Spatial Coordination through Social Potential Fields and Genetic Algorithms

Fabien Flacher[*,**] **Olivier Sigaud**[**]
[*]Dassault Aviation, DGT/DPR/ESA
78, Quai Marcel Dassault, 92552 St-Cloud Cedex
[**]AnimatLab-LIP6, 8, rue du capitaine Scott, 75015 PARIS
fabien.flacher@lip6.fr olivier.sigaud@lip6.fr

Abstract

In this paper, we address the problem of solving coordination problems in a continuous space with as few design effort as possible. Our approach relies on Potential Fields Methods combined with Genetic Algorithms. We compare it with other frameworks relying on Learning Classifier Systems and on Potential Fields Methods alone. Quantitatively, we show that we obtain better results than with Learning Classifier Systems. Qualitatively, we show that our framework requires less design effort than any other. The counterpart is that the controllers we obtain are harder to understand. We analyze a particularly efficient controller and conclude to the necessity of designing more formal tools to provide further insight on more complex controllers.

1. Introduction

There are two main motivations in the animat approach. This first one is to better understand nature by designing computational models used to validate hypotheses of natural mechanisms. The second one is to draw inspiration from nature in order to design efficient adaptive algorithms that can be used as software engineering tools. Using such tools can in turn contribute in significantly reducing the design effort necessary to solve many engineering problems. This paper is mainly concerned with the second motivation.

Outside of the animat research community, to solve an engineering problem, the prevailing methodology consists in a *functional decomposition* of the problem into smaller and easier subproblems. This top-down methodology has proved its efficiency as far as the problem is simple enough: it just stops when it reaches a level at which solutions to problems are obvious.

But, in the context of this paper, we will focus on specific problems which do not belong to that category: the design of controllers to solve collective problems requiring an efficient spatial coordination between several agents. These agents have to be:

- situated, *i.e.* they have only a local perception of the environment and are able to act on that environment;
- endowed with autonomous navigation capabilities, *i.e.* they are able, without any external assistance, of complex trajectories in a cluttered environment;
- adaptive, *i.e.* they are able to modify their nominal behavior to manage unpredictable situations as well as possible;
- coordinated, *i.e.* the global task must be achieved through the interdependent behavior of several agents.

Faced with these complex problems, the functional decomposition methodology performs poorly, because functional decomposition is not adapted to deal with interdependencies: the elementary interactions between agents cannot be isolated efficiently from one another. As a result, human designers spend a lot of time tuning low level parameters so as to adapt as much as possible their decomposition to the problem they face.

A minor improvement to this situation consists in adding to the top-down approach the use of adaptive algorithms in order to optimize the discretization boundaries that result from the functional decomposition. Optimizing the parameters of a system with adaptive algorithms makes it possible to spare a costly human intervention. Genetic Algorithms (GA) or Reinforcement Learning Algorithms can be used in that way, but this use does not imply reconsidering the functional decomposition approach itself.

In this paper, we show that a more radical improvement can be achieved on these complex coordination problems by giving up the functional decomposition approach. As an alternative, we present another methodology consisting in tuning the elementary interactions between the agents at the micro level so that the required global behavior emerges at the macro level. Relying on that methodology based on emergence gives us the ability to solve complex coordination problems with as few design effort as possible.

Our concrete aim is to exploit the robustness properties of adaptive behaviors in collective tasks where spatial coordination is necessary. Reaching this scientific goal endows to designers the ability to treat diverse applications, like the simulation of gregarious creatures in a video game, the test in simulation of new tactics of aerial raid or, as we will exemplify here, the control of a group of agents by another group.

In order to design the local interactions between agents, we use Potential Fields Methods (PFMs) (section 2.). But, whereas in most frameworks the potential fields have to be designed by hand, we use a GA to automate the tuning of the potential fields so that the agents optimize a global criterion (section 3.).

In order to evaluate the interest of our framework through a case study (section 4.), we first compare quantitatively its performance with controllers implemented with Learning Classifier Systems (section 5.). Then, we analyze one controller obtained with our framework (section 6.) and provide a robustness study of that controller (section 6.1). In the discussion section, we compare it with a similar controller obtained by hand by another research team and we stress the advantages of our methodology with respect to functional decomposition or the plain use of PFMs (section 7.). Finally, we highlight that the key problem with our approach lies in understanding rather than in designing the controller (section 8.).

2. Potential Fields Methods

At the origin of PFMs, the neurophysiological approach of Arbib demonstrated that some behaviors in frogs may be interpreted as a combination of attractions and repulsions induced by the environment (Arbib, 1981, Arbib and House, 1985). Such research was furthered by Partridge's work (Partridge, 1982) on fish schools and, above all, gave rise to Arkin's *schema theory* (Arkin, 1989) in robotics. In this framework, the behavior of an agent results from a decomposition into independent *schemas* expressed as attractive or repulsive potential fields which are combined to act on the agent as an electrical field acts on an electron (see figure 1).

Khatib (Khatib, 1985) was the first to apply PFMs to path planning among obstacles for simulated and actual robots. His approach was then generalized by Krogh (Krogh, 1984). Later on, Brooks (Brooks, 1986) and Arkin (Arkin, 1989) started to use these methods to control different robots with different kinds of sensors. Several methods were then proposed to solve the local minima problems and oscillations problems identified by Korenz and Borenstein (Korenz and Borenstein, 1991a), such as the use of noise (Arkin, 1989), evolutionary techniques (Pearce et al., 1992), or special purpose dead-lock detectors to avoid local minima (Korenz and Borenstein, 1991b, Piaggio et al., 2000). These new works raised a new surge of interest in PFMs in mobile robotics ((Matarić, 1994, Barraquand et al., 1992, Balch and Arkin, 1995)).

