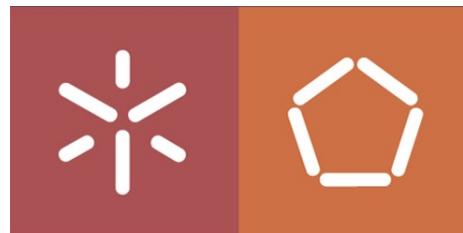


UNIVERSIDADE DO MINHO  
DEPARTAMENTO DE INFORMÁTICA



MEI - MESTRADO EM ENGENHARIA INFORMÁTICA

PERFIL EM COMPUTAÇÃO GRÁFICA

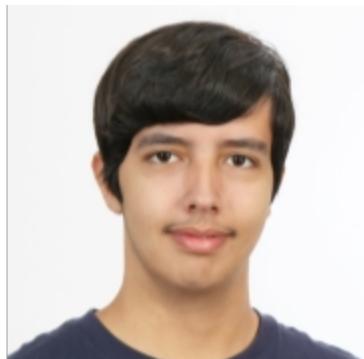
---

*Visualização em Tempo Real*

---

GRUPO 12

TERRAIN GENERATION



Eduardo Pereira PG53797



Filipa Rebelo PG53624



Pedro Fernandes A84313

Junho 2024

# Conteúdo

<b>1</b>	<b>Introdução</b>	<b>1</b>
<b>2</b>	<b>Estrutura do Projeto</b>	<b>1</b>
2.1	Shaders . . . . .	1
2.2	Texturas . . . . .	1
<b>3</b>	<b>Métodos Utilizados</b>	<b>2</b>
3.1	Terreno . . . . .	2
3.2	Noise Functions . . . . .	2
3.2.1	Fractal Brownian Motion . . . . .	2
3.2.2	Ridgenoise . . . . .	2
3.2.3	Noise . . . . .	2
3.3	Água . . . . .	2
3.4	Céu . . . . .	2
3.5	Estados . . . . .	3
3.6	Nevoeiro . . . . .	3
3.7	Iluminação . . . . .	3
3.8	Atributos . . . . .	3
<b>4</b>	<b>Resultados</b>	<b>4</b>
<b>5</b>	<b>Conclusão</b>	<b>4</b>

# 1 Introdução

O presente relatório visa abordar o trabalho desenvolvido no âmbito da Unidade Curricular de Visualização em Tempo Real, focando-se no tema Terrain Generation.

A geração de terreno em computação gráfica é um aspeto crucial que recebeu atenção significativa nos últimos anos. A geração de terreno realista tornou-se um componente desafiador, mas essencial em várias aplicações. Os terrenos desempenham um papel crucial em cenas tridimensionais numa infinidade de aplicações de computação gráfica. A geração de terrenos envolve algoritmos e metodologias sofisticadas para criar paisagens visualmente atraentes e realistas. Técnicas como renderização de terreno baseada em GPU foram adotadas para melhorar a aparência visual de terrenos em simulações como operações de escavadoras. Além disso, o uso de pirâmides Multicare e restrições de curvatura na generalização do terreno tem sido crucial em aplicações relacionadas à escala, visão de computador e generalização em computação gráfica.

A importância da geração de terreno estende-se a aplicações que vão além dos gráficos de computadores, como, por exemplo, na iluminação de terrenos para estudos ecológicos e geográficos. Modelos geradores de céu foram desenvolvidos para renderizar terrenos sob condições de mudança dinâmica do céu, incorporando inter-reflexões entre elementos de terreno. Além disso, a modelagem de paisagens com montanhas e rios é uma importante área de foco em computação gráfica, enfatizando a criação de redes de drenagem realistas em modelos de terreno. Além disso, a utilização de métodos de renderização de terrenos baseados em 'quadtree' têm sido instrumentais na aceleração de processos de visualização do terreno usando GPUs. A navegação assistida por terreno de precisão em tempo real para veículos aéreos não tripulados (UAVs) também se beneficiou de técnicas de computação paralela aplicadas a uma abundância de dados de terreno de alta resolução.

Em conclusão, a geração de terreno em computação gráfica é uma área multifacetada e essencial que impacta uma ampla gama de aplicações. O desenvolvimento de algoritmos e metodologias avançados para modelagem e renderização de terrenos desempenha um papel fundamental na criação de ambientes virtuais imersivos e realistas em vários domínios.

## 2 Estrutura do Projeto

O nosso projeto encontra-se dividido em 2 pastas, sendo elas:

### 2.1 Shaders

Os shaders são essenciais para renderizar gráficos em tempo real. Eles definem como cada píxel e vértice dos objetos na cena são processados visualmente.

