UNIVERSIDADE DE SÃO PAULO

FFCLRP - DEPARTAMENTO DE FÍSICA E MATEMÁTICA

PROGRAMA DE PÓS-GRADUAÇÃO EM FÍSICA APLICADA À MEDICINA E BIOLOGIA

Ana Paula Perini

Software para orientação de neurocirurgia guiada por um transdutor espacial 3D

v.1

Ribeirão Preto - SP
2007

ANA PAULA PERINI

Software para orientação de neurocirurgia guiada por um transdutor espacial 3D

Dissertação apresentada ao Departamento de Física e Matemática da Faculdade de Filosofia, Ciências e Letras de Ribeirão Preto da Universidade de São Paulo para a obtenção do título de Mestre em Ciências

Área de Concentração: Física Aplicada à Medicina e Biologia.
Orientador: Prof. Dr. Antonio Adilton Oliveira Carneiro.

v.1

Ribeirão Preto - SP
2007

FICHA CATALOGRÁFICA

Perini, Ana Paula.

Software para orientação de neurocirurgia guiada por um transdutor espacial 3D. Ribeirão Preto, 2007, p. 136.

Dissertação de Mestrado, apresentada ao Departamento de Física e Matemática da Faculdade de Filosofia, Ciências e Letras de Ribeirão Preto / USP. Área de concentração: Física Aplicada à Medicina e Biologia.

Orientador: Antonio Adilton Oliveira Carneiro.

1. Ressonância Magnética. 2. Visualization Toolkit (VTK). 3. Epilepsia. 4. Visualização Volumétrica.

Dedicatória

Aos meus pais, que mesmo longe, sempre me deram apoio e conforto e toda a dedicação para que eu desenvolvesse este trabalho.

Ao meu namorado e companheiro de todas as horas, que sempre me deu amor, carinho e confiança para que eu realizasse este trabalho.

Agradecimentos

Ao Senhor Deus por ter me dado força e perseverança durante o mestrado.

Ao meu orientador, Prof. Dr. Antonio Adilton Carneiro, pela orientação e apoio que contribuiu para meu desenvolvimento científico.

Ao Prof. Dr. Hélio Rubens Machado pela confiança, dedicação e apoio na realização da pesquisa.

Ao Centro de Pesquisas Renato Archer pelo apoio científico durante o desenvolvimento deste trabalho, em especial ao técnico Sr. Glauco Silva, Dr. Aílton Santa Bárbara e Ms. Jorge Vicente Lopes da Silva.

Ao Prof. Dr. Lucas Ferrari de Oliveira pelas discussões, amizade e apoio científico.

Aos meus amigos Prof. Dr. Evamberto Garcia de Góes e Marlene Cabral pela amizade, conselhos e pela ajuda nos momentos difíceis.

À Coordenação de Aperfeiçoamento de Pessoal de Nível Superior pelo apoio financeiro.

A todos que me apoiaram na realização deste projeto.

Aos colegas do grupo GIIMUS pela amizade.

À minha família pelo incentivo, amor e permanente dedicação.

E ao meu namorado, Lucio, que por fazer parte de minha vida sempre me incentivou, me ajudando nos momento mais difíceis.

Muito Obrigada a todos.

A sabedoria consiste em compreender que o tempo dedicado ao trabalho nunca é perdido.

Ralph Emerson.

Resumo

Neurocirurgia guiada por imagem permite ao neurocirurgião navegar dentro do cérebro do paciente, usando imagens pré-operatórias como orientação, através do uso de sistemas de rastreamento, durante o procedimento cirúrgico. Muitos sistemas desenvolvidos para neurocirurgia guiada por imagem, empregam imagens pré-operatórias para fornecer orientação ao cirurgião, durante o procedimento cirúrgico. Seguindo um procedimento de calibração, a posição tridimensional e orientação dos instrumentos cirúrgicos podem ser transmitidas ao computador. Estas informações espaciais são usadas para acessar a região de interesse nas imagens pré-operatórias com a finalidade de apresentá-las ao cirurgião durante o procedimento. Contudo, quando ocorre a craniotomia, o movimento dos tecidos do cérebro pode ser fonte significativa de erro nestes sistemas. A arquitetura implementada neste trabalho visa o desenvolvimento de um sistema que permite planejamento e orientação cirúrgica. Para orientação cirúrgica foi desenvolvido um *software* que permite extrair fatias do volume de imagens de ressonância magnética (IRM), com orientação fornecida por um transdutor de posição magnético (*Polhemus*®). As fatias extraídas serão, futuramente, correlacionadas com imagens de ultra-som (IUS) intra-operatórias para detectar e corrigir a deformação do tecido cerebral durante a cirurgia. A ferramenta para navegação pré-cirúrgica foi desenvolvida para fornecer três fatias ortogonais obtidas através do volume de imagens. Na metodologia usada para a implementação do *software*, foi utilizada a linguagem de programação *Python* e a biblioteca gráfica *Visualization Toolkit* (VTK). Os resultados mostraram que o programa de planejamento pré-cirúrgico, gerou uma alta resolução na visualização dos planos ortogonais e oblíquos das IRM, além de ser rápido e interativo. O programa de extrair fatias do volume de IRM permitiu a aplicação de transformações ao volume, com base nos valores de coordenadas fornecidos pelo transdutor de posição.

