# CS7032: AI & Agents: Ms Pac-Man vs Ghost League - AI controller project

TIMOTHY COSTIGAN 12263056
Trinity College Dublin

This report discusses various approaches to implementing an AI for the Ms Pac-Man vs Ghosts league. It implements a purely-reactive subsumption based agent to control Ms Pac-Man which consists of three modules : evade, hunt and gather which are arranged by priority. The behaviour of the agent can be adjusted by altering its *hunt distance* and *evade distance* parameters to determine when to chase, evade or ignore ghosts. The performance of the agent was evaluated across a range of parameter values for 100 trials at each point and its ideal average score was found to be at around : hunt distance = 75 and evade distance = 5. The results of the report suggest that a risk taking strategy is good for a reactive agent although alternative methods such as reinforcement learning or finite state machines may be better.

## 1. INTRODUCTION

In the early days of computer science, artificial intelligence was purely in the academic domain and even as computer games came on the scene in the 1970s and 80s, game AI was only an afterthought and could be very primitive indeed [Haahr 2010]. As computer technology has become more powerful and games more prolific however, much more complicated and effective AI techniques have crossed into the game domain [Haahr 2010]. It is with the above in mind, that this report discusses and implements an AI technique for a video game. Bizarrely, the game being used is one from the early days of computer games : Ms Pac-Man.

## 2. THE PROBLEM

### 2.1 The competition

For this project, we were faced with the challenge of designing and implementing a possible submission for the Ms Pac-Man vs Ghosts League. The competition was established by Philipp Rohlfshagen, David Robles and Simon Lucas of the University of Essex and has been running since 2011 [Philipp Rohlfshagen and Lucas 2012]. The goal of the competition is to implement an AI controller for either Ms Pac-Man, the team of four ghosts or both [Philipp Rohlfshagen and Lucas 2012].

DOI:http://dx.doi.org/0

The various submissions from the many competitors are then pitted against each other and judged based on the highest total average score in the case of Ms Pac-Man and the lowest average score in the case of the Ghosts [Philipp Rohlfshagen and Lucas 2012].

### 2.2 The rules

The rules for the league can be broken up into two categories, the competition rules which are those restricting the AI controller implementation and any failure to observe them will result in disqualification and the game rules which are enforced by the competition framework and determine the behaviour of the game world for example how points are awarded, levels progressed etc. [Philipp Rohlfshagen and Lucas 2012].

The competition rules are [Philipp Rohlfshagen and Lucas 2012]:

—AI controllers must finish initialisation within 5 seconds.
—AI controllers are restricted to a 512MB memory footprint.
—AI controllers must reside on a single thread.
—Files may only be read from or written to if they are in the controller's directory, are accessed only by the provided IO class and do not exceed 10MB.
—Levels last for 3000 ticks of 40ms with the game advancing to the next level when time runs out. In such an event, the score that would have been awarded from the remaining pills is halved and given to the controller.
—Each game can consist of a maximum of 16 levels.
—Ghosts cannot reverse.

The game rules are [Philipp Rohlfshagen and Lucas 2012]:

—Ms Pac-Man begins the game with three lives which are deducted whenever she is caught by a ghost. Additional lives can be gained through the collection of 10,000 points and if all lives are depleted the game ends.
—The game contains four mazes which are traversed in order until the game is complete or over. These mazes differ in terms of layout and pill placement.
—Pills give 10 points, power pills 50 points and ghosts (if edible) give initially 200 points with this amount doubling for each additional ghost.
—Ghosts become edible (and reverse direction) whenever Ms Pac-Man consumes a power pill. The time the ghost remains edible decreases with each level and if another power pill is eaten during this period, the score multiplier is reset.
—If Ms Pac-Man loses a life, the ghosts are reset and she spawns in her initial position.
—Once all pills are consumed or the time limit is up, the game progresses to the next level.

To be considered for entry to the competition, an AI controller must conform to these above rules.

### 2.3 Goal

This report implements just the Ms Pac-Man controller and not the ghosts controller. This is because traditionally the Pac-Man character is the one controlled by the player and as such it feels more natural to try and maximise its performance rather than hinder it.

## 3. POSSIBLE APPROACHES

In formulating a design for the controller, a number of different approaches were considered.

### 3.1 Learning methods

The first methods considered were those utilising some form of learning such as supervised learning or reinforcement learning techniques.

Supervised techniques were considered where the AI would be trained against other opposing AIs using some form of annotated training data but this approach was dropped as there was no way of knowing which AIs our controller would face [Luz 2012; Philipp Rohlfshagen and Lucas 2012]. As the behaviour of the ghosts could not be relied upon, the AI would not operate under the inductive learning assumption and as such techniques like supervised learning would not be ideal [Luz 2012].

Unsupervised reinforcement learning methods such as dynamic programming and temporal difference methods were then considered. These methods showed particular promise for the following reasons [Luz 2012]:

—Training data could be obtained automatically from direct interaction with the game.
—A "clearly defined adversary" is not necessary.
—They work well in environments such as Ms Pac-Man's where the search space could be quite large.

