

Charack: ferramenta para geração em tempo real de mundos virtuais pseudo-infinitos para jogos 3D

Fernando Bevilacqua
Cesar Tadeu Pozzer
Universidade Federal de Santa Maria

Abstract

Em jogos *massively multiplayer online* (MMO) a existência de um mundo virtual persistente é um tópico importante para manter o jogo atrativo e divertido ao jogador. Quanto maior o mundo a ser explorado, tecnicamente mais tempo o usuário passará jogando para conseguir explorar o maior número possível de lugares. A solução proposta neste trabalho é uma ferramenta capaz de gerar mundos virtuais pseudo-infinitos com diversificação de formas e relevos ao longo de sua extensão. Utilizando uma combinação de algoritmos e métodos de gerenciamento de conteúdo, a ferramenta é capaz de criar ilhas/arquipélagos, baías e costas contendo falésias, praias e rochedos que imitam as paisagens encontradas na natureza. Dentre as inovações apresentadas, estão a criação de um terreno virtual de vastas proporções com enfoque mais detalhado no que diz respeito à geração da costa.

Keywords:: MMO, mundos virtuais, geração de terreno, jogos 3D, ruído, geração procedural, multifractal

Author's Contact:

{fernando,pozzer}@inf.ufsm.br

1 Introduction

2 Introdução

The computer games market has been evolving considerably over the years. Since the first console, the hardware performance has increased and new graphic technologies were developed, resulting in a wide range of themes and game styles. In the multiplayer games, players interact with other human beings and also with NPCs, which are represented by virtual characters. The kind of game is popular and the social interaction between players is a matter of research [Griffiths et al. 2003; Ducheneaut et al. 2006]. In the category of multiplayer games there are the massively multiplayer online (MMO) ones, which are online games featuring large number of players interacting with each other in a persistent virtual world.

An MMO can feature millions of players, such as EverQuest [Sony Entertainment 2007] and World of Warcraft [Blizzard Entertainment 2007], the latter with more than 6 million subscribers [Clark 2006]. A persistent virtual world is an important topic to keep the game fun and attractive to the player. The bigger is the world to be explored, technically the bigger is the time the player has to spend in order to explore all the places. As a result of such huge virtual worlds, the creation (and subsequent upgrade) of those worlds is a complex task. EverQuest and World of Warcraft present a virtual world with a wide diversity of geographical features such as mountains, valleys, forests, fields, caves, etc., and the vast majority of them nominated and related the history of the game. The manual creation of those virtual worlds requires a team able to create heightmaps, adorn landscapes, ensure usability of the map (avoid unreachable places, for example), create interesting place for players, etc. To help on that task, the development of a tool able to generate complex virtual worlds is useful to speed up the development of 3D games such as MMOs.

The solution proposed in this work is a tool able to generate complex virtual worlds in real-time using noise-based techniques of terrain generation. The proposed tool is able to generate complete virtual world featuring continents, oceans, flat areas, mountains, etc,

with all those elements generated in real time, on demand as the player moves along the world.

3 Overview

The tool was designed to allow developers to use its features in order to generate 3D terrains for games, particularly MMOs, with minimal human intervention in the generation process. The size of the resulting world is customizable, but it can reach billions of pixels. The use of the term "pseudo" is necessary due physical limitations in computers hardware: an unsigned integer, for instance, can store a certain amount of data; if there were no physical limitations, the tool would be able to generate, in fact, an infinite world.

The content generation is made on demand. As the user moves along the world, the elements inside the user's view are processed and stored into the memory and the ones away from the user's view are removed. Even though the generation of all elements is based on random calculations, if the player visits a specific point A, then walks for miles generating a completely different set of landscapes, and returns to point A, the same previously seen landscape will be shown again.

Besides the heightmap generation, the tool is also able to create oceans and continents, all of them customizable. Unlike other related works which deal with the generation of continents as a consequence of the heightmap generation, this paper presents as its main contribution a new approach for content generation in virtual worlds. In this approach, the generation of continents, topography and coastlines are handled separately.

The shape of continents is defined on a macro level as a result of an algorithm for maps generation that work projecting pixels onto a sphere. The coast line generation is performed on a micro level. As a result, the tool is able to produce cliffs, beaches, bays and islands.

4 Related works

4.1 Virtual city

The procedural content generation is an old topic in the field of computer graphics. The application of such technique to generate a complete virtual world was used by [Greuter et al. 2005], whose goal was to create a virtual city that was visually interesting and complex.

In the approach the world were divided into a grid consisting of several squares, called cells. The location of each cell is used with a global seed as an input for a hash function [Wang 2000]. The result of this function is used as a seed for a pseudo random number generator and it will define all the characteristics of the buildings within a cell. As a consequence of that approach the contents of a cell is always the same, no matter how far the user moves or if the cell is removed from memory.

To ensure acceptable use of memory and CPU, only the elements inside the user's view are generated. As the user moves through the city, new cells are added to the user's view and the new content is generated. Whe the cell leaves the user's view, it is removed from memory and its resources are released. The cells are placed in square loops around the user and are considered inside the user's view if it is close enough and if it is located within a 120° view range.

4.2 Layered generation and texturization

Another related work was a tool used to build the SkyCastle multiplayer game engine [Häggström 2006; Häggström 2009]. For the heightmap generation, parameterized procedures and fractal based systems are combined in layered approach: starting with a base map, the application merges a new map with to the base in each iteration. The new maps are pre-calculated and generated using Perlin noise.

One of the texturization methods presented are the use of a large image able to cover the whole world. Although this approach is useful for small scenes, it is not suitable for large scenes nor virtual worlds, since the image size could reach prohibitive proportions. To avoid this problem, an reticulate approach is suggested [Cohen et al. 2003]; using that approach a texture cell is created so that the a combination of them produces a smooth and continuous and textured plane. The results are acceptable, but it is not visually attractive to the user, because the landscape features an unusual pattern that will never exist in the real world. In order to achieve a better visual result, an approach proposed by [Lefebvre and Neyret 2003] is suggested. In that approach, a set of pre-defined border textures is combined with reticulate textures. The idea is to apply the border textures on an already textured plane, but using transparency in the border textures. As a consequence, border textures will overlap the some regions of the plane and produce a less homogenous landscape, which creates a better visual result.

In order to decorate the virtual world, three procedural plant generation methods are used: L-System [Przemyslaw and Lindenmayer 1990], component-based generation [Lintermann and Deussen 1998] and parameterized trees [JasonWeber and Penn 1995].

4.3 Procedural planets

The use of fractals and their combinations with other methods are also used to procedurally generate planets [Linda 2007]. In this approach, the generated world is not infinite, but it is spherical and simulates the view of the planet Earth. Starting with a recursive subdivision of an octahedron, a spherical world is created and it is used as a first step to seed algorithms of heightmap generation. The heightmap is obtained as a result of several techniques, each one featuring its peculiarities and results. Among the techniques used are: generation using random failures, midpoint displacement (including a multifractal variation) and Perlin noise (and its many variations).

4.4 Fractais afetados por erosão

Outro trabalho analisado foi a geração de terrenos procedimentais em tempo real com a utilização de fractais afetados por erosão [Olsen 2004]. Conforme ressaltado, o aumento do poder de processamento de computadores domésticos possibilitou que jogos de computador façam uso de simulações de erosão em tempo quase real. Agregou-se à geração de relevo por fractais novas características decorrentes da aplicação de erosão, o que teve como resultado mapas mais reais e, ao mesmo tempo, customizáveis. Primeiramente, um mapa de altura foi criado a partir de um diagrama de Voronoi, sendo cada um dos vértices do diagrama chamados de *pontos de funcionalidade*; o valor de cada célula no mapa de altura é obtido através de uma combinação linear das distâncias entre os pontos de funcionalidade mais próximos. Depois que o mapa de altura é gerado, ele é combinado com um algoritmo de geração de ruído para produzir montanhas menos pontiagudas e, para remover as linhas bem definidas criadas pelo diagrama de Voronoi, um filtro de perturbação ainda é aplicado. Para a simulação de erosão no mapa de altura gerado, utiliza-se um método híbrido de duas técnicas de erosão: a termal [Musgrave et al. 1989] e a hidráulica.

