UNIVERSIDADE ESTADUAL DE CAMPINAS

FACULDADE DE ENGENHARIA ELÉTRICA

DEPARTAMENTO DE ENGENHARIA DA COMPUTAÇÃO E AUTOMAÇÃO INDUSTRIAL

# Visualização de Volumes Aplicada à Área Médica

por Alexandre Xavier Falcão

orientador Prof. Dr. Roberto de Alencar Lotufo

Dissertação submetida à Faculdade de Engenharia Elétrica da Universidade Estadual de Campinas, para preenchimento dos pré-requisitos parciais para obtenção do Título de Mestre em Engenharia Elétrica.

junho 1993

# Resumo

Este trabalho apresenta um tutorial sobre os principais métodos utilizados na visualização de dados biomédicos 3D e mostra os resultados da implementação de um conjunto de ferramentas de visualização volumétrica expandindo o sistema Khoros.

As principais técnicas de visualização apresentadas são classificadas em três categorias básicas: baseadas em superfície, baseadas em voxel binário e de *rendering* de volume semitransparente. Esta dissertação também discute outras técnicas mais simples de exploração dos dados para visualização. Estas são classificadas como métodos interativos e envolvem refatiamento, reprojeção aditiva e radiográfica, projeção de intensidade máxima e o uso de pseudo-cores.

Entre os métodos acima foram implementadas quatro técnicas baseadas em voxel binário, duas de *rendering* de volume semitransparente, as técnicas de reprojeção radiográfica e aditiva, e a projeção de intensidade máxima. O sistema Khoros permite usar pseudo-cores, gerar uma animação e combinar imagens geradas pelas técnicas acima. Este sistema também oferece um conjunto de ferramentas de manipulação e processamento de imagens que podem ser utilizadas no processo de visualização.

# Abstract

This work presents a tutorial on the fundamental methods used for visualization of 3D biomedical data and shows results from the implementation of a set of volumetric visualization tools as an enhancement to the Khoros system.

The main visualization methods described here are classified in three basic categories: surface-based techniques, binary voxel techniques and semi-transparent volume rendering techniques. This dissertation also discusses some other simple techniques for data visualization. These techniques are classified as interactive methods and involve reslicing, additive and radiographic reprojection, maximum intensity projection and the use of pseudo-color.

Among the methods mentioned above, four different binary voxel techniques, two semi-transparent volume rendering techniques, the additive and radiographic reprojection, and the maximum intensity projection technique were implemented. The Khoros system allows using pseudo-colors, creating an animation and combining images from the above techniques. This system also offers a set of tools for digital image processing and manipulation.

# Agradecimentos

Gostaria de agradecer aos meus amigos Jorge Diz, Mário Queiroz, Armando Delgado, Heraldo Madeira e José Olguín pelo suporte computacional e a todos os outros que de alguma forma me apoiaram durante este período.

Ao Prof. Roberto Lotufo pelo incentivo, colaboração, orientação e paciência.

Ao Prof. Lincoln Moura pelo incentivo e colaboração.

Ao colega Roberto Gonçalves pelo incentivo e colaboração.

Aos Professores do DECOM da FEE/UNICAMP, Dalton Arantes, Amauri Lopes e João Yabu-uti pela receptividade e apoio dados durante o meu primeiro ano na UNICAMP.

Ao Departamento de Radiologia do Hospital das Clínicas da UNICAMP pelo auxílio na aquisição das imagens do tomógrafo GE9800.

Ao *Medical Image Processing Group, Dept of Radiology, Univ of Pennsylvania*, pelo material fornecido.

À minha mãe, Darcy, e à minha avó, Maria Verônica.

# Conteúdo

# Lista de Figuras

# Capítulo 1

# Introdução

A visualização de dados biomédicos 3D tem como principal objetivo facilitar o entendimento de estruturas complexas tridimensionais internas ao corpo humano. A manipulação e análise destes dados também são ferramentas importantes. A manipulação serve para alterar estruturas interativamente com o objetivo principal de simular procedimentos cirúrgicos [41]. Em análise, o objetivo é obter medidas quantitativas, como por exemplo, medidas de distância, área, volume, e a localização espacial de estruturas de interesse.

## 1.1 Visão Geral

A visualização volumétrica tem várias áreas de aplicação [80]. Na medicina ela auxilia no planejamento de cirurgias, em diagnósticos clínicos, diagnósticos ortopédicos, no planejamento de tratamentos por radiação, na neurologia, na educação médica, etc. Outras áreas de aplicação são a microscopia, modelagem molecular, astrofísica, geofísica, química, engenharia, etc. Em todas as aplicações, a principal meta é auxiliar os especialistas na compreensão de fenômenos complexos tendo como base, dados relacionados com parâmetros físicos, tais como, medidas de densidade, pressão, velocidade, carga eletrostática e entropia.

Pesquisadores ao longo das últimas duas décadas têm se empenhado em desenvolver técnicas com o propósito da visualização, manipulação e análise de dados biomédicos 3D. Graças ao aumento da velocidade de processamento e capacidade de memória dos computadores, e ao surgimento de novas modalidades de aquisição de dados médicos 3D;

as tomografias por raios-x (CT), por emissão de pósitron (PET) e por emissão de fóton (SPECT), e o método de aquisição de imagens por ressonância magnética (MRI); um rápido crescimento no desenvolvimento de *hardware* e *software* para implementação dessas técnicas tem sido possível. Os dados também podem ser adquiridos por ultrasom ou, no caso de estruturas biológicas, através de uma câmera acoplada a um microscópio.

Na medicina os dados volumétricos são normalmente adquiridos em forma de imagens de fatias paralelas e uniformemente espaçadas, representando cortes transversais ao eixo longitudinal do paciente (figura 2.3). O empilhamento destas imagens, mantendo o espaçamento original entre elas, pode ser idealizado de forma que cada pixel (*picture element*) represente o volume de um pequeno cubóide (paralelepípedo) no espaço, denominado voxel (*volume element*). O conjunto de voxels forma uma representação digital de uma região cubóide em estudo no paciente e é denominado espaço voxel (figura 2.4), onde cada voxel tem normalmente associado um número inteiro proporcional ao tom de cinza do pixel na imagem correspondente. Cada um desses valores representa a integração de uma propriedade física que está sendo mensurada no interior do volume associado ao voxel.

O espaço voxel contém os valores amostrados de vários órgãos, mas não apresenta nenhuma manifestação visível destes órgãos (figura 2.6). Além disto, o espaçamento entre as amostras normalmente representa uma subamostragem dos órgãos. Técnicas de visualização, portanto, procuram identificar órgãos (ou fenômenos), reconstruir a estrutura tridimensional e visualizá-los de quaisquer direções. Para tanto é criada uma representação intermediária para estes órgãos (ou fenômenos). Os principais métodos de visualização são classificados neste trabalho de acordo com o tipo de representação intermediária (pontos, linhas, superfícies, cubos, gels, etc.) [59]. Estes métodos são divididos em técnicas baseadas em superfície, técnicas baseadas em voxel binário e técnicas de *rendering* de volume semitransparente. As técnicas baseadas em superfície procuram estimar o formato da superfície de interesse utilizando um conjunto de primitivas geométricas, que podem ser polígonos (normalmente triângulos) ou superfícies curvas. As técnicas baseadas em voxel binário representam o objeto por um conjunto de cubos opacos (voxels-1), ou sua superfície pelos polígonos que definem estes cubos. As técnicas de *rendering* de volume semitransparente associam uma cor e uma opacidade parcial a cada voxel dando à imagem final a aparência de um gel colorido e semitransparente.

Infelizmente nenhuma das técnicas desenvolvidas até hoje apresenta um resultado completamente satisfatório [27]. As técnicas baseadas em superfície levam a vantagem que as primitivas geométricas são compactas (tornando econômico o armazenamento e a transmissão dos dados) e possuem um alto grau de coerência espacial (tornando eficiente as operações de *rendering*). Entretanto, estas técnicas exigem uma classificação binária dos voxels; a superfície passa ou não passa através do voxel em questão; podendo gerar superfícies inexistentes no objeto real (falsas positivas) e/ou esconder superfícies reais (falsas negativas) na imagem final. Além disto, a reconstrução automática de uma superfície genérica raramente tem sucesso. As técnicas baseadas em voxel binário são relativamente fáceis de serem implementadas e as estruturas de dados utilizadas são simples de serem manipuladas, porém padecem dos mesmos problemas causados pela necessidade de classificação binária. As técnicas de *rendering* de volume semitransparente oferecem uma importante vantagem sobre as técnicas baseadas em superfície e em voxel binário substituindo a classificação binária por uma classificação nebulosa [82]. Entretanto, o ajuste de parâmetros do processo para obter uma boa imagem não é trivial. Além disto, estas técnicas normalmente necessitam ter toda a massa de dados na memória e o tempo para *rendering* cresce proporcionalmente ao número de voxels, o que faz com que sejam consideradas técnicas computacionalmente caras (imagens típicas usando estações de trabalho SUN *Sparc 2* necessitam de minutos e as vezes horas para serem geradas). Outra desvantagem é a dificuldade na extração de medidas, visto que estas técnicas não trabalham com estruturas topológicas definidas, mas com nuvens de tecidos de interesse.

Cada modalidade de aquisição fornece um tipo de informação diferente, baseada na relação entre a propriedade física mensurada e o valor numérico de cada voxel. Imagens de CT e MRI, por exemplo, fornecem melhor informação anatômica dos órgãos, enquanto imagens de PET e SPECT fornecem melhor informação funcional. Desta forma, informações extraídas de dados provenientes de diferentes modalidades de aquisição podem ser combinadas aumentando o conhecimento sobre a estrutura em questão. O espaço voxel também pode ser explorado para gerar imagens de cortes planares (ou não planares) em outras orientações (técnicas de refatiamento), que são fisicamente impossíveis de serem obtidas na maioria dos equipamentos de aquisição de dados. Técnicas de refatiamento junto as técnicas de reprojeção aditiva e radiográfica, projeção de máxima intensidade e o uso de pseudo-cores, são formas diretas de exploração do espaço voxel. Estas técnicas são

classificadas como métodos interativos [80].

## 1.2 Objetivos

A visualização de volumes é uma área de ponta no mundo inteiro. No Brasil ainda são muito poucos os trabalhos voltados a esta área. Entre eles podemos citar a tese de doutorado do Prof. Dr. Lincoln Assis de Moura [47] e alguns artigos recentemente publicados [3] [68] [67] [5] [29]. Sua importância e a escassez de pesquisas na área médica justificam a necessidade de um trabalho que forneça uma visão geral do assunto, abrindo caminhos para novos pesquisadores no Brasil. Baseado nestes fatos, os objetivos principais deste trabalho foram a confecção de um tutorial sobre as diversas técnicas de visualização de dados biomédicos 3D e a implementação em programa aberto de um conjunto mínimo de ferramentas de visualização volumétrica. Este trabalho faz parte das atividades do Grupo de Computação de Imagens do Depto. de Eng. da Computação e Automação Industrial da Faculdade de Eng. Elétrica da UNICAMP. O grupo tem como tema principal de pesquisa a Visualização, Modelagem, Manipulação e Análise de Dados Multidimensionais.

## 1.3 Materiais

Esta dissertação foi desenvolvida com base nas seguintes publicações: o tutorial do Udupa [39], o tutorial do SIGGRAPH'90 [59], o *survey* do Stytz et al. [62] e uma lista com mais de oitenta referências. O ambiente Khoros [35] de visualização de imagens [1] foi escolhido como plataforma de trabalho para a implementação de algumas técnicas de visualização. Os dados utilizados na implementação destas técnicas foram obtidos no Departamento de Radiologia da Faculdade de Ciências Médicas da UNICAMP e no *MIPG - Dept of Radiology, University of Pennsylvania, Philadelphia*. Estes dados são de tomógrafos de raios-x (CT9800) e equipamentos de ressonância magnética (MRI), embora os programas possam processar dados de quaisquer outras modalidades de aquisição. Os programas foram

[1] O Khoros é um pacote aberto de visualização de dados e processamento de sinais desenvolvido na Universidade do Novo México, EUA. Este programa possui como parte central um ambiente de programação visual que permite a interconexão de módulos de processamento e de controle. Assim é possível construir estruturas complexas de processamento a partir de uma coleção de blocos interligados.

testados em estações de trabalho SUN (*Sparc stations* 1+, 2 e 370). Este trabalho também dispôs dos *softwares* de visualização SOFTVU [76], desenvolvido no *MIPG, University of Pennsylvania*, e ANALYZE [71], desenvolvido na clínica Mayo, USA.

## 1.4 Estrutura do Trabalho

Inicialmente são descritos os conceitos básicos da visualização volumétrica (capítulo 2). Estes conceitos envolvem a terminologia da área, os espaços de trabalho, as técnicas de processamento de imagens e de computação gráfica utilizadas na visualização volumétrica, e uma visão detalhada do fluxo de dados e operações para obter a imagem final (figura 2.5). Em seguida são apresentados os principais métodos de visualização e os métodos interativos (capítulo 3). Este capítulo procura mostrar como os métodos fundamentais e interativos se encaixam no fluxo de dados da figura 2.5. Posteriormente são descritos os detalhes de implementação (capítulo 4). Este capítulo envolve os critérios de escolha do ambiente Khoros, uma visão geral da programação neste ambiente e a descrição das novas rotinas criadas neste trabalho, dando ênfase a como elas podem ser agrupadas na formação das técnicas de visualização. Finalmente são descritos os resultados obtidos e algumas sugestões para a continuação deste trabalho (capítulo 5), apresentando uma visão geral da importância das áreas de manipulação e análise nas aplicações clínicas, com ênfase ao estudo do problema de registro de dados 3D no planejamento de cirurgias e tratamentos.

# Capítulo 2

# Conceitos Básicos

## 2.1 Introdução

O objetivo deste capítulo é familiarizar o leitor com a terminologia utilizada na área de visualização de volumes e com as técnicas de processamento de imagens e de computação gráfica aplicadas a esta área, dando uma visão geral das etapas envolvidas no processo para visualização de estruturas 3D.

Inicialmente são descritos os conceitos relacionados com imagem digital (seção 2.2), como as imagens médicas podem ser adquiridas para a reconstrução 3D (seção 2.3) e os espaços de trabalho (seção 2.4). Em seguida, tem-se a definição do problema e suas dificuldades (seção 2.5). Esta seção também propõe um fluxo de dados genérico para a visualização volumétrica. As operações de processamento de imagens e de computação gráfica são agrupadas em blocos formando várias etapas no processo de visualização. Apenas as operações básicas do fluxo, que permitem a compreensão do problema, são descritas neste capítulo, deixando para o capítulo 3 uma descrição mais detalhada dos conceitos e operações diretamente relacionados com a classificação dos métodos fundamentais de visualização.

O fluxo inicia preparando os dados de entrada de modo a facilitar o processamento posterior (seção 2.6). Os dados pré-processados possuem dois caminhos alternativos: classificação, ou extração e modelagem (seção 2.7). Por serem mais gerais, são descritas as operações relacionadas com a extração e modelagem. Por fim, são descritas as operações

para a geração da imagem (seção 2.8), que será visualizada na tela de um monitor de vídeo (ou impressora).

## 2.2 Imagem Digital

Uma imagem digital é vista como uma matriz $m \times n$ onde os índices de linha e coluna identificam um ponto na imagem e o valor numérico do elemento correspondente na matriz está associado a alguma grandeza física medida neste ponto. A grandeza física depende do dispositivo que gera a imagem digital. Os elementos da matriz são chamados pixels, ou pels, representando a abreviação de *picture elements* [72].

Uma imagem contínua monocromática, por exemplo, é vista como uma função de intensidade de luz bi-dimensional $f(x, y)$, onde $x$ e $y$ representam coordenadas espaciais e o valor de $f$ para qualquer ponto $(x, y)$ é proporcional ao brilho (ou nível de cinza) da imagem neste ponto. A imagem digital correspondente pode ser gerada por uma câmera que discretiza ambas coordenadas espaciais e brilho da imagem $f(x, y)$. Neste caso a grandeza física medida pela câmera em cada ponto discretizado da imagem é o brilho (figura 2.1). O pixel é representado por um quadrado[1] com dimensões $\Delta x = \Delta y = p$. O sistema de coordenadas permite indexar os pixels na imagem pelos inteiros $(i, j)$, $i = 1, 2, ..., n$ e $j = 1, 2, ..., m$, onde os números representam respectivamente os índices $u$ e $v$ do pixel.

A resolução espacial de uma imagem digital é representada pelo número de linhas *versus* o número de colunas da matriz ($m \times n$ pixels). A profundidade de uma imagem digital corresponde a quantidade de bits utilizada para quantizar os valores dos pixels. O espaço de memória ocupado por uma imagem digital depende de ambas resolução e profundidade, da seguinte forma: uma imagem com resolução $200 \times 200$ pixels e profundidade 8 bits ocupa $200 \times 200 \times 8 = 40$Kbytes de memória.

[1] Considerando que normalmente os dispositivos de saída usam pixels igualmente espaçados nas direções dos eixos horizontal e vertical, para evitar distorções na visualização das imagens, adotam-se ao longo deste trabalho pixels quadrados.

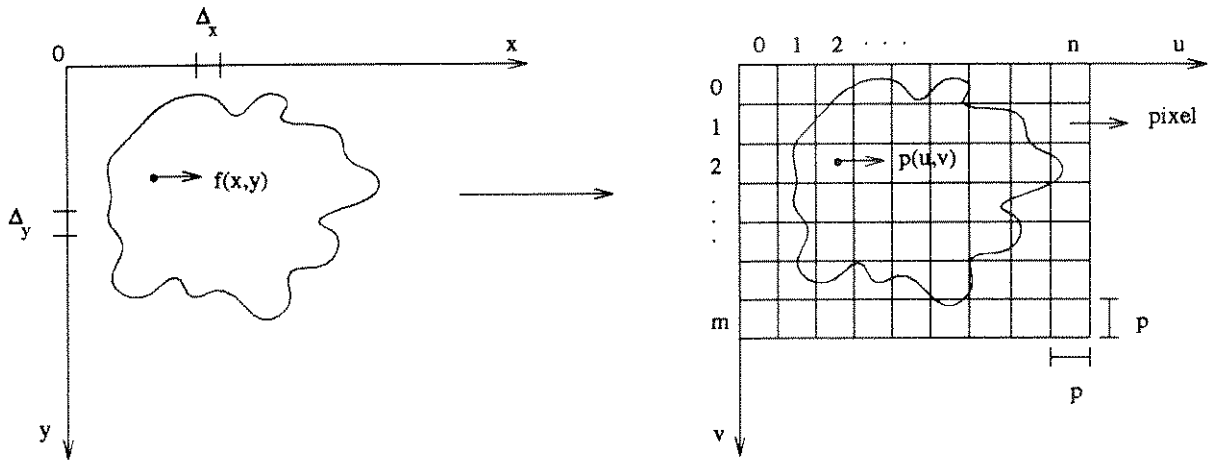


Figura 2.1: Formação de uma imagem digital.

A visualização de uma imagem digital na tela de um monitor de vídeo, por exemplo, é realizada pela conversão do valor numérico de cada pixel em luz. O tipo de luz emitida define uma classificação para a imagem:

**Imagem Monocromática** O valor numérico de cada pixel representa uma grandeza escalar associada ao brilho do ponto correspondente. Este tipo de imagem também é conhecido por imagem em tons de cinza. Quando os pixels só possuem dois valores distintos, é conhecido por imagem binária (normalmente é uma imagem preto e branco).

**Imagem Colorida** O valor numérico de cada pixel representa uma grandeza vetorial associada a cor do ponto correspondente. Uma cor pode ser caracterizada pela combinação de três componentes; matiz, luminância e saturação [38]. A matiz é utilizada para diferenciar as cores, tais como, vermelho, verde, amarelo, etc. A luminância envolve a noção acromática de intensidade de luz, que corresponde ao brilho da imagem monocromática. A saturação refere-se a quanto a cor está diluída pela luz branca. É o que diferencia o vermelho do rosa por exemplo. Entretanto, boa parte das cores percebidas pelo olho humano podem ser descritas como combinação de três cores primárias; vermelho, verde e azul; representando o modelo RGB de cores. Este modelo é normalmente utilizado nos monitores de vídeo e, portanto, é também adotado na descrição das técnicas de visualização que usam cores.

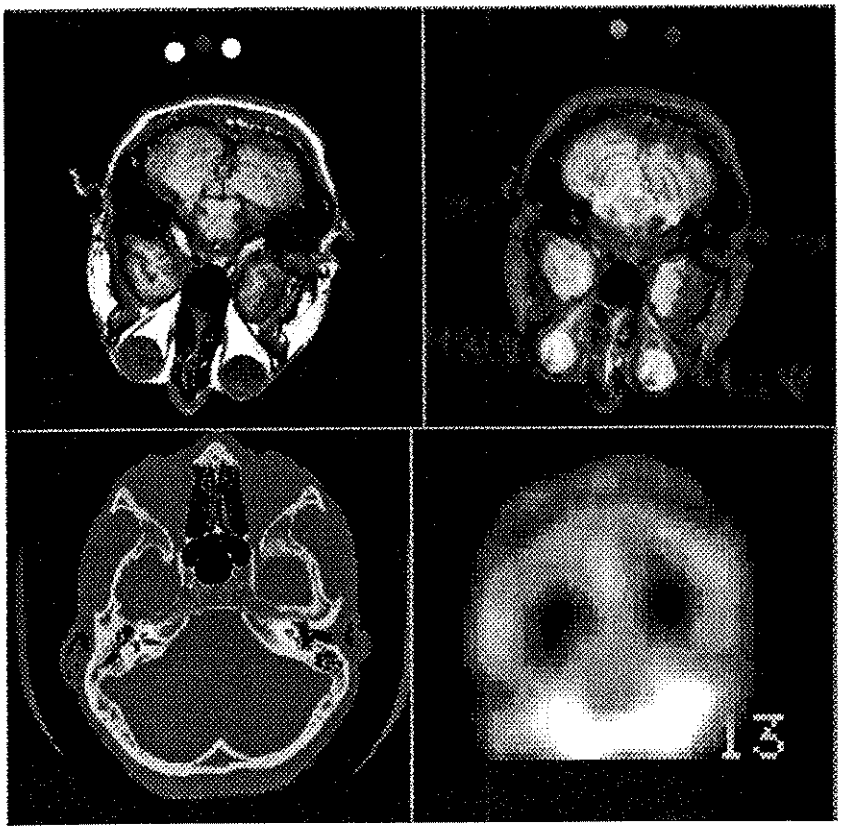

Figura 2.2: Fatias da cabeça: MRI com parâmetro T1 no canto superior esquerdo, MRI com parâmetro T2 no canto superior direito, CT no canto inferior esquerdo e SPECT no canto inferior direito.

## 2.3 Aquisição de Dados

As formas de aquisição de dados que desencadearam maior evolução nas técnicas de reconstrução 3D de imagens médicas foram a tomografia por raios-x (CT), a tomografia por emissão de pósitron (PET), a tomografia por emissão de fóton (SPECT) e a captura de imagens por ressonância magnética (MRI). As modalidades CT e MRI fornecem imagens de alta resolução (normalmente 512 × 512 pixels) e contribuem com a informação anatômica das estruturas orgânicas contidas nas imagens, sendo a modalidade CT melhor para tecidos duros e a modalidade MRI melhor para tecidos moles. As modalidades PET e SPECT fornecem imagens de baixa resolução (normalmente 64 × 64 pixels) e contribuem

com a informação funcional das estruturas contidas nas imagens. Uma grande vantagem da modalidade MRI, sobre as demais, é não utilizar radiação. Os dados também podem ser adquiridos através de ultrasom ou, no caso de estruturas histológicas, através de uma câmera acoplada a um microscópio. Quando a estrutura é opaca, a câmera obtém imagens de cortes previamente realizados sobre a estrutura, quando é transparente, as imagens são obtidas ajustando o foco do microscópio.

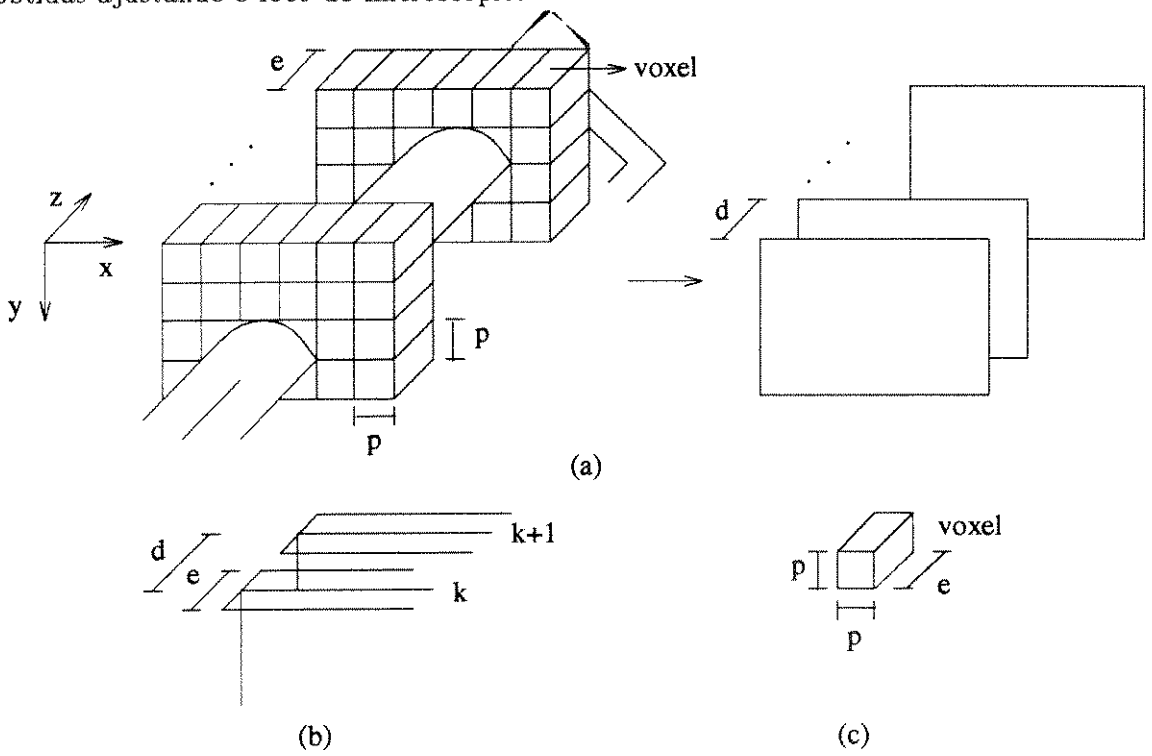


Figura 2.3: Tomografia computadorizada.

Os dados volumétricos são normalmente adquiridos em forma de imagens de fatias paralelas e uniformemente espaçadas, representando cortes transversais ao eixo longitudinal do paciente [2]. A tomografia de raios-x é um exemplo bastante didático no processo de aquisição de dados. A figura 2.3a ilustra como as imagens são geradas. Cada imagem está associada a uma localização $k$, $k = 1, 2, \cdots, l$, no eixo $z$ e a uma espessura $\Delta z = e$ em torno desta localização[3], formando o cubóide (caixa de faces retangulares) ilustrado na figura 2.3b.

[2] É comum a utilização de espaçamento variável entre os cortes quando existem regiões de maior interesse. Nestas regiões são feitos cortes mais próximos permitindo uma melhor visualização de detalhes.

[3] A espessura dos cortes pode variar de 1mm a 15mm e para estudos de alta resolução são tipicamente de 2mm [9].

A figura 2.3b também mostra o espaçamento $d$[4] entre os cortes. Este espaçamento é normalmente cerca de 2 a 15 vezes maior que a dimensão $p$ (normalmente $0mm < p \leq 1mm$) dos pixels [39]. O cubóide é subdividido em pequenos outros chamados voxels. O voxel representa uma abreviação para *volume element*. Cada pixel das imagens geradas está associado a um voxel. O voxel com dimensões $\Delta z = e$, $\Delta x = \Delta y = p$ é ilustrado na figura 2.3c. O valor numérico de cada pixel representa tipicamente a média das atenuações de raios-x no volume interno do corpo correspondente ao voxel [9]. Os valores destas atenuações são expressos na escala *Hounsfield*[5]. Estes valores são obtidos pela exposição do corpo ao bombardeamento de raios-x de várias direções, formando várias projeções de valores atenuados. Uma projeção em uma dada direção é obtida com o movimento da fonte de raios-x e um conjunto de detetores. A média das atenuações sofridas pelos raios-x corresponde a um valor medido no centro do voxel. Este cálculo é feito considerando todos os raios que passam através do voxel e usando o método *filtered back-projection* sobre os valores das projeções.

O valor associado a cada voxel é um número inteiro, proporcional ao tom de cinza do pixel na imagem correspondente, e representa a integração de alguma propriedade física que está sendo mensurada no interior do volume associado ao voxel. No caso da tomografia por raios-x, por exemplo, a grandeza física medida é a densidade do tecido. O conhecimento de como as propriedades medidas variam de um tecido para outro é crucial na determinação de uma estrutura de interesse. No caso da tomografia de raios-x, quanto maior a densidade do tecido maior serão as atenuações e, portanto, maior serão os valores dos pixels nas imagens dos cortes referentes a este tecido. A figura 2.2 mostra fatias da cabeça obtidas por MRI, CT e SPECT. A modalidade MRI mede a mobilidade do núcleo de hidrogênio. Isto é feito aplicando um campo magnético sobre a região em estudo e medindo o tempo de relaxação dos *spins* (parâmetros $T_1$ e $T_2$). Na modalidade SPECT, uma substância radioativa é injetada no paciente antes do exame e depois é feita uma medida da radiação *gamma* emitida pela região em estudo. Pode-se constatar que os tecidos moles são mais evidentes nas imagens de MRI e que o tecido ósseo é mais evidente na imagem de CT. Na imagem de SPECT observa-se a atividade metabólica do cérebro.

[4] Normalmente $d \geq e$ e em casos de fatiamento físico (digitalização de cortes de seções) $e = 0$.

[5] A escala *Hounsfield* é uma homenagem a G.Hounsfield, que ganhou o prêmio Nobel em 1979 por inventar o tomógrafo por raios-x.

## 2.4 Espaços de Trabalho

Nesta seção são definidos os espaços de trabalho em que são implementadas as técnicas de visualização volumétrica.

**Espaço Voxel** Empilhando as imagens das fatias, distanciadas de $d$, ao longo do eixo $z$ e paralelas ao plano $xy$, o espaço voxel pode ser idealizado como o cubóide que envolve toda a região rastreada pelo *scanner* (figura 2.4a). O cubóide é subdividido em outros pequenos cubóides denominados voxels. Cada voxel representa um volume $p \times p \times d$ formado ao se tomar uma distância $d/2$ em torno do pixel correspondente nas direções dos semi-eixos z+ e z-. O sistema de coordenadas permite indexar os voxels pelos três inteiros $(i, j, k)$, $i = 1, 2, \cdots, n$, $j = 1, 2, \cdots, m$ e $k = 1, 2, \cdots, l$, onde os números representam respectivamente os índices $x$, $y$ e $z$ do voxel e a resolução do espaço é $m \times n \times l$ voxels. O subespaço definido pelo conjunto de todos os voxels com coordenadas $(i, j, k)$ onde $i = 1, 2, ..., n$, $j = 1, 2, ..., m$ e $k$ é um valor inteiro fixo $r$ é chamado a $r$-ésima fatia (ou imagem) do espaço voxel. A figura 2.6 ilustra o espaço voxel formado pelo empilhamento de fatias de um joelho.

**Espaço Objeto** O objeto (ou estrutura) 3D de interesse na visualização está contido no espaço voxel. Um objeto de interesse pode ser parte de um orgão ou um orgão. O espaço objeto é, portanto, representado pela estrutura de dados que modela este objeto. Esta estrutura pode ser, por exemplo, um espaço voxel binário; onde a cada voxel é associado o valor 1, quando o voxel pertence ao objeto, ou o valor 0 no caso contrário; ou uma lista de triângulos representando a superfície do objeto.

**Espaço Imagem** O espaço imagem é representado por uma área retangular, no plano uv (plano de visão), na qual o objeto é projetado (figura 2.20). Esta área é subdividida em pequenos quadrados $p \times p$ denominados pixels. O sistema de coordenadas permite indexar os pixels pelos dois inteiros $(u, v)$, $u = 1, 2, \cdots, N$ e $v = 1, 2, \cdots, M$, onde os números representam respectivamente os índices $u$ e $v$ do pixel e a resolução do espaço é $M \times N$ pixels. Exemplos típicos de dados neste espaço associam ao valor dos pixels a distância do plano de visão à superfície do objeto (*z-buffer*), ou a intensidade de luz da imagem final.

Um detalhe importante é a redefinição de voxel: a dimensão $z$ do voxel depende apenas do espaçamento $d$ entre os cortes, sendo independente da espessura $e$ dos cortes (figura 2.4b). O valor numérico associado ao voxel é chamado densidade do voxel.

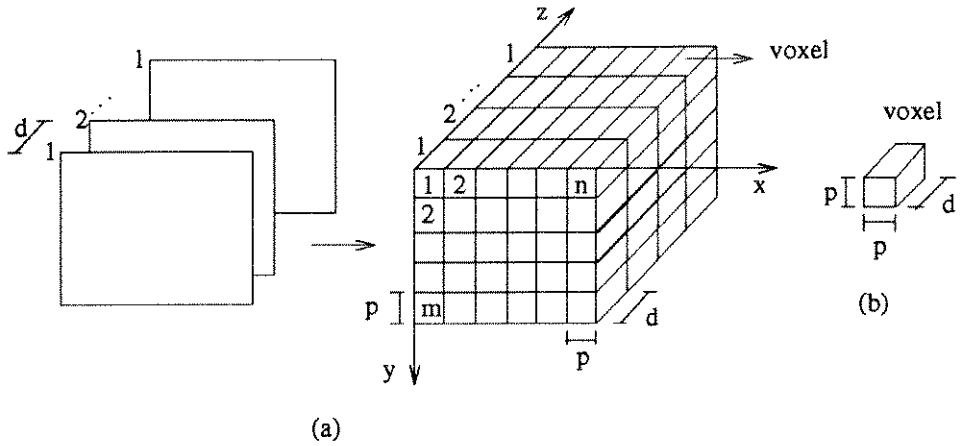


Figura 2.4: Formação do espaço voxel.

## 2.5 Definição do Problema

Após a aquisição dos dados pelos equipamentos de tomografia, as imagens são normalmente armazenadas em um arquivo linha por linha, da primeira fatia para a última: (1,1,1), (2,1,1), $\cdots$, (n,1,1), (1,2,1), (2,2,1), $\cdots$, (n,2,1), $\cdots$, (n,m,1), (1,1,2), (2,1,2), $\cdots$, (n,m,l). No cabeçalho do arquivo estão contidas informações sobre o exame, tais como, espessura das fatias, distância entre elas, dimensões dos pixels, resolução das fatias e número de fatias. Estas informações são essenciais no processamento dos dados. Este arquivo representa o espaço voxel original. A distribuição de densidade dos voxels deste espaço contém a informação anatômica (ou funcional) de estruturas 3D internas ao corpo humano. O problema tratado neste trabalho consiste em como extrair e visualizar esta informação.

