الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي و البحث العلمي
Ministère de l'enseignement supérieur et de la recherche scientifique

Université Mohamed Khider – Biskra
Faculté des Sciences Exactes, des Sciences de la nature et de la Vie
Département : Informatique
Réf : .....

جامعة محمد خيضر بسكرة
كلية العلوم الدقيقة و علوم الطبيعة و الحياة
قسم: الإعلام الآلي
المرجع : .....

Thèse présentée en vue de l'obtention
Du diplôme de

Doctorat LMD en Informatique

Spécialité : Image et Vie Artificielle

Techniques de contrôle bio-inspirées: application à la simulation réaliste des comportements de créatures artificielles

---

Présentée par :

Ahmed TIBERMACINE

Soutenue publiquement le Mercredi 9 décembre 2015

Devant le jury composé de :

Pr. BAARIR Zineddine	**Professeur**	**Président**	**Université de Biskra**
Pr. DJEDI NourEddine	**Professeur**	**Rapporteur**	**Université de Biskra**
Pr. BENMOHAMMD Mohammed	**Professeur**	**Examinateur**	**Université de Constantine 2**
Pr. MOUSSAOUI Abedelouahab	**Professeur**	**Examinateur**	**Université de Sétif 1**
Dr. BABAHENINI Med Chaouki	**MCA**	**Examinateur**	**Université de Biskra**
Pr. CHERIF Foudil	**Professeur**	**Examinateur**	**Université de Biskra**

DÉDICACES

Afin d'être reconnaissant envers ceux qui m'ont appuyé et encouragé à effectuer ce travail de recherche, je dédie ce mémoire :

À ma très chère mère Yamina et à mon très cher père Abderrahmane pour leur soutien moral, et pour tous les sentiments d'affection et d'amour qui représentent pour moi le pilier de tous mes efforts.

À tous les membres de ma famille sans aucune exception.

Et à tous ceux que ma réussite leur tient à cœur.

REMERCIEMENTS

Tout d’abord, j'exprime mes vifs remerciements à mon directeur de thèse, Monsieur **NourEddine DJEDI**, Professeur à l'Université de Biskra, qui m'a encadré durant la période de la thèse avec grande patience et bonne humeur. Son assistance, sa disponibilité, les encouragements qu'il n'a cessé de me prodiguer et enfin la confiance qu'il m’a toujours témoigné, m'ont été d'un grand secours pour l'accomplissement de ce travail. Je lui en suis très reconnaissant.

Je tiens à remercier profondément les membres de jury :

- Monsieur Zineddine BAARIR, Professeur à l'Université de Biskra, d'avoir accepté d'examiner ce travail et de m'avoir honoré de présider le jury.
- Monsieur Mohammed BENMOHAMMD, Professeur à l'Université de Constantine2, qui a accepté d'être examinateur. Je le remercie très sincèrement.
- Monsieur Abedelouahab MOUSSAOUI, Professeur à l'Université de Sétif 1, qui a accepté d'être examinateur. Je le remercie très sincèrement.
- Monsieur Mohammed Chaouki BABAHENINI, Maitre de conférence à l'Université de Biskra, qui a accepté d'être examinateur. Je le remercie très sincèrement.
- Monsieur Fodil CHERIF, Professeur à l'Université de Biskra, qui a accepté lui aussi d'examiner cette thèse. Je tiens à le remercier pour son extrême gentillesse.

je désire aussi remercie à exprimer ma profonde gratitude et mes remerciements aux enseignants du département d'Informatique de l’Université de Biskra, notamment Mr. T. ABABSA et le Docteur O. TIBERMACINE, qui m'ont soutenu et aidé durant cette période, ainsi que leurs conseils fructueux.

Enfin, mes remerciements vont également à ma famille et mes amis pour la patience dont ils font preuve pour me soutenir et venir à bout de ce projet.

Veuillez trouver ici l'expression de mon profond respect et de mon indéfectible attachement.

Résumé

Les robots autonomes ont toujours attiré l'attention des chercheurs dans le monde entier. Le défi principal de la robotique autonome traditionnelle est de concevoir un logiciel pour un robot physique donné de telle manière, que le robot équipé de ce logiciel est capable de réaliser, d'une manière autonome, les tâches données. Plusieurs approches ont été introduites pour la conception de systèmes de contrôle robotique. Dans le domaine de la conception manuelle des contrôleurs robotiques, les méthodes vont des techniques de planification classiques à la planification réactive. Une autre approche consiste à créer automatiquement des contrôleurs robotiques pour effectuer une tâche donnée dans un environnement simulé.

Dans cette thèse, nous présentons les principales recherches en vie artificielle et en réalité virtuelle qui s'inspirent de l'évolution naturelle pour concevoir automatiquement des contrôleurs pour des créatures artificielles. De cette manière, les robots peuvent exploiter la dynamique physique de leur environnement pour générer des comportements.

Nous commençons avec une série d'expérimentations d'organismes robotiques artificiels, dans lequel la morphologie de la créature est fixée, et seulement le contrôleur de la créature est optimisé utilisant l'évolution simulée. Nous proposons deux approches évolutionnaires pour l'auto-conception du contrôleur. La première fournit deux caractéristiques principales: (a) la topologie du contrôleur neuronal se développe progressivement en taille pour générer des comportements de plus en plus complexes, et (b) le processus d'évolution ne nécessite que les propriétés physiques du modèle de la créature et une fonction de fitness simple. Les expérimentations ont montré que l'approche proposée augmente significativement la performance de l'évolution sur la plupart des tâches testées.

La seconde approche est présentée dans la dernière section, dans lequel un modèle de développement artificiel, basé sur les réseaux de régulation génétique, est introduit pour faire évoluer le contrôleur des créatures artificielles. Nous avons pu démontrer que le modèle basé sur les réseaux de régulation génétique peut éventuellement être une solution viable pour l'évolution des solutions de contrôle efficace pour des machines physiques.

Les mots clés: réseaux de régulation génétique, créature virtuelle, contrôle comportemental, simulation physique, calcul évolutionnaire, NeuroEvolution, apprentissage automatique.

الملخص

تشد أثار مجال التحكم الذاتي للروبوتات اهتمام الباحثين في جميع أنحاء العالم. ويتمثل التحدي الرئيسي للروبوتات ذات التحكم الذاتي التقليدية في تصميم برمجيات للروبوت المادي حيث أن هذه البرامج تزود الربوت بالقدرة على أداء مهام معينة بطريقة تلقائية. وقد أدخلت عدة طرق لتصميم أنظمة التحكم الروبوتية. بالنسبة لمجال التصميم اليدوي لنظام تحكم في الروبوت، الأساليب تتنوع من أساليب التخطيط الكلاسيكية إلى أساليب التخطيط رد الفعل. من جهة أخرى، ظهر نهج آخر يهدف إلى إنشاء وحدات تحكم الروبوتية بصفة تلقائية و بدون أي يتدخل خارجي لتدريب الروبوتات على أداء مهمة معينة في بيئة محاكاة.

في هذه الأطروحة، سنقوم بعرض البحوث الرئيسية في حياة اصطناعية والواقع الافتراضي المستوحاة من التطور الطبيعي لتصميم التلقائي لأنظمة التحكم الكائنات الروبوتية الاصطناعية. وبهذه الطريقة يمكن أن تستغل الروبوتات الحركة الفيزيائية للبيئة لتوليد السلوك.

نبدأ مع مجموعة من التجارب للكائنات الروبوتية الاصطناعية بحيث أن جسم الكائن ثابت لا يتغير، وحدة التحكم وحدها تتطور باستخدام المحاكاة التطورية. في هذا السياق، اقترحنا تهجين نظاما تطوريا لتصميم وحدة التحكم. الأول يوفر سمتين رئيسيتين: (أ) حجم طوبولوجيا وحدة التحكم العصبي ينمو تدريجيا بالسماح للسلوك بالازدياد تعقيدا، و (ب) يتطلب عملية تطورية فقط للخصائص الفيزيائية للنموذج الكائن إضافة إلى دالة كفاءة بسيطة. لا يوجد هناك ضرورة لمعرفة مسبقة لأنماط الحركة المناسبة. وقد أظهرت التجارب أن النهج المقترح يزيد بشكل كبير من أداء تطور في معظم المهام التي تم اختبارها.
الفصل الأخير للأطروحة يعرض لنا النهج الثاني، و الذي يتضمن نموذجا للتنمية الصناعية مستوح من الشبكات التنظيمية الوراثية. هذا النموذج صمم أيضا لتطوير نظام تحكم الكائنات الروبوتية. ولقد أثبتت هذه التجارب أن نموذج الشبكة التنظيمية الجينية قد يكون حلا قابلا للتطبيق لتطوير نظم التحكم للآلات المادية.

كلمات البحث: شبكة الجينات التنظيمية، مخلوق افتراضي، التحكم في السلوك، المحاكاة الفيزيائية، الحساب التطوري، الشبكات العصبية التطورية، التعلم الآلي.

Abstract

Autonomous robots have always attracted attention of researchers around the world. The main challenge of traditional autonomous robotics is to design software for the given physical robot in such way, that the robot equipped with this software is able to autonomously perform given tasks. Several approaches have been introduced for designing robotic control systems. In the area of manually-designed robotic controllers, methods range from classical planning techniques to reactive planning. Another approach is to create robotic controllers automatically, training them for a given task in a simulated environment.

In this thesis, we present the major research in artificial life and virtual reality inspired by the natural evolution to automatically design of artificial creature's controller. In this way the robots can exploit the physical dynamics of their environment to generate behaviour.

We begin with a set of artificial creature experiments, in which the creature's body is fixed, and only the controller is optimized using simulated evolution. We proposed two evolutionary approaches to design the controller. The first provides two key features: (a) the topology of the neural controller gradually grows in size to allow increasingly complex behaviours, and (b) the evolutionary process requires only the physical properties of the creature model and a simple fitness function. No a priori knowledge of the appropriate cycles or patterns of motion is needed. Experiments have shown that the proposed approach significantly increases the performance of the evolution on most tested tasks.

The second approach is presented in the last section, in which a model of artificial development, based on genetic regulatory networks (GRNs), is introduced to evolve the controller system of artificial creatures. It is shown that gene regulatory network model may possibly be a viable solution for evolving control solutions for physical machines.

Keywords: Gene regulatory network, virtual creature, behavior control, physical simulation, Evolutionary computation, NeuroEvolution, machine learning.

Table des matières

Table des figures

CHAPITRE 2

LA VIE ARTIFICIELLE ET LES SYSTÈMES ÉVOLUTIONNAIRES

CHAPITRE 3

GÉNÉRATION BIO INSPIRÉE DE COMPORTEMENTS DANS UN ESPACE VIRTUEL

CHAPITRE 4

MODÈLE PROPOSÉ

CHAPITRE 5

NEUROEVOLUTION POUR LE CONTRÔLE: APPLICATION À LA SIMULATION DE CRÉATURES ARTIFICIELLES

CHAPITRE 6

RÉSEAU DE RÉGULATION GÉNÉTIQUE POUR LE CONTRÔLE ET LA SIMULATION DE LA LOCOMOTION DE CRÉATURES ARTIFICIELLES

Liste des tableaux

CHAPITRE 4

MODÈLE PROPOSÉ

CHAPITRE 5

NEUROEVOLUTION POUR LE CONTRÔLE: APPLICATION À LA SIMULATION DE CRÉATURES ARTIFICIELLES

1.Introduction générale

1.1. Introduction et motivations

De nombreux problèmes semblent exiger de la créativité humaine pour être résolus avec succès. Trouver une telle solution implique souvent la recherche des grands espaces multidimensionnels de toutes solutions possibles, qui, heureusement, les humains peuvent souvent faire intuitivement. Cependant, le processus de l'invention peut être difficile, fastidieux et fatigant. Par conséquent, plusieurs méthodes d'automatisation de ces processus ont été proposées au cours des dernières décennies. Une telle approche est inspirée par la théorie darwinienne de l'évolution. Plusieurs méthodes de recherche évolutionnaire efficaces ont été proposées et ont même conduit récemment à l'invention automatique de plusieurs résultats compétitifs à la créativité humaine.

Le domaine de calcul évolutionnaire est en expansion très rapide et les méthodes de recherche évolutionnaire ont déjà été appliquées avec succès à divers problèmes. Cette thèse se concentre sur les méthodes de recherche évolutionnaire pour la conception automatisée des organismes robotiques, où la morphologie d'une créature est fixe et seul son système de contrôle (SC) est automatiquement inventé et optimisé au cours de l'évolution.

L'emploi des algorithmes évolutionnaires pour la conception de robots offrent la flexibilité accrue d'être possible de ré-exécuté pour différentes morphologies, alléger le fardeau de la création manuelle d'un nouveau contrôleur pour chaque nouvelle morphologie. Toutefois, l'application de cette technique pour générer des stratégies de locomotion nécessite l'évaluation de plusieurs solutions possibles; cela pose deux défis majeurs pour le domaine. Premièrement, l'énorme quantité de temps nécessaire pour tester itérativement les contrôleurs potentiels sur un robot est prohibitive. Deuxièmement, le problème devient presque impossible si le concepteur souhaite essayer différents plans du corps de robot, ainsi que différents contrôleurs, parce que les robots doivent alors être construits à partir de scratch et la construction d'un seul robot peut souvent prendre des semaines ou des mois.

Nous exposons ici une série d'expériences dans lesquelles le calcul évolutionnaire est utilisé pour générer un grand nombre de contrôleurs pour des robots simulés qui agissent dans un

environnement virtuel, mais physiquement réaliste. En simulant les robots, il est possible d'accélérer considérablement le temps nécessaire pour «construire» un robot, et le tester dans un environnement physique. En permettant l'évolution artificielle de régler les relations entre le corps d'un robot, son cerveau, et son interaction avec son environnement, nous sommes en mesure de découvrir plusieurs interdépendances entre le cerveau et le corps en particulier, et comment ces deux sous-systèmes fonctionnent ensemble pour générer un comportement intelligent en général.

1.2. Problématique et objectifs

Notre problématique est donc de concevoir automatiquement des contrôleurs de comportement pour des créatures virtuelles adaptées à un environnement à partir de mécanismes d'évolution. Dans cette thèse, nous employons pour cela des modèles de calcul computationnel pour réaliser l'évolution de créatures artificielles.

Les objectifs de la réalisation de telles créatures peuvent répondre à quatre besoins.
Le premier est d'essayer de comprendre une partie des mécanismes les plus simples adoptés par le vivant pour concevoir des entités adaptées à un environnement. Le second, en réalité virtuelle, est d'automatiser l'animation d'entités virtuelles réalistes évoluant dans des environnements virtuels, mais physiquement réalistes. Le troisième est de développer un Framework pour l'évolution des comportements réaliste et optimal dans les créatures virtuelles qui peuvent être facilement transférées à des machines, en l'occurrence des robots. L'objectif étant d'avoir une machine qui puisse concevoir une autre machine. Le quatrième objectif de cette thèse est d'augmenter l'efficacité de l'évolution des créatures virtuelles en proposant une nouvelle méthode d'évolution de leurs systèmes de contrôle et d'étudier expérimentalement son impact sur le processus d'évolution des créatures.

1.3. Contribution

Dans le but de contrôler des créatures virtuelles, la thèse contribue sur deux plans: plan expérimental et plan théorique.
Dans le premier plan, nous avons développé quatre différents types de créature virtuelle; chacune est utilisée dans une expérimentation indépendante. Chaque créature représente une espèce entièrement différente des créatures artificielles qui se diffèrent dans le nombre et le type de jointures, de capteurs d'entrées, d'effecteurs et des corps rigides.

Dans le second, plan nous avons développé deux approches évolutionnaires de contrôle. La première approche introduit un système pour faire évoluer automatiquement les réseaux de neurones en tant que contrôleurs de locomotion, basés sur la physique, pour les créatures artificielles. Notre approche fournit deux principales caractéristiques: (a) la topologie du contrôleur de réseau de neurones se développe progressivement en taille pour générer des comportements de plus en plus complexes, et (b) le processus d'évolution ne nécessite que les propriétés physiques du modèle de caractère et une fonction de fitness simple. La seconde approche utilise un modèle computationnel du réseau régulateur génétique, qui s'inspire du mécanisme de contrôle cellulaire de cellules réelles pour faire évoluer les comportements des créatures virtuelles. Les résultats montrent que le modèle de réseau régulateur génétique peut éventuellement être une solution viable pour l'évolution des solutions de contrôle pour des machines physiques.

1.4. Plan de la thèse

Dans cette section, nous proposons un résumé des différents chapitres composant ce manuscrit tout en précisant les principaux résultats obtenus.

Le second chapitre de cette thèse pose les repères nécessaires à une compréhension du contexte dans lequel s'inscrit cette thèse. Après une introduction générale au domaine de la vie artificielle, nous présenterons les différents systèmes utilisés dans le cadre de ce projet de recherche. Ce chapitre *fournit* également une description *détaillée* des systèmes évolutionnaires qui, dans le cadre de la problématique donnée, joue un rôle primordial dans la conception du système comportemental pour des créatures artificielles et leur adaptation à leur environnement virtuel.

Le troisième chapitre présente un état de l'art sur les travaux relatifs aux créatures artificielles dans un environnement virtuel 3D. Ces travaux ont été classés en deux catégories, la première concerne les travaux se focalisant sur l'approche manuelle pour la conception des contrôleurs robotique, cette approche comporte des méthodes vont des techniques de planification classiques à la planification réactive. Une deuxième approche consiste à créer automatiquement des contrôleurs robotiques pour effectuer une tâche donnée dans un environnement simulé. Cette dernière représente notre problématique de recherche.

Le quatrième chapitre introduit les détails de notre modèle de contrôle proposé, de l'environnement virtuel dont lequel les créatures s'évoluent, de différentes morphologies de

créatures que nous avons développées et de l'union entre la méthode évolutionnaire, utilisée pour concevoir le cerveau, et la morphologie des créatures.

Le cinquième chapitre présente notre première proposition, elle introduit un système pour faire évoluer automatiquement les réseaux de neurones en tant que contrôleurs de locomotion, basés sur la physique, pour les créatures artificielles.

Le sixième chapitre concerne la deuxième proposition qui entre aussi dans le cadre du développement de méthode pour l'auto-conception évolutionnaire de contrôleur de créatures artificielles.

Enfin, le septième chapitre présente la conclusion de ce travail, il met en exergue l'apport de cette thèse, les limites de l'approche élaborée ainsi que les perspectives potentielles.

2. La vie artificielle et les systèmes évolutionnaires

2.1. Introduction

Aujourd'hui, la nature est utilisée comme une source d'inspiration pour le développement de nouvelles techniques et la résolution de problèmes complexes dans divers domaines, de l'ingénierie à la biologie, avec des adaptations innovantes sous investigation. Ce chapitre pose les repères nécessaires à une compréhension du contexte dans lequel s'inscrit cette thèse. Après une introduction générale au domaine de la vie artificielle, nous présentons les différents systèmes utilisés dans le cadre de cette investigation de recherche. Ce chapitre fournit également une description détaillée des systèmes évolutionnaires qui, dans le cadre de la problématique considérée, jouent un rôle primordial dans la conception du système comportemental pour des créatures artificielles et leur adaptation à leur environnement virtuel.

2.2. Qu'est-ce que la vie artificielle?

L'étude de la vie et ses systèmes a toujours été dominée par l'étude des organismes organiques à base de carbone. Cette étude est largement contenue dans le domaine de la biologie. Cette dernière applique une approche descendante[1] et tente de démonter les systèmes vivants existants pour découvrir les éléments essentiels.

L'approche biologique peut être assimilé à une tentative d'apprendre les principes d'un métier d'art en étudiant les travaux trouvés dans un musée ou crées par un artiste.

Bien qu'une telle approche va sûrement donner des idées inestimables, il est improbable que nous puissions apprendre tout ce qu'il y'a à savoir sur ce métier. D'autre part, une meilleure compréhension peut être acquise en essayant de créer nos propres travaux. De la même manière, au lieu d'étudier les phénomènes biologiques en démontant les organismes vivants pour mieux comprendre leurs principes de fonctionnement, la discipline de la vie artificielle

[1] On parle de « top-down approach »

tente de mettre en place des systèmes capables de générer des comportements similaires à ceux des organismes vivants. Elle cherche à mieux comprendre la vie et les systèmes de la vie par l'entremise de la simulation et la synthèse des systèmes de la vie naturelles.

Le terme «vie artificielle» est utilisé pour décrire la recherche dans les systèmes anthropiques[2] qui possèdent certaines propriétés essentielles de la vie. Cet effort est vraiment interdisciplinaire et couvre toute la gamme de la biologie, de la chimie et de la physique à l'informatique et l'ingénierie. Tandis qu'une grande partie de la vie artificielle est consacrée à la compréhension de la vie comme nous la savons [Langton, Taylor et al., 1992] (c'est-à-dire, la vie sur terre), un effort significatif concerne la recherche de principes des systèmes vivants qui sont indépendants d'un substrat particulier. Ainsi, la vie artificielle considère aussi la vie (comme il pourrait être) et explorer des alternatives artificielles à la chimie à base de carbone [Langton, Taylor et al., 1992]. La vie artificielle est souvent décrite comme une tentative de comprendre le comportement de haut niveau à partir des règles de bas niveau; par exemple, comment les règles simples de l'évolution darwinienne mènent à des structures de haut niveau, ou la façon dont les simples interactions entre les fourmis et leur environnement mènent à des comportements complexes tels que: la poursuite et la recherche de la nourriture [Langton, 1989].

Les principes séminaux de la vie artificielle sont matérialisés par un ensemble de paradigmes qui se traduisent finalement sous la forme de techniques informatiques [Bedau, 2003]. Ces dernières sont souvent obtenues par une analogie réalisée entre des systèmes existants dans le monde réel et une adaptation de ces systèmes sous forme d'algorithmes. L'inspiration vient souvent de processus biologiques, mais peut aussi être trouvée dans des systèmes chimiques ou bien encore dans des systèmes techniques mis au point par des hommes. Il est important de noter que de simplifications en optimisations diverses, les méthodes réellement utilisées n'ont fréquemment plus de rapport avec les systèmes qui les ont inspirées et sont pleinement entrées dans le domaine des techniques informatiques [Floreano and Mattiussi, 2008].

La figure 2.1 montre les trois objectifs essentiels de la vie artificielle. En plus d'étudier les enjeux biologiques et extraire les principes de comportements intelligents, les applications pratiques doivent être développées [Bedau, 2003].

[2] On parle de « human-made system »

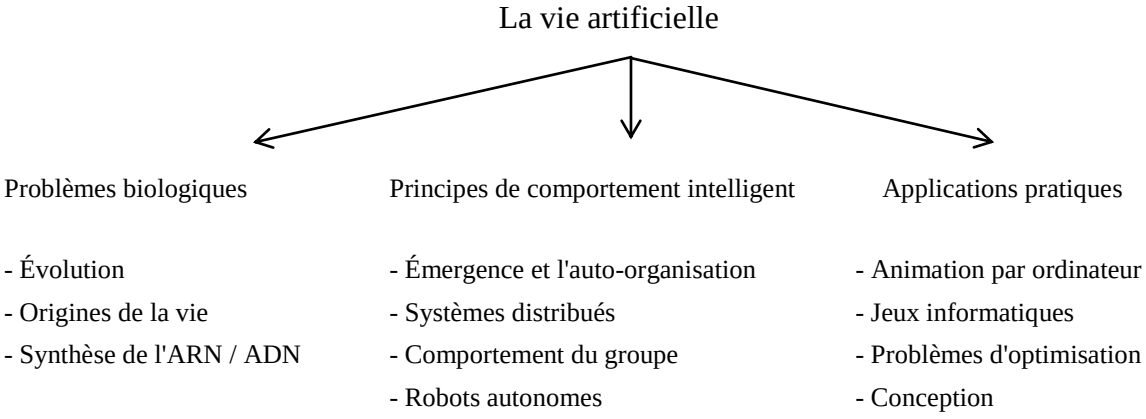


Figure 2. 1- Les objectifs de la vie artificielle.

2.3. Émergence de la vie artificielle comme discipline

Le terme «vie artificielle» a été inventé à la fin des années 1980 par le chercheur Christopher Langton qui l'a défini comme " l'étude des systèmes artificiels qui présentent un comportement particulier des systèmes de vie naturelle ". C'est la quête d'expliquer la vie dans toutes ses manifestations possibles, sans restriction aux exemples particuliers qui ont évolué sur terre. Ceci inclut les expérimentations biologiques et chimiques, les simulations informatiques ainsi que des efforts purement théoriques. Les processus qui se produisent au niveau moléculaire, social et les échelles évolutionnaires sont soumis à une investigation. Le but ultime est d'extraire la forme logique des systèmes vivants.

Probablement la première personne à activement étudier et écrire sur des sujets liés à la vie artificielle était le mathématicien John Von Neumann. Dans le milieu du 20e siècle, Von Neumann a présenté un papier intitulé « The General and Logical Theory of Automata », dans lequel il a discuté le concept d'une machine qui suit des règles simples et qui réagit à l'information de son environnement [Von Neumann and Burks, 1966]. Von Neumann a proposé que les organismes vivants ne soient que de telles machines. Il a également étudié le concept de la machine auto-réplicative, et conçut l'idée qu'une machine auto-réplicative, ou un organisme doivent contenir en lui-même une liste d'instructions pour produire une copie de lui-même. C'était plusieurs années avant que James Watson et Francis Crick, avec l'aide de Rosalind Franklin et Maurice Wilkins, ont découvert la structure de l'ADN [Von Neumann and Burks, 1966].

La première exposition des concepts de vie artificielle (VA) au grand public est apparue dans la colonne: Jeux mathématiques « Mathematical Games » du magazine « Scientific American ». Dans les années 1960, un professeur nommé John Conway a conçu un automate

cellulaire (CA) qui s'appelle le jeu de la vie. Les Automates cellulaires de Conway étaient assez simples, mais ils présentent des comportements étonnamment complexes et réalistes [Gardner, 1970].

Jusqu'à ce point, il n'y avait pas de discipline pour ces concepts que nous reconnaissons maintenant comme des concepts liés à la vie artificielle. Ce n'était pas jusqu'à la fin des années 1970 et au début des années 1980 qu'un programmeur non conventionnel, nommé Christopher Langton a inventé le terme "vie artificielle". Langton est également responsable de la première conférence, "International Conference on the Synthesis and Simulation of Living Systems" également appelé "Artificial Life 1", spécifiquement dédiée à cette nouvelle discipline.

2.4. Quelle est la différence entre la vie artificielle et l'intelligence artificielle?

La vie artificielle chevauche avec l'intelligence artificielle, mais les deux domaines sont très différents dans leurs approches et dans leur histoire [Langton, 1997; Bedau, 2003]. La vie artificielle s'intéresse aux algorithmes spécifiques orientés vers la vie tels que les algorithmes génétiques qui peuvent imiter la nature et ses lois et concerne donc plus la biologie [Langton, 1997], alors que l'intelligence artificielle tend à chercher la façon dont l'intelligence humaine peut être reproduite, elle est de ce fait plus liée à la psychologie [Jackson, 1985].

Cette question est beaucoup plus complexe qu'elle ne le paraît. Ceci est partiellement dû au fait qu'il y a beaucoup de désaccords entre les experts quant aux définitions des concepts de vie et d'intelligence. Cependant, il n'y a pas de définitions fermes pour ces termes, il est possible de différencier la vie artificielle et l'intelligence artificielle.

La vie artificielle dans sa forme pleinement réalisée serait toute «chose» anthropique vivante. Que ce cette "chose" soit composée de produits chimiques organiques, d'électrons circulant à travers le silicium ou tout autre matériel, ne ferait aucune différence. Les progrès rapides en informatique, ainsi que leur capacité à simuler les processus complexes, en font un outil idéal pour analyser, synthétiser et explorer les aspects et les éléments des systèmes vivants [Langton, 1989; Langton, Taylor et al., 1992; Rennard and Mange, 2002; Bedau, 2003].

L'intelligence artificielle est toute chose anthropique qui est capable de reproduire un comportement compatible avec l'intelligence humaine ou animale (organique). Étant donné

que toutes les formes connues de l'intelligence naturelle sont incorporées dans les systèmes vivants organiques, il n'est pas surprenant qu'il y ait un certain chevauchement entre les domaines de la vie artificielle et l'intelligence artificielle. Cependant, nous n'utilisons généralement pas le terme «intelligent» pour décrire tous les systèmes vivants. Les plantes, par exemple, sont vivantes, mais elles ne sont pas considérées comme intelligentes. Ainsi, il n'est pas indispensable que la vie artificielle doive inclure l'étude de l'intelligence artificielle. En même temps, tous les systèmes intelligents ne doivent pas être vivants. Un programme d'informatique qui joue aux échecs au niveau de « maître » présente un comportement intelligent, pourtant ce n'est pas une chose vivante [Jackson, 1985].

2.5. Les systèmes de la vie artificielle

Étudier l'origine des paradigmes de vie artificielle pour les classer n'est pas opportun, car un même système réel peut avoir donné lieu à des techniques très différentes. Hervé Luga a proposé dans son habilitation une approche fonctionnelle en discriminant les systèmes en fonction de leur fonctionnement vis-à-vis des structures sur lesquels ils portent.

2.5.1. Systèmes générateurs

Les systèmes générateurs sont des systèmes qui agissent sur un état initial, appelé axiome, et le faire évoluer progressivement au cours des générations successives. L'évolution s'effectue par l'application récursive de règles de réécriture dans le but de construire des objets plus complexes.

En rapprochant cette définition des systèmes vivants, on peut clairement classer les systèmes générateurs comme des systèmes agissant au niveau endogénétique. En effet, il n'existe pas de modifications endogènes des règles ni de l'axiome, seul l'état courant est modifié. De plus, si le système n'est pas dépendant du hasard ou d'actions extérieures il reste parfaitement déterministe.

Les systèmes générateurs comprennent les modèles de réaction diffusion et les automates cellulaires qui sont considérés comme les plus anciens. Ils sont inspirés par une simplification de l'évolution des concentrations de composés chimiques en solution. Ces modèles sont aujourd'hui encore le support de nombreuses études sur la complexité.

Les systèmes de type masse-ressort, les réseaux d'interaction et encore les systèmes à base de grammaires tels que les L-systèmes peuvent être aussi classés dans les modèles générateurs. De plus, ces modèles fournissent des outils puissants pour la création de structures complexes.

2.5.2. Système d'apprentissage

Un système d'apprentissage agit sur une structure de données définissant le comportement d'un système en modifiant cette structure de manière à adapter le comportement à un environnement. L'adaptation se fait généralement par retour d'un «feedback» fourni par cet environnement pour moduler la modification de la structure de données. Le «feedback» peut être exogène au système apprenant et on aura alors un type d'apprentissage par interaction directe entre l'environnement et le système. Il peut aussi être endogène et donc interne au système apprenant [Parizeau, 2004].

Deux catégories principales de techniques d'apprentissage existent: apprentissage supervisé et apprentissage non supervisé. Dans l'apprentissage supervisé, un «enseignant» est présent, qui "dit" au système comment il a résolu la tâche pendant la phase d'apprentissage (apprentissage par renforcement), ou fournit directement des valeurs de sortie désirées (apprentissage entièrement supervisé). Dans l'apprentissage non supervisé, le système découvre certaines propriétés des données d'entrée par lui-même et produit les données pertinentes à ces propriétés [Parizeau, 2004].

Les exemples les plus connus de systèmes d'apprentissages sont la famille des réseaux de neurones artificiels [Parizeau, 2004] et les systèmes de classeurs qui sont utilisés avec succès dans les applications de la vie artificielle.

Les réseaux de neurones artificiels

Un réseau de neurones artificiel est constitué d'un ensemble d'éléments actifs nommés «neurones artificiels» qui sont reliés entre eux (Voir la figure 1.3 pour un exemple d'une topologie de réseau de neurones), inspirées par les réseaux de neurones réels trouvés dans de nombreux organismes vivants complexes. Les modèles de réseaux neuronaux rencontrés dans la nature se sont révélés être un mode de calcul très puissant et adaptable, capable de traiter robustement des grandes quantités des données. Ceci a inspiré les chercheurs à imiter les réseaux de neurones dans un environnement simulé par ordinateur [Parizeau, 2004].

Un neurone artificiel[3] est considéré comme une unité élémentaire de traitement possédant généralement un comportement assez simple[4] qui est défini par une fonction de transfert *f(net)*, d'une somme pondérée *net,* permettant de calculer sa valeur de sortie en fonction de ses valeurs d'entrée [Parizeau, 2004].

[3] Appelé aussi souvent «neurone formel».

[4] La plupart des fonctions utilisées sont des fonctions mathématiques de type seuil ou sigmoïde.

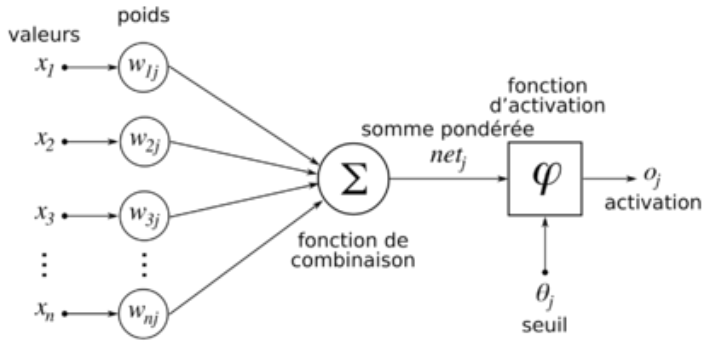


Figure 2. 2- neurone Formel

Les entrées et les sorties sont généralement à valeurs booléennes, entières ou réelles. Les liaisons et les neurones possèdent des caractéristiques[5] dont l'évolution au cours du temps va réaliser l'apprentissage.

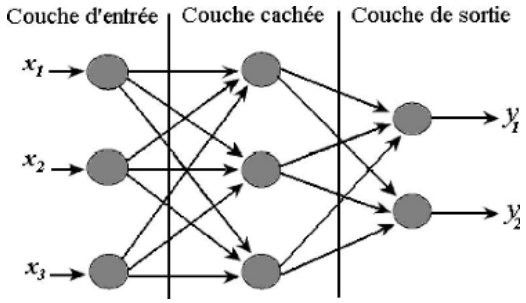


Figure 2. 3- Un réseau de type perceptron à une seule couche cachée.

Le lien entre un réseau de neurones et son environnement est réalisé par des «neurones d'entrée» du réseau dont la valeur est liée à des perceptions et des «neurones de sortie» liés à des effecteurs. L'ensemble constitué par les neurones artificiels ainsi que leurs connexions permet de définir la topologie du réseau. Les réseaux de neurones sont généralement vus comme organisés en «couches» [Parizeau, 2004]:

- La première couche du réseau appelée couche d'entrée;
- La dernière couche du réseau appelée couche de sortie;
- Les couches qui ne sont pas liées à des entrées ou à des sorties du réseau sont appelées «couches cachées». Le nombre de couches cachées va permettre de définir la complexité du traitement réalisé, car chaque couche correspond à l'addition d'un niveau de traitement sur l'information arrivant des capteurs.

Si on interdit la connexion entre un neurone d'une couche i et un autre d'une couche j<=i on parle de réseau feed-forward. Ces réseaux sont les plus courants, car les plus faciles à

[5] Parmi ces caractéristiques, on trouve principalement les points de départ et d'arrivée et un «poids» associé à chaque connexion représente l'atténuation ou l'amplification du signal lors du parcours de cette connexion.

entraîner. Nous trouvons dans cette catégorie, le perceptron qui est un réseau feed-forward où les couches cachées sont totalement interconnectées au début de l'apprentissage. La configuration initiale d'un tel réseau consiste donc en la définition du nombre de couches cachées et du nombre de neurones dans chacune de ces couches[6].

Plusieurs modèles de RNA ont été proposés. Le défi principal pour un algorithme de RNA est d'optimiser les propriétés du réseau de telle façon que le réseau de neurones soit capable de résoudre une tâche donnée. Les méthodes traditionnelles utilisent une topologie de réseau fixe, qui est fourni par l'expérimentateur à l'avance et seulement les poids des connexions sont optimisés. Cependant, les méthodes récentes ont montré que l'optimisation de la topologie du réseau de neurones peut ainsi augmenter la performance de l'algorithme d'apprentissage [Parizeau, 2004].

2.5.3. Systèmes évolutionnaires

Tous les systèmes biologiques résultent d'un processus évolutif. La sophistication, la robustesse et l'adaptabilité des systèmes biologiques représentent une motivation puissante pour reproduire les mécanismes d'évolution naturelle en essayant de générer des systèmes logiciels et matériels ayant des caractéristiques comparables à celles des systèmes biologiques [Floreano and Mattiussi, 2008]. Il y a plus de 40 ans, les informaticiens et les ingénieurs ont commencé à développer des algorithmes inspirés de l'évolution naturelle [Fogel, Owens et al., 1966; Rechenberg, 1973; Holland, 1975] pour générer des solutions aux problèmes qui ont été trop difficiles à aborder avec d'autres méthodes analytiques.

Le calcul évolutionnaire est rapidement devenu un domaine important de l'apprentissage automatique[7] « *machine learning* » et d'optimisation des systèmes et, plus récemment, il s'est répandu dans le domaine de la conception de matériel en exploitant de nouvelles technologies dans les circuits reconfigurables électroniques, la fabrication assistée par ordinateur, les technologies de la production matérielle ainsi que la robotique [Floreano and Mattiussi, 2008; Sumathi and Paneerselvam, 2010].

2.5.3.1. Les Piliers de la théorie de l'évolution

Le domaine de la Biologie fait des progrès continus dans la description des éléments qui composent les organismes vivants et la façon dont ces composants fonctionnent ensemble.

