**Run And Jump Game**

****

**Using Genetic Algorithm and Artificial Neural Networks**

**Victor Yu**

**StudentID:2000173**

***Abstract* –** With the development of the game industry, AI is becoming more and more widely used in game area and there are many Artificial Intelligence Algorithms that can help make the game more attractive.

The purpose of this project was to develop an application that can demonstrate the selected Artificial Intelligence Algorithms. The application is an endless run game and creates using the Unity game engines and C# programming language. The Artificial Intelligence Algorithms used in the application were Genetic Algorithm and Artificial Neural Network.

1. **Introduction**

The intent of this project was to design and develop a 2 – dimensional endless run game using Unity. It was not easy to decide what application is going to make at start as there are many AI Algorithms that can choose from. The original idea was to create a small 3d scenes where AI character can move towards the player and attack player by using the FSM (Finite state machine) and Artificial Neural Network. However, after a period of development and research, this plan did not continue.

The new idea of this game was inspired by the video found online.[1][2] The game used the Genetic Algorithms and Artificial Neural Network so that the AI will control and decide the movement of the character. The key learning for the AI was to decide when the character needed to jump to avoid hit. The game itself was not very difficult but the implementation of AI techniques was challenging.

## AI Techniques

The game consists of 2 type of AI techniques combine, they are Genetic algorithm and artificial neural network.

### Genetic Algorithm

Genetic algorithm are randomized search algorithms and first introduced by John Henry Holland in 1975. [3] The algorithms involving biological processes, usually mutation, crossover and selection.

* Selection - Select several individuals from the population and used for later process (usually crossover process). Figure 1.1

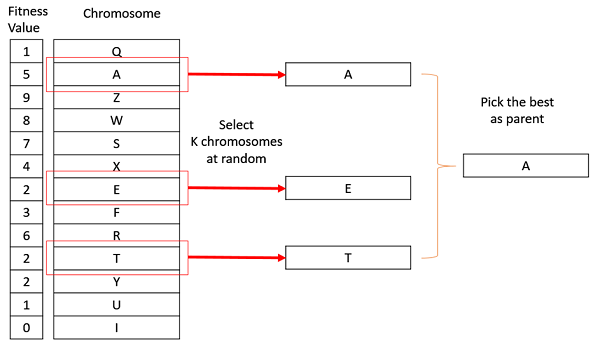


Figure 1.1 [4]

* Crossover – From Selection more than one parent is selected and used in crossover process, where it combines the genetic information from the parents and generate new offspring. for example, the gene is cut at the same position on the two parent chromosomes, then the front and back strings are combined to form two new child chromosomes. There are different types of crossover: Single Point Crossover, Two-Point Crossover and Uniform Crossover. Figure 1.2 [5]

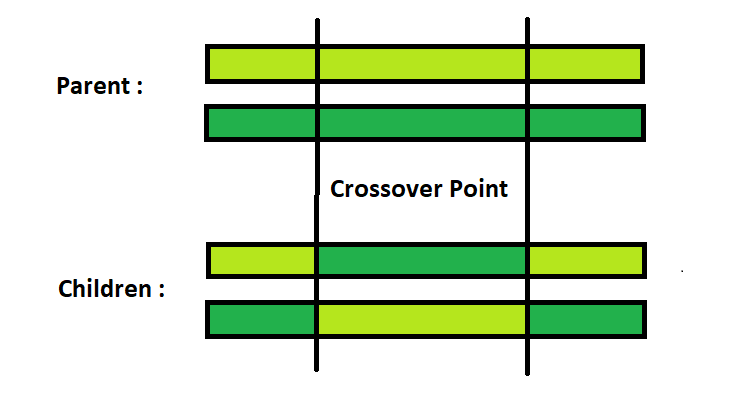


Figure 1.2 [5]

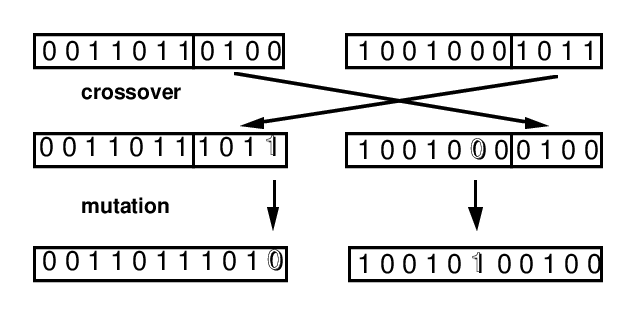
* Mutation –When the reproduction happens and there is small chance where mutation occurs by alters one or more gene values in the chromosome and produced new solution. And the difference between mutation and crossover is that the mutation only happens within one individual while crossover is between two. Figure 1.3 [6]