Abstract

Image guided neurosurgery enables the neurosurgeon to navigate inside the patient's brain using pre-operative images as a guide and a tracking system, during surgical procedure. Many image guided neurosurgery implementations employ pre-operative images as a guide to the surgeons throughout surgical procedure. Following a calibration procedure, three-dimensional position and orientation of surgical instruments may be transmitted to computer. The spatial information is used to access an interest region, in the pre-operative images, displaying them to the neurosurgeon during the surgical procedure. However, when a craniotomy is involved, movements of brain tissue can be a significant source of error in these systems. The architecture implemented in this work intends the development of a system to surgical planning and orientation. For surgical orientation, the software developed allows the extraction of slices from the volume of the magnetic resonance images (MRI) with orientation supplied by a magnetic position sensor (*Polhemus*®). In the future, the extracted slices will be correlated with intra-operative ultrasound images to detect and to correct the deformation of brain tissue during the surgery. Also, a tool for pre-operative navigation was developed, providing three orthogonal planes through the image volume. In the methodology used for the software implementation, the Python programming language and the Visualization Toolkit (VTK) graphics library were used. The results showed that the program of pre-operative navigation had high resolution in the visualization of orthogonal and oblique MRI planes. Furthermore, it was fast and interactive. The program to extract slices of the MRI volume allowed the application of transformations in the volume, using coordinates supplied by the position sensor.

Lista de Figuras

Sumário

Capítulo 1
Introdução

1.1 Motivação

1.1.1 Planejamento Pré-Cirúrgico

O planejamento pré-cirúrgico é indispensável para um bom resultado operatório em neurocirurgia. Por isso, um dos fundamentos do ato operatório consiste em determinar previamente o local em que deverá ser realizada a abertura do crânio (craniotomia), de tal forma que o acesso ao conteúdo intracraniano seja preciso, permitindo uma exploração segura através de sulcos e cisternas sem o comprometimento dos tecidos sadios. Muitas vezes, a superfície convexa do crânio pode propiciar erros de interpretação, gerando divergências no momento da transferência dos dados observados nas imagens bidimensionais da tomografia computadorizada (ITC) ou ressonância magnética (IRM), para o alvo correspondente, localizado logo abaixo da superfície da calota craniana. Há inúmeros relatos de acessos cirúrgicos insuficientes e deslocados do centro da lesão, ou, mais comumente, acessos feitos com amplas abordagens, expondo áreas sem interesse, fato que pode contribuir para aumentar as chances de lesões inadvertidas do cérebro ou de estruturas vasculares.

Mais recentemente, avanços no campo da radiologia e da informática permitiram reconstruir imagens tridimensionais (3D) geradas a partir de imagens tomográficas. As imagens podem ser pós-processadas de diversas formas, para melhor demonstração de lesões intracranianas e de suas relações com os elementos ósseos ou com outros acidentes anatômicos naturais, possibilitando, assim, uma melhor visão espacial do que a fornecida convencionalmente pelos cortes axiais.

Tipicamente, em imagens médicas, o dado volumétrico é apresentado como uma seqüência de fatias bidimensionais (*slices*, normalmente com resolução de 512x512 pontos), separadas por uma pequena distância (poucos milímetros). Cada fatia representa uma seção transversal da região de interesse e o conjunto das fatias permite a construção de uma imagem 3D desta região, auxiliando o médico no diagnóstico de uma doença, no tratamento mais adequado ou no planejamento de uma cirurgia.

Visando atender às necessidades de planejamento neurocirúrgico em pacientes com epilepsia, foi desenvolvido neste trabalho um *software* que permite navegar no volume de IRM de forma interativa. Utilizando métodos de renderização de volume, o volume de IRM foi reconstruído. Mediante a utilização de planos de corte sobre este volume, é permitido ao usuário o livre posicionamento destes planos. À medida que estes planos são movidos, as fatias correspondentes são imediatamente visualizadas em três janelas, que mostram os cortes sagital, coronal e axial.