[6] Ici encore il n'existe pas de méthode pour définir la topologie optimale d'un perceptron, l'expérience de la personne qui réalisera la mise en œuvre est donc primordiale.

[7] On parle de « machine learning »

Cependant, l'explication ultime se trouve dans la théorie de l'évolution naturelle. Dobzhansky a dit, « rien en biologie n'a de sens qu'à la lumière de l'évolution » [Dobzhansky, 1973]. Un nombre important de livres et d'articles ont été écrits sur la théorie de l'évolution naturelle, mais ses fondements sont assez simples et relativement élégants [Floreano and Mattiussi, 2008].

La théorie de l'évolution naturelle repose sur quatre piliers: la population, la diversité, l'hérédité et la sélection.

Population : La prémisse de l'évolution est l'existence d'une population que l'on peut définir comme étant un groupe de deux individus ou plus. En d'autres termes, nous ne pouvons pas parler de l'évolution pour un seul organisme.

Diversité: La diversité signifie que les individus de la population se diffèrent les uns des autres dans une certaine mesure.

Hérédité: L'hérédité indique que les caractères d'un individu peuvent être transmis aux descendants à travers la reproduction.

Sélection: la sélection indique qu'une seule partie de la population est capable de se reproduire et transmettre ses caractères aux générations suivantes. La sélection naturelle, proposée par Darwin (1859) et Wallace (1870) au dix-neuvième siècle, se fonde sur la prémisse: " *individuals tend to make several offspring and that not all of them may reproduce* ". La sélection des individus qui peuvent se reproduire n'est pas complètement aléatoire, mais régulée par les contraintes environnementales. Par exemple, si des individus cherchent de la nourriture disponible dans un environnement, ceux qui la trouvent et la consomment le plus vite auront une plus grande chance de survivre et de se reproduire [Nolfi and Floreano, 2000].

2.5.3.2. Biologie évolutionnaire

a. Génotype

En 1865, Mendel est arrivé à la conclusion que les individus se reproduisent en transmettant des particules spécifiques, connues comme étant leur matériel génétique[8], à leurs propres descendants. Les progrès récents dans le domaine de la génétique[9] et en génomique

[8] On parle de « genetic material ».

[9] On parle de « genetics », une discipline qui étude la structure et le comportement des gènes

fonctionnelle[10] ont fourni plusieurs indices pour les mécanismes et les processus moléculaires qui soutiennent l'héritage et la variation.

Le matériel génétique d'un individu s'appelle "génotype", alors que sa manifestation en tant qu'organisme représente le "phénotype". La sélection naturelle fonctionne uniquement sur le phénotype, mais le génotype est l'instrument ou le moyen ultime de l'héritage.

Les cellules contiennent une classe de molécules, appelées protéines, dont la forme, la concentration et le comportement déterminent les propriétés de la cellule. Par exemple, les cellules de cheveux et les cellules musculaires sont différentes parce qu'elles sont composées de protéines différentes. La définition des protéines spécifiques dépend d'une autre molécule, nommée ADN (acide désoxyribonucléique), qui à son tour dépend des protéines pour devenir active et sur la médiation d'un troisième type de molécule, appelé ARN (acide ribonucléique), qui est structurellement similaire à la molécule d'ADN.

L'ADN est le matériel génétique qui est transmis au fil des générations. Il est souvent enfermé dans le noyau de la cellule et toutes les cellules de l'organisme ont le même matériel génétique. Les molécules d'ADN (figure 2.4) sont de longues chaînes de brins complémentaires composés de quatre types d'unités chimiques (nucléotides): adénine (A), cytosine (C), guanine (G) et thymine (T). Les deux brins sont collés ensemble parce que les nucléotides peuvent se verrouiller les uns aux autres: L'adénine se lie à la thymine et la cytosine se lie à la guanine. Cette liaison spécifique signifie que les deux brins d'ADN sont parfaitement complémentaires. Si nous trouvons l'ACA de séquence sur un brin, nous savons que la partie correspondante du brin complémentaire affichera la séquence TGT (bien que certains décalages puissent se produire, mais très rarement) [Nolfi and Floreano, 2000; Floreano and Mattiussi, 2008; Sumathi and Paneerselvam, 2010].

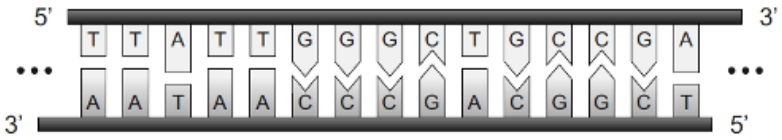


Figure 2. 4- Structure d'un morceau de molécule d'ADN représentant les deux brins avec les nucléotides correspondants. Les nombres 5 et 3 se réfèrent à la structure atomique de la molécule et affectent la manière dont la séquence de molécule est traduite en protéine. L'ordre de translation se poursuit toujours dans la direction de 5' vers 3'[Floreano and Mattiussi, 2008].

[10] On parle de « functional genomics ».Discipline qui étude le rôle des gènes dans les organismes

Le matériel génétique est organisé en plusieurs molécules d'ADN séparées, appelées chromosomes. En outre, dans plusieurs organismes, les chromosomes se produisent en paires (également appelés organismes diploïdes contrairement aux organismes haploïdes). Les deux chromosomes d'une même paire sont approximativement homologues dans le sens où les zones correspondantes produisent des protéines avec une fonctionnalité similaire dans des cellules similaires. Le nombre de paires de chromosomes et la longueur totale des molécules d'ADN varient d'une espèce à une autre. Par exemple, les humains ont 23 paires de chromosomes totalisant plusieurs centaines de millions de nucléotides (selon International Human Genome Sequencing Consortium 2001). La structure redondante du matériel génétique (deux chromosomes, deux brins) permet la réplication des molécules d'ADN lors de la réplication cellulaire [Floreano and Mattiussi, 2008].

b. Expression du gène

La séquence de quatre nucléotides le long de la chaîne d'ADN détermine les propriétés des cellules et le développement de l'organisme. Les quatre nucléotides sont effectivement les lettres de l'alphabet génétique. Les gènes sont fonctionnellement des séquences de nucléotides pertinents dans la chaîne d'ADN, qui peuvent produire des protéines.

Les protéines sont de longues chaînes moléculaires (figure 2.5) composées de centaines de sous-molécules, appelées acides aminés. Il existe 20 types d'acides aminés qui peuvent être combinés de différentes manières permettant de constituer un très grand nombre de protéines différentes. Lorsque les acides aminés sont enchaînés ensemble, ils se plient et se tordent dans l'espace à trois dimensions. Les propriétés d'une protéine sont déterminées principalement par sa forme. Chaque acide aminé correspond à une ou plusieurs séquences de trois nucléotides (codons) dans la chaîne d'ADN.

La production de protéines (figure 2.5) de l'ADN se fait par l'intermédiaire de l'ARN. Ce dernier est une longue molécule semblable à l'ADN, mais il se compose d'un seul brin de nucléotides. Elle est beaucoup plus courte et dispose d'uracile (U) à la place de l'ADN thymine (T). Au cours de la production de protéines, les deux brins de l'ADN sont séparés et une molécule d'ARN est assemblée le long d'une petite partie du brin d'ADN de manière à correspondre aux nucléotides correspondants [Edelman, 2008; Floreano and Mattiussi, 2008]. Ce processus est connu sous le nom transcription. La molécule d'ARN résultante est utilisée pour créer une protéine par l'assemblage d'une chaîne d'acides aminés correspondant à la séquence de nucléotides. Certaines protéines régulent la division cellulaire et l'expression

génétique des protéines. Il est important de noter que lorsque la séquence des nucléotides de l'ADN ne peut être modifiée par des protéines, la séquence de protéines « acides aminés » est plutôt déterminée par l'ADN. En d'autres termes, l'information coule dans une seule direction, des gènes vers les protéines. Ceci est la raison pour laquelle les modifications du phénotype, qui se produisent pendant la vie de l'individu sont causées par des phénomènes environnementaux, ne peuvent pas modifier directement le génotype et être héritées par la descendance (à l'exception de l'exposition aux rayonnements, ce qui peut affecter directement la séquence d'ADN). Ceci est également la raison pour laquelle le génotype est parfois considéré comme le plan directeur de l'organisme, c'est-à-dire la liste des instructions pour construire un système vivant qualifié [Edelman, 2008; Floreano and Mattiussi, 2008].

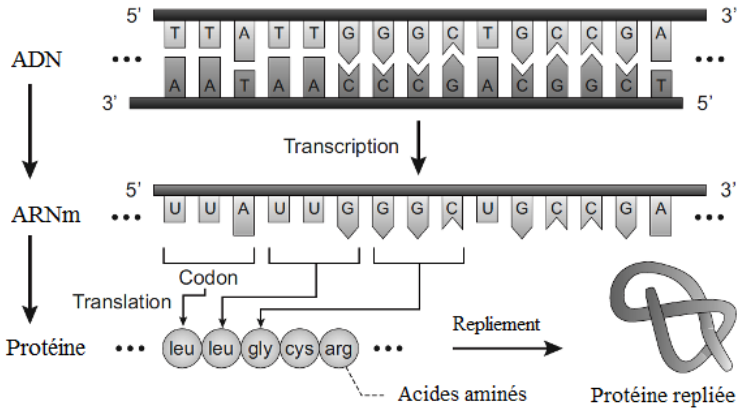


Figure 2. 5- Création d'une molécule de protéine. Une séquence d'un seul brin d'ADN est transcrite en une molécule d'ARN un ARN monocaténaire « single-stranded RNA molecule » lorsque l'uracile (U) est utilisé à la place de la thymine (T). Les triplets de nucléotides (codons) sont ensuite traduits en des molécules d'acides aminés correspondants (un acide aminé donné peut être généré par un ou plusieurs types de codons). Les acides aminés sont alors liés ensemble pour former la molécule de protéine qui se replie dans des formes différentes en fonction de la séquence spécifique d'acides aminés [Floreano and Mattiussi, 2008].

Le génotype comprend à la fois l'ADN génique et l'ADN non-génique. L'ADN génique est la partie de la molécule qui peut produire des protéines, tandis que l'ADN non-génique est la partie qui ne produit pas de protéines [Floreano and Mattiussi, 2008].

Un gène peut être actif, inactif, ou modérément actif. Le niveau d'activité d'un gène est utilisé pour indiquer le taux de production de la protéine correspondante par l'ARN. Les gènes sont structurés en deux régions le long de la molécule d'ADN (voir la figure 2.6): une région

codante « *coding region* » et une région régulatrice « regulatory region ». La région chargée du codage est constituée d'une séquence de nucléotides qui est traduite en une molécule d'ARN et finalement en la protéine correspondante. Le processus de translation et sa vitesse sont contrôlés par la présence de protéines spécifiques qui se lient à la région régulatrice du gène. Comme indiqué précédemment, la région régulatrice est une séquence de nucléotides qui ne produisent pas de protéines. La forme des protéines de liaison est telle qu'elles ne peuvent se lier qu'à des séquences spécifiques de nucléotides de la région régulatrice. Si une telle liaison se produit sur la région régulatrice d'un gène, dans certains cas, ce gène s'exprime en initiant la translation de la région codante à une molécule d'ARN. Dans d'autres cas, les protéines de liaison peuvent inhiber l'expression du gène ou interférer de plusieurs manières avec les autres protéines qui sont liées à différentes zones de la région régulatrice, de manière à accélérer ou ralentir le taux de translation de la région codante [Edelman, 2008; Floreano and Mattiussi, 2008].

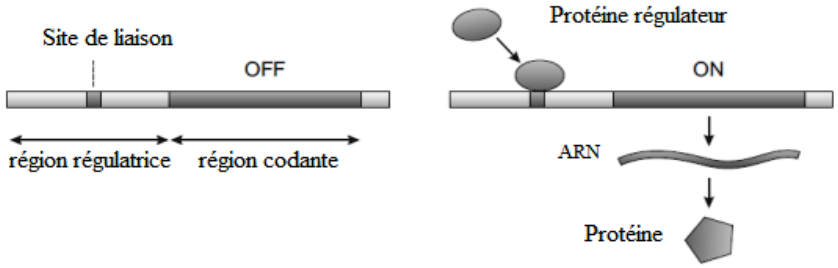


Figure 2. 6- Les gènes sont composés d'une région régulatrice et une région codante. L'activité de la région codante (c'est-à-dire, la production des protéines correspondantes) est déterminée par l'activation de la région régulatrice suivant la liaison d'une protéine spécifique. Puisque les protéines de liaison de l'ADN sont produites par des gènes, le modèle d'activation d'une molécule d'ADN est un réseau complexe d'interactions entre des parties des gènes et leurs produits protéiques [Floreano and Mattiussi, 2008].

Les protéines qui régulent l'expression du gène sont produites par des gènes, qui sont régularisées par d'autres protéines. Ces derniers produits par d'autres gènes, et ainsi de suite. En outre, les signaux chimiques provenant d'autres cellules ou induits par l'environnement peuvent affecter l'expression des gènes. L'image qui émerge est un réseau complexe d'interdépendances, également connu sous le nom *réseau de régulation génétique*, dont l'activité des gènes peut favoriser ou inhiber d'autres gènes. Une seule molécule d'ADN peut comporter plusieurs réseaux de régulation génétique, chacun correspondant à un modèle dynamique complexe de l'expression du gène. En conséquence, le rôle des gènes ne peut être

complètement compris en regardant simplement la séquence d'ADN ou en prenant une photo instantanée du modèle d'expression du gène à un moment donné en utilisant la technologie d'aujourd'hui "*DNA microarray chips*". La compréhension de la fonctionnalité génétique nécessite le dévoilement de réseaux régulateurs génétiques dans lesquels un gène participe et leur niveau d'expression au fil du temps. L'étude des réseaux régulateurs génétiques et leur relation avec la manière dont les organismes se développent et s'adaptent est connue sous le nom : la génomique fonctionnelle « *functional genomics » [Floreano, Husbands et al., 2008]*.

L'interprétation de l'ADN comme un ensemble d'instructions pour construire l'organisme, qui est souvent utilisé dans les descriptions classiques du matériel génétique et dans le calcul évolutionnaire, ne rend pas justice aux réseaux complexes d'interactions mutuelles entre les gènes et les protéines, les protéines et les cellules, les cellules et les organismes, ainsi que les organismes et l'environnement. Dans cette perspective plus large, les molécules d'ADN devraient plutôt être considérées comme la partie la plus constante d'un système dynamique et complexe qui se déroule dans un organisme. En d'autres termes, les molécules d'ADN sont la structure relativement immuable qui est transmise des parents aux enfants.

c. Mutations génétiques

Les molécules d'ADN peuvent s'échanger au moyen de mutations qui se produisent durant la réplication. Les mutations qui se produisent dans les cellules sexuelles peuvent affecter directement l'évolution des espèces. Ici, nous allons brièvement examiner certains types de mutations (voir la figure 2.7) qui sont également utilisées dans le calcul évolutionnaire.

Mutation de substitution

La mutation de substitution change un nucléotide en un autre (par exemple, de A à G ou de A à C). Dans le cas où, la modification ne correspond pas à la production d'un acide aminé différent de celui exprimé avant la mutation (le même acide aminé peut être produit par plusieurs triplets de nucléotides), la mutation est dite "synonyme" ou "silencieuse". La mutation non synonyme peut changer un codon de sorte qu'il produise un acide aminé différent [Floreano, Husbands et al., 2008].

Mutation d'inversion

Dans la mutation d'inversion, une longue séquence de double brin de la molécule d'ADN est mise en rotation de 180 degrés [Floreano, Husbands et al., 2008].

Mutation de recombinaison

La Mutation de recombinaison affecte les segments de nucléotides entre les séquences homologues de chromosomes homologues. Dans le cas de la recombinaison réciproque, les chromosomes sont croisés de sorte qu'ils échangent les séquences homologues. Dans la recombinaison non réciproque, la séquence d'un chromosome est remplacée par la séquence homologue de l'autre chromosome, mais la séquence remplacée est perdue. La mutation de recombinaison chez les individus sexuels correspond effectivement à mélanger les caractéristiques des parents parce que les chromosomes homologues sous recombinaison sont fournis séparément par le père et la mère de l'individu. Dans certains algorithmes évolutionnaires, la recombinaison assure un rôle important et il est considéré séparément de mutations [Floreano, Husbands et al., 2008].

Insertion et suppression

L'insertion et la suppression sont deux types de mutation où les longues séquences de nucléotides sont insérées ou supprimées, respectivement, dans une molécule d'ADN. Ceci peut se produire lors de la recombinaison entre les séquences mal alignées de deux chromosomes ou pendant la réplication avec le glissement d'un brin dans le même chromosome [Floreano, Husbands et al., 2008].

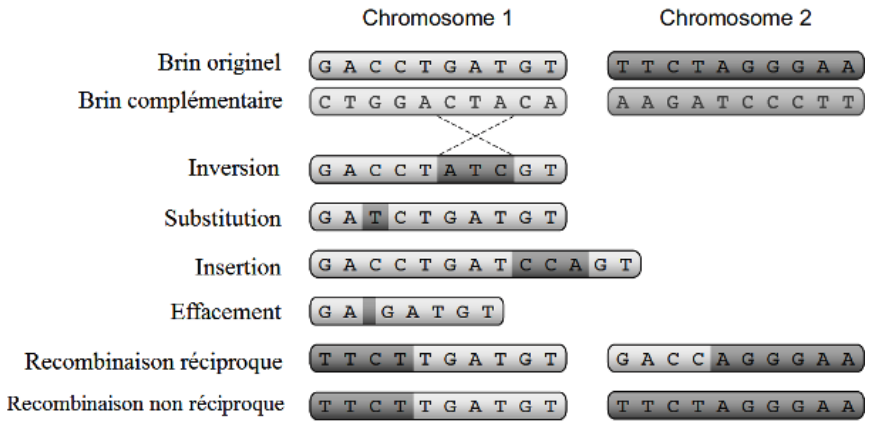


Figure 2.7- Types de mutations, dont certains impliquant l'échange de matériel de deux chromosomes homologues [Floreano and Mattiussi, 2008].

2.5.3.3. Évolution artificielle

L'évolution artificielle comprend un ensemble d'algorithmes qui s'inspirent des principes de l'évolution naturelle et de la génétique moléculaire pour trouver automatiquement des

solutions aux problèmes d'optimisation difficiles, améliorer les formes d'objets, découvrir de nouveaux programmes informatiques, concevoir des circuits électroniques et explorer plusieurs autres domaines qui sont habituellement traités par la conception humaine [Floreano and Mattiussi, 2008; Sumathi and Paneerselvam, 2010]. L'évolution artificielle est basée sur les mêmes quatre piliers de l'évolution naturelle:

1- le maintien de la population;
2- la création de la diversité;
3- les mécanismes de sélection;
4- les processus d'héritage génétique.

Dans le domaine de l'évolution artificielle, le phénotype de l'individu qui représente une solution à un problème, est soumis à un processus de sélection. Le génotype est une représentation génétique de cette solution, il est transmis au cours des générations et manipulé par les opérateurs génétiques. La mise en correspondance (mapping) entre la représentation génétique (génotype) et la description du problème (phénotype) peut prendre divers degrés de complexité allant des modèles directs à des modèles sophistiqués d'expression des gènes.

Nous devons souligner qu'il existe une différence majeure, et souvent négligée, entre l'évolution naturelle et l'évolution artificielle. Considérant que l'évolution naturelle n'a pas un objectif prédéfini et qu'elle est essentiellement un processus d'adaptation ouverte, alors que l'évolution artificielle est un processus d'optimisation qui tente de trouver des solutions à des problèmes prédéfinis. La fonction de résolution de problèmes de l'évolution artificielle est intégrée dans le processus de sélection, qui se compose de deux étapes:

1- l'évaluation du phénotype qui fournit un score quantitatif, aussi nommé, la valeur de fitness;
2- un opérateur de reproduction qui permet de générer un grand nombre de copies de génotypes qui correspondent à des phénotypes avec des valeurs de fitness élevées.

D'autre part, dans l'évolution naturelle, la fonction de fitness d'un individu est définie par son succès de reproduction (nombre de descendants), dans l'évolution artificielle la fitness d'un individu est une fonction qui permet de mesurer la performance d'un individu à résoudre un problème prédéfini.

La conséquence de cette différence est que l'évolution artificielle, telle que formulée aujourd'hui, ne peut pas éventuellement égaler la diversité et la créativité générée par

l'évolution naturelle parce que, par définition, tous les systèmes artificiellement évolués tendront à satisfaire le problème prédéfini.

Même dans le contexte de la résolution de problèmes, l'évolution artificielle est parfois critiquée par les ingénieurs, car elle contient des éléments de l'aléatoire dépourvus de preuves formelles de convergence. En effet, l'évolution artificielle est plus adaptée dans les situations où les techniques classiques d'optimisation ne peuvent pas trouver une solution satisfaisante, par exemple lorsque la fonction à optimiser est discontinue, non différentiables et/ou dans le cas de la présente de nombreux paramètres non linéairement liés.

2.5.3.4.Algorithmes évolutionnaires

a. Définition

La structure d'un algorithme évolutionnaire consiste en une procédure itérative simple appliquée sur une population d'individus génétiquement différents. Les phénotypes sont évalués selon une fonction de fitness prédéfinie, les génotypes des meilleurs individus sont copiés à plusieurs reprises et sont modifiées par les opérateurs génétiques. Les génotypes nouvellement obtenus sont insérés dans la population à la place des anciens. Cette procédure se poursuit jusqu'à ce qu'une solution "assez bonne" soit trouvée.

Les algorithmes évolutionnaires sont souvent utilisés sur des problèmes difficiles où les autres méthodes d'optimisation échouent ou sont piégées dans des solutions non optimales. Ces problèmes comprennent généralement des cas qui ont plusieurs paramètres libres avec des interactions complexes et non linéaires. Ces problèmes sont caractérisés par des fonctions non continues et présentent des données manquantes et corrompues, ou possèdent plusieurs optima locaux.

Les algorithmes évolutionnaires sont applicables dans de nombreux domaines aussi longtemps qu’une représentation génétique cohérente puisse être formulée. Les algorithmes évolutionnaires peuvent également être couplés à d'autres méthodes de recherche complémentaires pour augmenter la qualité des solutions. Par exemple, la technique "algorithme du gradient" pourrait être appliqué aux phénotypes avant l’évaluation de la fitness de sorte que le processus de sélection puisse reproduire des individus qui se trouvent dans de meilleures zones de l'espace de recherche. Les algorithmes évolutionnaires permettent également l'interaction et la collaboration avec les designers humains en laissant, par exemple, les humains substituer la fonction de fitness et sélectionner manuellement certains individus

pour la reproduction ou insérer dans la population les génotypes des individus ayant les caractéristiques désirées.

Il y a plusieurs types d'algorithmes évolutionnaires qui sont souvent différemment étiquetés pour des raisons historiques. Ces algorithmes mettent l'accent sur différents éléments, tels que le type de l'opérateur de mutation, ou sont adaptés pour certains types de problèmes tels que l'évolution des programmes d'ordinateur. Avant d'introduire les modèles des algorithmes évolutionnaires, nous allons fournir un aperçu sur les principales étapes nécessaires pour assembler un algorithme évolutionnaire approprié. Ces étapes sont:

1- Choix d'une représentation génétique;
2- Construction d'une population;
3- Élaboration d'une fonction de fitness;
4- Choix d'un opérateur de sélection;
5- Choix d'un opérateur de recombinaison;
6- Choix d'un opérateur de mutation;
7- Élaboration d'une procédure d'analyse de données.

Représentations génétiques

Une représentation génétique, également appelée codage génétique, décrit les éléments du génotype et la façon dont ces éléments sont mis en correspondance avec le phénotype. Une représentation génétique appropriée doit être conçue de sorte que:

a- les opérateurs de recombinaison et de mutation ont une probabilité élevée pour générer les meilleurs individus au cours de l'évolution et
b- l'ensemble de tous les génotypes possibles ayant une probabilité élevée de couvrir l'espace de solutions optimales pour le problème à résoudre.

Par conséquent, le choix d'une représentation génétique peut bénéficier de la connaissance des propriétés de l'espace de recherche. Nous pouvons citer, et ce n'est pas exhaustif, certaines des représentations génétiques communes: tableaux binaires, arbres génétiques, arbres syntaxiques, arbres binaires et langages naturels [Floreano and Mattiussi, 2008].

Population initiale

La population initiale doit être suffisamment large et diversifiée pour être sûr que les individus affichent des valeurs de fitness différentes parce que, si tous les individus ont la

même fitness, la sélection ne peut pas fonctionner correctement. La taille de la population dépend de deux paramètres:

a- les propriétés de l'espace de recherche;

b- le coût de calcul d'évaluation de tous les individus pour plusieurs générations.

Les problèmes pour lesquels la plupart des génotypes ont la même fitness ou pour lesquels les mutations aléatoires ont une faible probabilité pour générer une amélioration de la fitness nécessitent des populations plus grandes. Dans la plupart des cas, la taille de la population initiale est empiriquement déterminée mais peut être déterminée par les coûts de calcul. Dans la littérature, nous trouvons souvent des populations allant de plusieurs centaines à quelques milliers d'individus. Si l'évaluation d'un individu nécessite des expérimentations du monde réel, comme dans le cas d'évolution de robot, la taille de la population est souvent plus petite qu'une centaine d'individus [Floreano and Mattiussi, 2008].

Fonction de fitness

La fonction de fitness associe un score numérique à chaque phénotype dans la population. Il existe deux aspects importants intervenant dans la conception d'une fonction de fitness:

a- le choix et la combinaison des composants de la fonction de fitness;

b- la manière dont la fonction de fitness est évaluée.

L'évaluation de la fonction de fitness des individus est souvent la partie la plus fastidieuse d'un algorithme évolutionnaire parce que la qualité des solutions évoluées dépend de la façon d'évaluation exhaustive des individus.

- Objectifs multiples

Les fonctions de fitness tentent souvent d'optimiser des objectifs multiples d'un problème posé. Bien que l'optimisation multi-objective dans le contexte du calcul évolutionnaire ait été largement discutée, il n'y a pas une manière standard de combiner et pondérer les différents objectifs, sauf si on a une certaine connaissance des propriétés de l'espace de recherche qui mettent en lumière les relations entre les composants. Le choix est souvent arbitraire, basé sur une expérience antérieure ou sur le résultat d'une procédure "d'essai-erreur" [Floreano and Mattiussi, 2008].

- Fitness subjective

La fitness subjective est le nom utilisé lorsque des observateurs humains estiment la performance de l'évolution des individus par une inspection visuelle. L'utilisateur peut

sélectionner des individus pour la reproduction en cliquant sur leurs formes. La fitness subjective est souvent utilisée dans les domaines artistiques, tels que l'évolution de l'art figuratif et les structures architecturales, où il est difficile de formaliser les qualités esthétiques en fitness objectives. La fitness subjective peut également être combinée avec la fitness objective [Floreano and Mattiussi, 2008].

Sélection

Le rôle de la sélection est d'allouer un plus grand nombre de descendants aux meilleurs individus de la population. Différentes méthodes de sélection des individus existent selon les algorithmes :

- Sélection par tournoi: consiste à effectuer un nombre arbitraire de tournois entre n individus ($n \geq 2$) et de choisir le ou les plus performants. Plus n augmente, plus ce type de sélection induit une pression importante, étant donné que le tournoi a plus de chance d'impliquer les individus de la population ayant les plus hautes valeurs de fitness.
- Sélection par roulette: donne à chaque individu une probabilité d'être sélectionné proportionnellement à sa performance.
- Sélection steady-state : ce n'est pas une méthode particulière de sélection des chromosomes parents. L'idée principale est qu'une grande partie de la population puisse survivre à la prochaine génération. L'algorithme évolutionnaire fonctionne alors de la manière suivante: A chaque génération, sont sélectionnés quelques chromosomes (parmi ceux qui ont le meilleur coût) pour créer des chromosomes fils. Ensuite les chromosomes les plus mauvais sont retirés et remplacés par les nouveaux. Le reste de la population survit à la nouvelle génération.
- La sélection par rang : dans ce cas, la population est d'abord triée par rapport à la fitness. Chaque individu se voit associé un rang en fonction de sa position. La sélection se fait ensuite de la même manière que pour la roulette, mais les proportions sont en relation avec le rang plutôt qu'avec la valeur de l'évaluation.
- L’élitisme : à la création d'une nouvelle population, il y a de grandes chances que les meilleurs chromosomes soient perdus après les opérations de croisement et de mutation. Pour éviter cela, on utilise la méthode d'élitisme. Elle consiste à copier un ou plusieurs des meilleurs chromosomes dans la nouvelle génération. Ensuite, on génère le reste de la population selon l'algorithme de reproduction usuel. Cette méthode améliore

considérablement les algorithmes génétiques, car elle permet de ne pas perdre les meilleures solutions.

Opérateurs génétique

Les opérateurs génétiques capturent les effets des mutations biologiques sur le génotype. Dans cette section, nous allons décrire les opérateurs génétiques qui sont applicables aux représentations génétiques. Ces opérateurs sont conçus pour modifier les génotypes dont la longueur est fixe et qui n'incluent que les régions codantes. D'autres types d'opérateurs génétiques peuvent modifier la longueur des chaînes génétiques, tels que la suppression, l'insertion et la duplication qui sont utilisés pour des représentations plus avancées et qui sont adaptées à faire évoluer des problèmes spécifiques, tels que l'évolution de circuits électroniques analogiques [Floreano and Mattiussi, 2008].

Les opérateurs génétiques introduisent la diversité dans la population et permettent l'exploration de nouvelles solutions. La combinaison, ou le croisement, du matériel génétique des deux parents permet, sous certaines conditions, d'exploiter les blocs de construction génétique utiles pour les deux parents.

- Croisement

Le croisement assure que le descendant hérite des caractéristiques des parents en créant des recombinaisons paires des génomes d'individus sélectionnés. Cet opérateur est également

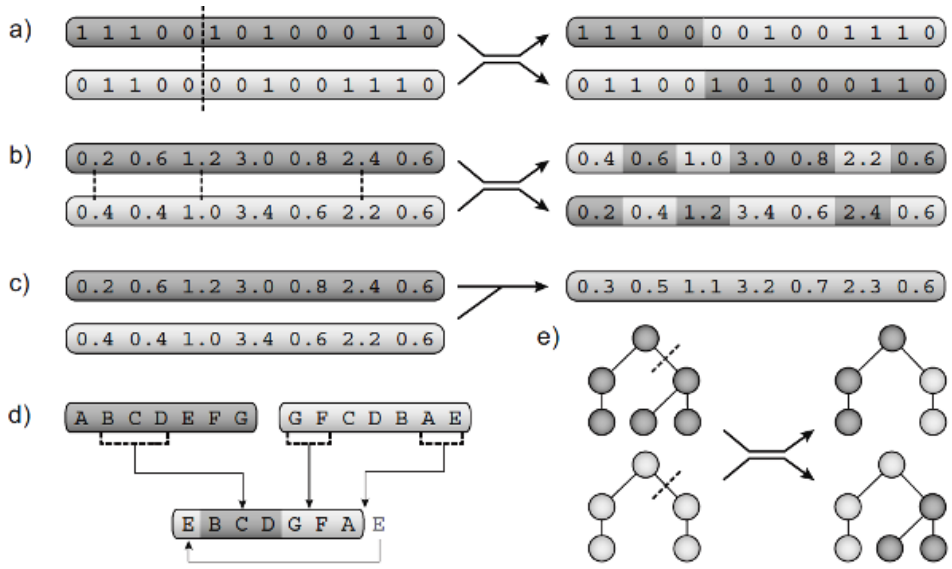


Figure 2. 8- Exemples d'opérateurs de croisement. a) one-point; b) uniforme; c) arithmétique; d) pour des séquences; e) Pour des arbres[Floreano and Mattiussi, 2008].

connu sous le nom de recombinaison.

Les descendants nouvellement créés sont appariés au hasard et les parties de leurs génotypes sont inversés par l'opérateur de croisement avec une probabilité p_c. L'opérateur de croisement peut avoir différentes formes (voir la figure 2.8), qui sont adaptés aux représentations génétiques. Le croisement du matériel génétique des deux parents permet d'exploiter les blocs de construction génétique utiles dans les deux parents.

- Mutation

La mutation fonctionne au niveau de l'individu. Les mutations sont de petites modifications aléatoires appliquées sur le génotype qui permettent à l'évolution d'explorer les variations des solutions existantes. L'opérateur de mutation doit être conçu de sorte que tous les points dans l'espace de la représentation génétique pourraient être potentiellement atteints. Les mutations sont utiles pour éviter les minima locaux et réaliser de nouveaux progrès dans les populations très convergentes où la recombinaison génétique a peu d'influence. Cependant, le nombre et la taille des mutations devraient être relativement bas pour prévenir la perte des solutions précédemment découvertes. La figure 2.9 présente les type de mutation les plus communes.

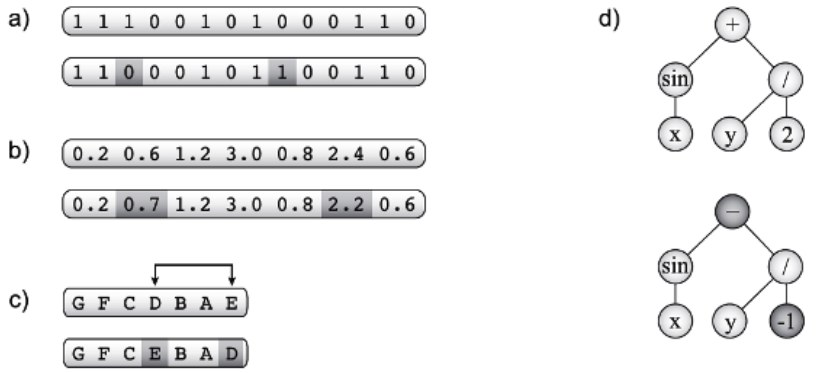


Figure 2. 9- Exemple de mutations. a) Commutation d'une position binaire; b) Ajout d'une valeur aléatoire en une position dans les représentations de valeur réelle; c) Permutation du contenu de deux positions dans une représentation de séquence; d) pour des arbres[Floreano and Mattiussi, 2008].

b. Principe de fonctionnement des algorithmes évolutionnaires

Une population de structures est maintenue pour l'évolution en fonction de règles de sélection et d'autres opérateurs, tels que la recombinaison et la mutation. Chaque individu de la population reçoit une mesure de sa fitness dans l'environnement. Durant la sélection, plus

d'attention est concentrée sur les individus ayant une grande fitness, exploitant ainsi les informations de fitness disponible. Les processus de recombinaison et de mutation perturbent ces individus, fournissant ainsi une heuristique générale pour l'exploration.

Bien que les scientifiques de la vie voient que ces algorithmes sont simples, ils sont suffisamment composites pour fournir des mécanismes de recherche d'adaptation robustes et puissants. La figure 2.10 décrit un algorithme évolutionnaire typique. La population est initialisée et subit ensuite un processus d'évolution d'une génération à une autre et ce par les opérations répétées comme l'évaluation, la sélection, la recombinaison et mutation. La taille de la population (N) est généralement constante dans un algorithme évolutionnaire. Généralement, un algorithme évolutionnaire initialise sa population au hasard, bien que la connaissance d'un domaine spécifique puisse également être utilisée pour solliciter la recherche. Le processus d'évaluation mesure la fonction de fitness de chaque individu. Ce processus peut être simple tel que le calcul d'une fonction de fitness ou aussi complexe tel que l'exécution d'une simulation élaborée. La sélection est souvent effectuée en deux étapes, la sélection des parents et de survie. La sélection des parents décide qui devient parents et détermine leur nombre d'enfants. Les enfants sont créés par une recombinaison qui échange les informations entre les parents, et la mutation, qui perturbe les enfants. Les enfants sont ensuite évalués. Enfin, l'étape de survie décide qui va survivre dans la population.

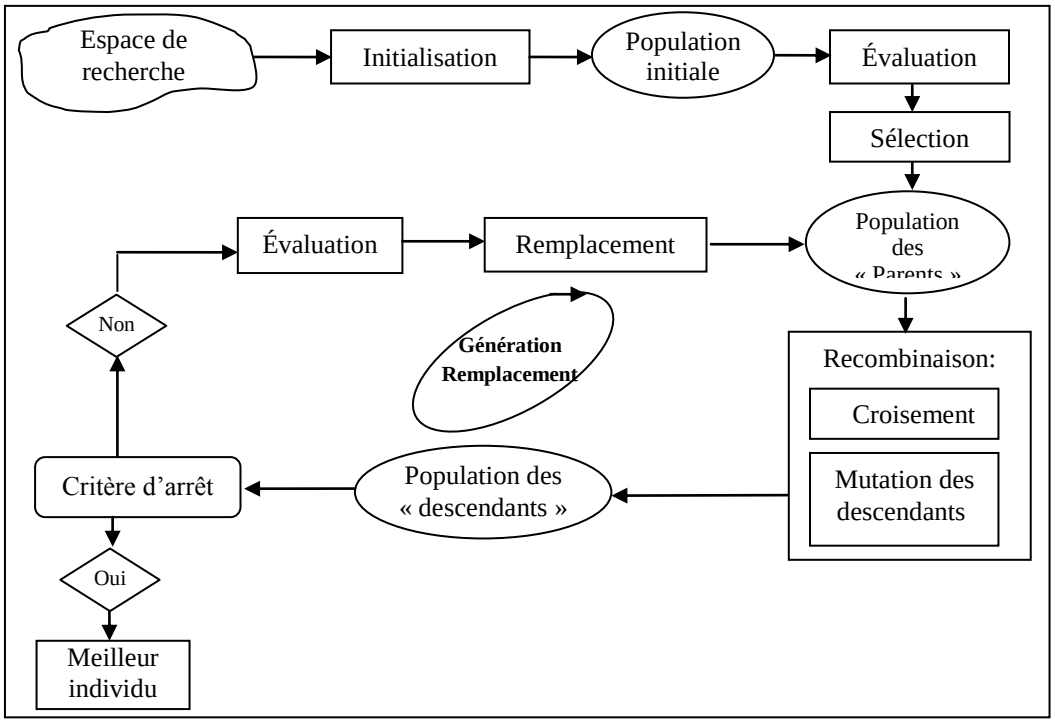


Figure 2. 10- Diagramme des algorithmes évolutionnaires

c. Modèles des algorithmes évolutionnaires

Les origines des algorithmes évolutionnaires remontent à la fin des années 50. Les méthodes qui ont émergé au cours des dernières décennies tels que: "la programmation évolutive (EP) [Fogel, Owens et al., 1966]", "les stratégies d'évolution (ES) [Rechenberg, 1973] ", "la programmation génétique (GP) [Koza, 1992] " et "les algorithmes génétiques (GA) [Holland, 1975]» sont discutés dans cette section. Bien que ces techniques présentent des similarités, chacune de ces méthodes implémente l'algorithme à sa manière unique. Les différences principales se trouvent presque sur tous les aspects des algorithmes évolutionnaires, y compris les choix de représentation pour les structures des individus, les types de mécanisme de sélection utilisés, les formes des opérateurs génétiques et les mesures de performance. Les différences et similarités importantes sont également illustrées dans cette section, en examinant une partie de la variété représentée par la famille actuelle des algorithmes évolutionnaires.

