ABOUT THE FITNESS OF SIMULATIONS WHOSE FUZZY RULES ARE LEARNED BY GENETIC ALGORITHMS

L.Gacôgne
LAFORIA CNRS - Université Paris VI 4 place Jussieu 75252 Paris 5°
tel : 44 27 70 02 fax : 44 27 70 00 mail : gacogne@laforia.ibp.fr
Institut d'Informatique d'Entreprise (CNAM) 18 allée J.Rostand 91025 Evry
tel : 60 77 97 40 fax : 60 77 96 99 mail : gacogne@iie.cnam.fr

Abstract

First, we give a new way to implement genetic algorithms as a random parallel research ending with cross-over. Intenting to enumerate the problems raised when setting a fuzzy controller, we examine then several experiments with numeric or symbolic formalizations about fuzzy control from two or more variables towards one or two outputs. Our experiments as various problems of guiding an autonomous vehicle show always that genetic algorithms starting with a random population, give a good answer to each heuristic we imagine, the hardness being to precise those heuristics.

Résumé

Nous offrons d'abord quelques modifications sur les algorithmes génétiques vus comme une recherche aléatoire en parallèle qui s'achève avec croisement des optimums obtenus. En tentant d'énumérer les problèmes rencontrés lors de la mise au point d'un contrôleur flou, nous examinons alors plusieurs expériences à propos d'un contrôle flou numérique ou symbolique de deux à quatre entrées vers une ou deux sorties. Nos expériences telles que quelques problèmes de guidage de véhicules automomes, montrent toujours que les algorithmes génétiques qui partent d'une populatiopn aléatoire, donnent une bonne réponse pour chaque heuristique que nous avons imaginé. Cependant que la difficulté réside dans la recherche de ces heuristiques.

Zusammenfassung

Zuerst bieten wir einige Veränderungen an den genetischen Algorythmen an, die als eine aleatorische und parallele Forschung betrachtet werden, eine Forschung die mit einer Kreuzung des Optimalen endet. Wir versuchen die Probleme aufzuwerfen und aufzuzählen, denen wir während des Nachstellens eines ungerauen Kontrollgrätes begegnet sind und wir prüefen dann einige Experimente einer ungenauen numerischen oder symbolischen Kontrolle von zwei bis vier Eingängen nach einem oder zwei Ausgängen. Unsere Experimente, wie die Führung eines Selbständigen Fahrzeuges beweisen immer daß die genetischen Algorythmen, die aus einer aleatorischen Bevölherung stammen, eine gute Antwort auf jede Heuristik bringen. Die Schwierigheit bleibt jedoch bei der Erfindung dieser Heuristik.

I A GENERAL FUZZY CONTROLLER FROM $[-1, 1]^n$ TO $[-1, 1]^p$

When (E1, n1) and (E2, n2) are two normed spaces and S a map from R^{+2} to R^+, defining N on the product E1*E2 by N(x, y) = S(n1(x), n2(y)), we have some properties between a supposed norm N on E1*E2 and a supposed t-conorm S [Gacôgne 94], for instance :
a) If S(a, b) = max(a, b) (Zadeh co-norm) then we have N(x, y) = sup (n1(x), n2(y))

b) If S(a, b) = min (1, a + b) (Lukasiewiecz) then the Hamming norm N(x, y) = n1(x) + n2(y)

c) S(a, b) = $(a2 + b2)^{1/2}$ (Yager), N is the euclidean norm.