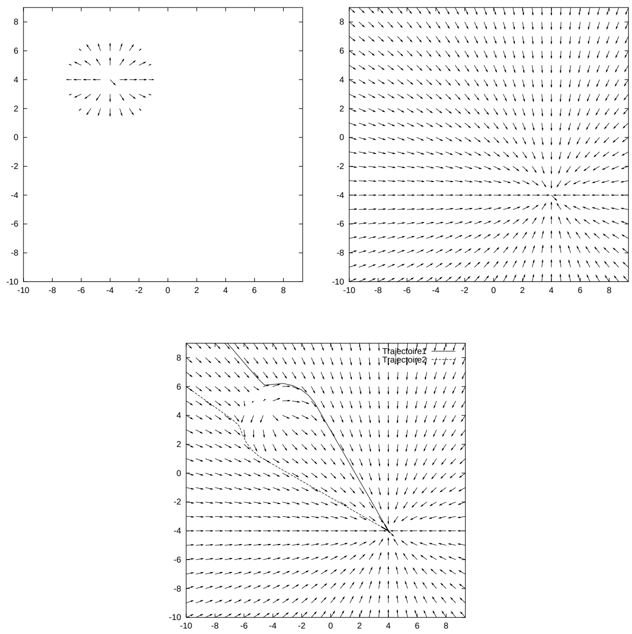

Figure 1: a) Repulsive field around an obstacle in B(-4,4); b) Attractive field towards a target in C(4,-4) c) Global field combining (a) with (b) and resulting trajectory of a mobile sensitive to both fields

Nowadays, the methodology is renewed by a trend applying it to collective behavior problems, giving rise to so called "Social Potential Fields". For instance, Panatier (Panatier et al., 2000) uses potential fields to build an internal model of the behavior of other agents expressed in tropistic terms. Then the agent can anticipate the behavior of other agents and to coordinate its behaviors with theirs. In the same vein, Simonin and Ferber (Simonin and Ferber, 2000) have shown the efficiency of potential fields methods to support signal communication between agents in a reactive multiagent coordination context.

Sharing common goals with our work and a similar formalism, Balch extends Arkin's schema theory (Balch and Arkin, 1995) to solve multiagent problems. In particular, his work was intended to keep a group of military ground vehicles in formation. More recently, drawing inspiration from crystal structures, Balch and Hybinette (Balch and Hybinette, 2000) also added to the classical framework a set of one to four *attachment sites*, arranged regularly around each agent, attracting other agents according to a *maintain formation* schema, giving rise to different group geometries depending on the positions of the sites.

The framework presented below can be seen as an extension of the original framework from (Balch and Arkin, 1995) towards less design effort from the designer. We also added the equivalent of *attachment sites* to our framework, but this is not presented here (see (Flacher, 2001)).

3. Our framework

3.1 The formalism

In multiagent simulations, entities of different types are present in the environment of each agent. We call such entities "landmarks" when they are relevant to the behavior of an agent. At each time step, the agent is located in a point A and sorts all its landmarks into lists depending on their type, according to their relative distance. With this way of sorting landmarks, we can implement easily a local perception limited to some range, so as to define *situated* agents in the sense given in the introduction. From the landmarks lists, the agent defines a set of *points of interest* P_i , each being the barycentre of some landmarks L_k. Hence, $\overrightarrow{AP_i} = \Sigma_{k=0}^{m_i} \beta_{ik}.\overrightarrow{AL_k}$, where β_{ik} are barycentric coefficients associated to P_i.

A normalized *function of magnitude* F_i modulates the influence exerted by the point P_i on the agent according to the distance $\| \overrightarrow{AP_i} \|$. This function is piecewise linear and defined by a set of q_i points whose coordinates are (x_{ik}, y_{ik}). Each function F_i generates an attractive or repulsive force towards P_i which acts on the agent. The force is attractive when the function is positive and repulsive when it is negative. This formalism generalizes the classical obstacle/target dichotomy in a unified formalism since the same *function of magnitude* can be either attractive or repulsive. The corresponding force is modulated by a coefficient G_i, which represents the relative gain of the point of interest with respect to other points.

Finally, the movement $\overrightarrow{D}$ resulting from the combination of all forces on the agent is given by the equation:

$$\overrightarrow{D} = \Sigma_{i=0}^{N} \left[G_i \times F_i(\| \overrightarrow{AP_i} \|) \times \overrightarrow{AP_i} \right] \quad (1)$$

This equation defines the controller of our agents. Each controller can be expressed by the set of all parameters involved in equation 1. The GA is applied to this set of parameters, so as to select controllers which perform well on the task they have to solve and discard controllers which get trapped into local minima.

3.2 Evolution of the Model

3.2.1 Genetic encoding

A genome is encoded as N vectors of real numbers representing all the parameters of our framework.These parameters are initially uniformly distributed over the range $[-1, +1]$. These parameters are the following:

- the coefficient G_i of all functions of magnitude,
- the ranges and types (r_{ik}, t_{ik}), together with the coefficient β_{ik} of all landmarks k defining a point of interest P_i;
- the coordinates (x_{il}, y_{il}) of the segment extremities defining all piecewise linear functions of magnitude.