- Les Algorithmes Génétiques (AG)

Les Algorithmes Génétiques (AG), développés à l'Université de Michigan (USA) par (J. Holland 1975, Goldberg 1989, L. Davis 1991 et Michalewicz 1992). Ils ont été imaginés comme outils de modélisation de l'adaptation. Ce sont les plus connus des algorithmes évolutionnaires; ils favorisent l'utilisation du croisement comme principal opérateur de recherche. Ils utilisent cependant la mutation avec un faible pourcentage de probabilité, et une méthode de sélection de type probabiliste dans laquelle la probabilité de sélection est proportionnelle en fonction de l'adaptation de l'individu. La représentation des individus génotype, qui est à l'origine de type binaire, a été par la suite développée en de nombreuses autres formes de représentation [Dupas, 2004; Sumathi and Paneerselvam, 2010].

- Les Stratégies d'Évolution (SE)

Les Stratégies d'Évolution (SE), (développées par Rechenberg et H.P. Schwefel, 1965, Berlin), ont pour objectif de résoudre des problèmes d'optimisation à variables réelles rencontrés dans les milieux industriels et pour lesquels il n'existe pas de fonction objective analytique; le contexte étant l'optimisation paramétrique. Ce sont les meilleurs algorithmes pour les problèmes purement numériques. Ce modèle de stratégies d'évolution utilise le principe de mutation sur les réels du modèle de la programmation évolutive avec un taux de mutation plus grand. Cette augmentation peut être interprétée par le fait que si la proportion

de mutation réussie est élevée, l'espace de recherche exploré est limité autour d'un optimum local; il faut donc diversifier la population en augmentant le taux de mutation. Ces approches utilisent un opérateur de sélection de type déterministe, les solutions dont la fonction d'adaptation est mauvaise sont éliminées de la population. En outre, dans le modèle originel, les populations des parents et de leurs descendants sont généralement de taille différente [Dupas, 2004; Sumathi and Paneerselvam, 2010].

- Programmation Évolutionnaire (PE)

La programmation évolutionnaire (PE) a été développée par (L.J. Fogel, 1964 et D.B. Fogel, 1991, 1995, Californie, USA). Ce modèle évolutionnaire accentue l'utilisation de la mutation et n'utilise pas dans sa version originale la recombinaison des individus par croisement. Développé à l'origine pour l'évolution des automates à état fini, ce modèle est souvent appliqué à la résolution de problèmes d'optimisation à variables réelles dans des espaces de recherche très variés. L'idée consiste à faire subir des mutations importantes aux mauvais individus et des mutations faibles aux bons individus. L'opérateur de sélection est de type probabiliste. Il est à noter que la représentation des individus n'a pas une forme spécifique de génome telle que dans une représentation linéaire de type chaîne binaire par exemple [Dupas, 2004; Sumathi and Paneerselvam, 2010].

- Programmation Génétique (PG)

La programmation génétique (PG) a été développée par (J. Koza, 1990, Californie, USA). Apparue initialement comme une extension du modèle d'apprentissage des algorithmes génétiques, elle est devenue une branche à part entière. La (PG) permet de générer des fonctions informatiques à partir des principes évolutionnaires, la population est un ensemble de codes de base de programmes informatiques. La spécificité des (PG) est l'espace de recherche où les individus formant une population sont des programmes candidats à la résolution d'un problème. Ces programmes sont exprimés sous la forme d'arbres sur lesquels des opérateurs génétiques produisent des transformations en vue d'obtenir un programme qui satisfait la résolution du problème choisi. Les (PG) cherchent à atteindre un des vieux rêves des programmeurs, faire écrire le programme par un autre programme [Dupas, 2004; Sumathi and Paneerselvam, 2010].

Les méthodes les plus répandues sont les algorithmes génétiques (AG). Globalement, les différences entre ces méthodes résident dans la stratégie de résolution (Bäck93, Park94,

Fogel94). Les (AG) sont considérés comme des méthodes de résolution "ascendantes", c'est-à-dire que la solution optimale peut être obtenue progressivement en assemblant des parties optimales des solutions partielles, les opérateurs de recombinaison jouent alors un rôle essentiel. Les stratégies évolutionnistes et la programmation évolutionniste sont vues comme des méthodes "descendantes", dans lesquelles l'environnement agit par pression pour faire apparaître la solution optimale. Les opérateurs de recombinaison y ont un rôle secondaire (Fogel94) [Francisci, 2002].

Les premières applications étaient marquées par l'utilisation de chaînes binaires de longueur fixe pour les AG, des vecteurs de réels pour les stratégies évolutionnistes, un automate à état fini pour la programmation évolutionnaire et des codes de programme pour la programmation génétique. Ceci jusqu'au début des années 80 où les recherches étaient principalement théoriques avec peu d'applications réelles (Goldberg89). Ensuite, après de nombreuses investigations, chaque nouvelle application apporta une nouvelle perspective à l'amélioration de la théorie [Francisci, 2002]. Dans l'état actuel des recherches, les algorithmes évolutionnaires sont encore à l'état de développement sur les fondements théoriques, notamment dans la résolution de problèmes d'optimisation et d'apprentissage.

2.6. Conclusion

Dans ce chapitre, nous avons introduit des généralités sur la vie artificielle et son émergence come une discipline. Nous avons ensuite fait un rappel sur ses différents systèmes. Ainsi, ce chapitre nous a permis de construire une vue générale sur les concepts de la biologie évolutionnaire et leurs mécanismes de fonctionnement afin de faciliter la lecture de la suite de ce mémoire.

D’autre part, les systèmes de la vie artificielle peuvent être combinés dans le but de créer des systèmes et résoudre des problèmes plus complexes. On trouve des applications qui utilisent des systèmes d’évolution pour la création de générateurs capables de produire le résultat recherché. Cela est le cas dans les approches appliquées à la génération de créatures artificielles (Voir le chapitre suivant). On peut aussi trouver des systèmes d’apprentissage tels que les réseaux de neurones dont la topologie est produite par évolution (dont une application est proposée dans le cinquième chapitre).

Selon un autre axe, la quasi-totalité des techniques de la vie artificielle sont utilisées avec des modifications de leur inspiration d'origine dans le seul objectif de produire des systèmes plus efficaces qui n'ont plus rien à voir avec la réalité des systèmes qui les sous-tendent.

Enfin, leur utilisabilité s'est accrue de pair avec l'accroissement de la performance des machines. Cette évolution leur permet aujourd'hui de s'attaquer à des problématiques jugées auparavant inatteignables.

Dans le chapitre suivant, nous allons passer en revue les différentes applications de ces systèmes dans le domaine de la robotique et de la simulation comportementale.

3.Génération bio inspirée de comportements dans un espace virtuel

3.1. Introduction

Les robots autonomes ont toujours attiré l'attention des chercheurs dans le monde entier. Le défi principal de la robotique traditionnelle est de concevoir un logiciel pour un robot physique donné de telle manière, que le robot équipé de ce logiciel soit capable de réaliser des tâches données, d'une manière autonome.

Plusieurs approches ont été introduites pour la conception de systèmes de contrôle robotique. Dans le domaine de la conception manuelle des contrôleurs robotiques, les méthodes vont des techniques de planification classiques (c.-à-d. STRIPS [Fikes and Nilsson, 1972]) à la planification réactive (la robotique à base de comportement; l'architecture de subsomption de Brooks [Brooks, 1986]). Une autre approche consiste à créer automatiquement des contrôleurs robotiques pour effectuer une tâche donnée dans un environnement simulé. Des exemples incluent l'algorithme d'apprentissage de rétro-propagation pour les réseaux de neurones artificiels ou les méthodes de recherche évolutionnaire, inspirées par l'évolution naturelle.

Les environnements simulés sont de bonnes fondations pour réaliser des expériences avec la structure physique du robot (il est plus facile de modifier le corps d'un robot dans un environnement simulé que dans la réalité). Par conséquent, plusieurs méthodes d'évolution de la morphologie et du système de contrôle de robots ont été proposées. Certains travaux ont même transféré les robots simulés en des robots réels. Ce chapitre fournit une introduction au domaine de l'évolution des créatures virtuelles et résume brièvement les méthodes d'évolution des organismes robotiques et leurs systèmes de contrôle. Les références proposées permettant de trouver l'information nécessaire à une compréhension plus complète des systèmes décrits.

3.2. Agent et comportement adaptatif

La recherche de comportements adaptatifs concerne l'étude de la façon dont les organismes peuvent faire évoluer des comportements en s'adaptant à l'environnement et à la

tâche qu'ils doivent accomplir d'une façon autonome. Cet objectif est atteint grâce à une méthodologie synthétique, c.-à-d. par la synthèse des créatures artificielles qui possèdent les caractéristiques suivantes:

1- avoir un corps;
2- être situé dans un environnement avec lequel ils interagissent;
3- posséder des caractéristiques qui varient durant le processus d'adaptation.

Dans le reste du chapitre, nous utiliserons le terme «agent» pour identifier les créatures artificielles qui possèdent les deux premières caractéristiques décrites ci-dessus et le terme «agent adaptatif» pour indiquer les créatures artificielles qui possèdent également la troisième caractéristique en addition aux deux premières.

Les agents et l'environnement peuvent être simulés ou réels. Dans le premier cas, les caractéristiques du corps d'agent, le moteur et le système sensoriel, les caractéristiques de l'environnement, et les règles qui régulent les interactions entre tous les éléments sont simulées sur un ordinateur. Ce type d'agent doit prendre ses décisions localement en fonction de la perception qu'il a de son environnement. Ceci permet de supprimer la vision globale du système qui, en plus de son manque de réalisme, est impossible dans un environnement virtuel possédant un grand nombre d'acteurs virtuels, comme dans une foule par exemple [Cussat-Blanc, 2009]. Dans le second cas, les agents sont constitués d'entités physiques (ex. robots mobiles) situées dans un environnement physique avec lesquelles ils interagissent sur la base des lois de la physique.

Les propriétés d'un agent

Un grand nombre de propriétés permettent de définir un agent. David Panzoli a donné dans sa thèse [Panzoli, 2008] la liste des principales propriétés suivantes :

- L'autonomie permet à l'agent de sélectionner les actions qu'il va accomplir sans intervention extérieure (et en particulier humaine). Pour cela, il doit posséder une représentation du problème, de son environnement ainsi que de son état interne.

  La propriété d'autonomie est primordiale pour un agent virtuel. Elle implique un mécanisme de sélection de l'action à effectuer par l'agent qui doit prendre en compte ses capacités (les actions que celui-ci est capable de réaliser). Ce mécanisme de sélection

d'action est souvent représenté dans une boucle Perception-Décision-Action que nous verrons par la suite [Cussat-Blanc, 2009].

- La réactivité est la capacité d'un agent à agir rapidement aux changements qui peuvent affecter l'environnement, et elle lui permet également de saisir les opportunités. La réactivité offre également le temps de réponse à l'agent lorsqu'il est confronté à un problème
- L'intentionnalité permet à l'agent de produire des plans à partir de buts explicites. Une intention exprime donc la volonté d'un agent d'atteindre un but ou d'effectuer une action. Elle peut s'opposer à la propriété de réactivité si le temps de production de ces plans est trop long.
- La sociabilité est la capacité de l'agent à interagir et à communiquer avec les autres agents de l'environnement.
- La situation: le comportement de l'agent est défini par son environnement au moyen de l'évolution ou l'apprentissage. La situation dénote l'importance de l'environnement pour la définition du comportement de l'agent.
- L'adaptabilité est l'aptitude de l'agent à s'adapter aux modifications de son environnement.

Le modèle général comportemental d'un agent

Dans le but de sélectionner la meilleure action à effectuer à chaque instant et montrer un comportement adaptatif, réaliste et efficace, la grande majorité des agents sont régis par un mécanisme de décision local, schématisé dans la figure 3.1. Ce schéma est synthétisé à travers une boucle (Perception-Décision-Action) qui se répète tant que la simulation n'est pas arrêtée, que l'agent est vivant ou qu'il n'a pas atteint son but. Cette boucle est composée de trois phases [Panzoli, 2008]:

1. Une phase de perception, durant laquelle l'agent détermine l'état de son environnement et les transforme en valeurs exploitables à l'aide de différents capteurs spécialisés.
2. Une phase de décision, qui a pour but de prendre en compte la perception précédemment réalisée pour choisir la meilleure action à effectuer en fonction du but de l'agent, de son environnement perceptible et de son état interne [Cussat-Blanc, 2009],

3. Une phase d'action, qui consiste à exécuter l'action précédemment choisie à travers les effecteurs de l'agent. Ces actions vont modifier l'état de l'environnement, l'état interne de l'agent et permettre ainsi d'agir sur le comportement de l'utilisateur.

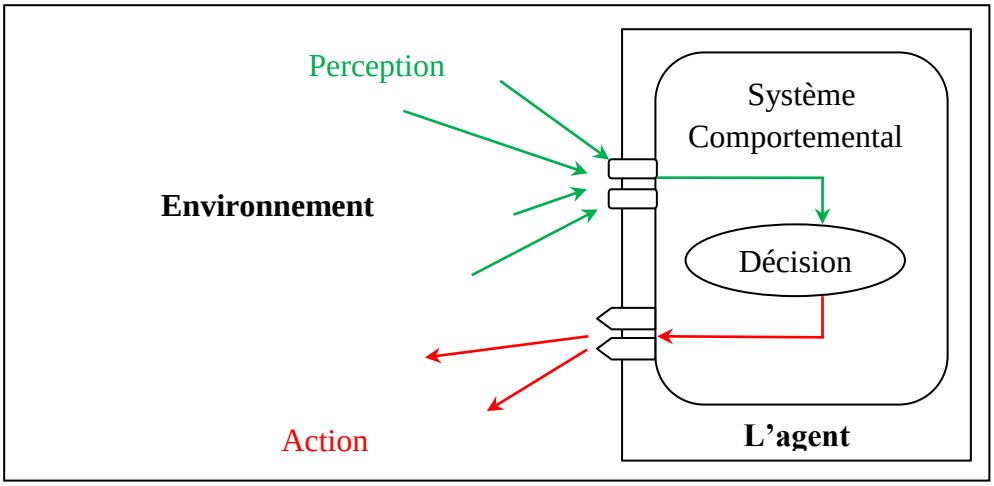


Figure 3.1- La boucle Perception-Décision-Action d'un agent virtuel lui permet de sélectionner la meilleure action en fonction de son environnement local et de son but propre.

3.3. Incarnation et Situation

La notion de l'incarnation[11] et de la situation [12] a été introduite pour caractériser les systèmes (c.-à-d. les organismes naturels et les robots) qui ont un corps physique et qui sont situés dans un environnement physique avec lequel ils interagissent. Dans ce qui suit, nous allons examiner brièvement les implications générales de ces deux propriétés fondamentales.

Une première implication importante d'être incarné et situé consiste dans le fait que ces agents et leurs parties qui se caractérisent par leurs propriétés physiques (i.e. le poids, la dimension, la forme, l'élasticité, etc..), sont soumis aux lois de la physique (ex. l'inertie, la friction, la gravité, la consommation d'énergie, la détérioration etc.), et interagissent avec l'environnement à travers la matière physique. Leur nature physique implique également qu'ils sont quantitatifs dans l'état et le temps. Le fait que ces agents soient quantitatifs en temps implique, par exemple, que les articulations qui relient les parties d'un bras robotisé peuvent assumer n'importe quelle position possible dans une plage donnée et que les effets de l'application d'une force à une jointure dépendent de la durée de son application.

[11] Ici on parle « embodiment »

[12] Ici on parle « situatedness »

Une seconde implication importante réside dans le fait que l'information mesurée par les capteurs n'est pas seulement une fonction de l'environnement, mais également de la position relative de l'agent dans l'environnement. Ceci implique que les actions effectuées par l'agent, en modifiant la relation "agent/environnement" ou l'environnement, co-déterminent les expériences sensorielles de l'agent.

Une troisième implication importante est que l'information mesurée par les capteurs fournit des informations sur l'environnement externe qui est égocentrique (dépend de la position actuelle et l'orientation de l'agent dans l'environnement), locale (fournir uniquement des informations relatives à la partie observable locale de l'environnement), incomplète (à cause de l'occlusion visuelle, par exemple) et soumis à un bruit. Des caractéristiques similaires s'appliquent aux actions des moteurs produites par les effecteurs de l'agent [Nolfi, 2009].

3.4. Les différentes approches de production de comportements

La manière de traitement de l'information parvenant au système comportemental permet de discriminer entre les principales approches de production des comportements cohérents et proches de ceux attendus. La partie suivante introduit ces différentes approches.

3.4.1. L'approche délibérative

L'approche délibérative, également appelée cognitive est issue de l'intelligence artificielle. Dans cette approche, le système de contrôle (SC) est organisé en couches horizontales, dont chacune représente une étape du processus de sélection de l'action à réaliser, comme il est illustré à la figure 3.2.

Les systèmes comportementaux délibératifs permettent à un agent au niveau de la couche de modélisation d'avoir une représentation générale de l'environnement, appelée modèle cognitif, afin de prédire son évolution en utilisant les informations issues de la couche perceptive qui est constituée d'un ensemble de capteurs.

La couche de planification du système de contrôle est la phase délibérative qui est considérée comme la partie cruciale dans la sélection de l'action. A partir de la situation de l'agent déterminée durant la phase précédente, cette couche permet de planifier la séquence des actions nécessaire pour satisfaire les buts courants de cet agent basé sur des techniques de

l'intelligence artificielle telles que le raisonnement à base de cas, la logique floue, les systèmes experts. Enfin, l'agent délibératif agit en fonction de ce plan à travers les effecteurs.

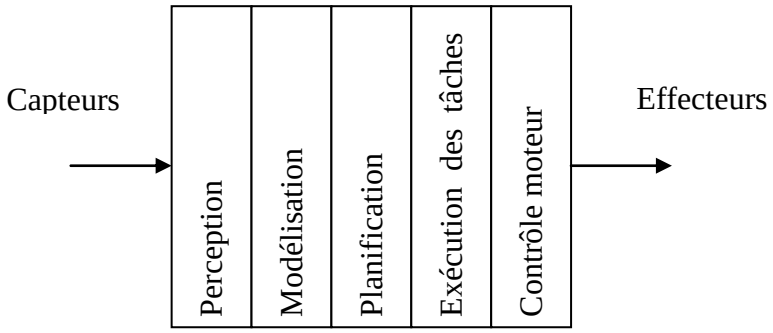


Figure 3. 2- L'architecture en couches de l'agent cognitif lui permettant d'avoir une représentation du monde avant de faire une planification dessus qui lui permettra de choisir la ou les actions à déclencher.

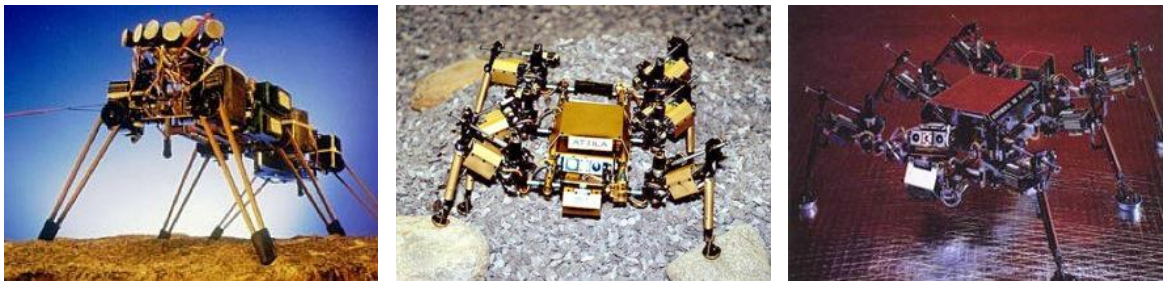
L'analyse a priori du monde ne prend pas en compte les modifications mineurs de l'environnement non prévu durant la création du système de contrôle, ce qui nécessite la recréation de ce dernier à cause de l'indisponibilité de la possibilité de le récréer automatiquement.

La modélisation et la planification sont des algorithmes coûteux en temps de calcul et il est difficile de garantir leur temps d'exécution. De plus, il est obligatoire de les relancer intégralement après chaque itération [Panzoli, 2008], ainsi le recalcule de la séquence optimale des actions dans le cas de modification de la perception sont les autres défauts de cette approche. Ceci entraine l'incapacité de choisir la bonne action dans un court temps de calcul et qui ne permet pas aux agents délibératifs d'intégré facilement dans les systèmes temps réels tels que les robots, car ils doivent parfois prendre des décisions rapides afin de protéger leur intégrité [Panzoli, 2008; Cussat-Blanc, 2009].

3.4.2. L'approche réactive

En 1991, Rodney Brooks critiqua le manque de réactivité de l'approche délibérative et il postula que la meilleure représentation de l'environnement est l'environnement lui-même. Il a également défini une architecture dite de « subsomption » comme une politique de coordination entre plusieurs comportements indépendants organisés en couches. Ces couches sont reliées directement au capteur de l'agent tandis que chacune de ces couches confinées à une tâche spécifique telle que l'évitement d'obstacle, l'exploration, la cartographie, la planification…, etc. Ces comportements sont en compétition permanente pour le contrôle de

l'agent sauf ce qui sert le même but. Cette compétition, impliquant un système d'arbitrage, permet de décider du comportement convenant à une situation particulière dans l'environnement. L'arbitre envoie ensuite les actions à effectuer aux effecteurs. Cette approche n'utilise aucune représentation interne de l'environnement facilitant ainsi le travail de l'agent [Panzoli, 2008; Cussat-Blanc, 2009].



(a) Genghis (b) Attila (c) Hannibal

Figure 3. 3- Les robots de Rodney Brooks.

D'autres techniques issues de la vie artificielle ont été utilisée pour la première fois pour la génération des comportements adaptatifs à la fin des années 80 par Reynolds [Reynolds, 1987]. Reynolds a développé un modèle qui permet de simuler une nuée, un troupeau

(a) Les boids de Reynolds sont très convaincants en groupe.

(b) Dans « Batman Returns », des centaines de boids envahissent les rues de Gotham City.

Figure 3. 4- Les boids de Reynolds et leur utilisation au cinéma [Reynolds, 1987]

d'individu se déplaçant dans un environnement visuellement très réaliste basé sur des agents autonome produisant des comportements très simples, appelé les *boids*, dont le comportement est définie par trois règles:

1. Evitement des collisions avec les autres *boids* ;

2. Alignement de la vitesse avec la vitesse globale du groupe ;
3. Centrage de la position du *boids* par rapport à son groupe.

Les travaux suivants vont bien plus loin dans l'idée d'adapter la créature à son comportement. Parmi les nombreux mécanismes de la vie artificielle, il en est un qui est largement exploité par la vie artificielle, les méthodes adaptatives en l'occurrence.

L'intégration de travaux de la vie artificielle en animation a été initiée par Karl Sims. Des travaux préliminaires [Sims, 1991] sur la production d'images avaient laissé penser que les techniques évolutionnaires répondaient parfaitement à l'attente de l'animation en terme de réalisme. Ses créatures [Sims, 1994a; Sims, 1994b], qui font encore référence aujourd'hui, sont basées sur l'évolution génétique de l'agent dans un environnement physique. Il en résulte des mouvements incroyablement réalistes et l'impression d'observer des créatures véritablement vivantes [Panzoli, 2008].

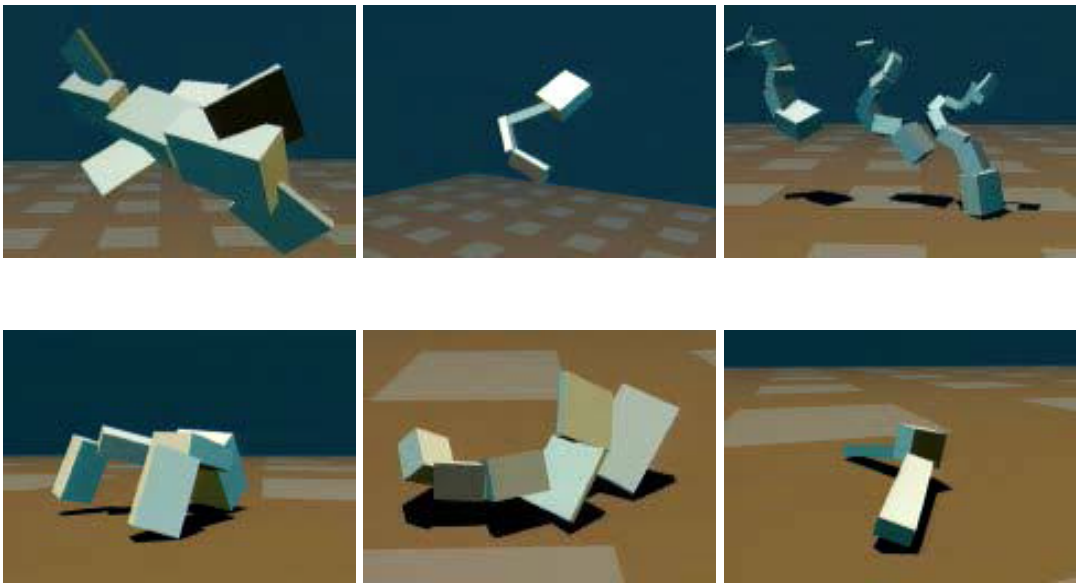
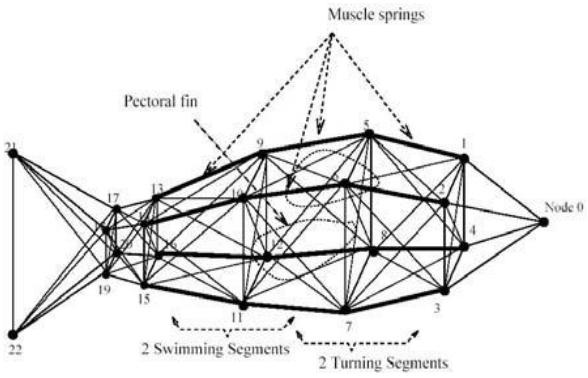


Figure 3. 5- Les créatures de Karl Sims sont capables de se déplacer aussi bien dans l'eau que sur terre, car elles ont évolué dans ce but.

Demetri Terzopoulos est l'un des premiers à avoir utilisé cette technique dans la simulation comportementale [Tu and Terzopoulos, 1994]. Il développa des poissons dont le squelette est dessiné par la main de l'homme (voir la figure 3.6 (a)) et dont le comportement défini par des réseaux de neurones évolue à l'aide d'un algorithme génétique. Il obtient grâce à cela des poissons développant une nage très réaliste dans un simulateur physique (voir la figure 3.6 (b)) [Cussat-Blanc, 2009].



(a) Le modèle utilisé pour construire le poisson

(b) Le poisson de Terzopoulos dans son environnement aquatique

Figure 3. 6- Les poissons de Terzopoulos [Tu and Terzopoulos, 1994].

L'avènement des réseaux de neurones [McCulloch and Pitts, 1943; Rosenblatt, 1958; Hopfield, 1982] et des systèmes de classeurs [Holland, 1975; Golberg, 1989; Wilson, 1994] couplés aux algorithmes évolutionnaire a permis une complexification des comportements des agents virtuels. Des comportements de fuite et de poursuite [Cliff and Miller, 1995a; Cliff and Miller, 1995b], des stratégies d'équipe [Sanza, 2001] ou d'anticipation [Panzoli, Luga et al., 2008] ont émergé en n'exprimant que le but global du système [Cussat-Blanc, 2009].

L'approche réactive a permis aux agents réactifs de rendre la phase de décision plus rapide pour sélectionner l'action que celle des agents délibératifs, mais elle ne leur offrait pas la possibilité d'atteindre des comportements complexes. L'approche hybride, présentée dans la partie suivante, a tenté de bénéficier des avantages des deux approches précédentes.

3.4.3. L'approche hybride

L'approche hybride consiste à concilier les propriétés des deux approches précédentes basées sur une architecture composée de trois couches (figure 3.7):

- Une couche délibérative : comporte une représentation de l'environnement, et des capacités d'inférences sur les connaissances. Cette couche permettant de planter des actions sur le long terme.
- Une couche réactive: elle est mise en œuvre lorsque l'agent hybride doit réagir rapidement.
- Une couche médiatrice appelée séquenceur jouant le rôle d'un conciliateur entre les entre les comportements délibératifs et les comportements réactifs. L'agent hybride est capable de planifier une action sur le long terme, en utilisant les connaissances qu'il possède de

son environnement, mais il est parallèlement capable de réagir rapidement, lorsqu'il rencontre un danger par exemple. Cette architecture est beaucoup plus utilisée dans le contrôle d'humanoïdes virtuels.

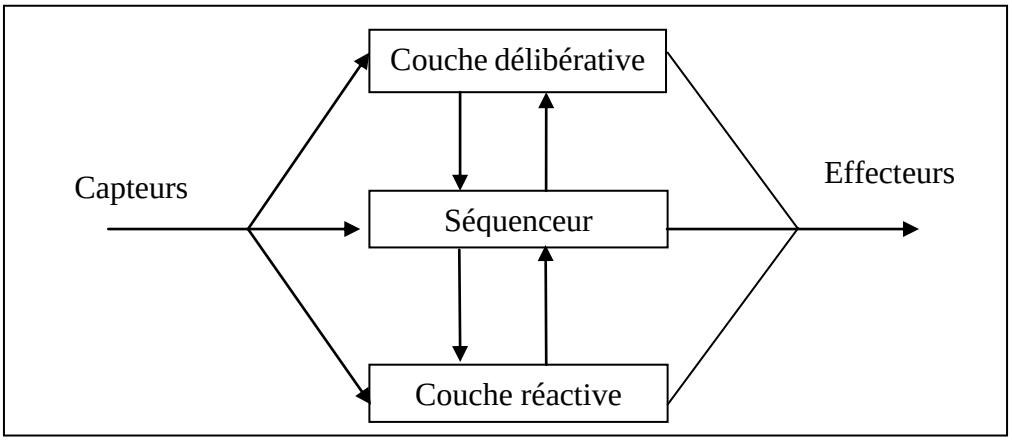


Figure 3. 7- Le modèle hybride ou TLA (Three Layers Architecture) comporte trois couches.

3.5. Méthodes adaptatives

Dans cette section, nous passons brièvement en revue les méthodes par lesquelles les agents incarnés et situés peuvent générer des comportements efficaces, intelligents et ce d'une manière autonome. Ces méthodes sont inspirées du processus adaptatif observé dans la nature: l'évolution, la maturation, le développement et l'apprentissage.

Nous allons nous concentrer en particulier sur les méthodes d'adaptation auto-organisés dans lesquelles le rôle de l'expérimentateur (Concepteur) est réduit au minimum et pour lesquelles les agents sont libres à développer leur stratégie pour résoudre leurs problèmes d'adaptation dans un grand nombre de solutions potentiellement alternatives.

Les approches actuelles, à cet égard, peuvent être regroupées en deux familles, qui seront illustrées dans les paragraphes suivants et qui comprennent des méthodes de la robotique évolutionnaire et des méthodes de la robotique développementale.

Le processus adaptatif qui réglemente la façon de changement des caractéristiques des agents pourrait consister en une population basée sur un processus évolutionnaire et/ou en un processus de développement/d'apprentissage. Dans le premier cas, les caractéristiques des agents ne varient pas pendant leur "durée de vie" (i.e. pendant le temps où les agents

interagissent avec l'environnement), mais, phylogénétiquement[13], lorsque les agents "se reproduisent". Dans le second cas, les caractéristiques des agents varient ontogénétiquement[14], pendant leur interaction avec l'environnement.

3.5.1. Méthodes de la robotique évolutionnaire

Les méthodes de la robotique évolutionnaire [Nolfi and Floreano, 2000; Floreano, Husbands et al., 2008] sont des méthodes qui permettent de créer des agents incarnés et situés capables de s'adapter à leur tâches et leur environnement d'une façon autonome à travers un processus adaptatif inspiré par évolution naturelle [Holland, 1975] et, récemment, par la combinaison de processus évolutionnaires, développementaux, et d'apprentissage.

Comme indiqué précédemment, les méthodes évolutionnaires [Rechenberg, 1973; Holland, 1975; Golberg, 1989] consistent en une application de la théorie de l'évolution en exploitant les principes de la sélection naturelle introduite Charles Darwin pour la résolution de problèmes complexes. Contrairement aux méthodes analytiques, qui nécessitent de représenter la démarche permettant de trouver une solution, l'approche évolutionniste se contente de décrire le problème puis de laisser la solution émerger par un processus d'évolution simulé [Panzoli, 2008]. Dans ce cas-là, seule une modélisation générique du contrôleur ainsi qu'une fonction d'évaluation « fonction de fitness » sont nécessaires. La fonction d'évaluation permet de noter les agents par rapport au résultat final qu'on attend d'eux. La pression de l'évolution génétique permettra l'émergence d'un contrôleur parfaitement adapté au problème posé [Cussat-Blanc, 2009]. Dans le cas de la robotique évolutionnaire, il suffit de considérer le comportement désiré comme la définition du problème, et modéliser un contrôleur générique. La pression de l'environnement sur les populations successives des agents, permet de faire émerger un comportement adéquat.

Les méthodes évolutionnaires peuvent être utilisées pour permettre aux agents de développer le comportement demandé à partir de scratch (c.-à-d. à partir des agents qui n'ont pas de comportement) ou de manière incrémentale (c.-à-d. à partir robots pré-évolué qui ont déjà une certaine capacité comportementale qui consiste, par exemple, dans la capacité de résoudre une version simplifiée du problème d'adaptation).

[13] Phylogénèse: Indique les variations des caractéristiques génétiques d'une population d'agents artificiels à travers les générations.

[14] Ontogenèse: Indique les variations qui se produisent dans les caractéristiques phénotypiques d'un agent artificiel (i.e. les caractéristiques du système de contrôle ou du corps de l'agent), lorsqu'il interagit avec l'environnement.

3.5.2. Méthode de la robotique développementale

Les méthodes de la robotique développementale, aussi appelée robotique épigénétique, sont des méthodes pour développer des agents incarnés et situés qui s'adaptent à leur tâche et leur environnement d'une façon autonome grâce à des processus inspirés par les processus de développement et d'apprentissage biologiques.

Bien que les méthodes de la robotique évolutionnaire et développementale partagent les mêmes hypothèses fondamentales, elles présentent également des différences en ce qui concerne la façon dont elles sont réalisées et le type de situations dans lesquelles elles sont généralement appliquées. En ce qui concerne le premier aspect, contrairement aux méthodes de la robotique évolutionnaire qui opèrent sur de «longues» échelles de temps phylogénétiques, les méthodes de développement fonctionnent généralement sur de «courtes» échelles de temps ontogénétiques. S'agissant du second aspect, contrairement aux méthodes évolutionnaires qui sont habituellement utilisées pour faire évoluer les compétences comportementales à partir de scratch, les méthodes du développement sont généralement adoptées pour modéliser le développement des compétences développementales complexes à partir des simples compétences préexistantes qui représentent des pré-requis pour le développement des compétences requises [Nolfi, 2009].

3.6. Évolution des créatures virtuelles articulées

Une des applications les plus intéressantes des algorithmes évolutionnaires est l'évolution créatures virtuelles autonome 2D ou 3D. La recherche dans ce domaine a été lancée par Karl Sims [Sims, 1994b]. Ses créatures en blocs à trois dimensions ont fait évolué avec succès diverses locomotions (nage, saut, course), ressemblant souvent à des comportements rencontrés dans la nature (la figure 3.5 représente un exemple des créatures évoluées par Sims). Sims a aussi simulé la compétition des créatures ayant un but commun (s'emparer d'un cube par exemple) [Sims, 1994a]. La particularité de ce système réside dans le fait que la morphologie et l'architecture de contrôle peuvent évoluer simultanément. Le génotype de la créature est basé les graphes orientés pour représenter la morphologie ainsi que les réseaux de neurones pour l'architecture de contrôle. Les créatures parviennent ainsi à faire évoluer leur morphologie en fonction du problème rencontré et on dénote l'émergence de stratégies et de contre-stratégies dans les expériences mettant en jeu des compétitions.

