UNIVERSIDADE FEDERAL FLUMINENSE

DIOGO TAVARES ROBAINA

Um visualizador para navegação através de cenários tridimensionais baseado em malhas adaptativas

NITERÓI

2006

Um visualizador para navegação através de cenários tridimensionais baseado em malhas adaptativas

DIOGO TAVARES ROBAINA

Dissertação de Mestrado submetida ao Programa de Pós-Graduação em Computação da Universidade Federal Fluminense como requisito parcial para a obtenção do título de Mestre em Computação. Área de concentração: Modelagem Computacional.

Orientadores:

Mauricio Kischinhevsky

Otton Teixeira da Silveira Filho

Niterói, Setembro de 2006.

Ficha Catalográfica elaborada pela Biblioteca da Escola de Engenharia e Instituto de Computação da UFF

R628 Robaina, Diogo Tavares.
Um visualizador para navegação através de cenários tridimensionais baseado em malhas adaptativas / Diogo Tavares Robaina. – Niterói, RJ : [s.n.], 2006.
70 f.

Orientadores: Maurício Kischinhevsky e Otton Teixeira da Silveira Filho.
Dissertação (Mestrado em Computação) - Universidade Federal Fluminense, 2006.

1. Computação gráfica. 2. Modelagem computacional. 3. Métodos numéricos. I. Título.

CDD 006.6

Um visualizador para navegação através de cenários tridimensionais baseado em malhas adaptativas

Diogo Tavares Robaina

Dissertação de Mestrado submetida ao Programa de Pós-Graduação em Computação da Universidade Federal Fluminense como requisito parcial para a obtenção do título de Mestre em Computação. Área de concentração: Modelagem Computacional.

Aprovada por:

Prof. Mauricio Kischinhevsky / IC - UFF (Presidente)

Prof. Otton Teixeira da Silveira Filho / IC - UFF

Prof. Anselmo Antunes Montenegro / IC - UFF

Profa. Denise Burgarelli Duczmal / ICEx - UFMG

Prof. Luiz Henrique de Figueiredo / IMPA

Niterói, Setembro de 2006.

À Nossa Senhora.

Agradecimentos

A Deus, pelo amor, pela saúde e por todas as coisas.

À minha família, em especial ao meu irmão, pela força e verdadeira amizade.

Aos meus amigos e namorada, Adriana, minhas fontes de alegria e esperança, que me fizeram entender que eu era capaz. Em especial a Betina Vath e Edilio Quintino.

Aos meus alunos que atentos ao que eu falo me dão força para não desistir de estudar.

A todos os professores que se dedicam verdadeiramente e principalmente aos professores Ana Kaleff, Mauricio Kischinhevsky, Otton Teixeira, Regina Leal e Cristina Boeres.

Resumo da Tese apresentada à UFF como requisito parcial para a obtenção do grau de Mestre em Computação (M.Sc.)

Um visualizador para navegação através de cenários tridimensionais baseado em malhas adaptativas

Diogo Tavares Robaina

Setembro/2006

Orientadores: Mauricio Kischinhevsky
Otton Teixeira da Silveira Filho
Programa de Pós-Graduação em Computação

Este trabalho apresenta um visualizador para navegação através de cenários tridimensionais nos quais a malha criada para a renderização das imagens é adaptativa e sua ordenação é baseada em uma modificação da curva de Hilbert de preenchimento do espaço. Trata-se de um sistema de Visualização Volumétrica, subárea da Visualização Científica, cujo foco relaciona-se com a extração de informação referente a problemas definidos no espaço tridimensional. A ferramenta desenvolvida é de uso simples e aplica-se a diversas áreas do conhecimento como engenharia, medicina, bioengenharia, geologia e meteorologia. A vantagem dessa estratégia adaptativa é, principalmente, a redução do número de pontos necessários para obter uma interpretação precisa do fenômeno em estudo. As imagens geradas são pouco refinadas onde os dados são facilmente interpretados e mais refinadas como, por exemplo, em uma região onde houver uma brusca variação de temperatura. O visualizador foi desenvolvido em C++ e utiliza as interfaces OpenGL e FLTK, permitindo sua utilização nos sistemas operacionais GNU/Linux, Windows e outros.

Abstract of Thesis presented to UFF as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

An adaptively refined mesh-based visualization tool for navigation through three-dimensional scenes

Diogo Tavares Robaina

September/2006

Advisors: Mauricio Kischinhevsky

Otton Teixeira da Silveira Filho

Department: Computer Science

This work introduces a visualization tool for navigation through tridimensional scenarios. The mesh employed for renderization of the images is adaptive and its ordering is based in a modification of the Hibert space-filling curve. It is a Volume Visualization system which aims at the information extraction related to problems defined in tridimensional domains. The tool discussed herein is simple and shall be applicable to several fields such as engineering, medicine, bio engineering, geology and meteorology. The advantage of this adaptive strategy is, mainly, the reduction of the number of points necessary to get a precise interpretation of the phenomenon in study. The generated images little refined where the data easily interpreted and more refined as, for example, in a region where it will have an brusque variation of temperature. The visualization tool was developed in C++ and uses the OpenGL e FLTK interfaces, allowing its use in the GNU/Linux, Windows and other operational systems.

Palavras-chave

1. Visualização Científica
2. Curvas de Preenchimento do Espaço
3. Curva de Hilbert modificada
4. Métodos Numéricos

Lista de Siglas e Abreviaturas

ALG	*Autonomous Leaves Graph* (Grafo de Folhas Autônomas)
DVR	*Direct Volume Rendering* (Renderização Volumétrica Direta)
FLTK	*Fast Light ToolKit*
GLUT	*OpenGL Utility Toolkit*
HSV	Modelo de cor, matiz, saturação e brilho (Hue, Saturation, Value)
LGPL	*GNU Library General Public License*
OpenGL	*Open Graphics Library*
RGB	Modelo de cor, componentes vermelha, verde e azul (Red, Green, Blue)
RSFC	*Recursive Space-filling Curve* (Curva de Preenchimento do Espaço Recursiva)
SciVis	*Scientific Visualization* (Visualização Científica)
SFC	*Recursive Space-filling Curve* (Curva de Preenchimento do Espaço)
VisHil3d	Visualizador para Navegação em Cenários Tridimensionais baseado na Curva de Hilbert Modificada Tridimensional

Sumário

Lista de Figuras

Capítulo 1

Introdução

A visualização de dados provenientes de simulações numéricas nem sempre é uma tarefa simples. Geralmente os algoritmos para visualização necessitam percorrer um grande volume de dados, que pode envolver tipicamente da ordem de milhões de elementos volumétricos a cada instante simulado. A representação visual destes dados é realizada com a utilização de diversas técnicas da Visualização Científica dentro da área de Computação Gráfica. Muitas vezes, devido ao desconhecimento dessas técnicas, simulações numéricas complexas e/ou em três dimensões dispõem de poucas possibilidades de análise intuitiva e visualmente interativa.

A eficiência no processamento, armazenamento e facilidade de acesso aos dados são fundamentais em aplicações que envolvem a Visualização Científica. Para tanto, a estratégia que deve ser utilizada para representar visualmente os dados é objeto de análise pormenorizada. A mais simples é desenhar todos os dados em estudo, sem a interpretação das particularidades do problema tratado, o que se mostra ineficiente quando o interesse do usuário está direcionado apenas a uma parte deles, ocasionando, assim, um custo computacional desnecessariamente alto. A segunda seria gerar imagens pouco refinadas onde os dados são facilmente interpretados e mais refinadas como, por exemplo, em uma região onde houver uma brusca variação de temperatura. Isso reduz drasticamente o número de pontos necessários para obter uma interpretação precisa, com reflexo no custo computacional.

A segunda estratégia mostra-se robusta, confiável e eficiente na resolução de problemas que envolvem Equações Diferenciais Parciais por diferenças finitas, volumes finitos e

elementos finitos. Burgarelli e colaboradores [1, 2, 3] propõem uma estratégia para resolução de EDPs bidimensionais baseada na construção de uma malha de pontos refinada localmente com baixo custo computacional. Propuseram um Grafo de Folhas Autônomas (*Autonomous Leaves Graph* - ALG), estrutura que dispensa o uso de uma árvore de refinamentos. A partir de uma malha dada inicialmente, representada pela raiz de uma árvore, criam-se nodos filhos dessa raiz, que são as representações das sub-malhas, com um grau de refinamento maior que a malha inicial. Esta estrutura se reproduz recursivamente, formando uma hierarquia de sub-malhas, cada vez mais refinadas. Nessa estrutura nem todos os nodos precisam gerar nodos filhos, permitindo-se, assim, que a malha fique com um maior número de pontos apenas em algumas regiões e malhas com diferentes graus de refinamento convivam numa mesma região. Para a ordenação da estrutura proposta, utiliza-se uma curva bidimensional, a curva de Hilbert modificada [1], baseada na curva de preenchimento do espaço (*space-filling curve* - SFC) de Hilbert.

A utilização das SFCs, em particular da curva de Hilbert, faz-se presente em diversos trabalhos publicados nos últimos anos, como por exemplo na compressão de imagens [4] e no armazenamento e recuperação de dados [5]. Além destes, [6, 7, 8, 9] apresentam outras aplicações da curva de Hilbert, bidimensional, tridimensional e dimensões superiores.

Neste trabalho apresenta-se uma técnica apropriada para visualização de dados provenientes de simulações numéricas através do desenvolvimento de um visualizador para geração de cenários tridimensionais baseado em um procedimento adaptativo que faz refinamento e desrefinamento das células que serão desenhadas, utilizando-se para ordenação das células uma curva de Hilbert modificada tridimensional, dando continuidade aos trabalhos iniciados por Burgarelli e colaboradores.

Para desenvolver a ferramenta computacional aqui apresentada, os conceitos referentes às SFCs, em particular à curva de Hilbert tridimensional e o algoritmo para sua construção

são discutidos. Um estudo sobre os principais componentes e características da Visualização Científica, bem como das interfaces de programação OpenGL e FLTK, foi desenvolvido.

Objetiva-se viabilizar um conjunto de métodos que auxiliem em análises numéricas e possibilitem o desenvolvimento de futuras aplicações, sem visar inicialmente implementar todas as características encontráveis em aplicativos de visualização.

O presente texto está organizado da forma a seguir.

No capítulo 2 encontra-se um breve histórico e conceituação da visualização, suas técnicas e aplicações. São apresentados os princípios que envolvem a Visualização Científica, exibindo alguns conceitos da Computação Gráfica e as características de um sistema de visualização.

No capítulo 3 apresentam-se as SFCs, em particular as curvas de Hilbert bidimensional, tridimensional e a curva de Hilbert modificada tridimensional utilizada para a representação dos dados estudados.

No capítulo 4 discutem-se as interfaces OpenGL e FLTK, os tipos de dados e a forma como o visualizador desenvolvido atua na aquisição, organização e renderização destes.

No capítulo 5 exibe-se a visualização de resultados relativos a simulações numéricas utilizando recursos do visualizador.

No capítulo 6 encerra-se o trabalho com conclusões e considerações finais, além de apresentar sugestões para desenvolvimentos futuros.

Capítulo 2

Visualização

O termo visualização é, em geral, empregado para referenciar o conjunto de técnicas para criação de imagens. Pode-se entendê-la como sendo o conjunto de métodos que permitem a interpretação de informações relevantes em conjuntos de dados provenientes de funções de difícil reconstrução, com o auxílio de técnicas de Computação Gráfica [10]. Constitui-se, assim, como uma ferramenta para a interpretação e análise de dados computacionais.

Visualização Científica (*Scientific Visualization* - SciVis) é o campo que se preocupa com a extração de informações de caráter científico a partir de conjuntos de dados que representam fenômenos complexos. McCormick [11] define a SciVis como o campo da Computação Gráfica que inclui interface gráfica, representação de dados, representação visual e outras. Para Brodlie [12], a SciVis está relacionada à exploração e ao entendimento mais profundo dos dados sob investigação, apoiando-se na habilidade de visualização do usuário. Muitas vezes, as ferramentas e técnicas de visualização são utilizadas para analisar e mostrar grandes volumes de dados; desta forma, para esses autores a Visualização Científica permite ao usuário extrair características e resultados de forma rápida e simples.

Diversas são as técnicas utilizadas pela SciVis e geralmente são específicas para tratar dados escalares, como temperatura; dados vetoriais, como deslocamento; ou ainda, tensoriais. Tais técnicas são baseadas em algoritmos da Computação Gráfica, da Visão Computacional e do Processamento de Dados, surgidas antes da larga utilização dos computadores [13]. A SciVis é hoje amplamente utilizada em diversas áreas, entre elas medicina, bioengenharia,

geologia, meteorologia e engenharia, incluindo representação de dados obtidos por métodos numéricos. Por exemplo, no campo da astronomia ela pode ser utilizada para a representação dos corpos celestes [14].

A área médica foi uma das primeiras a fazer uso da SciVis; dentre as aplicações destacam-se softwares para a visualização do corpo humano [15], e para o treinamento de procedimentos cirúrgicos com grande realismo com base em realidade virtual [16]. A visualização de dados obtidos de imagem por ressonância magnética e de escoamento de fluidos em três dimensões tem também bastante destaque, como ocorre na visualização da distribuição sangüínea no cérebro [17].

2.1 Visualização Volumétrica

Dados associados a regiões de volumes (informações tridimensionais) são chamados dados volumétricos e podem ser obtidos por simulação, na qual os dados são gerados a partir da solução de um modelo matemático ou por aquisição direta, em que são resultantes da medição de grandezas existentes na natureza. O conjunto de técnicas voltadas para a visualização de dados volumétricos é conhecido como Visualização Volumétrica e constitui uma subárea da Visualização Científica.

Contemporaneamente ainda são desenvolvidas pesquisas e trabalhos no campo da Computação Gráfica relacionados à Visualização Volumétrica. O desenvolvimento e o aperfeiçoamento de algoritmos buscam maior qualidade das imagens e a diminuição do tempo de geração das mesmas, permitindo o desenvolvimento de sistemas interativos. Há duas abordagens básicas para realizar a visualização volumétrica [10]: renderização volumétrica direta (Figura 2.1 a) e extração de superfícies (Figura 2.1 b). Na primeira os dados são visualizados

diretamente, sem a extração explícita de superfícies. Na segunda, são construídas representações poligonais de superfícies relacionadas às características desejadas da função de interesse (por exemplo, isosuperfícies) que são, em geral, visualizadas utilizando técnicas de renderização de polígonos.

