Asteroid AI

Inteligência Artificial em Jogos

Faculdade de Ciências da Universidade de Lisboa

Docentes

Luís Moniz

Discentes

Lucas Silva Nº55922

Francisco Madruga Nº55956

Rodrigo Nogueira Nº55721

Tabela de conteúdos

1 Introdução

Na proposta feita pelo docente da cadeira de Inteligência Artificial em Jogos, Luís Moniz, para o projeto final da cadeira, pedia a implementação de um AI num jogo à nossa escolha, ou tentar melhorar o AI de um jogo, utilizando técnicas referidas nas aulas. O nosso grupo preferiu a implementação em vez do melhoramento de um AI. Durante este projeto, foi desenvolvida uma variante do jogo ”Asteroids”, em pygame. Depois do jogo estar desenvolvido, passamos à implementação do AI, no nosso caso escolhemos fazê-lo com um algoritmo genético para a criação de ”Neural Networks” evolutivas. Sendo que, estas despertaram o nosso interesse devido ao potencial que os seus modelos podem obter com uma implementação relativamente simples.

2 Implementação do Jogo Asteroids

Para termos um jogo semelhante ao original, precisamos de quatro instâncias principais, uma Nave, os Mísseis que são disparados, os Asteróides, e uma instância Jogo, que trata de realizar todas as operações relacionadas com a pontuação e movimentos dos objetos.

2.1 Classes

2.1.1 Nave

Na criação da nave tentámos manter o mais próximo ao original, desta forma, temos que a nossa nave consiste de um triângulo isósceles, este pode movimentar para a frente, fazemos isto através do cálculo de um vector, usando o centro e o vértice situado na frente da nave, e com esse vector realizamos a translação dos vários vértices da nave. A nave pode também realizar rotações para ambos os lados, sendo que rotação é feita com um ângulo fixo que é adicionado ou subtraído, para se obter assim a nova posição da nave. Esta é também capaz de utilizar as bordas da janela de jogo para se movimentar para a borda da janela oposta, mantendo assim a movimentação original quase idêntica. Finalmente a nave é capaz de disparar mísseis que destroem os asteróides.

2.1.2 Mísseis

Para os mísseis, decidimos que estes tivessem um formato circular para facilitar o cálculo das colisões com os asteróides. Este apenas podem se movimentar numa só direção, sendo essa direção no sentido em que a nave estava quando lançou o míssil, para tal, simplesmente fornecemos o vector de movimento da nave no momento de disparo, ao novo míssil assim o míssil só pode avançar no sentido desse vector.

2.1.3 Asteróides

Para os asteróides, temos que são círculos e variam de raio (10, 20, 40, 60) conforme o tipo de asteróide. O tipo de asteróide é escolhido de forma aleatória ao serem introduzidos no jogo, e ao se realizar uma colisão com um míssil, divide-se em dois asteróides com o próximo maior valor de raio, por exemplo, um asteróide de raio 60 ao ser atingido por um míssil divide-se em dois asteróides com raio de 40, com vectores de movimento diferentes ao asteróide pai.

2.1.4 Jogo

Esta classe controla os diversos aspetos do jogo, sendo esta que trata de realizar a introdução dos asteróides na janela de jogo, para tal existe um limite de asteróides que podem estar em jogo e apenas são introduzidos mais caso o número de asteróides seja inferior ao definido. Inicializar a nave na sua posição inicial e os mísseis que esta dispara, e verificar a posição dos mesmos caso estes colidam ou não com algum dos asteróides.

Tal como, podemos ver na seguinte figura 1, temos as entidades presentes no decorrer do jogo desenvolvido.

