

An Evaluation of the Benefits of Look-Ahead in Pac-Man

Thomas Thompson, Lewis McMillan, John Levine and Alastair Andrew

Abstract—The immensely popular video game Pac-Man has challenged players for nearly 30 years, with the very best human competitors striking a highly honed balance between the games two key factors; the ‘chomping’ of pills (or pac-dots) throughout the level whilst avoiding the ghosts that haunt the maze trying to capture the titular hero. We believe that in order to achieve this it is important for an agent to plan-ahead in creating paths in the maze while utilising a reactive control to escape the clutches of the ghosts. In this paper we evaluate the effectiveness of such a look-ahead against greedy and random behaviours. Results indicate that a competent agent, on par with novice human players can be constructed using a simple framework.

I. INTRODUCTION

Pac-Man provides a point in history where video games moved into new territory; with players having their fill of the immensely popular space adventure genre with titles such as *Galaxian/Galaga*, *Asteroids* and the classic *Space Invaders*, Pac-Man brought players down to earth and more specifically trapped them in a maze. Historically this caused a shift in the focus of gaming developers, as the concept of maze-based video games became an overnight sensation, with many clones and bootlegs rapidly emerging in the market. Unbeknownst to many this style of game promoted not just a change in games that are played, but the thought processes required to play these games. The physical tools themselves had not changed, but the mental models required to process these tasks were dramatically different.

If we take *Space Invaders* as an example, we are presented the task of eliminating enemy ships whilst not succumbing to overwhelming numbers and firepower. The game required players to repeatedly mash the fire button whilst reactively moving away from any enemy fire, or towards enemies which were approaching the bottom of the screen. In comparison, Pac-Man presented little similarities, with the player focussing on overall strategy formulation; the difficult task on navigating throughout large mazes of varying shapes whilst avoiding the enemy ghosts provided something novel for it’s time.

While Pac-Man may be considered trivial in comparison to modern day video games in terms of scope and control, the challenge provided is still relevant today to humans and machines. To this day contenders still attempt to reach and (unlikely) break the world Pac-Man record (3,333,360 points held by Billy Mitchell) meanwhile academics seek to try their hand in developing agents capable of playing the game. Several attempts in developing agents has focussed on the use of training algorithms to develop agents using machine learning practices [1], [2], [3], while this is a noble attempt

we feel that such practices are best applied in small scope problems. While this could most certainly apply to ghost avoidance strategies, the ability to look ahead into the game world and begin to plan paths through the maze is often ignored. We consider the best means to attack the Pac-Man problem is to view at varying levels of reasoning; from high level strategy formulation to low level reactive control.

In our initial work in this domain, we sought to assess our hypothesis that the benefits of simple lookahead in Pac-man can be applied to generate a more competent agent than those using greedy or random decision processes. In this paper we give a recap of the Pac-Man domain and the particulars of our implementation in Section II and related work inSection III. The design for the agent architecture used in these experiments is provided inSection IV and the experiment breakdown and results are found inSection V. We discuss our findings in Section VI aswell as future directions for our research and give our concludingthoughts in Section VII

II. PAC-MAN

Pac-Man is an arcade game developed by Namco and released in 1980 (under the original title *Puck-Man* in Japan) with subsequent localisation and redistribution in America by 1981. Pac-Man was one of the most influential video games of the 1980’s by breaking away from the conventions of successful games such as *Asteroids* and *Space Invaders* and is to this day one of the most popular video games of all time [4].

The player is given charge of the titular character, with the task of consuming the ‘pills’ throughout the maze. To stop the player are four ghosts; commonly known as *Blinky* (red), *Inky* (cyan), *Pinky* (pink) and *Clyde* (orange). The ghosts are charged with hunting down the player, with contact between the Pac-Man and any ghost resulting in the loss of one of 3 lives. The only defence against the ghosts is the use of a ‘power pill’, which gives Pac-Man the temporary ability to consume the ghosts (encouraged by the significant bonus this adds to the players score). During this time the ghosts switch from aggressive to evasive in nature and avoid the Pac-Man as best as possible until the effects of the power pills consumption have faded. Only 4 power pills are provided in a given level, typically in the far corners. Furthermore, bonus items or ‘fruits’ appear in the centre of the level regularly throughout the game. These items also provide a large score bonus for the player.

There are a few changes that have been made in our own simulation of the Pac-Man game shown in Figure 1. While our version replicated the majority of the features of the original game, on both a functional and cosmetic level, we

Strathclyde Planning Group, University of Strathclyde, Glasgow, G1 1XH, UK, email: (forename.surname@cis.strath.ac.uk).



