Apprentissage de comportement comme résultat de l'intéraction avec la dynamique de l'environnement

- Programmation Génétique
- Problème
- paramètres, méthodes ...
- Résultats
- Simulation

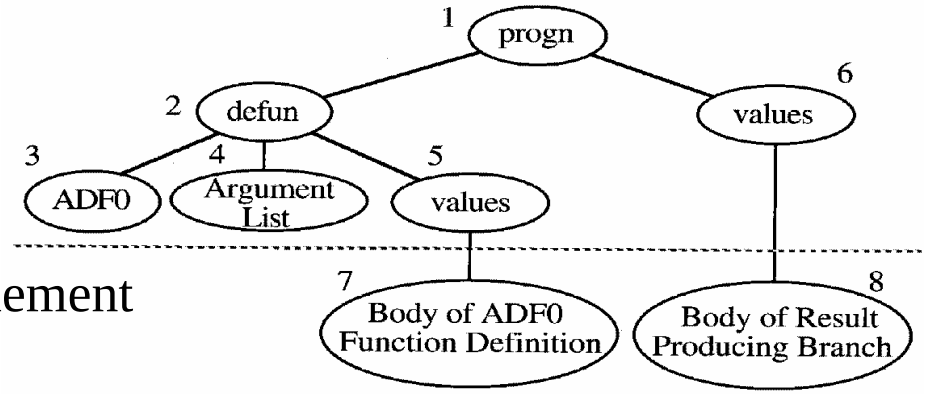
Problème

- « A case study in the behavior−oriented designof autonomous agents » Luc Steels'94
- « behavior learning and individual cooperation in autonomous agents as a result of interaction dynamics with the environment » S.Kamani, KOZA'95
- Etude de l'adaptation d'un système multi−agents dans un système dynamique par Programmation Génétique (GP).
- La coopération n'est pas explicitement programmée, mais doit émerger du comportement des agents à partager et agir de façon responsable dans un système dynamique.

Programmation Génétique

− dérivée des Algorithmes génétiques
− utilisant plutôt des programmes informatiques que des vecteurs fixes de chaînes ou structures.
− la population initiale composée de programmes aléatoires représentés sous formes d'arbres
− principe de mutation et de crossover, sélection par tounoi
− Result Production Branch (**RPB**): programme principal
− Automatically Defined Functions (**ADF**): fonction qui évolue dynamiquement pendant un fonctionnement d'un GP, appelé par le RPB qui est évolué en même temps.

Défini par:
− ensemble de terminaux
− ensemble de fonctions primitives
− mesure de Fitness
− paramètre de contrôle de fonctionnement
− méthode pour désigner le résultat
et critère d'arrêt de fonctionnement.



Valeurs standard en GP

Probabilité de crossover	90,00%
Probabilité de reproduction	10,00%
Probabilité de choix interne de points pour crossover	90,00%
Taille maximum du programme crée	17
Taille maximum pour les programmes aléatoires initiaux	6
Probabilité de mutation	0,00%
Probabilité de permutation	0,00%
Fréquence d'édition	0
Probabilité d'encapsulation	0,00%
Condition de décimation	Null
Pourcentage de décimation cible	0,00%
Méthode de génération initiale aléatoire	Rampe 1/2 1/2 (depth−grow
Méthode de sélection basique	Tournoi par groupe de 7
Méthode de sélection d'appariement	Tournoi par groupe de 7
Fitness ajustée	Non utilisée
Sur−sélection	Non utilisée
Stratégie élitiste	Non utilisée
Valeur aléatoire (si besoin) dans la création de la fitness	Fixée une fois pour toutes
Dans le crossover préservatif, assignation des types aux points non variants	Typage de branche

Avantages de la GP

- Décomposition de problèmes en sous−problèmes, puis réassemblage en solution du problème général
- utilisent les régularités, symétries, homogénéités, similarités, patterns et modularisation de l'environnement
- moins de calcul (car sous−problèmes)
- la solution trouvée est de taille réduite
- Toute fonction (ADF) peut accepter le retour d'autres fonctions ainsi que des terminaux (satisfaction de clôture (*closure*).

