COMPSYS 726 Assignment 1: Expert Systems – Mario

Johnny Shen (jshe979)

Introduction

With artificial intelligence (AI) becoming more advanced in recent years, many systems have incorporated the use of AI to emulate the behaviours of human intelligence to solve complex problems and make decisions.[1] Expert systems, a branch of artificial intelligence, can provide decision-making abilities like humans by reasoning about a problem. The most common type of Expert System is the rule-based expert system, which uses rules to make decisions. [2] Human experts create these rules in the form of 'if-else' statements. [3]

Super Mario Land is a platforming game developed by Nintendo for the Game Boy, which was like their existing product, Super Mario Bros. The game requires the player to move Mario in various ways to progress to the end of twelve levels.[4]

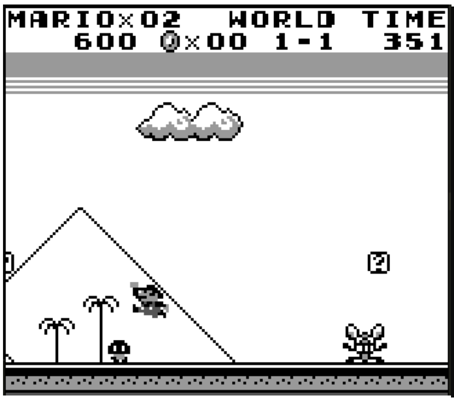
This report focuses on implementing an expert system to play the Super Mario Land game. The methodology section will explain the decisions made throughout the development of the system, and the results will be discussed later.

Methodology

Analysis:

The controls of the game are relatively simple. Left-Arrow and Right-Arrow will move Mario to the left or right, respectively. Pressing either button along with the B-button will cause Mario to move faster. Mario will jump if the player presses the A-button or travel down a pipe if Down-Arrow is pressed.

The game's environment can be examined using the self.environment.game_area() function, given by the MarioExpert script. By printing the game area onto the terminal, the game can be represented in a 16x20 matrix.




```
[[ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [13  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 13  0  0]
 [ 0  0  0  0  0  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 18 18  0  0  0]
 [ 0  0  0  0 15  0  0  0  0  0  0  0  0  0  0 18 18  0  0  0]
 [10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10]
 [10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10]]
```

Fig. 1: Game area examined

Given that for each render of a frame, the value of a specific object can be found. For example, Mario is represented by the value ‘1’, and a Goomba/Chibobo is represented by the value ‘15’. By obtaining the location of the enemy object and Mario, the distance between them can be calculated for our expert system. We can then use the data to distinguish between enemy objects and terrain to determine Mario’s correct action.

Implementation:

There are various types of expert systems, such as rule-based, fuzzy-logic, and knowledge-based. However, rule-based expert systems are better suited for integrating into Super Mario Land than fuzzy-logic ones since they utilise the traditional binary approach.

As mentioned previously, we can obtain the location of a given object to ‘calculate’ the correct action for Mario. Mario is represented by the value ‘1’ but in a 2x2 matrix. Therefore, Mario’s location is based on ‘1’ at the bottom right of the matrix. We will analyse the implementation using Goomba as our example. Throughout the game, Mario encounters different enemies. The expert system is implemented so that when an enemy is detected on the screen, its location and distance are calculated. If the enemy is in front of Mario (Same row), Mario will continue moving forward before jumping over it. Likewise, if the enemy is behind, Mario will move backwards. Due to the rendering, the jumping distance for Mario to the enemies varies from 1 to 3 spaces away. These actions only apply on flat levels.

Since every part of the stage changes in height, there must be a method to determine the ideal action. With the current implementation, if the enemy is on higher ground than Mario, Mario will remain stationary until the enemy approaches. The only exception is if the ledge is within jumping distance, which prevents Mario from jumping onto the enemy. Mario will move away until the rules on flat ground have been met. On the other hand, if the enemy is below Mario, Mario could jump on to them, but only if it is safe to do so.

Some enemies require a different approach to defeating them. One of them is Kumo, the jumping spider (18). Because Kumo is always jumping towards Mario, Mario must keep his distance until the time to jump on top of Kumo is met. Mario moves backward when Kumo jumps and forward when Kumo is on the ground.

Priorities are not set in this implementation; it is currently arranged in terms of enemy values (ascending order). There are approximately 25 enemies in the game [5], and we have only seen four in the first two stages. With the development of the expert system, it would be challenging to sort all the objects into specific priorities.

Finally, if no enemies are detected, Mario continues progressing forward, jumping over terrain obstacles and holes in the ground as necessary.

These rules have been written so Mario can complete the first stage accordingly.

Results

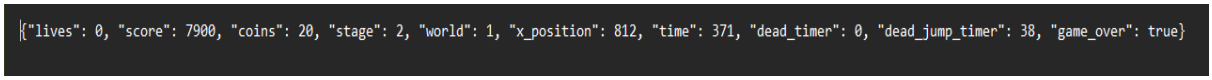


Fig. 2: Game result

The expert system implemented in this program to play Super Mario Land has completed the first stage with only one life lost. The agent demonstrated its attempt at completing the second stage but failed to do so. After analysing the reason for failure, it was found that Mario did not jump twice consecutively. He had jumped onto the platform, but a Nokobon (turtle) was on the screen, which met the requirement to jump again—however, the action needed to be reset before repeating, thus leading to failure.

Another factor that may have affected the progression in the second stage is that the expert system was implemented to detect ‘0’ along row 15 so that Mario would jump across the hole. Since the second stage was in the air, row 15 would always contain ‘0’s. As a result, the action that is always selected for that rule would be jump.

The results proved that the rules created by the human expert are only effective for specific stages. An expert system can complete the whole game, but that would require better-defined rules that could be applied to all levels/elements of the game, which would need deeper analysis to determine the standard rules across all levels.

In conclusion, the expert system chosen for this program is a rule-based expert system due to its straightforward implementation. The agent can complete the first stage, but more rules must be defined if the agent wishes to progress. Furthermore, there may be a better expert system than the rule-based one.

References

[1] “What is Artificial Intelligence? — AI New Zealand | Independent AI Advice For Businesses,” AI New Zealand. https://newzealand.ai/what-is-ai

[2] “What Are Expert Systems? | Baeldung on Computer Science,” www.baeldung.com, Mar. 22, 2023. https://www.baeldung.com/cs/expert-systems

[3] Wikipedia Contributors, “Expert system,” Wikipedia, Mar. 31, 2019. https://en.wikipedia.org/wiki/Expert_system

[4] “Super Mario Land,” Wikipedia, May 24, 2024. https://en.wikipedia.org/wiki/Super_Mario_Land

[5] “Super Mario Land,” Super Mario Wiki, Dec. 06, 2023. https://www.mariowiki.com/Super_Mario_Land