

**UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA**

**FERRAMENTA PARA GERAÇÃO DE
MUNDOS VIRTUAIS
PSEUDO-INFINITOS PARA JOGOS 3D
MMO**

DISSERTAÇÃO DE MESTRADO

Fernando Bevilacqua

Santa Maria, RS, Brasil

2008

FERRAMENTA PARA GERAÇÃO DE MUNDOS VIRTUAIS PSEUDO-INFINITOS PARA JOGOS 3D MMO

por

Fernando Bevilacqua

Dissertação apresentada ao Programa de Pós-Graduação em Informática
da Universidade Federal de Santa Maria (UFSM, RS), como requisito
parcial para a obtenção do grau de
Mestre em Informática

Orientador: Prof. Dr. Cesar Tadeu Pozzer (UFSM)

**Dissertação de Mestrado Nº 2
Santa Maria, RS, Brasil**

2008

**Universidade Federal de Santa Maria
Centro de Tecnologia
Programa de Pós-Graduação em Informática**

A Comissão Examinadora, abaixo assinada,
aprova a Dissertação de Mestrado

**FERRAMENTA PARA GERAÇÃO DE MUNDOS VIRTUAIS
PSEUDO-INFINITOS PARA JOGOS 3D MMO**

elaborada por
Fernando Bevilacqua

como requisito parcial para obtenção do grau de
Mestre em Informática

COMISSÃO EXAMINADORA:

(Presidente/Co-orientador)

Prof. Dr. Gerson Geraldo Homrich Cavalheiro (UFPel)

Prof^a Dr^a Iara Augustin (UFSM)

Santa Maria, 22 de Agosto de 2008.

“Não sabendo que era impossível, ele foi lá e fez.” — JEAN COCTEAU

RESUMO

Dissertação de Mestrado
Programa de Pós-Graduação em Informática
Universidade Federal de Santa Maria

FERRAMENTA PARA GERAÇÃO DE MUNDOS VIRTUAIS PSEUDO-INFINITOS PARA JOGOS 3D MMO

Autor: Fernando Bevilacqua

Orientador: Prof. Dr. Cesar Tadeu Pozzer (UFSM)

Local e data da defesa: Santa Maria, 22 de Agosto de 2008.

Resumo em português aqui.

Palavras-chave: MMO, mundos virtuais, geração de terreno, jogos 3D.

ABSTRACT

Master's Dissertation
Programa de Pós-Graduação em Informática
Universidade Federal de Santa Maria

VXDL: A LANGUAGE FOR INTERCONNECTION AND RESOURCES SPECIFICATION IN VIRTUAL GRIDS

Author: Fernando Bevilacqua
Advisor: Prof. Dr. Cesar Tadeu Pozzer (UFSM)

Grid computing has been defined as an infrastructure integrator of distributed resources. Although it is already used on a large scale in many areas, this type of computational infrastructure is still an area of active research, with many open questions. Today, new research works investigate the application of resources virtualization techniques to perform the composition of virtual grids. These grids can be defined as a high level abstraction of resources (computing and network), through which users have a view of a wide range of interconnected computers, that can be selected and virtually organized. In a virtual grid, as well in a real grid, users and middleware must have tools that allow the composition and management of the infrastructure. Among these tools, there are languages for resource description that allow the specification of components that will be used in the infrastructure. In a virtualized environment, the resources descriptions languages should offer attributes that interact with some peculiarities, such as the possibility of allocate multiple virtual resources (computing and network) on the same physical resource. In this context, this work presents VXDL, a language developed for the interconnections and resources description in virtual grids. The innovations proposed in VXDL allow the description, classification and parameter specification of all desirable components, including network topology and virtual routers. VXDL also allow the specification of a execution timeline, which can assist grid middleware in the tasks of resources sharing and scheduling. To evaluate the proposed language, this work presentes I) a comparative study between VXDL and other resources description languages and II) an analysis of results obtained with the benchmarks execution in virtual infrastructures composed using different VXDL descriptions.

Keywords: virtualization, virtual grids, virtual clusters, resources description language.