Figure 1.3 [6]

So how the Genetic algorithm flow works? They can be described as the follow steps: Figure 1.4[7]

1. The algorithms assess the fitness of the individual corresponding to each chromosome in the population.
2. using Selection to select two individuals from the population as the father and the mother by following the principle of higher fitness and greater selection probability.
3. Using the crossover function to extract the chromosomes of both parents and cross over to produce offspring.

1. Then the mutation function will make mutations to the chromosomes of the offspring
2. Repeat steps 2,3 and 4 until a new population is produced.

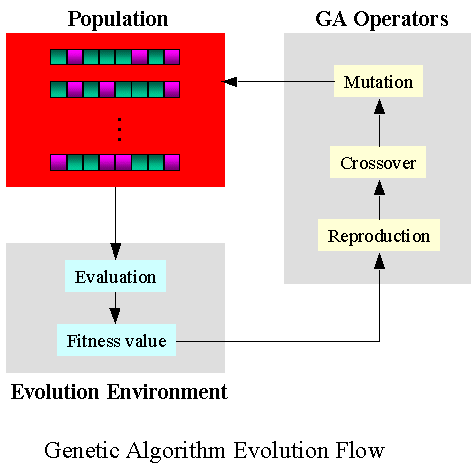


Figure 1.4 [7]

### Artificial Neural Network

The first Artificial neural network was developed in 1960s by Frank Rosenblatt, called perceptron.[8] It started at the same time as the artificial intelligence develop, but it has not achieved results like AI in more than 30 years until 1986 Maclelland and Rumelhart proposed the Back-Propagation algorithm to break through the technical bottleneck and become popular since then.[9]

Artificial neural network is one of the main tools in machine learning. It abstracts the human brain neuron network from the perspective of information processing, establishes a certain model and forms different networks according to different connection methods. This produce a brain-inspired system which can replicate the way that humans learn. So basically, a neural network is a computing model, which is composed of many nodes connected to each other and each node represents a specific output function, which is called an activation function. Each connection between two nodes represents a weighted value for the signal passing through the connection and this is called weight, which is equivalent tot the memory of an Artificial neural network. [10][11] Figure 2.1

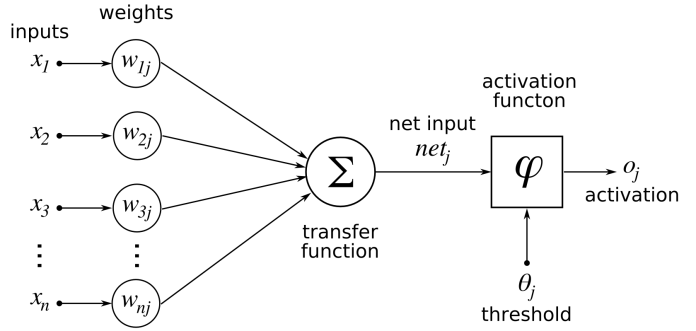


Figure 2.1[11]

It consists multiple layers:

* Input layer – accepts signal and data from the outside world
* Output layer – realizes the output of system processing results
* Hidden layer – perform the required calculations and output result to the output layer