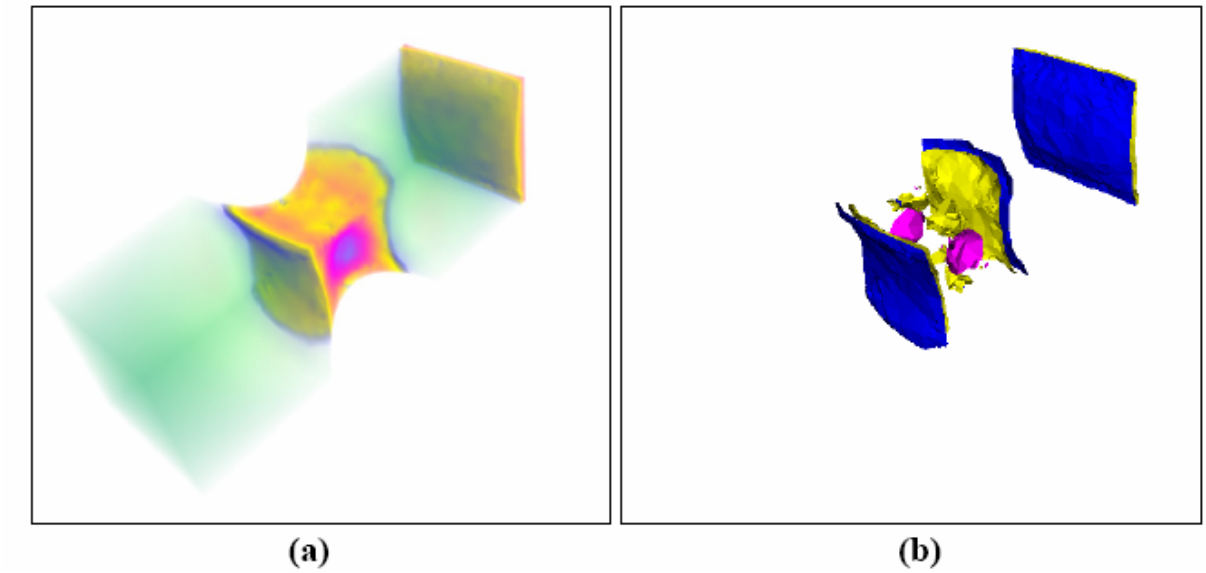


Figura 2.1: Técnicas de Visualização Volumétrica: (a) Renderização Volumétrica Direta, (b) Extração de Isosuperfícies. [18]

As técnicas de Renderização Volumétrica Direta (*Direct Volume Rendering* - DVR) permitem ao usuário enxergar "dentro" (ou "através") do volume de dados, possibilitando a visualização de mais informações do que as técnicas de renderização de superfícies. A imagem bidimensional resultante da renderização de volume pode ter a aparência de um gel semi-transparente. Os dados são mapeados por cor e transparência, contribuindo para a imagem final, o que pode causar confusão quanto à identificação de determinadas características, resultado indesejado em análises de resultados de simulações numéricas. A Figura 2.2 ilustra

este problema com a renderização volumétrica de um campo escalar. O mapeamento de cores no interior do modelo se torna confuso devido ao uso de transparências.

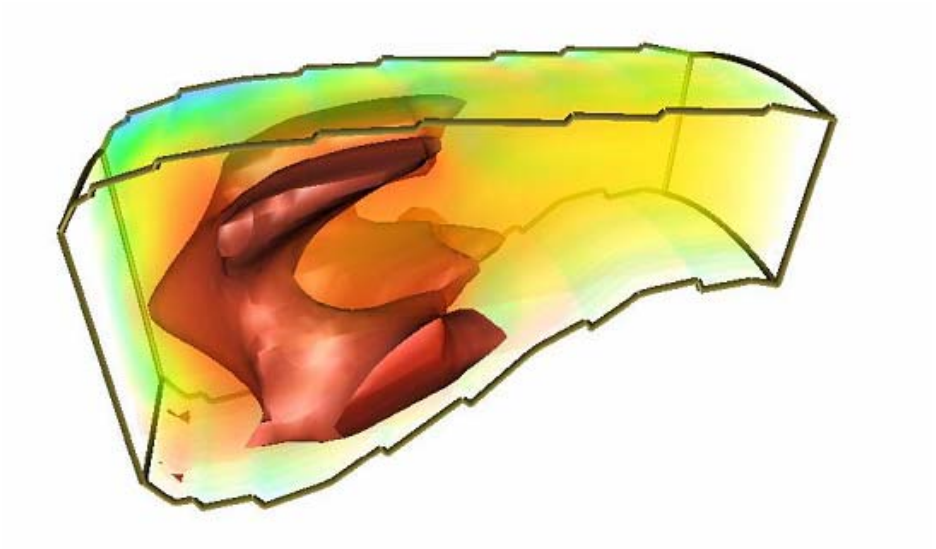


Figura 2.2: Renderização Volumétrica Direta de um campo escalar [19]

Uma vantagem da extração de superfícies é a possibilidade de identificação de estruturas nas medições ou simulações. Elas permitem análise em tempo real, pois, apesar de o processo de extração de superfícies ser lento, a renderização das superfícies resultantes pode ser muito eficiente. Entre as desvantagens observa-se a perda da informação relativa ao conteúdo do conjunto de dados fora dos valores selecionados para o traçado e nem sempre os dados podem ser descritos em termos de superfícies, como é o caso de objetos sem forma definida, como nuvens.

2.2 Princípios e Componentes da SciVis

2.2.1 Modelagem Geométrica

A geometria das formas dos objetos se constitui na base para visualização computacional. Para representação de objetos geométricos existem diversos modelos de dados. Os sistemas de visualização precisam manipular e modificar essas informações e a escolha de um modelo de dados é dependente dos requisitos da aplicação.

Existem diversos modelos geométricos, desde os que representam um mínimo de informações até os que detalham os modelos e permitem relação entre seus componentes. Esses modelos podem ser classificados como: gráficos, de decomposição, construtivos e de contorno [20].

Os modelos gráficos são constituídos de entidades simples como pontos, linhas e polígonos. Eles são úteis para visualizações simples e de baixa qualidade. Os de decomposição consistem de uma coleção de elementos básicos e operações de junção. Embora também simples, permitem algoritmos bastante eficientes. Os modelos construtivos também são constituídos por pontos, linhas e polígonos, mas têm operadores de combinação mais poderosos, comparado com os do modelo anterior. Com esse modelo, podem-se também desenvolver algoritmos bastante eficientes. Já os modelos de contorno (ou superfície) representam um objeto geométrico por sua superfície de contorno. Com essa representação, uma grande gama de operações complexas pode ser introduzida. Todos os modelos, exceto o gráfico, podem ser convertidos nesse.

Dentre os modelos de contorno, existem estruturas baseadas em polígonos, outras em vértices e outras, ainda, em arestas. Exemplo eficiente de estrutura de dados baseada em ares-

tas é a do tipo semi-aresta [21]. Esta estrutura é adequada para representar os modelos geométricos envolvidos nos métodos numéricos, possibilita um acesso eficiente a cada componente e é também possível operacionalizar as técnicas de visualização mais comumente utilizadas.

2.2.2 Iluminação e Cor

A maior parte do processo físico de iluminação pode ser facilmente simulada no computador, apesar de sua utilização na obtenção de imagens de alta qualidade implicar em elevados custos computacionais. Assim, é mais comum, a adoção de um modelo simplificado apenas com os processos rudimentares da reflexão da luz sobre uma superfície. A iluminação, de forma direta, não é muito importante em análises numéricas, mas, de forma indireta, ela pode dar a impressão de tridimensionalidade pelo efeito de sombreamento ou tonalização[1], facilitando a análise destes modelos. Para superfícies coloridas, a intensidade de cada componente da luz deve ser calculada individualmente.

A cor é de fundamental importância na geração de imagens relacionadas a resultados de métodos numéricos. A maioria dos sistemas gráficos trabalha com o modelo de cores RGB (*Red* (vermelho), *Green* (verde), *Blue* (azul)), modelo utilizado neste trabalho, e que é o mais próximo do hardware gráfico, que trabalha com fósforo destas três cores. No entanto, o resultado da combinação das componentes é, em geral, de difícil previsão. Porém, existem modelos de cores mais intuitivos, como o HSV [23] (*Hue*, *Saturation*, *Value*, ou matiz, saturação e brilho), que também estão disponíveis para a geração de tabelas de cores em muitos sistemas. No modelo HSV, a definição de cada cor básica se dá sobre um círculo de cores conforme representado pelo cone da Figura 2.3. O parâmetro de saturação define a proporção de quanto

[1] Tonalização (*shading)* é o processo de calcular a iluminação para cada ponto de uma superfície [22].

daquele matiz é utilizado, com valores mais saturados nas bordas do círculo e menos saturados próximos do centro (valores entre 0 e 1), e o parâmetro brilho define a luminosidade da cor (valores entre 0 e 1).

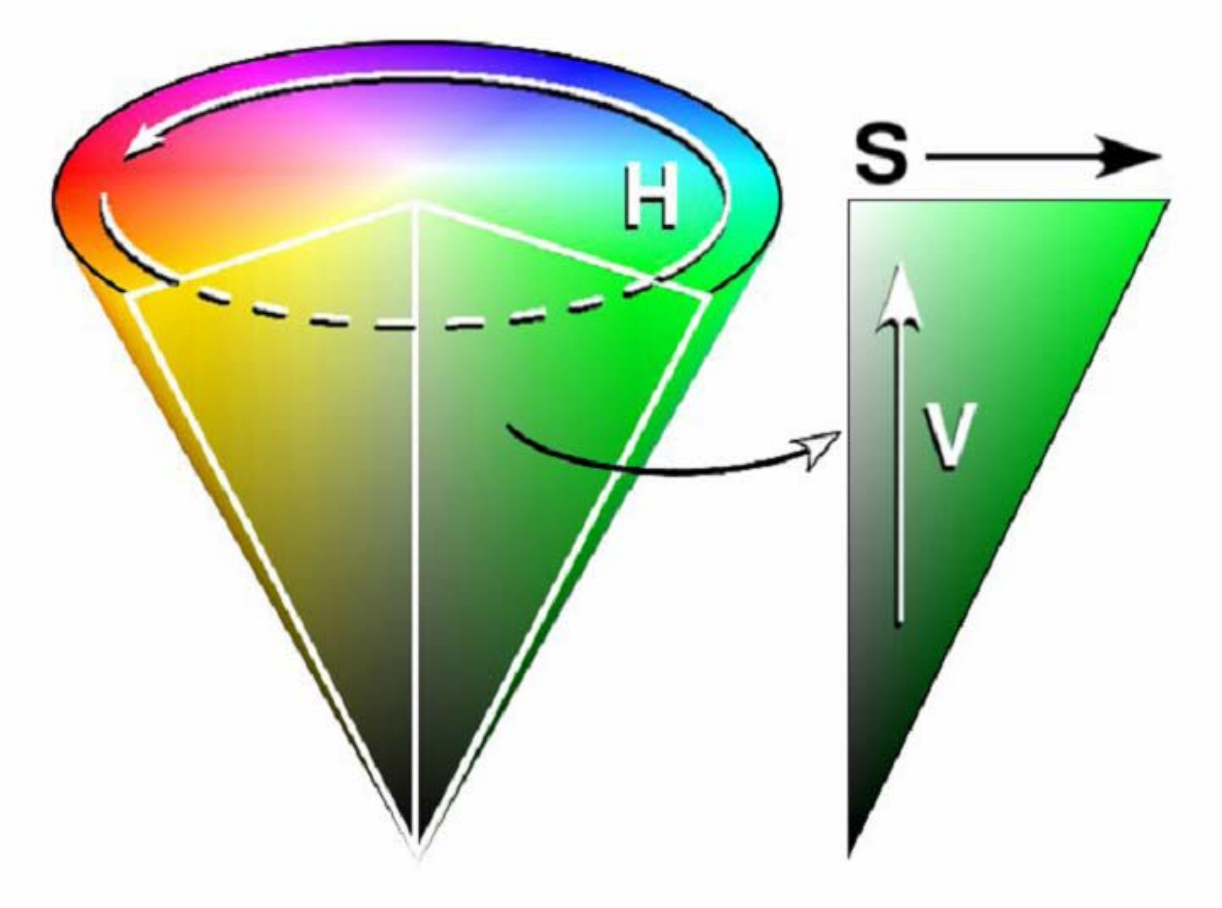


Figura 2.3: Cone do Modelo de Cor HSV [23]

A definição de cores e de tabelas de cores apropriadas para uma cena é um assunto de extrema importância em visualização. Tais tabelas normalmente são responsáveis pelo mapeamento de informação numérica e, portanto, assumem significado específico, como por exemplo, valores das temperaturas em um meio material.

2.2.3 Textura

Objetos reais não são suficientemente caracterizados apenas por sua geometria tridimensional, necessitando de informações adicionais a respeito de suas superfícies. Nos casos mais simples, as superfícies dos objetos são cobertas com uma cor constante. Como regra, as superfícies de objetos não são desestruturadas, mas possuem variação de cor dependente de posição, transparência, rugosidade e também geometria, que dão a característica de aparência de materiais. Essas propriedades são associadas ao termo textura.

Além de representar objetos reais, as texturas são usadas para dar suporte a aplicações científicas para percepção do espaço e forma de objetos tridimensionais. As texturas são tratadas como parâmetros de aparência como, por exemplo, cor, rugosidade, reflexão e brilho.

As texturas não são comumente usadas em análise por métodos numéricos, mas algumas técnicas as utilizam para representar dados vetoriais de maneira qualitativa [20].

2.2.4 Renderização

A renderização (rendering), processo de geração de uma imagem em uma tela a partir de uma descrição computacional [22], é fundamental na SciVis e, conseqüentemente, da Computação Gráfica. O processo de geração de imagens varia bastante, e muitas técnicas são utilizadas para sua realização, envolvendo desde a geração de desenhos em duas dimensões até técnicas sofisticadas de geração de imagens tridimensionais.

Uma técnica de renderização é composta de um procedimento de eliminação de superfícies escondidas e um modelo de shading (cor, luz e sombra). Essa eliminação faz com que

apenas as partes do modelo que são visíveis pelo observador sejam mostradas. O modelo de shading determina a aparência final de cada superfície na tela, de acordo com um modelo de iluminação, que leva em conta as condições de luz, propriedades ópticas das superfícies (cor, textura, reflexão, etc.) e a posição e orientação das superfícies em relação às fontes de luz, a outras superfícies e ao observador.

A maioria das técnicas tridimensionais, nas quais são baseados diversos algoritmos da SciVis, gera a imagem de um objeto por meio da simulação de sua iluminação ambiente, ou seja, fontes de luz. As técnicas de renderização baseadas nesse conceito variam consideravelmente: as mais simples desconsideram as reflexões entre objetos e a atenuação da luz no ambiente, enquanto que as técnicas mais complexas [22], e que produzem imagens bastante realísticas, adotam um modelo para a simulação da interação dos raios de luz com todos os objetos da cena, a um alto custo computacional.

2.2.5 Câmera Sintética

Uma câmera sintética precisa ser definida em cada cena tridimensional. Esta câmera sintética pode ser caracterizada de diversas maneiras como, por exemplo, um ponto de visão e centro de interesse que define o centro da imagem da câmera. Outra idéia é associar o ponto de visão com uma direção. A Figura 2.4 mostra uma representação da projeção de uma imagem em uma tela bidimensional, de acordo com a câmera.