A vector containing these parameters is called a chromosome, and each chromosome codes for one association (gain + point of interest + magnitude function).

3.2.2 Genetic Operators

The genetic operators, mutation and crossover, are adapted to our formalism to prevent the generation of meaningless controllers.

- *mutation* : a random value depending on a normal law is added with a probability P_M to each parameter of the genome. These parameters are then mapped using linear maps from $[-1, +1]$ to a given range of parameters values as follows:
  - type values $\in \{0, 1, ...N\}$,
  - range values $\in \{1, .., f_{MaxNb}(type)\}$,
  - barycentric weights of points of interest $\in [-10, 10]$,
  - gain values $\in [-5000, 5000]$,
- *crossover* : individuals I_1 and I_2 respectively have N_1 and N_2 chromosomes. The crossover is realized by selecting, with a probability P_C, N_3 chromosomes in the set $(N_1 + N_2)$ of chromosomes from I_1 and I_2. The selected chromosomes are then copied into the genome of the new individual.

As a result of the crossover operator, the number of chromosomes in genomes varies from one individual to another.

3.2.3 Description of the algorithm

The evolutionary algorithm used in the experiments described here is similar to the one proposed by (Kodjabachian, 1998). Thanks to the use of ecological niches, it prevents a premature convergence by distributing the population on a wheel and selecting candidates for evaluation into local windows around that wheel. A new individual is generated according to the following algorithm:

- a neighborhood window of size N_W which is proportional to the size of the population N_P is positioned on the wheel;

- two individuals I_1 and I_2 are selected in this window with a probability proportional to their fitness according to a roulette wheel selection scheme (Goldberg, 1989);
- genetic operators are applied to I_1 and I_2 to generate a new individual I_{Son};
- a bad individual I_B is selected in the window with a probability proportional to the inverse of its fitness;
- I_{Son} is tested;
- if the new individual I_{Son} is better than its parents, it replaces I_B.

We consider that the next generation is generated when this operation is repeated N_P times.

4. A Case Study

Our environment is inspired from the Robot Sheepdog Project, which involves a robot driving a flock of ducks towards a goal position in a circular arena (Vaughan et al., 1998). We extend this experimental setup to the case where the same task has to be solved by the coordinated effort of several simulated agents.

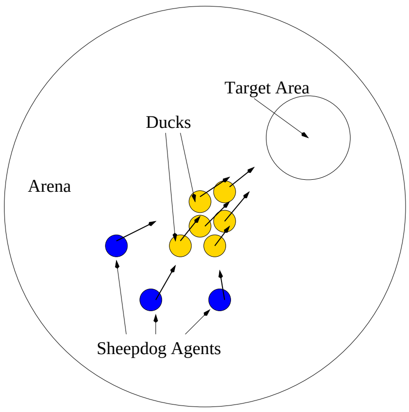

Figure 2: The arena, ducks and shepherds

The simulator shown on figure 2 includes a flock of ducks and some shepherds which must drive the flock towards a goal area. In the following experiments, all simulations always involve six ducks. The ducks and the shepherds have the same maximum velocity. The goal is achieved as soon as all the ducks are inside the goal area.

The behavior of the ducks results from a combination of three tendencies. They tend:

- to keep away from the walls of the arena;
- to join their mates when they are within their visual range;
- to flee from the shepherds which are within their visual range.

In the context of that case study, the type values and ranges for these values are given by the table 1.

Type	Landmark	$f_{MaxNb}(type)$
0	goal area	1
1	wall landmark	12
2	duck	6
3	shepherd	20

Table 1: Table of the parameters of our simulations

5. Empirical results

5.1 *Experiments with Learning Classifier Systems*

Basically, a Learning Classifier Systems (LCS) is a rule-based system able to improve its set of rules thanks to both GA and Reinforcement Learning Algorithms. Since the rules are (condition, action) couples, a LCS can define a controller, and its adaptive algorithms can be used to optimize this controller with respect to some criterion. For an introduction to LCS, the reader is referred to (Stolzmann et al., 2001). The work presented in this section is a very brief summary of (Sigaud and Gérard, 2001). In order to apply LCS to the problem defined in section 4., it is necessary to first design the set of inputs and actions considered by the classifiers. This in turn implies that a designer defines a general strategy to fulfill the task even before applying adaptive algorithms.

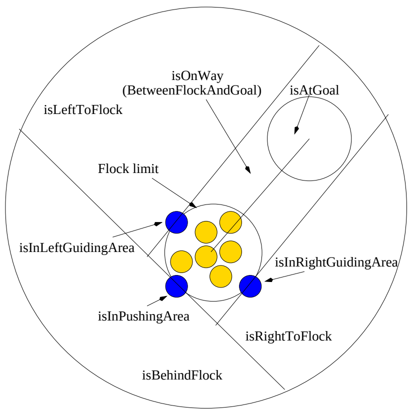

Figure 3: Representation used by Learning Classifier Systems

The strategy defined in our case consists in first surrounding the ducks so that there flock do not scatter when the agents get too close to them, and then drive the flock towards the goal.