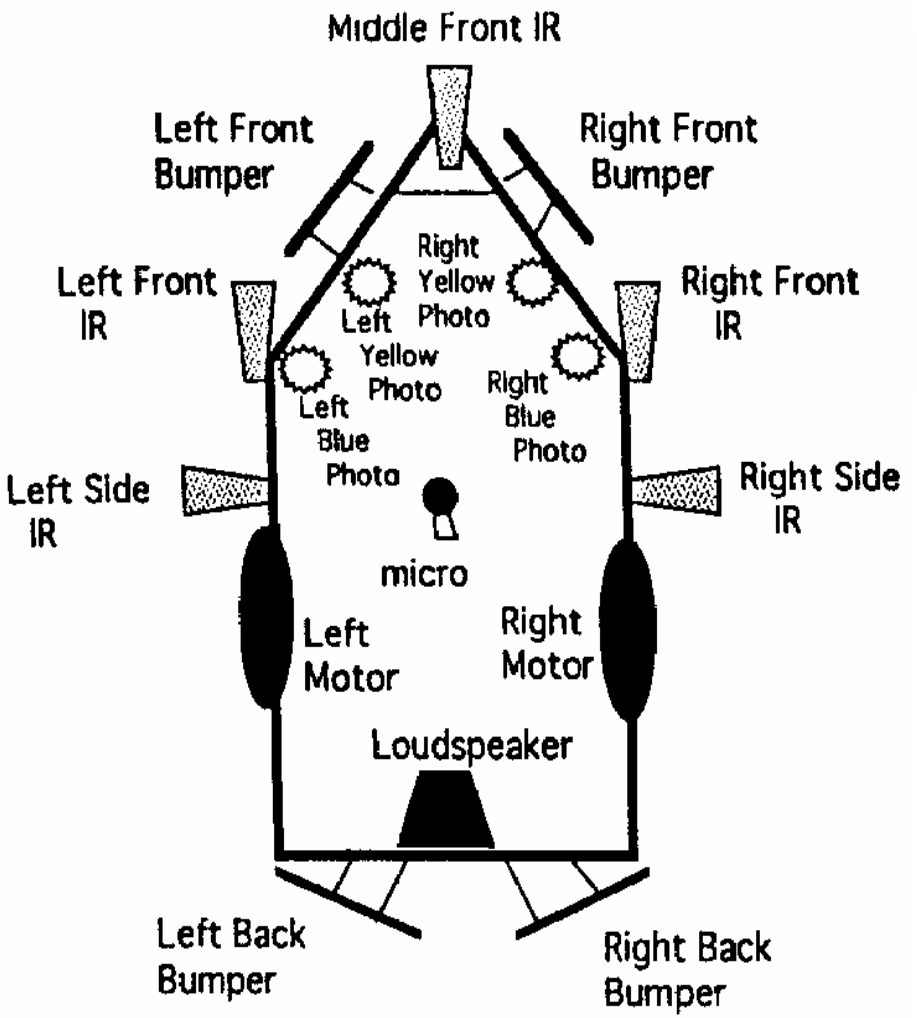
  La Fitness complètement définie (tous les cas de figures possibles)

  => Calculée d'après les conséquences du fonctionnement du programme.

