



Morphogenèse et Évolution de Créatures Artificielles

par

Nicolas Lassabe

Thèse présentée et soutenue le 14 mars 2008 en vue de
l'obtention du grade de Docteur de l'Université de Toulouse
(spécialité informatique)

Membres du jury :

Directeur de thèse : Yves Duthen (Université de Toulouse)

Rapporteur : Jean-Louis Deneubourg (Université Libre de Bruxelles)

Rapporteur : Marc Schoenauer (INRIA)

Examineur : John Stewart (CNRS)

Examineur : Jacques Tisseau (École Nationale d'Ingénieur de Brest)

Examineur : Bruno Gaume (CNRS)

Examineur : Hervé Luga (Université de Toulouse)

Pour Lingchih

Résumé

Comment fabriquer des créatures *virtuelles* ou *réelles* automatiquement ? L'ingénierie n'a pas de réponse à cela. Pour concevoir un robot, les ingénieurs font appel à leur créativité ou à l'expérience accumulée au fil des siècles. En opposition à cela, la nature peut créer des systèmes bien plus complexes en s'appuyant sur l'évolution et le hasard. Dans cette thèse, nous présentons les principales recherches en vie artificielle et en réalité virtuelle qui s'inspirent de l'évolution naturelle pour générer des créatures virtuelles ou des robots autonomes. Les outils issus du domaine de la vie artificielle permettent de faire évoluer conjointement la morphologie et le comportement de créatures virtuelles dans un monde virtuel 3D afin qu'elle soient adaptées à leur environnement. Dans la lignée de ces travaux, nous proposons d'utiliser un nouveau type de contrôleur, inspiré des systèmes de classifieurs, et validons notre approche en faisant évoluer des créatures plongées dans des environnements complexes.

Pour évaluer les résultats produits par nos simulations nous introduisons ensuite une nouvelle méthode, basée sur les propriétés des *small-worlds*, qui permet de mesurer la complexité et l'auto-organisation d'un environnement virtuel. Pour conclure, nous présentons les perspectives de telles approches dans les domaines des créatures virtuelles ou de la robotique autonome.

Les objectifs de la réalisation de telles créatures peuvent répondre à trois besoins :

- La compréhension des mécanismes adoptés par le vivant pour concevoir des entités adaptées à un environnement.
- L'automatisation de l'animation d'entités virtuelles réalistes, évoluant dans des environnements dynamiques et complexes tels que les jeux vidéos.
- La conception de machines, en l'occurrence des robots, à partir de créatures virtuelles générées par évolution. L'objectif étant d'avoir une machine qui puisse concevoir une autre machine.

Table des matières

Résumé	v
1 Introduction	1
1.1 Les motivations	1
1.1.1 La génération de créatures artificielles autonomes	1
1.1.2 La conception automatique de créatures artificielles	2
1.2 Les problèmes de la morphologie et du comportement des créatures	3
1.2.1 La modélisation et la réalisation des créatures artificielles	3
1.2.2 La modélisation du contrôleur	4
1.2.3 La gestion de l'évolution conjointe	5
1.2.4 L'adaptation à un environnement dynamique et complexe	5
1.3 Problématique : Génération et adaptation des créatures	5
1.3.1 Objectifs	6
1.3.2 L'approche écosystème	6
1.4 Propriétés des créatures artificielles virtuelles ou réelles	6
1.4.1 Deux types de créatures artificielles	7
1.4.2 Définition des créatures artificielles	8
1.5 Plan de la thèse	8
2 L'approche évolutionniste	11
2.1 Les mécanismes de la vie artificielle et ses moyens de sélection	11
2.1.1 La vie artificielle	11
2.1.2 Les algorithmes génétiques et leur utilisation	13
2.1.3 La sélection naturelle	15
2.2 Modèle d'entités virtuelles	16
2.2.1 Les premières utilisations de la vie artificielle	18
2.3 Travaux sur la morphologie figée	19
2.3.1 Évolution de la morphologie seule	19
2.3.2 Évolution de contrôleur pour une morphologie fixée	19
2.3.3 Les inconvénient d'une morphologie figée	21

2.4	Représentation de la morphologie	22
2.5	Les différents types de représentation de la morphologie	22
2.5.1	Représentation par arbre	23
2.5.2	Représentation par développement	23
2.6	Conclusion	26
3	Les créatures artificielles évolutionnistes et les écosystèmes	27
3.1	Travaux sur les créatures artificielles évolutionnistes	27
3.1.1	Les créatures de Karl Sims	27
3.1.2	Les créatures de Tim Taylor et Colm Massey	32
3.1.3	Les créatures de Thomas Ray	33
3.1.4	Les créatures de Gregory Hornby	35
3.1.5	Les créatures de Hod Lipson	35
3.1.6	Les créatures de Chaumont	39
3.1.7	Les créatures de Miconi	42
3.1.8	Les créatures de Shim	42
3.1.9	les créatures de Komosinski pour le projet Framsticks	45
3.2	Les écosystèmes	46
3.2.1	Tierra	46
3.2.2	Gene Pool	47
3.2.3	Life Drop	48
3.2.4	Les autres écosystème artificiels	49
3.2.5	Embryogenèse et ontogenèse artificielle	50
3.3	Synthèse	51
3.3.1	Morphologie et contrôleur des créatures 3D	53
3.3.2	L'environnement	54
3.3.3	Bilan	54
4	Modèle proposé	57
4.1	L'environnement	57
4.1.1	Environnement proposé	58
4.2	Le moteur physique	58
4.2.1	Le moteur physique proposé	59
4.2.2	L'évolution des moteurs physiques	59
4.2.3	L'implémentation	60
4.3	La morphologie	61

4.4	Morphologie proposée	61
4.4.1	Le génotype : Graftal	62
4.4.2	Génotype et phénotype	62
4.5	Le contrôleur	63
4.5.1	Notre modèle de contrôleur : la composition de <i>patterns</i>	64
4.5.2	L'assemblage de <i>patterns</i>	65
4.6	Application des <i>patterns</i> pour l'animation	65
4.7	Les opérateurs génétiques	67
4.7.1	La mutation	67
4.7.2	Croisement	68
4.8	Les problèmes de stabilité	69
5	Expériences et Résultats	71
5.1	Les conditions initiales	71
5.2	La validations de expérimentations	72
5.3	Le déplacement sur un sol plat	72
5.3.1	Présentation de l'expérience	72
5.3.2	Quelques créatures issus de l'évolution	73
5.4	Déplacement dans une direction spécifique	74
5.4.1	Analyse des résultats	75
5.5	La montée d'un escalier	76
5.5.1	Présentation de l'expérience	76
5.5.2	La fonction de fitness	77
5.5.3	L'intérêt de l'expérience	77
5.5.4	Analyse des résultats	77
5.6	Le franchissement de tranchées	79
5.6.1	Présentation des expériences	79
5.6.2	Interêt de l'experience	79
5.6.3	Analyse des résultats	79
5.7	Le déplacement sur un terrain accidenté	80
5.7.1	Présentation de l'expérience	80
5.7.2	Analyse des résultats	80
5.8	Le déplacement d'un mur en coopération	81
5.8.1	Présentation de l'expérience	81
5.8.2	Analyse des résultats	81
5.9	Le déplacement avec un objet mobile	82

5.9.1	Présentation de l'expérience	82
5.9.2	Analyse des résultats	83
5.10	Discussion du modèle	83
5.10.1	Apport de notre modèle	83
5.10.2	L'utilisation des patterns	84
5.10.3	Morphologie	85
5.11	Comparaison avec les autres modèles	86
5.11.1	Comparaison des résultats	86
5.11.2	Comparaison du contrôleur	87
5.12	L'apport du modèle sur les trois principaux objectifs	90
5.12.1	Compréhension du vivant pour concevoir des créatures artificielles	90
5.12.2	L'animation d'entités virtuelles réalistes	91
5.12.3	La conception d'une machine par une autre machine	91
6	Complexité	93
6.1	Résumé	93
6.2	Introduction	94
6.3	Pourquoi mesurer la complexité ?	95
6.4	Small-worlds	96
6.5	Simulation de fourmis artificielles	98
6.6	Expériences	100
6.6.1	Variation de la taille de la population	101
6.6.2	Variation de la quantité de nourriture	102
6.6.3	Répartition aléatoire de la nourriture	103
6.6.4	Déplacement	103
6.6.5	Mémoire	105
6.7	Synthèse	106
6.8	Exemples d'applications	106
6.9	Conclusions et Perspectives	108
7	Conclusion et perspectives	109
7.1	Conclusion générale	109
7.2	Perspectives de notre modèle	110
7.2.1	Contrôleur de <i>patterns</i>	110
7.2.2	Environnement plus complexe	110
7.2.3	Calcul de l'énergie dépensée	110

7.2.4	Embryogenèse	111
7.3	Perspectives générales	111
7.3.1	Limitation des représentations morphologiques	111
7.3.2	Robotique évolutionnistes	112
7.3.3	Parallélisation	112
7.3.4	La matérialisation	112
A	Représentation des Objets de l'environnement	115
A.1	Représentation du StakeBoard en langage Steve	115
A	Paramètres pour l'évolution des créatures artificielles	119
A.1	Paramètres principaux	119
A.2	Génération des patterns	120
A	Informations complémentaires sur les smallworlds	123
A.1	Paramètres principaux : main.h	123
A.2	Algorithme de déplacement des fourmis : ants.cpp	124
B	Proposition d'un modèle d'écosystème 3D	127
B.1	Écosystème	127
B.1.1	Introduction	127
B.1.2	La chaîne alimentaire	128
B.1.3	La reproduction des espèces	129
B.1.4	Le concept de poison	129
B.1.5	L'énergie	131
B.1.6	Conclusion	131
	Bibliographie	133

Liste des Figures

1.1	Vue d'ensemble des travaux sur les créatures artificielles	9
2.1	Travaux de Lipson et Bongard	20
2.2	Évolution d'un contrôleur pour un robot bipède simulé	21
2.3	Contraintes imposées par la connexion de segments	23
2.4	L'approche <i>top-down</i> pour la construction de structures	24
2.5	L'approche <i>bottom-up</i> pour la construction de structures	25
3.1	Créatures de Karl Sims	29
3.2	Les deux méthodes de croisements des <i>graftals</i> proposées par Karl Sims.	30
3.3	Créatures de Karl Sim	31
3.4	Les créatures de Tim Taylor et Colm Massey	34
3.5	Les créatures de Ray	36
3.6	Représentation des créatures d'Hornby par L-system	37
3.7	Les créatures de Hornby	38
3.8	Les créatures artificielles de Lipson dans l'environnement simulé.	40
3.9	Projet Golem de Lipson : robots	40
3.10	Projet Golem de Lipson : évolution	41
3.11	Les catapultes de Chaumont	43
3.12	Les créatures de Miconi inspirées de celles de Karl Sims.	44
3.13	Les créatures de Shim évoluent pour adapter les ailes au vol.	44
3.14	Ensemble des ailes générées par évolution.	44
3.15	Créature de Framsticks	45
3.16	Tierra, l'écosystème réalisé par Thomas Ray	47
3.17	Gene Pool, l'écosystème réalisé par Jeffrey Ventrella	49
3.18	Life Drop : Évolution d'un écosystème artificiel dans une goutte d'eau	50
3.19	Travaux de René Doursat sur l'embryogénèse	51
3.20	Travaux d'Yves Duthen et Arturo Chavoya	51
3.21	Les créatures de Bongare qui s'inspire de l'ontogenèse artificielle.	52

4.1	Architecture de notre application	60
4.2	Morphologie des créatures : génotype et phénotype	63
4.3	Signal composé	64
4.4	Classifieur	66
4.5	Décomposition du mouvement d'une créature	67
4.6	Mutations opérées sur une créature	68
5.1	Créatures se déplaçant sur un sol plat.	74
5.2	Créatures roulant sur un sol plat.	74
5.3	Créatures se déplaçant sur un sol plat.	75
5.4	Créatures devant aller dans une direction spécifique	75
5.5	Moyenne de nombre de blocs par modules	76
5.6	Créatures montant des escaliers	78
5.7	Créatures montant un escalier rapidement.	78
5.8	Créatures franchissant des tranchées avec l'aide de ses pattes.	80
5.9	Créatures se déplaçant sur un terrain accidenté.	81
5.10	Créatures poussant un mur.	82
5.11	Créature se déplaçant avec le skateboard.	84
6.1	Propriétés des small-worlds	98
6.2	Génération de small-world	100
6.3	Évolution dans le temps de la construction d'un graphe	101
6.4	Évolution du taux de clustering et de la densité	103
7.1	Perspectives	113
B.1	Une vue de notre écosystème.	128
B.2	Exemple de chaîne alimentaire.	130

Liste des Tables

3.1	Travaux sur les créatures artificielles 3D	55
3.2	Travaux sur les créatures artificielles	56
5.1	Nombre moyen de blocs par expérimentation	86
5.2	Travaux sur les créatures artificielles	88
5.3	Travaux sur les créatures artificielles	89
5.4	Points forts des différents travaux sur les créatures artificielles	90
6.1	Comparaison entre un graphe aléatoire et un <i>small-world</i>	99
6.2	Interprétation et signification des propriétés des <i>small-worlds</i>	99
6.3	Variation de la taille de la population de fourmis artificielles.	102
6.4	Variation de la quantité de nourriture	104
6.5	Répartition aléatoire de la nourriture	104
6.6	Le déplacement des fourmis est non Brownien	105
6.7	Fourmis utilisant de la mémoire.	106

Introduction

1.1 Les motivations

1.1.1 La génération de créatures artificielles autonomes

Modéliser des comportements de créatures virtuelles est un problème passionnant. Il est facile de concevoir que plus on désire obtenir un comportement complexe plus il pourra être difficile de le modéliser. Les premières approches dans la génération de comportements consistent à les programmer à l'aide de scripts créés par un programmeur. Cette méthode a l'avantage de permettre un parfait contrôle du comportement généré. Car on pourra établir pour des situations précises et connues, le comportement que doit adopter une créature virtuelle. Dès lors, il est facile de comprendre que la complexité des scripts grandira avec le nombre de situations à décrire. On se retrouve alors dépendant des capacités de calcul et de stockage des machines utilisant ces scripts. Un autre inconvénient est qu'il est souvent difficile, fastidieux, et en fait souvent impossible d'envisager toutes les situations auxquelles une créature sera confrontée. En effet, les créatures faisant partie elles-mêmes de l'environnement, elles le modifient par leur présence et leurs interactions. Il devient alors nécessaire de leurs donner plus d'autonomie afin de pouvoir généraliser les situations auxquelles elles seront confrontées.

Si l'on tente de rendre une créature autonome, cette créature devra avoir une représentation globale ou partielle du monde. Une représentation globale du monde est possible lorsqu'il s'agit de créatures virtuelles qui interagissent dans un environnement simulé. Cette représentation globale n'est plus possible dès lors que la créature est un vrai robot. Étant donné que nous voulons respecter cette contrainte dans cette thèse, nous ne nous intéresserons donc qu'aux créatures ayant une vision partielle du monde. Ce problème soulève la question de la représentation des perceptions. Nous donnerons dans le prochain chapitre notre définition des créatures

virtuelles ainsi que de leur perception du monde.

Nous pouvons dès lors établir que le comportement d'une créature n'a de sens que s'il permet à la créature d'agir sur (et dans) un environnement par le biais d'un corps. La question se pose alors de la représentation virtuelle ou physique des créatures et de leurs interactions dans et avec un environnement. En effet, il est difficile de modéliser une créature dont la forme et le comportement sont adaptés à un environnement inconnu. Ces problématiques se retrouvent en robotique car les capacités physiques des robots limitent leur interactions avec l'environnement. La morphologie et le contrôleur d'un robot sont donc primordiaux pour évoluer dans un environnement en constant changement. On peut en conclure que l'autonomie d'une créature dépend donc de sa morphologie et de son comportement. Il se pose la question de savoir quels mécanismes peuvent permettre l'automatisation de la conception de telles créatures virtuelles et réelles.

1.1.2 La conception automatique de créatures artificielles

Un des paradigmes de l'informatique et de la robotique est d'envisager qu'une machine puisse concevoir une autre machine [Lipson 2007]. Dans le cadre de la génération de créatures *virtuelles* ou *réelles* (physiques), il est intéressant de rechercher des solutions pour concevoir conjointement leur morphologie et leur contrôleur. Durant les deux derniers siècles, nous avons fait beaucoup de progrès dans la compréhension des phénomènes physiques comme la thermodynamique, la mécanique des fluides et l'électromagnétisme. Mais la production automatique en ingénierie est beaucoup plus lente bien que nous ayons les moyens de modéliser et simuler les comportements de produits avant qu'ils soient réalisés. Robert Willis en 1870 est certainement le premier à énoncer la possibilité de synthétiser des machines [Willis 1870] pouvant se mouvoir. À cette époque, il précisait qu'il était impossible de réaliser une machine de manière rationnelle et que seules l'intuition et l'expérience pouvaient être utilisées pour la réalisation. Pour la production de système complexe, deux approches sont en opposition. La première approche est celle utilisée par l'ingénierie qui utilise l'intelligence, la créativité et l'expérience [Schnaars 1989]. La seconde utilise l'émergence au travers d'une succession d'adaptations, c'est l'approche évolutionniste [Ziman 2003]. Dans le cadre de cette thèse, nous nous intéressons aux créatures virtuelles autonomes dont la morphologie et le comportement doivent être adaptés à leur environnement. Comment assembler automatiquement un ensemble de pièces dont on connaît les propriétés pour en faire une créature

ou un robot ? Il existe un nombre conséquent de combinaisons qu'aucune méthode déterministe ne peut résoudre en un temps raisonnable.

1.2 Les problèmes de la morphologie et du comportement des créatures artificielles

La simulation comportementale a pour but d'animer des agents en leur donnant un comportement. Bien souvent il n'est pas nécessaire de décrire la morphologie de l'agent ou bien celle-ci sera très implicite. Dès lors que l'on veut produire un agent dont le comportement est très complexe, il est important que l'agent puisse percevoir l'environnement dans lequel il évolue. Ces perceptions sont liées à sa morphologie. L'intelligence d'un agent n'a de sens que si elle lui permet de s'adapter à son environnement. L'environnement étant perçu par le corps de l'agent, il est primordial qu'il évolue en même temps. C'est l'approche qu'adoptent Jeff Bongard et Rolf Pfeifer dans leur livre "How the Body Shapes the Way We Think: A New View of Intelligence" [Bongard and Pfeifer 2006]. Dans cet esprit, nous nous intéressons au cours de cette thèse à l'évolution conjointe de la morphologie et du contrôleur afin de doter les créatures générées de capacités d'adaptation à leur environnement.

1.2.1 La modélisation et la réalisation des créatures artificielles

Il y a deux façons principales d'aborder le problème de la modélisation et de la réalisation de créatures, l'approche *bottom-up* et l'approche *top-down*. Durant les cinquante dernières années, l'intelligence artificielle a tenté avec beaucoup de difficultés et de désillusions de reproduire l'intelligence humaine en partant de concepts de haut niveau. C'est-à-dire en employant l'approche *top-down*. L'approche *top-down* traite uniquement les problèmes de manière déterministe avec une approche réductionniste et non dans leur ensemble. Cela rend l'approche *top-down* dépendante de la complexité combinatoire des problèmes. De plus l'intelligence artificielle tente de simuler des concepts de haut niveau difficiles à maîtriser et, malgré d'importants résultats ils sont le plus souvent limités par la quantité de données et les calculs nécessaires à leurs traitements. Au contraire, l'approche *bottom-up* modélise des concepts de bas niveau que l'on met en interaction pour faire émerger un système plus complexe.

Un bon exemple pour illustrer ce propos est la résolution du jeu d'échecs par des méthodes d'Intelligence Artificielle. Les échecs sont traités comme étant un problème combinatoire par l'approche *top-down*. Cette approche qui utilise la force brute ne per-

met pas de comprendre les mécanismes mis en place par l'homme pour y jouer¹. C'est ce qui limite l'extension de ce mode de résolution à des problèmes plus complexes tels que le jeu de go dont la combinatoire explose. De plus, toujours pour les échecs, leur résolution est traitée en trois parties distinctes l'ouverture, le milieu de partie et les finales. Le problème n'est pas traité dans son ensemble. Partant de ce constat, en 1996, Rodney Brooks propose huit challenges pour l'intelligence artificielle dont celui de concevoir un programme simulant un joueur d'échecs [Brooks et al. 1996]. Il en conclut que la résolution de ce problème permettrait certainement de mieux aborder la résolution du jeu de go.

Rodney Brooks partant de ce constat propose d'employer l'approche *bottom-up* pour la réalisation de créatures artificielles [Brooks 1991b]. En effet, au cours de notre évolution, la nature a passé plus de temps à concevoir les cellules et les êtres multicellulaires que l'intelligence. Il est donc primordial de comprendre et maîtriser les mécanismes qui permettront de générer des créatures plutôt que de se focaliser sur leur création. L'important n'étant pas les résultats que l'on produit mais les mécanismes que l'on met en place pour les produire.

En effet nous concevons la modélisation et la réalisation de créatures artificielles comme étant un assemblage de parties élémentaires et dont le tout dans son interaction fait émerger une créature. La question qui se pose est de savoir comment assembler ces parties. En s'inspirant des mécanismes de la nature, l'approche évolutionniste apporte une solution. En effet, elle permet grâce à la sélection naturelle de produire des créatures en utilisant des mécanismes de développement comme les *L-systems*, l'embryogenèse ou les graphes de développement que nous verrons dans cette thèse.

1.2.2 La modélisation du contrôleur

Comme nous l'avons vu précédemment, le contrôleur d'une créature est lié à sa morphologie et à ses capteurs qui lui permettent de percevoir le monde. Si nous voulons que le contrôleur émerge d'un assemblage d'éléments et non de la conception d'une intelligence et d'une expérience humaine, nous pouvons nous inspirer des mécanismes de la nature. Cette approche peut s'appliquer aux réseaux de neurones, en effet, l'assemblage de neurones permet de constituer une topologie dont un comportement pourra émerger. Les réseaux de neurones sont largement répandus dans l'ensemble des travaux traitant des créatures artificielles. Dans cette thèse, nous

¹Nous avons effectué quelques travaux dans ce domaine [Lassabe et al. 2006] mais qui ne sont pas intégrés dans ce document de thèse.

proposons d'utiliser des *patterns* assemblés au cours de l'évolution et actionnés par un mécanisme inspiré des systèmes de classeurs. En effet nous pensons que des contrôleurs qui composent des courbes pour produire un signal continu peuvent générer des comportements d'adaptation efficaces et réalistes.

1.2.3 La gestion de l'évolution conjointe

Nous verrons dans l'état de l'art, l'influence que peut avoir l'évolution d'un contrôleur sur une morphologie figée. En effet, rien ne garantit qu'il existe un contrôleur efficace pour une morphologie préétablie. Il semble donc intéressant que la morphologie et le contrôleur évoluent conjointement. Des éléments sont en évolution conjointe si leur évolution est interdépendante. Dans le cadre des créatures, cela signifie que la morphologie d'une créature évolue en même temps que son contrôleur en formant un tout. Nous parlerons aussi de coévolution lorsque plusieurs créatures interagissent ensemble et font émerger des stratégies d'adaptation.

1.2.4 L'adaptation à un environnement dynamique et complexe

Nous montrerons dans la synthèse en fin d'état de l'art qu'il peut être intéressant de confronter les créatures virtuelles générées à des environnements dynamiques et complexes. Une des raisons est que notre environnement réel est complexe. Si nous voulons concevoir des robots autonomes, ils doivent donc être adaptés à de tels environnements. En outre, il a été montré qu'un environnement complexe avait une influence positive sur la complexité des créatures qui en émergent [Seth 1998].

1.3 Problématique : Génération et adaptation des créatures dans un environnement complexe

Notre problématique est donc de concevoir des créatures virtuelles adaptées à un environnement complexe à partir de mécanismes d'évolution. Dans cette thèse, nous employons pour cela des algorithmes génétiques pour réaliser l'évolution de créatures artificielles. Nous proposons aussi, en perspective l'utilisation de la sélection naturelle pour un modèle d'écosystème.

1.3.1 Objectifs

Les objectifs de la réalisation de telles créatures peuvent répondre à trois 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 dynamiques et complexes comme les jeux vidéos. Le troisième objectif de ce type d'approche est de pouvoir concevoir des machines, en l'occurrence des robots, à partir de créatures virtuelles générées par évolution. L'objectif étant d'avoir une machine qui puisse concevoir une autre machine.

1.3.2 L'approche écosystème

Bien souvent, il est nécessaire pour résoudre un problème d'avoir une coopération entre plusieurs robots ou créatures. Comment générer un comportement complexe, c'est-à-dire qui émerge de l'interaction de plusieurs créatures ? La modélisation d'écosystèmes composés de créatures soumises à l'évolution est peut-être une solution envisageable. L'écosystème peut en effet être vu comme un système autorégulé. Ainsi les créatures émergent de l'écosystème et l'écosystème émerge de l'interaction des créatures qui le compose. Il est donc intéressant de les étudier afin d'en comprendre les mécanismes, mais nous verrons qu'il est difficile d'en mesurer la complexité. Dans cette thèse, nous proposons une approche basée sur l'utilisation des propriétés des *small-world* afin d'en mesurer la complexité.

1.4 Propriétés des créatures artificielles virtuelles ou réelles

L'homme a toujours voulu comprendre le vivant. Il a commencé par l'observer pour l'étudier, cela a donné naissance aux sciences de l'observation comme la botanique, la biologie, la paléontologie. Puis l'homme a voulu maîtriser le vivant pour se prémunir des maladies, il tente de le modifier pour le contrôler, les sciences issues de ces interventions sont entre autres la médecine et la génétique. De nos jours, l'homme s'inspire du vivant pour le modéliser, le simuler et le reproduire. Ainsi le domaine de la vie artificielle repose sur des mécanismes et des caractéristiques du vivant en vue de produire des créatures artificielles.

En nous référant à la définition de la vie artificielle de J. Doyne Farmer et Alleta Belin [Farmer and Belin 1989], nous proposons une définition des créatures artificielles. Les propriétés minimales d'une créature artificielle sont :

-
1. L'être humain a contribué au processus d'apparition de toute créature artificielle.
 2. Une créature artificielle est une unité.
 3. Une créature artificielle possède une morphologie²
 4. Une créature artificielle doit interagir avec son environnement.
 5. Une créature artificielle est autonome.

Les propriétés suivantes ne sont pas indispensables mais souhaitables :

6. Une créature artificielle s'auto-produit.
7. Une créature artificielle peut se reproduire elle-même.
8. Une créature artificielle s'auto-répare.
9. Une créature artificielle possède des mécanismes d'adaptation.

1.4.1 Deux types de créatures artificielles

Il est dès lors important de distinguer deux types de créatures. Les créatures issues de mécanismes d'évolution comme les algorithmes génétiques et les créatures ayant les caractéristiques du vivant comme la capacité à se reproduire (Fig. 1.1). Des créatures artificielles peuvent aussi posséder ces deux caractéristiques, c'est à dire être issues d'un processus évolutionniste et posséder les caractéristiques du vivant. On peut alors se poser la question s'il est raisonnable de vouloir produire des créatures artificielles ayant les caractéristiques du vivant sans utiliser des mécanismes d'évolution. En effet, les êtres vivants sont issus de l'évolution et il est certainement difficile d'obtenir les mêmes résultats sans ce procédé.

À la frontière de ces approches (Fig. 1.1), nous situons les écosystèmes artificiels car nous pensons qu'il n'est pas possible de les dissocier des créatures artificielles auxquelles ils apportent les ressources nécessaires à leur création et leur survie. Dans le cadre de cette thèse, nous traitons des créatures artificielles issues de mécanismes d'évolution. C'est une première étape qui permet par la suite de s'intéresser à l'intégration des caractéristiques du vivant. En effet, à partir de la mise en place des

²La morphologie correspond aux éléments extérieurs. Ces éléments sont généralement visibles et en interaction avec l'environnement. Plusieurs travaux ont été consacrés à la génération de morphologies. Dans cette thèse, nous nous intéressons à la morphologie issue d'une morphogenèse.

mécanismes d'évolution, nous montrerons en perspective qu'il est aisé d'intégrer des caractéristiques du vivant.

1.4.2 Définition des créatures artificielles

Nous définissons une créature artificielle évolutionniste comme étant une entité virtuelle ou réelle possédant une morphologie et un comportement issus de processus évolutionnistes.

À partir de cette définition et des propriétés que l'on a décrit dans la section précédente, on peut en déduire que les robots sont des créatures artificielles³.

Une créature artificielle virtuelle est une créature simulée qui a une représentation 2D/3D dans un environnement virtuel.

1.5 Plan de la thèse

Dans le prochain chapitre (Chap. 2), nous présentons l'approche évolutionniste et différentes représentations des créatures artificielles. Le chapitre suivant (Chap. 3) présente l'état de l'art sur la génération de créatures artificielles par des méthodes évolutionnistes. Nous distinguons deux catégories de travaux, ceux dont l'évolution est dirigée par une fonction d'évaluation et ceux dont l'évolution utilise une sélection naturelle inspirée du vivant⁴. La première catégorie de travaux se focalise sur l'évolution des créatures, la seconde sur l'évolution d'un écosystème dans le but que des stratégies émergent de l'interaction entre les créatures. Une synthèse des travaux de l'état de l'art, nous permettra d'établir un modèle de créatures artificielles ainsi qu'un ensemble d'environnements permettant leur évaluation (Chap. 4). Ce bilan nous permettra de remarquer qu'il est difficile d'évaluer la complexité d'un écosystème. Après avoir présenté les résultats de nos expérimentations sur notre modèle de créatures, nous discuterons de l'ensemble des résultats obtenus pour les comparer à ceux présentés dans l'état de l'art (Chap. 5). Puis nous présenterons une méthode basée sur les propriétés des *small-worlds* permettant d'évaluer "l'évolution de la complexité" d'un environnement. Nous appliquerons cette méthode à un environnement composé de fourmis artificielles (Chap. 6). Pour finir nous concluons et donnerons quelques perspectives pour obtenir des créatures artificielles plus complexes et ainsi que les applications en robotique (Chap. 7). En annexe, nous présentons

³En effet les robots sont une unité car ils peuvent avoir une indépendance énergétique par le biais d'une batterie. Ils interagissent physiquement avec l'environnement, ils peuvent être autonomes.

⁴Par exemple une modélisation de l'énergie.

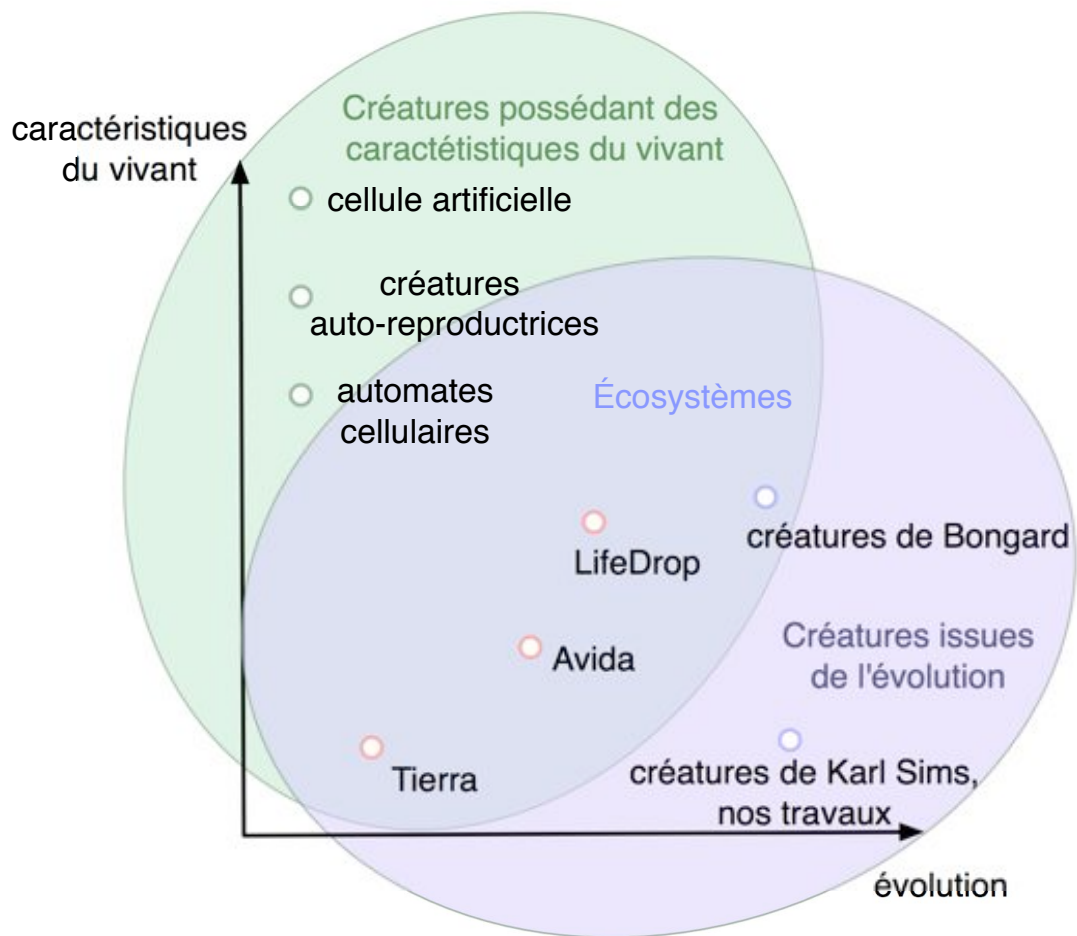


Figure 1.1: Vue d'ensemble des travaux de la vie artificielle concernant les créatures artificielles. Dans cette thèse, nous nous situons dans la catégorie des travaux portant sur les créatures issues des mécanismes d'évolution au même niveau que celles de Karl Sims.

un modèle d'écosystème ainsi que les environnements que nous avons utilisé pour obtenir nos résultats afin que d'autres personnes puissent les reproduire.

L'approche évolutionniste

Dans ce chapitre, nous donnons une définition de la vie artificielle et une brève description des algorithmes génétiques et de la sélection naturelle dans les écosystèmes artificiels.

2.1 Les mécanismes de la vie artificielle et ses moyens de sélection

La vie artificielle s'inspire du vivant pour élaborer des algorithmes adaptatifs similaires aux mécanismes d'adaptation et d'évolution trouvés dans la nature. Contrairement aux programmes classiques qui décrivent la démarche d'un algorithme en vue d'obtenir des solutions à un problème, les algorithmes évolutionnistes utilisés en vie artificielle, comme les algorithmes génétiques, la programmation génétique, les stratégies d'évolutions, se contentent de décrire le type de solutions recherchées, puis sélectionnent les solutions les plus adaptées au problème en vue de les reproduire à l'aide d'opérateurs de croisement et de mutations [Holland 1975; Goldberg 1989].

2.1.1 La vie artificielle

Il est possible de trouver deux principales définitions de la vie artificielle. Selon Langton [Langton 1989a], *"la vie artificielle a été définie comme étant l'étude de systèmes construits par l'homme qui présentent des comportements caractéristiques des systèmes vivants naturels."*

J. Doyne Farmer et Alleta d'A. Belin [Farmer and Belin 1989] proposent une autre définition sous forme de propriétés caractérisant un système de vie artificielle dont voici la liste :

1. L'être humain a contribué au processus d'apparition de tout système de vie artificielle.
2. Un système de vie artificielle est autonome.
3. Un système de vie artificielle est en interaction avec son environnement.
4. Il y a émergence de comportements dans un système de vie artificielle.

Les trois propriétés suivantes ne sont pas indispensables mais restent néanmoins très présentes.

5. Un système de vie artificielle peut se reproduire lui-même.
6. Un système de vie artificielle possède une capacité d'adaptation.
7. Un système de vie artificielle n'est pas une unité. À l'opposé de la vie, un système de vie artificielle peut être réparti en plusieurs endroits : par exemple, un robot et un ordinateur peuvent effectuer les calculs reliés par ondes.

En ce qui concerne la génération de créatures artificielles virtuelles, ces propriétés tendent à être toutes respectées sans pour autant être toutes simulées. Un exemple est la reproduction des créatures qui s'effectue le plus souvent sans se soucier de l'apport des ressources permettant de constituer la créature. Ces ressources généralement sont considérées comme étant en nombre illimité et pouvant s'assembler sans contraintes. La difficulté s'accroît lorsque l'on veut que ces robots puissent se *reproduire* (se dupliquer) physiquement, car les éléments subissent des contraintes physiques qui ne sont pas toujours simulées ou respectées dans les environnements virtuels. En effet, les créatures artificielles virtuelles évoluent dans un environnement qui ne peut pas simuler tous détails du monde réel. La reproduction physique des robots est donc une contrainte à difficile à surmonter. Malgré cela quelques travaux comme ceux de Victor Zykov, Efsthios Mytilinaios, Bryant Adams et Hod Lipson arrivent à obtenir les prémices d'une auto-reproduction [Zykov et al. 2005]. Effectivement leur robot constitué de cubes peut, si on lui fournit d'autres cubes, reconstituer son double.

Suivant les points de vue, on peut considérer un virus, une bactérie ou un cristal comme étant vivants. Si bien que les définitions précédentes peuvent paraître incomplètes selon les caractéristiques du vivant que l'on prend en compte. Pour éviter une définition de la vie (vie artificielle) sous forme de propriétés, il est possible de décrire la vie comme étant un processus qui s'auto entretient. La définition suivante a été donnée par Humberto Maturana et Francisco Varela [Maturana and Varela 1980]:

"Il s'agit de concevoir un être vivant comme une organisation autopoïétique. Cela caractérise le fait qu'un être vivant est un réseau de transformations dynamiques, fabricant ses propres composants (métabolisme) et qui construit une barrière topologique (membrane) qui, à son tour, est la condition nécessaire du fonctionnement en tant qu'unité du réseau de transformations qui l'a engendré. Ce qui signifie que les êtres vivants sont continuellement en train de s'auto-produire." On peut alors se poser la question s'il est nécessaire qu'un système adaptatif et auto-reproducteur (créatures artificielles ou robots) ait à s'auto-produire continuellement pour perdurer dans le temps et au fil des générations.

L'émergence et l'auto-organisation sont des mécanismes utilisés par la vie et la vie artificielle qu'il est important de définir. Nous utiliserons ces notions dans l'étude de la complexité des écosystèmes (Chap : 6).

Plusieurs définitions de l'émergence existent selon le contexte dans lequel nous plaçons. Dans le contexte de la vie artificielle, la définition de l'émergence, est la suivante [Kauffman 1993] : *"propriétés globales d'un système que l'on ne saurait prédire par la seule connaissance des règles qui gouvernent les éléments qui le composent."*

Une autre définition plus précise basée sur l'organisation [von Bertalanffy 1973] des constituants qui animent la vie est la suivante : *"l'organisation est un processus par lequel de la matière, de l'énergie et de l'information s'assemblent et forment une totalité ou une structure. Certaines totalités développent une forme d'autonomie ; elles s'organisent de l'intérieur ; on parle alors d'auto-organisation."*

Il existe deux sortes d'organisation : l'organisation en modules ou en sous-systèmes (qui renvoie aussi à l'organisation en réseaux) et l'organisation en niveaux hiérarchiques. L'organisation en sous-systèmes procède par intégration de systèmes déjà existants, tandis que l'organisation en niveaux hiérarchiques produit de nouvelles propriétés à chaque niveau supplémentaire. La notion d'organisation hiérarchiques est liée à l'émergence, dans la mesure où c'est le degré d'organisation qui fait passer d'un niveau hiérarchique à un autre et fait émerger de nouvelles propriétés. Dans ce contexte, l'émergence peut être définie comme étant le processus de création d'un niveau hiérarchique supérieur.

2.1.2 Les algorithmes génétiques et leur utilisation

Les algorithmes génétiques sont basés sur la théorie de l'évolution de Charles Darwin [Darwin 1859] qui considère que les êtres les plus adaptés à leur environnement survivent et propagent leurs caractéristiques. John Holland adapta cette théorie pour l'appliquer à l'information [Holland 1975]. Ainsi les algorithmes génétique permet-

tent d'obtenir une solution approchée à un problème d'optimisation. Ils appartiennent à la famille des algorithmes évolutionnistes comme les stratégies d'évolutions inventées par Rechenberg dans les années 60 [Rechenberg 1965], la programmation évolutionnaire inventée par Fogel, Owens et Walsh en 1965 [Fogel et al. 1966] et la programmation génétique démocratisée par Koza en 1992 [Koza 1992]. Les algorithmes évolutionnistes font eux partie de la famille des métaheuristiques comme le recuit simulé, les algorithmes de colonies de fourmis ou les algorithmes par essaims particuliers. Les algorithmes génétiques ont été popularisés à partir de 1989 notamment par des ouvrages tel que le livre de Goldberg [Goldberg 1989]. Ils sont une des solutions employées, lorsque les algorithmes déterministes sont inefficaces ou inadaptés pour essayer de résoudre des problèmes dont l'espace de recherche est trop grand.

Dans les algorithmes génétiques, l'information à optimiser est représentée sous forme de chromosome (appelé également individu-solution). Ce terme emprunté à la biologie désigne généralement une structure de données. Au début, on initialise, généralement aléatoirement, une population de chromosomes. Les individus sont ensuite évalués dans leur environnement par une fonction d'évaluation aussi appelée fonction de *fitness*. Cette fonction d'évaluation traduit le but que les individus doivent atteindre. Une fois les individus évalués, ils sont sélectionnés en fonction de leur note en vue d'être croisés ou mutés. Les nouveaux individus générés ainsi que les anciens individus les plus adaptés forment la nouvelle génération. L'algorithme recommence avec la nouvelle population jusqu'à ce qu'il converge vers des individus qui satisfassent la fonction de fitness.

Les algorithmes génétiques ne garantissent pas de trouver les solutions optimales en un temps fini [Holland and Reitman 1978; Goldberg 1989]. Ils comportent de nombreux paramètres qui dépendent du problème à résoudre et demandent une certaine expertise. De plus il n'est pas toujours trivial de déterminer la bonne fonction d'évaluation qui produira les résultats désirés ¹. Malgré cela le succès des algorithmes génétiques est grandissant et ils sont de plus en plus employés pour des problèmes difficiles. Kim récence les articles publiés dans le domaine de la vie artificielle et constate une croissance constante du nombre de publications [Kim and Cho 2006]. Quand à Koza, il récence les problèmes pour lesquels la programmation génétique arrive à égaler l'homme dans sa tâche de conception de système complexe [Koza et al. 2003].

Dans le cadre de la création des créatures artificielles l'intérêt de l'étude de

¹En effet, une fonction d'évaluation mal définie peut facilement entraîner des effets de bord non désirés.

tels mécanismes apparaît évident : reproduire et comprendre les mécanismes de l'évolution en les simulant ; produire des créatures artificielles dans un environnement similaire au nôtre en vue de les reproduire sous forme de robots ou de peupler des mondes virtuels. La problématique est alors d'avoir une évolution qui converge vers des créatures satisfaisant un but recherché. Pour cela, la *fitness* permettant de guider l'évolution des créatures doit être définie afin d'éviter des niches écologiques non optimales. Le phénotype doit restreindre l'espace de recherche sans pour autant diminuer l'espace des solutions souhaitées. Le paramétrage de toutes ces caractéristiques est déterminant pour obtenir l'évolution souhaitée. Au final, c'est l'évolution guidée par la description du problème qui permet de faire émerger la solution.

