1.Introdução

O primeiro trabalho prático tem como objectivo principal a introdução aos princípios base da Inteligencia artifícial.

Desta forma, para implementar uma inteligência mais simples foi utilizada a inteligência de uma personagem virtual de um jogo. A personagem interage com um jogador humano. O principal objetivo da personagem é impedir que os inimigos entrem numa zona à sua guarda, de tal forma que, o ambiente e o jogador proporcionam as diferentes ações da personagem.

De forma a implementar o comportamento de forma mais eficiente possível, foi-nos dado uma estrutura pré-definida da estrutura de classes e dos métodos. Desta forma, foi possível manter o nível de complexidade estável para o objetivo pretendido.

2.Desenvolvimento

2.1 Aspetos relevantes

2.1.1 Paradigma Simbólico

- Para definir o comportamento da personagem primeiramente teve-se em conta o facto de que a sua mecânica era com base numa entrada, o dispositivo computacional realiza uma transformação e é obtida uma saída.
- Cada entrada foi designada por um símbolo, ao qual ao conjunto de símbolos obtém-se um alfabeto. Neste caso ás entradas serão o alfabeto de entrada, designado pelo símbolo $\sum$, e ás saídas o alfabeto saída, designado pelo símbolo Z.
- Para a transformação do dispositivo foi associada a função transformação, designada pelo simbolo lambda λ, no qual se obtém:

$$\lambda : \Sigma \rightarrow Z$$

2.1.2 Sistema com memória

- Desta forma a função de transformação do sistema descrito passa a ter duas funções, a função de transição de estado e a função de saída.

  Função transição de estado:

  $$\delta : Q \times \Sigma \rightarrow Q$$

  Função de saída:

  $$\lambda : Q \times \Sigma \rightarrow Z$$

- Este tipo de paradigma serve para criar Máquinas de estados finitos

2.2 Entradas e saídas do sistema

Este tipo de arquitetura dá enfase ao acoplamento com o ambiente. No qual o agente, neste caso a personagem do jogo, reage a estímulos por parte do meio ambiente. Para cada estímulo surge uma ação distinta.

2.2.1 Estimulo

Um estimulo no presente caso será um evento, como qualquer evento é exporádico e pode surgir em qualquer altura. Para representar o estímulo foi criada uma interface Estimulo, que no caso presente serve como uma interface de marcação.

2.2.2 Ação

Ação será a atividade que a personagem irá realizar perante o estímulo obtido, sendo portante a sua função executar algum tipo de tarefa. Na representação da ação foi criada também uma interface que apenas possui o método executar que é a sua função respetivamente começar a executar alguma tarefa.

2.3 Componentes base do sistema

2.3.1 Ambiente

O ambiente será responsável por desencadear os eventos e mostrar esses mesmos eventos. Desta forma o ambiente possui três classes diferentes uma que representa o próprio ambiente, este então responsável por gerar eventos, evoluir após o evento terminar e mostrar os acontecimentos. Possui ainda dois enumeradores que são os nomes dos eventos possíveis e o nome da ação que foi desencadeada pelo evento.

2.3.2 Personagem

A personagem será a responsável por percepcionar o estimulo do ambiente, realizar o processamento do estímulo no qual resulta uma ação e de seguida realiza essa mesma ação. Para a personagem foi também criado uma classe personagem que possui os métodos percepcionar, processar e atuar, que realizão exatamente esse mesmo comportamento. Além da classe personagem existe ainda um conjunto de classes associadas ao comportamento, onde cada classe representa uma reação distinta que a personagem irá realizar. Existe ainda uma classe que faz a inicialização de todos esses comportamentos associado com o comportamento da máquina de estados, quer isto dizer que os comportamentos estão organizados por estados.

2.4.3 Jogos

O jogo será responsável por correr o programa, neste caso será a classe com o método main que chama a função executar para começar o jogo.

Para que não exista crescimento da complexidade por meio da má modelação do problema, foi ainda criada uma biblioteca que funciona para agentes reativos. Este tipo de biblioteca tem todo um conjunto de funcionalidades para de forma clara e abstrata representar a relação entre uma percepção e uma ação típico de agentes reativos.