If we note B(a, r) the fuzzy neighborhood of a $\in [-1, 1]^n$ with an r $\in R^+$as radius, defined by the fuzzy set whose membership function is $\mu_{B(a, r)}$ (x) = max (0, 1 - N(a - x) / r)
If T is a t-norm and N the associated norm (one of the three lasts), we have :

$\mu_{B(a, r)}$ (x) = T ($\mu_{B(a1, r)}$ (x1), ... , $\mu_{B(an, r)}$ (xn)) where a = (a1, ... , an) is the modal value and x = (x1, ... , xn)

The rules of a fuzzy control system from R^n to R^p, will be on the following model :
If $x = (x_1, x_2, \dots, x_n) \in B(a_i, r_i)$ then f(x) is equal to $c_i = (c_1, \dots, c_p)$
We define then :

Definition : f is the Sugeno function relied to the rules set $\{(a_1, r_1, c_1), \dots, (a_k, r_k, c_k)\}$ where $a_i \in [-1, 1]^n$, r $\in R^+$ and $c_i \in [-1, 1]^p$, if :
$\forall x \in [-1, 1]^n$ $f(x) = (\sum \alpha_i * c_i) / \sum \alpha_i$ with $\alpha_i = \mu_{B(ai, ri)}$ (x) for i from 1 to k.

Remark, we can also define a Mamdani function f relied to the rules set $\{(a_1, r_1, c_1, r_1'), \dots, (a_k, r_k, c_k, r_k')\}$ if for all x $\in [-1, 1]^n$ f(x) is the abscissa of the center of gravity of the union of the α_i-cuts of the fuzzy sets (c_i, r_i') with $\alpha_i = \mu_{B(ai, ri)}$ (x) for i from 1 to k.

II QUESTIONS RAISED BY THE SETTING OF A FUZZY CONTROLLER

Ordinary, when implementing a fuzzy controller, the questions raised are :

a) The chooses of the inputs and output(s), thus :
For the problem of control, we generally consider the error and its variation e, de --> aim
For the following of a line x, dx --> output dα or ωr, ωl angular speeds of the wheels.
Following of a passageway, it may be d1, d2, Δd1, Δd2 --> dα or d1, d2, d3 --> dα
For the pursuit of S by C, we can take the angles with the line CS : αC, αS --> dα
Obstacles sunting : we can take three or more distances (see further) or a simulation of a radar, taking the polar position (d, θ) of the nearest obstacle.

b) Domain of validity about these inputs-outputs.
The problem is generally simplified by the determination of parameters like A to reduce a measure from [-A, A] to [-1, 1]. Thus we have further a maximal steering AM and a maximal acceleration ACC.

c) Which predicates ?
Our investigation is simplified taking a binary family of triangular predicates on [-1, 1].
Applications of the last years are going towards five or more rarely three or seven predicates.
But many applications have triangular or trapezoidal predicates with variable bounds.
We have also gaussian curves.

d) Which rules ?

At the beginning, the "philosophy" of fuzzy control is the translating of typical situations by intuitive rules, and these rules must overlap themselves to realize interpolation and then to spare a modelization. But experiments prove that it is difficult to get optimization and even this question is difficult to formalize.

e) Which operators ? min for conjunction ?, Mamdani's min (that is to say truncating of the conclusions) for implication ?, max for aggregation ?, center of gravity for defuzzyfication ?

f) How to give an evaluation of the fitness ?
Each problem may be evaluated by some heuristics like the average of the distance with target or the number of iterations to be ε-near this aim. In the case of a supervised learning, the fitness function will be the average of error made by the Sugeno function built by a set of rules, on the set of examples.
Fuzzy control can be seen as an answer to many problems of multicriterion analysis where an example is an association (a1, a2, ..., an, b1, b2, ... bp) where the ranges of the parameters are numeric or symbolic.

Remark, many others problems less theoretical, like reliability of the sensors or distance evaluation on a screen when simulating raised also.