- Vertex Shader: responsável por processar cada vértice dos modelos 3D, aplicando transformações como rotação, escala e translação.
- Geometry Shader: atua entre o vertex shader e o fragment shader. Ele permite a manipulação geométrica adicional, como a geração de novos vértices, tessellation e processamento de primitivas.
- Fragment Shader: responsável por calcular a cor final de cada píxel renderizado no ecrã. Ele aplica texturas, sombras, iluminação e efeitos especiais, utilizando informações como normais, coordenadas de textura e dados de luz calculados pelos shaders anteriores.

Foram aplicados shaders para as cenas do céu, terreno e água. No qual, para o terreno foram criados os shaders vertex, geometry e fragment, enquanto para o céu e água foram apenas precisos os vertex e fragment shaders para cada cena.

### 2.2 Texturas

As texturas são imagens 2D aplicadas aos modelos 3D e ao terreno para adicionar detalhes visuais e realismo.

As texturas que utilizamos foram de terra, pedra, areia e neve para simular materiais e texturas naturais para níveis de altura diferentes. O método de criação de ruído (noise) foi utilizado para a criação de um terreno.

## 3 Métodos Utilizados

### 3.1 Terreno

Existem inúmeras estratégias para a criação de terreno, e desejamos implementar uma abordagem que gerasse terreno aleatoriamente, em vez de predeterminados por uma imagem. Foram a seguir examinados os métodos de gerar um mapa de altitude (heightmap) para gerar o terreno. No entanto, foi optada a estratégia mais frequentemente empregada, que envolve o uso de métodos de ruído. Os valores variáveis gerados no ruído representam os níveis de y.

De modo a alcançar uma diferenciação mais realista entre os diferentes níveis de altura, é essencial fornecer camadas para diferenciá-los. Desta forma, quatro níveis foram estabelecidos para representar a areia, a qual é principalmente coberta por corpos de água, a relva/terra, a pedra, que representa uma parte significativa do corpo da montanha, e o nível de neve, o que é o mais alto.

### 3.2 Noise Functions

Na geração de terreno, as funções de ruído são amplamente utilizadas para criar superfícies naturais e complexas processualmente. Estas funções são comumente usadas em métodos de procedimentos para geração de terreno instantânea, fornecendo uma variação aleatória nos valores de altura do terreno.

#### 3.2.1 Fractal Brownian Motion

Técnicas como o *Fractal Brownian Motion*, criado por somar funções de ruído de frequências e amplitudes variáveis, podem ser estendidas para gerar modelos artificiais de terreno rochoso em múltiplas dimensões. Que foi uma das razões pela escolha da técnica para o trabalho.

#### 3.2.2 Ridgenoise

Este método implementa o conceito de Ridge Noise, introduzindo picos e vales distintos no ruído gerado. Modificando o ruído padrão para amplificar ou atenuar extremos, ela é especialmente útil para simular características geológicas dramáticas como montanhas íngremes e vales profundos.

#### 3.2.3 Noise

Este método utiliza um método de interpolação entre pontos de grade para gerar ruído suave. Ela divide o espaço numa grade de células e calcula valores de ruído em cada ponto da grade, suavizando as transições entre valores adjacentes. Essa técnica é fundamental para criar texturas orgânicas e superfícies naturais em terrenos virtuais.

### 3.3 Água

Para a simulação da água, optamos por utilizar um modelo simples. A superfície da água é uma grade, em que apenas a colorimos com um tom de azul. Para representar a transparência da água, é utilizado um estado no material que permite realizar o teste do canal *alpha*. Assim, podemos passar como quarto componente da cor da água o valor desse canal, e o motor gráfico irá tratar da grade como um objeto transparente. Para que isto seja possível ser aplicado, o material é passado como um *injection map* em vez de um *material map*.

### 3.4 Céu

Para o céu, a nossa ideia inicial foi utilizar uma *Sky box*, por via de um *injection map*, tal como na água, para que a cor que havíamos utilizado para o céu fosse vista por dentro da *Sky box*, o que levou a criar um estado para trocar o tipo de *culling* do *default* para o *front*.

No entanto, nas etapas finais, decidimos criar uma opção de personalização do céu, que consiste em escolher se é utilizada a mesma cor referida anteriormente, ou uma textura 2D aplicada a todas as faces do cubo.

### 3.5 Estados

Os estados num ficheiro de configuração XML para renderização gráfica são essenciais para definir comportamentos específicos que afetam como os materiais ou objetos são processados durante o pipeline de renderização.

No nosso projeto foi definido um estado transparente para lidar com materiais que possuem características de transparência. Estas configurações são cruciais para garantir que objetos transparentes sejam renderizados corretamente, preservando a integridade visual da cena. Elas permitem que a transparência seja aplicada de forma eficaz, evitando artefactos visuais indesejados e garantindo uma representação precisa dos materiais na renderização gráfica.

