Snake & Ladder

## 

**A Project Work Report**

*Submitted in the partial fulfillment for the award of the degree of*

# BACHELOR OF ENGINEERING

**IN**

**Computer Science Engineering**

**(Spl. In Mobile computing)**

**Submitted by:**

**Aniket Garg**

**Under the Supervision of:**  L&T

**6th April-16th April(2021)**

# Abstract

Snake is a typical video game where the player maneuvers a line which grows in length, with the line itself being a primary obstacle. The concept originated in the 1976 arcade game Blockade, and the ease of implementing Snake has led to hundreds of versions (some of which have the word snake or worm in the title) for many platforms. The player controls a dot, square, or object on a bordered plane. Our goal is to make the snake has a specific length, so there is a moving tail a fixed number of units away from the head. The player loses when the snake runs into the screen border, a trail or other obstacle, or itself.

# 

## Table of Contents

1. **INTRODUCTION**
2. **PROBLEM FORMULATION**

Problem Definition

1. **METHODOLOGY USED**

Environment and Agent

Flowchart

Reward

1. **RESULT & OUTPUT**
2. **CONCLUSION**

# Introduction

We have embbed an output video in the git repo. If you haven’t watched it you can also find the video by the link given below:

https://drive.google.com/drive/u/0/folders/1eXDSP\_a9SWYBzJNB23EJd2\_ZdjyW6F

Snake is the common name for a video game concept where the player controls a dot, square, or object on a bordered plane. As it moves forward, it leaves a trail behind, resembling a moving snake. In another common scheme, the snake has a specific length, so there is a moving tail a fixed number of units away from the head. The player loses when the snake runs into the screen border, a trail or other obstacle, or itself.

# PROBLEM FORMULATION

## Problem Definition :-

In a 2D environment, there are three main components: snake which corresponds to white circle , boundary corresponds to # and fruit corresponds to F alphabet respectively. In this problem, our goal is to make the snake has a specific length, so there is a moving tail a fixed number of units away from the head. Player will need to decide which action to take for eating fruit and to avoid touching its own body or the boundary.

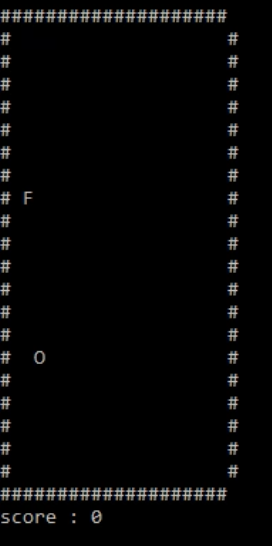


Figure 1

## 

## METHODOLOGY USED

## Environment and Agent :-

We develop our own snake environment. Our environment includes different setups. For example we can control the scale of the environment, also we can control whether the snake can go cross the boundary or not.

There are three components located in the squares: Snake,boundary and Fruit.

1. Snake is the agent in this environment. The triangle is its head, and the head controls the direction of movement, up, down, left, and right. Snake’s body has three blocks at beginning.Noted, snake cannot turn back because it will be considered as touching its body. Snake once touches its own body or boundary, it will die.

2. Boundary limits the size of the environment. At the same time, the snake will not allowed to come across the boundary once touching boundary.

3. Fruits mean reward for the snake. Environment includes only one kind of fruits: the white fruit. The white fruit appears and stays in fixed location until it is eaten by a snake and then environment will reproduce it in another place. If snake succesfully eat a white fruit,the score it get will add one. After eating the fruit, the small snake will gain growth and increase the length of its body.

**Flowchart :-**

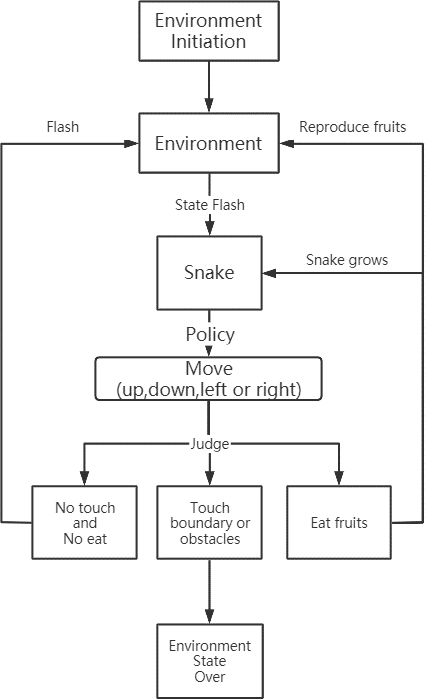


Figure 2: Environment and Agent

## Reward :-

1. Firstly we define the distance between snake and fruit as follow:

*dis* = (*x*1 *− x*0)2 + (*y*1 *− y*0)2

√

(*x*0*, y*0) is the positon of the fruit.

(*x*1*, y*1) is the postion of snake head

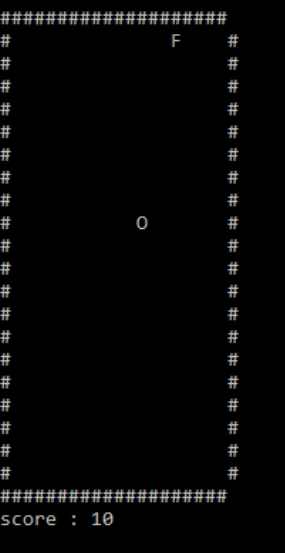
1. Then we can formulate the reward for the snake as following table:

|  |  |
| --- | --- |
| State | Reward |
| take a step and nothing happen | (1/max(1.0,dis)\*1 |
| take a step and game over | -1 |
| take a step and eat a fruit | +1 |

**Result & Output**

In our project, the player has to decide which action to take for eating fruit and to avoid touching its own body or the boundary. As the snake eats a fruit, a score of 10 is counted and if snake hits the hash # it will die and hence the program will end.

Output of the snake eats a fruit…



Result of Output

# 

# Conclusion

In our project, we create an environment of Snake game where Player will need to decide which action to take for eating fruit and to avoid touching its own body or the boundary and our project runs successfully.