It was for these above reasons that reinforcement learning methods were the first to be seriously considered.

Temporal difference learning was given particular attention as it does not require a model of the environment [Luz 2012]. Temporal difference learning instead of storing a large database of game states and their appropriate responses, controls the agent through a neural network [Tesauro 1995]. The neural network is trained by exposing the agent or controller to a number of games and adjusting the weights of the network units to approximate the desired output [Tesauro 1995]. When the game is being played using this network, the state is passed into it and the network outputs an approximate action [Tesauro 1995]. This system allows a very large amount of game states to be encoded into a considerably smaller memory footprint [Tesauro 1995]. The performance of temporal difference methods has also been shown to be rather good, for example when used to implement a backgammon AI it approached the ability of some of the world's best human players and shows significant potential for surpassing them [Tesauro 1995].

However, temporal difference learning or any other learning method was not used in the final implementation for a number of reasons. Reinforcement learning was introduced relatively late in the course and there was no introductory lab material for it so when it came to implementing it for this project there were many great difficulties. Difficulties included how best to represent the state to the learning method and how to determine values, rewards etc. Finally, it could not be determined definitively if enough training could be performed or if the quality of the training would be good enough to bring the controller's performance above simpler methods. It was also not certain if many of the issues detailed in Tesauro [1992] could be resolved. Essentially, more tried and tested methods were used in the end and these will be detailed in the next section.

### 3.2 Symbolic methods

After learning methods were abandoned, symbolic methods were then considered and in particular reactive agents. Reactive agents are relatively simple in comparison with the learning approaches discussed earlier. Reactive agents are stateless and operate as a hierarchical structure of $condition \rightarrow action$ rules meaning they react with a certain action for each state without considering previous or later states [Luz 2012]. The main appeal of this style of architecture over more advanced methods is that it is easier to understand, implement and test as the rules can be quite intuitive and the effects of changes can be seen immediately.

As will become clear in section 4, a particular type of reactive agent known as a subsumption architecture was used. A subsumption architecture is produced by determining what problem we wish to solve with the agent, decomposing that problem into a set of tasks and implementing each individually as a separate layer [A.Brooks 1986]. These independent layers provide a specific piece of functionality by themselves such as path finding, enemy evasion etc and by combining them together we can get a relatively advanced agent [A.Brooks 1986]. The main benefits of this system are that specific parts of the agent's behaviour can be implemented and tested independent of another part and new behaviours can be added without any major modification [A.Brooks 1986]. Subsumption architectures also work well in systems with multiple and perhaps conflicting goals like Ms Pac-Man's, for example Ms Pac-Man's goal to avoid yet hunt ghosts [A.Brooks 1986; Luz 2012]. It is for the above reasons that a subsumption architecture reactive agent based on A.Brooks [1986] was ultimately used for this project's final implementation.

## 4. IMPLEMENTATION

Implementation of the Ms Pac-Man controller began by determining its abstract architecture. The purpose of the abstract architecture was to help provide a formal framework on which to base the final implementation [Luz 2012].

### 4.1 Agent and environment properties

In order to produce the abstract architecture it was necessary to explore the system's agent and environment properties. The PEAS system (Performance, Environment, Actuators and Sensors) was used help model the AI controller's attributes [Luz 2012]. An alternative PAGE system (Percepts, Actions, Goals and Environment) was not used as it was thought that specifying the system's actuators and sensors would be more helpful when it came to coding [Luz 2012]. The PEAS properties were:

*Agent Type* Ms Pac-Man Game AI.

*Performance measure* Maximum average score.

*Environment* Differing mazes to traverse with hostile opponents in pursuit.

*Actuators* Determine direction of travel.

*Sensors* Number of remaining pills, number of ghosts, ghost edible time and many more.

From the PEAS agent properties, we could see that there were far too many sensors so to simplify the agent, it was decided to limit the sensors to just:

—The location and distance to the nearest hostile (inedible) ghost,
—The location and distance to the nearest edible ghost and
—The location of the nearest pill or power-pill (whichever is first).

After the agent properties were determined, finally the environment properties were identified [Luz 2012]. The environment properties are:

*Environment* Differing mazes to traverse with hostile opponents in pursuit.

*Observable* Partial, view restricted to simplify implementation.

*Deterministic* Depends, opposing AI controllers could implement random behaviour.

*Episodic* Yes as no history of states is maintained.

*Static* Dynamic, the opposing ghost agents act independently of the Ms Pac-Man agent.

*Discrete* Yes, the possible actions are limited to one of four directional changes.

*Agents* Single in the case of the Ms Pac-Man agent.