It appeared difficult to implement this simple strategy with LCS. The inputs of the shepherds are shown on figure 3. There are 16 basic behaviors, among which 8 are shown below:

•goToFlockCenter	•followFlockToGoal
•goBehindFlock	•goToPushingPoint
•goToLeftOfFlock	•goToClosestDuck
•goToOutmostDuck	•goAwayFromFlock

We finally had to attribute to each agent a role, either pushing the flock or guiding it by the left hand side or by the right hand side. Since a role binds to each agent a particular function, this solution can be seen as translating a spatial coordination problem into a functional coordination problem. A further improvement was achieved by giving to the shepherds the ability to exchange their roles. All the corresponding research is described in details in (Sigaud and Gérard, 2001).

From figure 4, it appears that despite an important design effort, we did not succeed in obtaining a system whose performance is improved when the number of agents is augmented. We will come back to the reasons for this failure in section 7.

The purpose of testing the controllers with shepherds groups of increasing size was to check whether these controllers were scalable. Scalability in terms of number of agents is an important issue in complex coordination problem studies. For instance, scalability is one of the key features of the approach of (Balch and Hybinette, 2000) presented in section 2., and we had to check this was the case of ours.

5.2 Experiments with PFM and AG

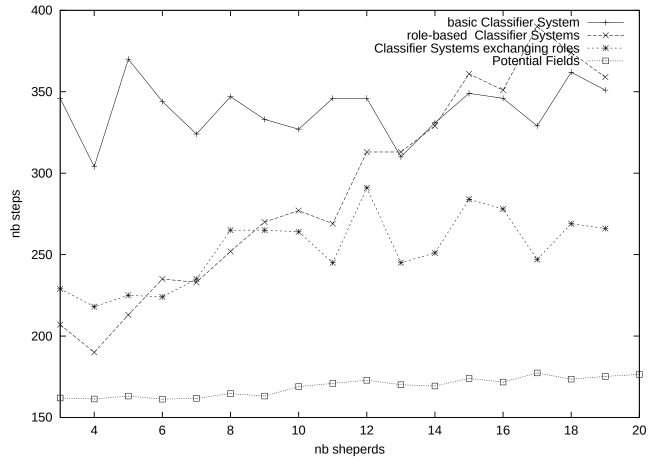

Figure 4: Average number of time steps to reach the goal over 100 trials of different controllers w.r.t. the number of shepherds (the lower, the better)

In our experiments with the framework presented in section 3., all the shepherds share the same controller. Their fitness function is defined by the remaining time after completing the task with respect to the maximum time allowed to fulfill it [1]. The population size N_P is 100 individuals, the size of the neighborhood window N_W is 5 individuals, the maximum time 500 time steps, and each evaluation of an individual involves 25 trials, with a random number of shepherds ranging between 3 and 20. The probability of mutation P_M is 5% and the probability of crossover P_C is 80%.

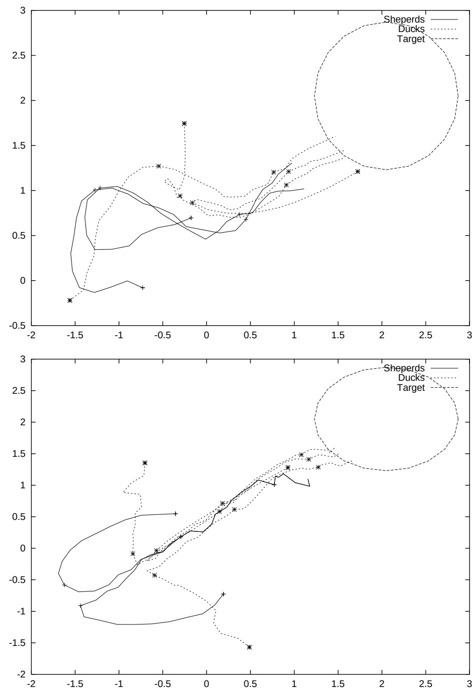

Figure 5: Various trajectories of ducks and shepherds endowed with the controller C_{150}. Positions are initialized randomly

Figure 4 shows the performance of a particularly efficient controller, called C_{150}, obtained among many others, compared to results of the previous studies based on LCS described in section 5.1 From figure 4, it is striking that the controller C_{150} is much more efficient than all controllers based on LCS. It can be seen that the controller C_{150} is scalable, since its performance hardly decreases as the number of agents is augmented.

6. Explaining the results

Figure 5 shows several trajectories of the agents obtained with the controller C_{150}. We only depict the behavior of 3 ducks and 2 shepherds for keeping the figure readable, but the trajectories shown are representative of the

[1] Hence, the higher, the better, in contrast with the performance shown in figure 4

whole group. These trajectories can be analyzed as follows: at initialization time, all the agents are randomly scattered in the arena. Then the completion of the task can be decomposed into two stages. During a first stage, the shepherds move away from the ducks, behind them with respect to the goal, leaving them form a flock. In the second stage, they surround the flock by the back with respect to the goal, and they drive the ducks towards the goal area by moving towards them.

Since the controller C_{150} is quite simple, it has been possible to understand how it works qualitatively. It only involves three points of interest:

- P_0 associated to the first [2] duck (with a coefficient $\beta_{D1} = 0.85$),
- P_1 associated to the third ($\beta_{D3} = 9.13$) and fifth ducks ($\beta_{D5} = 2.61$),
- P_2 associated to the goal ($\beta_G = 9.08$), to the fourth shepherd ($\beta_S 4 = 2.42$), and to the closest wall landmark ($\beta_{W0} = 0.77$).