III LEARNING METHODS

The genetic algorithm to synthetize a fuzzy controller

We introduce a genetic strategy which has some difference with the traditional one [Goldberg 89] [Holland 93]. When we compare genetic algorithms with the other methods of the operational research, we can say that the most important distinction is the fact that we operate with an initial point in the first case, and with a set of initial points with the second. If we see a population as an important set and the process of research as the action of various methods to give neighborhood points, we do a parallel research and we shall bring the following modifications :
a) A chromosome has not a fixed length, moreover it may be structured as (x, y, z ... ((NB PB) ZE PS NB) ((ZE PB PB) NS NS ANY) ....) with numeric and symbolic values mixed for instance, to always keep the concrete meaning of individuals.
b) Those methods are the genetic transitions (mutation, transpositions, gaussian noise if numeric ...) which are randomly generated at the beginning in an other population.
If the aim of the algorithm is to get a minimum of a cost function computed on the population whose elements are potentials solutions to a given problem, we also give a cost to the genetic transitions and make a sort of them at each generation. We note that roughly crossover is going best after action of transitions like suppression.
c) When a transition is applied to an element, we sort immediately the father and the son (the four parents and children in the case of cross-over). By this way, when a falling in a well of the cost function is primed, we keep only the last descendant.
d) In the same idea, we don't use cross-over in the first phase of the evolution and then we can hope to get all minimums of a function.

Remark

If a function from R to R has n minimums in an interval and if we start from k points (with a uniform random distribution), the probability to reach a number X of those n minimums is given by :

$$pr(X = p) = \sum_{i=0}^{p} (-1)^{p+i} C_p^i \left(\frac{i}{n}\right)^k$$

For example if n = p = 3 and k = 9, we have a probability 0.922 only with p = 3 therefore we often work with populations with cardinality 10.

IV SIMULATIONS AND RESULTS

IV - 1 Reaching a target with ε

We got good results, defining the cost as the sum of the steps to reach the target with an ε-proximity and leave inside.

One of the simplest table we got is :

PB			PS		
PS		NS			
ZE	PB				NB
NS				PS	
NB			NS		
dx / x	NB	NS	ZE	PS	PB

IV - 2 Following crenels

Evaluation is the relative (%) average error with the aim minus the number of empty boxes. When completing the array, we get the best solution in about 25 generations.

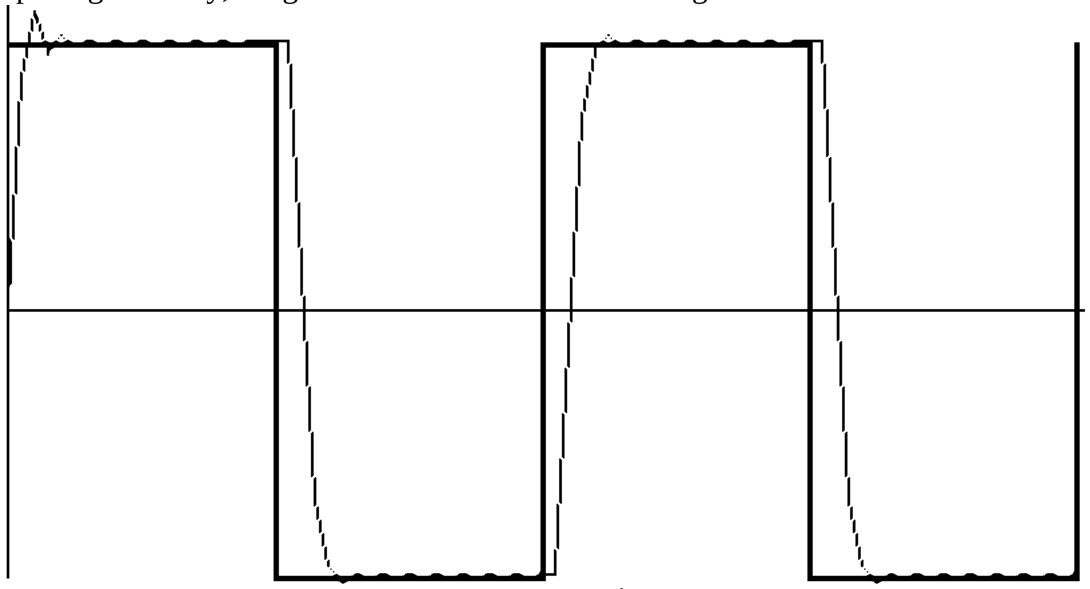

Figure 1

The best solution (7% average error) is gotten in about 25 generations with the chromosome (pb nul nul ps pb nul ze nul nul nb ns ze) which is in connection to the table :