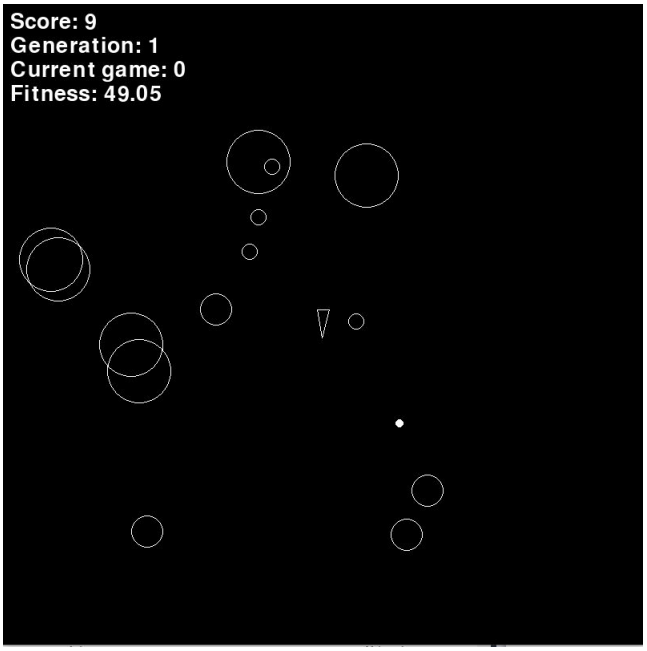

Figure 1: Entidades do Jogo Asteroids

3 NeuroEvolution of Augmenting Topologies

NEAT, ou NeuroEvolution of Augmenting Topologies, consiste num algoritmo genético de redes neuronais artificiais e evolutivas. Esta técnica baseia-se em seleção natural, onde os indivíduos mais aptos serão usados para criar uma nova geração que terá em conta os valores recebidos pelos antepassados, através de *cross-over*, podendo ser alvo de mutações.

Na nossa implementação, começamos com uma população de N naves/players, todas a jogar jogos diferentes um de cada vez. Estes players terão os seus comportamentos relativos aos valores dos weights que lhes foram atribuídos através das redes neuronais, e tentarão ter o fitness score mais alto possível. O fitness score é influenciado da seguinte forma: Por cada asteróide destruído, sobe K pontos, por cada segundo de sobrevivência, sobe X pontos, por cada míssil falhado, perde Y pontos e finalmente, eram retirados pontos se a nave estivesse a uma distância menor ou igual a Z. Apesar de estes serem os parâmetros definidos para o fitness, nós fizemos testes com outros a tentar encontrar mais possibilidades. Depois de todos os players perderem, começa a nova geração de players

utilizando os indivíduos mais aptos, falados anteriormente. Passadas algumas gerações é esperado que seja encontrado algum player que consiga sobreviver para sempre.

4 Redes Neuronais

As redes neuronais precisam de inputs de forma a conseguirem produzir um output viável, e estes podem variar bastantes conforme o contexto do problema, no caso deste decidimos que os inputs mais vantajosos para as redes seriam, a posição do centro da nave, a posição do vértice da frente, o ângulo da nave relativamente a um eixo imaginário criado, e posição, vetor de movimento, raio, distância à nave e ângulo com a nave dos N asteróides mais próximos.

Também decidimos utilizar vários valores para as *hidden layers*, sendo que estas são pequenos neurónios que transferem os dados de treino, de layer para layer até alcançar a layer de output. Finalmente, temos que os outputs vão ser as ações que podem ser realizadas, para tal, temos 4 neurónios um para cada uma das ações, ou seja, temos um neurónio que decide se deve ou não avançar, outro se deve ou não disparar, e caso deva ou não girar para cada um dos lados (esquerda, direita). Podemos ver um exemplo, de uma rede neuronal no contexto deste projeto na figura 2.

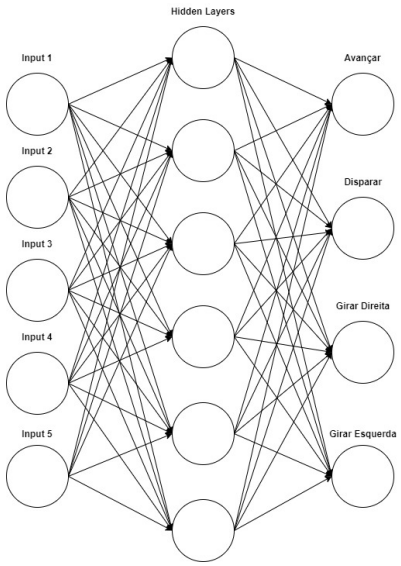

Figure 2: Exemplo Rede Neuronal Utilizada

5 Resultados

Consoante os parâmetros das redes neuronais, os inputs que lhes eram passados, o número de gerações e o tamanho da população do algoritmo genético, obtemos agentes com estratégias diferentes para abordar o jogo.