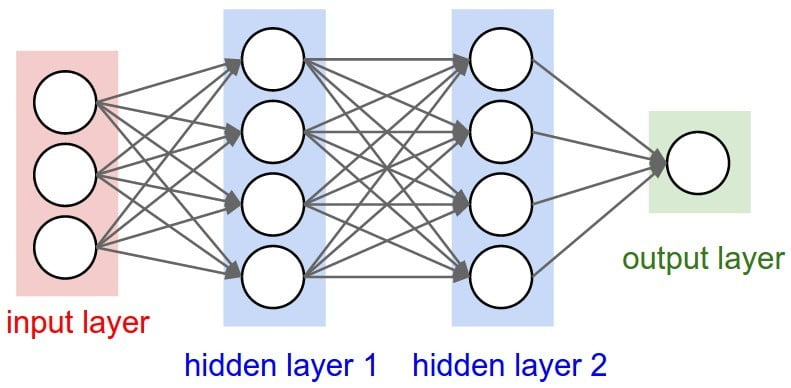
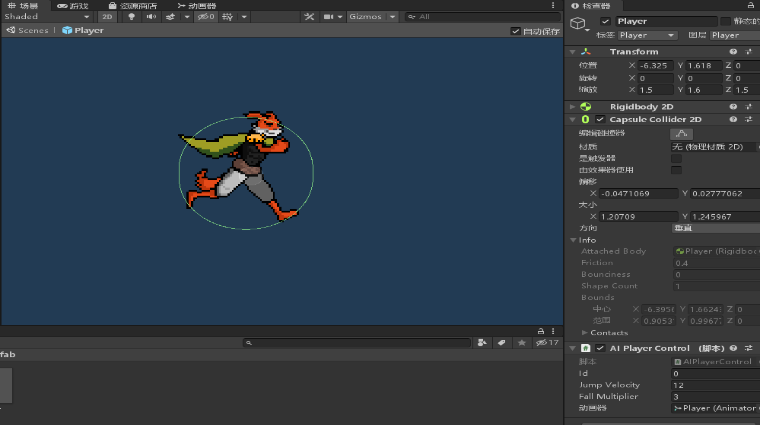


Figure 2.2 [10]

## Unity

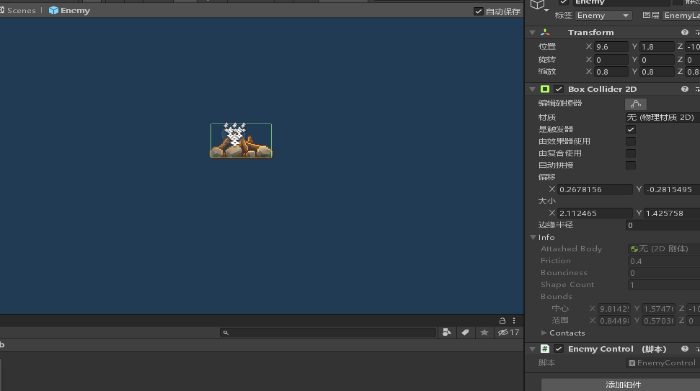
### Prefabs

* Player:



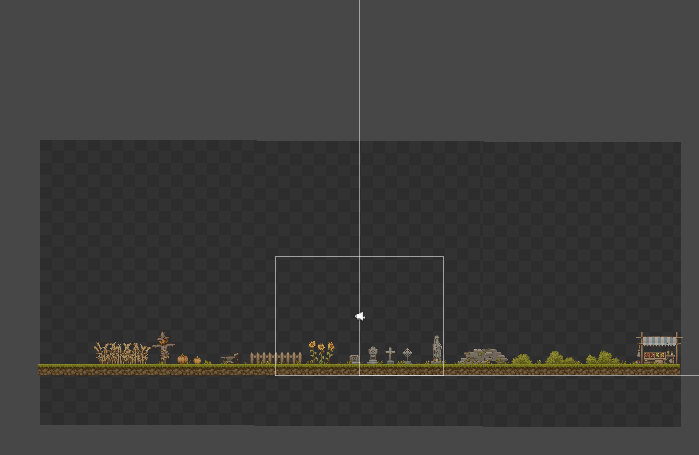
Player prefab contains AIplayerControl scripts, the variable jump velocity and fall multiplier can be set manually. This prefab with animation is come from the unity Asset store for free.[12]

* Enemy:



Enemy prefab contains Enemy scripts, the script which used to set the default of the enemy’s transform and spawn position. It also contains the partical system for the fire. This prefab was create using the Asset from the unity Asset store for free.[13]

### Scene



The scene was create using the Asset from the unity Asset store for free.[13] The floor have script attach to it so that it can transform and always stay in the camera area to make the floor endless, same as the background. The scene also have an GameControl system attach to it. This gamecontrol system is used to setup the game setting such as speed, the time of enemy spawn and music. For the testing purpose there is another control called AIControl, in this system the number of AI player spawn can be set so that can be used as the population for genetic algorithms.

## Scripts

### Normal Scripts

* AIPlayerControl -

This script was created to allow the player to move in the game and animate on each move such as run and jump. The scripts have function check is player alive or not using the collider.

* EnemyControl –

A simple script writes to make enemy prefab move toward player and destroy itself if it is outside the camera view.

* Enemy –

This script was used to spawn based on the spawn interval time set.

