

ICE Pambush 2

Hiroshi Matsumoto, Chota Tokuyama, and Ruck Thawonmas
Intelligent Computer Entertainment Laboratory
Department of Human and Computer Intelligent, Ritsumeikan University
ruck [at] ci.ritsumeii.ac.jp

1. Introduction

This is our Ms Pac-man software controller in Java for the IEEE CEC 2009 contest. Its name is ICE Pambush 2. We developed it based on, but significantly different from, the sample controller (pac.zip) available from the contest site.

ICE Pambush 2 uses a much lighter original image-processing mechanism, than the one in the aforementioned sample controller, to extract all relevant

objects: Ghosts, Pills, Power-Pills, and Ms PacMan. As a result, this leads to a drastic increase in performance compared to our previous controllers ICE Pescape and ICE Pambush for IEEE WCCI 2008. Tested with [the web version of Ms Pac-Man using Internet Explorer](http://www.webpacman.com), ICE Pambush 2 scored in average more than 10,000 and achieved a maximum score more than 20,000. A demo video clip of ICE Pambush 2 is available at the site below:

<http://jp.youtube.com/watch?v=nrpCVy-PPyo>

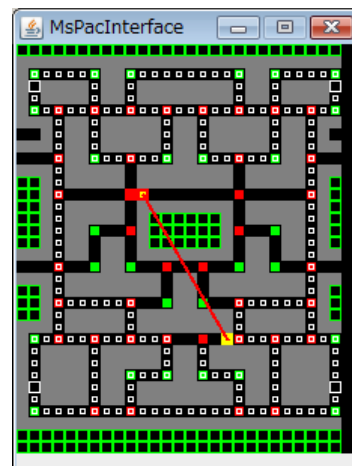
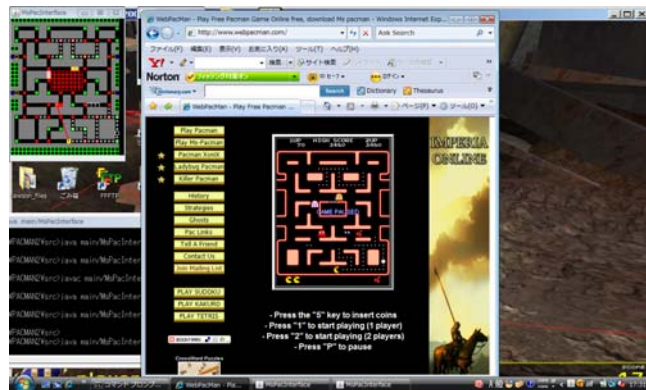
Hope that similar performance will be achieved at the contest!!

The left figure shows one representation of the game. This window, titled MsPacInterface, usually appears on the top left of the display. There, all pills are in white boxes and power pills in larger white boxes. Each corner and cross point are superimposed by green and red, respectively. A ghost is represented by its own color, and our Ms PacMan is represented by yellow. The red line points from Ms PacMan to the nearest ghost.

Our Ms PacMan moves along the black paths towards the target location. To exploit warp points, the game map used in our controller is an extended version of the one shown in the MsPacInterface window. This map concatenates the right half + the whole map + the left half. With this map, our Ms Pacman will go through a warp point if the target is located near the opposite warp point.

If ICE Pambush 2 performs well at the contest, we will make our controller and screen-capture kit available at <http://www.ice.ci.ritsumeii.ac.jp/~ruck/downloads.html>. If you cannot wait, email us!

We describe below how to set up and execute ICE Pambush 2 in Sec. 2 and the controller itself in Sec. 3.



2. Set-up and execution

ICE Pambush 2 can be run by moving to /src/ and executing

```
java main/MsPacInterface.
```

If the current parameter setting is correct, the MsPacInterface window will have the content similar to that shown in the previous figure, where all objects are shown in the colors described above; our controller should easily earn more than 3000 scores.

According to our experience, there are mainly two problems that might occur, causing the score to be very low, say, below 3000. The first one is related to the position of the game window, and the second one is related to color settings.

If one (or some) of the left top, right top, left bottom, or right bottom corners, all in green boxes, is (are) missing in the MsPacInterface window, the first problem occurs. This problem can be solved by first pausing the game and then adjusting the game window such that all those four corners appear in the MsPacInterface window and that there is one row of black boxes surrounding by green on the top and two rows of such boxes on the bottom as shown in the MsPacInterface figure.

If, during the play, Ms Pacman moves strangely or the controller still scores less than 3000, the second problem occurs. This might require more time to fix, compared to the first one. For this, the corresponding color related parameters in **MsPacInterface.java** (in /src/main/) must be changed accordingly. Below are our hints on which parameters to try first!