4.5 Divisões estocásticas de uma quadtree

Outro trabalho analisado foi a geração de mundos virtuais proceduralmente através de técnicas de subdivisão estocásticas [Dollins 2002]. O objetivo foi criar um mundo virtual de grandes

proporções, porém gerando o seu conteúdo sob-demanda de forma procedural e com multi-resolução. Para a geração do relevo do terreno, utilizou-se sub-divisões de uma *quadtree* em um processo recursivo. A forma do terreno é definida pelo ponto médio de cada uma das células, que na figura são representados pelos círculos pequenos no centro de cada quadrado. Para cada nível de subdivisão, os novos pontos médios das células filho criadas são definidos em função das nove células pai que estão ao redor da célula sendo dividida. A altura dos demais pontos da malha, como os vértices que estão nos cantos das células, é definida pela interpolação dos pontos médios mais próximos. A cada camada de detalhamento, mais células são subdivididas, e a célula que sofreu a subdivisão é substituída pelas novas células filho geradas.

4.6 Gerador de mapas para planetas

Outro trabalho analisado foi um gerador de mapas para planetas, que consiste na geração de mundos redondos através da subdivisão recursiva de um tetraedro [Mogensen 2009]. Embora o trabalho não seja apresentado como uma ferramenta gráfica na qual o usuário pode explorar um ambiente 3D, todas as informações geradas são parte de um mundo virtual completo, com continentes e oceanos altamente customizáveis. A ideia apresentada é criação de um mapa como resultado da projeção de pixels numa esfera, numa abordagem que se assemelha ao *ray tracing*. Assim como é feito nesse último, no qual os raios que partem da câmera e colidem com algo geram um pixel visível, cada pixel do mapa a ser gerado é projetado para uma esfera. O algoritmo que faz a projeção atribui uma altura ao pixel encontrado, o que gera um mapa com relevo.

Para renderizar um mapa, inicialmente encontram-se os pontos dele que são visíveis na superfície da esfera. Em seguida, para calcular a altura de cada um desses pontos, aplica-se o seguinte algoritmo:

- Inserir a esfera dentro do tetraedro.
- Cortar o tetraedro em dois tetraedros menores.
- Escolher em qual dos dois tetraedros está localizado o ponto sendo analisado.
- Repetir os passos anteriores até que o tetraedro seja pequeno o suficiente
- Usar a altura média dos vértices do tetraedro para calcular a altura do ponto sendo procurado.

Inicialmente a esfera é colocada dentro de um tetraedro irregular e informações sobre altura e uma semente para cálculos aleatórios é colocada em cada um dos vértices dele. Em seguida, ele é dividido em dois pelo plano formado entre o ponto médio da aresta mais longa e os dois pontos finais da aresta oposta ao ponto médio. Em suma, cada um dos dois tetraedros resultantes terão três vértices a partir do tetraedro original mais um novo vértice posicionado no ponto médio da maior aresta. A altura e a semente do novo vértice são calculados a partir da altitude e semente dos pontos finais da linha que é dividida (maior aresta). O processo é repetido recursivamente para cada um dos tetraedros até que ele fique pequeno o suficiente para conter apenas o ponto desejado.

5 Organização da ferramenta

As seções seguintes explicam como a ferramenta, doravante denominada *Charack*, foi estruturada para permitir que os objetivos citados fossem atingidos. O texto está organizado de forma a apresentar uma abordagem *top-down*: parte-se da explicação sobre a geração de continentes (visão mais ampla) e termina-se com a explicação sobre a geração de relevo para cada um dos pixels a serem desenhados na tela (visão mais específica e pontual).

5.1 Análise inicial

Existem diversos trabalhos na área de geração de mundos virtuais, sejam eles finitos ou infinitos. O trabalho desenvolvido por [Greuter et al. 2005] cria uma cidade infinita que é apresentada ao usuário sob demanda à medida que esse passeia sobre o terreno. Esse foi o trabalho semente de onde nasceu Charack, porém a idéia

original foi alterada a fim de que mais conteúdos fossem criados (montanhas, planícies, continentes, etc.), não só ruas e prédios. A criação de conteúdo sob demanda, na qual o usuário só enxerga aquilo que está dentro do seu campo de visão, foi mantida da abordagem original.

O mundo procedural gerado por [Linda 2007] aproxima-se muito do conceito buscado para a criação de conteúdo de Charack. No referido trabalho, um planeta esférico é criado a partir da divisão recursiva de uma forma geométrica e, sem seguida, funções de ruído são aplicadas à malha para a geração de conteúdo. Não existem abordagens separadas para a criação dos continentes e do relevo dentro deles, ou seja, os continentes são resultado da inundação do relevo criado pelas funções de ruído; como consequência, a costa dos continentes também não possui um tratamento diferenciado, ela surgirá baseada na altura do nível do mar e da quantidade de ondulações presentes no relevo. O mundo criado é esférico e o jogador pode aproximar ou distanciar a câmera do terreno, porém o conteúdo não é gerado sob demanda. Charack foi criado a partir de uma evolução dessa idéia, porém aplicando-se algumas limitações. O mundo criado por Charack trata de forma completamente diferente a geração de continentes, do relevo dentro deles e da costa. Cada um desses elementos possui sua forma de atuar, que permite ao programador customizar detalhes pontuais sem interferir nas demais áreas do mundo. Esse é o caso da criação de praias mais longas na costa sem modificação da estrutura básica do relevo ou do continente na que essa praia está inserida. Além disso, Charack gera o conteúdo sob demanda num mundo com proporções muito maiores, entretanto sem o uso da abordagem esférica; o mundo criado está contido dentro de um plano.

Em outro trabalho analisado, o terreno criado sofre ação de erosão [Olsen 2004]. Charack não utiliza esse conceito, porém faz uso de uma idéia semelhante para fazer o casamento dos conteúdos gerados nos diferentes módulos da aplicação. Um exemplo simples é o encontro do relevo do continente com o oceano, no qual as alturas de cada pixel são atenuadas para que um relevo de praia seja criado.

Por fim o trabalho criado por [Dollins 2002] aproxima-se muito do objetivo proposto. Nesse trabalho, um mundo virtual de proporções gigantescas é criado e o seu conteúdo é gerado sob demanda à medida que o usuário se move. O relevo é criado de forma parametrizada e com multi-resolução, ou seja, quanto mais próximo o usuário estiver de um determinado local no mundo virtual, maior será a quantidade de detalhes nesse local. Também não são apresentadas abordagens separadas para a criação de continentes/costa/relevo. O funcionamento da geração de relevo de Charack baseia-se nas idéias desse trabalho (geração sob demanda e parametrizável), porém a criação de continentes e costa é completamente diferente.

Analizando-se cada um dos trabalhos citados, é possível constatar que mundos virtuais de diferentes proporções são gerados, porém em nenhum deles há uma preocupação com a separação dos métodos de geração dos conteúdos (continentes, relevo, costa). Embora existam variações na forma como o relevo é criado, a geração de continentes fica a cargo da ação de um plano de água interceptando o plano de terra. Essa abordagem permite que esforços sejam concentrados na geração de conteúdo do miolo do mundo, que são as faixas de terra, porém ela tem um aspecto simples em relação aos demais elementos. No que se refere à heterogeneidade de conteúdo, como diferentes configurações de continentes e costa, o resultado final é pobre. A principal idéia e contribuição de Charack é a geração dos diferentes tipos de conteúdo de forma separada, com abordagens agressivas e pontuais em cada um dos tópicos, entretanto criando um resultado único que contempla todos os tópicos. Como a grande maioria dos trabalhos analisados foca na criação de relevo, entende-se que a contribuição de Charack será maior se os esforços forem focados no que diz respeito à geração de continentes e costa.