### 3.6 Nevoeiro

Para aumentar o realismo e o apelo visual do nosso terreno gerado, implementamos um sistema de névoa atmosférica. Este recurso não só adiciona profundidade e atmosfera à cena, mas também fornece um mecanismo para controlar a visibilidade de características de terreno distantes. A nossa implementação de névoa usa um modelo de nevoeiro linear, que calcula a intensidade da névoa com base na distância da câmara. A densidade da névoa aumenta linearmente com a distância, criando uma transição suave de visibilidade clara no primeiro plano para características obscurecidas no fundo. A cor da névoa é definida por omissão como cinza suave (RGB: 128, 153, 179), que combina bem com a maioria dos tipos de terreno e condições de iluminação. O efeito de névoa é aplicado no fragment shader, onde interpolamos entre a cor do terreno e a cor da névoa com base no fator de intensidade do nevoeiro calculado. No entanto, existe um limite de intensidade do nevoeiro aplicado ao céu visto que se o mesmo estiver a utilizar textura queremos que seja visível essa mesma textura.

### 3.7 Iluminação

Para o modelo de iluminação, optamos por implementar um simples modelo de reflexão, de *Lambert*, que utiliza apenas a direção da luz e a normal num vértice, o que resulta num cosseno do ângulo entre os dois vetores, que é proporcional à intensidade da luz que atinge a superfície. Esse valor é multiplicado à cor escolhida, criando zonas mais claras onde a superfície é diretamente atingida pela luz, e zonas escuras onde a luz não atinge.

### 3.8 Atributos

Existem atributos relativos ao terreno que podem ser alterados na visualização para ver esses valores a afetar a própria cena do terreno. As variáveis são:

- scale: Controla a escala global do terreno, afetando a amplitude das deformações geradas pelo ruído.
- gain: Define a amplitude relativa entre cada octave consecutivo no ruído. Ajusta a intensidade das transições entre as diferentes escalas de frequência no ruído, afetando diretamente o contraste e a nitidez das características do terreno.
- min.h: Especifica a altura mínima do terreno gerado pelo ruído. Define o nível base do terreno, garantindo que não haja partes subterrâneas ou abaixo do nível desejado.
- noiseSeed: Define a semente inicial para a geração de ruído, afetando a aleatoriedade e a repetibilidade do padrão gerado. Útil para criar variações sem alterar outros parâmetros.
- showWater: Controla a visibilidade da água na cena.
- fogIntensity: Define a densidade da névoa atmosférica na cena. Afeta a visibilidade e a atmosfera geral da cena, criando uma sensação de profundidade e distância.
- skyColorOrTex: Permite flexibilidade na representação do céu, permitindo alternar entre uma textura específica ou uma cor sólida, dependendo do valor atribuído.
- lacunarity: Determina o fator de escala entre os diferentes níveis de frequência dos octaves no ruído. Influência a textura do terreno, controlando como as variações de frequência são distribuídas ao longo das camadas geradas pelo ruído.

- frequency: Especifica a frequência inicial do ruído para gerar o terreno. Controla a taxa de mudança das características do terreno, determinando a distância entre os picos e vales na superfície gerada.
- octaves: Controla o número de camadas utilizadas no algoritmo de ruído para gerar detalhes fractais. Quanto maior o número de octaves, maior a complexidade e o detalhe do terreno, resultando em características mais naturais.