Currently, all color related parameters used at our environment during development (for [the web version of Ms Pac-Man using Internet Explorer](#)) are as follows:

```
blinky = -65536; //blinky
pinky = -18210; //pinky = -18689 in the sample controller
inky = -16711714; //inky=-16711681 in the sample
sue = -18361; // sue = -18859 in the sample
pacMan = -256;
edible = -14605858; //edible ghost = -14408449 in the sample
pill1 = -2171170; //pill at the first stage = -2434305 in the sample
pill2 = -256; //pill color at the second stage
pill3 = 0; //pill color at stage 3 (not used)
pill4 = 0; //pill color at stage 4 (not used)
back = -16777216; // color for the background
```

Please try first to modify **those parameters in red**. After modifying those parameters, please recompile (JDK 1.6) the program by

```
javac main/MsPacInterface.java
```

Hope this helps!

3. Controller

Our controller moves Ms PacMan based on path costs. We adopted two variants of the A* algorithm to find the best path, the one with the lowest cost, between Ms PacMan and the target location. The Manhattan distance is used in our search algorithm. At each iteration, one of the following six rules (in decreasing priority) is fired.

```
#Rule 1: If distance(Ms_Pacman_to_nearest_power_pill) ≤ 5(3) AND
distance(MS_Pacman_to_the_nearest_ghost) ≥ 4(4) AND
distance(MS_Pacman_to_ghost_nearest_to_the_nearest_power_pill) ≥ 6(4),
then stop moving to ambush near the nearest power pill for a ghost to come closer,
```

where $\text{distance}(\text{Ms_Pacman_to_the_nearest_power_pill})$ is the distance from Ms Pacman to the nearest power pill, $\text{distance}(\text{MS_Pacman_to_the_nearest_ghost})$ is the distance from Ms Pacman to the nearest ghost, and $\text{distance}(\text{MS_Pacman_to_ghost_nearest_to_the_nearest_power_pill})$ is the distance from Ms Pacman to the ghost nearest to the power pill nearest to Ms Pacman, and the numbers in the parentheses are those for the second stage.

#Rule 2: If $\text{distance}(\text{Ms_Pacman_to_the_nearest_power_pill}) \leq \text{distance}(\text{MS_Pacman_to_ghost_nearest_to_the_nearest_power_pill})$, then move to the nearest power pill with the A* version one if $\text{distance}(\text{Ms_Pacman_to_nearest_power_pill}) \geq 6$; otherwise, with the A* version two.

#Rule 3: If the distances from Ms Pacman to the nearest ghost and to the nearest edible ghost are less than or equal to 8, then move to the nearest edible ghost.

#Rule 4: If the distance from Ms Pacman to the nearest ghost is less than or equal to 8, then move to the nearest pill.

#Rule 5: If the distance from Ms Pacman to the nearest ghost is more than or equal to 9 and that to the nearest edible ghost is less than or equal to 8, then move to the nearest edible ghost.

#Rule 6: If the distance from Ms Pacman to the nearest ghost is more than or equal to 9, then move to the nearest pill.

For rules 3 and 4, the A* version one is used; and for rules 5 and 6, the A* version two is used. In the former version, the 3-depth distance cost, 2-depth ghost cost, corner cost, and power pill cost are used. In the latter version, only the 10-depth distance cost is used. Cost is accumulated to each cross point or corner. The definition of each cost is as follows:

#Distance cost at location X

$$= [\text{distance}(\text{Ms_Pacman_to_X}) + \text{distance}(\text{X_to_target}) - \text{distance}(\text{Ms_Pacman_to_target})] * 1000,$$

where X is the i th point (either a cross point or a corner) from Ms Pacman, $i = 1$ to 3 or 1 to 10 (depending on the maximum search depth), $\text{distance}(\text{Ms_Pacman_to_X})$ is the distance from Ms Pacman to X, $\text{distance}(\text{X_to_target})$ is the distance from X to the target, and $\text{distance}(\text{Ms_Pacman_to_target})$ is the distance from Ms Pacman to the target.

#Ghost cost at location X away from ghosts

$$= 500,000 / [\text{Manhattan distance}(\text{ghost_to_X})^2],$$

where X = the i th point (either a cross point or a corner) from a ghost of interest, and $i = 1, 2$.

Ghost cost at location X with a ghost on it

$$= 1,000,000$$

#Cost at a corner X

$$= 5,000$$

This cost was introduced to reduce a chance to get trapped by ghosts.

#Power pill cost at corners or cross points nearest to a power pill when the distance from Ms Pacman to the nearest ghost is beyond 9.

$$= 100,000$$

This cost was introduced to reduce a chance to wastefully obtain a power pill when a ghost is still far away.

4. Future work

As future work, we plan to further improve the image-processing part and to introduce supervised and/or non-supervised learning mechanisms for training Ms Pacman in some critical situations. Part of the training data will be from the first member whose maximum score is above 70,000.