5.2 Estrutura básica

Para a criação do mundo virtual nos moldes pretendidos, adotou-se uma estratégia hierárquica para a geração de conteúdos. O fluxo de dados de Charack tem início na visão mais ampla do mundo, que

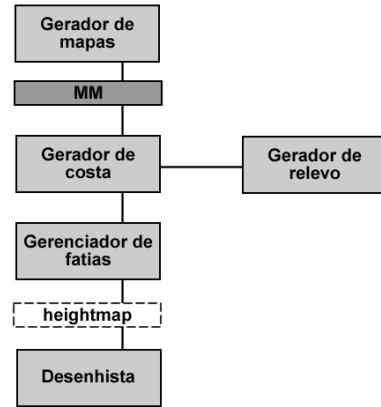


Figure 1: Estrutura básica para geração de conteúdo

são os continentes, evoluindo ao longo do processo até terminar na visão mais específica e pontual, que é a geração de conteúdo para cada um dos pixels que serão desenhados na tela. A figura 1 ilustra a estrutura básica.

5.2.1 Gerador de mapas

No topo da cadeia encontra-se o gerador de mapas, que é responsável por criar os continentes que existirão ao longo do mundo virtual. Esse módulo é um encapsulamento da solução criada por [Mogensen 2009], que foi descrita na seção 4.6. Quando a ferramenta é iniciada, ela utiliza uma semente definida pelo usuário para gerar todos os continentes que existirão no mundo virtual. Depois que os continentes são gerados, as informações sobre os tipos de terreno (terra, água e costa) são armazenadas em uma matriz, chamada **macro-matriz** (MM), que é utilizada para consulta pelos demais algorítimos da ferramenta.

5.2.2 Gerenciador de fatias

Abaixo da MM e do gerador de mapas existe o gerenciador de fatias (GF). Ele é responsável por recortar uma porção do mundo virtual (campo de visão do usuário) e, em seguida, fornecer essa informação ao desenhista através de um *heightmap*. Para obter as informações necessárias para a criação do *heightmap*, o gerenciador de fatias faz uso do gerador de costa (GC), que por sua vez faz uso da MM e do gerador de relevos (GR).

No contexto do GF, não existe informação sobre terra, água ou afins, ele apenas tem conhecimento de um conjunto de pixels do mundo virtual e qual a altura de cada um deles. Utilizando a posição do usuário como guia, o GF realiza o corte do mundo virtual e, para cada um dos pixels coletados, ele faz consultas ao GC para descobrir qual a altura desse pixel.

5.2.3 Gerador de relevo

O gerador de relevo é responsável pela definição da altura que cada pixel do mundo virtual possui. Para garantir que o programador possa criar um relevo customizável baseado em suas necessidades, o gerenciador de relevo permite que novas funções de geração de conteúdo sejam acopladas de forma simples. Com essa abordagem, diferentes tipos de relevo podem ser adicionados à aplicação de forma fácil, basta que as funções adicionadas sigam o protótipo esperado pelo módulo.

5.2.4 Gerador de costa

O gerenciador de costa (GC) irá mapear cada pixel do gerenciador de fatias para a MM e descobrir qual é o tipo de terreno associado. Se o pixel em questão é mapeado para um local na MM que é descrito como água, então o GC atribuirá uma altura igual ao nível do mar para o pixel e, em seguida, irá retorná-lo para o GF. Se o pixel em questão é mapeado para um local descrito como terra (simples), então o GC utilizará as informações do GR para descobrir qual a

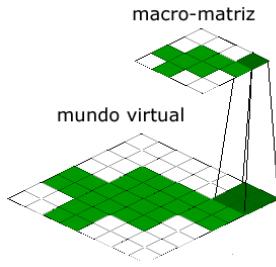


Figure 2: Mapeamento da MM para o mundo virtual: cada pixel da MM é mapeado para diversos pixels no mundo virtual.

altura desse pixel, o que definirá qual relevo ele representa. Por fim, se o mapeamento terminar em um terreno descrito como costa, então o GC utilizará sua própria estrutura (em conjunto com a MM) para definir o relevo do pixel.

O mundo virtual gerado possui, tecnicamente, altura e largura definidas pelo tamanho máximo que um inteiro pode suportar, o que torna fisicamente inviável a geração de uma MM com tais proporções. Uma vez que a MM é menor que o mundo virtual, uma entrada (i, j) dela corresponde a diversos pixels dentro do mundo virtual. A figura 2 ilustra o mapeamento da MM para o mundo virtual.

Observando-se a figura é possível constatar que quanto menor for a MM, mais pixels do mundo virtual serão mapeados para uma mesma entrada. Se o mundo virtual tiver dimensões 1000×1000 e a MM 10×10 , por exemplo, isso quer dizer que para cada pixel da MM existem 100 pixels no mundo virtual; se um desses pixels da MM for do tipo costa, então existirá uma área de 100×100 pixels no mundo virtual que precisa ser uma costa. É exatamente nesse ponto que o GC atua no que diz respeito à geração de conteúdo para a costa; quando qualquer um dos pixels dessa área em especial chegar aos cuidados do GC, ele irá trabalhá-los e retornará ao desenhista informações diferentes das originais, o que permitirá que uma costa seja desenhada na tela, não uma área inteiramente preenchida por terra ou por água.

5.2.5 Desenhista

O desenhista é responsável por renderizar na tela o *heightmap* criado nas demais fases do processo. O resultado final é desenhada como uma malha de triângulos que é texturizada conforme a altura de cada vértice.

6 Implementação

O principal problema encontrado durante a implementação de Charack foi a geração de conteúdo sob demanda. Partindo do fato que o usuário só consegue enxergar aquilo que está dentro do seu campo de visão, todos os algoritmos de geração de conteúdo, seja para relevo, caminhos ou cidades, precisam levar em consideração única e exclusivamente as informações que estão disponíveis dentro desse campo. Essa abordagem é eficiente para a utilização racional de recursos (processar somente o que o usuário está vendo), porém ela aumenta a complexidade dos algoritmos envolvidos.

Para o algoritmo de geração de cadeias montanhosas, por exemplo, não é possível determinar onde a cadeia termina, visto que o mundo fora do campo de visão tecnicamente não existe ainda, ele será gerado conforme o usuário avança pelo terreno. Uma abordagem seria utilizar uma função matemática que descrevesse a cadeia montanhosa, porém essa função não deveria depender de um ponto de início e fim, porque eles poderiam inexistir em um determinado momento. Se a função de geração de cadeias montanhosas não dependesse de um ponto de início e fim, ela precisaria, ao menos, depender da posição do usuário no mundo virtual para que o conteúdo correto fosse gerado. Depender de uma localização implicaria que a cadeia montanhosa gerada pela função fosse pré-posicionada no mundo virtual, o que iria contra o conceito de geração de conteúdo sob demanda.



Figure 3: Conjunto de imagens utilizadas para texturização do terreno

Além disso, os algoritmos são sensivelmente afetados pelo fato de que as informações que eles recebem em um determinado instante podem desaparecer por completo na próxima iteração, visto que o usuário pode se mover e mudar o conteúdo do campo de visão. Utilizando o exemplo da geração de cadeias montanhosas, uma montanha poderia sofrer uma alteração em sua composição de forma abrupta, apenas porque os pontos que estavam sendo utilizados para a geração do relevo mudaram.

Para contornar esses problemas e focar os esforços de desenvolvimento em soluções pontuais, a geração do mundo virtual foi dividida em três grandes etapas: terreno infinito, continentes e relevo. A geração de conteúdo sob demanda afeta de forma diferenciada cada uma dessas etapas e a descrição da implementação de cada uma delas, junto com os problemas associados, é descrito nas seções seguintes.

6.1 Terreno infinito

A base para a geração do mundo virtual proposto é a possibilidade do usuário poder andar, de forma infinita, sobre a superfície do mundo e, conforme anda, visualizar novos conteúdos. A medida que o usuário anda, a ferramenta precisa ser capaz de identificar em qual local do mundo o observador se encontra para então gerar os conteúdos à sua volta. Para solucionar esse problema, utilizou-se uma variação da técnica descrita por [Greuter et al. 2005].