Rodney A. Brooks, fut en 92, l'un des premiers à s'intéresser à l'utilisation de l'approche vie artificielle pour des robots réels [Brooks 1992]. À cette époque les comportements des robots étaient programmés à la main. À la même période, Langton organise les premières conférences sur la vie artificielle où les programmes évolutionnistes générant des contrôleurs d'entités virtuelles étaient présentés [Christopher G. Langton and Rasmussen 1992; Langton 1989b]. On doit par la suite, les premiers travaux donnant des résultats sur les créatures artificielles à Karl Sims [Sims 1994a; Sims 1994c], qui, sur une *Connection Machine*, est arrivé à générer par évolution des créatures adaptées à se déplacer dans un environnement virtuel. Ses travaux sont présentés dans les prochaines sections ainsi que des travaux plus récents comme ceux d'Hornby, Komosinski, Lipson et Taylor [Hornby and Pollack 2001; Komosinski 2000; Lipson and Pollack 2000; Taylor and Massey 2000].

2.1.3 La sélection naturelle

L'utilisation de la fonction d'évaluation n'est pas la seule solution lorsque l'on souhaite résoudre un problème par le biais de l'évolution. Il est aussi possible de simuler la sélection naturelle sans utiliser de fonction d'évaluation. L'écosystème artificiel peuplé de créatures est un parfait exemple de sélection naturelle. Chaque individu possède une quantité d'énergie et évolue dans l'environnement. Cette quantité d'énergie sert à la reproduction et aux déplacements. L'évolution n'est pas dirigée comme avec une fonction d'évaluation. L'évolution se fait par la mort des créatures et l'arrivée des nouvelles par le biais des naissances. Ainsi, les individus évoluent tous en même temps dans l'environnement ce qui peut nécessiter une grande capacité de calcul. Les avantages sont que l'évolution est laissée libre et peut

aboutir à des résultats intéressants tel que ceux obtenus par Ventrella avec Gene Pool [Ventrella 2005] que nous allons décrire dans les sections suivantes.

D'ailleurs en partant de ce principe, il est possible d'utiliser un nouveau type d'algorithmes génétiques qui au lieu d'attribuer une évaluation, attribut une quantité d'énergie en fonction de cette évaluation. Cette énergie sert ensuite pour la reproduction des individus. Un individu meurt, c'est à dire est éliminé de la population, quand il n'a plus d'énergie. La taille de la population devient ainsi variable. L'évolution se termine quand un individu satisfait la fonction d'évaluation ou quand la population n'a plus d'individus. S'il n'y a pas de convergence, les individus finissent par épuiser leur énergie en se reproduisant, ce qui entraîne une diminution de la taille de la population puis sa disparition.

2.2 Modèle d'entités virtuelles

L'environnement d'une créature artificielle peut être, soit simulé par un moteur physique [Klein 2003], soit utiliser l'espace réel dans le cadre de robots. Les deux composantes principales sont la morphologie de la créature et son contrôleur qui lui permet d'interagir avec son environnement par le biais de capteurs et d'effecteurs.

Récemment beaucoup de travaux sur les créatures artificielles ont été réalisés comme ceux de Bongard, Eggenberger, Hornby, Lipson, Miconi, O'Kellym, Shim, Sims, Taylor et Teo². Mise à part ceux de Komosinski [Komosinski 2000], ces travaux se sont souvent focalisés sur la morphologie des créatures ou le comportement d'une ou deux créatures mais n'ont pas utilisé d'environnement complexe. En contre partie, des simulations se sont focalisées sur les interactions de populations composées de créatures très simples. Nous voulons essayer de prendre en compte les deux approches et donc faire évoluer des créatures dans des environnements plus complexes.

Après une présentation des travaux de référence, nous présenterons les choix vers lequel nous nous sommes orientés.

Il existe plusieurs solutions pour représenter les morphologies et les contrôleurs des créatures artificielles. Les créatures de Karl Sims sont certainement les plus évoluées [Sims 1994a; Sims 1994c; Sims 1994b]. Ce dernier réalise une évolution qui comporte peu de contraintes et obtient ainsi une grande variété de créatures ayant des aptitudes pour se déplacer. Pour engendrer la morphologie de ces créatures, il utilise

²[Bongard and Paul 2000; Eggenberger 1997; Hornby and Pollack 2001; Lipson and Pollack 2000; Miconi and Channon 2006b; O'Kellym and Hsiao 2004; Shim et al. 2004; Sims 1994a; Sims 1994c; Taylor and Massey 2000; Teo and Abbass 2005]

des *graftals* que nous détaillerons dans les prochaines sections.

Par la suite beaucoup de travaux ont tenté de reproduire les résultats de Sims avec différentes approches pour la représentation de la morphologie et du contrôleur, tel que les *L-systems* [Lindenmayer 1968] pour les créatures d'Hornby [Hornby and Pollack 2001]. Son projet, tout comme le projet Golem de Lipson [Lipson and Pollack 2000], a pour objectif de réaliser des robots sans la moindre d'intervention humaine. Pour cela, il utilise des mécanismes d'évolution afin de trouver la morphologie et le contrôleur de robots. Les éléments sont ensuite produits par une imprimante 3D qui les imprime couche par couche à l'aide de collagène. L'homme n'intervient sur le processus de fabrication qu'à la fin où il assemble les moteurs aux structures imprimées.

Une autre approche pour obtenir des comportements complexes est d'optimiser des créatures 2D dans un écosystème artificiel [Ventrella 1998b; Ventrella 1998a]. Le mécanisme de sélection est basé sur la sélection naturelle. Chaque créature a une quantité d'énergie et essaie de survivre en recherchant de la nourriture dans l'environnement (*eating/mating*). Bien que les créatures soient moins complexes, cette approche permet de simuler beaucoup de créatures simultanément. Ainsi l'évolution des créatures dépend de l'environnement et de l'interaction des créatures entre elles. Il est ainsi possible d'étudier le comportement global des populations et leur dynamique. Sur des principes proches, Komosinski réalise un environnement 3D [Komosinski 2000]. Ces travaux se distinguent par le fait que l'évolution est soit dirigée par une fonction d'évaluation soit par sélection naturelle³. Récemment, EvolGL project [Garcia Carbajal et al. 2004] propose de simuler un écosystème virtuel dont les créatures sont des vers. Les vers peuvent évoluer en tant qu'herbivores, carnivores ou omnivores en développant différentes stratégies de survie.

Des résultats récents ont été obtenus en utilisant une fonction d'évaluation comme l'évolution de créatures combattantes d'O'Kellym [O'Kellym and Hsiao 2004] et l'évolution de créatures volantes de Shim [Shim et al. 2004]. Ce dernier fait évoluer les ailes et le contrôleur de créatures afin qu'elles puissent suivre un chemin.

Dans ses travaux, Miconi a reproduit les créatures de Karl Sims sans obtenir de meilleurs résultats, mais il a par contre amélioré les algorithmes de coévolution [Miconi and Channon 2005; Miconi and Channon 2006a; Miconi and Channon 2006b].

Pour conclure, Adam et Seth ont montré chacun dans leurs travaux l'influence

³Les individus meurent lorsqu'ils n'ont plus d'énergie.

de l'environnement sur la complexité [Adami et al. 2000; Seth and Edelman 2004; Seth 1998]. Ainsi des entités soumises à un environnement plus complexe deviennent elles aussi plus complexes. Partant de ce constat, une perspective serait d'utiliser des environnements plus complexes afin d'obtenir des créatures aux comportements adaptatifs plus complexe.

2.2.1 Les premières utilisations de la vie artificielle

En 1991, Langton suggère d'utiliser la programmation génétique afin de générer les comportements de base des robots [Langton 1989b; Christopher G. Langton and Rasmussen 1992]. La même année, des résultats prometteurs avaient été obtenus par Koza [Koza 1991] en appliquant la programmation génétique au langage *Lisp*. Cependant, l'évolution de programmes pour des robots nécessite de nombreuses évaluations qui prennent beaucoup de temps. Pour remédier à cela, il est possible d'automatiser les évaluations en les simulant ce qui apporte néanmoins d'autres problèmes. En effet, un des premiers écueil est de perdre du temps à générer des comportements simulés qui ne correspondent pas à la réalité. Le deuxième problème est qu'il est difficile et même impossible d'avoir une simulation qui reflète la réalité avec précision comme l'a indiqué entre autre Brooks dans ses travaux [Brooks 1992]. Les environnements simulés sont plus simples et trop parfaits en comparaison avec la réalité dont les mesures sont toujours approximatives même dans des environnements stables. De plus dans le monde réel les perceptions des objets et leurs relations sont difficiles à déterminer. Les actions sur l'environnement peuvent avoir des conséquences inattendues. Comme le rappelle Brooks, le cerveau humain consacre 50% de ses capacités à la perception du monde. Pour le développement, il propose [Brooks 1992] de réduire l'espace de recherche en facilitant l'apparition de structures naturelles telles que la symétrie et la modularité et de faire évoluer les programmes de manière incrémentale à mesure que des solutions viables apparaissent. Les différents capteurs sont ainsi ajoutés au cours de la simulation. Enfin Brooks propose au cours des générations de valider les résultats simulés sur un vrai robot cela afin d'orienter les recherches dans les bonnes directions et d'être en mesure au final d'obtenir les solutions souhaitées.

2.3 Travaux sur la morphologie figée

Dans les prochaines sections nous allons présenter les raisons pour lesquels il est intéressant d'avoir une morphologie qui évolue.

2.3.1 Évolution de la morphologie seule

En dehors, des travaux sur les plantes artificielles, il y a peu de recherches où l'on fait évoluer seulement une morphologie sans lui donner un comportement. Certains travaux en embryogenèse se consacrent uniquement au processus de développement des formes comme ceux de Duthen, Chavoya, Doursat et Fleischer [Chavoya and Duthen 2007a; Chavoya and Duthen 2007b; Doursat 2006; Doursat 2007a; Doursat 2007b; Fleischer 1996]. Nous traiterons de ces travaux prometteurs à la fin de cette section.

2.3.2 Évolution de contrôleur pour une morphologie fixée

En 1990, McKenna et Zeltzer sont parmi les premiers à réaliser un contrôleur pour une entité évoluant dans un environnement physique [McKenna and Zeltzer 1990]. Par la suite, de nombreux travaux sur l'évolution de contrôleurs pour une morphologie fixe ont apparu comme ceux par exemple de Ijspeert, Lee, van de Panne, Terzopoulos [Ijspeert 2000; Lee et al. 1996; van de Panne 1993; Terzopoulos et al. 1994]. Ces méthodes ont ensuite inspiré Nolfi S. and Floreano D. qui travaillaient sur des contrôleurs pour des robots à roues [Nolfi et al. 1994]. En parallèle de ces travaux, Bongard et Lipson [Bongard and Lipson 2004a; Bongard and Lipson 2004b; Lipson 2005] fabriquent des contrôleurs pour des créatures à pattes. Ici, il n'est pas question d'imiter la marche mais de trouver par évolution un contrôleur qui permette à une créature de se déplacer (Fig : 2.1). La créature évolue dans un environnement physique réaliste. Elle possède quatre pattes munies de capteurs de contact pour détecter le sol, de capteurs d'angle au niveau des articulations et de deux capteurs de phéromones. Le comportement de la créature est contrôlé par un réseau de neurones dont la topologie et le poids évoluent. Au commencement de la simulation, les neurones relient les capteurs aux actionneurs. La simulation comporte deux cents individus. Au bout de cinquante générations, le contrôleur produit arrive à contrôler la créature pour qu'elle se déplace en ligne droite ou si on le désire pour qu'elle suive la piste de phéromones. L'évolution a donc produit un contrôleur avec deux fonctions indépendantes. En 1995, Gruau obtient des résultats pour contrôler

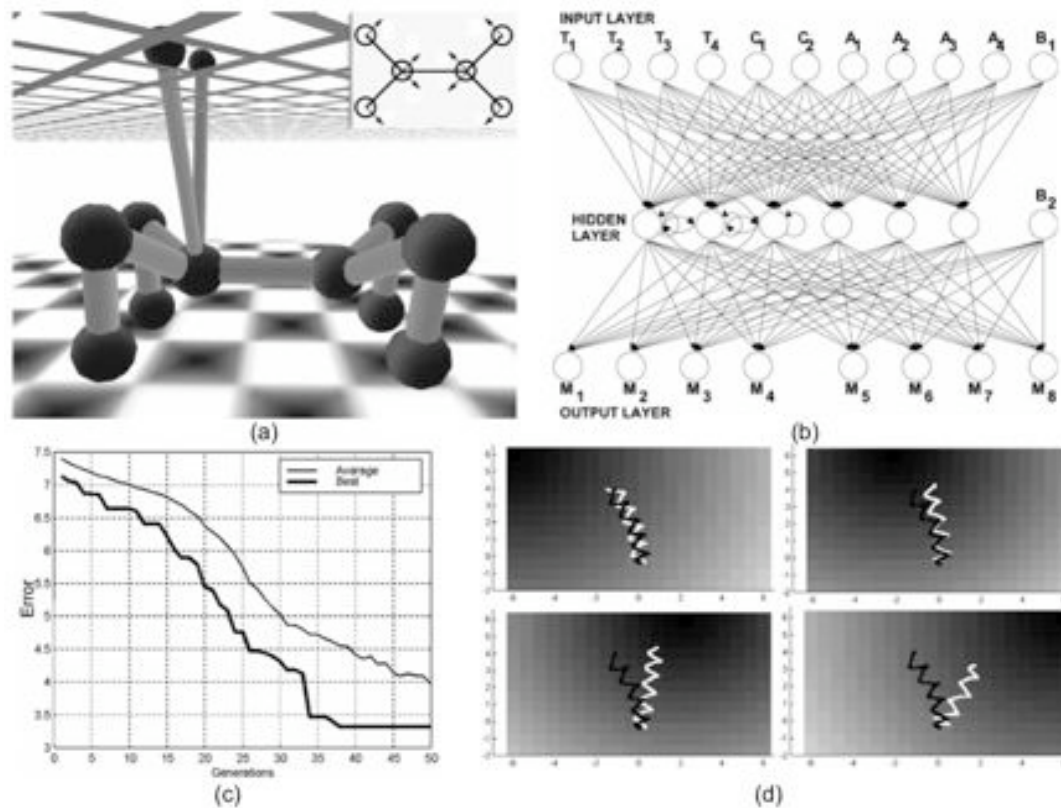


Figure 2.1: Travaux de Lipson et Bongard [Bongard and Lipson 2004a]. (a) Morphologie de la créature comportant quatre pattes, huit actionneurs et un capteur de contact par patte. (b) Réseau de neurones reliant les capteurs aux actionneurs. (c) La progression de l'erreur (distance qui sépare la créature de la plus forte concentration de phéromone) aux travers des générations. (d) Affichage de la concentration en phéromone (dégradés de gris). Le chemin en blanc est le trajet suivi par la créature quand les capteurs de phéromones sont actifs et le trajet est en noir quand les capteurs sont désactivés.

un hexapode avec des réseaux de neurones modulaires [Gruau 1995]. Dans leurs travaux, Cliff et Miler [Cliff and Miller 1996] font coévoluer les contrôleurs d'une proie et d'un prédateur créant ainsi des surenchérissements de stratégies de la part des deux créatures. Dans d'autres travaux Gritz utilise la programmation génétique pour faire évoluer le contrôleur de personnages en 3D [Gritz and Hahn 1997]. L'exemple traité montre une lampe articulée se déplaçant par saut. Le contrôleur le plus évolué permet à la lampe d'éviter les obstacles sur son passage. En 1997, Hervé Luga réalise des contrôleurs par algorithmes génétiques qui permettent des déplacements d'objets par coopération [Luga 1997].

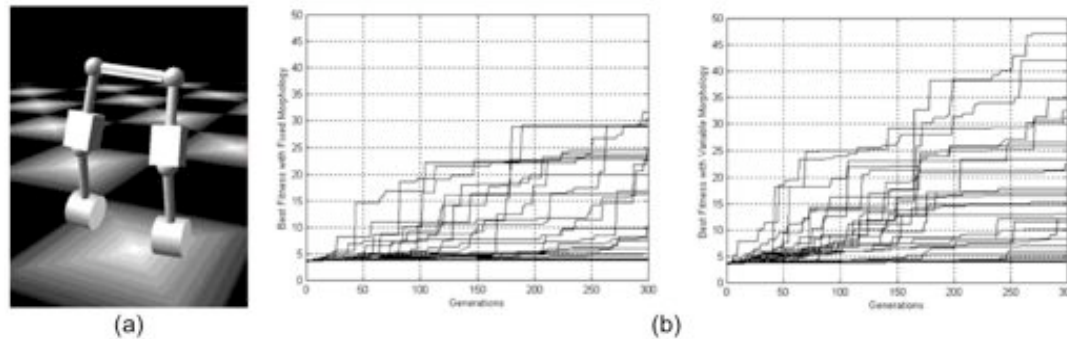


Figure 2.2: Évolution d'un contrôleur pour un robot bipède simulé [Paul and Bongard 2001] : (a) Représentation de la morphologie du robot dans le simulateur physique. (b) Comparaison des fonctions *fitness* au cours des générations. À gauche la morphologie est figée, à droite la répartition des masses évoluent. On voit clairement que l'évolution de la morphologie permet une meilleure convergence de la fonction *fitness*.

2.3.3 Les inconvénient d'une morphologie figée

Afin de voir quel était l'impact de l'évolution de la morphologie sur l'évolution des contrôleurs, Paul C. and J. C. Bongard propose une comparaison entre une simulation où la morphologie est figée avec une où elle évolue [Paul and Bongard 2001]. Dans cette simulation, un robot bipède muni de six actionneurs, deux capteurs de contact au niveau des pieds et des capteurs d'angle pour chaque articulation est modélisé dans un environnement physique simulé. Le but est de faire évoluer le contrôleur, un réseau de neurones, afin qu'il maximise la fonction de *fitness* qui traduit la plus grande distance parcourue en un temps fini. Dans la simulation où la morphologie est figée, seul le contrôleur évolue. Dans celle où la morphologie n'est pas figée, on fait évoluer la répartition des masses du robot. Lors de l'expérience, les simulation comprennent trois cents contrôleurs qu'on laisse évoluer sur une période de trois cents générations. À la fin des expérimentations, la simulation où la morphologie évolue montre des résultats nettement supérieurs (Fig. 2.2). Les auteurs en concluent que l'évolution conjointe de la morphologie et du contrôleur donne de meilleurs résultats. En effet, figer la morphologie restreint l'espace des solutions ce qui contraint l'adaptation du contrôleur. De plus, rien ne garantit que la morphologie d'un robot est adaptée pour la marche. Il y a donc un risque de faire évoluer une simulation pour un but que l'on ne peut pas atteindre.

2.4 Représentation de la morphologie

Lipson préconise de ne pas utiliser une représentation directe de la morphologie des créatures [Lipson 2005] tel que cela a été fait dans ses travaux précédents [Lipson and Pollack 2000; Pollack et al. 2003]. En fait, si cela peut fonctionner pour quelques éléments, cela n'est pas concevable pour les espèces que l'on retrouve dans la nature et qui comportent des milliards de cellules. Il est donc souhaitable que le phénotype ne soit pas un représentant direct du génotype. En effet, il est préférable que le phénotype suive un développement ou une croissance traduite par son génotype afin que la représentation soit indirecte. De ce fait la représentation génétique pourra être modifiée et évoluer, ce qui apporte des avantages tels que la réutilisation de certains composants ainsi qu'une compression de la représentation.

2.5 Les différents types de représentation de la morphologie

Ainsi une solution pour décrire la morphologie et un réseau de neurones peut-être un graphe comme dans les travaux de Karl Sims [Sims 1994a; Sims 1994c]. Pour leur part, Luke et Spector [Luke and Spector 1996] étudient les possibilités et les différentes représentations d'un graphe et de son développement. Une des solutions pour le développement d'une morphologie ou d'un réseau de neurones est l'utilisation d'une grammaire *context free* tel que les *L-systems*. Les *L-systems context free* peuvent avoir un haut degré de connectivité. Ce qui est souhaitable pour un réseau de neurones⁴, mais moins pour créer une créature se déplaçant. Cela est dû au fait que les *L-systems* peuvent imposer des contraintes de forte connexion entre les blocks (Fig: 2.3). Si les contraintes sont trop fortes, la probabilité de générer une créature pouvant se déplacer devient donc faible. De ce fait, il faut utiliser une représentation qui évite de générer des contraintes morphologiques. Ainsi les *L-systems context-sensitive* permettent d'éviter ces contraintes car elles prennent en compte le contexte dans lequel elles sont appliquées et évitent ainsi de générer des structures non viables. C'est ce type de représentation qu'Hornby utilise dans les travaux que nous décrirons par la suite [Hornby and Pollack 2001].

⁴Nous avons travaillé sur l'évolution par algorithme génétique de réseaux de neurones générés par *L-system* servant à évaluer des positions de jeu d'échecs [Autones et al. 2004]. Ces travaux ne sont pas intégrés dans ce document de thèse.

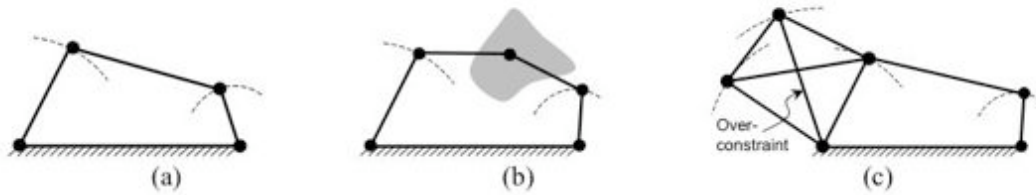


Figure 2.3: Contraintes imposées par la connexion de segment. Celles-ci peuvent apparaître aussi lorsqu'on utilise une grammaire *L-System context-free* (a) Pas de contrainte. (b) Contrainte par une aire délimitée. (c) Sur contrainte sur un segment.

2.5.1 Représentation par arbre

Une représentation par arbre [Lipson 2004] permet une construction du phénotype de manière *top-down* ou *bottom-up*. L'approche *top-down* part de la racine et descend jusqu'aux feuilles. Cette approche permet le développement du génotype. Les nœuds parents sont donc construits avant leurs fils. Cette méthode part donc d'une structure embryogénique qui est enrichie par exemple par deux opérateurs D et T (Fig: 2.4.(b-c)). L'opérateur D crée un nouveau nœud et le connecte à un ensemble de points par la création de nouveaux tubes. L'opérateur T crée un nouveau point au milieu d'un tube existant et relie ce point à un autre point existant localement proche. L'application de ces opérateurs est montrée par (Fig: 2.4.d). La structure se développe à partir d'une structure de base (Fig: 2.4.a). La construction de la même structure est présentée (Fig: 2.5.c) mais en utilisant l'approche *bottom-up*. C'est à dire que la structure est assemblée à partir d'éléments au lieu de se développer.

2.5.2 Représentation par développement

Une autre approche pour développer la morphologie d'une créature consiste à appliquer un ensemble de règles *context-free* à un axiome de départ. Ce type d'approche est similaire aux *L-systems* ou aux automates cellulaires. Elle peut être aussi bien appliquée à la morphologie qu'au développement d'un contrôleur. Deux règles très simples et un axiome de départ peuvent rapidement donner une structure très complexe. Dans le même esprit les travaux de Gruau sur la génération de réseaux de neurones par codage cellulaire peuvent aussi être utilisés pour le développement de contrôleurs et de morphologies [Gruau 1994].

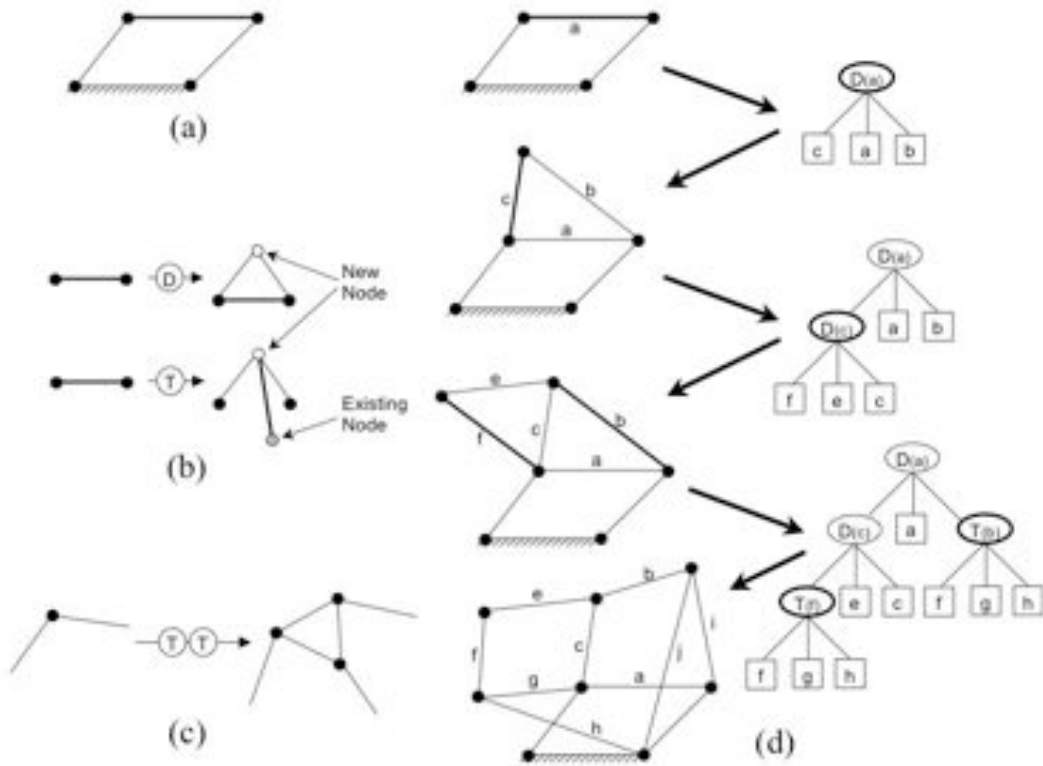


Figure 2.4: L'approche *top-down* permet à l'aide d'opérateur le développement d'une structure. En effet, la structure se développe en parcourant un arbre de sa racine (*top*) jusqu'à ses feuilles qui donnent plus de détails (*down*) (a) Structure de base (*top*). (b-c) Description des opérateurs. (d) Séquence montrant l'évolution de la construction d'une structure à partir de son embryon. Les détails de la structure augmentent plus on descend dans l'arbre, d'où le nom approche *top-down*.

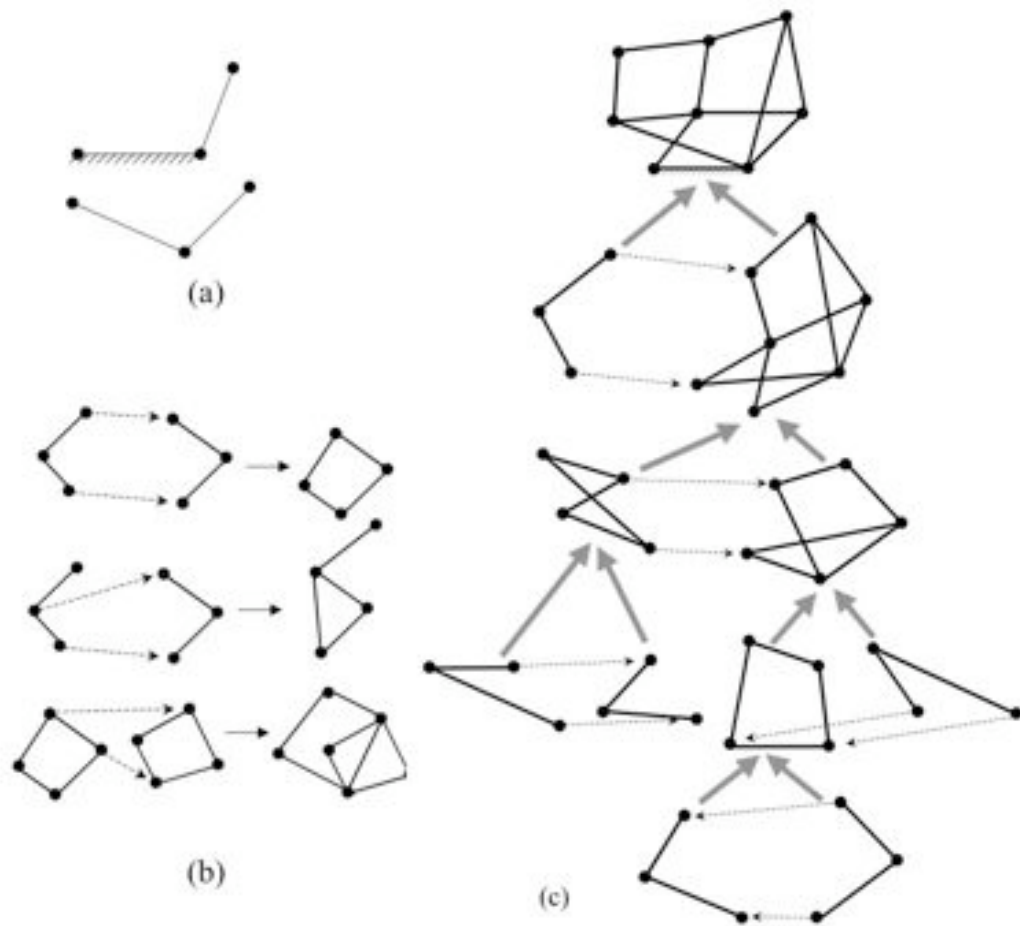


Figure 2.5: L'approche *bottom-up* est utilisée pour construire la même structure qu'à la figure précédente (Fig: 2.4). En ici, au lieu de développer la structure en la détaillant, elle est assemblée à partir d'éléments.

2.6 Conclusion

Dans ce chapitre, nous avons décrit les différents mécanismes d'évolution servant pour générer des créatures artificielles. Nous avons présenté les différentes représentations de ces créatures ainsi que les raisons pour lesquelles il est utile de faire évoluer conjointement la morphologie et le contrôleur. Dans le prochain chapitre, nous présenterons les travaux de références sur les créatures artificielles issues de ces approches.

Les créatures artificielles évolutionnistes et les écosystèmes

Dans ce chapitre, nous présentons l'ensemble des principaux travaux menés pour générer automatiquement des créatures artificielles. Cette section se termine par une synthèse sur l'état actuel des travaux de ce domaine.

3.1 Travaux sur les créatures artificielles évolutionnistes

3.1.1 Les créatures de Karl Sims

Les créatures de Karl Sims [Sims 1994a; Sims 1994c; Sims 1994b] restent parmi les travaux les plus convaincants et les plus aboutis du domaine. La motivation de ces travaux était qu'en réalité virtuelle [Krueger 1991], il était difficile de réaliser une animation et d'en garder le contrôle. Si par exemple, on définit tous les angles et mouvements d'un squelette, il sera difficile d'avoir une animation réaliste. Au contraire, si l'on a une simulation réaliste, c'est à dire où l'on définit les forces appliquées et où l'on calcule les lois de la physique, on aura un mouvement réaliste mais on en aura moins le contrôle. En 1994, il existait des techniques spécifiques pour animer des entités dans des environnements physiques [McKenna and Zeltzer 1990; Miller 1988; Raibert and Hodgins 1991]. Karl Sims propose d'utiliser les algorithmes génétiques [Holland and Reitman 1978; Goldberg 1989] afin de générer la morphologie et le comportement de créatures artificielles. Ce type d'évolution artificielle peut-être dirigée de deux façon :

- soit l'utilisateur choisi esthétiquement les individus à reproduire comme dans les travaux de Ray [Ray 2000] décrits plus bas ou dans [Dawkins 1986; Sims 1991; Sims 1992; Todd and Latham 1992].

- soit une fonction de *fitness* permet de faire évoluer les individus.

Bien que l'on perde du contrôle, Sims propose d'utiliser une fonction de *fitness* afin d'automatiser l'évaluation des créatures. L'utilisateur garde tout de même un certain contrôle en définissant la fonction de *fitness*. Karl Sims justifie ce choix par les succès rencontrés par ses prédécesseurs sur la génération automatique de contrôleurs [de Garis 1990; Ngo and Marks 1993; van de Panne 1993]. Malgré cela, les travaux de Karl Sims diffèrent car en plus de générer le contrôleur, il fait évoluer la forme de ses créatures en 3D dans un environnement physique très réaliste.

Pour la morphologie de ces créatures, il s'inspire des fractales, des *L-systems* et des *graftals* [Hart 1992; Kitano 1990; Lindenmayer 1968; Mjolsness et al. 1989; Smith 1984]. Le génotype des créatures est représenté par un graphe orienté. Les nœuds contiennent les informations décrivant les différentes parties rigides du corps. Le graphe ne correspond pas directement au corps de la créature mais à son processus de développement (Fig. 3.1, 3.3). La construction de la morphologie se développe par récursivité à partir de la racine du graphe¹. Les parents peuvent avoir plusieurs connexions vers le même fils. Les nœuds contiennent la description des blocs à savoir leur taille, le type de liaison qui les relie à leurs parents. Les types de liaisons peuvent être *rigide*², *universal*³, *revolute*⁴, *bend-twist*, *twist-bend* ou *spherical*⁵. Il ne donne pas les définitions exactes de ce qu'il entend par *bend-twist* mais on peut l'interpréter comment étant une liaison *resolute* associée avec une liaison *twist*. Les blocs contiennent aussi les neurones associés qui contrôlent les articulations. Les connections entre les blocs indiquent le placement du bloc relativement à son bloc parent, la symétrie, l'échelle et l'orientation.

Le contrôle s'effectue grâce à des capteurs qui mesurent différents aspects du monde tel que le contact entre deux surfaces, l'angle entre deux blocs. Un réseau de neurones interne composé de diverses fonctions permet de moduler le signal (Fig. 3.1). Sims s'est intéressé aux travaux de [Kitano 1990; Mjolsness et al. 1989] pour le développement du réseau de neurones. Des effecteurs agissent ensuite dans le monde virtuel en actionnant les muscles au niveau des liaisons. Chaque bloc possède un réseau de neurones, les réseaux de neurones sont reliés entre eux par un réseau de neurones qui supervise le comportement de la créatures. Chaque neurone représente

¹Il faut noter que des nœuds du graphe peuvent être connectés sur eux-mêmes ce qui permet la récursivité.

²Aucune rotation.

³Simule la rotation d'un Cardan.

⁴Rotation avec un degré de liberté, c'est à dire deux blocs s'articulent sur un seul axe.

⁵Rotation entre deux blocs à trois degrés de libertés.

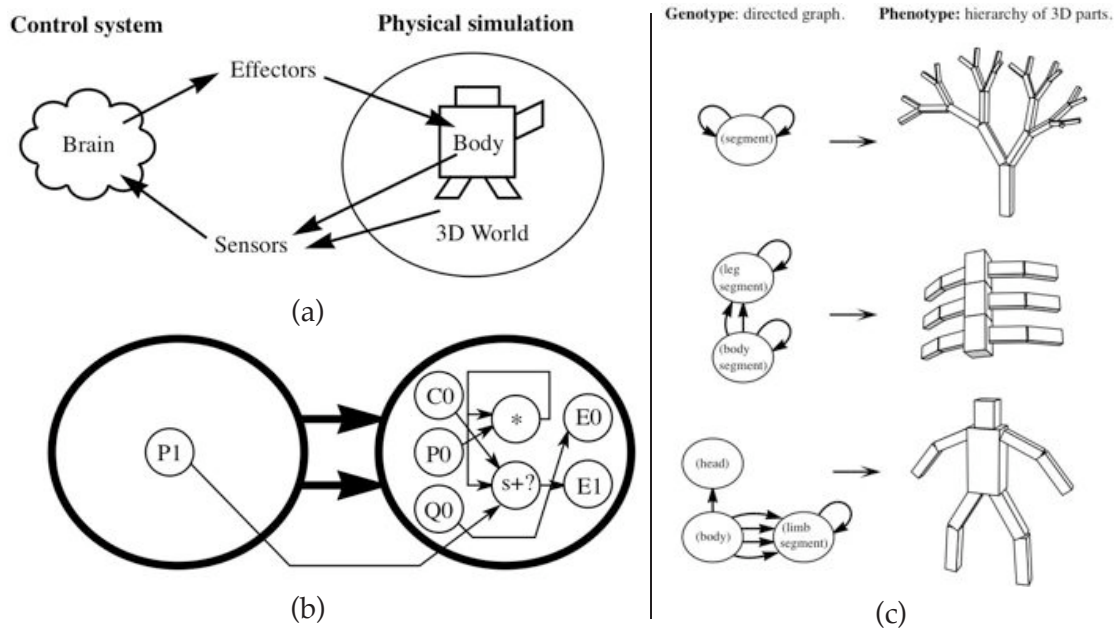


Figure 3.1: Créatures de Karl Sims [Sims 1994c]. (a) Simulation comportementale : Cycle entre le contrôleur, le corps de la créature et l'environnement. (b) Exemple de neurones à l'intérieur de noeuds d'un *graftal*. (c) Morphologie et processus de développement des créatures de Karl Sims

une fonction mathématique tel que des fonctions sinusoïdales qui modifient le signal.

L'environnement 3D, dont la gravitation et la détection de collisions sont simulées, est constitué d'un plan 2D sur lequel les créatures évoluent. La simulation physique utilise la méthode récursive décrite par Featherstone pour calculer les mouvements des créatures [Featherstone 1987]. Sims préconise que la simulation soit la plus stable possible car l'évolution peut facilement exploiter un bug pour satisfaire sa fonction d'évaluation⁶. L'évolution des créatures se fait grâce à un algorithme génétique. Au début de la simulation, une population est initialisée aléatoirement. Après évaluation des phénotypes par la fonction d'évaluation, les créatures sont sélectionnées pour ensuite être reproduites par croisement ou subir des mutations (Fig. 3.2). Il s'en suit une nouvelle génération et l'algorithme recommence jusqu'à que les créatures satisfassent la fonction d'évaluation ou atteignent un nombre limite d'itérations.

Durant la simulation les créatures sont évaluées pour une tâche spécifique dans l'environnement 3D. Après plusieurs heures de calculs⁷ et plusieurs simulations sur une *Connection Machine*, les résultats obtenus sont les suivants, l'évolution a sélectionné différents individus pouvant se mouvoir en rampant, nageant et sautant

⁶Un exemple est décrit dans nos résultats sur les escaliers (Sec.5.5.4).

⁷Environ 3 heures par simulation.

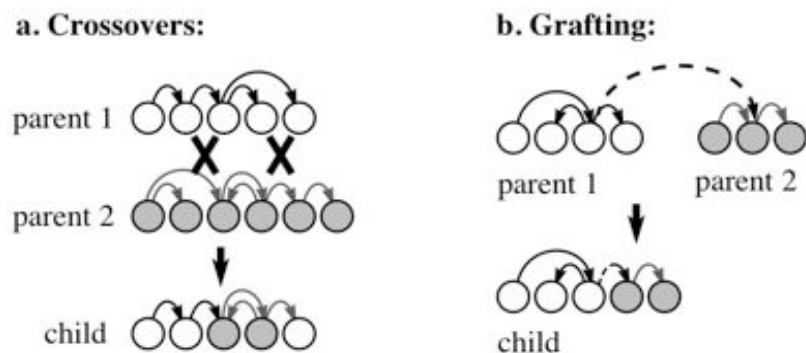


Figure 3.2: Les deux méthodes de croisements des *graftals* proposées par Karl Sims [Sims 1994c].

(Fig. 3.3). Parmi les créatures les plus élaborées, l'une d'entre elles peut suivre une source de lumière en nageant. Ainsi les résultats obtenues offrent une grande diversité de créatures ayant des stratégies très différentes. Bien que la récursivité soit une caractéristique importante de la morphologie des créatures de Sims, elle apparaît rarement dans les solutions trouvées. Ce qui n'influe pas sur les mouvements qui sont très réalistes et bien qu'il n'y ait pas de mécanisme de minimisation de l'énergie. En effet les créatures peuvent se déplacer en effectuant les mouvements qu'elles désirent sans limitation d'énergie.

Dans la seconde partie de ses travaux, il met directement les créatures en confrontation pour essayer de s'approprier un cube. Différentes stratégies intéressantes sont adoptées : certaines éloignent le cube à l'opposé de leur adversaire alors que d'autres tentent d'en garder la possession grâce à un membre articulé. Ces résultats sont spectaculaires, d'autant qu'ils n'ont jamais été reproduits avec un tel réalisme.

En perspective de ces travaux, Karl Sims propose d'améliorer ses créatures afin qu'elles interagissent avec des environnements plus complexes. Il énonce aussi la possibilité de construire des robots à partir des créatures les plus évoluées. C'est ce que Lipson et Pollack réalisèrent avec le projet Golem [Lipson and Pollack 2000].