PB	15 PB				25 NB
PS		14 PS	18 ZE	21 NS	
ZE	6 PB		13 ZE		20 NB
NS		5 PS	8 ZE	12 NS	
NB	1 PB				11 NB
dx / x	NB	NS	ZE	PS	PB

Whithout NB in 11 (and PB in 15) the reaching from an aim to the other is worse describing a step (value 10%).

IV - 3 Poursuit

We had some good results for a pursuer C when the two inputs are the angles tc and ts respectively made by the directions of C and the pursued S with CS.

The pursued S is runing m times less speed than C but is able to steer m times better then C. If the angles are brought to [-4π/5, 4π/5], we define the cost of a strategy by the number of steps used to reach S summed in 5 random startings for S.

	PS	ZE		
	PB		NB	
		ZE	NS	

		PS		
PB		PB		
		NB		NB
		NS		

With tc in line and ts in column the rules of C (left table) are gotten in 20 generations and the rules of S (right table) are gotten when the performance of a strategy is defined by the total sum of steps run by S when C has the strategy of the left table with 6 initial orientations. [Gacôgne 93]

IV - 4 Following a road with Sugeno numeric rules

First we introduced in [Gacôgne 94] the "road-holding" as a number in percentage whose range is [0, 100] :

$$tr = \left| \frac{|da|}{am} + \frac{dp}{2acc} - \frac{1}{2} \right| \in [0, 1]$$

More the average of tr is near 0, more we estimate the trajectory good, when close to 50, we estimates bad, and more tr is near 100, more the trajectory is the opposite of a good driving.

We simulate three problems in the next parts, following a line or a route (figure a), reaching a line of markers (figure b) and passing a slalom (figure c).

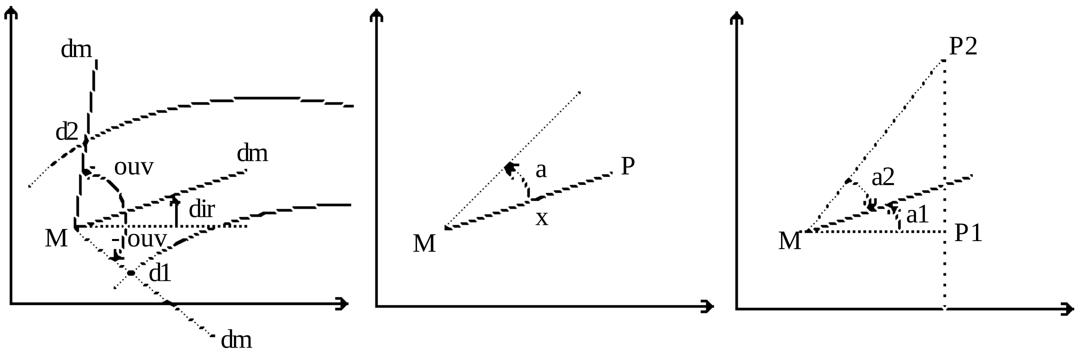

Figure 2

a) d1 $\in$ [-dm, dm], dd1 $\in$ [-vm, vm] or d1rigth, dfront, d2left $\in$ [0, dm] $\rightarrow$ ∂dir, ∂step to follow a road.
b) x $\in$ [0, dm], a $\in$ [$-\pi$, π] $\rightarrow$ ∂dir, ∂step to reach a sequence of markers.
c) a1, a2 $\in$ [$-\pi$, π] $\rightarrow$ ∂dir, ∂step to pass through slalom gates.

