Licenciatura em Engenharia Informática e Multimédia
2º Semestre Letivo 2021/2022

Interação em Ambientes Virtuais

Projeto Final

Docente:
Arnaldo Abrantes

Autores:
42356, Érica Pereira, 61D

Lisboa, 17 de julho de 2022

Índice

1. Introdução

Este trabalho foi desenvolvido em *Unity* (2020.3.0f1 – LTS) e *Visual Studio*, na linguagem de programação *C#*.

O trabalho é baseado no jogo *Minecraft* – um jogo de blocos tridimensional de interação com o ambiente virtual, com foco nos temas de geração procedimental de conteúdos, inteligência artificial e interação pessoa-máquina.

Este projeto engloba trabalhos anteriormente realizados, de cada um dos temas, onde se tinha dado como atingidos os seguintes objetivos:

- Para a geração procedimental de conteúdos, conseguiu-se gerar procedimentalmente um ambiente virtual semelhante ao mundo real, afinando os parâmetros dos geradores de *Ruído de Perlin* para obter uma distribuição dos blocos mais realista;

- Adicionou-se outros tipos de blocos para além dos blocos de terra, relva, pedra e ar, do tipo carvão, granito, diamante, flor, erva, raiz de árvore, madeira e folhas;

- Para a interação pessoa-máquina, utilizou-se o rato para destruir e construir blocos no ambiente, o teclado para ativar funcionalidades no mundo e o microfone para interagir com o mundo virtual, associando a deteção de palmas no áudio do microfone para alterar a textura/tipo dos blocos no ambiente e a deteção de assobios para criar e crescer uma árvore no ambiente.

Neste trabalho, tem-se como objetivos:

- No tema de inteligência artificial, criar e treinar um agente a visitar todas as peças no ambiente em ordem crescente, similar a um dos exemplos do github/*mlagents;*

- No tema de interação pessoa-máquina, adicionar a interação com o rato de destruir e construir, aos blocos do tipo *Untouchables*, como por exemplo, aos blocos de flor e erva;

- Adicionar o conceito de dia/noite ao mundo virtual, através da luz do telemóvel.

2. Resultados Obtidos

2.1. Mundo *Minecraft* – Geração Procedimental de Conteúdos

2.1.1. Superfície

Resumindo o primeiro trabalho, para tornar mais realista a distribuição dos blocos do mundo a construir, primeiramente, incrementou-se um dos parâmetros dos geradores de *Ruído de Perlin*: o número de oitavas e, posteriormente, ajustou-se a suavidade e a persistência (oitavas: 8, persistência: 0.87, suavidade: 0.00055).

Colocar a persistência com um valor menor e próximo de 100%, serviu para dar mais importância às primeiras oitavas, produzindo um terreno com mais montanhas do que planícies (Figura 1).

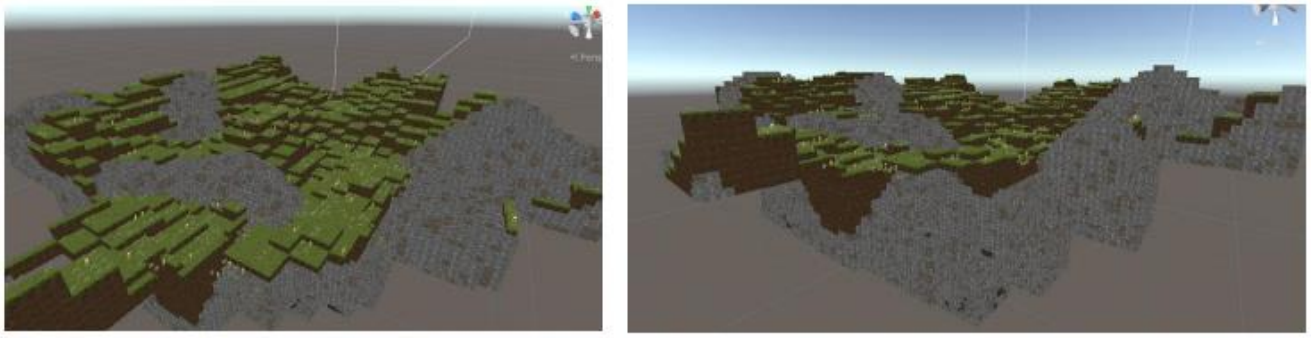

Figura 1 - Mundo virtual gerado procedimentalmente com o auxílio do Ruído de Perlin.