1.1.2 Orientação Cirúrgica

Orientação precisa e localização de ferramentas cirúrgicas dentro do cérebro são essenciais para o sucesso de vários procedimentos neurocirúrgicos, como biópsia e ressecção tumoral. Além disso, a mínima interferência dentro do tecido cerebral saudável reduz o risco de complicações pós-operatórias para o paciente.

Nos dias de hoje, em muitas situações, o procedimento cirúrgico é acompanhado por alguma modalidade de imagem médica. Cirurgias guiadas por imagem permitem ao cirurgião navegar dentro do cérebro do paciente usando imagens pré-operatórias e estruturas estereotáticas para sua orientação [Comeau *et al*. 2000]. Sistemas estereotáticos fornecem orientação para o cirurgião com base em imagens tomográficas, como ITC e IRM. Existem vários tipos de estruturas estereotáticas, porém todas elas fornecem a

mesma função básica, a de fornecer um sistema de referência rígido que estabelece um sistema de coordenadas relativo ao paciente [Peters 2006]. O primeiro sistema estereotático foi denominado *frames*, este foi aderido à cabeça do paciente durante a aquisição das imagens pré-operatórias e durante o procedimento cirúrgico [Kitchen *et al*.1993]. O sistema estereotático tem um sistema de coordenadas 3D inerente, que é associado, através de transformações de coordenadas, com o sistema de coordenadas das imagens pré-operatórias. Apesar dos sistemas estereotáticos oferecerem alta precisão, eles apresentavam algumas desvantagens, como [Pagoulatos *et al*.1999]:

- São volumosos e, portanto, interferem no procedimento cirúrgico;
- O planejamento cirúrgico e localização do alvo nas coordenadas da estrutura estereótatica demandam muito tempo;
- São invasivos.

Com o avanço da tecnologia, uma nova geração de estruturas estereotáticas foi desenvolvida [Pagoulatos *et al.* 1999]. Estes sistemas usam um transdutor de posição para, interativamente, orientar a posição de uma ferramenta cirúrgica durante o caminho da cirurgia. A principal limitação encontrada na utilização de estruturas estereotáticas é que a orientação intra-operatória é baseada em imagens pré-operatórias, o que torna o sistema impreciso, visto que o cérebro sofre deformações após a craniotomia [Comeau *et al*. 1998].

Imagens intra-operatórias do cérebro tem sido uma solução alternativa, fornecendo ao cirurgião imagens em tempo real durante a cirurgia, como ITC [Butler *et al.*1998], IRM [Tronnier *et al*. 1997] e IUS [Comeau *et al*. 2000]. A utilização do ultrasom em cirurgia oferece vantagens em relação às outras modalidades, visto que não utiliza radiação ionizante, é de fácil manuseio em salas cirúrgicas e fornece imagens em tempo real [Pagoulatos *et* al. 1999].

Através da análise de vários artigos, como os citados anteriormente, pôde-se perceber que existem muitas pesquisas na área de neurocirurgia guiada por imagem. Nestes artigos, são usadas várias metodologias, alguns fazem uso de transdutores de posição óptico e imagens pré-operatórias [Adams *et al*. 1996], já outros, fazem uso de imagens intra-operatórias além do transdutor ótico e das imagens pré-operatórias [Comeau *et al*. 1998, Gobbi *et* al. 1999, Gobbi *et al*. 2000 e Comeau *et al*. 2000]. O que se pôde observar mediante as metodologias descritas nos artigos é que a grande maioria faz uso de transdutores óticos.

As cirurgias guiadas por imagem envolvem etapas muito complexas para seu desenvolvimento, como a implementação de um *software* específico, desenvolvimento de um *phantom* com marcadores que simule o cérebro, tanto em IUS como em IRM, que servirá para auxiliar nos procedimentos de calibração. Desta forma, foram feitas neste projeto, algumas etapas, desenvolvidas em conjunto com outro projeto, para futuramente alcançar um sistema de neuronavegação completo.

No sistema de neuronavegação, que está em desenvolvimento, o médico terá acesso as IRM pré-operatórias e IUS intra-operatórias em tempo real para corrigir qualquer tipo de deslocamento ocorrido após a craniotomia. Através do uso de um transdutor de posição magnético, aderido ao transdutor do ultra-som, será possível comparar as IRM e IUS em tempo real. É importante salientar, que este sistema difere dos sistemas de neuronavegação comercias já existentes, pois o mesmo faz uso de um transdutor magnético 3D (*Polhemus*®), enquanto que os demais sistemas fazem uso de transdutores óticos. Este diferencial é importante quando se considera o aspecto financeiro, visto que transdutores magnéticos são mais baratos. Além disso, deve ser levado em conta o fato de a sala cirúrgica ter um espaço limitado, sendo melhor optar por sistemas mais compactos, como o caso dos transdutores magnéticos.