The idea of using genetic algorithms to determine symbolic tables is very natural because the easiness of codage. Yet, when we would change predicates or other specific parameters to each problem, it is possible [Glorennec] however costly according to the tall of research space, to use transitions like gaussian noises. Moreover our chromosomes are a collection of rules with different lengths. In the first experiment, a rule is on the type :

(ouv dm vm am acc pm (a b r al ac) (a' b' r' al' ac') .....)

where a mobile tries to follow a line taking as inputs the algebraic distance d1 to the line (with an angle "ouv" to its direction) and the variation of it.

Each period, the vehicle determines by Sugeno' algorithm two outputs in the same time : the variation ∂dir of dir, and the variation ∂step of the speed.

$x \in [-dm, dm]$, $dx \in [-vm, vm] \rightarrow x / dm, dx / vm \in [-1, 1] \rightarrow fuzzy_C \rightarrow u \in [-1, 1] \rightarrow am*u \in [-am, am]$

There is a maximum dm to estimate distances, a maximum vm for the variations, and also a maximum pm for each step (a minimum ds is also fixed) acc is the maximum of acceleration and we try to generate all these parameters in addition to the rules.

For this first experiment we impose to have the symmetric of each rule :

sym (a b r al ac) = (-a -b r -al ac) and the cost of an individual is the rest (in %) of the run + road holding + number of rules.

After 300 generations we got the performances 44% of the run and 26% road-holding (value 69) with the rules :

(d_{right} is ZE) and (d_{front} is PS) and (d_{left} is PS) -> (∂dir is PS) and (∂step is NB)
and (d_{right} is PS) and (d_{front} is PS) and (d_{left} is ZE) -> (∂dir is NS) and (∂step is NB)
{ ze ps pb force to turn left and a strongly braking}
(d_{right} is ZE) and (d_{front} is PS) and (d_{left} is PB) -> (∂dir is PB) and (∂step is NS)
and (d_{right} is PB) and (d_{front} is PS) and (d_{left} is ZE) -> (∂dir is NB) and (∂step is NS)
(d_{right} is PS) and (d_{front} is PS) and (d_{left} is PB) -> (∂dir is PB) and (∂step is NS)
and (d_{right} is PB) and (d_{front} is PS) and (d_{left} is PS) -> (∂dir is NB) and (∂step is NS)
(d_{right} is PS) and (d_{front} is PB) and (d_{left} is PB) -> (∂dir is NS) and (∂step is PB)
and (d_{right} is PB) and (d_{front} is PB) and (d_{left} is PS) -> (∂dir is PS) and (∂step is PB)
(d_{right} is PS) and (d_{front} is PB) and (d_{left} is ANY) -> (∂dir is NS) and (∂step is PB)
and (d_{right} is ANY) and (d_{front} is PB) and (d_{left} is PS) -> (∂dir is PS) and (∂step is PB)
(d_{right} is PS) and (d_{front} is PB) and (d_{left} is PS) -> (∂dir is PB) and (∂step is PB)
or (∂dir is NB) and (∂step is PB)
{ If we try the conclusion ∂dir is ZE (only one rule) we gets a worse result }
(d_{right} is ANY) and (d_{front} is PB) and (d_{left} is ANY) -> (∂dir is PS) and (∂step is PS)
or (∂dir is NS) and (∂step is PS)
{strong acceleration if front distance is very big}