2.1.2. Montanhas de pedra

Utilizou-se um *Ruído de Perlin* diferente para os blocos de pedra, para criar montanhas de pedras à superfície (oitavas: 5, persistência: 1.04, suavidade: 0.00275).

2.1.3. Grutas

Foram estabelecidas grutas, ou seja, blocos de ar, entre os blocos de pedra, usando um *Ruído de Perlin* 3D com os parâmetros – oitavas: 5, persistência: 1.7, suavidade: 0.00165. Sendo que o *Ruído de Perlin* devolve valores entre 0 e 1, determinou-se o limiar de decisão de 0.51, em que todos os valores abaixo deste são representantes de grutas e os restantes são blocos de pedra.

2.1.4. Diamantes

Os blocos de diamantes só aparecem perto das grutas, fazendo uma orla de diamantes à volta do limiar de decisão das grutas. Para não haver blocos de diamantes juntos, em grandes quantidades, adicionou-se um outro *Ruído de Perlin 3D* (oitavas: 11, persistência: 1.5, suavidade: 0.00165), para determinar um posicionamento mais disperso dos diamantes inseridos dentro da orla criada em volta das grutas. Aparecem blocos de diamante quando o *Ruído de Perlin 3D* retorna valores entre 0.44 e 0.4300995 (Figura 2).

2.1.5. Carvão

Os blocos de carvão estão dispersos pelos blocos de pedra (Figura 2), com um Ruído de Perlin 3D próprio, com os seguintes parâmetros – oitavas: 7, persistência: 2.4, suavidade: 0.00165.

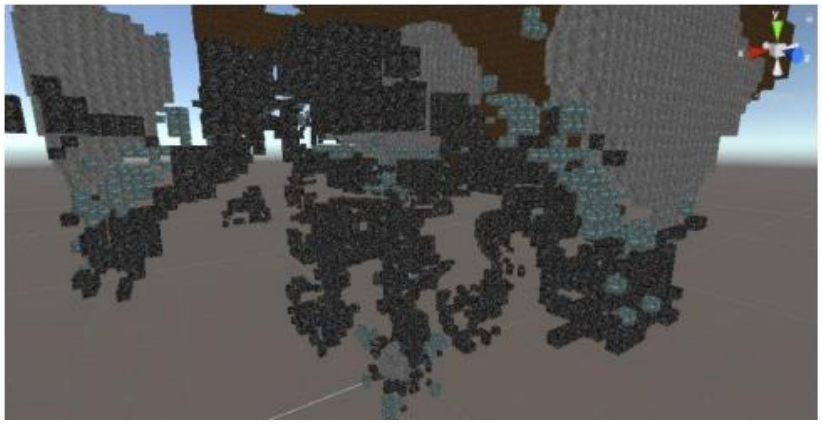

Figura 2 - Orla de diamantes depois do *Ruído de Perlin* 3D com 2 limiares de decisão e blocos de carvão com *Ruído de Perlin* 3D (sem os blocos de pedra envolventes para fins demonstrativos).

2.1.6. Flores e Erva daninha – *Untouchable Chunks*

Os blocos de erva daninha foram gerados através de um *Ruído de Perlin 3D* (oitavas: 10, persistência: 1.2, suavidade: 0.00165), com dois limiares de decisão, para obter pequenos 'grupos' de ervas daninha espalhados pelos *Chunk's*. Foram atribuídos mais 2 limiares de decisão no mesmo *Ruído de Perlin* para distribuir blocos de flores por entre a erva daninha (Figura 3).