Neste projeto, além das ferramentas de visualização em 3D, foi realizada a implementação de um *software* que permite a extração de fatias do volume de IRM, que é uma parte do sistema de neuronavegação cirúrgica em desenvolvimento. Para a extração destas fatias foi utilizada uma transformação calculada a partir das coordenadas espaciais obtidas com um transdutor de posição magnético 3D.

1.2 Objetivos

O objetivo deste trabalho é parte de um projeto maior: “Neuronavegação em Cirurgias de Epilepsia”. Uma outra etapa deste projeto, que está sendo desenvolvido por outro aluno de mestrado, consiste na construção de um *phantom* de cabeça e calibração do sistema de posicionamento 3D. Com o término destas duas etapas, serão realizados os procedimentos de calibração adequados.

O principal objetivo deste trabalho é o desenvolvimento de um programa computacional rápido e interativo de planejamento e orientação cirúrgica, em casos de epilepsia.

Para alcançar este objetivo, foram realizados os seguintes objetivos secundários:

- Estudar os métodos de visualização tridimensional que existem e analisar suas características (qualidade da imagem, eficiência);
- Aprender a linguagem de programação Python;
- Aprender a utilizar a ferramenta de visualização científica *Visualization Toolkit* (VTK);
- Investigar como funciona um transdutor de posição magnético.

1.3 Contribuições

As principais contribuições deste trabalho foram:

- A implementação de *softwares* para planejamento e orientação cirúrgica;
- A utilização da linguagem Python, o que possibilita, futuramente, tornar o *software* implementado um produto comercial;
- O desenvolvimento de *softwares* interativos, obtidos através da utilização de uma ferramenta de visualização científica;
- A possibilidade de trazer, ao grupo de pesquisa, uma nova linguagem de programação, além de ferramentas de visualização, como VTK, muito importantes para processamento de imagens médicas;
- A aplicação do programa desenvolvido em outros projetos como: elastografia de cérebro e reconstrução de IUS para visualização 3D.

1.4 Organização do Documento

- **Capítulo 2**: Revisão de alguns conceitos envolvidos em epilepsia;
- **Capítulo 3**: Princípios básicos das IRM e suas aplicações. Este capítulo não compromete o entendimento do trabalho, visto que as IRM só foram utilizadas para validar o programa;
- **Capítulo 4**: Revisão sobre transformações geométricas;
- **Capítulo 5**: Principais métodos de visualização volumétrica em medicina;
- **Capítulo 6**: Revisão das ferramentas de visualização científica existentes, dando destaque ao VTK;
- **Capítulo 7**: Revisão dos tipos de transdutores de posição existentes;

- **Capítulo 8**: Listagem e explicação dos materiais utilizados, bem como a metodologia empregada;
- **Capítulo 9**: Discussão e análise dos resultados;
- **Capítulo 10**: Conclusões sobre o trabalho e sugestões de trabalhos futuros.

Capítulo 2

Epilepsia

A epilepsia é um distúrbio do cérebro que se expressa por crises epilépticas repetidas. A crise epilética é resultado de uma descarga súbita e excessiva dos neurônios cerebrais. O tratamento das epilepsias é feito inicialmente através de medicamentos e, quando estes não são efetivos, a cirurgia passa a ser a forma de tratamento mais viável. Neste capítulo serão abordados os fundamentos para o conceito, causas e tratamento da epilepsia.

2.1 Conceito

Não há uma definição completamente satisfatória de epilepsia. Trata-se, geralmente de uma condição crônica, compreendendo um grupo de doenças que têm em comum crises epilépticas, que ocorrem na ausência de doença tóxico-metabólica.

Crises epilépticas são eventos clínicos que refletem disfunção temporária de uma pequena parte do cérebro (crises locais) ou de uma área mais extensa, envolvendo os dois hemisférios cerebrais (crises generalizadas). A crise epiléptica é causada por descarga anormal excessiva e transitória das células nervosas. Os sintomas de uma crise dependem das partes do cérebro envolvidas na disfunção febril.