3. Biblioteca Reacao

A ideia por de trás dos agentes reactivos é um agente percepciona uma alteração no ambiente, essa percepção desencadeia uma reação e o agente atua sobre a mesma. Esta biblioteca possui uma entação com outra biblioteca designada maqest. Esta biblioteca tem como interesse principal fazer de forma clara e precisa a transição entre um estímulo e uma ação.

3.1 Reação

A reação é o que permite guardar um estímulo e uma resposta e ativa uma ação a partir de um estímulo de acordo com um comportamento. A reação é uma classe que tem um comportamento associado que é um contrato com uma interface designada Comportamento, e permite activar uma ação a partir de um estímulo.

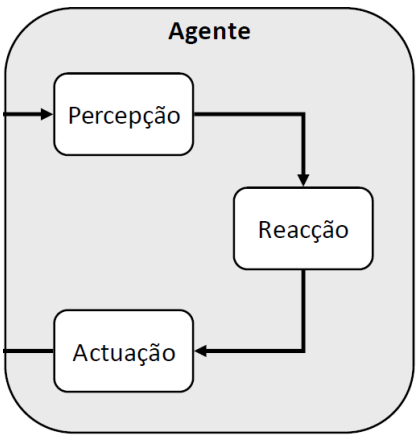

Figura 1 - Reação

3.2 Comportamento

O comportamento é uma abstração para algo que um agente realiza com base no fim que pretende atingir. Deste modo o comportamento é um contrato estabelecido sob a forma de uma interface entre o tipo de comportamento do agente, neste caso o comportamento hierárquico e o comportamento da máquina de estados.

3.3 Comportamento hierárquico

O comportamento hierarquico como o próprio nome indica faz a seleção das ações por meio de uma hierarquia fixa. Prevalecendo o comportamento com maior nível de prioridade em relação aos restantes. Deste modo a classe ComportHierarq permite a partir de um conjunto de comportamentos ativar a ação correspondente ao estimulo por meio dessa hierarquia.

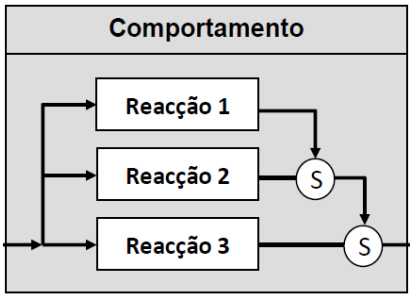

Figura 2 - Comportamento Hierarquico

3.4 Comportamento Máquina Estados

O comportamento da máquina de estados é guardar estimulos que serão representados sob a forma de estados. Cada um desses estados tem um comportamento distinto, sendo necessário fazer a sua correspondência. A classe possui um HashMap que permite exatamente essa funcionalidade de guardar e corresponder dois tipos de objetos que no caso são estados e comportamentos. Para fazer a atribuição de um estímulo a um estado, o estado internamente é parametrizado. A classe internamente possui um compromisso para com o comportamento e por isso permite activar uma ação a partir de um estímulo. Reescrevendo desta forma o método da interface comportamento permitindo a sua abstração. Permite ainda obter o estado que fez a associação com o estímulo.

Para que a classe do comportamento da máquina de estados funcione primeiramente precisamos de uma máquina de estados e para isso foi criada a biblioteca maquest.

4. Biblioteca maqest

4.1 Estado

Um estado é referente a uma entrada do problema, quer isto dizer que um estado será uma abstração para a máquina de estados sobre uma entrada do problema. Desta forma a classe Estado é parametrizada com um tipo genérico, esse tipo genérico será substituído pela dita entrada do problema que no caso presente é um estimulo. A partir do estado é possivel transitar de estado e processar o estado.

4.2 Máquina Estados

A máquina de estados é o que permite guardar estados. Deste modo a o que ela realiza é guardar um estado e processar o estado a partir de um dado evento.