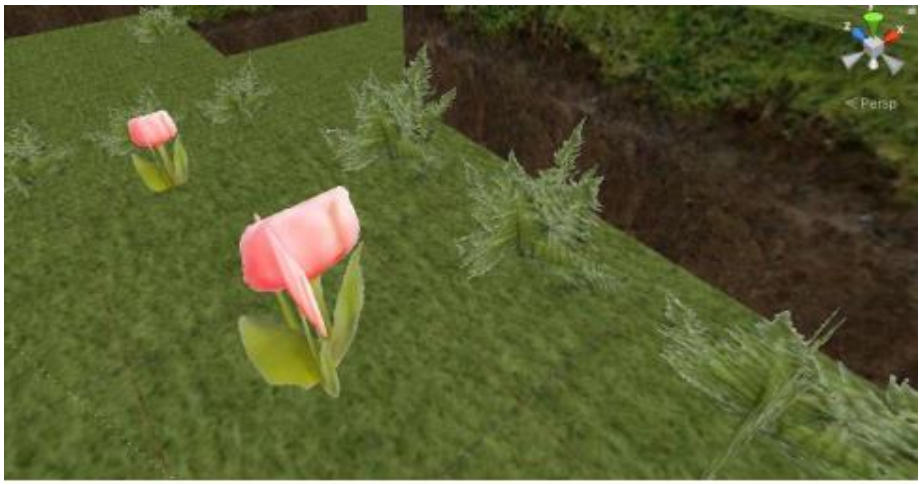

Figura 3 - Blocos de flor e de erva daninha.

Este tipo de bloco pertence a uma classe derivada do *Chunk*: *UntouchableChunk*. Este 'filho' do *Chunk* é constituído somente por blocos de erva daninha, flores e ar, e não possui um *Collider*, o que permite que o utilizador passe livremente por entre este tipo de blocos.

2.2. MLAgents: Inteligência Artificial

Criou-se para este tema um ambiente circular com 14 telhas numeradas aleatoriamente (Figura 4). O objetivo do agente fazendeiro é pressionar todas as peças da sala circular por ordem crescente.

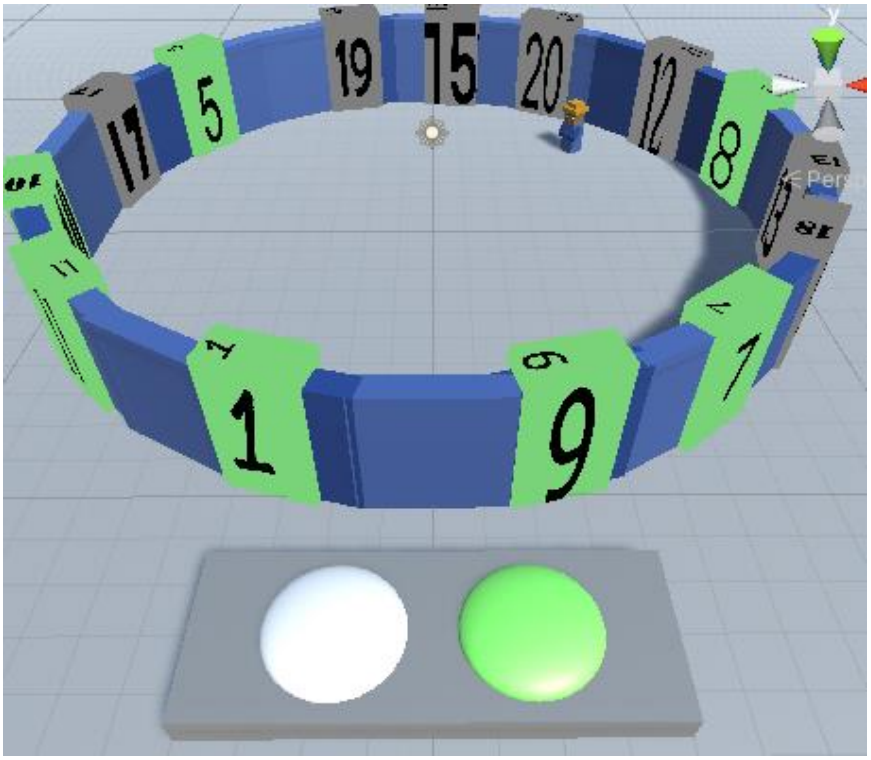

Figura 4 - Sala circular com telhas numeradas aleatoriamente para treinar 1 agente.

