Proc. Asia Simulation Conference 2009 (JSST 2009), October7-9, 2009, Shiga, Japan.

Automatic Controller of Ms. Pac-Man and Its Performance:

Winner of the IEEE CEC 2009 Software Agent Ms. Pac-Man Competition

Ruck Thawonmas[1] and Hiroshi Matsumoto[1]

[1] Graduate School of Science and Engineering, Ritsumeikan University, Japan

Abstract—In this paper, we describe the outline of our Ms. Pac-Man controller, ICE Pambush 2, which is the winner of the IEEE CEC 2009 Software Agent Ms. Pac-Man Competition. One striking feature of ICE Pambush 2 is its ability to lure ghosts and ambush them. It is also equipped with improved image processing and decision making modules, leading to more than 8000 scores higher than its predecessor, ICE Pambush, submitted to the previous competition in the IEEE WCCI 2008. In addition, the score of ICE Pambush 2 is higher than those of controllers reported in the literature. At the time of writing this paper, ICE Pambush 2 is holding the world records for both the maximum score of 24640 and the average score of 13059, among ten trials, in this series of Software Agent Ms. Pac-Man Competitions.

Keywords— Ms. Pac-Man, controller, software agent, path finding, A*

1 Introduction

A series of Ms. Pac-Man controller competitions [1] have been held recently as a benchmark for advancing artificial intelligence and/or computational intelligence techniques. The first competition was held in affiliation with IEEE CEC 2007. We have participated in the competition since the second one at IEEE WCCI 2008. In this paper, we describe the outline and discuss the performance of our controller ICE Pumbush 2 which was developed for the third competition held at IEEE CEC 2009 and won the competition [2].

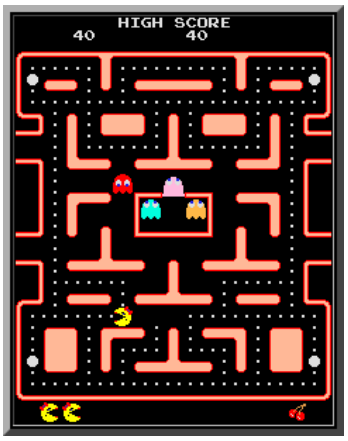


Fig. 1. A screenshot of Ms.Pac-Man.

Ms. Pac-Man (Fig. 1) was made in 1981 as a variant of Pac-Man. Main differences between these two are the appearance of the player character, with Ms Pac-Man having the ribbon on its head, and ghost behaviours, with ghosts in Ms. Pac-Man behaving almost non-deterministically. Objects in the game include Ms. Pac-Man, ghosts, pills, power pills, and items. Major game rules are as follows:

(a) In order to finish a given stage and proceed to the next stage, Ms. Pac-Man must eat all available pills and power pills.
(b) The life value, initially set to 3, is decremented each time Ms. Pac-Man is hit by a ghost, and the life value is added by one when the score reaches ten thousand,
(c) If Ms. Pac-Man eats a power pill, all available ghosts outside of the ghost cage will become edible for a particular period; if Ms. Pac-Man eats an edible ghost, a relatively high score can be earned.
(d) If Ms. Pac-Man eats an item, a score can be earned according to the item type.

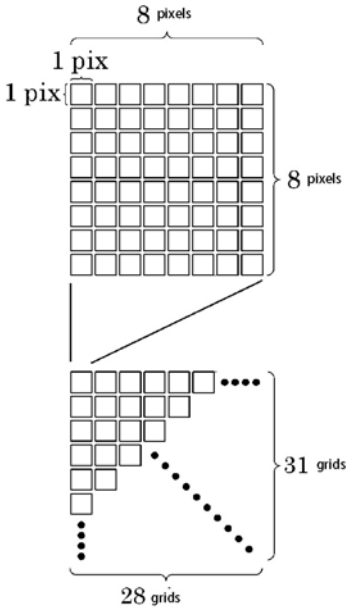


Fig. 2. Game map partition.

As shown in Fig. 2, the game map for all game stages consists of 28 x 31 grids, where each grid is composed of 8 x 8 pixels. Each stage has different objects and a map layout. All moving objects, i.e., Ms. Pac-Man, ghosts, and items, move along their paths in a pixel level.

ICE Pumbush 2 consists of two parts: one executed when the controller starts and the other executed iteratively for deciding which direction to move. The former part is done at the image processing module described in detail later. The latter one copes with controlling of Ms. Pac-Man movements. Because information in the game screen is updated every 67 milliseconds (ms), there exists a time constraint in the decision making process. This leads to a need for a strategy balancing the quality of decision results and the computation time. ICE Pambush 2's

mechanisms employed in these two modules have been significantly improved, from those of its predecessor ICE Pambush, in order to meet the aforementioned time constraint.