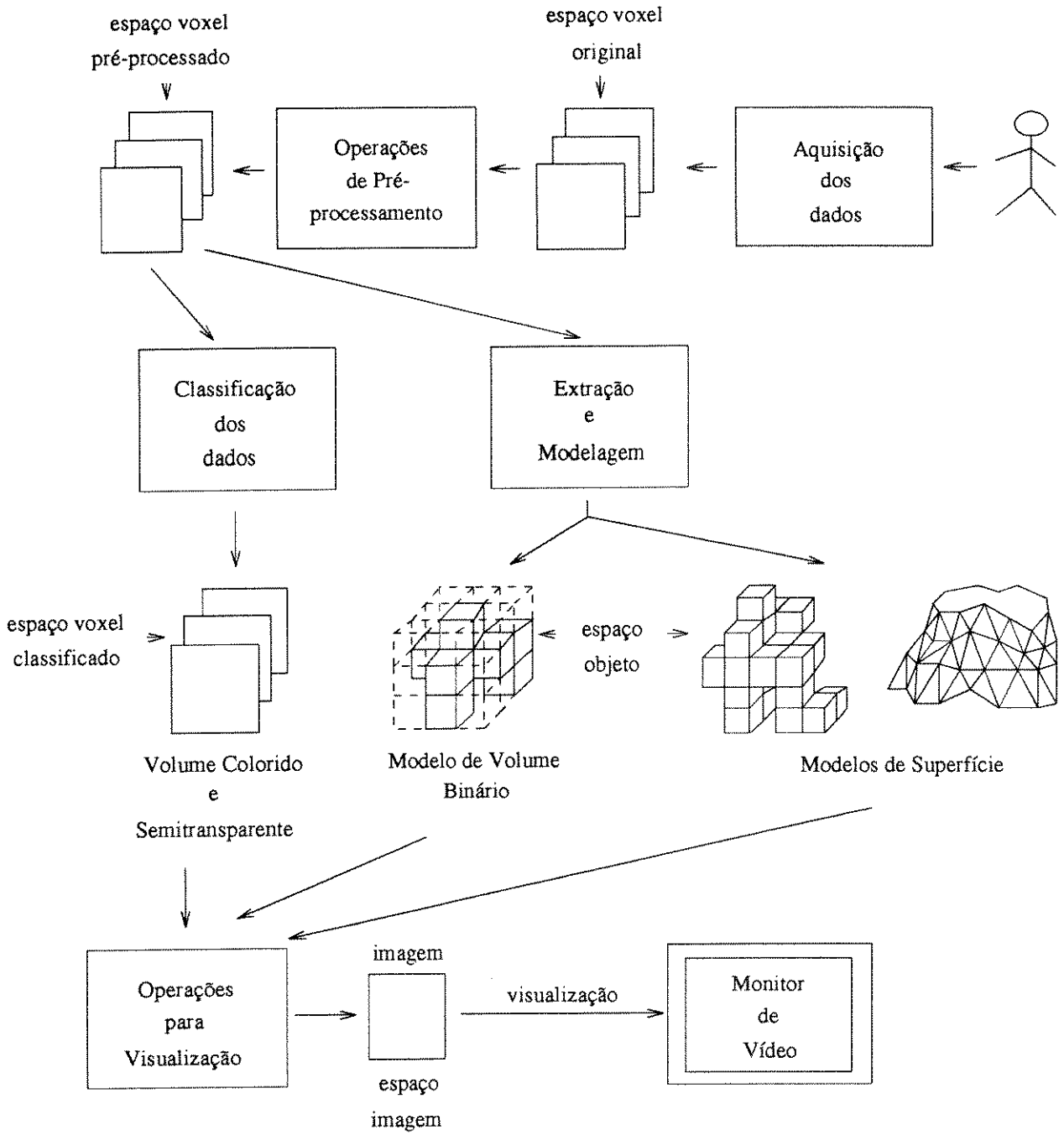


Figura 2.5: Fluxo de dados.

As imagens das fatias (figura 2.2) mostram que estruturas diferentes podem aparecer com pouco contraste nas bordas, dificultando a identificação dos voxels perten-

centes a uma estrutura de interesse. Quando "não há transparência", as estruturas de interesse contidas no interior do espaço voxel não podem ser visualizadas (ver figura 2.6 e figura 2.7). Uma amostragem anisotrópica ocorre quando o espaçamento entre as fatias faz com que os pontos, onde foram calculadas as amostras (centro geométrico dos voxels), não estejam igualmente espaçados nas direções dos eixos principais (x,y e z). Esta anisotropia deve ser levada em conta antes da geração da imagem final, para não provocar distorções na visualização da imagem, devido ao espaçamento desigual entre as amostras que serão projetadas no plano de visão (figura 2.20) nas direções dos eixos u e v. Por fim, o arquivo de dados normalmente ocupa muita memória: um exemplo típico é um arquivo de saída de um CT com 60 imagens de $512 \times 512 \times 16$ bits, que equivale a aproximadamente 32 Mbytes de memória. Neste caso, este arquivo pode ser reduzido para aproximadamente 8 Mbytes convertendo as imagens para $256 \times 256 \times 16$ bits ou, dependendo da disponibilidade de memória, o arquivo pode ser reduzido para aproximadamente 4 Mbytes convertendo as imagens para $256 \times 256 \times 8$ bits. Mesmo assim, 4 Mbytes de memória ainda é relevante para alguns computadores.

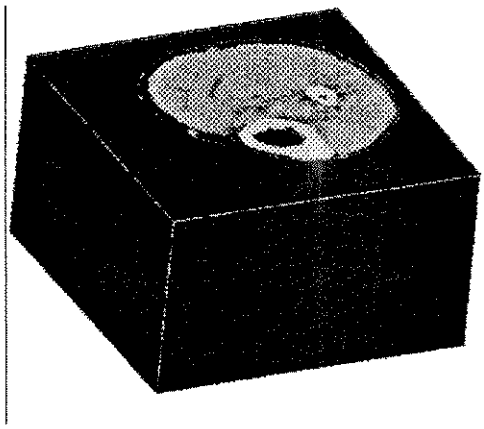

Figura 2.6: Espaço voxel resultante do empilhamento de fatias de um joelho.

O diagrama de blocos da figura 2.5 esquematiza um fluxo de dados genérico com as principais etapas encontradas nas técnicas de visualização volumétrica. Cada etapa é representada por um conjunto de operações de processamento de imagens e computação gráfica, seguindo uma ordem didática de explicação. No entanto, é possível omitir, acrescentar e inverter a ordem das operações de uma etapa, ou entre etapas, implementando várias soluções para o problema. O primeiro bloco apresenta as operações de pré-processamento.

Estas operações podem ser utilizadas, por exemplo, para fazer o alinhamento entre fatias, quando este é perdido no processo de aquisição; reduzir as dimensões do espaço voxel, economizando memória; tratar as imagens das fatias, facilitando a identificação das estruturas de interesse; e tornar a amostragem dos dados isotrópica (igualmente espaçadas nas direções x, y, e z), através da interpolação de outras amostras gerando um novo espaço com voxels cúbicos.

A visualização de estruturas de interesse pode ser feita por dois caminhos no fluxo. Um caminho procura identificar os voxels do espaço pré-processado que pertencem a uma dada estrutura e extrair esta informação, criando um modelo de superfície, ou volume, para a estrutura. O outro caminho está interessado em visualizar todo o espaço voxel como se fosse translúcido. Neste caso, não existem estruturas topológicas explicitamente definidas, mas regiões de tecidos que podem aparecer com maior, ou menor, ênfase na imagem final. A ênfase é dada através da classificação dos dados que associa níveis de transparência aos voxels. Uma cor pode ser associada aos voxels dando à imagem final a aparência de um gel colorido e semitransparente. Devido as operações de classificação dos dados estarem diretamente relacionadas com uma das metodologias fundamentais, a descrição destas operações é feita no capítulo 3.

Após a modelagem das estruturas de interesse, o próximo bloco da figura 2.5 contém as operações para visualização. Estas operações consistem da idealização de um ambiente de visualização com seus elementos básicos (observador, fontes de luz, objetos, etc.), onde são feitas transformações geométricas e o *rendering* (projeção e tonalização) dos modelos para gerar imagens que representem a visualização tridimensional do objeto de interesse.

## 2.6 Operações de Pré-processamento

As operações de pré-processamento podem ser consideradas transformações do espaço voxel no espaço voxel. O objetivo do pré-processamento é tornar o espaço voxel mais adequado, facilitando a implementação da técnica de visualização volumétrica desejada. Estas operações envolvem registro de imagens, volume de interesse, filtragem e interpolação.

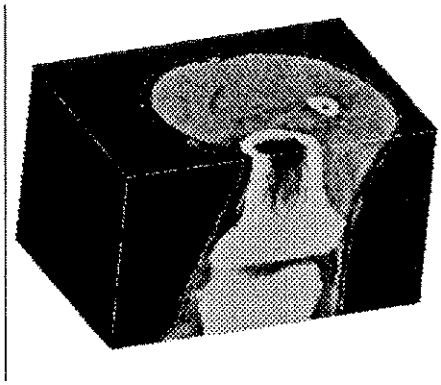

Figura 2.7: Visualização do interior do espaço voxel através de um corte.

### 2.6.1 Registro de Imagens

Existem algumas circunstâncias nas quais o alinhamento físico entre as imagens das fatias é perdido no processo de aquisição [47]. Um caso típico é o fatiamento serial em patologia. Nesta aplicação, o bloco contendo o material biológico é fisicamente fatiado, produzindo uma série de seções transversais. Antes de qualquer operação é interessante, portanto, devolver o alinhamento físico entre as fatias do espaço voxel. Esta operação é conhecida por registro de imagens, que consiste em estimar os parâmetros das operações de translação e rotação de imagens de modo que elas fiquem alinhadas.

Dependendo do material biológico é possível criar marcas de referência antes do fatiamento. As marcas de referência definem um sistema de eixos utilizado no processo. Infelizmente para estruturas pequenas esta técnica não é confiável. Outra forma é dispor de um especialista que localize marcas anatômicas conhecidas nas imagens das fatias, mas este processo, além de exigir um especialista, é lento. Entretanto, quando o espaçamento entre as fatias é pequeno o suficiente para que duas imagens consecutivas sejam consideradas quase iguais, o alinhamento automático pode ser idealizado. O conceito básico assume que duas imagens estão alinhadas, quando uma certa função de similaridade entre elas é máxima. Portanto, técnicas de registro podem ser idealizadas usando o conceito de técnicas de casamento. Exemplos de funções de similaridade são o erro quadrático médio, a autocorrelação e o coeficiente de correlação.

Algumas técnicas de registro são implementadas após a segmentação

(seção 2.7.1), fatia por fatia, da estrutura de interesse. Estas técnicas podem explorar a coordenada do centro de massa do objeto em cada fatia para efetuar a translação, fazendo com que o centro de massa de duas fatias consecutivas coincidam em um mesmo eixo imaginário, e a transformada de *Hotelling* [72] para fazer a rotação das fatias. Em alguns casos é preferível o alinhamento entre os contornos externos das estruturas segmentadas em cada fatia (seção 2.7.4). Pedrini et al. [29] utiliza o centro de massa dos contornos externos e marcas de referência antes do fatiamento para fazer o registro, mostrando que, dependendo da estrutura, fazer com que o centro de massa de todos os contornos coincidam com um eixo imaginário, não afeta muito a precisão no processo de reconstrução 3D.

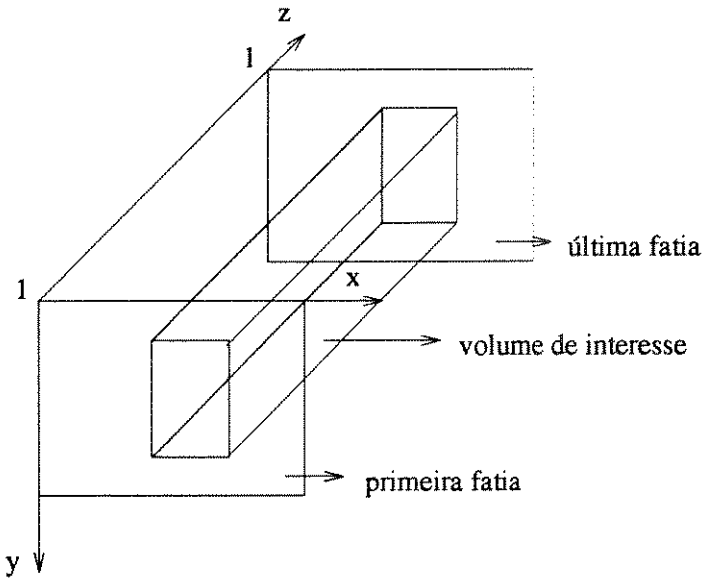


Figura 2.8: Volume de interesse.

### 2.6.2 Volume de Interesse (VOI)

A operação volume de interesse também é conhecida por subregionamento [23], ou região de interesse (ROI), por visar a extração de uma subregião do espaço voxel, que consiste do menor cubóide que contém as estruturas de interesse (figura 2.8). Esta operação gera um novo espaço voxel formado por um subconjunto de voxels do espaço original.

A principal proposta da operação VOI é minimizar o espaço de memória [39]. Um exemplo de como este espaço pode ser bastante representativo é supor o espaço original

com $256 \times 256 \times 60$ voxels, o que significa 4 milhões de voxels ou 4Mbytes de memória, caso as densidades dos voxels sejam quantizadas em 8 bits. Sendo $50 \times 70 \times 60$ voxels a subregião que contém a estrutura de interesse, o número de voxels reduz para 210 mil, o que significa uma redução para 210kbytes de memória, mais de dezenove vezes menos memória. Isto implica uma redução substancial do tempo para processamento dos dados e menor acesso à memória auxiliar. Uma outra razão para esta operação pode ser a exclusão de parte de um orgão para que o espaço resultante revele o interior deste orgão.

### 2.6.3 Filtragem

A idéia básica da filtragem é adequar a imagem ao problema pelo tratamento de componentes da função a ser filtrada. O domínio da frequência para uma dada função está associado ao domínio do espaço pela Transformada de *Fourier*. Variações locais abruptas de densidade no domínio do espaço correspondem a componentes em alta frequência, assim como variações suaves correspondem a componentes em baixa frequência. A função de densidades da imagem e o filtro podem ser expressos em ambos domínios e, portanto, a operação de filtragem pode ser feita no domínio da frequência ou no domínio do espaço dependendo da facilidade de implementação. Aqui a ênfase é dada ao domínio do espaço.

As principais operações de filtragem na visualização de volumes são : suavização e realce, sendo que normalmente deseja-se uma suavização de imagem que preserve as características das estruturas do objeto de interesse [39]. Diferente da operação VOI, o número de voxels após a filtragem é o mesmo de antes, porém com densidades diferentes. A filtragem associa um valor de densidade a um voxel $v$ do espaço de saída baseada nas densidades dos voxels em uma pequena vizinhança de $v$ no espaço de entrada. As vizinhanças comumente utilizadas nas operações de filtragem são ilustradas na figura 2.9. O voxel $v$ é em todos os três casos o voxel central $v_0$. A filtragem 2D utilizando os oito mais próximos voxels vizinhos de $v_0$ em uma dada fatia (figura 2.9a) não é afetada pelas fatias adjacentes. Já as filtragens 3D com base nas vizinhanças definidas nas figuras 2.9b e 2.9c utilizam, respectivamente, os seis voxels vizinhos por uma face adjacente e os vinte e seis voxels vizinhos que formam um cubo em torno de $v_0$.

### 2.6.3.1 Suavização

A suavização também é conhecida como filtragem passa-baixas frequências, pois as variações locais de densidade são suavizadas. A idéia principal da suavização é suprimir ruído. A nova densidade $d_s(v)$ de cada voxel $v = v_0$ é a média ponderada das densidades de seus vizinhos, incluindo a sua, no espaço de entrada segundo a equação geral abaixo:

$$d_s(v) = \frac{\sum_{j=0}^{n} p_j d_e(v_j)}{\sum_{j=0}^{n} p_j} \tag{2.1}$$

onde $d_e(v_j)$ é a densidade do voxel $v_j$ no espaço de entrada, $p_j$ é o peso (qualquer número positivo finito) associado a $v_j$ e $v_1, v_2, \cdots, v_n$ são os vizinhos de $v = v_0$. Vários tipos de filtro passa-baixas podem ser obtidos dependendo de como os pesos $p_j$ são escolhidos. Um filtro gaussiano com vizinhança de 8 voxels, por exemplo, pode ser obtido pela escolha de $p_0 = 4, p_1 = p_3 = p_5 = p_7 = 2$ e $p_2 = p_4 = p_6 = p_8 = 1$. Os pesos normalmente são escolhidos de maneira que os voxels mais próximos de $v_0$ tenham maior influência no resultado final em relação aos mais afastados.

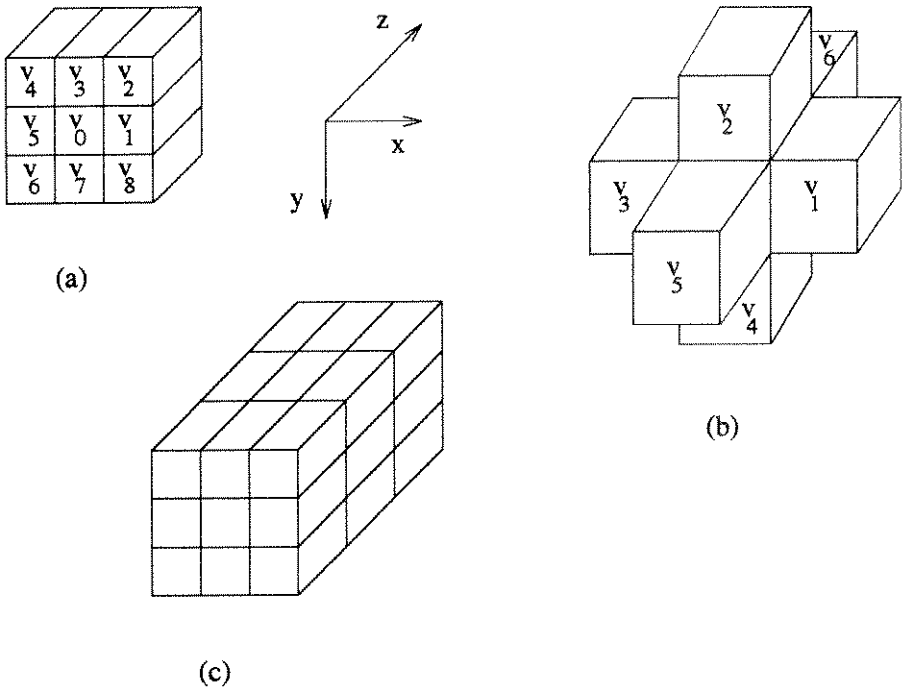


Figura 2.9: Vizinhanças.

#### 2.6.3.2 Realce

A idéia principal do realce é enfatizar as bordas das estruturas contidas no espaço voxel. As bordas normalmente apresentam variações locais de densidade relevantes. Portanto, esta operação também é conhecida como filtragem passa-altas. A ênfase nas bordas é feita calculando as diferenças de densidades entre voxels vizinhos, como pode ser observado nos dois exemplos dadas abaixo [39].

(i) Usando a vizinhança da figura 2.9a:

$$d_s(v) = \left|\frac{p_2 d_e(v_2) + p_1 d_e(v_1) + p_8 d_e(v_8)}{(p_2 + p_1 + p_8)} - \frac{p_4 d_e(v_4) + p_5 d_e(v_5) + p_6 d_e(v_6)}{(p_4 + p_5 + p_6)}\right| +$$

$$\left|\frac{p_6 d_e(v_6) + p_7 d_e(v_7) + p_8 d_e(v_8)}{(p_6 + p_7 + p_8)} - \frac{p_2 d_e(v_2) + p_3 d_e(v_3) + p_4 d_e(v_4)}{(p_2 + p_3 + p_4)}\right| \quad (2.2)$$

(ii) Usando a vizinhança da figura 2.9b:

$$d_s(v) = \frac{p_1|d_e(v_1) - d_e(v_3)| + p_2|d_e(v_4) - d_e(v_2)| + p_3|d_e(v_6) - d_e(v_5)|}{p_1 + p_2 + p_3} \quad (2.3)$$

Nestas equações $|x|$ representa o valor absoluto de $x$ e os pesos $p_i$ são quaisquer números inteiros positivos.

#### 2.6.3.3 Suavização, Realce e Preservação de Partes Selecionadas

A suavização, o realce e a preservação de partes selecionadas são utilizadas com o objetivo de reduzir o ruído, suavizando as regiões internas às estruturas, e realçar ou preservar, as bordas. Um problema encontrado na suavização é que, além de suavizar os interiores dos objetos, as bordas também são suavizadas. Já no realce, o problema é a enfatização do ruído . A união das operações de suavização e realce pode ser feita utilizando a filtragem passa-altas para identificar as descontinuidades das bordas, realçando-as ou preservando-as, e a filtragem passa-baixas para suprimir o ruído apenas nos interiores dos objetos.

O filtro mediano é um bom exemplo da suavização de interiores e preservação de bordas. A densidade de saída $d_s(v)$ é obtida neste tipo de filtro como o valor mediano

das densidades de entrada $d_e(v_0), d_e(v_1), \cdots, d_e(v_n)$. Um exemplo é descrito utilizando a vizinhança da figura 2.9b e supondo que os valores de $d_e(v_0), d_e(v_1), \cdots, d_e(v_6)$ são 70, 75, 80, 85, 90, 100 e 300. Arranjando as densidades em ordem crescente 70, 75, 80, 85, 90, 100 e 300, escolhe-se o valor mediano 85 para a densidade $d_s(v)$ do voxel $v = v_0$ no espaço de saída. Pode ser observado que a densidade $d_e(v_6) = 300$ representa um *speckle noise* (ruído pontual), que é eliminada na operação.

### 2.6.4 Interpolação

Analisando o processo de aquisição de dados (figura 2.3a), percebe-se que a relação entre $d$ e $e$ dita a qualidade da amostragem na direção $z$ e que a dimensão $p$ dos pixels dita a qualidade nas direções $x$ e $y$. A relação entre $d$ e $p$ dita o grau de anisotropia da amostragem. A interpolação tem por objetivo melhorar a qualidade da amostragem, estimando valores amostrados em uma nova escala e gerando uma amostragem isotrópica. Devido aos pixels nos dispositivos de saída terem posicionamento isotrópico, a interpolação evita possíveis distorções na visualização. Uma outra correção realizada na geração de um espaço isotrópico é quando o espaçamento $d$ entre as fatias é variável. Esta situação ocorre, por exemplo, no planejamento de cirurgias craniofaciais [39]. Quando o local da deformidade é uma região de órbitas, as estruturas ósseas são finas e de baixa densidade, sendo desejáveis, portanto, informações mais detalhadas, através de cortes mais próximos nesta região. Outra vantagem da interpolação é que medidas quantitativas, como distância, área e volume, podem ser extraídas do espaço voxel isotrópico de forma realística e correta.

Um exemplo ilustrativo da transformação ocorrida no espaço voxel na operação de interpolação é apresentado na figura 2.10. O espaço de entrada tem resolução $2 \times 2 \times 2$ voxels ($m = n = l = 2$) e as dimensões dos voxels são $\Delta x = \Delta y = p$ e $\Delta z = 2p$. Desejando-se obter voxels cúbicos com dimensões $\Delta x = \Delta y = \Delta z = p/2$, novas amostras podem ser interpoladas nas fatias 1 e 2, aumentando a resolução das fatias para $4 \times 4$ pixels, e novas fatias com resolução $4 \times 4$ pixels podem ser interpoladas entre as fatias 1 e 2. A base para a interpolação são as densidades dos oito voxels do espaço original. Neste caso, as fatias originais tiveram suas localizações no espaço preservadas. Entretanto, a idéia genérica de interpolação permite que o processo realizado nas fatias, alterando a localização e intensidade dos pixels originais, seja feito também na direção $z$, alterando a localização e

a densidade dos voxels originais. Udupa [39] apresenta um algoritmo de interpolação linear com este propósito para a direção $z$.

O exemplo acima mostra que para conseguir voxels cúbicos, com dimensões $\Delta x = \Delta y = \Delta z = p$, basta apenas interpolar amostras na direção $z$. Esta é a forma mais comum de interpolação. Entretanto, a interpolação nas direções $x$, $y$ e $z$ é a mais genérica. A interpolação também pode ser utilizada quando se amostra sob o trajeto de um raio sob um ângulo qualquer (seção 3.4.2), para gerar fatias em direções arbitrárias (seção 3.5.1), e para reduzir a resolução do espaço voxel, através de uma subamostragem, visando economia de memória. Em todos estes casos a interpolação foi aplicada no espaço voxel, no entanto, ela também pode ser aplicada no espaço objeto (seção 2.7.6).

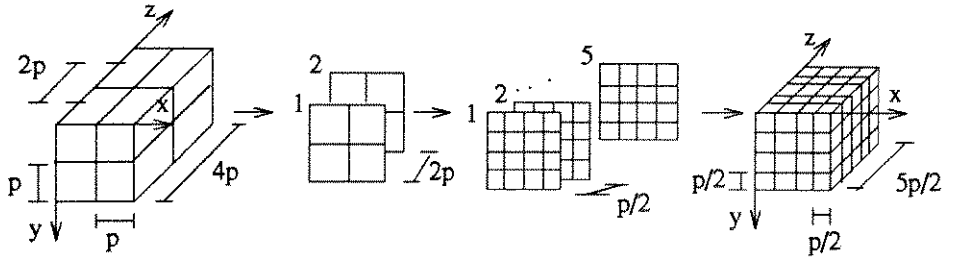


Figura 2.10: Exemplo de Interpolação.

#### 2.6.4.1 Interpolação na Direção $z$

A *nearest neighbor rule* [39] (interpolação por vizinho mais próximo) e a interpolação linear podem ser citadas entre as técnicas de interpolação na direção $z$. A figura 2.11 ilustra o processo para interpolar uma fatia $m$ entre as fatias $n$ e $n+1$ do espaço de entrada. O sistema de coordenadas foi rotacionado e as fatias foram empilhadas na vertical para facilitar a visualização do processo. Uma vertical $(i, j, k)$, $i$ fixo, $j$ fixo e $n \leq k \leq n+1$, que passa pelo voxel $v_m$ a ser interpolado, é inicialmente fixa.

Na interpolação por vizinho mais próximo, a densidade $d_s(v_m)$ do voxel $v_m$ é igual a densidade do voxel do espaço de entrada mais próximo: $d_e(v_n)$ (caso $l_i < l_s$) ou $d_e(v_{n+1})$ (caso $l_s < l_i$, onde $d_e(v_k)$ é a densidade do voxel $v_k$, $k = n, n+1$, do espaço de entrada, $l_i$ é a distância entre o voxel $v_m$ e o voxel inferior $v_n$, e $l_s$ é a distância entre o voxel $v_m$ e o voxel superior $v_{n+1}$.

Na interpolação linear assume-se que a variação de densidade é linear na direção z entre os voxels $v_n$ e $v_{n+1}$. A densidade $d_s(v_m)$ é obtida segundo a equação abaixo:

$$d_s(v_m) = d_e(v_n) + \frac{[d_e(v_{n+1}) - d_e(v_n)]l_i}{l_s + l_i} \tag{2.4}$$

onde $l_s + l_i = d$ (espaçamento entre as fatias $n$ e $n+1$). O mesmo procedimento é repetido nas outras verticais para calcular as densidades de todos os outros voxels da camada $m$.

É fácil concluir que a interpolação por vizinho mais próximo equivale a duplicação das fatias do espaço de entrada em fatias mais próximas. Já a interpolação linear é uma aproximação bem melhor que a anterior e normalmente apresenta bons resultados.

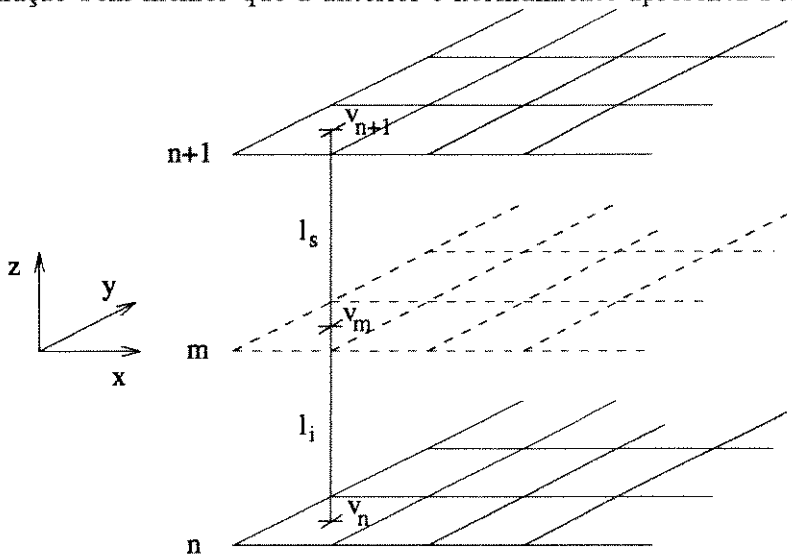


Figura 2.11: Interpolação linear.

### 2.6.4.2 Interpolação nas Direções $x$, $y$ e $z$

Um método normalmente utilizado para interpolação nas direções $x$, $y$ e $z$ é o método de interpolação trilinear. A figura 2.12a mostra o processo de interpolação de uma fatia $m$ entre as fatias $n$ e $n+1$ do espaço de entrada, e o sistema de coordenadas rotacionado. Neste caso, a densidade $d_s(v_m)$ leva em consideração as densidades dos oito voxels vizinhos mais próximos do voxel $v_m$: os voxels $v_1$, $v_2$, $v_3$ e $v_4$ na fatia inferior e $v_5$, $v_6$, $v_7$ e $v_8$ na fatia superior. A união dos centros dos voxels $v_j$, $j = 1, 2, 3, \cdots, 8$, forma o cubóide da figura 2.12b. A interpolação trilinear assume que a variação de densidade é

linear em qualquer aresta do cubóide. Portanto, utilizando as medidas de distância, $d_1$, $d_2$ e $d_3$ (figura 2.12b), o espaçamento $d$ entre as fatias $n$ e $n+1$, e a dimensão $\Delta x = \Delta y = p$ dos voxels do espaço de entrada, calcula-se a densidade $d_s(v_m)$ após sete operações similares a equação 2.4: primeiro é calculada a densidade em $p_1$ (figura 2.12b) com base nas densidades em $v_1$ e $v_2$, em $p_2$ com base nas densidades em $v_3$ e $v_4$, e depois em $p_3$ com base nas densidades em $p_1$ e $p_2$. Similarmente a densidade em $p_6$ é calculada usando mais três operações. Finalmente a densidade $d_s(v_m)$ é calculada conhecendo as densidades em $p_3$ e $p_6$. É fácil concluir que, se o voxel $m$ estiver em uma das faces do cubóide $v_1, v_2, \cdots, v_8$, apenas três operações para estimar $d_s(v_m)$ são necessárias, e se estiver em uma aresta do cubóide, apenas uma operação é necessária. O processo acima é repetido para todos os outros voxels da camada $m$.

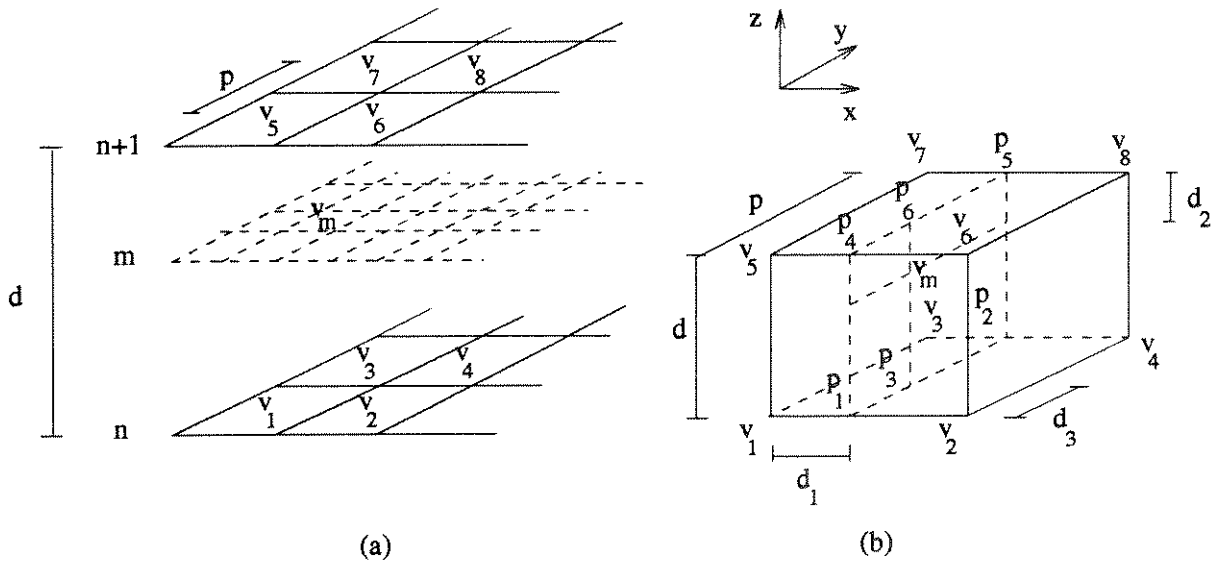


Figura 2.12: Interpolação trilinear.

A interpolação trilinear também pode utilizar como base, mais do que oito voxels vizinhos mais próximos. As interpolações linear e trilinear são operações de primeira ordem nas direções correspondentes, entretanto, existem interpolações de ordem maior.

## 2.7 Extração e Modelagem

Nesta seção são descritas as técnicas de processamento de imagens para identificação dos voxels que pertencem aos objetos de interesse e para extração desta informação, através da construção de um modelo de superfície, ou de volume, para cada objeto. Estas técnicas envolvem segmentação, representação *octree* de dados binários, *masking*, extração de contornos e interpolação de superfícies. A extração e modelagem representa uma transformação do espaço voxel no espaço objeto. Dois tipos de modelos caracterizam o espaço objeto: modelo de superfície e modelo de volume binário (figura 2.5). Diferentes estruturas de dados podem ser utilizadas para cada tipo de modelo, visando um armazenamento e manipulação mais eficiente dos objetos. Nesta seção, também é visto que a interpolação visando um espaço voxel com amostragem isotrópica (seção 2.6.4), pode ser feita no espaço objeto representado pelo modelo de volume binário.

### 2.7.1 Segmentação

O objetivo da segmentação é identificar os voxels que pertencem a uma estrutura de interesse. Para tanto são exploradas características de similaridade, descontinuidade e conectividade decorrentes da distribuição de densidade dos voxels. A principal diferença entre as técnicas de segmentação está em como estas características são exploradas. As técnicas de segmentação mais utilizadas são *thresholding* (métodos baseados na escolha de valores de limiar), métodos baseados no crescimento de regiões, métodos baseados em detecção de bordas e *feature plot method* (métodos baseados em plotagem de características). Os métodos de escolha de limiar exploram a característica de similaridade, os métodos de crescimento de regiões exploraram a característica de similaridade e conectividade, os métodos baseados em detecção de bordas exploram a característica de descontinuidade, e os métodos baseados em plotagem de características podem explorar todas elas.

#### 2.7.1.1 Métodos Baseados na Escolha de Valores de Limiar

Supondo $n$ o número de objetos de interesse, os métodos baseados na escolha de valores de limiar são aplicados com sucesso quando é possível escolher $n$ intervalos disjuntos

de densidade de voxels que caracterizam cada objeto. Um limiar de densidade inferior e outro superior são escolhidos para cada intervalo. Um número inteiro $k$, tal que $1 \leq k \leq n$, é associado ao voxel, cuja densidade está no intervalo correspondente ao $k$-ésimo objeto de interesse. O número zero é associado ao voxel que não pertence a nenhum dos intervalos. O espaço objeto resultante da operação é um espaço voxel $(n+1)$-ário (possui $n+1$ valores de densidade distintos). O caso mais simples, e comum, é quando existe um único objeto de interesse. Um exemplo é a identificação de uma estrutura óssea envolvida por músculos, gordura e pele, cujas imagens foram obtidas através da tomografia por raios-x. Supondo a profundidade do espaço voxel 8 bits, existem 256 $(0-255)$ tons de cinza para representar as imagens. Devido a estrutura óssea ser a mais densa, é natural que os voxels desta estrutura apresentem densidade elevada . A escolha de um limiar inferior de densidade 170 e um limiar superior 255, por exemplo, torna possível a identificação de todos os voxels pertencentes a estrutura óssea com densidades neste intervalo. O espaço objeto resultante desta operação é um espaço voxel binário, que representa um modelo de volume binário (figura 2.5).