## 4 Resultados

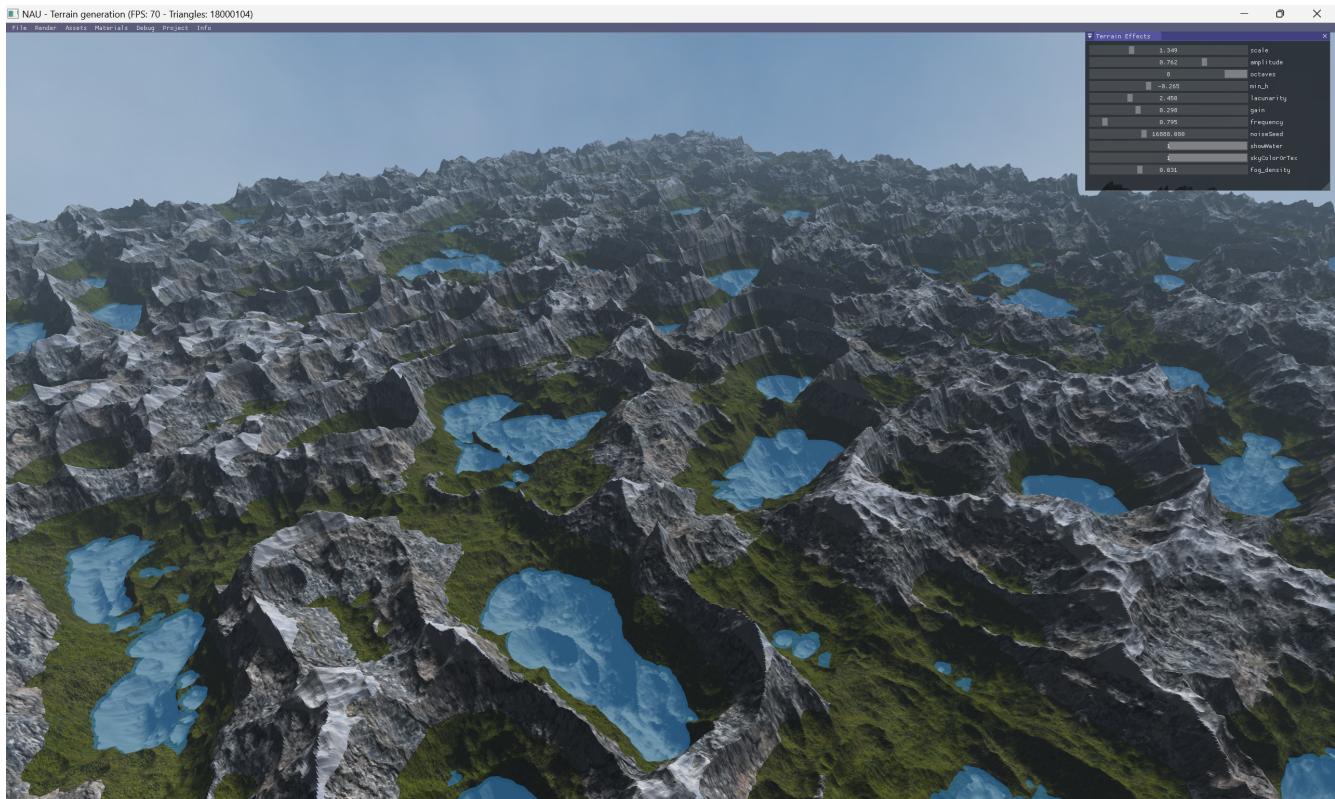


Figura 2: Exemplo de um resultado possível

## 5 Conclusão

A realização deste projeto permitiu consolidar os conceitos abordados nas aulas desta Unidade Curricular.

As principais realizações do nosso esforço incluem:

- A utilização de funções de ruído para implementar um sistema versátil de geração de terreno que permite desenvolver paisagens que sejam realistas e diversificadas.
- A criação de uma abordagem de terreno de várias camadas que distingue os níveis de areia, relva/terra, pedra e neve, de modo a melhorar orealismo.
- A integração de elementos de água e céu, que inclui um sistema de céu personalizável que suporta cores sólidas e texturas.
- A implementação de um sistema de nevoeiro que facilita o controle dinâmico da visibilidade da cena e melhora a percepção de profundidade.
- A renderização eficiente de terrenos complexos e efeitos ambientais é alcançada através do uso de tecnologia dos shaders, incluindo vertex, geométrico e fragment.

- O desenvolvimento de um sistema de personalização que permite o ajuste em tempo real de uma variedade de parâmetros de terreno, como escala, características de ruído e características ambientais.

No entanto, é de realçar que, não obstante, como não estávamos a conseguir aplicar a textura por dentro, seja 2D ou *SamplerCube*, decidimos fazer um *workaround* que consiste em criar 6 sky boxes e colocá-las à volta do terreno, para o efeito resultante ser idêntico à nossa ideia original, seja para cor ou textura. Para o caso de utilizar cor, o resultado é satisfatório, mas para o caso de se optar por utilizar textura, já não o podemos verificar, visto não termos conseguido encontrar uma textura que tivesse uma transição *seamless* entre as diferentes faces, resultando em quebras visíveis nos cantos superiores e inferiores conectados entre cubos.

Era também possível, na nossa opinião, encontrar imagens de textura que trouxessem mais detalhe às diferentes camadas de elevação do terreno, e que o próprio terreno acomodasse o detalhe dessas texturas para o terreno ser mais suave e não muito triangulado.

Em resumo, este projeto atingiu os seus objetivos iniciais de desenvolver um sistema de geração de terreno, mas também ofereceu intuições valiosas sobre as complexidades e o potencial das técnicas de visualização em tempo real. Os conhecimentos e a experiência adquiridos a partir deste projeto irão, sem dúvida, ser benéficos nos futuros esforços no campo da computação gráfica e além.

## Referências

- [1] Geração de terrenos com a utilização de *ridge noise*: <https://www.redblobgames.com/maps/terrain-from-noise/>
- [2] *Lambertian Reflectance*: [https://en.wikipedia.org/wiki/Lambertian\\_reflectance](https://en.wikipedia.org/wiki/Lambertian_reflectance)
- [3] <https://github.com/AsuosOnurb/terrain-generation-and-rendering>