

Evolving Evil: Optimizing Flocking Strategies through Genetic Algorithms for the Ghost Team in the Game of Ms. Pac-Man*

[Published in EVOGames 2014 (EVOApplications)]

F. Liberatore, A.M. Mora, P.A. Castillo, J.J. Merelo

Departamento de Arquitectura y Tecnología de Computadores.

CITIC-UGR, ETSIIT.

University of Granada, Spain

`federico.liberatore@urjc.es`

`{amorag,pacv,jmerelo}@geneura.ugr.es`

Abstract. Flocking strategies are sets of behavior rules for the interaction of agents that allow to devise controllers with reduced complexity that generate emerging behavior. In this paper, we present an application of genetic algorithms and flocking strategies to control the Ghost Team in the game Ms. Pac-Man. In particular, we define flocking strategies for the Ghost Team and optimize them for robustness with respect to the stochastic elements of the game and effectiveness against different possible opponents by means of genetic algorithm. The performance of the methodology proposed is tested and compared with that of other standard controllers. The results show that flocking strategies are capable of modeling complex behaviors and produce effective and challenging agents.

1 Introduction and Problem Description

The game of Ms. Pac-Man was released in 1981, presenting a new female protagonist, who has to collect all the pills in a maze while avoiding four ghosts chasing her. As in the original game of Pac-Man, if Ms. Pac-Man is touched by a ghost the player loses one life, then she is relocated at the initial position, and the ghosts respawn from the center of the maze. The *powerpills* turn the ghosts vulnerable for a short time, allowing Ms. Pac-Man to “eat” them. When a ghost gets eaten, it disappears from the game and respawns at the center of the maze after a certain amount of time. As the levels are cleared, the game becomes more difficult by changing certain parameters such as respawn time, length of time the ghosts are vulnerable, and ghosts’ speed. The main difference with respect to the original game of Pac-Man, in addition to the new maze designs and, of course to the new main character, is that this game has elements of randomness,

* This is a summary of the work [3] previously published in EVOGames track of EVOApplications 2014 Conference, included in EVO* 2014

apparently included to make the game more engaging. Thus, occasionally, there is a global reversal event when all the ghosts suddenly change direction.

Given its multiple challenges, the game has been chosen for the Ms. Pac-Man vs Ghosts competition, a game AI competition started in 2009 [4], where participants can submit controllers for both Ms. Pac-Man and the Ghost Team. The aim of Ms. Pac-Man agents is to maximize the final score, while the aim of Ghost Team controllers is to minimize it. During the competition, the controllers are ranked according to the results of random matches between two controllers of the same kind (e.g., Ghosts controllers) against two other controllers of the opposite kind (e.g., Ms.Pac-Man controllers). The controllers of each type that get the best score win the match and increase their rank. The competition is won by the controller of each kind having the highest rank.

The version of the game implemented for the competitions differs slightly from the original one. A thorough description of the game rules can be found in [8]. For the purposes of this work, the relevant restrictions for the Ghost Team are briefly enlisted in the following:

- A ghost can never stop and, when it is in a corridor, it can only move forward.
- A ghost can choose its direction only at a junction. Specifically, a ghost can only move into a corridor different from the one it is coming from. As a result, a ghost cannot turn back on itself.
- Every time a ghost is at a junction the controller has to provide a direction (i.e., UP, DOWN, LEFT, or RIGHT) from the set of feasible directions, i.e., those directions corresponding to corridors different from the one the ghost is coming from. If no direction or an unfeasible direction is returned by the controller, the game framework chooses a random direction from the set of feasible directions.
- At every tick of the game all the ghosts obligatorily reverse their direction according to a small random probability, set in the game implementation to 0.005.
- After 2000 game ticks, a level is considered completed: Ms. Pac-Man is rewarded with the points of the remaining pills and the game moves on to the next level.

