FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE D
COIMBRA

Departamento de Engenharia Informática

Introdução à Inteligência Artificial
2019/2020 - 2º Semestre

Trabalho Prático Nº1:
Reactive D31: The AI Awakens

Nota: A fraude denota uma grave falta de ética e constitui um comportamento inadmissível num estudante do ensino superior e futuro profissional licenciado. Qualquer tentativa de fraude levará à anulação da componente prática tanto do facilitador como do prevaricador, independentemente de acções disciplinares adicionais a que haja lugar nos termos da legislação em vigor. Caso haja recurso a material não original, as **fontes** devem estar explicitamente indicadas.

1 Introdução

Os Agentes reactivos são agentes que percepcionam o ambiente e executam acções mediante essas mesmas percepções através de um sistema de regras simples. Embora se baseiem em processos simples, não implica que com este tipo de agentes não se consiga observar comportamentos complexos. Um exemplo emblemático deste tipo de agentes é descrito pelo neuro-anatomista Valentino Braitenberg no seu livro "Vehicles – Experiments in Synthetic Psychology".

Neste trabalho prático pretende-se a implementação de agentes reactivos autónomos. Neste caso o robo $D31$ encontra-se num ambiente povoado por obstáculos e recursos, e precisa de ajuda para navegar no mesmo.

A implementação será feita através de Unity. O Unity é um motor de jogos desenvolvido pela Unity Technologies, especialmente notável pela sua portabilidade entre plataformas. Entre as suas especificações é de distinguir os motores de física e gráfico, a extensa biblioteca e respectiva documentação, assim como suporte para as linguagens C# e JavaScript. Apesar da sua finalidade ser o suporte ao desenvolvimento de jogos, estas características permitem também que o Unity seja utilizado como ferramenta de simulação, com aplicabilidade na Inteligência e Vida Artificial, capaz de funcionar como ambiente para simulações realistas. Neste caso será utilizado para implementação e teste em tempo real de diversos agentes reactivos.

2 Objectivos Genéricos

O presente trabalho prático tem como objectivos genéricos:

1. A aquisição de competências de desenvolvimento de aplicações no Unity como ambiente de simulação;

2. A aquisição de competências relacionadas com a análise, desenvolvimento, implementação e teste de agentes reactivos autónomos.

Estes objectivos genéricos serão alcançados através do trabalho em grupo e da experimentação, promovendo-se, assim, estas capacidades.

3 Enunciado

Conforme o nome do trabalho prático deixa adivinhar, pretende-se personalizar, tanto ao nível funcional como estético do agente fornecido. Tal implica

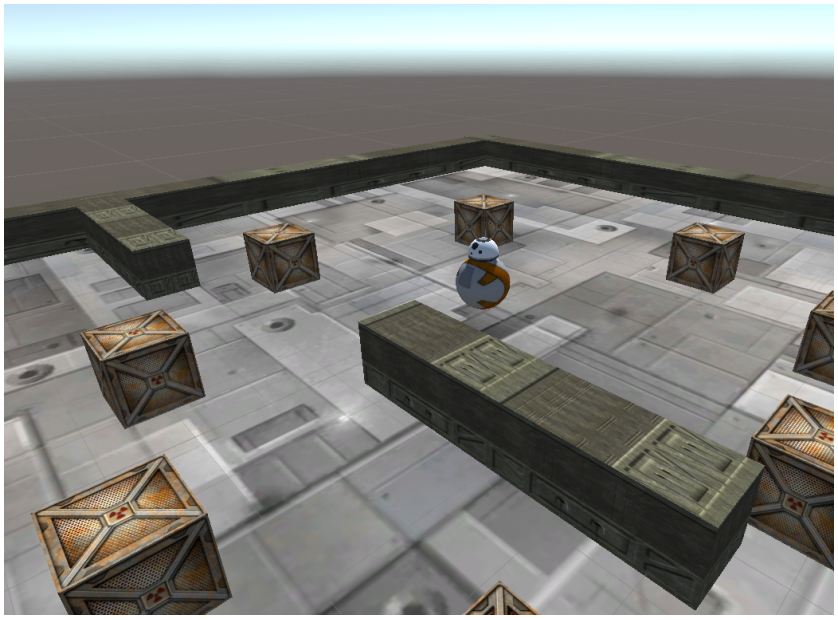

Figura 1: Exemplo do ambiente do agente D31.

expandir os "prefabs" distribuídos com o projecto, por forma a acrescentar novos sensores, obstáculos, etc...

O presente trabalho prático encontra-se dividido em 2 metas distintas:

1. Meta 1 – Sense it
2. Meta Final – Tune it & Test it

3.1 Meta 1 – Sense it

O prefab "d31-r" disponibilizado apenas prevê um tipo de sensor para recursos, que reage à existência dos mesmos no campo de visão do agente.