A solução encontrada foi fazer com que o jogador veja na tela em uma determinado momento um pedaço do mundo virtual existente, porém sem a utilização de células ou quadrantes. Diferentemente do que foi feito em [Greuter et al. 2005], no qual o campo de visão é um cone, o campo de visão de Charack é um quadrado centrado no usuário. A partir da posição (x, y, z) do usuário, calculase qual é o conteúdo visualizável ao redor desse ponto e, então, "recorta-se" essa fatia do mundo e a desenha-se ela na tela. Ao chegar na borda limite do mundo, que pode ser a distância máxima de um eixo, por exemplo, o usuário é impedido de avançar e nenhum conteúdo é mostrado além da borda limite.

Depois que a fatia visível é recortada, ela é desenhada na tela. Para texturar o conteúdo renderizado, utiliza-se um conjunto de imagens gerenciadas através de GLSL [OpenGL 2007]. Baseado na altura do pixel sendo desenhado, calcula-se o peso que cada uma das texturas possui naquele local e, em seguida, cria-se uma interpolação delas. Cada uma das texturas do conjunto possuem um intervalo de altura na qual podem atuar: areia para alturas baixas, pedra para alturas médias, cascalho para alturas intermediárias e grama para as alturas maiores que as já estipuladas. Um exemplo de funcionamento desse método é a texturização da areia da praia; nesse caso, todos os pixels da praia possuem uma altura que faz com que a textura de areia tenha o peso máximo, ao passo que as demais alturas tenham um peso zero. A figura 3 demonstra o conjunto de texturas existentes.

6.2 Relevo

Considerando que a ferramenta é capaz de criar um mundo pseudo-infinito, a geração de uma malha de relevo tão grande, de uma só vez, é fisicamente inviável; a quantidade de vértices que precisariam ser armazenados tornaria o consumo de armazenamento da aplicação muito grande, o que poderia ser um empecilho para a utilização da ferramenta. Além do problema de armazenamento, outro tópico importante que considerado é a geração de conteúdo contínuo, ou seja, um relevo que não tenha falhas abruptas de continuidade, como uma cadeia montanhosa que termina inesperada-

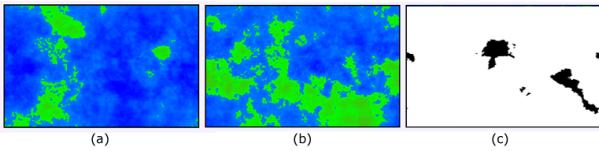


Figure 4: Planetas aleatórios gerados com diferentes sementes de entrada. (a) e (b) mapas com informações de relevo; (c) mapa com informações somente sobre o que é terra/mar.

mente. Os algoritmos de geração de relevo devem trabalhar apenas com as informações existentes no campo de visão, evitando comportamentos estranhos causados pela renovação de pontos dentro do campo de visão à medida que o usuário se move.

Para a ilustração do problema de geração de conteúdo contínuo, supõe-se o seguinte cenário: a geração de relevo utiliza a posição do vértice no mundo virtual como semente para uma função pseudo-aleatória de cálculo de relevo; supõe-se também que a geração de montanhas, por exemplo, é definida por pontos chave que indicam a espinha dorsal da cadeia montanhosa. Se o algoritmo se basear apenas nesses pontos chave para calcular as montanhas, o usuário só irá enxergar a montanha quando um desses pontos chave entrar dentro do campo de visão. Isso irá causar falhas abruptas de continuidade, porque se o usuário costear a espinha dorsal da montanha, sem que os pontos chave entrem no campo de visão, ele não verá a cadeia de montanhas, sendo que ele deveria ver um relevo ascendente até o cume dessas montanhas.

A solução encontrada para a geração de relevo foi utilizar uma função paramétrica que informa a característica de cada um dos vértices do mundo virtual. Nessa abordagem, cada ponto do mundo é utilizado como parâmetro para a função de relevo que, por sua vez, calcula a altura que esse ponto deve ter. Como resultado, tem-se uma função capaz de descrever todo o relevo do mundo virtual, independente do tamanho que ele possua; a quantidade de recursos computacionais que serão utilizados para gerar o conteúdo do campo de visão é diretamente proporcional ao tamanho do campo de visão, não ao tamanho do mundo, visto que a função de relevo utiliza as informações de cada ponto para calcular a sua característica.

Para a criação da função de relevo, utilizou-se o método de geração de ruído de Perlin, que também foi utilizado por [Linda 2007]. Para garantir um controle sobre o nível de perturbação do relevo utilizou-se apenas as propriedades da função de ruído de Perlin; para a obtenção de relevos mais "enrugados", utilizou-se mais oitavas na função de ruído.

6.3 Continentes

A geração de continentes e oceanos foi proposta para quebrar a monotonia de uma paisagem composta apenas por terra no mundo virtual, bem como para aumentar o nível de semelhança com as paisagens encontradas na natureza. A solução encontrada para garantir que os continentes sejam mais customizáveis foi pré-processar as faixas de terra existentes no mundo e, então, armazenar essa informação para os demais cálculos da ferramenta. Com essa abordagem, a geração de conteúdo sob-demanda foi parcialmente quebrada, uma vez que os continentes serão gerados por completo antes de todos os demais conteúdos, porém isso garante um melhor controle sobre o que é oceano e o que é terra.

A geração dos continentes é baseada na abordagem de [Mogensen 2009], descrita em detalhes na seção 4.6. Utilizou-se esse gerador de planetas porque ele possui diversas opções de parametrização, como estipulação da semente que será usada para os cálculos aleatórios, definição da altura/largura do mapa gerado, suavidade das curvas e zoom. A figura 4 (a), (b) e (c), ilustra os resultados obtidos com o gerador de planetas de [Mogensen 2009].

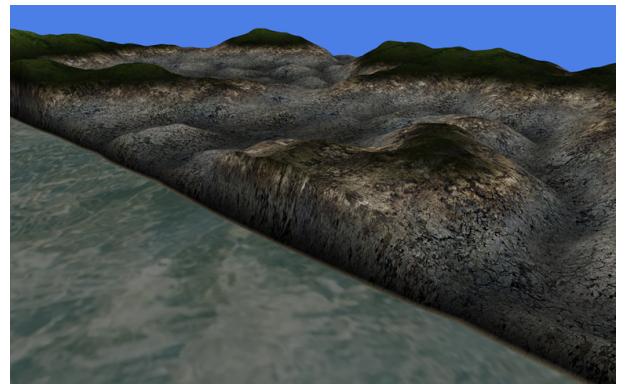


Figure 5: Resultado gráfico obtido quando nenhum algoritmo extra de geração de conteúdo é utilizado para preencher as discrepâncias de mapeamento da MM

6.3.1 Problemas na geração de continentes

Conforme explicado na seção 5.2, cada pixel da MM é mapeado para diversos pixels no mundo virtual. Uma consequência direta desse mapeamento desproporcional é a geração de grandes faixas de terra retilíneas; se fosse possível a geração de uma matriz com o tamanho exato do mundo virtual, ela conteria a resolução necessária para a ferramenta decidir com precisão se um determinado pixel é ou não terra, em uma proporção de 1:1 (um pixel da MM corresponde a um pixel do mundo virtual). Essa abordagem, porém, não é viável, visto que uma matriz com tais proporções consumiria muitos recursos para ser construída e processada. Embora a ferramenta permita que o tamanho da MM seja ajustado para qualquer valor arbitrário, testes realizados mostraram que um tamanho de 800x800 é o bastante para que informações suficientes sejam processadas pelos demais algoritmos da ferramenta.

A figura 5 ilustra o resultado gráfico obtido pela ferramenta quando nenhum algoritmo extra de geração de conteúdo é utilizado para preencher os espaços retilíneos entre um mapeamento e outro.