Tim Taylor et Colm Massey dans un article présentant l'ensemble des travaux sur les créatures évolutionnistes donnent leur avis sur les résultats de Karl Sims [Taylor and Massey 2000]. Ils affirment que les résultats de Sims, en particulier le réalisme des mouvements des créatures sont dues à la puissance de calculs et à la précision de son moteur physique et non au contrôleur. Je pense au contraire que c'est en grande partie le contrôleur qui donne le réalisme aux créatures, même si le moteur physique y joue aussi son rôle. Un bon moteur physique ne suffit pas à rendre

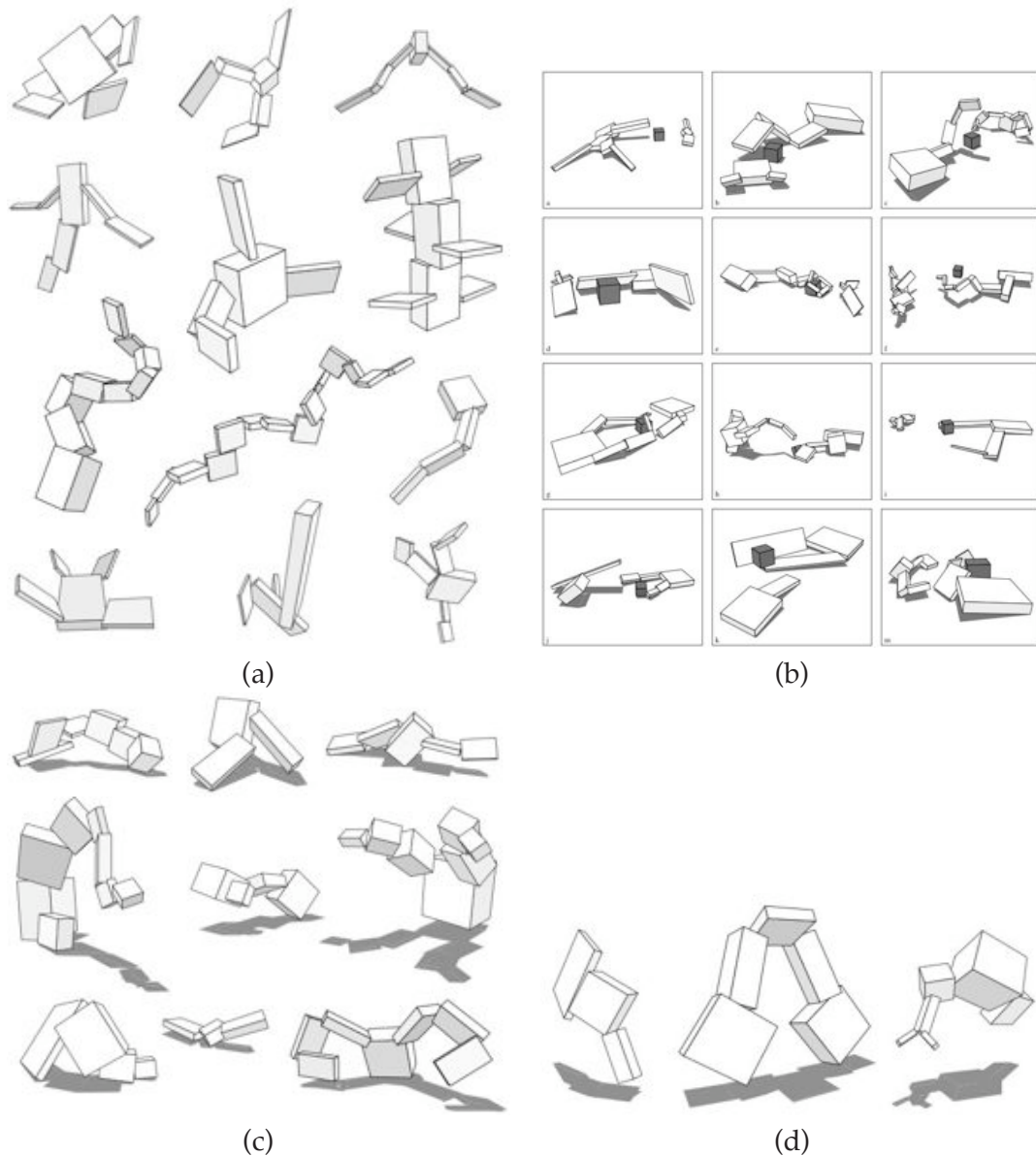


Figure 3.3: Créatures de Karl Sims [Sims 1994c] pouvant nager, ramper et sauter. (a) Sims obtient une large variété de créatures ayant des stratégies de déplacement très originales. En comparaison avec la symétrie, la récursivité sur un même bloc est peu présente dans les créatures générés. (b) Créatures de Karl Sims se battant pour la possession d'un cube. Ses travaux n'ont jamais été reproduits avec un tel réalisme. (c) Créatures de Karl Sims pouvant se déplacer. De même que pour la nage, Sims obtient une grande diversité de créatures avec comme stratégie de déplacement de sauter, de ramper ou de rouler. Il est encore plus surprenant de voir qu'aucune créature n'utilise la récursivité. Elle devrait pourtant faciliter la réutilisation du contrôleur. (d) Les résultats sont moins spectaculaires que pour la nage et la marche. Cela vient peut-être du fait qu'il est difficile d'obtenir des forces suffisantes pour propulser les créatures en hauteur. Mais s'il avait pu générer des forces plus grandes certainement qu'il apparaîtrait des comportements non désirés.

les créatures réalistes. Faut-il encore définir ce que l'on entend par réalisme. Mais pour ma part, je considère que ce qui donne en partie le réalisme est la continuité du signal. Les contrôleurs de Karl Sims génèrent moins de bruits car ils sont composés de sinusoides. De plus les morphologies et les stratégies comportementales des créatures de Karl Sims restent les plus élaborées⁸. Ce qui aujourd'hui donne encore le plus d'importance aux travaux de Karl Sims est que malgré les tentatives récentes [Miconi and Channon 2005; Miconi and Channon 2006a; Miconi and Channon 2006b; Miconi 2007], personne n'a reproduit ces résultats. Il faut noter qu'à cette époque il travaillait pour la compagnie qui produisait les *Thinking Machines* et de ce fait, il était entouré par une équipe qui l'aidait et dont le but était de promouvoir la société par le biais de ses travaux.

3.1.2 Les créatures de Tim Taylor et Colm Massey

Tim Taylor et Colm Massey ont programmé des créatures évolutionnistes [Taylor and Massey 2000; Taylor 2000] en utilisant le moteur MathEngine. Ce moteur utilise la méthode des multiplicateurs de Lagrange [Baraff 1996] avec la stabilisation de Baumgarte [Baumgarte 1972] pour résoudre les équations du mouvement. Leur but était de réaliser des créatures pouvant se déplacer. Pour les capteurs et les actionneurs, ils utilisent un réseau SAN *sensor- actuator-network* [van de Panne 1993]. La morphologie est décrite comme dans les travaux de Sims par un graphe de développement. Contrairement à Sims, ils utilisent des cylindres à la place de parallépipèdes rectangles cela pour des raisons de performance. La simulation commence avec une population générée aléatoirement qui évolue soit sur un sol dur soit dans un environnement liquide. Ils se sont heurtés rapidement au problème de la définition de la fonction d'évaluation. Dans l'eau par exemple, la simulation privilégie les créatures qui n'ont qu'un seul mouvement de poussée et qui s'éloignent du point de départ. Ainsi, cette catégorie de créatures, si le reste de la population est moins performante, peut diriger l'évolution vers un optimum local. Afin de résoudre ce problème, une solution consiste à d'augmenter le temps d'évaluation en espérant qu'une mutation modifie le contrôleur pour qu'il produise des mouvements de poussée cycliques. Le dilemme est de trouver le temps d'évaluation nécessaire à l'apparition de telles créatures. Une autre solution proposée par Sims est de mesurer la vitesse de la créature en fin de simulation et de lui donner un poids dans la fonction

⁸Mais certainement que la prise en compte et la minimisation de la dépense énergétique permettrait de rendre les mouvements des créatures encore plus réalistes.

d'évaluation. Cette méthode améliore les résultats de Taylor et Massey mais pas de façon significative et leur simulation tombe souvent dans un optimum local. Après de nombreuses simulations, ils finissent par trouver les bons paramètres pour leur simulation. La taille de leur population est autour de 300 individus et le nombre de génération pour obtenir un résultat est autour de 50-100 générations. Ils observent que maintenir une grande diversité de la population [Whitley et al. 1999] leur évite de tomber dans des optimum locaux et améliore ainsi la performance de leur simulation. Le nombre de blocs de chaque créature est compris entre 4 et 10. Le nombre de nœud dans chaque partie de la créature est entre 5 et 8. Ils favorisent les contrôleurs dont le signal est sinusoïdal. Afin de conserver une population diversifiée, ils conservent une petite partie de la population et renouvelle en partie le reste⁹. Chaque créature prend 10 à 50s pour être évaluée. Une simulation dure donc entre 4h et 8h sur un simple PC celeron 400MHz (année 2000). Les simulations des créatures de faibles performances ou presque immobiles après un tiers de la période d'évaluation, sont écourtées. Après plusieurs simulations, ils obtiennent ainsi une grande variété de créatures ayant la capacité de mouvoir dans l'eau ou sur un sol (Fig 3.4). Au final, les contrôleurs obtenus ne sont pas décrits et peu de résultats quantitatifs présentés. Cependant les travaux de Taylor restent tout de même importants car ils présentent l'état de l'art du domaine et montrent qu'il est envisageable de reproduire les résultats de Sims.

3.1.3 Les créatures de Thomas Ray

Dans ces travaux, Ray génère des créatures qui sont sélectionnées par l'utilisateur [Ray 2000]. Le but de son approche est de produire une émotion chez l'utilisateur qui pourra être surpris de l'évolution des créatures virtuelles. Il permet aussi à l'utilisateur d'aborder et de comprendre les mécanismes de l'évolution. Ainsi ces travaux se rapprochent des biomorphes de Richard Dawkins [Dawkins 1986] tout en s'inspirant de ceux de Karl Sims [Sims 1994c; Sims 1994a]. Pour la morphologie, il utilise des *graftals*. Pour l'esthétique, un gradient de couleur est appliqué sur les blocs récursifs (Fig. 3.5). Pour le contrôleur les créatures utilisent un réseau de neurones. Le moteur physique employé est mathengine. En ce qui concerne l'environnement, les créatures évoluent dans un liquide. En ce qui concerne les expériences, il y a peu de

⁹ Les premiers 20% de la population sont conservés pour la génération suivantes et forme l'élite. Le reste, 80% de la population, est régénéré en les sélectionnant selon la méthode des tournois (taille deux). Les nouveaux individus générés ont 60% de chance d'être introduit dans la population sinon on conserve leurs parents (40% de chance). Parmi les 60% de la population reproduite, 30% le sont par croisement et 30% par *grafting* [Sims 1994c]. Le second parent est choisi parmi les 20% de la population élite. La mutation ne s'effectue que sur la population élite.

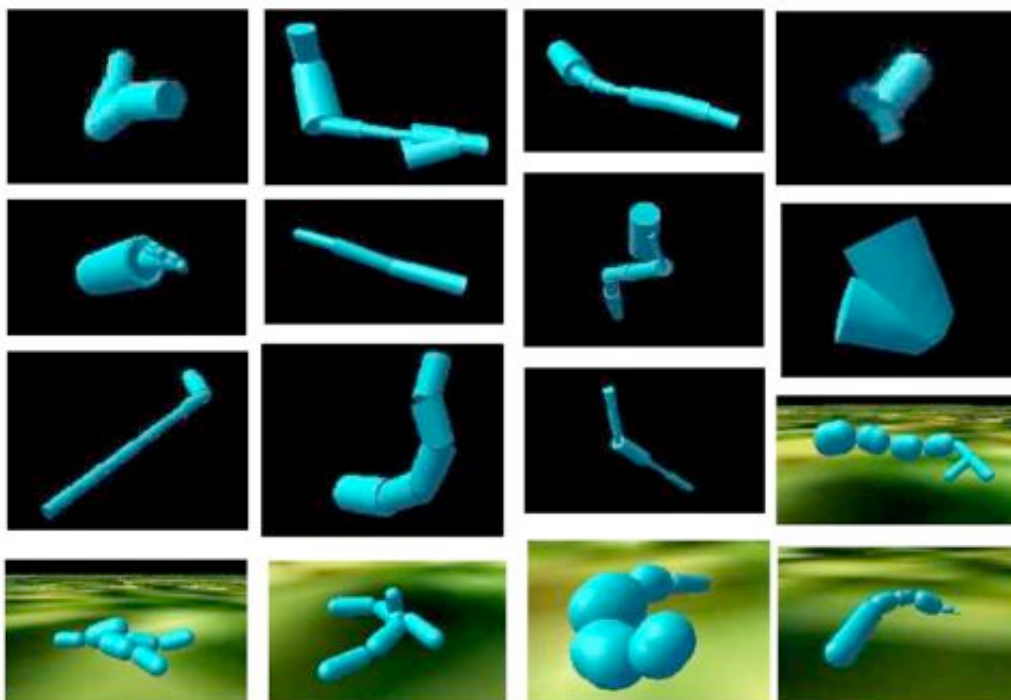


Figure 3.4: Les créatures de Tim Taylor et Colm Massey [Taylor and Massey 2000]. Elles ont le mérite de reproduire la méthode employée par Sims mais leur résultats est moins spectaculaire.

détails sur les paramètres employés durant les simulations. Pour autant les résultats donnent naissance à une grande variété de créatures très colorées et esthétiques. Étant donné qu'elles sont sélectionnées par l'utilisateur dans un but esthétique, leurs mouvements ne sont pas forcément adaptés à un déplacement. Ainsi, si la démarche n'est pas totalement scientifique comme le souligne Tim Taylor [Taylor and Massey 2000], elle est intéressante par la diversité des créatures générées. De plus, ces simulations ont permis de résoudre de nombreux problèmes de stabilité dus au moteur physique. En effet, la sélection par l'utilisateur (évolution interactive) permet d'explorer des cas extrêmes pour lesquels le moteur physique n'est pas toujours adapté. Pour ce qui est de l'originalité de ces travaux, Ray n'est pas le premier à permettre à l'utilisateur d'intervenir dans la sélection des créatures artificielles, en effet Karl Sims utilisa ce procédé sur des plantes artificielles dans les projets Genetic Image et Galapagos [Sims 1991].

3.1.4 Les créatures de Gregory Hornby

Les créatures de Hornby [Pollack et al. 2003; Hornby and Pollack 2001; Hornby and Pollack 2002] ont une morphologie basée sur les *L-systems context-sensitive* (Fig. 3.6). Cela permet d'avoir des structures très modulaires avec un grand nombre de segments (Fig. 3.7). Le contrôleur utilise des réseaux de neurones. Le moteur physique a été réalisé pour obtenir des bonnes performances sur ce type de structures. En ce qui concerne les simulations, elle durent autour de 250 générations. Les créatures ainsi obtenues sont d'une grande variété. De plus les meilleures créatures issues de l'évolution ont servi de modèles pour la réalisation de robots. Plus récemment, Hornby s'inspira ces créatures pour faire évoluer des structures statiques comme des tables et des antennes pour les satellites.

3.1.5 Les créatures de Hod Lipson

Les travaux de Lipson et Pollack [Lipson and Pollack 2000; Pollack et al. 2003] sont certainement les plus finalisés par le fait que leurs créatures ont été par la suite construites sous forme de robots (Fig. 3.8, 3.9). Ce sont les seules avec celles d'Hornby à avoir suivi tout le processus d'évolution jusqu'à leur réalisation physique. Les composants virtuels doivent donc avoir la possibilité d'être reproduits sous forme réelle. Pour cela, les créatures sont composées essentiellement d'éléments dit 1D tubes et de pistons (Fig. 3.9). Elles ne comportent pas de capteurs. Les créatures peuvent se mouvoir en compressant leurs pistons. Les joints entre les pistons peuvent

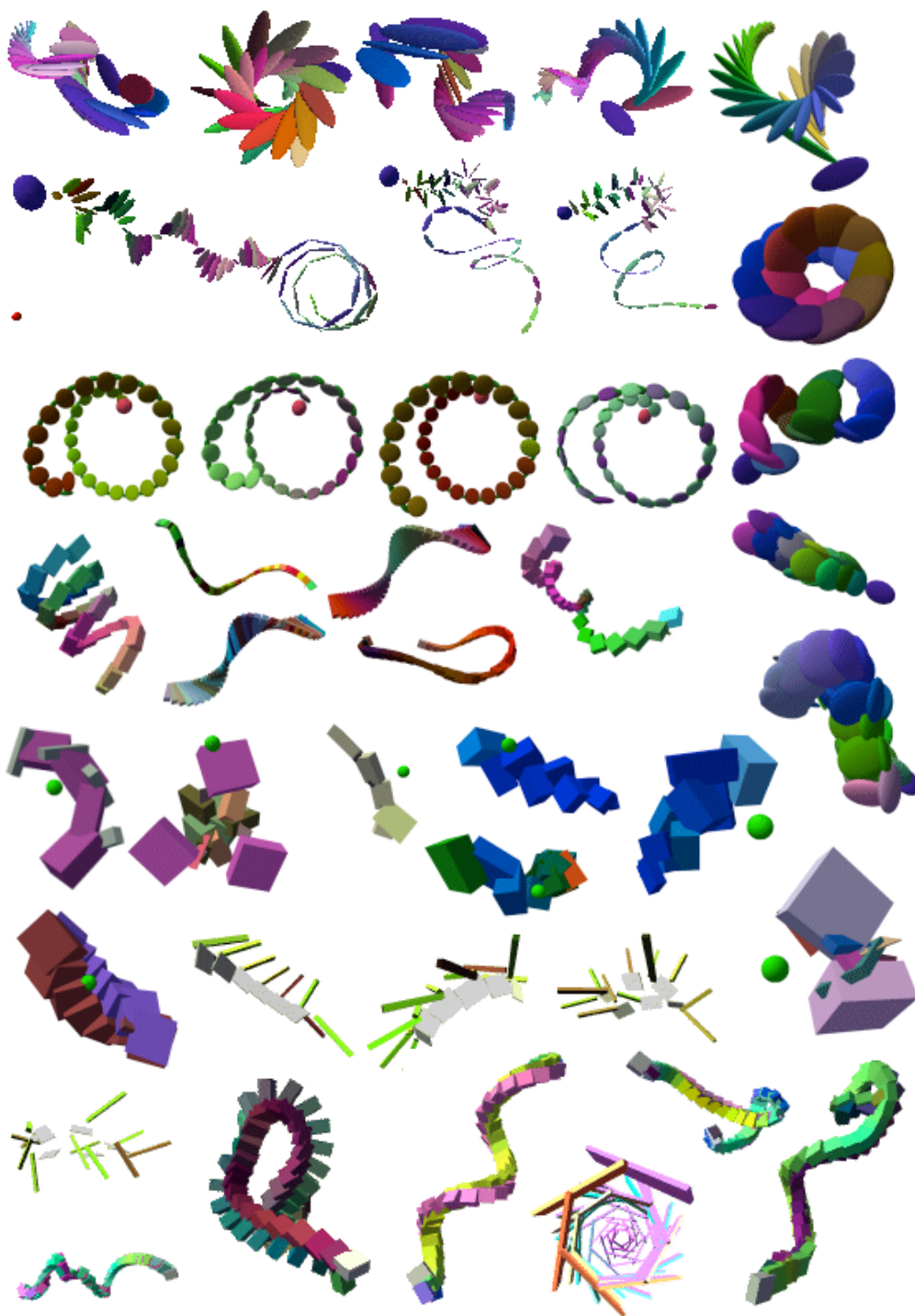


Figure 3.5: Les créatures de Ray [Ray 2000]. Son auteur permet à l'utilisateur d'intervenir dans la sélection des créatures. Le but de son approche est de produire une émotion chez l'utilisateur qui pourra être surpris de l'évolution des créatures virtuelles qu'il produit.

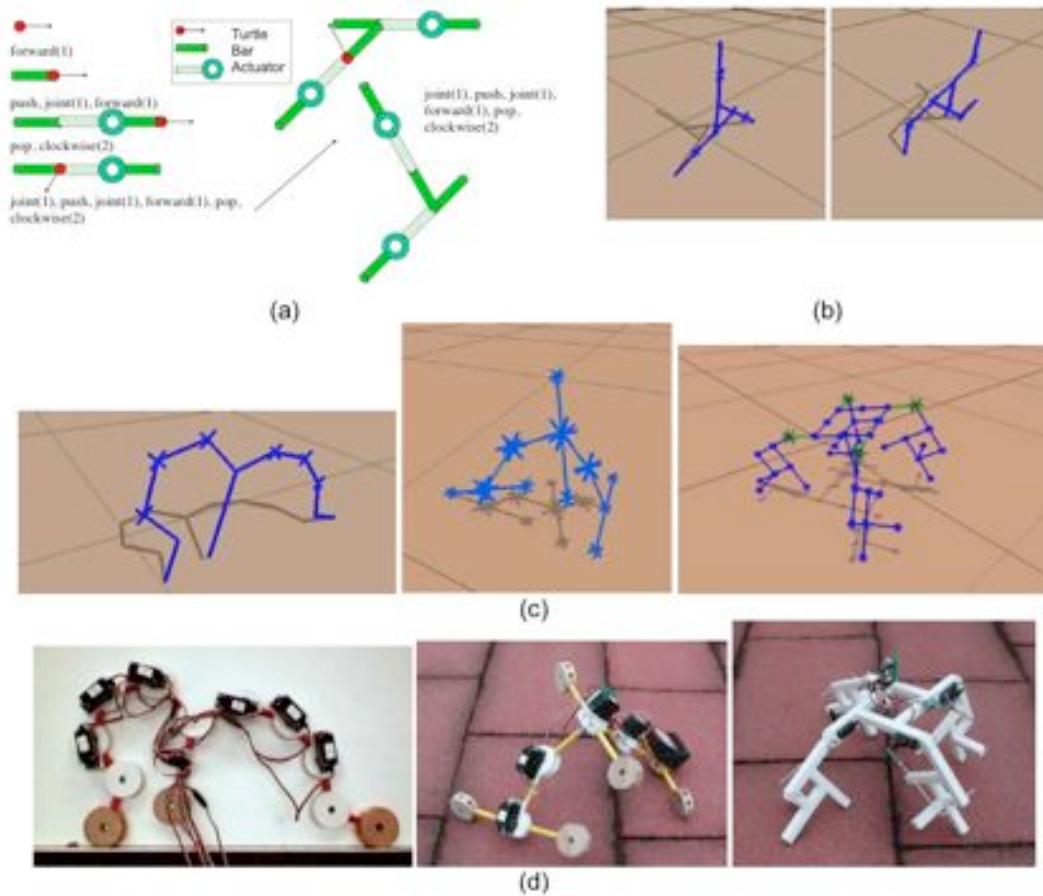


Figure 3.6: Représentation des créatures d'Hornby par L-system [Hornby and Pollack 2001]. (a) Représentation par *L-system*. (b)(c) Évolution dans le simulateur des créatures. (d) Réalisation des créatures sous forme de robots.

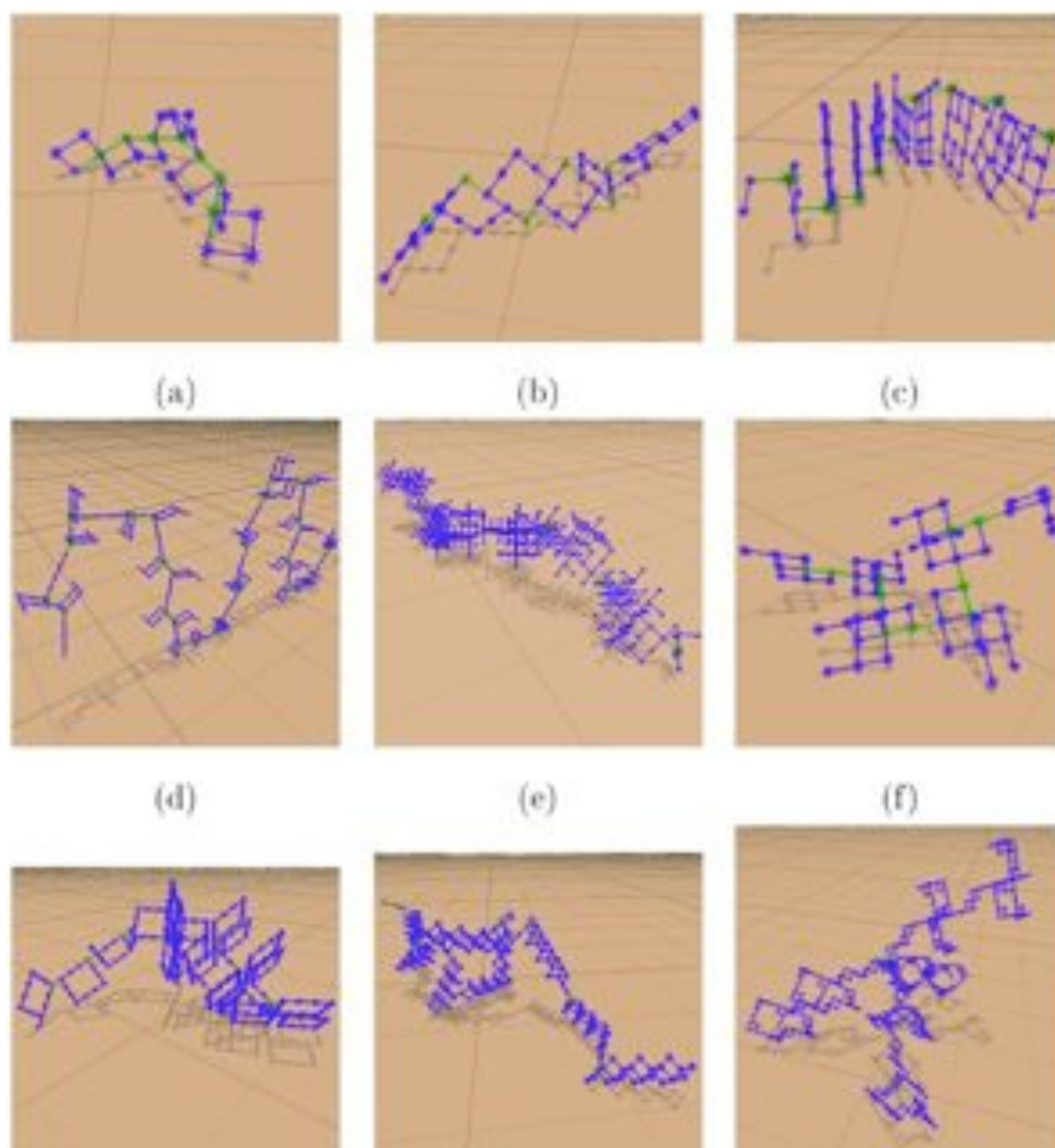


Figure 3.7: Les créatures générées par Hornby possèdent un grand nombre de segments (entre 59 et 629 segments) [Hornby and Pollack 2001].

être rigides ou permettre un certain degré de liberté. Les actionneurs sont contrôlés par un réseau de neurones produisant des fonctions sinusoïdales et permettant des topologies récurrentes (Fig. 3.10.a). L'évolution permet de modifier la morphologie des créatures ainsi que leur contrôleur.

Au début de la simulation, la population est de deux cents individus dépourvues de la moindre structures et du moindre contrôleur¹⁰. Les mutations sont alors la seule issue pour ajouter des nouveaux éléments aux créatures. La fonction de *fitness* détermine leur habilité à se déplacer en mesurant la distance parcourue par leur centre de gravité. La sélection permet aux créatures ayant le plus de potentiel de se reproduire. Au cours de la simulation¹¹, après plusieurs dizaines de générations les premiers mouvements de créatures apparaissent. Les *patterns* des morphologies et des neurones permettant de se mouvoir sont répandus dans la population par le processus de sélection et de croisement. Au cours de l'évolution la population se diversifie tout en étant sujette à des extinctions (Fig. 3.10.c). Il faut noter que ces créatures obtenues sont pour la plupart symétriques bien que cela n'ait pas été spécifié dans le programme. Cela s'explique par le fait que ces créatures symétriques sont plus performantes pour parcourir une longue distance. De plus les créatures sont robustes aux variations induites par les mutations, en effet les variations de la longueur des tubes ne réduit pas les performances des créatures. Les meilleures créatures à la fin de la simulation sont réalisées physiquement par une imprimante 3D. Ces robots comportent généralement une vingtaine de pièces. Ainsi la vie artificielle apportent de nouvelles perspectives dans l'élaboration des robots dont les architectures sont vouées à devenir de plus en plus complexes.

3.1.6 Les créatures de Chaumont

Récemment Nicolas Chaumont [Chaumont et al. 2007] s'inspire de la morphologie et des contrôleurs de Karl Sims pour réaliser ses créatures. Il propose, en plus d'évaluer les créatures sur leur capacité à se déplacer, de les évaluer sur leur capacité à catapulter un bloc. Il rencontre le même problème d'instabilité en début de simulation que celui rencontré par Tim Taylor [Taylor and Massey 2000] à savoir que les créatures peuvent rouler si on les laisse tomber sur le sol et qu'il peut y avoir des problèmes d'instabilité si elles sont posées sur le sol. Pour éviter cela au début de chaque évaluation, il

¹⁰Contrairement à la majorité des autres approches, les créatures sont généralement initialisées avec une morphologie et un contrôleur aléatoire.

¹¹La durée d'une simulation est de 300 à 600 générations.

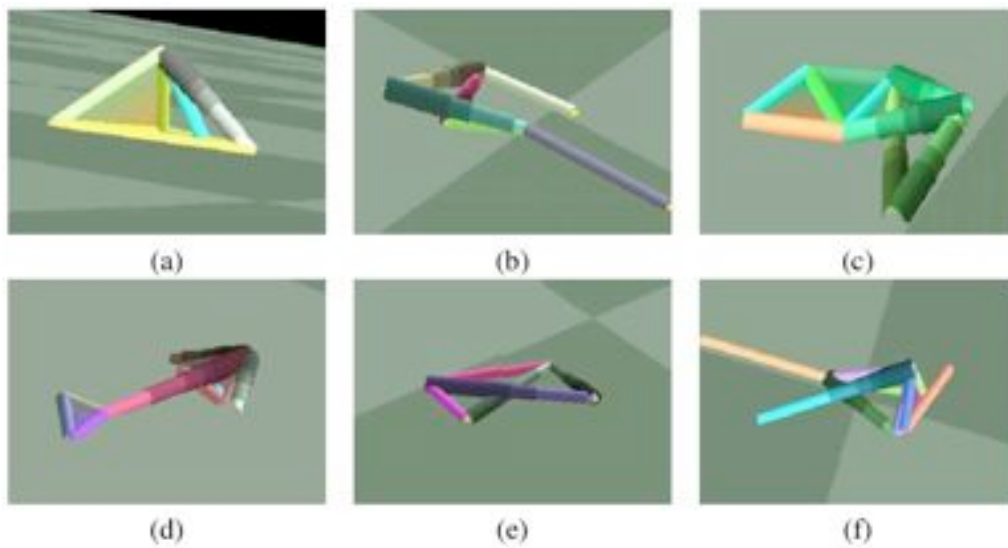


Figure 3.8: Les créatures artificielles de Lipson dans l'environnement simulé [Lipson and Pollack 2000].

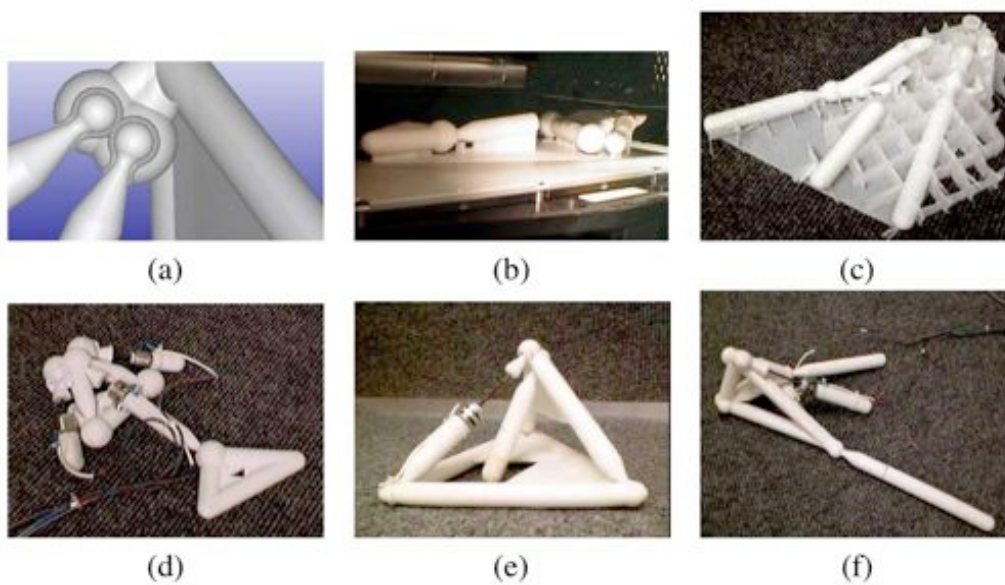


Figure 3.9: Projet Golem de Lipson [Lipson and Pollack 2000] dans lequel des robots sont réalisés à partir de créatures simulées.

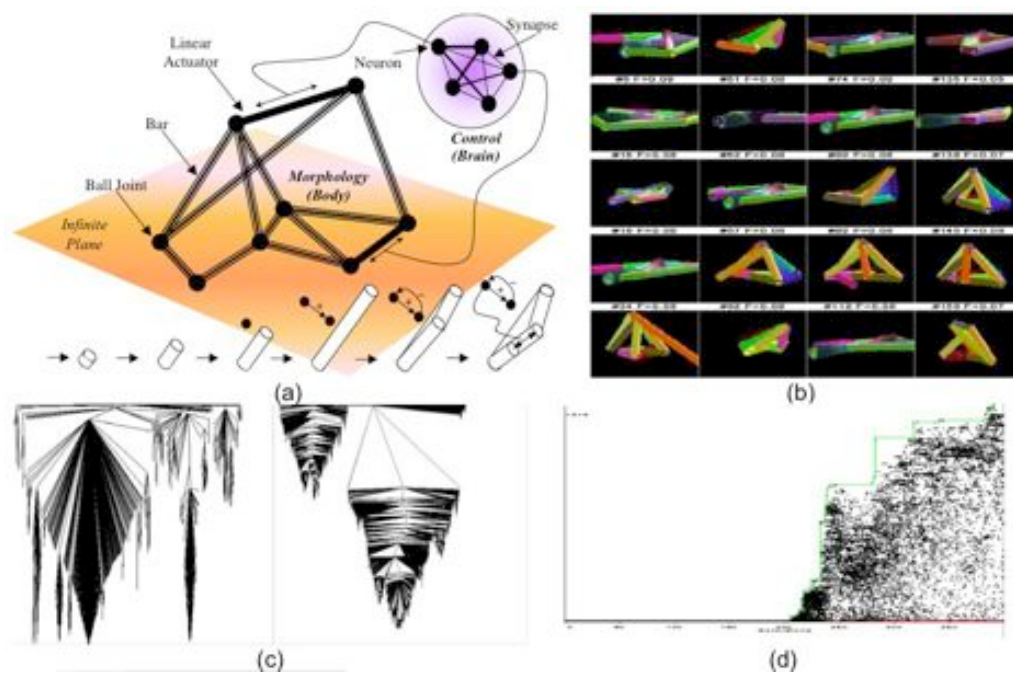


Figure 3.10: Projet Golem de Lipson [Lipson and Pollack 2000]. (a) Les actionneurs sont contrôlés par un réseau de neurones produisant des fonctions sinusoïdales et permettant des topologies récurrentes. (b) Les créatures produites à la fin de l'évolution. (c) Au cours de l'évolution la population se diversifie tout en étant sujette à des extinctions. (d) Évolution de la *fitness* des différentes créatures durant une simulation (chaque point représente une créature et son contrôleur).

désactive le contrôleur des créatures afin que celles-ci se stabilisent¹². En ce qui concerne les résultats, ils apparaissent trois types de stratégies : certaines créatures lancent le bloc, d'autres le poussent en tapant dedans, d'autres jonglent avec (Fig. 3.11). On peut en conclure grâce à ces nouveaux résultats que les créatures évolutionnistes peuvent s'adapter à des situations complexes, il serait donc intéressant de trouver quels sont les limitations des travaux de Karl Sims et comment les améliorer.

3.1.7 Les créatures de Miconi

Miconi très récemment a tenté de reproduire les créatures de Karl Sims [Miconi and Channon 2005; Miconi and Channon 2006a; Miconi and Channon 2006b; Miconi 2007]. S'il a réussi à améliorer les algorithmes de coévolution, les créatures obtenues sont moins efficaces et réalistes que celle de Sims (Fig. 3.12.a). Miconi a réalisé aussi des créatures qui se combattent¹³ (Fig. 3.12.b) tel que dans [Komosinski 2000] et [O'Kellym and Hsiao 2004]. À la différence de Sims, il utilise des réseaux de neurones de type McCulloch-Pitts [McCulloch and Pitts 1943]. On peut en conclure que les travaux de Miconi sont parmi les plus proches de ceux de Karl Sims, tout en offrant de nouvelles perspectives.

3.1.8 Les créatures de Shim

Shim commence par produire des créatures avec la possibilité de se mouvoir. Ces créatures ressemblent morphologiquement à celles de Tim Taylor vues précédemment [Taylor and Massey 2000]. Puis il s'intéressa à l'évolution de créatures volantes [Shim et al. 2004; Shim and Kim 2003; Shim and Kim 2007]. Il fait donc évoluer le contrôleur et les ailes de ces créatures (Fig. 3.13). Il obtient des résultats très intéressants avec une grande variété d'ailes adaptés au vol (Fig. 3.14). Récemment des travaux similaires ont été menés par Meyer et al. sur le projet Robur [Doncieux and Meyer 2003]. Leur travaux ont pour but de construire physiquement des créatures ayant la capacité de voler.

¹²Par la suite, nous verrons dans la proposition de notre modèle une solution qui permet de résoudre simplement les problèmes d'instabilité sans avoir recours à la désactivation du contrôleur.

¹³Cependant, il n'est pas toujours facile d'évaluer les stratégies développées par les créatures qui se combattent.

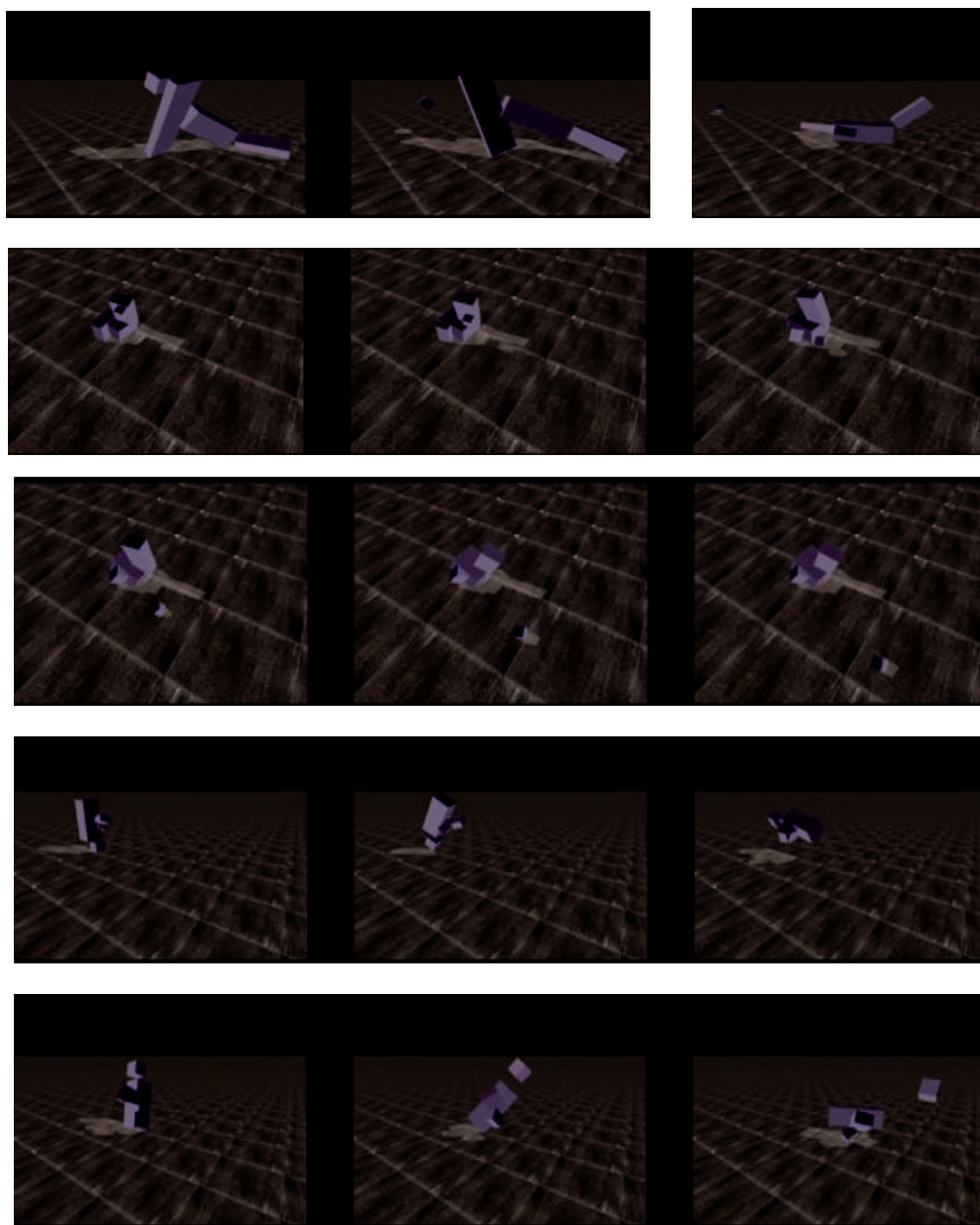


Figure 3.11: Les catapultes de Chaumont [Chaumont et al. 2007] : Différentes stratégies sont obtenue pour catapulter un bloc, certaines créatures lancent le bloc, d'autres le poussent dans tapant dedans, d'autres jonglent avec.

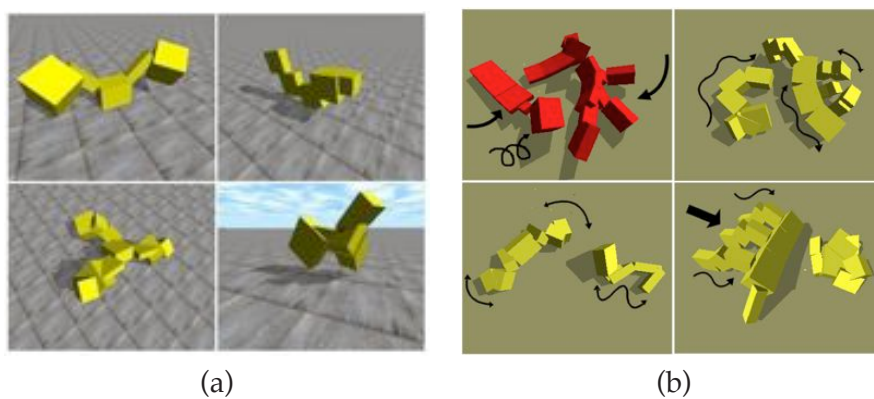


Figure 3.12: Les créatures de Miconi [Miconi and Channon 2005] inspirées de celles de Karl Sims. (a) Les créatures mettent en place différentes stratégies de placements. (b) Miconi utilise la coévolution pour réaliser un tournoi où les créatures se combattent en duel.



Figure 3.13: Les créatures de Shim évoluent pour adapter les ailes au vol [Shim et al. 2004].

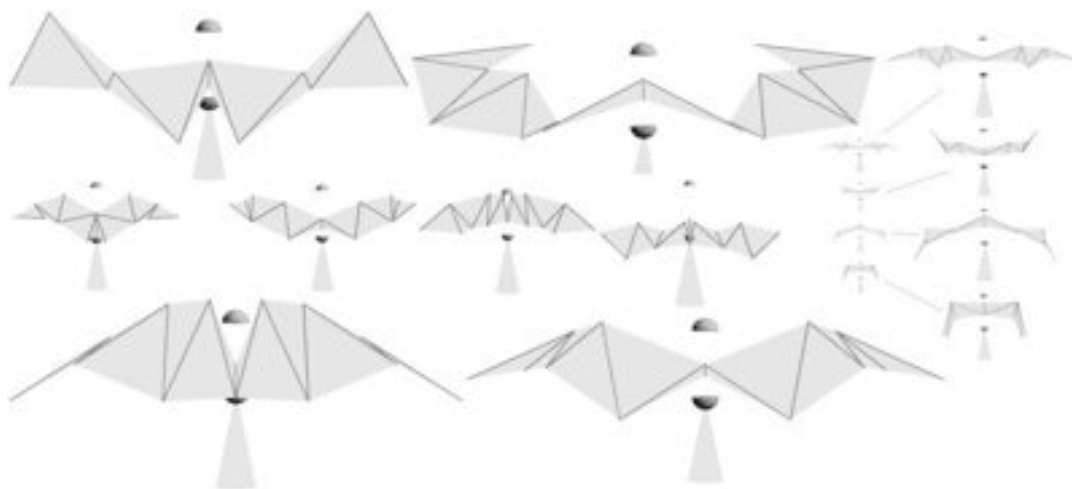


Figure 3.14: Ensemble des ailes générées par évolution [Shim et al. 2004].



