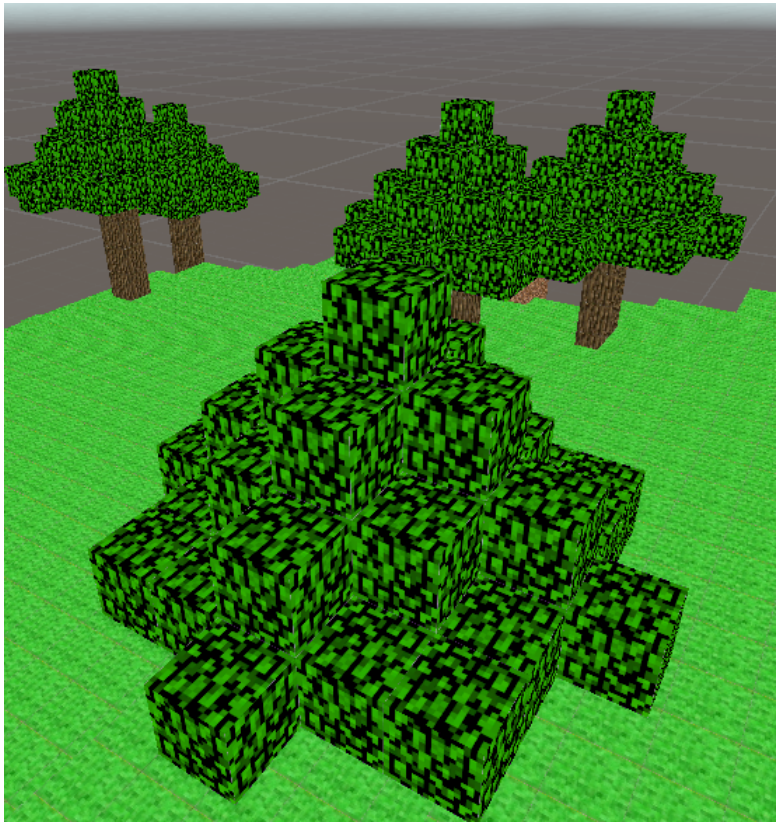
Instituto Superior de Engenharia de Lisboa

Licenciatura em Engenharia Informática e Multimédia

Interacção em Ambientes Virtuais - 2023/2024 SV

Trabalho nº 1 - Mundo Minecraft



Docente Arnaldo Abrantes

Realizado por :
Pedro Silva 48965

19 de setembro de 2024

Conteúdo

Lista de Figuras

1 Introdução

Este primeiro trabalho pretende consolidar e avaliar os conceitos adquiridos ao longo da disciplina Interação em Ambientes Virtuais sendo o objetivo a criação do nosso próprio ambiente Minecraft. Ao longo das aulas foi desenvolvido código para a construção do terreno e interação base do jogador com esse mesmo. Para a construção do terreno foram usadas técnicas procedimentais como o ruído de Perlin e a sobreposição de várias funções para produzir movimento browniano fractal. Eu decidi aprofundar a diversificação do terreno no que toca ao subterrâneo assim como o terreno superior. Assim, foram adicionados ao submundo todos os minerais base do jogo Minecraft (Carvão, Ferro, Ouro e Diamante) e ao mundo superior 2 tipos de árvore diferentes (Carvalho, Bétula). Também foram alterados vários parâmetros da construção do mundo com o intuito de obtermos um melhor desempenho do jogo.