Crises epilépticas são sintomas comuns de doenças neurológicas agudas (tais como meningoencefalite, trauma cranioencefálico, doenças cerebrovasculares) ou doenças clínicas (tais como anóxia, estado hipoglicêmico não cetótico, insuficiência renal ou hepática) [Guerreiro e Guerreiro 1996].

2.2 A Epilepsia do Lobo Temporal Mesial

Epilepsia do lobo temporal mesial (ELTM), na maioria das vezes, começa na infância tardia ou adolescência. Há geralmente história de convulsão febril [Janszky *et al*. 2003]. Embora os pacientes possam apresentar-se com crises secundariamente generalizadas, o tratamento com medicamentos suprime a maioria destas, e as crises parciais complexas tornam-se o tipo predominante [Cendes *et al*. 1993a].

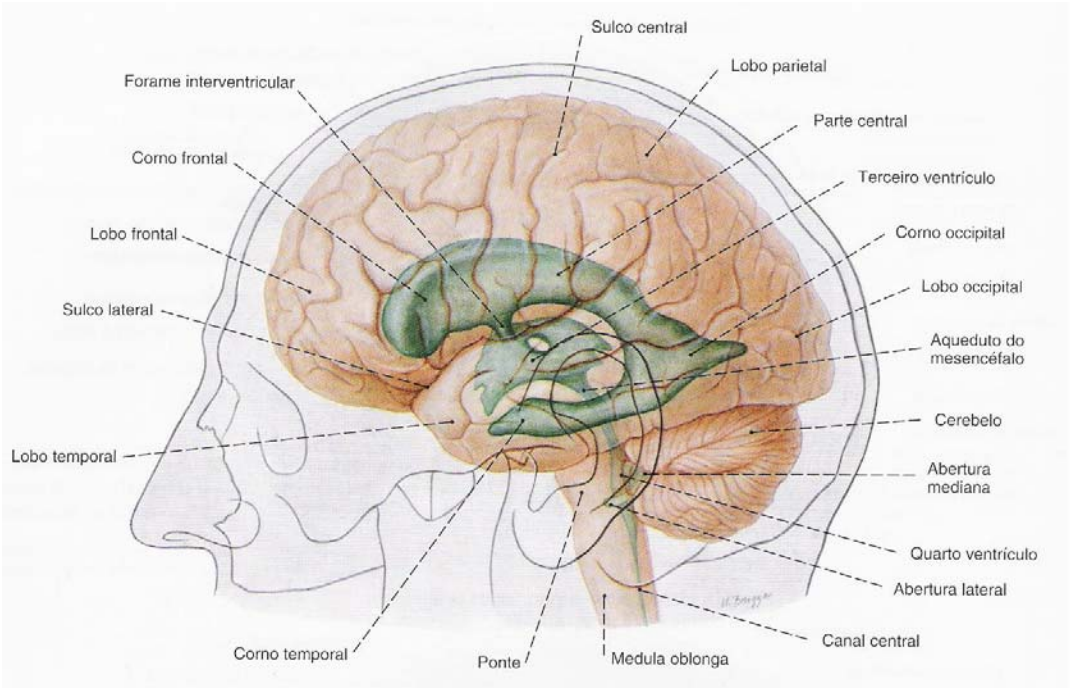
A ELTM é a forma mais comum de epilepsia parcial em adultos jovens e também a mais freqüente em cirurgias. Este tipo de epilepsia é originado a partir de padrões anormais de descargas elétricas na matéria cinzenta e suas causas ainda estão sendo investigadas [Webb *et al*. 1999]. A esclerose mesial temporal (EMT) é considerada principal causa de ELTM; presente em 60-70% dos pacientes submetidos ao tratamento cirúrgico por crises refratárias [Cendes 2005].

A etiologia da EMT ainda é uma questão não resolvida, embora alguns fatores precipitantes estejam presentes na maioria dos casos como crise febril prolongada (em aproximadamente 60% dos casos), infecções e traumas [Cendes *et al*. 1993a].

Dentre os parâmetros usados para interpretação da EMT pode-se destacar: atrofia do lóbulo temporal anterior, assimetria dos cornos temporais, atrofia hipocampal e alteração da intensidade de sinal do hipocampo. Os dois primeiros parâmetros citados anteriormente não são sinais específicos para EMT, já a atrofia hipocampal e as alterações de sinal são os indicadores mais confiáveis [Guerreiro e Guerreiro 1996].