Figure 3.15: Créature de Framesticks [Komosinski 2000] : (a) L'énergie est représentée sous forme de sphère lumineuse. (b) Une créature réalisée directement à l'aide du langage script et son contrôleur est lui évolué par algorithme génétique.

3.1.9 les créatures de Komosinski pour le projet Framsticks

À la différence de Karl Sims, Komosinski¹⁴ [Komosinski 2000] crée un environnement en 3D plus riche dans lequel plusieurs créatures tentent de survivre. On peut considérer Framsticks comme étant le premier écosystème 3D bien que son but est de faire évoluer des créatures et non l'environnement¹⁵. Pour cela, elles doivent s'approprier de l'énergie¹⁶(Fig. 3.15.a). La morphologie des créatures est composée de tubes, *sticks*, qui peuvent avoir des caractéristiques différentes. Les capteurs sont plus nombreux mais les contrôleurs sont du même type que ceux de Karl Sims, à savoir des réseaux de neurones composés de fonctions mathématiques élémentaires. Les résultats ne sont pas aussi convaincants que ceux de Karl Sims mais un langage approprié permet de créer facilement ses propres créatures (Fig. 3.15.b) et de les faire évoluer dans un but défini par l'utilisateur ou par sélection naturelle en les mettant en compétition avec d'autres créatures.

¹⁴L'ensemble des travaux de Komosinski qui concernent le projet Framsticks : [Komosinski 2005; Adamatzky and Komosinski 2005; Komosinski 2003; Komosinski and Ulatowski 1999; Komosinski and Kubiak 2001; Komosinski et al. 2001; Komosinski and Rotaru-Varga 2001].

¹⁵En effet, Framstick n'offre pas tous les outils qui permettraient l'observation et l'étude de la dynamique des populations comme cela se fait dans les écosystèmes 2D et que nous allons voir par la suite.

¹⁶L'énergie est représentée sous forme de sphère lumineuse.

3.2 Les écosystèmes

Dans cette section, nous présentons l'ensemble des écosystèmes artificiels dont le but est de faire évoluer des créatures artificielles. Contrairement aux travaux précédemment présentés qui se focalisent sur les créatures dont une seule est généralement présente à la fois dans l'environnement, les écosystèmes artificiels s'intéressent aux stratégies et aux adaptations qui émergent de l'interaction d'une multitude de créatures.

3.2.1 Tierra

Durant les années 1991-2001, Thomas Ray développe le projet Tierra [Ray 1991a; Ray 1991b] en s'inspirant des programmes Core Ware et Darwin [McIlroy et al. 1971]. Il propose de réaliser un écosystème dont l'environnement est la mémoire de l'ordinateur et les individus des programmes. Les programmes se disputent les ressources du microprocesseur et de la mémoire afin de survivre. Ils évoluent par mutation, croisement et réplication. Le génotype et le phénotype sont confondus car ils sont représentés par le programme ce qui limite l'évolution par rapport à une représentation indirecte. Au commencement du projet, Ray espérait que simplement avec la mutation et le croisement il puisse produire au bout d'un certain temps un programme avec la capacité de s'auto-reproduire. Il abandonna finalement cette idée au vue de la faible probabilité que cela apparaisse. Il initialisa donc son environnement avec des programmes ayant la capacité de se dupliquer. Ces organismes digitaux qui ne sont soumis à aucune fonction d'évaluation se battent pour la conquête des ressources de l'ordinateur comme la mémoire et le processeur. Durant les simulations il observe des stratégies différentes. Certains programmes continuent à se répliquer comme le programme originel. D'autres n'ont plus la capacité de se dupliquer mais parasitent d'autre programme pour profiter de cette fonction. Il observe aussi que des programmes parasites sont parasités à leur tour par d'autres programmes (Fig. 3.16).

Cependant on peut reprocher à Ray d'initialiser Tierra avec des individus qui possèdent des fonctions de réplication au lieu d'individus générés aléatoirement. Malgré l'aide apportée à l'évolution de Tierra, Russell Standish montra que la complexité des programmes générés durant les simulations n'était pas si élevée [Standish 2004]. Une autre reproche faite au projet Tierra est que les programmes évoluent tous dans le même espace et un programme peut en écraser un autre. De ce fait, il devient difficile de retracer l'évolution des programmes et ce qui se passe

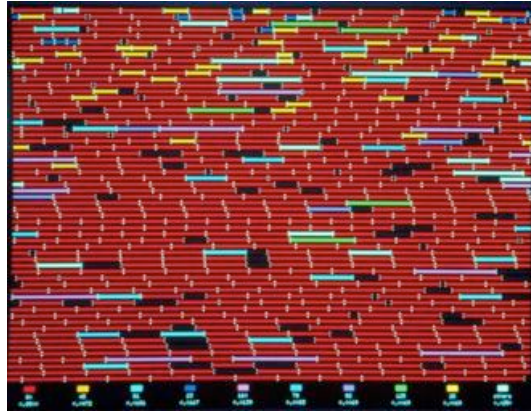


Figure 3.16: Tierra [Ray 1991a], l'écosystème réalisé par Thomas Ray dont on voit ici une représentation de la mémoire peuplé de programmes. Ceux-ci ont des couleurs différentes selon leur taille, les programmes les plus petits en jaune sont des parasites qui profitent des capacités de réplifications des autres programmes.

dans l'environnement. Le projet Avida s'inspire de Tierra [Adami and Brown 1994; Wilke et al. 2001; Lenski et al. 2003] et résout ce problème en utilisant une mémoire séparée et un microprocesseur virtuel pour isoler l'exécution de chacun des programmes.

Si les programmes de Tierra (qui sont des organismes digitaux) ne sont pas des créatures artificielles, on peut tout de même considérer des créatures dans un écosystème comme étant des métaphores des programmes de Tierra. La question reste ouverte quant-a leur possibilité à acquérir des propriétés du vivant [Stewart and Mossio 2007].

3.2.2 Gene Pool

Jeffrey Ventrella produit des créatures sous une sorte de microscope dans un environnement 2D simulant un laboratoire [Ventrella 1994; Ventrella 1998a; Ventrella 1998b; Ventrella 1996; Ventrella 2005]. Ses créatures, les *swimbots*, ont une morphologie et un contrôleur plus simple que celles de Karl Sims mais l'intérêt des travaux réside dans l'environnement (Fig. 3.17). L'environnement physique étant plus simple cela permet de faire évoluer un grand nombre de créatures simultanément. La morphologie des créatures est composée de rectangles aux angles arrondis qui sont reliés par des articulations. Chaque créature est composée en moyenne d'une dizaine de blocs. Leur gène est constitué de 16 nombres réels.

Chaque créature possède une quantité d'énergie qui diminue lors de ses

déplacements et de sa reproduction. Les créatures sont sexuées et la sélection est naturelle. Elles peuvent regagner de l'énergie si elles trouvent de la nourriture répartie initialement dans l'environnement. Pour cela, elles ont la possibilité de détecter la nourriture qui est à proximité. Elles essaient ensuite de s'orienter vers cette nourriture en actionnant leurs membres contrôlés par des fonctions sinusoïdales pondérées. Les créatures cherchent aussi à se reproduire selon leur état qui est décrit par un diagramme d'états comportant quatre possibilités et deux actions : recherche d'un partenaire en vue de se reproduire, se diriger vers le partenaire, recherche de nourriture, se diriger vers la nourriture, se reproduire et manger. La recherche de partenaire se fait à l'aide de cinq critères : la taille (petit, grand), la corpulence (mince, gros), la couleur (similaire, opposée), la réactivité (actif, mou) et la silhouette (droit, tordu). Ces caractéristiques déterminent ainsi des espèces. Les espèces qui ne sont pas adaptées pour se nourrir ou se reproduire finissent par disparaître lorsqu'elles n'ont plus d'énergie.

La simulation commence avec une population initiale tirée aléatoirement. La nourriture y est présente. L'utilisateur peut alors observer en temps réel l'évolution des espèces grâce une interface et une grande panoplie d'options. Parmi elles, la possibilité de manipuler les créatures en les plaçant dans des tubes à essais virtuels, de les reproduire, de les supprimer, de modifier leur énergie etc. Il est aussi possible de modifier leur environnement, de leur ajouter de la nourriture pour en observer les conséquences.

Au cours de ces simulations, Ventrella observe et met en évidence de nombreux comportements. Par exemple, lorsqu'il favorise la reproduction des individus dont les couleurs sont opposées, très vite il y observe une grande diversité des créatures colorées. À l'opposé, lorsqu'il favorise la reproduction des créatures dont les couleurs sont similaires, très vite chaque espèce converge vers une couleur uniforme. Il y observe aussi les capacités d'adaptation de la sélection naturelle lorsqu'il change brutalement les critères d'affinité de la reproduction.

3.2.3 Life Drop

Life Drop [Métivier et al. 2002] est un écosystème comparable à celui de Gene Pool [Ventrella 2005] dont le but est aussi d'essayer d'analyser et de comprendre la complexité de l'évolution d'un écosystème [Heudin 1998]. Ces travaux sont dans la lignée de Tierra, Avida et Gaia [Ray 1991a; Adami and Brown 1994; de Garis 1990]. L'environnement prend place dans une goutte d'eau artificielle (Fig. 3.18). L'environnement prend en compte des données comme le pH , la fluidité

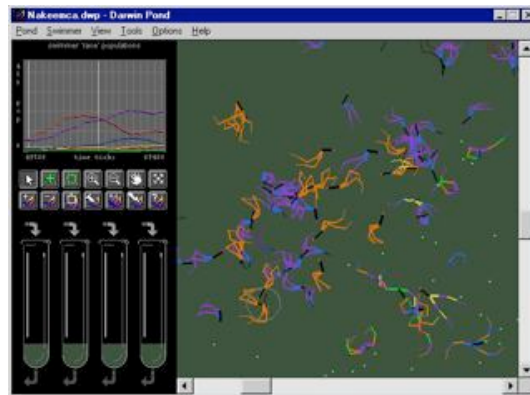


Figure 3.17: Gene Pool, l'écosystème réalisé par Jeffrey Ventrella [Ventrella 2005].

de l'eau. La morphologie des créatures appelées *biomorph* sont inspirées des travaux Dawkins [Dawkins 1986]. L'état interne et la perception des créatures est basée sur le modèle de Brooks [Brooks 1991a]. Le modèle physique qui permet aux créatures de se déplacer dans l'environnement s'inspire de celui de Reynolds [Reynolds 1999]. Une créature se compose donc de quatre couches : son génotype composé de 32 bits, son métabolisme primaire, son comportement réactif et son comportement cognitif. La créature possède plusieurs états possibles comme la vitalité, la faim, la fécondité et le stress. Elles interagissent avec l'environnement par le biais de plusieurs actions comme chercher un partenaire, chercher de la nourriture, fuir un danger, se déplacer aléatoirement etc. Les créatures se reproduisent entre elles suivant leur affinité génétique et leur niveau de stress. Il faut noter que c'est généralement le phénotype qui est utilisé pour l'affinité. Trois types d'expérience sont mises en place. La première prend en compte un environnement favorable. Le nombre d'espèces se stabilise dans le temps. Dans les deux dernières expériences, l'environnement subit des crises, comme la hausse du *pH* de l'eau qui est moins favorable aux créatures. Le nombre d'espèces diminue lors des crises et augmente à nouveau en fin de crise. Les expériences montrent ainsi l'implication du stress sur la dynamique des populations.

3.2.4 Les autres écosystème artificiels

Il existe d'autres travaux que nous ne détailleront pas ici comme Avida [Adami and Brown 1994] qui s'est inspiré de Tierra. Larry Yaeger a développé Poly-Word [Yaeger 1994], un environnement aussi basé sur Tierra. Dellaert et Beer qui étudient le développement morphologique de créatures 2D dans un univers discret [Dellaert and Beer 1996].

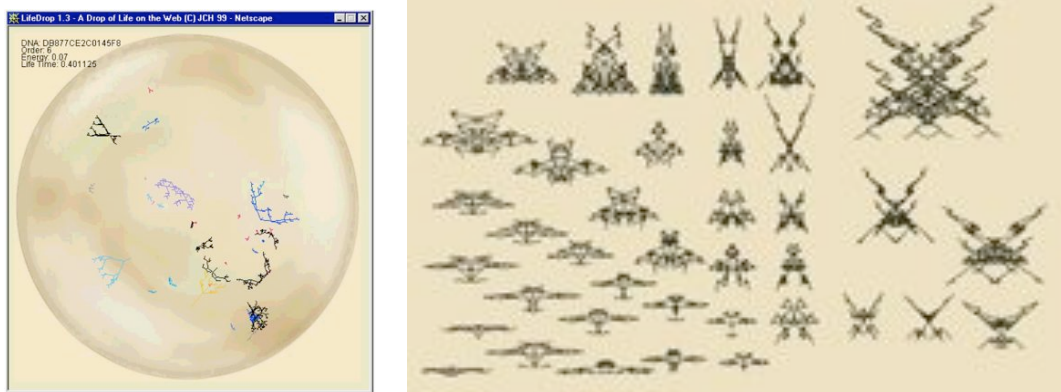


Figure 3.18: Life Drop [Métivier et al. 2002] : Évolution d'un écosystème artificiel dans une goutte d'eau. Il est peuplé de créatures appelés *biomorph*.

3.2.5 Embryogenèse et ontogenèse artificielle

L'embryogenèse artificielle est certainement l'une des voies les plus prometteuses pour générer des créatures artificielles. L'embryogenèse s'inspire du vivant pour générer la morphologie, voir le contrôleur d'une créature. Le but n'est plus de générer directement une créature mais de la développer à partir d'une cellule oeuf. La créature va ainsi croître par division cellulaire contrôlée par un ensemble de gènes et par régie par un réseau régulateur de gènes. Un des premier travaux intéressants traitant de la croissance cellulaire est celui de Kurt Fleischer [Fleischer 1996]. Plus récemment, René Doursat propose une méthode pour générer des structures (Fig. 3.19) basées sur l'embryogenèse [Doursat 2007b; Doursat 2006; Doursat 2007a]. Pour cela, il fait évoluer des réseaux régulateurs de gènes qui servent à la représentation des *patterns* et des formes de ces créatures. Tout aussi récent, Arturo Chavoya et Yves Duthen ainsi que Sylvain Cussat-Blanc proposent chacun un modèle de développement cellulaire dont les objectifs finaux sont de réaliser des créatures artificielles [Chavoya and Duthen 2007b; Chavoya and Duthen 2007a; Cussat-Blanc et al. 2007] (Fig. 3.20). Dans ces travaux, Bongard s'inspire de l'ontogenèse artificielle¹⁷ pour réaliser des créatures artificielles (Fig. 3.21) qui utilisent un réseau régulateurs de gènes comme dans les travaux de Doursat, Duthen et Chavoya [Bongard and Pfeifer 2003]. Ses travaux montrent qu'il y a une évolution séparée entre le réseau de neurones et le corps des créatures

¹⁷L'ontogenèse artificielle décrit l'ensemble des processus de développement des créatures artificielles de leur création jusqu'à leur fin, il inclue donc embryogenèse. Bongard emploie le terme ontogenèse car il s'intéresse au processus de développement des créatures tout au long de leur cycle de vie et non uniquement à leur création.

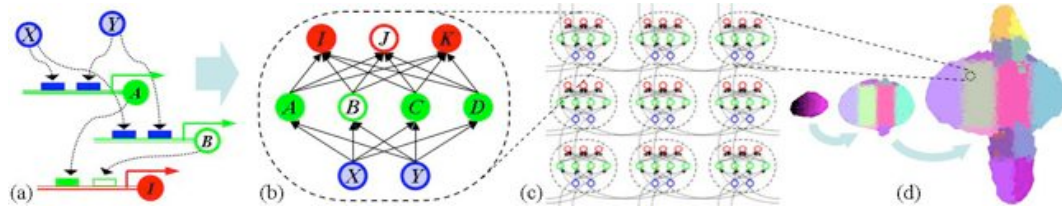


Figure 3.19: Travaux de René Doursat sur l'embryogénèse [Doursat 2007b]. La forme est créée à partir d'un réseau régulateur de gènes.

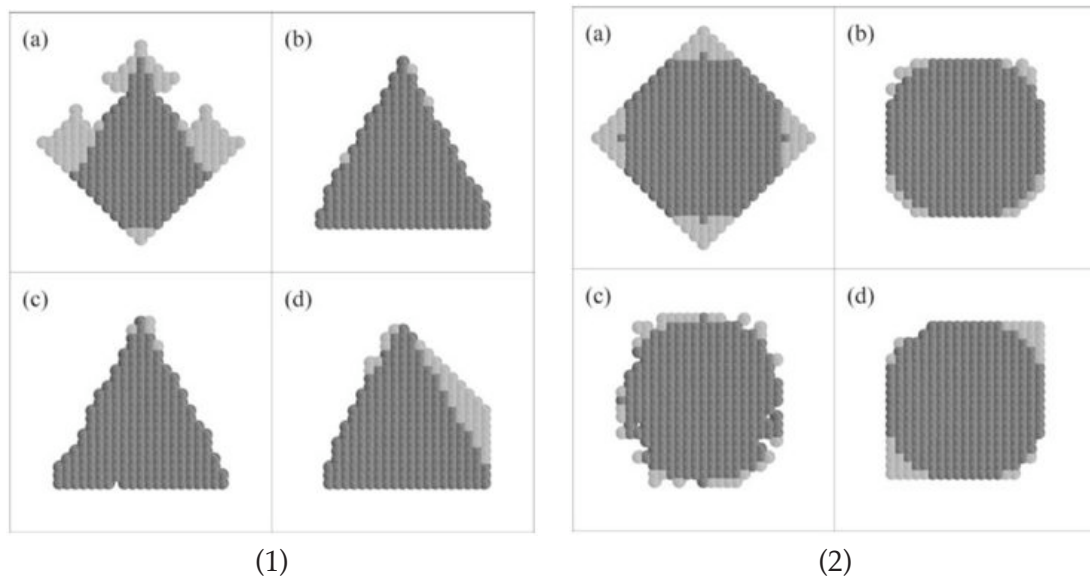


Figure 3.20: Travaux d'Yves Duthen et d'Arturo Chavoya [Chavoya and Duthen 2007b]. Développements cellulaires optimisés par algorithme génétique permettant de générer un triangle ou un cercle à partir de voisinages différents : (a) Von Neumann, (b) Moore, (c) 2-Radial et (d) Margolus. Ces primitives pourraient par exemple servir de briques élémentaires pour la construction de créatures artificielles.

(Fig. 3.21.c).

3.3 Synthèse

Afin d'établir les intérêts et les perspectives de l'apport des différents travaux, nous en établissons une synthèse. Par la suite, celle-ci orientera et justifiera nos choix pour le modèle que nous allons proposer.

Étant donné qu'il est difficile de reproduire les résultats de la plupart des auteurs, il existe peu de comparaisons quantitatives. Cela est dû au fait que les résultats obtenus ne sont le plus souvent jugés que visuellement. De plus les résultats

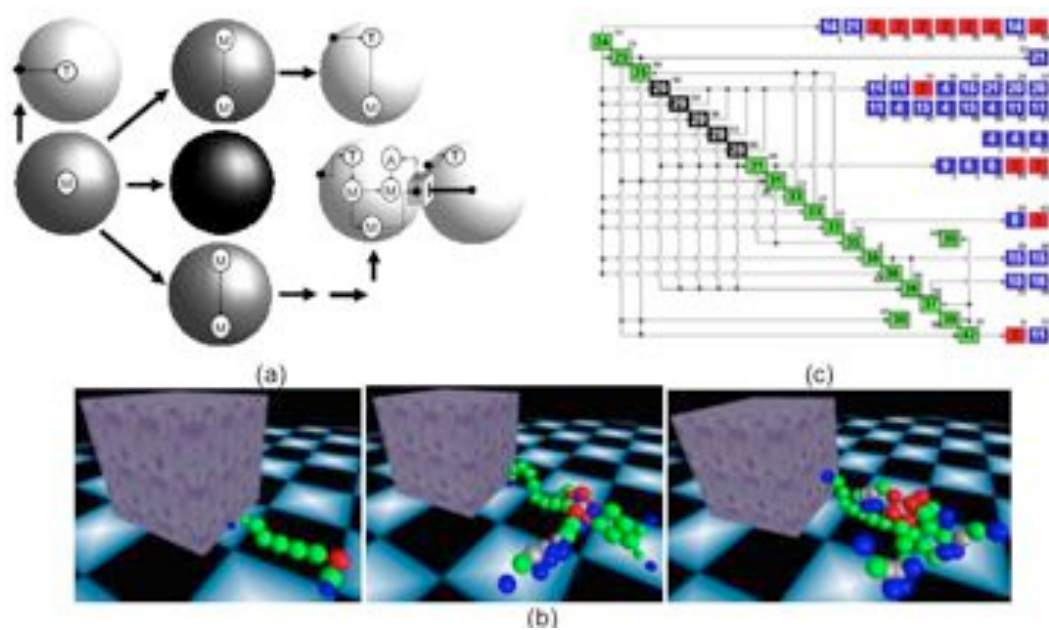


Figure 3.21: Les créatures de Bongard qui s'inspire de l'ontogenèse artificielle [Bongard and Pfeifer 2003]. (a) Les créatures se développent grâce à un réseau régulateur de gènes. (b) Trois différentes créatures issues de l'évolution qui poussent un cube. (b et c) En rouge, on peut observer que les neurones se sont développés essentiellement qu'en une seule zone du corps.

dépendent de nombreux paramètres et des environnements utilisés. Nous proposons donc de comparer les travaux par le biais de tableaux de caractéristiques et à travers différents critères comme le type de contrôleur, de morphologie et d'environnement employés.

3.3.1 Morphologie et contrôleur des créatures 3D

On peut remarquer dans le tableau récapitulatif que les contrôleurs sont exclusivement basés sur des réseaux de neurones (Tab : 3.1). Les réseaux de neurones sont très performants lorsqu'il s'agit d'approcher une fonction mathématique et ils peuvent facilement composer des fonctions sinusoïdales pour produire un signal qui actionne les effecteurs des créatures. Leur utilisation est aussi due au fait que Karl Sims les a employés et que beaucoup de travaux ont pour objectif de refaire ses créatures ou de s'en inspirer. Néanmoins, il peut être intéressant d'envisager des alternatives à ce type de contrôleur afin de diversifier les méthodes pour explorer de nouvelles pistes de recherche et peut-être permettre d'obtenir de nouveaux résultats. De plus, il est important dans une optique d'application à la robotique de considérer les types de contrôleurs adaptés à une motorisation.

Le tableau (Tab : 3.1) montre que les morphologies utilisent par contre des représentations différentes, souvent liées aux performances du moteur physique ou aux contraintes mécaniques. Ainsi en robotique, Lipson et Hornby emploient des éléments 1D qui facilitent la construction des robots. Pour ce qui est des morphologies à base de cylindres, si elles permettent d'obtenir de meilleures performances lors des calculs des interpénétrations, elles présentent l'inconvénient de produire des morphologies peu stables¹⁸ en comparaison avec les parallélépipèdes rectangles. Pour les autres types de morphologies, personne n'a pour l'instant tenté de produire des structures déformables pourtant plus réalistes. Mais ce type de morphologie sera certainement utilisé dès que les puissances de calculs le permettront¹⁹.

L'embryogenèse sera certainement l'une des prochaines voies employée par les chercheurs car elle permet de modéliser le développement de créatures artificielles. Ainsi l'embryogenèse rapprochera les créatures artificielles de celles déjà existantes dans la nature. Elle sera certainement utile dans la confection des robots modulaires

¹⁸En effet, lors d'une collision avec un plan, un parallélépipède rectangle offre généralement plus de points d'appui qu'un cylindre. Ainsi une morphologie composée de parallélépipèdes rectangles permet une plus grande stabilité.

¹⁹Néanmoins, il est tout de même possible suite à une évolution d'habiller les créatures avec une structure déformable et élastique.

et reconfigurables. Ainsi cette approche est prometteuse si on veut avoir des créatures artificielles, voir des robots, avec la capacité de s'auto-réparer car l'embryogenèse prend généralement cette caractéristique en compte dans ses modèles.

3.3.2 L'environnement

On peut constater que les environnements dans lesquels les créatures évoluent sont souvent constitués d'un simple sol plat. De ce fait, comme nous l'avons vu dans l'état de l'art, il serait souhaitable de les complexifier afin d'obtenir des comportements de créatures plus complexes. De plus, on peut constater qu'il y a peu de tests proposant d'évaluer l'aptitude des créatures (Tab : 3.2). Ainsi nous proposons dans nos expériences de nouveaux environnements auxquels sont confrontées nos créatures pour montrer leur capacité d'adaptation.

Pour ce qui est des écosystèmes, il est très difficile d'évaluer leur complexité bien qu'on puisse observer des stratégies d'adaptations assez variées. Il serait donc intéressant d'envisager la création d'outils permettant la mesure de leur complexité de manière plus objective afin obtenir de meilleures évaluations des stratégies d'adaptation qui en découlent.

Une autre constatation est que le parallélisme est peu employé dans l'évolution de créatures artificielles. Bien que Karl Sims ait utilisé une machine multiprocesseurs l'évolution n'est restée concentrée que sur une seule population. Il serait intéressant d'envisager des évolutions issues de populations multiples afin d'assurer une grande diversité de solutions. Ainsi il serait possible d'utiliser des écosystèmes composés d'îlots au lieu d'un ensemble de populations.

3.3.3 Bilan

La reproduction des résultats de Karl Sims qui s'est retiré de la scène scientifique donnant peu de détails de son modèle, semble être une étape décisive avant que d'autres travaux prennent des directions différentes. Cependant, les puissances de calculs, la démocratisation de la robotique et la venue de nouvelle discipline comme l'embryogenèse artificielle semblent montrer des perspectives prometteuses pour les créatures artificielles.

Ainsi nous avons orienté notre choix vers un nouveau type de contrôleur afin de nous démarquer des travaux précédents et afin d'obtenir des fonctions plus complexes que des sinusoides. Pour ce qui est de la morphologie, il aurait été souhaitable d'envisager des mécanismes basés sur une embryogenèse artificielle pour générer

nos créatures, mais au moment de notre choix cela était encore difficilement envisageable²⁰ et semblait être une voie trop complexe pour une première approche du problème. En ce qui concerne, l’environnement nous avons tenu à le complexifier afin d’évaluer nos créatures dans des situations difficiles. Cependant il aurait peut-être été plus intéressant de réaliser un écosystème complet en 3D malgré la capacité de calcul que cela impose. De ce fait, nous présentons en perspective un modèle d’écosystème et proposons des outils pour en mesurer la complexité.

Auteurs	Project	Moteur	Morphologie	Contrôleur
Sims		Personnel	Graphe orienté et parallélépipède rectangle	Réseau de neurones
Komosinski	Framsticks	Personnel	Stick	Réseau de neurones
Lipson	Golem	Personnel	Tubes et pistons	Réseau de neurones
Hornby	Genre	Personnel	L-System	Réseau de neurones
Taylor		MathEngine	Cylindres	Réseau de neurones
Miconi		ODE	Graphe orienté et parallélépipède rectangle	Réseau de neurones
Chaumont		ODE	Graphe orienté et parallélépipède rectangle	Réseau de neurones

Table 3.1: Travaux sur les créatures artificielles 3D. Comparaison des moteurs physiques, de la morphologie et du type de contrôleur. On retrouve l’utilisation des réseaux de neurones dans les principaux travaux du domaine. Il faut aussi noter que depuis l’apparition et la démocratisation de moteurs physiques commerciaux ou gratuits, de moins en moins d’auteurs développent leur propre moteur.

²⁰En effet, il semblait personnellement difficile de trouver un modèle cohérent avec un grand nombre de cellules interagissant dans un environnement physique.

auteurs	environnement					créature										application				évaluation (points forts)					
	représentation		écosystème	fitness	naturelle	morphologie					aptitude					animation	vie artificielle	biologique	robotique	environnement	physique	mouvement	morphologie	diversité	stratégie
	2D	3D				programme	cellule	bloc	embryogenèse	nager	rampier	marcher	sauter	voler	combattre										
Sims		x		x				x			x	x				x	x			3	5	5	4	5	5
Lipson		x		x				x			x							x		3	5	4	4	4	4
Hornby		x		x				x			x						x		x	3	4	4	5	5	4
Bongard		x		x			x	x	x		x						x			4	4	4	5	4	5
Miconi		x		x				x			x						x	x		3	5	4	4	4	4
Shim		x		x				x					x		x		x	x		3	5	4	4	4	3
Taylor		x		x				x	x								x	x		3	5	4	3	4	3
Chaumont		x		x				x			x					x	x			4	5	4	4	4	4
Ray		x						x									x			-	5	3	5	5	-
Framsticks		x	x	x	x			x	x					x	x	x		x		5	4	3	5	5	3
Ventrella	x		x	x				x		x							x			5	3	4	4	4	5
Life Drop	x		x	x				x		x							x	x		5	3	3	5	5	5
Tierra			x	x	x											x	x			3	-	-	-	3	5
Avida			x	x	x											x	x			3	-	-	-	4	5

Table 3.2: Dans ce tableau, nous comparons sur différents critères les principaux travaux sur les créatures artificielles. Il s'agit de critères et d'avis qui sont personnels et qui tentent d'être objectifs. Une notation de 1 à 5 permet de juger les points forts et les insuffisances des différents modèles. Ainsi les notes de 1 à 2 montrent que le modèle est insuffisant dans ce domaine, 3 que c'est satisfaisant mais peu intéressant, 4 montre que c'est une caractéristique intéressante, 5 un point fort et "-" qu'on ne peut évaluer ce critère. Ainsi ce tableau montre que l'évolution de créatures artificielles est peu employée pour la robotique. À noter aussi qu'il y a encore peu d'écosystèmes en 3D, peu de travaux utilisant l'embryogenèse. De plus, on peut remarquer que les créatures sont toujours évaluées dans les mêmes domaines sans que leur environnement ne soit plus complexe. Aucune simulation n'a réussi à faire émerger des créatures bipèdes avec la faculté de marcher. On peut aussi noter, bien que les travaux de Karl Sims soit parmi les premiers et date de 1994, qu'ils dominent encore le domaine pour de nombreux critères.

Modèle proposé

Dans ce chapitre, nous présentons notre modèle de créatures artificielles. Il a la particularité d'utiliser un nouveau type de contrôleur basé sur la composition de *patterns* de fonctions mathématiques. Ces travaux sont décrits dans les articles suivants [Lassabe et al. 2007b; Lassabe et al. 2006; Lassabe et al. 2007a; Lassabe et al. 2008].

4.1 L'environnement

Au cours de l'état de l'art, nous avons pu discerner plusieurs types d'environnements : les ressources d'un ordinateur pour le projet Tierra [Ray 1991a], des environnements 2D pour des écosystèmes tel que Gene Pool de Ventrella [Ventrella 2005] ou encore 3D par exemple dans les travaux de Karl Sims [Sims 1994a]. Le choix de l'environnement conditionne le type de créatures que l'on peut obtenir. La réciproque est aussi vraie, obtenir un certain type de créature ne peut être réalisé que dans des environnements particuliers. Cette dualité montre le lien qui existe entre les créatures et l'environnement dans lequel elles sont susceptibles d'évoluer. Par exemple, les ressources d'un ordinateur comme la mémoire, le processeur sont des environnements adaptés à l'évolution de programmes mais pas à l'évolution de créatures physiques 2D ou 3D. On peut alors se poser les questions suivantes :

- pour un environnement donné, quel est l'ensemble des créatures que l'on peut obtenir ?
- pour une créature donnée, quel est l'environnement minimal nécessaire à son évolution ?

Dans le cas de créatures issues de mécanismes d'évolution, l'environnement est une contrainte à respecter, le choix de l'environnement guide donc l'évolution vers certains types de solutions. Si on sélectionne les créatures pour leur aptitude au

déplacement, leur morphologies et les contrôleurs produits ne seront pas les mêmes suivant qu'on utilise un environnement simulant un milieu aquatique, un sol plat ou un sol irrégulier. L'environnement est donc une contrainte et un moyen de sélection au même titre que la fonction d'évaluation.

4.1.1 Environnement proposé

En ce qui concerne le choix de notre environnement, pour faire évoluer nos créatures nous nous orientons vers un environnement 3D simulant les lois de la physique. En effet, un environnement 3D est plus proche du monde réel. Il permet la simulation d'écosystème et donne en plus la possibilité de simuler des robots contrairement à un environnement 2D qui ne permet que de simuler des mini-écosystèmes. Cependant un environnement 3D nécessite de plus grandes capacités de calculs mais celles-ci deviennent de plus en plus accessibles.

Par rapport aux travaux que nous avons listé dans l'état de l'art, nous nous orientons vers des environnements incorporant des contraintes différentes de celle proposées par Karl Sims et ses successeurs. En effet nous pensons qu'il est possible d'obtenir des créatures adaptées à des environnements plus complexes. Par exemple, nous proposons d'évaluer le déplacement de créature en les confrontant non plus à un sol plat mais à la montée d'escaliers ou au franchissement de tranchées. Nous voulons aussi nous orienter vers des environnements plus dynamiques en faisant interagir les créatures avec des objets mobiles. Nous proposons donc d'observer les solutions obtenues si les créatures ont pour contrainte de se déplacer avec un *skateboard*.

4.2 Le moteur physique

L'implémentation d'un moteur physique est une tâche difficile qui requiert une grande quantité de travail. Avant les années 2000, il pouvait être courant de réaliser son propre moteur physique car ceux qui étaient proposés n'avaient pas forcément les qualités recherchées étaient trop spécialisées. En 2000, Tim Taylor recense les principaux travaux sur les créatures artificielles ainsi que les moteurs physiques utilisés [Taylor and Massey 2000]. Il dénote beaucoup de problèmes d'instabilité des modèles dans certaines conditions telles que les interpénétrations d'objets par exemple¹. Au moment de notre choix, *ODE, open dynamic engine*, semblait être le moteur le mieux

¹Cela reste vrai en 2007.

documenté et offrait les fonctionnalités que nous recherchions, du moins parmi les moteurs non commerciaux. Il présente de plus l'avantage d'être open source et a souvent été employé pour la réalisation de créatures artificielles ou de simulations en robotique.

4.2.1 Le moteur physique proposé

Notre choix s'est donc porté sur *Breve* réalisé par Jon Klein [Klein 2003], un environnement basé sur *ODE*. Cet environnement est dédié principalement aux simulations de vie artificielle et permet de réaliser des systèmes particuliers, des systèmes multi-agents ou des simulations physiques tout en proposant un ensemble de primitives de haut niveau pour gérer les simulations de corps articulés. De plus, *Breve* inclut un ensemble de fonction de bas niveau pour gérer le moteur physique : définition des couples, des forces appliquées, de la gravité, des détections des collisions... Par exemple, *Breve* dispose déjà de beaucoup fonctionnalités utiles à la simulation d'écosystèmes comme la quantité de lumière perçue par un objet que nous planifions d'utiliser dans nos futures expériences. En ce qui concerne les instabilités liées aux interpénétrations d'objets, nous avons choisi de désactiver la détection de collisions entre parties d'une même créature. Bien que dans certains cas, cela puisse paraître moins réaliste, cela nous permet d'obtenir des solutions que l'on ne trouverait pas autrement, d'améliorer la stabilité des simulations et de réduire significativement les temps de calculs.

4.2.2 L'évolution des moteurs physiques

Depuis la démocratisation de la physique dans les jeux vidéos, de nombreux moteurs physiques existent. Ils ne sont pas tous adaptés à la simulation physique de précision, mais sont optimisés pour fonctionner en temps réel. Il serait donc intéressant de reconsidérer le choix du moteur physique car leur évolution est constante depuis ces trois dernières années surtout avec l'apparition d'un nouveau type de carte dédiée aux calculs physiques². Le moteur *Breve* utilise son propre langage scripté interprété, *Steve*, qui est avantageux lorsque l'on veut faire un prototype de son application mais qui devient un inconvénient si on veut accéder à toutes les ressources du moteur physique qui lui est écrit en langage C. En effet, il devient fastidieux de réaliser toutes les interfaces car *Breve* n'implémente pas tous les types de données (Fig. 4.1). Un autre l'inconvénient de *Breve* est le manque de contrôle sur les fonctionnalités

²Appelé PPU (Physics Processing Unit). Exemple : la carte *physiX* d'Ageia (Nvidia)

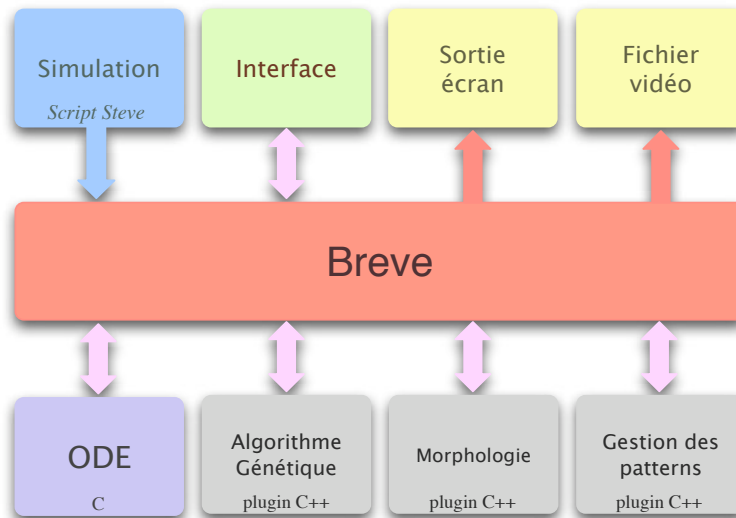


Figure 4.1: Architecture de notre application : En gris les plugins C++ qui gèrent la morphologie et le contrôle des créatures ; le cadre *simulation* contient le script en langage *Steve* qui initialise l'application et contrôle la simulation ; l'interface permet d'interagir avec les créatures. La modularité permet aux plugins d'être réutilisables dans une autre architecture, ce qui permettrait de s'affranchir de *Breve* et d'utiliser un environnement différent ou directement un moteur physique.

et les paramètres du moteur physique. S'il est possible de réaliser une application avec quelques créatures, il est difficile pour l'instant d'envisager d'utiliser *Breve* pour réaliser un écosystème complet qui compterait un grand nombre d'objets en interaction. De plus il est difficile de déboguer le langage *Steve* par manque d'outils adaptés, surtout lorsqu'on fait appel à des fonctions C++ provenant d'un plugin. Cependant cet environnement dédié aux simulations de vie artificielle est lui aussi en constante amélioration et son auteur réalise régulièrement de nouvelles mises à jour et versions.

4.2.3 L'implémentation

Pour ce qui concerne l'implémentation, notre programme est écrit en C++ et fonctionne dans *Breve* sous forme de plugins (Fig. 4.1). De ce fait, notre programme n'est pas dépendant de *Breve* et peut être utilisé comme une librairie avec un autre moteur physique. Ainsi l'architecture comporte trois plugins principaux qui implémentent l'algorithme génétique, la morphologie des créatures et la gestion des *patterns*. Cependant un inconvénient est qu'il est nécessaire de réécrire en langage *Steve* chacune des classes C++ que l'on veut interfacer.

4.3 La morphologie

La morphologie détermine la forme de la créature ainsi que la façon dont elle s'articule. Dans l'état de l'art, nous avons vu qu'il existait plusieurs solutions pour modéliser les morphologies, comme les L-systems employés par Hornby [Hornby and Pollack 2001], les *graftals* par Karl Sims [Sims 1994a] ou l'ontogenèse artificielle employée par Bongard [Bongard and Pfeifer 2003]. L'ontogenèse est l'une des voies les plus prometteuses pour les créatures artificielles. En effet, la modélisation et l'évolution de créatures artificielles a en partie pour objectif de trouver des solutions pour la morphologie de robots afin que ceux-ci puissent acquérir des caractéristiques du vivant. Si bien que les études sur l'ontogenèse permettront certainement dans un futur plus ou moins proche de modéliser puis de réaliser des robots avec la faculté de se développer et de s'auto-réparer.

4.4 Morphologie proposée

Pour obtenir nos créatures artificielles 3D utilisant un modèle physique, nous sommes orientés vers des créatures dont la morphologie n'évolue pas au cours de la simulation³. En ce qui concerne la représentation (Génotype) de la morphologie, les expériences de Karl Sims n'étant pas reproduites, nous avons supposé que son modèle était suffisante pour un grand nombre d'applications et nous sommes orientés donc vers les *graftals* qu'il employait. Ils ont l'avantage de produire des formes modulaires ce qui facilite la réutilisation des structures. Le génotype de nos créatures est donc représenté par un *graftal* qui est ensuite interprété pour réaliser un phénotype composé primitives géométriques.

Pour les primitives géométriques, nous proposons d'utiliser des parallélépipèdes rectangles. Les parallélépipèdes sont des primitives simples, ce qui simplifie les calculs de détection des collisions car leurs forme ne nécessite pas l'utilisation de boîtes englobantes⁴ Ils permettent d'obtenir des structures stables pour faciliter les déplacements et surtout éviter que les créatures se renversent. Il aurait pu être envisagé d'utiliser des tubes de faible diamètre, que l'on désigne aussi par *sticks*, composés par des L-systems. Mais les L-systems, s'ils permettent d'obtenir des créatures très modulaires, nécessitent de créer un moteur physique spécifique afin qu'il puisse

³La morphologie est donc figée lors de l'évaluation et n'évolue que par le biais des opérateurs génétiques

⁴Les détections de collisions avec un cylindre nécessitent encore moins de calculs mais les cylindres offrent moins de stabilité et d'appuis qu'un parallélépipède.

gérer un grand nombre d'éléments⁵ comme a fait Hornby pour des créatures qui comportent parfois plus de 600 segments [Hornby and Pollack 2001].

4.4.1 Le génotype : Graftal

Nos travaux utilisent donc une morphologie basée sur les *graftals* de Karl Sims [Sims 1994a; Sims 1994c; Sims 1994b]. Ils présentent l'avantage d'avoir été largement décrits et utilisés pour les créatures artificielles [Taylor and Massey 2000; Miconi and Channon 2005; Chaumont et al. 2007]. Le génotype représente un graphe dont les noeuds décrivent les blocs qui composent la créature et dont les arcs indiquent leur agencement. Pour chaque créature, nous définissons un noeud comme racine de notre graphe (fig: 4.2). Chaque noeud contient l'information d'un bloc. Les blocs sont représentés par des solides. Les données des solides sont leur taille en x , y et z , la couleur et le lien de parenté avec un autre bloc. Seul le bloc racine n'a pas de parent. Les liens entre deux noeuds sont représentés par un type de joint dans le phénotype. Un bloc pourra être relié sur lui-même, ce qui permet la récursivité de l'expression d'un noeud. La modularité est très visible comme le montrent nos résultats. Nous avons choisi de ne pas représenter explicitement la symétrie dans nos créatures comme Karl Sims l'avait fait dans ses travaux afin de voir si l'évolution peut la faire apparaître. Le type d'articulation entre les blocs est sélectionné aléatoirement, l'articulation peut être sur 1, 2, ou 3 axes.