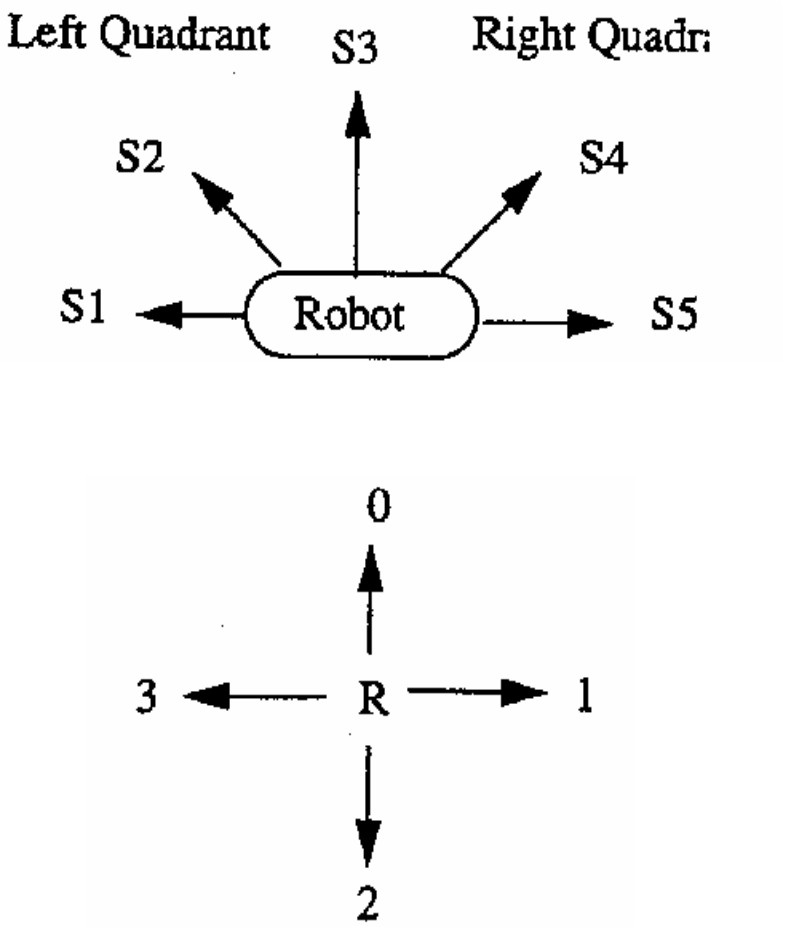
Robot à GP

- Deux ou plusieurs robots
- batteries rechargeables
- 6 senseurs, moteurs, actuateurs
- lampes et station de chargement (CS) prenant leur énergie de la même source
- le système n'a plus d'énergie si toutes les lampes sont complètement chargées
- les robots heurtant les lampes les déchargent
- les robots se rechargent sur la CS
- => encodage de schémas de motivation pour Steels
- => apprentissage de ces schémas par les GP et le calcul de Fitness