Os valores das telhas são aleatórios entre 1 e 20. Estes valores mudam aleatoriamente a cada episódio. Quando o Agente pressiona uma telha, esta fica verde. O episódio acaba quando todas as telhas tenham sido pressionadas corretamente ou quando o tempo acaba.

2.2.1. Recompensas

As recompensas extrínsecas do agente são: +1 por pressionar a telha certa, -1 por visitar a errada e -0.0002 por pena existencial.

2.2.2. Observações

As observações no ambiente são 4 por cada telha presente na sala circular: 2 *floats* para as coordenadas (x, z) da direção do agente à telha, 1 *int* para o valor do número atribuído à telha e 1 booleano para se a telha já foi pressionada ou não.

Para 14 telhas presentes, o número de observações é de: 14 x 4 = 56 observações.

2.2.3. Ações do Agente

O agente possui 3 ações discretas, que correspondem:

- ao movimento para a frente, para trás ou fazer nada, tomando a primeira ação 3 possíveis valores;
- ao movimento para a direita, para a esquerda ou fazer nada, tomando a segunda ação 3 possíveis valores;
- à ação de pressionar ou não pressionar uma telha, tomando a terceira ação 2 possíveis valores.

2.2.4. Treino

Fez-se inicialmente um treino com aprendizagem por imitação, onde se realizaram 24 episódios com uma recompensa média de 12.14.

Utilizaram-se 9 agentes para treinar a rede neuronal simultaneamente (Figura 5). Após alguns *step's*, notou-se que a tendência dos agentes era estagnar as recompensas, não mexendo do mesmo sítio (Figura 6).

Concluiu-se que o agente 'optou' por receber uma média de recompensas de -2.000, sendo esta obtida quando o episódio atinge o seu tempo limite, ou seja, quando o agente alcança o número máximo de *step's* por episódio.