The contributions of this paper are as follows:

(1) Description of the ICE Pambush 2's efficient image processing module,
(2) Description of the ICE Pambush 2's effective decision making module, and
(3) Other useful information for new comers to facilitate their participation to the competition.

2 Image Processing

We proposed two mechanisms: on for speeding up the acquisition of the map information and the other for enabling effective movements between the corresponding warp points. The basic idea behind the former one is to detect all non-moving objects only once in the beginning of the game and iteratively trace the moving ones during the game. For the latter mechanism, the game map used in our controller is an extended version of the one shown in Fig. 1 by concatenating the right half of the original map + the original map + the left half of the original map. With this extended map (Fig. 3), our Ms. Pac-Man will go through a warp point of interest if its target location is near the corresponding warp point on the other side. The first mechanism contributes to a fast decision making time of 30 ms, compared with the previous version of 60 ms in ICE Ambush when the original map and the rules in ICE Ambush were used, and the second one contributes to more effective movements than ICE Ambush.

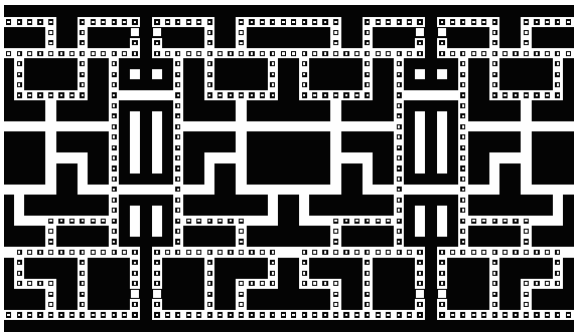


Fig. 3. The extended map used in ICE Ambush 2.

For acquisition of the map information, object detection is performed during a raster scan of all grids (each with 8 x 8 pixels) in the extended map. In the first scan, in order to derive the map layout, all pills, power pills, and path grids are detected by matching them with the patterns shown in Fig. 4. Then the landmarks in the map, all corners and warp points, as well as the ghost cage are detected. All of these objects are no longer considered in the subsequent scans, each of which is performed at the beginning of every iteration; when a pill or power pill is eaten by Ms. Pac-Man, the corresponding grid will be simply changed to a path grid. Thereby, the target recognition objects in a subsequent raster scan are limited to Ms. Pac-Man and all ghosts outside of the cage, leading to much less computation time, compared to our previous version of object recognition considering all objects in each scan.

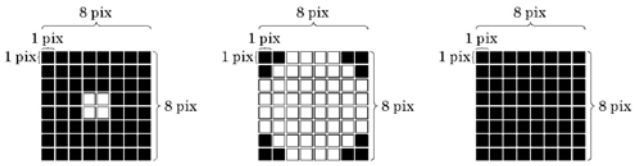


Fig. 4. Patterns of the pill, power pill and path gird.

However, since all moving objects move in a pixel level, as mentioned earlier, a measure must be taken to enable detection of them in a grid level. We do this by counting the number of object colors, in terms of the number of pixels, in a grid of interest and locating to the grid a particular object if the grid's most representative color is similar to that of the object by meeting a specific threshold. Although misdetection slightly occurs, this mechanism for detection of the moving objects reduces computation time and gives room for the decision making process described in the next section.

3 Decision Making

Our controller moves Ms. Pac-Man over the path with the lowest cost by sending the mouse-click command directing to the next corner or cross point. To ensure Ms. Pac-Man makes a turn of its direction properly in time, the mouse-click command with the direction to the second next corner or cross point will also be issued in advance if necessary. We adopted two variants of the A* algorithm [3], versions one and two, to find the lowest-cost path between Ms. Pac-Man and the target location, where the Manhattan distance is used and represented in the grid unit.

3.1 Rules

At every decision-making iteration, one of the following seven rules (in decreasing priority) is fired.

#Rule 1:
If distance(nearest_power_pill) ≦ 5(3)
AND
distance(nearest_ghost) ≧ 4(4)
AND
distance(ghost_nearest_to_the_nearest_power_pill) ≧ 6(4),
then *stop moving and ambush at the corner or cross point near the nearest power pill waiting for a ghost to come closer*,
where distance(nearest_power_pill) is the distance from Ms. Pac-Man to the nearest power pill, distance(nearest_ghost) the distance from Ms. Pac-Man to the nearest ghost, and distance(ghost_nearest_to_the_nearest_power_pill) the distance from Ms. Pac-Man to the ghost nearest to the power pill nearest to Ms. Pac-Man, and the numbers in the parentheses are those for the second stage of the game.