Le travail de Sims a inspiré plusieurs autres travaux sur les créatures virtuelles tridimensionnelles. Shim et Kim ont réussi à faire évoluer diverses créatures volantes avec des types d'ailes différents [Shim and Kim, 2003]. Miconi, dans l'un des travaux les plus récents, reproduisit avec succès les créatures originales de Sims, avec une course aux armements co-évolutionnaires [Miconi and Channon, 2006]. Chaumont et al. a également repris les créatures de Karl Sims en utilisant la même technique d'évolution de la morphologie et le même type de contrôleur [Chaumont, Egli et al., 2007]. Il a proposé, de plus, d'évaluer les créatures sur leur capacité à se déplacer, d'une part, ou à catapulter un bloc, d'autre part.

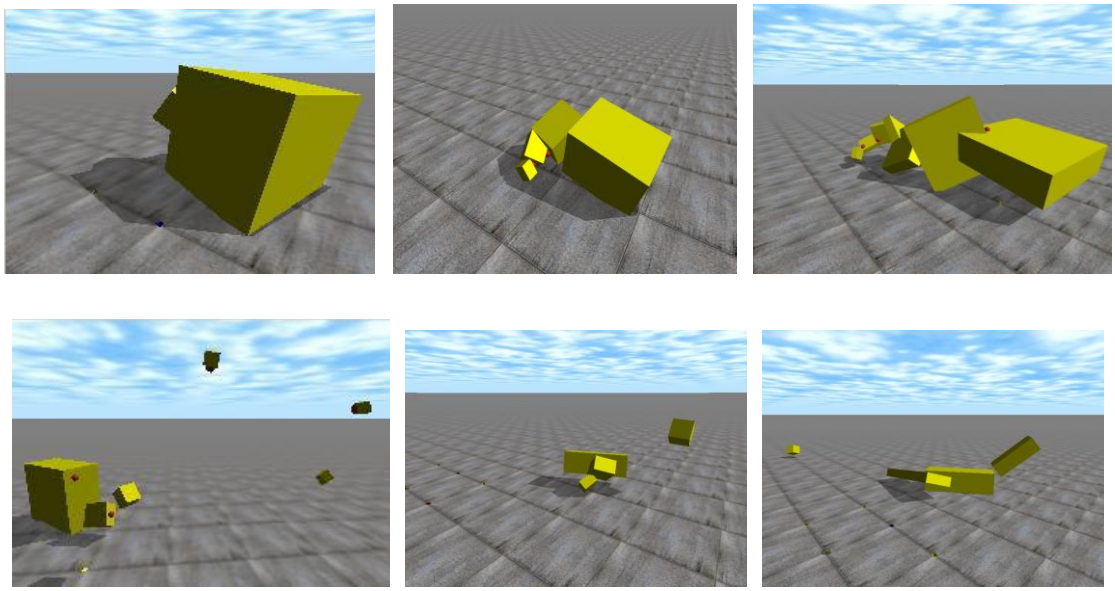


Figure 3. 8- Les créatures de Nicolas Chaumont qui se déplacent et qui catapultent un bloc [Chaumont, Egli et al., 2007]

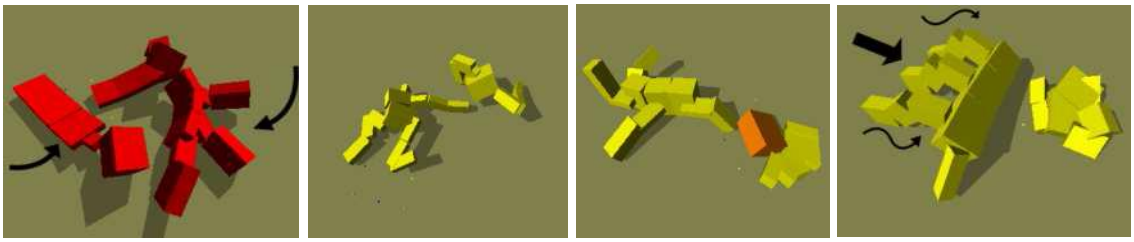


Figure 3. 9- Les créatures de Miconi qui mènent un combat afin de s'accaparer un cube [Miconi and Channon, 2006]

Ray a créé un système pour l'évolution interactive des animaux virtuels tridimensionnels colorés simulés dans un environnement physique [Ray, 2001]. L'objectif du travail de Ray était de faire évoluer des créatures utilisant la sélection émotionnelle et esthétique. Ray a fait

évoluer de manière interactive diverses créatures intéressantes (voir la figure 3.10 pour un exemple d'une créature évoluée).

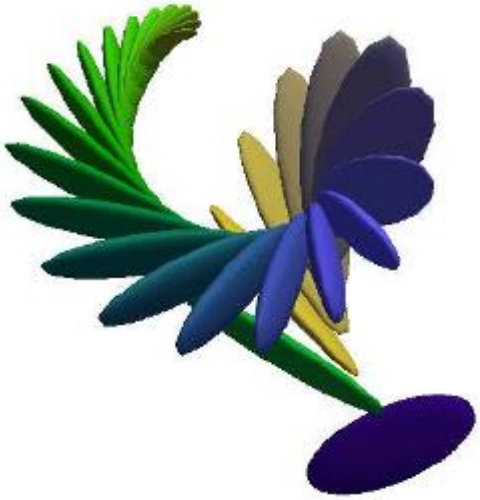


Figure 3. 10- Une créature évoluée par évolution interactive basée sur la sélection esthétique [Ray, 2001]

Spector a récemment introduit un Framework pour étudier l'évolution ouverte[15] dans un écosystème 3D simulé [Spector, Klein et al., 2007]. Au début de l'évolution, les organismes sont de simples blocs (c.-à-d. Blocs de division) contrôlées par un réseau de neurones artificiel. Tous les organismes sont placés dans une arène virtuelle unique, où ils recueillent l'énergie du soleil virtuel en utilisant la photosynthèse et, de ce fait, ils grandissent, rétrécissent, se divisent, forment des jointures et échangent des ressources avec d'autres organismes. Bien que cette étude s'inspire des travaux de Karl Sims [Sims, 1994b], aucun algorithme génétique explicite n'a été utilisé pour faire évoluer les créatures et aucun test explicite de fitness comportementale n'a été fourni. Au lieu de cela, les organismes doivent faire évoluer des paramètres de reproduction et de mutation optimaux afin de survivre dans un écosystème difficile. Les résultats préliminaires montrent que les organismes développent façon répétitive des stratégies de coopération pour des transactions sur les ressources.

Komosinski et Ultowski ont créé le projet «*Framsticks* » pour l'évolution des créatures virtuelles [Komosiński and Ulatowski, 1999; Komosinski, 2000]. Ce modèle «*Framsticks*», est une conjonction entre les modèles avancés de la vie artificielle et des outils de simulation avancés. Les agents peuvent interagir entres eux ainsi qu'avec l'environnement. Ces créatures

[15] Ici on parle de « open-ended evolution »

ont donc une morphologie adaptative codée génétiquement et contrôlée par un réseau de neurones. Le simulateur physique permet entre autre de gérer les frictions (statiques et dynamiques), les forces (action et réaction), les pertes d'énergie après déformation, la gravitation, la pression et la flottabilité. Les créatures sont formées à partir de bâtons (élément de base) ayant des propriétés variables (taille, poids, récepteur, neurones).

Figure 3. 11- exemple des créatures évolué dans le projet « *Framsticks* » [Komosiński and Ulatowski, 1999]

Ces bâtons sont reliés par des jonctions symbolisant les muscles. Les travaux effectués permettent de faire évoluer les créatures en les adaptant à des tâches simples telles que la marche ou la nage.

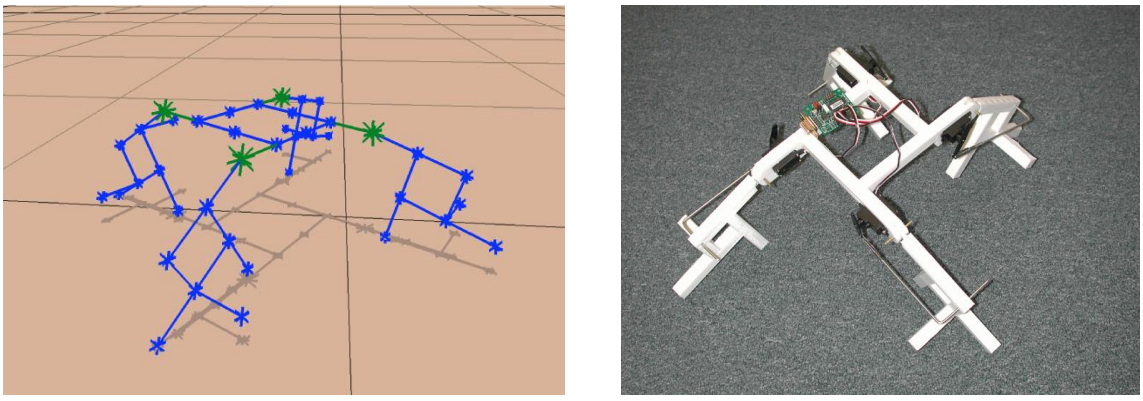
Lipson et Pollack ont fait évoluer des robots simulés capables de se mouvoir dans un environnement 3D simulé. Ils ont également introduit une méthode pour leur fabrication automatique dans le monde réel [Lipson and Pollack, 2000; Lipson and Pollack, 2006] (voir figure 3.12 pour un exemple d'une créature évoluée). Pour ce faire, ils ont utilisé un encodage direct pour représenter leurs créatures. Par ailleurs, un algorithme génétique simple a été utilisé avec une mutation et sans reproduction « croisement ».

(a) une créature dans un environnement simulé

(b) une créature physique construit automatiquement

Figure 3.12- exemple d'une créature évolué dans le projet « *Golem* » [Lipson and Pollack, 2000; Lipson and Pollack, 2006]

Hornby et Pollack se sont basés sur les travaux de Lipson et Pollack et ont conçu un encodage développemental basé sur les L-systèmes pour la morphologie et le système contrôle des créatures. Cet encodage développemental a été comparé à un encodage non-développemental basé sur une séquence de commandes de construction explicites[Hornby and Pollack, 2001a; Hornby and Pollack, 2001b]. Les deux différentes formes de codages ont été comparées sur une tâche de locomotion terrestre dans un environnement 3D.



(a) La créature dans un monde simulé

(b) le robot issu de la créature artificielle

Figure 3. 13- Les robots de Hornby et Pollack, générés par L-Systèmes[Hornby and Pollack, 2001b].

Les résultats ont montré que l'encodage développemental est plus efficace que l'encodage non-développemental et a permis de faire évoluer plus rapidement des créatures ayant des

comportements plus efficaces. Hornby et Pollack ont également construit des créatures dans le monde réel, en profitant de la nature modulaire de l'encodage L-système [Hornby, Lipson et al., 2001].

Bongard et al [Bongard, Zykov et al., 2006] ont récemment proposé une méthode de récupération automatique des dommages inattendus pour des robots physiques à pattes, à travers l'auto-modélisation adaptative. Pendant les expériences, le robot utilise ses capteurs et ses actuateurs pour déduire les dommages potentiels de sa morphologie et utilise ensuite cette auto-modélisation pour se déplacer vers l'avant. L'auto-modélisation « Self-model » est continuellement mise à jour, et agitr lorsque le robot est soudainement endommagé. Elle s'adapte et le robot peut l'utiliser pour reprendre le mouvement vers l'avant. Cette technique dite « auto-modélisation » a expérimentalement prouvé son efficacité à améliorer les capacités de locomotion d'un robot endommagé.

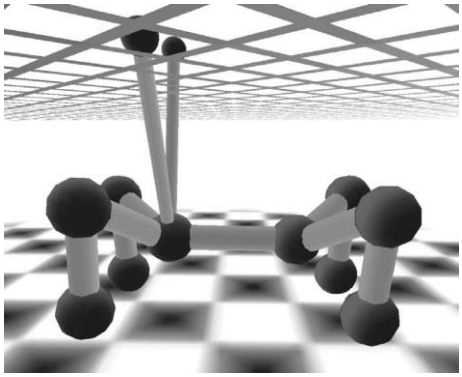


Figure 3.14- l'agent de Bongard [Bongard, Zykov et al., 2006]

Lassabe et Duthen [Lassabe, Luga et al., 2007] ont présenté des améliorations au niveau de l'évolution de la morphologie et du comportement de créatures virtuelles articulées plongées dans des environnements simples ou complexes. Le but est de savoir comment l'évolution des créatures réagit face aux situations différentes de rampement et de marche à des activités plus complexes tels que monter un escalier ou faire de la planche à roulette. Ces créatures utilisent des blocs solides tridimensionnels et les *graphtals* comme les créatures de Sims [Sims, 1994b] et un nouveau type de contrôleur inspiré des systèmes de classifieurs.
En ce qui concerne le choix de son environnement, et pour faire évoluer ses créatures, ils se sont orientés vers un environnement tridimensionnel simulant les lois de la physique. Son choix s'est donc porté sur *Breve* [Klein, 2003].

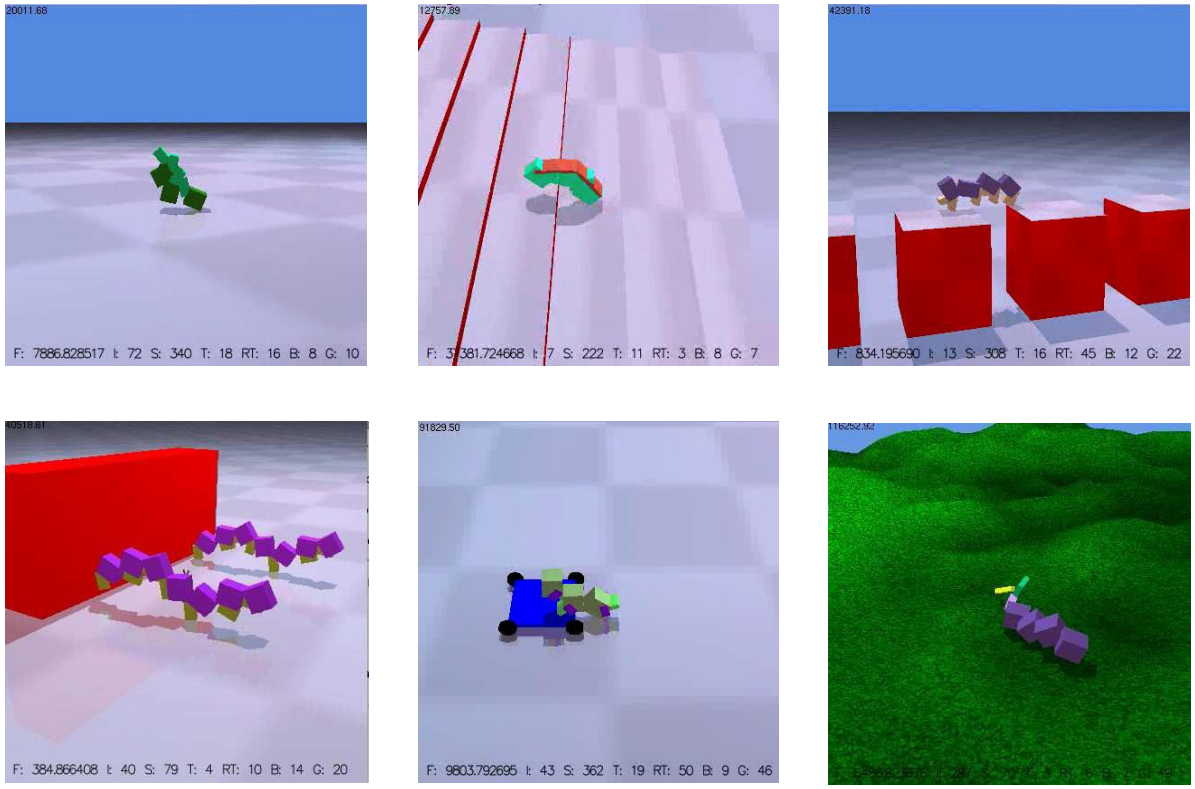


Figure 3.15- Exemples de créatures de Nicolas Lassabe

En résumé, plusieurs travaux ont permis la reproduction avec succès de l'évolution de créatures virtuelles intéressantes et plusieurs tentatives ont réussi de passer de la créature artificielle simulée au des robots physique réel évoluant dans le monde réel.

3.7. Conclusion

Les différents modèles qui ont été produits pour générer des créatures artificielles ont montré la puissance des systèmes de la vie artificielle et leur capacité à fournir des solutions face aux problèmes de différentes complexités en offrant de nombreuses façons de générer des créatures de tailles, de formes variées et qui sont capables de produire des comportements crédibles dans des environnements virtuels comportant, non seulement des interactions géométriques, mais également des interactions physiques entre les objets de l'environnement.

4.Modèle proposé

4.1. Introduction

Les méthodes de calcul évolutionnaire sont empruntées des systèmes de la vie naturelle. En raison de notre désir de faire évoluer des comportements intelligents pour des créatures articulées dans un environnement physiquement simulé, nous employons plusieurs différents algorithmes évolutionnaires. Comme nous l'avons mentionné précédemment, le but du présent travail est de développer un Framework pour l'évolution des comportements réalistes et optimaux associés à des créatures virtuelles qui peuvent être facilement transférées en des robots, dans le monde réel. Ce système est constitué d'algorithmes évolutionnaires utilisés dans l'évolution du cerveau de chaque créature, de l'environnement virtuel qui est peuplé par les créatures, et de la morphologie des créatures. Dans ce chapitre, nous discutons les détails de l'environnement, les morphologies des agents, ainsi que de l'union entre la méthode évolutionnaire, utilisée pour concevoir le cerveau, et la morphologie des créatures.

4.2. L'environnement virtuel et le moteur physique

Les environnements virtuels jouent un rôle très important dans l'évolution des créatures artificielles. Ces environnements sont en tridimensionnels et doivent modéliser avec précision les lois de la physique qui existent dans la nature afin de s'assurer que les créatures existantes dans l'environnement virtuel soient incapables de générer des mouvements ou des forces qui sont impossibles à reproduire dans le monde réel. Pour assurer ce réalisme, l'environnement virtuel utilise un moteur physique sophistiqué qui lui permet de simuler minutieusement la dynamique des corps rigides, des jointures, des collisions, de la friction, de l'inertie et de la gravité en simulant la physique du monde naturel.

De nombreux moteurs physiques tels que Breve [Klein, 2003], ODE [Smith, 2005], Physx et bien d'autres sont à l'usage public. Notre choix s'est porté sur le moteur physique ODE (Open Dynamic Engine)[Smith, 2005]. Ce dernier étant une librairie open source multi plateforme C/C++, permettant de simuler la dynamique des corps rigides (détection de collisions, frottements, …). Les créatures présentes dans cet environnement sont composées d'une série de corps rigides reliés entre eux par des jointures. Le moteur physique agit directement sur les

corps rigides des créatures et ainsi, les créatures elles-mêmes sont soumises à des contraintes imposées par le moteur physique.

Les créatures virtuelles qui peuplent cet environnement sont composées d'une série de corps rigides interconnectés par des jointures. Le moteur physique agit directement sur les corps rigides des créatures; par conséquent, les créatures sont même soumises aux lois de la physique qui régissent l'environnement. Chaque corps rigide peut être contraint par l'utilisation d'une jointure. Cette dernière peut connecter deux ou plusieurs corps rigides. Elle peut être permanente ou temporaire (qui résulte à la collision de deux corps rigides). Lorsqu'un des membres de la créature entre en collision avec la surface de la terre ou un autre objet de l'environnement, une jointure de contact temporaire va être créée.

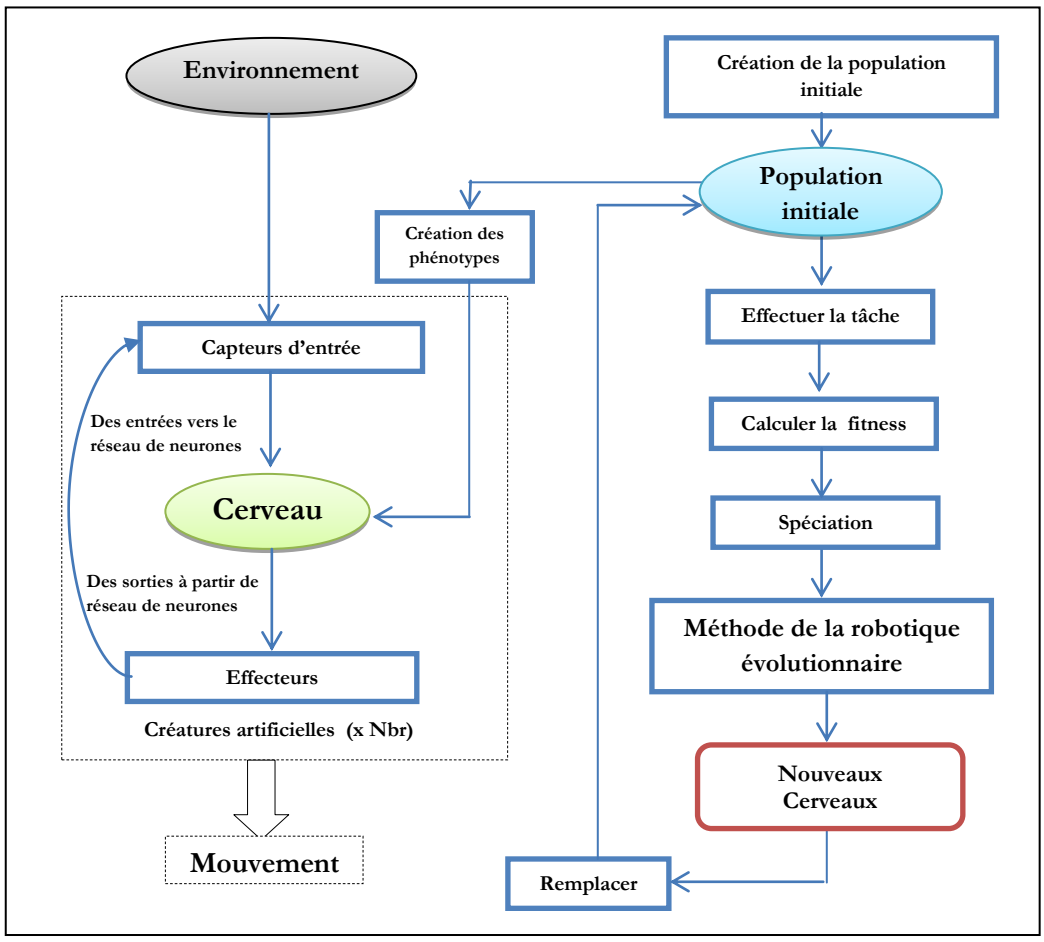


Figure 4. 1- Organigramme de déroulement du système

La force gravitationnelle du monde virtuel est fixée à environ 9,8 m/s^2 pour fournir une bonne approximation des forces gravitationnelles du monde réel au niveau de la mer. Le frottement avec le sol est nécessaire pour la locomotion et il est modélisé au sein d'une

jointure de contact. Le frottement au sein des jointures de la créature est également modélisé par l'utilisation d'un petit couple négatif dans la jointure elle-même. D'autres forces telles que la résistance du vent sont tellement exiguës, ainsi, nous choisissons de ne pas les simuler afin de réduire la complexité de calcul du système.

L'environnement virtuel dans lequel les créatures vivent est complètement stérile. C'est un terrain plat horizontal infini avec un coefficient de frottement similaire à celui de l'asphalte. Ce terrain sur lequel les créatures virtuelles peuvent utiliser leurs membres pour générer une force.

Le déroulement de notre système est illustré par la figure 4.1 qui donne un aperçu sur la manière de faire évoluer des créatures artificielles dans l'environnement virtuel dont le but est de permettre à des stratégies de locomotion à développer.
Le système va suivre un certain nombre des cycles, qui correspondent à chaque fois au calcul de la génération suivante des individus :

1. Tout d'abord, le système crée une population initiale des génotypes, dont le nombre équivalant au nombre des créatures nécessaires pour la simulation (Nb = 80) qui servira de base de départ. Le système crée ensuite le phénotype correspondant à chaque génotype de la population initiale et on les affecte aux créatures en tant que leurs cerveaux.
2. Les créatures effectuent les tâches édictées par une fonction d'évaluation prédéfinie. Dans ce cas, les créatures utilisent une variété de capteurs d'entrée pour recueillir des données sur leur état interne et sur leur environnement. Ces données sont traitées par le système comportemental pour générer des sorties. Ces derniers sont envoyés aux effecteurs des créatures qui leur permettent de déplacer physiquement leurs articulations, leurs membres et générer ainsi le mouvement.
3. Le système calcule la fonction de fitness en basant sur la façon dont les agents ont été en mesure d'effectuer leurs tâches à l'étape 2.
4. Le système fait évoluer automatiquement les contrôleurs des agents par un processus évolutif basé sur les propriétés physiques du modèle et d'une fonction de fitness statique, en générant de nouveaux individus qui possèdent des propriétés héritées de ceux dont ils sont issus.
5. Enfin, le système remplace la génération courante par ses descendants.

Ce cycle est répété jusqu'à l'atteinte du critère d'arrêt satisfaisant nos besoins. La relation symbiotique entre la créature et le système évolutionnaire peut être clairement vue dans l'organigramme dont lequel la méthode de la robotique évolutionnaire a pour tâche de concevoir le système de contrôle.

4.2.1. Les corps rigides « blocs rigides »

Le corps rigide possède diverses propriétés. Certaines propriétés changent au cours du temps:

1. Le vecteur de position (x, y, z) du point de référence d'un corps rigide. Dans un premier temps, le point de référence doit correspondre au centre du corps de masse.
2. Le vecteur de vitesse linéaire (v_x, v_y, v_z) du point de référence du corps rigide.
3. L'orientation du corps, représentée par un quaternion (q_s, q_x, q_y, q_z) ou bien par une matrice de rotation 3x3.
4. Le vecteur de vitesse angulaire (w_x, w_y, w_z) qui décrit comment l'orientation change au cours du temps.

Les autres propriétés sont généralement constantes au cours du temps:

1. La masse du corps.
2. La position du centre de masse par rapport au point de référence. Dans l'implémentation actuelle, le centre de masse et le point de référence doivent coïncider.
3. La matrice d'inertie. Il s'agit d'une matrice 3x3 qui décrit la façon dont la masse du corps est répartie autour du centre de masse.

Ces propriétés sont utilisées au sein du moteur physique pour calculer les forces et les couples qui affectent les corps rigides [Smith, 2005].

Conceptuellement, chaque corps possède un système de coordonnées (x,y,z) qui lui est intégré. Il bouge et tourne avec l'origine de ce système de cordonnées ainsi qu'avec le point de référence du corps comme il est indiqué dans la figure 4.2.

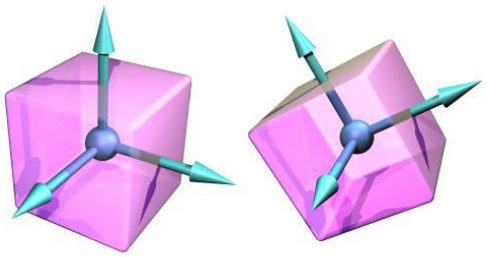


Figure 4. 2- représentation des corps rigides [Smith, 2005]

4.2.2. Les jointures

Il s'agit d'une relation entre deux corps rigides qui doivent la respecter, de sorte que ces deux corps ne puissent avoir que certaines positions et orientations l'un par rapport à l'autre. Ces jointures peuvent être permanentes: les jointures Hinge, Slider et Ball/socket, ou bien temporaires: la jointure de contact qui permet de reproduire la collision de deux objets dans l'environnement virtuel.

a. La jointure « Hinge »

La jointure « Hinge », comme le montre la figure 4.3, limite le mouvement des deux blocs rigides fixés en rotation autour d'un axe. Cette jointure est la plus simple avec seulement deux capteurs qui peuvent fournir, l'angle courant de la jointure et le taux de l'angle courant de la jointure par rapport à la créature.

Soit h (t) l'angle courant entre deux blocs à un instant t. Le taux d'angle de la jointure r(t) est défini comme la dérivée temporelle de l'angle de la jointure:

$$r(t) = \frac{d}{dt} h(t) \tag{4.1}$$

La valeur retournée pour l'angle de la jointure sera comprise entre -pi et pi. Les informations reçues des capteurs peuvent être utilisées pour calculer l'angle courant et la vitesse angulaire et les fournir au système de contrôle de la créature. La jointure dispose également d'un effecteur qui accepte la vitesse désirée comme entrée.

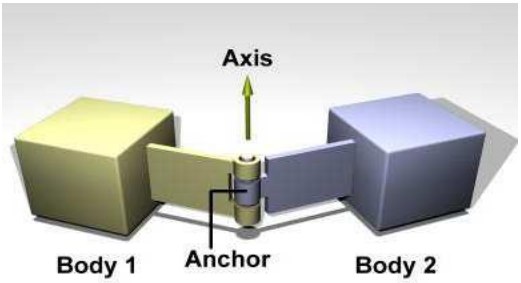


Figure 4. 3- jointure « Hinge » [Smith, 2005].

b. La jointure « Slider »

La jointure « Slider », comme le montre la figure 4.4, contraint le mouvement des blocs interconnectés le long d'un axe prédéfini. Cette jointure est également une jointure simple qui fournit seulement un seul degré de liberté. Les deux capteurs de la jointure fournissent les informations suivantes: la position courante de la jointure et le taux de

changement de position. Soit s(t) la position courante entre les deux corps à l'instant t, le taux de position de la jointure p(t) est défini comme la dérivée temporelle de la position de la jointure:

$$p(t) = \frac{d}{dt} s(t) \tag{4.2}$$

La position courante est retournée comme valeur entre -1 et 1. La jointure « Slider » possède 2 capteurs fournissant des informations sur la position linéaire et la vitesse linéaire au système de contrôle de la créature virtuelle. Cette jointure possède un seul effecteur qui accepte la vitesse désirée comme une entrée.

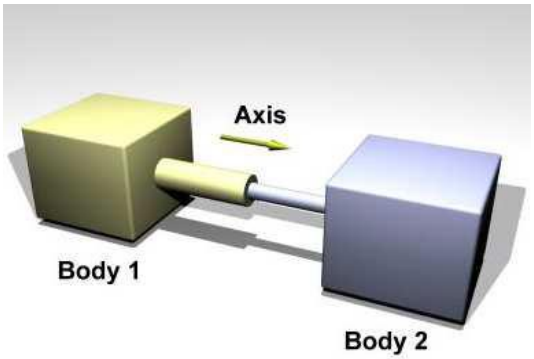


Figure 4. 4- jointure « Slider» [Smith, 2005]

c. La jointure « Ball/Socket »

La jointure « Ball/Socket », illustrée dans la figure 4.5, est actuellement la jointure la plus complexe utilisée dans la conception de la morphologie des créatures articulées. Cette jointure possède trois degrés de liberté sur les trois axes. Elle dispose de deux capteurs pour chaque axe qui fournissent les informations suivantes: l'angle et le taux de l'angle de la jointure par rapport à l'axe donné.

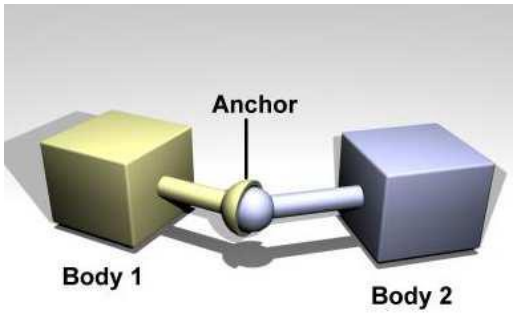


Figure 4. 5- jointure « Ball/socket» [Smith, 2005]

Le taux de l'angle pour un axe est calculé en prenant la dérivée par rapport au temps de l'angle courant sur l'axe particulier. En raison des trois degrés de liberté, les deux capteurs peuvent fournir jusqu'à six éléments d'information au système de contrôle.

d. Jointure de contacte

La jointure de contact prévient l'interpénétration de deux objets (objet 1 et objet 2) au point de contact. Elle le fait en autorisant les corps d'avoir une force "sortante" dans le sens du contact normal, comme illustré dans la figure 4.6. Les jointures de « contact » sont généralement créées et supprimées en réponse à la détection de collisions. Elles simulent le frottement lors du contact par l'application de forces dans les deux directions de frottement qui sont perpendiculaires au vecteur normal.

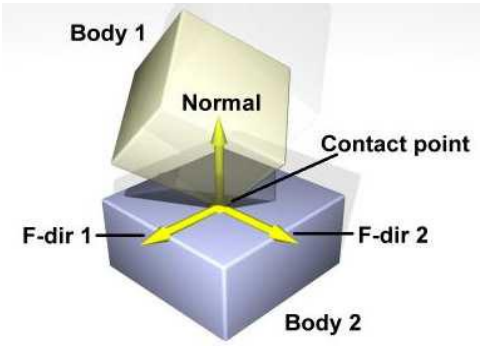


Figure 4. 6- jointure de contact [Smith, 2005]

4.2.3. Les Entrées sensorielles

Les créatures artificielles utilisent une variété des capteurs leur permettent de continuellement recueillir les données provenant de leur environnement, acquérir les informations nécessaires sur leur état interne et les fournir au système comportemental. Le tableau 4.1 montre les capteurs utilisés dans la conception et la simulation des agents, ainsi que la représentation des données sensorielles à l'intérieur du système comportemental. Le tableau montre également la plage des valeurs que les capteurs sont capables de détecter. Les capteurs mettent à jour leurs valeurs de sortie à chaque pas de temps de simulation et ils sont plutôt fiables car ne générant pas de données fausses ou erronées.

Les quatre des huit capteurs effectuent une mise à l'échelle linéaire de leurs sorties avant de les présenter au système comportemental de la créature. Les huit capteurs produisent une plage de valeurs en fonction de leurs entrées à l'exception du capteur tactile, qui peut

reproduire deux valeurs discrètes. Le capteur tactile est modélisé après un interrupteur à bascule momentanée, avec deux positions: ON et OFF.

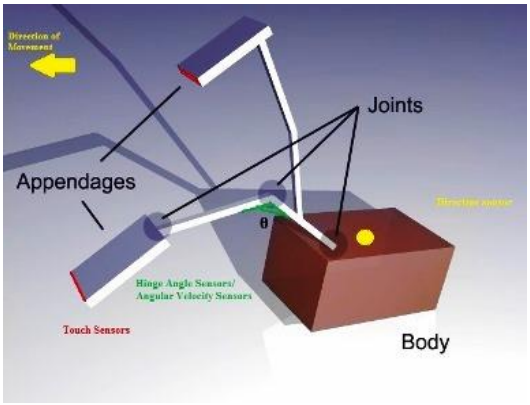


Figure 4. 7- Arrangement des capteurs

L'arrangement des capteurs est représenté par la figure 4.6. Sur cette figure, la créature artificielle a neuf capteurs diffus sur tout son corps. Les autres morphologies ont un agencement de capteurs similaire, avec une seule différence sur le type ou le nombre de capteurs.

Tableau 4. 1- Représentation les données des capteurs

Type de capteur	Plage de sortie	Représentation dans le SC
Hinge	[-pi,pi]	[-3.1416, 3.1416]
Slider	[-1,1]	[-1,1]
Ball/Socket	[-pi,pi]	[-3.1416, 3.1416]
Vitesse Linéaire	[-300,300]	[-1,1]
Vitesse Angulaire	[-100, 100]	[-1,1]
Hauteur	[0, 1000]	[0,1]
Direction	[0,2pi]	[-1,1]
Touche	ON / OFF	-1,1

4.2.4. Les Sorties effectrices et la dynamique angulaires

Chaque jointure possède un moteur connu comme un effecteur. L'effecteur applique un couple sur chaque degré de liberté des jointures pour les faire pivoter ou glisser à la vitesse désirée. La quantité maximale du couple qui peut être générée par les effecteurs est limitée, ils sont incapables d'appliquer plus d'une valeur maximale donnée sur une jointure. Les effecteurs de l'agent lui permettent de contrôler la vitesse angulaire (ou linéaire) relative de deux corps reliés par une jointure, leurs appendices et produire un mouvement.

Tableau 4. 2- Représentation les données des effecteurs

Effecteur/ Jointure	**Force $_{max}$ / Couple**	**Plage d'entrée**
Hinge	800 gm-cm	[-10,10] rad/sec
Slider	1200 gm-cm	[-100,100] cm/sec
Ball/socket	600 gm-cm	[-10,10] rad/sec

Le système de contrôle utilise les entrées sensorielles fournies par les capteurs pour calculer les sorties. Ces derniers sont connectés directement aux entrées des effecteurs. L'effecteur prend la valeur d'entrée et le convertit en vitesse désirée par la mise à l'échelle linéaire des entrées d'effecteurs à la plage d'entrée, comme indiqué dans le tableau 4.2. La force/couple maximum est prédéterminée pour chaque type d'effecteur et elle est conçue pour être proche de plusieurs types communs de moteurs électriques à courant continu.

En utilisant les équations ci-dessous, le moteur physique peut rapidement déterminer l'accélération subie par les organes (les corps), liés à la jointure, basée sur la vitesse désirée qui est fixée par le système de contrôle.

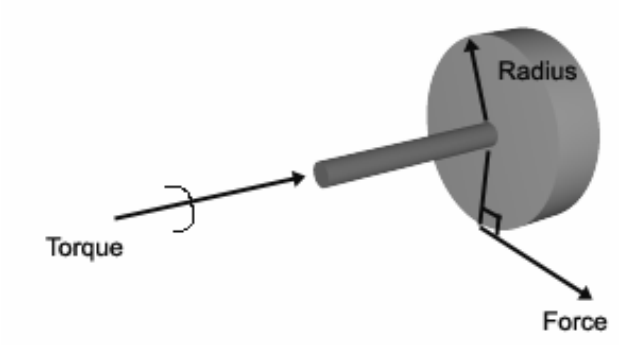


Figure 4. 8- Modèle d'effecteur pour des jointures de type « Hinge » et « Ball/Socket » [Ruebsamen, 2002]

Dans nos expérimentations, les effecteurs utilisent un modèle simple de moteurs de la vie réelle, comme illustré à la figure 4.7. La différence avec le modèle de Sims est qu'un effecteur ne précise pas une force ou un couple, mais plutôt une vitesse désirée. Le moteur tentera constamment d'atteindre la vitesse désirée, avec la contrainte que le couple total exercé ne peut pas être supérieur à un maximum spécifié, ce maximum étant une constante du système. Ce mécanisme correspond à un modèle très simple de servomoteurs. En utilisant ce mécanisme, nous pouvons effectivement réduire le nombre des sorties requises par le système de contrôle à un seul par effecteur: la vitesse désirée.