Simulation



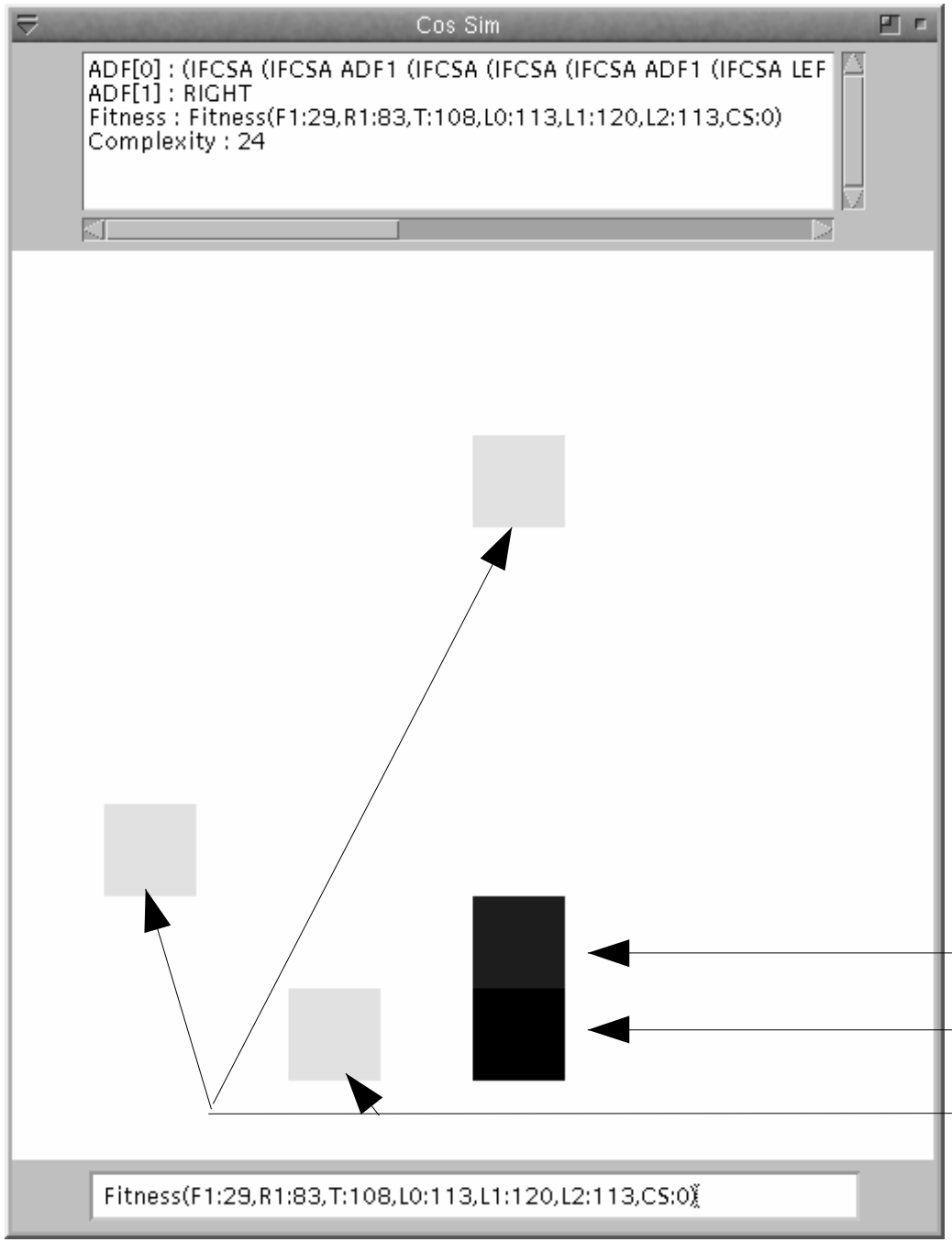
Robot utilisé par Steels



Equivalent pour la GP

Monde simulé

- Monde toroïdal de 10x10
- 3 lampes, 1 station de recharge
- les lampes initialement ont 12 d’énergie, s’accroissent de 2/timestep jusqu’à 30, sont déchargées de 7 si heurtées par un robot
- les robots ont 80 d’énergie, perdent 1 par mouvement et peuvent se recharger de 6/timestep à la CS
- connaissance interne de leur état et de celui du système par des fonctions



Simulateur

En Java 1.1basé sur la librairie gpsys−1.1 de Adil QURESH (University College London)

> java gpsys.cos.CosSim fichier RandomSeed ***population génération***

Charging Station (Blue photoaxis)

Robot

Lampes

Fitness

- Raw Fitness: le temps vécu par le système + nombre de robots en vie + nombre de lampes pas au maximum
- => forcer le robot à ne pas avoir un comportement qui laisse le système dépérir (perdre son énergie)
- le système est arrêté si le robot meurt ou toutes les lampes sont au maximum