A primeira tarefa a desempenhar é expandir o código fornecido de forma a permitir diversos tipos de sensores de objectos. O novo agente deve suportar sensores:

1. Recursos[1]
2. Obstáculos

Estes sensores devem responder, respectivamente, a:

1. Recursos

[1]Este sensor já está implementado no prefab "d31-r" fornecido.

2. Blocos

A saída dos sensores é calculada em função do objecto que se encontra mais próximo do seu campo de visão, condicionado pela direção entre o agente D31 e o objecto.

Deve testar assim todas as funcionalidades através da construção de variantes do agente D31 que tirem partido destes sensores. Teste os agentes nos ambientes fornecidos **mapa1a** e **mapa1b** e outros que venha a criar, que ilustrem as diferentes funcionalidades.

3.2 Meta Final – Tune it & Test it

3.2.1 Tune it

Na versão original do agente "d31-r" a função de activação dos sensores é, por omissão, linear.

O cálculo da **energia** com base no sensor de recursos é feito da seguinte forma:

$$energia = \frac{1}{distancia(obj, sensor) + 1} \tag{1}$$

onde *obj* é o ponto mais próximo do objecto sensoriado.

Utilizando a função de activação linear (no código fornecido, a função *GetLinearOutput()*), a saída do sensor é a calculado utilizando a Eq. 1 sem qualquer alteração ou filtro.

No entanto, a função de activação linear impõe restrições sérias e impossibilita a programação de vários comportamentos interessantes. Deve proceder às alterações necessárias por forma a que seja possível especificar, **para cada sensor**:

1. A função de activação desejada linear, gaussiana ou logarítmica;
2. Limiar (*threshold*) de activação mínimo e máximo.
3. Limite superior e inferior;

Na figura 2 apresentam-se exemplos destas funções.

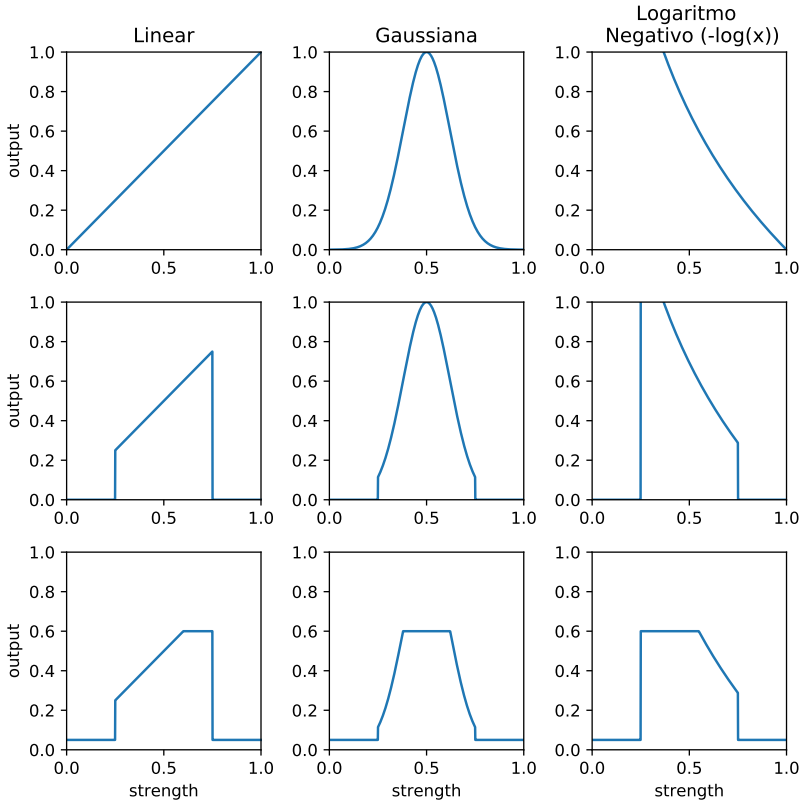

Figura 2: Exemplos de funções de activação, linear, gaussiana, negativo do logaritmo com diferentes parâmetros. A função gaussiana tem $\mu = 0.5$ e $\sigma = 0.12$. Na segunda linha são aplicados limiares de 0.25 e 0.75 (sobre os valores de x). Na terceira linha aplicamos os mesmo limiares, acrescentando limites de 0.05 e 0.6 (sobre os valores y).