2.3 A Epilepsia do Lobo Temporal: Aspectos em Ressonância Magnética

A introdução do método de ressonância magnética (RM), desde 1986, tem permitido um estudo mais preciso das patologias do Sistema Nervoso Central. A RM é uma técnica de obtenção de imagens digitalizadas, que permite estudar especialmente as partes moles com mais detalhes do que a tomografia computadorizada (TC) e não utiliza radiação, evitando, portanto, os artefatos por difração de raio X nas estruturas próximas ao osso. O estudo da localização lóbulo temporal (Figura 2.1), foi amplamente facilitado com esta técnica de imagem. Além disso, a RM possibilita a obtenção de imagens em múltiplos planos, sem a mudança de decúbito do paciente, o que permite a visualização das estruturas temporais mesiais em plano coronal, fundamental para análise destas estruturas.



[Figura 2.1] Localização do lobo temporal (Figura retirada de [Sobotta 1995]).

A utilização da RM modificou a atitude dos profissionais que trabalham com epilepsias resistentes ao tratamento clínico. A RM não apresenta riscos de toxicidade biológica conhecidos, e portanto pode ser aplicada em qualquer idade, além disso, é comprovadamente a técnica neurodiagnóstica mais eficaz para identificação de lesões vasculares, tumorais e distúrbios de migração neuronal. Este exame é também capaz de detectar alterações estruturais associadas com EMT [Bruton 1988]. Tais alterações incluem atrofia e distúrbios da morfologia das estruturas temporais mesiais. Até o advento da RM, tais patologias não podiam ser diagnosticadas com precisão antes do exame histopatológico. A presença de alterações na RM compatíveis com EMT revolucionou a avaliação diagnóstica e pré-operatória de pacientes com ELTM [Jack 1990].

Diversos trabalhos na literatura têm proposto que as estruturas mesiais do lóbulo temporal podem ser bem visualizadas pela RM e que a presença de atrofias está correlacionada com as alterações patológicas da EMT [Kuzniecky *et al*. 1987]. Mais recentemente, vários trabalhos demonstraram que os estudos volumétricos computadorizados, usando RM, permitem detectar atrofia das estruturas mesiais do lóbulo temporal. Além disso, o grau de atrofia está correlacionado com a intensidade das alterações vistas no exame histopatológico [Cendes *et al*. 1993b].

2.4 O Tratamento Cirúrgico

A cirurgia de epilepsia foi a modalidade terapêutica anti-epiléptica que mais se desenvolveu nas últimas décadas e a que mais se beneficiou do extraordinário desenvolvimento científico-tecnológico ocorrido no período. Melhorou significativamente a capacidade de investigação e diagnóstico, em particular pelo advento de sofisticadas técnicas de monitoração vídeo-eletroencefalográfica e de

neuroimagem estrutural e funcional. Embora a cirurgia de epilepsia não seja uma modalidade terapêutica propriamente nova, somente nos últimos anos é que deixou de ser uma forma de tratamento quase experimental e restrita a um número limitadíssimo de centros.

O maior objetivo da cirurgia de epilepsia é o de promover a melhoria na qualidade de vida dos pacientes epilépticos, o que se consegue usualmente, porém, não exclusivamente, com o controle ou redução significativa da freqüência das crises, através de cirurgias ditas curativas (visam à cura das epilepsias, como por exemplo, as ressecções focais) ou mesmo paliativas (buscam primordialmente o alívio das crises, como por exemplo, a calosotomia) [Guerreiro e Guerreiro 1996].

O processo de avaliação cirúrgica deve permitir a localização da zona epileptogênica (região cerebral responsável pela geração das crises) tão precisamente quanto possível, incluindo seus prováveis limites, e, adicionalmente, permitir a identificação de todo o córtex eloqüente presente no campo da ressecção.

A avaliação cirúrgica se baseia em um conjunto de testes que pode ser dividido em três categorias: 1) estruturais; 2) funcionais e 3) de excitabilidade cortical.

1) Testes estruturais

Visam à detecção de lesões intracerebrais potencialmente epileptogênicas. O principal teste é a IRM que, nos dias atuais, constitui parte obrigatória da avaliação pré-cirúrgica, sendo de grande valia na investigação de pacientes com epilepsia focal [Barkovich *et al*. 1995]. Além de diagnosticar lesões expansivas (tumores), a IRM permite um diagnóstico de outras lesões como: distúrbios de migração neuronal, malformações vasculares, angiomas, cistos, granulomas e, principalmente, a EMT.

A literatura sobre EMT e IRM é bastante extensa. É suficiente aqui mencionar que existem protocolos específicos para otimizar sua detecção e visualização, sendo fatores críticos a orientação dos cortes que devem ser finos, transversais ao maior eixo do hipocampo, com imagens coronais amplificadas e ponderadas em T1 e T2 [Guerreiro e Guerreiro 1996]. Para fins de avaliação pré-cirúrgica é suficiente a detecção da esclerose, já que ela é uma lesão, na maioria das vezes, facilmente detectável à IRM.