4.4.2 Génotype et phénotype

Le génotype est construit aléatoirement. On détermine tout d'abord un nombre aléatoire de noeuds dont le *graftal* sera constitué. On détermine ensuite toujours aléatoirement les caractéristiques des blocs, c'est-à-dire leur taille, leur couleur. Un des noeuds est choisi ensuite comme racine puis les autres noeuds sont connectés un par un à un noeud du graphe et l'on indique pour le phénotype les coordonnées des points qui relieront les blocs par l'intermédiaire d'une articulation. Ensuite, une capacité de récursion est attribuée pour chacun des noeuds du graphe. Afin de limiter le nombre de blocs lors de la création du phénotype, nous limitons la profondeur de la récursivité ainsi que le nombre de blocs sur lesquels nous appliquons la récursivité. Le phénotype est exprimé en fonction du génotype, pour cela le graphe est interprété récursivement depuis son bloc racine. C'est le phénotype qui sera ensuite évalué lors

⁵En effet, les moteurs physiques génériques ne supportent pas des structures composées d'un grand nombre d'éléments.

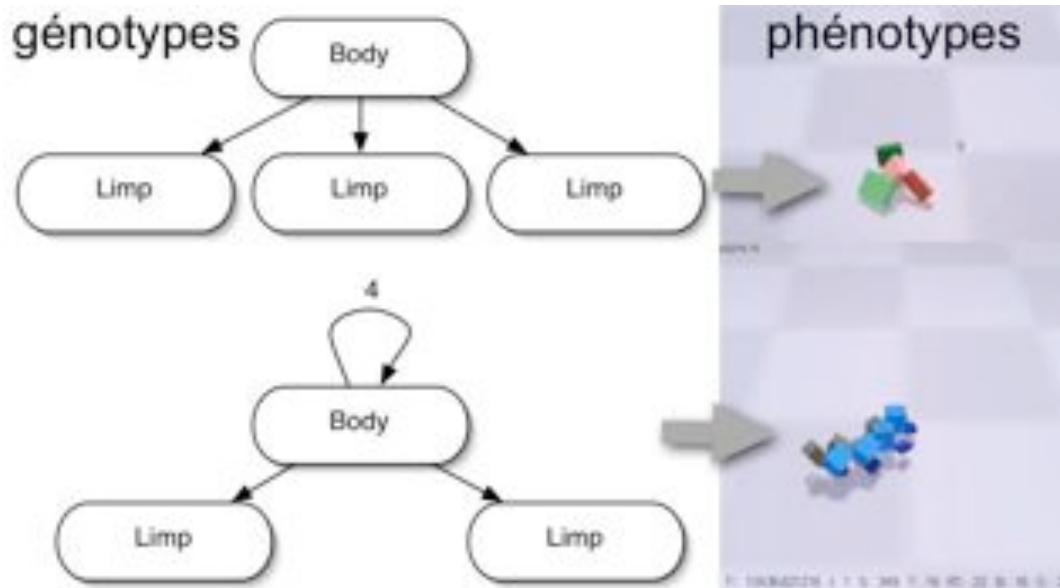


Figure 4.2: Le génotype d'une créature est représenté par un *graftal*, son phénotype est ensuite évalué lors des simulations dans l'environnement physique *Breve*.

des simulations. La figure (Fig. 4.2) montre deux exemples simples de génotypes exprimés sous forme de phénotypes.

4.5 Le contrôleur

Il y a plusieurs méthodes pour représenter un contrôleur, nous pouvons citer parmi les plus connues : un ensemble de fonctions logiques, une machine à états finis, un programme, un ensemble d'équations différentielles ou un réseau de neurones. Dans le domaine des créatures artificielles, les réseaux de neurones sont principalement utilisés (Tab. 3.1). L'apprentissage se fait non pas durant la simulation, mais lors de l'évolution du graphe. Soit en ajustant aléatoirement les poids des entrées des neurones s'ils utilisent une fonction d'activation. Soit en modifiant leur fonction mathématique s'il s'agit d'une composition de fonctions comme dans le cas des réseaux utilisés par Karl Sims [Sims 1994a].

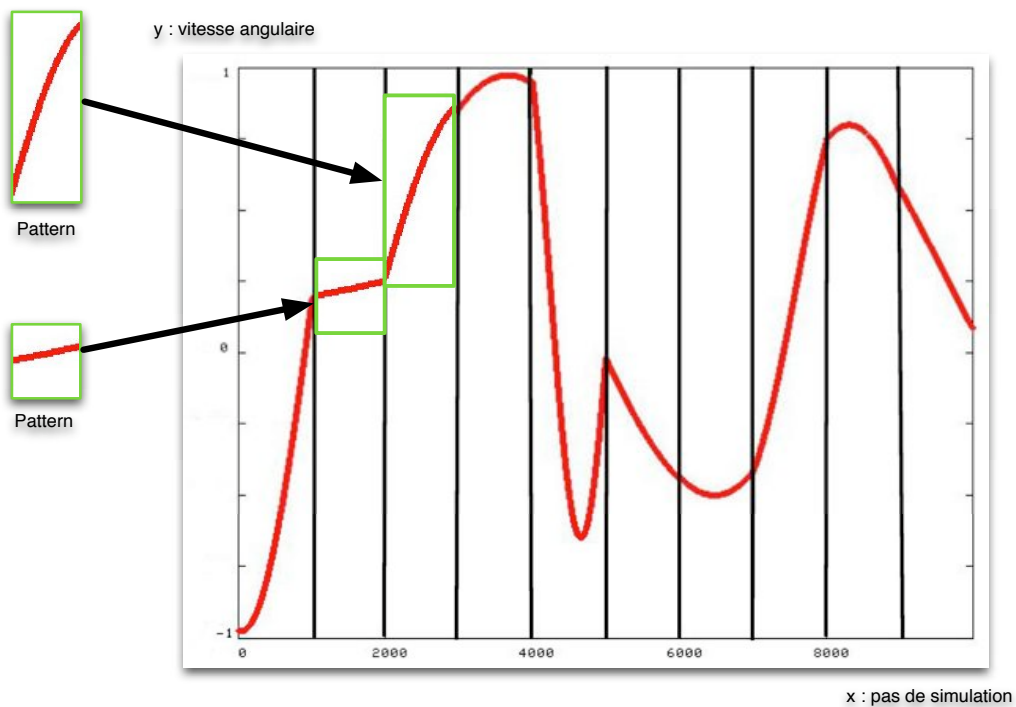


Figure 4.3: Signal composé de pour le contrôle d'un muscle d'une créature. (axe x : pas de simulation, are y : vitesse angulaire.).

4.5.1 Notre modèle de contrôleur : la composition de *patterns*

Les contrôleurs doivent activer les articulations correctement tout au long de la simulation car des erreurs⁶ peuvent provoquer des instabilités au niveau du moteur physique. Le signal généré se doit d'être continu et de varier au cours du temps en tenant compte des valeurs issues des capteurs. Le signal est généralement une courbe sinusoïdale [Komosinski 2000] ou une composition de fonctions périodiques. Nous proposons comme alternative de composer un signal à partir de *patterns* prédéfinis (Fig. 4.3). Ainsi les *patterns* sont activés lors des simulations par un système inspiré des classifieurs. Comme pour les réseaux de neurones cités précédemment l'apprentissage des classifieurs se fait durant l'évolution des créatures et non durant les simulations. Dans le même esprit, Wilson utilise aussi des systèmes de classeurs afin approximer des fonctions [Wilson 2002].

⁶En effet, pour les cas limites il devient difficile aux moteurs physiques de calculer avec précisions les positions et les forces appliquées aux créatures ce qui provoque des comportements non-réalistes.

4.5.2 L'assemblage de *patterns*

Pour cette approche, le problème principal est de conserver la continuité du signal tout en sélectionnant les *patterns* les plus adaptés. Nous commençons par définir une base de donnée d'un millier de *patterns* générés aléatoirement à partir de fonctions mathématiques telles que des fonctions périodiques ⁷. Nous définissons alors une règle de composition : deux *patterns* peuvent être additionnés pour composer une partie du signal de contrôle si seulement si les conditions suivantes sont satisfaites :

- Les *patterns* sont définis entre [-1,1] et les mêmes bornes sont définies pour le minimum et le maximum.
- Soient P_1 et P_2 deux *patterns*, P_2 peut être ajouté à P_1 , si :

$$(P_1(1) - P_2(-1)) < \varepsilon$$

où $\varepsilon = 0.01$

Le classifieur contrôle la sélection des *patterns* (Fig. 4.4). Les entrées du classifieurs proviennent des capteurs de la créature. En accord avec l'état des capteurs, un ensemble de règles pointant sur des *patterns* de la base permettent de composer le signal. Ce dernier est alors utilisé par les effecteurs pour contrôler les muscles de la créature (Fig. 4.5). Les règles appliquées durant la simulation sont rétribuées à la fin par l'évaluation donnée à la créature. L'avantage de cette méthode est qu'il est facile d'enrichir la base avec de nouveau *pattern* afin d'obtenir des signaux plus complexes. Pour remplacer les *patterns*, une autre approche pourrait être d'utiliser des fonctions mathématiques, lesquelles pourrait être générées par l'utilisation d'un programme génétique.

4.6 Application des *patterns* pour l'animation

Une des premières raisons pour lesquelles Karl Sims a voulu créer des créatures artificielles était l'animation [Sims 1994a]. À cette époque, comme nous l'avons vu dans l'état de l'art, beaucoup de travaux portaient sur la réalisation de contrôleurs pour animer des personnes comme par exemple ceux de van de Panne [van de Panne 1993]. L'animation de personnage utilise souvent la capture de mouvement (motion capture) afin d'être plus réaliste. Ces séquences pourraient être utilisées comme *patterns* afin

⁷Le détail de la génération d'un *pattern* est en annexe.

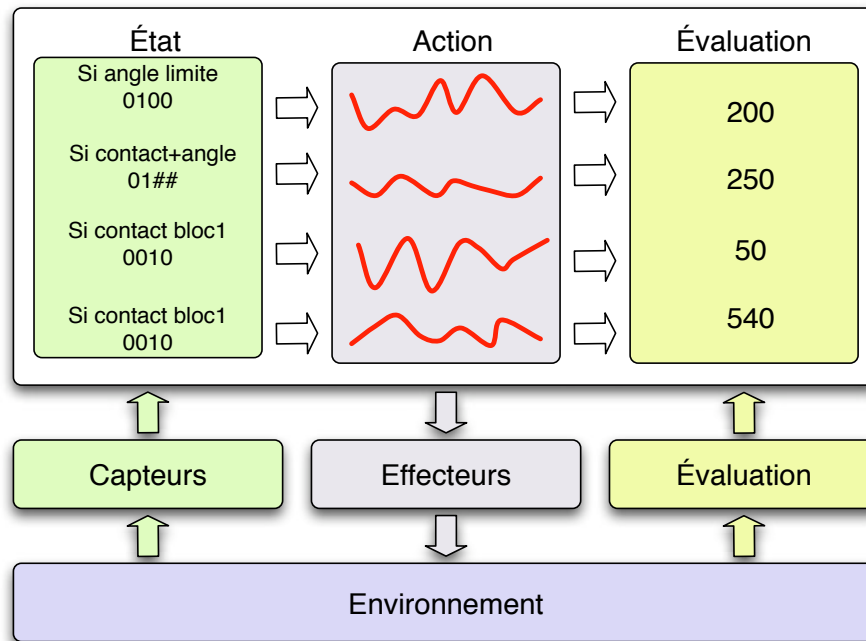


Figure 4.4: Chaque degré de liberté de la créature est contrôlé par un classeur. Il y a donc au moins un classeur par articulation. Chaque classeur est relié aux capteurs binaires des deux blocs articulés. Ainsi on peut compter un capteur binaire de toucher pour chaque bloc (deux au total) et deux capteurs binaires de limite d'angle qui indiquent si l'articulation est en butée. Suivant l'état de chacune des articulations un *pattern* est sélectionné. Il est possible qu'il y ait deux règles sélectionnées pour un même état, dans ce cas c'est la règle qui a la meilleure évaluation qui est sélectionnée. Les règles qui ont été activées au cours d'une simulation sont évaluées (l'évaluation est tout simplement égale à la valeur de *fitness* que la créature a reçue). Au niveau des capteurs le # est utilisé pour généraliser les états des créatures. L'évolution des *patterns* se produit uniquement lors de l'évolution des créatures. Ainsi des mutations peuvent transformer un signal. Lors des croisements des créatures, des croisements entre les classeurs (règles) interviennent entre les articulations qui sont au niveau des points de croisement du graphe. Si une règle correspondant à un état est manquante, elle est ajoutée aléatoirement.



Figure 4.5: La créature bouge grâce aux blocs bleus qui sont en mouvement, ce mouvement est dû à l'action d'un signal sinusoïdal. La créature avance en prenant appui sur un bloc pour s'allonger (a-c) et puis se replie avant de s'allonger à nouveau (e-d).

de générer les mouvements d'une personne. On pourrait ainsi trouver l'enchaînement des séquences qui permettrait par exemple la montée d'un escalier. On peut imaginer l'utilisation d'opérateurs afin d'ajuster les différentes séquences. Par évolution, on pourrait ainsi trouver un contrôleur utilisant des séquences d'animations réalistes. Il serait aussi possible de modifier la morphologie des personnages afin de leur faire correspondre les contrôleurs obtenus.

Afin de stocker moins de *patterns* et d'enrichir la base de *patterns*, il serait envisageable d'utiliser les symétries. Ainsi des opérateurs pourraient indiquer qu'on utilise les *patterns* symétriques. Cependant, il faudra toujours vérifier la continuité du signal composé.

4.7 Les opérateurs génétiques

4.7.1 La mutation

Les mutations sont réalisées sur le génotype. Toutefois, la procédure de mutation est adaptée à sa structure interne et portera donc sur le *graftal* sous-jacent. Chaque mutation a autant de chance d'être appliquée.

Les mutations possible sur des *graftals* que nous utilisons sont :

- Un nouveau bloc peut être ajouté aléatoirement au *graftal*.
- Un bloc peut être effacé aléatoirement à partir du graphe. Il est à ce moment nécessaire de vérifier si cette opération est possible. L'opération ne peut être appliquée au bloc racine car elle produirait l'effacement de tout le graphe. Le sous graphe connecté au bloc effacé est lui aussi effacé.
- Une mutation peut légèrement et aléatoirement déplacer ou recentrer ⁸ :

⁸Les points de contact d'un bloc peuvent être déplacés d'une distance de 30% de la longueur ou



Figure 4.6: Mutations opérées sur des créatures au cours d’une simulation en sélectionnant les plus aptes à se déplacer (a) La créature a une morphologie composée de six blocs bleus et six blocs blancs excentrés, son génotype ne comporte que deux noeuds dont l’un est récursif six fois. (b) Après quelques générations, les nouvelles créatures ne comportent plus que cinq blocs blancs et bleus. Une mutation a donc modifié la récursivité. De plus, on constate que les blocs blancs ne sont plus excentrés grâce aux mutations et les blocs bleus sont plus larges. (c) Quelques générations plus tard, les mutations ont encore réduit la récursivité et un nouveau bloc marron est apparu qui a amélioré les performances de cette créature.

- le point de contact du bloc fils en x ou y sur la surface du bloc parent.
- le point de contact du bloc parent en x ou y sur la surface du bloc fils.
- La taille en x , y ou z d’un bloc choisi aléatoirement peut changer.
- Le nombre d’itération d’un noeud peut être aléatoirement changé, incrémenté d’un ou décrétementé d’un.
- L’axe de rotation d’un joint peut être changé.

Si une mutation est effectuée sur un noeud récursif, la mutation affectera tous les blocs issus à partir de ce noeud, ce qui permet de conserver la modularité.

4.7.2 Croisement

Pour reproduire deux créatures, nous utilisons la méthode *graphing* décrite dans les travaux de Karl Sims [Sims 1994a; Sims 1994c]. Pour cela, nous sélectionnons aléatoirement un noeud dans le graphe de la première créature à croiser et un deuxième dans celui de la seconde puis nous permutons les sous-graphes qui ont pour racine ces noeuds (Fig. 3.2).

largeur de la surface d’un bloc autour du point de contact. Ils peuvent être recentrés n’importe où entre le centre de la surface et l’ancien point de contact.

Suite à une mutation, il est possible qu'une créature se retrouve constituée uniquement d'un bloc. Ces créatures sont éliminées car elle ne peuvent pas agir dans l'environnement. En plus du *grafting*, nous utilisons aussi comme croisement des *graftals* la méthode de *crossover* définie par Karl Sims pour la génération de ses créatures (Fig. 3.2).

4.8 Les problèmes de stabilité

De nombreux articles recensent des problèmes de stabilité causés par le moteur physique. Au début de la simulation, il est par exemple possible de voir passer la créature à travers le sol⁹ et tomber sans retenue sous l'effet de la gravité et ainsi obtenir un score important lors de l'évaluation. Cela est dû au fait que la créature est posée directement sur le sol en début de simulation et qu'il peut arriver qu'une partie de la créature soit interpénétrée par le bloc qui constitue le sol. Pour remédier à cela, les auteurs préconisent de laisser tomber les créatures d'une certaine hauteur afin que celles-ci ne soient pas initialement en contact avec le sol. Mais cela entraîne un autre problème, les créatures peuvent rouler ou rebondir sur le sol et ainsi obtenir une évaluation importante sans effort. Cela peut aussi compromettre l'évolution, car en roulant au début de la simulation des créatures issues d'une même souche peuvent se retrouver initialisées avec des situations différentes empêchant de reproduire les résultats de leurs ancêtres. Dans nos expériences, nous choisissons de calculer la hauteur moyenne des créatures afin de les faire tomber d'une hauteur assez basse. Ensuite nous diminuons la gravité afin d'en contrôler la chute. Les créatures se retrouvent ainsi toujours dans les mêmes conditions d'initialisation ce qui permet aussi une meilleure convergence.

⁹Le sol est généralement un simple plan 2D ou un rectangle de grande taille et de faible épaisseur.

Expériences et Résultats

Dans ce chapitre, nous proposons d'étudier l'évolution des créatures face à différentes situations comme ramper, marcher et pour des déplacements plus complexes comme la montée d'escalier et l'interaction avec des objets mobiles. Pour cela, nous utilisons le modèle de créatures décrit dans la section précédente. Les résultats présentés regroupent ceux des publications suivantes [Lassabe et al. 2007b; Lassabe et al. 2007a; Lassabe et al. 2006; Lassabe et al. 2008].

5.1 Les conditions initiales

La population de la simulation est initialisée avec cent individus ¹. Chaque génotype est créé aléatoirement en accordant un nombre maximal de blocs. Pour ce type de simulation, ce nombre est généralement égal à dix. Ce nombre peut librement augmenter par la mutation du nombre d'itération de chaque nœud prévue dans la définition de la modélisation de notre génotype ou par croisement. Chaque individu est ensuite évalué dans l'environnement durant 40s (40 000 pas dans le simulateur). La fonction d'évaluation utilisée pour évaluer les créatures est décrite dans chaque expérience. Pour le processus de sélection, nous utilisons le mode tournoi à trois individus². Le taux de croisement est de 70% de la population et celui de mutation de 10%. L'évolution est arrêtée quand la fonction atteint une valeur désirée ou après un certain nombre de générations déterminés pour chaque expérience. Les expériences convergent généralement entre 50 et 120 générations.

¹Les paramètres sont détaillés dans les annexes.

²Le choix de cette valeur s'appuie sur démarche empirique. Il faut toutefois noter que les expériences convergent tout aussi bien pour d'autres valeurs autour de trois.

5.2 La validations de expérimentations

À la fin d'une expérimentation, il est nécessaire d'analyser les résultats. Les défaillances³ d'une évolution peuvent avoir diverses origines. Il est possible que les objectifs définis par la fonction d'évaluation soient trop difficiles à atteindre ou qu'on n'ait pas assez de puissance de calcul. Dans ces cas-là, il est soit nécessaire de définir des objectifs moins ambitieux, soit redéfinir la fonction d'évaluation de manière à faire évoluer les solutions par palier, c'est-à-dire en récompensant les individus qui ne satisfont qu'une partie des objectifs. Il est aussi possible que la fonction d'évaluation ne décrive pas correctement l'objectif que l'on désire. Dans cette situation, l'évolution peut faire apparaître des solutions optimales mais qui ne sont pas désirées. C'est l'une des contraintes des systèmes évolutionnistes. Nous verrons un exemple de ce type de comportement dans le cas de la montée de l'escalier (Sec: 5.5). Afin de détecter, les simulations défaillantes, les évaluations des individus sont enregistrées sous forme de vidéos. On conserve les vidéos dont l'évaluation dépasse un certain seuil ainsi que le meilleur individu dans le cas où le seuil d'enregistrement de la fonction de *fitness* n'est pas atteint. L'évaluation de chaque individu est aussi enregistrée après chaque génération dans un fichier⁴. Durant la simulation, on constate pour de bonnes conditions⁵ que la moyenne de l'évaluation augmente. Ceci traduit le fait que la population progresse globalement vers l'objectif défini et que le patrimoine des meilleurs individus est bien transmis au reste de la population. La durée des expériences sur un *Mac Pro Bi-Xeon* peut varier de 3h à 12h de calculs suivant le type d'expérience, le nombre de blocs dont les créatures sont composées et la complexité des interactions avec l'environnement qu'elles engendrent. Dans la section suivante, nous décrivons l'ensemble des expériences que nous avons mis en place pour évaluer notre modèle.

5.3 Le déplacement sur un sol plat

5.3.1 Présentation de l'expérience

La première expérience est le déplacement sur un sol plat, c'est celle qui est la plus classique. Cette expérience est similaire aux travaux de Karl Sims [Sims 1994a],

³C'est à dire les cas où l'évolution n'a pas produit les résultats souhaités.

⁴On peut aussi connaître l'évolution de la fonction de *fitness* des meilleurs individus ainsi que l'évolution de la moyenne de la fonction de *fitness* de la population.

⁵Lorsque la fonction de *fitness* est bien définie et l'algorithme génétique bien paramétré. Ainsi l'augmentation de la moyenne de la fonction d'évaluation est un indicateur pour estimer si la simulation tend à converger.

qui l'a présentée le premier. Beaucoup de travaux, ont essayé de la reproduire [Lipson and Pollack 2000; Taylor and Massey 2000; Hornby and Pollack 2001; Miconi and Channon 2005; Chaumont et al. 2007]. Son but est d'optimiser le déplacement de créatures sur un sol plat en faisant évoluer leur morphologie et leur comportement.

Dans cette expérience, notre fonction de *Fitness* F n'est pas la distance parcourue mais la vitesse moyenne nécessaire après un temps t pour parcourir la distance entre le point départ et la position courante de la créature à ce temps t . $F = d/t$ où t est la durée de l'expérience. On utilise la vitesse moyenne car cela permet d'avoir des durées d'expérience différentes et de toujours pouvoir les comparer.

Chacune des créatures présentées est issue d'évolutions différentes. Il est très rare d'avoir deux créatures différentes et performantes au cours d'une même simulation⁶. Ceci peut-être résolu soit en utilisant des populations parallèles, soit en essayant de conserver une diversité dans la population. Les créatures de Sims sont elles aussi issues d'évolutions différentes.

5.3.2 Quelques créatures issus de l'évolution

Voici les différentes catégories de créatures que l'on obtient :

- Pour cette première expérience, un des meilleurs résultats que nous avons obtenu est un tripode (Fig. 5.1.a). Cette créature n'est pas symétrique. Elle est composée simplement de trois blocs qui constituent les pattes et d'un bloc représentant le corps. La modularité n'apparaît pas dans cette créature.
- D'autres créatures utilisent des stratégies efficaces en se déplaçant en crabe (Fig. 5.1.b). La modularité est très présente dans cette créature puisqu'elle réutilise quatre fois un même groupe de trois blocs et aussi un même groupe de contrôleurs. On peut remarquer que certains blocs agissent comme des pattes puis que le corps principal de la créature ne touche pas le sol. Parmi les créatures modulaires, certaines ont leur corps excentré par rapport aux positionnements de leurs pattes. Leur corps étant très fin cela leur permet de garder l'équilibre (Fig. 5.1.c). La symétrie n'étant pas activée, les créatures avec des pattes ont généralement des paires de pattes différentes mais qui sont presque symétriques.

⁶Car les créatures les plus performantes finissent par transmettre des variantes de leurs gènes à toute la population.

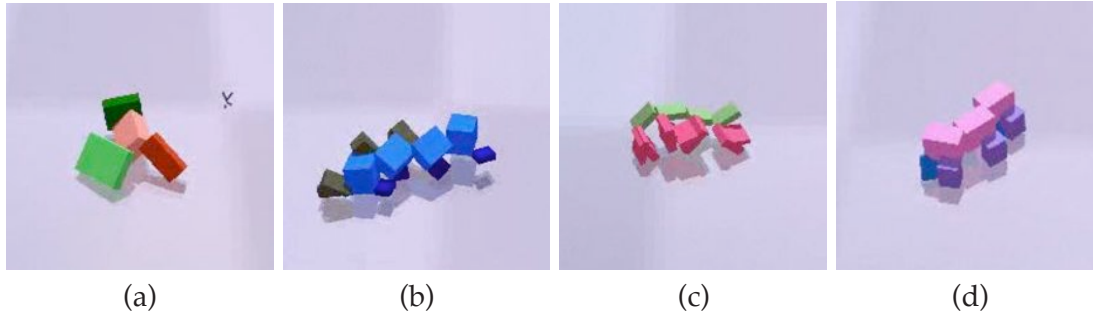


Figure 5.1: (a) Créature évolutionniste se déplaçant comme un tripode. (b) Créature se déplaçant comme un crabe. (c)(d) Créatures avec des morphologies modulaires.

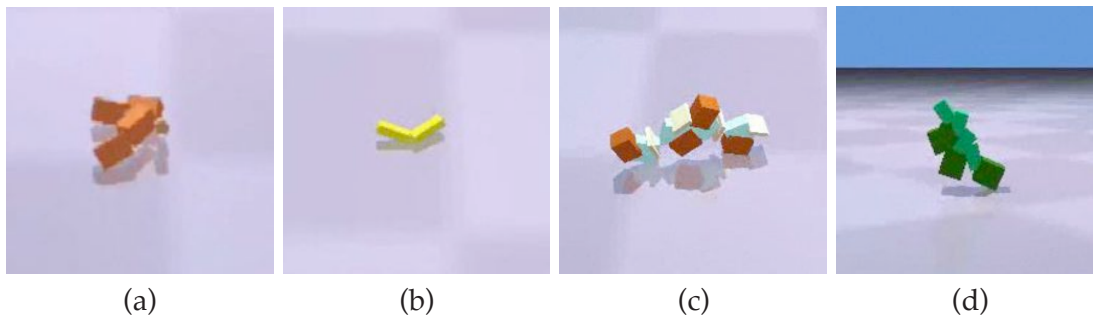


Figure 5.2: (a)(b) Créature roulant. (c) Créature ressemblant à un robot (d) Créature avec un profil humanoïde.

- Une autre catégorie de créatures roule, parmi ces créatures on en trouve une comparable à celle obtenue par Karl Sims (Fig. 5.2.a).
- L'interpénétration des blocs permet d'obtenir des résultats intéressants comme cette créature à la démarche saccadée comparable à celle d'un robot (Fig. 5.2.c).
- Une catégorie de créatures arrive à se redresser sur leurs pattes arrière afin de s'élaner vers l'avant et ainsi d'être plus efficaces. (Fig. 5.2.d).
- Une des créatures les plus originales commence à se déplier pour déposer une patte supplémentaire au sol et finit par entamer sa marche (Fig. 5.3.d).

5.4 Déplacement dans une direction spécifique

Dans cette expérience, nous gardons la même simulation et les mêmes paramètres, mais nous changeons la fonction d'évaluation. Nous définissons la fonction

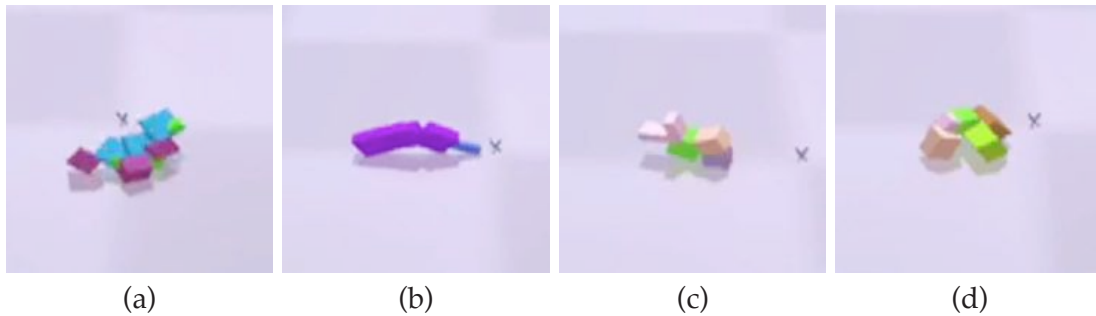


Figure 5.3: (a) Créature qui rampe en ondulant. (b) Créature ressemblant à un ver. (c) Créatures non-modulaires (d) Créature non-modulaires qui se déploie avant d’avancer.

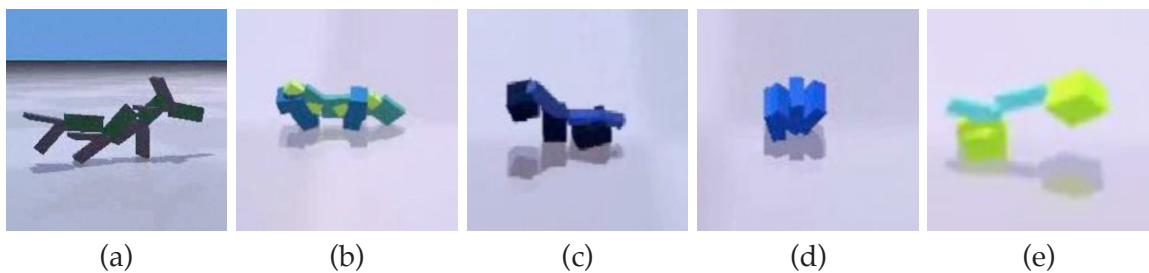


Figure 5.4: (a) Créature non-symétrique marchant dans une direction spécifique. (b)(c) Créatures modulaires et symétriques avec des sortes de pattes. (d) Créature qui roule sur elle-même. (e) Créature faisant de grands pas.

d’évaluation afin d’indiquer aux créatures d’aller dans une direction spécifique, par exemple, suivre l’axe x . La nouvelle fonction d’évaluation devient ainsi $F = |x|/t$

5.4.1 Analyse des résultats

Il semble qu’il soit plus difficile pour les créatures d’aller dans une direction spécifique, cela se traduit par une diversité plus restreinte par rapport à l’expérience précédente. Cependant, une caractéristique intéressante des morphologies des créatures obtenues est qu’elles sont composées de corps modulaires (Fig. 5.5). Cette modularité leurs donne une grande efficacité. Parmi les meilleures créatures, des genres de vers possèdent de longues pattes (Fig. 5.4(b-c)). Les créatures n’ont pas pour autant de capteur qui leur indique la direction à prendre, c’est par le biais de l’évolution qu’elles prennent la bonne direction.

On aurait pu imaginer que les résultats soient les mêmes que dans l’expérience précédente. En fait, les créatures sont optimisées pour aller dans une direction et cela se manifeste par une parfaite symétrie des meilleures créatures (Fig. 5.4(b-d)).

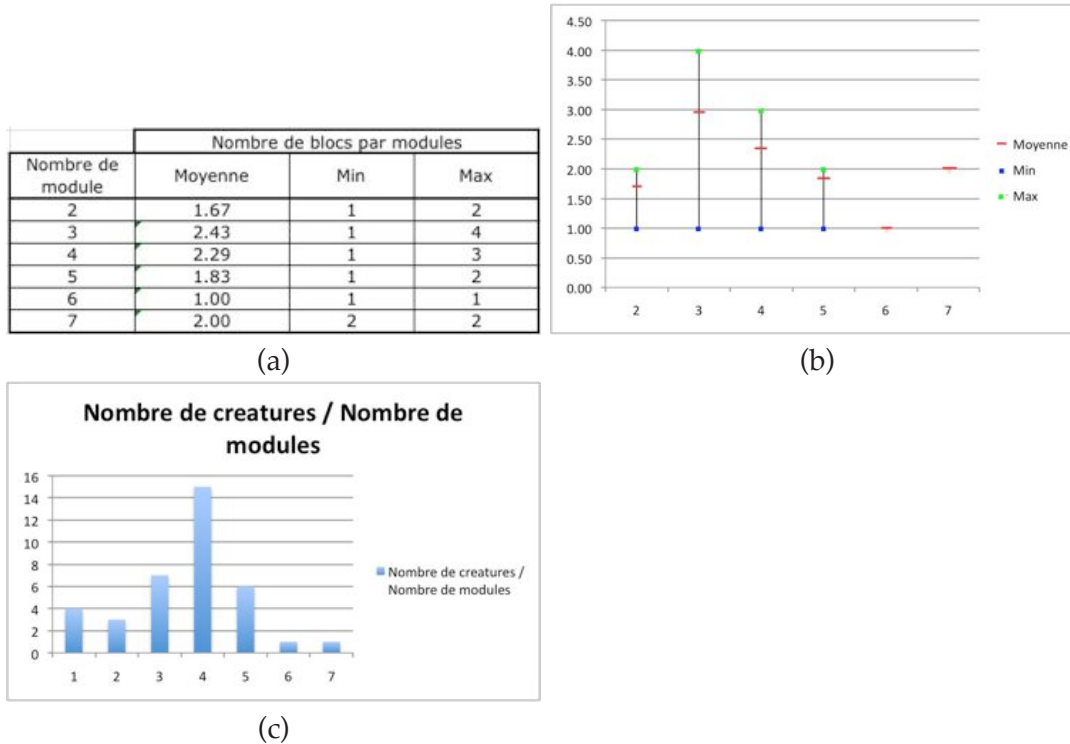


Figure 5.5: (a-b) Moyenne de nombre de blocs par modules en fonction du nombre de module (Réalisé sur 36 expériences). (c) Nombre de créatures en fonction du nombre de modules quelles possèdent.

Parfois, la modularité offre des stratégies différentes, certaines créatures font de larges pas pour se déplacer plus vite (Fig. 5.4(e)). Il faut noter que la symétrie n'est pas spécifiée dans les gènes, elle provient d'une mutation qui aide à recentrer les blocs. D'autres expériences pourraient être menées afin de voir l'impact de cette mutation par exemple en changeant la probabilité de mutation et en regardant si l'on obtient de résultats plus rapidement. Cela permettrait de montrer réellement l'implication de cette mutation dans la symétrie des créatures.

5.5 La montée d'un escalier

5.5.1 Présentation de l'expérience

Dans les travaux présentés dans l'état de l'art, les environnements dans lesquels les créatures évoluent sont plats et dépourvus d'obstacles. Pour augmenter graduellement la difficulté, nous essayons de voir si l'évolution peut générer une créature pouvant monter un escalier. La difficulté est définie par la hauteur des marches, le type

d'escalier étant fixe. Dans ce test, cette dernière est définie pour représenter la moitié de la hauteur des créatures et la largeur est égale à la largeur des créatures.

5.5.2 La fonction de fitness

La fonction d'évaluation est définie pour indiquer la direction du sommet de l'escalier et optimiser la vitesse de déplacement des créatures. Elle est la même que dans l'expérience précédente: $F = |x|/t$, où x indique la direction de l'escalier. Il faut noter que les créatures ne perçoivent pas l'escalier. C'est donc l'évolution qui, les guidant dans la bonne direction, adapte leur morphologie à l'ascension.

5.5.3 L'intérêt de l'expérience

On trouve partout dans notre quotidien des escaliers de différents types. Ils constituent, dans le domaine de la robotique, une difficulté de franchissement importante. Trouver la morphologie adaptée pour un vrai robot, son équilibre et la synchronisation de ses mouvements pour monter un escalier est très difficile comme le montrent les travaux de Dalvand et Moghadam [Dalvand and Moghadam 2006]. Ce problème est donc toujours d'actualité en robotique.

5.5.4 Analyse des résultats

Les meilleures créatures peuvent monter en haut de l'escalier rapidement. Différentes stratégies émergent. Certaines créatures se déplacent comme des vers capables de monter deux marches à la fois (Fig. 5.6.(c)). D'autres sont adaptées pour sauter de marche en marche (Fig. 5.6.(a)).

Le corps des meilleures créatures est modulaire et possède des axes de symétrie (Fig. 5.7). On peut noter que si certaines créatures ne sont pas totalement symétriques mais tout de même efficaces, l'évolution aura tendance à les rendre symétriques pour optimiser leur performance.

Pour augmenter la difficulté de l'expérience, nous avons augmenté la hauteur des marches. Par rapport aux simulations précédentes, la meilleure créature utilise une nouvelle stratégie pour monter les marches en s'accrochant et se hissant avec ses pattes (Fig. 5.6.(d)).

Malgré ce cas isolé, nous avons rencontré peu de problèmes d'instabilité par rapport à ceux qui sont souvent recensés dans les travaux de référence. Ces instabilités

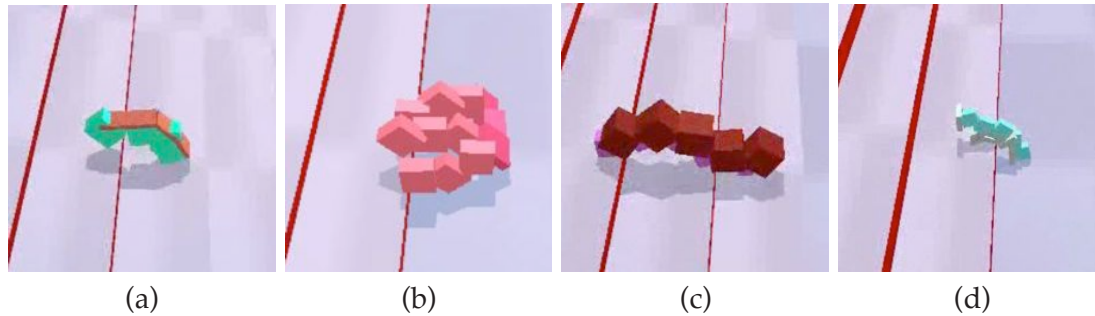


Figure 5.6: (a) Créatures sautant de marche en marche. (b) Une créature qui ressemble à une main, chacun ses doigts l'aide à monter l'escalier. (c) Une créature qui ressemble à un ver. Son corps modulaire est optimisé pour monter rapidement les escaliers. (d) Dans cette simulation la hauteur des marches est deux fois plus grande. L'un des meilleures créatures utilise une nouvelle stratégie pour monter les marches en s'accrochant et se hissant avec ses pattes.

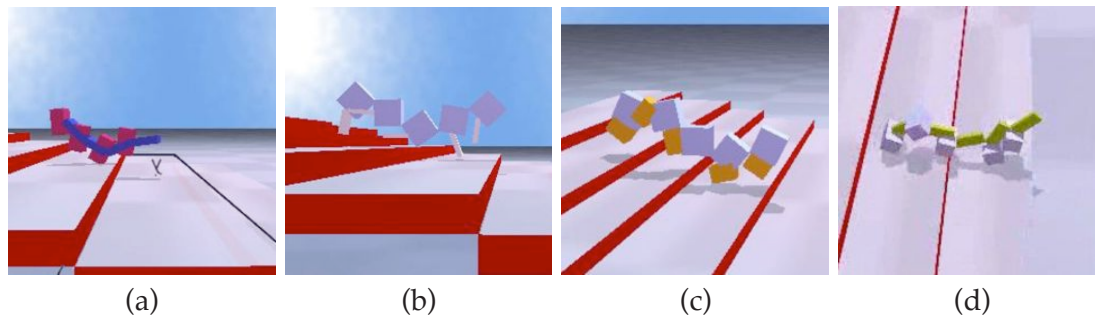


Figure 5.7: Créatures symétriques possédant des pattes. Le corps modulaire des créatures est optimisé pour monter rapidement les escaliers.

sont peut-être une des raisons pour lesquelles il y a eu peu d'évolutions faisant interagir des créatures et des environnements plus complexes. Cependant, quelques créatures ont réussi à exploiter des problèmes de détections de collisions. En effet à partir d'une défaillance du moteur physique des créatures ont pu se catapulter par-dessus l'escalier en s'accrochant à une marche et ont ainsi obtenu des évaluations hors normes.

Par la suite il serait intéressant de faire évoluer les créatures pour différents types d'escalier afin de voir si les stratégies d'adaptation sont les mêmes et voir si l'on peut avoir des créatures qui s'adaptent indifféremment aux différents types d'escalier.

5.6 Le franchissement de tranchées

5.6.1 Présentation des expériences

Pour cette expérience, nous isolons une créature sur le haut d'un cube. D'autres cubes identiques et alignés avec le premier sont répartis le long d'un axe parallèle à l'axe x et séparés d'une distance constante (Fig. 5.8). La créature a pour but de rejoindre le cube le plus éloigné de son point de départ. Plusieurs stratégies sont possibles comme sauter de cube en cube ou développer une morphologie assez grande et ramper jusqu'au cube suivant.

5.6.2 Interêt de l'expérience

L'intérêt de cette expérience repose sur le fait que des situations similaires apparaissent en robotique et qu'elles sont difficiles à résoudre. Il est de plus facile d'augmenter la difficulté de la tâche en changeant la largeur des tranchées. Afin de compliquer le problème, il serait souhaitable aussi d'alterner la position et la hauteur des cubes⁷.

5.6.3 Analyse des résultats

Dans la simulation que nous allons décrire les tranchées sont assez larges. Elles ont une longueur de la moitié de la longueur des cubes. La fonction de *fitness* est définie pour attirer les créatures sur le dernier cube. Si la créature tombe, sa *fitness* est divisée par dix. Ceci pousse les créatures à aller dans la direction du prochain cube. Il ne devrait pas être possible pour une créature de remonter sur un cube⁸.

La meilleure créature utilise ses pattes pour franchir les tranchées (Fig. 5.8.(b-c)). Il faut noter que l'évolution a été plus difficile pour cette expérience car beaucoup de créatures tombent dans les tranchées. Cependant l'évolution a trouvée comme solution d'augmenter petit à petit par mutation la taille des créatures afin de leur permettre d'atteindre le cube opposé. En effet, au début de la simulation, les créatures ont une taille largement inférieure à la largeur du cube sur lequel elles reposent, alors qu'en fin de simulation les meilleures créatures ont une taille d'un cube et demi (Fig. 5.8.a).