Les jointures de type Slider sont simplement modélisées en utilisant la deuxième loi de Newton pour calculer la force F:

$$Force : F = m * a \tag{4.3}$$

Où **m** représente la masse du corps rigide et **a** est l'accélération.

Les jointures de type Hinge et Ball/Socket sont modélisées en utilisant la deuxième loi de Newton pour la rotation afin d'atteindre les équations de base de couple en fonction de la force, l'angle et le rayon. Le couple est défini comme suit:

$$Torque : \tau = F * r * \sin\theta \tag{4.4}$$

où **τ** est le couple, **r** représente le rayon, **F** est la force, et **θ** est l'angle entre les deux vecteurs **r** et **F**. L'équation de base du couple est utilisée pour calculer le couple à appliquer, par l'effecteur, sur les jointures mentionnées auparavant.

Une autre force de rotation que le moteur physique doit prendre en considération est le moment d'inertie. Le moment d'inertie d'un corps rigide mesure sa difficulté pour commencer à tourner et cela dépend de la position de l'axe de rotation (soit une jointure ou bien le centre de gravité d'un corps tournant librement) et la masse de l'objet. Le moteur physique calcule le moment d'inertie **I** comme suit : il commence premièrement par la rupture du corps rigide en plusieurs petits morceaux, puis en multipliant la masse de chaque pièce par le carré de la distance de son axe de rotation **r** et en ajoutant tous ces produits jusqu'à:

$$Moment\ d'inertie : I = \sum m * r^2 \tag{4.5}$$

Les corps rigides de cette simulation ont une distribution homogène de la masse sur le centre de masse du corps rigide.

4.3. Les morphologies des créatures

La morphologie des créatures virtuelles est complètement prédéfinie avant l'évolution, elle n'évolue pas au cours de la simulation. Quatre différents types de morphologie sont introduits dans la simulation; chacune est utilisée dans une expérimentation indépendante. Chaque morphologie d'une créature représente une espèce entièrement différente des créatures artificielles qui se différencient par le nombre et le type des jointures, des capteurs d'entrées, des effecteurs et des corps rigides, comme indiqué dans le tableau 4.3.

Des capteurs permettent l'entrée de données provenant de l'environnement et les effecteurs agissent comme des moteurs attachés aux appendices et permettent à la créature d'interagir physiquement avec l'environnement. Chaque agent est composé d'une hiérarchie de parties de corps rigides 3D qui sont reliées par l'intermédiaire des jointures. Comme décrites précédemment, chacune des jointures possède des effecteurs (muscles) entre elles qui exercent une force de traction ou de poussée sur l'un des degrés de liberté (jusqu'à trois degrés de liberté pour une jointure Ball/Socket) et peut modéliser les forces de flexion et d'extension qui sont généralement exercées par des paires de muscles biologiques ou par le couple d'un motoréducteur.

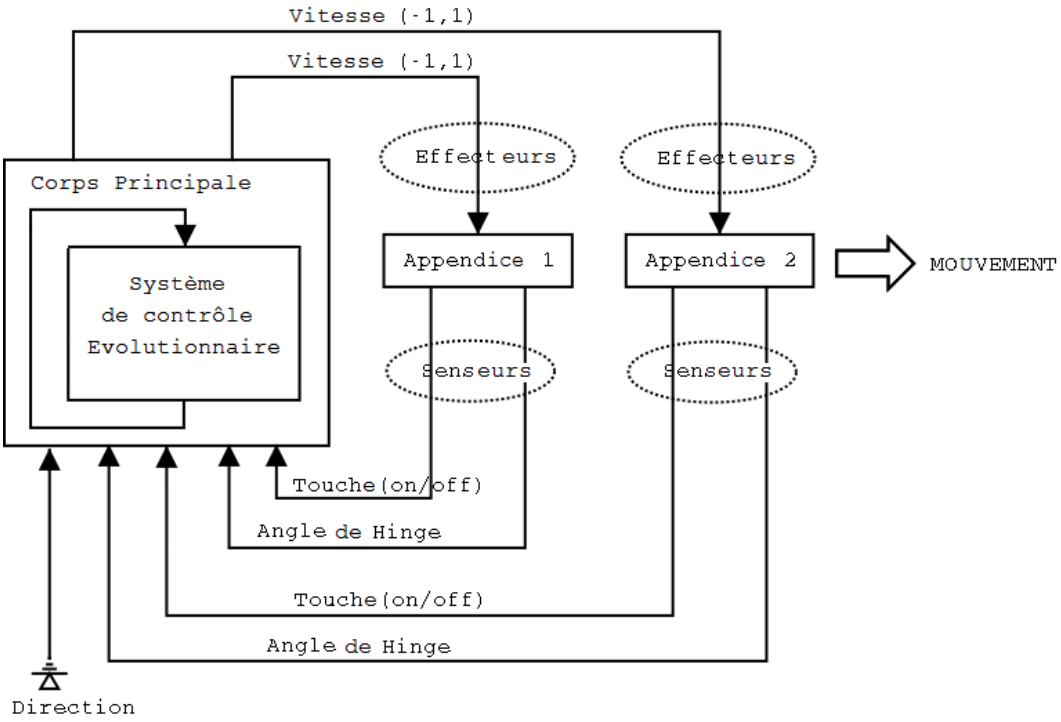


Figure 4. 9- Organigramme de créature artificielle.

L'organigramme de la créature artificielle, illustré par la figure 4.9, décrit la façon dont une créature peut faire évoluer les comportements de locomotion. Bien que l'organigramme décrive la morphologie de la créature de type « Rampant », il reste applicable à toutes les autres morphologies, car la seule différence, du point de vue du système de contrôle, réside dans le nombre d'entrées et de sorties. L'organigramme peut être divisé en quatre étapes:

1- La créature reçoit les informations de son environnement à partir les capteurs attachés à ses appendices ;
2- Le système de contrôle traite les données sensorielles entrantes ;

3- Le système de contrôle envoie les signaux de sortie aux effecteurs ;

4- Les effecteurs contrôlent le mouvement des appendices.

Les quatre étapes sont continuellement répétées pour générer le mouvement.

Tableau 4. 3- la morphologie des créatures.

Créatures	Rampant	Arm-Based	Trémie	Coureur
Nombre de Jointure	2	6	4	2
Corps rigides	3	7	5	3
Type de jointure	Hinge	Hinge	Hinge et Slider	Ball/Socket
Capteurs d'entrée	5	9	9	10
Effecteurs	2	6	4	6

L'objectif d'utiliser une variété de morphologies peut répondre à trois besoins:

1. Développement d'une variété des stratégies de locomotion.
2. Expérimentation des capacités de généralisation de la technique de robotique évolutionnaire pour faire évoluer des solutions de locomotion intelligente lorsqu'elles sont présentées avec une variété de morphologies.
3. Démonstration des capacités du système de modéliser une variété de jointures et de structures existantes dans la nature.

Comme indiqué précédemment, la modélisation précise des machines du monde réel peut théoriquement permettre la transplantation des structures comportementales évoluées (le système de contrôle évolué) à partir des créatures virtuelles de l'environnement simulé aux machines de l'environnement naturel.
La composition des morphologies de l'agent est détaillée dans le tableau 4.3.

4.3.1. La morphologie de la créature de type « Rampant »

La première créature artificielle est composée de trois corps rigides interconnectés comme le montre la figure 4.10. Deux appendices se saillent de l'avant du corps principal "torse" et sont attachés au torse par des jointures de type « Hinge ». Cette créature possède 5 capteurs liés directement au système comportemental : deux capteurs fournissant des informations sur la vitesse angulaire des deux jointures « Hinge », un capteur tactile sur chaque appendice et un capteur de direction agissant comme une boussole. Cette créature est également équipée de deux effecteurs contrôlant la vitesse angulaire des moteurs d'articulation.

Cette morphologie a été conçue de telle sorte que la créature utilise ses appendices pour ramper vers l'avant. Elle a été introduite pour montrer que la technique évolutionnaire utilisée pour l'auto-conception de systèmes de contrôle est une technique efficace pour entrainer des créatures simples qui effectuent des tâches impliquant la manipulation d'un système de contrôle simple.

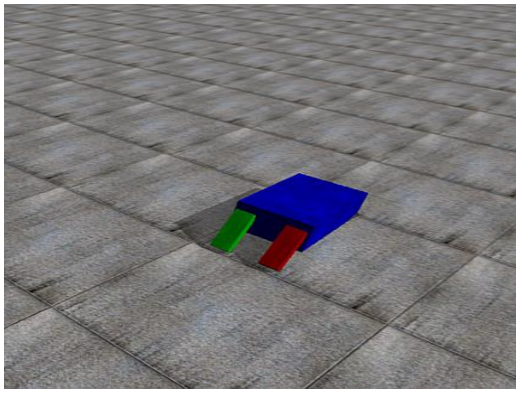


Figure 4. 10- Créature de type **«** Rampant **»**

4.3.2. La morphologie de la créature de type « Arm-Based »

La seconde créature se compose de deux structures complexes " en forme de bras" se saillant du haut de son corps principal, comme le montre la figure 4.11. Chaque bras est composé de trois corps rigides reliés entre eux par des jointures de type « Hinge » et reliés au corps principal par une jointure du même type « Hinge ». Les «mains» sont en forme de pagaies pour fournir une plus grande surface de contact avec le sol. Cette créature possède 9 capteurs liés directement au système comportemental : six capteurs fournissent les informations sur la vitesse angulaire de toutes les six jointures « Hinge », un capteur tactile sur chaque main de créature et un capteur de direction agissant comme une sorte de boussole. Cette créature utilise six effecteurs contrôlant la vitesse angulaire des moteurs d'articulation.

Cette morphologie a été conçue de telle sorte que la créature apprendrait à alterner le mouvement de ses "bras", pour permettre à chacun de tirer le torse vers l'avant et chronométrer son mouvement. Cette morphologie a été introduite pour tester les capacités du système comportemental à contrôler des créatures morphologiquement plus complexes et déterminer s'il est capable d'apprendre à manipuler d'une manière efficace des membres articulés complexes qui sont composés de plusieurs corps rigides et plusieurs jointures.

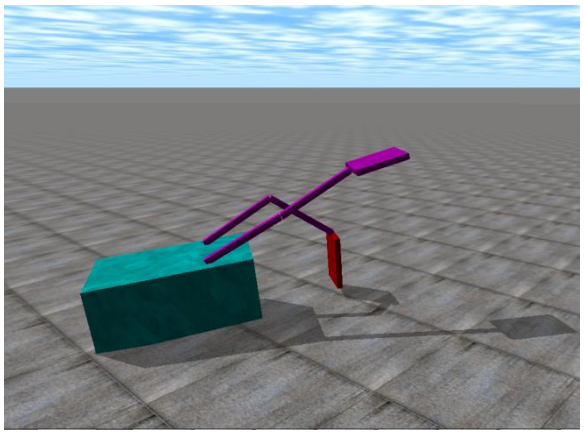


Figure 4. 11- Créature de type « Arm-Based »

4.3.3. La morphologie de la créature de type « Trémie »

La troisième créature est conçue pour obtenir des comportements de culbute qui représente le comportement de locomotion le plus complexe de toutes les espèces. Cette créature requerrait au moins une certaine connaissance du mouvement d'un projectile et être en mesure de déterminer où il va atterrir après chaque mouvement du saut successif. La créature « Trémie » se compose de deux jambes attachées par une jointure Hinge à la face inférieure du corps principal qui est structurellement semblable à un box. Chaque jambe à son tour se compose de deux blocs liés par une jointure « Slider », comme le montre la figure 4.12. La jointure « Slider » offre aux deux jambes la possibilité de sauter rapidement.
Cette morphologie a été introduite afin de déterminer si le système de contrôle peut être capable de générer des comportements complexes et apprendre à manipuler efficacement beaucoup d'informations de son environnement. Cette créature comporte 13 capteurs d'entrées fournissant au système de contrôle les informations suivantes :

- Deux capteurs pour la vitesse angulaire de chaque jointure « Hinge » ;
- Deux capteurs pour l'extension de chaque jointure « Slider » ;
- Un capteur pour la vitesse linéaire du corps principal par rapport à l'environnement ;
- Deux capteurs pour la position angulaire de chaque jointure « Hinge » ;
- Deux capteurs pour la distance de l'extension de chaque jointure ;
- Un capteur pour la position angulaire du corps principal par rapport au terrain plat horizontal ;
- Deux capteurs de touche sur la face inférieure de la deuxième partie de chaque jambe ;
- Un capteur qui fournit la hauteur du corps principal sur le terrain de l'environnement.

Cette créature utilise 4 effecteurs qui lui permettent de contrôler la vitesse d'extension de chaque jambe et la vitesse angulaire des deux jointures Hinge en fonction des sorties fournies par le système de contrôle.

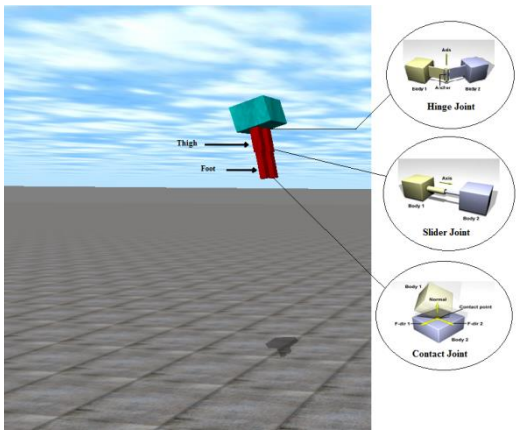


Figure 4. 12- Créature de type « Trémie »

4.3.4. La morphologie de la créature de type « Coureur »

La troisième créature, illustrée dans la figure 4.13, se compose de trois parties: un corps principal formé d'un box et de deux bras dont chacun est attaché à un côté du corps principal à travers une jointure de type Ball/socket qui fournit trois degrés de liberté pour le mouvement et la rotation de chaque bras. À cause de ces deux degrés de liberté supplémentaires, le système comportemental est contraint à traiter beaucoup plus d'informations que la créature précédente utilisant la jointure Hinge avec un seul degré de liberté. Cette créature possède 10 capteurs d'entrées fournissant les informations nécessaires au système de contrôle: 6 capteurs pour déterminer la vitesse angulaire de chaque degré de liberté pour chaque bras, un capteur de contact à l'extrémité de chaque bras, un capteur de contact sur la face inférieure du corps principal et un capteur de direction. La créature utilise 6 effecteurs contrôlant la vitesse angulaire des moteurs d'articulation. Cette morphologie a été introduite pour démontrer l'efficacité du système de contrôle à faire évoluer des comportements efficaces pour des créatures employant des articulations complexes.

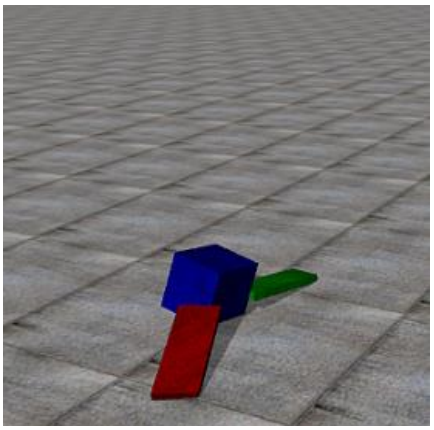


Figure 4. 13- Créature de type « Coureur ».

4.4. La fonction d'évaluation

Le réseau de neurones NEAT utilise une fonction de fitness prédéfinie afin d'évaluer la performance de chaque individu de la population. La fonction de fitness doit être choisie de telle sorte qu'elle récompense les comportements efficaces et pénalise ceux considérés comme inutiles. Elle se base sur la distance parcourue par la créature virtuelle durant une période de temps prédéterminée. Les meilleures valeurs des fonctions de fitness sont attribuées aux créatures parcourant de grandes distances durant le temps alloué (7.5 secondes).

Les créatures qui se déplacent hors de leur trajectoire sont pénalisées par une réduction de leur fonction de fitness. Par contre, celles qui sont capables de générer correctement le comportement désiré voient les valeurs de leurs fonctions de fitness croitre vers des valeurs plus élevées.

Mathématiquement, pour déterminer la fitness d'une créature Cr_i, il faut d'abord calculer la distance qu'elle a parcouru $dis(Cr_i)$ comme suit :

$$dis(Cr_i) = a(-y - y_0) - (|\, x - x_0 \,|^{\beta}) \tag{4.6}$$

Tel que x_0 et y_0 sont la position initiale de départ de la créature virtuelle sur les axes X, Y respectivement, α est le facteur de multiplication de la distance parcourue et β est le facteur de pénalité. Ainsi, la fonction de fitness de chaque créature est définie comme suit :

$$Fitness = \begin{cases} F(x) & si \quad F(x) > 0 \\ 0 & si \quad F(x) = 0 \end{cases} \tag{4.7}$$

La distance parcourue le long de l'axe X (représente une créature qui se déplace hors de sa trajectoire) est déterminée par (x - x_0). Tandis que la distance parcourue le long de l'axe y est représentée par (y - y_0).

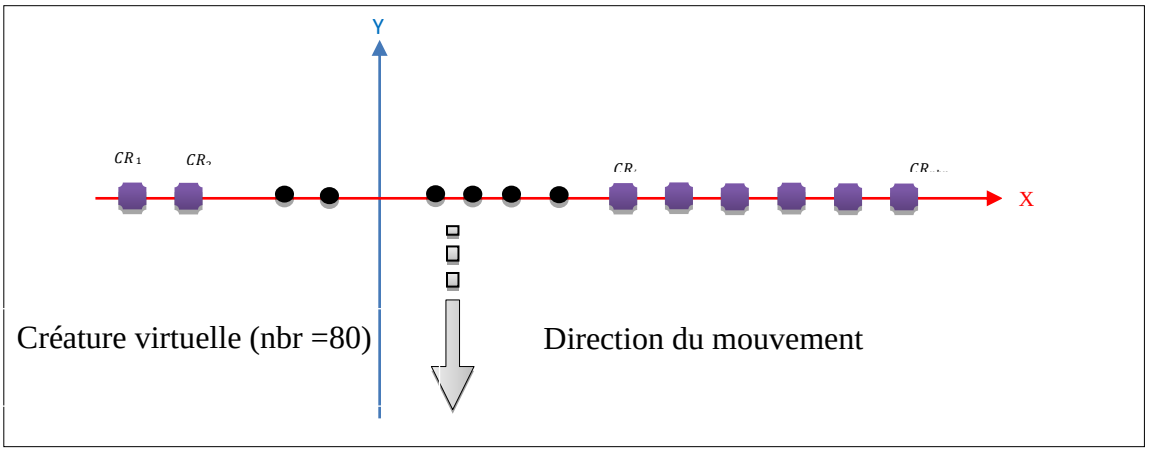


Figure 4. 14- Départ de chaque génération

La fonction de fitness renforce positivement les créatures qui sont capables de parcourir de longues distances le long de l'axe Y en conservant leurs trajectoires.

4.5. Conclusion

Ce chapitre propose une méthodologie pour l'évolution des comportements des créatures artificielles dans un environnement physiquement réaliste comme une étape importante vers l'élaboration d'une technique évolutionnaire pour l'émergence de stratégies de locomotion qui utilisent des techniques inspirées par les systèmes évolutionnaires biologiques. Les différentes morphologies réalisées dans ce chapitre fournissent une bonne base d'expérimentation qui sert à mesurer la performance et tester les capacités des méthodes de la robotique évolutionnaires à proposer dans les prochaines sections.

5.NeuroEvolution pour le contrôle: Application à la simulation de créatures artificielles

5.1. Introduction

Dans ce chapitre, de nouveaux résultats dans le domaine de l'évolution du réseau de neurones sont combinés avec des travaux sur la conception d'un contrôleur pour des créatures virtuelles. L'objectif de ce chapitre est d'élaborer une nouvelle méthode d'évolution des contrôleurs de locomotion pour une variété d'organismes anthropomorphes, décrits dans le chapitre précédent en utilisant l'algorithme NEAT. Ce chapitre nous permet également d'appliquer les résultats récents du domaine de l'intelligence computationnelle pour simuler et contrôler des créatures virtuelles, de montrer que les avantages de la méthode NEAT démontrés dans d'autres domaines peuvent également être exploités avec succès pour concevoir automatiquement différents systèmes de contrôle de créatures et d'élaborer une plateforme pour faire évoluer des stratégies de locomotion réalistes et optimales pour des créatures virtuelles qui peuvent être facilement transférées et portées vers des robots réels. Ainsi, dans ce chapitre, nous traitons brièvement, dans un premier temps, les approches classiques utilisées pour faire évoluer un réseau de neurones artificiel, puis nous décrivons la plateforme que nous avons mise en œuvre et les résultats nous avons pu obtenir.

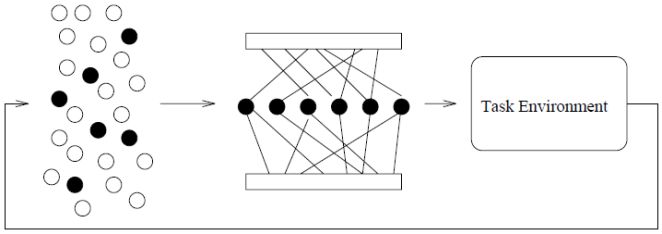
5.2. NeuroEvolution (NE)

L'évolution des réseaux de neurones est devenue une technique d'apprentissage par renforcement dotée d'une efficacité avérée. Un problème de repère, le balancement de deux pôles[16], est souvent utilisé pour comparer les différentes méthodes d'apprentissage par renforcement. Dans ce problème, un réseau de neurones contrôle le mouvement d'un chariot avec deux poteaux verticaux qui s'y rattachent. Le but est de garder les deux pôles en position verticale. Le problème est adapté à l'analyse comparative, parce que sa difficulté évolue

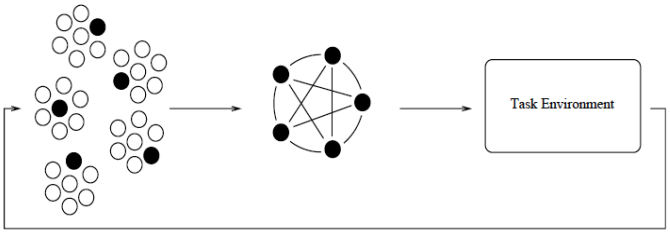
[16] On parle de « double pole-balancing problem »

facilement en modifiant les longueurs relatives aux deux pôles (ce problème devient plus difficile lorsque la différence de longueur des deux pôles devient plus petite). Ce Benchmark a été introduit en 1991 par Wieland [Wieland, 1991] pour tester l'une des premières méthodes d'évolution des réseaux de neurones. Plus tard, il a été adopté comme benchmark standard pour les algorithmes d'apprentissage par renforcement.

Plusieurs approches ont étudié l'évolution des poids des connexions d'une topologie fixe de réseaux de neurones[Whitley, Dominic et al., 1994; Saravanan and Fogel, 1995]. Wieland a utilisé, à la fois, le croisement et la mutation pour l'évolution des valeurs des poids des réseaux de neurones dont la topologie est fixe [Wieland, 1991]. Son approche est considérée comme une méthode de NE conventionnelle. Un exemple d'une technique plus avancée de l'évolution des réseaux de neurones de topologie fixe est SANE (symbiotique, adaptative, NE) [Moriarty and Mikkulainen, 1996]. SANE tente de faire évoluer les neurones cachés au lieu de tout le réseau. Cette technique réalise l'évolution des réseaux de neurones avec une seule couche cachée. Chaque neurone caché dans la population est décrit par un ensemble de poids de son entrée et de sa sortie.



(a) SANE (symbiotic, adaptive NeuroEvolution)



(b) ESP (Enforced Sub-populations)

Figure 5. 1- Une comparaison des algorithmes SANE et ESP. Les deux algorithmes font évoluer les neurones au lieu du réseau entier. SANE réalise l'évolution de tous les neurones dans une seule population, alors que l'algorithme ESP divise la population en espèces distinctes, et chaque espèce contient des neurones candidats pour un seul nœud caché dans le réseau [Gomez and Miikkulainen, 1999].

Durant l'évaluation de la fonction de fitness, une taille fixe aléatoire d'un ensemble de neurones est choisie parmi l'ensemble de la population pour former une couche cachée dans un réseau de neurones. Ce réseau est ensuite évalué sur un problème donné (c.-à-d. balancement de pôles).

Ce test est effectué à plusieurs reprises pour différents ensembles aléatoires de neurones, jusqu'à ce que chaque neurone ait apparu dans au moins 10 tests. La fonction de fitness d'un neurone représente la fitness moyenne de tous les tests dont lesquels il a participé (voir figure 5.1 (a)). SANE a supplanté d'autres techniques d'apprentissage par renforcement, telles que le Q-learning, dans le problème de balancement de deux pôles [Moriarty and Mikkulainen, 1996], mais il a été dépassé par des méthodes de NE plus récentes.

La première technique de NeuroEvolution qui a fait évoluer la topologie d'un réseau de neurones était la technique Cellular Encoding (CE) [Gruau, Whitley et al., 1996]. Cette dernière a été motivée par la croissance de la cellule biologique, qui se produit par division cellulaire. Elle a utilisé un ensemble de règles de grammaire pour représenter d'une manière compacte la topologie du réseau de neurones. Les règles de grammaire représentent un arbre de grammaire, qui décrit le processus de croissance cellulaire. Chaque cellule contient une tête de lecture pointant vers le nœud dans l'arbre de la grammaire, qui instruit la cellule sur la façon de réaliser la division (plusieurs types de divisions cellulaires sont utilisés). La croissance commence par une seule cellule et après chaque division, la tête de lecture du premier enfant avance à la sous-arborescence gauche et la tête de lecture du second enfant avance à la sous-arborescence droite. Finalement, lorsque le développement de la cellule se termine, la cellule se transforme en un neurone. C'est l'une des premières méthodes qui a introduit une phase de développement dans le processus de construction du phénotype (similaire au développement de l'embryon dans les organismes naturels). C'est aussi la première méthode qui a résolu le problème de balancement de pole.

Un disciple direct de SANE est la technique ESP (Enforced Sub-populations) [Gomez and Miikkulainen, 1999]. Celle-ci aborde le problème difficile de la spécialisation des neurones dans la méthode SANE. Dans cette méthode, Les neurones se concurrencent et couplent avec les neurones de toute la population rendant peu probable, que les différents groupes de neurones se développent pour différentes fonctions. La technique ESP résout ce problème en divisant explicitement la population de neurones en *n* espèces, où *n* est le nombre désiré de neurones dans la couche cachée. Les neurones sont autorisés à se recombiner en ne

concourant uniquement que dans leur propre espèce (voir la figure 5.1 (b)). En plus de permettre aux neurones de se spécialiser au cours du processus d'évolution, la spéciation permet également l'évolution des réseaux récurrents. Ceci est possible dans ESP (par opposition à SANE), parce que la spécialisation des neurones permet à un neurone dans une espèce d'attendre des valeurs d'un autre neurone dans une autre espèce. ESP fait évoluer les réseaux de topologie fixe, cependant, lorsque l'évolution stagne (c.-à-d. la recherche bloque sur un optimum local), ESP redémarre avec un nombre aléatoire de nœuds cachés. ESP a supplanté la technique SANE dans le benchmark de balancement de deux pôles, et a constitué un algorithme de pointe pour la tâche de balancement de pole [Stanley and Miikkulainen, 1996], jusqu'à ce que l'algorithme NEAT (NeuroEvolution d'augmenter topologies) ait été proposé.

5.3. Les réseaux de neurones NEAT

Les réseaux de neurones NEAT sont des algorithmes évolutionnaires puissants capables de trouver des solutions à des problèmes complexes où d'autres en sont incapables. L'algorithme arrive à de tels résultats grâce à plusieurs innovations majeures. Les réseaux de neurones NEAT commencent leurs recherches avec des petits réseaux, ce qui permet d'augmenter la rapidité en début de recherche. L'évolution incrémentale permet aux utilisateurs d'éviter de devoir deviner la topologie à utiliser pour résoudre le problème. De plus, cette façon crée des réseaux beaucoup plus petits (optimaux).

Les réseaux de neurones NEAT essaient de nombreuses façons de résoudre un problème, et ce qui lui permet d'éviter de rester coincé sur un minimum local. Les réseaux de neurone NEAT ont été utilisés avec succès dans de nombreuses applications telles que: le problème de balancement de pole [Stanley and Miikkulainen, 2002a], les systèmes de contrôle et d'avertissement des véhicules [Kohl, Stanley et al., 2006], la course automobile [Cardamone, Loiacono et al., 2009], le contrôle robotique ainsi que les jeux [Stanley and Miikkulainen, 2004; Stanley, Bryant et al., 2005; Wittkamp, Barone et al., 2008]. Les réseaux de neurones NEAT ont démontré leur efficacité par rapport aux autres approches classiques dans le cas le plus difficile du problème de balancement de pole [Stanley and Miikkulainen, 2002b].

Les Réseaux de neurones NEAT utilisent l'encodage direct pour décrire la structure du réseau et les poids des connexions. Ils marquent tous les gènes avec un identifiant unique, permettant de tracer d'où sont issus les gènes, d'éviter le problème de permutation, et donc d'effectuer un croisement plus efficace et plus robuste. Un Réseau de neurones NEAT utilise la spéciation

de ses individus, et protège ainsi l'innovation. Il part d'une population de réseaux simples sans neurones cachés et augmente leur complexité par l'ajout progressif des neurones et de connexions au cours d'évolution [Stanley and Miikkulainen, 2002b].

5.3.1. Le génome NEAT

Les génomes sont des représentations linéaires de la connectivité du réseau. La structure d'un génome NEAT est composée de deux listes : la liste des gènes de nœuds (neurones), et la liste des gènes de connexions dont chacun se réfère à deux gènes de neurones connectés. La première décrivant la fonction du neurone au sein du réseau qu'il soit un neurone d'entrée, un neurone de sortie, un neurone caché, ou un biais qui peuvent être connectés, tandis que chaque gène de neurone possède un identifiant unique. La seconde contient les informations

Génotype (génome)

Liste des gènes de nœud (neurones)

identifiant:1 **Type**: neurone d'entrée	**identifiant**: 2 **Type**: neurone d'entrée	**identifiant**:3 **Type**: neurone d'entrée	**identifiant**:4 **Type**: neurone de sortie	**identifiant**:5 **Type**: neurone caché

Liste des gènes de connexions

de: 1 **à** : 4 **poids** : 1.2 **activé** : oui **récurrent** : non **innovation** : 1	**de**: 2 **à** : 4 **poids** : -3 **activé** : non **récurrent** : non **innovation** : 2	**de**: 3 **à** : 4 **poids** : 0.7 **activé** : oui **récurrent** : non **innovation** : 3	**de**: 2 **à** : 5 **poids** : -2.1 **activé** : oui **récurrent** : non **innovation :** 4	**de**: 5 **à** : 4 **poids** : 1.1 **activé** : oui **récurrent** : non **innovation** : 5	**de**:1 **à** : 5 **poids** : 0.8 **activé** : oui **récurrent** : non **innovation** : 6	**de**:4 **à** : 5 **poids** : -1 **activé** : oui **récurrent** : oui **innovation** : 7

Phénotype (Réseau)

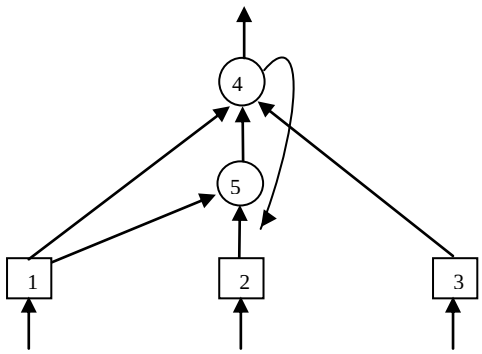


Figure 5. 2- Encodage d'un réseau de neurones artificiel NEAT

sur deux neurones connectés, le poids associé à cette connexion, un indice pour indiquer si le lien est activé, un indice pour indiquer si le lien est récurrent, et un numéro d'innovation inchangeable qui permet de trouver les gènes correspondants. La figure 5.2 illustre les deux listes de gènes d'un génome présentant un réseau simple.

5.3.2. Mutation

La mutation dans les réseaux de Neurones NEAT peut modifier, à la fois, les poids des connexions et la structure du réseau. Les poids des connexions mutent comme dans tout système de NeuroEvolution tandis que les changements structurels sont assurés par deux opérateurs spéciaux de mutation qui augmentent la complexité.

Le premier consiste à ajouter une nouvelle connexion entre deux neurones non connectés précédemment comme illustré dans la figure 5.3. Un seul gène de la connexion associée à un poids aléatoire est ajouté à la fin du génome.

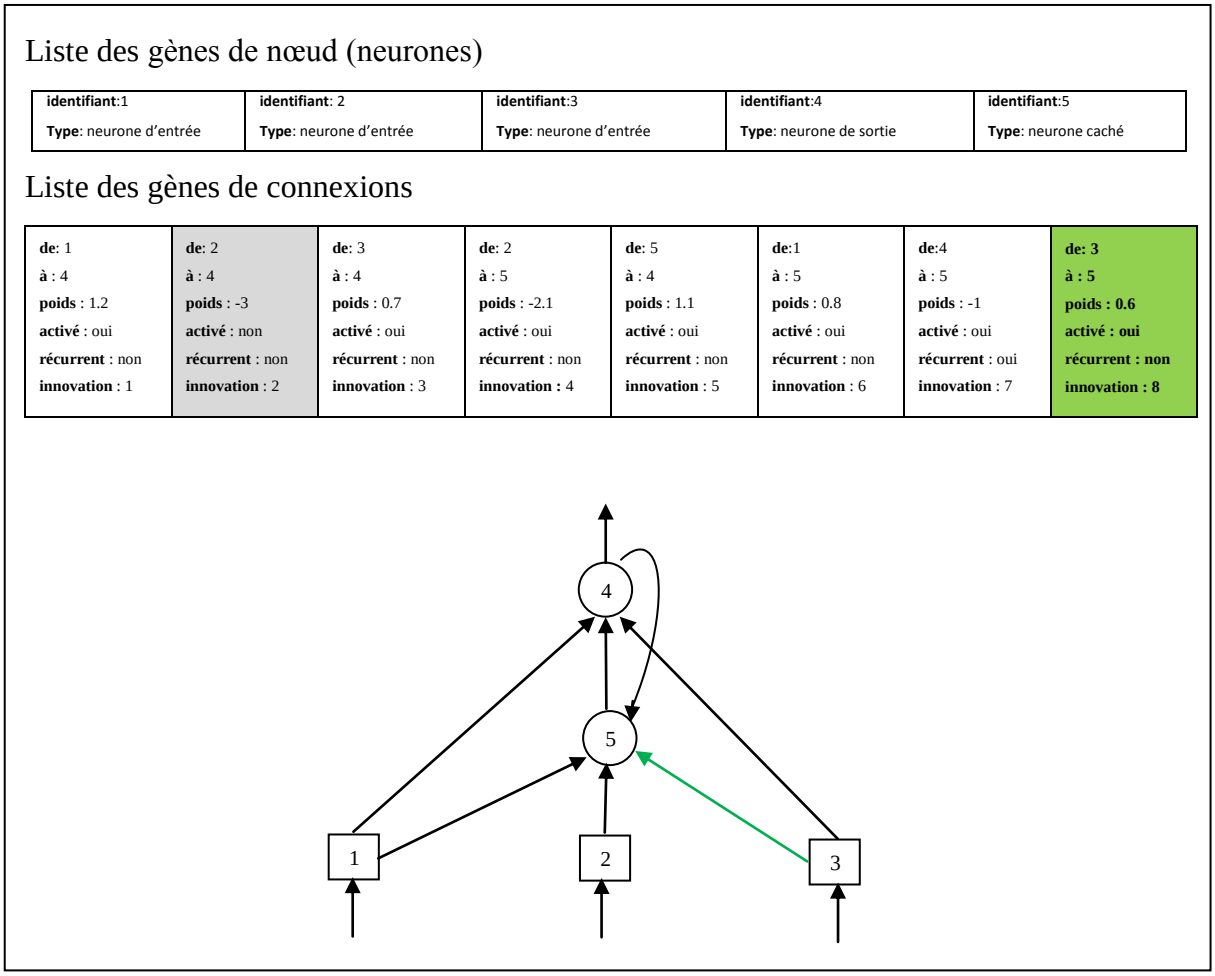


Figure 5. 3- Mutation structurelle dans un réseau NEAT en ajoutant une connexion

On peut distinguer trois types différents des liens à ajouter :

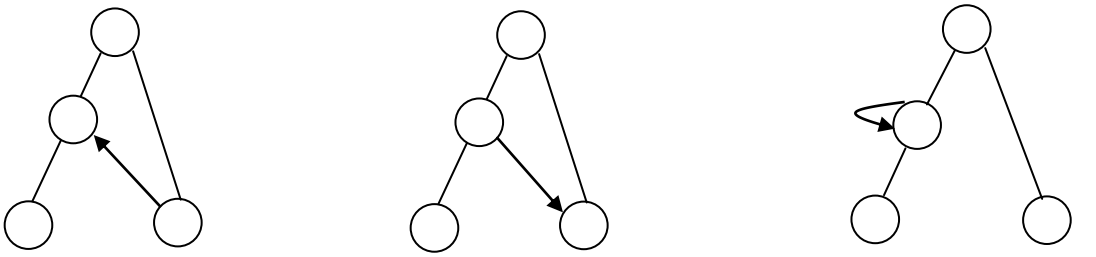


Figure 5. 4- les différents modèles de connexions

Le second est l'ajout d'un nœud, illustré à la figure 5.5. Une connexion doit être choisie, puis divisée en deux nouvelles connexions et le nouveau neurone caché se place entre elles. Le gène de la connexion divisée est désactivé et trois innovations sont créées et ajoutées au génome: une pour le gène du neurone et deux pour les gènes des nouvelles connexions. Le poids de la connexion divisée est utilisé comme le poids de la nouvelle connexion sortant du neurone ajouté, et le poids de la nouvelle connexion entrant au neurone est mis à 1.