## 4.2 Abstract architecture

Using the agent and environment properties alongside the sensor simplifications from section above, it was possible to produce the abstract architecture. The abstract architecture defines the rough structure of the agent as a tuple of 4 values:

$$Arch_s =< S, A, action, env >$$

where S represents all possible environment states, A represents all possible actions, action represents the agent's behaviour and env describes the environments behaviour [Luz 2012]. Using the agent properties, the possible states of the environment were found to be:

*s0* No edible ghosts.

*s1* No hostile ghosts.

*s2* Edible ghost and hostile ghost.

*s3* No pills remaining.

making $S = \{s0, s1, s2, s3\}$. The possible actions are:

*a0* Go towards nearest pill.

*a1* Go towards nearest edible ghost.

*a2* Go away from nearest hostile ghost.

making $A = \{a0, a1, a2\}$. In a standard abstract architecture, the action part of the tuple would be modelled as:

$$action : S^* \rightarrow A$$

where $S^*$ represents all sequences of states, however as it was believed that there was no need to maintain a history of states, a purely reactive agent was used instead. A purely-reactive agent is stateless and performs actions based only on the current episode and is modelled like so[Luz 2012]:

$$action : S \rightarrow A$$

Using the state-action rule format, the possible actions, states and desired responses, the following rules were formed:

—$s0 \rightarrow a0 \vee a2$ : a0 if ghost is far enough away and a2 if ghost is too close.
—$s1 \rightarrow a0 \vee a1$ : a0 if ghost is too far away and a1 if close enough.
—$s2 \rightarrow a0 \vee a1 \vee a2$ : a0 if hostile and edible ghosts are too far away, a1 if edible ghost is close enough and hostile ghost is far enough away and a2 if the hostile ghost is too close.
—$s3 \rightarrow a0$ : as no pills marks the start of the next round.

Finally, the environment can be modelled as:

$$env(s_j, a_k) = S'$$

which means that "performing an action $a_k$ on an environment whose state is $s_j$ results in a number of scenarios (S')" [Luz 2012].

## 4.3 Concrete architecture

Once the abstract architecture was completed, the final implementation could be built.

As can be seen in the list of state-action rules in the abstract architecture, the distance between Ms Pac-Man and either the nearest hostile or edible ghost is important so the implementation uses these distances (hunt distance and evade distance) as parameters to alter the agent's behaviour. As three possible actions were identified (go towards nearest pill, go towards nearest edible ghost and go away from nearest hostile ghost), it seemed natural to use the subsumption architecture and implement each action as a separate layer as in figure 5.

The responsibilities of the layers are:

*Evade Layer* This layer determines if a hostile ghost is too close to Ms Pac-Man (within *evade distance*) and if so provides a possible move to escape, otherwise no move is produced.

*Hunt Layer* This layer determines if an edible ghost is close enough to Ms Pac-Man (within *hunt distance*) and provides a possible move towards consuming it, otherwise no move is produced.

*Gather Layer* This layer provides a move towards the nearest pill, it always produces a move.

These layers are ordered by priority with evading first as losing a life reduces any chances of completing the game, hunting second as ghosts provide significantly more points than pills and gathering last. The implementation every cycle simply checks each layer in order and acts on the first move given.

## 5. EVALUATION

The performance of the AI controller was evaluated across a range of parameter values with the average score for 100 trials being used as the performance metric. The two parameters for the AI controller *hunt distance* and *evade distance* were input across a range of 0 to 95 in increments of 5. Ideally, a larger range with a smaller increment step size should have been used but the time required to do so would have been prohibitive and even the relatively modest plots produced for this project took several minutes to output.

## 5.1 Results

Using the results of the test discussed above, the graph in figure 1 was produced. The graph shows that the highest average score for the given range was around 5431 with the parameter values of 5 for evade distance and 75 for hunt distance. The full table of results can be seen in figure 3. To verify that 75 or thereabouts was the ideal hunt distance, another plot of results was produced but this time with a greater hunt distance range (0 to 195) which can be seen in figure 2. Figure 2 does indeed confirm that a distance of around 75 is ideal and that no great score difference can be observed with any higher distance value (most likely as we are reaching the width or height of the playing field).

## 5.2 Discussion

The results in figures 1,2,3 and 4 allow us to make a few observations. It would seem that as the evade distance is decreased towards around 5, the average score (with a few exceptions) seems to increase. Below 5, the score begins to decrease as at this point in the event of a pursuit, ghosts are nearly on top of the Ms Pac-Man agent like in figure 6. Based on these results it would appear that risk taking is more rewarding than playing it safe. Another observation is that as the hunting distance is increased, the average score increases which shows that it is better to opportunistically chase edible ghosts than continue gathering pills. The levelling out of the average score as the hunt distance goes beyond 75 is most likely due to the limited size of the map and it could be expected to increase further on bigger maps. Once again the above observations support this report's opinion of risk taking being a desirable trait of the Ms Pac-Man agent.

The performance of this AI seems to be respectable when compared with the average scores of other AIs in the competition, however this report's AI has only been tested using the default Ghost AI controller while those in the league have used other custom AIs and as such no direct comparison can be made [Philipp Rohlfshagen and Lucas 2012].