⁷Des premiers testes dans ce sens semblent montrer que les temps convergences sont nettement plus longs et la taille des créatures semblent augmenter lorsque les cubes sont plus éloignés. On obtient des créatures avec des corps modulaires dont le nombre de modules augmente par mutation

⁸Rapport de proportionnalité, jamais amélioré lors des simulations.

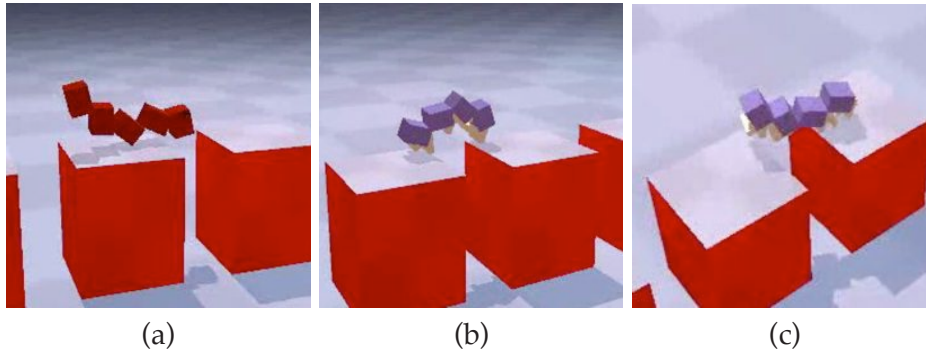


Figure 5.8: (a) La créature a une taille nettement plus grande que les créatures de la population initiale. (b-c) Cette créature s’aide de ses pattes pour franchir les tranchées.

5.7 Le déplacement sur un terrain accidenté

5.7.1 Présentation de l’expérience

Le monde est rarement parfait et plat, donc dans cette expérience, nous avons défini un environnement légèrement vallonné. L’amplitude des hauteurs de l’environnement peut aller jusqu’à dix fois la hauteur moyenne des créatures (Fig. 5.9). La fonction d’évaluation est la distance parcourue divisé par le temps comme dans les deux premières expériences $F = d/t$.

5.7.2 Analyse des résultats

Les créatures ont beaucoup plus de difficulté dans ce type d’environnement. Cela est dû au fait qu’elles ne peuvent pas prédire les irrégularités de l’environnement. Les meilleures créatures sont moins efficaces que celles qu’on a pu observer sur des sols plats, mais quelques unes obtiennent tout de même de bons résultats (Fig. 5.9).

Un autre obstacle majeur à l’évolution est le fait que les créatures ont tendance à rouler si elles se retrouvent sur des plans trop inclinés. Une solution à évaluer serait de les équiper d’un gyroscope comme dans les travaux Komosinski [Komosinski and Ulatowski 1999] qu’elles puissent se stabiliser ou détecter qu’elles sont dans des positions défavorables.

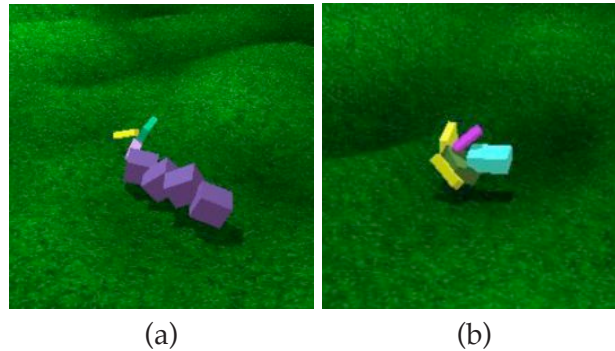


Figure 5.9: Créatures se déplaçant sur un terrain accidenté.

5.8 Le déplacement d'un mur en coopération

5.8.1 Présentation de l'expérience

Dans cette expérience, le but est de pousser un bloc dans une direction définie. Le bloc est défini de façon à ne pas pouvoir être poussé par une seule créature (Fig. 5.10). Afin de réaliser cette tâche, une créature et son clone⁹ sont initialisées à chaque début de simulation dans l'environnement. La fonction de *fitness* est de maximiser la distance parcourue par le centre de gravité du mur dw dans une direction (axes x): $F = dw_x$. Il faut aussi noter qu'il n'y a aucun moyen de synchronisation et de communication entre les créatures et que la direction du mur n'est pas indiquée aux créatures et c'est donc uniquement par évolution qu'elles parviennent à la trouver.

5.8.2 Analyse des résultats

Dans cette expérience, les meilleures créatures sont nettement plus grandes ce qui leur permet d'être plus rapides et d'avoir plus de force pour pousser le mur (Fig. 5.10). Nous pouvons observer qu'au cours de l'évolution les créatures grandissent, en effet au début de l'évolution (Fig. 5.10.a) les créatures ne sont composées que de cinq blocs. Par la suite ces créatures ont évolué, les blocs sont devenus plus gros, un nouveau bloc les a allongées, une mutation a fait apparaître des sortes de pattes... En fin de simulation, elles sont composées de quatorze blocs. À la fin de l'évolution, les créatures arrivent presque à faire tomber le bloc alors qu'au début, elles ont les plus grande difficulté à le trouver et à le faire bouger.

⁹Bien que ces créatures ont la même morphologie, elles peuvent agir différemment car si leur position est différente leurs points de contacts sont également différents.

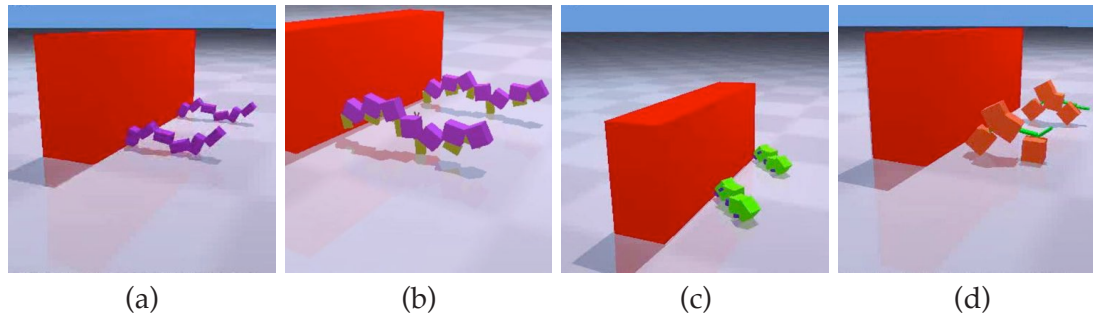


Figure 5.10: (a)(b) Evolution des créatures après quelques générations. (c) Créatures poussant le mur mais de taille trop petite pour être réellement efficaces. (d) Coopération entre deux créatures dont le corps est très fin sur lequel des blocs très gros sont répartis.

Les créatures plus petites ont plus de difficulté pour pousser le mur (Fig. 5.10.c) alors les créatures plus grandes avec de gros blocs ont plus de facilité (Fig. 5.10.d). En effet ces dernières créatures ont un corps très fin avec des blocs énormes¹⁰ servant à pousser le mur.

Dans le futur, nous planifions d'avoir deux créatures différentes afin d'obtenir diverses **stratégies** d'adaptations. Il est certain qu'avoir deux créatures avec des morphologies différentes augmente la difficulté et nécessite une certaine synchronisation. Il sera donc intéressant de voir si les créatures auront tendance à avoir des morphologies similaires pour faciliter leur tâche.

5.9 Le déplacement avec un objet mobile

5.9.1 Présentation de l'expérience

L'interaction avec l'environnement sera primordial si l'on désire obtenir des créatures artificielles de plus en plus complexes. Cependant, nous sous-estimons la capacité et l'adaptabilité des créatures évolutionnistes actuelles. En effet beaucoup d'expériences auraient besoin d'être réalisées pour trouver les limites et les potentialités réelles des créatures artificielles. Nous proposons, une nouvelle expérience dont le but est d'interagir avec un objet mobile.

Dans cette expérience, nous définissons une sorte de skateboard composé de quatre roues et d'une large planche. Le but est de voir si des créatures peuvent l'utiliser pour se déplacer dans une direction spécifique. Pour cela, les créatures sont déposées sur le skateboard. La créature et le skateboard sont deux objets différents et n'ont

¹⁰La masse des blocs est proportionnelle à leur volume.

pas de connections entre eux. La créature ne possède pas de capteur spécifique pour définir l'emplacement du skateboard. La fonction d'évaluation est définie pour maximiser la distance d_s parcourue par le skateboard tout en minimisant la distance entre la créature et le skateboard $|d - d_s|$.

$$F = d + d_s - |d - d_s|$$

. Pour le lecteur intéressé par la reproduction de ces expériences, nous mettons en annexe la description en langage *Steve* du skateboard (Annexe: A).

5.9.2 Analyse des résultats

Les résultats montrent l'emploi de différentes stratégies :

- Des créatures sautent sur le skateboard et se laissent glisser. Les créatures répètent plusieurs fois cette opération et parcourent ainsi une grande distance (Fig. 5.11.(a-b)).
- D'autres créatures poussent le skateboard et courent derrière (Fig. 5.11.c).
- D'autres créatures composées seulement de quatre blocs, dont deux pour le corps modulaire et deux pour les pattes, avancent en basculant sur la patte arrière servant de pivot et font avancer le skateboard en le soulevant à l'aide d'une force appliquée vers le haut par l'autre patte. (Fig. 5.11.d).

Ces résultats montrent que l'on peut obtenir des créatures évolutionnistes adaptées à leur environnement. Si ces résultats peuvent sembler spectaculaires du point de vue du comportement des créatures, il faut rappeler que les créatures ne perçoivent pas le skateboard, c'est donc l'évolution et les contraintes de l'environnement qui ont fait évoluer leur morphologie et leur comportement.

De plus, on peut aussi noter que la fonction d'évaluation reste simple par rapport au problème traité. En ce qui concerne les temps de calcul de ces expériences, elles ont nécessité environ six heures de calculs sur un *Mac Pro Bi-Xeon*.

5.10 Discussion du modèle

5.10.1 Apport de notre modèle

Notre modèle a apporté aux créatures des capacités à pousser un bloc en coopération, à faire du skate-board, à monter des escaliers, à évoluer sur un sol irrégulier et à passer

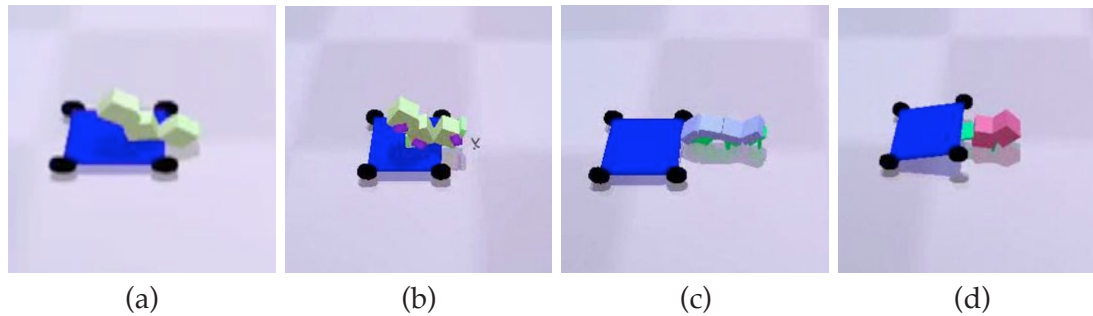


Figure 5.11: (a-b) Créatures se déplaçant en sautant sur un skateboard. Quelques générations plus tard, elle obtient des pattes qui lui permettent être beaucoup plus efficace en parcourant une distance dix fois plus grande. (c) Créatures poussant le skate. (d) Cette créature soulève le skatebord pour le déplacer et bascule sur une patte servant de pivot pour avancer.

des tranchées. Cependant nous pensons que les autres modèles ont pour la plupart les capacités à produire des résultats similaires. De ce point de vue, il est possible que les auteurs précédents aient sous-estimé les capacités d'adaptations des créatures artificielles. Une des raisons pour lesquelles on voit très peu d'environnement complexes est peut-être aussi le fait qu'il est difficile d'obtenir des créatures physiquement stables. Ainsi, en ayant désactivé l'interpénétration des blocs pour nos créatures, nous avons pu obtenir des résultats qui n'étaient peut-être pas accessibles sans une puissance de calcul supplémentaire. Nous en concluons qu'il serait intéressant, d'élargir l'ensemble des expériences permettant d'évaluer les créatures artificielles afin de connaître leur réelles limitations d'adaptation et de réaliser davantage de tests afin de déduire l'impact que peut avoir la gestion de l'interpénétration des blocs sur les résultats.

5.10.2 L'utilisation des patterns

L'utilisation des patterns est une maladresse dans le sens où sa réalisation apporte une solution mais ne répond pas à la démarche que nous voulions adopter. En effet, le contrôleur mis en place ne peut émerger de l'évolution. Bien que l'ensemble des patterns en soit issu, le contrôleur en lui-même vient de la créativité et de l'expérience de son concepteur à la différence des réseaux de neurones dont la topologie et le fonctionnement peuvent eux émerger d'une évolution. En effet, un réseau de neurones est composé d'éléments similaires, les neurones, et le contrôleur dépend de l'assemblage de ces neurones. Le système entier peut-être donc être inclus dans le réseau de neurones, ce qui lui donne la possibilité d'évoluer durant les simulations. Un système de

classeur quant à lui ne être produit que par l'intelligence et a donc moins de raisons d'être utilisé dans une approche émergente. En effet, le système de classeur n'est pas soumis à l'évolution, seules les règles évoluent et non le système complet. Cependant, nous pensons qu'il est utile d'explorer les capacités de ce type de contrôleur car même s'il ne répond pas à entièrement notre démarche il n'en reste pas moins performant. De plus le système de classeurs évite l'effet *black box* des réseaux de neurones, ce qui est un avantage par exemple pour l'implantation dans un robot réel.

5.10.3 Morphologie

Les créatures que nous obtenons dans nos résultats sont modulaires. En effet, le corps principal des créatures est généralement composé de blocs identiques issus d'un même nœud du *graftal*. Ainsi beaucoup de créatures ressemblent à des vers. Cette prédominance est due au fait qu'il est facile d'obtenir un ver qui a la capacité de se déplacer. En effet il peut être obtenu simplement à partir d'un *graftal* composé d'un seul bloc récursif et d'un contrôleur générant des mouvements ondulatoires. On retrouve ce type de créature dans des travaux antérieurs mais en moins grande proportion [Sims 1994a; Miconi and Channon 2006a]. Une des raisons peut être que la récursivité est moins privilégiée¹¹. Une autre explication est peut-être, le fait d'inclure la symétrie dans les créatures, celle-ci permet d'obtenir des solutions différentes au détriment de la modularité. Pour comparer les résultats des expériences en fonction du nombre blocs, nous avons répertorié dans un tableau leur nombre minimal et maximal ainsi que leur moyenne pour chaque expérimentation (Tab. 5.1).

En ce qui concerne les expériences dont le but est d'aller dans une direction spécifique, il faut noter que les solutions obtenues sont en majorité des vers ou des créatures avec des pattes. Cela vient du fait, comme nous l'avons vue précédemment, que ce type de morphologie est facile à obtenir et son efficacité augmente facilement au cours de l'évolution (ajout de pattes, agrandissement de la taille des blocs). Ce constat est d'autant plus important lorsque le but est de franchir des obstacles. Dans ce dernier cas, cela est aussi certainement dû au fait que les créatures peuvent facilement subir des mutations et conserver leurs caractéristiques tout en devenant de plus en plus grandes ou grosses suivant l'adaptation nécessaire à leur environnement. De plus ces créatures peuvent franchir des obstacles en redressant la partie avant de leur corps ce qui leur donne un avantage sur les autres créatures que l'on a obtenues.

¹¹Moins de blocs récursifs sont autorisés lors de la génération ou de la mutation des créatures.

Expériences	min	moy	max
déplacement sur sol plat (Sec. 5.1)	2	7,7	12
direction spécifique (Sec. 5.2)	4	8	12
montée d'un escalier (Sec. 5.3)	8	9,8	12
franchissement de tranchées (Sec. 5.4)	5	8,5	12
déplacement sur sol irrégulier (Sec. 5.5)	6	6,5	7
coopération (Sec. 5.6)	8	9	14
skateboard (Sec. 5.7)	4	8,3	12
moyenne :	5,3	8,4	11,6

Table 5.1: Les créatures sont en moyenne composées de huit à neuf blocs . On constate que le nombre maximal de blocs reste constant quelles que soient les expériences (entre onze et douze), sauf pour une expérience où il chute à sept. Pour ce qui concerne le nombre minimal de blocs, la moyenne est autour de cinq-six blocs. On peut remarquer que pour la montée des escaliers, il y a peu d'écart entre le nombre minimal et maximal de blocs (quatre blocs), ce qui peut sembler logique car les stratégies trouvées sont très semblables. Alors que pour les déplacements sur sol plat les stratégies sont très diverses et l'écart est très grand (dix blocs). Il faut noter que les résultats pour le sol irrégulier sont un peu faussés par le faible nombre de solutions (2) que l'on a trouvé (Para. 5.7)

5.11 Comparaison avec les autres modèles

Comme nous l'avons vu dans à la fin de l'état de l'art, il n'est pas facile de comparer qualitativement les différents modèles (Sec. 3.3.3). Nous avons donc repris le tableau comparatif des principaux travaux (Tab. 3.2) afin de confronter nos résultats.

5.11.1 Comparaison des résultats

On peut constater dans le tableau suivant (Tab. 5.3) où les notes sont purement subjectives, que les travaux sur les créatures artificielles focalisent leur effort sur le moteur physique et la morphologie au détriment d'un environnement complexe et d'un nombre varié d'expérimentations. Cela vient certainement du fait que le moteur physique et la représentation de la morphologie sont dans un premier temps indispensables à la réalisation de créatures artificielles évolutionnistes.

En ce qui concerne le réalisme des mouvements obtenus, la diversité des résultats et les stratégies d'adaptations on peut constater que les travaux de Karl Sims, bien qu'étant les plus anciens, dominent les résultats des autres travaux.

Pour ce qui est de nos travaux, les points forts sont :

- le nombre d'expérimentations réalisées qui permet de montrer que les créatures artificielles peuvent s'adapter à de nombreuses situations.

-
- notre environnement physique qui ne se résume pas à un simple plan 2D mais incorpore différents types d'obstacles
 - la première utilisation d'un objet mobile par les créatures évolutionnistes.
 - la diversité et les stratégies d'adaptations trouvées pour résoudre l'ensemble des expérimentations.

Les points que l'on peut améliorer par rapport aux autres travaux sont :

- La morphologie de nos créatures peut être améliorée en utilisant par exemple l'embryogenèse et en ajoutant des capteurs comme des accéléromètres, des gyroscopes et des capteurs de distance.
- Le mouvement des créatures pourrait être amélioré en prenant en compte l'énergie dépensée lors des déplacements ou en interpolant les signaux utilisés pour actionner les muscles des créatures.
- Notre modèle doit prendre davantage en compte le fait que les créatures puissent être utilisées comme modèle pour la conception de robots en y intégrant de nouvelles contraintes spécifiques à la robotique et à l'environnement réel (exemple : prendre en compte un environnement bruité pour les capteurs, modéliser les caractéristiques et les imperfections de vrai moteur pour les articulations, prendre en compte la densité des différents matériaux).

Dans le tableau suivant (Tab. 5.3), nous identifions le point fort de chacun des principaux travaux. Il donne une vue d'ensemble de ce que devra intégrer les futurs modèles de créatures artificielles évolutionnistes.

5.11.2 **Comparaison du contrôleur**

Nous n'avons pas réalisé de mesures quantitatives de notre contrôleur. Cependant une question que l'on peut se poser est : est-ce que l'usage d'une base de patterns en nombre fini peut égaler un réseau de neurones qui lui a la possibilité d'approcher n'importe quelle fonction. En effet, nous pensons qu'un réseau de neurones permet plus de souplesse car il peut généraliser des fonctions mathématiques, notre modèle présente lui l'avantage d'avoir un espace de recherche plus petit qui dépend de la taille de sa base de patterns.

En partant de ce constat, il serait donc intéressant de comparer ces deux types de contrôleurs sur des créatures dont la morphologie est figée puis sur des morphologies

auteurs	année	évaluation (points forts)						
		nb aptitudes	environnement	physique	mouvement	morphologie	diversité	stratégie
Sims	1994	5	3	5	5	4	5	5
Lipson	2000	1	3	5	4	4	4	4
Hornby	2001	1	3	4	4	5	5	4
Bongard	2003	2	4	4	4	5	4	5
Miconi	2006	3	3	5	4	4	4	4
Shim	2004	2	3	5	4	4	4	3
Taylor	2000	2	3	5	4	3	4	3
Chaumont	2007	2	4	5	4	4	4	4
Komosinski	2000	5	5	4	3	5	5	3
Lassabe	2007	7	5	5	4	4	5	5
moyenne :		3	3,6	4,7	4	4,2	4,6	4

Table 5.2: Travaux sur les créatures artificielles. La notation de 1-5 correspond à celle que nous avons décrit dans le tableau comparatif précédent (Tab. 3.2). Nous avons ajouté l'estimation qui reste subjective de nos résultats ainsi qu'une nouvelle colonne indiquant le nombre d'aptitudes auxquelles les créatures artificielles sont confrontées dans les différents travaux (marche, nage, saut *etc*). Nous avons calculé une moyenne générale des notes de chacune des caractéristiques. Cette moyenne montre que nos travaux se démarquent par le nombre d'expériences réalisées et obtiennent de bons résultats pour l'environnement et les stratégies d'adaptation obtenues.

auteurs	publications	évaluation (points forts)						
		nb aptitudes	environnement	physique	mouvement	morphologie	diversité	stratégie
Sims	[Sims 1994a]	5	3	5	5	4	5	5
Lipson	[Lipson and Pollack 2000]	1	3	5	4	4	4	4
Hornby	[Hornby and Pollack 2001]	1	3	4	4	5	5	4
Bongard	[Bongard and Pfeifer 2003]	2	4	4	4	5	4	5
Miconi	[Miconi and Channon 2006a]	3	3	5	4	4	4	4
Shim	[Shim et al. 2004]	2	3	5	4	4	4	3
Taylor	[Taylor and Massey 2000]	2	3	5	4	3	4	3
Chaumont	[Chaumont et al. 2007]	2	4	5	4	4	4	4
Komosinski	[Komosinski 2000]	5	5	4	3	5	5	3
Lassabe	[Lassabe et al. 2007b]	7	5	5	4	4	5	5
moyenne :		3	3,6	4,7	4	4,2	4,6	4

Table 5.3: Travaux sur les créatures artificielles. La notation de 1-5 correspond à celle que nous avons décrit dans le tableau comparatif précédent (Tab. 3.2). Nous avons ajouté l'estimation qui reste subjective de nos résultats ainsi qu'une nouvelle colonne indiquant le nombre d'aptitudes auxquelles les créatures artificielles sont confrontées dans les différents travaux (marche, nage, saut *etc*). Nous avons calculé une moyenne générale des notes de chacune des caractéristiques. Cette moyenne montre que nos travaux se démarquent par le nombre d'expériences réalisées et obtiennent de bons résultats pour l'environnement et les stratégies d'adaptation obtenues.

auteurs	les points forts des travaux.
Sims	La coévolution (combat pour le partage d'un cube).
Lipson	La fabrication de robots à partir d'une évolution de créatures virtuelles.
Hornby	La fabrication de robots et la morphologie modulaire.
Bongard	L'embryogenèse.
Miconi	Les réseaux de neurones.
Shim	L'évolution de créatures volantes.
Taylor	L'utilisation et l'état de l'art sur les moteurs physiques.
Chaumont	L'évolution de catapultes.
Komosinski	L'environnement des créatures.
Lassabe	Le nombre de nouveaux résultats (escalier, skateboard, tranchées).

Table 5.4: Points forts des différents travaux sur les créatures artificielles. On peut constater que ces différents travaux sont complémentaires.

évolutives. Pour une telle expérience, il est possible que l'usage des patterns soit plus performant pour une morphologie non figée car l'évolution peut permettre de compenser les carences du contrôleur en adaptant la morphologie.

Pour ce qui est de la convergence lors nos expérimentations, si on s'en tient au nombre de générations calculé lors d'une évolution, il semble que notre modèle soit en accord avec les autres travaux du domaine.

5.12 L'apport du modèle sur les trois principaux objectifs

Nous avons défini dans l'introduction trois principaux objectifs à long terme pour les créatures artificielles (Sec. 1.3.1). L'un des objectifs concerne la compréhension des mécanismes adoptés par le vivant pour concevoir des créatures virtuelles, le suivant l'animation d'entités dans le domaine de la réalité virtuelle et le dernier la fabrication d'une machine par une autre machine. Dans les trois prochaines sections, nous allons décrire comment notre modèle peut participer à la réalisation de ces objectifs. Nous proposerons par la suite des perspectives qui nous semblent envisageables dans ces domaines.

5.12.1 Compréhension du vivant pour concevoir des créatures artificielles

Dans ce domaine notre modèle n'apporte pas de nouveauté par rapport aux travaux déjà présentés dans l'état de l'art. En effet, notre modèle, si ce n'est l'évolution artificielle, n'emploie pas de nouveau mécanisme inspiré du vivant. Si contribution, il y a,

elle se situe dans les différents environnements que nous avons proposés pour évaluer les créatures artificielles. Néanmoins, en plus de notre modèle de créature artificielle, nous proposons dans le chapitre suivant une approche pour évaluer la complexité d'un écosystème (Chap. 6). En effet, nous pensons qu'il est nécessaire d'avoir des outils pour évaluer des futurs résultats qui seront de plus en plus difficiles à interpréter. Toujours dans l'objectif de concevoir des créatures artificielles adaptées à un environnement complexe nous proposons en annexe un modèle d'écosystème artificiel (Annexe. B). On peut en conclure, qu'il reste beaucoup à faire dans ce domaine.

5.12.2 L'animation d'entités virtuelles réalistes

Comment nous l'avons déjà mentionné, nous pensons que notre modèle de contrôleur est particulièrement adapté à l'animation d'entités virtuelles car il permet l'utilisation de patterns d'animation tout en les confrontant à l'évaluation de l'animation dans un environnement physique réaliste. Il faut toute fois reconnaître que si l'animation est efficace du point de vue qualitatif cela n'en garantit pas pour autant son esthétisme. Notre méthode pourrait tout de même être appréciée dans le domaine de la robotique où l'efficacité de l'animation est généralement privilégiée à l'esthétisme.

5.12.3 La conception d'une machine par une autre machine

C'est certainement l'objectif le plus ambitieux des trois. L'une des difficultés reposent sur les contraintes physiques que nous impose l'environnement réel. Nous pensons donc que cet objectif n'est réalisable que s'il prend en compte un environnement complexe qui fourni l'apport des ressources nécessaires à la confection de ces machines. En ce sens, notre contribution reste encore très modeste bien que nous pensons que les créatures évolutionnistes et les écosystèmes artificiels sont certainement une voie intéressante pour parvenir à atteindre ce but.

Complexité

6.1 Résumé

Comment aborder la complexité d'une simulation de vie artificielle ou d'un écosystème ? Comment savoir si une simulation est plus complexe qu'une autre ? Comment exploiter des mesures lors de simulations pour comprendre l'auto-organisation d'un système ?

Ce chapitre présente une nouvelle approche pour mesurer la complexité d'une simulation de vie artificielle. Ces travaux ont été menés en collaboration avec Bruno Gaume qui travaille sur les *small-worlds* [Lassabe et al. 2007; Lassabe et al. 2008]. Notre approche commence par une identification de chaque paramètre qui semble pertinent pour une mesure de la complexité de la simulation et de l'auto-organisation. Durant la simulation, l'évolution de ces paramètres est utilisée pour construire un graphe. Nous analysons l'évolution de ce dernier en utilisant les quatre propriétés définissant les petits mondes : *small-worlds*. Ces propriétés donnent des mesures de la *hiérarchisation* et du taux de *clustering* qui permettent de mieux comprendre l'auto-organisation de la simulation.

Dans les résultats, nous donnons des exemples d'applications de cette méthode et nous validons son application en utilisant une simulation de fourmis artificielles. Nous exécutons différentes simulations de fourmis avec des paramètres différents et nous comparons leur évolution et leur complexité grâce aux propriétés des *small-worlds*. Alors que générer les données qui permettent de créer des *small-worlds* avec toutes leur propriétés est un problème difficile (généralement les données sont collectées¹), nous montrons que les simulations de fourmis artificielles peuvent être un outil efficace à la production de tels graphes.

¹Les données proviennent de données collectées, c'est à dire elles ne sont pas générées à partir d'un algorithme.

Ces *small-worlds* artificiels sont intéressants car ils peuvent être comparés aux *small-world* réels et nous aident à comprendre la complexité de l'auto-organisation. En perspective, il pourrait être intéressant de voir, dans le cadre de simulations de vie artificielle, si l'on peut établir une relation entre les mesures que fournisse la théorie de l'information et celles provenant des propriétés des *small-worlds*.

6.2 Introduction

Il n'est pas évident de mesurer la complexité d'une simulation de vie artificielle ou d'un écosystème. Des simulations similaires pourraient être comparées grâce à la mesure de leur complexité. Les sociologues, les économistes, les biologistes et les mathématiciens essaient par des modèles de représenter la société, son organisation et ses interactions avec l'environnement. Pour cela ils collectent beaucoup de données qu'ils analysent ensuite. Une possibilité pour représenter ces données est sous forme de graphe. Ainsi les réseaux routier², les connexions dans notre cerveau [Achacoso and Yamamoto 1992], les relations entre les personnes [Gaure 1990; Wasserman and Faust 1994], le web [Albert et al. 1999], les galeries d'une fourmilière [Buhl et al. 2004] ou même l'évolution des espèces peuvent être représentés par des graphes. Beaucoup de solutions existent maintenant pour analyser ces graphes et leurs propriétés [Watts and Strogatz 1998; Newman 2003]. Cependant, il peut-être difficile de collecter ou d'obtenir ces données. Lors de simulations de vie artificielle, nous pouvons obtenir les données, mais nous n'avons pas toujours les outils pour analyser leur complexité [Adami et al. 2000]. Donc, si la vie artificielle essaie de reproduire la vie, pourquoi ne pas utiliser les outils qui ont déjà été utilisés pour mesurer la complexité de nos sociétés réelles ?

Dans ce chapitre, nous proposons d'utiliser les propriétés des *small-worlds* [Newman 2003; Watts and Strogatz 1998] pour mesurer la complexité d'une simulation de vie artificielle. Dans la prochaine section, nous expliquons pourquoi il est intéressant de mesurer la complexité d'une simulation de vie artificielle et quelles sont les propriétés que nous voulons mesurer. Nous montrons que s'il est facile de mesurer la complexité de chacune des parties d'un système, il est plus difficile d'en mesurer la complexité globale. Par la suite, nous expliquons pourquoi nous utilisons les propriétés des *small-worlds* dans la mesure de la complexité. Nous donnons ensuite différents exemples pour lesquels il est possible d'appliquer cette méthode, puis nous

²Nous avons travaillé sur les algorithmes de plus court chemins [Lassabe et al. 2007], mais ces travaux ne sont pas intégrés dans cette thèse.

développons le cas de simulations artificielles de fourmis [Bongard and Paul 2000]. Dans la dernière partie, nous montrons que ces méthodes sont très intéressantes pour générer des *small-worlds*. Enfin, quelques perspectives seront proposées.

6.3 Pourquoi mesurer la complexité ?

Un système complexe est un système composé de nombreux sous-systèmes en interaction [Prigogine and Stegers 1987]. Lorsque de nouvelles propriétés émergent de la somme de ces sous-systèmes, la complexité du système augmente [Kauffman 1993; Bar-Yam 1997]. Dans le cadre d'écosystèmes ou de simulations de vie artificielle qui font intervenir l'émergence, il est donc intéressant de voir si la complexité augmente durant les simulations afin de savoir si de nouvelles propriétés ont émergé [Seth 1998; Adami et al. 2000]. Une des possibilités pour mesurer la complexité est d'utiliser la théorie de l'information [Shannon 1948]. Par exemple, si nous considérons un écosystème artificiel, il est possible d'appliquer³ cette méthode afin de voir si l'information contenue dans la représentation de ses créatures est complexe. Mais pour connaître la complexité globale de cet écosystème, la connaissance de la complexité de chaque créature ne suffit pas. Cette information porte en effet sur les créatures et non sur l'ensemble des interactions du système. De plus, si de nouvelles propriétés doivent émerger, c'est par le biais des interactions entre les créatures. Il faut donc tenir compte des interactions pour mesurer la complexité globale d'un écosystème. Les interactions peuvent être représentées par un graphe dont les noeuds sont les créatures et les arcs les relations qui les relie. Par exemple, dans le modèle d'extinction des espèces [Abramson 1997], les arcs peuvent représenter les liens de parentés entre les espèces (noeuds). Un graphe peut ainsi donner des informations sur l'interaction entre les agents. Pour analyser un graphe, les propriétés des *small-worlds* décrites dans la section suivante sont intéressantes. Ces propriétés, comme la hiérarchisation, le *clustering*, sont fréquentes dans le monde réel dès qu'il y a une forme d'organisation [Bar-Yam 1997; Camazine et al. 2001], donc si les simulations de vie artificielle sont réalistes [Kauffman 1993], nous devrions retrouver ces propriétés. Si nous ne les retrouvons pas, qu'est ce que cela signifie pour les simulations ? Ces études peuvent donner certainement des éléments pour comprendre par simulation une partie des mécanismes de la vie.

³Une proposition pour connaître la complexité d'une créature est d'évaluer les potentialités de l'information contenue dans son code génétique. Pour cela on évalue toutes les différentes créatures qui peuvent être issues d'une mutation à partir de ce code génétique.

Dans nos précédents travaux sur les créatures artificielles et les écosystèmes [Lassabe et al. 2006; Lassabe et al. 2007b], nous recherchions un moyen de mesurer la complexité des interactions et l'organisation du système. Il pourrait être intéressant de voir si nous pouvons établir des relations entre la théorie de l'information et les propriétés des *small-worlds* pour une même simulation. Que seront les différences s'il y en a ? Analyser la topologie d'un graphe pour obtenir la complexité d'un système revient à représenter la complexité sous une forme topologique. L'avantage de cette représentation est que nous possédons des outils pour l'analyser. Cette représentation facilite la comparaison de deux simulations issues d'un même système, voir de systèmes différents. Elle permet une vision directe de la hiérarchisations du système et de son organisation.

6.4 Small-worlds

Dans cette section, nous présentons les propriétés des *small-worlds*. Pour plus de détails, nous recommandons les travaux de Newman sur la structure et les fonctions des réseaux complexes [Newman 2003] qui définissent les *small-worlds* par quatre propriétés décrites par la suite.

Commençons par définir un graphe G par un ensemble de sommets V et par un ensemble d'arcs E tel que $G = (V, E)$. Nous considérons G comme étant graphe non-orienté.

- La première propriété des *small-worlds* est leur faible densité. Cela signifie qu'ils ont un nombre réduit d'arcs par rapport à leur nombre n de sommets. Le nombre d'arcs est inférieur ou égal à n^2 , il est généralement égal à $O(n \log(n))$ [Watts and Strogatz 1998]. Cette densité D est définie par $D = k / (n(n - 1)/2)$ où k est le nombre d'arcs et $(n(n - 1)/2)$ le nombre maximal d'arcs.
- La seconde propriété est la longueur L du graphe. C'est la moyenne des longueurs des plus courts chemins entre deux sommets quelconques du graphe. Dans les *small-worlds* cette longueur L est très petite comparée à la longueur L d'un graphe aléatoire. Elle est comprise entre 3 et 5 pour les graphes lexicaux comme le montre Bruno Gaume dans ces travaux [Gaume 2008].
- La troisième propriété est le taux de *clustering* ou d'agrégation C définie par Watts et Strogatz [Watts and Strogatz 1998]. Le *clustering* d'un noeud du graphe,

C de v , C_v , est définie par $C_v = A_v / ((K_v(K_v - 1))/2)$ ou K_v est le nombre de voisins de v et A_v est le nombre d'arcs entre les voisins de v .

C de G que l'on notera C_G , est la moyenne des C de chaque sommet de G. C de G est toujours compris entre 0 et 1. Dans les *small-worlds*, cette valeur C_G est élevée par rapport aux autres graphes (> 0.1).

- La quatrième propriété est la loi de puissance P (*power-law*) [Newman 2003]. Elle correspond à la distributivité des sommets dans le graphe. Bien que cette propriété n'est pas incluse dans la définition des *small-worlds* de Watts et Strogatz [Watts and Strogatz 1998], nous considérons que les *small-worlds* doivent aussi avoir la propriété de la *power-law* car il donne de informations sur la hiérarchie du système⁴ Dans les *small-worlds*, il y a beaucoup de sommets avec quelques voisins et peu de sommets avec beaucoup de voisins. Plus les sommets ont d'arcs : moins ils sont nombreux. Ce qui revient à dire que la probabilité d'avoir un noeud v avec i voisins décroît quant i augmente.

Pour le visualiser, la somme des sommets avec le même nombre d'arcs est représentée dans un repère loglog avec sur l'axe x le nombre d'arcs connectés par sommet et sur les y la somme des sommets ayant le même nombre d'arcs. Cette fonction qui doit se rapprocher de la forme d'une droite est analysée par m le coefficient de la pente de la droite. m est obtenu par la méthode des moindres carrés où r est le coefficient de corrélation. m est généralement inférieur à -0.1 . r doit être inférieur -0.8 pour être un *small-world*.

Pour conclure, les *small-worlds* sont un ensemble de graphes avec une faible densité d'arcs, un haut coefficient de *clustering* C, une longueur L faible entre ses sommets et une loi de puissance qui se rapproche d'une droite (Fig. 6.1). Les graphes avec les quatre propriétés sont rares (Tab. 6.4) et sont très caractéristiques de l'organisation que l'on retrouve dans la nature ou dans nos sociétés [Achacoso and Yamamoto 1992; Gaure 1990; Wasserman and Faust 1994; Newman 2003; Watts and Strogatz 1998]. Leur production par un algorithme est un problème important car il est difficile d'obtenir les quatre propriétés en même temps. Dans la section suivante, nous proposons d'analyser une simulation de fourmis grâce aux *small-worlds*. Si l'analyse est importante, il sera aussi nécessaire d'interpréter ce que signifie cette analyse. Cette

⁴En effet, la distributivité des sommets permet d'identifier les noeuds fortement connectés, de plus ces noeuds sont généralement aux centres des *clusters* et sont les points de passages des plus court chemins qui traversent le graphe.

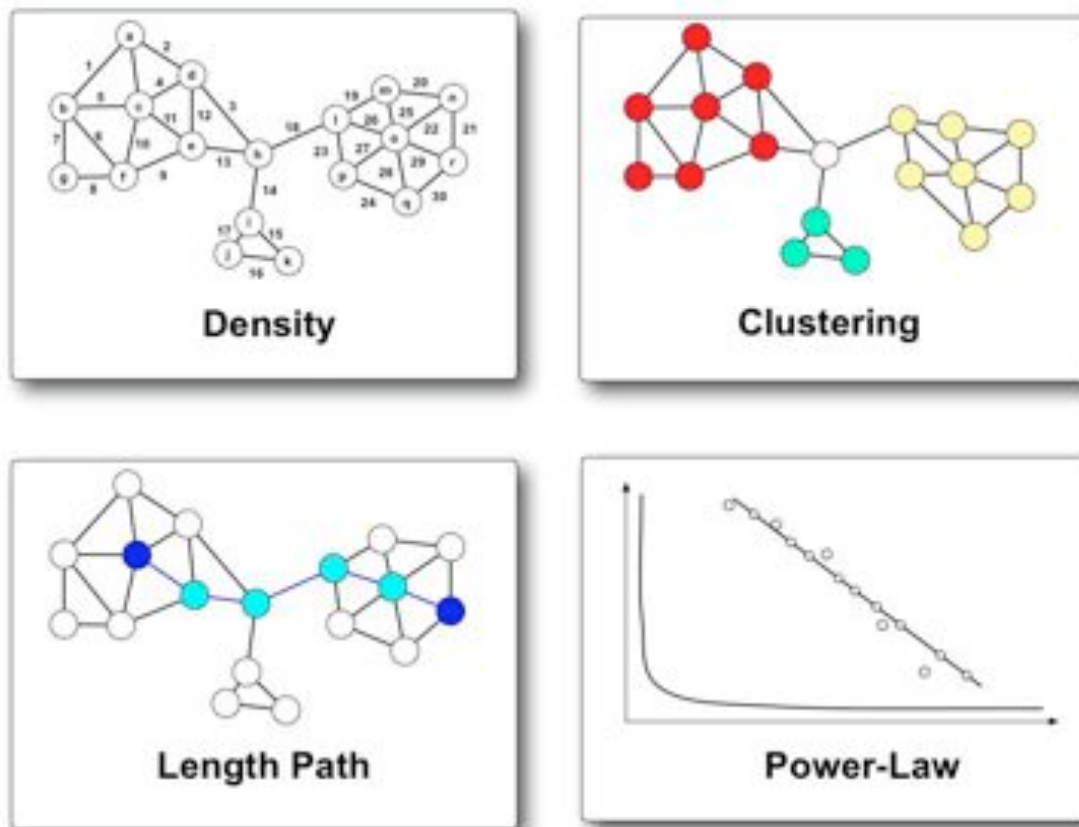


Figure 6.1: Densité : Le nombre d'arcs est peu élevé. Clustering : les noeuds sont très connectés localement. Longueur moyen des chemins : La distance entre deux noeuds est courte. *Power-law* : distributivité des sommets dans le graphe.

interprétation sera dans la plupart des cas spécifique aux systèmes analysés. Nous donnons une interprétation générale aux travers de ce tableau (Tab. 6.4).

6.5 Simulation de fourmis artificielles

En 1983, Deneubourg et ces collègues proposent un modèle de comportement collectifs des fourmis qui montre que les fourmis trouvent toujours pour se déplacer le plus court chemin [Deneubourg et al. 1983], par la suite, ces résultats ont inspiré les premiers travaux sur les colonies de fourmis artificielles de Manderick et Moyson [Manderick and Moyson 1988] puis ceux de Dorigo sur l'optimisation par colonie de fourmis [Dorigo 1992]. En s'inspirant de ces travaux, nous proposons un modèle très simple de colonie de fourmis qui tend à regrouper la nourriture en un même endroit.

Les fourmis artificielles sont dans un environnement de forme carrée avec de la

Types de Graphe	Densité D	Clustering C	Longueur L	Distributivité P
Small-World	faible	fort	court	loi de puissance
Graphe Aléatoire	dépend du processus de construction	faible	court	loi de poisson

Table 6.1: Comparaison entre un graphe aléatoire et un *small-world*.