GP

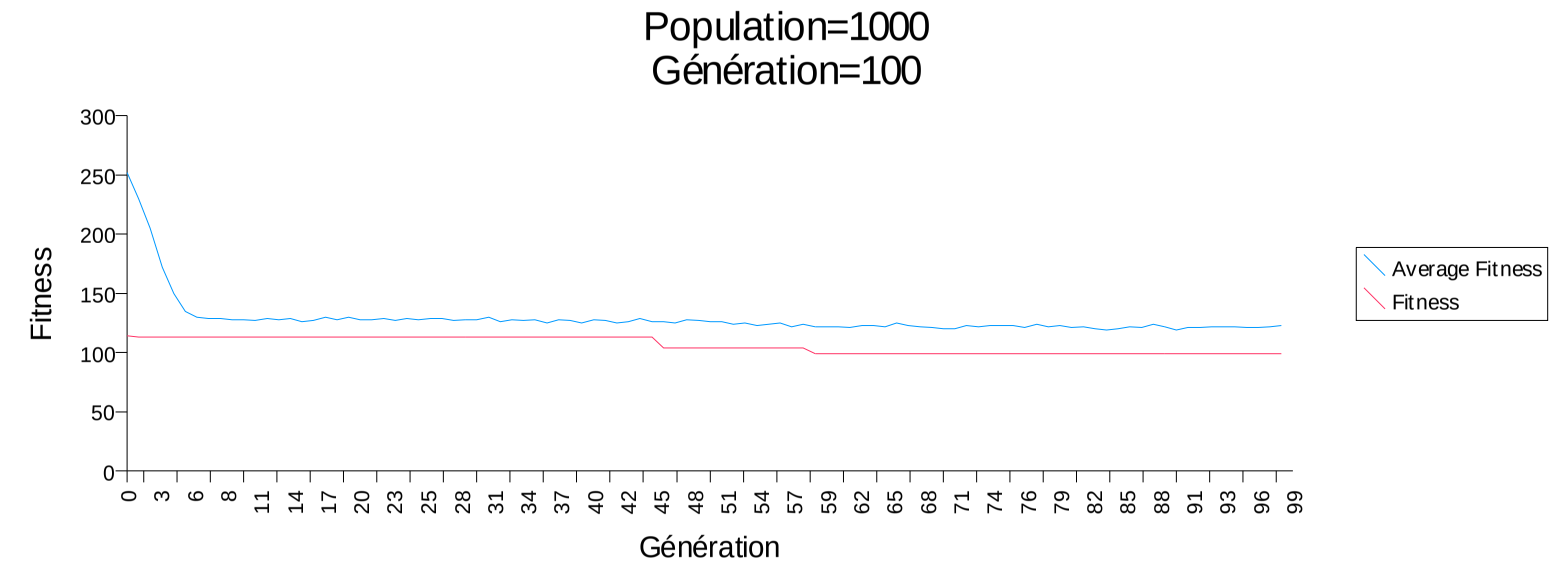
- Terminaux RPB: LEFT, RIGHT, FORWARD, BACK
- Fonctions RPB: IFSYSUNSAFE, IFCHARGED, IFAL, IFCSA, IFLGR, IFDIE, IFOBA
- Terminaux ADF: LEFT, RIGHT, FORWARD, BACK, HALT
- Fonctions ADF: IFACS, IFCHARGED, IFSYSUNSAFE
- Paramètres: M=10 000, G=100