Nessa ilustração está sendo mostrado um local no mundo virtual que representa a transição entre dois pontos diferentes da MM (um ponto de terra e outro ponto de água). Para explicar o que acontece, são necessárias duas suposições: 1) a ferramenta está desenhando o mundo na posição (x, y, z) , que é o mapeamento do ponto (i, j) na MM, e 2) esse mapeamento corresponde a uma faixa de água. À medida que a ferramenta incrementa a coordenada do mundo para poder desenhá-lo, supondo $(x + 1, y, z)$, esse ponto também é mapeado para a MM. Se o resultado do mapeamento da coordenada $(x + 1, y, z)$ ainda for o ponto (i, j) na MM, então a ferramenta irá novamente desenhar um pixel de água na tela. Supondo que somente no ponto $(x + 10, y, z)$ o mapeamento mude na MM para $(i + 1, j)$ (e que esse ponto na MM seja terra), então todos os pontos anteriores a esse serão água e todos os pontos seguintes serão terra (até que o mapeamento na MM mude novamente).

A figura mostra claramente o momento em que o mapeamento na MM muda, que é quando a ferramenta substitui a renderização da água pela renderização da terra. Como não há algoritmos de geração de conteúdo atuando nessa transição, o usuário pode andar por essa linha reta da costa e não verá qualquer alteração em sua forma ou direção, a menos que outra quebra de mapeamento seja encontrada. Nesse caso, o usuário poderá ver outra costa retilínea perpendicular à aquela que ele está seguindo ou outra costa no mesmo sentido (que é a continuação da costa atual).

6.3.2 Quebra de linearidade da costa

O mapeamento dos pixels do mundo virtual com as informações da MM em baixa resolução resulta em efeitos gráficos muito irreais. As praias naturais possuem uma curvatura característica e dificilmente terão um comprimento de 20Km em uma configuração perfeitamente retilínea, como as praias geradas pela ferramenta. Embora o ob-

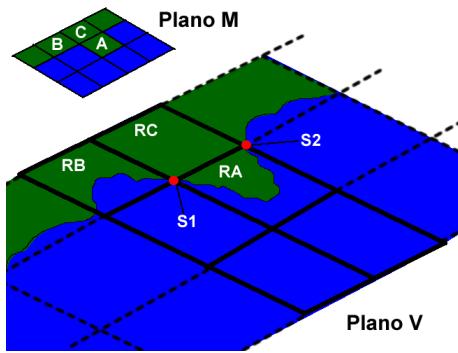


Figure 6: Funcionamento do algoritmo de quebra de linearidade da costa

jetivo do presente trabalho não seja criar paisagens fotorrealistas, praias tão irreais não são aceitáveis. Para contornar esse problema, criou-se uma forma de quebrar a linearidade da costa através da adição de conteúdo aos locais onde o mapeamento da MM é calculado entre dois pontos, um de água e outro de terra. O funcionamento do algoritmo é descrito a seguir.

A MM possui a descrição completa do que é terra e do que é água no mundo virtual. Cada um de seus pixels possui um descritor associado, que informa aos demais algoritmos da ferramenta para qual tipo de terreno um pixel do mundo virtual está sendo mapeado na MM. Inicialmente a ferramenta foi criada com três tipos de terreno: água, terra (continente) e costa (terra em contato com a água). Depois que as faixas de terra são pré-processadas e essas informações são armazenadas na MM, existem apenas informações sobre terra (continente) e água.

A partir desse momento, o primeiro passo do algoritmo de quebra de linearidade da costa é executado. Utilizando como entrada a MM atual, o algoritmo varre cada um dos seus elementos, atualizando o descritor de informação dos pixels que são costa. Um pixel é dito costa quando pelo menos um de seus vizinhos é água. Depois que o algoritmo termina o seu processamento, a MM contém os três tipos de terreno descritos anteriormente. O próximo passo para a quebra de linearidade da costa é a geração de conteúdo com base no descritor de informação de cada um dos pixels da MM. Quando a ferramenta estiver gerando conteúdo para desenhar na tela, o procedimento de geração testa qual o tipo de terreno que está descrito no mapeamento da MM para os pixels que ela está desenhando atualmente. Se o mapeamento terminar em um pixel da MM que é terra, então a função irá gerar um relevo para aquele ponto. Se o mapeamento terminar em um pixel da MM que é água, então a função irá gerar o relevo para o oceano (que é uma altura padrão representando o nível do mar). Se o mapeamento terminar em um pixel da MM que é costa, então a função irá criar alterações nas informações de terra/água esse mapeamento, o que irá resultar em uma costa não-retilínea e mais realista. A figura 6 ilustra o funcionamento do algoritmo.

Os pixels A e B da MM possuem um descritor de informação indicando que eles são costa. Alguns outros pixels da figura também são costa, porém eles não serão detalhados por fins de clareza. O Plano M descreve a MM e o Plano V descreve o resultado do mapeamento dela no mundo virtual. Embora não esteja descrito na ilustração, cada um dos blocos do Plano V é composto por vários pixels, enquanto cada bloco do Plano M corresponde a apenas um pixel da MM. O pixel C da MM é mapeado para um bloco maciço de terra no Plano V, já que o seu descritor informa à ferramenta que ele é um pixel de terra. O pixel A seria mapeado para um bloco maciço de terra também, porém com a atuação do algoritmo de quebra de linearidade ele é mapeado para uma configuração diferenciada. No momento da geração de conteúdo para os pixels do mundo virtual que estão dentro do bloco RA, o algoritmo de quebra de linearidade distorce as informações de terra/mar para cada um dos pixels, o que faz com que o bloco não seja composto inteiramente de pixels de terra.

A implementação desse processo é baseada em ruídos e números

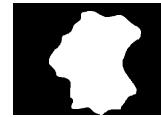


Figure 7: Representação gráfica do espectro de ruído utilizado para criação da costa

aleatórios, na qual uma função paramétrica é usada para decidir o que é terra/mar dentro de um bloco que é mapeado para costa na MM. Utilizando como base a posição do pixel dentro do bloco RA, a função mapeia essa informação para dentro de um espectro de valores criados a partir de uma função de ruído de Perlin. Em linhas gerais, o que a função faz é testar se o hash do pixel em questão está dentro ou fora do espectro; o processo pode ser imaginado como um teste de altura em um *heightmap* de pequenas proporções (que é criado como resultado do espectro de ruído): se o retorno da função de ruído para o pixel em questão for maior que um determinado valor (que é a granularidade do bloco sendo analisado), então ele é terra, caso contrário ele é água. Quanto maior a granularidade do bloco, maior será a quantidade de terra no local. A figura 7 ilustra o desenho do *heightmap* fictício gerado pela função de quebra de linearidade no bloco RA.

6.3.3 Praias

A quebra de linearidade da costa elimina em grande parte o problema de linhas irreais nos continentes, porém o resultado final ainda tende para algo que não é aceitável na natureza. Quando a ferramenta está renderizando uma fatia do mundo, para cada pixels que é descrito como terra um relevo é associado a ele; o mesmo se aplica para os pixels que são descritos como água, porém nesse caso o relevo criado possui sempre a mesma altura (o nível do mar). Como uma consequência direta disso, se a ferramenta estiver desenhando um conjunto de pixels que descreve uma cadeia montanhosa e, logo em seguida, os próximos pixels são descritos como água, a paisagem resultante apresentará um "degrau". Isso acontece porque a cadeia montanhosa foi gerada muito próxima da água, o que faz com que a sua renderização seja abruptamente interrompida no momento que a ferramenta encontra água. Embora existam falésias no mundo real, elas não estão presentes em todas as costas, somente em algumas. Para contornar esse problema, criou-se um algoritmo capaz de gerar praias em terminadas áreas, o que torna a paisagem gerada mais realista.