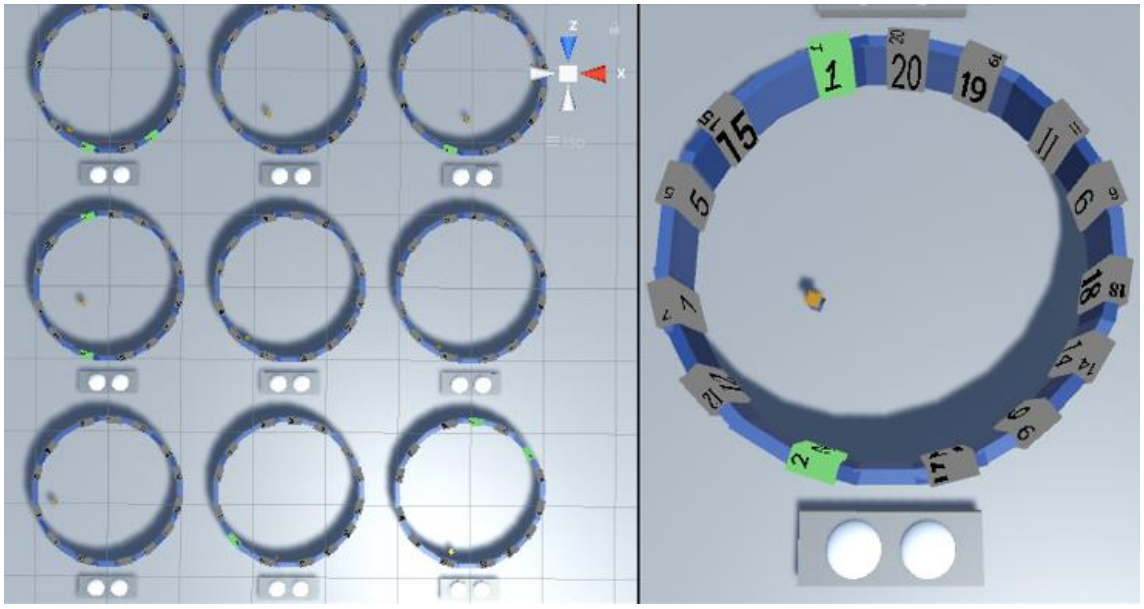

Figura 5 – 9 agentes a serem treinados simultaneamente.

```
[INFO] Sorter14. Step: 10000. Time Elapsed: 54.385 s. No episode was completed since last summary. Training.
[INFO] Sorter14. Step: 20000. Time Elapsed: 99.297 s. Mean Reward: -258.333. Std of Reward: 82.723. Training.
[INFO] Sorter14. Step: 30000. Time Elapsed: 141.132 s. No episode was completed since last summary. Training.
[INFO] Sorter14. Step: 40000. Time Elapsed: 183.167 s. Mean Reward: -25.667. Std of Reward: 11.245. Training.
[INFO] Sorter14. Step: 50000. Time Elapsed: 222.189 s. No episode was completed since last summary. Training.
[INFO] Sorter14. Step: 60000. Time Elapsed: 263.021 s. Mean Reward: -3.111. Std of Reward: 2.079. Training.
[INFO] Sorter14. Step: 70000. Time Elapsed: 300.387 s. No episode was completed since last summary. Training.
[INFO] Sorter14. Step: 80000. Time Elapsed: 337.636 s. Mean Reward: -3.222. Std of Reward: 2.043. Training.
[INFO] Sorter14. Step: 90000. Time Elapsed: 376.605 s. Mean Reward: -2.000. Std of Reward: 0.000. Training.
[INFO] Sorter14. Step: 100000. Time Elapsed: 419.402 s. Mean Reward: -2.000. Std of Reward: 0.000. Training.
[INFO] Sorter14. Step: 110000. Time Elapsed: 461.003 s. Mean Reward: -2.556. Std of Reward: 1.571. Training.
[INFO] Sorter14. Step: 120000. Time Elapsed: 501.262 s. No episode was completed since last summary. Training.
[INFO] Sorter14. Step: 130000. Time Elapsed: 543.484 s. Mean Reward: -2.000. Std of Reward: 0.000. Training.
[INFO] Sorter14. Step: 140000. Time Elapsed: 584.092 s. No episode was completed since last summary. Training.
[INFO] Sorter14. Step: 150000. Time Elapsed: 624.198 s. Mean Reward: -2.000. Std of Reward: 0.000. Training.
[INFO] Sorter14. Step: 160000. Time Elapsed: 663.221 s. No episode was completed since last summary. Training.
[INFO] Sorter14. Step: 170000. Time Elapsed: 716.300 s. Mean Reward: -2.000. Std of Reward: 0.000. Training.
[INFO] Sorter14. Step: 180000. Time Elapsed: 762.727 s. Mean Reward: -2.000. Std of Reward: 0.000. Training.
[INFO] Sorter14. Step: 190000. Time Elapsed: 803.696 s. Mean Reward: -2.000. Std of Reward: 0.000. Training.
[INFO] Sorter14. Step: 200000. Time Elapsed: 842.383 s. Mean Reward: -2.000. Std of Reward: 0.000. Training.
[INFO] Sorter14. Step: 210000. Time Elapsed: 880.658 s. No episode was completed since last summary. Training.
[INFO] Sorter14. Step: 220000. Time Elapsed: 925.159 s. Mean Reward: -2.000. Std of Reward: 0.000. Training.
[INFO] Sorter14. Step: 230000. Time Elapsed: 968.025 s. No episode was completed since last summary. Training.
[INFO] Sorter14. Step: 240000. Time Elapsed: 1009.944 s. Mean Reward: -2.000. Std of Reward: 0.000. Training.
[INFO] Sorter14. Step: 250000. Time Elapsed: 1049.282 s. No episode was completed since last summary. Training.
[INFO] Sorter14. Step: 260000. Time Elapsed: 1092.345 s. Mean Reward: -2.000. Std of Reward: 0.000. Training.
```