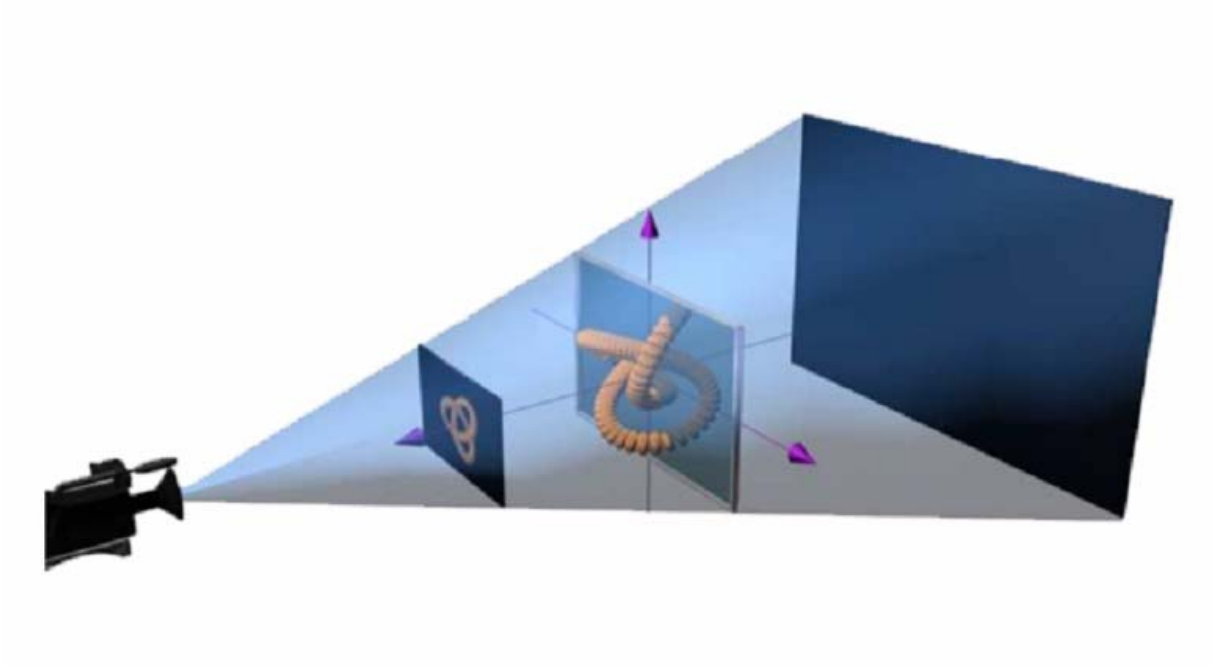


Figura 2.4: Projeção de Cena Tridimensional em imagem Bidimensional [24]

O movimento de câmera é o mais utilizado para representar o observador em sistemas gráficos e de visualização [22]. O posicionamento e a animação de objetos se dão tanto pela alteração da posição do objeto em relação a um sistema de coordenadas global da cena, como pela alteração pura e simples da posição da câmera.

As projeções mais comuns para a criação de uma imagem são as perspectivas e a paralela. Em análises de métodos numéricos a projeção paralela ortogonal é preferida à perspectiva pois, enquanto a primeira distorce as dimensões, dando uma impressão tridimensional, a segunda mantém proporções e dimensões. A manutenção dessas características é importante para uma compreensão global do comportamento do modelo, mesmo diminuindo a sensação de tridimensionalidade.

2.2.6 Interatividade e Animação

Interação pode ser entendida como o conjunto de ações do usuário e a reação do computador durante o processo de visualização. Sua importância relaciona-se à exploração e compreensão do problema em estudo, seja através do mouse, teclado ou outro dispositivo de entrada. Ela permite a navegação pelo modelo numérico através de operações de rotação, translação, zoom, e a mudança da técnica utilizada, de pontos de visão e de parâmetros dos dados.

Pode-se entender animar como trazer à vida [20]. Na prática, significa definir mudanças em parâmetros no tempo ou aplicar operações em objetos no tempo. Numa animação comum, os intervalos de tempo entre um passo e outro devem ser tais que possam assegurar a sensação de continuidade, como num filme, produzindo a ilusão de movimento contínuo. Para criar esta ilusão é necessário o uso de interpolações entre uma etapa e outra quando os dados estão esparsos no tempo. Esse efeito de continuidade não é fundamental em análises gráficas utilizando métodos numéricos.

Capítulo 3

Estrutura de Dados

Um conjunto de dados pode ser considerado como um conjunto de valores em certo intervalo de variação definido sobre algum domínio de variáveis independentes [25]; possui, em geral, certa organização e consiste de uma malha ou grade de células. De acordo com a organização definida pela estrutura, pode haver vários tipos de malhas, cf. Tabela 1.

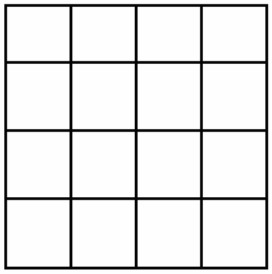
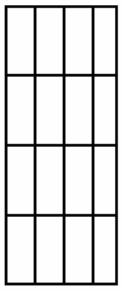
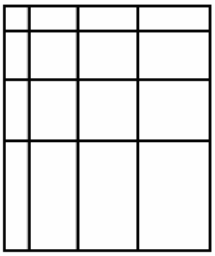
Malha Cartesiana	**Malha Regular**	**Malha Retilínea**
		
Todos os elementos são quadrados ou cubos idênticos, alinhados aos eixos principais.	Todos os seus elementos são idênticos, alinhados aos eixos e retângulos (ou paralelepípedos) regulares.	Seus elementos são quadriláteros ou hexaedros alinhados aos eixos e não são, necessariamente, idênticos.

Tabela 1: Tipos de Malhas [22]

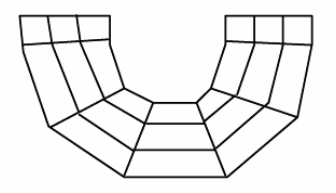
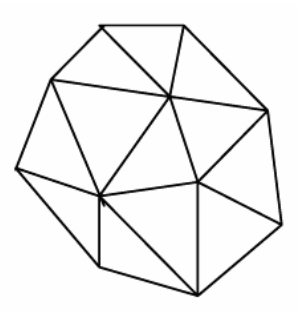
Malha Estruturada	**Malha Irregular**
	
Os elementos são quadriláteros ou hexaedros não alinhados aos eixos principais, como, por exemplo, os que aparecem em grades esféricas ou curvilíneas.	Contém polígonos ou poliedros sem qualquer padrão explícito de conectividade. No caso tridimensional, as células podem ser tetraedros, hexaedros ou outro tipo.

Tabela 1: Tipos de Malhas [22]

Existem também as malhas sem conectividade, onde os pontos são esparsos e podem ter posição regular ou irregular, dependendo do espaço entre os pontos, e as malhas híbridas que são uma combinação de quaisquer tipos apresentados na Tabela 1.

A estrutura de dados desenvolvida neste trabalho utiliza uma malha tridimensional retilínea formada por hexaedros regulares e a estratégia proposta por [1] para a ordenação desta através da curva de Hilbert modificada. Este assunto será abordado detalhadamente na seção 3.5.

3.1 Estrutura da Malha Adaptativa

Como proposto por Burgarelli [1], considere inicialmente um domínio tridimensional formado pelo cubo unitário. Dividindo este cubo em oito cubos iguais obtêm-se oito células cúbicas com aresta $\frac{l}{2}$, conforme a Figura 3.1.

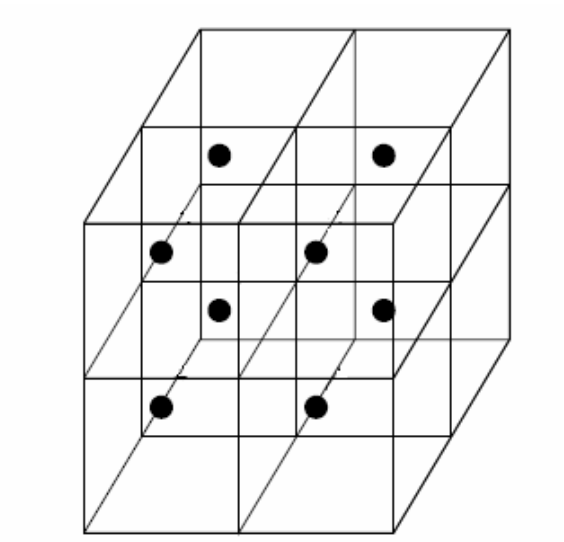


Figura 3.1: Cubo Unitário [1]

Em cada um desses cubos escolhe-se um ponto, denominado nodo preto; este representará o centro da célula e será denotado pictoricamente. Suas coordenadas espaciais e as informações referentes aos dados volumétricos serão associadas a esses nodos.

Adota-se a convenção de que as células vizinhas se situam a norte, sul, leste e oeste, em analogia com as direções geográficas, acrescidas de frente e trás, completando as três direções espaciais. Cada nodo possui, então, seis ponteiros que apontam para os vizinhos do norte, sul, leste, oeste, frente e trás. Para indicar que um nodo não tem vizinho em uma deter-

minada direção, ou seja, que a célula correspondente pertence ao bordo da malha, é conectado a um outro nodo especial, chamado nodo terra.

Além dos seis ponteiros que apontam para cada direção, os nodos pretos ainda possuem mais dois ponteiros que apontam para a próxima célula e para a célula anterior. Ou seja, os nodos pretos são organizados dentro de uma lista duplamente encadeada e ordenada. Assim, cada vez que é feito um refinamento ou desrefinamento ocorrem apenas perturbações locais na estrutura.

Conforme Burgarelli [1], define-se profundidade de uma célula como o número de passos realizados para sua obtenção e convenciona-se que as células de uma malha composta de oito células iguais estão no nível 1 de profundidade.

A conexão entre nodos de células de profundidades diferentes é feita com auxílio dos nodos de transição ou, simplesmente, nodos brancos. Estes possuem 5 ponteiros e tem a função de conectar um nodo preto ou um branco de profundidade *n* em quatro nodos pretos ou brancos de profundidade *n + 1* (Figura 3.2).

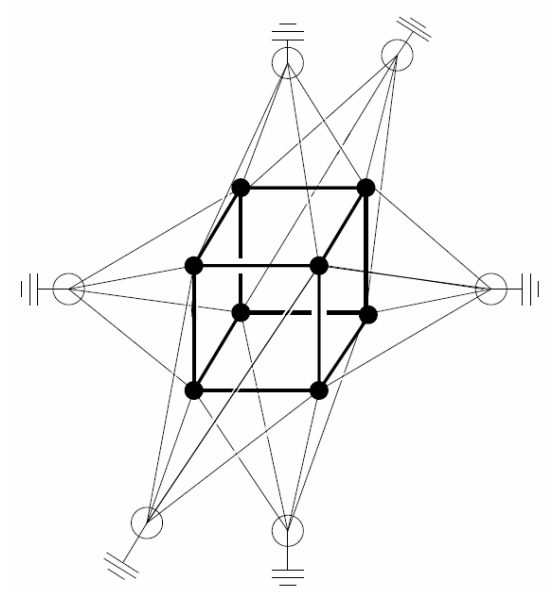


Figura 3.2: Estrutura de conexão de nodos [1]

3.2 Refinamento da Malha

Considere o cubo unitário e as oito células de aresta $\frac{1}{2}$ como na Figura 3.3.

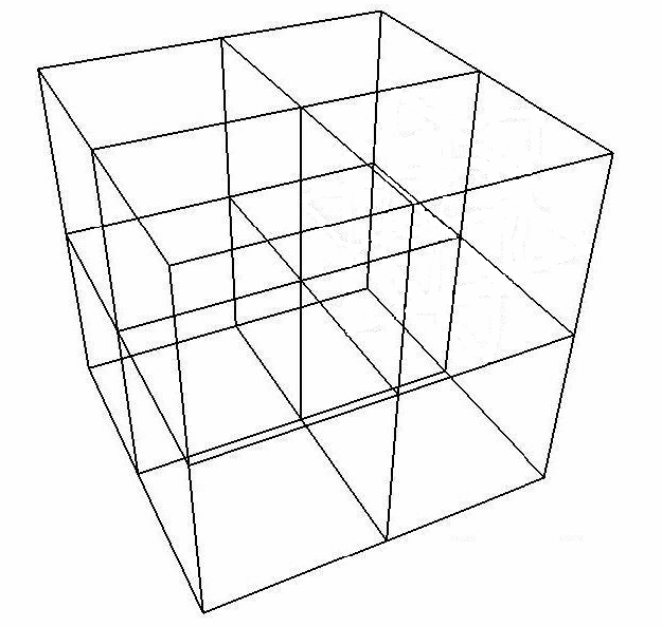


Figura 3.3: Cubo Unitário e Células da Malha [26]

Refinar a célula de centro (a, b, c) e aresta l significa substituí-la por oito células de aresta $\frac{l}{2}$ (Figura 3.4) de centros:

$$\left(a-\frac{l}{4},b-\frac{l}{4},c-\frac{l}{4}\right),\left(a-\frac{l}{4},b-\frac{l}{4},c+\frac{l}{4}\right),\left(a-\frac{l}{4},b+\frac{l}{4},c+\frac{l}{4}\right),\left(a-\frac{l}{4},b+\frac{l}{4},c-\frac{l}{4}\right),$$

$$\left(a+\frac{l}{4},b-\frac{l}{4},c-\frac{l}{4}\right),\left(a+\frac{l}{4},b+\frac{l}{4},c-\frac{l}{4}\right),\left(a+\frac{l}{4},b-\frac{l}{4},c+\frac{l}{4}\right),\left(a+\frac{l}{4},b+\frac{l}{4},c+\frac{l}{4}\right)$$

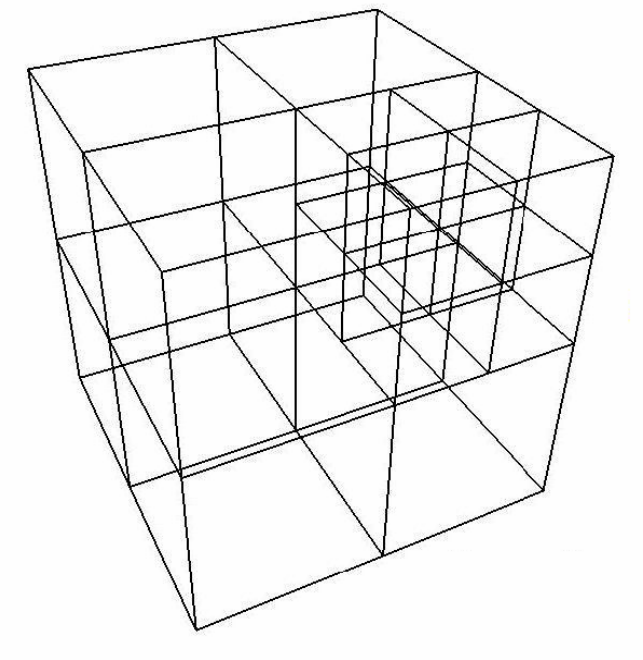


Figura 3.4: Cubo Unitário com Célula Refinada [26]

O processo de subdivisão do domínio ou, simplesmente, refinamento da malha, não é necessariamente uniforme. Uma célula de nível *n* de profundidade, ao ser refinada, é substituída por 8 células de nível (*n + 1*) de profundidade. Se uma célula do cubo unitário possui aresta de comprimento igual a $\frac{1}{2^n}$, sua profundidade, no refinamento, será igual a *n*.

Ao refinar uma célula criam-se, inicialmente, oito novos nodos pretos e seis nodos brancos, já dispostos como na malha inicial. Os nodos pretos de mesma profundidade são ligados entre si e os de profundidades diferentes são ligados através de um nodo branco, conforme a Figura 3.5. Se nodos brancos de mesma profundidade, após o refinamento, aparecem ligados faz-se necessária uma simplificação na estrutura [1]. Essa simplificação possibilita que o funcionamento do algoritmo de busca dos vizinhos de um nodo preto funcione corretamente.