#Rule 2:
If at least one power pill exists
AND
distance(nearest_ghost) ≦ 8
AND
distance(nearest_power_pill) ≦ 6
AND
distance(nearest_power_pill) ≦
distance(ghost_nearest_to_the_nearest_power_pill),
then *move to the nearest power pill with the A* version two.*

#Rule 3:
If at least one power pill exists
AND
distance(nearest_ghost) ≦ 8
AND
distance(nearest_power_pill) ≦
distance(ghost_nearest_to_the_nearest_power_pill),
then *move to the nearest power pill with the A* version one.*

#Rule 4:
If at least one edible ghost exists
AND
distance(nearest_ghost) ≦ 8
AND
distance(nearest_edible_ghost) ≦ 8,
then *move to the nearest edible ghost with the A* version one,*
where distance(nearest_edible_ghost) is the distance from Ms. Pac-Man to the nearest edible ghost.

#Rule 5:
If distance(nearest_ghost) ≦ 8,
then *move to the nearest pill with the A* version one.*

#Rule 6:
If at least one edible ghost exists
AND
distance(nearest_ghost) ≧ 9
AND
distance(nearest_edible_ghost) ≦ 8,
then *move to the nearest edible ghost with the A* version two.*

#Rule 7:
If distance(nearest_ghost) ≧ 9,
then *move to the nearest pill with the A* version two.*

Figure 5 shows screenshots where Ms. Pac-Man ambushes ghosts at the left-bottom corner according to Rule 1 (a), moves to the nearest power pill according to Rule 2 (b), and moves to the nearest ghost according to Rule 6 (c) (d). A YouTube video clip of ICE Pambush 2 is also available at [4]

To find a path, the A* version one considers the distance cost, the ghost cost, and the node cost while the A* version two considers only the distance cost, where cost definitions are given in the next subsection. The former version of A* is used in more critical situations, such as when a ghost is nearby, than the latter one. In addition, the depth of search is also different between these two versions. The *n* depth indicates that the search space covers all paths from the current grid of Ms. Pac-Man to the *n* th corner or cross point in the same quarter of the target location, with the constraint that the distance from Ms. Pac-Man to the *i* th corner or cross point is always larger than that of the *i*-1 th one. In particular, the search depth of the A* version one and two is 3 and 10, respectively.

3.2 Cost Definitions

Costs are defined such that Ms. Pac-Man can manage to reach the target location without being hit by a ghost. The definition of each cost is given below, and the costs below are accumulated for the corresponding corner or cross point.

#Distance Cost at point *X*:
= [distance(*X*) + distance(*X*_to_target) – distance(target)]*1000,
where *X* is the *i* th-level search corner or cross point from Ms. Pac-Man, *i* = 1 to 3 (A* version one) or 1 to 10 (A* version two), distance(*X*) is the distance from Ms. Pac-Man to *X*, distance(*X*_to_target) is the distance from *X* to the target location, and distance(target) is the distance from Ms. Pac-Man to the target location.

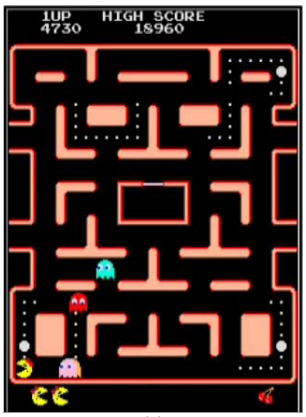
#Ghost Cost I at point *X* away from ghost *Y*:
= 500,000/[distance(*Y*_to_*X*)^2],
where *X* = the *i* th corner or cross point from ghost *Y*, *i* = 1 and 2, and distance(*Y*_to_*X*) is the distance from ghost *Y* to point *X*.

Ghost Cost II at a corner or cross point where there is a ghost on it:
= 1,000,000

#Ghost Cost III at a corner or cross point behind a ghost moving on the same straight path as Ms. Pac-Man:
= 6,000,000

Corner Cost at each corner:
= 5,000

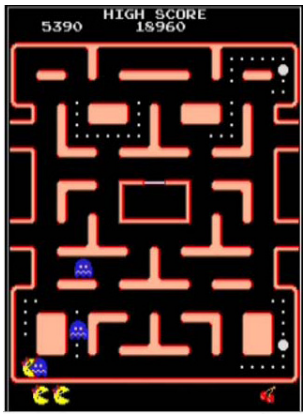
The above corner cost is for suppressing Ms. Pac-Man from moving to an area with many corners because of a high chance in being trapped by ghosts there.



(a)

(b)

(c)



(d)

Fig. 5. Ambushing of ghosts.