Inicialmente começamos por obter agentes que se mantinham no centro uma vez que era a zona que os asteróides demoravam mais tempo a chegar. Para resolver este problema decidi-mos recompensar os agentes que acertam nos asteróides e penalizar os que tem asteróides próximo de si. No entanto isto apresentou agentes com uma nova estratégia que consiste em rodar e estar sempre a disparar. Como este ainda não era o comportamento pretendido, decidimos penalizar os tiros que falhavam os asteróides, desta forma começamos a obter agentes que se desviavam dos asteróides e só disparavam quando tinham a certeza de que iriam acertar.

6 Dificuldades

Durante o desenvolvimento do projeto, não foram encontradas muitas dificuldades que tenham influenciado o projeto. No entanto, ainda enfrentámos alguns problemas que atrasaram um pouco o desenvolvimento. A primeira dificuldade com que nos deparámos foi no desenvolvimento do jogo, ao tentar obter a posição da nave. Apesar de tudo, isto foi algo ultrapassado bastante rápido, sem grandes problemas.

O mais difícil mesmo neste projeto foi tentar encontrar os parâmetros certos para obter o melhor resultado para o AI. A procura por estes valores foi bastante demorada e levou a bastantes dúvidas sobre se iríamos realmente encontrar uns que satisfizessem minimamente o propósito do jogo. Outro problema aconteceu durante a procura por estes valores, visto que tínhamos de correr várias instâncias do jogo com diferentes parâmetros, de forma a encontrar o ideal. Isto levou a que os computadores utilizados ficassem lentos, consequentemente, deixando o programa e os diferentes jogos bastante lentos estragando os resultados e aumentando o tempo de desenvolvimento do projeto. Foi necessário, de forma a contornar isso, correr os jogos de cada geração sequencialmente, com cada jogo durando o tempo de vida do player e o próximo começando apenas depois

do jogo anterior ter terminado.

Por fim, o que mais nos afetou no final deste projeto, foi que, apesar de termos encontrado um jogador bastante bom na simulação, não conseguimos guardar os seus parâmetros devido a um erro, o que nos comprometeu uma demonstração final com um bom AI para o jogo. Isto aconteceu já bem perto da entrega, o que tornou impossível correr de novo todas as gerações possíveis para o encontrar.

7 Conclusão

Com este projeto foi-nos possível desenvolver e principalmente aprender sobre algumas técnicas de Inteligência Artificial em Jogos, mais concretamente técnicas utilizando redes neuronais. O projeto correu bem durante praticamente todo o tempo de desenvolvimento. No entanto, como explicado anteriormente, deparámo-nos com alguns problemas que nos atrasaram e comprometeram até certo ponto o projeto.

Este projeto serviu principalmente como aprendizagem já que, os elementos do grupo não tinham muita experiência na utilização de redes neuronais e algoritmos genéticos, podendo então aprender bastante sobre estes assuntos enquanto se trabalhava diretamente sobre eles.

Concluindo esta reflexão sobre o projeto apresentado, pensamos que o objetivo traçado no início e o desafio lançado pelo docente, foi alcançado e bem ultrapassado, respetivamente, apesar da infelicidade final.

Referências

[1] Hunter Heidenreich. “NEAT: An Awesome Approach to NeuroEvolution”. In: (2019). URL: https://towardsdatascience.com/neat-an-awesome-approach-to-neuroevolution-3eca5cc7930f.

[2] Robert MacWha. “Evolving AIs using a NEAT algorithm”. In: (2019). URL: https://macwha.medium.com/evolving-ais-using-a-neat-algorithm-2d154c623828.

[3] Kenneth O. Stanley. “Evolving Neural Networks through Augmenting Topologies”. In: (2002). URL: http://nn.cs.utexas.edu/downloads/papers/stanley.ec02.pdf.