LISTA DE FIGURAS

Figura 3.1 – Problema da geração de conteúdo sob demanda.	12
Figura 3.2 – Organização do sistema de coordenadas do mundo virtual	14
Figura 3.3 – Campo de visão do usuário: a área visualizada corresponde a uma fatia do mundo virtual existente.	15
Figura 3.4 – Mapeamento das coordenadas do mundo virtual para as coordenadas de desenho da tela.....	16

LISTA DE TABELAS

SUMÁRIO

1	INTRODUÇÃO	10
1.1	Contexto e Motivação	10
1.2	Objetivos e Contribuição	10
1.3	Organização do Texto	10
2	REVISÃO DE LITERATURA	11
2.1	Mundos pseudo-infinitos.....	11
2.2	Funções de ruído	11
2.3	Fractais para geração de relevo.....	11
3	IMPLEMENTAÇÃO.....	12
3.1	Terreno infinito	13
3.1.1	Abordagem utilizada	13
3.1.2	Renderização de conteúdo	16
3.2	Continentes.....	17
3.3	Relevo	17
4	RESULTADOS	18
5	CONCLUSÃO E TRABALHOS FUTUROS	19
	REFERÊNCIAS	20

1 INTRODUÇÃO

1.1 Contexto e Motivação

1.2 Objetivos e Contribuição

1.3 Organização do Texto

isso eh um texto [1]

2 REVISÃO DE LITERATURA

2.1 Mundos pseudo-infinitos

– falar aqui das zilhões de abordagens para geração de mundos infinitos que nós encontramos. Parece que todo mundo resolveu fazer pesquisa sobre isso de uma hora para outra.

– Aqui vai a explicação sobre o artigo frances sobre geração de cidades pseudo-infinitas (nosso artigo semente), aquela engine para geração de mundos virtuais, aquela outra ferramenta para geração de terrenos infinitos.

– <http://landscapestudio.omgames.co.uk/screenshots.html> – <http://www.vterrain.org/>
 – <http://www.pandromeda.com/products/mojoworldpro.php> – <http://www.wcg.in.tum.de/Research/Publication>
 – <http://www.howardzzh.com/research/terrain/> – <http://www.cs.brown.edu/scd/world/home.html>
 (exatamente o que queremos fazer) – <http://www.earth3d.org/>

2.2 Funções de ruído

– Como utilizamos muito funções de ruído e afins, falar aqui do Perlin e do Musgrave. Falar bastante do livro deles, porque é uma coisa que é bem relacionada com o nosso trabalho e que a gente vai utilizar bastante.

– Só utilizamos o noise do Perlin aqui e o conceito de multi-fractal, mas sobre fractais eu vou falar depois.

2.3 Fractais para geração de relevo

– Falar aqui das 3 formas que encontramos para geração de relevo através de fractais (deposição de sedimentos, alteração do ponto médio e divisão de não sei o que).

– Falar também sobre os multifractais que tem no livro do Musgrave, que ele usa para geração de costas de continentes muito bonitas.

3 IMPLEMENTAÇÃO

A ideia original do trabalho foi desenvolver um mundo virtual completo, semelhante em grande parte com o mundo real. Dentre as funcionalidades previstas, encontravam-se a divisão do terreno em relevos específicos (desertos, florestas, planícies, etc), cidades/vilas, caminhos entre as cidades, rios e cadeias montanhosas. A combinação de todos esses elementos seria capaz de criar um mundo virtual muito próximo da realidade, fato que seria de suma importância para garantir um bom resultado da ferramenta.

Quando o planejamento foi finalizado, a complexidade de determinadas funcionalidades previstas tornou proibitiva a sua implementação. A grande maioria dos problemas encontrados é uma consequência da abordagem de geração dinâmica de conteúdo sob demanda (a medida que o usuário se move, novos elementos são colocados na tela). A Figura 3.1 ilustra o problema da 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