O algoritmo de criação de praias atua pouco antes do conteúdo ser renderizado na tela. Depois que a ferramenta mapeia para a MM os pixels e depois que o algoritmo de quebra de linearidade da costa atua, o resultado é um *heightmap* pronto para ser renderizado. Antes de ser desenhado na tela, esse *heightmap* é tratado pelo algoritmo de criação de praias. O procedimento varre cada um dos pixels existentes no mapa e, para cada um deles, checa qual a distância que o pixel atual está de um pixel de tipo água à sua volta, porém fazendo o mapeamento diretamente na MM, não no *heightmap*. A checagem é feita em quadro direções (direita, esquerda, cima e baixo), sendo que a ferramenta avança até encontrar um pixel água ou até que uma distância de N pixels seja percorrida. Em seguida, as quatro distâncias até um pixel de tipo água são somadas e utilizadas em uma fórmula para o cálculo da altura da praia. Os resultados possíveis são os seguintes:

- Se o pixel analisado estiver à uma distância de $4N$, isso quer dizer que a ferramenta percorreu às quatro direções possíveis e não encontrou água. Nesse caso, o pixel em questão não tem a sua altura recalculada; esse caso descreve o que acontece com todos os pixels que estão dentro do continente ou na costa porém longe da água: eles não formam uma praia e sua altura é definida pela função de relevo principal;
- Se o pixel analisado estiver à uma distância inferior a $4N$, então a sua altura será recalculada. Quanto maior for a distância calculada, maior será a altura do pixel, porém essa variação da altura é calculada dentro de um intervalo definido $[T, B]$, onde T é a altura máxima e B é a altura mínima de uma praia, respectivamente. O resultado dessa abordagem é

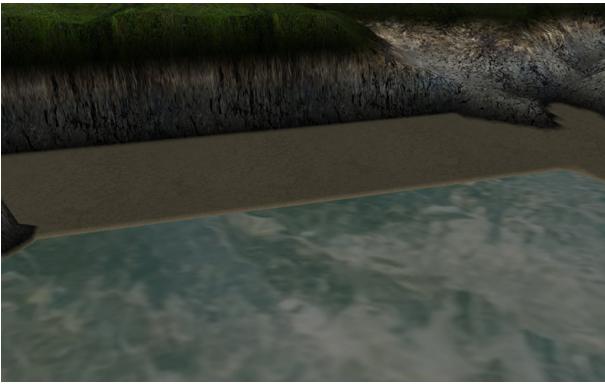


Figure 8: Praia gerada ao longo da costa



Figure 9: Resultado obtido com a aplicação do diferenciador de praias

uma praia que inicia alta no continente e, à medida que se aproxima da água, fica mais baixa.

A figura 8 ilustra o resultado obtido com a geração de prais.

6.3.4 Arquipélagos e praias diferenciadas

Utilizando os algoritmos de quebra de linearidade e criação de praias ao longo da costa, a ferramenta passou a gerar paisagens mais realistas. O resultado final em relação à costa e à praia, porém, apresentou um padrão muito definido, o que é incomum de acontecer no mundo real, no qual as linhas e paisagens naturais tendem a seguir um princípio aleatório ou menos padronizado. Se o usuário viajasse pelo mundo virtual apenas pela costa, ele veria prais com a mesma configuração (mesmo tamanho) e nenhuma ilha ou arquipélago ao longo do oceano. Para melhorar esse aspecto, criaram-se dois novos algoritmos que atuam na costa: um diferenciador de prais e um gerador de arquipélagos.

O **diferenciador de prais** atua perturbando a distância utilizada para o cálculo dos pixels água vizinhos a um determinado pixel. Em vez de utilizar uma distância fixa N para o cálculo da distância até a água, o diferenciador utiliza a posição do pixel como semente para uma função de ruído, que tem como retorno a nova distância que será utilizada nos cálculos. Utilizando essa técnica, o diferenciador é capaz de alterar o tamanho e forma da praia, o que faz com que determinadas regiões apresentem uma maior quantidade de areia do que outras. A figura 9 ilustra os resultados obtidos.

O **gerador de arquipélagos** atua criando novas faixas de terra em terminados pixels da MM. Depois que a MM é criada e todos os descriptores de informação de cada um de seus pixels é configurado, o gerador de arquipélago itera sobre os pixels que representam a costa e, para alguns deles, adiciona a informação que essa região possui ilhas. No momento em que a ferramenta estiver renderizando os pixels que são mapeados para essa região da MM, o desritor de informação será consultado e a ferramenta saberá que essa região necessita de um conteúdo novo, além do conteúdo utilizado para a quebra de linearidade da costa. Esse conteúdo é criado utilizando

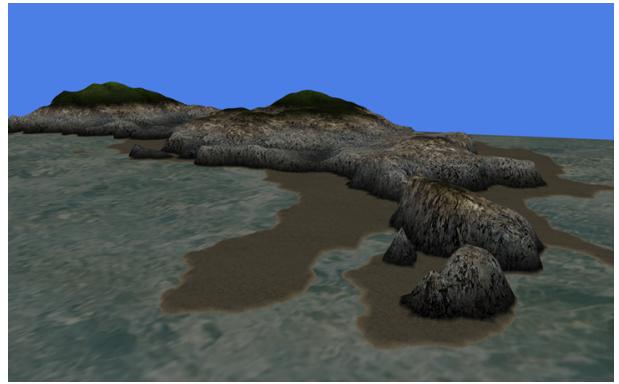


Figure 10: Ilha obtida com o gerador de ilhas

a mesma abordagem do algoritmo de quebra de linearidade, porém utilizando como pixels de análise aqueles pixels que são água. Para cada pixel sendo analisado, a sua posição é utilizada como um hash que é testado contra um espectro criado por uma função de ruído. Os ruídos utilizados para esse espectro são diferentes daqueles do algoritmo de quebra de linearidade, visto que o resultado esperado são pequenas porções de terra, não um bloco maciço dela. Para conseguir esse efeito, aumentou-se o número de oitavas da função de ruído de Perlin e aumentou-se o limite que é utilizado nos testes para saber se o pixel é ou não terra. O resultado do gerador de arquipélagos é combinado com o algoritmo de quebra de linearidade, o que faz com que as ilhas sejam geradas, em determinados casos, muito próximas à costa. O resultado final obtido com o gerador de arquipélagos são ilhas de diversos tamanhos ao longo da costa em determinados locais do mundo virtual. A figura 10 ilustra os resultados obtidos.

7 Resultados

Essa seção tem por objetivo avaliar cada uma das técnicas utilizadas na geração de conteúdo da ferramenta, mostrando os resultados que foram obtidos com cada abordagem. Leva-se em consideração a flexibilidade proporcionada pela ferramenta durante a utilização da funcionalidade e o resultado gráfico alcançado.

É importante frisar que o objetivo da ferramenta, no escopo do presente trabalho, não é a geração de conteúdos reais ou foto-realísticos, mas sim de elementos que possam ser utilizados para a criação de um cenário num jogo 3D. Entende-se por graficamente aceitável todo o resultado obtido que se enquadre dentro de um jogo e não destoe daquilo esperado pelo jogador, como uma montanha composta por ondulações senoidais suaves ao invés de um conjunto de cristas piramidais.

7.1 Avaliação dos continentes

O tempo para a geração dos continentes é diretamente proporcional ao tamanho da macro-matriz especificada. Em testes realizados, a redução da macro-matriz para valores inferiores a 800×800 rendeu aumentos de desempenho consideráveis. Em contra partida, quanto menor o tamanho da macro-matriz, mais quadrados e lineares serão as linhas da costa de cada um dos continentes, o que pode produzir um resultado gráfico não muito aceitável. Para contornar esse problema, é possível ajustar o algorítimo de geração da costa para que ele faça alterações mais agressivas nas bordas dos continentes, o que irá quebrar a linearidade gerada por uma macro-matriz de baixa resolução. O ajuste desses dois elementos abre um leque de possibilidades para o programador, que pode encontrar um ponto de equilíbrio entre tempo de geração dos continentes e linearidade da costa. A figura 11 mostra os resultados obtidos em relação a continentes e oceanos gerados pela ferramenta.