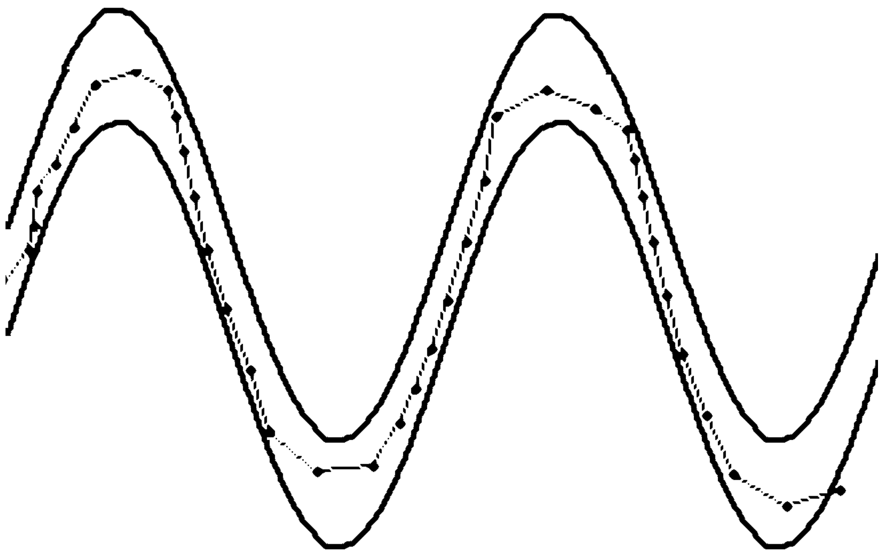

figure 2

IV - 5 Following a markers line

Our next experiment is composed by a robot which must overtake a sequence of points (crosses on the figure) step by step without memory of previous run. Considering the two inputs : distance to the point to join, and angle between the cap of the robot and the cap to join, the rules to discover must have an action on two outputs : variation of the direction and growing of the speed. That is to say :

$x \in$ [-dm, dm], $a \in$ [-vm, vm] $\rightarrow$ $fuzzy_C$ $\rightarrow$ (∂dir, ∂step) $\in$ [-am, am]*[-acc, acc]

We define sym (∂dir ∂step x a) = (-∂dir ∂step x -a)
Where dm = 50 acc = 7 vm = 90°, the steering limit am = 50° pm = 35 the best road holding is 19%
The evaluation of a trajectory is made by the average error plus the number of rules. We now use a symbolic but Sugeno algorithm which means that we look only to the modal value of the predicates when they are in conclusion.
After only 25 generations we get a short solution :

pb ns --> ps pb,	{if far from the point, we must turn to join the cap and accelerate}
ps pb --> nb nb	{if we are not too far, then we have to brake}

and their symmetric rules (value 27) whose meaning is when the point is far and the cap medium, it must turn and accelerate medium, and when distance is medium and cap is very bad, then it must brake and turn strongly .

After 150 generations we get the 7 rules

pb pb --> nb ze,
pb ps --> nb pb,
ze nb --> pb nb, {if we are very close to the point, we can turn strongly}
any ze --> ze pb

and the symmetric rules : pb nb --> pb ze, pb ns --> pb pb, ze pb --> nb nb. (value 21)

Figure 3 is gotten by the 8 rules (value 25) :

(x is ANY) and (a is ZE) -> (∂dir is ZE) and (∂step is PB)
{strong acceleration when cap is very good}
(x is ZE) and (a is ZE) -> (∂dir is ZE) and (∂step is NB)
{strongly braking when very near}
(x is PB) and (a is PB) -> (∂dir is NS) and (∂step is PB)
(x is PB) and (a is NB) -> (∂dir is PS) and (∂step is PB)
{turn fairly when far of the point}
(x is PB) and (a is PS) -> (∂dir is NB) and (∂step is ZE)
(x is PB) and (a is NS) -> (∂dir is PB) and (∂step is PB)
{turn strongly with the same speed when far of the point, those 4 rules balance themselves}
(x is ZE) and (a is PB) -> (∂dir is NB) and (∂step is NB)
(x is ZE) and (a is NB) -> (∂dir is PB) and (∂step is NB)
{turn strongly when near, but not in the good direction}