Without other controllers implemented using some other game AI technique, it is difficult to say how this AI would compare, however for certain methods a few differences are likely. The reactive nature of this report's AI while simplifying implementation does limit the power of the agent. The agent is incapable of learning from its mistakes unlike methods such as temporal difference learning and can rather naively walk into traps that even a small amount of look-ahead or look-back would have prevented [Tesauro 1995].

The subsumption architecture while sufficient for the purposes of this report is probably not ideal for game AIs. Subsumption architecture seems to be more ideally suited for robots where a series of robust redundant systems is desirable while in games, the state of the world can be exact and reliable [A.Brooks 1986]. A system such as finite state machines would be more commonly used to control AI in an environment like Ms Pac-Man [Haahr 2010]. Finite state machines instead of using a series of modules layered according to priority, consist of a series of states which are moved between by transition conditions [Haahr 2010]. Finite state machines have the advantage that additional states don't interfere with existing states as only one is executed at a time while in a subsumption architecture each layer operates independently and may conflict [A.Brooks 1986; Haahr 2010].

## 6. CONCLUSION

In conclusion, this report has discussed the implementation of a simple AI controller for Ms Pac-Man from concept to evaluation. This report has discussed some of the varied approaches to creating a game AI from reinforcement learning to reactive agents. It has detailed the construction of a purely reactive agent based upon the subsumption architecture from abstract architecture to final implementation and has assessed its performance across a range of parameter values. It has been shown that for a reactive Ms Pac-Man agent that risk taking is desirable and that hunting is one of the best ways to increase the average score. Finally some of the shortcomings of such an approach such as the inability to have long term goals or learn from mistakes have been highlighted and the alternative of finite state machines has been noted.

## APPENDIX

### REFERENCES

Rodney A.Brooks. 1986. A Robust Layered Control System For A Mobile Robot. *IEEE JOURNAL OF ROBOTICS AND AUTOMATION* RA-2, 1 (March 1986), 14–23.

Mads Haahr. 2010. Autonomous Agents: Introduction. (2010). Retrieved January 19, 2013 from http://www.cs.tcd.ie/Mads.Haahr/CS7056/notes/001.pdf

Mads Haahr. 2010. Autonomous Agents State-Drive Agent Design. (2010). Retrieved January 19, 2013 from http://www.cs.tcd.ie/Mads.Haahr/CS7056/notes/002.pdf

Saturnino Luz. 2012. AI, agents and games: cs7032 course reader. (2012). Trinity College.

David Robles Philipp Rohlfshagen and Simon Lucas. 2012. Ms Pac-Man vs Ghosts League. (2012). Retrieved January 13, 2013 from http://www.pacman-vs-ghosts.net/

Gerald Tesauro. 1992. Practical Issues in Temporal Difference Learning. *Machine Learning* 8 (1992), 257–277.

Gerald Tesauro. 1995. Temporal Difference Learning and TD-Gammon. *Commun. ACM* 38, 3 (March 1995), 58–68.

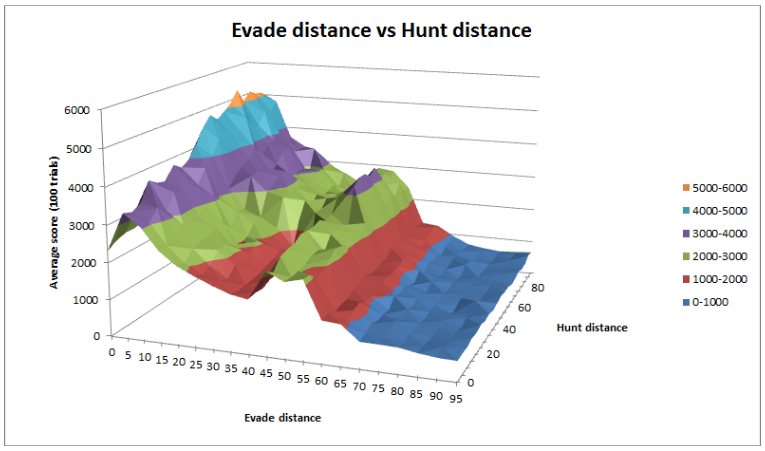

Fig. 1. Plot of the average score over 100 trials depending on the values of the hunt distance (from 0 to 95) and evade distance (from 0 to 95) parameters.

