SNAKE GAME

Name :- Mannat Sapra , Dipendar Kumar

Reg.no :- 11802769,11802820

Advisor: - Sagar Pandey Sir

Department of Computer Science and Engineering, Lovely Professional University, India

**INTRODUCTION**

Snake game is a ideal computer game, in which we control a snake to move around and collect food in a map. In this paper a controller is developed using artificial intelligence by using A\* algorithm. Before elaborating how to design the controller,  the snake game which is implemented and the objective defined will be explained initially.

In the game , the snake is allowed to pass through all the area around a 2-Dimensional playing field game map which is surrounded by walls. At each distinct interval (a time step), the snake should move forward, turn left, turn right, as the snake requires and the snake cannot stop moving. The game will be generated randomly and a piece of food will be placed anywhere in the map , whenever there is no food on the map. When the snake moves towards the food and if the food is eaten then the length of the snake will be increased by one. The goal of the game is to eat as many food without getting collide to the wall or by itself. The objective of the game is to maximize the score. score. The above-mentioned simple strategy may keep the snake alive, but without moving toward the apples efficiently it cannot get a high score. Thus, it is needed to be  designed with more intelligent controller, which is the topic of this paper.

**Acknowledgement**

We have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. We would like to extend my sincere thanks to all of them.

We are highly indebted to Sagar Pandey Sir for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project.

We would like to express our gratitude towards our parents & member of Lovely Professional University for their kind co-operation and encouragement which help us in completion of this project.

**HISTORY**

The Snake design dates back to the arcade game Blockade developed and published by Gremlin in 1976. It was cloned as BigfootBonkers the same year. In 1977, Atari released two Blockade-inspired titles: the arcade game Dominos and Atari VCS game Surround. Surround was one of the nine Atari VCS (later the Atari 2600) launch titles in the United States and was also sold by Sears under the name Chase. That same year, a similar game was launched for the Bally Astrocade as Checkmate.

Nibbler (1982) is a single-player arcade game where the snake fits tightly into a maze, and the gameplay is faster than most snake designs. Another single-player version is part of the 1982 Tron arcade game, themed with light cycles. It reinvigorated the snake concept, and many subsequent games borrowed the light cycle theme.

Starting in 1991, Nibbles was included with MS-DOS for a period of time as a QBasic sample program. In 1992, Rattler *Race* was released as part of the second MicrosoftEntertainmentPack. It adds enemy snakes to the familiar apple-eating gameplay.

MeercaChase is a snake game available on Neopets.

Slither*.io* (2016) is a popular multiplayer interpretation of Snake*.*

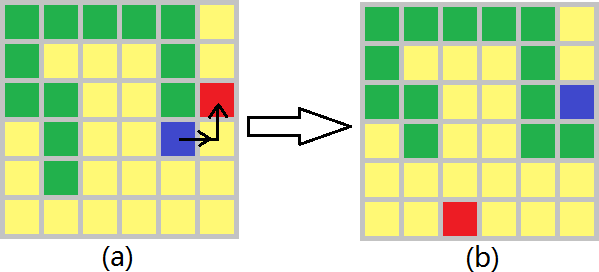
In 2017, Google released their version of the game as an easter egg, whenever the phrases "snake", "play snake", "snake game" and "snake video game" are typed.