The functions of magnitude generated by the GA are shown on figure 6. The corresponding G coefficients are: $G_0 = 1271.62$, $G_1 = 1758.97$ and $G_2 = 2918.08$. In order to compare the relative forces exerted by these three points of interest on the agent, we define $H_i = G_i \times f_i$, where f_i is the value of the function of magnitude F_i in its constant part, *i.e.* for a distance greater than 0.6. Then we have $H_0 = 771.13$, $H_1 = 660.73$ and $H_2 = 1161.59$. The explanation of the behavior is the following:

- Thanks to their three tendencies, if they are left alone, the ducks tend to converge from random widespread locations to form a single flock away from the walls of the arena.
- If they are alone in the arena, the shepherds are only sensitive to P_2 (P_0 and P_1 are dedicated to ducks). Since F_2 is negative for a distance over 0.5 and since the main contribution to P_2 is the landmark associated to the goal, the shepherds tend to be repulsed from the goal towards the walls of the arena. The point of interest P_2 stays close to the goal because β_G is bigger than $\beta_S 4$. But $\beta_S 4$ guarantees that, if there are more than 3 shepherds, they won't stay grouped into a compact pack.
- Since H_2 is bigger than H_0 and H_1, as long as the ducks and shepherds are located randomly, the effect of P_2 is stronger than the effects of P_0 and P_1. As a consequence, the shepherds scatter away from the goal. That helps the ducks to form a flock. Furthermore, when the flock gets formed, the shepherds are already positioned behind the flock with respect to the goal (see figure 7).
- But, as soon as the flock is formed, P_0 and P_1 are close to each other, the attraction vectors they generate get collinear, their attractive effects are combined and, since $H_0 + H_1$ is bigger than H_2, the shepherds move towards the ducks (see figure 8).
- Thanks to P_2, while they are attracted by P_0 and P_1, the shepherds also tend to go exactly behind the ducks with respect to the goal, which ensures that all agents finally reach the goal (see figure 9).

[2] *i.e.* closest at that particular time step, since the landmarks are sorted according to their distance to the agent at each time step

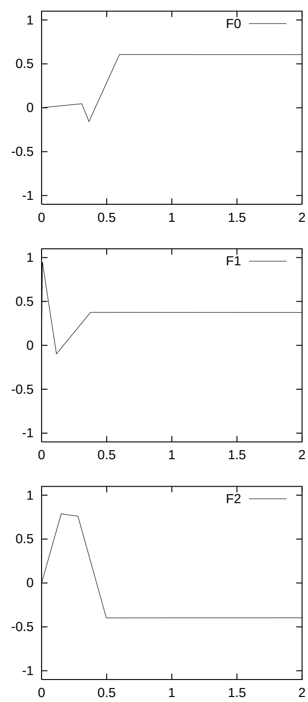

Figure 6: Functions of magnitude associated to the three points of interest

6.1 *Robustness study*

Robustness is a crucial issue of the validation of the solutions obtained by GA. We have already shown that the controller C_{150} is *scalable*, *i.e.* robust with respect to the number of shepherds. But the variation of the number of shepherds was introduced into the evaluation of a controller by our GA, though not systematically. It is also robust with respect to the initial positions changes. Indeed, in figure 4, the performance of the evolved controllers was the average performance over 1000 different initial situations. In spite of the fact that these controllers were evolved on 25 runs only, the obtained controllers are not only better than the LCS controllers, but

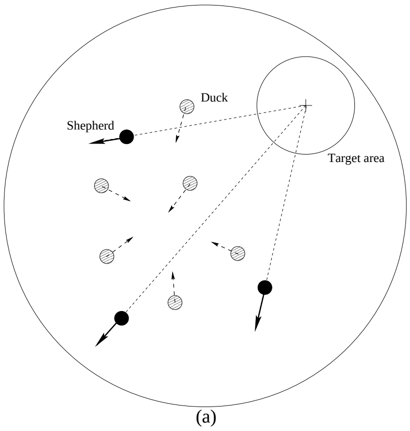

Figure 7: Initialization: initial positions are random. The shepherds go away from the goal, the ducks towards each other

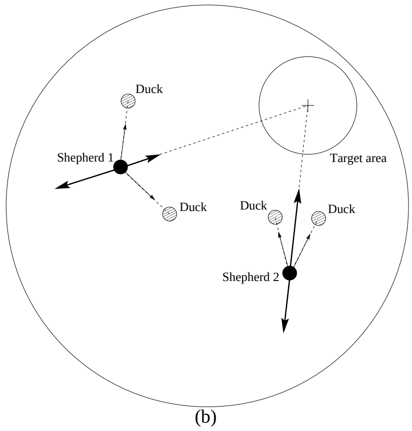

Figure 8: Balance between attraction towards the ducks and repulsion from the goal: if the ducks are scattered, repulsion wins (cf. shepherd 1). If the ducks are close to each other, attraction gets stronger than repulsion (cf. shepherd 2)

they are also robust with respect to the initial positions changes.