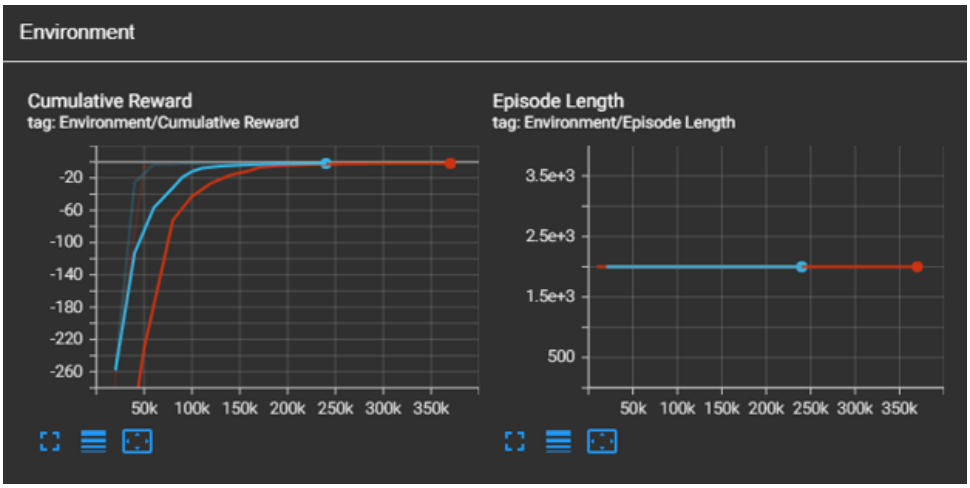

Figura 6 – Resultados obtidos na linha de comandos e os gráficos do TensorBoard da recompensa acumulativa, estagnando no -2 e o tempo que leva o episódio a acabar.

2.3. Interação Pessoa-Máquina

2.3.1. Destruir e contruir blocos

Resumindo o terceiro trabalho, o utilizador pode interagir com o mundo virtual *Minecraft* destruindo blocos já existentes ou construindo novos blocos, utilizando o rato. O bloco ou a posição do novo bloco é adquirido através de *RayCast*.

2.3.1.1. Untouchable Blocks

Neste trabalho, acrescentou-se a interação de destruir e construir blocos *Untouchables*, ou seja, os blocos por onde o utilizador passa sem tocar, como por exemplo, blocos de flor e blocos de erva daninha.

2.3.1.2. Barra de Ferramentas

Ao contruir um novo bloco no mundo virtual é necessário identificar primeiro o tipo de bloco pretendido. Para isso, neste trabalho criou-se uma barra de ferramentas na interface *Canvas* (Figura 6), com 8 tipos de blocos possíveis. A escolha do bloco é feita através da roda do rato.

No trabalho anterior, adicionaram-se 5 *Texts*, para o utilizador ir recebendo notificações das ações relacionadas com o microfone (Figura 6).

Figura 6 - Interface do GameObject Canvas, com uma barra de ferramentas de 8 tipos de bloco diferentes e 5 texts com mensagens informativas.

2.3.2. Microfone: Assobios

Resumindo o trabalho anteriormente feito, a interação realizada através da deteção de assobios com o microfone baseia-se na criação e geração de uma árvore, constituída por um conjunto empilhado de blocos do tipo madeira que formam o tronco, e blocos de folha aleatoriamente dispostas na parte de cima da árvore (Figura 7).

Uma árvore só pode ser criada se a câmara do utilizador estiver a apontar para um bloco do tipo raiz ou, se já criou uma árvore onde estava a raiz, estiver a apontar para o primeiro bloco de madeira a partir do chão, da árvore acabada de criar.

Após uns testes, concluiu-se que os assobios costumam situar-se acima dos 40 decibéis, com um pico de frequência entre os 1100 e os 2300 Hz e uma concentração à volta do pico de frequência maior do que 90%. A altura da árvore criada é dada pelo pico de frequência obtido do assobio detetado. A geração das folhas da árvore é feita com o auxílio do algoritmo *Flood Fill*.