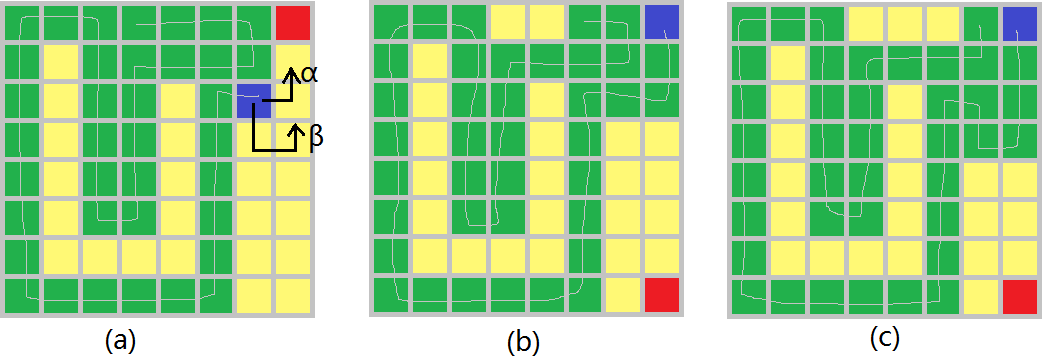
**ABSTRACT**

Methods in the domain of artificial intelligence (AI) have been applied to develop agents capable of playing a variety of games. Snake game is a computer game, whose goal is to control a snake to move and collect foods in the map. The single-player variant of Snake is a well-known and popular video game that requires a player to navigate a line-based representation of a snake through a two-dimensional playing area, while avoiding collisions with the walls of the playing area and the body of the snake itself. A score and the snake length are increased whenever the snake is moved through items representing food. The game thus becomes more challenging as the score increases. The application of AI techniques to playing the game of Snake has not been very well explored. In this project we work on techniques which are empirically compared against three Snake playing agents in terms of several performance measures(GREEDY, Hamilton, and DQN).

WORKING OF SNAKE GAME

A∗ is guaranteed to find an optimal path if it exists. In Snake, A∗ Search uses the Manhattan distance as a heuristic. Up to some extend, A∗ is comparable to Breadth First Search as both can find the optimal path. However, A∗ Search considers heuristic information and only expands nodes that can potentially lead to an optimal path, therefore expanding less nodes than Breadth First Search. Even though it can outperform Breadth First Search or Depth First Search, A∗ Search can still lead a dead end if the snake is long enough.In this case, A∗ Search makes the snake succeed in eating the apple but after that the snake can not do anything but move forward until there are no more moves available. This is incurred by the property of A∗ Search that it only considers current situation .

  
*A*∗ Search can always find the shortest path as long as a valid path to the goal exists. However, it does not consider the effects when the snake succeeds in eating the apple. In this figure, *A*∗ Search chooses to move right and up. However, after eating the apple, the snake does not have any other choice but to continue going up until no more moves are available. *A*∗ will get stuck in the dead end.



The forward checking will check several steps ahead to see if this move incurs in a dead end. This upgrade is expected to improve performance when comparing it to the plain *A*∗ Search.However, *A*∗ Search with forward checking also canlead to an avoidable dead end. Fig. 10 demonstrates such an example. With the current state in (a), *A*∗ Search will choose to move right and up. This easily leads to a dead end, as shown in (b). However, *A*∗ Search with a 3-step forward checking will make the snake move down by one unit, before moving right and up. This will result in (c), which is **NOT** a dead end at present, because the snake can move left and the tip of the tail will move downward to give way to the head. This is determined by the snake’s growth mechanism. But as the snake keeps moving left for some more steps, the dead end will be eventually be reached. Still, this dead end can be avoided if the snake moves down by 5 steps at (a), which will make room around the tip of the tail. In other words, checking for more steps will eliminate more dead ends that will otherwise occur with fewer forward checking steps.

*×*

There is still one more possibility in which *A*∗ with forward checking can lead to a dead end. When using Breadth First Search to look for possible dead ends, apple

placement is not considered. The placement of the apple can turn states that were initially thought not to be part of a dead end into one. Imagine a case were the head and the first parts of the snake are moved in a way that fit in a certain region of the board, the placement of the apple in that region will make the snake grow by one unit potentially making the computations made by the Breadth First Search algorithm useless.

Therefore, *A*∗ Search with forward checking can still reach a dead end if the forward check depth is not

high enough. Even though checking for more steps will eliminate more dead ends, the time spent on the check- ing, can slow down the process significantly, making the automated snake game solver unpractical.

**Packages which we are using for Snake Game**

**Numpy:-**

NumPy is a general-purpose array-processing package. It provides a highperformance multidimensional array object, and tools for working with these arrays.It is the fundamental package for scientific computing with Python. As the whole project is based on whole complex stats ,we will use this fast calculations and provide results.

Random:-

Python offers randommodule that can generate random numbers. These are pseudo-random number as the sequence of number generated depends on the seed. If the seeding value is same, the sequence will be the same.

Pygame:-

Pygame uses the Simple DirectMedia Layer (SDL) library,with the intention of allowing real-time computer game development without the low-level mechanics of the C programming language and its derivatives.Pygame is a cross-platform set of Python modules designed for writing video games. It includes computer graphics and sound libraries designed to be used with the Python programming language.

Tkinter:-

Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit.

CONCLUSION

It is our group hope that this document will be of huge help with understanding of our little project as we have used a A\* algorithm for building a Snake Game which has proved beneficial for us and easy for us to understand the vast ocean that is Artificial Intelligence.

REFERENCES

* <http://en.wikipedia.org/wiki/Snake>
* [www.edureka.co](http://www.edureka.co)
* www.tutorialspoint.com