A escolha dos valores de limiar inferior e superior é um conhecimento *a priori*, que pode ser obtido pela análise das densidades dos voxels no *display* das fatias, ou calculado com base no histograma (espectro de densidades associado aos voxels de um dado objeto) do espaço voxel de entrada.

#### 2.7.1.2 Métodos Baseados no Crescimento de Regiões

Os métodos baseados na escolha de valores de limiar falham quando dois, ou mais, intervalos de densidade de voxels não são disjuntos: um voxel que pertence a interseção de dois intervalos, não pode ser associado com seguraça a nenhum dos objetos. Entretanto, se os voxels na vizinhança da borda de um dos objetos, não pertencerem ao outro objeto, os métodos baseados no crescimento de regiões conseguem com segurança segmentar os objetos. Estes métodos se baseiam, além dos intervalos de densidade, na escolha de um ponto semente e de um critério de conectividade para cada objeto de interesse [72]. O resultado genérico desta operação também é um espaço voxel $(n+1)$-ário.

Um exemplo é o algoritmo de conectividade usado por Cline et al. [30] para reconstrução 3D de um cérebro, utilizando imagens obtidas por ressonância magnética. O cérebro é o único objeto de interesse, o ponto semente é o ponto de partida do algoritmo e é

representado por um voxel escolhido no interior, ou na borda, do objeto. A escolha do ponto semente pode ser feita através do *display* das fatias. Um critério de conectividade diz que dois voxels estão conectados, quando ambos pertencem ao mesmo intervalo de densidades e existe um caminho entre eles, através de voxels adjacentes, que também pertence. Inicialmente, o ponto semente é marcado e os seis voxels adjacentes por uma face são pesquisados, quatro na mesma fatia e dois na mesma localização $(x, y)$ das fatias adjacentes. Neste caso o critério de conectividade usa vizinhança 6 (figura 2.9b). Os pontos pesquisados que estão conectados ao ponto semente são marcados e a densidade 0 é atribuída aos desconectados. Desmarca-se o ponto semente e atribui a ele densidade 1. Cada voxel marcado passa ser um novo ponto semente e o mesmo processo é repetido para cada um deles até que não haja mais pontos marcados. Aos outros pontos da fatia é atribuída densidade 0. Os pontos da borda do objeto são aqueles que pelo menos um dos seis vizinhos fica com densidade 0. Os pontos do interior do objeto são aqueles que todos os seis vizinhos ficam com densidade 1.

Uma ilustração do algoritmo de conectividade, para o caso 2D, é apresentada na figura 2.13. Neste caso, o critério de conectividade usa vizinhança 4. A figura 2.13a mostra a imagem de uma fatia com um ponto semente marcado por um quadrado. Supondo que o objeto de interesse é identificado pelos voxels conectados ao ponto semente, com densidade maior ou igual a 3 (limiar inferior) e menor ou igual a 5 (limiar superior). A figura 2.13b mostra a situação após algumas etapas e a figura 2.13c mostra a situação final.

| | | | | |
|---|---|---|---|---|
| 7 | 3 | 5 | 6 | 6 |
| 3 | 6 | 2 | 3 | 2 |
| 5 | 4 | 4 | 2 | 2 |
| 2 | 3 | [5] | 4 | 8 |
| 9 | 2 | 3 | 2 | 3 |
| 2 | 8 | 7 | 9 | 2 |

(a)

| | | | | |
|---|---|---|---|---|
| 7 | 3 | 5 | 6 | 6 |
| [3] | 0 | 0 | 3 | 2 |
| 1 | 1 | 1 | 0 | 2 |
| 0 | 1 | 1 | 1 | 0 |
| 9 | 0 | 1 | 0 | 3 |
| 2 | 8 | 0 | 9 | 2 |

(b)

| | | | | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 |

(c)

Figura 2.13: Exemplo do algoritmo de conectividade em 2D.

#### 2.7.1.3 Métodos Baseados em Detecção de Bordas

A borda de um objeto de interesse é representada pelas faces que separam voxels-1 de voxels-0 [6]. O objetivo destes métodos é extrair a borda, utilizando-a como modelo de superfície cubiculada (figura 2.5). Estes métodos se baseiam na escolha inicial de um voxel candidato a pertencer a borda do objeto de interesse. Esta escolha pode ser feita através do *display* das fatias. O algoritmo pesquisa os voxels vizinhos para encontrar uma transição voxel-1 para voxel-0, ou vice-versa. Encontrada a face inicial, o algoritmo procede encontrando as faces adjacentes que também pertencem a borda até voltar a face inicial. Alguns objetos possuem borda externa e bordas internas, relativas a buracos no seu interior. Neste caso, o algoritmo pode ser aplicado para todas as bordas.

Métodos baseados em detecção de bordas podem ter como entrada o espaço voxel em tons de cinza, ou o espaço objeto resultante dos métodos baseados na escolha de valores de limiar e métodos baseados no crescimento de regiões (espaço voxel binário). Em ambos os casos, podem existir voxels-1 próximos a borda, em pequenos grupos ou isolados, que não pertençam ao objeto de interesse (ruído). Neste caso, uma definição matemática precisa para o objeto pode ser necessária. O objeto pode ser definido, por exemplo, como o conjunto de voxels-1 conectados por uma face. Quando a entrada é o espaço voxel em tons de cinza, uma exigência básica para o sucesso destes métodos é que a velocidade de variação das densidades dos voxels, nas proximidades da borda do objeto de interesse, seja bem maior que afastado dela. A principal diferença entre estes métodos são as diferentes formas de detetar as descontinuidades para localizar a borda. A filtragem passa-altas (seção 2.6.3.2), por exemplo, pode ser utilizada com este propósito. Similarmente, diferenças locais de densidade de voxels são calculadas com o objetivo de estimar o vetor gradiente no ponto $(x, y, z)$, correspondente ao centro geométrico da região de voxels vizinhos utilizada. O módulo do vetor gradiente estimado indica a velocidade da variação de densidade para cada ponto $(x, y, z)$, e sua direção indica a direção de máxima variação.

O espaço objeto resultante destes métodos é representado por uma estrutura de dados que armazena apenas a coordenada do centro geométrico das faces. Um exemplo de estrutura eficiente para manipulação são seis arquivos que guardam, respectivamente,

[6] Voxels-1 são voxels que têm densidade 1, ou que por algum critério são considerados pertencentes ao objeto de interesse. Voxels-0 têm conceito complementar.

as coordenadas das faces vistas das seis direções principais $(x^+, x^-, y^+, y^-, z^+, z^-)$. Esta estrutura representa um modelo de superfície para o objeto de interesse (figura 2.5). A aparência externa do modelo de volume binário é a mesma deste modelo de superfície. A principal diferença entre os modelos é que o primeiro armazena todos os voxels que formam o objeto de interesse e o segundo armazena apenas a casca. Dependendo da estrutura de dados escolhida, pode haver vantagens e desvantagens entre os modelos, com relação a facilidade de manipulação e o espaço de memória para armazenamento.

Udupa [39] apresenta um algoritmo de detecção de bordas no espaço voxel em tons de cinza e no espaço voxel binário. No primeiro caso é utilizada a informação da magnitude do vetor gradiente em cada passo do algoritmo. Os métodos de detecção de bordas no espaço voxel binário são conhecidos por *surface tracking method*, por caminharem sobre a superfície do modelo de volume binário. Stytz et al. [62] cita as referências [15] [24] [25] [10] [40] como exemplos de *surface tracking methods*, entre as quais o primeiro trabalho voltado à visualização de volumes foi descrito por Herman e Liu [25].

#### 2.7.1.4 Métodos Baseados em Plotagem de Características

Métodos baseados em plotagem de características são similares aos métodos de escolha de limiar, porém o valor do pixel possui duas ou mais dimensões (características). Nestes métodos, o modelo de volumes binário é gerado baseado no reconhecimento de um *cluster* (aglomerado) de pontos, que identifica os voxels do objeto de interesse. O aglomerado pode ser visualizado plotando, em um sistema de coordenadas cartesianas, os valores das características para todos os voxels do espaço de entrada. Cada característica é associada a um eixo do sistema. Normalmente são utilizadas duas características. A figura 2.14a mostra o caso em que foram plotados os valores de duas características, $c_1$ e $c_2$, e os pontos aglomerados foram isolados dos outros pontos traçando uma linha[7]. Para cada ponto do aglomerado, atribui-se densidade 1 ao voxel correspondente no espaço. A densidade 0 é atribuída a todos os outros voxels.

Um exemplo da situação acima é um infarto no miocárdio, com imagens obtidas por ressonância magnética. Na figura 2.14b $T_1$ representa o miocárdio, $T_2$ a região infartada e $T_3$ a região externa ao miocárdio [39]. Supondo que $T_1$, $T_2$ e $T_3$ representam três regiões

[7] Outro tipo de curva é as vezes necessário para fazer o isolamento.

de tecidos diferentes em um corte transversal, tal que, as densidades médias, $t_1$, $t_2$ e $t_3$, dos voxels nestas regiões sejam da ordem $t_1 \geq t_2 \geq t_3$. A região de fronteira entre $T_1$ e $T_3$ contém provavelmente ambos tipos de tecidos ($T_1$ e $T_3$), os seus voxels têm densidades entre $t_1$ e $t_3$, e portanto muito próximas a $t_2$. Este fenômeno é conhecido por *partial volume artifact*. Extrair $T_2$ escolhendo um limiar superior $t_2 + \Delta t$ e um inferior $t_2 - \Delta t$, onde $\Delta t$ é escolhido, para cobrir o espectro de densidades dos voxels internos a $T_2$, faz com que os voxels da região de fronteira entre $T_1$ e $T_3$ venham junto. Incluindo uma segunda carcterística, tal como, a magnitude do vetor gradiente para cada voxel, é possível extrair apenas $T_2$, devido a magnitude do vetor gradiente para os voxels da fronteira entre $T_1$ e $T_3$ ser bem maior que para os voxels do interior de $T_2$. A plotagem desta situação é apresentada na figura 2.14c. Três aglomerados de pontos podem ser identificados próximos ao eixo da densidade $c_1$ (baixos valores de magnitude). Estes aglomerados correspondem, respectivamente, às regiões internas dos tecidos $T_1$, $T_2$ e $T_3$. O arco formado pelos pontos entre os aglomerados $T_1$ e $T_3$ representa os voxels da fronteira entre estas duas regiões. O mesmo pode ser dito para os outros arcos. A região $T_2$ é portanto identificada no aglomerado de pontos central e pode ser extraída sem a fronteira entre $T_1$ e $T_3$.

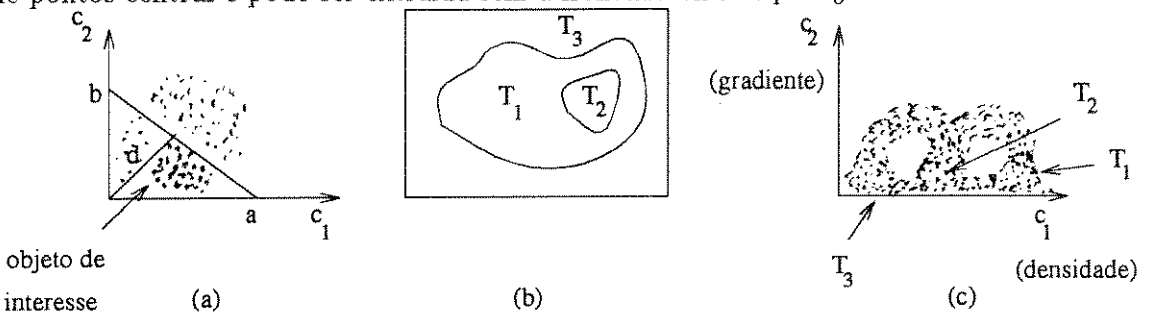


Figura 2.14: Plotagem de características.

### 2.7.2 Representação *Octree* de Dados Binários

Existem várias formas interessantes para armazenar o modelo de volume binário (figura 2.5). A mais simples é a própria representação do espaço voxel (linha por linha, fatia por fatia). Esta representação leva a vantagem de acesso direto aos voxels, mas pode ocupar mais memória do que as outras formas de representação. Outro tipo de estrutura mais complexa, e que permite uma compactação dos dados seguindo uma hierarquia, é a

representação *octree*. Esta representação resulta de repetidas subdivisões do espaço voxel em subespaços até que todos os subespaços tenham apenas voxels-1 ou voxels-0. A estrutura final é uma árvore, onde cada nó tem oito ramos. Quando o espaço tem mais de um objeto de interesse é feita uma representação *octree* para cada objeto isoladamente. Apesar da construção da estrutura *octree* representar uma etapa a mais no processamento dos dados e do acesso aos voxels não ser direto, operações simples (como união, interseção, e diferença entre objetos; translação, rotação, e escalamento; detecção de interferências e remoção de superfícies escondidas) podem ser feitas acessando cada nó da árvore no máximo uma vez. Além disto, estas operações requerem aritmética simples, tais como, adições de inteiros, *shifts* (deslocamento de bits) e comparações [21]. A figura 2.15 ilustra a idéia da representação *octree* de um modelo de volume binário. O espaço voxel (figura 2.15a) é dividido em oito subespaços iguais, que são identificados por números associados (figura 2.15b). O processo de subdivisão termina para os subespaços que contêm apenas voxels-0 (vazios) ou voxels-1 (cheios). No caso de subespaços parciais, cada subespaço é dividido em oito novos subespaços seguindo a mesma lógica de numeração da figura 2.15b. O resultado final é a estrutura árvore octal da figura 2.15c, onde os subespaços cheios e vazios representam as folhas e os subespaços parciais representam os nós com oito ramos cada.

A idéia da representação *octree* é uma expansão para 3D da representação *quadtree* utilizada em processamento de imagens. Entretanto, outras variações para 3D podem ser feitas. Um exemplo é a mistura da estrutura *quadtree* com a estrutura *run-lenght encoding* (ver seção 2.7.4), onde a estrutura *quadtree* é aplicada visualizando o plano yz do espaço voxel e a estrutura *run-lenght encoding* é aplicado no eixo x [46].

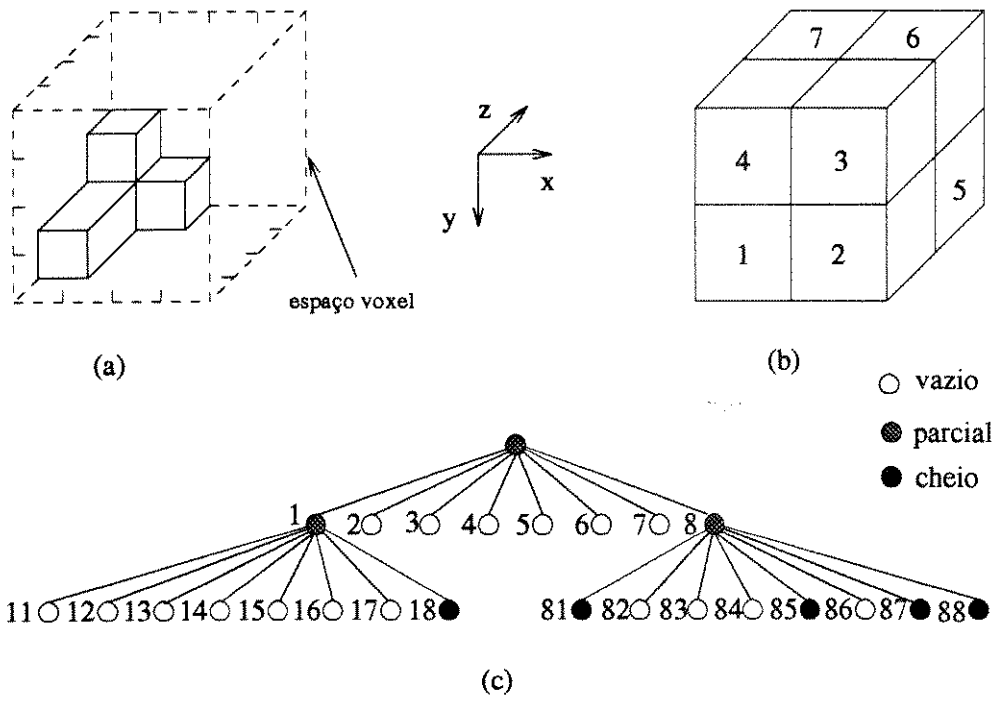


Figura 2.15: Estrutura de dados *octree*.

### 2.7.3 Mascaramento

A segmentação automática do objeto de interesse nem sempre é realizada com sucesso. Quando isto ocorre, a única alternativa é ter um médico experiente que desenhe sobre as fatias do espaço voxel de entrada uma curva em torno do objeto de interesse, que possa identificá-lo após a segmentação. No pior caso, o médico tem que desenhar precisamente a borda do objeto para todas as imagens. Para auxiliar o trabalho do médico, é suficiente desenhar uma curva fechada e precisa, nos trechos críticos de algumas fatias, e nos outros trechos, apenas garantir que o objeto está no seu interior. Cada curva deve ser traçada de forma a garantir que a segmentação automática funcione no seu interior. Algumas curvas podem ser aproveitadas em outras fatias. A região envolvida por cada curva é conhecida por região da máscara. Em seguida, as curvas traçadas são transferidas para um outro arquivo, gerando um espaço voxel binário da seguinte forma: classificando, os voxels em cada linha das imagens na ordem ascendente da coordenada $x$, atribui-se densidade 1 aos voxels entre todo par sucessivo de voxels da curva na lista classificada, e

densidade 0 aos demais. Todos os voxels com densidade 1 nas imagens binárias geradas estão no interior ou sobre a curva da imagem correspondente.

A segmentação automática é feita usando o espaço voxel em tons de cinza e o espaço voxel binário (máscara) como entradas. Esta operação é realizada considerando apenas os voxels do espaço em tons de cinza que têm densidade 1 no espaço binário, e atribuindo densidade 0 aos demais. Um exemplo de mascaramento é descrito por Udupa [39], quando a segmentação automática da articulação de uma junta não é possível devido ao fenômeno de *partial volume artifact* descrito na seção 2.7.1.4.

### 2.7.4 Extração de Contornos

Os contornos são representados por pontos pertencentes aos voxels, que compõem a borda do objeto extraídos de cada fatia. Quando o objeto possui mais de uma borda, pode ocorrer um ou mais contornos por fatia. Cada contorno pode ser aproximado por uma curva fechada com regiões interna e externa bem definidas, utilizando os pontos como pontos de controle para esta estimativa (ex: B-spline [38] [13]). Entretanto, a forma mais comum de aproximação é interligando os pontos por segmentos de reta. Neste caso, os contornos podem ser visualizados em forma de linhas (figura 2.16a e figura 2.16b), como um conjunto de segmentos de reta interligados por pontos, ou em forma de tiras (figura 2.16c), onde cada segmento representa uma face quadrada da borda do objeto. O modelo de superfície, representado pelo empilhamento dos contornos em forma de linhas (figura 2.16d), é conhecido por *wireframe contour* (modelo de contornos aramados). Este modelo serve para dar uma idéia limitada da anatomia da superfície do objeto. O modelo de superfície representado pelo empilhamento dos contornos em forma de tiras, fornece uma aproximação discreta da anatomia da superfície lateral [8] do objeto.

A extração de contornos pode ser feita utilizando como entrada o espaço voxel, ou o espaço objeto resultante das operações de segmentação. No espaço voxel, a extração de contornos é feita durante a aplicação dos métodos de segmentação baseados em detecção de bordas (ver seção 2.7.1.3). No espaço objeto, a informação dos contornos do objeto extraídos de cada fatia está embutida no modelo de superfície cubiculada ou no modelo de

[8] A visualização deste modelo só faz sentido se for de um ponto de vista cuja direção de visualização seja paralela ao plano $xy$ das fatias.

volume binário (figura 2.5). A extração de contornos utilizando o modelo de volume binário como entrada, normalmente é feita aproveitando a forma natural em que os dados estão disponíveis; linha por linha, fatia por fatia (figura 2.5). Os contornos podem ser obtidos processando uma fatia do espaço voxel binário por vez. Em Udupa [39] pode ser visto um exemplo deste tipo de algoritmo. Neste caso, dois pontos do contorno, representando uma face da borda do objeto, são inicialmente escolhidos [9]. Em seguida, o algoritmo percorre a borda do objeto pesquisando as densidades dos voxels vizinhos até voltar a face inicial. Kim et al. [77] apresenta uma variação interessante deste tipo de algoritmo. A imagem da fatia binária é inicialmente lida, da esquerda para a direita e de cima para baixo, montando uma estrutura de corridas [10]. Cada corrida recebe uma classificação. A estrutura de corridas classificadas representa uma compressão da imagem da fatia conhecida por *run-length encoding* (codificação por comprimento de corridas). Os contornos são, portanto, extraídos desta estrutura em forma de *chaincode* (codificação em cadeia) [11]. Duas vantagens importantes do algoritmo de Kim et al. [77] em relação ao algoritmo do Udupa [39], são a rapidez de processamento e a facilidade para determinar a hierarquia entre os contornos de uma mesma fatia (quem está contido em quem). Um terceira vantagem é a obtenção de contornos suavizados (figura 2.16b): no algoritmo do Udupa [39], os contornos são considerados discretos (figura 2.16a); pois as possíveis direções dos segmentos de reta, que unem os pontos de um contorno, acompanham as direções das faces da borda do objeto. Quando deseja-se interpolar uma superfície para o objeto a partir do modelo de contornos aramados (ver seção 2.7.5), necessita-se de um algoritmo de suavização para estes contornos [39]. Também é comum eliminar os pontos do contorno que pertencem a segmentos consecutivos de mesma declividade, armazenando apenas os pontos de mudança de declividade.

[9] A escolha automática dos pontos iniciais pode ser feita lendo a imagem binária da fatia, da esquerda para direita e de cima para baixo, até encontrar a primeira transição de voxel-0 para voxel-1.

[10] Uma corrida é definida pelo conjunto de voxels-1 em uma mesma linha da imagem, iniciando em uma transição voxel-0 para voxel-1 e terminando na próxima transição voxel-1 para voxel-0.

[11] Na codificação em cadeia tem-se a coordenada do primeiro ponto (centro geométrico do voxel inicial) e uma sequência de inteiros entre 0 e 7 que identificam 8 possíveis direções para cada segmento de reta do contorno.

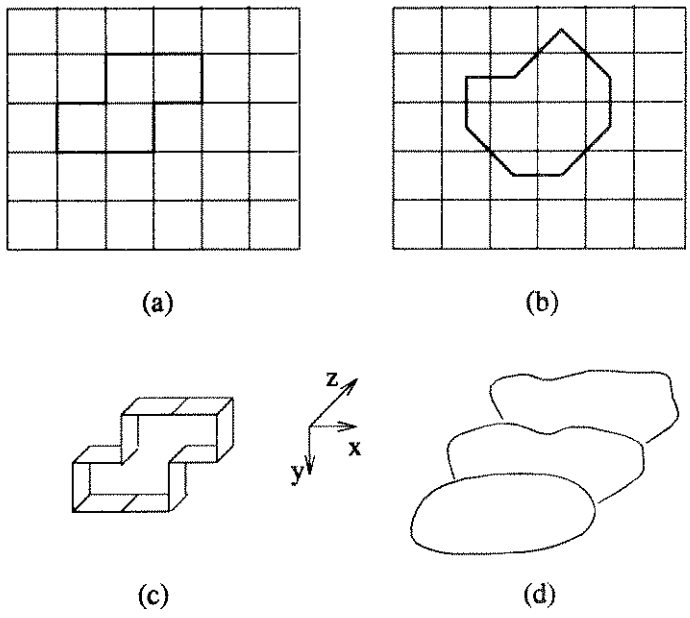


Figura 2.16: a) Contorno discreto b) Contorno suave c) Contorno discreto em forma de tira d) Modelo de contorno aramado.

### 2.7.5 Interpolação de Superfícies

A interpolação de superfícies procura obter uma estimativa mais próxima possível da anatomia real da superfície do objeto. O modelo de superfície utilizado consiste de um conjunto de primitivas geométricas (*patches*), dando o aspecto de superfície ladrilhada [12]. As primitivas geométricas podem ser polígonos ou superfícies curvas. Os pontos de contornos extraídos das fatias podem ser utilizados, por exemplo, como pontos de controle para interpolar superfícies *B-spline* [38] [13]. A superfície também pode ser interpolada a partir do espaço voxel pré-processado [85] [30]. Entretanto, a forma mais comum utiliza o modelo de contornos aramados e compõem a superfície utilizando triângulos para unir contornos de cada duas fatias sucessivas [28] [2] (ver um dos modelos de superfície da figura 2.5). Cada face triangular da superfície é formada por um segmento de reta, que une dois pontos de um contorno, e duas diagonais conectando os pontos extremos do segmento com um ponto comum no contorno adjacente. Cada diagonal pertence a duas faces triangulares formando

[12] Devido ao aspecto de superfície ladrilhada, técnicas de interpolação de superfícies também são conhecidas por *tiling method*.

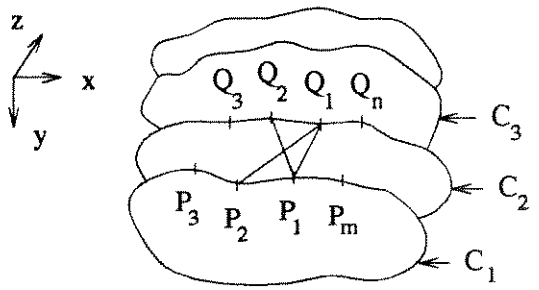


Figura 2.17: Interpolação de superfície usando triângulos.

uma superfície fechada. O processo de escolha dos triângulos pode utilizar critérios ótimos ou heurísticos. Um exemplo utilizando um critério heurístico é ilustrado na figura 2.17. Sejam $C_1$ $(P_1, P_2, \cdots, P_m)$ e $C_2$ $(Q_1, Q_2, \cdots, Q_n)$ os contornos de duas fatias sucessivas. O algoritmo inicia pela escolha de uma diagonal inicial [13]. Os triângulos podem ter dois vértices em $C_1$ e um vértice em $C_2$ ou ter dois vértices em $C_2$ e um vértice em $C_1$. Supondo que a aresta inicial é $P_1Q_1$, existem duas arestas possíveis para formar o primeiro triângulo: $P_1Q_2$ ou $P_2Q_1$. A escolha de $P_1Q_2$ resulta o triângulo $P_1Q_2Q_1$ e a escolha de $P_2Q_1$ resulta o triângulo $P_2Q_1P_1$. Utilizando como critério heurístico a aresta de menor comprimento e supondo que $P_1Q_2 < P_2Q_1$, a próxima escolha deve ser entre as arestas $P_1Q_3$ e $P_2Q_2$. O processo continua até $P_1Q_1$ fazer parte novamente de um triângulo. A próxima etapa é aplicar o mesmo algoritmo entre os pontos do contorno $C_2$ e o contorno $C_3$ da terceira fatia. O algoritmo é repetido de duas em duas fatias sucessivas até toda superfície ser formada.

A interpolação a partir do modelo de contornos aramados utiliza contornos suaves e com o menor número de pontos possível. Portanto, se o algoritmo utilizado para extrair contornos gera contornos discretos, é necessária uma etapa de pré-processamento que suavize os contornos e reduza o número de pontos. Udupa [39] apresenta um algoritmo com este propósito. Durante a interpolação podem ocorrer situações de ambiguidade, quando existe mais de um contorno por fatia, devido a existência de ramificações ou junções no objeto de interesse, podendo tornar necessária a intervenção do usuário no algoritmo. Os métodos de interpolação de superfícies são descritos com mais detalhes no capítulo 3.

[13] Existem muitas formas de determinar uma diagonal inicial. Uma delas é escolhendo o par de pontos entre os quais a distância é mínima comparada com todas as distâncias possíveis entre pontos de $C_1$ e $C_2$.

### 2.7.6 Interpolação no Espaço Objeto

A interpolação visando voxels cúbicos (seção 2.6.4) também pode ser realizada no espaço objeto. Um exemplo é o método *euclidean shape-based interpolation* [79]. Este método transforma inicialmente o espaço voxel binário $E_b$ em um espaço voxel $E_d$ (Transformada de Distância): para cada fatia de $E_d$ associa-se à densidade de cada voxel $v$, a distância $d$ entre o centro de $v$ e o ponto mais próximo na borda do objeto. Valores positivos de $d$ são associados aos voxels do interior do objeto (voxels-1 em $E_b$) e valores negativos de $d$ aos voxels do exterior (voxels-0 em $E_b$). Em seguida, pode ser aplicado ao espaço voxel $E_d$ qualquer método de interpolação para estimar fatias intermediárias em um espaço voxel em tons de cinza (seção 2.6.4). No final, o espaço interpolado $E_{d'}$ é transformado em um espaço voxel binário $E_{b'}$, associando a cada voxel $v'$ de $E_{b'}$ densidade 1, se o voxel da posição correspondente em $E_{d'}$ tem densidade positiva, e densidade 0 em qualquer outro caso. Uma variação deste método combina a interpolação no espaço voxel original com a do espaço objeto, usando o gradiente local como fator de normalização na combinação [68]. Um outro exemplo de interpolação no espaço objeto é o método utilizado por Lin Wei-Chung e Chen Shiuh-Yung [84] para estimar contornos intermediários no modelo de contornos aramados.

## 2.8 Operações para Visualização

As operações para visualização consistem da idealização de um ambiente de visualização, com seus elementos básicos; fontes de luz, observador, objetos, etc; onde são realizadas transformações geométricas e operações de *rendering*, gerando imagens de vários pontos de vista. Exemplos de objetos para visualização são os modelos de superfície, o modelo de volume binário e o volume de dados colorido e semitransparente (figura 2.5). O observador tem o mesmo ponto de vista do usuário do sistema de visualização. Sendo a visualização normalmente utilizada para auxiliar em diagnósticos médicos. O objetivo principal é simplificar o máximo possível os efeitos visuais, provocados pelas reflexões das fontes de luz sobre os objetos, sem comprometer a qualidade da imagem. Estes efeitos podem dificultar a interpretação das imagens, prejudicando o diagnóstico. Outra vantagem da simplificação é a obtenção de imagens com rapidez, tornando o sistema de visualização mais interativo com o usuário. Esta simplificação é normalmente obtida utilizando um única fonte de luz

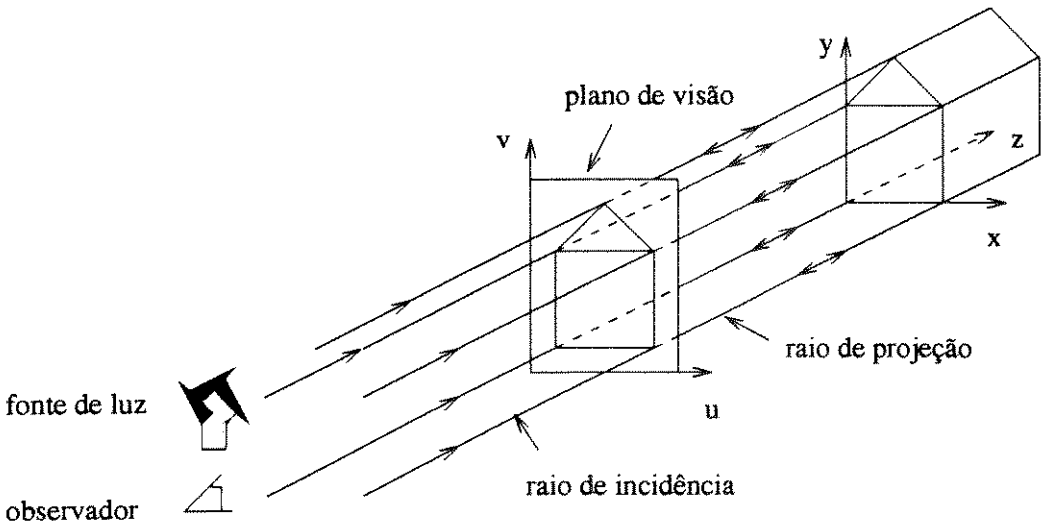


Figura 2.18: Projeção ortogonal.

na mesma posição do observador, e considerando os raios de incidência da fonte sobre os objetos e a direção de visualização paralelos entre si (figura 2.18). Esta condição é adotada na descrição das técnicas deste trabalho, e faz com que os raios de projeção [14] sejam paralelos entre si e ortogonais ao plano de visão (projeção ortogonal) [38]. A projeção ortogonal e o posicionamento da fonte de luz no observador simplificam o processamento, eliminam a possibilidade de sombras na imagem e facilitam a extração de medidas, identificadas diretamente no plano de visão, por preservar as distâncias entre os raios de projeção. Múltiplas reflexões de luz em uma mesma superfície, ou entre objetos, também são desconsideradas.

As operações para visualização podem ser consideradas transformações do espaço objeto, representado pelo sistema de coordenadas $(x, y, z)$, no espaço imagem, representado pelo sistema de coordenadas $(u, v)$ (figura 2.18). Modelos de superfície por primitivas geométricas (figura 2.5) e por contornos aramados (figura 2.16a) são mais comuns em computação gráfica, e as operações de *rendering* para estes modelos podem ser encontradas nos livros básicos da área [38] [13]. Também existem pacotes gráficos, como o PHIGS, que executam as transformações geométricas e o *rendering* destes modelos. Portanto, neste trabalho as operações de *rendering* são descritas visando os modelos baseados em voxel binário; o modelo de volume binário, o modelo de superfície cubiculada (figura 2.5), e o

[14] Os raios de projeção equivalem aos raios da fonte de luz refletidos na superfície dos objetos em direção aos olhos do observador (paralelos à direção de visualização).

modelo de contornos em forma de tiras (figura 2.16c). Por simplicidade, a descrição das operações de visualização utiliza o modelo de volume binário, mas é facilmente adaptada aos outros modelos, pois todos eles consistem de um conjunto de pontos, que representam o centro de uma pequena área quadrada da superfície.

### 2.8.1 Transformações Geométricas

Inicialmente os elementos do ambiente de visualização têm o posicionamento absoluto definido com relação ao sistema de coordenadas $(x, y, z)$ (espaço objeto). As transformações geométricas podem ser aplicadas a qualquer dos elementos do ambiente, representando uma transformação do espaço objeto no espaço objeto. Estas transformações definem o posicionamento relativo entre os elementos do ambiente, consistindo das operações de translação, rotação e escalamento aplicadas a cada ponto $(x, y, z)$ que define o elemento no espaço objeto, onde $(x', y', z')$ são as coordenadas cartesianas do ponto $(x, y, z)$ após a transformação.

Neste trabalho, o posicionamento inicial adotado para os elementos do ambiente de visualização é ilustrado na figura 2.20. O observador e a fonte de luz estão situados no infinito do semi-eixo $y-$, o plano de visão $uv$ tem sua origem no centro geométrico do espaço imagem $(0, Yo, 0)$, sobre o semi-eixo $y-$ entre o observador e o objeto (representado pelo modelo de volume binário), de forma que não ocorram cortes despropositais sobre o objeto. O sistema de coordenadas $(x, y, z)$, que segue a regra da mão direita, tem sua origem no centro geométrico do espaço voxel binário. Dois graus de liberdade são definidos pelos ângulos $\alpha$ e $\beta$ para que observador, fonte de luz e plano de visão $uv$ possam girar em torno da origem do sistema $(x, y, z)$, obtendo a visualização de qualquer das faces do espaço voxel (uma forma mais trabalhosa é manter estes elementos fixos e rotacionar apenas os elementos do espaço voxel). Os ângulos de rotação $\alpha$ e $\beta$ equivalem, respectivamente, aos movimentos de rotação (spin) e inclinação (tilt), do paciente em torno dos eixos longitudinal (eixo z) e lateral (eixo x).