Thinking in this competition, and focusing in the Ghosts Teams, it can be noticed that it is a group of individuals that perform simple actions (i.e., moving up, down, left, or right), so it seems a natural proving ground for algorithms based on the paradigm of Swarm Intelligence (SI) [1]. SI is the term used to describe the type of coordinated intelligence that arises from the collective behavior of decentralized, self-organized systems, either natural or artificial.

Moreover, in this work we have focused on the so-called Flocking Strategies (FSs). Flocking refers to a SI technique proposed by Reynolds [7] for the coordinated movement of multiple AI agents. Originally, flocking algorithms have been developed to mimic lifelike behaviors of groups of beings such as herds of animals and schools of fishes. A flocking system typically consists of a population of simple *agents* (or *boids*) interacting locally with one another depending on the distance between them. The agents follow very simple steering behaviors:

- *Separation* makes the agent steer away from close flock mates.
- *Alignment* makes the agent steer toward the average heading of the flock.
- *Cohesion* makes the agent steer toward the average position of distant flock mates.

Despite the lack of a centralized control structure dictating how individual agents should behave, the interactions between such agents lead to the emergence of “intelligent” global behavior, unknown to the individual agents [10]. Due to this desirable property, the easiness of implementation, and the reduced computational cost, flocking algorithms have been extensively applied to many fields, such as cinematography, art, medicine, etcetera. A presentation of flocking algorithms applications in videogames can be found in [9] and [6].

In this work we have combined this Ghost Team controller with Genetic Algorithms (GAs) [2]. Offline (not during the game) GAs are used to design FSs for the Ghost Team that are effective at minimizing Ms. Pac-Man final score and that are also robust with respect to the stochastic elements of the game. To the best of the authors knowledge this is the first work to actually applying flocking algorithms to the game of Pac-Man. Our objective is to understand how the proposed methodology would perform in comparison to controllers that use different approaches.

2 Ghost Team AI: Evolutionary Flocking

Every Flocking Rule (FR) for boids (ghosts, in this case), considers a number of concentric ring-shaped neighborhoods centered on the ghost. They are limited to different radii. When an agent (e.g., another ghost or Ms. Pac-Man) falls into one of these neighborhoods a steering force is applied on the ghost according to the magnitude associated to the neighborhood.

To find the steering force on an agent A resulting from the interaction with agent B, a difference vector and the Euclidean distance between the two agents are calculated. The magnitude is determined by the neighborhood where agent B belongs to. A negative magnitude corresponds to the behavior of separation, while a positive magnitude corresponds to the behavior of cohesion. No alignment behavior is included in this strategy model as it would make the ghosts very predictable.

Differently from the basic flocking algorithm where only one type of agent is considered, in the game Ms. Pac-Man a variety of different actors are present. Also, the ghosts can be in different states: *HUNTER* is the “normal” state of a ghost (i.e., kills Ms. Pac-Man if touched). When Ms. Pac-Man eats a powerpill all the ghosts become *HUNTED* for a certain length of time (i.e., is killed by Ms. Pac-Man on contact). When this period is about to expire, every ghost blinks to warn the player; we call this state *BLINKING*.

In the same way, we can define a set of possible actors in the game, namely: *PACMAN*, *POWERPILL*, *HUNTER*, *HUNTED*, *BLINKING*, referring the three later to ghosts in that state. We can now define a Flocking Strategy

(FS) for the Ghost Team as a function that, given a ghost state and the type of actor considered, returns the flocking rule that has to be applied to calculate the steering force on the ghost resulting from the interaction with the actor.

As explained in Section 1, every time a ghost is at a junction the game needs to calculate its next move, so the controller based on the FS provides it.

A FS could be manually designed by an expert with decent results. Nevertheless, given as the number of parameters and the inherent complexity of the game, it is desirable to automatize the definition of an effective strategy by means of an optimization algorithm, such as GAs.

In the GA proposed, every *individual is represented by a FS*. The *initial population* is created as a random set of FSs, but ensuring that most of the magnitudes are close to zero and there are assign similar probabilities to the appearance of cohesion, separation, and no interaction behaviors.