Fig. 1. A screenshot of our implementation of the Pac-Man game. Note the series of nodes and connecting edges that are superimposed onto the maze. These represent our graph model that is used for navigation as described in Section IV.

have made several major and minor modifications to enhance and improve the domain. A list of the notable inclusions, omissions and maintained features are given below.

- Firstly, our ghosts differ from those in the original given that they are non-deterministic in nature. In the original Pac-Man game, the agents are strictly deterministic which allows for players to learn means of avoiding them in play. To combat this, we have added a stochastic element to their behaviour with a relatively low probability. While the ghosts operate as faithfully to the original ghosts as possible (with each ghost operating under a unique strategy), there is now a small probability that at a given intersection the ghost will make a random movement, thus breaking the deterministic nature of the player without absolving the agent of it's original nature. The actual behaviour of each ghost is a hot topic in game programmer forums, despite disagreement over particular facets of their behaviour. The general consensus suggests unique behaviours for each ghost as described below:
 - *Blinky (Red)*: Blinky always tries to shorten the horizontal or vertical distance between himself and Pacman. The axis chosen is dictated by whichever is the longest.
 - *Inky (Blue)*: Inky mimics the behaviour of Blinky when in close proximity of the Blinky agent. Otherwise it will try and follow the Pacman by moving towards his last known location.
 - *Pinky (Pink)*: Pinky will attempt to intersect the Pacman by moving towards the junction Pacman

is heading towards. However should Pacman be in a tunnel, be more than a third of the game board away or has simply stopped moving, Pinky will act like Blinky.

- *Sue (Orange)*: Sue isolates which quadrant of the game field Pacman is in and will head towards the power pill in that quadrant.

- Bonus items (fruits) have been removed from our simulation. Given that we are interested in the agent's ability to traverse the maze and avoid ghosts as effectively as possible, we felt that the inclusion of the bonus fruits provided little contribution to our research interests.
- In accordance with the original game, the ghosts move slightly faster than the Pac-Man, with Blinky increasing in speed after a certain number of pills are consumed.
- Our ghosts do not slow down in the tunnels.
- The 2 maps we provide for the player are taken from Ms. Pac-Man (the sequel to Pac-Man) and not the original (there is no scientific reason for this design choice).
- The nest for the ghosts still exists in the centre of the arena, we mimic the original games behaviour such that all consumed ghosts return to the nest and wait for a small period of time before being reinserted into the game arena.
- No additional lives are attainable after certain score markers, while this would be provide a more accurate reproduction of the game it does not contribute to the overall behaviours the agents generate.
- The scoring system for consumption of ghosts is replicated; with the first ghost worth 200 points, the second 400, the third at 800 points and the last ghost generating 1600 points for the player.

Despite the minor modifications to the game, we are pleased in knowing we operate under one of the more accurate interpretations of the game by refraining from reducing the problem as often occurs in Pac-Man research (see Section III for more info). However our efforts could benefit from greater clarification as to which game we are playing, given we use (modified) Pac-Man ghost behaviours on Ms. Pac-Man maps.

The immediate challenge for players to overcome is to focus on overall goal achievement (the consumption of all pills on the screen) while ensuring all ghosts in the game world are avoided and possibly consumed when in an energised state. To achieve this it requires the player to consider both local and distant goals and develop mental models that allow

for the player to deal with immediate threats as well as plan movements for future consumption of pills on other parts of the maze. This is not to suggest that Pac-Man is merely a two tier problem, but instead promotes the notion that it is not a simple problem to solve using only one form of control. While this work is preliminary, we hope it provides sufficient evidence for us to expand into further work on Pac-Man as time progresses.

III. RELATED WORK

One of the earliest works involving Pac-Man was conducted by Koza as means to assess the effectiveness of genetic programming for task prioritisation [1]. This work utilised a modified version of the Pac-Man domain, with a maze that replicated the first level of Ms. Pac-Man. The ghost behaviours did not mimic those of the original game and all 4 ghosts used the same strategy (unlike the original game where each ghost operates on its own unique behaviour pattern). Using a series of predefined control primitives for perception and action control, agents could generate solutions to solve certain mazes (solutions were structure dependant) at human novice level.

Kalyanpur and Simon [5] applied a genetic algorithm to try to improve ghost strategies in a domain similar to the Pac-Man game. Solutions produced were also considered to be a list of directions for ghosts to traverse. A neural network is used to determine suitable crossover and mutation rates based on experimental data from the game world.