Une base de données globale de toutes les innovations est utilisée dans le but de sauvegarder les numéros d'identification de chaque innovation. Ainsi, à chaque fois qu'un nouveau neurone ou lien est ajouté, la base de données est référencée pour voir si l'innovation a été créée précédemment. Si c'est le cas, alors le nouveau gène est attribué au numéro d'identification d'innovation existant. Sinon, une nouvelle innovation est créée et ajoutée à la base de données, et le gène est étiqueté avec le numéro d'identification de l'innovation nouvellement créée.

À la création de la population initiale, l'algorithme utilisé par le réseau de neurones NEAT définit automatiquement les innovations pour tous les neurones et les connexions de départ. Ainsi, la base de données de l'innovation contient, avant la mutation, les numéros d'innovation des gènes de chaque génome représentant un réseau de neurones de la population initiale. De cette façon, les gènes contiennent un historique de tous les changements structurels. Cette information est indispensable pour la conception d'un opérateur de croisement valide.

La mutation dans les réseaux de neurones NEAT augmente progressivement la taille des génomes induisant des structures différentes: leurs représentations ne seront pas nécessairement équivalentes. Selon les cas, les génomes peuvent avoir des tailles différentes.

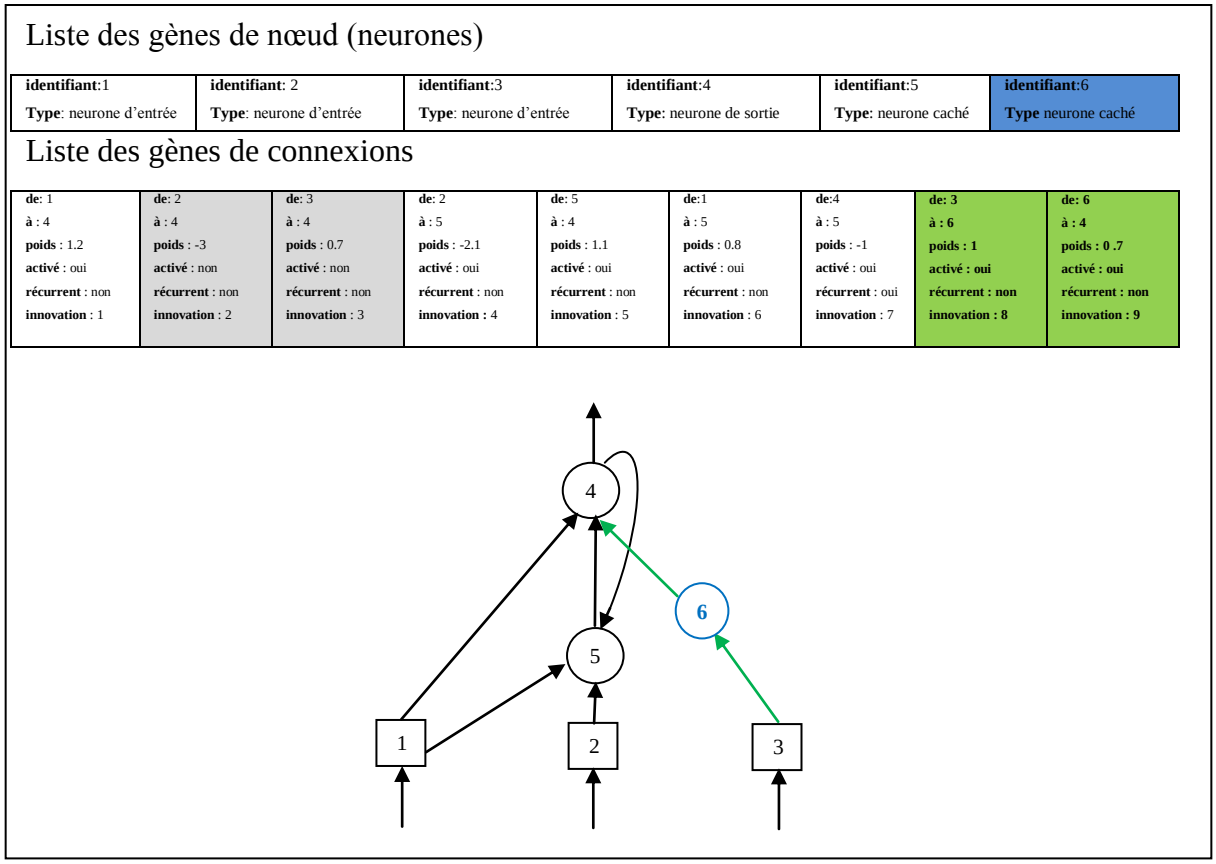


Figure 5. 5- Mutation structurelle dans un réseau NEAT en ajoutant un neurone.

Dans d'autres cas, des gènes dans la même position sur différents génomes peuvent exprimer des caractéristiques complètement différentes. En outre, des gènes exprimant la même caractéristique peuvent apparaître dans des positions différentes sur des génomes différents.

5.3.3. Croisement utilisant les Marqueurs historiques

Pour effectuer le croisement, le système doit être en mesure de spécifier quels sont les gènes qui correspondent entre les individus d'une population dont la topologie est diversifiée. À cet effet, les réseaux de neurones NEAT gardent la trace de l'origine historique de chaque gène. À l'apparition d'un nouveau gène à travers la mutation, un numéro d'innovation global est incrémenté et affecté à ce gène. Ce numéro représente la chronologie de chaque gène dans le système.

Lors d'un croisement, le réseau de neurones NEAT réalise un alignement des gènes des deux génomes. Les gènes présents chez les deux génomes avec le même numéro d'innovation sont nommés «*matching genes*». Les gènes présents dans un seul des deux génomes se distinguent par leur position dans le génome de leurs parents: ceux qui ne correspondent pas au milieu

sont dits «*disjoint genes*», tandis que les «*excess genes* » ne correspondent pas à la fin. Les deux représentent la structure qui n'est pas présente dans l'autre génome.
Pour générer les descendants, les gènes sont choisis aléatoirement parmi l'un des parents pour les «*matching genes*», tandis que les «*disjoint genes*» et les «*excess genes* » sont hérités du parent le plus apte. Dans le cas où la fitness des deux parents est la même, le choix des gènes est dû aussi au hasard.

Les numéros d'innovation sont des marques historiques offrant au réseau de neurones NEAT une nouvelle capacité consistant en la mise en œuvre de croisements beaucoup plus robustes et sans analyse topologique coûteuse. Ainsi, des génomes de structures différentes restent compatibles tout au long de l'évolution, et le problème de l'appariement des différentes topologies est essentiellement évité. De cette façon, le réseau de neurones NEAT évite le problème de permutation et garantit la création de descendants valides. La figure 5.6 illustre le procédé de croisement entre deux individus.

5.3.4. Spéciation pour protéger l'innovation

Les marques historiques offrent aussi aux réseaux de neurones NEAT la possibilité de la spéciation de la population basée sur la similarité topologique, de sorte que les individus concourent principalement au sein de leurs propres espèces au lieu de concourir avec la population en général. De cette façon, les innovations topologiques sont protégées et ont le temps pour optimiser leur structure avant qu'ils n'entrent en concurrence avec les autres espèces de la population.

Les réseaux de neurones NEAT utilisent l'une des techniques les plus populaires de nichage «*niching*»: le partage explicite de performance « *explicit fitness sharing* ». C'est une méthode dans laquelle les individus de la population sont regroupés en fonction de la similitude de leurs génomes. Ensuite, la fitness de chaque individu est ajustée et ce en la partageant entre les membres de ce groupe avant que toute sélection ne se produise. Cela garantit que les individus similaires (les individus qui sont trop proches les uns des autres) dans la population soient punis, et concourt à la conservation de la diversité.

Afin de regrouper la population dans des niches de génomes similaires, à chaque génération, les génomes sont classés séquentiellement en espèces. Un génome est placé dans l'espèce du premier représentant compatible avec ce génome. Le représentant d'une espèce consiste en un individu choisi aléatoirement dans l'espèce à la génération précédente. Ils sont

compatibles seulement si la distance δ entre ce génome et le représentant de l'espèce est inférieure à un seuil de compatibilité δ_τ.

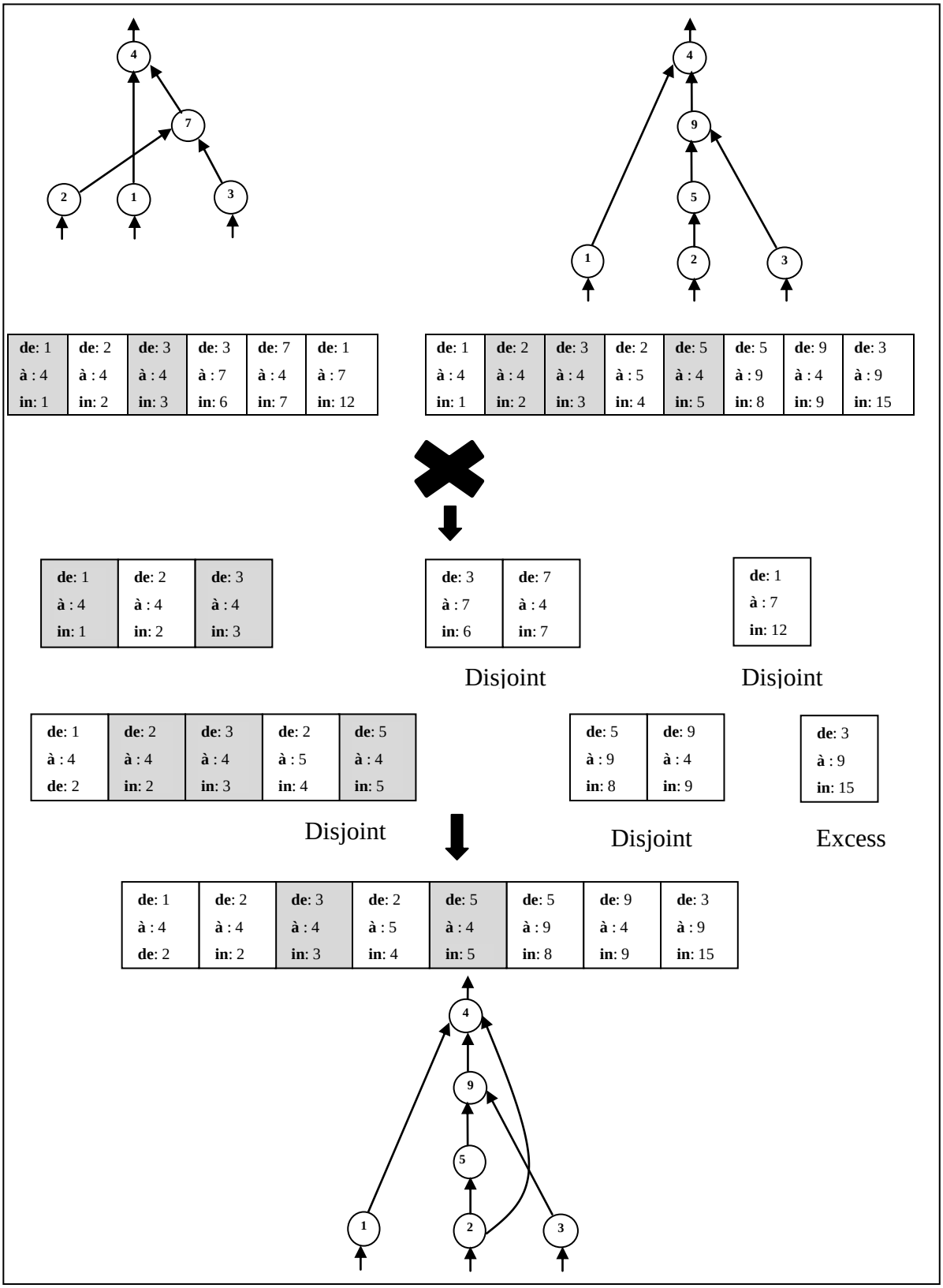


Figure 5. 6- Croisement dans les réseaux de neurones NEAT

Le nombre des gènes qui ne correspondent pas entre deux génomes permet de définir une distance de compatibilité:

$$\delta = \frac{c_1 E}{N} + \frac{c_2 D}{N} + c_3 W \tag{5.1}$$

On note ***E*** le nombre d'*excess genes*, ***D*** le nombre de *disjoint genes*, ***W*** la différence moyenne des poids des *matching genes* y compris aussi les gènes désactivés et ***N*** le nombre de gènes du plus gros génome. Les coefficients c_1, c_2 et c_3 sont ajustés selon l'importance apportée à chaque facteur.

Dans le cas où un génome n'appartient à aucune espèce, une nouvelle espèce est créée avec ce génome comme représentant.

La fonction de fitness ajustée « *shared fitness* » f'_i d'un individu *i* est calculée en fonction de sa distance δ par rapport à chaque autre organisme j dans la population:

$$f'_i = \frac{f_i}{\sum_{j=1}^{n} sh(\delta(i, j))} \tag{5.2}$$

Où *sh* désigne la fonction de partage « *sharing function* » qui dépend de la distance entre deux individus:

$$sh(\delta(i, j)) = \begin{cases} 0 \, si \quad \delta(i, j) > \delta_t \\ 1 \, sinon \end{cases} \tag{5.3}$$

Ce qui signifie que la fitness de chaque individu est divisée par la taille de l'espèce. Ainsi, une espèce ayant une meilleure fitness ne va pas s'imposer, car si sa taille devient trop grande, sa fitness va fortement diminuer.

$$f'_i = \frac{f_i}{\textit{le nombre d'organisme de l'espèce du génome i}} \tag{5.4}$$

Chaque espèce est affectée d'un nombre potentiellement différent de descendants en proportion à la somme des fitness ajustées f_i de ses organismes membres. Les espèces se reproduisent ensuite en éliminant d'abord les membres les moins performants de la population. La population entière est alors remplacée par les descendants des organismes restants dans chaque espèce. Dans le cas où la fitness de l'ensemble de la population ne s'améliore pas pendant plus de 20 générations, seules les deux premières meilleures espèces

sont autorisées à se reproduire, en recentrant la recherche dans les espaces les plus prometteurs.

5.3.5. Minimisation de la dimensionnalité

Contrairement aux autres approches de NeuroEvolution, les réseaux de neurones NEAT commencent avec une population uniforme de réseaux simples sans neurones cachés. De nouvelles structures sont introduites progressivement par les deux opérateurs de mutation structurelle au cours de l'évolution. Seuls les ajouts qui améliorent la performance sont retenus, ce qui aide les réseaux NEAT à découvrir les petits réseaux sans structure superflue et à réaliser un apprentissage rapide. Ceci nous permet d'affirmer que les réseaux NEAT explorent l'espace de recherche progressivement. Ils cherchent toujours à travers un espace de recherche de dimension minimale et passent à un espace plus grand lorsque la performance dans les petits espaces de recherche stagne. La réduction de la dimensionnalité de l'espace de recherche permet de chercher la solution la plus efficace possible.

L'une des raisons qui n'a pas permis aux autres méthodes d'évolution de la topologie et des poids du réseau de neurones réside dans le fait de ne pas commencer avec une population minimale de réseaux simples. Ceci est dû au fait que sans la présence de la diversité topologique dans la population initiale, les innovations topologiques ne survivraient pas. Le problème de protection de l'innovation n'est pas traité par ces méthodes de sorte que des réseaux avec les principaux ajouts structurels sont susceptibles de ne pas se produire. Tandis que la spéciation permet aux réseaux de neurones NEAT d'utiliser au départ une population minimale.

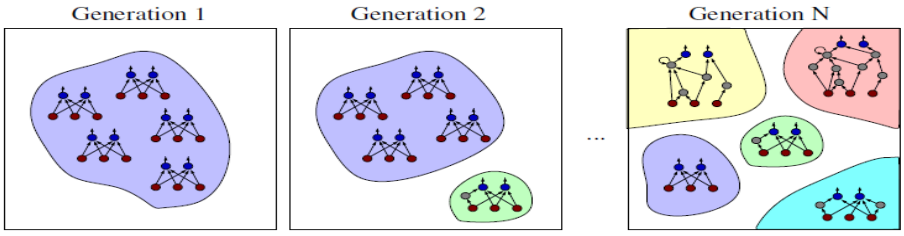


Figure 5. 7- un exemple d'évolution d'une population de réseaux de neurones NEAT [Kohl, 2009]

5.4. Application des réseaux de neurones NEAT pour le contrôle des créatures artificielles

5.4.1. Configuration des paramètres des réseaux de neurones NEAT

Les mêmes paramètres expérimentaux sont utilisés dans toutes les expérimentations; ils ne sont pas réglés spécifiquement pour une expérience particulière. Ces valeurs sont empiriquement choisies par une série des cas de test. Chaque configuration est testée sur 20 exécutions de simulation. Chaque exécution est arrêtée après 300 générations.

Les quatre expérimentations ont utilisé une population de 80 réseaux NEAT. Les multiplicateurs utilisés pour modifier le score final de la mesure de compatibilité sont c_1 = 1.0, c_2 = 1.0, et c_3 = 0.4. Dans toutes les expérimentations, le seuil de compatibilité δ_t = 0.26. Si la fitness maximale d'une espèce ne s'améliore pas dans les 15 générations, les réseaux dans les espèces stagnantes ne sont pas autorisés à se reproduire. Le représentant de chaque espèce est transféré à la nouvelle population sans mutation; ceci est fourni par l'élitisme de l'espèce. Dans chaque génération, la probabilité de croisement de deux génomes est 80%. Dans chaque espèce, la probabilité d'une nouvelle mutation de nœud est de 3% et la probabilité de l'ajout d'une nouvelle connexion est 7%. Il y a une probabilité de 10% pour que les poids de connexion d'un génome soient mutés; dans ce cas, chaque poids a une chance de 10% de se voir attribué une nouvelle valeur aléatoire et une chance de 90% d'être uniformément perturbé.

La fonction d'activation utilisée dans toutes les unités des réseaux de neurones NEAT est la sigmoïde bipolaire:

$$f(x) = \frac{2}{1+e^{(-\partial.x)}} - 1 \qquad (5.5)$$

Ou x est la somme pondérée des entrées d'un neurone et le paramètre ∂détermine la pente de la fonction sigmoïde bipolaire qui est fixée dans ce travail à 2.

Les données environnementales recueillies par les capteurs sont représentées en virgule flottante qui sont ensuite introduits dans le système de contrôle neuronal. L'utilisation de la sigmoïde bipolaire comme fonction d'activation permet au réseau de neurones NEAT d'accepter une gamme de valeurs d'entrée qui peuvent être des nombres positifs ou négatifs. La fonction d'activation sigmoïde bipolaire permet également au réseau de produire des sorties compatibles avec les effecteurs des agents incarnés, qui exigent des données présentées

dans un format bipolaire. Par conséquent, en utilisant cette fonction de transfert, le réseau NEAT est capable de traiter efficacement les données sensorielles bipolaires et générer, le cas échéant, les sorties bipolaires pour le bon fonctionnement des effecteurs des créatures virtuelles. La fonction sigmoïde bipolaire est illustrée par la figure 5.8.

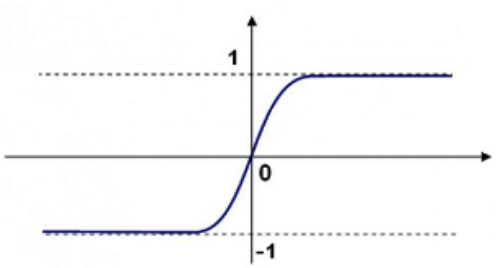


Figure 5. 8- Graphique de la fonction sigmoïde bipolaire

5.4.2. Résultats des expérimentations

Nous montrons ici la capacité de notre système à développer une variété de stratégies de locomotion pour quatre espèces des créatures dont la morphologie est complètement différente. Les graphiques de la fonction de fitness représentent la fitness du meilleur et du pire individu ainsi que la fitness moyenne de la population à chaque génération.

a. Résultat de la créature de type « Rampant »

L'objectif de cette expérimentation est de faire évoluer des créatures capables de se déplacer, dans un environnement tridimensionnel, sur la plus grande distance à partir de la position de départ dans le temps alloué en adoptant une stratégie d'alternance pour le mouvement du "bras" de la créature pour tirer le reste du corps vers l'avant. La figure 5.9 montre les progrès réalisés au cours des 300 générations.

Au début de l'évolution les créatures commencent rapidement à apprendre à interagir avec l'environnement en effectuant des mouvements aléatoires, mais la population apprend rapidement à chronométrer le mouvement du bras en séquences alternées, entraînant ainsi un score de la fonction de fitness plus élevé. Dans la 51ème génération, la meilleur fitness a presque été trouvée et peu d'améliorations ont pu être réalisées dans les générations suivantes. La meilleure fitness a été enregistrée à la génération 111 quand un score de fitness de 63.20 est obtenu.

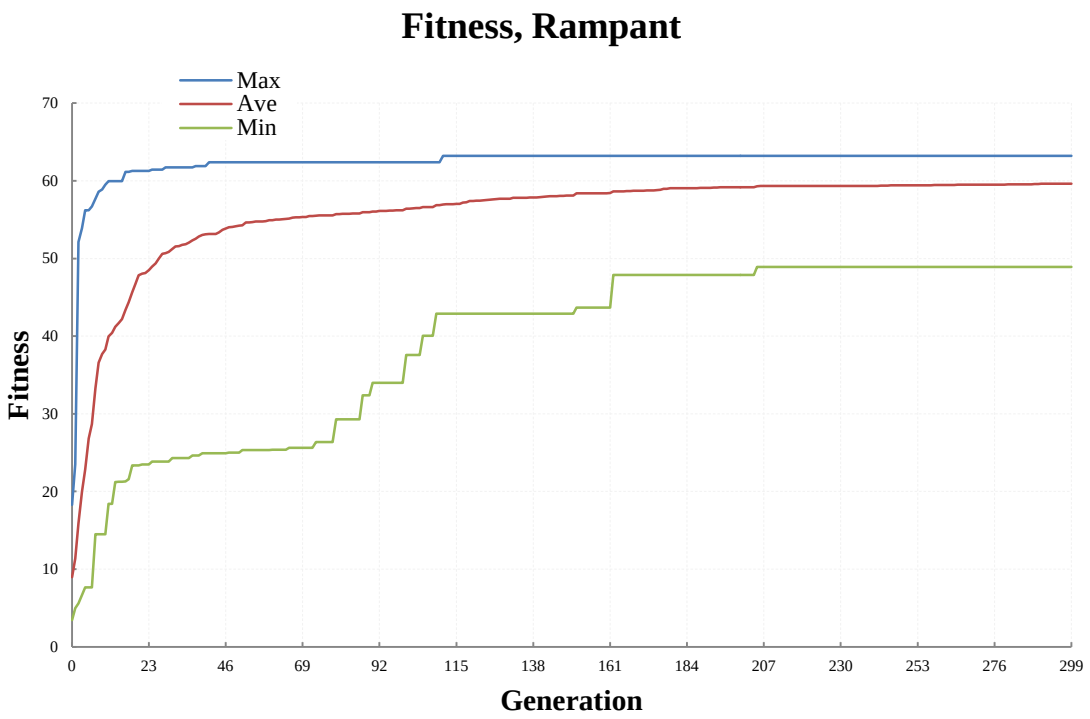


Figure 5. 9- Graphique de la fonction de fitness du premier type " Rampant ". Évolutions de la meilleure fitness, de la pire fitness et de la fitness moyenne de la population utilisant le réseau de neurone NEAT sur 20 exécutions, en moyenne.

Le résultat de cette simulation montre que le réseau de neurones NEAT permet de développer des stratégies de locomotion impliquant la manipulation d'un système de contrôle simple. Un simple contrôle, consiste en la jointure de type « Hinge » avec 1 degré de liberté et les effecteurs contribuent à la convergence rapide d'une solution pour cette morphologie. Des séquences de mouvements évolués pour la morphologie « Rampant » sont illustrées par la figure 5.10.

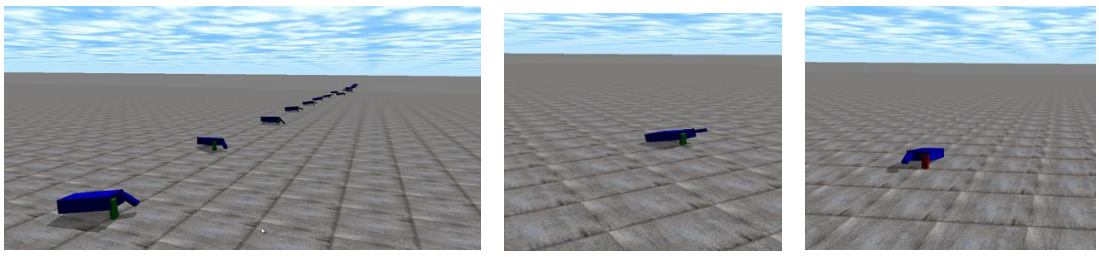


Figure 5. 10- Séquences de mouvements évolués pour la morphologie de type " Rampant ".

b. Résultat de la créature de type «Arm-Based»

Les courbes de la figure 5.11 tracent l'évolution des meilleures fitness, des pires fitness et des fitness moyennes de la population. La meilleure fitness a été enregistrée à la génération 257 quand un score de fitness de 202,21 est obtenu.

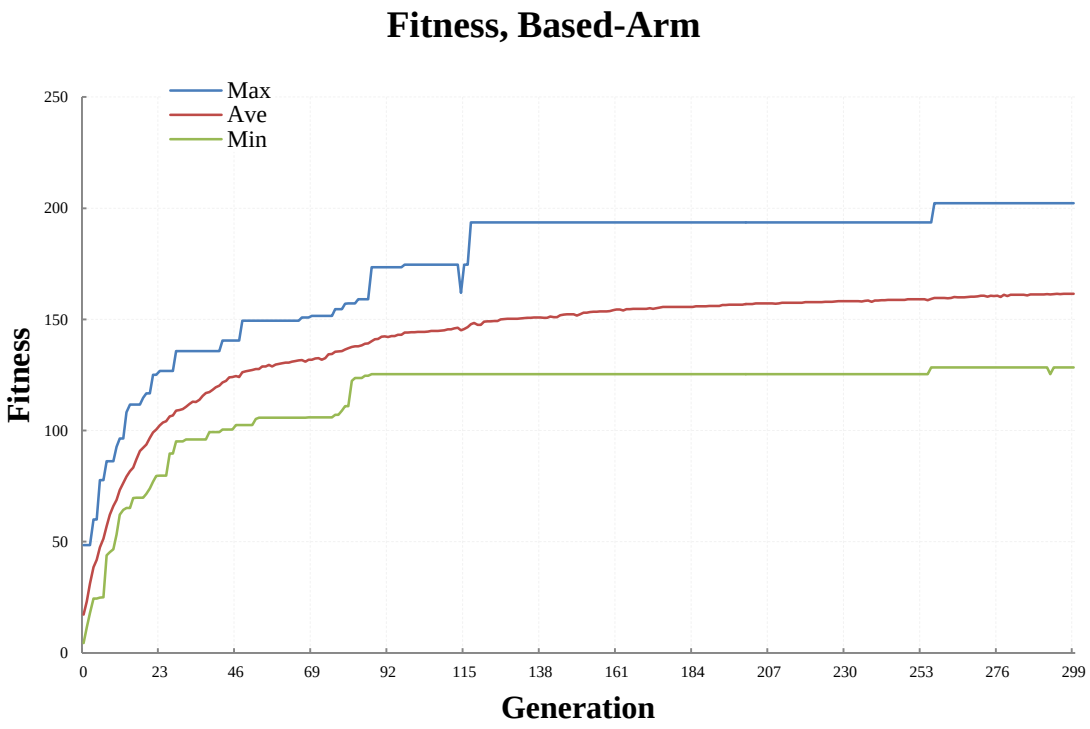


Figure 5. 11- Graphique de la fonction de fitness de la deuxième variété " Arm-based ". Évolutions de la meilleure fitness, de la pire fitness et de la fitness moyenne de la population utilisant le réseau de neurone NEAT sur 20 exécutions, en moyenne.

Dans cette expérience, le réseau de neurones NEAT n'a pas pu apprendre à utiliser idéalement les deux bras et de faire évoluer parfaitement le comportement désiré. Au début de l'évolution, la créature déhanche ses longs "bras" pour se déplacer vers l'avant de sorte que les deux bras ne touchent pas le sol. Ce comportement de locomotion est rapidement supplantée par des individus qui ont commencé à utiliser un seul "bras" pour se tirer vers l'avant en agrippant leur «main» dans le sol et en tirant le torse vers l'avant et balançant l'autre «bras» pour tenter de propulser le reste du corps vers l'avant en même temps. Dans cette expérience, le réseau de neurones NEAT n'a pas réussi à apprendre comment alterner le mouvement des bras pour permettre à chacun des deux de tirer le reste de corps vers l'avant et chronométrer leur mouvement. Des séquences de mouvements évolués pour la morphologie « Arm-Based» sont illustrées par la figure 5.12.

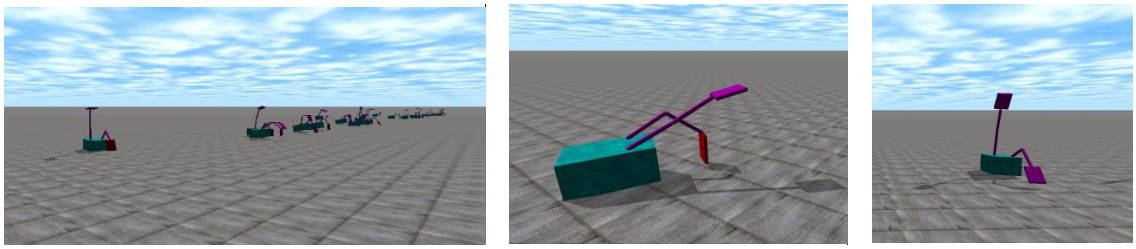


Figure 5. 12- Séquences de mouvements évolués pour la morphologie " Arm-Based ".

c. Résultat de la créature de type « Trémie »

La culbute est considérée comme la stratégie de locomotion la plus complexe à développer, où la créature « Trémie » a besoin d'estimer la force à appliquer correctement sur les jointures pour générer le comportement désiré et être capable de manipuler ses effecteurs de manière à avoir les pieds en position pour un atterrissage.

En d'autres termes, pour reproduire un comportement de culbute, la créature doit avoir un minimum de connaissance du mouvement de projectile et être en mesure de déterminer où l'atterrissage va s'effectuer après chaque mouvement des sauts successifs réalisés. Par ailleurs, la créature doit avoir également la capacité de chronométrer précisément la rotation de son corps pour permettre à ses pieds d'influer sur l'atterrissage de sorte qu'il puisse immédiatement effectuer une autre séquence de culbute.

Au début de la simulation, la plupart des créatures de la population chutent rapidement sur le terrain, ou sautent dans une direction aléatoire et omettent d'atterrir sur leurs pieds à cause des sorties aléatoires affectées aux effecteurs de la créature. Après quelques générations, certains individus apprennent à contrôler la direction de leur premier saut et améliorent ainsi leurs fitness par rapport aux créatures sautant dans des directions aléatoires. Toutefois, après leur premier saut, ils ne peuvent pas encore atterrir sur leurs pieds. Ce comportement s'est poursuivi continué jusqu'à la génération 26, lorsque plusieurs créatures ont appris à chronométrer leur saut initial de sorte que le corps entier fasse un tour complet dans l'air (culbute) et atterrisse sur ses pieds pour continuer le saut en culbute.

Les résultats, représentés graphiquement dans la figure 5.13, montrent un type d'évolution nommé équilibre ponctué « punctuated equilibrium» où la population subie de longues périodes de stagnation, avec une augmentation spectaculaire du score de la fonction de fitness maximale entre ces longues périodes de stagnation. La meilleure fitness a été enregistrée à la génération 282 quand un score de fitness de 221,97 est obtenu. Des séquences de mouvements évolués pour la morphologie « Trémie » sont illustrées par la figure 5.14.

Les résultats de cette simulation démontrent la robustesse des solutions évoluées utilisant les réseaux de neurone NEAT pour la génération et la reproduction des comportements de locomotion complexe. Les créatures «Trémie» ont pu réussir à apprendre la façon d'appliquer correctement des forces à leurs effecteurs afin de contrôler leur trajectoire.

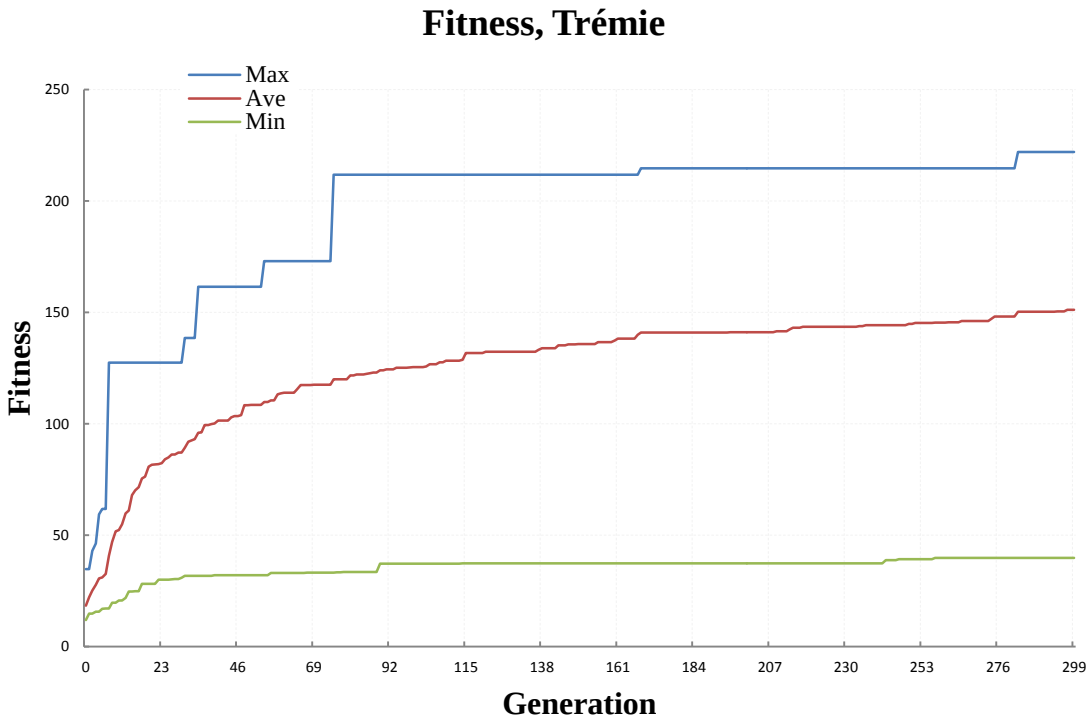


Figure 5. 13- Graphique de fitness de la troisième variété " Trémie ". Évolutions de la meilleure fitness, de la pire fitness et de la fitness moyenne de la population utilisant le réseau de neurone NEAT sur 20 exécutions, en moyenne.

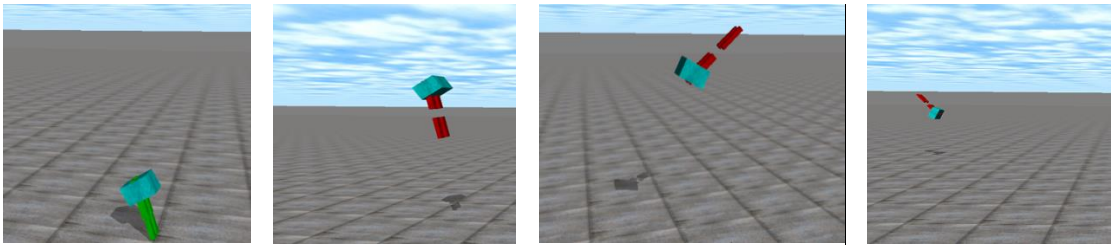


Figure 5. 14- Séquences de mouvements évolués pour la morphologie " Trémie ".

d. Résultat de la créature de type « Coureur »

Dans cette expérimentation, les créatures artificielles doivent apprendre à manipuler correctement leurs appendices afin de produire des comportements de locomotion intelligents

et efficaces leurs permettant de se déplacer vers l'avant. Les créatures de type « Coureur » ont rapidement appris à utiliser les deux appendices en les balançant simultanément vers l'avant, puis en les mettant sur le sol et en les poussant pour produire un mouvement vers l'avant. Au cours des générations, l'amélioration progressive est presque imperceptible, mais la distance parcourue par chaque génération a augmenté progressivement, comme illustré à la figure 5.15. La meilleur fitness a été enregistrée à la génération 282 quand un score de fitness de 146.88 est obtenu. Des séquences de mouvements évolués pour la morphologie « Coureur » sont illustrées par la figure 5.16.