The proposed GA should generate Ghost Team strategies that perform well against any possible Ms. Pac-Man strategy and, at the same time, should be resilient to the random ghosts reverse direction events (see Section 1). To achieve this result, each flocking strategy is pitted against two different Ms. Pac-Man controllers included in the Ms. Pac-Man vs Ghosts competition framework: *StarterPacMan* (SPM) and *NearestPillPacMan* (NPPM) (for a description of the controllers, please refer to the competition framework documentation¹). The game is simulated 30 times for each Ms. Pac-Man controller. Thanks to that we can take advantage of the central limit theorem to compute a relatively precise 95% confidence interval of the final score obtained by the Ms. Pac-Man controllers. This is done to minimize the effect of noise present in this problem and in videogames in general [5] since, due to the stochastic elements of the game, the same FS could perform very well sometimes and quite bad some others.

The *Fitness function* is defined as the sum of the inverses of the average scores obtained by each of the controllers in the 30 runs:

$$\text{FITNESS} = \frac{1}{\text{SCORE}_{\text{SPM}}} + \frac{1}{\text{SCORE}_{\text{NPPM}}} \quad (1)$$

After all the individuals (FSs) of the current generation have been evaluated, the offspring will be generated. For each individual to be generated, two parents are chosen by *roulette-wheel selection* (i.e., every member of the population has a probability of being chosen proportional to its fitness). The children individual is created by random recombination of the parameters of the parents.

During the recombination, *Mutations* can occur with certain probability. When a mutation happens, the current parameter is re-initialized to a random value. Initially, the probability value is set to $p^{\text{mut}} = 0.00125$. At each iteration t , its value changes depending on a coefficient of variation of the current population fitness, which measures the degree of variability of the population in terms of fitness. When the variability is low, we increment the mutation probability to introduce new chromosomes in the genetic pool of the population. When the variability is too big, the mutation probability is set to a low initial value.

¹ <http://www.pacman-vs-ghosts.net/>, last visited on May 23, 2014

3 Experiments, Results and Future Work

In this section, it will be tested how well a GA evolved controller performs, compared to non-evolutionary strategies. The standard Ghost Team controllers included in the competition framework will be used as a comparative basis. In the experiments, the GA described in the previous section has been run for 50 generations with a population of 50 candidate strategies. At each iteration, the next generation was constituted by 49 recombined individuals plus the best solution of the current generation.

A first experiment has been conducted comparing the performance of the Ghost Team controllers obtained with different values of the parameter N (i.e., the number of neighborhoods considered in the Flocking Rules).

	$N = 1$	$N = 2$	$N = 3$	$N = 4$	$N = 5$
Best FITNESS ⁻¹	783.38	726.84	815.66	766.96	720.20
Avg. FITNESS ⁻¹	871.30±55.45	861.57±70.13	876.31±65.06	905.86±62.66	863.17±72.15
Worst FITNESS ⁻¹	951.75	969.20	1,032.39	986.48	980.50
Avg. CPU time (s)	1373±150.66	1484.3±122.01	1561±193.94	1562.60±109.90	1473.00±74.02

Table 1. Performance of the controllers for the Ghost Team obtained by the GA using different numbers of neighborhoods.

Table 1 shows the performance of the evolved controllers over 10 runs of the GA with $N = 1, \dots, 5$. Each column is associated to a different number of neighborhoods. The first row displays the inverse fitness of the best individual found. The second row presents the average controllers fitness; the standard deviation is also reported after the plus-minus sign. Next, the third row illustrates the fitness value of the worst controller found. Finally, the last row reports the average optimization CPU time in seconds over the 10 runs and the corresponding standard deviation. By observing the table some conclusions can be drawn.

According to the results the best controller is obtained for $N = 5$ (the higher number of neighbours, the better). As a clarification we use the inverse fitness (FITNESS⁻¹) as a clearer measure of performance, so a lower value corresponds to a better controller.

In the next experiment, we compare our controllers to the five Ghost Team controllers included in the competition framework. Their FITNESS⁻¹ values, computed exactly as per the GA solutions, are presented in Table 2.