Work conducted by Gallagher and Ryan in [6] and more recently in [2] by Gallagher and Ledwich, show two different approaches to generating Pac-Man players. The former applies a set of hand-coded rules for gameplay in a reduced version of the Pac-Man game (with no power pills and one deterministic ghost), with specific parameters of the rules trained using an evolutionary algorithm. The learning process was successful, with the authors contemplating the benefits and drawbacks of the approach. One interesting observation is that the agent can consider only localised information [6]; an important factor in our observations of Pac-Man work to date. The latter publication attempts a more low level approach, by using the same reduced domain the agent applies neuroevolution. With a minimal amount of information on the game rules and the game state and a relatively simple fitness function, the agent learned basic playing ability.

Finally, work by Lucas in [3] proposed the concept of neural network move evaluators in an implementation of Ms. Pac-Man, with the decision to focus on the sequel due to the pseudo-random behaviour of the ghosts, providing a challenge to an agent by eliminating possibility of deterministic evaluations. Utilising a feature vector composed of handcrafted data (such as 'shortest path distance to nearest pill', to 'distance to nearest junction') agents achieved good performance with agents performing better against the deterministic of the game compared to the non-deterministic version.

IV. AGENT DESIGN

The behaviour of our Pac-Man agent is facilitated by using two simple components; a knowledge base and a graph model of the game world. The former is a series of rules that allows for the agent to generate quick, real-time decisions for ghost avoidance and hunting. Utilising data from the game world that allows us to understand the threat a ghost presents or the best pill to eat given the position of the agent. Meanwhile the latter provides us with a representation of the maze allowing for a range of different strategies to be deployed in consuming all the pills in the world. The overall agent control is achieved through the use of a Finite State Machine (FSM) as shown in Figure 2. The FSM dictates agent behaviours and decisions at a given time-point based on it's current state. The agent has three states:

- *Normal*: In this state the agent operates according to traditional parameters. Focussing on the consumption of pills depending on the strategy deployed (described below). In this state ghosts are acknowledged but otherwise considered harmless until they are within range. Once they are within a specified range of the agent, then the agent moves into the 'Ghost is Close' state. The range is dictated by a tile representation across the game map, with any ghost within 3 tiles of the agent deemed a threat (see Figure 3).
- *Ghost is Close*: In this state the agent reacts to the nearby ghost that has triggered the current state. The agent employs ghost avoidance or hunting strategies depending on whether a power pill has been consumed. Only once all ghosts are out with the vicinity of the agent shall a state transition occur, with the transition dependant on the whether the agent is energised or not.
- *Energised*: Given that the Pac-man has now consumed a power pill, we consider the ghosts as the primary target and switch to this state. With the agent hunting the closest ghosts provided they are within sufficient range. Should the ghosts be too far away for the agent to consume ghosts given the limited time of the energised state, the Pac-Man will continue to consume pills as normal. As soon as the agent is no longer energised it will transfer to the appropriate state based on current in-world data.

For the purposes of navigation, the maze is represented as a graph structure with nodes representing intersections in the maze, with any available paths from these intersections represented as a weighted edge. The topology if this graph can be see as nodes and connectors superimposed on the maze in Figure 1. The edge of the weight is considered as the number of pills lying upon it. At each node in the graph the agent makes a decision based on the strategy employed. We have 3 strategies that agent can apply:

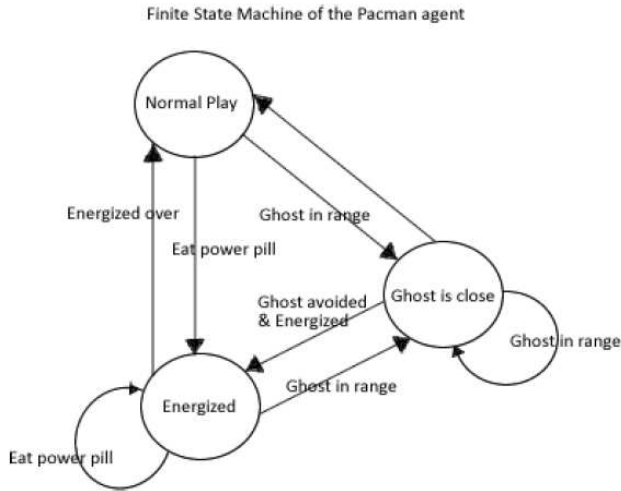


Fig. 2. This figure shows the Finite State Machine that controls the Pac-Man agent. The agent can exist in 3 states; Normal Play, Ghost is Close and Energised, with the transition arrows dictating the changes in the environment that trigger state changes.