4 Performance Evaluation

Table 1 shows the scores that ICE Pambush and ICE Pambush 2 obtained at the IEEE WCCI 2008 competition [5] and IEEE CEC 2009 competition [2], respectively. Therein, the average and maximum scores were derived from 10 games executed at the organizer environment, a Dell XPS laptop. From the competition result in [2], ICE Pambush 2 is the winning entry among the five submitted controllers. Figure 6 shows the score histograms of ICE Pambush 2 and the second place controller, Max Chan. In addition, ICE Pambush 2 is so far holding the world records for both average and maximum scores for all three competitions [1] as shown in Table 2. Its scores are also superior to those reported in the literature [6-9].

Table 1. Comparison between ICE Ambush and ICE Ambush 2.

Version	Average Scores	Maximum Scores
ICE Ambush (IEEE WCCI 2008)	4694	6390
ICE Ambush 2 (IEEE CEC 2009)	13059	24640

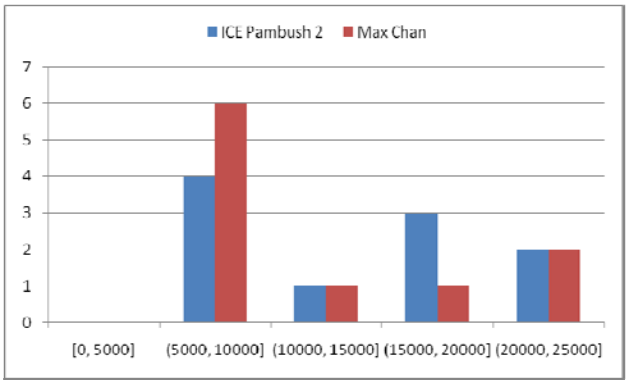


Fig. 6. Score histograms of ICE Pambush 2 and Max Chan.

Table 2. Comparison between the winning entries of three competitions.

Competition	Average Scores	Maximum Scores
IEEE CEC 2007 (Default)	2269	3810
IEEE WCCI 2008 (USM)	11167	15970
IEEE CEC 2009 (ICE Ambush 2)	13059	24640

5 Conclusions and future work

This paper described the outline of our Ms. Pac-Man controller, ICE Pambush 2, which is the winning entry of the IEEE CEC 2009 Software Agent Ms. Pac-Man Competition and the world record holder for this series of competitions. Its high performance is due to the efficient image processing mechanisms and the effective decision making mechanisms. However, compared to the world record of 921360 made in 2005 by a human player, a large number of challenging issues for improvement of the controller remain.

At the time of writing this paper, Evolutionary Strategy has been adopted in our new controller, to be submitted to the IEEE CIG 2009 competition, for optimizing a number of parameters in use such as cost constants and distance thresholds. In near future, we plan to explore learning techniques such as those in [7] and [8] as well as examine the technique in [9]. A variety of other artificial intelligence and/or computational intelligence techniques will be exploited in order to shrink the gap between the best automatic controller and the best human player of Ms. Pac-Man.

Acknowledgements

The authors would like to express our deepest gratitude to Yuna Dei for his contributions to bot development. We wish to thank also the organizer of the Software Agent Ms. Pac-Man Competition for their hard work.

References

[1] Ms. Pac-Man Competition, http://dces.essex.ac.uk/staff/sml/pacman/PacManContest.html

[2] Ms. Pac-Man Competition IEEE CEC 2009 Result, http://cswww.essex.ac.uk/staff/sml/pacman/CEC2009Results.html

[3] I. Millington. Artificial Intelligence for Games. The Morgan Kaufmann Series in Interactive 3D Technology, 2006.

[4] YouTube video clip of ICE Pambush 2, http://www.youtube.com/watch?gl=JP&hl=ja&v=nrpCVy-PPyo

[5] Ms. Pac-Man Competition IEEE WCCI 2008 Result, http://cswww.essex.ac.uk/staff/sml/pacman/WCCI2008Results.html

[6] S. M. Lucas. Evolving a neural network location evaluator to play Ms. Pac- Man. In IEEE Symposium on Computational Intelligence and. Games 2005, pp. 203-210, 2005.

[7] I. Szita and A. Lorincz. Learning to Play Using Low-Complexity Rule-Based Policies: Illustrations through Ms. Pac-Man. In the Journal of Artificial Intelligence Research, 30, pp. 659-684, 2007.

[8] H. Handa. Constitution of Ms.PacMan Player with Critical-Situation Learning Mechanism. In Fourth International Workshop on Computational Intelligence & Applications IEEE SMC Hiroshima Chapter (IWCIA 2008), pp. 48-53, 2008.

[9] N. Wirth and M. Gallagher. An Influence Map Model for Playing Ms. Pac-Man. In IEEE Symposium on Computational Intelligence and Games (CIG'08), pp. 228-233, 2008.

[10] Ms. Pac-Man world record broken (2005), http://uk.gamespot.com/news/2005/08/11/news_6130815.html?sid=6130815