Controller	<i>AggressiveGhosts</i>	<i>Legacy</i>	<i>Legacy2TheReckoning</i>	<i>RandomGhosts</i>	<i>StarterGhosts</i>
FITNESS ⁻¹	1893.13	2210.94	1429.20	4200.70	1603.49

Table 2. Performance of the standard Ghost Team controllers included in the competition framework.

According to these results, the best controller is *Legacy2TheReckoning*, followed by *StarterGhosts*. Nevertheless, their FITNESS⁻¹ value is twice that of the best evolved FS found, approximately. These results support the claim that FSs are a viable option for the definition of intelligent controllers.

It is possible to see a video of the best evolved controller at <https://www.youtube.com/watch?v=I9rL0jUwHhk>. It is pitted against the Ms.

Pac-Man controller *StarterPacMan*. The video illustrates that, despite the lack of explicit coordination between them, the ghosts show complex strategic behaviors, entrapping and ambushing Ms. Pac-Man in a cooperative emergent way. This is obtained without including complex rules, which is a desirable feature in this type of problems (i.e., AI in games). This, in turn, results in the ghosts behaving in a “intelligent” fashion although they are not explicitly programmed with this objective in mind.

Some possible future lines of research include a fitness function extension by including more Ms. Pac-Man controllers. This should result in a Ghost Team controller that performs better against a wider range of opponents. Moreover, it would be interesting to compare the controllers obtained by applying the presented methodology with the best Ghost Team controllers that took part to the Ms. Pac-Man vs Ghosts competition. This would allow us to really understand the limits of FSs.

It would be also interesting to investigate the effectiveness of optimization methods that allows for small changes in the solutions parameters (instead of abrupt ones as in GAs), such as Particle Swarm Optimization (PSO).

References

1. Beni, G., Wang, J.: Robots and Biological Systems: Towards a New Bionics?, NATO ASI Series F: Computer and Systems Sciences, vol. 102, chap. Swarm intelligence in cellular robotic systems, pp. 703–712. SPRINGER-Verlag (1993)
2. Goldberg, D., Korb, B., Deb, K.: Messy genetic algorithms: motivation, analysis, and first results. *Complex Systems* 3(5), 493–530 (1989)
3. Liberatore, F., Mora, A.M., Castillo, P.A., Merelo, J.J.: Evolving evil: Optimizing flocking strategies through genetic algorithms for the ghost team in the game of ms. pac-man. In: Proc. Applications of Evolutionary Computing: EvoApplications 2014. p. In Press. Springer, LNCS (April 2014)
4. Lucas, S.: Ms Pac-Man versus ghost-team competition. In: Proceedings of IEEE Symposium on Computational Intelligence and Games (CIG 2009). p. 1 (2009)
5. Mora, A., Fernández-Ares, A., Guervós, J.M., García-Sánchez, P., Fernandes, C.: Effect of noisy fitness in real-time strategy games player behaviour optimisation using evolutionary algorithms. *Journal of Computer Science and Technology* 27(5), 1007–1023 (2012)
6. Rabin, S.: introduction to Game Development, Second Edition, chap. Artificial Intelligence: Agents, Architecture, and Techniques, pp. 521–557. Charles River Media (2010)
7. Reynolds, C.: Flocks, herds and schools: a distributed behavioral model. *Computer Graphics* 21(4), 25–34 (1987)
8. Rohlfschagen, P., Lucas, S.: Ms Pac-Man versus Ghost Team CEC 2011 competition. In: Proceedings of the 2011 IEEE Congress on Evolutionary Computation (CEC 2011). pp. 70–77. IEEE Press (2011)
9. Scutt, T.: AI Game Programming Wisdom, chap. Simple swarms as an alternative to flocking, pp. 202–208. Charles River Media (2002)
10. Spector, L., Klein, J., Perry, C., Feinstein, M.: Emergence of collective behavior in evolving populations of flying agents. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2003). pp. 61–73 (2003)