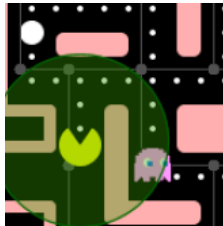


Fig. 3. This figure shows the invisible field of influence on the Pac-Man agent. This field is what will trigger an agent to recognise an enemy as a threat if the enemy is noted to be within the field.

- *Random*: At each node, the agent will make a random move down any of the available corridors of the maze.
- *Greedy-Random*: The agent traverses the maze such that at any given node the agent will take the path with the largest number of pills available. Should the agent reach a position where there are no pills on any connected edge, the agent will operate akin to the random behaviour described above.
- *Greedy-Lookahead*: Our agent will operate in the same greedy fashion as described in the Greedy-Random strategy. Should the agent reach a position where there are no nearby available pills, the agent will search across all available edges in the graph until a node is found in the graph with any pill filled edges. Once this node has been found the agent deploys an A* search algorithm

TABLE I

A BREAKDOWN OF SCORES ACHIEVED WHEN RUNNING 50 COMPLETE GAMES OF PAC-MAN USING EACH STRATEGY AS WELL AS RUNS CONDUCTED BY A NOVICE HUMAN PLAYER.

Strategy	Min	Max	Mean	Std. Dev
Random-Only	280	6360	2107.8	± 1158.64
Greedy-Random	3660	10670	5749.4	± 1344.98
Greedy-Lookahead	4550	11940	7826.4	± 1309.05
Human Player	1950	13640	8120	± 2876.24

to find the fastest path to this node. Once a path is determined the agent will immediately execute this path and continue to operate as specified.

While these strategies provide movement across the graph, they do not take the ghosts into consideration. Ghost avoidance is activated using the FSM, with a series of small hand-crafted rules that dictate the appropriate action the agent should take. Should the agent be approaching on a particular axis, the default action is to turn around and head back towards the last node visited at which point the agent will rebuild the next path to take. It is important to note that should an agent be running on ‘Greedy-Lookahead’, once it reaches the next node the path it has taken will be attributed a strong negative weight to prevent the agent from using that edge.

When the agent is in the energised state, the agent will immediately seek to discover whether an enemy is within the vicinity for consumption. This is achieved by calculating the distance to the nearest ghost using Manhattan distance. If this is greater than half the maze size then the ghosts are ignored and the agent continues the consumption of pills. However if the ghosts are close-by, the agent approaches the nearest ghost by closing the shortest distance along either the x or y axis.

V. EXPERIMENTS & RESULTS

In order to assess the validity of our hypothesis, we apply each of our 3 strategies in a series of experiments. Each strategy is executed in 50 complete matches, i.e. a standard game of Pac-Man until the agent has lost all 3 lives where each pill consumed is worth 10 points. In doing so we calculate the maximum, minimum, mean and standard deviation of scores from each batch of 50 matches. To provide an interesting comparison, we also give the scores from 50 human trials using the same domain.

For our human trials we used 10 students in the department who were moderately skilled in Pacman. The students placed strong emphasis on local pill consumption; eating pills in the nearby vicinity until threatened by ghosts. Should they successfully shake off the ghosts then they would continue to clear the area of the maze they were focussing on.

The results from our experiments are provided in Table I, it is clear that of our agent players the lookahead agent generates the best results. The progression of scores from the random agent to the lookahead agent show that as expected,

scores improve as the agent makes more rational decision processes at each node. The random-only agent generates very poor results in some instances with a minimum of only 280 points. However it is interesting to note that a good random run can generate a score that is greater than the poorest greedy random and lookahead strategies. Further interesting results indicate only a difference of approx. 1300 points between the best greedy random player and the greedy lookahead. Fortunately the mean values tell a different story, with our lookahead generating the highest mean. With an average of at least 2000 points more than the greedy agent. What is promising to see is that the standard deviations across the board are relatively close to one another, hence providing us with a stronger clarification and validity of the mean values.

