Trabalho Prático Nº1

Reactive D31: The AI Awakens

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE D COIMBRA

Fundamentos de Inteligência Artificial

Departamento de Engenharia Informática

2020/2021 - 2º Semestre

Nome	Nº	Email	PL
Eduardo Cruz	2018285164	eduardo.cruz@student.uc.pt	PL3
Gustavo Bizarro	2018298933	uc2018298933@student.uc.pt	PL9
Rodrigo Sobral	2018298209	uc2018298209@student.uc.pt	PL7

Contexto

Inicialmente, o agente ‘d31-r’ previa apenas um tipo de sensor, para recursos, que lhe permitia detetar um ou mais recursos num raio definido de visibilidade, selecionando e aproximando-se do mais próximo a uma velocidade inversamente proporcional a essa distância.

Na meta 1, o nosso objetivo era, através do código fornecido, desenvolver um sensor semelhante, mas relativo aos obstáculos existentes nos mapas (paredes), que permitisse ao agente, detetar e afastar-se suavemente do obstáculo mais próximo, evitando colisões e sem interferir no principal objetivo do agente, recolher, rapidamente, todos os recursos no mapa.

Dando continuidade ao trabalho desenvolvido na meta 1, na meta 2 desenvolvemos três funções de ativação, linear, gaussiana e logarítmica negativa, e implementámos a aplicação de limiares e limites ao input (strenght) e output destas funções. Estas funções foram aplicadas ao agente, posteriormente, tanto para os *resources* como para os obstáculos. Com o agente reativo desenvolvido, procedemos à exploração e recolha do conjunto de parâmetros que permitia obter uma melhor prestação temporal, em cada um dos mapas. Para esta tarefa, começámos por analisar cada uma das funções de ativação e, tendo em conta as dificuldades presentes em cada mapa, experimentámos soluções que melhor pareciam resolvê-las.

Objetivos Alcançados

Conseguimos realizar todos os objetivos estabelecidos para ambas as metas. Primeiramente começámos por criar e adicionar à Head do ‘d31-r’ um sensor a que chamámos ‘*WallsSensor*’, ao qual foi associado o script *SensorController.cs* e *BlockDetectorScript.cs*. Este último script, contém código equivalente ao encontrado em *ResourcesDetectorScript.cs* que está associado ao *ResourcesSensor,* no entanto o raio de visão que definimos para o sensor de obstáculos varia, assim como para o sensor de recursos. Considerámos que a variável *angleOfSensors* de ambos os sensores deveria permanecer com o valor 10, dado que permitem uma recolha eficiente e eficaz dos dados de ambos os sensores.

Na classe *LinearRobotUnitBehaviour*, adicionámos as variáveis *wallAngle* e *wallValue,* em que *wallAngle* armazena o ângulo entre o agente e o obstáculo mais próximo e o *wallValue* consiste no valor da energia (*strength*) multiplicada por *weightWall (*inferior a *weightResource).* Esta variável, assim como *weightResource*, permite definir um grau de prioridade para os obstáculos e para os recursos, respetivamente, fazendo com que o agente demonstre uma atitude de especial interesse pela recolha de recursos enquanto se afasta dos obstáculos. Estas variáveis traduzem-se na redução da energia com que o agente se afasta dos obstáculos, comparativamente com a energia com que se aproxima dos recursos. Na fase de

experimentação foram variados, entre outros, estes valores, para cada mapa, de modo a encontrarmos o melhor compromisso entre rapidez e eficiência na recolha dos recursos.

Através da implementação e utilização de outras funções de ativação, foi-nos possível resolver alguns mapas num tempo inferior ao conseguido apenas com a função de ativação linear, dado que estas funções permitem obter agentes com um comportamento completamente distinto, de função para função, e de limiar/limite para limiar/limite. Tanto os limiares como os limites desempenharam um papel fundamental na melhoria da performance do agente, por exemplo, foi apenas através da aplicação de limiares e limites nas funções de ativação, que conseguimos resolver o map2a, cuja maior dificuldade era conseguir evitar que o agente caísse na lava, fazendo com que ele parasse logo após ter recolhido o único resource do mapa.