2 Desenvolvimento

2.1 Desempenho do Jogo

A performance do jogo apenas com o código fornecido nas aulas ficou um pouco aquém do desejável por isso decidi tentar melhorá-la. O primeiro teste que fiz foi num primeiro momento diminuir o raio de chunks para 2 em vez de 3 e, vendo que não tinha os melhores resultados, num segundo momento aumentar o tamanho de cada chunk para 24 (normal é 16). Diminuir o raio melhorou o desempenho, como é lógico, mas tornava-se distrativo o jogador conseguir ver o final do mapa pois este tinha que se chegar muito perto para o próximo chunk renderizar. Para tentar combater isto aumentei o tamanho do chunk para 24, só que isto piorou significativamente a performance do jogo. Esta queda deveu-se o facto de como aumentámos o número de blocos por chunk, também aumento a complexidade do mesmo logo, fez com que se demorasse mais a criar cada chunk pois o seu número de triângulos aumentou.

Com isto surgiu a ideia de voltarmos para a configuração inicial mas diminuir a altura a que as grutas começam de 10 blocos abaixo da linha do mar para 20. Esta mudança tem o propósito de diminuir a complexidade do submundo, quando se está na superfície, o que faz com que não exista a necessidade de se tornarem tantos blocos visíveis quando não se está no subterrâneo, ou seja, vai ser preciso cavar mais para se chegar aos minérios. Depois também poderíamos diminuir o raio da criação de chunks pois quando estamos fechados numa gruta não precisamos de saber o que está à volta.

Esta mudança afetou o nosso mundo na medida em que agora são precisos à volta de 7 segundos para o mundo ser criado e se estabilizar enquanto que antes era por volta dos 10. A criação de novos chunks também se tornou mais suave dando resultado a menos ”bugs”. Estas melhorias tornam-se óbvias quando comparamos o número de triângulos entre as duas versões, enquanto que a versão original tinha num momento inicial 83k triângulos a nova versão tem 70k um decréscimo de 13 mil triângulos o que torna lógico esta melhoria no desempenho.

2.2 Diversificação do Interior

Com o código das aulas apesar de já serem criadas grutas estas são apenas feitas de pedra o que as torna desinteressantes de explorar, para combater esse aspeto decidi adicionar novos blocos e minérios ao interior do mundo. Foram adicionados então gravilha, terra, carvão, ferro, ouro e diamante.

Figura 1: Distribuição dos minérios

Nesta imagem conseguimos ver todos os blocos adicionados aos níveis inferiores do mapa e a sua distribuição tal como no jogo original sendo os minérios mais abundantes perto do solo o carvão e ferro e os mais afastados ouro sendo o diamante extremamente raro aparecendo uma vez nesta imagem. Este processo foi possível ao manipularmos o código fornecido, ao longo das aulas, desta forma:

```
if (worldY <= hs)
{
    float under = Utils.fBM3D(worldX, worldY, worldZ, 2, 0.5f);
    if (under < 0.39f && worldY <= hs - 5)
        chunkData[x, y, z] = new Block(Block.BlockType.DIRT, pos, this, material);
    else if (under < 0.40f && worldY <= hs - 2 && worldY > hs - 20)
        chunkData[x, y, z] = new Block(Block.BlockType.COAL, pos, this, material);
    else if (under < 0.405f && worldY <= hs - 4 && worldY > hs - 25)
        chunkData[x, y, z] = new Block(Block.BlockType.IRON, pos, this, material);
    else if (under < 0.408f && under >= 0.405f && worldY <= hs - 35)
        chunkData[x, y, z] = new Block(Block.BlockType.DIAMOND, pos, this, material);
    else if (under < 0.412f && under >= 0.408f && worldY <= hs - 20)
        chunkData[x, y, z] = new Block(Block.BlockType.GOLD, pos, this, material);
    else if (under < 0.42f && worldY <= hs - 2)
        chunkData[x, y, z] = new Block(Block.BlockType.GRAVEL, pos, this, material);
    else if (under < 0.515f)
        chunkData[x, y, z] = new Block(Block.BlockType.STONE, pos, this, material);
    else
        chunkData[x, y, z] = new Block(Block.BlockType.AIR, pos, this, material);
}
```

Figura 2: Código da Distribuição dos minérios

Usando o ruído de Perlin criado anteriormente bastou dar limites de acordo com os valores da função e limites da altura para tornar esta distribuição o mais parecido com a original possível. Os limites de altura abaixo do nível da pedra são então, por ordem de blocos dado anteriormente, $\leq -2, \leq -5, entre -2 e -20, entre -4 e -25, \leq -20, \leq -35$.