2) Testes funcionais

Os testes funcionais mais comuns são os neuropsicológicos e os de neuroimagem funcional.

Avaliação neuropsicológica: conjunto de testes que integram a avaliação neuropsicológica permitindo avaliar, além do quadro cognitivo global (QI global), funções cognitivas específicas, sendo as mais relevantes, para fins de avaliação pré-cirúrgica, a memória, a linguagem e a atenção. Existem alterações cognitivas específicas que sugerem lateralização, por exemplo envolvimento preferencial do hemisfério dominante ou não dominante, e alterações que sugerem localização, por exemplo envolvimento temporal.

Neuroimagem funcional: os testes de neuroimagem funcional mais comumente utilizados na avaliação pré-cirúrgica são a tomografia por emissão de pósitron (PET) e a tomografia por emissão de fóton único (SPECT). O PET comumente utiliza moléculas de glicose marcadas com flúor-18 para permitir medidas diretas da taxa de metabolismo regional. O PET é especialmente útil nos casos de epilepsia de lobo temporal, onde alcança correlação positiva com outros métodos de localização superior a 90%. Suas principais limitações são a baixa resolução temporal e espacial. O SPECT, por outro lado, mede as variações do fluxo sanguíneo cerebral regional e, evidencia,

habitualmente, zonas de hipofluxo no período interictal e zonas de hiperfluxo no período ictal [Guerreiro e Guerreiro 1996].

3) Testes de excitabilidade cortical

Na avaliação pré-cirúrgica não basta demonstrar e localizar a presença de lesões estruturais, zonas de déficit funcional focal ou mesmo zonas irritativas, é extremamente necessário comprovar que estas lesões ou áreas são normalmente excitáveis ou epileptogênicas, ou seja, capazes de gerar crises. Na prática isso é feito através do registro de potenciais marcadores de excitabilidade cortical. O eletroencefalograma (EEG) é o método de registro de potenciais marcadores da epilepsia, sendo imprescindível na definição da zona irritativa, na lateralização e localização da zona epileptogênica [Guerreiro e Guerreiro 1996].

Na prática, o EEG é realizado através da colocação de eletrodos sobre o couro cabeludo do paciente (Figura 2.2), que são conectados a um amplificador de corrente elétrica. Esse amplificador aumenta a amplitude do sinal elétrico gerado pelo cérebro milhares de vezes e, através de um dispositivo chamado galvanômetro, as oscilações são desenhadas numa tira de papel sob a forma de ondas.

Fisiologicamente sabe-se que as características das ondas elétricas cerebrais variam conforme o funcionamento (situação funcional) do órgão. As maiores variações se observam entre os estados de vigilância, ou seja, entre o estar acordado, dormindo, sonolento, em coma, etc.

Para o diagnóstico de epilepsia, o EEG serve para localizar com exatidão os focos epilépticos ou tumores cerebrais. Os focos epilépticos são pequenas regiões no cérebro onde a atividade elétrica se apresenta anormal. Pela observação do EEG, o

neurologista que o interpreta é capaz de deduzir exatamente onde esta anormalidade está situada.

[Figura 2.2] Localização dos eletrodos sobre o couro cabeludo (Figura retirada de [Kgu 2007]).

Capítulo 3

Imagem de Ressonância Magnética

Neste capítulo, serão fornecidos alguns fundamentos físicos para obtenção de imagens utilizando a técnica de ressonância magnética. A descrição dessa técnica será restrita àquelas necessárias ao entendimento da formação de IRM; mais informações podem ser obtidas na referência [Smith e Ranallo 1989]. Além da apresentação de alguns conceitos físicos, também serão apresentadas algumas aplicações clínicas desta modalidade de imagem.

A IRM é indicada para a visualização de tecidos moles da região da cabeça, do pescoço, do tórax, do abdômen, da pélvis, do sistema muscular e ósseo além da espinha. Este exame possui alguns riscos e contra-indicações, como para pessoas com problemas cardíacos ou com aneurisma. Algumas de suas limitações incluem a necessidade de contraste intravenoso em casos especiais, a sensibilidade ao movimento e a detecção de calcificação. Entre as vantagens que possui pode-se citar a não utilização de radiação ionizante, a capacidade multiplanar, a possibilidade de demonstrar o fluxo sanguíneo e de detectar a má formação intracraniana [Manssour 1998].