Another important robustness test is about the variation of the speed of shepherds. A higher speed may be a very detrimental factor since, if the shepherds can move faster than the ducks, they can rush into the flock and the ducks will scatter, which might be very detrimental to the completion of their task. This new criterion was not varied among the different runs during the evolution of controllers. Thus we must check if there is a mechanism in the controller C_{150} which prevents the agents to get too close to the flock. In order to investigate this question, we doubled the speed of our agents so that they could be twice faster than the ducks and tested the controller C_{150} again. As it can be seen on figure 10, under high speed conditions the controller C_{150} is a bit less adapted with few agents, but it gets even better than at normal speed when there are more agents. This good result is all the more striking than controller C_{150} was not evolved at that higher speed. Furthermore, it appears in these new experiments that the agents, even if they are faster than ducks, do not get too close to them. This proves that there is a mechanism in controller C_{150} which prevents the agents to do so.

There are two candidates for such a mechanism, that we will call M_1 and M_2.

- M_1 is provided by the shape of the magnitude functions: as can be seen on figure 6, when the distance between the agent and the flock gets too small, the attractions towards the ducks generated by F_0 and F_1 become repulsions. The distance between the agent and a point of interest should stabilize where the attraction/repulsion function is null.

- M_2 is the mechanism explained on figure 8: if the agent gets too close, the ducks will scatter, but then the repulsive force from F_2 will get bigger again than the combination of F_0 and F_1, and the agent will stop getting closer.

So as to determine which of these mechanisms is involved in the regulation of the distance to the flock, we conducted a further experiment which consists in replacing the magnitude functions of the controller C_{150} by the constant functions obtained by extending the constant part of F_0, F_1 and F_2 up to $d = 0$, both at normal and double speed. The results are shown on figure 10.

In the case of normal speed, the controller with constant magnitude functions is even more efficient than the controller C_{150}. This is not a complete surprise, since in our explanation of the behavior given in section 6., the only part of the magnitude functions considered are the constant parts after $d = 0.6$. Thus having a piecewise linear magnitude function is a useless feature in that particular context. In particular, the attractive part of F_2 is of no use since the agents never get into the goal area before the trial ends. We can also conclude that M_1 is not involved in that case: if the agents do not rush into the flock, this is both because the ducks are as fast as them and because of the mechanism M_2.

In the case where the agents have a double speed, on the contrary, the controller C_{150} performs better than the controller with constant magnitude functions. In that case, thus, M_1 is clearly involved.

As a conclusion of this further study, a nice feature of the controller C_{150} is the fact that several dynamical mechanisms are involved in the spatial coordination of the behavior of the agents. This results in a very robust performance in varying conditions of use.

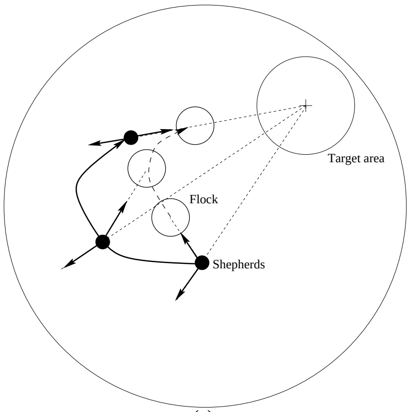

Figure 9: Driving the flock towards the goal: repulsion from the goal guarantees that the shepherds go behind the flock until they finally get aligned w.r.t. the goal

7. Discussion

Since our experiments were conducted on a task inspired from (Vaughan et al., 1998), comparing the strategy implemented by our controller with the one hand-crafted by Vaughan is insightful.

In (Vaughan et al., 1998), Vaughan proposes an equation to control his robot on the same flock control task:

$$\overrightarrow{D} = K_1.\overrightarrow{RF} - (\frac{K_2}{|RF|^2}).\overrightarrow{RF} - K_3.\overrightarrow{RG}$$

where G is the center of the goal area, F is the center of the flock, R the position of the shepherd robot and K_1, K_2 and K_3 are manually tuned parameters. This equation can be seen as the combination of three terms.

- The first two terms generate an attraction of the robot towards the flock, but if $|RF|$ gets too small, the negative term $-\frac{K_2}{|RF|^2}$ repulses the robot away from the flock. Thus this combination should stabilize the robot at a distance from the flock, thus prevent the ducks from scattering.
- The third term, $-K_3.\overrightarrow{RG}$ repulses the robot away from the goal.

In a latter paper (Vaughan et al., 2000), a new equation was proposed:

$$\overrightarrow{D} = (K_{r1}|GF|).\overrightarrow{RF} - K_{r2}.\overrightarrow{RG}$$

where K_{r1}, K_{r2} are also manually tuned parameters.

This time, the first term, $(K_{r1}|GF|).\overrightarrow{RF}$, attracts the robot towards the flock but according to its distance to the goal. The further the flock is from the goal, the more the robot is attracted by the flock. The second term, $-K_{r2}.\overrightarrow{RG}$ was already present in the previous equation and repulses the robot away from the goal.