2.3 Diversificação do Exterior

Depois de expandirmos o interior está na altura de fazer o mesmo com o exterior, para isso vamos plantar dois tipos de árvores, carvalho e bétula, utilizando mais uma vez o Ruído de Perlin. Foi necessário adicionar ao atlas de texturas fornecido pelo docente duas texturas para atingir o nosso objetivo. Foi então adicionado a textura das folhas assim como a vista superior do tronco da bétula. Estas texturas estão indicadas com setas vermelhas na seguinte imagem:

Figura 3: Atlas de Texturas

Começamos pela distribuição das árvores pelo terreno:

```
else if (worldY == h)
{
    float tree = Utils.fBM(worldX * 0.1f, worldZ * 0.1f, 3, 0.9f);

    if (tree > 0.54 && tree < 0.58 && NoTreeNeighbors(chunkData, x, y, z))
        GenerateTree(chunkData, x, y, z, Block.BlockType.OAK);
    else if (tree > 0.55 && NoTreeNeighbors(chunkData, x, y, z))
        GenerateTree(chunkData, x, y, z, Block.BlockType.BIRCH);
    else
        chunkData[x, y, z] = new Block(Block.BlockType.GRASS, pos, this, material);
}
```

Figura 4: Código da Distribuição das árvores

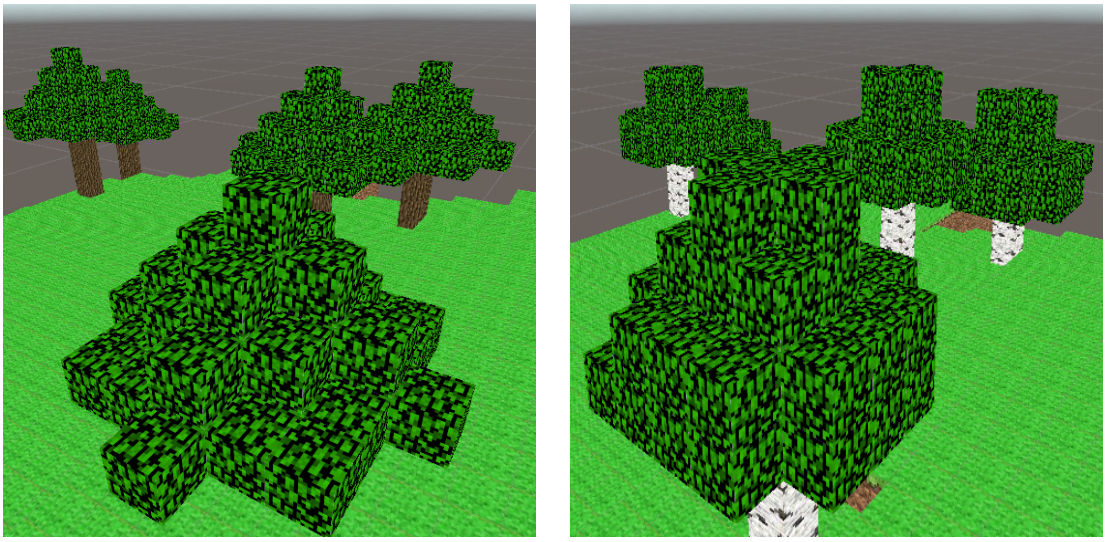
No código acima conseguimos ver o processo de escolha no que toca à colocação das árvores decidindo entre o tipo de árvore ou se mantemos relva. Existe também uma verificação de que a árvore não se encontra muito perto do final do chunk nem tem vizinhos demasiado próximos de modo a não termos árvores deficientes, claro que isto pode sempre acontecer mas estamos a diminuir essa probabilidade significativamente.