A operação de rotação de um ângulo $\alpha$ no sentido anti-horário em torno do eixo

z equivale a seguinte equação em coordenadas homogêneas [15]:

$$\begin{bmatrix} x' & y' & z' & 1 \end{bmatrix} = \begin{bmatrix} x & y & z & 1 \end{bmatrix} \times \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 & 0 \\ \sin\alpha & \cos\alpha & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.5}$$

A rotação de um ângulo $\beta$ no sentido anti-horário em torno do eixo x equivale:

$$\begin{bmatrix} x' & y' & z' & 1 \end{bmatrix} = \begin{bmatrix} x & y & z & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\beta & -\sin\beta & 0 \\ 0 & \sin\beta & \cos\beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.6}$$

Uma translação do plano de visão $\vec{T} = (T_x, T_y, T_z)$ também é possível para efetuar cortes no objeto. O efeito de escalamento do objeto pode ser realizado mais facilmente no espaço imagem, após as operações de *rendering*. A combinação das matrizes de rotação e translação é dada pela equação:

$$M = \begin{bmatrix} cos\alpha & -sen\alpha cos\beta & sen\alpha sen\beta & 0 \\ sen\alpha & cos\alpha cos\beta & -cos\alpha sen\beta & 0 \\ 0 & sen\beta & cos\beta & 0 \\ T_x & T_y & T_z & 1 \end{bmatrix} \tag{2.7}$$

### 2.8.2 *Rendering*

Definido o posicionamento relativo entre os elementos do ambiente de visualização, o *rendering* pode ser definido como o conjunto de operações necessárias para obter a imagem do objeto, que está sendo visualizada pelo observador neste instante. As operações de *rendering* consistem de recorte, projeção com remoção de superfícies escondidas e *shading* (tonalização da imagem). O resultado destas operações é uma transformação do espaço objeto no espaço imagem.

[15] O uso de coordenadas homogêneas é importante para que todas as matrizes de transformação sejam multiplicativas e, portanto, possam ser combinadas em uma única matriz [49].

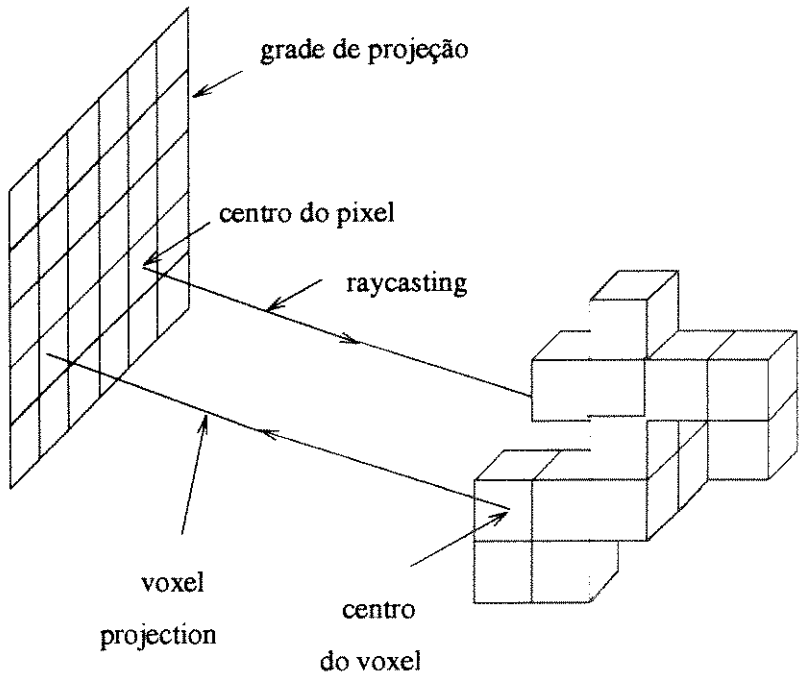


Figura 2.19: Projeção de voxels e *raycasting*.

#### 2.8.2.1 Recorte

O corte, ou não, de partes selecionadas do objeto na imagem é feito pela escolha adequada das dimensões do espaço imagem; $u_{max}$, $u_{min}$, $v_{max}$ e $v_{min}$ (figura 2.20). Normalmente é desejável fixar estas dimensões de forma a garantir que o objeto possa ser visualizado de qualquer ângulo sem cortes. Isto pode ser feito, por exemplo, definindo a largura e a altura iguais a $(a^2 + b^2 + c^2)^{\frac{1}{2}}$, onde $a, b$ e $c$ são as dimensões do menor cubóide que envolve o objeto.

#### 2.8.2.2 Projeção com Remoção de Superfícies Escondidas

Existem duas maneiras de fazer a projeção do objeto no espaço imagem para os modelos baseados em voxel [39]: projeção de voxels e *raycasting* (figura 2.19). Métodos que utilizam projeção de voxels, rastreiam os voxels do espaço objeto projetando os voxels-1 no espaço imagem (*object-order methods*). O método *raycasting* lança um raio de procura (figura 2.20) partindo de cada pixel em direção ao espaço objeto, até encontrar o primeiro

voxel-1 interceptado pelo raio, projetando-o no espaço imagem (*image-order method*).

A principal diferença entre os métodos que utilizam *voxel projection* é o tipo de algoritmo de remoção de superfícies escondidas [62]: *depth-sort*, *z-buffer*, *back-to-front* (BTF) e *front-to-back* (FTB). A remoção de superfícies escondidas está em identificar para cada pixel no espaço imagem, qual voxel-1 do espaço objeto, entre os voxels-1 que são projetados sobre o pixel, está mais próximo do observador, e utilizá-lo como uma pequena área da superfície visível. No algoritmo *z-buffer*, por exemplo, para cada pixel do espaço imagem são associados dois valores: um valor de tonalização (ver seção 2.8.2.3) e o valor da distância, entre o voxel-1 que está sendo projetado e o pixel, medida sobre o raio de projeção. Um novo valor de tonalização e um novo valor de distância, referentes a um outro voxel-1, só são associados ao pixel se o novo valor de distância for menor que o anterior. Isto garante que entre os voxels-1 que são interceptados pelo raio de projeção, o voxel visível é o que está mais próximo do observador. Os outros métodos que utilizam *voxel projection* e os problemas encontrados são descritos no capítulo 3. Uma forma simples de implementar *voxel projection* aproveitando a configuração utilizada para os elementos no ambiente de visualização (figura 2.20), rotaciona os voxels do espaço objeto usando a equação abaixo:

$$\begin{bmatrix} x' & y' & z' & 1 \end{bmatrix} = M \begin{bmatrix} x_0 & y_0 & z_0 & 1 \end{bmatrix} \tag{2.8}$$

onde:
$(x_0, y_0, z_0)$ é a coordenada de um voxel do espaço objeto
$(u, v) = (x', -z')$ é a coordenada de sua projeção no espaço imagem
$|Yo + y'|$ é a distância do ponto da superfície ao plano de visão, representando o valor de projeção associado ao pixel $(u, v)$ do espaço imagem (formação do *z-buffer*).

O método *raycasting* tem intrínseco o algoritmo de remoção de superfícies escondidas. Este método é adotado na implementação das técnicas deste trabalho e pode ser descrito da seguinte forma: para cada pixel do espaço imagem é lançado um raio partindo do pixel em direção ao espaço objeto (figura 2.20). Os raios são paralelos entre si e perpendiculares ao espaço imagem (direção $\vec{v}$ de visualização). Quando um raio penetra no espaço objeto, ele caminha até encontrar o primeiro voxel-1. A projeção do voxel-1 sobre o espaço imagem é efetuada associando ao pixel correspondente, sua distância ao voxel-1 medida sobre o raio de projeção [16] (formação do z-buffer).

[16] O raio de projeção tem a mesma direção do raio de procura, mas sentido oposto.

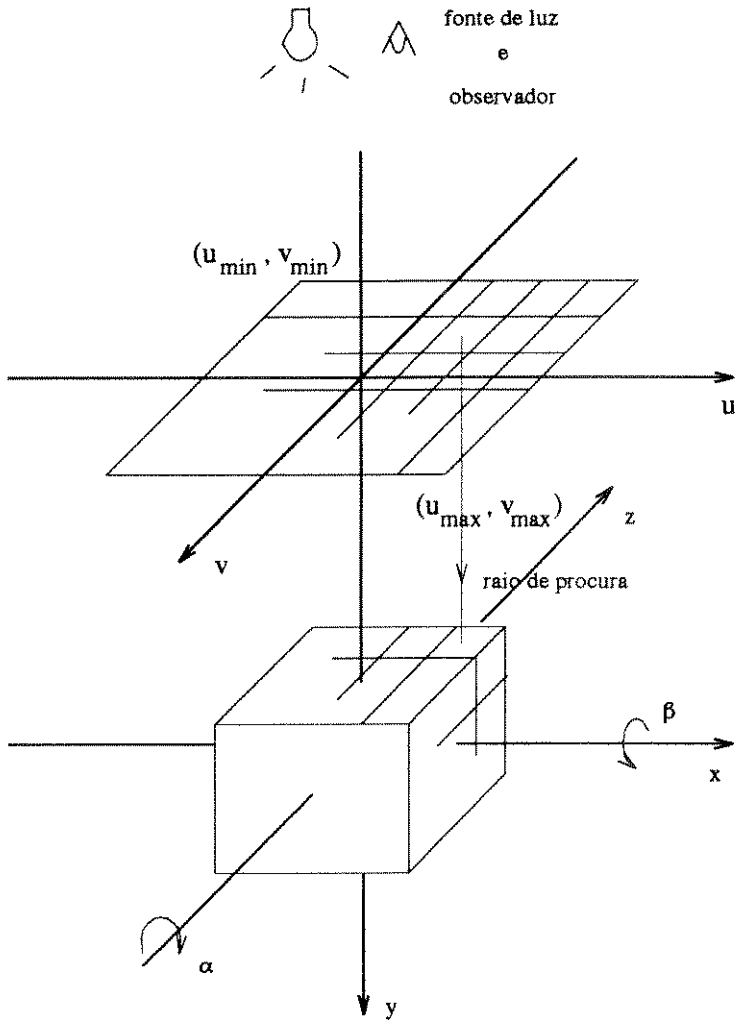


Figura 2.20: Posicionamento inicial dos elementos do ambiente de visualização.

A equação paramétrica de um raio lançado do pixel $(u = x_0, y_0, v = z_0)$ na direção inicial de visualização $\vec{v} = (0, 1, 0)$ é:

$$\begin{bmatrix} x(t) & y(t) & z(t) \end{bmatrix} = \begin{bmatrix} x_0 & y_0 & z_0 \end{bmatrix} + t \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \tag{2.9}$$

Sobre esta equação é aplicada a transformação M em coordenadas homogêneas:

$$\begin{bmatrix} x(t)' & y(t)' & z(t)' & 1 \end{bmatrix} = M \begin{bmatrix} x(t) & y(t) & z(t) & 1 \end{bmatrix} \tag{2.10}$$

Com efeito chega-se a equação recursiva utilizada pelo algoritmo para caminhar sobre o raio

de procura:

$$X(t_i) = X(t_{i-1}) + \Delta t \left[ \begin{array}{cccc} sen\alpha & cos\alpha cos\beta & -cos\alpha sen\beta & 1 \end{array} \right] \quad (2.11)$$

onde:
$X(t_i) = \left[ \begin{array}{cccc} x(t_i) & y(t_i) & z(t_i) & 1 \end{array} \right]$
$t_i$ é o valor do parâmetro t no instante i ($t_0 = 0$)
$\Delta t = t_i - t_{i-1}$

#### 2.8.2.3 Tonalização dos Pixels na Imagem

A operação de tonalização visa a distribuição de luminosidade sobre o espaço imagem, fornecendo a ilusão de tridimensionalidade perdida na projeção do objeto. A imagem é gerada associando a cada pixel da grade um tom de cinza. O valor desse tom de cinza deve ser tal que simule a intensidade de luz na superfície do objeto em questão. Diferentes tipos de tonalização podem ser obtidos para um mesmo modelo de iluminação previamente escolhido. O modelo mais comum na visualização de dados médicos é o modelo de Phong [6]. Neste modelo algumas propriedades do comportamento da luz são levadas em conta, para definir como a luz ambiente e a fonte de luz influenciam no cálculo do valor de tonalização:

**Influência da Fonte de Luz** A intensidade de luz refletida de cada ponto da superfície em direção ao observador cai com o aumento da distância entre o ponto e o observador. Este efeito é dado variando o valor de tonalização linearmente com à distância normalizada entre o ponto da superfície e o plano de visão (o valor do *z-buffer*) [17], e pode ser representado pela equação 2.12. Quanto mais próximo do plano de visão, mais próximo do observador, e portanto mais clara é a tonalização. Quanto mais afastado do plano de visão, mais longe do observador, e portanto mais escura é a tonalização.

Duas formas de reflexão da luz incidente são consideradas no modelo de iluminação:

**Reflexão Difusa** A intensidade de luz refletida de um ponto da superfície é a mesma em todas as direções, portanto atingindo o observador independente de

[17] Também consegue-se bons resultados usando variação não linear.

sua direção, mas varia com o cosseno do ângulo $\theta$ entre a direção do raio de luz incidente e a direção normal a superfície neste ponto (figura 2.21a).

**Reflexão Especular** A intensidade de luz refletida é máxima na direção $\vec{D}$ de reflexão que também forma um ângulo $\theta$ com a normal, no lado oposto ao da luz incidente, e vai diminuindo com o afastamento em relação à direção $\vec{D}$ de acordo com alguma regra (figura 2.21b). Esta intensidade varia com o cosseno do ângulo $\alpha$, entre a direção $\vec{D}$ e a direção do observador, elevado a um expoente n. Como a direção do observador e da fonte são iguais, $\alpha = 2\theta$.

**Influência da Luz Ambiente** A intensidade luz refletida pela superfície do objeto devido a luminosidade natural do ambiente e que chega aos olhos do observador é modelada por um fator constante.

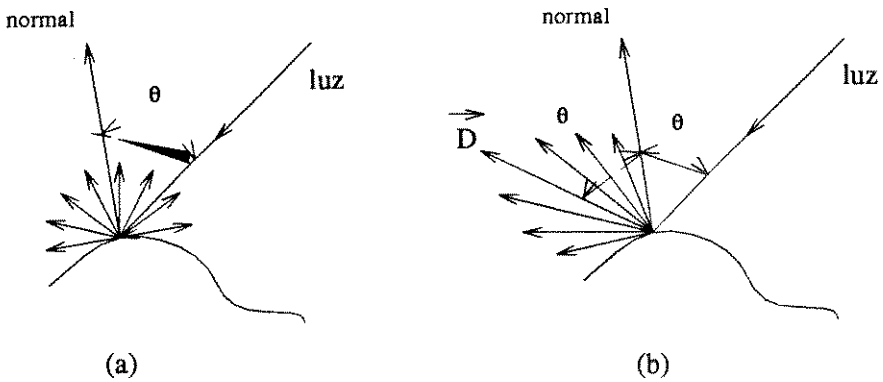


Figura 2.21: a) Reflexão difusa b) Reflexão especular.

O modelo de iluminação acima pode ser representado pelas equações abaixo:

$$I_{dist}(u,v) = \frac{I_{max} - I_{min}}{d_{min} - d_{max}}[d(u,v) - d_{max}] + I_{min} \tag{2.12}$$

$$I(u,v) = I_{max}K_a + I_{dist}(u,v)(K_d cos\theta + K_s cos^n 2\theta) \tag{2.13}$$

onde:

$I_{max}$ é a máxima intensidade desejada na imagem

$I_{min}$ é a mínima intensidade desejada na imagem

$d_{max}$ é a maior distância da superfície do objeto ao plano de visão (o maior valor do *z-buffer*)

$d_{min}$ é a menor distância da superfície do objeto ao plano de visão (o menor valor do *z-buffer*)

$d(u,v)$ é a distância do pixel $(u,v)$ à superfície do objeto (o valor contido na posição $(u,v)$ do *z-buffer*)
$I(u,v)$ é o nível de cinza associado ao pixel $(u,v)$
$K_a$ é o coeficiente de reflexão para a luz ambiente
$K_d$ é o coeficiente de reflexão difusa
$K_s$ é o coeficiente de reflexão especular
$\theta$ é o ângulo entre o vetor normal à superfície e o raio de incidência. Devem ser considerados apenas os valores de $\theta$ entre 0 e 90 graus para a componente difusa, e valores entre 0 e 45 graus para a componente especular.

A equação 2.13 pode ser vista como a equação de Phong [6] modificada. A figura 2.22 mostra as quatro componentes principais da equação 2.13 atuando separadamente. Um resultado completo pode ser visto na figura 3.16. Os valores dos parâmetros da equação 2.13 são escolhidos empiricamente até a obtenção de um resultado satisfatório. Os coeficientes de reflexão da superfície ($K_d$, $K_s$ e $K_a$) devem apresentar valores entre 0 e 1, e o expoente $n$ um valor no mínimo 0. Os cossenos dos ângulos $\theta$ e $\alpha$ são obtidos pelo produto interno entre os vetores normalizados das respectivas direções. A direção normal à superfície em cada ponto é representada por uma estimativa do vetor normal. Devido sua influência na imagem resultante, diferentes técnicas de estimar o vetor normal são vistas apenas no capítulo 3, junto aos métodos de visualização baseados em voxel binário (seção 3.3.2).

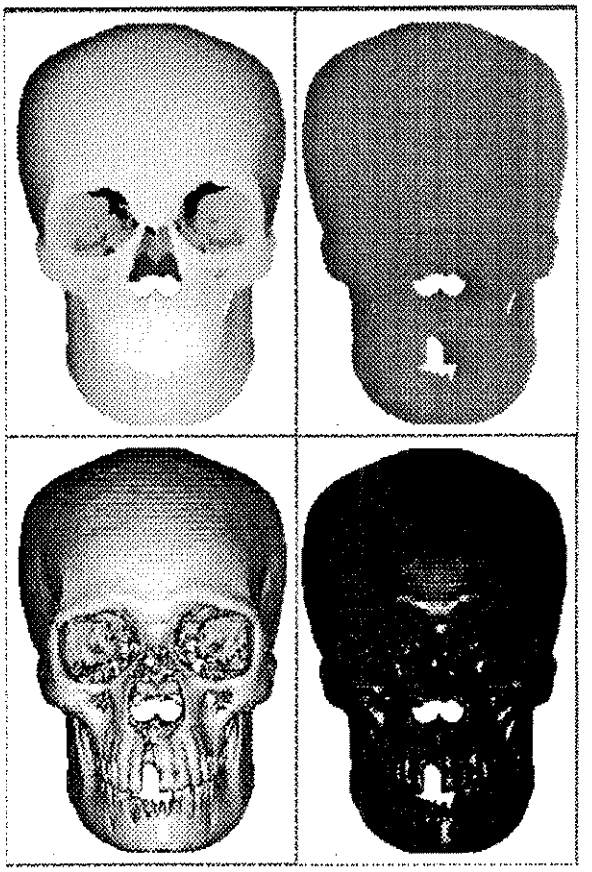

Figura 2.22: Componentes principais da tonalização: o efeito da variação da intensidade da fonte com a distância no canto superior esquerdo, o efeito da luz ambiente no canto superior direito, o efeito da reflexão difusa no canto inferior esquerdo e o efeito da reflexão especular no canto inferior direito.

# Capítulo 3

# Métodos Fundamentais

## 3.1 Introdução

O objetivo deste capítulo é apresentar uma classificação para as principais técnicas de visualização e quais as operações do fluxo de dados (figura 2.5) que caracterizam estas técnicas. O fluxo de dados apresenta duas soluções alternativas para o problema do espaço voxel pré-processado não apresentar nenhuma manifestação visível das estruturas no seu interior: classificação dos dados, ou extração e modelagem. Em ambas soluções, uma representação intermediária é criada baseada em pontos, linhas, gels, cubos, superfícies, etc. Os métodos fundamentais de visualização podem ser classificados de acordo com o tipo de representação intermediária [59] [27]. Estes métodos são divididos em técnicas baseadas em superfície, técnicas baseadas em voxel binário e técnicas de *rendering* de volume semitransparente. As técnicas baseadas em superfície e em voxel binário, usam extração e modelagem, e as técnicas de *rendering* de volume semitransparente usam a classificação dos dados.

As técnicas baseadas em superfície procuram aproximar o formato real da superfície utilizando a interpolação de superfícies por primitivas geométricas (seção 2.7.5). As primitivas geométricas podem ser polígonos ou superfícies curvas, e a superfície do objeto é modelada por um conjunto destes elementos (figura 2.5). As técnicas baseadas em voxel binário representam o volume do objeto por um conjunto de cubos opacos (modelo de volume binário), ou sua superfície pelos polígonos que definem estes cubos (modelo de

superfície cubiculada). As técnicas de *rendering* de volume semitransparente associam uma cor e uma opacidade parcial a cada voxel do espaço pré-processado, e cada pixel no plano de visão é função de uma composição das opacidades e cores dos voxels que estão sobre a reta de projeção. O resultado é uma imagem com a aparência de um gel colorido e semitransparente.

Neste capítulo são apresentados inicialmente os principais métodos de visualização de volumes, suas vantagens e desvantagens, dando ênfase às técnicas de computação gráfica que diferenciam os algoritmos em cada método. Nas técnicas baseadas em superfície são descritas a terminologia, as principais dificuldades e soluções encontradas na interpolação de superfícies (seção 3.2). Nas técnicas baseadas em voxel binário são apresentados outros algoritmos de projeção com remoção de superfícies escondidas e as diferentes formas de estimar o vetor normal para aproximar o formato real da superfície do objeto, caracterizando diferentes técnicas de tonalização (seção 3.3). Nas técnicas de *rendering* de volume semitransparente são discutidas as operações envolvidas na classificação dos dados e como é feito o *rendering* do volume de dados colorido e semitransparente (figura 2.5). No final do capítulo são apresentados os métodos interativos (seção 3.5), que apresentam resultados bem mais rápido que os métodos fundamentais. Estes resultados são limitados, mas representam informações complementares, e muitas vezes suficientes, à informação tridimensional obtida com os métodos fundamentais. Estes métodos envolvem técnicas de refatiamento, técnicas de reprojeção e o uso de pseudo-cores.

## 3.2 Técnicas Baseadas em Superfície

Técnicas baseadas em superíficie reconstroem a superfície a partir de contornos planares, usando o modelo de contornos aramados (figura 2.17), ou a partir do espaço voxel pré-processado (figura 2.5). As duas principais diferenças entre estas técnicas são a escolha das primitivas geométricas (normalmente triângulos) e a escala que as define [27]. Uma das principais vantagens destes métodos é a utilização dos algoritmos já desenvolvidos em *hardware* e *software* para visualização destas superfícies. As primitivas geométricas são compactas (tornando econômico o armazenamento e a transmissão dos dados) e possuem um alto grau de coerência espacial (tornando eficiente as operações de *rendering*). Contudo,

a construção automática de uma superfície genérica raramente tem sucesso. Os algoritmos que unem os contornos de planos consecutivos, por exemplo, têm dificuldades para resolver ambiguidades em situações de múltiplos contornos por plano, exigindo a intervenção do usuário. Estas técnicas também exigem uma classificação binária [1] do espaço voxel pré-processado; a superfície passa ou não passa através do voxel em questão; podendo gerar superfícies inexistentes no objeto real (falsas positivas) e/ou esconder superfícies reais (falsas negativas) na imagem final.

### 3.2.1 Reconstrução da Superfície a partir de Contornos Planares

Na reconstrução da superfície a partir de contornos planares, o objeto é normalmente representado por um poliedro de faces triangulares cujos vértices são pontos dos contornos planares. O problema geral de reconstrução da superfície encara estes pontos como um conjunto S de n pontos distibuídos arbitrariamente no espaço sem nenhuma informação de conectividade entre eles. A solução deste problema, segundo Ekoule et al. [2], foi proposta por Hermeline [20] e Boissonnat [36]. Ambos os métodos baseiam-se na triangulação *Delaunay* para resolver o problema em um espaço n-dimensional. Um outro método desenvolvido por O'Rourke [34] envolve o uso de um procedimento interativo para modificar o *convex hull* [2] do conjunto convexo S e obter uma superfície poliédrica de área mínima. Posteriormente, Boissonnat [37] propôs um método também baseado na triangulação *Delaunay*, mas utilizando a informação de conectividade entre os pontos de um mesmo contorno para tratar casos de objeto com múltiplos contornos por plano, onde o número de contornos varia entre planos adjacentes, múltiplos objetos e objetos com buracos. Os planos (ou fatias) não necessariamente precisam representar cortes paralelos, mas os contornos precisam ser orientados e fechados definindo claramente a parte interna e externa de cada objeto. Diferente dos métodos descritos abaixo, o algoritmo de Boissonnat reconstrói o volume do objeto de interesse e não sua superfície. O volume do objeto é representado por uma malha de tetraedros e sua superfície pode ser extraída formando um poliedro cujas faces são faces triangulares dos tetraedros.

[1] A classificação binária não gera necessariamente um espaço voxel binário, pois o tom de cinza dos voxels que pertencem à superfície pode ser mantido.

[2] O *convex hull* de um conjunto S de pontos distribuídos arbitrariamente no espaço é formado pelo menor poliedro convexo que contém todos os pontos de S.

Devido à complexidade normalmente encontrada no problema de triangulação aplicado à área médica, um grande número de algoritmos têm sido propostos nos últimos dezessete anos. Infelizmente nenhum deles foi ainda capaz de tratar todos os possíveis casos automaticamente. Mesmo com a intervenção do usuário, os algoritmos exigem para casos mais complexos um conhecimento *a priori* sobre a natureza da estrutura em questão, tornando-se lentos e trabalhosos. Os principais algoritmos propostos seguem mais ou menos uma mesma terminologia e um conjunto de regras gerais, encontram as mesmas dificuldades, e propõem soluções limitadas e diferentes para o problema.

#### 3.2.1.1 Terminologia e Regras Gerais

O problema mais simples consiste em aproximar a superfície do objeto de interesse por um poliedro de faces triangulares, quando existe apenas um contorno fechado por plano e os planos são paralelos (figura 3.1a). Cada face triangular é formada por um segmento de contorno [3] e duas *spans* (diagonais esquerda e direita) conectando os pontos extremos do segmento de contorno com um ponto comum no contorno adjacente. O processo de triangulação é feito para cada dois planos consecutivos respeitando as duas restrições propostas por Fuchs et al. [28] para construção de uma superfície aceitável:

1. Cada segmento de contorno deve aparecer em exatamente um triângulo. Se um contorno possui $m$ pontos e o outro $n$ pontos, a superfície poliédrica formada entre estes dois contornos possuirá um conjunto de $m + n$ faces triangulares [74].
2. Se uma aresta de um triângulo do conjunto aparece como diagonal esquerda (direita), ela deve aparecer como diagonal direita (esquerda) de exatamente um outro triângulo do conjunto garantindo uma superfície fechada.

A figura 3.1b ilustra os dois casos acima que não devem ocorrer no processo de triangulação. As diagonais $\overline{P_0Q_1}$ e $\overline{P_1Q_2}$ infringem o caso 2 e os triângulos $P_{m-1}P_0Q_4$ e $P_{m-1}Q_1P_0$ infringem o caso 1.

[3] O segmento de contorno é uma aproximação linear da curva que conecta dois pontos consecutivos do contorno [28].

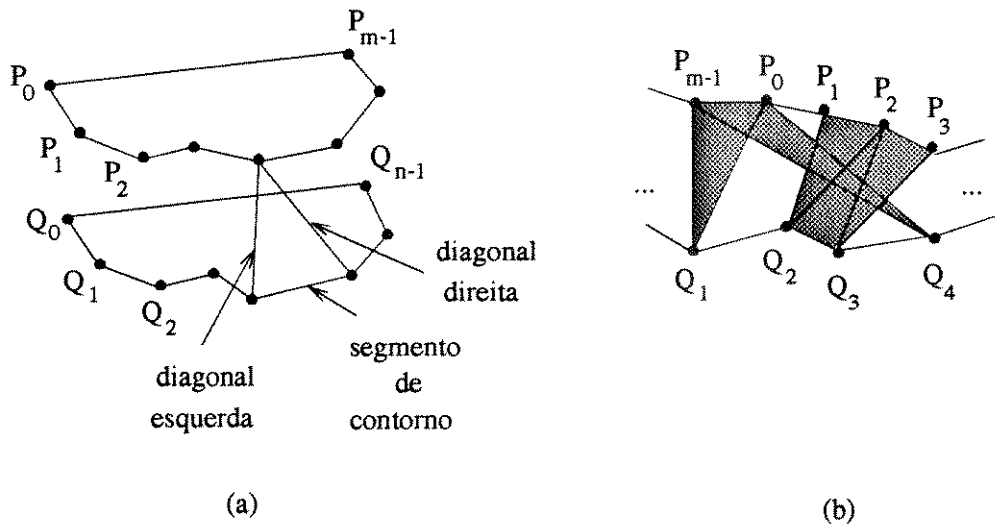


Figura 3.1: a)Polígono elementar b)Duas situações que não devem ocorrer na formação de uma superfície aceitável.

O problema básico da triangulação consiste no fato dos contornos não conterem informação suficiente a respeito do sistema de gradientes associado à superfície que eles descrevem [16]. O aspecto combinatorial do problema torna-se evidente quando são considerados os possíveis arranjos de triângulos que podem ser usados na formação de uma superfície aceitável. O conjunto de todas as superfícies aceitáveis que podem ser utilizadas para um par de contornos $P_0, P_1, \cdots, P_{m-1}$ e $Q_0, Q_1, \cdots, Q_{n-1}$ (figura 3.1a) pode ser representado por um grafo direcionado G[V,A] [28] (figura 3.2). O conjunto de vértices $V = \{v_{ij} | i = 0, 1, 2, \cdots, m-1; j = 0, 1, 2, \cdots, n-1\}$ corresponde o conjunto de todas as possíveis diagonais $\overline{P_i Q_j}$ entre o contorno superior e o contorno inferior. O conjunto de arcos $A = \{< v_{kl}, v_{st} > | ((s = k) \wedge (t = l + n^1)) \vee ((s = k + m^1) \wedge (t = l))\}$ corresponde a todas possíveis faces triangulares. Um arco $< v_{kl}, v_{st} >$ é sempre incidente do vértice que representa a diagonal esquerda $\overline{P_k Q_l}$ no vértice que representa a diagonal direita $\overline{P_s Q_t}$ deste triângulo. Os arcos horizontais $< v_{ij}, v_{i(j+n^1)} >$ unem as colunas $j$ e $j + n^1$, e os arcos verticais $< v_{ij}, v_{(i+m^1)j} >$ unem as linhas $i$ e $i + m^1$.

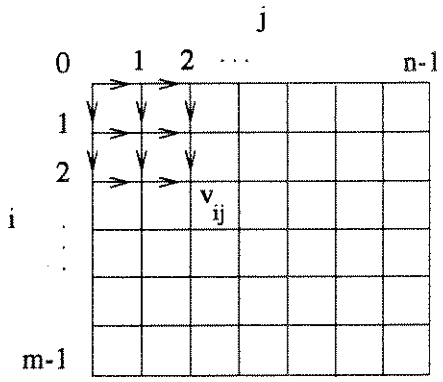


Figura 3.2: Grafo direcionado.

Um conjunto qualquer de faces triangulares é representado por um caminho S no grafo G, chamado subgrafo. Um subgrafo S corresponde uma superfície aceitável se e somente se (1) S contém exatamente um arco horizontal entre quaisquer duas colunas adjacentes e exatamente um arco vertical entre quaisquer duas linhas adjacentes e (2) S é *eulerian* [4] (euleriano) (*Teorema 1* do Fuchs et al. [28]). A figura 3.3 ilustra um exemplo de superfície aceitável para um cilindro.

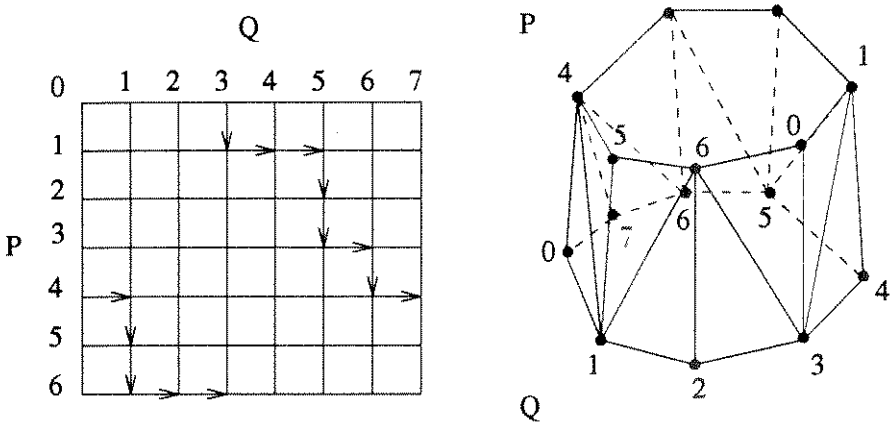


Figura 3.3: Superfície aceitável para um cilindro.

[4] Um grafo direcionado é euleriano se, e somente se, ele pode ser percorrido por um caminho fechado no qual todo arco do grafo ocorre exatamente uma vez.

#### 3.2.1.2 Dificuldades Encontradas

O número de superfícies aceitáveis para o par de contornos da figura 3.1a, por exemplo, é uma função A[m,n] representada pela seguinte equação [74]:

$$A[m,n] = \frac{[(m-1)+(n-1)]!}{(m-1)!(n-1)!} \tag{3.1}$$

onde m e n são o número de pontos em cada contorno, A[m,1]=1 $\forall$ m$\in N^*$, e A[1,n]=1 $\forall$ n$\in N^*$ . A expressão demonstra que para um número relativamente pequeno de pontos, o número de possíveis combinações de superfícies aceitáveis é muito grande. Para n=m=10 pontos, por exemplo, tem-se aproximadamente $5 \times 10^4$ superfícies de formatos diferentes. Considerando $\varphi : A \rightarrow R$ uma função-custo que mapeia o conjunto de faces triangulares em números reais, associa-se a cada arco $a$ um peso $\varphi_a$. O caminho de uma superfície aceitável tem exatamente $m+n$ arcos $a_1, a_2, \cdots, a_{m+n}$. O caminho escolhido é tal que a função $\phi = \sum_{k=1}^{m+n} \varphi_{ak}$ satisfaz algum critério predefinido (local ou global). Devido o grande número de possíveis superfícies aceitáveis, um método para selecionar a superfície mais adequada para o objeto de interesse exige normalmente uma pesquisa exaustiva tornando os algoritmos lentos.

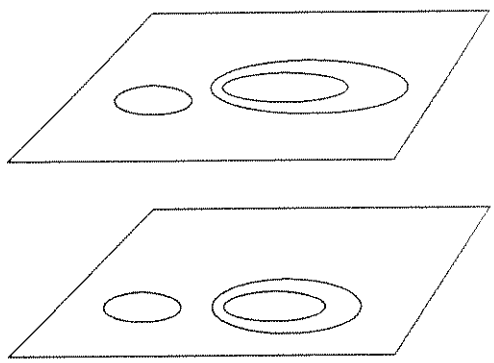

Figura 3.4: Situação de múltiplos contornos por fatia.

As dificuldades comumente encontradas no processo de triangulação de estruturas mais complexas são situações de múltiplos contornos por fatia (figura 3.4) (objeto com buracos, ramificações, ou múltiplos objetos de interesse), mapeamento parcial de contornos (figura 3.5) ou contornos abertos, e contornos descentralizados, com formatos e tamanhos não similares, e não totalmente convexos (figura 3.6). Estas dificuldades geram ambiguidades que nem sempre podem ser resolvidas automaticamente, requerendo a intervenção do

usuário no processo.

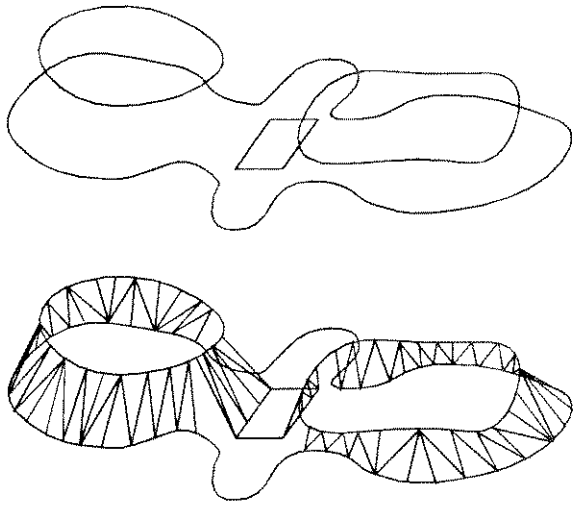

Figura 3.5: Situação de mapeamento parcial entre contornos.