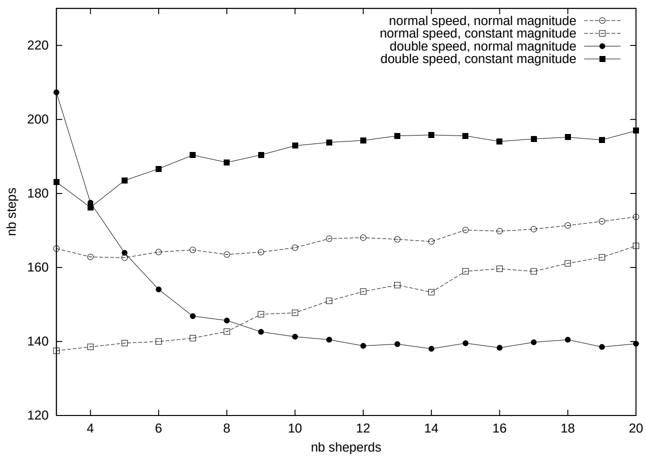

Figure 10: Performance of the controller C_{150} when the speed of the shepherds is normal or doubled and when the magnitude functions are normal or constant. The results are averaged over 1000 runs, each starting with random initial positions

Thus the general approach of the strategies handcrafted by Vaughan is very similar to the one implemented by the controller C_{150} and explained in section 6.: it consists in a careful combination of attraction towards the flock and repulsion from the goal which drives the robot to get aligned with the goal and the flock from behind and, thereafter, pushes it towards the goal, as explained on figure 9. Unfortunately, since the various K parameters are not given in (Vaughan et al., 1998, Vaughan et al., 2000) and since they use one robot only, we cannot provide a quantitative comparison of the performance of our controllers.

Anyway, our framework has found without human intervention a very robust and smart solution similar to another one that was carefully designed by a research team through successive publications.

Thus it seems necessary to stress in this discussion the issue of amount of design effort necessary to solve complex coordination problems with different frameworks.

On the experimental problem described in section 4., the less efficient methodology probably consists in discretizing the description of the problem and tuning by hand a controller to solve the problem in this discretized representation. With that respect, applying adaptive techniques such as LCS to this discretized representation so as to automate the tuning process is already an improvement.

But, as explained in section 5.1, our experiments described in (Sigaud and Gérard, 2001) have shown that applying LCS to such a problem requires that the designer identifies an *ad hoc* set of discrete inputs and basic behaviors. This identification is generally achieved through a functional decomposition: decomposing the global problem into subproblems leads to designing basic behaviors, and identifying in which conditions these behaviors should be fired results in a list of discrete inputs. Since the controller of the agents is expressed in terms of rules connecting inputs to basic behaviors, the quality of the resulting behavior heavily depends on this preliminary design effort, even if the rules themselves are optimized by some adaptive algorithms.

Rather than using discrete representations and basic behaviors, another family of methodologies consists in trying to use directly a continuous representation. The resulting continuous controller can be defined by hand in a purely *ad hoc* fashion. This is the case for instance of Vaughan's work described in the previous section.

A better methodology of the same family consists in defining a set of continuous basic attraction or repulsion schemas and combining them to solve a particular problem. One argument in favor of this methodology is that the same schemas can be reused in different contexts. An example of such a methodology is the PFM defined by Arkin and Balch. By designing by hand their schemas, they succeed in generating effective controllers for navigation and spatial coordination without any preliminary discretization of the environment, and schemas like obstacle avoidance are reused on different problems.

One of the key differences between the use of basic behaviors and basic attraction or repulsion schemas is that, in the case of basic behaviors, what must be found is the correct sequencing of the activation of these behaviors. This is done through tuning the conditions of activation of each basic behavior and it turns out to be difficult and time consuming. In the case of basic schemas, on the contrary, all schemas can be active at all time, thus the sequencing problem simply disappears.

Our formalism is very close to the one defined by Balch and Arkin. Indeed our concept of point of interest is equivalent to that of point aimed by the schema, our function of magnitude is equivalent to Balch and Arkin's one and we both modulate it by a gain. Our approach shares with that of Balch and Arkin the design simplicity provided by PFM. But, instead of defining a set of basic schemas by hand, we just let a GA find a global combination of a set of local attraction/repulsion functions. The only thing that the designer has to do in our framework is to state, for each agent, which other landmarks in the simulation are of interest to it.

Our position with respect to schema reuse, as Balch and Arkin do, is twofold. First, since in our method involving PFMs and GAs, the design effort necessary to define each basic schemas is not required anymore, the system will find by itself each time new schemas adapted to the particular problem at hand.

But the situation might be different if we were pretending to address even more complex problems implying a set of different goals and of different strategies to reach sequentially these different goals. In such a case, relying on a GA might prove infeasible because of constraints on the time it takes to converge to a satisfying solution. In that case, our formalism does not prevent us from reusing and combining already evolved basic schemas exactly as Balch and Arkin do. We have tested this possibility in experiments not described here.

Thus we are quite confident about the fact that our approach is among the ones which requires the smallest design effort. However, what we have spared in design must now be spent in understanding the obtained solution. In the case of the controller presented in section 6., understanding the behavior has been possible because it was simple and because the underlying mechanism did not involve too many interactions between agents. In particular, in section 6., we have shown how we could investigate fine grained phenomena in the behavior of our agents without calling upon complex mathematical tools from dynamical systems theory. However, such favorable circumstances will probably be uncommon in more complex problems.

8. Future Work and Conclusion

The fact that controllers generated through the methodology advocated in this paper are probably difficult to analyze suggests an agenda of research devoted to identifying useful formal tools, possibly calling upon dynamical systems theory, that would help understanding how and why such controllers work. What we have shown here needs to be generalized to other applications in order to identify which mechanisms deserve a more precise formalization. We are already engaged in new experiments where we try to apply our formalism to maintaining a patrol of military aircrafts into formation during a mission involving incursion in enemy territory (see (Flacher, 2001)).