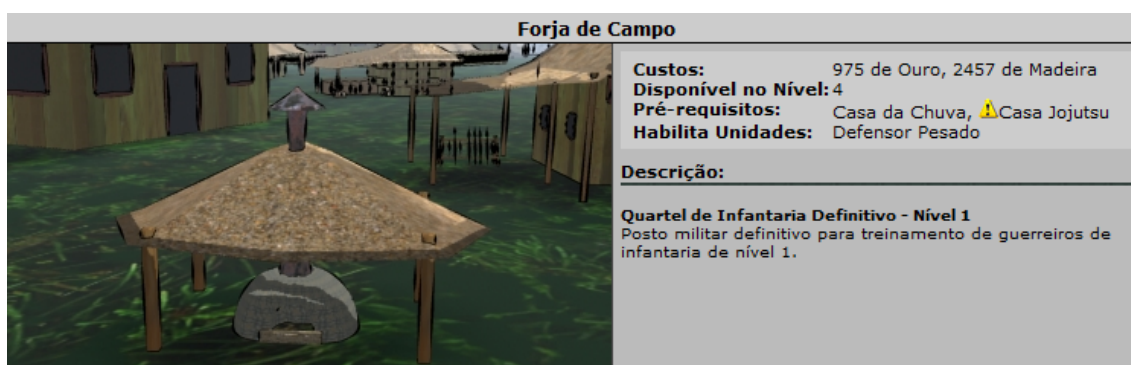


Figura 3.1: Problema da geração de conteúdo sob demanda.

a complexidade dos algoritmos envolvidos na ferramenta.

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.

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.

3.1 Terreno infinito

3.1.1 Abordagem utilizada

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 [?]. Na abordagem dos autores em questão, o mundo virtual pseudo-infinito é dividido em células quadradas e, à medida que o usuário anda, as células são adicionadas e/ou re-

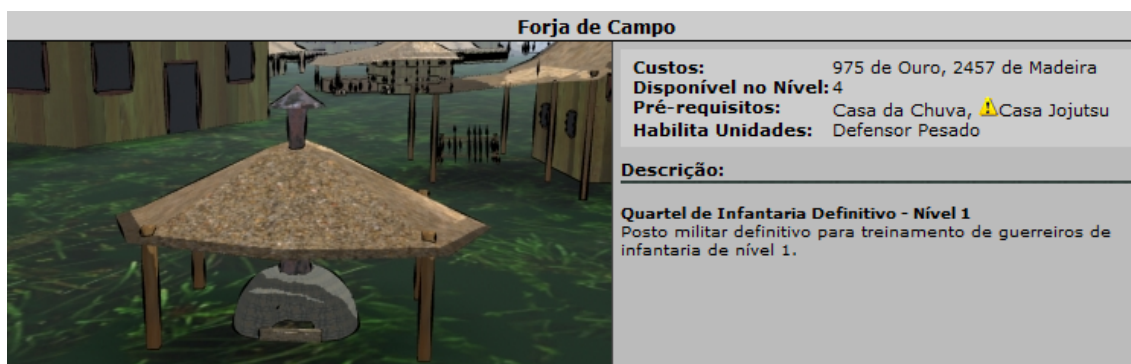


Figura 3.2: Organização do sistema de coordenadas do mundo virtual

movidas do campo de visão. Cada célula possui um conteúdo próprio e auto-contido, ou seja, a célula não precisa de informações de vizinhos para gerar o seu conteúdo. Isso garante que as células não entrem em uma dependência recursiva infinita entre elas para conseguirem gerar o seu conteúdo. Além disso, essa abordagem é vantajosa para garantir o uso racional de recursos, uma vez que só serão carregados para a memória os blocos que o usuário realmente consegue ver.

Para o posicionamento do usuário no mundo virtual, os autores utilizam um vetor 3D no formato (x, y, z) . Conforme o usuário se move horizontalmente pelo mundo, as coordenadas x e y são atualizadas. Se o usuário se move verticalmente, a coordenada Z é alterada. A abordagem utilizada para a ferramenta dessa dissertação baseou-se nesses conceitos. A figura 3.2 ilustra a organização do mundo virtual.

A origem do mundo virtual é o ponto $(0, 0, 0)$ e os eixos que definem o plano horizontal são o X e Z , sendo a altura controlada pelo eixo Y . A distância máxima que o usuário consegue percorrer em qualquer um dos eixos é o número máximo suportado por um inteiro de 32 bits com sinal.

O que o jogador consegue ver na tela em um determinado momento é um pedaço do mundo virtual existente. Esse pedaço foi chamado de *view frustum*, ou campo de visão. Diferentemente do que foi feito em [?], no qual o campo de visão é um cone, o campo de visão da presente ferramenta é um quadrado centrado no usuário. A figura 3.3 ilustra o funcionamento do campo de visão.

A partir da posição (x, y, z) do usuário, a ferramenta calcula qual é o conteúdo visualizável ao redor do referido ponto. 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.