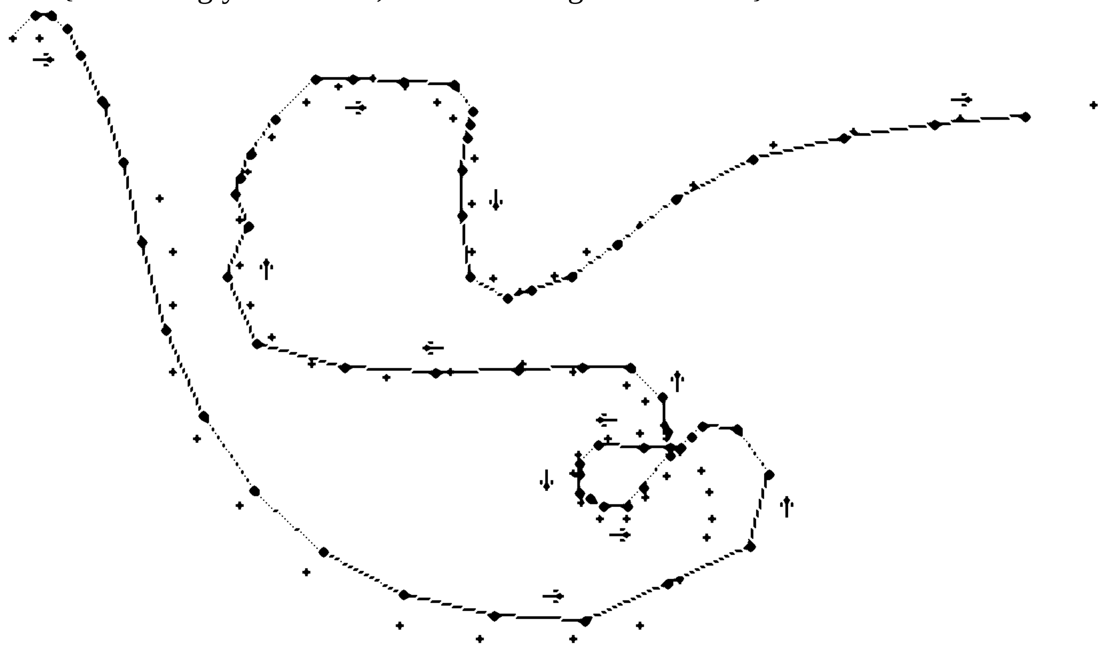

figure 3

IV - 6 Slalom

We now want to force a robot to cross slalom gates by learning rules about the two angles it makes between its direction and the two sides of the gate which is front of it.

We define sym (∂dir ∂step a1 a2) = (-∂dir ∂step -a2 -a1).

The cost function is here :

value (C) = tax for missed gates (sum of the distances to the nearer side of the gate)
+ tax for gates no reached (5 times their number)
+ the road-holding
+ number of rules

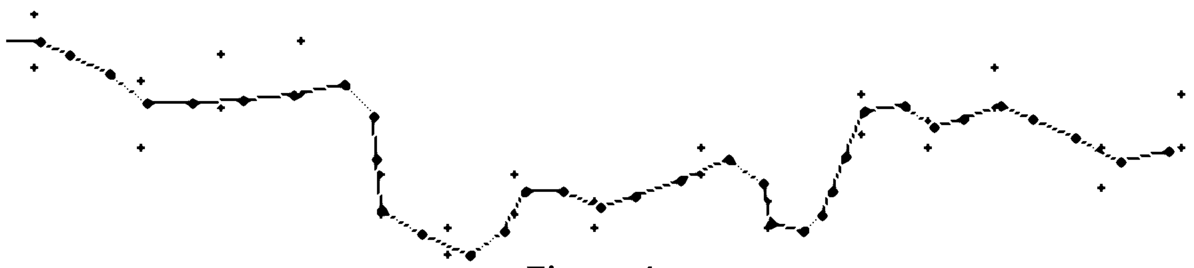

Figure 4

For vm=57, acc=4, am=60, pm=20 we get in 35 steps 99% of the run, 0 missed gates, a road holding 10% and then the best cost 17 after about 200 generations with the rules corpus :