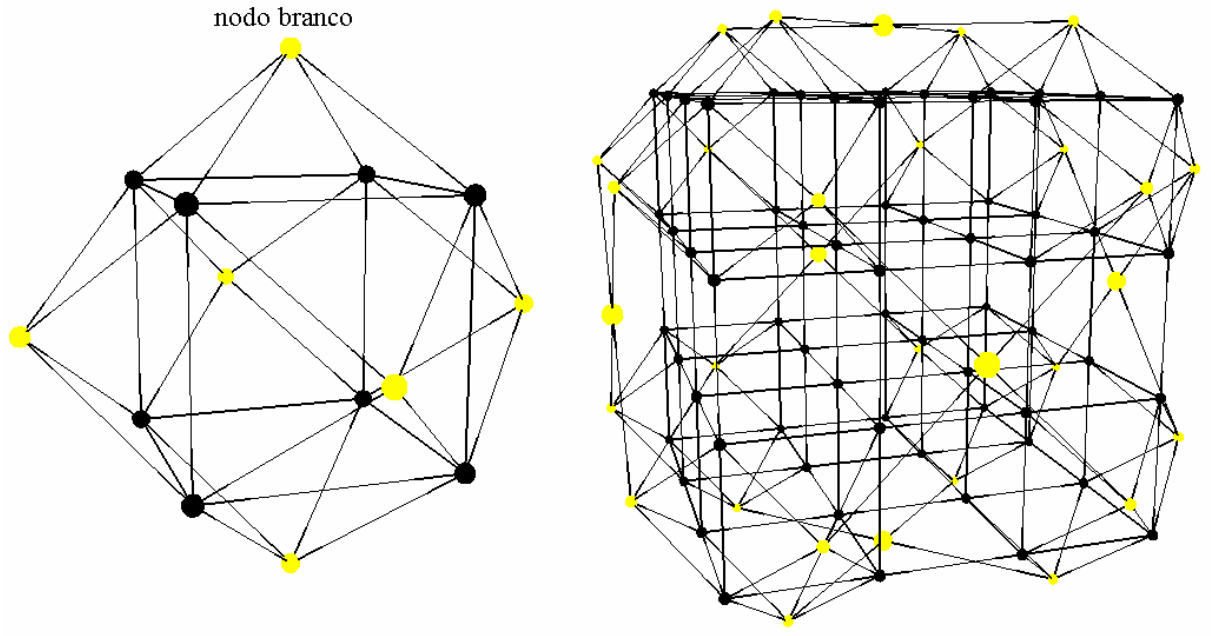


Figura 3.5: Refinamento com Estrutura

No final do processo os nodos pretos criados são ordenados pela curva de Hilbert modificada tridimensional.

3.3 Desrefinamento da Malha

O processo de desrefinamento da malha só pode ocorrer se forem desrefinadas células que possuem exatamente sete irmãs em um mesmo nível de profundidade. Ou seja, oito células de profundidade n são substituídas por apenas uma célula de profundidade $(n - 1)$. Essas células devem ser originadas do mesmo pai para garantir a configuração anterior da malha após o desrefinamento Figura 3.6.

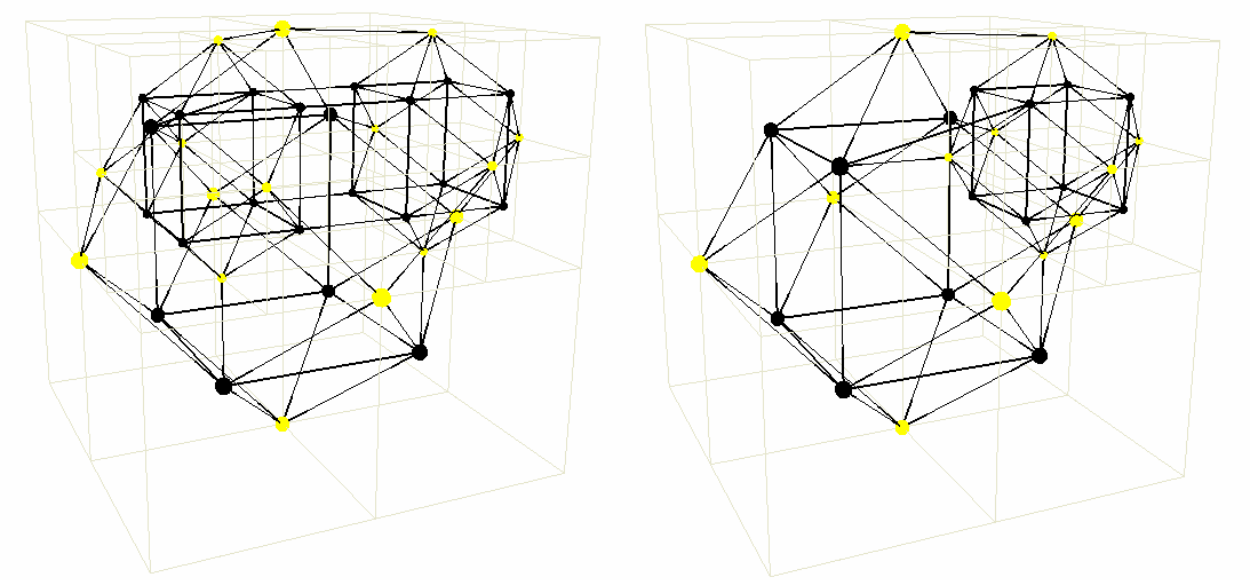


Figura 3.6: Desrefinamento com Estrutura

As etapas necessárias para o desrefinamento são:

1) criação de um novo nodo preto;

2) o nodo preto criado armazena suas coordenadas espaciais e sua profundidade;

3) ligação do nodo preto com seus vizinhos;

4) ordenação da malha pela curva de Hilbert;

5) liberação da área de memória dos nodos brancos e pretos que passam a não existir.

Como no processo de refinamento, o processo de desrefinamento pode acarretar a destruição de nodos brancos uma vez que nodos brancos de mesma profundidade não devem coexistir e nodos pretos filhos de um mesmo pai devem ser conectados diretamente [1].

3.4 Curvas de Preenchimento do Espaço

O conceito das curvas de preenchimento do espaço (space-filling curves - SFCs) surgiu no século XIX e originalmente é atribuído a Peano [27]. Ele expressou o conceito em termos matemáticos e o representou através de pontos no espaço. A primeira representação gráfica das SFCs é atribuída a Hilbert [28].

Em seu artigo publicado em 1891, Hilbert construiu passo a passo uma curva de preenchimento do espaço em duas dimensões com uma seqüência infinita de curvas finitas. Hilbert mostrou que, repetindo o processo indefinidamente definiria uma curva contínua e não diferenciável [29] que passa por todos os pontos de um espaço, denominada *curva de Hilbert*. As idéias de Peano e Hilbert podem ser formalizadas da seguinte forma:

Definição 3.1: Uma *curva plana* é uma aplicação contínua $C : I \to R^2$ do intervalo unitário $I = [0,1]$ da reta real, no plano euclidiano $R^2 = \{(x, y); x, y \in R\}$.

Definição 3.2: A imagem $c(I)$ é chamada de *traço da curva* C.

Definição 3.3: Uma SFC bidimensional é uma curva contínua tal que seu traço preenche todo o quadrado unitário $I^2 = [0,1] \times [0,1]$ do R^2. Para cada ponto $P \in I^2$, existe um número real $t \in I$ tal que $c(t) = P$. Isso significa que uma SFC oferece uma maneira ordenada de visitar todos os pontos do quadrado apenas variando t entre 0 e 1.

Para construir uma SFC bidimensional c considere a seqüência $c_n : I \to I^2$ tal que:

$$c = \lim_{n\to\infty} c_n$$

As curvas c_n formam aproximações de c e, à medida que n cresce, mais pontos do quadrado unitário são visitados. A Figura 3.7 apresenta exemplos de SFCs bidimensionais e a Figura 3.8 as tridimensionais definidas similarmente às bidimensionais.

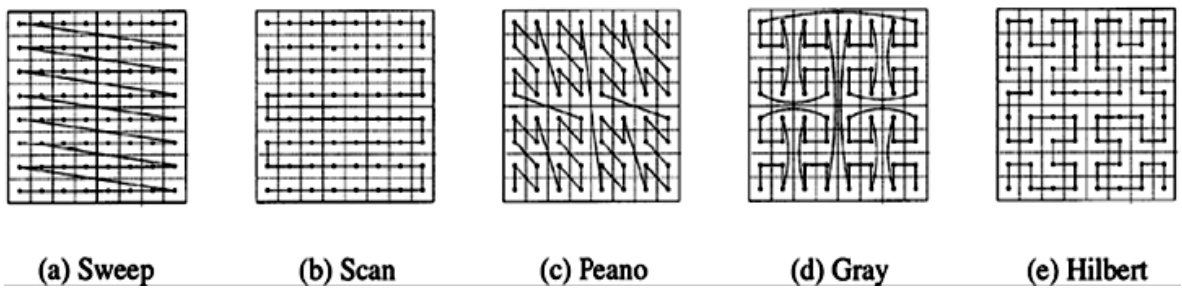


Figura 3.7: SFCs Bidimensionais [30]

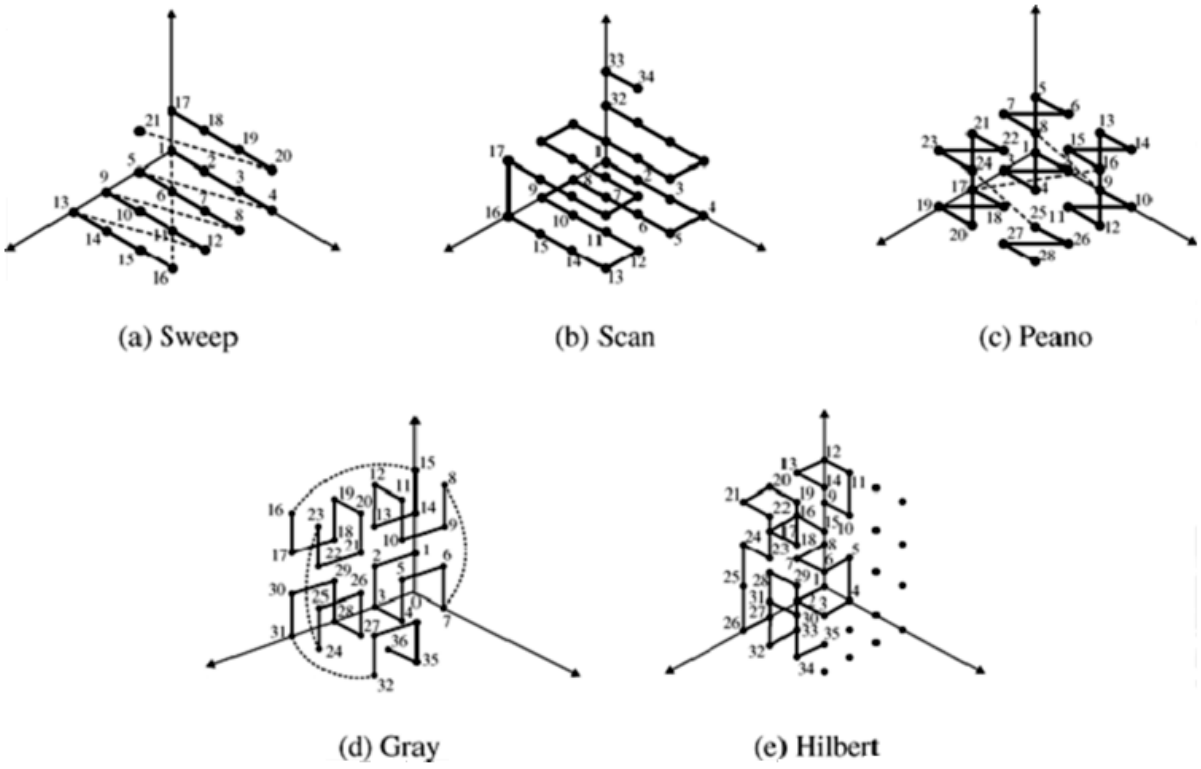


Figura 3.8: SFCs Tridimensionais [30]

As SFCs podem ser classificadas em: *recursivas* (*recursive SFCs* - RSFCs) ou *não-recursivas* [6]. Uma RSFC bidimensional é uma SFC que pode ser divida recursivamente em quatro RSFCs de igual tamanho. Exemplos de RSFCs são as de Peano (figura 3.6(c)), a de Gray (figura 3.6 (d)) e a de Hilbert(figura 3.6 (e)). As SFCs constituem exemplos de *fractais*, porém nem todos os fractais são SFCs e nem mesmo curvas [6].

3.4.1 Curva de Hilbert

Lawder [6] define a ordem de uma curva como o número de passos, ou iterações, realizadas no processo de construção de determinado estágio da curva. A curva de Hilbert bidimensional de primeira ordem, denotada H_1, está representada na Figura 3.9. Para construir uma curva de

ordem *i*, cada vértice da curva H_i é trocado pela curva de ordem *i-1*, que deve ser convenientemente girada ou refletida. A Figura 3.9 também apresenta curvas de ordens 2, 3 e 4, ou seja, H_2, H_3 e H_4, respectivamente. Uma curva de Hilbert de ordem *i* é dita uma aproximação da curva de Hilbert de preenchimento do espaço, logo as curvas H_i não são curvas de Hilbert, mas sim aproximações.

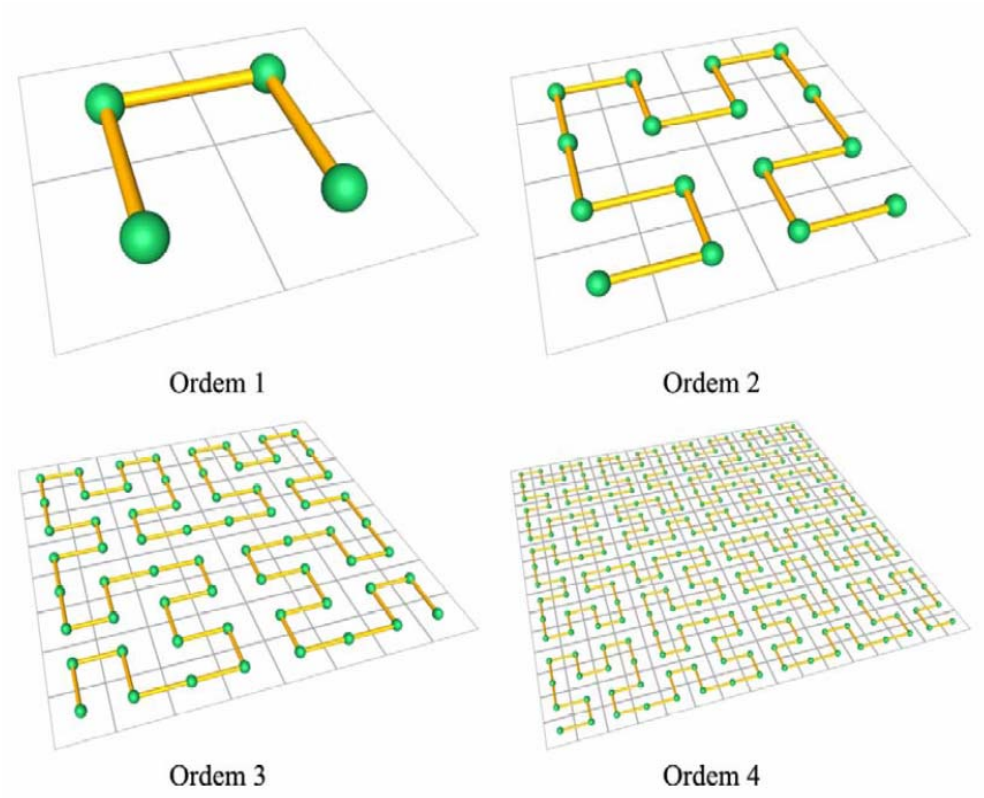


Figura 3.9: Curvas de Hilbert de Ordens 1, 2, 3 e 4 [31]

A curva de Hilbert tridimensional, definida em um cubo unitário, é uma generalização, não trivial, da curva de Hilbert bidimensional. Na bidimensional as curvas de ordem *i* são construídas a partir da união de curvas de ordem $i - 1$ convenientemente giradas ou refletidas, na curva tridimensional, dependendo da curva de primeira ordem obtida, pode não ser possível a obtenção das curvas de outras ordens. Por exemplo, considere uma curva de primeira

ordem como a da Figura 3.10, observa-se que com oito curvas iguais a esta não é possível a construção de uma curva de segunda ordem.