#### 3.2.1.3 Soluções Propostas

A metodologia utilizada pelos algoritmos de reconstrução de superfície a partir de contornos planares pode ser classificada em duas linhas básicas: métodos ótimos e métodos heurísticos [74]. Os métodos ótimos são os que otimizam alguma característica da superfíce gerada. Um critério de otimização para a função-custo $\phi = \sum_{k=1}^{m+n} \varphi_{ak}$ é definido para encontrar o caminho ótimo no grafo G. Estes métodos normalmente geram bons resultados, mas consomem bastante tempo de processamento. Os métodos heurísticos são os que constroem a superfície baseados em critérios locais e heurísticos de decisão. A cada passo do algoritmo escolhe-se um arco $a_k$ baseado nos valores locais $\varphi_{ak}$ dos possíveis arcos. Estes métodos são bem mais rápidos e são considerados aproximações razoáveis do caso ótimo por apresentarem superfícies visualmente corretas.

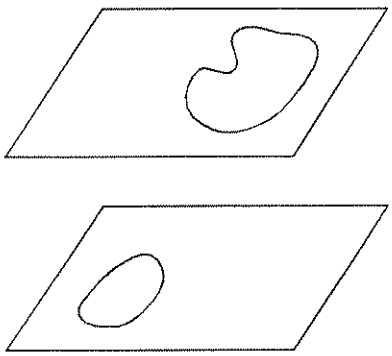

Figura 3.6: Contornos dissimilares em forma e tamanho, e descentralizados.

**3.2.1.3.1 Métodos Ótimos** A primeira solução proposta foi a de Keppel [16]. Keppel definiu a função-custo $\phi$ como o volume envolvido por uma superfície aceitável e seu critério de otimização maximiza esta função. O algoritmo trata pares de contornos fechados maximizando o volume das partes convexas e minimizando o volume das partes côncavas. Superfícies definidas desta forma têm muitas aplicações, particularmente nos campos da terapia de radiação e cirurgia plástica [74]. Posteriormente, Fuchs et al. [28] definiu a função-custo $\phi$ como a área de uma superfície aceitável e seu critério de otimização minimiza esta função. Os algoritmos de Fuchs et al. e de Keppel fazem uma triangulação completa, contorno a contorno, entre contornos adjacentes admitindo um contorno fechado por fatia. Portanto, estes algoritmos não são capazes de tratar casos mais complexos de múltiplos contornos, mapeamento parcial e contornos abertos, por exemplo. Segundo Zyda et al. [61], os métodos que tratam mapeamento parcial também são capazes de tratar contornos abertos.

**3.2.1.3.2 Métodos Heurísticos** Influenciado pelos problemas encontrados por Fuchs et al. e Keppel, Christiansen e Sederberg [33] introduziram um método heurístico capaz de tratar automaticamente situações de simples ramificações entre planos adjacentes (figura 3.7a). O algoritmo introduz um novo nó no meio da ramificação entre os dois pontos mais próximos, um em cada contorno coplanar, com coordenada z igual a média das coordenadas z dos planos adjacentes (figura 3.7b). O algoritmo também permite um procedimento interativo com o usuário para tratar situações de múltiplos contornos um pouco mais complexas. Apesar do algoritmo funcionar melhor para situações de triangulação de

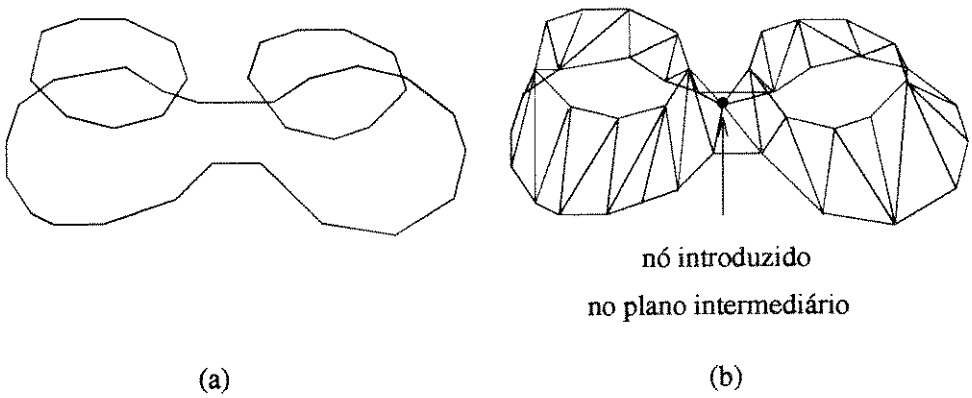


Figura 3.7: a)Ramificação simples b)Triangulação com introdução de um nó intermediário.

pares de contornos semelhantes em forma e tamanho, e mutualmente centralizados, os pares de contornos não satisfazendo este critério podem ser mapeados em um quadrado unitário com centro na origem, e o algoritmo é então aplicado aos contornos normalizados [74]. A heurística do método de Christiansen e Sederberg é a escolha da "diagonal mais curta" na formação dos triângulos: partindo de uma diagonal inicialmente escolhida, o peso $\varphi_a$ de um arco $a$ é definido como o comprimento da diagonal que forma um triângulo com a diagonal inicial. O triângulo (ou arco) escolhido é o de menor peso. O algoritmo repete o procedimento partindo da nova diagonal escolhida até a diagonal inicial fazer parte de um novo triângulo. A limitação deste algoritmo é sua incapacidade para tratar contornos abertos, mapeamento parcial e casos mais complexos de múltiplos contornos, exceto através de um alto custo de interação com o usuário [61]. Posteriormente, Ganapathy e Dennehy [74] propuseram uma heurística na qual o critério de decisão local não cria dependência entre os contornos e é influenciado pelas decisões anteriormente tomadas (não hiterto) permitindo, por exemplo, tratar casos de planos de contornos não paralelos. Entretanto, este algoritmo não provê interação com o usuário, não trata casos de múltiplos contornos, mapeamento parcial e contornos abertos. A heurística utilizada é a das "diferenças absolutas" onde o peso $\varphi_a$ de um arco $a$ é definido como o comprimento do segmento de contorno do respectivo triângulo dividido pelo perímetro do contorno ao qual o segmento pertence. Triângulos são acrescentados à superfície de forma tal que a diferença absoluta entre a soma dos pesos dos arcos horizontais e a soma dos pesos dos arcos verticais seja mínima a cada passo do algoritmo (figura 3.8): entre $P_jPj+1Q_i$ e $P_jQ_iQ_{i+1}$, por exemplo, $P_jPj+1Q_i$ é escolhido se $|\phi_{h'}+\varphi_h-\phi_{v'}| < |\phi_{v'}+\varphi_v-\phi_{h'}|$. Quando uma superfície aceitável é encontrada esta

diferença é nula, mas a recíproca não é verdadeira.

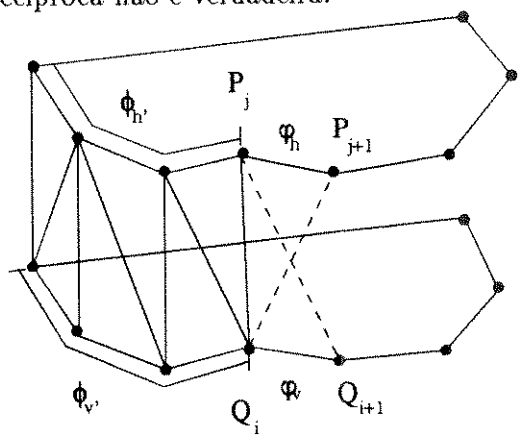


Figura 3.8: Heurística das diferenças absolutas.

Segundo Zyda et al. [61], baseado nos métodos de Fuchs et al., e Christiansen e Sederberg, Shantz [60] propôs um algoritmo capaz de tratar uma situação genérica de m e n contornos em planos adjacentes provocada por objetos altamente ramificados e com buracos. O algoritmo procura inicialmente concatenar os contornos de cada plano isoladamente em um único contorno. Após esta etapa usa o mecanismo de Fuchs et al. para a triangulação de pares de contornos fechados sobre os contornos concatenados e, após a formação da superfície, o algoritmo remove as conecções que não pertencem ao objeto reconstruído. As ambiguidades durante o processo são resolvidas interativamente de forma similar ao método de Christiansen e Sederberg. Sua principal limitação é a incapacidade de tratar contornos abertos e mapeamento parcial. O método de Zyda et al. [61], entretanto, é capaz de tratar mapeamento parcial e contornos abertos, e também provê interação com o usuário para tratar as ambiguidades durante o processo. Este algoritmo trata situações de múltiplos contornos, mas no caso particular de ramificações com um contorno em um plano e dois ou mais contornos no plano adjacente, o algoritmo apenas sugere a solução descrita por Christiansen e Sederberg.

A necessidade de intervenção do usuário para resolver problemas mais complexos de múltiplos contornos e os erros na aproximação da superfície provocados em casos de contornos não convexos ou pares de contornos não similares, são problemas comumente encontrados nos algoritmos até agora descritos. Influenciado por estes problemas, Ekoule et al. [2] propôs um método heurístico para triangulação da superfície externa de um objeto

definido por um conjunto de contornos planares paralelos, que é capaz de tratar desde um caso de simples ramificação (figura 3.7a), cujos contornos têm formatos diferentes [5], até uma situação com qualquer número de contornos por plano, operando automaticamente. Os algoritmos e arquiteturas utilizados para fazer o *rendering* destes modelos são clássicos e podem ser encontrados nos livros básicos de computação gráfica [38] [13].

### 3.2.2 Reconstrução da Superfície a partir do Espaço Voxel Pré-processado

Duas técnicas interessantes de estimar a superfície sem precisar extrair contornos, ou gerar um espaço voxel binário, foram descritas por Lorensen e Cline [85], e Cline [31]: *marching cubes* e *dividing cubes*. Estas técnicas utilizam em cada etapa do processamento, para cada duas fatias adjacentes do espaço voxel pré-processado (figura 2.5), todos os cubos formados por cada oito voxels vizinhos; quatro na fatia superior e quatro na inferior (figura 2.12); para estimar a forma que a superfície do objeto passa, ou não, por eles. Operações boleanas (interseção, união e subtração) podem ser usadas, estendendo a versatilidade dos algoritmos. *Marching cubes* é bastante utilizado na área de visualização científica. Uma aplicação para a visualização de dados metereológicos pode ser vista em Battaiola [3]. Battaiola apresenta um algoritmo vetorizado para a geração de superfícies tridimensionais. Superfícies bi-cúbicas também podem ser utilizadas como primitivas geométricas [73].

#### 3.2.2.1 *Marching Cubes*

*Marching cubes* cria uma lista de triângulos representando a superfície e, portanto, o *rendering* da superfície resultante pode ser feito utilizando os algoritmos e arquiteturas convencionais da mesma forma que as técnicas descritas na seção anterior. Os triângulos não necessariamente representam superfícies conectadas, porém algumas extensões deste algoritmo procuram conectar estes triângulos. Esta técnica representa os dados em ponto flutuante e é lenta para aplicações médicas típicas, o que a torna mais eficiente apenas para espaços voxels de baixa resolução [59]. O algoritmo *marching cubes* consta das seguintes etapas:

[5] Mesmo os contornos tendo formatos diferentes, o algoritmo admite que os *convex hull* dos contornos têm formatos similares.

1. Formação de um cubo usando os oito voxels vizinhos como vértices; quatro na fatia $k$ e quatro na fatia $k+1$ (figura 3.9a); o qual tem seus vértices e arestas rotulados conforme a figura 3.9b, por exemplo.

2. Um número inteiro de 0 a 255 é associado ao cubo onde cada um dos oito dígitos binários do número está associado a um vértice do cubo. Um vértice é classificado com dígito 1, por exemplo, se sua densidade é maior ou igual a um valor de densidade constante para a superfície, previamente escolhido pelo usuário [6], significando que o vértice pertence ao interior ou a superfície do objeto. Um vértice é classificado com dígito 0 no caso contrário, significando que pertence ao exterior do objeto (figura 3.10). A parte da superfície estimada é subdividida em triângulos cujos vértices se encontram sobre as arestas do cubo que têm um vértice no interior e o outro no exterior do objeto. Existem, portanto, 256 possíveis casos, mas uma tabela com 14 casos [7] é suficiente para gerar através de operações de simetria rotacional e complementariedade os 256 casos (figura 3.11).

3. Uma lista de arestas do cubo, que contêm os vértices dos triângulos, é obtida utilizando o número associado ao cubo para acessar a tabela de possíveis casos (Ex: figura 3.12 - caso 9).

4. Para cada aresta da lista, a coordenada de um dos vértices do triângulo que a intercepta pode ser encontrada por interpolação linear das densidades nos vértices da aresta (Ex: figura 3.13a): $x = i + \frac{(v_s - f(i))}{(f(i+1) - f(i))}$, onde $x$ é a coordenada no eixo x do vértice do triângulo, $v_s$ é o valor constante escolhido para a superfície, $i$ e $i+1$ são as coordenadas no eixo x dos vértices da aresta, e $f(i)$ e $f(i+1)$ são suas respectivas densidades associadas. O vetor normal à superfície em cada vértice dos triângulos pode ser estimado, por exemplo, como o vetor gradiente normalizado. Para cada vértice (i,j,k) do cubo calcula-se o vetor gradiente usando os seis voxels vizinhos face a face (figura 2.9b) pelo método das diferenças centrais [9] ao longo dos três eixos principais:

$$G_x(i,j,k) = \frac{f(i+1,j,k) - f(i-1,j,k)}{2}$$

[6] É importante lembrar que no conteúdo deste trabalho a frase *valor de densidade constante para a superfície* não significa o valor de densidade do material, mas o número que pode estar associado aos voxels do espaço voxel cortados pela superfície.

[7] Dependendo da combinação de casos entre cubos vizinhos e do ângulo de visão, buracos podem aparecer na visualização da superfície. Este erro conceitual foi apontado e resolvido por Battaiola [3].

$$G_y(i,j,k) = \frac{f(i,j+1,k) - f(i,j-1,k)}{2}$$

$$G_z(i,j,k) = \frac{f(i,j,k+1) - f(i,j,k-1)}{2} \quad (3.2)$$

onde $f(i,j,k)$ é a densidade do voxel (i,j,k) e $\Delta_x, \Delta_y, \Delta_z$ são as distâncias entre os voxels vizinhos do voxel (i,j,k) nas respectivas direções [8]. Os vetores gradientes são normalizados e interpolados para estimar o vetor normal nos vértices dos triângulos (figura 3.13b).

5. A saída do algoritmo é a coordenada de todos os vértices do poliedro de faces triangulares e suas respectivas normais. O vetor normal a cada vértice pode ser utilizado no *rendering* da superfície para produzir uma imagem com tonalização de Gouraud [38]. Alternativamente, para qualquer modelo que utiliza uma superfície poliédrica de faces poligonais, o vetor normal a um vértice pode ser calculado posteriormente a formação da superfície interpolando os vetores normais aos polígonos das faces que compartilham o vértice.

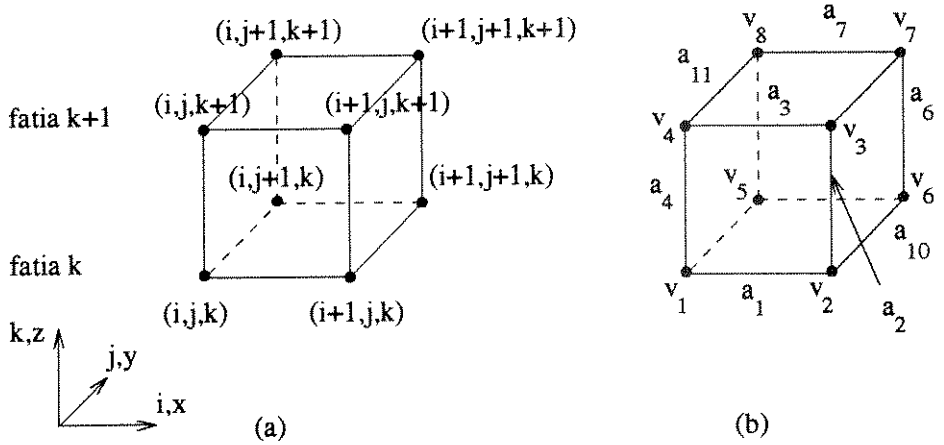


Figura 3.9: a)Criação do cubo b)Criação de rótulos para os vértices e arestas do cubo.

[8] Nota-se que para calcular o vetor gradiente em todos os vértices do cubo é necessário ter na memória quatro fatias consecutivas.

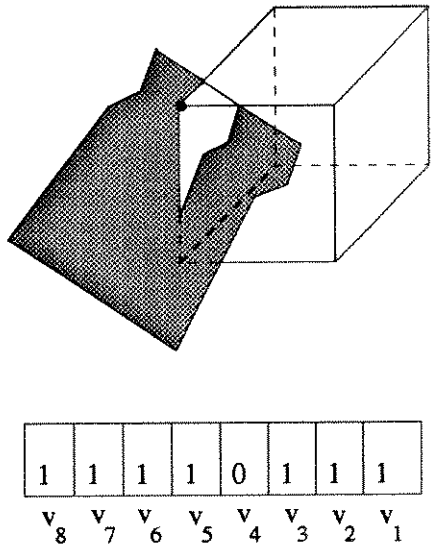


Figura 3.10: Classificação dos vértices e criação de um *index*

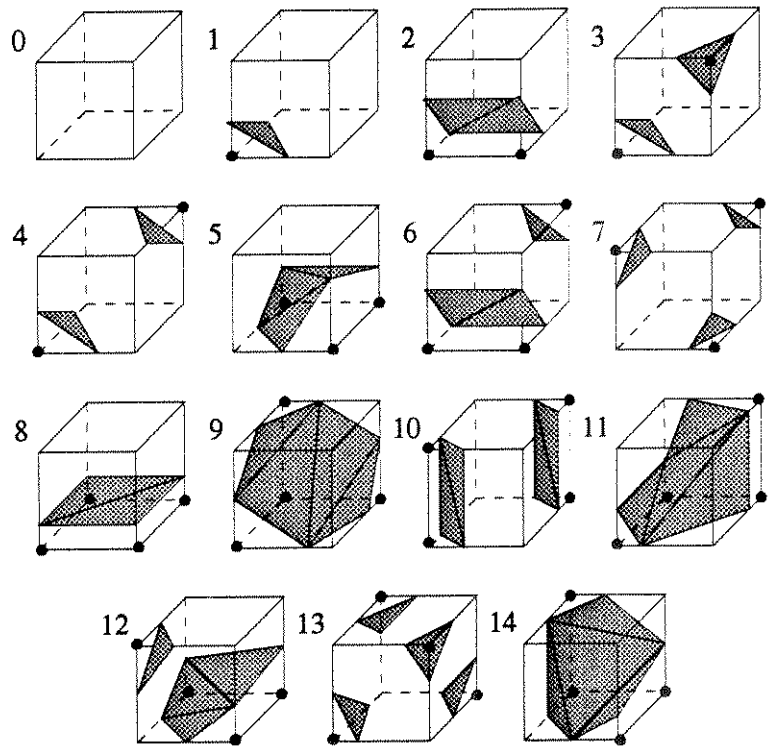


Figura 3.11: Tabela de possíveis casos.

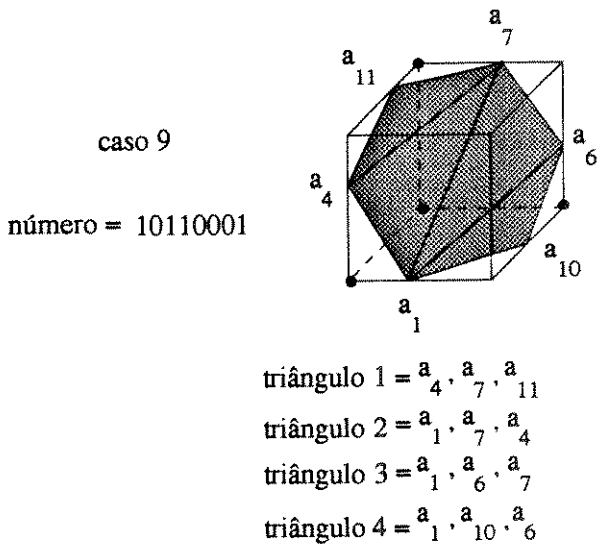


Figura 3.12: Obtenção de uma lista de arestas que contém os vértices dos triângulos.

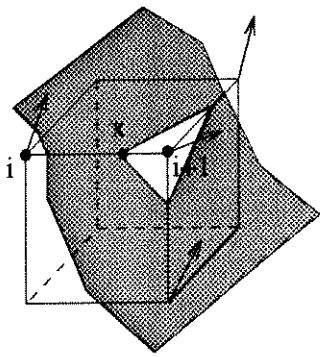


Figura 3.13: a)Cálculo da densidade em cada vértice dos triângulos b)Cálculo do vetor normal em cada vértice dos triângulos.

#### 3.2.2.2 *Dividing Cubes*

*Dividing cubes* cria uma lista de pontos representando amostras de pequenas áreas da superfície com uma estimativa de suas respectivas normais. Neste caso, o *rendering* da superfície gerada pode ser feito similarmente à forma descrita na próxima seção. Esta técnica representa os dados em ponto fixo e é rápida para aplicações médicas típicas, o que a torna eficiente também para espaços voxels de alta resolução [59]. A conectividade entre os pontos também fica a critério do algoritmo. O *rendering* da superfície pode ser feito, por

exemplo, usando o algoritmo *z-buffer* para projeção dos pequenos cubos com remoção de superfícies escondidas, conforme descrito na seção 2.8.2, e o vetor normal pode ser usado na equação 2.13 para calcular a tonalização do pixel correspondente. O algoritmo *dividing cubes* envolve as seguintes etapas:

1. Formação do cubo conforme a primeira etapa do *marching cubes*, porém não utiliza rótulos para os vértices e arestas.
2. Classificação de cada vértice do cubo comparando sua densidade com um valor de densidade constante previamente escolhido para a superfície. Três casos podem ocorrer: todos os vértices têm valor de densidade acima do valor da superfície significando, por exemplo, que o cubo está no interior do objeto, ou todos os vértices têm valor de densidade abaixo do valor da superfície (o cubo está no exterior do objeto), ou nenhum dos dois casos anteriores significando que a superfície corta o cubo.
3. Se a superfície passar pelo cubo, subdivide-se o cubo nas direções principais x, y e z em pequenos outros cubos de forma que a resolução do espaço voxel final em cada uma das seis faces principais coincida com a resolução da imagem final. Se a resolução do espaço voxel é, por exemplo, $256 \times 256 \times 128$ voxels e a resolução da imagem desejada é $512 \times 512$ pixels, subdivide-se 2 vezes em x e y, e 4 vezes em z (figura 3.14). Calcula-se as densidades dos novos cubos por interpolação trilinear das densidades nos vértices do cubo maior. Para cada pequeno cubo, repete-se a etapa 2 para verificar quais os pequenos cubos cortados pela superfície. Para os pequenos cubos cortados pela superfície estima-se a normal como o vetor gradiente calculado usando a equação 3.2 e posteriormente normalizado.
4. A saída do algoritmo é a coordenada de todos os centros dos pequenos cubos e suas respectivas normais.

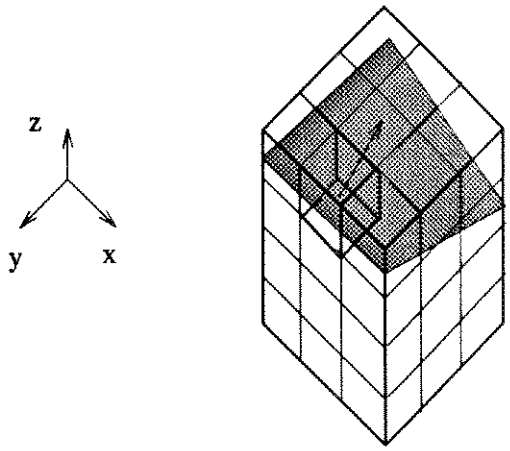


Figura 3.14: Algoritmo *dividing cubes*.

## 3.3 Técnicas Baseadas em Voxel Binário

Técnicas baseadas em voxel binário normalmente utilizam a segmentação baseada na escolha de valores de limiar (seção 2.7.1.1) para gerar o espaço voxel binário (espaço objeto). Segundo Udupa [39], imagens de boa qualidade podem ser obtidas utilizando a seguinte ordem para as operações do fluxo de dados da figura 2.5: filtro mediano nas operações de pré-processamento (seção 2.6.3.3), segmentação gerando o espaço voxel binário (seção 2.7.1) e interpolação no espaço objeto (seção 2.7.6). Em seguida, *surface-tracking methods* podem ser utilizados para extrair uma superfície conectada gerando o modelo de superfície cubiculada. A superfície conectada facilita a extração de medidas e a manipulação do modelo. O modelo de superfície cubiculada representa o objeto como uma casca oca, impossibilitando a visualização de seu interior, através de cortes no modelo, mas dependendo da estrutura de dados escolhida, pode tornar mais rápida a execução das operações para visualização e, em alguns casos, representar boa economia de memória.

As principais vantagens das técnicas baseadas em voxel binário é a facilidade de manipulação com os dados para extrair medidas quantitativas (distância, área e volume) e a rapidez no *rendering* de seus modelos. A desvantagem está em exigir a classificação binária do espaço voxel pré-processado, tornando-as vulneráveis aos mesmos problemas encontrados nas técnicas baseadas em superfície [27]. As características mais importantes

que diferenciam os algoritmos baseados em voxel binário estão nas diferentes formas de projeção com remoção de superfícies escondidas e de tonalização. Escolhidos o tipo de projeção e de tonalização, o *rendering* de seus modelos é similar, gerando imagens de mesma qualidade.

### 3.3.1 Projeção com Remoção de Superfícies Escondidas

Conforme descrito no capítulo 2 (seção 2.8.2.2), existem duas formas de projeção dos modelos baseados em voxel: projeção de voxels e *raycasting* (figura 2.19). A principal diferença entre os algoritmos que usam *voxel projection* está no tratamento de superfícies escondidas. Os mais comuns são *depth-sort*, *z-buffer*, *back-to-front* (BTF) e *front-to-back* (FTB). Os algoritmos *depth-sort* e *z-buffer* são clássicos na área de computação gráfica e podem ser utilizados em ambos os modelos. Os algoritmos BTF e FTB são comuns apenas para o modelo de volume binário. O algoritmo *depth-sort* classifica os elementos do espaço voxel de acordo com a distância deles ao plano de visão e depois projeta-os no espaço imagem na ordem em que a distância decresce; do mais afastado para o mais próximo do observador. Este processo de classificação torna-o mais lento e um pouco mais complexo que o algoritmo *z-buffer* descrito na seção 2.8.2.2. O algoritmo BTF [21] explora a informação de proximidade com o observador implícita na estrutura do modelo de volume binário (linha por linha, fatia por fatia), evitando a classificação de distâncias. O algoritmo acessa, após a rotação, os voxels linha por linha, fatia por fatia, na ordem em que eles se aproximam do observador, fazendo com que os mais próximos sejam projetados no espaço imagem sobre os mais afastados. O algoritmo FTB [12] baseia-se na mesma idéia, mas acessa os elementos do modelo de volume binário, linha por linha, fatia por fatia, na ordem em que eles se afastam do observador e, portanto, quando é feita a projeção de um voxel no espaço imagem, nenhum outro elemento pode ser projetado no mesmo pixel (ou região de pixels) associado ao voxel. Stytz et al. [62] cita outras técnicas de projeção com remoção de superfícies escondidas que visam acelerar o processo acima, reduzindo o tempo de acesso aos voxels e/ou reduzindo o número de voxels examinados [78] [69] [70]. Um problema encontrado nas técnicas que usam projeção de voxels é que alguns pixels no espaço imagem podem não ser preenchidos, devido a problemas de erros de truncamento e precisão aritmética, ficando com a cor do fundo, o que acusa a existência de pequenos "buracos" na imagem final. Uma forma de corrigir

este problema é calcular a tonalização destes pixels por interpolação da tonalização de seus vizinhos. Na técnica *raycasting* [32] este problema não ocorre, porque necessariamente existe um raio lançado a partir de cada pixel do espaço imagem. Esta técnica também elimina naturalmente as superfícies escondidas.

A relação entre as dimensões dos pixels no espaço imagem e as dimensões dos voxels no espaço objeto, pode fazer com que mais de um elemento do modelo (voxel-1 ou face de voxel-1) ocupe áreas parcialmente sobrepostas em um mesmo pixel, ou mais de um pixel seja coberto pela área de um elemento projetado. O segundo caso não altera em nada os algoritmos, mas o primeiro caso pode ser contornado, por exemplo, associando ao pixel o valor de tonalização resultante da média ponderada dos valores de tonalização de cada elemento, onde os pesos são suas respectivas áreas de projeção descobertas. Normalmente assume-se as dimensões dos pixels iguais as dimensões dos voxels cúbicos. Problemas de *aliasing* na imagem, provocados por estar-se trabalhando em todas as etapas do fluxo de dados com amostras de funções contínuas, são geralmente tratados utilizando inicialmente pixels com dimensões menores, e depois mediando os valores de tonalização de conjuntos de pixels, para achar a tonalização dos pixels de tamanho desejado. Esta técnica é conhecida como *supersampling* (superamostragem de valores). Por exemplo, calculando a imagem com resolução $1024 \times 1024$ pixels e apresentando-a com resolução $512 \times 512$ pixels, os valores de cada quatro pixels são mediados para achar o valor de um pixel na imagem final [62].

### 3.3.2 Tonalização

O modelo de iluminação de Phong, representado pelas equações 2.12 e 2.13, necessita de duas informações básicas para cada pixel: a distância do ponto da superfície do objeto ao pixel correspondente (o valor do *z-buffer* para o pixel) e a estimativa do vetor normal à superfície neste ponto. Os modelos de volume binário e de superfície cubiculada representam uma aproximação grosseira do objeto. O papel do vetor normal é, portanto, prover uma estimativa mais aproximada da anatomia real da superfície, ditando a forma que a superfície corta cada elemento do modelo (voxel-1 ou face). Sua importância está em devolver os pequenos e mal definidos aspectos da estrutura real perdidos na segmentação. A estimativa do vetor normal pode ser feita no espaço voxel, no espaço objeto, ou no espaço imagem (figura 2.5). As formas e critérios adotados para fazer esta estimativa podem alterar

bastante a imagem do objeto e sugerem os diferentes tipos de tonalização.

#### 3.3.2.1 Tonalização por Distância

A forma mais simples de tonalização é conhecida por *depth shading* (tonalização por distância), que leva em conta apenas a informação contida no *z-buffer*. Este tipo de tonalização pode ser representada pela equação 2.12, onde os tons de cinza $I_{dist}(u, v)$ associados aos pixels $(u, v)$ variam linearmente [9] com o conteúdo do *z-buffer* na posição $(u, v)$. A figura 3.15 apresenta uma imagem, tonalizada por distância, de um crânio seco. Percebe-se que este tipo de tonalização fornece uma idéia aproximada da estrutura em estudo, o que algumas vezes é suficiente, porém pequenos detalhes e a informação de curvatura da superfície são perdidos [9]. Apesar do método ser o mais rápido e o mais simples de ser implementado, os pequenos detalhes e a informação de curvatura da superfície só são possíveis em métodos que utilizam o vetor normal estimado (equação 2.13).

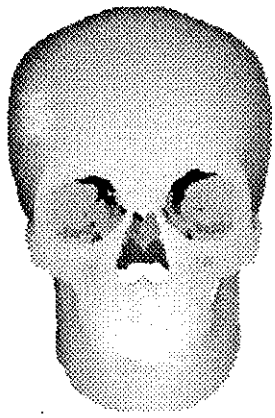

Figura 3.15: Tonalização por distância.

#### 3.3.2.2 Tonalização Estimando a Normal no Espaço Voxel

As técnicas de tonalização que estimam o vetor normal no espaço voxel (volume cinzento) [42] [75] [86] usam as densidades de voxels vizinhos no espaço pré-processado (figura 2.5). A coordenada dos elementos do modelo do objeto é utilizada para acessar

[9] Também consegue-se bons resultados usando variação não linear.

sua localização no volume cinzento. Estas técnicas utilizam o vetor gradiente normalizado, representando a direção de máxima variação de densidade, como estimativa do vetor normal. Esta aproximação só funciona bem quando é possível admitir que a superfície estimada pode ser representada por uma superfície de densidade de voxel constante, pois a componente do vetor gradiente na direção tangencial à superfície tem valor nulo e, portanto, a direção do vetor gradiente é normal à superfície. Em imagens de ressonância magnética (figura 2.2), por exemplo, a anatomia de estruturas ósseas não é bem representada pela densidade dos voxels, fazendo com que a estimativa da normal no volume cinzento não apresente bons resultados de tonalização.

As vizinhanças 6 (figura 2.9b) e 26 (figura 2.9c) do voxel central $v_0$ são as mais utilizadas. A estimativa do vetor normal $\vec{N} = (N_x, N_y, N_z)$ usando vizinhança 6, por exemplo, feita pelo método das diferenças centrais [9] para cálculo do gradiente, é representada pela equação abaixo.

$$\begin{aligned} N_x &= \frac{f(x+1,y,z) - f(x-1,y,z)}{2} \\ N_y &= \frac{f(x,y+1,z) - f(x,y-1,z)}{2} \\ N_z &= \frac{f(x,y,z+1) - f(x,y,z-1)}{2} \end{aligned} \tag{3.3}$$

onde:
$f(x,y,z)$ é o tom de cinza associado ao voxel $(x,y,z)$. Esta estimativa também é conhecida por *gray-level gradient shading* [82]. Quando $(x,y,z)$ é um voxel de borda do espaço voxel, utiliza-se o método das diferenças progressivas (ou retrógradas) [9], por exemplo:

$$N_z = f(x,y,z+1) - f(x,y,z) \tag{3.4}$$

se z é a coordenada da primeira fatia (diferença progressiva), ou

$$N_z = f(x,y,z) - f(x,y,z-1) \tag{3.5}$$

se z é a coordenada da última fatia (diferença retrógrada). Em alguns casos, é possível que a espessura da superfície do objeto seja insuficiente para o cálculo, mesmo usando vizinhança 6 [82]. Nestes casos é comum utilizar uma estimativa adaptativa (*adaptive gray-level gradient*

*shading*), por exemplo:

$$N_x = f(x,y,z) - menor[f(x+1,y,z), f(x-1,y,z)] \tag{3.6}$$

se $f(x,y,z) > f(x+1,y,z)$ e $f(x,y,z) > f(x-1,y,z)$;

$$N_x = maior[f(x+1,y,z), f(x-1,y,z)] - f(x,y,Z) \tag{3.7}$$

se $f(x,y,z) < f(x+1,y,z)$ e $f(x,y,z) < f(x-1,y,z)$;

$$N_x = \frac{f(x+1,y,z) - f(x-1,y,z)}{2} \tag{3.8}$$

em qualquer outro caso.

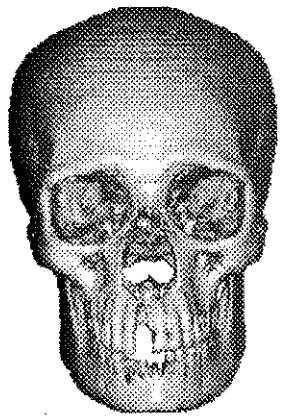

Figura 3.16: Tonalização estimando a normal no espaço voxel.

Quando trabalha-se com o modelo de superfície cubiculada e deseja-se estimar o vetor normal no centro $P$ de uma face usando a equação 3.3, precisa-se escolher no espaço voxel pré-processado quais os seis valores de densidade em torno do ponto $P$ vão ser usados na equação. Uma alternativa é substituir $P$ pelo voxel $v$ mais próximo de $P$ no espaço voxel pré-processado e utilizar as densidades dos seis voxels vizinhos face a face do voxel central $v$. Outra alternativa é calcular por interpolação as densidades $d(P_1), d(P_2), \cdots, d(P_6)$ dos seis pontos $P_1, P_2, \cdots, P_6$ equidistantes de $P$ ao longo dos eixos principais do espaço voxel [39].