Clustering		Distributivité		Longueur		Densité	
faible	fort	loi de puissance	loi de poisson	court	long	faible	forte
peu de connexions entre voisins	beaucoup de connexions entre voisins	bonne hiérarchie	pas de hiérarchie	bonne optimisation des distances	faible optimisation des distances	peu de chemins inutiles	beaucoup de chemins redondant
$C < 0.1$	$C > 0.1$	$r \leq -0.8$	$r > -0.8$	$L < 15$	$L > 15$	$D \leq 1$	$D > 1$

Table 6.2: Interprétation et signification des propriétés des *small-worlds*

nourriture régulièrement répartie sur la surface de l'environnement. Le comportement des fourmis est très simple :

- quand une fourmi trouve de la nourriture elle la prend. Elle la dépose quand elle trouve un nouveau site avec de la nourriture.
- le déplacement des fourmis est soit aléatoire⁵, soit plus naturel en s'approchant du déplacement réel de fourmis comme nous le décrivons dans les expériences suivantes.

Dans les différentes expériences, nous modifions les caractéristiques des fourmis et nous ajoutons des paramètres tels que la présence d'une mémoire pour chacune des fourmis, leur mode déplacement, la répartition et la quantité de nourritures, ainsi que le nombre de fourmis.

La question qui se pose est alors de mesurer la différence de complexité de ces expériences. Pour cela un graphe (Fig. 6.2) est construit en utilisant les sites où se trouve la nourriture comme des sommets. **Deux sommets V_1 et V_2 sont connectés dans le graphe quand une nourriture provenant de V_1 est déposée en V_2 .** Durant la simulation, la nourriture sera transportée par les fourmis causant ainsi la disparition de sites au profit d'autres qui grossiront. À la fin de la simulation, toute la nourriture est généralement regroupée sur quelques sites. L'analyse du graphe construit est réalisée en utilisant les propriétés des *small-worlds*. Il est possible de réaliser cette

⁵mouvement Brownien

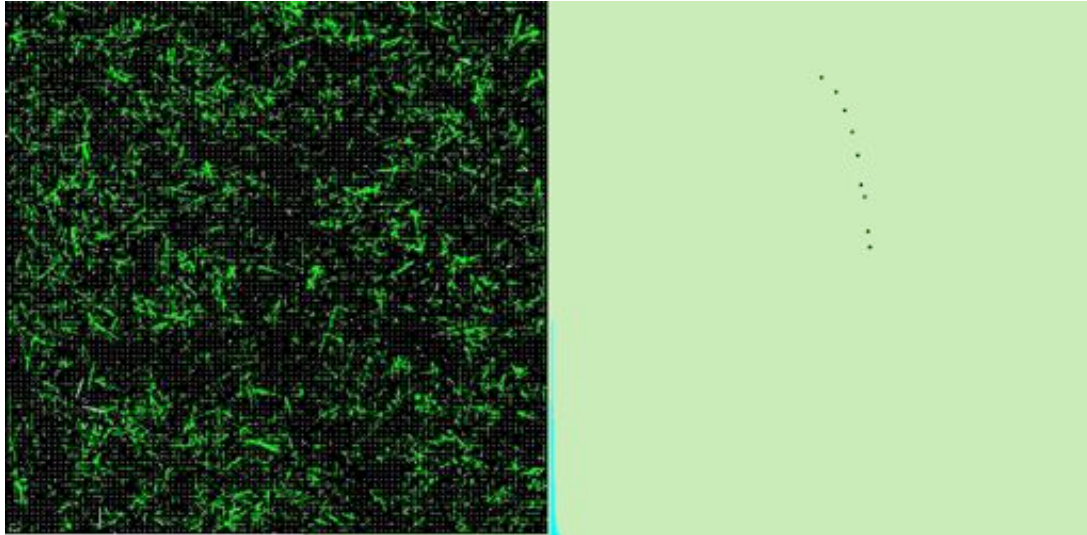


Figure 6.2: Capture d'écran de notre simulation de fourmis artificielles. La visualisation en temps réel de la simulation et de la distributivité des sommets permet de juger visuellement de l'orientation que prend l'expérimentation. Sur la droite, se trouve le graphe en cours de construction. À gauche, sont représentés en bleu le degré de distribution des arcs par sommets et en noir sa représentation par une loi de puissance dans un repère *loglog*.

analyse durant la simulation et ainsi d'obtenir un graphe dynamique dans le temps (Fig. 6.3). Ainsi pour comparer, deux simulations, il suffit de comparer l'analyse issue des graphes.

6.6 Expériences

Dans chaque expérience, nous observons l'influence de la variation de chaque paramètre en fonction de la taille de la population de fourmis, la quantité de nourriture, les différents moyens de locomotion et la mémoire des fourmis. Les variations modifient les structures du graphe, nous en mesurons les changements grâce aux propriétés définies plus haut.

Durant les simulations de chacune des expériences les résultats de l'analyse des graphes sont reportés dans des tableaux. Les calculs des propriétés interviennent quand au minimum 10000 arcs ont été ajoutés au graphe. Cela donne entre trois et six mesures, ce qui est suffisant pour montrer l'évolution de l'organisation dans le temps de chaque simulation⁶. Il faut signaler aussi que tous les graphes générés

⁶Il est difficile d'analyser le graphe à chaque modification à cause des temps de calcul que cela engendre. mais une solution envisageable serait de calculer incrémentalement les valeurs des propriétés du graphe.

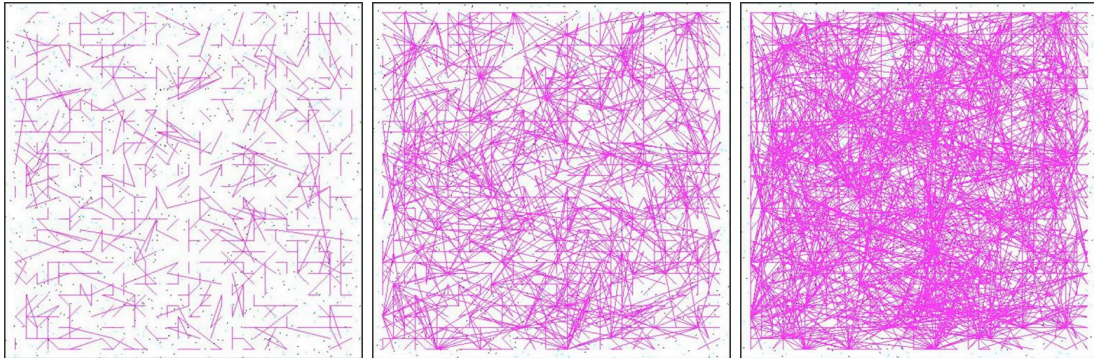


Figure 6.3: Évolution dans le temps de la construction d'un graphe par simulation de déplacement de fourmis artificielles.

sont connexes. Le nombre de sommets est égal à la quantité de nourriture⁷ qui est généralement répartie uniformément sur une grille carrée. On peut ainsi obtenir des graphes plus grands si on l'augmente la quantité de nourriture en début de simulation. Chaque simulation n'a été réalisée qu'une seule fois⁸. On peut noter qu'il n'est pas nécessaire de réaliser plusieurs simulation car celles-ci n'auraient pas trop d'impact sur les résultats obtenus étant donné que le nombre de fourmis qui participent à la construction du graphe est conséquent. De plus si on obtient des *small-worlds*, ces graphes ont la propriété de conserver les propriétés même si une partie de leur noeuds ou arcs est modifiée ou supprimée. Ces propriétés sont donc très robustes, l'exécution de même simulations malgré le déplacement aléatoire des fourmis donnera donc des graphes ayant des propriétés dont les paramètres les caractérisant seront très similaires.

6.6.1 Variation de la taille de la population

Nous observons ce qui se passe si nous fixons la quantité de nourriture à 1024 (grille 32x32) et que nous changeons la taille de la population. La taille de la population pour ces expériences est de 256, 512 et 1024 c'est à dire 1/4, 1/2 et 1 fois le nombre de sites avec de la nourriture. La table (Tab. 6.6.1) résumant les expériences montre que nous obtenons de bonnes caractéristiques des *small-worlds*. Leur taux de clustering C est supérieur à 0.1 et le coefficient de corrélation r de la *power-law* inférieur à -0.8 (voir [Newman 2003] pour plus de détails sur ces valeurs). Le *clustering* augmente

⁷Il y a au début de chaque simulation une unité de nourriture par site.

⁸La durée d'une simulation dure environ une à deux heures.

Small-world Propriétés	C	P		L	Densité
		m	r		
Population	256				
Arcs 10000	0.1225	-1.3734	-0.8253	4.1869	0.019
Arcs 20000	0.1588	-1.1904	-0.8527	3.1286	0.038
Arcs 30000	0.1755	-1.1855	-0.8510	2.8069	0.057
Population	512				
Arcs 10000	0.1436	-1.3475	-0.8703	4.1146	0.019
Arcs 20000	0.1788	-1.2303	-0.8184	3.0480	0.038
Arcs 30000	0.2043	-1.0951	-0.8022	2.7042	0.057
Arcs 50000	0.2537	-0.9619	-0.8279	2.4158	0.095
Arcs 60000	0.2827	-0.9236	-0.8290	2.3398	0.114
Population	1024				
Arcs 10000	0.1851	-1.3652	-0.8402	3.5575	0.019
Arcs 30000	0.3007	-0.9780	-0.8278	2.4094	0.057
Arcs 50000	0.4346	-0.8580	-0.7767	2.1638	0.095
Arcs 80000	0.5515	-0.7930	-0.7538	2.0656	0.152

Table 6.3: Variation de la taille de la population et évolution du nombre d'arcs. La quantité de nourriture (sommets) est fixée à 1024 et répartie sur une grille de 32x32. Le mouvement des fourmis est un mouvement Brownien. Tous les graphes générés sont des *small-worlds* exceptés les deux derniers. La meilleure valeur pour la taille de la population est 512 (Pour cette valeur, le graphe le plus grand a 60000 arcs).

durant les simulations tout en gardant une densité faible (Fig. 6.4). Ce qui montre que les arcs ajoutés servent à complexifier le graphe en connectant localement les noeuds voisins et les clusters entre eux. Les paramètres de la *power-law* montrent une bonne distributivité des connections, ce qui signifie que les graphes sont aussi hiérarchiques. Le meilleur résultat est obtenu pour une population de 512 fourmis (les quatre propriétés sont présentes durant toute la simulation). Ce qui n'est pas réciproque lorsque la taille de population est 1024 car la distributivité est moins respectée dès que le graphe à plus de 50000 arcs mais le *clustering* est alors très fort. Avec 256 fourmis, on obtient de bons résultats mais le graphe reste de plus petite taille et le *clustering* est plus faible que pour une population de 512. On peut en conclure que la taille de la population donne de bons résultats lorsqu'elle représente environ la moitié de la quantité de nourriture.

6.6.2 Variation de la quantité de nourriture

Dans cette expérience, nous fixons la taille de la population à 1024 fourmis. 512 étant la meilleure valeur pour laquelle on a obtenu des *small-worlds* pour l'expérience

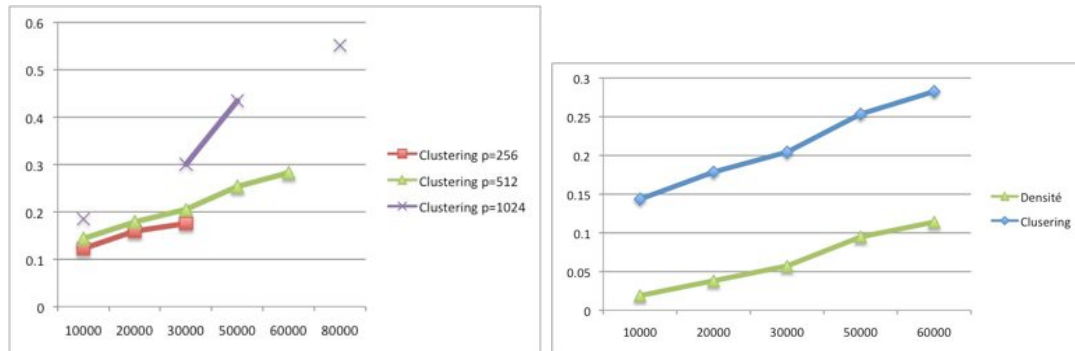


Figure 6.4: Le *clustering* augmente durant les simulations tout en gardant une densité faible.

précédente, nous doublons cette valeur afin d’obtenir des graphes d’une plus grande taille et voir si les propriétés des *small-worlds* sont conservées. La quantité de nourriture change pour chaque expérience (Tab. 6.6.2). Les valeurs sont 1024, 2025, 4096 et 9216 respectivement réparties sur des grilles de 32x32, 45x45, 64x64 et 96x96. Les quantités de nourriture ont été choisies afin d’être correctement réparties sur une grille et d’être doublées pour chaque expérience. Cette expérience confirme la précédente, la meilleure valeur est pour une quantité de nourriture de 2025. C’est à dire pour la moitié de la taille de la population de fourmis. Nous pouvons remarquer que nous avons réussi à augmenter la taille des graphes par rapport à l’expérience précédente.

6.6.3 Répartition aléatoire de la nourriture

Les conditions de l’expérience sont les mêmes que la précédente. Seule la répartition de la nourriture est maintenant aléatoire pour voir l’effet que la répartition de la nourriture a sur la complexité de la simulation. La taille de la population est de 512 et la quantité de nourriture est de 2025 pour pouvoir le comparer avec les simulations précédentes. On a donc un rapport de 1/4. Le graphe obtenu est un *small-world* (Tab. 6.5). Le *clustering* est plus important lorsque la nourriture est répartie sur une grille. Les autres propriétés restent similaires. Cependant des simulations supplémentaires sont nécessaires pour réellement conclure.

6.6.4 Déplacement

Dans cette expérience (Tab. 6.6), nous utilisons un autre type de déplacement de fourmis. Le déplacement est plus proche de celui d’une vraie fourmi mais sans utilisation

Small-world Propriétés	C	P		L	Densité
		m	r		
Nourriture	1024				
Arcs 10000	0.1851	-1.3652	-0.8402	3.5575	0.019
Arcs 30000	0.3007	-0.9780	-0.8278	2.4094	0.057
Arcs 50000	0.4346	-0.8580	-0.7767	2.1638	0.095
Arcs 80000	0.5515	-0.7930	-0.7538	2.0656	0.152
Nourriture	2025				
Arcs 10000	0.1128	-1.6865	-0.8751	7.7934	0.0048
Arcs 30000	0.1586	-1.3332	-0.7932	3.6897	0.014
Arcs 50000	0.1824	-1.2578	-0.8076	3.0466	0.024
Arcs 80000	0.2043	-1.1504	-0.8428	2.6868	0.038
Nourriture	4096				
Arcs 10000	0.0833	-2.3838	-0.8643	17.8707	0.0011
Arcs 30000	0.1273	-1.6291	-0.8264	6.4254	0.0033
Arcs 50000	0.1430	-1.5341	-0.8439	4.6484	0.0055
Arcs 100000	0.1568	-1.3572	-0.8462	3.4279	0.011
Arcs 150000	0.1673	-1.3027	-0.8463	3.0729	0.0165
Nourriture	9216				
Arcs 30000	0.0962	-2.6756	-0.8759	25.0928	0.00070
Arcs 50000	0.1233	-2.2949	-0.8208	12.4055	0.0011
Arcs 80000	0.1384	-2.0286	-0.8176	7.4871	0.0018
Arcs 100000	0.1450	-1.8505	-0.8214	6.1711	0.0025

Table 6.4: Variation de la quantité de nourriture et évolution du nombre de sommets. La taille de la population est fixée à 1024. Le déplacement utilisé par les fourmis est le mouvement Brownien. La meilleure valeur est pour 2025 sites de nourritures (Pour cette valeur, le graphe le plus grand a 60000 arcs, soit 2000 arcs de plus que dans l'expérience précédente.).

Small-world Propriétés	C	P		L	Densité
		m	r		
Nourriture	Répartition de la nourriture				
Arcs 10000	0.1059	-2.2604	-0.8327	8.7386	0.0048
Arcs 30000	0.1415	-1.3400	-0.8377	3.8863	0.014
Arcs 50000	0.1565	-1.2211	-0.8251	3.2169	0.024
Arcs 100000	0.1876	-1.1532	-0.8707	2.7079	0.048
Arcs 150000	0.2170	-1.0221	-0.8664	2.5239	0.072

Table 6.5: Dans cette expérience la répartition de la nourriture est aléatoire. La taille de la population est de 512 et la quantité de nourriture est de 2025.

Small-world Propriétés	C	P		L	Densité
		m	r		
Motion	Déplacement non Brownien				
Arcs 10000	0.0135	-1.3635	-0.8585	3.7066	0.0048
Arcs 20000	0.0434	-1.3367	-0.8508	2.9351	0.0096
Arcs 30000	0.0878	-1.1960	-0.8786	2.6733	0.014
Arcs 50000	0.1703	-0.9954	-0.8712	2.4357	0.024
Arcs 60000	0.2019	-0.9645	-0.8850	2.3859	0.028

Table 6.6: Le déplacement des fourmis non Brownien. La taille de la population est de 512 et la quantité de nourriture est de 2025 (grille 35x35).

de phéromone⁹. Il est défini tel que dans 90% des cas la fourmi va tout droit, elle a ensuite 5% de chance de tourner à droite et 5% à gauche. Le nombre de fourmis est de 512 et la quantité de nourriture 2025 (grille 35x35). Par rapport à l'expérience précédente, le graphe obtenu est plus petit mais à valeur égale son *clustering* est plus grand. Il est surprenant d'avoir un *clustering* plus fort car le type de déplacement employé par les fourmis couvre des zones plus grandes. On pourrait donc penser qu'il est plus difficile d'avoir des sommets voisins fortement connectés que lorsque l'on a un mouvement qui couvre une zone locale (comme le mouvement Brownien).

6.6.5 Mémoire

La taille de la population est de 2048. Le mouvement des fourmis est aléatoire. Chaque fourmi a une mémoire M qui stocke la quantité maximale de nourriture rencontrée durant la simulation. Il est possible de déposer de la nourriture sur un site si seulement la quantité de nourriture de ce site est supérieure ou égale à $M - 1$. Les déplacements de nourriture étant contraint par la mémoire, ils sont donc moins fréquents. Il est donc utile d'augmenter le nombre de fourmis pour avoir une durée de simulation correcte. La distributivité des connexions n'est pas fortement hiérarchiques ($r > -0.8$). La mémoire qu'on l'on a définie n'est clairement pas avantageuse (Tab. 6.7). Cela vient du fait qu'en fin de simulation il devient difficile de satisfaire les conditions imposées par la mémoire. Les fourmis devant toujours trouver des tas de plus grands ce qui est de plus en plus rare au fur et à mesure de l'agrégation des grains de nourriture.

⁹Les phéromones sont des substances que les fourmis déposent afin d'indiquer un chemin à leurs congénères.

Small-world Propriétés	C	P		L	Densité
		m	r		
Mémoire	Fourmis 512 , Nourriture 2025				
Arcs 9372	0.1996	-1.0604	-0.7998	3.0426	0.0045
Arcs 13946	0.3518	-0.9490	-0.7545	2.3354	0.0068

Table 6.7: Résultats pour les fourmis utilisant de la mémoire. Le coefficient de corrélation est légèrement inférieur à 0.8

6.7 Synthèse

Nous venons de voir l'évaluation de la complexité d'une simulation de fourmis par le biais de constructions dynamiques de graphes. Il faut noter que malgré que toute la nourriture soit regroupée sur quelques sites à la fin de chaque simulation, on constate que l'organisation qui a permis ces résultats peut parfois légèrement différer. Les graphes générés ont des hauts degrés de *clustering* ($C > 0.1$) ; cela montre que les simulation ont un bon niveau d'organisation. La majorité des graphes sont des *small-worlds* possédant les quatre propriétés : *clustering*, faible densité d'arcs, *power-law* et faible longueur des chemins. Ainsi l'évaluation grâce aux propriétés des *small-worlds* permet d'avoir une autre vision de la complexité du système. Elle révèle l'organisation qui a permis d'aboutir aux résultats tel que le nombre de fourmis nécessaire, l'influence de la répartition de la nourriture et les effets d'une mémoire sur une fourmi. La difficulté reste la sélection du paramètre à analyser. En effet, il aurait été intéressant et complémentaire d'analyser le graphe des interactions des fourmis. Il serait aussi intéressant de maintenir les simulations actives en ajoutant régulièrement de la nourritures, et de regarder par quelle phase la complexité du graphe passe et évolue.

Le but de cet exemple n'étant pas de faire une analyse détaillée mais de montrer qu'il est possible d'utiliser les propriétés des *small-world* pour analyser des simulations, nous donnons dans la prochaine section d'autres exemples d'applications.

6.8 Exemples d'applications

Dans cette section, nous citons des exemples où il est possible d'appliquer les propriétés des *small-worlds* pour en mesurer la complexité. Cette méthode est applicable à tout système qui présente une forme d'organisation.

-
- Nous proposons d'utiliser cette méthode dans d'autres types de simulation comme les écosystèmes. Dans notre cas, il est envisageable d'utiliser les *small-worlds* pour évaluer la complexité d'un écosystème composé de créatures artificielles. Si les noeuds représentent les créatures, il serait possible de réaliser un graphe retraçant les reproductions entre les créatures. On devrait ainsi observer un fort *clustering*. Chaque cluster correspond à une espèce. Aussi, il serait possible de réaliser le graphes des connaissances. C'est à dire l'ensemble des créatures qui ont été en contact ou à proximité les unes des autres. Nous proposons en annexe un modèle d'écosystème qui pourrait être sujet à ce type de mesures. Dans un premier temps, l'écosystème Gene Pool de Ventrella que nous avons décrit dans l'état de l'art pourrait aussi être utilisé comme base d'étude [Ventrella 2005].
 - Les applications multi-agents peuvent aussi être analysées de manière similaire aux simulations de fourmis. En effet, dans les systèmes multi-agents, des agents sont décrits par des propriétés et des solutions émergent de leurs interactions. Nous pensons que notre proposition de la mesure de complexité s'associe bien à ce type d'exemple.
 - En vie artificielle, des travaux portent sur l'émergence du langage. On sait que les graphes lexicaux sont des *small-worlds* [Gaume 2008]. L'émergence de langage entre robots ou entités virtuelles devrait aussi produire des *small-worlds*. Il serait donc intéressant d'analyser les graphes produits et de comparer ainsi leur structure avec celle de langues connues. Cela peut permettre de savoir au sens linguiste si un langage est bien généré.
 - En chimie, il a été montré que le graphe des interactions des protéines est un *small-world* [Jeong et al. 411]. Il serait donc intéressant d'utiliser les *small-worlds* pour analyser les chimies artificielles proposé en vie artificielle et ainsi de pouvoir comparer les modèles entre eux et avec la réalité.
 - Dans le cadre des réseaux de neurones, il y aurait deux possibilités pour utiliser notre proposition. La première serait d'évaluer les propriétés des larges réseaux de neurones artificiels pour permettre de comparer leurs différents modèles de topologies et ainsi les améliorer. Une deuxième possibilité serait dans l'évolution de topologie de réseaux de neurones comme ceux présentés par Stanley [Stanley and Miikkulainen 2002]. En effet, ces évolutions par algorithme génétique pourraient utiliser dans les fonctions d'évaluations une

mesure des propriétés de la topologie des réseaux et ainsi améliorer la qualité de leur génération. Ainsi, une mesure du taux de *clustering*, de la densité, de la répartition des connexions et de la longueur des chemins permettrait en partie d'orienter l'évolution de topologie de réseaux vers une catégorie de solutions souhaitées.

6.9 Conclusions et Perspectives

Nous avons proposé une méthode permettant de mesurer la complexité d'un système global. Dans cette section, nous avons montré qu'il est possible d'utiliser les propriétés des *small-world* pour analyser les simulations de vie artificielle. Cela permet de montrer les différences de complexité entre les simulations. Généralement les méthodes pour générer des *small-worlds* obtiennent difficilement les quatre propriétés simultanément.

Les simulations de fourmis constituent une nouvelle méthode pour générer des *small-worlds*. Il est donc intéressant de synthétiser de tels graphes car il est difficile de les obtenir de manière sûre à partir d'algorithmes classiques. En perspective, nous envisageons d'améliorer cette méthode en intégrant des paramètres pour contrôler le type de *small-worlds* générés. Nous voulons aussi mieux déterminer la mesure de la complexité à l'aide de nouvelles simulations et essayer d'établir des relations entre la théorie de l'information et les résultats obtenus. Il serait aussi intéressant de voir s'il est possible à partir de cette méthode de générer des graphes pour des réseaux de neurones artificiels pour lesquels il est souvent difficile d'obtenir une topologie efficace.

Il peut aussi être envisagé d'utiliser les propriétés des *small-worlds* comme une fonction de *fitness* d'un environnement en cours d'évolution tel qu'un écosystème composé de créatures. La fonction de *fitness* donnerait ainsi peu de contraintes tout en privilégiant l'auto-organisation du système.

Conclusion et perspectives

7.1 Conclusion générale

Cette thèse a présenté les principaux travaux sur les créatures et les écosystèmes artificiels. Depuis les premiers articles de Karl Sims qui ont été précurseurs dans le domaine, beaucoup de chercheurs ont tenté d'obtenir - au moins - les mêmes résultats. Nos travaux s'inscrivent dans cette lignée en essayant de proposer un nouveau type de contrôleur et un ensemble de nouvelles expérimentations. Ainsi nos résultats semblent montrer que les créatures évolutionnistes peuvent s'adapter à des environnements plus complexes que ceux initialement proposés par Karl Sims. En effet, nos expérimentations sont les premières à faire évoluer des créatures artificielles avec un skateboard, sur des escaliers ou des tranchées. Ainsi ces nouvelles expérimentations peuvent servir aux validations des autres travaux du domaine et ainsi permettre une meilleure évaluation de leurs résultats.

Notre modèle de contrôleur a l'originalité de ne jamais avoir été utilisé auparavant pour l'évolution de créatures artificielles, domaine dans lequel les réseaux de neurones sont très présents. Notre contrôleur a l'avantage d'utiliser une base de patterns qui est extensible et qui peut produire des signaux complexes. Les comportements obtenus lors de nos expériences semblent montrer des caractéristiques intéressantes pour ce type de contrôleur notamment dans l'animation d'entités virtuelles. Cependant, il aurait été souhaitable d'en mesurer réellement les performances dans d'autres contextes et de le comparer avec les autres contrôleurs du domaine.

Nous avons aussi proposé d'employer les propriétés des *small-worlds* pour évaluer l'évolution de la complexité d'environnement virtuel. Cette méthode a l'avantage de mesurer la globalité de la complexité de l'environnement. Cette approche a été appliquée, pour l'exemple, sur un environnement peuplé de fourmis artificielles. Les résultats montrent que l'on obtient une bonne mesure de la com-

plexité. Ce type de simulation peut donc constituer un outil pour produire des jeux de données ayant les propriétés des *small-worlds*.

Notre contribution porte aussi sur les écosystèmes dont nous avons proposé un modèle en annexe qui peut servir d'environnement pour l'évolution de créatures artificielles. Celui-ci incorpore de nombreux concepts destinés à faire émerger des cycles de vie et des stratégies d'adaptations complexes.

7.2 Perspectives de notre modèle

Nous proposons dans cette section quelques perspectives afin d'améliorer notre modèle.

7.2.1 Contrôleur de *patterns*

Comme nous l'avons vu, notre contrôleur s'inspire des modèles de classifieurs pour sélectionner les *patterns* qui actionnent les muscles de nos créatures. Notre modèle s'est limité à des états binaires, il serait donc intéressant, à partir de travaux comme ceux de Stéphane Sanchez et Olivier Heguy sur les classifieurs génériques [Heguy et al. 2004], d'intégrer des valeurs réelles, pour simuler par exemple des capteurs de distance. Cela permettrait aux créatures d'avoir une meilleure perception de l'environnement et de ce fait une meilleure adaptation à l'environnement. Une validation sur un vrai robot permettrait de valider le modèle.

7.2.2 Environnement plus complexe

Afin d'obtenir des créatures de plus en plus complexes nous pensons qu'il est utile d'avoir des environnements aussi de plus en plus complexes. Cela peut être réalisé en utilisant des écosystèmes artificiels qui semblent être l'une des suites logiques pour faire évoluer les créatures.

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 ainsi, nous pourrions sélectionner 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 c'est elle qui détermine la survie des créatures.

7.2.4 Embryogenèse

L'embryogenèse comme nous l'avons décrit au cours de cette thèse aurait certainement de nombreuses applications pour l'évolution des créatures. Nous avons inscrit nos travaux dans un niveau que nous pouvons qualifier de "l'organe à la créature". Réaliser le lien avec le niveau "cellule vers organe" permettrait à partir d'un composant indifférencié "cellule" de générer une créature complète.

7.3 Perspectives générales

À l'aide d'un graphique (Fig 7.1), nous présentons des perspectives personnelles et générales sur cinq et dix ans. Ces perspectives s'orientent dans le but de résoudre les objectifs principaux que nous avons défini pour les créatures artificielles à savoir l'auto-réplication, l'animation réaliste pour des environnements virtuels et l'adaptation à des environnements complexes :

7.3.1 Limitation des représentations morphologiques

Si l'évolution trouve des solutions, on s'aperçoit qu'elles sont limitées par la représentation de la morphologie utilisée. Ainsi les stratégies trouvées avec les *graphals* leur sont certainement propres. Il en est de même avec les *L-systems* qui permettent de trouver des morphologies différentes. Le choix de la représentation de la morphologie a donc un impact important sur le type de solutions trouvées. Il serait donc intéressant d'avoir un meta-langage afin de faire évoluer l'ensemble des systèmes générateurs et ainsi permettre d'avoir une plus grande diversité de créatures. Cela reste tout de même ambitieux, ainsi l'embryogenèse pourrait être une étape intermédiaire comme nous le montre notre diagramme des perspectives (Fig. 7.1). Dans un premier temps, grâce à des machines de plus en plus performantes, il sera certainement possible d'utiliser des primitives géométriques plus complexes que de simples cubes ou cylindres. Dans le même esprit l'apparition des simulations d'objets déformables permettront de produire des créatures encore plus réalistes.

7.3.2 Robotique évolutionnistes

Une perspective intéressante lorsque l'on désire produire physiquement les créatures issues d'une évolution serait de les produire en simulant auparavant plusieurs types de matériaux et en utilisant plusieurs types d'actionneurs (différents moteurs), cela afin de réduire les coûts de fabrication et d'augmenter la fiabilité tout en préservant leur fonctionnalité. Ainsi on peut imaginer avoir un ensemble de critères qui permettent d'obtenir le robot le plus fiable et rentable pour un comportement désiré. Pour l'instant, il est coûteux de réaliser des simulations physiques très réalistes mais cette approche sera certainement réalisable dans quelques années. Cela permettra aussi d'obtenir des créatures physiques plus proches de celle simulées. Dans ce sens Bongard et Lipson proposent déjà de faire évoluer l'environnement simulé afin qu'il corresponde au mieux à la réalité [Bongard and Lipson 2004b].

7.3.3 Parallélisation

Les créatures évolutionnistes demandent beaucoup de puissance de calcul, il est donc souhaitable de paralléliser les calculs et d'utiliser des cartes physiques qui seront bientôt très répandues.

7.3.4 La matérialisation

La matérialisation est la confection des éléments physiques qui constitueront une créatures physiques comme un robot. Cette étape est essentielle dans la matérialisation des créatures artificielles virtuelles en vue de la conception automatique de robots par une machine. Ainsi les imprimantes 3D, comme celle employée par Hod Lipson pour son projet Golem [Lipson and Pollack 2000], sont certainement une première étape vers la confection des éléments intégrant de plus en plus de composants divers. Pour la suite, nous pensons que les évolutions apportées dans ce domaine auront des retombés sur la robotique évolutionniste et la robotique modulaire en permettant de produire des éléments physiques au cours d'une évolution.

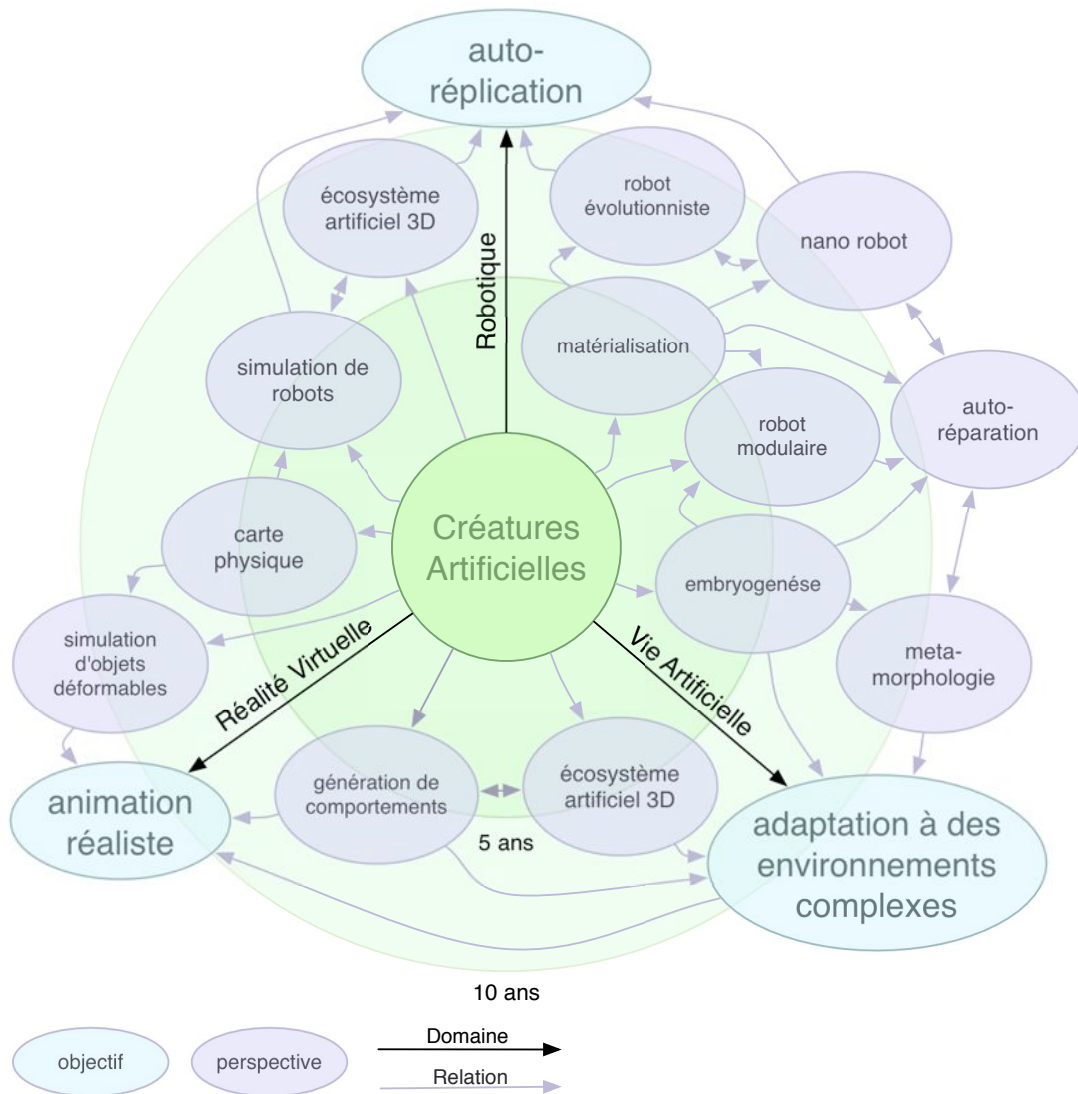


Figure 7.1: Une vision personnelle des perspectives sur les créatures artificielles et de leurs relations avec les objectifs à long terme que nous avons défini dans cette thèse.

Représentation des Objets de l'environnement

A.1 Représentation du StakeBoard en langage Steve

Le code suivant donne la description en langage *Steve* du skateboard qui est utilisé pour nos simulations. Ce code peut être directement inclu sous forme de librairie dans *Breve*,

```
@use PhysicalControl .
@use Shape .
@use Link .
@use MultiBody .
@use Joint .

MultiBody: Skate{
+ variables:
  body_shape(object).
  body(object).
  joint_dot_L(object).
  joint_dot_R(object).
  joint_dot_LB(object).
  joint_dot_RB(object).

  wheel_radio(double).
  wheel_height(double).
  wheel_L(object).
  wheel_R(object).
  wheel_LB(object).
  wheel_RB(object).
  wheel_shape(object).
```

```
LENGTHROBOT (double).
HEIGHTROBOT (double).

+ to init:
  LENGTHROBOT=3.
  HEIGHTROBOT=LENGTHROBOT/15.

body_shape=(new Cube init-with
  size (LENGTHROBOT,HEIGHTROBOT,LENGTHROBOT)).

body=new Link.
body set-shape to body_shape.
body set-color to (0,0,1).

self set-root to body.
wheel_radio=HEIGHTROBOT*2.
wheel_height=HEIGHTROBOT.

wheel_shape=(new PolygonDisk init-with radius wheel_radio
  sides 40 height wheel_height).

wheel_L=new Link.
wheel_L set-shape to wheel_shape.
wheel_L set-color to (0,0,0).
wheel_R=new Link.
wheel_R set-shape to wheel_shape.
wheel_R set-color to (0,0,0).

wheel_LB=new Link.
wheel_LB set-shape to wheel_shape.
wheel_LB set-color to (0,0,0).
wheel_RB=new Link.
wheel_RB set-shape to wheel_shape.
wheel_RB set-color to (0,0,0).

joint_dot_L=new RevoluteJoint.
joint_dot_L set-relative-rotation around-axis(0,0,1) by 1.57.

joint_dot_R=new RevoluteJoint.
```

```
joint_dot_R set-relative-rotation around-axis(0,0,1) by 1.57.

joint_dot_LB=new RevoluteJoint.
joint_dot_LB set-relative-rotation around-axis(0,0,1) by 1.57.

joint_dot_RB=new RevoluteJoint.
joint_dot_RB set-relative-rotation around-axis(0,0,1) by 1.57.

self move to (5,LENGTHROBOT,0).

joint_dot_L link parent body to-child wheel_L with-normal (1,0,0)
with-parent-point (LENGTHROBOT/2,0,-LENGTHROBOT/2)
with-child-point (0,HEIGHTROBOT/2,0).

joint_dot_R link parent body to-child wheel_R with-normal (1,0,0)
with-parent-point (-LENGTHROBOT/2-wheel_height,0,-LENGTHROBOT/2)
with-child-point (0,HEIGHTROBOT/2,0).

joint_dot_LB link parent body to-child wheel_LB with-normal (1,0,0)
with-parent-point (LENGTHROBOT/2,0,LENGTHROBOT/2)
with-child-point (0,HEIGHTROBOT/2,0).

joint_dot_RB link parent body to-child wheel_RB with-normal (1,0,0)
with-parent-point (-LENGTHROBOT/2-wheel_height,0,LENGTHROBOT/2)
with-child-point (0,HEIGHTROBOT/2,0).

joint_dot_L set-joint-velocity to 0.
joint_dot_R set-joint-velocity to 0.
joint_dot_LB set-joint-velocity to 0.
joint_dot_RB set-joint-velocity to 0.

joint_dot_L set-strength-limit to 0.1.
joint_dot_R set-strength-limit to 0.1.
joint_dot_LB set-strength-limit to 0.1.
joint_dot_RB set-strength-limit to 0.1.

joint_dot_L set-joint-damping to 0.
joint_dot_R set-joint-damping to 0.
joint_dot_LB set-joint-damping to 0.
```

```
joint_dot_RB set-joint-damping to 0.  
}
```

Paramètres pour l'évolution des créatures artificielles

A.1 Paramètres principaux

```

POPULATION_SIZE      100

SELECTION             0.7
// taux de selection

MUTATION              0.1
// taux de mutation : chance qu'une mutation soit appliquée sur un
// individu. Chaque mutation a par la suite autant de chance d'être
// appliquée.

REPLACING.....1
// 0: Les enfants sont ajoutés à la population et les parents sont
// conservés. Les individus les moins bien notés en surplus de la
// population sont supprimés
// 1: Les parents sont remplacés par leurs enfants.

TOURNAMENT_SIZE.....3

KEEP_FIRST.....1
// indique le nombre de meilleurs individus de la
// population que l'on conserve d'une génération à l'autre.

Paramètre pour l'initialisation des créatures :
```

```

RECURSIVITY.....3
//_Ce_nombre_peut_augmenter_lors_de_croisements_ou_de_mutations

NB_MAXNODE.....10
//_nombre_de_bloc_maximum_lors_de_l'initialisation.
//_Ce_nombre_peut_augmenter_lors_de_croisement_ou_de_mutations

NB_MAX_ITERATION    5
//_initialise_la_modularité_lors_de_l'initialisation
//_Ce_nombre_peut_augmenter_lors_de_croisements_ou_de_mutations

```

A.2 Génération des patterns

Procédure générant un pattern.

```

PatternSig::PatternSig(){
    float a,b,c,d;
    int i;
    signalPat=new float[SIZE];

    i=random()%2;
    a=(random()%314)/100.0;
    b=(random()%100)/1000.0;
    c=((random()%20000)/1000.0)-10.0;
    d=(random()%10)/10.0;

    if(i==0)
        for(int n=0;n<SIZE;n++)
            signalPat[n]=b*cos(a+2*M_PI*(float)n/((float)SIZE*c));

    if(i==1)
        for(int n=0;n<SIZE;n++)
            signalPat[n]=b*sin(a+2*M_PI*(float)n/((float)SIZE*c));

    if(i==2)
        for(int n=0;n<SIZE;n++)
            signalPat[n]=b*tan(a+2*M_PI*(float)n/((float)SIZE*c))/40.0;

    for(int n=0;n<SIZE;n++){
        if(signalPat[n]>1)

```

```
    signalPat[n]=cos(signalPat[n]);  
  
    if (signalPat[n]<-1)  
        signalPat[n]=cos(signalPat[n]);  
    }  
}
```

Informations complémentaires sur les smallworlds

Afin de faciliter la réimplémentation des algorithmes vous trouverez dans cette section les différents paramètres de la simulation de fourmis ainsi que l'algorithme de déplacement des fourmis. L'ensemble des simulations et des codes sources sont aussi disponibles à l'adresse suivante : <http://www.irit.fr/Nicolas.Lassabe/smallworld/>

A.1 Paramètres principaux : main.h

```
// Affichage graphique
#define GRAPHIC 1

// Quantité de nourriture dans l'environnement
#define NB_FOOD 1024

// Nombre de fourmis dans l'environnement
#define NB_ANTS 1024

// Taille de l'environnement (carré)
#define SIZE_WORLD 495

// Mode de debug si DEUG est à 1
#define DEBUG 0

// Affichage des données du graphe sur la sortie standart
#define OUT 1

// Affichage des labels du graphe sur la sortie standard
```

```
#define LABEL 0

// Répartition de la nourriture sur une grille
#define GRID 1

// Vérification de l'algorithme : Controle que la quantité de
// nourriture reste constante durant la simulation
// (quantité transportée + quantité sur les sites = NBFOOD)
#define CONTROL 0

// Affiche la densité du graphe sur la sortie standard
#define DENSITE 0

// Affiche les connexions du graphe dans l'environnement graphique
#define LINE 1

// Active le mouvement aléatoire des fourmis
#define RAND_MOVE 1

// Active la mémoire des fourmis qui retient la quantité de
// nourriture détectée sur les différents sites
#define MEMORY 0

// Affichage le nombre de connexion sur la sortie standard
#define EDGE 0

// Nombre maximun de connexions dans le graphe
#define MAX_EDGE 50000
```

A.2 Algorithme de déplacement des fourmis : ants.cpp

```
void Ants::move(){
    int nx,ny;
    if (RAND_MOVE==1){
        do{
            ox=random()%3-1;
            oy=random()%3-1;
        } while (ox==0&&oy==0);
    }
}
```

```
else{
  int direction=random()%100;
  if (direction >9)
    direction=0;
  else
    if (direction >4)
      direction=-1;
    else
      direction=1;

  if (ox==0){
    ox=direction;
  }
  else
    if (oy==0){
      oy=direction;
    }
    else{
      if (ox!=0&&oy!=0&&direction!=0){
        if (random()%2)
          ox=0;
        else
          oy=0;
      }
    }
}
nx=x-ox;
ny=y-oy;
ox=x-nx;
oy=y-ny;
if (nx>0&&nx<SIZE_WORLD)
  x=nx;

if (ny>0&&ny<SIZE_WORLD)
  y=ny;
}
```


Proposition d'un modèle d'écosystème 3D

Nous proposons dans cette section les prémices d'un modèle simple d'écosystème artificiel. Il permet d'illustrer en plus des modèles présentés dans l'état de l'art ce que pourrait être un écosystème artificiel peuplé de créatures.