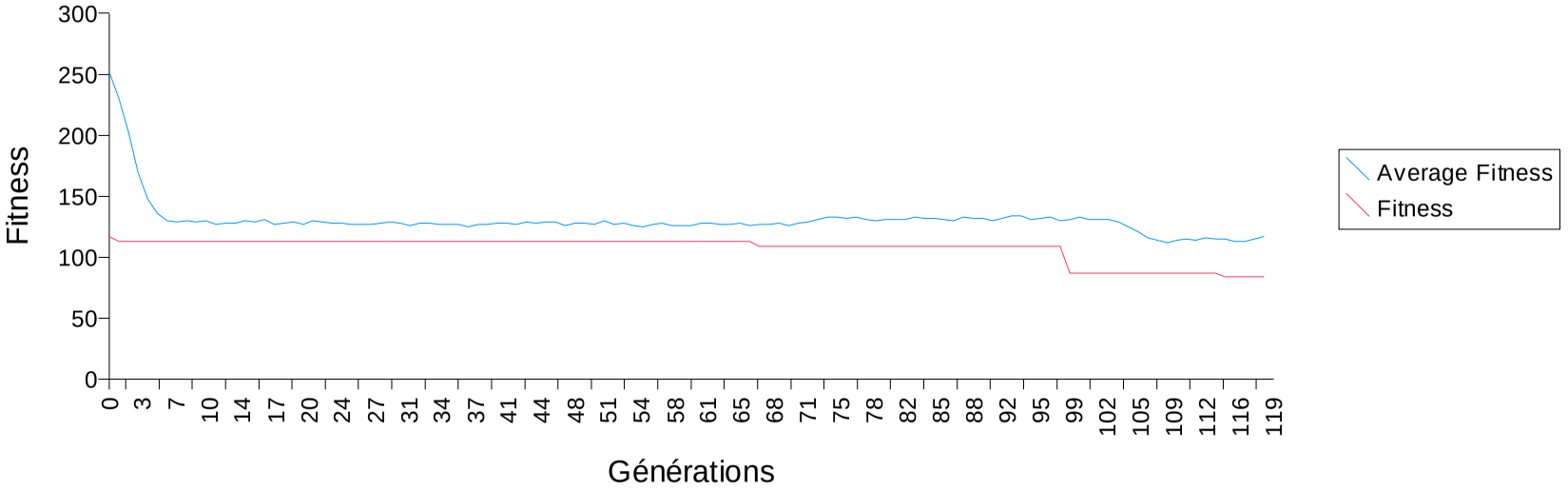
Simulation

Paramètres ajoutés pour la simulation

Taille Tournoi	7
Mutation	5,00%
Arbre	*taille 9 maximum* *5 maximum à la création* *3 maximum pour le croisement*