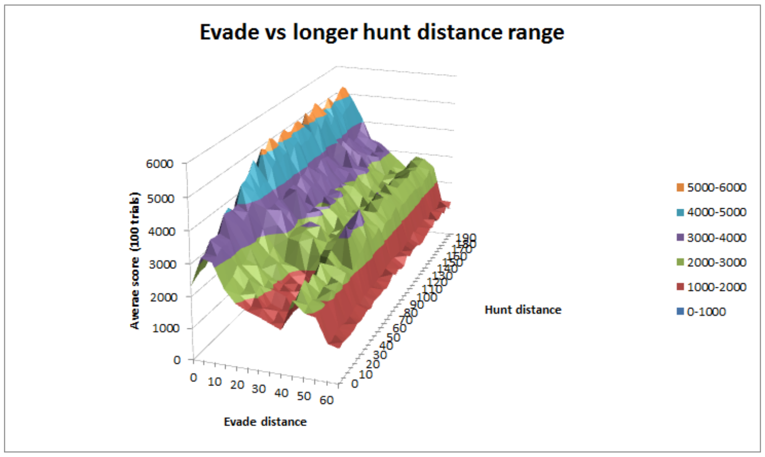

Fig. 2. Plot of the average score over 100 trials depending on the values of the hunt distance (from 0 to 195) and evade distance (from 0 to 65) parameters.

| | Evade | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Hunt | 0 | 5 | 10 | 15 | 20 | 25 | 30 | 35 | 40 | 45 | 50 | 55 | 60 | 65 | 70 | 75 | 80 | 85 | 90 | 95 |
| 0 | 2318.2 | 3368.7 | 3043.8 | 2474.5 | 2124.5 | 1905.8 | 1711.2 | 1539.2 | 1458.9 | 2245.4 | 2021 | 2141.8 | 1142.5 | 1105.1 | 715.6 | 709.2 | 694.2 | 612.4 | 559.3 | 546.7 |
| 5 | 2469.8 | 3281.3 | 3052.9 | 2476.1 | 2074.1 | 1805.4 | 1697.3 | 1473.5 | 1483.5 | 2071.3 | 2005.4 | 2115 | 1191.2 | 1028.1 | 773 | 733.5 | 640.4 | 622.2 | 665.8 | 557.9 |
| 10 | 2552.7 | 3410.2 | 3324.2 | 2562.3 | 2310 | 1893.7 | 1603.2 | 1484.8 | 1447.9 | 2346.3 | 2020.1 | 1885.1 | 1038 | 1154.4 | 858.9 | 806.6 | 694.9 | 579.7 | 590.5 | 548.9 |
| 15 | 2550 | 3901.4 | 3128.9 | 2778.3 | 2372.9 | 2195.9 | 1835.1 | 1495.4 | 1544.6 | 2413.1 | 2048.8 | 1919.8 | 1087.3 | 1191.4 | 762.2 | 818.7 | 660.9 | 587.9 | 590.5 | 610 |
| 20 | 2632.1 | 3734.1 | 3503.1 | 2785.8 | 2421 | 2223.9 | 1793.4 | 1524.8 | 1540.9 | 2424.1 | 2151.7 | 2111.9 | 1081.4 | 1061.2 | 778.5 | 698.7 | 674.7 | 618.4 | 565.5 | 575.3 |
| 25 | 2830.4 | 3669.1 | 3452.5 | 2768.6 | 2490.3 | 2059.4 | 1893 | 1597.9 | 1606.8 | 2565.1 | 2158.8 | 2271.3 | 1188.8 | 1052.4 | 736.9 | 816.5 | 614.9 | 630.5 | 529.9 | 626.7 |
| 30 | 3112.6 | 4022.5 | 3531 | 2943 | 2491.7 | 1955 | 2085.5 | 1645.7 | 1576 | 2522.1 | 2001.4 | 2195 | 1168.1 | 1051.2 | 959.7 | 720.2 | 665.4 | 637.3 | 606.4 | 572.1 |
| 35 | 3094.1 | 3843.7 | 3395.1 | 3096.8 | 2265.3 | 2050.6 | 1933.1 | 1758.9 | 1643.7 | 2513.3 | 2340.3 | 2026.3 | 1302.5 | 1197.4 | 874.8 | 808.4 | 668.9 | 500.9 | 553 | 605.3 |
| 40 | 3126.7 | 3986.3 | 3773 | 2936.4 | 2595.8 | 2286.8 | 2041.8 | 1647.5 | 1536.4 | 2570.8 | 2362.5 | 2055.6 | 1176.4 | 1180.8 | 929.8 | 799.7 | 681.4 | 591.2 | 755.6 | 539.5 |
| 45 | 3094 | 4324.5 | 4019.3 | 3096.5 | 2542.4 | 2356.4 | 2087 | 1571.6 | 1487.8 | 2430.1 | 2350.1 | 2125.6 | 1156.1 | 1134 | 822.6 | 672.6 | 695.7 | 621.4 | 570.5 | 586.9 |
| 50 | 3751 | 4723.7 | 4396.3 | 3397.1 | 3399.8 | 2984.3 | 2855.5 | 2248.6 | 2493.1 | 2494.6 | 2147.1 | 2328.6 | 1211.7 | 1148.7 | 740.4 | 831.8 | 647.9 | 625.5 | 590 | 633.7 |
| 55 | 3911.4 | 4990.6 | 4380.5 | 3534.3 | 3432.7 | 3063.5 | 2900.2 | 2689.6 | 2467 | 2853.6 | 3224.3 | 2239.4 | 1207.7 | 1208.5 | 949.8 | 811.3 | 723.7 | 644.4 | 569.4 | 600.9 |
| 60 | 4014.6 | 4786.5 | 4335 | 3651 | 3438.2 | 3237.9 | 2900 | 2442 | 2694.7 | 2782.4 | 3378.7 | 2432.4 | 1380.6 | 1221.1 | 774.2 | 703.6 | 665.7 | 636.5 | 619.4 | 602.1 |
| 65 | 3925 | 4932.6 | 4406.7 | 3874.1 | 3121.8 | 3276.3 | 2857.3 | 2646.5 | 2527.4 | 2689.5 | 3379.3 | 2463.3 | 1436.4 | 1142.4 | 888.1 | 761.1 | 601.4 | 636.1 | 586.3 | 521.3 |
| 70 | 3841.4 | 5105.1 | 4711.9 | 3451.2 | 3352.6 | 2889.2 | 2783.4 | 2513.3 | 2619.9 | 2706.2 | 3216.5 | 2406.6 | 1398.3 | 1193.8 | 833.7 | 730.3 | 814.5 | 639.3 | 630.9 | 557.6 |
| 75 | 3704 | 5431.1 | 4320.7 | 3816.5 | 3602.6 | 3047 | 2812.4 | 2524.7 | 2454.1 | 2797.9 | 3361.6 | 2586.9 | 1341.1 | 1037.6 | 871.9 | 792 | 798.9 | 595.4 | 602.1 | 622.3 |
| 80 | 3717.3 | 5020.9 | 4282.8 | 3853.6 | 3698.1 | 3046.5 | 3029.2 | 2610.5 | 2214.7 | 2831.2 | 2679 | 2407.1 | 1308.6 | 1154 | 738.7 | 772.7 | 655.1 | 603.5 | 557.8 | 555.8 |
| 85 | 3850.1 | 5241.6 | 4641.3 | 4013.7 | 3456 | 2785.7 | 2958.4 | 2456.6 | 2306.2 | 2919.6 | 2979.2 | 2745.4 | 1391.9 | 1230.4 | 851 | 768.6 | 631.3 | 564.9 | 620 | 574.6 |
| 90 | 3830.1 | 5104.6 | 4564.4 | 3509.9 | 3419.5 | 3477.9 | 2951.1 | 2760.4 | 2536.6 | 2838.9 | 2693.3 | 2366.4 | 1344 | 1297.7 | 886.9 | 717.3 | 699.2 | 586.1 | 583 | 707.5 |
| 95 | 3833.8 | 5057.4 | 4809.1 | 3697.1 | 3422.3 | 3018.4 | 2830.7 | 2605.4 | 2215.2 | 2913.2 | 2843.6 | 2369.6 | 1276 | 1224.9 | 949.6 | 730 | 649.3 | 601.2 | 628.2 | 663.6 |