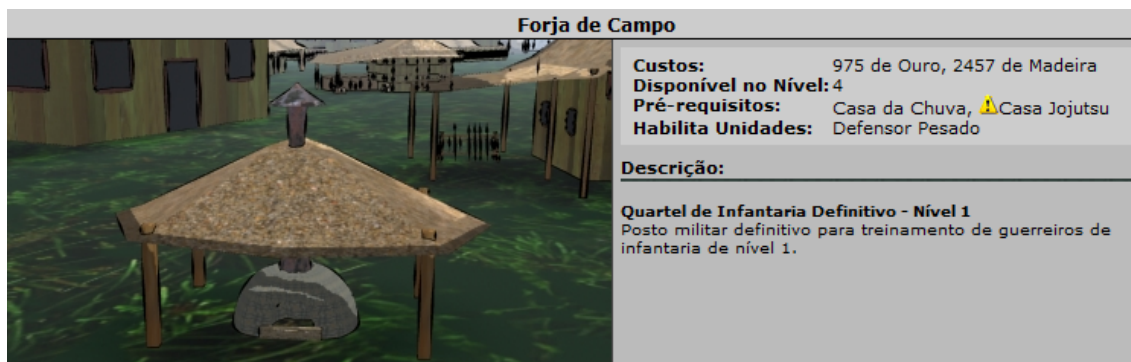


Figura 3.3: Campo de visão do usuário: a área visualizada corresponde a uma fatia do mundo virtual existente.

Inicialmente planejou-se a utilização de quadrados para dividir o mundo virtual em células, conforme é feito na outra abordagem citada anteriormente. A utilização de células garantiria que o mundo virtual fosse subdividido em blocos menores, o que viabilizaria um melhor controle sobre o que o usuário consegue ver no seu campo de visão e, também, um melhor controle sobre a geração de conteúdo. O maior problema encontrado nessa abordagem, que foi a razão pela qual ela foi abandonada, é a dependência que as células precisam ter entre si para que o terreno infinito seja gerado.

Utilizando o exemplo da geração de relevo, que é tratada em mais detalhes na seção ??, se o mundo virtual fosse dividido em células, cada uma delas deveria possuir um relevo perfeitamente nivelado com a célula vizinha, caso contrário o relevo gerado teria diversos "degraus". Analisando o problema um pouco mais a fundo, no caso da geração de montanhas, por exemplo, se na metade de uma célula a ferramenta decidisse que uma cadeia montanhosa deveria começar, a célula vizinha deveria obrigatoriamente ter a continuação dessa cadeia montanhosa, caso contrário a montanha em questão seria fatiada pela metade. A forma de corrigir esse problema é adicionar um nível mínimo de dependência entre as células: o conteúdo de uma célula é gerado com base nas informações da própria célula e também com base em alguma "dica" da célula vizinha. No exemplo da cadeia montanhosa, a célula vizinha ao começo da cadeia saberia que o conteúdo que ela deve gerar é a continuação da cadeia montanhosa, visto que a sua célula vizinha possui o começo da cadeia.

Essa dependência de conteúdo gera um encadeamento recursivo infinito entre as células. Se a célula A, por exemplo, for gerar o seu conteúdo, ela irá fazer isso com base nas suas informações e também com base nas informações de sua célula vizinha, B. A célula

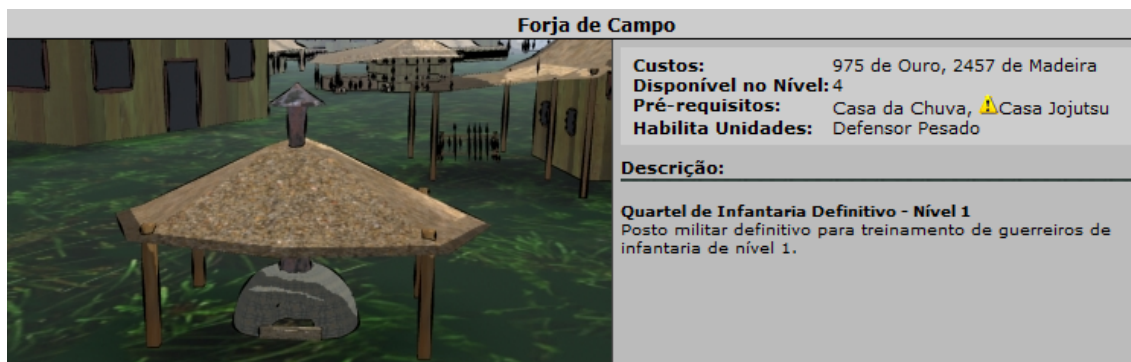


Figura 3.4: Mapeamento das coordenadas do mundo virtual para as coordenadas de desenho da tela