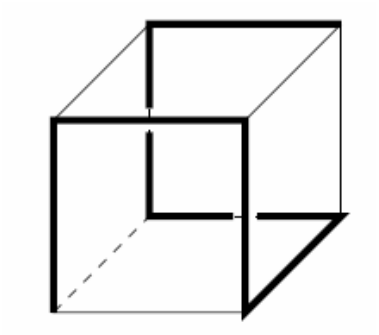


Figura 3.10: Estrutura de Conexão que não dá origem a Curvas de Segunda Ordem [6]

Tal fato justifica o cuidado que se deve ter ao desenvolver ou utilizar um algoritmo de construção da curva de Hilbert. Por exemplo, no presente trabalho, optou-se por um algoritmo que gera uma curva de Hilbert tridimensional de primeira ordem como a da Figura 3.11.

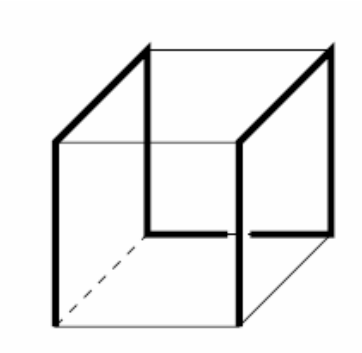


Figura 3.11: Curva de Hilbert Tridimensional de Primeira Ordem [6]

Com oito curvas iguais à curva apresentada na Figura 3.10, obtém-se uma curva de segunda ordem como a da Figura 3.12.

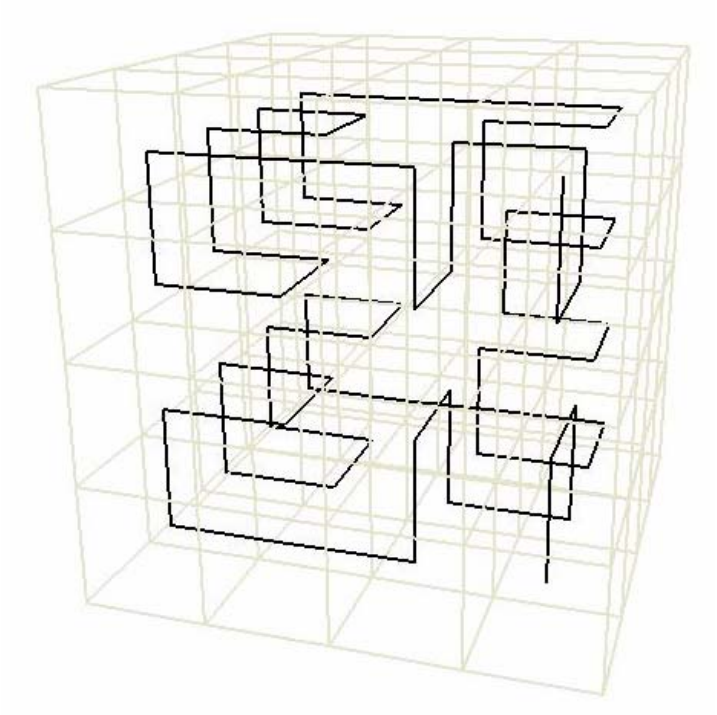


Figura 3.12: Curva de Hilbert Tridimensional de 2ª Ordem

3.5 Curva de Hilbert Modificada

A curva de Hilbert modificada [1] não é refinada uniformemente, ou seja, possibilita o refinamento das regiões do domínio (células) de maior interesse. Assim, diferentes níveis de refinamento podem ocorrer numa mesma etapa da construção da curva. Esta característica permite, no caso da Visualização Científica, uma redução significativa no tempo de processamento das imagens pretendidas, como ocorre na solução de equações diferenciais parciais [1, 2, 3].

Conforme Costa [26], considere *C* uma sub-árvore em um nível de profundidade n. Deseja-se percorrer as quatro sub-árvores de *C* pela curva de Hilbert bidimensional.

Numere as sub-árvores de *C* como na Figura 3.13.

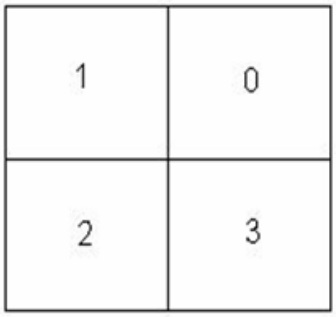


Figura 3.13: Sub-árvore *C* [26]

Os nodos 0, 1, 2, 3 representam os pais das sub-árvores de *C*. O algoritmo que percorre as sub-árvores de *C* na seqüência 0, 1, 2, 3, deve ser iniciado na sub-árvore 0 e terminar na sub-árvore 3, por simplificação, denominado algoritmo 03. Na Figura 3.14 o nodo 1 é folha, portanto não há o que percorrer em 1, mas 0 não é.

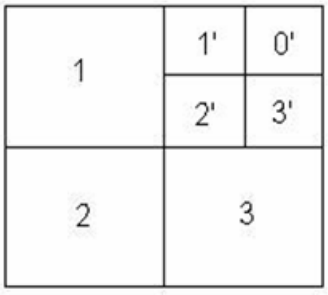


Figura 3.14: Sub-árvore 0 [26]

Para percorrer as sub-árvores de 0 e continuar obedecendo ao algoritmo 03, a curva que percorre 0 deve terminar em 1’ ou 2’. Terminando em 1’, a seqüência deve ser 0’, 3’, 2’, 1’, e em 2’ a seqüência deve ser 3’, 0’, 1’, 2’. Convencionando que a curva deve começar pela folha que estiver mais ao nordeste e terminar na folha mais ao noroeste da sub-árvore *C*. Res-

ta então a opção de percorrer 0 na seqüência 0’, 3’, 2’, 1’. Simplificando, como no algoritmo 03, defina esta seqüência como algoritmo 01. Percorrendo 0 com o algoritmo 01, obtém-se a Figura 3.15.

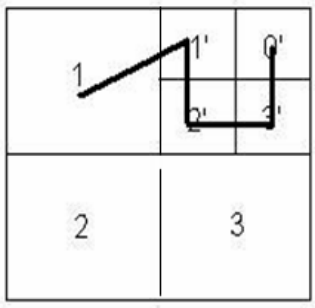


Figura 3.15: Percorrendo a Sub-árvore 0 [26]

A próxima etapa é percorrer 1. Se 1 é folha não há o que fazer, caso contrário, pode-se explicitar as quatro sub-árvores de 1: 0”, 1”, 2”, 3” (Figura 3.16). Pelo algoritmo 01 deve-se partir necessariamente de 0”. A única possibilidade de chegar em 2 é fazer o percurso: 0”, 1”, 2”, 3”. Portanto para percorrer 1 utiliza-se o algoritmo 03.

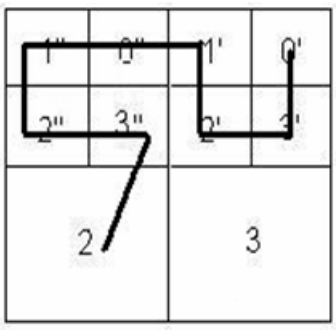


Figura 3.16: Percorrendo a Sub-árvore 1 [26]

Seguindo o mesmo raciocínio e fazendo as hipóteses semelhantes, conclui-se que a sub-árvore 2 deve ser percorrida pelo algoritmo 03 e que a sub-árvore 3 pelo algoritmo 23, onde 23 deve ligar a folha mais ao sudoeste à folha posicionada mais ao sudeste.

Portanto apresenta-se o Algoritmo 1:

Algoritmo 1 Algoritmo 03

1: **Função:** 03 (Sub-árvore C)

2: **se** C não é folha **então**

3: 01 (C → 0)

4: 03 (C → 1)

5: 03 (C → 2)

6: 23 (C → 3)

7: **fim se**

Além disso, pode-se definir de forma compacta o algoritmo 03, como segue:

03 : 01 (0), 03 (1), 03 (2), 23 (3).

E de forma análoga a 03:

01 : 03 (0), 01 (3), 01 (2), 21 (1).

23 : 21 (2), 23 (1), 23 (0), 03 (3).

21 : 23 (2), 21 (3), 21 (0), 01 (1).

Segundo Costa [26], por indução verifica-se que, se M é uma malha de nível máximo de profundidade, os quatro algoritmos 01, 03, 23 e 21 percorrem toda a malha M.

Utilizando esses quatros algoritmos obtém-se uma curva semelhante às curvas geradas nas etapas de construção da curva de Hilbert. A diferença é que, no processo de construção da curva de Hilbert, ela é refinada uniformemente em cada etapa e na curva obtida com a estratégia proposta não, cf. Figura 3.17.

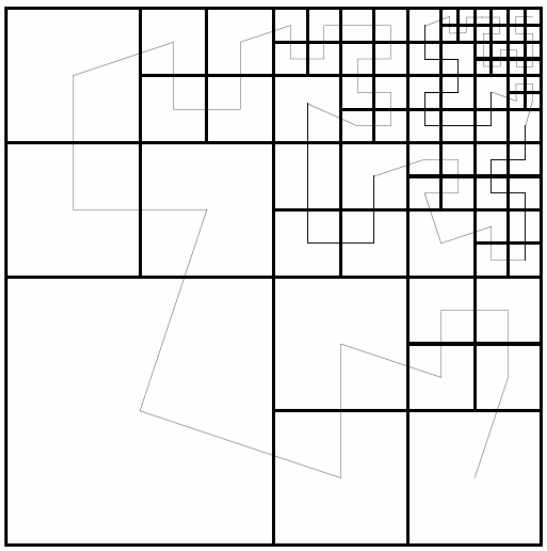


Figura 3.17: Curva de Hilbert Modificada Bidimensional [1]

3.5.1 Curva de Hilbert Modificada Tridimensional

A estratégia para construção da curva de Hilbert modificada tridimensional é baseada no caso bidimensional. No caso tridimensional, cada nodo que não é folha aponta para oito sub-árvores, numeradas como na Figura 3.18:

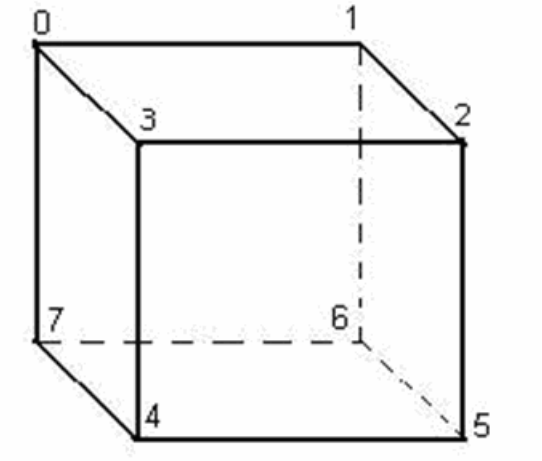


Figura 3.18: Representação das Sub-árvores [26]

Os nodos 0, 1, 2, 3, 4, 5, 6, 7 representam os pais das sub-árvores de C. O algoritmo que percorre as sub-árvores de C nessa seqüência será denominado algoritmo 07 (Figura 3.19), como no caso bidimensional.

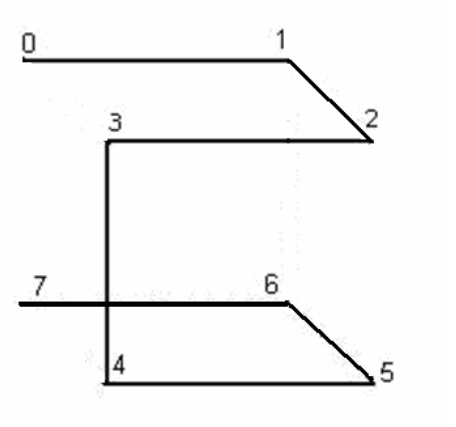


Figura 3.19: Algoritmo 07

Aplicando os procedimentos descritos no caso bidimensional, pode-se encontrar os algoritmos para ordenação da malha pela curva de Hilbert modificada tridimensional. Entre eles, alguns, embora diferentes, possuem origem e destino comuns produzindo o mesmo efeito

e podendo ser substituídos uns pelos outros. Após simplificar tais redundâncias, restam apenas doze curvas e usando a mesma nomenclatura utilizada no caso bidimensional, os algoritmos para a ordenação da curva serão:

07: 01(0) 03(1) 03(2) 25(3) 25(4) 47(5) 47(6) 67(7)

01: 07(0) 03(7) 03(4) 45(3) 45(2) 21(5) 21(6) 61(1)

03: 01(0) 07(1) 07(6) 65(7) 65(4) 43(5) 43(2) 23(3)

25: 23(2) 21(3) 21(0) 07(1) 07(6) 65(7) 65(4) 45(5)

47: 45(4) 43(5) 43(2) 21(3) 21(0) 07(1) 07(6) 67(7)

67: 61(6) 65(1) 65(2) 23(5) 23(4) 47(3) 47(0) 07(7)

45: 43(4) 47(3) 47(0) 01(7) 01(6) 65(1) 65(2) 25(5)

21: 25(2) 23(5) 23(4) 47(3) 47(0) 01(7) 01(6) 61(1)

61: 67(6) 65(7) 65(4) 43(5) 43(2) 21(3) 21(0) 01(1)

65: 67(6) 61(7) 61(0) 03(1) 03(2) 25(3) 25(4) 45(5)

43: 45(4) 47(5) 47(6) 61(7) 61(0) 03(1) 03(2) 23(3)

23: 21(2) 25(1) 25(6) 67(5) 67(4) 43(7) 43(0) 03(3)

Assim, apresentam-se doze algoritmos para construção da curva de Hilbert modificada tridimensional (Figura 3.19) válidos e utilizados para ordenação dos nodos pretos (seção 3.1) criados pela estrutura de dados utilizada no visualizador desenvolvido no presente trabalho.

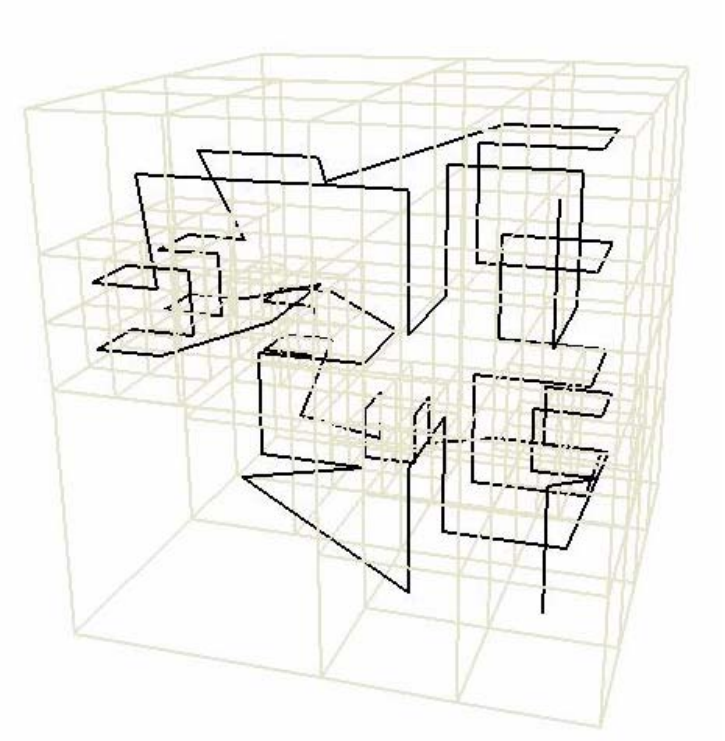


Figura 3.17: Curva de Hilbert Modificada Tridimensional

Capítulo 4

VisHil3d – O Visualizador Desenvolvido

A Visualização Científica envolve a exploração, transformação e mapeamento de dados em objetos gráficos. Neste sentido, o sistema de visualização deve ser usado como parte integrante do processo de investigação.