Dificuldades Iniciais

Na meta 1 dedicámos algum tempo para compreendermos, na íntegra, o que o código disponibilizado nos vários ficheiros fazia e de que forma interagiam entre si e com a interface *Unity*. Após alguma análise em grupo conseguimos incrementar algum do código ao, até então, fornecido, e realizar as diversas configurações de interface de forma a obtermos algo funcional.

Posteriormente, reunindo informações dadas pelos docentes durante as aulas práticas de todos os membros do grupo e conseguimos aperfeiçoar o que tínhamos feito.

Já na meta final, as principais dificuldades passaram por tentarmos prever o tipo de implicações, que as funções de ativação implementadas tinham, na prática, nos diversos ambientes. Em alguns casos deparámo-nos com pontos do mapa em que o agente entrava num comportamento totalmente aleatório e imprevisível do qual não conseguia sair. Um dos nossos focos foi evitar que esse tipo de situações ocorresse.

Funcionalidades Implementadas

Ao longo deste trabalho prático desenvolvemos mecanismos que nos permitiram ter um agente somente reativo, sem qualquer tipo de memória ou decisão complementar além dos cálculos resultantes das funções de ativação, que recolhesse todos os *resources* existentes nos diversos mapas, afastando-se dos obstáculos e evitando cair do mapa, tudo isto claro, no melhor tempo possível.

Foi também criado um novo mapa, '*Extra_Map*', que se encontra na imagem abaixo, e que nos permitiu testar algumas das limitações do agente simples, tais como a ausência de resources nas proximidades, ou curvas que 'cortam' a visão do sensor do agente, o que nos ajudou a chegar a soluções melhoradas, mais conscientes.

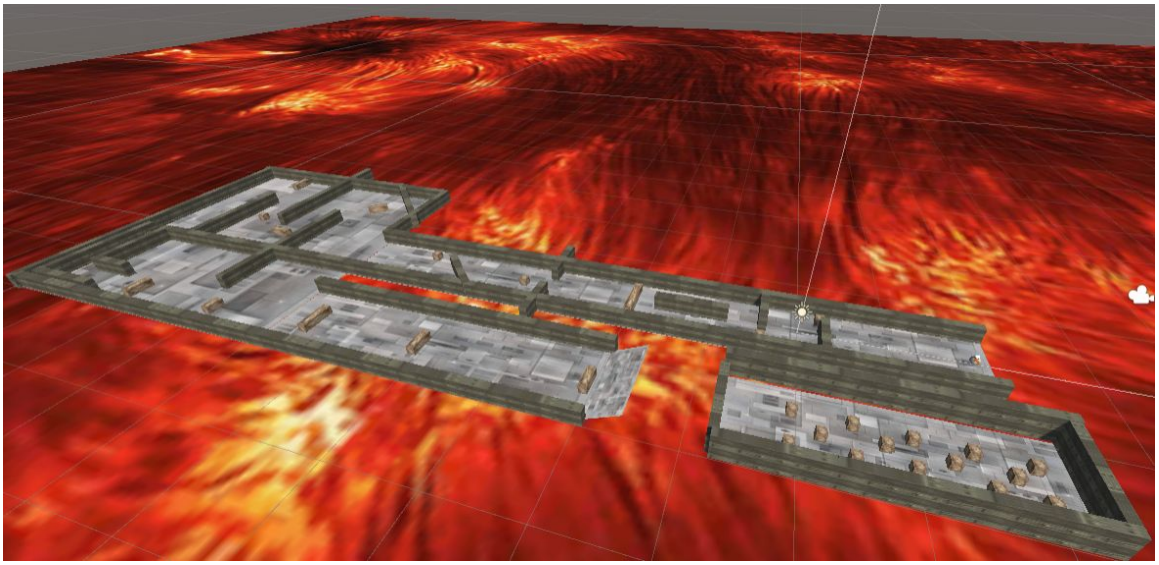

Figura 1 - *Extra_Map*

Abaixo apresentamos também o gráfico das três funções de ativação exploradas neste trabalho (sem limites ou limiares).