7.2 Avaliação do relevo

O relevo gerado pela ferramenta é inteiramente parametrizável e customizável pelo programador. Segundo o protótipo da função

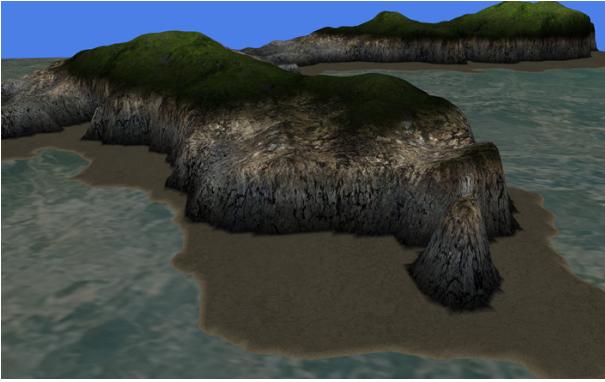


Figure 11: Continentes e oceanos gerados pela ferramenta

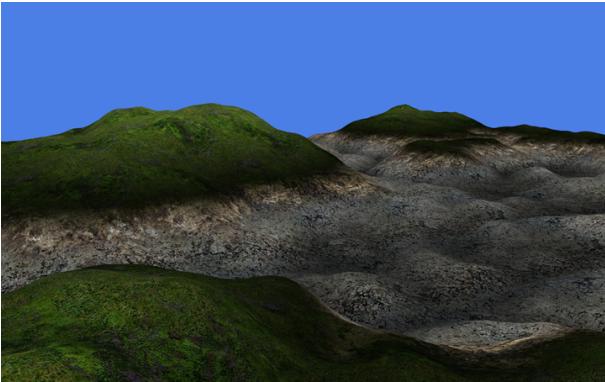


Figure 12: Relevo gerado pela ferramenta

esperada pela ferramenta para o cálculo do relevo, o programador pode criar qualquer código para a geração de relevo, o que garante flexibilidade nesse aspecto. A ferramenta possui embutida um gerador de relevo baseado na função de ruído de Perlin, que produz uma paisagem semelhante àquela encontrada no mundo real. A figura 12 ilustra o relevo criado pela ferramenta utilizando-se as funções de geração embutidas.

A função de relevo que foi implementada na ferramenta é capaz de gerar paisagem que imitam de forma aceitável o que é encontrado na natureza, porém a grande maioria dos esforços de desenvolvimento não foram empregadas sobre esse tópico. Como consequência, a ferramenta não possui um grande leque de possibilidades de geração de conteúdos para o interior dos continentes, como cadeias montanhosas, planícies e planaltos. O relevo gerado atualmente é simplista no sentido de não apresentar grandes diversidades geológicas, entretanto se o objetivo da utilização da ferramenta não for criar um terreno rico em diversidade, o objetivo é plenamente atingido.

A geração atual de relevo foi desenvolvida visando-se a obtenção de ondulações de média/baixa frequência nas funções de ruído, o que faz com que não existam cadeias montanhosas pontudas ou depressões abruptas. A baixa frequência nos ruídos da função de geração de relevo faz com que o usuário veja montanhas com um ângulo muito suave de inclinação, além de áreas com poucas ondulações, que podem ser interpretadas como planaltos. A figura 13 ilustra uma área do mundo virtual que possui relevo com baixa frequência.

Se a função de geração de relevo utilizar frequências muito baixas e o seu espectro de valores for estendido à todo o mundo virtual, as montanhas e ondulações que serão produzidas serão muito suaves. Esse fenômeno acontece porque baixas frequências não criam alterações acentuadas de altura nos polígonos do relevo. Na utilização de altas frequências, a grande quantidade de ruído cria montanhas mais pontiagudas, porém as áreas intermediárias entre elas são muito onduladas, que é o reflexo da propagação de um espectro de valores muito ruidoso para o mundo virtual. Uma das formas de contornar esse problema é replicar a extensão do espec-

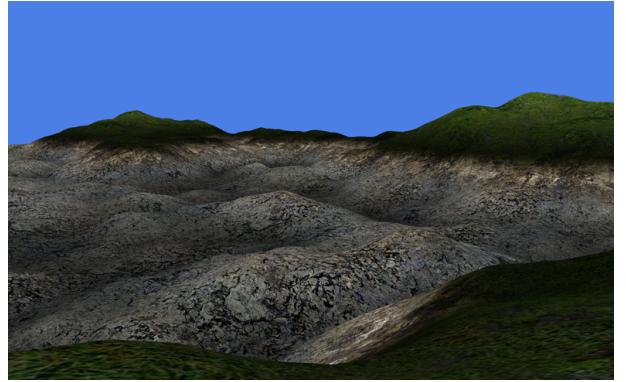


Figure 13: Área com relevo de baixa frequência

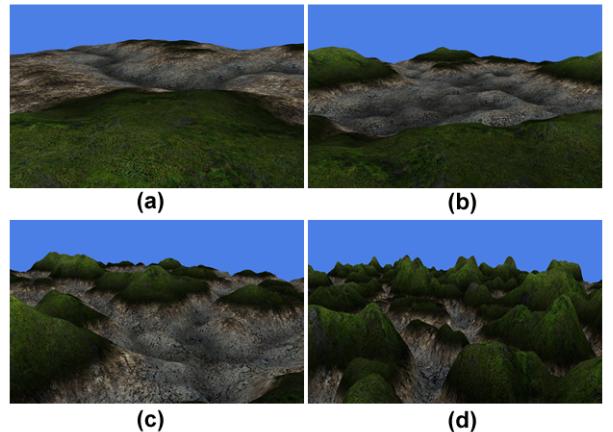


Figure 14: Diversos relevos gerados a partir de diferentes espectros de ruídos. (a) Pouco ruído estendido para o mundo inteiro sem replicação. (b) Pouco ruído replicado 200 vezes. (c) Bastante ruído com replicação de 200 vezes. (d) Bastante ruído sem replicação.

tro de valores, ou seja, em vez de utilizar um espectro que cubra o mundo inteiro, utilizar esse mesmo espectro três vezes ao longo do mundo. Isso permite que pouco ruído seja utilizado, porém evita que o relevo resultante tenda ao plano. A ferramenta utiliza essa abordagem para criar um relevo com montanhas consideravelmente onduladas, porém sem áreas de ondulação estranha entre elas. A figura 14 ilustra os diferentes tipos de relevo obtidos com a variação do tamanho do espectro de valores.

7.3 Avaliação da costa

A geração da costa é composta por dois pilares principais, um de aspecto mais global e outro mais local. No aspecto global, a ferramenta utiliza apenas dados encontrados na MM para criar as linhas que compõem a costa, conforme descrito na seção 6.3.1. O resultado final para essa abordagem são costas completamente retilíneas, o que é irreal do ponto de vista do usuário. A figura 15 ilustra duas linhas da costa completamente retilíneas.

Depois que o método para quebra de linearidade da costa foi implantada, a ferramenta passou a apresentar paisagens mais aceitáveis. As figuras 16, 17 e 20 ilustram a criação de pequenas baías em certos locais da costa resultantes da aplicação do algoritmo. Isso acontece porque nesses locais o algoritmo de quebra de linearidade da costa criou braços de terra curvos partindo da linha reta do continente e, ao mesmo tempo, o algorítimo de criação de praias reduziu ao máximo a quantidade de areia encontrada no local. A ferramenta também é capaz de criar golfos, que são baías de grandes proporções, porém não é possível prever o local exato que isso irá acontecer, porque tal resultado depende da combinação de valores e coordenadas que variam sensivelmente ao longo do mundo virtual.