Para cada tipo de árvore existe também um algoritmo diferente para a criação do dossel. Os carvalhos vão ter um dossel básico estilo pirâmide enquanto que as bétulas vão ter o design clássico do minecraft composto por 4 filas de folhas:

- A fila superior está um nível acima da altura da árvore e tem sempre exatamente 5 folhas: uma acima do tronco e quatro adjacentes ortogonalmente a ele, formando um formato de +.

- A segunda fila é o topo do tronco e também tem 4 folhas adjacentes a ele. Blocos de folhas diagonais crescem a partir de um mínimo de 1 e um máximo de 3 cantos do tronco.
- A terceira fila tem o tronco no meio, cercado ortogonalmente e diagonalmente por blocos de folhas. Esses 8 blocos de folhas também são cercados ortogonalmente, para um adicional de 12 blocos de folhas. Pode-se pensar nisso como um espaço de 5×5 onde cada bloco é de folhas, exceto os quatro cantos. Esses cantos são preenchidos aleatoriamente com entre 0 a 4 folhas.
- A quarta fila é gerada com as mesmas regras da terceira e, portanto, tem um mínimo de 20 e um máximo de 24 blocos de folhas.

Ficam assim cada uma das árvores:



(a) Carvalho **(b)** Bétula

Figura 5: Os dois tipos de árvore

Devido às funções mencionadas anteriormente conseguimos subjugar cada tipo de árvore a estar nas redondezas da sua própria família criando o efeito de pequenas florestas.

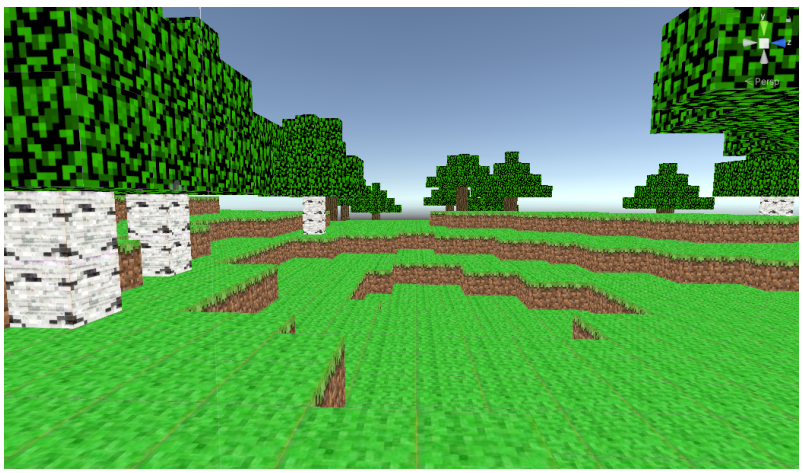


Figura 6: Código da Distribuição das árvores

3 Conclusões

O objetivo deste primeiro trabalho era o de consolidar e avaliar os conceitos adquiridos na disciplina Interação em Ambientes Virtuais usando-os para aprofundar o nosso mundo Minecraft. Conseguimos diversificar ambos os locais principais deste mundo ao utilizarmos as técnicas procedimentais como o ruído de Perlin e a sobreposição de várias funções para produzir movimento browniano fractal. Gostava de ter tido mais tempo para otimizar ainda mais o desempenho do jogo, arranjado uma forma de fazer com que as árvores não tenham uma restrição tão restrita no que toca ao passarem para o próximo chunk e a criação de novos biomas algo que estava nos planos mas devido à interrupção letiva não foi possível. Também gostava de ter adicionado mais interação com o mundo mas esse não pareceu o aspeto mais relevante nesta primeira fase da disciplina. Apesar disto, creio que o projeto desenvolvido está de acordo com o pretendido pelos docente, demonstrando o domínio que tenho sobre toda a matéria lecionada ao longo do semestre.