Um modelo do processo de visualização orientado a dados teve profunda influência no desenvolvimento de ferramentas. Estes dados são transformados em um número de passos lógicos até serem representado como imagem [32]. O modelo supõe que existe uma fonte de dados, que pode ser uma simulação ou um conjunto de observações ou medições. A Figura 4.1 mostra o esquema geral deste modelo.

Figura 4.1: Esquema do Modelo de Processo de Visualização [12]

O primeiro passo é o de filtro, no qual os dados de interesse são derivados da fonte original crua. Este passo pode envolver uma variedade de operações como, por exemplo, interpolação a partir dos dados esparsos em uma malha regular, extração de uma seção específica dos dados, ou ainda, realização de alguma operação de suavização.

A segunda etapa, que corresponde à construção da representação dos dados, é denominada de mapeamento. Normalmente quando se cria um modelo as informações gráficas armazenadas neste, (coordenadas, tamanhos, cores, espessuras, etc.) dizem respeito à aplicação e não ao dispositivo que está sendo usado. Por exemplo, quando aplicação é um sistema de desenho de plantas, as dimensões de uma porta ou de uma parede, são armazenadas em metros ou polegadas e não em pontos de tela. Outro exemplo deste fato são sistemas de traçado de curvas, por exemplo, no caso do traçado da função "seno" os valores sobre o eixo X variam entre 0 e 2π e sobre Y de -1 até +1. Traçar estes valores diretamente pode ocasionar alguns problemas, pois serão vistos apenas alguns pontos no canto superior esquerdo da tela. Para permitir a visualização deste tipo de modelos faz-se necessário realizar uma conversão dos valores do modelo para valores compatíveis com as dimensões da tela. A esta conversão dá-se o nome de mapeamento.

O último passo desse modelo de processo de visualização é a renderização, ou seja, a representação geométrica é convertida em imagem, a qual pode ser desenhada.

Embora esse modelo seja bastante importante, ele pode dar a falsa impressão de que o processo de visualização está centrado nos dados [12]. Os dados só são úteis se for possível conhecer o comportamento do modelo do qual eles foram extraídos.

O desenvolvimento do *Visualizador* para navegação em cenários tridimensionais baseado na curva de *Hilbert* modificada *Tridimensional* ou, simplesmente, VisHil3d (Figura 4.2), foi orientado tendo como principal parâmetro a exploração das potencialidades do computador e os princípios e componentes da SciVis (seção 2.2), desde a placa gráfica aos periféricos, de modo a torná-lo mais eficiente. Foi valorizada a facilidade com que as tarefas pretendidas eram executadas e a comodidade ao executá-las. A implementação foi realizada em C++, com a utilização de duas interfaces (seções 4.1 e 4.2) e está disponível para os sistemas operacionais GNU/Linux e Windows.

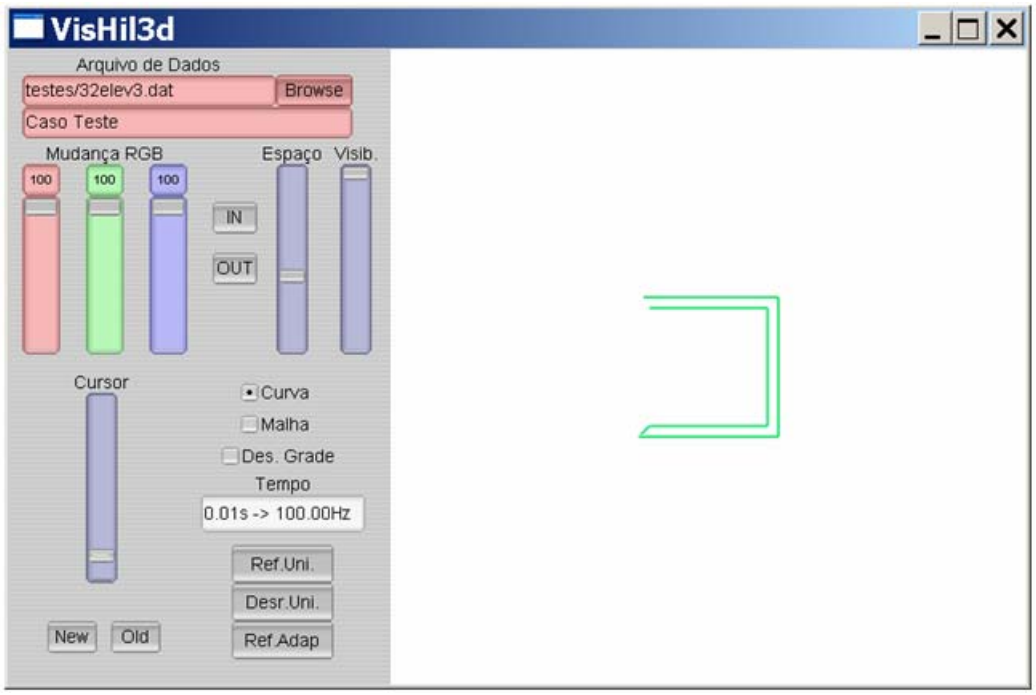


Figura 4.2: O Visualizador Desenvolvido

As escolhas e decisões a tomar ao longo do processo de criação foram feitas com base na compreensão dos usuários. As necessidades de ferramentas voltadas para a visualização de resultados numéricos, por parte dos usuários, foram identificadas e, a partir daí, buscou-se a criação de uma ferramenta útil, utilizável, agradável e simples.

4.1 Interface de Programação - OpenGL

A biblioteca *Open Graphics Library,* ou simplesmente OpenGL, é uma API (*Application Programmer's Interface*) aberta para o desenvolvimento de aplicações gráficas tridimensionais que pode ser incorporada a qualquer sistema de janelas (*MS Windows*, *X Window,* por exemplo). Com ela o programador pode produzir imagens através do controle direto do seu conjunto de rotinas. Embora não disponha de licença livre, não exige licença para utilização e está disponível gratuitamente.

A OpenGL pode ser integrada a outras ferramentas para gerenciamento de rotinas de criação e gerenciamento da renderização. O VisHil3d utiliza a biblioteca alternativa GLUT (*OpenGL Utility Toolkit* [33]) para a geração dos objetos gráficos tridimensionais. Para traçar um cubo, por exemplo, executa a rotina glutSolidCube(), onde o argumento definirá o tamanho de sua aresta. Uma vantagem em utilizar a GLUT é a compatibilidade com quase todas as implementações OpenGL em Windows e X.

OpenGL possui dois modos diferentes para tratar cor: o modo RGBA e o modo indexado de cor. A GLUT provê uma rotina denominada glutInitDisplayMode(), onde a seleção é feita através dos parâmetros GLUT_RGBA ou GLUT_INDEX. Caso não seja especificado nenhum dos modos o parâmetro *default* é o GLUT_RGBA. Este modo possui as componentes vermelho, verde, azul e alfa. Os três primeiros representam as cores primárias e são lineares (variando de 0.0 a 1.0). A componente alfa é utilizada em operações de *blending* (mistura) e transparência e representa a opacidade da cor, variando de 0.0, totalmente transparente, até 1.0, totalmente opaca. O valor alfa não é visível na tela, sendo usado apenas para determinar como o dado será exibido.

Entre os modelos locais de iluminação, os mais conhecidos são os de Gouraud [34], default do OpenGL, e de Phong [35]. No método de Phong, a normal nos pontos dentro de uma célula é calculada através da interpolação das normais nos vértices da mesma. Já no método de Gouraud a iluminação é calculada nos vértices da célula e interpolada em seu interior. O método utilizado neste trabalho é o de Gouraud, pois no do visualizador dá origem a imagens melhores do que o método de Phong, e com baixo custo computacional [36].

4.2 Interface com o Usuário - FLTK

A criação de uma aplicação requer o uso de uma série de ferramentas de programação auxiliares que permitam a criação e manipulação de componentes visuais tais como menus, botões de comando, caixas de diálogo. Tais ferramentas são oferecidas sob a forma de uma biblioteca de rotinas ou *toolkits.*

No VisHil3d a ferramenta utilizada foi o *Fast Light ToolKit* (FLTK) [37]. Ela é uma ferramenta de desenvolvimento de interfaces gráficas para a linguagem C++, disponível para diversos sistemas operacionais (GNU/Linux – através do sistema gráfico X11, Windows, Mac OS X, OS/2 e Solaris). A escolha de tal ferramenta deve-se ao suporte gráfico tridimensional (comum em jogos modernos e visualização científica) via OpenGL, bem como suporte à biblioteca GLUT.

Através do FLTK é possível separar o código da aplicação, em *C* ou *C++*, por exemplo, do código que define e manipula a interface, escrito em uma linguagem interpretada. Esta separação se justifica, pois o desenho da interface é a parte que sofre maiores alterações desde o primeiro estágio até o produto final.

O FLTK fornece um rico conjunto de componentes gráficos e aplicações construídas com ele tendem a ser eficientes (mesmo quando ligados estaticamente). O FLTK é um Software Livre distribuído sob a licença LGPL (*GNU Library General Public License*), permitindo inclusive que os programas criados sejam comercializados sem distribuição do código-fonte.

4.3 Dados Representados

Neste trabalho os dados tratados são de natureza escalar e definidos através de uma grade tridimensional com valores escalares associados a cada ponto da grade. Podem-se entender os dados tratados como uma n-upla <i,j,k,E> que define um ponto amostrado do campo escalar na posição (i,j,k) e com valor E associado.

4.3.1 Gerenciamento dos Dados

O gerenciamento dos dados é uma tarefa importante na Visualização Científica. Algoritmos de gerenciamento de dados são responsáveis pela importação e exportação de arquivos e dados. Esses algoritmos também são responsáveis pelo controle da memória interna utilizada na aplicação.

Uma das grandes dificuldades na implementação de uma ferramenta de visualização é a organização dos dados. A organização interna é fundamental para a eficácia de uma técnica de visualização. Aspectos importantes são: eficiência de armazenamento e execução e o suporte à análise.

4.3.2 Formato dos Arquivos de Entrada

Neste trabalho, os dados tratados consistem de uma seqüência de informações que são facilmente lidas e escritas, tanto manual quanto computacionalmente. Os dados tratados são de

natureza escalar e estão associados às coordenadas dos centros das células da malha gerada pela curva de Hilbert de preenchimento do espaço de maior ordem. Esta ordem é uma potência de dois igual à raiz cúbica da quantidade de pontos que devem ser lidos ou a menor potência de dois maior que esta raiz.

O formato do arquivo de dados deve conter três partes básicas:

a) cabeçalho, que consiste de um conjunto de no máximo 256 caracteres, ocupando apenas a primeira linha. Ele pode ser usado para descrever os dados ou qualquer outra informação pertinente;

b) um número inteiro especificando o número de pontos que deverão ser armazenados;

c) seqüência de valores escalares.

O VisHil3d gerencia os dados relacionando os mesmos às coordenadas dos centros das células geradas pela curva de Hilbert tridimensional. Portanto, para garantir a correta utilização do visualizador desenvolvido o arquivo de dados de entrada deve ser gerado segundo o *Algoritmo 2*.

Algoritmo 2 Gerando o Arquivo de Dados

1: Escreva o cabeçalho do arquivo, no máximo 256 caracteres;

2: Escreva a quantidade de dados que deverão ser lidos pelo VisHil3d, chamada ***quant***;

3: De ***z*** = 0 até ***quant***, Faça:

4: De ***y*** = 0 até ***quant***, Faça:

5: De ***x*** = 0 até **quant**, Faça:

6: Escreva o valor associado;

7: ***x*** = ***x*** + 1;

8: Fim De;

9: ***y*** = ***y*** + 1;

10: Fim De;

11: ***z*** = ***z*** + 1;

12: Fim De.

4.3.3 Organização Interna dos dados

Os dados podem ser gerados de duas maneiras: medidas experimentais, através da amostragem de processos empíricos, ou simulações computacionais. Desta forma, são conhecidos apenas em pontos discretos. Isso cria uma dificuldade, pois uma das atividades relevantes em visualização científica é justamente determinar valores em pontos arbitrários. A solução mais utilizada para solução desse problema é o uso de funções de interpolação, através das quais se infere uma relação entre valores de dados vizinhos. A função de interpolação mais usual é a linear, mas existem funções quadráticas, cúbicas, do vizinho mais próximo (grau zero), entre outras.

O VisHil3d organiza os dados obtidos admitindo que somente será gerada uma curva de Hilbert tridimensional de maior ordem possível. Desta forma, foi desenvolvido um algoritmo de interpolação para determinar os valores referentes aos centros das células que não estão totalmente refinadas. Tal algoritmo é utilizado apenas no momento em que os dados são adquiridos e os valores resultantes são armazenados na memória. Ele ordena os valores com uma curva de Hilbert tridimensional de maior ordem possível, faz uma média dos valores das células irmãs e atribui este valor à célula pai deste ramo, ou seja, esta interpolação, utilizada pelo seu baixo custo computacional, é um caso especial da interpolação linear. Os problemas relacionados à interpolação foram considerados no desenvolvimento do visualizador, apesar de refletirem uma limitação que pode ser tratada em trabalhos futuros.

4.3.4 Função de Transferência

A função de transferência é uma função matemática que associa os valores dos dados volumétricos a propriedades ópticas, como cor e opacidade. Para um campo escalar s: $(x, y, z) \rightarrow F(x, y, z)$, utilizando o sistema de cores RGBA, tem-se $FT: s \mapsto (R(s), G(s), B(s), \alpha(s))$, onde α representa a opacidade.

Essa função é fundamental para o desenvolvimento de ferramentas de visualização científica, pois transforma dados abstratos em características que podem ser visualizadas. A função de transferência pode ser especificada pelo usuário ou gerada por métodos semi-automáticos e ser bastante complexa. É possível também se utilizarem funções de transferência multidimensionais definidas em função das propriedades relacionadas aos dados volumétricos; por exemplo, em função de um campo escalar e o gradiente deste campo.

A ferramenta desenvolvida neste trabalho mapeia os valores contidos no arquivo de entrada de forma a associá-los a uma faixa de cor (Figura 4.3) relacionada ao esquema "arco-íris" (Algoritmo 3) [36] e permite que o usuário controle a opacidade, que é inicialmente 0. A opção por este esquema deve-se à qualidade das imagens geradas para o Caso Teste (seção 5.1).

Algoritmo 3 Cores Arco-íris

1: Dado u_0 definido entre 0 e 1;

2: $r = u_0 * u_0$;

3: $b = (u_0 - 1.0) * (u_0 - 1.0)$;

4: $g = 1.0 - (b-r) * (b-r)$;

5: Retorna r, g, b;

Figura 4.3: Esquema de Cores Arco-Íris [36]

Capítulo 5

Resultados

5.1 Caso Teste

O visualizador desenvolvido foi testado em um problema de condução do calor em um cubo constituído de um material homogêneo isotrópico, ou seja, neste cubo o calor flui de pontos quentes para pontos frios na direção em que a diferença de temperatura é a maior. A equação de condução de calor tridimensional descrita, por exemplo, em [38], pode ser expressa como:

$$u_t = a^2(u_{xx} + u_{yy} + u_{zz})$$

onde a representa o coeficiente de condutividade térmica. De posse dessa equação, por uma expansão em série de Taylor, que dá origem ao *método das diferenças finitas* [39], ter-se-á:

$$v_{i,j,k,l+1} = v_{i,j,k,l} + a^2\Delta t\left(\begin{array}{l}\dfrac{v_{i-1,j-1,k-1,l} - 2v_{i,j-1,k-1,l} + v_{i+1,j-1,k-1,l}}{\Delta x^2} \\ +\dfrac{v_{i-1,j-1,k-1,l} - 2v_{i-1,j,k-1,l} + v_{i-1,j+1,k-1,l}}{\Delta y^2} \\ +\dfrac{v_{i-1,j-1,k-1,l} - 2v_{i-1,j-1,k,l} + v_{i-1,j-1,k+1,l}}{\Delta z^2}\end{array}\right)$$

em que i é o contador para a posição x, j é contador para a posição y, k é o contador para a posição z e l o contador para o tempo.