As figuras 18 e 19 ilustram a adição de conteúdo à costa

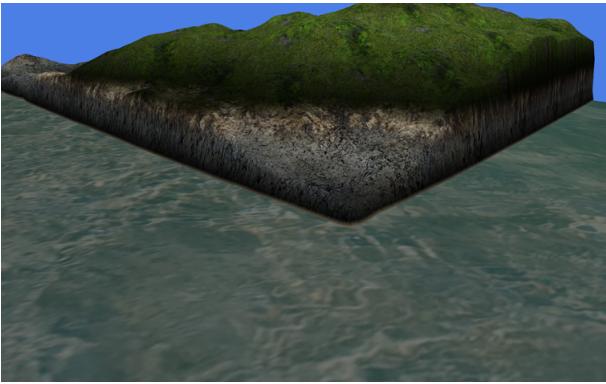


Figure 15: Local de encontro de duas linhas da costa sem a aplicação de qualquer algoritmo de geração de conteúdo extra



Figure 16: Pequena baía com rochas

retilínea do continente, resultando numa paisagem mais convincente para o usuário. As duas figuras mostram com bastante detalhe a combinação da função de geração de relevo, o diversificador de praias e o algoritmo de quebra de linearidade da costa na criação de conteúdo; a figura 18 apresenta uma rocha imediatamente à esquerda da falésia principal, resultado obtido a partir de um espectro de ruído que gerou dois pontos principais indicando a existência de terra: o maior sendo a falésia e o menor sendo a rocha grande à esquerda. Além disso, o diversificador de praias removeu a areia na parte de baixo da falésia, porém à esquerda ele fez o processo contrário. Em conjunto, o algoritmo de geração de relevo com replicação de valores criou pontas na extremidade esquerda da falésia.

A 19 apresenta a mesma combinação do algoritmo da figura anterior, porém o resultado obtido variou em função da coordenada do mundo virtual no local. Ao contrário do que aconteceu anteriormente, o diversificador de praias adicionou areia à base da falésia e, à esquerda da imagem, criou uma extensão da praia em forma de "braço". Também à esquerda, o diversificador de praias removeu a areia que fica na base do continente, criando um aspecto menos padronizado de conteúdo. À direita, é possível observar que o algoritmo de geração de relevo criou um declive que termina de forma relativamente suave na praia (relevo de baixa frequência), ao passo que na parte esquerda da imagem o relevo termina de uma forma mais abrupta (relevo de alta frequência).

8 Conclusão

A criação automatizada de mundos virtuais procedimentais é uma das formas existentes para auxiliar desenvolvedores de jogos a criarem ambientes mais ricos em detalhes, em menos tempo e com menos recursos humanos. Ao contrário da abordagem puramente não automatizada, no qual um *game designer* ou artista deve desenhar e modelar o mundo virtual por completo, na abordagem automatizada a adição de detalhes e modelagem de locais fica a cargo da ferramenta geradora. Dependendo do grau de realismo e da complexidade dos algoritmos de geração de conteúdo, uma ferramenta



Figure 17: Baía de médio porte gerada a partir da criação de um braço de terra originada no continente



Figure 18: Extremidade de um continente



Figure 19: Costa de um continente com praias de tamanho variável



Figure 20: Baía criada a partir da geração de dois braços de terra originados no continente

automatizada pode ser capaz de gerar um mundo virtual aceitável para diversos escopos de projeto.

Existem diversas pesquisas em relação à essa área, com abordagens diferentes e focadas em resultados diferenciados. Pode-se destacar nelas a preocupação em criar um mundo virtual de proporções infinitas sem que isso comprometa o desempenho da aplicação. Para atingir esses resultados, diversas técnicas são utilizadas, como aplicação de LOD e quadtrees para gerenciamento de malhas, além de diversos algoritmos procedimentais para a geração de conteúdo, como criação de relevo a partir de fractais, multi-fractais e/ou combinação de funções de ruído.

Este trabalho apresentou o desenvolvimento de uma ferramenta capaz de gerar mundos virtuais pseudo-infinitos com diversificação de formas e relevos ao longo de sua extensão. Utilizando uma combinação de algoritmos e métodos de gerenciamento de conteúdo, a ferramenta é capaz de criar praias, ilhas/arquipélagos, baías e costas que imitam as paisagens encontradas na natureza. Além disso, a possibilidade de parametrizar cada um desses elementos dá ao desenvolvedor um controle maior sobre o resultado que será obtido.

Dentre as inovações apresentadas, estão a criação de um terreno virtual de vastas proporções com enfoque mais detalhado no que diz respeito à geração da costa. Fruto de um aprofundamento nas pesquisas já realizadas na área, o desenvolvimento da ferramenta foi focado em tratar de forma diferenciada a criação de conteúdo para os diversos elementos existentes (como continentes, relevo, etc). Uma das prioridades do trabalho foi a criação de bordas dos continentes, não a criação de conteúdo para o seu interior; esse relevo está aceitável, porém ainda está muito aleatório e sem grandes resultados. A ferramenta é capaz de gerar continentes contendo falésias, praias e rochedos em suas extremidades. Em testes realizados, o tempo que um jogador levaria para chegar de uma ponta do mundo virtual até a outra num ambiente com o tamanho máximo suportado por um inteiro sem sinal, andando 100 pixel por segundo, seria de um 1 ano e 3 meses.

References

- BLIZZARD ENTERTAINMENT, 2007. World of warcraft. Disponível em: <http://www.blizzard.com>.
- CLARK, N. L. 2006. *Addiction and the Structural Characteristics of Massively Multiplayer Online Games*. Master's thesis, University of Hawai, Hawai.
- COHEN, M., SHADE, J., HILLER, S., AND DEUSSEN, O. 2003. Wang tiles for image and texture generation. In *Siggraph 03 Conference proceedings*.
- DOLLINS, S. C. 2002. *Modeling for the Plausible Emulation of Large Worlds*. PhD thesis, Brown University, Estados Unidos da América.
- DUCHENEAUT, N., YEE, N., NICKELL, E., AND MOORE, R. J. 2006. Alone together exploring the social dynamics of massively multiplayer online games. In *CHI 2006 Proceedings*.
- GREUTER, S., PARKER, J., STEWART, N., AND LEACH, G., 2005. Realtime procedural generation of 'pseudo infinite' cities.
- GRIFFITHS, M. D., DAVIES, M. N., AND CHAPPELL, D., 2003. Breaking the stereotype the case of online gaming.
- HÄGGSTRÖM, H. 2006. *Real-time generation and rendering of realistic landscapes*. Master's thesis, University of Helsinki, Finlândia.
- HÄGGSTRÖM, H., 2009. Skycastle - free multiplayer game engine focusing on player creativity and world simulation. Disponível em: <http://www.skycastle.org>.
- JASONWEBER, AND PENN, J., 1995. Creation and rendering of realistic trees.
- LEFEBVRE, S., AND NEYRET, F. 2003. Pattern based procedural textures. In *Proceedings of the 2003 symposium on Interactive 3D graphics*.
- LINDA, O. 2007. Generation of planetary models by means of fractal algorithms. Tech. rep., Czech Technical University.
- LINTERMANN, B., AND DEUSSEN, O., 1998. A modelling method and user interface for creating plants. Computer Graphics Forum.
- MOGENSEN, T., 2009. Instant planet generator. Disponível em: <http://www.eldritch.org/erskin/roleplaying/planet.php>.
- MUSGRAVE, F. K., KOLB, C. E., AND MACE, R. S., 1989. The synthesis and rendering of eroded fractal terrains. Computer Graphics, Volume 23, Number 3, July 1989, pages 41 to 50.
- OLSEN, J., 2004. Realtime synthesis of eroded fractal terrain for use in computer games.
- OPENGL, 2007. Opengl shading language. Disponível em: <http://www.opengl.org/documentation/glsli>.
- PRZEMYSŁAW, P., AND LINDEMAYER, A., 1990. *The algorithmic beauty of plants*. Springer-Verlag.
- SONY ENTERTAINMENT, 2007. Everquest. Disponível em: <http://everquest2.station.sony.com>.
- WANG, T., 2000. Integer hash function. Available at: <http://www.concentric.net/Ttwang/tech/inthash.html>.