Para além do cálculo de energia, existem parâmetros de configuração para cada sensor que deve considerar: (i) alcance do sensor (range of sensor); (ii) incremento no ângulo dos raios do sensor (angle of sensor), e.g. dado que os sensores actuam em redor do agente (360 graus), definir 10 neste parâmetro (valor por defeito no sensor fornecido) implica que os 360 graus em volta do sensor sejam divididos em raios a cada 10 graus (36 raios igualmente espaçados em redor do sensor).

3.2.2 Test it

Tirando partido dos diferentes tipos de parâmetros, funções de activação, limiares e limites, crie agentes que consigam resolver os mapas fornecidos:

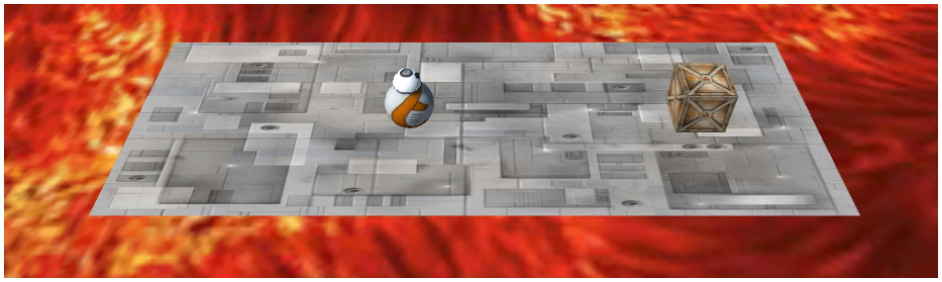

Figura 3: Colecionar o recurso mas sem cair!

Figura 4: Ambiente de recursos e obstáculos.

- map2a - O objectivo será colecionar a recursos sem que o agente caia da plataforma (figura 3).
- map2b - O objectivo será colecionar todas os recursos desviando-se dos blocos e sem cair da plataforma (figura 4). Neste mapa pode ajustar a posição inicial do agente dentro da zona com uma textura mais escura.

Através do uso de funções de activação, limiares e thresholds, **crie um agente** que explore o meio ambiente sem colidir com os blocos existentes. Não existe qualquer restrição quanto ao número de sensores ou funções de activação, deve no entanto tentar encontrar um bom compromisso entre desempenho e economia de recursos.

Para além de resolver os ambientes propostos, procure construir ambi-

entes que permitam ilustrar as propriedades dos diferentes agentes e que ponham à prova as suas capacidades. Analise o comportamento dos agentes indicando as suas forças e fraquezas. Pode ainda utilizar as informações fornecidas, tempo e recursos apanhados, para comparar os seus agentes.

4 Datas e Modo de Entrega

4.1 Meta 1 – Sense it

Material a entregar:

- O código desenvolvido até ao momento, devidamente comentado;
- Um breve documento (max. 3 páginas), em formato pdf, com a seguinte informação:
  - Identificação dos elementos do grupo (Nomes, Números de Estudante, e-mails, Turma(s) Prática(s))
  - Informação pertinente relativamente a esta meta, objectivos alcançados e dificuldades.

Modo de Entrega:

Entrega electrónica através do Inforestudante. **Data Limite: 1 de Março de 2020**

4.2 Meta Final – Tune it & Test it

Material a entregar:

- O código desenvolvido, devidamente comentado, para cada uma das metas;
- Um relatório (max. 10 páginas), em formato pdf, com a seguinte informação:
  - Identificação dos elementos do grupo (Nomes, Números de Estudante, e-mails, Turma(s) Prática(s))
  - Informação pertinente relativamente à globalidade do trabalho realizado

Num trabalho desta natureza o relatório assume um papel importante. Deve ter o cuidado de descrever detalhadamente todas as funcionalidades implementadas, dando particular destaque aos problemas e soluções encontradas. Deve ser fácil ao leitor compreender o que foi feito e ter por isso capacidade de adaptar/modificar o código.

Para cada agente desenvolvido, deve descrever o comportamento esperado e a forma como esse comportamento foi alcançado.

A **experimentação** é uma parte essencial do desenvolvimento de aplicações de IA. Assim, deve descrever detalhadamente as experiências realizadas, analisar os resultados, extrair conclusões e efectuar alterações (caso se justifique) em função dos resultados experimentais, por forma a optimizar o desempenho dos seus agentes.

O relatório deve conter informação relevante tanto da perspectiva do utilizador como do programador. Não deve ultrapassar as 10 páginas, formato A4. Todas as opções tomadas deverão ser devidamente justificadas e explicadas.

Modo de Entrega:

Entrega electrónica através do Inforestudante. **Data Limite: 15 de Março de 2020**.

5 Bibliografia

- **Inteligência Artificial: Fundamentos e Aplicações**
  Ernesto Costa, Anabela Simões
- **Vehicles: Experiments in Synthetic Psychology**
  Valentino Braitenberg