As condições para a garantia de convergência são

$$0 < \frac{\Delta t}{\Delta x^2} a^2 \leq \frac{1}{2}, \qquad 0 < \frac{\Delta t}{\Delta y^2} a^2 \leq \frac{1}{2} \qquad \text{e} \qquad 0 < \frac{\Delta t}{\Delta z^2} a^2 \leq \frac{1}{2}$$

de tal forma que

$$\Delta x, \Delta y, \Delta z \to 0 \Rightarrow v \to u$$

Na resolução desse problema de condução de calor tridimensional, a estrutura utilizada foi um cubo. Num passo posterior, o corpo poderia tomar a forma que se quisesse (por exemplo uma esfera, ou apresentar partes interiores a temperaturas diferentes, ou mesmo um corpo com uma cavidade), desde que as condições de contorno e forma fossem condizentes.

Retomando a idéia inicial, o cubo, apresenta seis faces, oito vértices e doze arestas, tal que quando da diferença de temperaturas de contorno das faces, resolveu-se fazer a média aritmética na aresta de duas faces adjacentes, bem como a média aritmética no ponto de intersecção (vértice) de três faces adjacentes.

Na seção 5.2 encontram-se os resultados referentes ao aquecimento de um cubo de aresta 1 metro de comprimento, com uma temperatura inicial igual a 0ºC e as a 100º C, com a igual a 1, aquecida ao longo de 20 segundos.

5.2 Desempenho

Os testes de desempenho foram executados em uma máquina com processador *Intel Pentium IV,* de *2.28 GHz* e *256MB* de memória *RAM*. A placa gráfica utilizada foi uma *NVIDIA GeForce4 MX 4000*.

Para uma mesma condição inicial e de fronteira, após 40 tentativas, obtiveram-se os tempos apresentados na Tabela 2. Estes valores referem-se ao processo de refinamento da curva de ordem máxima, para efeito de simplificação denominada Grade, sem utilizar a curva de Hilbert modificada tridimensional. A Figura 5.1 apresenta a Malha gerada pelo teste com a Grade 16x16x16.

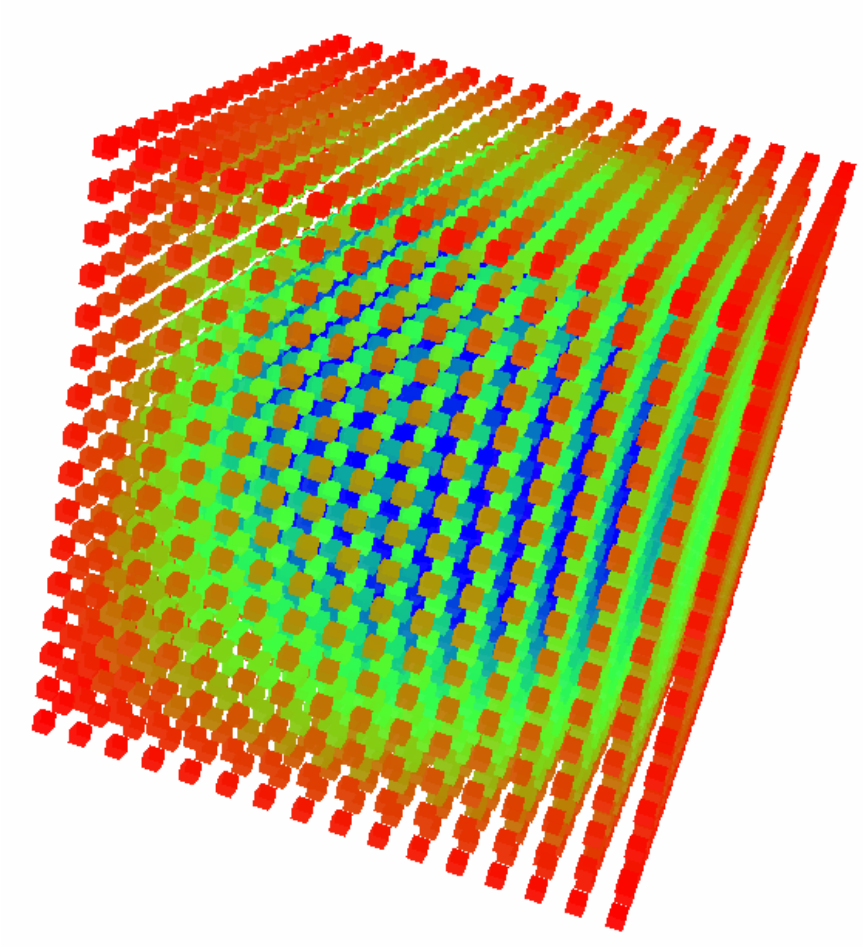


Figura 5.1: Malha Gerada com Grade 16x16x16

Grade	Curva		Malha	
	Min.	Máx.	Min.	Máx.
64x64x64	0.44	0.54	0.44	0.56
126x126x126	0.64	0.81	2.08	2.25
256x256x256	1.82	1.89	7.64	7.85

Tabela 2: Desempenho (em segundos): Curva de Hilbert Tridimensional

Na Tabela 3 apresentam-se os resultados referentes a ordenação dos dados pela curva de Hilbert modificada tridimensional de maior refinamento possível. Como critério para o refinamento define-se que uma célula deve ser refinada se o valor referente a esta célula é diferente de 0 e se os valores de suas filhas, células geradas com o seu refinamento, são diferentes entre si. A Figura 5.2 apresenta a curva gerada pela Curva de Hilbert modificada Tridimensional no teste com Grade 64x64x64.

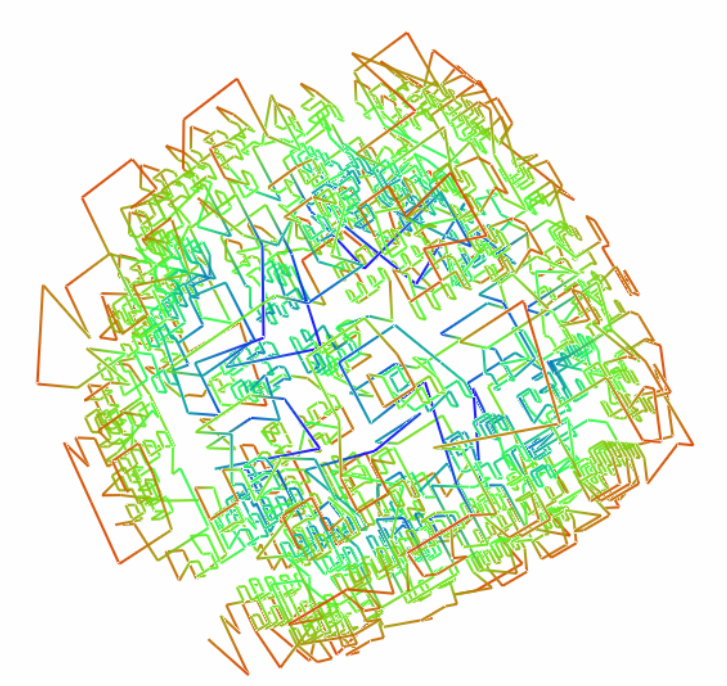


Figura 5.2: Curva Gerada pela Curva de Hilbert Modificada Tridimensional com Grade 64x64x64

Grade	Curva		Malha	
	Min.	**Máx.**	**Min.**	**Máx.**
64x64x64	0.06	0.06	0.17	0.18
126x126x126	0.09	0.12	0.25	0.32
256x256x256	0.21	0.24	0.68	0.75

Tabela 3: Desempenho (em segundos): Curva de Hilbert Modificada Tridimensional

Na Tabela 4 apresentam-se a comparação entre os tempos de geração das imagens com a curva de Hilbert modificada tridimensional referente às respectivas grades para um usuário no interior do cubo unitário e um usuário no exterior do mesmo.

Grade	Interior		Exterior	
	Min.	**Máx.**	**Min.**	**Máx.**
64x64x64	0.03	0.05	0.09	0.12
126x126x126	0.10	0.13	0.21	0.24
256x256x256	1.04	1.43	1.13	2.55

Tabela 4: Desempenho (em segundos): Interior **x** Exterior

Os tempos apresentados na Tabela 4, quando comparados, evidenciam a eficiência da estrutura desenvolvida em relação à geração de imagens relacionadas ao interior do domínio em estudo, mesmo quando a curva gerada passa por todos os pontos, inclusive os fora da cena.

Para comparação do desempenho e da qualidade das imagens produzidas com a estratégia utilizada neste trabalho, foi desenvolvida uma função baseada nas estratégias propostas

por Espinha [18], implementado em linguagem Cg [40] através de programação da placa gráfica. Este programa utiliza renderização volumétrica direta de volumes.

O melhor tempo para renderização da imagem com grade 16x16x16 foi de 7.45s e, como esperado, a possibilidade de interpretação do fenômeno de aquecimento do cubo foi prejudicada (Figura 5.3). Em análises de resultados provenientes de métodos numéricos a renderização volumétrica direta pode originar imagens onde ocorre uma mistura de informações e cores, dificultando a avaliação de cada dado separadamente, mas pode ser útil para uma avaliação prévia e geral de todo o modelo.

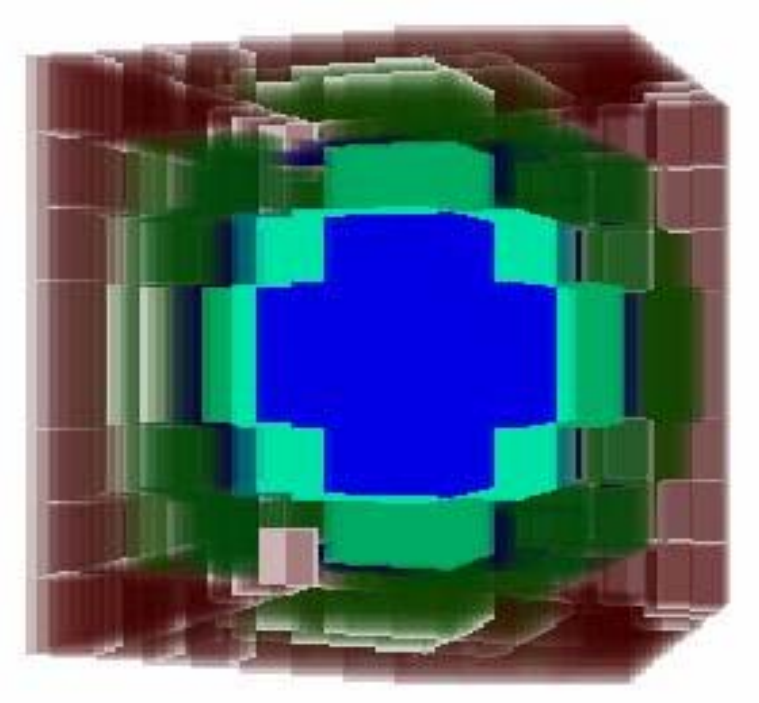


Figura 5.3: Renderização Direta de Volumes Utilizando a Linguagem Cg

5.3 Modificação Interativa da Escala de Cores

No VisHil3d é possível redefinir tabelas de cores e a opacidade da imagem. Elas podem ser redefinidas pelo usuário depois de uma primeira exploração dos dados, sendo que os intervalos são ajustados interativamente, até gerar um resultado aceitável.

No caso do aquecimento do cubo, por exemplo, podem-se visualizar apenas as regiões associadas às maiores temperaturas, cf. Figura 5.4, ou temperaturas mais baixas, cf. Figura 5.5.

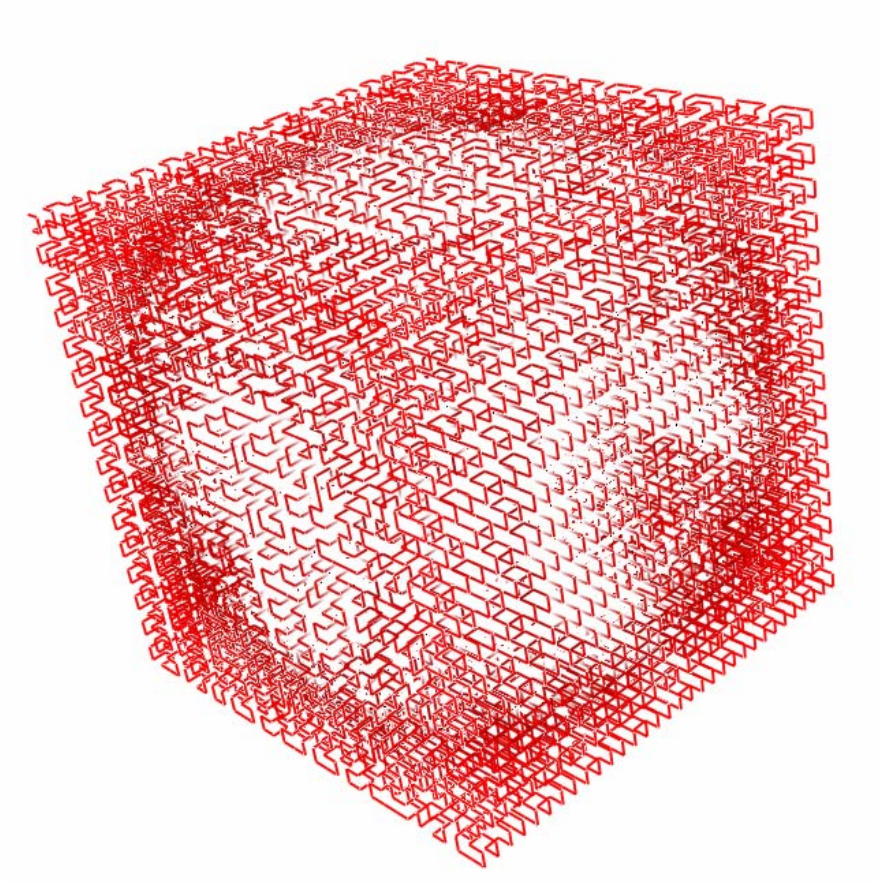


Figura 5.4: Região associada às temperaturas elevadas (32x32x32) – Malha Uniforme

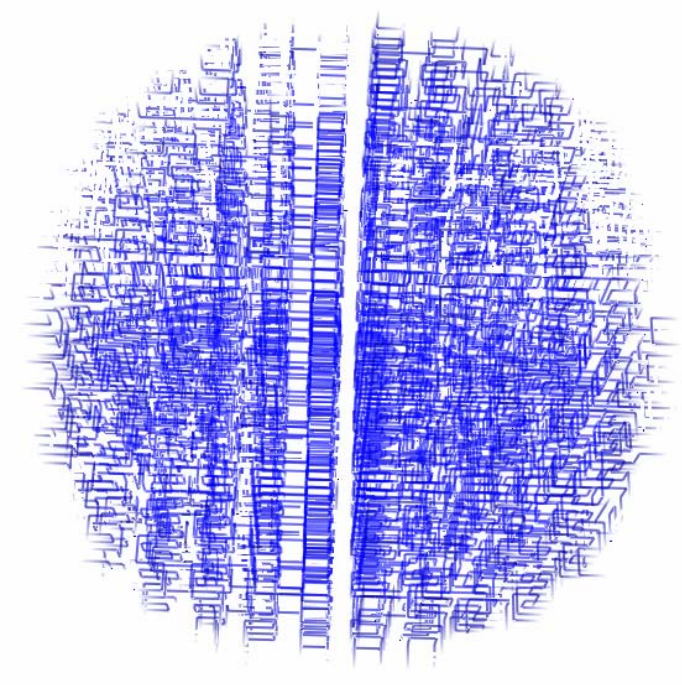


Figura 5.5: Região associada às temperaturas mais baixas (32x32x32) – Malha Uniforme

5.4 Seleção de Regiões

Baseado nas características fractais da curva de Hilbert [6] foi desenvolvida uma função que realiza a geração de uma curva de Hilbert modificada tridimensional dentro das regiões da imagem de maior interesse do usuário. A justificativa para a criação de tal função está nos problemas onde o usuário está interessado em visualizar o "interior" do problema tratado da mesma maneira que visualiza todo o fenômeno. A Figura 5.6 apresenta uma Malha definida pela curva de Hilbert modificada Tridimensional gerada por um arquivo de dados de ordem 32x32x32 e a Figura 5.7 apenas a região selecionada.