Fig. 3. Table of results used to produce the graph in figure 1.

| | Evade | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Hunt | 0 | 5 | 10 | 15 | 20 | 25 | 30 | 35 | 40 | 45 | 50 | 55 | 60 | 65 |
| 0 | 2319.9 | 3249.7 | 3137.4 | 2390.7 | 1990.6 | 1809.6 | 1695.2 | 1538.8 | 1389 | 2341.4 | 2010.4 | 1933.8 | 1191.9 | 1094.4 |
| 5 | 2444.1 | 3261.8 | 3223.9 | 2464.8 | 2123.5 | 1827.6 | 1817 | 1476.9 | 1465.5 | 2220.3 | 2124.7 | 1983.2 | 1110.9 | 1144.1 |
| 10 | 2531 | 3567.7 | 3357.5 | 2526.1 | 2156.6 | 2089.4 | 1716.8 | 1645.8 | 1494.4 | 2460.3 | 2155.8 | 2036.1 | 1185.6 | 1103.9 |
| 15 | 2536.1 | 3531.7 | 3313.5 | 2743.4 | 2361 | 2205.3 | 1696.9 | 1562.6 | 1534.6 | 2584.9 | 2024.9 | 1943.5 | 1227.1 | 1082 |
| 20 | 2671.7 | 3526.1 | 3363.3 | 2629 | 2310.9 | 2026.9 | 1964.9 | 1825.9 | 1498.9 | 2337.2 | 2092.7 | 2217.4 | 1215.7 | 1092.9 |
| 25 | 2702.8 | 3980.6 | 3343.3 | 2684.2 | 2534.6 | 2138.2 | 1804.1 | 1851.9 | 1506.4 | 2517.6 | 2169 | 2058.2 | 1146.8 | 1154.5 |
| 30 | 2818.6 | 4096 | 3565.7 | 2926.4 | 2295 | 2069.2 | 1906.9 | 1727 | 1786.7 | 2337.5 | 2100.9 | 2241.2 | 1189.5 | 1096.8 |
| 35 | 2996.3 | 3920.1 | 3622.4 | 3049.3 | 2488.1 | 2390.1 | 1880.3 | 1860.8 | 1644.7 | 2619.3 | 2212.5 | 2033.6 | 1260.2 | 1130 |
| 40 | 2866.6 | 3988.7 | 3779.1 | 2861.4 | 2507.2 | 2271.2 | 1899.6 | 1585.3 | 1604 | 2315.7 | 2425.1 | 2255.6 | 1304.4 | 1179.9 |
| 45 | 3122.1 | 4496.4 | 4022.4 | 3190.1 | 2661.8 | 2209.6 | 2090.7 | 1890.9 | 1425.8 | 2535.1 | 2217.5 | 2117.7 | 1172.2 | 1082.8 |
| 50 | 3798.6 | 4555 | 4199.8 | 3702.4 | 3246.1 | 2856.6 | 2786 | 2405.4 | 2420.3 | 2658.8 | 2230.1 | 2094.4 | 1332.2 | 1202.8 |
| 55 | 3777.8 | 4717.9 | 4220 | 3654.9 | 3601.9 | 2953.3 | 2968.5 | 2841.6 | 2487.8 | 2515.3 | 3366.7 | 2226.2 | 1274.9 | 1161.5 |
| 60 | 3741.2 | 5006.3 | 4326.7 | 3668.2 | 3155.5 | 2955.8 | 2959 | 2702.6 | 2614.8 | 2652.7 | 3141.5 | 2358.1 | 1323.5 | 1142.3 |
| 65 | 3913 | 4527.8 | 4233 | 3706.1 | 3369.4 | 2874.6 | 3107.8 | 3006.3 | 2567.5 | 2625 | 3020.2 | 2426.9 | 1219.1 | 1167.7 |
| 70 | 3569.2 | 5171.5 | 4564.7 | 3580.9 | 3188.2 | 2776.8 | 2552.6 | 2391.1 | 2357.2 | 2990.1 | 3133.1 | 2664.3 | 1279.1 | 1086.4 |
| 75 | 3550.6 | 5060.5 | 4705.6 | 3745 | 3835.9 | 3111.7 | 2852.1 | 2390.8 | 2500 | 2673.4 | 3461.1 | 2387.2 | 1278.8 | 1167.5 |
| 80 | 3945.4 | 5337.6 | 4527.8 | 3758 | 3438.1 | 3061.9 | 3137.2 | 2424.8 | 2230.4 | 2698.8 | 2911.5 | 2384 | 1230.9 | 1305.7 |
| 85 | 3884.2 | 5223.9 | 4729.2 | 3777.1 | 3097 | 3111.9 | 2752.6 | 2426.2 | 2508.1 | 2762.3 | 2789.4 | 2414.9 | 1325.1 | 1148.6 |
| 90 | 3735 | 4905.1 | 4715.7 | 3647.2 | 3394.8 | 2884.2 | 3060.6 | 2762.1 | 2481.4 | 3053 | 2802 | 2566.3 | 1352 | 1231.4 |