Simulations



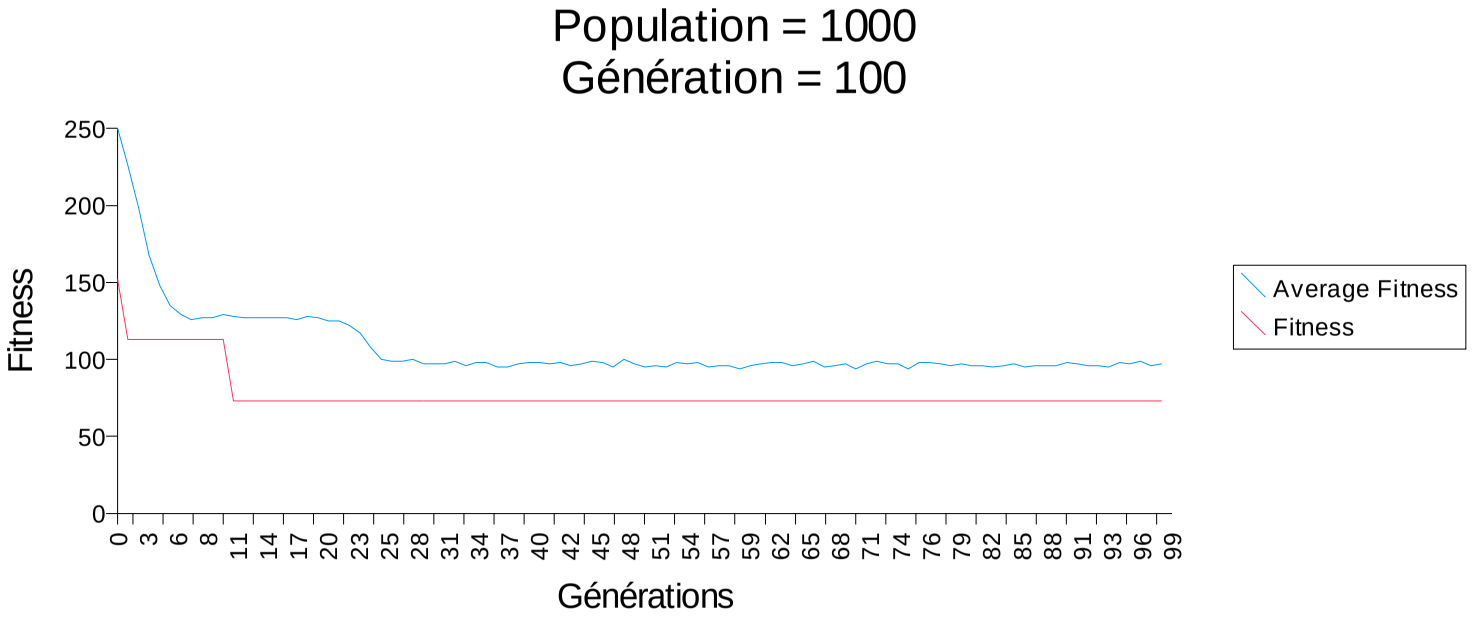


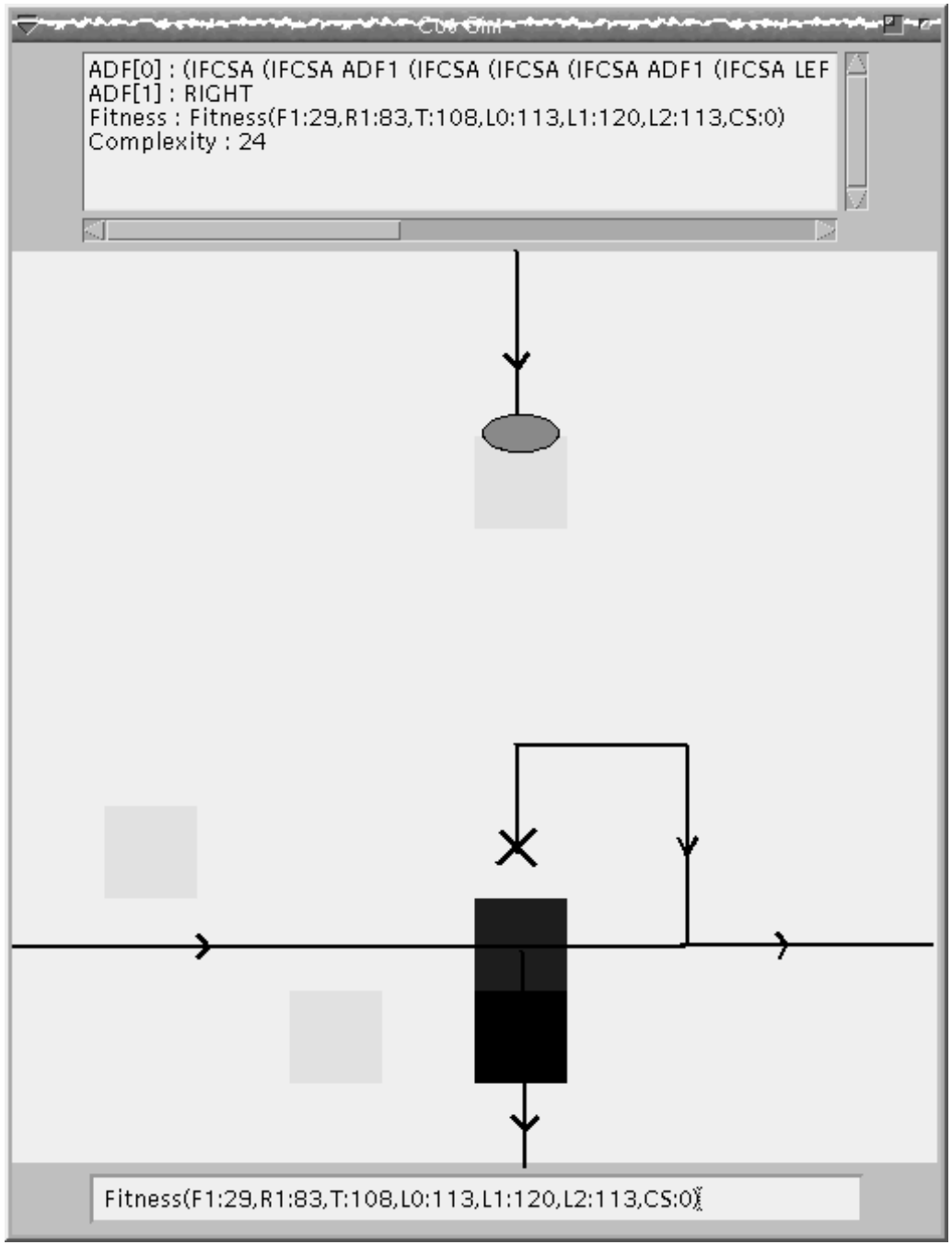
Meilleure Simulation

ADF[0] : (IFSYSUNSAFE (IFSYSUNSAFE (IFDIE (IFOBA (IFLGR ADF1 LEFT) RIGHT) FORWARD) (IFSYSUNSAFE BACK ADF1)) (IFSYSUNSAFE (IFSYSUNSAFE (IFSYSUNSAFE (IFDIE (IFOBA (IFLGR ADF1 LEFT) RIGHT) FORWARD) (IFLGR (IFSYSUNSAFE ADF1 BACK) (IFSYSUNSAFE BACK ADF1))) (IFSYSUNSAFE (IFAL ADF1 LEFT) (IFCSA FORWARD ADF1))) (IFSYSUNSAFE RIGHT (IFCSA FORWARD ADF1))))

ADF[1] : (IFACS (IFACS (IFCHARGED BACK BACK) RIGHT) (IFSYSUNSAFE HALT RIGHT))

Fitness : Fitness(**F1:73**,R1:25,T:275,L0:120,L1:97,L2:120,CS:0)

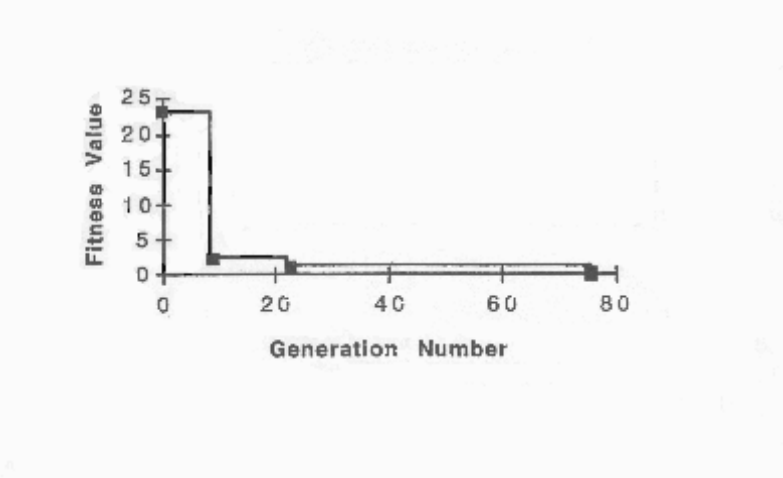




Meilleure Simulation

Résultat

- Dans le cas de deux robots, l'algorithme semble converger rapidement si l'on en croit le graphe donné dans l'article (Figure 3), avec comme solution optimale:
- Robot1: (IFAL LEFT ADF0)
- Robot2: (IFCSA ADF0 ADF0)
- ADF0: (IFSYSUNSAFE FORWARD RIGHT)



Conclusion

- Nous avons vu l'émergence de comportements en relation avec l'émergence du système, dans le cas d'un seul robot.
- Dans le cas de deux, cela semble plus intéressant, avec une rapide convergence.
- Un calcul de Fitness approprié et des constantes permettent d'obtenir le comportement désiré plus rapidement (6 générations au lieu de 10000 habituellement).
- Il serait intéressant de vérifier ce comportement de coévolution quoique plus difficile techniquement à réaliser
- On pourrait rajouter des signaux comme dans [Steels], et un comportement de parquage à la CS