A figura 3.16 apresenta uma imagem tonalizada do mesmo crânio seco da figura 3.15, onde foi utilizada a estimativa da normal no espaço voxel, utilizando vizinhaça 6. O resultado é uma imagem bem mais rica em detalhes, no entanto, é importante ressaltar

que além dos pequenos detalhes e da informação de curvatura da superfície, este método também pode reproduzir detalhes errados, ocorridos durante as etapas anteriores. Segundo Udupa [39], uma variedade de métodos mais sofisticados têm sido propostos para eliminar o ruído, mantendo os detalhes reais [43].

#### 3.3.2.3 Tonalização Estimando a Normal no Espaço Objeto

A estimativa do vetor normal no espaço objeto é baseada na geometria do modelo [53] [51] [64] [64]. No modelo de volume binário, esta estimativa pode ser feita em vizinhança 6 [22], por exemplo, usando a equação 3.3, onde $f(x, y, z)$ é 1 se o voxel $(x, y, z)$ pertence ao objeto e 0 no caso contrário. No modelo de superfície cubiculada, cada face $f$ possui quatro faces vizinhas que compartilham uma aresta com $f$ (figura 3.17). A normal à superfície no centro da face $f$ pode ser estimada, por exemplo, como a média ponderada dos vetores normais ao plano de $f$ e ao plano de suas quatro faces vizinhas $N_0, N_1, \cdots, N_4$ [52]. A figura 3.18 apresenta a imagem do crânio seco, onde a normal foi estimada no modelo de volume binário, usando a equação 3.3. O resultado apresenta uma piora de qualidade em relação ao anterior, mas não significa que este método é inferior em todos os casos.

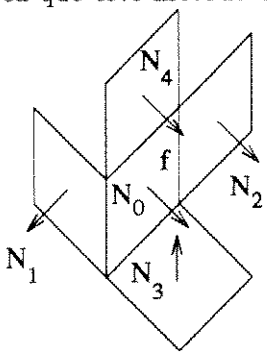


Figura 3.17: Cálculo da normal no espaço objeto para uma superfície cubiculada.

#### 3.3.2.4 Tonalização Estimando a Normal no Espaço Imagem

A tonalização estimando a normal no espaço imagem [46] [12], representado pelo *z-buffer*, também é conhecida por *z-buffer gradient shading* [82]. Cada pixel $(u, v)$ do *z-buffer* está associado a posição de uma pequena área $\delta a$ da superfície do objeto. Portanto,

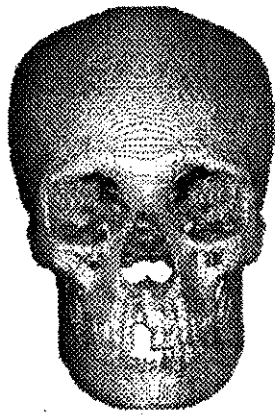

Figura 3.18: Tonalização estimando a normal no espaço objeto.

se existe uma continuidade espacial da parte da superfície projetada sobre o pixel $(u, v)$ e seus vizinhos imediatos na vertical (eixo v) e horizontal (eixo u) do *z-buffer*, o vetor normal $\vec{N} = (N_u, 1, N_v)$ estimado no centro da pequena área $\delta a$, usando a notação adotada na figura 2.20, é dado pelo método das diferenças centrais:

$$\begin{aligned} \vec{N} &= (N_u, 1, N_v) \\ N_u &= \frac{d(u+1, v) - d(u-1, v)}{2} \\ N_v &= \frac{d(u, v-1) - d(u, v+1)}{2} \end{aligned} \tag{3.9}$$

onde:
$d(u, v)$ é o valor do z-buffer para o pixel $(u, v)$.

Quando a continuidade espacial não ocorre, devido a bordas e cumes na superfície, uma estimativa errada da normal manifesta-se em forma de anéis escuros na imagem [39]. Algumas técnicas procuram minimizar este erro [11] [83] [9]. Em D. Geist e M.W. Vannier [9], por exemplo, usa-se um limiar de identificação de borda e aplica-se o método das diferenças progressivas (ou retrógradas) de forma similar às equações 3.4 e 3.5. A figura 3.19 apresenta o crânio seco, com normal estimada no espaço imagem. O resultado mostra que a estimativa da normal no espaço imagem contribui para uma tonalização de melhor qualidade que a da figura 3.18, mas a estimativa no espaço voxel (figura 3.16) apre-

sentou o melhor resultado entre todas as tonalizações. Isto se deve ao fato do tom de cinza relacionado aos voxels do objeto, no espaço voxel pré-processado, estarem relacionados com a informação anatômica do objeto [10].

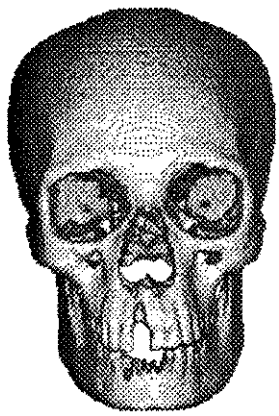

Figura 3.19: Tonalização estimando a normal no espaço imagem.

### 3.3.3 Combinação de Imagens

Imagens resultantes do *rendering* de superfícies distintas também podem ser combinadas em uma única imagem, usando o conceito de opacidade parcial [39] [26]. O uso de pseudo-cores pode ser útil para distinguir as superfícies distintas na imagem final [64] [18]. Esta técnica pode ser útil quando desejamos localizar na superfície externa a posição de alguma estrutura interna. A opacidade é representada por um valor real entre 0% e 100%, proporcional a evidência desejada na imagem final, para o tecido ao qual está sendo associada. Uma aplicação simples, por exemplo, é o *rendering* separado de duas superfícies, como a face e o crânio, utilizando os mesmos parâmetros de posição de visualização e depois a combinação das duas imagens geradas, em uma terceira imagem usando a álgebra de composição de imagens desenvolvida por Porter e Duff [81], onde o crânio é considerado opaco e a face semitransparente.

$$I(u,v) = \alpha_f I_f(u,v) + \alpha_c I_c(u,v)(1-\alpha_f) \tag{3.10}$$

[10] No caso, as imagens do crânio seco foram obtidas por tomografia de raios-x, que mostram com boa qualidade a anatomia de estruturas ósseas.

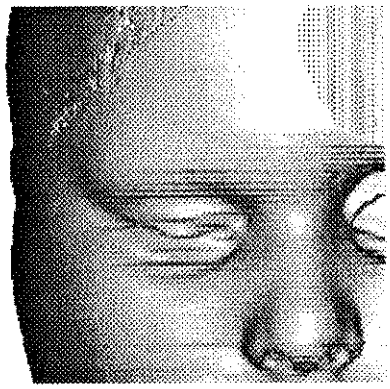

Figura 3.20: Visualização da face.

onde:
$I(u,v)$ é a intensidade de luz resultante da composição face-crânio, associada ao pixel $(u,v)$
$\alpha_f$ é a opacidade da face
$\alpha_c$ é a opacidade do crânio
$I_f(u,v)$ é a intensidade de luz refletida pela face, no ponto correspondente ao pixel $(u,v)$
$I_c(u,v)$ é a intensidade de luz refletida pelo crânio, no ponto correspondente ao pixel $(u,v)$.

Outra forma simples de combinar imagens é fazendo uma média ponderada. A figura 3.20 apresenta uma imagem da face e a figura 3.21 a imagem do crânio sob o mesmo ângulo de visão. A combinação destas imagens pode ser obtida pela equação abaixo (figura 3.22):

$$I(u,v) = \frac{\sum_{i=0}^{n} \alpha_i I_i(u,v)}{\sum_{i=0}^{n} \alpha_i} \tag{3.11}$$

onde:
$I(u,v)$ é o nível de cinza associado ao pixel $(u,v)$
$n$ é o número de imagens
$\alpha_i$ é o peso associado a $i$-ésima imagem
$I_i(u,v)$ é o nível de cinza associado ao pixel $(u,v)$ da $i$-ésima imagem.

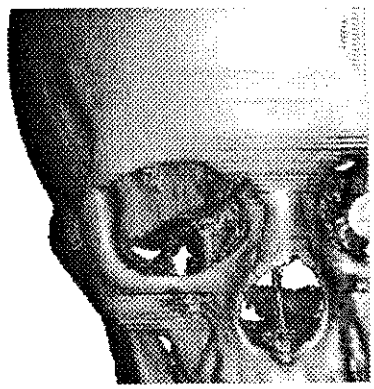

Figura 3.21: Visualização do crânio.

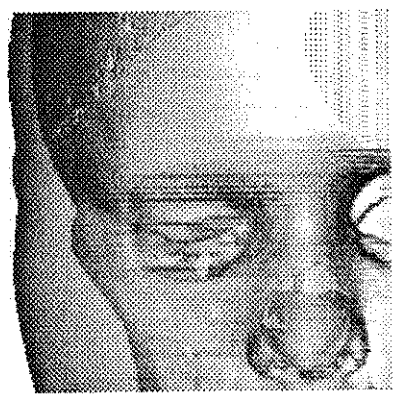

Figura 3.22: Visualização da face e do crânio na mesma imagem.

## 3.4 Técnicas de *Rendering* de Volume Semitransparente

As técnicas de *rendering* de volume semitransparente resolvem o problema da invisibilidade de estruturas internas ao espaço voxel pré-processado, seguindo o caminho da classificação de dados no fluxo da figura 2.5. Uma representação intermediária hipoteticamente criada para o espaço voxel é a de um volume colorido e semitransparente, onde a luz incidente pode ser transmitida e, através de sua reflexão, diferentes tipos de tecido podem ser visualizados ao mesmo tempo pelo observador. Para tanto, estas técnicas exploram as densidades dos voxels, selecionando tecidos distintos e dando maior ênfase (mais opacos), ou menor ênfase (mais transparentes), aos tecidos, segundo o grau de interesse do usuário.

Nestas técnicas não há a idéia de modelos geométricos para objetos, mas de regiões (ou nuvens) de interesse.

Pesquisadores de Pixar em 1985 foram os primeiros a usar estas técnicas de forma mais sofisticada em dados médicos 3D [59]. A técnica deles foi descrita em termos gerais por Smith [4] e apresentada em detalhes por Drebin et al. [66]. Ela consiste em estimar a fração de cada material (ar, músculo, gordura, osso) no interior dos voxels e usar estas frações para calcular uma cor e uma opacidade parcial para cada voxel. O método de Levoy [55] é similar a este método, mas calcula as cores e opacidades diretamente do valor escalar de cada voxel. Outros trabalhos importantes de *rendering* de volume semitransparente foram realizados por Westover [44] [45], Upson e Keeler [7] e Sabella [63]. Todos os métodos acima geram imagens similares e usam a álgebra de composição de imagens desenvolvida por Porter e Duff [81], para combinar o efeito dos voxels que influenciam em cada raio de projeção. Entretanto, uma diferença conceitual entre eles é que os modelos de *rendering* utilizados por Levoy, Drebin et al., Upson e Keeler, e Westover são reflexivos; a luz é emitida da fonte incidindo nos voxels do espaço, que refletem uma parte e transmitem a outra; e o modelo utilizado por Sabella é emissivo; a luz é emitida de cada voxel variando com sua densidade, e é transmitida através dos voxels seguintes.

As técnicas de *rendering* de volume semitransparente oferecem uma importante vantagem sobre as técnicas baseadas em superfície e em voxel binário, substituindo a classificação binária (segmentação) pela classificação nebulosa [82]. A classificação nebulosa e a cuidadosa reamostragem dos dados, por interpolação de valores no raio de projeção, reduzem os erros de quantização e *aliasing*. Entretanto, estas técnicas normalmente necessitam ter todo o volume de dados na memória e o tempo para *rendering* cresce proporcionalmente ao número de voxels, o que faz com que sejam consideradas técnicas computacionalmente caras (imagens típicas usando estações de trabalho SUN *Sparc 2* necessitam de minutos e às vezes horas para serem geradas). Para reduzir o tempo de *rendering* é possível tirar vantagens da coerência espacial do volume de dados e do processo de projeção (composição volumétrica). Em Levoy [56] [58] esta coerência é explorada acelerando o processo através da ultilização de decomposição *octree* do espaço voxel e decomposição *quadtree* do espaço imagem respectivamente [59]. A composição volumétrica pode ser explorada utilizando um limiar de opacidade para cessar os cálculos durante a projeção, ou utilizando menos que um raio por pixel para regiões de baixa complexidade (ou interesse) e aumentando gradualmente

o número de raios de acordo com a complexidade (ou interesse) [27]. Outra desvantagem é a dificuldade de extração de medidas, visto que estas técnicas não trabalham com estruturas topológicas definidas, mas com nuvens de tecidos de interesse. A falta de versatilidade por não trabalhar com objetos geometricamente definidos também pode ser verificada em muitas aplicações clínicas, que requerem a combinação de dados amostrados e uma geometria definida analiticamente na mesma imagem. Exemplos podem ser a superimposição de raios do tratamento de radiação sobre a anatomia do paciente, para o oncologista, e o *display* de próteses médicas, para o ortopedista. Uma solução é a conversão de ambas representações em uma representação comum: os algoritmos vistos nas técnicas baseadas em superfície podem ser utilizados para converter os dados amostrados na representação poligonal do objeto de interesse [28] [31], ou ambas representações são convertidas para o modelo de volume binário, usando os algoritmos vistos nas técnicas baseadas em voxel binário sobre os dados amostrados e o algoritmo de Kaufman [1], por exemplo, para conversão *scan* 3d do objeto poligonalmente definido. Uma solução alternativa é o *display* de ambas representações usando um algoritmo de *rendering* híbrido [14] [19] [57]. A primeira referência faz o *rendering* separado gerando duas imagens e depois faz a composição em uma terceira usando a classificação de distâncias para remoção de superfícies escondidas. As duas últimas são uma extensão de um *raytracing* convencional.

As principais diferenças entre as técnicas de *rendering* de volume semitransparente são as diferentes formas de classificação dos dados e de composição volumétrica durante o *rendering*.

### 3.4.1 Classificação dos Dados

A classificação dos dados pode ser vista como uma segmentação nebulosa, associando a cada voxel uma cor e uma opacidade parcial para gerar o volume colorido e semitransparente. Nos casos em que a segmentação automática necessita de auxílio, usa-se o mascaramento (seção 2.7.3) onde é associada uma opacidade 0% aos voxels que pertencem a tecidos indesejáveis na visualização, ou são associadas opacidades distintas a diferentes partes de um mesmo tecido. Um exemplo deste último caso é a visualização da pele na metade de uma face e do osso na outra metade. Um caso particular de volume colorido e semitransparente é o modelo de volume binário das técnicas baseadas em voxel binário.

Este caso pode ser obtido associando-se uma opacidade 100% aos voxels que pertencem à região que caracteriza o objeto de interesse e opacidade 0% aos demais. Na situação mais geral, tem-se múltiplos objetos de interesse e considera-se uma cor diferente para cada objeto, e uma opacidade entre 0% e 100% baseada na mistura de material contida em cada voxel [39].

Existem várias formas de fazer uma classificação nebulosa. Como ilustração são descritas a seguir as classificações de opacidade: linear, linear usando o gradiente e probabilística.

#### 3.4.1.1 Classificação Linear

A classificação linear de opacidade pode ser utilizada quando intervalos disjuntos de densidade de voxel podem ser usados para identificar o tipo de material contido em cada voxel. Um valor de opacidade constante entre 0% e 100% é associado a cada intervalo (figura 3.23). A um voxel associa-se o valor de opacidade referente ao intervalo que contém a densidade do voxel. Uma única cor pode ser associada a todos os voxels ou, por exemplo, uma cor pode ser associada a um voxel usando a mesma idéia de classificação linear da figura 3.23.

$$\alpha_{v_{linear}} = \frac{(\alpha_{n+1} - \alpha_n)}{f_{n+1} - f_n}(f_v - f_n) + \alpha_n$$

$$C_v = \frac{(C_{n+1} - C_n)}{f_{n+1} - f_n}(f_v - f_n) + C_n \tag{3.12}$$

onde:

$\alpha_{v_{linear}}$ é a opacidade associada ao voxel $v = (x, y, z)$

$C_v = (R_v, G_v, B_v)$ é a cor associada ao voxel v

$f_v$ é a densidade do voxel v

$f_n$ é a densidade no início do intervalo n

$\alpha_n$ é a opacidade associada à densidade $f_n$

$C_n = (R_n, G_n, B_n)$ é a cor associada à densidade $f_n$.

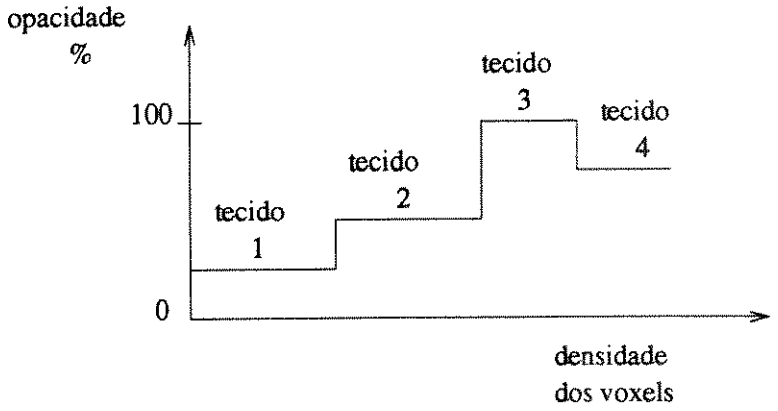


Figura 3.23: Classificação linear.

### 3.4.1.2 Classificação Linear Usando o Gradiente

A classificação linear usando o gradiente procura dar maior ênfase (maior opacidade) às regiões de interface entre tecidos e menor ênfase (se possível suprimir) às regiões internas aos tecidos [55], onde cada região de tecido é especificada por um intervalo de densidades disjunto (figura 3.24). A magnitude do vetor gradiente é usada para identificar as regiões de interface entre tecidos. Associando um valor de opacidade a cada voxel proporcional a magnitude do vetor gradiente, calculado no centro do voxel, consegue-se maior contraste nas interfaces de tecidos [11]. A cor associada a cada voxel é especificada pelas características de reflexão especular e difusa de cada comprimento de onda da luz incidente no voxel (i.e. Usa-se uma equação de Phong [6] modificada).

$$\begin{aligned} \alpha_v &= |\vec{g_v}|\alpha_{v_{linear}} \\ C_v &= C_f(K_a + \frac{1}{K_1 d_v + K_2}(K_d \cos\theta + K_s(\cos 2\theta)^n) \end{aligned} \tag{3.13}$$

onde:

$\alpha_v$ é a opacidade associada ao voxel $v = (x, y, z)$

$C_v = (R_v, G_v, B_v)$ é a cor associada ao voxel v

$|\vec{g_v}|$ é o módulo do vetor gradiente calculado no voxel v

$C_f = (R_f, G_f, B_f)$ é a cor da fonte de luz

[11] É interessante observar que a magnitude do vetor gradiente deve ser sempre $\leq 1$ para limitar a opacidade máxima em 100%. O gradiente é calculado pela equação 3.3.

$K_a = (k_{ar}, k_{ag}, k_{ab})$ é coeficiente de reflexão para a luz ambiente

$K_d = (k_{dr}, k_{dg}, k_{db})$ é o coeficiente de reflexão difusa

$K_s = (k_{sr}, k_{sg}, k_{sb})$ é o coeficiente de reflexão especular

$\theta$ é o menor ângulo entre o vetor normal à superfície e o raio de incidência

$K_1$ e $K_2$ são constantes de linearização

$d_v = d(u, v)$ é o valor do *z-buffer* para o voxel v.

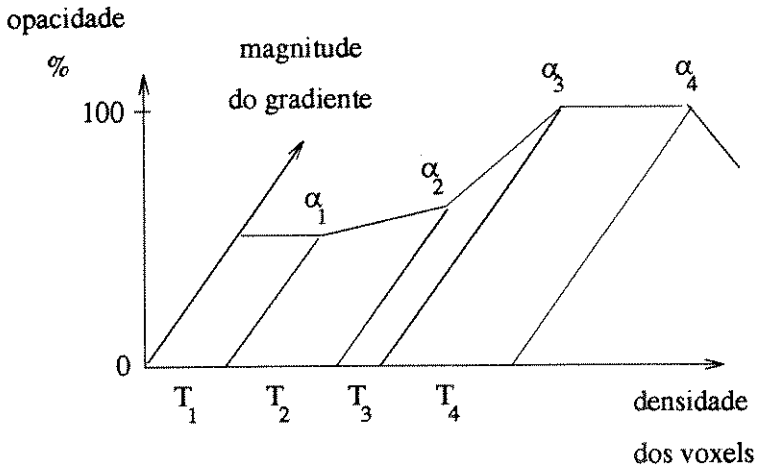


Figura 3.24: Classificação linear usando o gradiente.

### 3.4.1.3 Classificação Probabilística

Um método mais sofisticado [39] consiste em associar uma cor e uma opacidade parcial para cada voxel baseado na mistura de material (ou tecido) contida no voxel [66]. Conhecendo-se como a distribuição de probabilidade de cada tecido varia com a desidade dos voxels, é possível obter o gráfico percentagem dos tecidos *versus* densidade dos voxels (figura 3.25). Dada uma densidade x de um voxel, calcula-se o percentual de mistura $p_i$ de cada tecido $i$ contido neste voxel. Uma cor $C_i = (R_i, G_i, B_i)$ e uma opacidade $\alpha_i$ são associadas a cada tipo de tecido baseando-se na imagem desejada (ex: osso opaco 100% e pele semitransparente 20%). Uma cor $C = (R, G, B)$ e uma opacidade $\alpha$ efetivas para cada voxel podem ser calculadas pela seguinte equação:

$$R = \sum_{i=1}^{n} p_i R_i$$

$$
\begin{aligned}
G &= \sum_{i=1}^{n} p_i G_i \\
B &= \sum_{i=1}^{n} p_i B_i \\
\alpha &= \sum_{i=1}^{n} p_i \alpha_i
\end{aligned} \tag{3.14}
$$

onde $n$ é o número de tecidos.

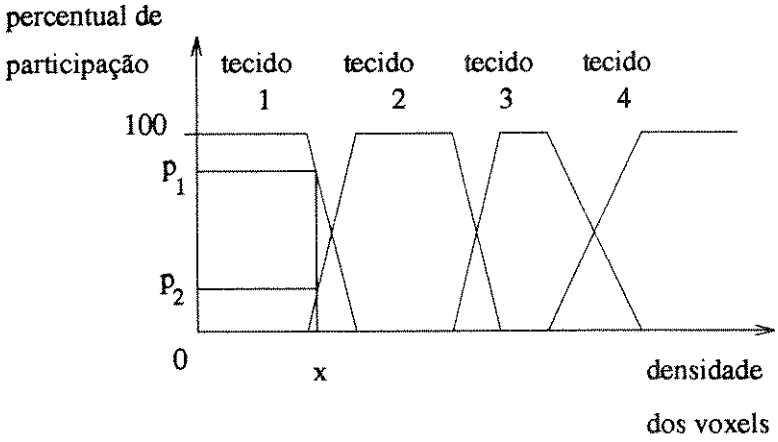


Figura 3.25: Classificação probabilística.

### 3.4.2 *Rendering*

Uma metodologia que combina várias imagens criando uma nova imagem é conhecida por *volumetric compositing* (composição volumétrica) [62]. A composição volumétrica aplicada ao espaço voxel é qualquer técnica que combina, para cada raio de procura, a influência de cada voxel $v_1, v_2, \cdots, v_n$ (figura 3.26a) no cálculo da cor do pixel correspondente. Definida uma cor e opacidade para cada voxel, volta-se a situação apresentada na figura 2.20. Os raios são lançados do centro de cada pixel do espaço imagem igual ao método *raycasting* (seção 2.8.2.2), porém para o *rendering* de volume semitransparente, os raios penetram no espaço voxel atravessando-o de um lado para o outro. O resultado é uma espécie de *raytracing* simplificado, onde a cor associada a cada pixel na imagem é o resultado da combinação da luz refletida e sucessivamente transmitida através de cada voxel

atravessado pelo raio de procura . Uma extensão para 3D do algoritmo de Bresenham [38], por exemplo, pode ser usada na identificação dos voxels, como se estivéssemos traçando uma reta em um espaço tridimensional discreto. Entretanto, um método comum é a reamostragem de $n$ pontos no raio, igualmente espaçados, desde o ponto que o raio penetra no espaço voxel até o ponto que o deixa, representando o centro de novos voxels $v_1, v_2, \cdots, v_n$ (figura 3.26a). Na reamostragem, as grandezas escalares e vetorial representadas pela densidade, opacidade e cor destes voxels são normalmente calculadas por interpolação trilinear das mesmas grandezas, relativas aos oito voxels mais próximos de cada novo ponto de amostragem (figura 3.26b).

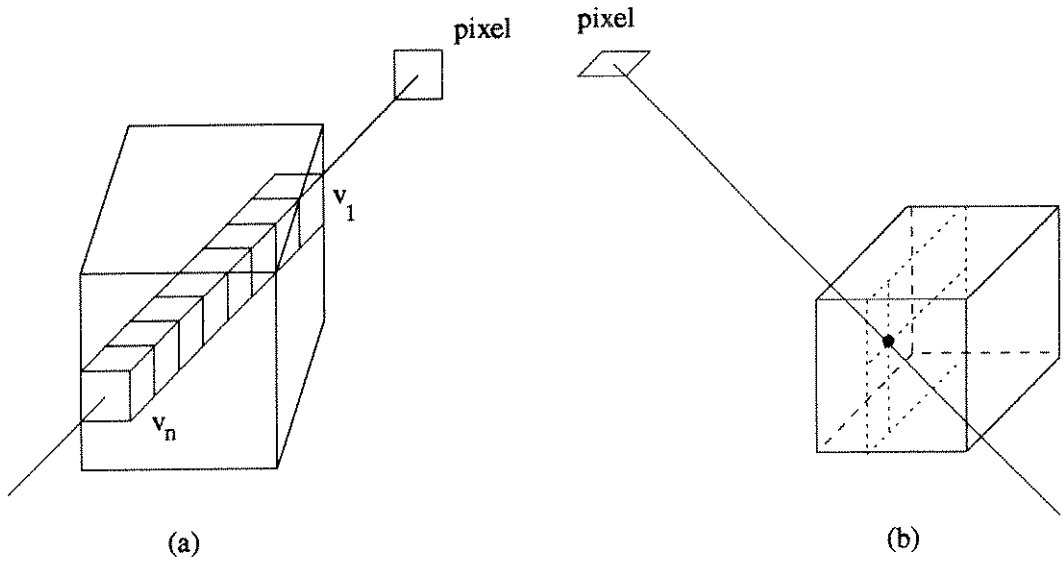


Figura 3.26: a) Escolha de voxels igualmente espaçados no raio de procura b) Cálculo da cor e opacidade para cada voxel do raio de procura usando interpolação trilinear.

A composição volumétrica pode ser feita no sentido *front-to-back* (de $v_1$ para $v_n$) ou *back-to-front* (de $v_n$ para $v_1$):

**Front-to-Back** [62]

$$C_{out}\alpha_{out} = C_{in}\alpha_{in} + C_i\alpha_i(1-\alpha_{in})$$

$$\alpha_{out} = \alpha_{in} + \alpha_i(1-\alpha_{in}) \tag{3.15}$$

onde para o i-ésimo voxel atravessado pelo raio:

$C = C_{out}\alpha_{out}$ é a cor associada ao pixel (u,v) no i-ésimo estágio do algoritmo

$C_{in}\alpha_{in}$ é a cor no (i-1)-ésimo estágio do algoritmo

$C_i$ e $\alpha_i$ são respectivamente cor e opacidade associadas ao i-ésimo voxel

$\alpha_{in}$ é a opacidade acumulada no (i-1)-ésimo estágio

$\alpha_{out}$ é a opacidade acumulada no i-ésimo estágio.

**Back-to-Front** [55]

$$C_{out} = C_{in}(1 - \alpha_i) + \alpha_i C_i \tag{3.16}$$

onde $C = C_{out}$ é a cor associada ao pixel (u,v) no i-ésimo estágio do algoritmo. A vantagem do *front-to-back* é que a opacidade acumulada pode ser usada para acelerar o algoritmo (i.e. Se $\alpha_{out} \geq 1$, o algoritmo inicia o processamento do raio seguinte). A figura 3.27 mostra um *rendering* de volume semitransparente usando classificação linear de opacidade e cor (equação 3.12) para a região ocular de um paciente do Hospital das Clínicas da UNICAMP. A figura 3.28 mostra um *rendering* de volume semitransparente de um joelho usando classificação linear e gradiente segundo a equação 3.13.

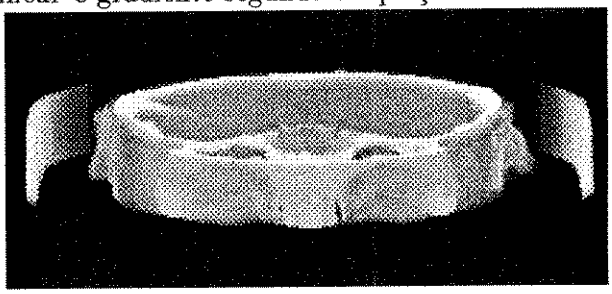

Figura 3.27: *Rendering* de volume semitransparente usando classificação linear.

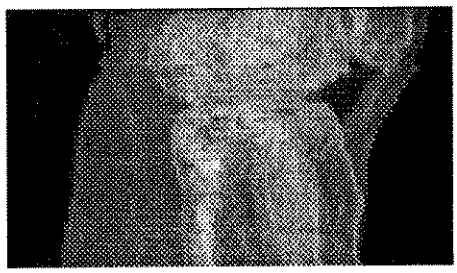

Figura 3.28: *Rendering* de volume semitransparente com classificação linear e gradiente.

A composição apresentada acima supõe uma distribuição homogênea de tecido para cada voxel, desconsiderando ou considerando o resultado efetivo (seção 3.4.1.3) quando uma superfície entre tecidos corta um voxel. Outro método um pouco mais complexo é apresentado por Udupa [39], que utiliza a equação 3.16 em três etapas para cada voxel,

quando existe uma forte evidência de uma superfície entre os tecidos que compõem um voxel.

## 3.5 Métodos Interativos

Em muitas aplicações clínicas é importante a interação do usuário com o sistema de processamento e visualização de dados. A meta principal é obter um sistema que permita, além da interação com o usuário, a obtenção de informações no menor tempo possível. Os métodos interativos procuram cumprir este papel, fornecendo informações limitadas, mas complementares, e muitas vezes suficiente, à informação tridimensional obtida com os métodos fundamentais. Estes métodos envolvem técnicas de refatiamento, técnicas de reprojeção e o uso de pseudo-cores.

### 3.5.1 Técnicas de Refatiamento

As técnicas de refatiamento podem ser definidas como operações de reamostragem feitas no espaço voxel, para visualizar uma ou mais fatias em qualquer direção. Estas técnicas também permitem que medidas quantitativas (distância e área) possam ser feitas sobre as novas fatias. Assumindo o espaço voxel formado por fatias paralelas ao plano xy e transversais ao eixo z (figura 2.4), considera-se transaxial esta direção. A operação mais simples é a visualização da k-ésima fatia. Uma operação mais complexa pode obter a visualização de novas fatias nas direções sargital e coronal (fatias paralelas aos planos yz e xz, respectivamente) [39]. Supondo voxels cúbicos, pode-se escolher a resolução [12] e espessura das novas fatias de forma a obter a densidade dos novos voxels, nas direções sargital e coronal, apenas trocando (ou acessando) a posição dos voxels no espaço original. Supondo que a densidade dos voxels (x,y,z) é armazenada no computador, linha por linha, da primeira fatia para a última: (1,1,1), (2,1,1), $\cdots$, (n,1,1), (1,2,1), (2.2,1), $\cdots$, (n,2,1), $\cdots$, (n,m,1), (1,1,2), (2,1,2), $\cdots$, (n,m,l). Um refatiamento coronal, por exemplo, acessaria a densidade dos voxels na seguinte ordem: (1,1,1), (2,1,1), $\cdots$, (n,1,1). (1,1,2), (2,1,2), $\cdots$, (n,1,2), $\cdots$, (n,1,l), (1,2,1), (2,2,1), $\cdots$, (n,m,l). Em um espaço voxel $3 \times 3 \times 3$ com densidades 15, 20,

[12] Considerando que normalmente a tela dos monitores usam pixels quadrados (ou de alguma forma simétricos), para evitar distorções na visualização, escolhe-se a resolução das fatias usando pixels quadrados.

50, 30, 42, 18, 17, 13, 12, 10, 50, 20, 60, 30, 45, 48, 19, 20, 15, 18, 13, 8, 6, 10, 11, 20, 19, o resultado seria um novo espaço $3 \times 3 \times 3$ com densidades 15, 20, 50, 10, 50, 20, 15, 18, 13, 30, 42, 18, 60, 30, 45, 8, 6, 10, 17, 13, 12, 48, 19, 20, 11, 20, 19.

A orientação de uma nova fatia pode ser ortogonal, oblíqua ou curva em relação ao espaço voxel original. Sua localização, resolução e espessura também são parâmetros arbitrários. Portanto, o centro de um novo voxel e o volume ocupado por ele, nem sempre coincidem com o centro e o volume de um voxel do espaço original. Neste caso, torna-se necessária a utilização de técnicas de interpolação (ver seção 2.6.4) para estimar a densidade dos novos voxels. Uma forma simples do usuário especificar os parâmetros de localização e orientação do refatiamento é desenhar com um *mouse* uma linha de intersecção (ou uma outra curva qualquer) no *display* de uma fatia original, para definir o plano (ou superfície) de refatiamento perpendicular ao plano desta fatia. Esta operação pode ser repetida sucessivas vezes no *display* das novas fatias, gerando cortes em qualquer direção.

Uma grande vantagem do refatiamento é que o paciente não precisa se submeter a um novo exame para que fatias em outras orientações sejam obtidas. No caso das tomografias, evita que o paciente receba uma dose maior de radiação. Em outros casos, a orientação desejada para as fatias é fisicamente impossível de ser obtida pelo equipamento de aquisição de dados. Um exemplo é o planejamento de implante dentário, quando deseja-se visualizar fatias de CT ortogonais à curvatura média do maxilar inferior (ou superior). Para compensar a ausência de informação tridimensional, uma ilusão de tridimensionalidade pode ser criada com uma animação de fatias obtidas consecutivamente em uma dada orientação. Esta técnica também é conhecida por *sweeping planes* [80].

### 3.5.2 Técnicas de Reprojeção

Relembrando o processo de formação das fatias tomográficas, a densidade dos voxels é obtida a partir de projeções geradas de várias direções. No caso do CT, por exemplo, uma projeção representa a atenuação sofrida pelos raios-x que atravessaram o corpo do paciente na direção correspondente. As técnicas de reprojeção, portanto, simulam este processo usando os raios de procura e o espaço voxel em lugar dos raios-x e do corpo do paciente. Estas técnicas realizam algum tipo de operação com as densidades dos voxels, que pertencem a um mesmo raio de procura, para gerar a projeção correspondente. A operação

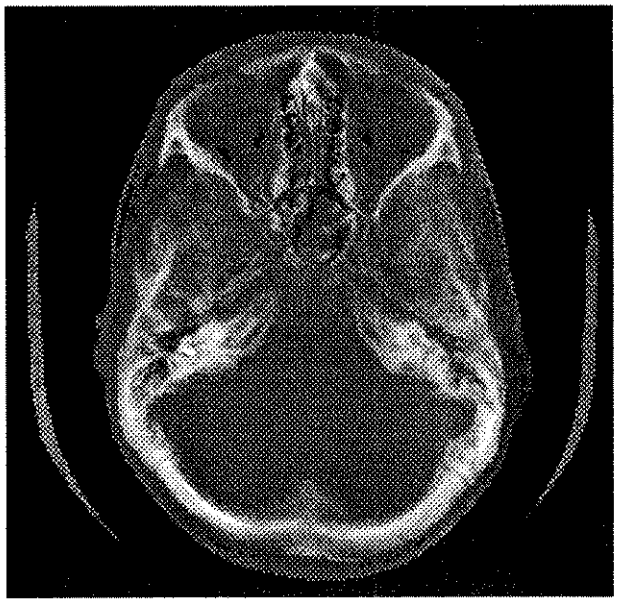

Figura 3.29: Vista superior da região ocular - reprojeção aditiva.

escolhida visa extrair algum tipo de informação contida na distribuição de densidade dos voxels, e pode ser vista como uma transformação do espaço voxel no espaço imagem.