(a1 is PB) and (a2 is NB) -> (∂dir is ZE) and (∂step is PB) {acceleration if good direction}
(a1 is PB) and (a2 is NS) -> (∂dir is NS) and (∂step is NS)
(a1 is PS) and (a2 is NB) -> (∂dir is PS) and (∂step is NS)
{2 rules for a small rectification, braking is balanced with the first rule and the nexts}
(a1 is ANY) and (a2 is ZE) -> (∂dir is ZE) and (∂step is PB)
(a1 is ZE) and (a2 is ANY) -> (∂dir is ZE) and (∂step is PB)
{acceleration if the mobile is guided towards a side of the gate}
(a1 is ANY) and (a2 is PS) -> (∂dir is NS) and (∂step is PS or NS)
(a1 is NS) and (a2 is ANY) -> (∂dir is PS) and (∂step is PS or NS)
{4 rules to turn moderate if the one of the angles is not too bad }
(a1 is ANY) and (a2 is PB) -> (∂dir is NB) and (∂step is NS or NB)
(a1 is NB) and (a2 is ANY) -> (∂dir is PB) and (∂step is NS or NB)
{4 rules if very bad orientation, brake strongly and turn in consequence}

Conclusions

We show that it is more or less possible to find "good solutions" to various problems of multi-variable fuzzy control. But, except the first one, those experiments give a notation to a global comportment by an average of different performs. We can remark that most of the improvements are caused by arrival of new rules during evolution of generations or shrinking in the set of rules. New rules whose intuitive interpretation is always possible. About those experiments, we observe, of course, that more we bring our ambition, more long is getting on good solutions.

The search of "good" rules to fuzzy control is an example of difficult problem where the applications of genetic algorithms realise a kind of "boot-strap".

References

Franz Non-linearities in generic adaptative search Doctoral dissertation 1972 in Dissertation Abstracts International 33 (11) 5240B-5241B University of Michigan

Gacôgne L. Compte rendu sur la recherche de table de contrôleur flou par algorithme génétique, Rapport 93/26 Laforia Université Paris VI 1993

Gacôgne L. Apprentissage génétique global d'un contrôleur flou à deux variables basé sur la simulation d'un véhicule autonome. Congrès IPMU 1994

Glorennec P.Y. Applications des algorithmes génétiques pour l'optimisation des fonctions d'appartenance d'un réseau neuro-flou, Deuxièmes journées sur les applications des ensembles flous, pp 219-226 Nîmes 1992

Goldberg David Edward Genetic algorithms in search optimization and machine learning, Addison Wesley 1989

Holland Une échappatoire à la précarité : les possibilités de l'application des algorithmes généraux d'apprentissage aux systèmes parallèles à base de règles, Apprentissage symbolique pp 523-554 (Cépaduès 1993)

Kamada H. Yoshida M. A visual control system using image processing and fuzzy system. IEEE round table discussion on vision-based vehicule guidance Tokyo 1990

Karr Charles L. Design of an adaptative fuzzy logic controller using a genetic algorithm Proceedings of the fourth International Conference on genetic algorithm pp 450-457 San Diego 1991

Mamdani E. H. Assilian S. Learning control algorithm in real dynamic systems. Proceedings of the 4th international IFAC IFIP Zürich 1974

Michalewicz Z. Genetic algorithms + data structures = evolution programs, Springer Verlag 1992

Siler W. Ying H. Fuzzy control theory : the linear case FSS n°33 pp 275 1989

Thrift Philip Fuzzy logic synthesis with genetic algorithm Proceedings of the fourth International Conference on genetic algorithm pp 509-513 San Diego 1991

Yamakawa Stabilization of an inverted pendulum by a high speed fuzzy contoller hardware system FSS n° 32 p 161-180 1989