Encore une fois, les résultats de cette simulation démontrent la capacité de notre système à générer des solutions robustes et intelligentes sur une période raisonnable pour une morphologie complexe impliquant des membres connectés à travers des jointures « Ball and Socket Joints », très similaire à la façon dont les bras et les jambes d'un être humain sont attachés.

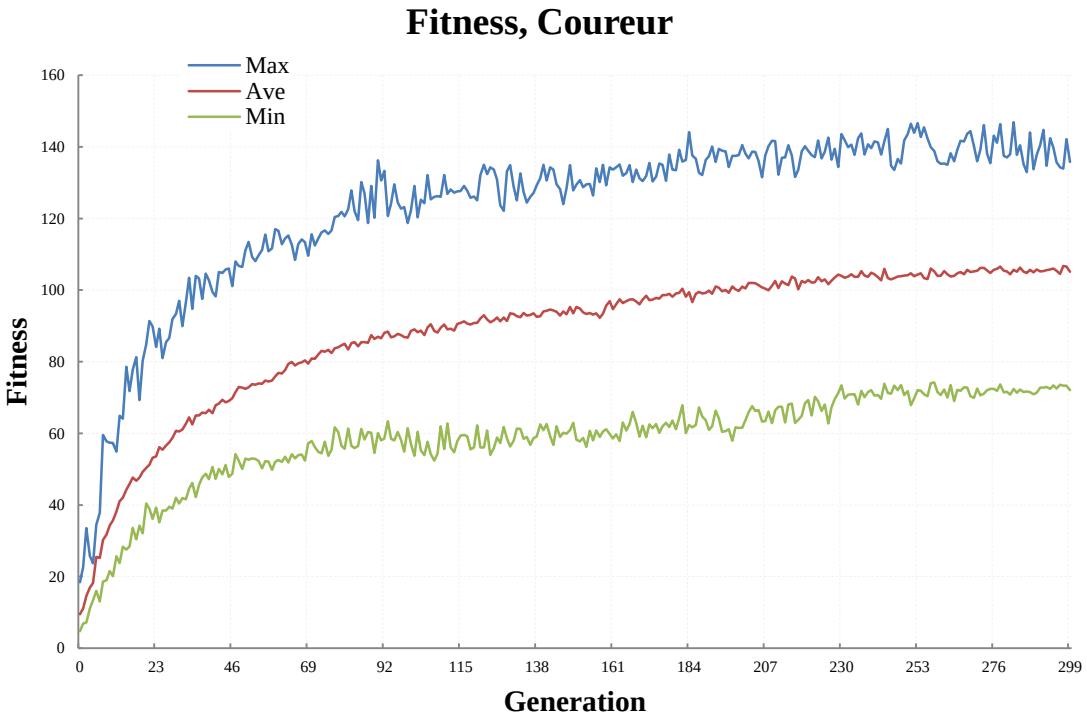


Figure 5. 15- Graphique de fitness de la quatrième variété " Coureur ". Évolutions de la meilleure fitness, de la pire fitness et de la fitness moyenne de la population utilisant le réseau de neurone NEAT sur 20 exécutions, en moyenne.

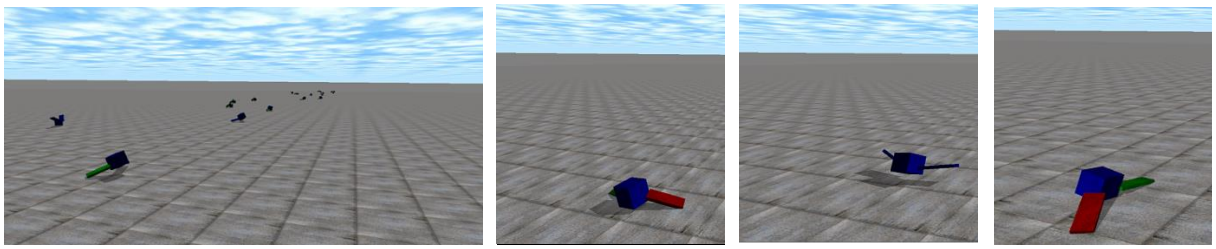


Figure 5. 16- Séquences de mouvements évolués pour la morphologie " Coureur ".

5.5. Discussion de notre modèle

L'objectif de cette thèse est de développer un Framework pour l'évolution des comportements des créatures artificielles. Les lois de l'environnement virtuel ont été inspirées du monde réel dont le but est de transférer les solutions évoluées des créatures virtuelles vers des machines physiques. Les résultats des expérimentations montrent que l'évolution des comportements intelligents dans un environnement virtuel limité par les lois de la physique est réalisable. Les créatures montrent la capacité d'apprendre selon un style d'apprentissage défini: «apprentissage par renforcement ». Les créatures ont également acquis la capacité de coordonner et chronométrer leurs mouvements pour générer des comportements de locomotions efficaces, et ce faisant, elles sont capables d'apprendre à s'adapter à leur environnement. En outre, elles sont capables de faire évoluer ces méthodes efficaces de locomotion sans aucune intervention externe. Bien que les simulations exploitent quatre morphologies de créature prédéterminées, les créatures démontrent leur robustesse et leur capacité de faire le meilleur usage de leur morphologie pour faire évoluer des stratégies de locomotion efficaces.

Tableau 5. 1- Résumé de nos résultats

Créature	Score Maximale	Taille de génome	Génération	Difficulté
Rampant	63.20	17	111	0.37
Arm-Based	202,21	78	257	0.85
Trémie	221,97	37	282	0.94
Coureur	146.88	119	282	0.94

La méthode de NeuroEvolution NEAT n'est pas optimisée pour une morphologie particulière. Cependant, certaines morphologies se prêtent à des solutions de locomotion simples, comme illustré dans le tableau 5.1. Les résultats les plus surprenants ont été obtenus avec la morphologie de type « Trémie », qui se révèle être la morphologie la plus difficile à faire évoluer une solution de locomotion optimale. En comparaison, la morphologie la plus simple,

Rampant, requiert seulement 37% de temps de simulation pour faire évoluer un comportement intelligent. La créature de type *Coureur* utilise un réseau de neurones plus complexe que celui utilisé par la créature de type *Trémie*, avec 119 gènes. La créature de type *Coureur* nécessite le même temps de calcul que la créature, moins complexe, de type *Hopper* pour faire évoluer un comportement intelligent.

La créature de type *Trémie*, avec ses 37 gènes requis 94% de temps de simulation pour faire évoluer un comportement de locomotion que l'on pourrait qualifier comme «adéquat». En réalité, la créature de type *Trémie* n'a pas encore atteint son plein potentiel relativement à sa fonction de fitness dans les 300 générations prévues dans nos simulations. Bien que, nous avons essayé d'augmenter le nombre de génération, à chaque exécution de simulation, jusqu'à plus de 1500 génération, la morphologie de type *Trémie* semble encore être en évolution. Il est probable que la difficulté rencontrée par le processus de NeuroEvolution NEAT avec la créature de type *Trémie* est due aux moyens de locomotion restrictifs qui sont imposés par sa morphologie. La créature de type *Trémie* ne peut se déplacer qu'en sautant d'un endroit à un autre; ce comportement nécessite la planification, le chronométrage et la connaissance de l'environnement. En outre, le type de saut effectué par la créature trémie (effectuant des culbutes de 360 degrés dans l'air), nécessite un timing précis pour perfectionner les débarquements et effectuer plusieurs sauts consécutifs. Ce type de comportement implique le contrôle précis des sorties effectrices et une connaissance approfondie des rouages de l'environnement. Ceci est probablement la raison pour laquelle la population de trémies continue à évoluer même après les 1500 générations de chaque exécution de simulation.

Tableau 5.2- Résumé des résultats obtenus utilisant les réseaux de neurones évolués génétiquement [Ruebsamen, 2002]

Créatures	Score Maximale	Génération	Difficulté
Rampant	30.79	72	0.24
Arm-Based	190.30	221	0.74
Trémie	119.89	276	0.92
Coureur	95.64	200	0.67

Les résultats démontrent également que la complexité du génome n'est pas un facteur important par rapport aux contraintes imposées par la morphologie des créatures.

Les créatures générant les comportements de locomotion nécessitent une connaissance précise de l'environnement, telles que d'être capable de prédire la force nécessaire pour sauter une distance basée sur l'attraction gravitationnelle. Ceci exige une augmentation de temps de

calcul pour faire évoluer un comportement locomotif intelligent. Ces créatures utilisent le temps supplémentaire pour faire évoluer les modèles de leur environnement qui peuvent inclure le mouvement de projectile, la friction, la dynamique des corps rigides et la rotation. En outre, les créatures font évoluer la capacité d'utiliser ces modèles en apprenant à contrôler leurs structures physiques par l'application de forces par l'intermédiaire de leurs effecteurs.

Le tableau 5.2 illustre le résumé des résultats obtenus des quatre expériences dans [Ruebsamen, 2002]; où les réseaux neuronaux, dont la topologie est fixe et seulement les poids des connexions du réseau qui sont évolués génétiquement, sont utilisés comme une technique d'évolution des contrôleurs de locomotion capable de générer les mêmes comportements obtenus dans nos expériences pour les même créatures. Bien que, les réseaux de neurones évolués génétiquement convergent plus vite vers la meilleure solution optimale (qui peuvent l'atteindre durant les 300 générations), les scores de la fitness maximale obtenus avec notre modèle, basé sur les réseau de neurone NEAT, montrent que les solutions de locomotion fournies par notre modèle sont plus efficaces que celles obtenues dans [Ruebsamen, 2002]. La lenteur de la convergence de notre modèle vers la solution optimale et efficace peut être expliquée par la manière de rechercher dans l'espace des solutions adopté par les réseaux de neurone NEAT. Ces derniers cherchent dans l'espace des solutions d'une façon progressive, ils partent d'une population de réseaux simples sans neurones cachés et augmentent leur complexité au cours de l'évolution jusqu'à correspondre à la complexité du problème (dans notre cas, jusqu'à atteindre la topologie de réseaux qui peut contrôler le mouvement des créatures et qui peuvent générer le comportement désiré). Ce processus prend un peu de temps pour atteindre la solution optimale, mais il offre un avantage qui est la génération de solutions plus efficaces ayant une complexité de topologie moins élevée(i.e. la créature de type *Trémie* a besoin d'un génotype qui se compose de 660 gènes [Ruebsamen, 2002] et la créature de type *Coureur* a besoin d'un génotype ayant plus de 976 gènes [Ruebsamen, 2002] tandis que dans note modèle le génotype de la créature de type *Trémie* se compose de 37 gènes et le génotype de la créature de type *Coureur* se compose de 119 gènes. En résumé, Notre modèle, basé sur la méthode NeuroEvolution NEAT, a induit plusieurs avantages: des solutions de locomotion plus efficaces pour une utilisation optimale de la morphologie de la créature avec une petite topologie optimale du contrôleur neuronal.
Les résultats montrent la capacité des réseaux de neurones de type NEAT à faire évoluer des stratégies de locomotion intelligentes pour des créatures virtuelles autonomes dont la

morphologie est fixe. En conséquence, les performances des réseaux de neurones NEAT déjà démontrées dans d'autres domaines sont également confirmées dans le domaine de la simulation comportementale. En outre, les résultats démontrent que les réseaux de neurones NEAT peuvent éventuellement être une solution viable pour l'évolution des solutions de contrôle efficace pour des machines physiques.

5.6. Conclusion

Dans ce travail, nous avons exploité les réseaux de neurones NEAT pour le développement d'une variété de stratégies de locomotion dans un environnement virtuel physique. Nous avons testé les capacités de généralisation des réseaux de neurones NEAT sur quatre tâches: le rampement, la course, la culbute et le mouvement basé sur de long bras. Bien que les simulations utilisent quatre créatures dont la morphologie est différente, les créatures montrent une robustesse et une capacité d'exploitation optimale de leur morphologie pour faire évoluer les comportements souhaités sans la nécessité d'une connaissance à priori de la topologie du réseau de neurones. En conclusion, ce travail a confirmé que les performances des réseaux de neurones NEAT démontrés précédemment dans d'autres domaines peuvent être exploitées avec succès dans la simulation comportementale.

6.Réseau de régulation génétique pour le contrôle et la simulation de la locomotion de créatures artificielles

6.1. Introduction

Ce chapitre étudie l'application d'une technique de calcul évolutionnaire pour faire évoluer les comportements de créatures virtuelles qui peuplent un environnement virtuel réaliste. Notre approche utilise un modèle computationnel du réseau de régulation génétique, qui s'inspire du mécanisme de contrôle cellulaire de cellules réelles. Habituellement utilisés pour contrôler les cellules virtuelles dans les modèles de développement, des travaux récents ont montré que les réseaux de régulation génétiques sont également capables de contrôler différents types d'agents. Ce chapitre explique la façon dont un réseau de régulation génétique peut être évolué afin de contrôler une variété des créatures virtuelles articulées. Pour ce faire, les entrées et les sorties du réseau sont directement liées aux capteurs et effecteurs des créatures. Pour évaluer cette approche, nous avons comparé sa performance à l'une des plus récentes méthodes évolutionnaires abouties, l'algorithme NEAT. Les résultats montrent que le modèle de réseau de régulation génétique peut éventuellement être une solution viable pour l'évolution des solutions de contrôle pour des machines physiques**.**

6.2. Réseau de régulation génétique

Les réseaux de régulation génétique (Gene Regulatory Network : GRN) sont des structures biologiques qui déterminent le comportement interne des cellules vivantes. Ils régulent l'expression des gènes en renforçant et en inhibant la transcription de certaines parties de l'ADN. À cet effet, les cellules utilisent des capteurs de protéines distribués sur leurs membranes; ceux-ci fournissent des informations cruciales pour guider les cellules pendant leur cycle. La figure 6.1 présente le fonctionnement du réseau. Beaucoup des modèles informatiques modernes de ces réseaux existent. Ils sont utilisés à la fois pour simuler les réseaux de régulation génétique réels [Reil, 1999; Banzhaf, 2003; Hotz, 2003]

ainsi que pour contrôler les agents [Doursat, 2008; Guo, Meng et al., 2009; Nicolau, Schoenauer et al., 2010; Joachimczak and Wróbel, 2011; Cussat-Blanc and Pollack, 2012b; Cussat-Blanc, Sanchez et al., 2012].

Lorsqu'il est utilisé à des fins de simulation, le GRN est généralement codé dans une chaîne de bits, comme l'ADN est codé dans une chaîne de nucléotidiques. De même que pour dans l'ADN réel, une séquence de gènes commence par une séquence particulière, appelée le promoteur en biologie [Lifton, Goldberg et al., 1978].
Dans l'ADN réel, cette séquence est représentée par un ensemble de quatre protéines: TATA où T représente la thymine et A l'adénine. Dans [Reil, 1999], Torsten Reil est l'un des premiers à proposer un modèle biologiquement plausible de réseaux de régulation génétique. Le modèle est basé sur une séquence de bits dans laquelle le promoteur est constitué de quatre bits 1010. Le gène est codé directement après ce promoteur, alors que les éléments de régulation sont codés avant le promoteur.
Pour visualiser les propriétés de ces réseaux, il utilisa la visualisation graphique pour observer la variation de la concentration des différentes protéines du système. Il souligne trois différents types de comportements des réseaux de régulation génétique générés d'une façon aléatoire: stables, chaotiques et cycliques. Il observa également que ces réseaux sont capables de se remettre des changements aléatoires du génome, en produisant le même pattern quand ils sont mutés d'une manière aléatoire.

En 2003, Wolfgang Banzhaf formula un nouveau réseau de régulation génétique fortement inspiré de la biologie [Banzhaf, 2003]. Il utilisa un génome composé de multiples entiers de 32 bits codé comme une chaîne de bits. Chaque gène commence par un promoteur codé par n'importe quel nombre entier se terminant par la séquence "XYZ 01010101". Cette séquence se produit avec une probabilité 2^{-8} (0,39%). Le gène suivant ce promoteur est ensuite codé avec cinq nombres entiers de 32 bits (160 bits) et les éléments de régulation sont codés en amont du promoteur par deux nombres entiers, un pour le renforcement et l'autre pour l'inhibition cinétique. Le modèle de Banzhaf confirme l'hypothèse soulignée par le modèle de Reil[Reil, 1999]; les mêmes propriétés émergent de son modèle.

À partir de ces modèles séminaux, de nombreux modèles computationnels ont été initialement utilisés pour contrôler les cellules des modèles de développement artificiels [Hotz, 2003; Doursat, 2008; Joachimczak and Wróbel, 2011]. Ils ont permis de simuler la

première étape de l'embryogenèse des organismes vivants et plus particulièrement les mécanismes de différenciation cellulaire. Un des problèmes initiaux de ce domaine de recherche est le problème du drapeau français [Wolpert, 1969] dans lequel un organisme virtuel doit produire un rectangle composé de trois couleurs (bleu, blanc et rouge). Cela stimule la capacité de différenciation dans un environnement spatial des cellules. De nombreux modèles ont abordé ce benchmark avec des cellules contrôlées par un réseau de régulation génétique [Knabe, Schilstra et al., 2008; Cussat-Blanc, Bredeche et al., 2011; Joachimczak and Wróbel, 2011]. Plus récemment, les réseaux de régulation génétique ont prouvé leur capacité à contrôler des comportements complexes dans diverses situations: ils sont utilisés pour contrôler les agents virtuels [Panzoli, Luga et al., 2008; Nicolau, Schoenauer et al., 2010; Cussat-Blanc, Sanchez et al., 2012], les essaims réels et les robots modulaires [Guo, Meng et al., 2009; Cussat-Blanc and Pollack, 2012a].

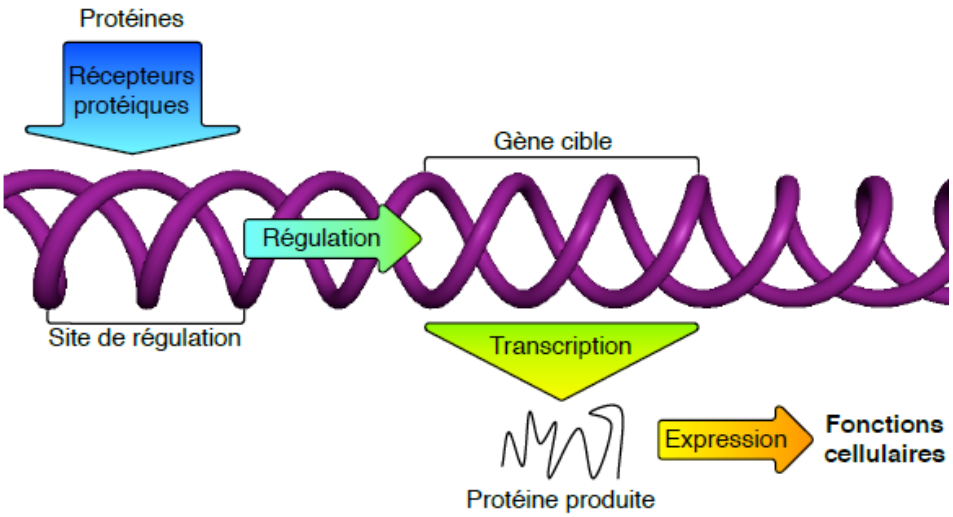


Figure 6. 1- Schéma de l'action du réseau de régulation génétique durant la division cellulaire. Dans les cellules réelles, le réseau utilise des signaux externes pour renforcer ou inhiber la transcription de gènes cibles. L'expression de ces gènes permettra de déterminer le comportement final de la cellule [Cussat-Blanc, 2009].

6.3. Notre modèle

Le réseau de régulation génétique utilisé pour développer des stratégies de locomotion est un modèle simplifié basé sur le modèle Banzhaf [Banzhaf, 2003]. Ce type de réseaux a été déjà utilisé avec succès dans d'autres applications. Il est capable de développer des

morphologies de robots modulaires [Cussat-Blanc and Pollack, 2012a], produire des images 2D [Cussat-Blanc and Pollack, 2012b], contrôler des cellules conçues pour optimiser l'aménagement du parc éolien [Wilson, Awa et al., 2013], contrôler des paramètres d'apprentissage par renforcement [Harrington, Awa et al., 2013] et conduire une voiture de course virtuelle [Sanchez and Cussat-Blanc]. Ce modèle est conçu pour des objectifs computationnels uniquement et ne pas pour simuler le réseau biologique.

Ce modèle est composé d'une série de protéines abstraites. Une protéine *pr* est composée de trois étiquettes:

- L'étiquette de protéine id_{pr} qui identifie la protéine ;
- L'étiquette d'activateur « enhancer » enh_{pr} qui identifie le facteur de correspondance de renforcement entre deux protéines ;
- L'étiquette inhibitrice « inhibtor » inh_{pr} qui identifie le facteur de correspondance d'inhibition entre deux protéines.

Ces étiquettes sont codées avec un nombre entier appartenant à l'intervalle [0, p] où la limite supérieure *p* peut être réglée pour contrôler la précision du réseau. En plus de ces étiquettes, une protéine est également définie par sa concentration qui varie au cours du temps avec une dynamique particulière, nous la décrirons ultérieurement. Une protéine peut être:

- Une protéine d'*entrée*: c'est une protéine dont la concentration est assurée par l'environnement, elle régule d'autres protéines, mais elle n'est pas régulée par les autres protéines ;
- Une protéine de *sortie*: c'est une protéine possédant une concentration utilisée comme une sortie du réseau, elle est régulée par les autres protéines, mais elle ne régule pas les autres protéines ;
- une protéine *régulatrice*: une protéine interne qui régule les autres protéines et qui est régulée par les autres protéines.

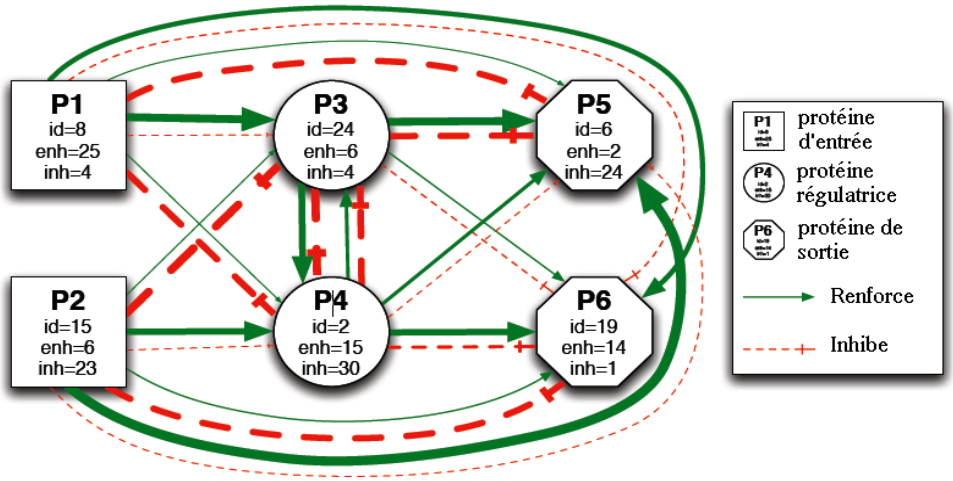


Figure 6. 2- Représentation graphique du GRN. Les nœuds sont les protéines et les arêtes représentent l'affinité de renforcement et d'inhibition entre deux protéines. Plus grosses sont les arêtes, plus proches sont les protéines.

Avec cette structure, la dynamique du GRN est calculée en utilisant les étiquettes de protéine. Elles déterminent le taux de productivité de l'interaction entre deux protéines. Pour cela, l'affinité d'une protéine pr_a pour une autre protéine pr_b est donnée par le facteur de renforcement $u^+_{pr_a pr_b}$ et le facteur d'inhibition $u^-_{pr_a pr_b}$ qui sont calculés comme suit:

$$\begin{aligned} u^+_{pr_a pr_b} &= p - |enh_{pr_a} - id_{pr_b}| \\ u^-_{pr_a pr_b} &= p - |inh_{pr_a} - id_{pr_b}| \end{aligned} ; \qquad (6.1)$$

Les protéines sont ensuite comparées par paires en fonction de leurs facteurs de renforcement et d'inhibition. Pour une protéine pr_a, le renforcement g_{pr_a} et l'inhibition h_{pr_a} totale sont donnés par:

$$\begin{aligned} g_{pr_a} &= \frac{1}{N}\sum_{pr_b}^{N} c_{pr_b} e^{\beta(u^+_{pr_a pr_b} - u^+_{\max})} \\ h_{pr_a} &= \frac{1}{N}\sum_{pr_b}^{N} c_{pr_b} e^{\beta(u^-_{pr_a pr_b} - u^-_{\max})} \end{aligned} ; \qquad (6.2)$$

Où N est le nombre total de protéines dans le réseau, c_{pr_b} est la concentration de la protéine pr_b , $u^+_{\max}$ est le facteur maximum de renforcement observé, $u^-_{\max}$ est le facteur maximum d'inhibition observé et β est un paramètre de contrôle qui sera détaillé plus tard. À

chaque pas de temps, la concentration d'une protéine pr_a change conformément à l'équation différentielle suivante:

$$\frac{dc_{pr_a}}{dt} = \frac{\delta(g_{pr_a} - h_{pr_a})}{\phi} \tag{6.3}$$

Où ϕ est facteur de normalisation pour s'assurer que la somme totale de la concentration de protéine régulatrice et de protéine de sortie est égale à 1. β et δ sont deux facteurs scalaires qui ont une influence sur les taux de réaction du réseau. β affecte l'importance des facteurs de correspondance et δ est utilisé pour modifier le niveau de production des protéines dans l'équation différentielle. En résumé, la régulation serait souple quand les deux valeurs sont minimes. Cependant, le comportement de la régulation devient inattendu avec l'augmentation des valeurs.

La figure 6.2 résume le fonctionnement du modèle. Les arêtes représentent le facteur de correspondance de renforcement (en vert) et d'inhibition (en rouge) entre deux protéines. Leur épaisseur représente la valeur: Plus grosses sont les arêtes, plus proches sont les protéines.

6.4. L'utilisation des GRN pour faire évoluer les comportements dans de créatures artificielles

6.4.1. Lier le GRN aux capteurs et effecteurs de créature

Comme nous l'avons mentionné précédemment, les créatures ont des capteurs pour recueillir des données provenant de l'environnement. Ces données sont introduites au GRN qui retourne des valeurs pour résoudre le problème donné. Le tableau 4.1 (voir le chapitre 4) présente la variété des capteurs utilisés dans cette simulation ainsi que la plage de valeurs que les capteurs sont capables de détecter.

Pour utiliser le Réseau de régulation génétique pour l'évolution des comportements dans créatures virtuelles, notre souhait principal est de maintenir la connexion entre le GRN et les capteurs et les effecteurs de la créature la plus simple possible. À notre avis, l'approche doit être capable de gérer la réactivité nécessaire pour contrôler une créature, le bruit possible des capteurs et les situations inattendues. Pour atteindre cet objectif, nous convertissons les signaux d'entrée et de sortie en des valeurs de concentration normalisées, et ce dans le but de trouver une bonne façon de relier les effecteurs de créature au GRN. Avant d'être calculées

par le GRN, les données environnementales collectées par les capteurs sont normalisées dans l'intervalle [0,1] avec la formule suivante:

$$norm\left(v_{sensor_i}\right) = \frac{v_{sensor_i} - \min_{sensor_i}}{\max_{sensor_i} - \min_{sensor_i}} \tag{6.4}$$

où v_{sensor_i} est la valeur du capteur (i) à normaliser, $\min_{sensor_i}$ est la valeur minimale du capteur et $\max_{sensor_i}$ est la valeur maximale du capteur.

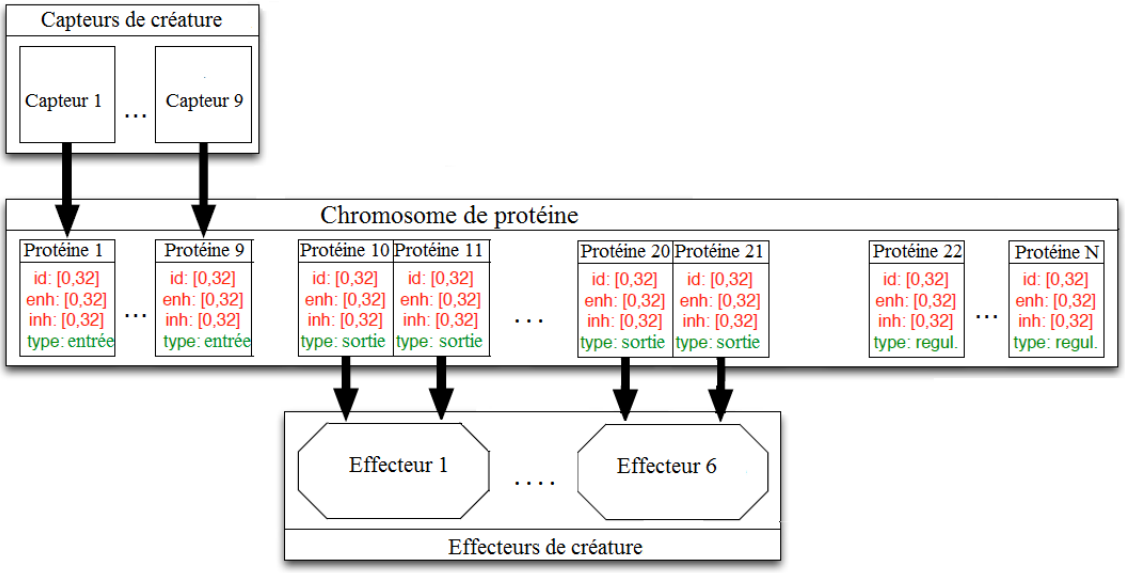


Figure 6. 3- Organisation du chromosome de protéine et la liaison aux capteurs et effecteurs de créature.

Dès que les concentrations de protéine d'entrée de GRN sont mises à jour, la dynamique du GRN est exécutée une fois pour propager la modification de concentration sur tout le réseau. La concentration des protéines de sortie est ensuite introduite directement aux entrées d'effecteurs qui la convertissent en la vitesse désirée par la mise à l'échelle linéaire des entrées d'effecteurs aux valeurs correspondantes de la plage des entrées indiquées dans le tableau 4.2 (voir le chapitre 4).

Selon le type de la liaison utilisée, les effecteurs exercent une force de traction/poussée ou une force d'extension/flexion sur un des degrés de liberté de la jointure. Par conséquent, deux protéines, O^+ pour la force de poussée ou la force de flexion et O^- pour la force de traction ou la force d'extension, sont nécessaires pour déterminer les valeurs finales de la vitesse désirée

du moteur, qui doivent être fournies aux effecteurs des créatures. La vitesse désirée est calculée comme suit:

$$\text{vitesse} = \frac{c(o^+) - c(o^-)}{c(o^+) + c(o^-)} \tag{6.5}$$

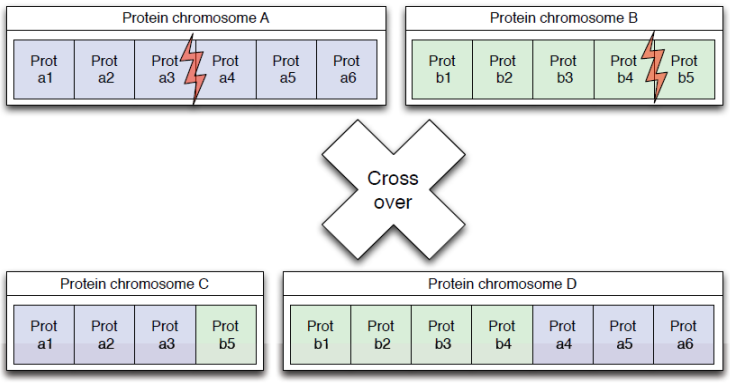
où $c(o^*)$ est la concentration de la protéine de sortie o^*. La figure 6.3 montre la connexion du GRN à la créature virtuelle.

L'utilisation de deux protéines de sortie pour contrôler la vitesse d'un seul moteur permet au GRN de produire un signal de sortie compatible avec les effecteurs des créatures virtuelles. Ces effecteurs nécessitent des données présentées dans un format bipolaire. Le cycle continu d'entrées et de sorties vers et depuis le GRN est le mécanisme qui peut produire des comportements de locomotion efficaces et intelligents.

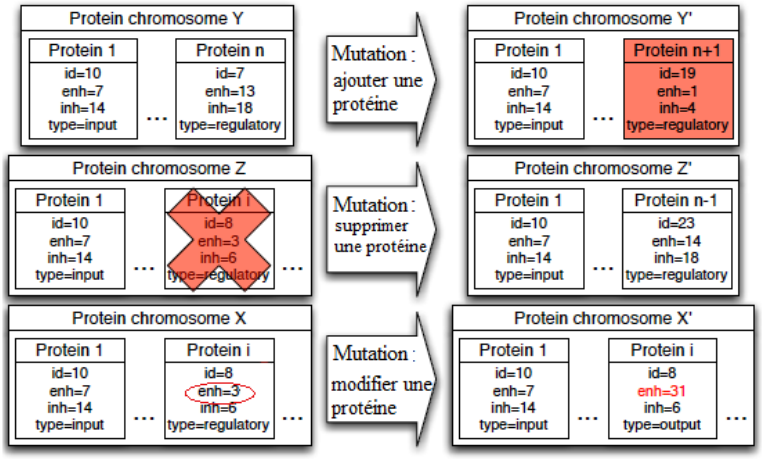
6.4.2. Encodage génétique du GRN

Le Schéma de codage génétique du GRN est conçu pour permettre aux gènes des protéines correspondantes d'être facilement alignés lorsque deux génomes sont croisés pendant la recombinaison. Chaque génome comprend deux chromosomes indépendants. Le premier est défini comme un chromosome de longueur variable des protéines indivisibles. Chaque protéine est codée par trois nombres entiers compris entre 0 et p, qui correspondent aux trois étiquettes. Dans ce travail, p est fixé à 32 et les protéines du génome sont organisées comme suit: les protéines d'entrée en premier, suivi par les protéines de sortie et puis les protéines régulatrices. Les entrées et les sorties présentées dans la section précédente sont toujours liées à la même protéine, comme illustré sur la figure 6.3. Le deuxième chromosome est codé par des valeurs à virgule flottante en double précision, ces valeurs correspondent aux deux facteurs scalaires présentés dans la section précédente.

Avant que le GRN ne puisse contrôler les créatures, le réseau de régulation génétique doit être optimisé. Dans ce travail, nous utilisons un algorithme génétique avec des opérateurs de croisement et de mutation particuliers, représentés dans la figure 6.4, pour optimiser le premier chromosome:



(a) Croisement.



(b) Mutation

Figure 6. 4- Opérateurs de croisement et de mutation appliqués sur le chromosome de la protéine

Le croisement ne peut s'effectuer ni entre deux protéines ni entre deux étiquettes d'une même protéine, ce qui permet d'assurer l'intégrité des deux sous-réseaux lorsque le GRN est divisé en deux réseaux. Lors de l'assemblage d'un autre nouveau GRN, les connexions locales sont maintenues avec cet opérateur et seules les nouvelles connexions entre les deux réseaux sont créées.

La mutation dans ce modèle de GRN peut modifier à la fois une étiquette au sein d'une protéine sélectionnée au hasard et la structure du réseau. Les étiquettes protéiques sont mutées comme dans la mutation standard, une étiquette choisie au hasard qu'elle soit perturbée ou non à chaque génération. La mutation structurelle se produit de deux manières, chaque mutation dilate ou rétrécit la taille du génome en ajoutant ou en supprimant une ou plusieurs protéines. Dans la mutation, (ajout d'une protéine) une seule nouvelle protéine régulatrice

avec un identifiant aléatoire ainsi que des valeurs du facteur de renforcement et d'inhibition aléatoires sont ajoutées au premier chromosome. Dans la mutation (suppression d'une protéine) une protéine régulatrice existante est enlevée du premier chromosome.
Le deuxième chromosome est utilisé pour faire évoluer la dynamique des variables β et δ. Ce chromosome utilise les méthodes de mutation et de croisement standards.

6.5. Résultats des expérimentations

Dans chaque expérimentation, la performance du GRN est comparée à la performance de la méthode de neuro-évolution NEAT pour montrer si le GRN améliore les performances de la recherche évolutionnaire ou non. Pour chaque expérience, les deux algorithmes sont testés (les résultats obtenus avec la méthode NEAT ont été bien discutés dans le chapitre précédent). Les graphiques des fitness sont représentatifs de la moyenne de 20 exécutions de simulation par morphologie.

6.5.1. Paramètres de GRN

Les mêmes paramètres expérimentaux sont utilisés dans toutes les expérimentations; ils ne sont pas réglés spécifiquement pour une telle expérience particulière. Ces valeurs sont empiriquement choisies par une série des cas de test. Chaque configuration est testée dans 20 exécutions de simulation dont chacune est arrêtée après 300 générations. Les niveaux de signification sont calculés en utilisant le test de Student.
Cette section présente les paramètres expérimentaux que nous avons utilisés dans ce travail. Les exécutions évolutionnaires utilisent une population de 80 génomes de GRN. À chaque génération, la probabilité de croisement de deux génomes est de 70% et celle de la mutation est fixée à 80%. Le nombre minimum et maximum de protéines régulatrices de chaque chromosome est 4 et 20, respectivement. Les deux facteurs scalaires β et δ sont évoluent dans l'intervalle [0.5, 2]. Les valeurs inférieures à 0,5 produisent des réseaux réactifs tandis que les valeurs supérieures à 2 produisent des réseaux très instables.