#### 3.5.2.1 Reprojeção Aditiva

A reprojeção aditiva associa a cada pixel (u,v) do plano de visão uma média aritmética das densidades dos voxels visitados pelo raio de procura (figura 3.29).

$$I(u,v) = \frac{\sum_{v=0}^{n} f_v}{n} \tag{3.17}$$

onde:
$I(u,v)$ é o nível de cinza associado ao pixel $(u,v)$
$f_v$ é a densidade do n-ésimo voxel atravessado pelo raio
$n$ é o número de voxels atravessados pelo raio.

#### 3.5.2.2 Reprojeção Radiográfica

No caso de existir um maior interesse em algum tipo de tecido, a reprojeção aditiva leva desvantagem quando a maior parte dos voxels no raio pertencem a outros

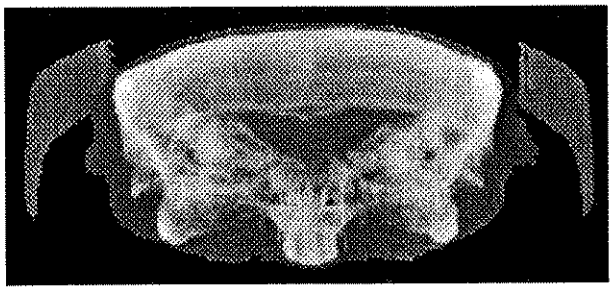

Figura 3.30: Reprojeção radiográfica.

tecidos. Uma forma de evitar este problema é associar a cada pixel (u,v) do plano de visão o resultado da média ponderada das densidades dos voxels visitados pelo raio, onde o peso atribuído a um voxel é proporcional ao interesse em visualizar o tecido contido no voxel [48] (figura 3.30). Os pesos podem ser escolhidos utilizando uma distribuição de opacidades como a da figura 3.23.

$$I(u,v) = \frac{\sum_{v=0}^{n} \alpha_v f_v}{\sum_{v=0}^{n} \alpha_v} \tag{3.18}$$

onde:
$I(u,v)$ é o nível de cinza associado ao pixel $(u,v)$
$f_v$ é a densidade do n-ésimo voxel atravessado pelo raio
$\alpha_v$ é a opacidade do n-ésimo voxel atravessado pelo raio
$n$ é o número de voxels atravessados pelo raio.

#### 3.5.2.3 Projeção de Máxima Intensidade

A projeção de máxima intensidade tem como objetivo identificar as regiões de maior densidade do conjunto de dados 3D. Esta técnica associa a cada pixel (u,v) do plano de visão o nível de cinza correspondente a maior densidade de voxel encontrada pelo respectivo raio de procura. Um resultado desta técnica pode ser visto na figura 3.31, onde regiões mais claras representam maior densidade.

### 3.5.3 O Uso de Pseudo-Cores

Um pixel colorido tem associado três atributos; matiz, luminância e saturação; enquanto um pixel monocromático tem associado apenas a luminância. Portanto, uma ima-

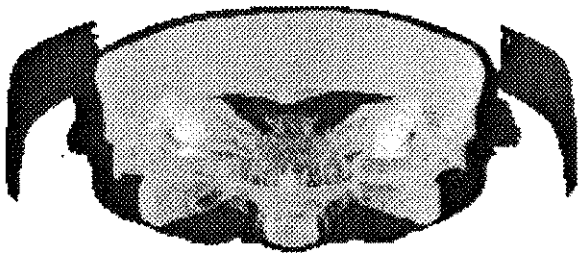

Figura 3.31: Projeção dos voxels de máxima intensidade.

gem colorida pode mostrar mais detalhes do que uma imagem monocromática. Entretanto, a escolha imprecisa das cores pode passar uma idéia errada da estrutura em questão. O uso de pseudo-cores é válido para a imagem de uma fatia do espaço voxel, como para qualquer imagem resultante de um dos métodos monocromáticos descritos anteriormente, e serve para identificar na imagem regiões de mesmo tecido, ou de tecidos diferentes.

A escolha das cores pode ser feita de diversas formas. Nos casos de *display* de estruturas tridimensionais distintas e previamente segmentadas, uma cor pode ser atribuída a cada estrutura durante a formação da imagem. Em um estudo de articulação, por exemplo, uma cor pode estar associada a cada osso que forma a articulação. Nos casos em que é possível uma segmentação por limiar, uma cor pode ser associada a cada intervalo de intensidade de pixels que caracteriza uma região de interesse na imagem. Em uma imagem com 256 tons de cinza, por exemplo, pixels entre 50 e 60 podem ser representados por 11 tons de vermelho, entre 90 e 110 por 21 tons de verde, entre 130 e 138 por 9 tons de amarelo, e entre 180 e 240 por 61 tons de azul. A escolha destes intervalos depende da natureza dos dados [13]. A técnica *painting planes* [17] é um exemplo típico feito sobre as imagens das fatias do espaço voxel. Uma matiz é associada a cada intervalo de densidade de voxel e as fatias coloridas são visualizadas de trás para frente, em relação ao observador, sobrepostas com um pequeno deslocamento vertical e horizontal, em relação a fatia anterior. Esta técnica pode representar oito formas diferentes de projeção oblíqua do espaço voxel (quatro de traz para frente e quatro de frente para traz). O aumento da luminância de cada matiz, quão mais próximo a fatia está do observador, completa o efeito de tridimensionalidade. Uma dificuldade pode ser encontrada na seleção das cores, devido o padrão RGB, normalmente

[13] Normalmente aconselha-se o uso de no máximo quatro cores, para não tornar difícil a interpretação da imagem final.

utilizado para *display*, não ser facilmente relacionado com a percepção humana. Neste caso pode-se definir as cores no padrão HLS (matiz, luminância e saturação), e depois utiliza-se um programa para converter o padrão HLS em padrão RGB.

# Capítulo 4

# Implementação

## 4.1 Introdução

Este capítulo tem por objetivo descrever a implementação de algumas técnicas de visualização apresentadas no capítulo 3. As técnicas escolhidas são baseadas em voxel e formam um pequeno sistema de visualização volumétrica, que pode ser utilizado não só para dados médicos como para dados metereológicos, dados de sensoriamento remoto e biológicos.

O sistema de visualização volumétrica está integrado ao ambiente Khoros [35]. O Khoros é um pacote aberto de processamento e visualização de dados desenvolvido na Universidade do Novo México, EUA [1]. Este programa possui como parte central um ambiente de programação visual que permite a interconexão de módulos de processamento e de fluxo de controle. Assim é possível construir estruturas complexas de processamento a partir de uma coleção de blocos interligados. O Khoros possui mais de 260 programas envolvendo as seguintes áreas de aplicação: manipulação de imagens, processamento de imagens, processamento de sinais, análise de imagens, classificação estatística, sistema de informação geográfica e sensoriamento remoto. Este trabalho colabora para a expansão do ambiente Khoros, acrescentando o processamento de dados 3D envolvido na visualização volumétrica. O Khoros foi desenvolvido utilizando o sistema *XWindows*, sendo, portanto,

[1] O Khoros está disponível para aplicações não comerciais na forma de código fonte no ftp *anonymous pprg.eece.unm.edu*.

compatível com qualquer plataforma que suporte este sistema (Apollo, Cray, DEC, HP, IBM, MIPS, NeXT, SGI e SUN). Os programas foram feitos em linguagem C e testados em estações SUN (*Sparc stations* 1+, 2 e 370).

O capítulo inicia com uma descrição do ambiente Khoros dando ênfase a sua facilidade de integração com o programador e com o usuário (seção 4.2). Cinco rotinas foram incorporadas: `soft2viff`, `vinter`, `zbuffer`, `shading` e `raytrace`. A rotina `soft2viff` (seção 4.3) converte os dados 3D do formato SOFTVU [76] para o formato VIFF do Khoros [2]. A rotina `vinter` (seção 4.4) implementa algumas técnicas de interpolação [3] apresentadas nas seções 2.6.4 e 2.7.6. As rotinas `zbuffer` e `shading` (seção 4.5) implementam quatro técnicas baseadas em voxel binário, e a rotina `raytrace` (seção 4.6) implementa três técnicas de reprojeção (métodos interativos) e duas técnicas de *rendering* de volume semitransparente. Na seção 4.7 são descritas as possibilidades de integração das rotinas do Khoros com as rotinas acrescentadas para gerar outras formas de visualização volumétrica.

## 4.2 O Sistema Khoros

A principal componente do sistema Khoros é sua linguagem de programação visual representada pelo módulo `cantata` (figura 4.1). O usuário constrói um programa de aplicação no `cantata` conectando blocos de processamento denominados *glyphs*, formando um grafo de fluxo de dados. Os dados podem ser imagens, sinais, pontos de uma curva ou de uma superfície. Cada *glyph* representa uma rotina disponível no Khoros. As rotinas desenvolvidas pelo programador são integradas ao sistema como *toolboxes*. Os dados podem ser visualizados em qualquer etapa do fluxo (figura 4.1). *Glyphs* de fluxo de controle (ex: `count_loop` da figura 4.8), permitindo a execução condicional e iterativa, e um analisador de expressões estendem a funcionalidade da linguagem de programação visual do `cantata`.

Existem duas categorias de programas no Khoros: *vroutines* e *xvroutines*. As *vroutines* funcionam como um filtro, não permitindo a interação do usuário com os dados durante sua execução. As *xvroutines* são mais complexas e se caracterizam por permitirem

[2] SOFTVU [76] é um sistema de visualização volumétrica desenvolvido no *Medical Image Processing Group (MIPG), Dept of Radiology, Univ of Pennsylvania, Philadelphia*, disponível no depto. de Eng. da Computação e Automação Industrial da Faculdade de Engenharia Elétrica da UNICAMP.

[3] As rotinas `soft2viff` e `vinter` foram desenvolvidas pelo Prof. Dr. Roberto de Alencar Lotufo durante estágio no MIPG.

a interação do usuário com os dados. Os programas feitos neste trabalho são *vroutines*. Um exemplo de *xvroutine* é o módulo **editimage**, que permite ao usuário editar textos, linhas e figuras na imagem, mudar o mapa de cores, pseudo-colorir, visualizar outras bandas, etc. O conceito de imagem multibanda no Khoros pode ser encarado de duas formas: uma imagem colorida possui três bandas, uma para cada primária, ou no caso do espaço voxel, por exemplo, o arquivo imagem possui várias bandas, onde cada banda é a imagem de uma fatia. As *vroutines* e *xvroutines* podem ser executadas no **cantata** em forma de *glyphs* ou direto da linha de comando. As imagens no Khoros são armazenadas no formato VIFF, ocupando 1024 bytes de cabeçalho com as informações relacionadas à imagem (tamanho do pixel, resolução das bandas, número de bandas, etc.) e, em seguida, os valores dos pixels da imagem. Os pixels são armazenados linha por linha e banda por banda. Cada pixel na imagem (ou voxel) pode ser armazenado como bit, byte, int, long, float, complex, double ou dcomplex. O Khoros suporta vários modelos de cor; NTSC, CIE, UCS, RGB, CMY, YIQ, HSV e IHS; e também tem como suporte um conjunto de rotinas de conversão entre diversos formatos de armazenamento de imagens.

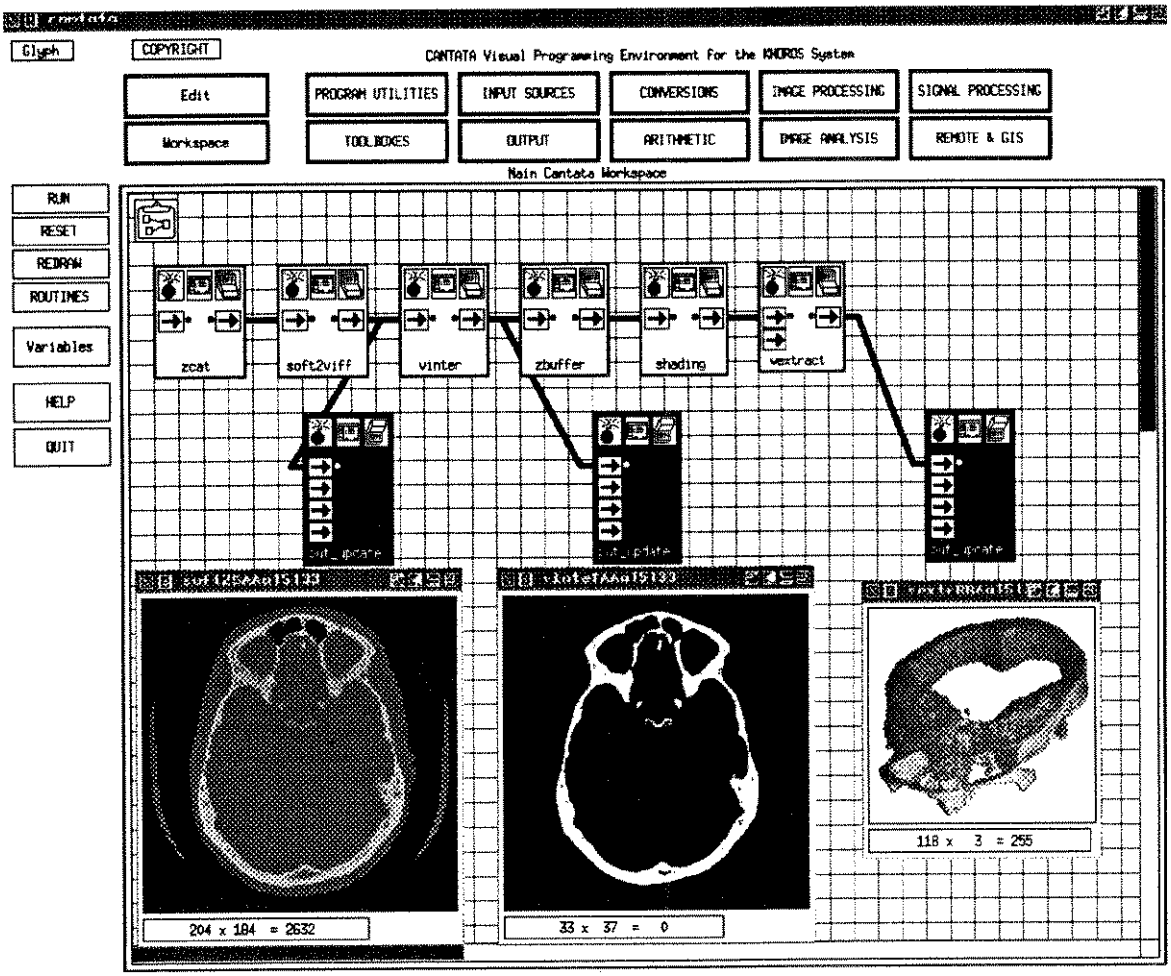


Figura 4.1: Fluxo de dados no módulo cantata de programação visual.

O sistema Khoros possui o módulo `composer` de interface com o programador. No `composer` o programador especifica o *layout* do painel de entrada e saída de parâmetros de sua rotina (figuras 4.2 a 4.6), através de um programa de especificação de interface com o usuário (UIS) [4]. O código fonte e as informações necessárias para a utilização da rotina são escritos em um arquivo base com extensão `prog`. Baseado nestas informações e em um arquivo de configuração, o `composer` instala automaticamente a rotina do programador, integrando-a ao sistema na forma de *toolbox*. A instalação consta da geração de códigos fontes, códigos executáveis e do código de interface com o usuário, através dos manuais, do

[4] Os parâmetros de entrada e saída utilizados nas rotinas `soft2viff` a `shading` do fluxo da figura 4.1 são mostrados a título de ilustração nos painéis que aparecem nas figuras 4.2 a 4.5.

painel e do *help on-line*.

As características apresentadas acima junto as vantagens de acesso a uma lista aberta de discussões via *e-mail*, o acesso a rotinas desenvolvidas por outros programadores e a sua portabilidade, justificam a escolha do Khoros para a implementação das técnicas deste trabalho. Outras componentes importantes do Khoros são a habilidade para traçar gráficos 2D e 3D, a animação de sequência de imagens, o *rendering* de superfícies de terrenos e o registro interativo de imagens.

## 4.3 Aquisição de Dados

Uma grande dificuldade na área de visualização volumétrica é a obtenção dos dados gerados pelos equipamentos médicos. A maior parte dos fabricantes destes equipamentos não revelam o formato da fita ou do disco em que os dados gerados são armazenados. Uma exceção à esta regra existe em relação aos equipamentos da GE. Foi feito um contrato em que foi possível programar um módulo de leitura do formato de fita do tomógrafo GE9800, existente no Depto. de Radiologia do Hospital das Clínicas da Unicamp, e converter para o formato SOFTVU, já que antes das rotinas do Khoros só tínhamos este sistema de visualização. A primeira etapa de programação no Khoros foi, portanto, a rotina `soft2viff`, para converter arquivos do formato SOFTVU para o formato VIFF do Khoros. A figura 4.2 mostra o painel contendo os dois parâmetros necessários para a rotina `soft2viff`: o nome do arquivo de entrada no formato SOFTVU e o nome do arquivo de saída no formato VIFF. Este trabalho também dispôs de dados de CT e MRI obtidos no MIPG do Depto. de Radiologia da Universidade da Pensilvânia, na fase de teste das rotinas. Atualmente dispõe-se de dados de SPECT, adquiridos no Depto. de Medicina Nuclear do Hospital das Clínicas da UNICAMP, e de dados de CT e MRI que fazem parte do *software* ANALYZE da Clínica Mayo, EUA [71]. O ANALYZE também consegue ler os dados do GE9800 e o SOFTVU consegue ler os dados do ANALYZE.

Os arquivos em qualquer formato constam de um cabeçalho, contendo as informações sobre o exame do paciente, e das imagens das fatias, representando o espaço voxel original (figura 2.5). Como o formato VIFF não possui um campo específico no cabeçalho para armazenar a informação de distância entre os cortes, esta informação é armazenada

no campo *fspare1* disponível no cabeçalho.

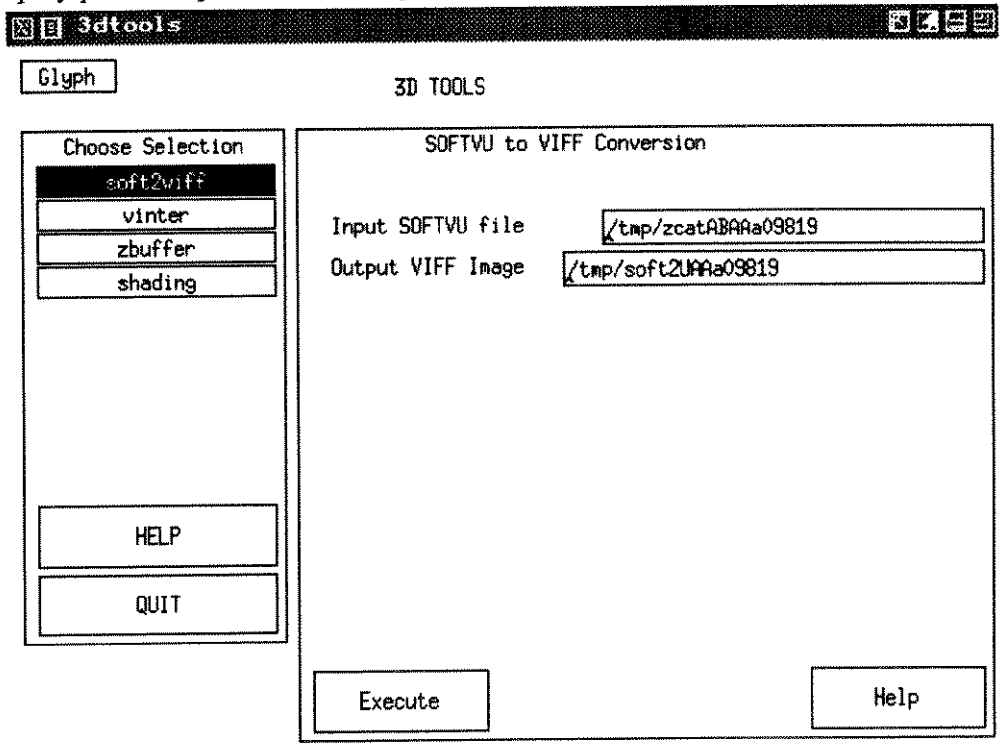


Figura 4.2: Painel do módulo de conversão de dados do formato SOFTVU para o formato VIFF.

## 4.4 Pré-processamento

As operações de pré-processamento descritas na seção 2.6 envolvem técnicas de registro entre fatias, volume de interesse, filtragem 3D e interpolação. O registro entre fatias e o volume de interesse podem ser obtidos com rotinas disponíveis no Khoros. Filtros 3D são disponíveis em alguns *toolboxes* feitos por outros programadores, mas também podem ser facilmente programados com pequenas alterações nas rotinas de filtragem 2D disponíveis no Khoros. A interpolação, entretanto, requer um tratamento mais específico e é representada pelo módulo `vinter`.

A figura 4.3 apresenta o painel dos parâmetros de entrada e saída de dados do programa `vinter`. O arquivo de entrada pode ser o espaço voxel original (seção 2.6.4) ou

um modelo de volume binário (seção 2.7.6), podendo ser aplicado antes ou depois do bloco de extração e modelagem da figura 2.5. Na figura 4.1, por exemplo, a entrada é o espaço voxel original (saída do `soft2viff`) e a saída é o espaço voxel pré-processado (entrada do `zbuffer`), conforme a ordem apresentada no fluxo de dados da figura 2.5. O programa `vinter` implementa as técnicas de interpolação discutidas nas seções 2.6.4 e 2.7.6, que são apresentadas por Lotufo et al. [68]. O programa suporta interpolação linear e do tipo *spline* cúbica modificada, para gerar fatias intermediárias, operando com imagens do tipo bit, byte e 2 bytes por pixel. Quando a entrada é o espaço voxel original, `vinter` tem a opção de fazer uma segmentação por limiar (seção 2.7.1.1), caso o usuário deseje que a interpolação seja aplicada a um modelo de volume binário, ou que seja feita uma segmentação no volume cinzento resultante da interpolação. O número de fatias na saída pode ser escolhido pelo usuário, ou pode ser calculado automaticamente gerando o espaço voxel isotrópico, onde as informações de dimensão dos pixels e de espaçamento entre as fatias contidas no cabeçalho do arquivo de entrada são utilizadas. As opções de interpolação oferecidas pelo `vinter` são:

- Interpolação clássica no volume cinzento (equação 2.4)
  entrada: espaço voxel original
  saída: espaço voxel pré-processado ou modelo de volume binário
- *Euclidean shape-based interpolation*
  entrada: espaço voxel original ou modelo de volume binário
  saída: modelo de volume binário
- *Combined shape-based interpolation*
  entrada: espaço voxel original
  saída: modelo de volume binário

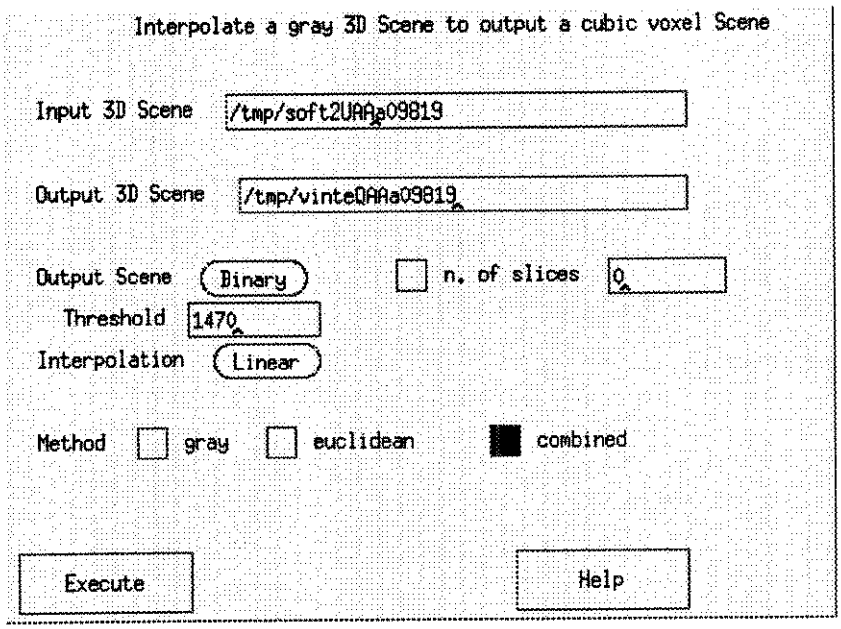


Figura 4.3: Painel do módulo de interpolação de fatias.

## 4.5 Técnicas Baseadas em Voxel Binário

As técnicas baseadas em voxel binário são representadas pelos programas `zbuffer` e `shading`. O programa `zbuffer` opera com imagens do tipo bit, byte e 2 bytes por pixel. Dado um ponto de vista do observador, `zbuffer` calcula o *z-buffer* e, opcionalmente, o vetor normal para cada ponto visível da superfície do objeto. A saída do `zbuffer` é um arquivo com uma ou, opcionalmente, duas bandas do tipo float. A primeira banda contém o *z-buffer* ($d(u,v)$ da equação 2.12) e a segunda banda contém o ângulo $\theta$ entre o vetor normal e a direção do raio de incidência (ver equação 2.13). O programa `shading` utiliza a saída do `zbuffer` para fazer a tonalização, gerando a imagem final (figura 4.1).

O arquivo de entrada no `zbuffer` (painel da fig. 4.4) pode ser o espaço voxel pré-processado ou um modelo de volume binário (figura 2.5). O programa parte da situação apresentada na figura 2.20 e implementa um algoritmo de *raycasting* (seção 2.8.2.2). O usuário escolhe no painel os ângulos de rotação, $\alpha$ e $\beta$, do observador em torno dos eixos $z$ e $x$, respectivamente (figura 2.20). Para cada raio lançado de um pixel do espaço imagem,

o algoritmo acha inicialmente a intersecção do raio com uma das seis faces do cubóide que envolve o espaço voxel, e aplica a equação 2.11 até encontrar um ponto de intersecção com a superfície do objeto. O usuário escolhe o passo $\Delta t$ que o algoritmo caminha sobre o raio e a distância em relação a origem do sistema $(x, y, z)$, que deseja posicionar o espaço imagem, para que o algoritmo calcule a translação necessária e efetue um corte no objeto. O ponto de intersecção do raio com a superfície do objeto pode ser representado por um voxel do volume cinzento, que tem densidade maior que um valor de limiar escolhido no painel, embutindo uma segmentação por limiar, ou por um voxel-1 do modelo de volume binário. O programa **zbuffer** oferece quatro tipos de saída. A saída mais simples envolve apenas o cálculo do *z-buffer*. As outras três opções envolvem também o cálculo do vetor normal: no espaço imagem (equação 3.9), no espaço objeto (equação 3.3) e no espaço voxel (equação 3.3).

O programa **shading** (painel 4.5) pode gerar quatro tipos de tonalização: tonalização por distância (seção 3.3.2.1), tonalização estimando a normal nos espaços voxel (seção 3.3.2.2), objeto (seção 3.3.2.3) e imagem (seção 3.3.2.4), dependendo da opção escolhida anteriormente no programa **zbuffer**. As três últimas são classificadas no painel da figura 4.5 como tonalização por gradiente. A primeira implementa a equação 2.12 e a tonalização por gradiente implementa a equação 2.13. Os parâmetros de ambas equações são escolhidos no painel pelo usuário, justamente com a opção de escolha de cor de fundo.

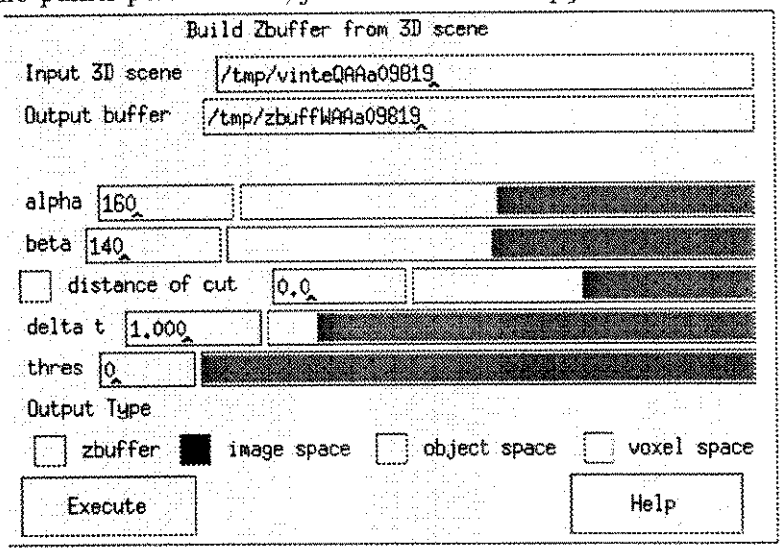


Figura 4.4: Painel do módulo de geração do *z-buffer* e cálculo de normais.

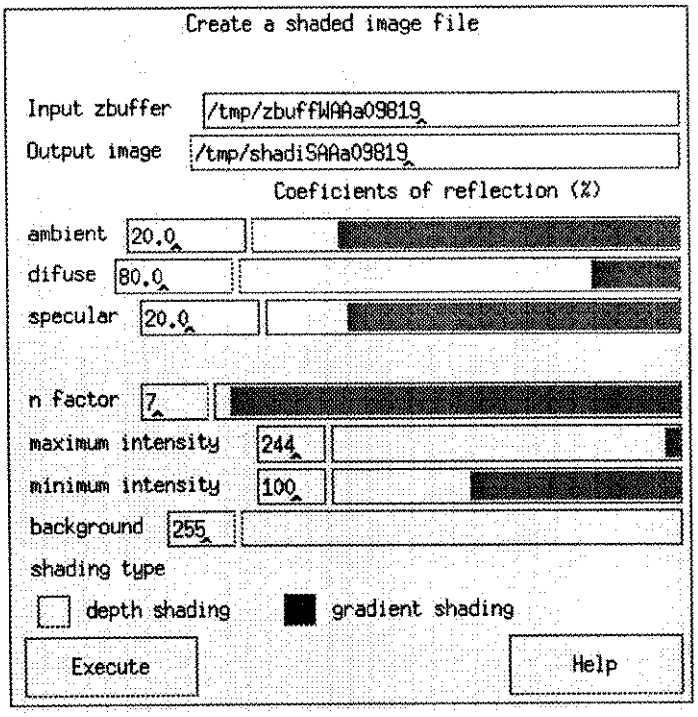


Figura 4.5: Painel do módulo de tonalização.

## 4.6 Técnicas de Reprojeção e *Rendering* de Volume Semitransparente

O programa `raytrace` (painel 4.6) implementa três técnicas de reprojeção (seção 3.5.2): reprojeção aditiva (equação 3.17), reprojeção radiográfica (equação 3.18) e projeção de máxima intensidade; e duas técnicas de *rendering* de volume semitransparente (equação 3.15): uma com classificação linear (equação 3.12) e a outra com classificação linear usando o gradiente (equação 3.13). Para ter um maior controle na distribuição de densidade dos voxels, facilitando o ajuste de parâmetros envolvidos na classificação, o `raytrace` opera apenas com imagens do tipo 1 byte por pixel. O arquivo de entrada é o espaço voxel pré-processado e a saída é a imagem final, que nas técnicas de reprojeção é uma imagem em tons de cinza (8 bits por pixel) e nas técnicas de *rendering* de volume semitransparente é uma imagem colorida do tipo RGB (3 bandas formando 24 bits por pixel).

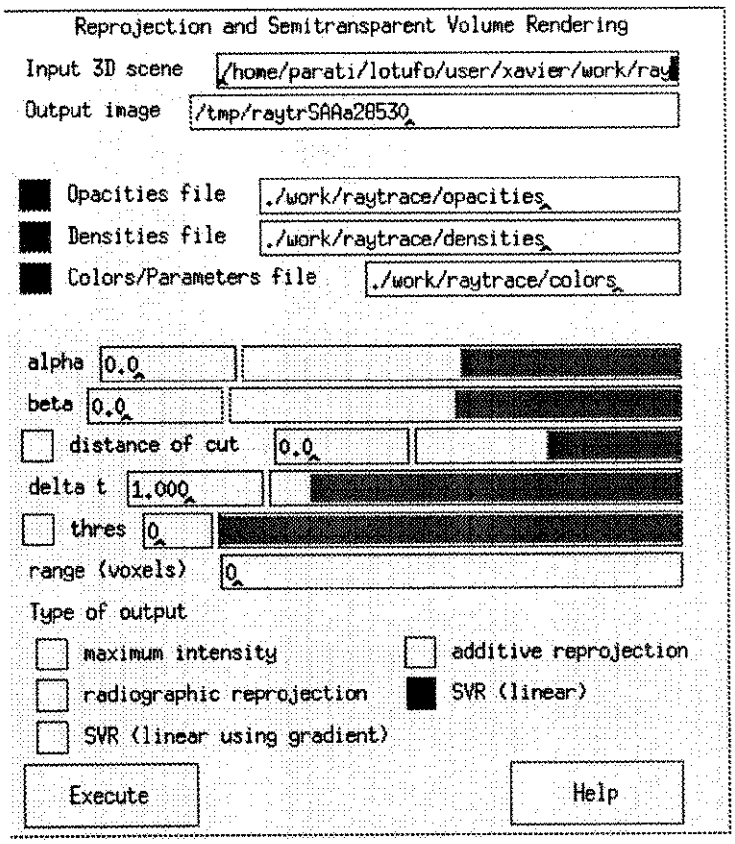


Figura 4.6: Painel do módulo de implementação das técnicas de reprojeção e *rendering* de volume semitransparente.

O posicionamento inicial observador-objeto é o apresentado na figura 2.20. O algoritmo é similar ao utilizado no `zbuffer`, tal que os parâmetros $\alpha$, $\beta$, distância de corte e $\Delta t$ do painel da figura 4.6 têm a mesma função. A principal diferença é que o algoritmo utiliza para calcular a cor de cada pixel no espaço imagem, todos os voxels atravessados pelo raio. Um valor de limiar e um intervalo de voxels em torno deste valor, podem ser utilizados para aplicar o algoritmo apenas na determinada espessura de voxels. Os pontos de mudança de declividade da curva opacidade $\times$ densidade (figura 3.23), utilizada nas técnicas de reprojeção radiográfica e *rendering* de volume semitransparente com classificação linear, são informados através de dois arquivos ASCII; um com a coordenada de opacidade de cada ponto e o outro com a coordenada de densidade. No caso do *rendering* com classificação linear, usa-se também um arquivo ASCII com as coordenadas de cor RGB para cada ponto. No caso do *rendering* com classificação linear usando o gradiente, uma opacidade é inicial-

mente determinada para cada voxel usando os arquivos ASCII, e depois multiplicada pelo gradiente local (equação 3.3) normalizado. Neste caso, um outro arquivo ASCII é utilizado para entrar com os parâmetros de cálculo da cor de cada voxel (equação 3.13).

## 4.7 Explorando o Khoros

As figuras 4.7 e 4.8 mostram duas formas interessantes de explorar as rotinas do Khoros para obter novas formas de visualização.

A figura 4.7 implementa a técnica de combinação de imagens apresentada pela equação 3.11. As rotinas `vscale` são utilizadas para introduzir a multiplicação de cada intensidade $I_i(u,v)$ pelos pesos $\alpha_i$ (no caso $\alpha_1 + \alpha_2 = 1$) e a rotina `vadd` para adicionar os resultados.