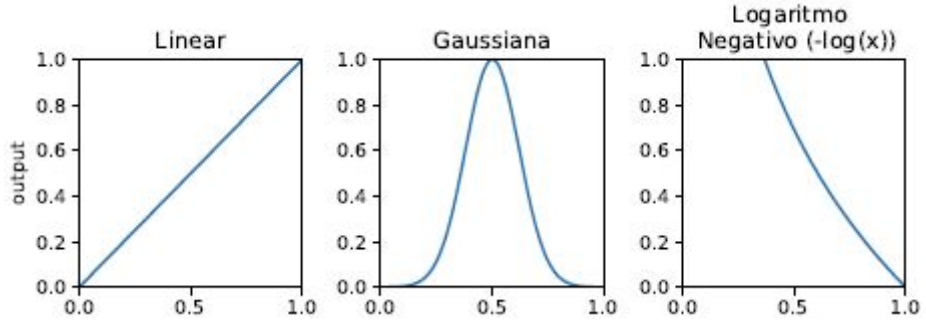

Figura 2 - Funções de Ativação

Soluções Encontradas

Para efetuarmos toda a experimentação, programámos tanto os sensores como o *LinearRobotUnit* de modo a permitirem, através do *Unity*, variar todos os parâmetros relevantes, de uma maneira mais simples. Ao clicarmos no componente *Body* em cada mapa, ao qual o ficheiro *LinearRobotUnit* está associado, podemos selecionar a função de ativação que pretendemos usar para cada sensor, assim como a prioridade *(weight)* de cada objeto. Para definirmos valores para os limites e limiares de cada função, entre outros parâmetros, é necessário selecionar o componente *ResourceSensor* ou *WallSensor,* existentes na *Head* do agente.

O cálculo da *strenght* é dado pela expressão seguinte:

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

Daqui temos que, quanto maior for a distância entre o sensor e o objeto, menor é a *strenght* do agente (proporcionalidade inversa). No caso em que o sensor do agente não estiver a detetar o objeto, por exemplo quando já todos os resources foram recolhidos, a *strength* assume o valor 0.

A partir desta observação e da análise dos gráficos das funções de ativação apercebemo-nos que, para *strenght*=0, situação em que o sensor não deteta qualquer resource, o output da função de ativação logarítmica negativa, assume o valor do limite superior, definido para o output, dado que log(x), com x a tender para 0 é igual a infinito. Para contornarmos este fator, que demonstrou ser um problema em alguns mapas, definimos um limiar inferior para a strength diferente de 0, por exemplo, 0.01.

Cada mapa exige um conjunto de parâmetros, por vezes, bastante distintos, dado que cada mapa apresenta particularidades que devem ser tidas em conta aquando da escolha desse conjunto de parâmetros a experimentar e a variar.

Como é visível nos resultados apresentados abaixo, existem parâmetros que se mantêm iguais de teste para teste. Isto justifica-se com o facto de, após várias experimentações com outros valores, termos percebido que esse conjunto de valores imutáveis permitiam obter uma solução mais eficiente e equilibrada em termos de performance/tempo. A variação do grau de prioridade atribuído ao sensor de cada objeto (resource ou wall) demonstrou, em alguns casos, ser bastante efetiva na obtenção de um melhor resultado, dado que este parâmetro afeta diretamente o 'vetor' de força aplicado ao agente.

Resultados Levantados

Os seguintes dados são fruto do levantamento do melhor resultado obtido para todas as combinações de funções de ativação.

Para limites e limiares, variamos naturalmente os limites inferiores e superiores do *x* e *y* das funções, que são, neste contexto, a *strength* aplicada no agente e o *output* da função, respetivamente. No caso particular da função gaussiana podemos variar o micro e sigma, que faz com que o pico da função seja mais ou menos precoce. No entanto decidimos manter os valores 0,5 e 0,12 respetivamente pois são aqueles que fornecem resultados mais consistentes em todos os mapas. Além destes são usados os *weights* para definir a prioridade que as paredes e os recursos têm no cálculo do *output*, o *angle* para definir o ângulo entre os sensores (isto é particularmente essencial no *extra_map*, onde há muitas esquinas e devemos manter um ângulo baixo para o agente ter visão para um recurso "quase escondido" pela parede) e por fim o *range* dos sensores, cuja manipulação também se torna crucial em mapas de maiores dimensões onde, precisamos dum *range* maior para os recursos terem impacto na movimentação do agente.