B.1 Écosystème

Nous proposons un modèle d'écosystème artificiel composé de créatures, d'insectes, de plantes artificielles afin de permettre la génération de formes de vie adaptées à un environnement dynamique et ainsi obtenir la spécification d'espèces et l'émergence de cycle de vie (Fig. B.1). En biologie, les écosystèmes permettent une meilleure compréhension de l'évolution et des extinctions des espèces comme le montre les travaux de G. Abramson [Abramson 1997]. En robotique, l'évolution de la morphologie et du comportement dans des écosystèmes a un intérêt pour la conception des robots et de leurs contrôleurs.

B.1.1 Introduction

Le but de notre modèle d'écosystème est de produire des stratégies de survie plus adaptées et plus complexes grâce à une interaction importante entre les créatures.

Nous ne voulons pas nous focaliser sur une représentation spécifique de la morphologie ou du contrôleur des créatures. Ainsi nous voulons que notre écosystème puisse être capable facilement de supporter différent type de créatures et permettre leurs interactions.

Nous aimerions aussi qu'il permette le développement de stratégies de proies et



Figure B.1: Une vue de notre écosystème.

de prédateurs afin d'obtenir des chaînes alimentaires complètes (Fig. B.2). Dans ce modèle, chaque créature possède une quantité d'énergie provenant de la lumière, de l'apport de nourriture consommée. Une créature meurt quand il ne lui reste plus d'énergie ou quand elle a été mangée par une autre créature.

B.1.2 La chaîne alimentaire

L'alimentation d'une créature est représentée dans ses gènes. Ainsi les liens déterminent dans la chaîne alimentaire quelle créature est autorisée à en manger une autre mais aussi les aliments qui sont en bas de la chaîne (Fig. B.2). Une créature peut manger cinq type aliments : oeuf, graines, plantes, insectes et autres créatures. Les liens peuvent donc être encodés dans un des gènes de la créature en utilisant cinq bits. Ainsi si un bit est à un, la créature pourra manger l'aliment correspondant. Par exemple, le gène 01010 permet à une créature de manger seulement des insectes et des graines. Une créature, si son gène le permet, ne pourra manger que les créatures d'une autre espèce et que si elle est plus petite quelle.

Afin obtenir des comportements complexes, nous définissons pour chaque créature une couleur qui leur est propre, une couleur d'attraction définie par un autre gène et une couleur répulsive. Par l'intermédiaire de ces nouveaux gènes nous

espérons voir l'apparition de stratégies d'adaptation supplémentaire. Ainsi on peut imaginer que la couleur d'un prédateur peut évoluer afin d'attirer les créatures qu'il est autorisé à manger. Cela provoquera des stratégies d'adaptation par le biais du changement de gènes de couleurs des proies et ainsi rétablira un nouvel équilibre des populations.

B.1.3 La reproduction des espèces

La reproduction s'effectue par rapport à la compatibilité entre les gènes¹. Ainsi plusieurs reproductions successives doivent garantir aux créatures d'une même espèce de faibles variations de leurs caractéristiques. On peut imaginer que la distance tolérée entre les gènes soit une caractéristique du génome et devienne ainsi variable. Toutefois, les créatures doivent respecter une certaine différence de taille qui elle aussi pourra évoluer. On pourra considérer que les processus de reproduction de créatures prend plus de temps en proportion de leur taille. Ainsi on peut prédire que la stratégie de survie des petites espèces devraient être de se reproduire en grand nombre afin d'être moins vulnérable.

B.1.4 Le concept de poison

Nous proposons aussi d'introduire le concept de poison dans notre écosystème. Cela permettrait d'obtenir des stratégies de défense pour les plantes ou les petites créatures. Le poison serait représenté par un gène de huit bits. Ainsi, la possibilité d'une créature à manger une autre créature sans être empoisonnée serait basée le fait que ces huit bits soient complémentaires avec ceux de la créature mangée. Par exemple, si le poison d'une espèce est 0101110101, le gène complémentaire permettant de consommer cette espèce est 1010001010. Ainsi lorsqu'une créature mange une autre et que celle-ci n'est pas comestible elle perd une certaine quantité d'énergie. Pour connaître, le pourcentage d'énergie perdue lors d'un empoisonnement, nous calculons le pourcentage de bits qui ne sont pas complémentaires. Ainsi si aucun bit n'est complémentaire la créature mourra. Si la moitié les sont, les créatures perdra cinquante pour cent de ses capacités totales². De plus, nous proposons d'utiliser le caractère # pour permettre la représentation du 0 et du 1. Ainsi, le gène ##### protège contre tous les poisons, mais il ne permet pas aux créatures de se défendre car

¹Les gènes compatibles font partie d'une même espèce. La compatibilité entre deux génomes peut se mesurer grâce à la distance de Hamming [Hamming 1950].

²Ces valeurs sont subjectives.

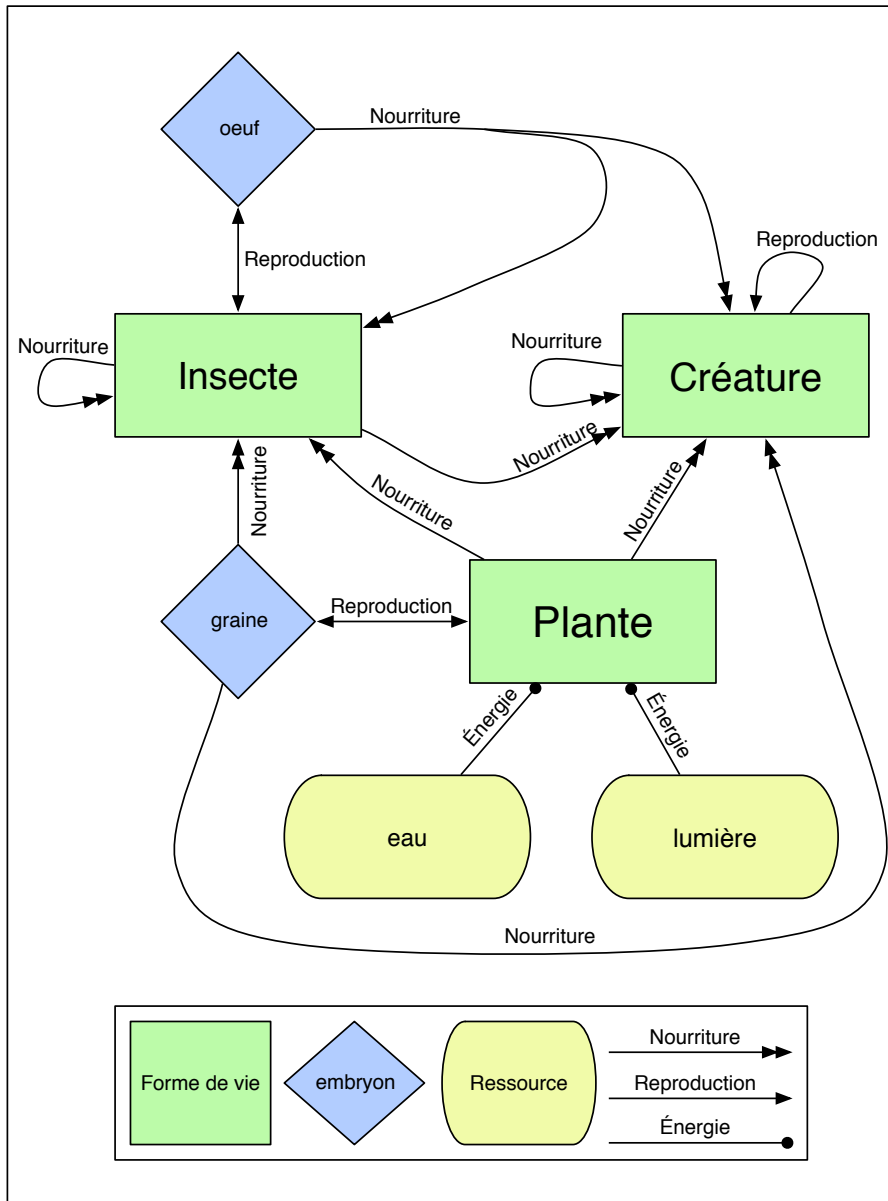


Figure B.2: Exemple de chaîne alimentaire qui pourrait être produit par un écosystème artificielle.

son poison est inoffensif.

B.1.5 L'énergie

L'énergie est nécessaire à la survie des créatures. Elles l'utilisent entre autre pour se déplacer, ainsi si leur mouvement n'est pas efficace, elle pourront difficilement trouver de la nourriture. La quantité d'énergie dépensée pour les déplacements dépend du poids des créatures et de l'ampleur des mouvements qu'elles effectuent. Les créatures utilisent aussi l'énergie pour se reproduire.

En ce qui concerne les plantes, l'énergie permet de produire des graines. Ainsi une quantité importante d'énergie offre de meilleures de chances de se reproduire. Pour ce qui concerne la croissance des plantes, l'énergie leur permet de grandir plus vite à condition quelles disposent de ressources en eau. L'énergie des plantes provient de ses feuilles qui la collectent à partir de la lumière. Il faut noter que plus une plante est grande est plus elle aura besoin d'eau. Une des stratégies pour les plantes sera de trouver un bon compromis entre le nombre de feuilles, le nombre de graines qu'elles produisent et la quantité d'eau qui est nécessaire à leur survie.

B.1.6 Conclusion

Afin de valider notre modèle d'écosystème artificiel, une première étape sera de l'initialiser avec des populations aléatoires afin de voir vers quel état l'écosystème converge. En effet, il peut converger vers un équilibre stable si par exemple sa population se réduit et finit par disparaître ou il peut se maintenir dans un équilibre instable. Dans ce dernier cas, il est intéressant d'observer les mécanismes et les stratégies d'adaptations qui permettent cet équilibre instable, c'est à dire une auto-régulation du système. Dans le cas d'une convergence rapide vers un équilibre stable, il faudra déterminer les causes et les paramètres qui entraînent la disparition des espèces. Dans la mesure du possible, il serait intéressant de trouver l'ensembles des valeurs limites qui permettent l'auto-régulation du système.

Le modèle d'écosystème que nous proposons comporte de nombreux paramètres dont il peut être difficile de mesurer l'influence. Il serait donc intéressant d'évaluer leur influence en les activant individuellement au cours de différentes simulations. Des outils comme celui qui nous avons présenté pour mesurer l'évolution de la complexité peuvent permettre de comparer et d'améliorer ces simulations [Lassabe et al. 2007].

En ce qui concerne le concept de poison et d'attraction par le biais de la couleur des créatures, il serait intéressant d'étudier l'évolution du nombre d'espèces, leur extinction et l'évolution de leur population pour ainsi déterminer les stratégies d'adaptations qui apparaissent.

Bibliographie

- ABRAMSON, G. 1997. Ecological model of extinctions. *Phys. Rev. E* 55, 1, 785. (pp.95, 127)
- ACHACOSO, T. B. AND YAMAMOTO, W. S. 1992. *AYs Neuroanatomy of C Elegans for Computation*. CRC-Press. (pp.94, 97)
- ADAMATZKY, A. AND KOMOSINSKI, M. 2005. *Artificial Life Models in Software*. Springer-Verlag. (p.45)
- ADAMI, C. AND BROWN, C. T. 1994. Evolutionary learning in the 2d artificial life system "avida". In *Artificial Life IV* (1994), pp. 337. (pp.47, 48, 49)
- ADAMI, C., OFRIA, C., AND COLLIER, T. C. 2000. Evolution of biological complexity. *PROC.NAT.ACAD.SCI* 97, 4463. (pp.18, 94, 95)
- ALBERT, R., JEONG, H., AND BARABASI, A. L. 1999. The diameter of the world wide web. *Nature* 401, p1300131. (p.94)
- AUTONES, M., BECK, A., CAMACHO, P., LASSABE, N., LUGA, H., AND SCHARFFE, F. 2004. Evaluation of chess position by modular neural network generated by genetic algorithm. In *Genetic Programming 7th European Conference* (2004), pp. 1–10. (p.22)
- BAR-YAM, Y. 1997. *Dynamics of complex systems*. Perseus Books. (p.95)
- BARAFF, D. 1996. Linear-time dynamics using Lagrange multipliers. *Computer Graphics 30*, Annual Conference Series, 137–146. (p.32)
- BAUMGARTE, J. 1972. Stabilization of constraints and integrals of motion in dynamical systems. In *Computer Methods in Applied Mechanics and Engineering 1* (1972), pp. 1–16. (p.32)
- BONGARD, J. AND PAUL, C. 2000. Investigating morphological symmetry and locomotive efficiency using virtual embodied evolution. In J.-A. M. ET AL. Ed., *From Animals to Animats: The Sixth International Conference on the Simulation of Adaptive Behaviour* (2000). (pp.16, 95)

- BONGARD, J. AND PFEIFER, R. 2003. Evolving complete agents using artificial ontogeny. In *In Morpho-functional Machines: The New Species* (2003), pp. 237–258. (pp. 50, 52, 61, 89)
- BONGARD, J. AND PFEIFER, R. 2006. *How the Body Shapes the Way We Think: A New View of Intelligence*. Bradford Books. (p. 3)
- BONGARD, J. C. AND LIPSON, H. 2004a. Integrated design, deployment and inference for robot ecologies. In *Proceedings of Robosphere 2004* (2004). (pp. 19, 20)
- BONGARD, J. C. AND LIPSON, H. 2004b. Once more unto the breach, automated tuning of robot simulation using an inverse evolutionary algorithm. In *Proceedings of the Ninth Int. Conference on Artificial Life (ALIFE IX)* (2004). (pp. 19, 112)
- BROOKS, R. 1992. Artificial life and real robots. In *European Conference on Artificial Life* (1992), pp. 3–10. (pp. 15, 18)
- BROOKS, R. A. 1991a. How to build complete creatures rather than isolated cognitive simulators. In *Architectures for Intelligence* (1991), pp. 225–240. (p. 49)
- BROOKS, R. A. 1991b. Intelligence without representation. *Artificial Intelligence* 47, 1-3, 139–159. (p. 4)
- BROOKS, R. A., SELMAN, B., DEAN, T., HORVITZ, E., MITCHELL, T. M., AND NILSSON, N. J. 1996. Challenge problems for artificial intelligence. In *Thirteenth National Conference on Artificial Intelligence - AAAI* (1996), pp. 1340–1345. (p. 4)
- BUHL, J., GAUTRAIS, J., SOLÉ, R. V., KUNTZ, P., VALVERDE, S., AND DENEUBOURG, J. L. 2004. Efficiency and robustness in ant networks of galleries. *European Physical Journal B* vol. 42, pp. 123–129. (p. 94)
- CAMAZINE, S., FRANKS, N. R., SNEYD, J., BONABEAU, E., DENEUBOURG, J.-L., AND THERAULAZ, G. 2001. *Self-Organization in Biological Systems*. Princeton University Press. (p. 95)
- CHAUMONT, N., EGLI, R., AND ADAMI, C. 2007. Evolving virtual creatures and catapults. *Artificial Life* 13, 2, 159–187. (pp. 39, 43, 62, 73, 89)
- CHAVOYA, A. AND DUTHEN, Y. 2007a. An artificial development model for cell pattern generation. In *Progress in Artificial Life, Third Australian Conference* (2007), pp. 61–71. (pp. 19, 50)
- CHAVOYA, A. AND DUTHEN, Y. 2007b. Use of a genetic algorithm to evolve an extended artificial regulatory network for cell pattern generation. In *GECCO* (2007), pp. 1062. (pp. 19, 50, 51)

-
- CHRISTOPHER G. LANGTON, J. D. F., CHARLES TAYLOR AND RASMUSSEN, S. Eds. 1992. *Artificial Life II, Proceedings of the Workshop on Artificial Life. Held February, 1990 in Santa Fe, New Mexico, Proceedings Volume X, Santa Fe Institute Studies in the Sciences of Complexity* (1992). (pp.15, 18)
- CLIFF, D. AND MILLER, G. F. 1996. Co-evolution of pursuit and evasion II: Simulation methods and results. In P. MAES, M. J. MATARIC, J.-A. MEYER, J. B. POLLACK, AND S. W. WILSON Eds., *From animals to animats 4* (Cambridge, MA, 1996), pp. 506–515. MIT Press. (p.20)
- CUSSAT-BLANC, S., LUGA, H., AND DUTHEN., Y. 2007. Using a single cell to create an entire organ. In *International Conference on Artificial Reality and Telexistence* (2007), pp. p300–301. (p.50)
- DALVAND, M. M. AND MOGHADAM, M. M. 2006. Stair climber smart mobile robot (msrox). *Auton. Robots* 20, 1, 3–14. (p.77)
- DARWIN, C. 1859. *The origin of species by Means of Natural Selection*. John Murray, London. (p.13)
- DAWKINS, R. 1986. *The blind watchmaker*. Longmans. (pp.27, 33, 49)
- DE GARIS, H. 1990. Genetic programming: building artificial nervous systems using genetically programmed neural network modules. In *Machine Learning: Proceedings of the Seventh International Conference* (1990). (pp.28, 48)
- DELLAERT, F. AND BEER, R. 1996. A developmental model for the evolution of complete autonomous agents. In *SAB '96* (1996), pp. 393–401. (p.49)
- DENEBOURG, J., PASTEELS, J., AND VERHAEGH, J. 1983. Probabilistic behaviour in ants : a strategy of errors? *Journal of Theoretical Biology* 105, 259–271. (p.98)
- DONCIEUX, S. AND MEYER, J.-A. 2003. Evolving neural networks for the control of a lenticular blimp. In *euroGP* (2003). (p.42)
- DORIGO, M. 1992. *Optimization, Learning and Natural Algorithms*. PhD thesis, Politecnico di Milano. (p.98)
- DOURSAT, R. 2006. The growing canvas of biological development: multiscale pattern generation on an expanding lattice of gene regulatory networks. *InterJournal Complex Systems*. (pp.19, 50)
- DOURSAT, R. 2007a. *Organic Computing*, Chapter Organically Grown Architectures: Creating Decentralized, Autonomous Systems by Embryomorphic Engineering. Springer-Verlag. (pp.19, 50)

- DOURSAT, R. 2007b. The self-made puzzle: Integrating self-assembly and pattern formation under non-random genetic regulation. In *7th International Conference on Complex Systems* (2007). (pp.19, 50, 51)
- EGGENBERGER, P. 1997. Evolving morphologies of simulated 3D organisms based on differential gene expression. In P. HUSBANDS AND I. HARVEY Eds., *Proceedings of the Fourth European Conference on Artificial Life, ECAL97*, pp. 205–213. MIT Press. (p.16)
- FARMER, D. F. AND BELIN, A. 1989. Artificial life: the coming evolution. In *Artificial Life* (1989), pp. 815–840. (pp.6, 11)
- FEATHERSTONE, R. 1987. *Robot Dynamics Algorithms*. Kluwer Academic Publishers. (p.29)
- FLEISCHER, K. 1996. Investigations with a multicellular developmental model. In *Artificial Life V* (1996), pp. 229–236. (pp.19, 50)
- FOGEL, L., OWENS, A. J., AND WALSH, M. J. 1966. *Artificial Intelligence through Simulated Evolution*. wiley. (p.14)
- GARCIA CARBAJAL, S., MORAN, M. B., AND MARTINEZ, F. G. 2004. EvolGL: life in a pond. In J. POLLACK, M. BEDAU, P. HUSBANDS, T. IKEGAMI, AND R. A. WATSON Eds., *Artificial Life XI Ninth International Conference on the Simulation and Synthesis of Living Systems* (Boston, Massachusetts, 12-15Sept. 2004), pp. 75–80. The MIT Press. (p.17)
- GAUME, B. 2008. Mapping the forms of meaning in small world. *International Journal of Intelligent Systems* 23, 1–15. (pp.96, 107)
- GAURE, J. 1990. Six degrees of separation: A play. *Vintage Books, New York*. (pp.94, 97)
- GOLDBERG, D. E. 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley. (pp.11, 14, 27)
- GRITZ, L. AND HAHN, J. K. 1997. Genetic programming evolution of controllers for 3-D character animation. In J. R. KOZA, K. DEB, M. DORIGO, D. B. FOGEL, M. GARZON, H. IBA, AND R. L. RIOLO Eds., *Genetic Programming 1997: Proceedings of the Second Annual Conference* (1997), pp. 139–146. Morgan Kaufman. (p.20)
- GRUAU, F. 1994. *Neural Network Synthesis using Cellular Encoding and the Genetic Algorithm*. PhD thesis, France. (p.23)

-
- GRUAU, F. 1995. Modular genetic neural networks for six-legged locomotion. In *Artificial Evolution* (1995), pp. 201–219. (p.20)
- HAMMING, R. W. 1950. Error detecting and error correcting codes. *Bell System Technical Journal* 26, 2, p147–160. (p.129)
- HART, J. C. 1992. The object instancing paradigm for linear fractal modeling. In *Proceedings of Graphics Interface '92* (1992), pp. 224–231. (p.28)
- HEGUY, O., SANCHEZ, S., BERRO, A., AND LUGA, H. 2004. Generic classifiers system and learning behaviours in virtual worlds. In *CW '04: Proceedings of the 2004 International Conference on Cyberworlds (CW'04)* (Washington, DC, USA, 2004), pp. 113–120. IEEE Computer Society. (p.110)
- HEUDIN, J.-C. 1998. *Virtual Worlds: Synthetic Universes, Digital Life, and Complexity*, pp. 1–28. Persus Books. (p.48)
- HOLLAND, J. H. 1975. *Adaptation in Natural and Artificial Systems*. University of Michigan Press. (pp.11, 13)
- HOLLAND, J. H. AND REITMAN, J. S. 1978. Cognitive systems based on adaptive algorithms. *SIGART Bull.* 1, 63, 49–49. (pp.14, 27)
- HORNBY, G. AND POLLACK, J. 2002. Creating high-level components with a generative representation for body-brain evolution. (p.35)
- HORNBY, G. AND POLLACK, J. B. 2001. Evolving l-systems to generate virtual creatures. *Computers & Graphics* 25, 6, 1041–1048. (pp.15, 16, 17, 22, 35, 37, 38, 61, 62, 73, 89)
- IJSPEERT, A. J. 2000. A 3-d biomechanical model of the salamander. In *Virtual Worlds* (2000), pp. 225–234. (p.19)
- JEONG, H., MASON, S. P., BARABASI, A.-L., AND OLTVAI, Z. N. 411. Lethality and centrality in protein networks. *Nature* 41, 2001. (p.107)
- KAUFFMAN, S. A. 1993. *The Origins of Order: Self-Organization and Selection in Evolution*. Oxford University Press. (pp.13, 95)
- KIM, K.-J. AND CHO, S.-B. 2006. A comprehensive overview of the applications of artificial life. *Artificial Life* 12, 1, 153–182. (p.14)
- KITANO, H. 1990. Designing neural networks using genetic algorithms with graph generation system. *Complex Systems* 4, 4, 461–476. (p.28)

- KLEIN, J. 2003. Breve: a 3d environment for the simulation of decentralized systems and artificial life. In *ICAL 2003: Proceedings of the eighth international conference on Artificial life* (Cambridge, MA, USA, 2003), pp. 329–334. MIT Press. (pp. 16, 59)
- KOMOSINSKI, M. 2005. Framsticks: a platform for modeling, simulating and evolving 3D creatures. In *Artificial Life Models in Software*, Chapter 2, pp. 37–66. Springer-Verlag. (p. 45)
- KOMOSINSKI, M. 2000. The world of framsticks: Simulation, evolution, interaction. In *VW '00: Proceedings of the Second International Conference on Virtual Worlds* (London, UK, 2000), pp. 214–224. Springer-Verlag. (pp. 15, 16, 17, 42, 45, 64, 89)
- KOMOSINSKI, M. 2003. The Framsticks system: versatile simulator of 3D agents and their evolution. *Kybernetes: The International Journal of Systems & Cybernetics* 32, 156–173. (p. 45)
- KOMOSINSKI, M., KOCZYK, G., AND KUBIAK, M. 2001. On estimating similarity of artificial and real organisms. *Theory in Biosciences* 120, 3-4, 271–286. (p. 45)
- KOMOSINSKI, M. AND KUBIAK, M. 2001. Taxonomy in Alife. Measures of similarity for complex artificial organisms. In *Advances in Artificial Life. Lecture Notes in Artificial Intelligence* 2159, pp. 685–694. Springer-Verlag. (p. 45)
- KOMOSINSKI, M. AND ROTARU-VARGA, A. 2001. Comparison of different genotype encodings for simulated 3D agents. *Artificial Life Journal* 7, 4, 395–418. (p. 45)
- KOMOSINSKI, M. AND ULATOWSKI, S. 1999. Framsticks: towards a simulation of a nature-like world, creatures and evolution. In *Advances in Artificial Life. Lecture Notes in Artificial Intelligence* 1674, pp. 261–265. Springer-Verlag. (pp. 45, 80)
- KOZA, J. R. 1991. Evolution and co-evolution of computer programs to control independent-acting agents. In J.-A. MEYER AND S. W. WILSON Eds., *From Animals to Animats: Proceedings of the First International Conference on Simulation of Adaptive Behavior, 24-28, September 1990* (Paris, France, 1991), pp. 366–375. MIT Press. (p. 18)
- KOZA, J. R. 1992. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press. (p. 14)
- KOZA, J. R., KEANE, M. A., J. STREETER, M., MYDLOWEC, W., YU, J., AND LANZA, G. 2003. *Genetic Programming IV: Routine Human-Competitive Machine Intelligence*. Kluwer Academic Publishers. (p. 14)
- KRUEGER, M. W. 1991. *Artificial Reality 2 (2nd Edition)*. Addison-Wesley Professional. (p. 27)

-
- LANGTON, C. 1989a. Artificial life. In *Artificial Life* (1989), pp. 1–47. (p. 11)
- LANGTON, C. Ed. 1989b. *Proceedings of the Interdisciplinary Workshop on the Synthesis and Simulation of Living Systems (ALIFE '87), Los Alamos, NM, USA, September 1987*, Volume 6 of *Santa Fe Institute Studies in the Sciences of Complexity* (1989). Addison-Wesley. (pp. 15, 18)
- LASSABE, N., BERRO, A., AND DUTHEN, Y. 2007. Improvement of a shortest routes algorithm. In *International Conference on Intelligent Transportation Systems IEEE (2007)*, pp. p. 613–617. (p. 94)
- LASSABE, N., GAUME, B., LUGA, H., AND DUTHEN, Y. 2007. Is there a small-world in the small world of ants? In *Internationnal Conference on Complex Systems* (2007). (pp. 93, 131)
- LASSABE, N., GAUME, B., LUGA, H., AND DUTHEN, Y. 2008. Is there a small-world in the small world of ants? *InterJournal of Complex Systems* 2200. (p. 93)
- LASSABE, N., LUGA, H., AND DUTHEN, Y. 2006. Evolving creatures in virtual ecosystems. In *16th International Conference on Artificial Reality and Telexistence* (2006), pp. 11–20. (pp. 57, 71, 96)
- LASSABE, N., LUGA, H., AND DUTHEN, Y. 2007a. Interaction of evolving artificial creatures. In *International Conference on Enactive Interfaces* (2007), pp. p. 137–140. (pp. 57, 71)
- LASSABE, N., LUGA, H., AND DUTHEN, Y. 2007b. A new step for evolving creatures. In *IEEE-ALife'07* (2007), pp. 243–251. IEEE. (pp. 57, 71, 89, 96)
- LASSABE, N., LUGA, H., AND DUTHEN, Y. 2008. New skills for evolving artificial creatures. *International Journal of Information Technology and Intelligent Computing, IEEE-CIS to appear* 4. (pp. 57, 71)
- LASSABE, N., SANCHEZ, S., LUGA, H., AND DUTHEN, Y. 2006. Genetically programmed strategies for chess endgame. In *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation* (New York, NY, USA, 2006), pp. 831–838. ACM Press. (p. 4)
- LEE, W.-P., HALLAM, J., AND LUND, H. H. 1996. A hybrid gp/ga approach for co-evolving controllers and robot bodies to achieve fitness-specified tasks. In *Proceedings of the 1996 IEEE International Conference on Evolutionary Computation* (1996), pp. 84–89. IEEE Press. (p. 19)

- LENSKI, R., OFRIA, C., PENNOCK, R., AND ADAMI, C. 2003. The evolutionary origin of complex features. *Nature* 423, 139–145. (p.47)
- LINDENMAYER, A. 1968. Mathematical models for cellular interaction in development, parts i and ii. *Journal of Theoretical Biology* 18, 180–315. (pp.17, 28)
- LIPSON, H. 2004. How to draw a straight line using a GP: Benchmarking evolutionary design against 19th century kinematic synthesis. In M. KEIJZE Ed., *Late Breaking Papers at the 2004 Genetic and Evolutionary Computation Conference* (2004). (p.23)
- LIPSON, H. 2005. Evolutionary robotics and open-ended design automation. (pp.19, 22)
- LIPSON, H. 2007. Curious and creative machines. In *50 Years of AI* (2007), pp. 316–320. (p.2)
- LIPSON, H. AND POLLACK, J. B. 2000. Automatic design and manufacture of artificial lifeforms. *Nature* 406. (pp. 15, 16, 17, 22, 30, 35, 40, 41, 73, 89, 112)
- LUGA, H. 1997. *Vie artificielle et synthèse d'images : étude des mécanismes évolutionnistes pour la synthèse de formes et de comportements*. PhD thesis, Université Paul Sabatier. (p.20)
- LUKE, S. AND SPECTOR, L. 1996. Evolving graphs and networks with edge encoding: Preliminary report. In J. R. KOZ Ed., *Late Breaking Papers at the Genetic Programming* (1996), pp. 117–124. Stanford Bookstore. (p.22)
- MANDERICK, F. AND MOYSON, B. 1988. The collective behavior of ants: An example of self-organization in massive parallelism. *Proceedings of the AAAI Spring Symposium on Parallel Models of Intelligence*. (p.98)
- MATURANA, H. AND VARELA, F. 1980. *Autopoiesis and Cognition: The realization of the living*. D.Reidel, Boston. (p.12)
- MCCULLOCH, W. AND PITTS, W. 1943. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics* 7, 115 – 133. (p.42)
- MCILROY, M. D., MORRIS, R., AND VYSSOTSKY, V. A. 1971. Darwin, a game of survival of the fittest among programs. Technical report, Bell Labs. (p.46)
- MCKENNA, M. AND ZELTZER, D. 1990. Dynamic simulation of autonomous legged locomotion. In *SIGGRAPH '90: Proceedings of the 17th annual conference on Computer graphics and interactive techniques* (1990), pp. 29–38. (pp.19, 27)

-
- MÉTIVIER, M., LATTAUD, C., AND HEUDIN, J. 2002. A stress-based speciation model in lifestream. In *Proceedings of the 8th Int. Conf. On Artificial Life* (2002), pp. 121–126. MIT Press. (pp. 48, 50)
- MICONI, T. 2007. *THE ROAD TO EVERYWHERE Evolution, complexity and progress in Nature and in computers*. PhD thesis, The University of Birmingham. (pp. 32, 42)
- MICONI, T. AND CHANNON, A. 2005. A virtual creatures model for studies in artificial evolution. (pp. 17, 32, 42, 44, 62, 73)
- MICONI, T. AND CHANNON, A. 2006a. An improved system for artificial creatures evolution. (pp. 17, 32, 42, 85, 89)
- MICONI, T. AND CHANNON, A. 2006b. The n-strikes-out algorithm: A steady-state algorithm for coevolution. In *the IEEE Congress on Evolutionary Computation* (2006). (pp. 16, 17, 32, 42)
- MILLER, G. S. P. 1988. Dynamic simulation of autonomous legged locomotion. In *SIGGRAPH* (1988), pp. 169–173. (p. 27)
- MJOLSNESS, E., SHARP, D. H., AND ALPERT, B. K. 1989. Scaling, machine learning, and genetic neural nets. *Advances in Applied Mathematics* 10, 137–163. (p. 28)
- NEWMAN, M. E. J. 2003. The structure and function of complex networks. *SIAM Review* 45, 2, 167–256. (pp. 94, 96, 97, 101)
- NGO, J. T. AND MARKS, J. 1993. Spacetime constraints revisited. *Computer Graphics* 27, 343–350. (p. 28)
- NOLFI, S., FLOREANO, D., MIGLINO, O., AND MONDADA, F. 1994. How to evolve autonomous robots: different approaches in evolutionary robotics. In *Proceedings of the International Conference Artificial Life IV* (1994), pp. 190–197. (p. 19)
- O’KELLYM, M. J. T. AND HSIAO, K. 2004. Evolving simulated mutually perceptive creatures for combat. In I. T. BEDAU M., HUSBANDS P. AND W. R. A. Eds., *Proc. of Artificial Life IX, ed.* (2004), pp. p111–118. (pp. 16, 17, 42)
- PAUL, C. AND BONGARD, J. 2001. The road less travelled: Morphology in the optimization of biped robot locomotion. In *International Conference on Intelligent Robots and Systems, Volume 1* (2001). (p. 21)
- POLLACK, J. B., HORNBY, G. S., LIPSON, H. L., AND FUNES, P. 2003. Computer creativity in the automatic design of robots. *LEONARDO* 2, 36, 115–121. (pp. 22, 35)

- PRIGOGINE, I. AND STEGERS, I. 1987. *Order out of Chaos, Bantam Books*. Bantam Books. (p.95)
- RAIBERT, M. AND HODGINS, J. K. 1991. Animation of dynamic legged locomotion. In *SIGGRAPH 91 (1991)*, pp. 349–358. (p.27)
- RAY, T. S. 1991a. An approach to the synthesis of life. In *Artificial Life II (1991)*, pp. 371–408. (pp.46, 47, 48, 57)
- RAY, T. S. 1991b. Evolution and optimization of digital organisms. In *Billingsley K. R., E. Derohanes, H. Brown, III [eds.], Scientific Excellence in Supercomputing: The IBM 1990 Contest Prize Papers, Athens, (1991)*. (p.46)
- RAY, T. S. 2000. Aesthetically evolved virtual pets. In *Artificial Life 7 workshop proceedings (2000)*, pp. p158–161. (pp.27, 33, 36)
- RECHENBERG, I. 1965. Cybernetic solution path of an experimental problem. *Royal Aircraft Establishment Library Translation 1122*. (p.14)
- REYNOLDS, C. W. 1999. Steering behaviors for autonomous characters. In *Game Developer Conference (1999)*, pp. 763–782. (p.49)
- SCHNAARS, S. P. 1989. The evolution of technology : George basalla, (cambridge university press, new york, 1988). *International Journal of Forecasting 5, 4*, 611–612. (p.2)
- SETH, A. K. 1998. The evolution of complexity and the value of variability. In *ALIFE VI (1998)*, pp. 209–218. (pp.5, 18, 95)
- SETH, A. K. AND EDELMAN, G. M. 2004. Environment and behavior influence the complexity of evolved neural networks. *Adaptive Behavior 12, 1*, 5–20. (p.18)
- SHANNON, C. E. 1948. A mathematical theory of communication. *Bell System Technical Journal 27*, p379–423. (p.95)
- SHIM, Y.-S. AND KIM, C.-H. 2003. Generating flying creatures using body-brain co-evolution. In *SCA '03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation (2003)*, pp. 276–285. (p.42)
- SHIM, Y.-S. AND KIM, C.-H. 2007. Evolving physically simulated flying creatures for efficient cruising. *Artificial Life 12, 4*, 561–591. (p.42)
- SHIM, Y.-S., KIM, S.-J., AND KIM, C.-H. 2004. Evolving flying creatures with path following behaviour. In *Proceedings of 9th International Conference on the Simulation and Synthesis of Living Systems/ (ALIFE IX) (2004)*, pp. 125–132. *Artificial Life*. (pp.16, 17, 42, 44, 89)

-
- SIMS, K. 1991. Artificial evolution for computer graphics. In *SIGGRAPH (1991)*, pp. 319–328. (pp.27, 35)
- SIMS, K. 1992. Interactive evolution of dynamical systems. In 171-178 Ed., *Towards a Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life (1992)*. (p.27)
- SIMS, K. 1994a. Evolving 3d morphology and behavior by competition. In *Artificial Life IV (1994)*, pp. pp. 28–39. Artificial Life: MIT Press. (pp. 15, 16, 22, 27, 33, 57, 61, 62, 63, 65, 68, 72, 85, 89)
- SIMS, K. 1994b. Evolving 3d morphology and behavior by competition. *Artificial Life 1*, 4, 353–372. (pp. 16, 27, 62)
- SIMS, K. 1994c. Evolving virtual creatures. In *Siggraph'94 (1994)*, pp. pp. 15–22. (pp. 15, 16, 22, 27, 29, 30, 31, 33, 62, 68)
- SMITH, A. R. 1984. Plants, fractals, and formal languages. In *SIGGRAPH '84: Proceedings of the 11th annual conference on Computer graphics and interactive techniques (1984)*, pp. 1–10. (p.28)
- STANDISH, R. K. 2004. The influence of parsimony and randomness on complexity growth in tierra. In *ALife IX Workshop and Tutorial Proceedings (2004)*. (p.46)
- STANLEY, K. O. AND MIIKKULAINEN, R. 2002. Efficient reinforcement learning through evolving neural network topologies. In *GECCO '02: Proceedings of the Genetic and Evolutionary Computation Conference (2002)*. (p.107)
- STEWART, J. AND MOSSIO, M. 2007. Is "life" computational? In *International Conference on Enactive Interfaces (2007)*, pp. 271–276. (p.47)
- TAYLOR, T. 2000. Artificial life techniques for generating controllers for physically modelled characters. In *Proceedings of the First International Conference on Intelligent Games and Simulation (2000)*. (p.32)
- TAYLOR, T. AND MASSEY, C. 2000. Recent developments in the evolution of morphologies and controllers for physically simulated creatures. *Artificial Life 7*, 1, 77–87. (pp. 15, 16, 30, 32, 34, 35, 39, 42, 58, 62, 73, 89)
- TEO, J. AND ABBASS, H. A. 2005. Multiobjectivity and complexity in embodied cognition. *IEEE Trans. Evolutionary Computation 9*, 4, 337–360. (p.16)
- TERZOPOULOS, D., TU, X., AND GRZESZCZUK, R. 1994. Artificial fishes : Autonomous locomotion, perception, behavior, and learning in a simulated physical world. *Artificial Life 1*, 4, 327–351. (p.19)

- TODD, S. AND LATHAM, W. 1992. *Evolutionary Art and Computers*. Academic Press. (p.27)
- VAN DE PANNE, M. 1993. Sensor-actuator networks. In *SIGGRAPH '93: Proceedings of the 20th annual conference on Computer graphics and interactive techniques* (1993), pp. 335–342. ACM. (pp.19, 28, 32, 65)
- VENTRELLA, J. 1994. Explorations in the emergence of morphology and locomotion behavior in animated characters. In *Proceedings of the 4th International Workshop on the Synthesis and Simulation of Living Systems ArtificialLifeIV* (Cambridge, MA, USA, 1994), pp. 436–441. MIT Press. (p.47)
- VENTRELLA, J. 1996. Sexual swimmers: Emergence morphology and locomotion behavior. In *Artificial Life IV* (1996), pp. 484–493. (p.47)
- VENTRELLA, J. 1998a. Attractiveness vs. efficiency (how mate preference affects location in the evolution of artificial swimming organisms). In *ALIFE: Proceedings of the sixth international conference on Artificial life* (Cambridge, MA, USA, 1998), pp. 178–186. MIT Press. (pp.17, 47)
- VENTRELLA, J. 1998b. Designing emergence in animated artificial life worlds. In *VW '98: Proceedings of the First International Conference on Virtual Worlds* (London, UK, 1998), pp. 143–155. Springer-Verlag. (pp.17, 47)
- VENTRELLA, J. 2005. *Artificial Life Models in Software*, Chapter Gene Pool: Exploring the Interaction Between Natural Selection and Sexual Selection. Springer-Verlag. (pp.16, 47, 48, 49, 57, 107)
- VON BERTALANFFY, L. 1973. *Théorie générale des systèmes*. Dunod. (p.13)
- WASSERMAN, S. AND FAUST, K. 1994. *Social Network Analysis: Methods and Applications*. Cambridge University Press. (pp.94, 97)
- WATTS, D. J. AND STROGATZ, S. H. 1998. Collective dynamics of “small-world” networks. *Nature* 393, 440–442. (pp.94, 96, 97)
- WHITLEY, D., RANA, S., AND HECKENDORN, R. B. 1999. Exploiting separability in search: The island model genetic algorithm. *Journal of Computing and Information Technology* 7, 1, 33–47. (p.33)
- WILKE, C., WANG, J., OFRIA, C., LENSKI, R., AND ADAMI, C. 2001. Evolution of digital organisms at high mutation rate leads to survival of the flattest. *Nature* 412, 331–333. (p.47)

-
- WILLIS, R. 1870. *Principles of Mechanism*. London : Longmans, Green, New York, J.Willey. (p.2)
- WILSON, S. W. 2002. Classifiers that approximate functions. *Natural Computing: an international journal* 1, 2-3, 211–234. (p.64)
- YAEGER, L. 1994. Computational genetics, physiology, metabolism, neural systems, learning, vision, and behavior or polyworld: Life in a new context. In *Artificial Life III* (1994). (p.49)
- ZIMAN, J. 2003. *Technology innovation as an evolutionary process*. Cambridge University Press. (p.2)
- ZYKOV, V., MYTILINAIOS, E., ADAMS, B., AND LIPSON, H. 2005. Self-reproducing machines. *Nature* 435, 7038, p163–164. (p.12)