B, por sua vez, só poderá informar a A o seu próprio relevo quando ela o gerar; para gerar o seu relevo, ela precisa das suas informações e das informações da sua vizinha, C, e C precisa de D e assim por diante. Dessa forma, para gerar o conteúdo de A, a ferramenta teria que obrigatoriamente percorrer todas as células do mundo virtual.

Uma saída alternativa para esse problema da recursão infinita é definir um nível de consulta de informações. Embora essa abordagem limite o nível de consulta entre as células vizinhas, ela não soluciona o problema de continuidade de conteúdo. Se o conteúdo de A for gerado com seis níveis de recursão, por exemplo, quando o usuário se mover, novas células vizinhas serão consultadas para a geração do conteúdo; se a última célula que A consultou foi G, depois que o usuário se mover, G terá novas informações para o seu relevo, porque agora ela pode solicitar informações de suas vizinhas. Isso fará com que todas as células dependentes de G mudem o seu conteúdo, o que resultaria em um relevo diferente a cada movimentação do usuário. Em virtude da complexidade descrita e dos problemas mapeados, a abordagem de divisão do mundo virtual em células foi abandonada e substituída pelo modelo de campo de visão quadrado.

3.1.2 Renderização de conteúdo

Depois que o modelo de controle da geração de terrenos infinitos foi definido, iniciaram-se os trabalhos de computação gráfica para que as informações pudessem ser desenhadas na tela. Embora o mundo virtual tenha coordenadas pseudo-infinitas, o máximo de conteúdo que o usuário enxerga na tela é a área do campo de visão, que possui sempre o mesmo tamanho e coordenadas de desenho. A figura 3.4, quadrado 1, ilustra as coordenadas envolvidas no desenho do conteúdo do campo de visão.

O plano A representa o mundo virtual gerado pela ferramenta, enquanto o plano B

representa a fatia do mundo virtual que o usuário consegue visualizar. A medida que o usuário se move, o centro do campo de visão é alterado e o usuário passa a ver novos conteúdos. Independente da movimentação que o usuário faça, o plano B pode sempre ser mapeado como se estivesse na origem, porque ele é apenas uma fatia do mundo virtual. O que a ferramenta faz para desenhar esse conteúdo na tela é extrair essa fatia e, então, mapeá-la para um mapa de altura (*height map*). Depois que o mapa de altura é definido, as coordenadas dos eixos X e Z do mundo virtual que foram utilizadas durante a extração não são mais relevantes para o processo de desenho. Dessa forma, depois de extraído, o mapa de altura possui sempre as mesmas coordenadas nos eixos X e Z , que são as coordenadas de desenho da tela. A única informação do mundo virtual que é mantida é a altura de cada um dos pontos da malha do mapa de altura, conforme ilustra a figura 3.4, quadrado 2.

3.2 Continentes

– falar que utilizamos o algoritmo maluco do professor dos EUA para fazer a costa. Falar que existe o mundo é gigante o suficiente para fazer com que uma matrix que descreve água/terra seria inviável. Para solucionar esse problema, falar que utilizamos o conceito de isLang local e isLang global. O isLand global dá uma dica se o lugar é água ou terra, e o isLand local utiliza multifractais para fazer o desenho das bordas dos continentes.

– É importante frisar que essa seção é onde está a nossa contribuição na pesquisa: mundo pseudo-infinito, com relevo gerado on-the-fly e com costas de continentes com multiresolução.

3.3 Relevo

sub:relevo

– O relevo é gerado on-the-fly por funções matemáticas (explicar essas funções); falar que isso é totalmente customizável e que a qualquer momento o relevo pode ser mostrado.

4 RESULTADOS

5 CONCLUSÃO E TRABALHOS FUTUROS

–Trabalhos futuros: – Melhoramento nas funções que geram relevo; – Texturização conforme altura do terreno e afins; – Rios; – GPU;

REFERÊNCIAS

- [1] A. Aho and J. D. Ullman. *Compiladores: conceitos, técnicas e implementação*. Makron Books, 1993.