map1a

func. ativacao w / r	weight w / r	angle sens w / r	range sens w / r	lim. inf. out(y) w / r	lim. sup. out(y) w / r	lim. inf. stren(x) w / r	lim. sup. stren(x) w / r	gauss micro w / r	gauss sigma w / r	tempo (s)
linear / linear	-1.5 / 2	10 / 10	10 / 30	0 / 0.1	1 / 1	0 / 0.1	1 / 1	-- / --	-- / --	7
linear / gauss	-1.5 / 2	10 / 10	10 / 30	0 / 0.1	1 / 1	0 / 0.1	1 / 1	-- / 0.5	-- / 0.12	7
linear / log	-0.75 / 1	10 / 10	10 / 30	0 / 0.1	1 / 1	0 / 0.1	1 / 1	-- / --	-- / --	9
--	--	--	--	--	--	--	--	--	--	--
gauss / gauss	-1.5 / 2	10 / 10	10 / 30	0 / 0.1	1 / 1	0 / 0.1	1 / 1	0.5 / 0.5	0.12 / 0.12	5
gauss / linear	-0.75 / 1	10 / 10	10 / 30	0 / 0.1	1 / 1	0 / 0.1	1 / 1	0.5 / --	0.12 / --	8
gauss / log	-1.5 / 2	10 / 10	10 / 30	0 / 0.1	1 / 1	0 / 0.1	1 / 1	0.5 / --	0.12 / --	4
--	--	--	--	--	--	--	--	--	--	--
log / log	-1.5 / 2	10 / 10	10 / 30	0 / 0.1	1 / 1	0 / 0.1	1 / 1	-- / --	-- / --	4
log / gauss	-0.5 / 2	10 / 10	10 / 30	0 / 0.1	1 / 1	0 / 0.1	1 / 1	-- / 0.5	-- / 0.12	6
log / linear	-0.5 / 0.75	10 / 10	10 / 30	0 / 0.1	1 / 1	0 / 0.1	1 / 1	-- / --	-- / --	9

O melhor resultado pertence à combinação das funções gaussiana com logarítmica nas paredes e recursos, respetivamente, e logarítmica com logarítmica. Isto deve-se ao ambiente em causa, neste caso trata-se apenas dum campo pequeno e fechado, no qual não há obstáculos e a velocidade do agente a seguir os recursos e afastar-se das paredes à sua volta assume um fator muito importante, para que consiga, coletar os 3 recursos disponíveis por ordem crescente de distância. Desta forma percebe-se que os melhores resultados se obtêm evitando funções lineares, pois estas não oferecem crescimentos abruptos de velocidade.

map1b

func. ativacao w / r	weight w / r	angle sens w / r	range sens w / r	lim. inf. out(y) w / r	lim. sup. out(y) w / r	lim. inf. stren(x) w / r	lim. sup. stren(x) w / r	gauss micro w / r	gauss sigma w / r	tempo (s)
linear / linear	-0.6 / 2.5	10 / 10	10 / 50	0 / 0	1 / 1	0 / 0	1 / 1	-- / --	-- / --	10
linear / gauss	-0.5 / 2	10 / 10	10 / 50	0 / 0	0.4 / 1	0 / 0	1 / 1	-- / 0.6	-- / 0.12	10
linear / log	-0.5 / 2	10 / 10	10 / 50	0 / 0.05	1 / 0.3	0 / 0.1	1 / 1	-- / --	-- / --	10
--	--	--	--	--	--	--	--	--	--	--
gauss / gauss	-0.5 / 2	10 / 10	10 / 50	0 / 0	0.6 / 1	0 / 0	1 / 1	0.7 / 0.3	0.12 / 0.12	9
gauss / linear	-0.5 / 2	10 / 10	10 / 50	0.1 / 0	1 / 1	0 / 0	1 / 1	0.8 / --	0.12 / --	10
gauss / log	-0.5 / 2	10 / 10	10 / 50	0.1 / 0.1	1 / 1	0 / 0.05	1 / 1	0.8 / --	0.12 / --	9
--	--	--	--	--	--	--	--	--	--	--
log / log	-0.5 / 3	10 / 10	10 / 50	0 / 0.15	0.6 / 1	0.4 / 0.25	1 / 1	--/ --	-- / --	8
log / gauss	-0.5 / 2	10 / 10	10 / 50	0.2 / 0	1 / 1	0 / 0	0.25 / 0.2	-- / 0.4	-- / 0.12	11
log / linear	-0.5 / 2	10 / 10	10 / 50	0.25 / 0	1 / 1	0 / 0	1 / 0.2	-- / --	-- / --	10