As a general conclusion, we have shown that combining PFMs with GAs is a very powerful methodology which requires a very small design effort and is more efficient in terms of performance than functional decomposition. Our framework extends Arkin and Balch's approach in a promising direction, which extensively relies on the self-organization mechanisms provided by GA, resulting in a lesser involvement of the designer. As a consequence, however, this methodology requires additional research on analysis tools that would help understanding more accurately how the corresponding controllers really work.

References

Arbib, M. (1981). Perceptual Structure and Distributed Motor Control. In Brooks, (Ed.), *Handbook of Psychology – The nervous system II*, pages 1441–1465.

Arbib, M. and House, D. (1985). Depth and Detours: An essay on Visually Guided Behavior. COINS TR 85-20, University of Massachussetts, Amherst, MA.

Arkin, R. C. (1989). Motor Schema-Based Mobile Robot Navigation. *The International Journal of Robotics Research*, 8(4):92–112.

Balch, T. and Arkin, R. C. (1995). Motor Schema-Based Formation Control for Multiagent Robot Teams. In *Proceedings of the First International Conference on Multiagent Systems*, pages 10–16.

Balch, T. and Hybinette, M. (2000). Social Potentials for Scalable Multi-Robots Formations. In *IEEE International Conference on Robotics and Automation (ICRA-2000)*.

Barraquand, J., Langois, B., and Latombe, J. C. (1992). Potentials Fields for Robot Motion Planning. *IEEE Transactions on Systems, Man and Cybernetics*, pages 224–241.

Brooks, R. A. (1986). A Robust Layered Control System for a Mobile Robot. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume RA-2, pages 14–23.

Flacher, F. (2001). *Emergence de comportements collectifs : cas du maintien de formation*. Masters Thesis, University PARIS VI and DASSAULT AVIATION, St-Cloud, http://animatlab.lip6.fr/Flacher.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison Wesley.

Khatib, O. (1985). Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 500–505.

Kodjabachian, J. (1998). *Développement et évolution de réseaux de neurones artificiels*. Thèse de 3ième cycle, Université Pierre et Marie Curie.

Korenz, Y. and Borenstein, J. (1991a). Potential Field Methods and their Inherent Limitations for Mobile Robot Navigation. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 489–493.

Korenz, Y. and Borenstein, J. (1991b). The Vector Field Histogram – Fast Obstacle-Avoidance for Mobile Robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 7, pages 278–288.

Krogh, B. H. (1984). A Generalized Potential Field Approach to Obstacle Avoidance Control. In *International Robotics Researh Conference*.

Matarić, M. J. (1994). *Interaction and Intelligent Behavior*. PhD thesis, MIT AI Mobot Lab.

Panatier, C., Sanza, C., and Duthen, Y. (2000). Adaptive Entity thanks to Behavioral Prediction. In Meyer, J.-A., Wilson, S. W., Berthoz, A., Roitblat, H., and Floreano, D., (Eds.), *From Animals to Animats 6: Proceedings of the Sixth International Conference on Simulation of Adaptive Behavior (supplement book)*, pages 295–303, Cambridge, MA. MIT Press.

Partridge, B. L. (1982). The Structure and Functions of Fish Schools. *Scientific American*, pages 114–123.

Pearce, M., , Arkin, R., and Ram, A. (1992). The Learning of Reactive Control Parameters through Genetic Algorithms. In *Proceedings 1992 International Conference on Intelligent Robotics and Systems (IROS)*, pages 130–137.

Piaggio, M., Sgorbissa, A., and Zaccaria, R. (2000). Micronavigation. In Meyer, J.-A., Wilson, S. W., Berthoz, A., Roitblat, H., and Floreano, D., (Eds.), *From Animals to Animats 6: proceedings of the Sixth International Conference on Simulation of Adaptative Behavior*, pages 209–218, Paris. MIT Press.

Sigaud, O. and Gérard, P. (2001). Being Reactive by Exchanging Roles: an Empirical Study. In Hannebauer, M., Wendler, J., and Pagello, E., (Eds.), *LNAI 2103 : Balancing reactivity and Social Deliberation in Multiagent Systems*. Springer-Verlag.

Simonin, O. and Ferber, J. (2000). Modeling Self Satisfaction and Altruism to Handle Action Selection in Reactive Cooperation. In Meyer, J.-A., Wilson, S. W., Berthoz, A., Roitblat, H., and Floreano, D., (Eds.), *From Animals to Animats 6: Proceedings of the Sixth International Conference on Simulation of Adaptative Behavior (supplement book)*, pages 314–323, Paris. Cambridge MA.

Stolzmann, W., Lanzi, P.-L., and Wilson, S. W. (2001). *LNAI 1996 : Advances in Classifier Systems*. Springer-Verlag.

Vaughan, R., Stumpter, N., , Henderson, J., Frost, A., and Cameron, S. (2000). Experiments in Automatic Flock Control. *Robotics and Autonomous Systems*, 31:109–117.

Vaughan, R., Stumpter, N., Frost, A., and Cameron, S. (1998). Robot Sheepdog Project achieves Automatic Flock Control. In Pfeifer, R., Blumberg, B., Meyer, J.-A., and Wilson, S. W., (Eds.), *From Animals to Animats 5: Proceedings of the Fifth International Conference on Simulation of Adaptive Behavior*, pages 489–493, Cambridge, MA. MIT Press.