* BackgroundC –

Script to make the background realistic and always in the game scene.

* GroundControl –

Script to make the floor move and always in the game scene.

### Manager Scripts

### GameControl –

### Script to setup the default setting for the game, contain method to update the score and the ability to reset the scene when player all died.

### AIManager-

### Script to setup the population of player and initial the player AI.

### AI Scripts

### NeuralNetowrk –

### This script was used to produced randomized weights.

### GeneticOps –

### Sigmoid () – this is the sigmoid activation function for the artificial neural network.

### This function return a sigmoid value between 0 and 1 by using the algorithm S(x) = 1/(1 + (e^x)). The code Mathf.Exp((float)matrix[I,j])) which gives the result of e^x.

### Genetic –

### CrossOver() – this method combine the crossover and mutation together.

### What this function does is it create an offspring by copying the weights of one player’s neural network and make a new one with the previous creature’s weights and adding some mutation to it. [14]

## Test

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No.Population | Score | Generation | mutationRate | time |
| 5 | 5569 | 26 | 0.01f | 5m |
| 5 | 406 | 47 | 0.01f | 5m |
| 5 | 7939 | 21 | 0.5f | 5m |
| 5 | 581 | 25 | 0.5f | 5m |
| 5 | 4800 | 36 | 0.2f | 5m |
| 5 | 5385 | 24 | 0.2f | 5m |
| 10 | 3362 | 17 | 0.5f | 5m |
| 10 | 8562 | 12 | 0.5f | 5m |
| 10 | 5992 | 21 | 0.01f | 5m |
| 10 | 2424 | 30 | 0.01f | 5m |
| 10 | 6469 | 23 | 0.2f | 5m |
| 10 | 4000 | 16 | 0.2f | 5m |

The above table is the test data gathered from the testing. The mutation Rate and number of populations were changed to test the game and the testing period is 5 minutes each.

Author set the number of populations to 5 and test it 2 times for each mutation rate. As a result, when the mutation rate is set to 0.01 the generation period were 26 and 47, when the mutation rate is 0.2, the result were 24 and 36, and when the mutation rate is 0.01 the result were 21 and 25. This shows that as the mutation rate increase, the less generation required for the AI to become intelligence. Another factor that can impact the generation is the number of populations. This is because when the population become large there were more gene to be compare with and because for the game the AI in the start was set to random gene so when the population increase there is higher chance for good gene to be generate.

## Conclusion

The overall result was good, the game was successfully created by using those two algorithms combined.

However, as the gene was randomly giving at the beginning there is small chance when the AI player become very intelligence at very beginning and as the game is endless run so the generation might not goes on as the current generation might stay really long time, this was consider to be update in the future.

**Video**

**https://youtu.be/9lQn4kc5Lcg**

**Reference**

1. “Genetic algorithm. Learn to jump over ball” <https://www.youtube.com/watch?v=Gl3EjiVlz_4>
2. “Neural Network + Genetic Algorithm + Game” <https://towardsdatascience.com/neural-network-genetic-algorithm-game-15320b3a44e3>
3. “adaptation in natural and artificial systems” <https://search.proquest.com/openview/c78067a39019fb29daf134cf5dfbb2d6/1?pq-origsite=gscholar&cbl=30748>
4. “Selection”<https://www.tutorialspoint.com/genetic_algorithms/genetic_algorithms_parent_selection.htm>
5. “Crossover”<https://www.geeksforgeeks.org/crossover-in-genetic-algorithm/>
6. “Mutation”<https://www.researchgate.net/figure/Crossover-and-mutation-operators-in-the-genetic-algorithm_fig3_2263212>
7. “Genetic Evolution flow”

<https://www.ewh.ieee.org/soc/es/May2001/14/Begin.htm>

1. “Perceptron

<https://en.wikipedia.org/wiki/Perceptron>

1. “back-Propagation

<https://en.wikipedia.org/wiki/Backpropagation>

1. “what is ANNS”

<https://www.digitaltrends.com/cool-tech/what-is-an-artificial-neural-network/>

1. <https://bdtechtalks.com/2019/08/05/what-is-artificial-neural-network-ann/>
2. <https://assetstore.unity.com/packages/2d/characters/fox-the-fox-full-version-59175>
3. <https://assetstore.unity.com/packages/2d/environments/pixel-art-platformer-village-props-166114>
4. <https://www.youtube.com/watch?v=lu5ul7z4icQ>