Apesar de também pequeno e fechado, ao contrário do anterior, possui alguns obstáculos, mas não os suficientes para fazer com que os resultados sejam muito distintos do mapa *map1a* e, por isso, as funções logarítmicas continuam a liderar.

map2a

Nota: Aqui não temos paredes, logo só precisamos da função de ativação, limites e limiares relativos aos sensores de recursos (weight w=0 ; weight r=1)

funcao de ativacao	weight	angle sens	range sens	limite inf out(y)	limite sup out(y)	limite inf stren(x)	limite sup stren(x)	gauss micro	gauss sigma	tempo(s)
linear	1	10	30	0.1	0.2	0	0.02	--	--	7
gauss	1	10	30	0.1	0.2	0	0.03	0.5	0.12	7
log	1	10	30	0.1	0.2	0	0.02	--	--	7

Por não possuir qualquer tipo de obstáculo, incluindo paredes, foi o mais peculiar de experimentar, uma vez que, se os limites e limiares não fossem bem definidos, facilmente o agente caía do mapa. Os tempos resultantes foram todos iguais porque, pela sua simplicidade, as diferenças entre funções não são significativas, fazendo com que os limites se tornem o foco de manipulação nesta experimentação. Estes por sua vez não são valores muito grandes nem se alteram muito, já que não podemos ter grandes variações nem grandes velocidades, senão o agente cai.

map2b

func. ativacao w / r	weight w / r	angle sens w / r	range sens w / r	lim. inf. out(y) w / r	lim. sup. out(y) w / r	lim. inf. stren(x) w / r	lim. sup. stren(x) w / r	gauss micro w / r	gauss sigma w / r	tempo (s)
linear / linear	-0.7 / 1.5	10 / 5	10 / 100	0 / 0.1	1 / 1	0 / 0.1	1 / 1	-- / --	-- / --	25
linear / gauss	-0.6 / 1	5 / 10	10 / 100	0 / 0.2	1 / 2	0 / 0.1	1 / 1	-- / 0.5	-- / 0.12	41
linear / log	-0.6 / 1	5 / 10	10 / 100	0 / 0.2	1 / 2	0 / 0.1	1 / 1	-- / --	-- / --	30
--	--	--	--	--	--	--	--	--	--	--
gauss / gauss	-0.3 / 1.5	10 / 5	10 / 100	0 / 0.1	1 / 1	0 / 0.1	1 / 1	0.5 / 0.5	0.12 / 0.12	28
gauss / linear	-0.3 / 1.5	10 / 5	10 / 100	0 / 0.1	1 / 1	0 / 0.1	1 / 1	0.5 / --	0.12 / --	43
gauss / log	-0.3 / 0.8	10 / 5	10 / 100	0 / 0.1	1 / 1	0 / 0.1	1 / 1	0.5 / --	0.12 / --	57
--	--	--	--	--	--	--	--	--	--	--
log / log	-0.5 / 2	10 / 10	10 / 50	0 / 0.4	0.4 / 0.8	0.01 / 0.2	1 / 1	-- / --	-- / --	22
log / gauss	-0.2 / 1.5	10 / 5	10 / 100	0 / 0.1	1 / 1	0 / 0.1	1 / 1	-- / 0.5	-- / 0.12	32
log / linear	-0.5 / 2	10 / 10	10 / 50	0.01 / 0	0.5 / 1	0 / 0	1 / 1	-- / --	-- / --	28

Este foi, a nosso ver, o mapa mais difícil de testar, uma vez que por ser grande e estruturalmente complexo, faz com que uma pequena alteração cause uma diferença enorme de movimentação do agente ao longo do mapa. Por isso precisámos de encontrar um par de *weights* que fizesse o agente priorizar sempre os recursos, mas sem ganhar muita velocidade para não cair do mapa ou perder o seu caminho padrão e consequentemente deixar um resource para trás ou ficar preso num certo espaço e não recolher o resto dos resources.