| 95 | 3969.1 | 5187.4 | 4409.3 | 3865.9 | 3549 | 2841.3 | 3000.5 | 2680.2 | 2440.1 | 2950.1 | 2600.7 | 2247.7 | 1360.8 | 1107 |
| 100 | 3966.3 | 5281.3 | 4747.3 | 3846.3 | 3540.1 | 3150.8 | 3068 | 2611.4 | 2174.1 | 2755.1 | 2795.8 | 2433.6 | 1345.1 | 1086.9 |
| 105 | 3782.7 | 5089.7 | 4823.8 | 3998.8 | 3490.4 | 3061.7 | 2943.6 | 2585.2 | 2350.6 | 2633.9 | 2730.6 | 2451.8 | 1326.4 | 1323.7 |
| 110 | 3581.4 | 5054.3 | 4568.9 | 3722.5 | 3676.4 | 3208.5 | 3156.3 | 2418.7 | 2455 | 2935.5 | 2865.2 | 2393.3 | 1310 | 1176.3 |
| 115 | 3914.8 | 5240.4 | 4641.9 | 3741.9 | 3280.5 | 2938.4 | 2848.9 | 2452.2 | 2327.3 | 2801.2 | 2996.7 | 2523.7 | 1201.5 | 1260.4 |
| 120 | 3648.4 | 5229.7 | 4574 | 3682 | 3394.1 | 2928.5 | 2871.7 | 2389.9 | 2307 | 3027.9 | 2757.3 | 2606.1 | 1279.1 | 1264.9 |
| 125 | 3745.4 | 4882.2 | 4764 | 3864.5 | 3496.6 | 3083.5 | 2709.7 | 2259.4 | 2438.1 | 2966.9 | 2869.2 | 2275.6 | 1309.8 | 1216.1 |
| 130 | 3765.1 | 5343.7 | 4515.3 | 3840.1 | 3337.2 | 2999.5 | 2919.9 | 2603.9 | 2416.1 | 2592.5 | 2778 | 2456.2 | 1284.5 | 1106.5 |
| 135 | 3768.6 | 5103 | 4876.2 | 3803 | 3434.3 | 3035.8 | 2979.1 | 2476.5 | 2264.8 | 3013.7 | 2704.9 | 2413.9 | 1439.5 | 1090.1 |
| 140 | 3625.2 | 5236.5 | 4624.5 | 3427 | 3363.4 | 3199 | 2921.3 | 2355.8 | 2398.4 | 2790.8 | 2819.6 | 2428.6 | 1335.8 | 1165.2 |
| 145 | 3591.9 | 5461.5 | 4934 | 3566.2 | 3270.1 | 3245.8 | 2771.7 | 2547.1 | 2401 | 2829.6 | 2832.5 | 2485.2 | 1373.5 | 1181.2 |
| 150 | 3824.8 | 5158.2 | 4974.9 | 4060.1 | 3789.5 | 3248.3 | 3076.6 | 2701.5 | 2388.4 | 2912.4 | 2846.3 | 2455.9 | 1331.7 | 1248.1 |
| 155 | 3546.2 | 5037.2 | 4429.7 | 3528.4 | 3273.9 | 3089.5 | 3096 | 2191.7 | 2215.5 | 2680.8 | 2871.7 | 2503.6 | 1430.5 | 1192.9 |
| 160 | 3780.9 | 5041.4 | 4724.6 | 3896.9 | 3619 | 3392.6 | 2785.7 | 2431.2 | 2454.7 | 2631.9 | 2794.1 | 2407 | 1262.5 | 1362.2 |
| 165 | 3727.5 | 5320.9 | 4608.9 | 3673.1 | 3514.2 | 2998.2 | 2830.9 | 2465.1 | 2247.8 | 2722.6 | 2948.1 | 2385.6 | 1132.8 | 1205 |
| 170 | 3841.5 | 5153.2 | 4448.3 | 3848.2 | 3491.5 | 3258.8 | 2768.3 | 2570.6 | 2504.6 | 2665.9 | 2834.8 | 2288.6 | 1399.5 | 1215.6 |
| 175 | 3860.5 | 4974.2 | 4550.5 | 3515.1 | 3441.9 | 3064 | 3120.2 | 2446.8 | 2579.6 | 2786.1 | 2759.1 | 2431.5 | 1386.3 | 1303.9 |
| 180 | 3719.8 | 5083.9 | 4648 | 3529.9 | 3293.5 | 3169.9 | 3060.5 | 2582 | 2326.9 | 2676.9 | 2813.9 | 2437.6 | 1259.4 | 1147.2 |
| 185 | 3725.5 | 5276 | 4679.5 | 3594.3 | 3291.7 | 3091 | 3133.8 | 2591.7 | 2155.7 | 2907.8 | 2853.1 | 2620.4 | 1216.1 | 1318.4 |
| 190 | 3850.5 | 5339.5 | 4852.8 | 3765.7 | 3494.6 | 3274.4 | 3157 | 2646.5 | 2427.2 | 2841.3 | 2877.6 | 2270.8 | 1383.5 | 1109.8 |
| 195 | 3922.3 | 5227 | 4718.7 | 4094.6 | 3353.3 | 3253.8 | 2766.3 | 2687.9 | 2398.5 | 2879.5 | 2797.6 | 2530.8 | 1288.4 | 1270.3 |