Figura 7 – Árvore gerada com assobio de pico de frequência igual a 1850 Hz, que resultou numa árvore de 15 blocos de altura.

2.3.3. Microfone: Palmas

A interação feita com a deteção de palmas consiste na alteração aleatória do tipo de bloco, do bloco para onde a câmara do utilizador está a apontar (Figura 8). Após uns testes, concluiu-se que as palmas costumam situar-se entre os 30 e os 60 decibéis, com um pico de frequência entre os 1000 e os 3000 Hz e uma concentração à volta do pico de frequência entre os 20 e os 50%.

A taxa de sucesso da deteção de uma palma no áudio do microfone é satisfatória, não capturando uma ou outra palma em 20 tentativas, porém, sons do tipo gargalhadas, risos ou alguns sons da televisão são por vezes detetados como uma palma, não sendo este o resultado ideal desta deteção.

Figura 8 – Blocos com as texturas alteradas aleatoriamente pela deteção de palmas no som do microfone.

Neste trabalho, acrescentou-se uma música ambiente ao mundo virtual, que deixa de tocar quando o microfone é ligado e retoma a tocar quando o microfone é desligado.

2.3.4. OSC: Luminosidade

Neste trabalho, acrescentou-se a interação do utilizador com a luz detetada no telemóvel. Para isto, utilizou-se a aplicação móvel *OscHook v2*, para detetar valores de luminosidade.

Os valores obtidos de luminosidade, detetados pela aplicação, foram utilizados de maneira a criar o conceito de dia/noite no ambiente virtual. Com 2 limiares de decisão, determinou-se as noções de dia – tarde – noite, sendo considerado 'noite' quando o valor de luminosidade se encontra abaixo de 26, considerado 'dia' quando se encontra acima de 499 e considerado 'tarde' quando se encontra entre estes 2 limiares.

Este efeito foi concebido alterando a rotação em X da *Directional Light* (Figura 9).

Figura 9 - Ambiente virtual com dia, tarde e noite.

3. Conclusões

3.1. Geração Procedimental de Conteúdos

Os objetivos deste tema foram atingidos, com a exceção de certos *Ruídos de Perlin* realizados. O ideal seria diminuir o número de oitavas e, por consequente, alterar os outros parâmetros para obter resultados semelhantes aos atuais, uma vez que a quantidade de oitavas em certos *Ruídos de Perlin* acaba por ter uma carga computacional desnecessariamente alta.

3.2. Inteligência Artificial

O agente tendia a não se mexer de um dado sítio. É possível que, a quantidade de recompensas negativas que o agente recebia ao errar a telha a pressionar era tão grande que a melhor opção era acabar o episódio com a penalização existencial possível.

Uma hipótese para resolver este problema possivelmente seria penalizar o agente ao tocar na parede com uma recompensa negativa ou acabar o episódio. Ou tentar treinar mais o agente com aprendizagem por imitação, fazendo mais episódios.

Também gostaria de ter incorporado o agente com o mundo virtual criado de forma a interagir com este, como por exemplo, depois de o agente concluir o objetivo de pressionar todas as telhas por ordem crescente, era gerada uma galinha no mundo virtual.

3.3. Interação Pessoa-Máquina

Todos os objetivos do trabalho foram alcançados, com exceção do aperfeiçoamento da deteção de palmas. Este foi um objetivo que não chegou a ser alcançado desde o último trabalho.

Também seria interessante mexer o jogador ou um outro personagem através do acelerómetro, movimentado o personagem em questão com a movimentação do telemóvel.

4. Webgrafia

- Vídeos disponibilizados para a cadeira de Interação em Ambientes Virtuais;

- Exemplo “Sorter” dos exemplos de mlagents do site *github.com/Unity-Tecnologies:*

https://github.com/Unity-Technologies/ml-agents/blob/release_16_docs/docs/Learning-Environment-Examples.md#sorter