A melhor combinação de funções de ativação é a linear / linear, já que, apesar de não resultar no melhor tempo, é a mais robusta, isto é, aquela que nos dá mais garantias de sucesso, isto porque, mantendo crescimentos e decrescimentos constantes de velocidade, o agente consegue facilmente passar pelas paredes sem

sofrer fortes repulsões e seguir os recursos sem se desviar do caminho padrão do mapa.

extra_map

func. ativacao w / r	weight w / r	angle sens w / r	range sens w / r	lim. inf. out(y) w / r	lim. sup. out(y) w / r	lim. inf. stren(x) w / r	lim. sup. stren(x) w / r	gauss micro w / r	gauss sigma w / r	tempo (s)
linear / linear	-0.5 / 3	10 / 5	10 / 100	0 / 0.1	1 / 1	0 / 0.1	1 / 1	-- / --	-- / --	51
linear / gauss	-0.5 / 2	10 / 5	10 / 100	0 / 0.1	1 / 1	0 / 0.1	1 / 1	-- / 0.5	-- / 0.12	71
linear / log	-0.5 / 1	10 / 5	10 / 100	0 / 0.1	1 / 1	0 / 0.1	1 / 1	-- / --	-- / --	49
--	--	--	--	--	--	--	--	--	--	--
gauss / gauss	-0.4 / 2.5	10 / 5	10 / 100	0 / 0.1	1 / 1	0 / 0.1	1 / 1	0.5 / 0.5	0.12 / 0.12	59
gauss / linear	-0.3 / 3	10 / 5	10 / 100	0 / 0.1	1 / 1	0 / 0.1	1 / 1	0.5 / --	0.12 / --	58
gauss / log	-0.3 / 3	10 / 5	10 / 100	0 / 0.1	1 / 1	0 / 0.1	1 / 1	0.5 / --	0.12 / --	53
--	--	--	--	--	--	--	--	--	--	--
log / log	-0.4 / 1	10 / 5	10 / 100	0 / 0.1	1 / 1	0.1 / 0.1	1 / 1	-- / --	-- / --	46
log / gauss	-0.2 / 2	10 / 5	10 / 100	0 / 0.1	1 / 1	0.1 / 0.1	1 / 1	-- / 0.5	-- / 0.12	51
log / linear	-0.4 / 2.8	10 / 5	10 / 100	0 / 0.1	1 / 1	0.1 / 0.1	1 / 1	-- / --	-- / --	73

Por ser um mapa com várias esquinas, curvas e contracurvas, espaços estreitos, afunilamentos e até rampas, torna-se um mapa com muita variedade de reações pretendidas por parte do agente.

Para a realização deste mapa começámos por diminuir o ângulo dos sensores dos recursos. Desta forma, quando o agente ficasse numa esquina na qual teria pouca visão para o recurso, com um ângulo pequeno, ele iria facilmente detetá-lo e prosseguia o seu percurso. Relativamente aos *weights*, foi preciso definir uma valor muito alto para os recursos e fazer com que o agente consiga adquirir velocidade suficiente para saltar a rampa e cair no local pretendido, e, um valor muito mais baixo para as paredes, para fazer com que o agente desse sempre muito mais prioridade aos recursos e não recuasse nos afunilamentos do percurso.

Conclusões

Com este projeto, concluímos que um agente reativo simples é bastante limitado para aplicações que exijam movimentos mais exigentes, que requerem memória, por exemplo, ou até mesmo tomadas de decisão.

Para trabalhos mais complexos, em vez de um agente reativo, devemos implementar um agente decisivo, com memória associada, decisões baseadas em redes neurais e capacidade de auto aprendizagem.