Fig. 4. Table of results used to produce the graph in figure 2.

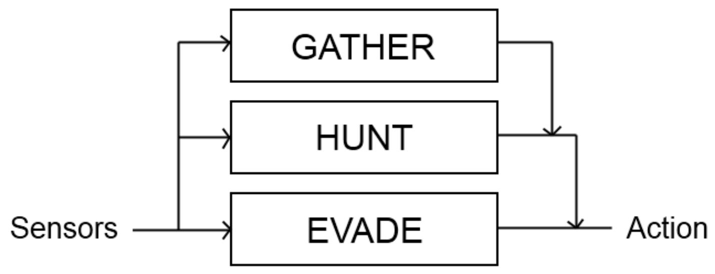

Fig. 5. A simple example of the AI controller's layered architecture.

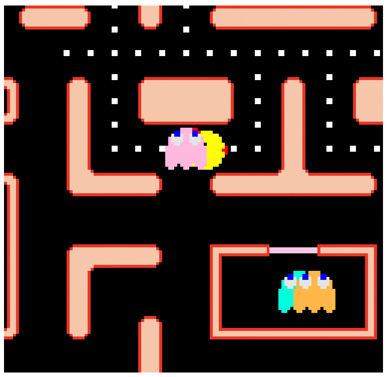

Fig. 6. An example of this project's AI controller's risk taking behaviour.