6.5.2. Résultats et mesure de la performance

Dans l'expérimentation de rampement, les deux méthodes GRN et NEAT sont capables de trouver constamment des solutions efficaces permettant à la créature de type *Rampant* d'apprendre à utiliser ses deux appendices frontaux pour se tirer vers l'avant (voir la figure 5.9 et 6.5). Le GRN a réalisé un score de 80,85 tandis que l'algorithme NEAT a réalisé un score

de 63,20 après les 300 générations d'évolution. Le GRN a ainsi dépassé l'algorithme NEAT d'une valeur de 21,78% (test de Student, la valeur-p <0,005), ce qui confirme que former de simples créatures virtuelles pour effectuer des tâches qui impliquent la manipulation d'un système de contrôle simple est favorable aux réseaux de régulation génétique.

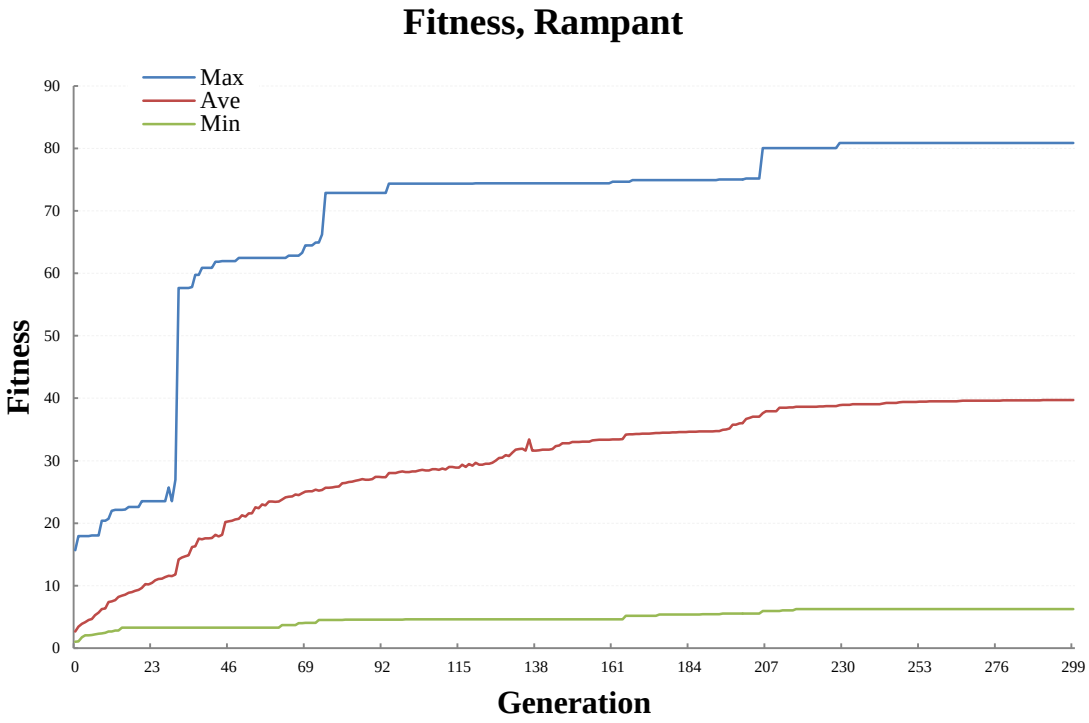


Figure 6. 5- Évolution de la fitness de la première variété "Rampant". Évolution de la meilleure fitness, de la pire fitness et de la fitness moyenne de la population utilisant le GRN en moyenne sur 20 exécutions.

Avec la créature *Arm-Based*, les deux méthodes n'ont pas réussi à maîtriser la façon d'alterner systématiquement le mouvement des deux bras de la créature pour permettre à chacun de tirer le reste du corps vers l'avant et chronométrer leur mouvement. Le contrôleur, qui a été évolué par le GRN et les réseaux de neurones NEAT permettent à cette créature d'utiliser un seul "bras" pour avancer vers l'avant en positionnant sa «main» sur le sol et en tirant le reste du corps vers l'avant. La performance du modèle proposé basé sur les réseaux de neurones NEAT, avec un score de 202,20, dans ce cas est meilleure que celle du modèle utilisant le GRN dont le score réalisé est de 147,42 (bien que la différence est statistiquement significative, la valeur-p <0,0001). Les courbes des figures 5.11 et 6.6 nous montrent les scores obtenus par les créatures *Arm-Based* produites par le GRN et les RN-NEAT.

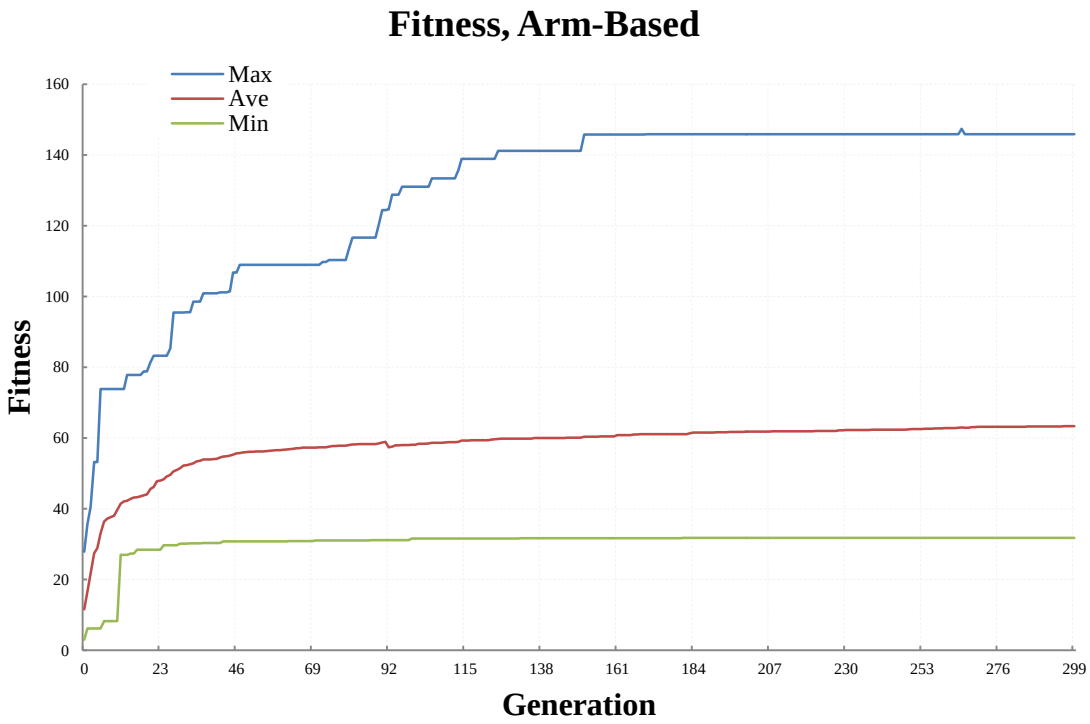


Figure 6. 6- Évolution de la fitness de la deuxième variété "Arm-based". Évolution de la meilleure fitness, de la pire fitness et de la fitness moyenne de la population utilisant le GRN en moyenne sur 20 exécutions.

Dans l'expérience du saut en culbute, les deux techniques GRN et NEAT ont réussi à trouver constamment de bonnes solutions permettant à la créature trémie d'apprendre à contrôler ses culbutes et la réalisation de nombreux sauts en culbutes consécutivement. Le GRN a réalisé un score de 303,69 sur les 300 générations d'évolution et le NEAT a obtenu un score de 221,97 après les 300 générations d'évolution. Ainsi, le GRN a supplanté le réseau NEAT d'une valeur équivalente à 26,9%. Cependant, les différences entre le GRN et le réseau NEAT ne sont pas statistiquement significatives ($p > 0,2$) en raison d'un grand écart-type dans cette expérience particulière, mais le GRN surpasse NEAT sur les 20 exécutions.

Les résultats, comme illustré dans les figures 5.13 et 6.7, démontrent un type d'évolution connu comme l'équilibre ponctué. Dans ce type d'évolution, la population subit de longues périodes de stagnation avec une augmentation spectaculaire en score de la fitness maximale entre ces longues périodes de stagnation.

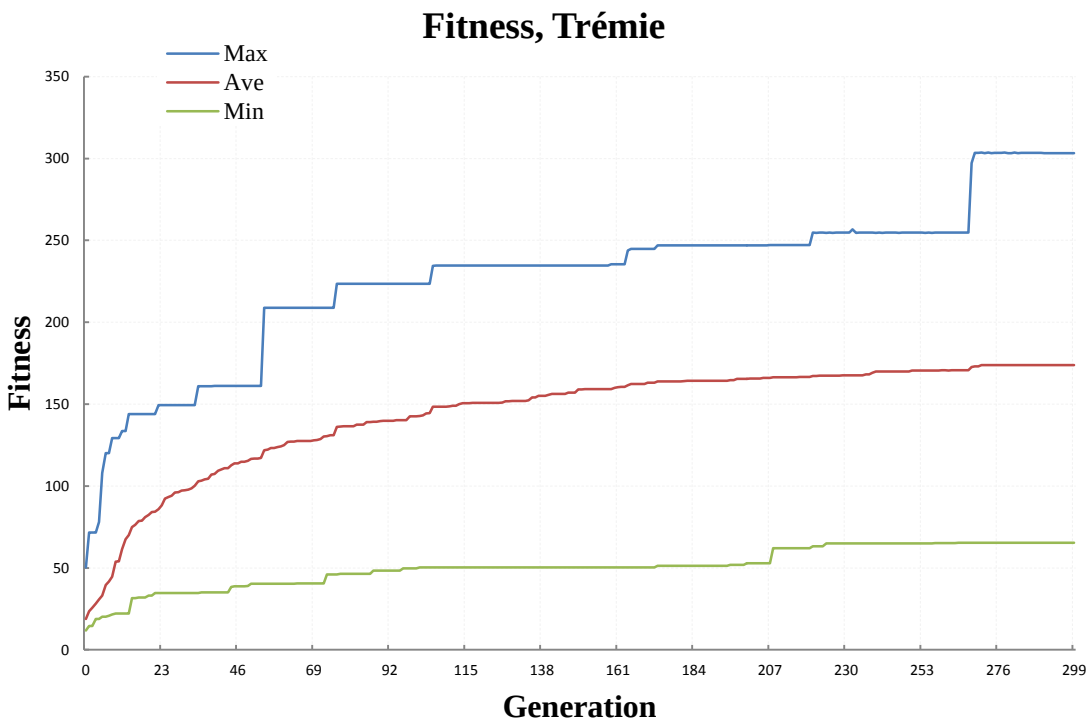


Figure 6. 7- Évolution de la fitness de la troisième variété "Trémie ". Évolution de la meilleure fitness, de la pire fitness et de la fitness moyenne de la population utilisant le GRN en moyenne sur 20 exécutions.

Encore une fois, le GRN a démontré avec succès sa capacité à faire évoluer des comportements de locomotion complexes. Cette créature a pu apprendre la façon d'appliquer la force sur ses effecteurs en vue de chronométrer précisément la rotation de son corps et d'atterrir avec une bonne maîtrise sur ses pieds, afin qu'elle puisse effectuer immédiatement une autre séquence de saut en culbute et de contrôler efficacement sa trajectoire.

Dans l'expérimentation de la créature de type *Coureur*, les deux méthodes, GRN et NEAT, sont capables de trouver constamment des solutions efficaces (voir la figure 5.15 et 6.8) permettant à la créature de type *Coureur* d'apprendre à manipuler les deux bras en les balançant simultanément vers l'avant, afin de les placer sur le sol et les repousser dans le but de produire un mouvement vers l'avant. Le GRN a réalisé un score de 85,82 et le réseau de neurones NEAT a obtenu un score de 146,88 sur les 300 générations d'évolution. Ainsi, le réseau de neurone NEAT a ainsi dépassé le GRN d'un taux égal à 41,45%. La différence entre les deux méthodes s'est avérée extrêmement statistiquement significative ($p < 0{,}0001$).

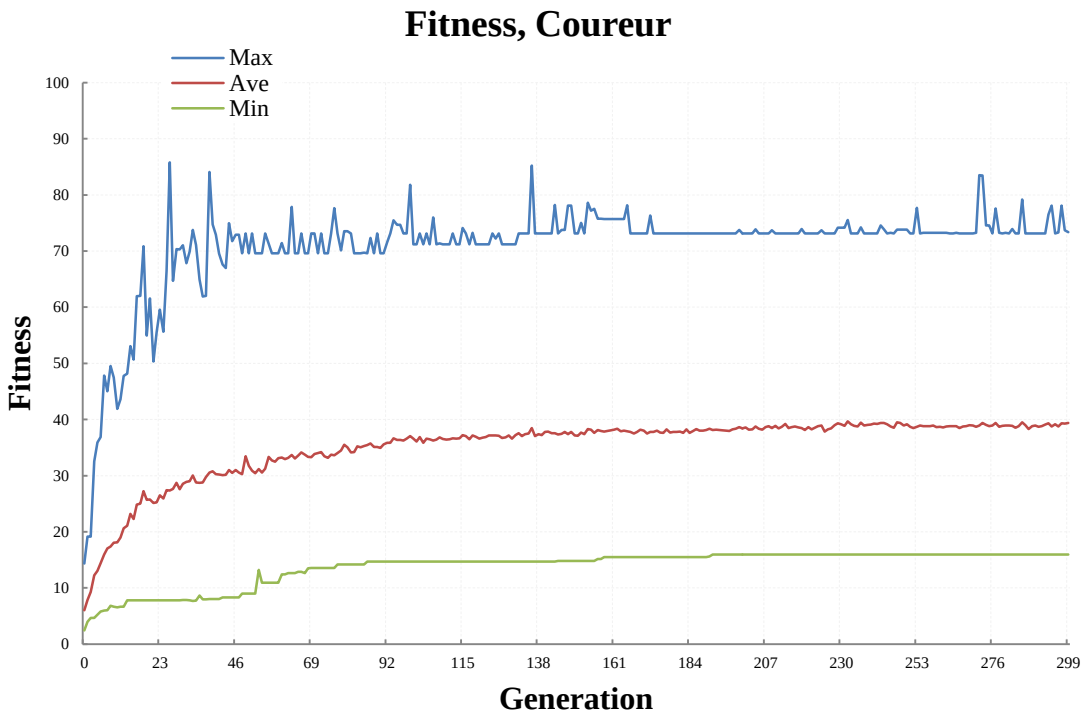


Figure 6. 8- Graphique de fitness de la quatrième variété "Coureur ". Évolution de la meilleure fitness, de la pire fitness et de la fitness moyenne de la population utilisant le GRN en moyenne sur 20 exécutions.

Cette simulation montre que les réseaux de neurones NEAT sont plus efficaces que les GRN dans la tâche d'évolution des comportements de locomotion pour des créatures autonomes employant des jointures complexes possédant jusqu'à trois degrés de liberté. Néanmoins, notre modèle GRN a montré sa capacité à générer des comportements dans un délai raisonnable pour des morphologies impliquant des appendices connectés via des jointures, très similaires à la façon dont les bras ou les jambes d'un être humain sont assemblés.

6.6. Discussion

Les résultats de ces expérimentations montrent que l'utilisation des GRN pour faire évoluer des comportements dans un environnement virtuel est réalisable. Les créatures qui utilisent le GRN pour l'évolution du contrôleur montrent une capacité de traiter et d'utiliser les données sensorielles provenant des jointures et des capteurs d'entrée pour produire des solutions de locomotion efficaces, et, ce faisant, ils sont capables d'apprendre à s'adapter à leur environnement. Les résultats de la simulation montrent que le GRN est une méthode efficace pour la simulation de créatures virtuelles simples capables d'effectuer des tâches qui

impliquent la manipulation d'un système de contrôle simple. Les résultats les plus imprévisibles ont été obtenus avec la morphologie de la trémie, qui s'est avérée être la morphologie la plus difficile à pouvoir faire émerger une solution de locomotion. Le GRN semble être plus efficace lorsqu'il traite des exemples qui nécessitent beaucoup d'informations provenant de l'environnement. L'expérimentation de rampement démontre que lorsque la jointure est très complexe, le GRN ne fournit pas un avantage significatif par rapport au NEAT. En revanche, l'expérimentation de la créature *Arm-based* montre que lorsque la morphologie de la créature est composée de structures symétriques, complexes avec des parties similaires où le système de comportement est nécessaire pour coordonner son mouvement pour générer le comportement souhaité, le GRN ne parvient pas à fournir une solution de locomotion efficace.

Cette baisse de performance peut être expliquée par la mise en correspondance de génotype/phénotype qui fonctionne habituellement avec des problèmes simples, mais ne pouvant traiter les domaines ayant de très grandes solutions phénotypiques qui contiennent des parties similaires. Ce problème pourrait être traité par la recherche d'un codage génératif approprié pour le génome du GRN qui lui permette de compresser la description de la solution de telle sorte que l'information puisse être réutilisée et permette à la solution finale de contenir plus de composants que la description elle-même. En nous basant sur les quatre expérimentations, nous avons conclu que le GRN surpasse les réseaux de neurones NEAT dans la tâche de contrôle des organismes anthropomorphes, dans un environnement 3D physiquement simulé, pour des organismes dont la morphologie est relativement simple, mais cet avantage disparait dès lors que la morphologie devient de plus en plus complexe.

6.7. Conclusion

La contribution présentée dans ce chapitre est la proposition d'une nouvelle approche pour l'évolution des comportements dans les créatures articulées. L'algorithme proposé est inspiré par le GRN - un algorithme pour contrôler les comportements dans les modèles de développement cellulaires. Nous avons démontré les avantages et les inconvénients du GRN sur quatre tâches: le rampement, le mouvement basé sur de longues armes, le saut en culbute et le jogging. Concernant les résultats des deux expérimentations, les mouvements basés sur de longs bras ainsi que le jogging ont confirmé que le GRN ne fournit pas un avantage pour les tâches où la créature se compose de structures complexes avec des parties similaires employant des jointures complexes. Cependant, les deux autres expérimentations, le

rampement et le saut en culbute, ont montré que si la morphologie de la créature est composée de structures simples et de jointures qui sont utilisées pour les assembler sont également simples, l'algorithme du GRN supplante les réseaux de neurones NEAT même si le comportement à générer est complexe ou lorsqu'il a besoin beaucoup d'informations sur son environnement. Les résultats de nos expérimentations en utilisant le GRN comme un contrôleur de créatures soulignent également que cette approche peut être utilisée dans de nombreuses applications à base d'agents, la seule exigence est qu'elle soit capable de convertir les signaux d'entrée et de sortie en valeurs de concentration normalisées et de trouver une bonne façon pour relier les capteurs et les effecteurs de l'agent au GRN. La contribution majeure de ce travail n’est pas simplement un bon résultat, mais plutôt une nouvelle direction de recherche pour faire évoluer des comportements de locomotion efficace en utilisant le codage direct.

7.Conclusion et perspectives

7.1. Conclusion générale

La Conception et la fabrication automatique d'organismes de robots autonomes sont progressivement transférées du domaine de la science-fiction à la science contemporaine. Karl Sims représente l'un des précurseurs dans ce domaine. Celui-ci a réussi à faire évoluer avec succès des créatures virtuelles dont les mouvements sont remarquablement similaires à ceux observés par l'organisme dans le monde réel. Récemment, plusieurs chercheurs ont repris le travail de Sims en adoptant des approches évolutionnaires différentes.

La contribution principale de cette thèse est la proposition de nouvelles approches pour faire évoluer des créatures artificielles. La première approches proposée est inspirée des réseaux de neurones NEAT- un algorithme pour faire évoluer des réseaux neuronaux complexes. Nous avons ainsi réussi à reproduire un nombre important d'expérimentations dans le but de tester les capacités de généralisation des réseaux de neurones NEAT sur quatre types de tâches. Bien que les simulations utilisent quatre créatures dont la morphologie est différente, les créatures montrent une robustesse et une capacité d'exploitation optimale de leur morphologie pour faire évoluer les comportements souhaités sans la nécessité d'une connaissance à priori de la topologie du réseau utilisé. Les réseaux de neurones NEAT ont démontré ainsi leur capacité, non seulement comme un outil d'amélioration des performances des systèmes de NeuroEvolution, mais également en tant que technique qui suit sa propre démarche. Cette démarche dépend de trois propriétés complémentaires (les marques historiques, la protection de l'innovation à travers la spéciation, et la croissance progressive de la structure minimale) et permet ainsi de produire un système capable d'évoluer en faisant émerger des solutions de complexité optimale. En conclusion, ce travail a confirmé que les performances des réseaux de neurone NEAT démontrées précédemment dans d'autres domaines peuvent être exploitées avec succès dans le domaine de la simulation comportementale.

La seconde approche, que nous avons proposée et développée, est inspirée des réseaux GRN – dont le principe est représenté par un algorithme de contrôle des comportements dans des modèles de développement cellulaires. Des expérimentations ont également été reproduites à l'effet de mesurer les diverses propriétés de l'algorithme. Les résultats de ces

expérimentations démontrent que l'algorithme augmente significativement les performances du processus d'évolution des stratégies de locomotion pour des créatures artificielle. Néanmoins, ce gain de performance se voit réduit lorsque la morphologie devient de plus en plus complexe. Les résultats de nos expérimentations en utilisant le GRN comme outil de contrôle de créatures artificielles soulignent également que cette approche peut être utilisée dans de nombreuses applications à base d'agents, la seule exigence est d'être capable de convertir les signaux d'entrée et de sortie en valeurs de concentrations normalisée et de trouver une bonne façon pour relier les capteurs et les effecteurs de l'agent au GRN. La contribution majeure de ce nouveau modèle computationnel que nous avons proposé, n'est pas simplement un résultat probant, mais plutôt une nouvelle direction de recherche pour faire évoluer des comportements de locomotion efficaces en utilisant le codage direct.

En résumé, les objectifs de cette thèse ont été remplis avec succès. L'étude contribue au domaine de la robotique évolutionnaire en proposant deux nouvelles approches d'évolution des stratégies de locomotion de créatures virtuelles. Les résultats obtenus peuvent être utilisés en tant que base pour la recherche future dans ce domaine.

7.2. Perspectives

Cette section décrit les différentes investigations pouvant être entreprises dans un avenir proche en continuité à cette thèse. Comme expliqué précédemment, l'objectif à long terme de cette étude est de faire évoluer des comportements intelligents qui peuvent être implantés dans des machines et les robots réels utilisés dans le monde réel. Ce qui suit est une liste de recommandations pour de futures améliorations aux systèmes proposés.

7.2.1. Système de contrôle

Les développements futurs devraient prendre en considération de nouvelles améliorations dans le modèle des GRN afin de traiter le problème de contrôle des morphologies complexes et d'améliorer son efficacité dans la manipulation les jointures complexes. Une autre perspective consiste à utiliser de nouvelles techniques adoptées dans le domaine de la NeuroEvolution précisément avec les réseaux de neurones NEAT pour résoudre les problèmes qui contournent les réseaux de neurone NEAT dans la tâche de contrôle des créatures.

7.2.2. Environnement et comportement plus complexe

Il est intéressant d'explorer la possibilité d'évolution des comportements plus complexes, tels que le comportement de poursuite, le comportement de recherche de nourriture et générer par évolution des comportements de coopération avec d'autres créatures pour réaliser un but. Une autre perspective consiste à faire évoluer des créatures virtuelles dans des environnements physiques plus complexes. Ceci peut être réalisé en utilisant des écosystèmes artificiels qui semblent être l'une des suites logiques pour faire évoluer d'une manière réaliste des créatures et des environnements artificiels.

7.2.3. Calcul de l'énergie dépensée

Il est intéressant de permettre à notre modèle de calculer l'énergie dépensée par les mouvements des créatures. En effet, nous pourrions sélectionner ainsi les créatures qui génèrent des mouvements minimisant l'énergie consommée. Du point de vue de l'animation, cette perspective permet de produire des mouvements plus réalistes.
En ce qui concerne la robotique, les batteries des robots sont un obstacle à leur autonomie, minimiser leur dépense énergétique augmenterait leur durée de fonctionnement. Dans la perspective de réaliser un écosystème basé sur la sélection naturelle, il est indispensable calculer l'énergie dépensée car elle représente un élément déterminant dans le processus de survie des créatures [Lassabe, 2008].

7.2.4. Embryogenèse artificielle

L'embryogenèse artificielle simule le développement des créatures entières possédant un métabolisme, une morphologie et un comportement à partir d'une cellule unique. L'embryogenèse aurait certainement de nombreuses applications pour l'évolution des créatures artificielles. Nous avons inscrit nos travaux dans un niveau que nous pouvons qualifier de « de l'organe à la créature ». Réaliser le lien avec le niveau « de la cellule à l'organe » permettrai de générer une créature complète à partir d'un composant indifférencié (cellule).

7.2.5. Parallélisme

L'évolution et la simulation des créatures artificielles dans des environnements 3D physiquement simulé demandent beaucoup de puissance de calcul, il est donc souhaitable de paralléliser ces calculs et d'utiliser les cartes physiques qui sont de plus en plus répandues.

Bibliographie

Banzhaf, W. (2003). Artificial regulatory networks and genetic programming. Genetic programming theory and practice, Springer: 43-61.

Bedau, M. A. (2003). "Artificial life: organization, adaptation and complexity from the bottom up." Trends in cognitive sciences 7(11): 505-512.

Bongard, J., V. Zykov, et al. (2006). "Resilient machines through continuous self-modeling." Science 314(5802): 1118-1121.

Brooks, R. A. (1986). "A robust layered control system for a mobile robot." Robotics and Automation, IEEE Journal of 2(1): 14-23.

Cardamone, L., D. Loiacono, et al. (2009). Evolving competitive car controllers for racing games with neuroevolution. Proceedings of the 11th Annual conference on Genetic and evolutionary computation, ACM.

Chaumont, N., R. Egli, et al. (2007). "Evolving virtual creatures and catapults." Artificial life 13(2): 139-157.

Cliff, D. and G. F. Miller (1995a). "Co-evolution of pursuit and evasion II: Simulation methods and results."

Cliff, D. and G. F. Miller (1995b). Tracking the red queen: Measurements of adaptive progress in co-evolutionary simulations. Advances In Artificial Life, Springer: 200-218.

Cussat-Blanc, S. (2009). Créatures artificielles: développement d'organismes à partir d'une cellule unique, Université des Sciences Sociales-Toulouse I.

Cussat-Blanc, S., N. Bredeche, et al. (2011). Artificial gene regulatory networks and spatial computation: A case study. ECAL.

Cussat-Blanc, S. and J. Pollack (2012a). A cell-based developmental model to generate robot morphologies. Proceedings of the fourteenth international conference on Genetic and evolutionary computation conference, ACM.

Cussat-Blanc, S. and J. Pollack (2012b). Using Pictures to Visualize the Complexity of Gene Regulatory Networks. Artificial life.

Cussat-Blanc, S., S. Sanchez, et al. (2012). Simultaneous cooperative and conflicting behaviors handled by a gene regulatory network. Evolutionary Computation (CEC), 2012 IEEE Congress on, IEEE.

Dobzhansky, T. (1973). Nothing in biology makes sense except in the light of evolution.

Doursat, R. (2008). Organically grown architectures: Creating decentralized, autonomous systems by embryomorphic engineering. Organic computing, Springer: 167-199.

Dupas, R. (2004). "Amélioration de performance des systèmes de production: apport des algorithmes évolutionnistes aux problèmes d'ordonnancement cycliques et flexibles." Université d'Artois.

Edelman, G. M. (2008). Biologie de la conscience, Odile Jacob.

Fikes, R. E. and N. J. Nilsson (1972). "STRIPS: A new approach to the application of theorem proving to problem solving." Artificial intelligence 2(3): 189-208.

Floreano, D., P. Husbands, et al. (2008). Evolutionary robotics. Springer handbook of robotics, Springer: 1423-1451.

Floreano, D. and C. Mattiussi (2008). Bio-inspired artificial intelligence: theories, methods, and technologies, MIT press.

Fogel, L. J., A. J. Owens, et al. (1966). EVOLUTION OF FINITE AUTOMATA FOR PREDICTION, DTIC Document.

Francisci, D. (2002). "Algorithmes évolutionnaires et optimisation multi-objectifs en data mining." Laboratoire informatique, signaux et systèmes de Sophia Antipolis UMR 60702002.

Gardner, M. (1970). "Mathematical games: The fantastic combinations of John Conway's new solitaire game "life"." Scientific American 223(4): 120-123.

Golberg, D. E. (1989). "Genetic algorithms in search, optimization, and machine learning." Addion wesley 1989.

Gomez, F. J. and R. Miikkulainen (1999). Solving non-Markovian control tasks with neuroevolution. IJCAI.

Gruau, F., D. Whitley, et al. (1996). A comparison between cellular encoding and direct encoding for genetic neural networks. Proceedings of the 1st annual conference on genetic programming, MIT Press.

Guo, H., Y. Meng, et al. (2009). "A cellular mechanism for multi-robot construction via evolutionary multi-objective optimization of a gene regulatory network." BioSystems 98(3): 193-203.

Harrington, K. I., E. Awa, et al. (2013). Robot coverage control by evolved neuromodulation. Neural Networks (IJCNN), The 2013 International Joint Conference on, IEEE.

Holland, J. (1975). "Genetic Algorithms, computer programs that evolve in ways that even their creators do not fully understand." Scientific American: 66-72.

Hopfield, J. J. (1982). "Neural networks and physical systems with emergent collective computational abilities." Proceedings of the national academy of sciences 79(8): 2554-2558.

Hornby, G. S., H. Lipson, et al. (2001). Evolution of generative design systems for modular physical robots. Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on, IEEE.

Hornby, G. S. and J. B. Pollack (2001a). Body-brain co-evolution using L-systems as a generative encoding. Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001).

Hornby, G. S. and J. B. Pollack (2001b). "Evolving L-systems to generate virtual creatures." Computers & Graphics 25(6): 1041-1048.

Hotz, P. E. (2003). "Combining developmental processes and their physics in an artificial evolutionary system to evolve shapes." On Growth, Form and Computers: 302-318.

Jackson, P. C. (1985). Introduction to artificial intelligence, Courier Corporation.

Joachimczak, M. and B. Wróbel (2011). Evolution of the morphology and patterning of artificial embryos: scaling the tricolour problem to the third dimension. Advances in Artificial Life. Darwin Meets von Neumann, Springer: 35-43.

Klein, J. (2003). Breve: a 3d environment for the simulation of decentralized systems and artificial life. Proceedings of the eighth international conference on Artificial life.

Knabe, J., M. Schilstra, et al. (2008). "Evolution and morphogenesis of differentiated multicellular organisms: autonomously generated diffusion gradients for positional information." Artificial Life XI.

Kohl, N., K. Stanley, et al. (2006). Evolving a real-world vehicle warning system. Proceedings of the 8th annual conference on Genetic and evolutionary computation, ACM.

Kohl, N. F. (2009). "Learning in fractured problems with constructive neural network algorithms."

Komosinski, M. (2000). The world of framsticks: simulation, evolution, interaction. Virtual worlds, Springer.

Komosiński, M. and S. Ulatowski (1999). Framsticks: Towards a simulation of a nature-like world, creatures and evolution. Advances in Artificial Life, Springer: 261-265.

Koza, J. R. (1992). Genetic programming: on the programming of computers by means of natural selection, MIT press.

Langton, C., C. Taylor, et al. (1992). Artificial Life II: Proceedings of the Workshop on Artificial Life, vol. X of SFI Studies in the Sciences of Complexity, Addison-Wesley, Redwood City, CA.

Langton, C. G. (1989). Artificial Life, Santa Fe Institute Studies on the Sciences of Complexity. Proc.

Langton, C. G. (1997). Artificial life: An overview, Mit Press.

Lassabe, N. (2008). Morphogenese etEvolution de Creatures Artificielles, Université de Toulouse.

Lassabe, N., H. Luga, et al. (2007). "A new step for evolving creatures." IEEE-ALife 7: 243-251.

Lifton, R., M. Goldberg, et al. (1978). The organization of the histone genes in Drosophila melanogaster: functional and evolutionary implications. Cold Spring Harbor symposia on quantitative biology, Cold Spring Harbor Laboratory Press.

Lipson, H. and J. Pollack (2006). Evolving physical creatures. Artificial Life VII: Proceedings of the Seventh International Conference on Artificial life.

Lipson, H. and J. B. Pollack (2000). "Automatic design and manufacture of robotic lifeforms." Nature 406(6799): 974-978.

McCulloch, W. S. and W. Pitts (1943). "A logical calculus of the ideas immanent in nervous activity." The bulletin of mathematical biophysics 5(4): 115-133.

Miconi, T. and A. Channon (2006). "An improved system for artificial creatures evolution." Proceedings of Artificial Life X: 255-261.

Moriarty, D. E. and R. Mikkulainen (1996). "Efficient reinforcement learning through symbiotic evolution." Machine learning 22(1-3): 11-32.

Nicolau, M., M. Schoenauer, et al. (2010). Evolving genes to balance a pole. Genetic Programming, Springer: 196-207.

Nolfi, S. (2009). Embodied and Situated Agents, Adaptive Behavior in. Encyclopedia of Complexity and Systems Science, Springer: 2844-2859.

Nolfi, S. and D. Floreano (2000). Evolutionary robotics: The biology, intelligence, and technology of self-organizing machines, MIT press.

Panzoli, D. (2008). Proposition de l'architecture" Cortexionist" pour l'intelligence comportementale de créatures artificielles, Université de Toulouse, Université Toulouse III-Paul Sabatier.

Panzoli, D., H. Luga, et al. (2008). A Reactive Architecture Integrating an Associative Memory for Sensory-Driven Intelligent Behavior. Intelligent Virtual Agents, Springer.

Parizeau, M. (2004). "Réseaux de neurones." GIF-21140 et GIF-64326 124.

Ray, T. S. (2001). "Aesthetically evolved virtual pets." Leonardo 34(4): 313-316.

Rechenberg, I. (1973). "Evolutionary strategie: Evolution." Frommann-Holzboog, Stuttgart.

Reil, T. (1999). Dynamics of gene expression in an artificial genome—implications for biological and artificial ontogeny. Advances in Artificial Life, Springer: 457-466.

Rennard, J.-P. and D. Mange (2002). Vie artificielle: où la biologie rencontre l'informatique: illustré avec Java, Vuibert.

Reynolds, C. W. (1987). "Flocks, herds and schools: A distributed behavioral model." ACM Siggraph Computer Graphics 21(4): 25-34.

Rosenblatt, F. (1958). "The perceptron: a probabilistic model for information storage and organization in the brain." Psychological review 65(6): 386.

Ruebsamen, G. D. (2002). Evolving Intelligent Embodied Agents Within a Physically Accurate Environment, California State University, Long Beach.

Sanchez, S. and S. Cussat-Blanc "Gene regulated car driving: using a gene regulatory network to drive a virtual car." Genetic Programming and Evolvable Machines: 1-35.

Sanza, C. (2001). Evolution d'entités virtuelles coopératives par système de classifieurs, Toulouse 3.

Saravanan, N. and D. B. Fogel (1995). "Evolving neural control systems." IEEE Intelligent Systems 10(3): 23-27.

Shim, Y.-S. and C.-H. Kim (2003). Generating flying creatures using body-brain co-evolution. Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation, Eurographics Association.

Sims, K. (1991). Artificial evolution for computer graphics, ACM.

Sims, K. (1994a). "Evolving 3D morphology and behavior by competition." Artificial life 1(4): 353-372.

Sims, K. (1994b). Evolving virtual creatures. Proceedings of the 21st annual conference on Computer graphics and interactive techniques, ACM.

Smith, R. (2005). "Open dynamics engine."

Spector, L., J. Klein, et al. (2007). Division blocks and the open-ended evolution of development, form, and behavior. Proceedings of the 9th annual conference on Genetic and evolutionary computation, ACM.

Stanley, K. O., B. D. Bryant, et al. (2005). "Real-time neuroevolution in the NERO video game." Evolutionary Computation, IEEE Transactions on 9(6): 653-668.

Stanley, K. O. and R. Miikkulainen (1996). "Efficient reinforcement learning through evolving neural network topologies." Network (Phenotype) 1(2): 3.

Stanley, K. O. and R. Miikkulainen (2002a). "Efficient Evolution of Neural Network Topologies." Network (Phenotype) 1(2): 3.

Stanley, K. O. and R. Miikkulainen (2002b). "Evolving neural networks through augmenting topologies." Evolutionary computation 10(2): 99-127.

Stanley, K. O. and R. Miikkulainen (2004). Evolving a roving eye for go. Genetic and Evolutionary Computation–GECCO 2004, Springer.

Sumathi, S. and S. Paneerselvam (2010). Computational intelligence paradigms: theory & applications using MATLAB, CRC Press.

Tu, X. and D. Terzopoulos (1994). Artificial fishes: Physics, locomotion, perception, behavior. Proceedings of the 21st annual conference on Computer graphics and interactive techniques, ACM.

Von Neumann, J. and A. W. Burks (1966). "Theory of self-reproducing automata." IEEE Transactions on Neural Networks 5(1): 3-14.

Whitley, D., S. Dominic, et al. (1994). Genetic reinforcement learning for neurocontrol problems, Springer.

Wieland, A. P. (1991). Evolving neural network controllers for unstable systems. Neural Networks, 1991., IJCNN-91-Seattle International Joint Conference on, IEEE.

Wilson, D., E. Awa, et al. (2013). On learning to generate wind farm layouts. Proceeding of the fifteenth annual conference on Genetic and evolutionary computation conference, ACM.

Wilson, S. W. (1994). "ZCS: A zeroth level classifier system." Evolutionary computation 2(1): 1-18.

Wittkamp, M., L. Barone, et al. (2008). Using NEAT for continuous adaptation and teamwork formation in Pacman. Computational Intelligence and Games, 2008. CIG'08. IEEE Symposium On, IEEE.

Wolpert, L. (1969). "Positional information and the spatial pattern of cellular differentiation." Journal of theoretical biology 25(1): 1-47.