VI. DISCUSSION

The results provided in Table I are essentially a proof of concept. Our intentions were to generate a small controller framework that would generate capable gameplay in the Pac-Man game and investigate our hypothesis of rational decision processes. While this has been achieved, with our lookahead agents generating strategies that can perform almost on par with a human of varying ability. In fact it would be expected for the human player to still be able to surpass our attempts on average (as shown in Table I), since as we move down through the strategies we are gradually providing a more focussed, rational approach to tackling the Pac-Man problem (though we imagine the increase from greedy-lookahead to human cognitive functions is somewhat significant). The more sophisticated the decision process as well as the mental models of the environment, we assume that the player will become better at solving the overall task. The importance of lookahead in our observations is that a human player will often apply lookahead of some description to solve problems such as Pac-Man; where the state of the world is fully visible with strong assumptions of how the world may change in the immediate future. Allowing for a player to devise where next in the maze to visit to consume pills. The two poorer strategies we apply suffer from a lack of lookahead, the players operate in a reactive capacity, immediately grabbing nearby pills (by greedy or mere random behaviour) and unsurprisingly suffer in the rankings.

The use of lookahead to plan further into the world combined with low level control is part of our visualisation of how to solve the Pac-Man problem. In time we intend to expand upon this work, as one of the major criticisms at present is that the controller operates heavily on a predefined, handcrafted rules. However we feel it is important to expand the design to provide greater flexibility to the decision processes, to move away from hand crafted solutions to more autonomous decision processes and controllers. Below we provide a series of logical expansions to the current system that we hope to investigate in the near future:

- Firstly, we may consider expansion of the FSM to become more expressive of the particular states that

exist in the game world. While this may allow for the world to be expressed at greater level of granularity, the complexity of control may increase considerably.

- While our lookahead strategy works well in comparison with other strategies employed here. It would be of interest to apply more sophisticated strategies for more robust behaviour.
- The agent sphere of influence for ghost avoidance currently uses a 3 tile radius to recognise the nearest threat. Furthermore the agent will only consider one enemy at a time in our rule set for avoidance, with instances where agents are approaching from different directions causing problems for the agent. This is certainly an area that requires further investigation.
- Expanding on the previous remark, it would be of great interest to apply a perceptron and train the controller to learn how to avoid ghosts in a local context. Since our design relies only on immediate ghost avoidance for the nearest ghost, it would be interesting if we could deliver a fast, reactive system that can compensate for 1 or more ghosts in the vicinity and pull the player out of difficult situations. Given the amount of work that has been conducted in dealing with this particular subproblem as part of Pac-Man research [3], [2] this is an avenue that while challenging should be immediately accessible to researchers.
- Perhaps the most audacious expansion that we have considered is to apply real-time planning as part of the maze traversal algorithm. By incorporating the graph like structure into a planning domain description language such as PDDL and apply real-time planning for not only pill consumption but possibly long-term ghost avoidance and eating tactics. One of the actions at present our agent is incapable of making is staying still, thus preventing such strategies as drawing ghosts towards Pac-Man as the agent approaches or is sitting next to a power pill. Furthermore using planning could allow for the agent to prune the search space of future actions and prevent agents from trapping themselves in scenarios that the ghost avoidance controller (trained perceptron based or otherwise) would struggle to recover from.

VII. CONCLUSION

In this paper we have explored the application of random and basic informed strategies for traversing the haunted mazes of Pac-Man. Using a non-reduced reproduction of Pac-Man that incorporates features of Pac-Man and Ms Pac-Man, we have found application of local greedy pill consumption with path formulation for distant areas of the maze provides behaviours that when incorporated into our FSM construct can generate agents capable of competing with novice human players. We believe that this satisfies our hypothesis that Pac-Man solutions must consider controller differentiation

by function for maze traversal, pill consumption strategy and ghost avoidance. We hope to consider this preliminary research into our agent architecture, with intentions of expanding into autonomous ghost avoidance control and informed pill consumption strategies.

REFERENCES

- [1] J. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, 1992.
- [2] M. Gallagher and M. Ledwich, "Evolving Pac-Man Players: Can We Learn from Raw Input?" *Computational Intelligence and Games, 2007. CIG 2007. IEEE Symposium on*, pp. 282–287, 2007.
- [3] S. Lucas, "Evolving a neural network location evaluator to play ms. pac-man," *Proceedings of the IEEE Symposium on Computational Intelligence and Games*, pp. 203–210, 2005.
- [4] G. McLemore, "The top coin-operated videogames of all times. killer list of videogames." wWW Address: "<http://www.klov.com/TOP100.html>".
- [5] A. Kalyanpur and M. Simon, "Pacman using Genetic Algorithms and Neural Networks," Retrieved from <http://www.ece.umd.edu/~adityak/Pacman.pdf> (19/06/03), 2001.
- [6] M. Gallagher and A. Ryan, "Learning to play Pac-Man: an evolutionary, rule-based approach," *Evolutionary Computation, 2003. CEC'03. The 2003 Congress on*, vol. 4, 2003.