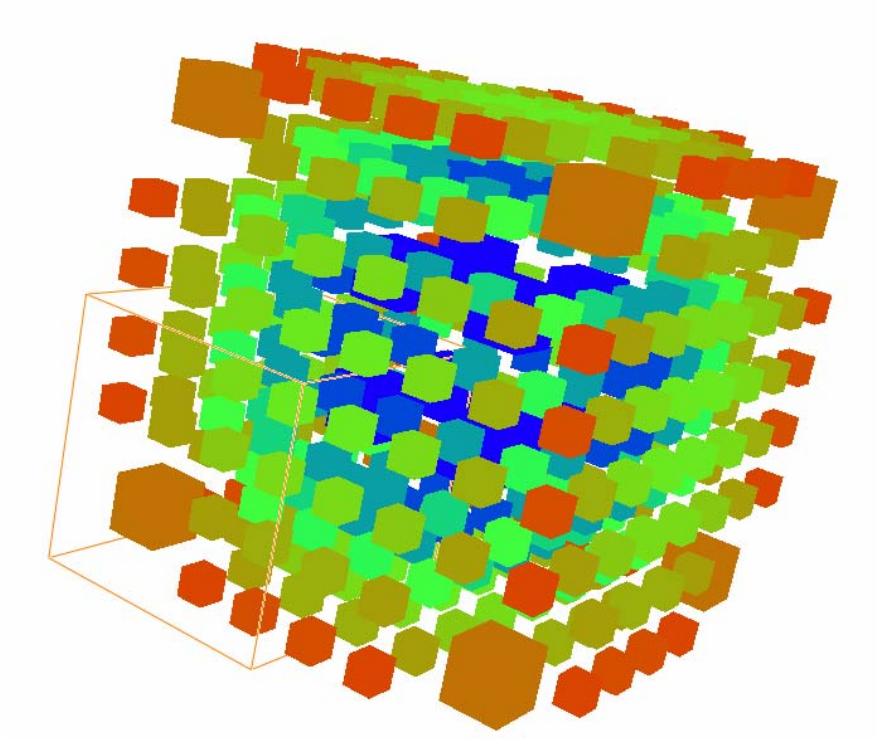


Figura 5.6: Malha gerada pela Curva de Hilbert Modificada tridimensional (Grade 32x32x32)

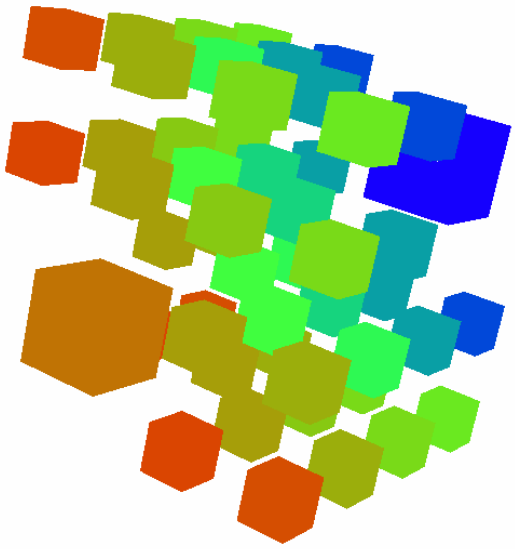


Figura 5.7: Seleção de Região

A utilização desta função dá maior controle ao usuário para que o mesmo possa decidir sobre qual região do domínio deve direcionar sua atenção.

Capítulo 6

Conclusão

Neste trabalho buscou-se o desenvolvimento de um aplicativo que utiliza um procedimento adaptativo baseado em uma modificação da curva de Hilbert tridimensional de preenchimento do espaço para a organização e renderização dos dados provenientes de simulações numéricas de fenômenos tridimensionais. Para tanto, foi realizado um estudo sobre a Visualização Cientifica e as curvas de preenchimento do espaço, em particular a curva de Hilbert tridimensional e o algoritmo, não trivial, para sua construção.

Apresentou-se uma estratégia baseada nas características fractais da curva de Hilbert modificada tridimensional que permite uma maior exploração e compreensão dos dados, permitindo que o usuário navegue pelo modelo numérico, possibilitando selecionar as regiões de maior interesse.

Em Computação Gráfica e Métodos Numéricos é fundamental a eficiência tanto de processamento quanto de armazenamento de dados. O VisHil3d, visualizador desenvolvido neste trabalho, é um aplicativo que satisfaz estas exigências, exibe imagens de boa qualidade e possui uma interface simples de usar.

O aplicativo desenvolvido não exige que o usuário seja especialista em Computação Gráfica possibilitando o seu uso em diferentes áreas do conhecimento como Engenharia, Física e Geografia, por exemplo. A eficiência na geração das imagens no VisHil3d transmite ao usuário a sensação de que a imagem realmente segue suas diretivas.

6.1 Trabalhos Futuros

Uma extensão deste trabalho é utilizar um método numérico para resolução de equações diferencias acoplado ao VisHil3d. Desta maneira, poderiam ser geradas imagens simultaneamente à resolução da equação diferencial, aumentando o poder de decisão do usuário, assim como facilitando a interpretação do problema tratado.

Os dados de entrada do VisHil3d são de natureza escalar. A possibilidade de tratamento de dados de natureza vetorial pode permitir a implementação de uma ferramenta de navegação segundo um campo de direções. O desenvolvimento de tal ferramenta pode, então, prover um visualizador com outras funções como traçar a trajetória de um determinado evento a partir da definição de um sistema de funções e um valor inicial, impressão do campo de vetores, e associação de uma função de classificação à direção ou ao módulo dos vetores adquiridos.

As estratégias de ordenação adotadas neste trabalho permitem a resolução de problemas em paralelo de forma bastante eficiente, minimizando a troca de mensagens entre os processadores. Portanto, outra sugestão de trabalho é implementar um visualizador utilizando a curva de Hilbert modificada tridimensional em um sistema paralelo, a fim de comprovar sua eficiência.

Referências

[1] BURGARELLI, D. Modelagem computacional e simulação numérica adaptativa de equações diferenciais parciais evolutivas aplicadas a um problema termoacústico. 1998. 86 f. *Tese de Doutorado*, PUC-Rio, Rio de Janeiro, 1998.

[2] BURGARELLI, D.; KISCHINHEVSKY, M.; PAES LEME, P. J.; SILVEIRA, O. T. Refinamento de malha adaptativa em paralelo, (*C3AD*) *(Colloquia em Computação Científica de Alto Desempenho)*, Rio de Janeiro, 1995.

[3] BURGARELLI, D.; KISCHINHEVSKY, M.; BIEZUNER, R. A new adaptive mesh refinement strategy for numerically solving evolutionary PDE´s. *Journal of Computational and Applied Mathematics*, vol. 196/1, p 115-131, 2006.

[4] OLIVEIRA, L. F.; OLIVEIRA, A. A.; ESPERANÇA, C.; Compressão de imagens usando transformada wavelet e curva Peano-Hilbert. *XI Simpósio Brasileiro de Computação Gráfica e Processamento de Imagens*, Rio de Janeiro, 1998.

[5] KAMATA, S.; KAWAGUCHI, E.; NIIMI, M. An interactive analysis method for multi-dimensional images using a Hilbert curve. *Systems and Computers in Japan*, 1995. p.83-92.

[6] LAWDER, J. K. The Application of Space-filling Curves to the Storage and Retrieval of Multi-dimensional Data. 1999. 221 f. *PhD Thesis* – University of London, London, 2000.

[7] WITTEN, I. H.; WYVILL, B. On the generation and use of space-filling curves. *Software - Practice and Experience*, 1983. p. 519-525.

[8] KAMEL, I.; CHRISTOS, F. Hilbert R-Tree: An improved R-Tree using fractals. *Proceedings of 20th International Conference on Very Large Data Bases*, Santiago de Chile, Chile, Sept. 1994. p. 500-509.

[9] BIALLY, T. Space-filling curves: Their generation and their application to bandwidth reduction. *IEEE Transactions on Information Theory*, IT - Nov. 1969. p. 658-664

[10] PAIVA, A. C.; SEIXAS, R. B.; GATTASS, M. Introdução à Visualização Volumétrica. *Departamento de Informática, PUC-Rio, Inf. MCC03/99*, Rio de Janeiro, 1999.

[11] MCCORMICK, B. H.; DEFANTI, T. A.; BROWN, M. D. Visualization in Scientific Computing. *Computer Graphics* (special edition), v. 21, n. 6, nov. 1987.

[12] BRODLIE, K. A classification scheme for scientific visualization. In: EARNSHAW , R. A.; WATSON , D. (Eds.) *Animation and Scientific Visualization:* tools & applications. Academic Press, 1993. p. 125-140.

[13] COLLINS, B. M. Data visualization: has it all been seen before? In: EARNSHAW , R. A.; WATSON , D. (Eds.) *Animation and Scientific Visualization: tools & applications.* Academic Press, p. 3-28, 1993.

[14] GELLER, M. J.; FALCO, E. E . Graphic voyages through the Universe. *IEEE Computer Graphics and Applications*, p. 7-11, IEEE, 1994.

[15] SVAKHINE, N.; EBERT, D. S.; STREDNEY, D . Illustration Motifs for effective medical volume illustration. *IEEE Computer Graphics and Applications*, p. 31-39. IEEE, 2005.

[16] BERKLEY , J. et al. Real-time finite element modeling for surgery simulation: an application to virtual suturing. *IEEE Transactions on Visualization and Computer Graphics,* v. 10, p. 314-325. IEEE, 2004.

[17] WENGER, A. et al., Interactive volume rendering of thin thread structures within multivalued scientific data sets. *IEEE Transactions on Visualization and Computer Graphics*, v. 10, n. 6, p. 664-672. IEEE, 2004.

[18] ESPINHA, R. S. Visualização volumétrica interativa de malhas não estruturadas utilizando placas gráficas programáveis. 2005. 96 f. *Tese de Mestrado*, PUC-Rio, Rio de Janeiro, 2005.

[19] KITWARE. VTK home page. Disponível em: <http://www.vtk.org>. Acesso em: 2 jun. 2005.

[20] ENCARNAÇÃO, J. L. Advanced research and development topics in animation and scientific visualization. In: EARNSHAW , R. A.; WATSON , D. (Eds.) *Animation and Scientific Visualization: tools & applications.* Academic Press, 1993. p. 37-73.

[21] MÄNTYLÄ, M. *An Introduction to Solid Modeling.* Rockville: Computer Science Press, 1988.

[22] OLIVEIRA, M. C. F. de; MINGHIM, R. Uma introdução à Visualização Computacional. In: JORNADA DE ATUALIZAÇÃO EM INFORMÁTICA (JAI), 16., 1997, Brasília, Ed. SBC, 1997. p. 85-131.

[23] HSV color space - Wikipedia, the free encyclopedia. Disponível em: <http://en.wikipedia.org/wiki/HSV_color_space>. Acesso em: 12 jun. 2004.

[24] SETRED. Disponível em: <http://www.setred.com/technology>. Acesso em: 14 jun. 2005.

[25] TREINISH, L. A. Unifying principles of data management for scientific visualization. In: EARNSHAW, R. A.; WATSON, D. (Eds.) *Animation and Scientific Visualization:* tools & applications. Academic Press, 1993. p. 141-170.

[26] COSTA, F. N. Refinamento de Malha Adaptativa Ordenada por uma Curva de Hilbert Modificada Tridimensional. Disponível em: <http://homepages.dcc.ufmg.br/~fabrimac>. Acesso em: 02 jun. 2005.

[27] HILBERT, D. Ueber stetige abbildung einer linie auf ein achenstuck. *Mathematische Annalen*, p. 459-460, 1891.

[28] P. GIUSEPPE. Sur une courbe, qui remplit toute une aire plane (on a curve which completely fills a planar region). Mathematische Annalen, p. 157-160, 1890.

[29] SAGAN, H. Space-Filling Curves. *Springer-Verlag*, 1994.

[30] MOKBEL M. F.; AREF W. G. Analysis of Multi-Dimensional Space-Filling Curves. *GeoInformatica*, 2003. p. 179-209.

[31] HOFMAN M. S. Tratamento Eficiente de Visibilidade Através de Árvores de Volumes Envolventes. 2002. 148 f. *Tese de mestrado*, PUC-Rio, Rio de Janeiro, 2002.

[32] UPSON, C. et al. The Advanced Visualization System: a computational environment for Scientific Visualization. *IEEE Computer Graphics & Applications.* IEEE, p. 30-42, 1989.

[33] M. J. KILGARD. The OpenGL Utility Toolkit (GLUT) Programming Interface API Version 3. Technical report, SGI. Disponível em: <http://www.opengl.org/developers/documentation/glut>. Acesso em: 12 jan. 2005.

[34] GOURAUD, H., *Continuous Shading of Curved Surfaces*, IEEE Transactions of Computers, Vol. 20, No. 6, pp. 623-629, 1971.

[35] PHONG, B. T., *Illumination for Computer Generated Pictures*, Communications of ACM, Vol. 18, No. 6, pp. 311-317, 1975.

[36] *Computer Graphics: Programming, Problem Solving, and Visual Communication.* Disponível em: < www.cs.csustan.edu/~rsc/NSF/>. Acesso em: 3 jan. 2006.

[37] FLTK. Disponível em: <http://www.fltk.org>. Acesso em: 3 jan. 2006.

[38] BIEZUNER R. J. Introdução às Equações Diferenciais Parciais. Disponível em: <www.mat.ufmg.br/~rodney/notas_de_aula/edp.pdf>. Acessado em: 03 jan. 2006.

[39] CUMINATO J. A., MENEGHETTE JR, M. Discretização de Equações Diferenciais Parciais: Técnicas de Diferenças Finitas, USP, São Paulo 2002. Disponível em: <http://www.icmc.usp.br/~jacumina>. Acesso em: 10 jan. 2006.

[40] NVIDIA CORPORATION. *NVIDIA GPU Programming Guide Version 2.2.1.* NVidia Corporation, novembro de 2004. Disponível em: <http://www.nvidia.com/object/gpu_programming_guide.html>. Acesso em: 20 abr. 2005.