O Khoros permite agrupar vários blocos (ou rotinas) em um único bloco, formando uma *procedure*. A *procedure* `preprocessing` na figura 4.7 é um exemplo que agrupa o subfluxo representado pelas rotinas `zcat`, `soft2viff` e `vinter` da figura 4.1 em um único bloco e a *procedure* `display` agrupa as rotinas `vextract` e `put_update`.

A figura 4.8 utiliza as rotinas de fluxo do controle `count_loop`, onde os programas `zbuffer` e `shading` são executados diversas vezes gerando os quadros de uma animação. Os ângulos de rotação $\alpha$ e $\beta$ do painel da figura 4.4 são variáveis definidas nas rotinas `count_loop` e são incrementados (ou decrementados) a cada passo. Os quadros gerados são combinados na rotina `vbandcomb` e finalmente animados pela rotina `animate`, que funciona de forma similar a um vídeo cassete. A rotina `animate` pode também ser usada para mostrar uma animação das fatias transversais (técnica *sweeping plane* - seção 3.5.1).

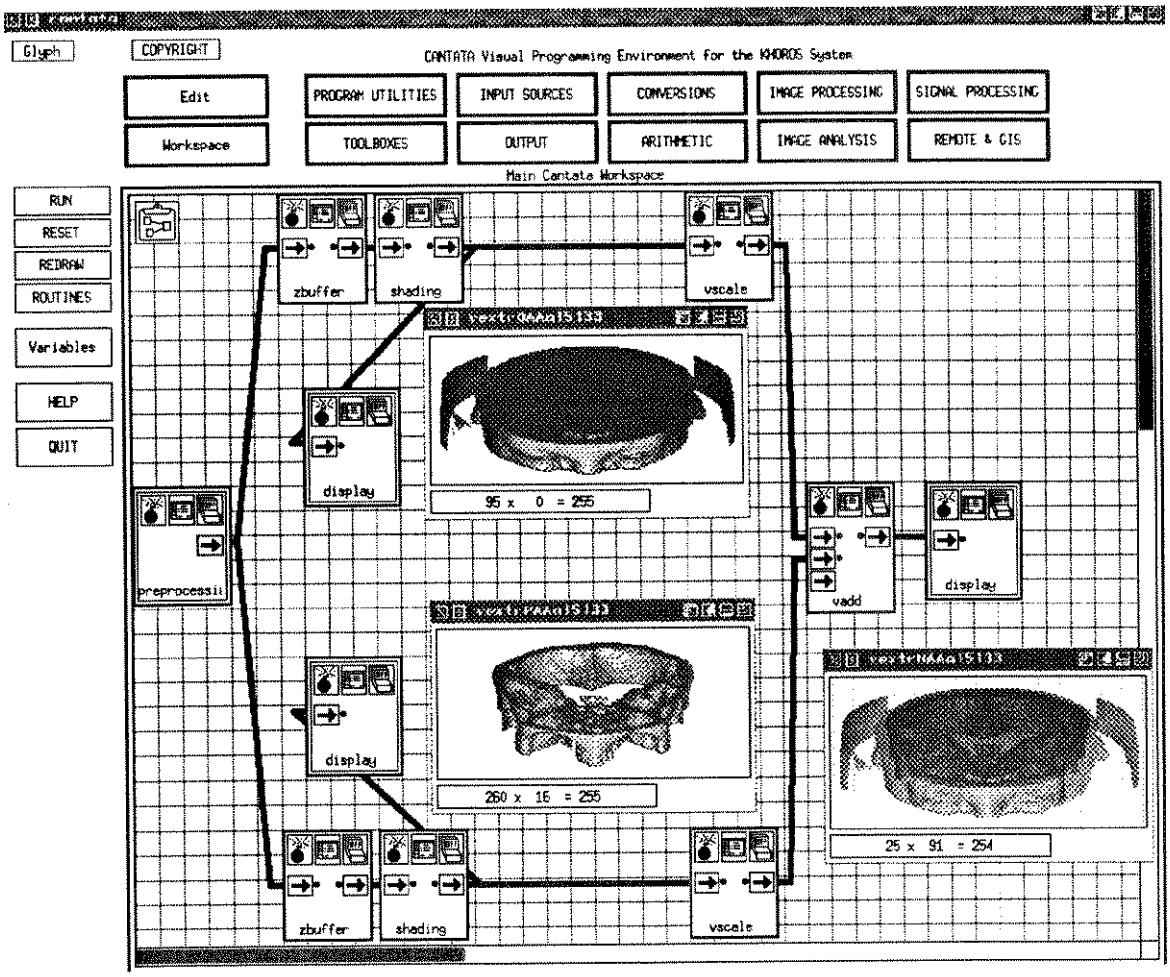


Figura 4.7: Fluxo de Combinação de Imagens.

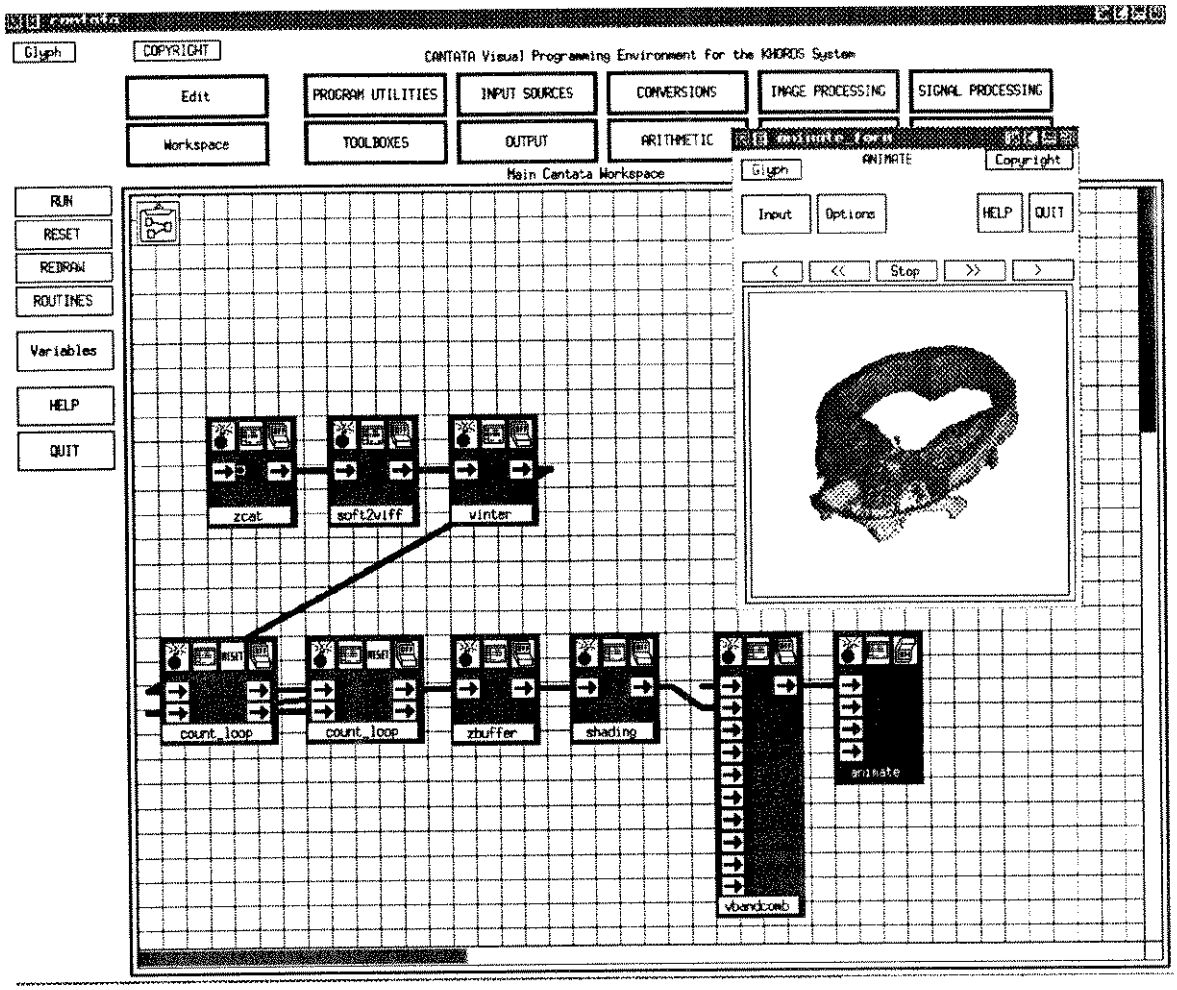


Figura 4.8: Fluxo para geração de uma animação.

# Capítulo 5

# Conclusão

Este capítulo apresenta os resultados obtidos e algumas sugestões para a continuação deste trabalho. A título de ilustração é descrita a importância das áreas de manipulação e análise em aplicações clínicas, dando ênfase ao estudo do problema de registro de dados 3D no planejamento de cirurgias e tratamentos.

## 5.1 Resultados

Os métodos de visualização apresentados envolvem a maior parte das técnicas encontradas na literatura. Apesar de algumas outras técnicas de visualização importantes, tais como, *stereo display*, espelhos varifocais e holografia, não serem discutidas neste trabalho, este tutorial representa uma boa fonte de estudo para os pesquisadores que desejam iniciar na área. O trabalho também teve como resultado um ambiente interativo de visualização volumétrica, com quatro técnicas de *rendering* de volume binário, duas técnicas de *rendering* de volume semitransparente, três técnicas de reprojeção e as facilidades oferecidas pelo ambiente Khoros para combinar imagens, fazer uma animação, etc. A experiência quanto ao uso do Khoros confirmou como sendo um ambiente apropriado para a pesquisa e ensino. Sua flexibilidade permite explorar diversas combinações de operações primitivas e escolher a melhor técnica de visualização para cada tipo de dado. As rotinas `soft2viff`, `vinter`, `zbuffer` e `shading` estão documentadas e disponíveis na Universidade do Novo México, pelo ftp *anonymous* `pprg.eece.unm.edu`. Outros resultados são os artigos Falcão

et al. [5] e Lotufo et al. [67], e um tutorial sobre visualização volumétrica apresentado em novembro de 1992 no V Simpósio Brasileiro de Computação Gráfica e Processamento de Imagens.

## 5.2 Sugestões

A continuação do trabalho na área de visualização pode ser feita com o estudo mais aprofundado das técnicas de *rendering* de volume semitransparente. Entretanto, ainda existem alguns pontos críticos no processo de visualização que influenciam bastante a qualidade da imagem final. É difícil distinguir um artefato de um dado real na imagem final. Estes pontos envolvem o estudo mais aprofundado de técnicas de interpolação, de segmentação e de estimativa do vetor normal. Isto torna a visualização uma ferramenta não plenamente confiável em diagnósticos clínicos, sendo utilizada apenas como verificação de um resultado já esperado ou como complementação à análise de parâmetros quantitativos (volume, distância, área, etc.). Portanto em aplicações clínicas, a visualização só tem sentido como auxílio às áreas de manipulação e análise. Estas áreas têm apresentado novos desafios e um grande interesse dos pesquisadores no exterior, principalmente pela variedade de aplicações. Um trabalho interessante no planejamento de cirurgias pode envolver o estudo de um modelo adequado, para simular com facilidade as possíveis modificações na anatomia de um orgão, ou de técnicas de registro de dados 3D. Como ilustração são apresentadas algumas aplicações que envolvem técnicas de registro como ferramenta básica para a manipulação e análise no planejamento de cirurgias e tratamentos.

### 5.2.1 Técnicas de Registro

Quando dispõe-se de imagens tomográficas, o primeiro problema enfrentado no planejamento de cirurgias e tratamentos, é o mapeamento entre um ponto (ou pixel) da imagem de uma fatia e sua localização precisa (ou voxel) no corpo do paciente. Um problema mais abrangente surge quando precisa-se fazer o mapeamento entre um ponto (ou voxel) do corpo do paciente com seu ponto correspondente em imagens, ou modelos 3D de um mesmo objeto, provenientes de diferentes modalidades ou adquiridas em instantes diferentes. Uma sugestão para a solução deste problema está no estudo de técnicas de registro. Estas técnicas

procuram um sistema de coordenadas comum ao paciente e às imagens, ou modelos 3D, para corrigir as possíveis distorções entre eles, encontrando uma correlação entre os pontos e fazendo, quando necessário, um alinhamento.

Uma das principais aplicações de técnicas de registro é descrita por Pelizzari et al. [8], visando a correlação das informações anatômica e funcional do corpo humano. Pelizzari et al. propõe uma técnica de registro, baseada nos contornos externos da cabeça do paciente extraídos das fatias, para que neurologistas, neurocirurgiões e neuroradiologistas possam correlacionar lesões anatômicas com anomalias fisiológicas, e também correlacionar as mudanças estrutural e funcional do cérebro após os procedimentos cirúrgicos. A informação anatômica é extraída de imagens de tomografia de raios-x (CT) e ressonância magnética (MR), e a informação funcional é extraída de imagens de tomografia por emissão de pósitron (PET).

Outra técnica de registro, proposta por Schad et al. [50], visa o planejamento no tratamento por radiação de tumores cerebrais. Esta técnica utiliza uma estrutura rígida e fixa na cabeça do paciente, que pode ser classificada como um sistema de localização estereotáctico [1], para encontrar um sistema de coordenadas comum às imagens, usando as marcas da estrutura em cada imagem. Neste caso, os objetivos principais são a localização precisa do tumor, o que evita a danificação de estruturas sadias pelo feixe de radiação (ex: nervo ótico), e seu volume. Imagens de ressonância magnética (que fornecem ótimo contraste para tecidos macios) são utilizadas na obtenção do volume do tumor e identificação de estruturas críticas ao seu redor, que precisam ser removidas. Entretanto, as imagens de tomografia por emissão de pósitron (que contribuem com a informação metabólica) são utilizadas para auxiliar na localização do tumor. Estas informações são utilizadas na dosimetria da radiação. Devido a problemas de homogeneidade nas imagens de ressonância magnética, a tomografia de raios-x é utilizada para a localização de estruturas ósseas/ar (que também são importantes na dosimetria). O controle eficaz na dosimetria, evita que doses muito altas possam danificar tecidos sadios em torno do tumor e doses muito baixas não resolvam o problema. Outras técnicas que utilizam um sistema de localização estereotáctico e auxiliam as neurociências são descritas por Bergström et al. [54] para alinhamento de imagens de CT e PET, e por Fox et al. [65] para a localização anatômica entre o paciente e imagens de

[1] O termo estereotáctico está relacionado com estereotáxico, que em cirurgia significa, método apropriado para fazer perfurações no crânio a fim de atingir o ponto desejado no cérebro.

PET, utilizando um atlas da anatomia padrão do cérebro.

Com relação ao registro de modelos 3D, em muitas aplicações clínicas existe a necessidade de fazer análise comparativa e análise de composição [41]. A análise comparativa é feita quando deseja-se medir as modificações que o objeto sofreu ao longo do tempo. Um exemplo é a medida do volume de enxertos ósseos implantados durante uma cirurgia corretiva de crânio. Devido a dificuldade de segmentação entre o enxerto ósseo e o osso natural do paciente em uma representação 3D obtida usando a tomografia por raios-x (CT) [2], esta segmentação é feita por subtração de duas representações 3D; uma obtida antes da cirurgia craniofacial e a outra obtida depois. A análise de composição é feita quando deseja-se combinar informações complementares sobre o objeto vindas de dispositivos de aquisição diferentes. Um exemplo da análise de composição é o seguinte estudo de caso [62]:

"Um paciente chega a um hospital sofrendo sucessivas convulsões. Como primeira etapa do tratamento são obtidas, por ressonância magnética (MR), 63 imagens de fatias da cabeça do paciente. Estas imagens não revelam qualquer anormalidade. Em seguida um modelo tridimensional do cérebro do paciente é reconstruído a partir destas imagens. Este modelo revela achatamento nas circonvulsões cerebrais, na parte inferior das faixas motora e sensorial, o que não foi notado nas imagens das fatias. Os médicos então fazem um segundo exame usando tomografia por emissão de pósitron (PET) para retratar a atividade metabólica do cérebro. O modelo tridimensional do exame PET revela a atividade metabólica cortical média. Neste modelo é observado um volume de atividade hipermetabólica. Devido a tomografia por emissão de pósitron gerar imagens de baixa resolução, a localização precisa da anormalidade não pode ser correlacionada com as marcas anatômicas conhecidas da superfície cerebral. Para correlacionar os resultados dos dois exames, os médicos combinam os dois modelos 3D usando técnicas de registro [3]. O modelo combinado provê as informações anatômica e metabólica requeridas para a localização precisa da anormalidade na parte inferior das faixas motora e sensorial. Os médicos usam um programa para simular a cirurgia no modelo combinado do cérebro. Um *display* lado a lado da imagem do modelo combinado do cérebro e da superfície da pele que está em sua volta é feito para guiar os médicos. Usando um *mouse*, os médicos contornam na imagem do

[2] A tomografia por raios-x é a modalidade de aquisição de dados mais indicada quando trabalha-se com osso.

[3] Técnicas de registro também permitem que imagens de volumes adquiridas em tempos diferentes possam ser combinadas em uma única imagem para *display*.

cérebro a área da anormalidade. O programa usa o contorno marcado para fazer uma craniotomia simulada. Fotos da operação simulada são retiradas para guiar os procedimentos na cirurgia real. Um eletroencefalograma intraoperativo confirma uma atividade de ataque no local prescrito pelas imagens médicas. Após a remoção da área anormal cessa a atividade de ataque do paciente".

O principal problema é a dificuldade em obter um mesmo posicionamento do paciente para todos os exames. Apesar dos métodos estereotácticos, e métodos que usam marcadores externos, surtirem bons resultados, estes métodos não são aplicáveis em estudos que necessitam de dados adquiridos anteriormente na rotina da prática clínica. Outras dificuldades são provenientes das diferentes características dos *scanners*: tamanho do pixel, espessura das fatias e o espaçamento entre elas, e distorções nas imagens ( imagens de MR apresentam algumas distorções devido a falta de homogeneidade do campo magnético principal e não linearidade dos gradientes). Imagens de PET e SPECT também apresentam problemas por serem de baixa resolução: em alguns problemas de registro, o sistema de coordenadas comum é encontrado através da ajuda de um *especialista* que localiza nas imagens, marcas anatômicas conhecidas do paciente. Em imagens de PET e SPECT dificilmente consegue-se um número suficiente destas marcas.

# BIBLIOGRAFIA


[1] Kaufman A. "Efficient Algorithms for Scan-Converting 3D Polygons". *Computer & Graphics*, 12(2):213–219, 1988.

[2] Ekoule A.B., Peyrin F.C. e Odet C.L. "A Triangulation Algorithm from Arbitrary Shaped Multiple Planar Contours". *ACM Transactions on Graphics*, 10(2):182–199, Apr 1991.

[3] Battaiola A.L. "Implementação de um Algoritmo Vetorizado para a Geração de Superfícies Tridimensionais". In *SIBGRAPI'92 - V Simpósio Brasileiro de Computação Gráfica e Processamento de Imagens*, pages 145–154, Águas de Lindóia, SP, Brasil, Nov 1992.

[4] Smith A.R. "Volume Graphics and Volume Visualization: A Tutorial". Technical Report Technical Memo 176, Pixar Inc., San Rafael, California, May 1987.

[5] Falcão A.X., Lotufo R.A. e Gonçalves R.J. "Visualização de Volumes Aplicada à Área Médica". In *SIBGRAPI'92 - V Simpósio Brasileiro de Computação Gráfica e Processamento de Imagens*, pages 125–134, Águas de Lindóia, SP, Brasil, Nov 1992.

[6] Phong B.T. "Illumination for Computer Generated Pictures". *Communication of the ACM : Graphics and Image Processing*, 18(6):311–317, Jun 1975.

[7] Upson C. e Keeler M. "VBUFFER: Visible Volume Rendering". *Computer & Graphics*, 22(4):59–64, Aug 1988.

[8] Pelizzari C.A., Chen G.T.Y., Spelbring D.R., Weichselbaum R.R. e Chen C.T. "Accurate Three-Dimensional Registration of CT, PET, and/or MR Images of the Brain". *Journal of Computer Assisted Tomography*, 13(1):20–26, Jan/Feb 1989.

[9] Geist D. e Vannier M.W. "PC-Based 3D Reconstruction of Medical Images". *Computer & Graphics*, 13(2):135–143, Sep 1989.

[10] Gordon D. e Udupa J.K. "Fast Surface Tracking in Three-Dimensional Binary Images". *Computer Vision, Graphics and Image Processing*, 45(2):196–214, Feb 1989.

[11] Gordon D. e Reynolds R.A. "Image-Space Shading of Three-dimensional Objects". Technical Report MIPG85, Medical Image Processing Group, Dept. of Radiology, Univ. of Pennsylvania, Philadelphia, 1983.

[12] Gordon D. e Reynolds R.A. "Image-Space Shading of Three-dimensional Objects". *Computer Vision, Graphics and Image Processing*, 29:361–376, 1985.

[13] Rogers D.F. *Procedural Elements for Computer Graphics.* McGraw-Hill, 1985.

[14] Goodsell D.S., Mian S. e Olson A.J. "Rendering of Volumetric Data in Molecular Systems". *Molecular Graphics*, 7(1):41–47, Mar 1989.

[15] Artzy E. "Display of Three-Dimensional Information in Computed Tomography". *Computer Graphic and Image Processing*, 9:196–198, 1979.

[16] Keppel E. "Approximating Complex Surfaces by Triangulation of Contour Lines". *IBM Journal of Research and Development*, 19(1):2–11, Jan 1975.

[17] Farrel E.J. "Color Display and Interactive Interpretation of Three-Dimensional Data". *IBM Journal of Research and Development*, 27(4):356–366, Jul 1983.

[18] Farrel E.J., Zappulla R. e Yang W.C. "Color 3-D Imaging of Normal and Pathologic Intracranial Structures". *IEEE Computer Graphics and Applications*, 4(10):5–17, Sep 1984.

[19] Johnson E.R. e Mosher Jr. C.E. "Integration of Volume Rendering and Geometric Graphics". In *State of the Art in Data Visualization, Siggraph'89 tutorial*, Jul 1989.

[20] Hermeline F. "Triangulation Automatique d'un Polyhèdre en Dimension N". *RAIRO, Modél. Math. Analyse Numérique*, 16(3):211–242, 1982.

[21] Frieder G., Gordon D. e Reynolds R.A. "Back-to-Front Display of Voxel Based Objects". *IEEE Computer Graphics and Applications*, 5(1):52–60, Jan 1985.

[22] Jense G.J. e Huijsmans D.P. "Interactive Voxel-Based Graphics For 3D Reconstruction of Biological Structures". *Computer & Graphics*, 13(2):145–150, Fev 1989.

[23] Herman G.T. "From 2D to 3D Representation". *Mathematics and Computer Science in Medical Imaging*, F39:197–220, 1988.

[24] Herman G.T. e Liu H.K. "Dynamic Boundary Surface Detection". *Computer Graphic and Image Processing*, 7:130–138, 1978.

[25] Herman G.T. e Liu H.K. "Three-Dimensional Display of Human Organs from Computed Tomograms". *Computer Graphic and Image Processing*, 9:1–21, 1979.

[26] Herman G.T., Udupa J.K., Kramer D., Lauterbur P.C., Rudin A.M. e Schneider J.S. "Three-Dimensional Display of Nuclear Magnetic Resonance Images". *Optical Engineering*, 21(5):923–926, Sep/Oct 1982.

[27] Fuchs H., Levoy M. e Pizer S.M. "Interactive Visualization of 3D Medical Data". *IEEE Computer*, 22(8):46–50, Aug 1989.

[28] Fuchs H., Kedem Z.M. e Uselton S.P. "Optimal Surface Reconstruction from Planar Contours". *Communications of the ACM*, 20(10):693–702, Oct 1977.

[29] Pedrini H., Almeida e Silva L.M. e Tozzi C.L. "Avaliação da precisão no processo de reconstrução 3D a partir de seções seriadas". In *SIBGRAPI'92 - V Simpósio Brasileiro de Computação Gráfica e Processamento de Imagens*, pages 155–161, Águas de Lindóia, SP, Brasil, Nov 1992.

[30] Cline H.E., Dumoulin C.L., Hart Jr. H.R., Lorensen W.E. e Ludke S. "3D Reconstruction of the Brain from Magnetic Resonance Images Using a Connectivity Algorithm". *Magnetic Resonance Imaging*, 5:245–352, Apr 1987.

[31] Cline H.E., Lorensen W.E., Ludke S., Crawford C.R. e Teeter B.C. "Two Algorithms for the Three-Dimensional Reconstruction of Tomograms". *Medical Physics*, 15(3):320–327, May/Jun 1988.

[32] Tuy H.K. e Tuy L.T. "Direct 2-D Display of 3-D Objects". *IEEE Computer Graphics and Applications*, 4(10):29–33, Oct 1984.

[33] Christiansen H.N. e Sederberg T.W. "Conversion of Complex Contour Line Definitions Into Polygonal Element Mosaics". *Computer & Graphics*, 12(3):187–192, 1978.

[34] O'Rourke J. "Triangulation of Minimal Area of 3D Object Model". In *Proceedings of International Joint Conference on Artificial Intelligence*, pages 664–666, 1981.

[35] Rasure J. e Williams. "An Integrated Visual Language and Software Development Environment". *Visual Languages and Computing*, 2:217–246, 1991.

[36] Boissonnat J.D. "Representing 2D and 3D Shapes with Delaunay Triangulation". In *Proceedings of IEEE and ICASSP*, pages 745–748, 1984.

[37] Boissonnat J.D. "Shape Reconstruction from Planar Cross Sections". *Computer Vision, Graphics and Image Processing*, 44:1–29, 1988.

[38] Foley J.D., Van Dam A., Freiner S.K. e Hughes J.F. *Fundamentals of Interactive Computer Graphics.* Addison Wesley Publishing Company, 2nd edition, 1990.

[39] Udupa J.K. "Computer Aspects of 3D Imaging in Medicine: A Tutorial". Technical Report MIPG 153, Medical Image Processing Group,Dept. of Radiology, Univ. of Pennsylvania, Philadelphia, 1989.

[40] Udupa J.K. e Ajjanagadde V.G. "Boundary and Object Labeling in Three-Dimensional Images". *Computer Vision, Graphics and Image Processing*, 51:355–369, 1990.

[41] Toennies K.D., Udupa J.K., Herman G.T., Wornom III I.L. e Buchman S.R. "Registration of 3D Objects and Surfaces". *IEEE Computer Graphics and Applications*, pages 52–62, May 1990.

[42] Hohne K.H. e Bernstein R. "Shading 3D Images from CT Using Gray-Level Gradients". *IEEE Transactions on Medical Images*, MI-5(1):45–47, Mar 1986.

[43] Chuang K.S., Udupa J.K. e Raya S.P. "High-Quality Rendering of Discrete Three-dimensional Surfaces". Technical Report MIPG130, Medical Image Processing Group, Dept. of Radiology, Univ. of Pennsylvania, Philadelphia, 1988.

[44] Westover L. "Interactive Volume Rendering". In *Workshop on Volume Visualization*, pages 9–16, Chapel Hill, North Carolina, USA, May 1989.

[45] Westover L. "Footprint Evaluation for Volume Rendering". *Computer & Graphics*, Aug 1990. In Press.

[46] Chen L.-S. e Sontag M.R. "Representation, Display and Manipulation of 3D Digital Scenes and Their Medical Applications". *Computer Vision, Graphics and Image Processing*, 48(2):190–216, Nov 1989.

[47] Moura Jr. L.A. *System for Reconstruction, Handling and Display of Three-Dimensional Medical Structures.* PhD thesis, University of London, Nov 1988.

[48] Harris L.D., Robb R.A., Yuen T.S. e Ritman E.L. "Non-Invasive Numerical Dissection and Display of Anatomic Structure Using Computerized X-ray Tomography". In *Proceedings of the Society of Photo-Optical Instrumentation Engineers*, volume 152, pages 10–18, Bellingham,WA, 1978. SPIE.

[49] Magalhães L.P. *Computação Gráfica: Interfaces em Sistemas de Computação Gráfica.* ed. UNICAMP/PAPIRUS, 1986.

[50] Schad L.R., Boesecke R., Schlegel W., Hartmann G.H., Sturm V., Strauss L.G. e Lorenz W.J. "Three Dimensional Image Correlation of CT, MR, and PET Studies in Radiotherapy Treatment Planning of Brain Tumors". *Journal of Computer Assisted Tomography*, 11(6):948–954, Nov/Dec 1987.

[51] Chen L.S., Herman G.T., Meyer C.R., Reynolds R.A. e Udupa J.K. "3D83: An Easy-to-Use Software Package for Three-Dimensional Display from Computed Tomograms". In *Proceedings of International Symposium on Medical Images and Icons*, pages 309–316, Arlington,Virginia, 1984. IEEE.

[52] Chen L.S., Herman G.T., Reynolds R.A. e Udupa J.K. "Surface Shading in the Cuberille Environment". *IEEE Computer Graphics and Applications*, 5(12):33–43, Dec 1985.

[53] Cook L.T., Dwyer III S.J., Batnitzky S. e Lee K.R. "A Three-Dimensional Display System for Diagnostic Imaging Applications". *IEEE Computer Graphics and Applications*, 3:13–19, 1983.

[54] Bergström M., Boëthius J., Eriksson L., Greitz T., Ribbe T. e Widén L. "Head Fixation Device for Reproducible Position Alignment in Transmission CT and Positron Emission Tomography". *Journal of Computer Assisted Tomography*, 5(1):136–141, Feb 1981.

[55] Levoy M. "Display of Surfaces from Volume Data". *IEEE Computer Graphics and Applications*, 8(3):29–37, May 1988.

[56] Levoy M. "Efficient Ray Tracing of Volume Data". *ACM Transactions on Graphics*, 9(3):245–261, Jul 1990.

[57] Levoy M. "A Hybrid Ray Tracer for Rendering Polygon and Volume Data". *IEEE Computer Graphics and Applications*, 10(2):33–40, Mar 1990.

[58] Levoy M. "Volume Rendering by Adaptative Refinement". *Visual Computer*, 6(1):2–7, Fev 1990.

[59] Levoy M., Hanrahan P., Hoehne K.H., Kaufman A. e Lorensen W. "Course Notes: Volume Visualization Algorithms and Architectures". In *SIGGRAPH 1990, 17th International Conference On Computer Granphics and Interactive Techniques*, Dallas, Aug 6th-10th 1990.

[60] Shantz M. "Surface Definition for Branching Contour-Defined Objects". *Computer & Graphics*, 15(2):242–270, Jul 1981.

[61] Zyda M.J., Jones A.R. e Hogan P.G. "Surface Construction from Planar Contours". *Computer & Graphics*, 11(4):393–408, 1987.

[62] Stytz M.R., Frieder G. e Frieder O. "Three-Dimensional Medical Imaging: Algorithms and Computer Systems". *ACM Computing Surveys*, 23(4):421–499, Dec 1991.

[63] Sabella P. "A Rendering Algorithm for Visualizing 3D Scalar Fields". *Computer & Graphics*, 22(4):51–58, Aug 1988.

[64] Heffernan P.B. e Robb R.A. "A New Method for Shaded Surface Display of Biological and Medical Images". *IEEE Transactions on Medical Images*, MI-4:26–38, 1985.

[65] Fox P.T., Perlmutter J.S. e Raichle M.E. "A Stereotactic Method of Anatomical Localization for Positron Emission Tomography". *Journal of Computer Assisted Tomography*, 9(1):141–153, Jan/Feb 1985.

[66] Drebin R.A., Carpenter L. e Hanrahan P. "Volume Rendering". *Computer & Graphics*, 22(4):65–74, Aug 1988.

[67] Lotufo R.A., Falcão A.X. e Gonçalves R.J. "Expansão do Ambiente Khoros para Visualização de Estruturas Biomédicas 3D". In *I Fórum Nacional de Ciência e Tecnologia em Saúde'92*, pages 288–291, Caxambú, MG, Brasil, Nov 1992.

[68] Lotufo R.A., Herman G.T. e Udupa J.K. "Combining Gray-level into Shape-based Interpolation". In *Visualization in Biomedical Computing 1992 (VCB'92)*, pages 13–16, Chapel Hill, North Carolina, USA, Oct 1992.

[69] Reynolds R.A. *Fast Methods for 3D Display of Medical Objects*. PhD thesis, Dept. of Computer and Information Science, Univ. of Pennsylvania, Philadelphia, 1985.

[70] Reynolds R.A., Gordon D. e Chen L.-S. "A Dynamic Screen Technique for Shaded Graphics Display of Slice-Represented Objects". *Computer Vision, Graphics and Image Processing*, 38(3):275–298, Jun 1987.

[71] Robb R.A. e Baarillot C. "Interactive 3–D Image Display and Analysis". In *Proceedings of SPIE: Hibrid Image and Signal Processing*, volume 939, pages 173–202, 1988.

[72] Gonzalez R.C. e Wintz P. *Digital Image Processing*. Addison Wesley Publishing Company, 2nd edition, 1987.

[73] Gallagher R.S. e Nagtegaal J.C. "An Efficient 3-D Visualization Technique for Finite Element Models and Other Coarse Volumes". *Computer & Graphics*, 23(3):185–194, Jul 1989.

[74] Ganapathy S. e Dennehy T.G. "A New General Triangulation Method for Planar Contours". *ACM Computer Graphics*, 16(3):69–75, 1982.

[75] Goldwasser S. "Rapid Techniques for the Display and Manipulation of 3-D Biomedical Data". In *Proceeding of NCGA'86 Tutorial*, Anaheim, CA, May 1986.

[76] Raya S. "SOFTVU – a Software Package for Multidimensional Medical Image Analysis". In *Proceedings of SPIE*, volume 1232, pages 162–166, 1990.

[77] Kim S.-D., Lee J.-H. e Kim J.-K. "A New Chain-Coding Algorithm for Binary Images Using Run-Length Codes". *Computer Vision, Graphics and Image Processing*, 41:114–128, 1988.

[78] Goldwasser S.M. e Reynolds R.A. "Real-Time Display and Manipulation of 3-D Medical Objects: The Voxel Processor Architecture". *Computer Vision, Graphics and Image Processing*, 39:1–27, 1987.

[79] Raya S.P. e Udupa J.K. "Shape-Based Interpolation of Multidimensional Objects". *IEEE Transactions on Medical Images*, 1989.

[80] Elvins T. "Tutorial on Volume Visualization". Transparências do SIBGRAPI'91 - IV Simpósio Brasileiro de Computação Gráfica e Processamento de Imagens, Jul 1991.

[81] Porter T. e Duff T. "Compositing Digital Images". *Computer & Graphics*, 18(3):253–259, Jul 1984.

[82] Tiede U., Hoehne K.H., Bomans M.. Pommert A., Riemer M. e Wiebecke G. "Surface Rendering - Investigation of Medical 3D-Rendering Algorithms". *IEEE Computer Graphics and Applications*, pages 41–53. Mar 1990.

[83] W.A.Barrett, Raya S.P. e Udupa J.K. "A Low-Cost PC-Based Image Workstation for Dynamic Interactive Display of Three-Dimensional Anatomy". In *Proceedings of the Society of Photo-Optical Instrumentation Engineers*, volume 1091, pages 346–355, Bellingham,WA, 1989. SPIE.

[84] Lin W.C., Chen S.Y. e Chen C.T. "A New Surface Interpolation Technique for Reconstruction 3D Objects from Serial Cross-Sections". *Computer Vision, Graphics and Image Processing*, 48(1):124–143, Oct 1989.

[85] Lorensen W.E. e Cline H.E. "Marching Cubes: A High Resolution 3D Surface Construction Algorithm". *Computer & Graphics*, 21(4):163–169, Jul 1987.

[86] Trousset Y. e Schmitt F. "Active-Ray Tracing for 3D Medical Imaging". In *Proceedings of Eurographics'87*, 1987. ed. G.Marechal, Elsevier Science Publishers B.V., North-Holland.