3.1 Princípios Básicos da Imagem de Ressonância Magnética

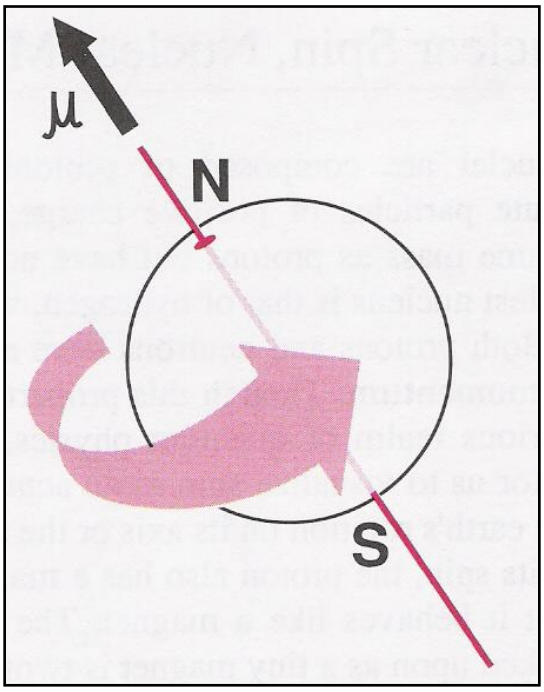
A ressonância é um dos fenômenos mais estudados pela física, pois aparece em quase todos os sistemas mecânicos, acústicos, ópticos, elétricos e magnéticos, sempre que ocorrem estímulos cuja freqüência seja próxima à das suas vibrações naturais. O fenômeno de ressonância manifesta-se também nos níveis moleculares, atômico,

eletrônicos e nucleares. Neste último caso, é de natureza magnética sendo por isso chamada de ressonância magnética nuclear (RMN).

A RMN é um fenômeno puramente quântico, que aqui será tratado, para efeito didático, seguindo um modelo clássico. Apesar de não estar rigorosamente correto, esse modelo gera uma visão mais intuitiva do processo, facilitando a compreensão da formação de imagens por ressonância. O texto a seguir, para explicação dos fundamentos físicos envolvidos na formação de IRM, foi baseado no livro [Smith e Ranallo 1989].

Todos os núcleos atômicos são compostos de prótons e nêutrons. Prótons são partículas pequenas de carga positiva, enquanto os nêutrons apesar de terem a mesma massa dos prótons, não possuem carga. O núcleo do átomo mais simples que existe é o do hidrogênio, que consiste de um único próton. Tanto os prótons quanto os nêutrons têm uma propriedade chamada *spin* ou momento angular.

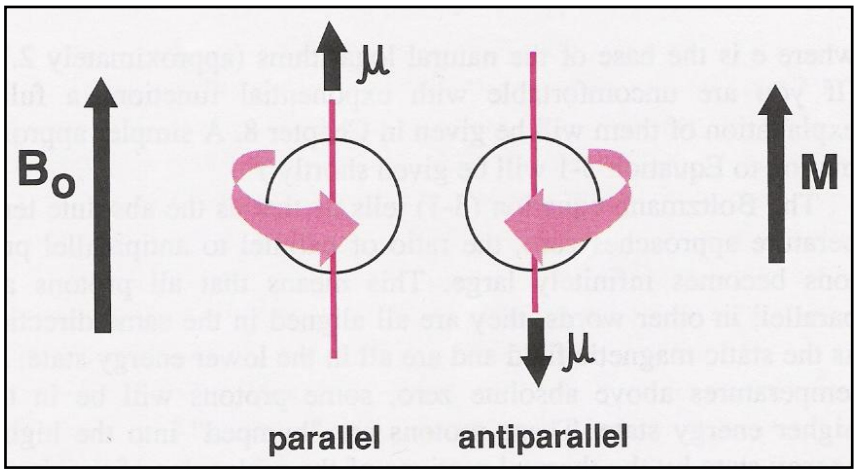
Em adição a estes *spins*, o próton também tem um momento magnético, o que significa que ele se comporta como um pequeno imã. Este comportamento pode ser explicado pelas seguintes razões: (1) o próton tem carga elétrica, e (2) ele rotaciona ao redor de seu próprio eixo em um movimento chamado de *spin.* Qualquer objeto carregado eletricamente em movimento irá gerar um campo magnético ao seu redor, e quando o movimento for de *spin*, o objeto é denominado como sendo um dipolo magnético. Portanto, o próton é um dipolo magnético (Figura 3.1). A intensidade e orientação do dipolo magnético são dadas por um vetor chamado de momento de dipolo magnético ou simplesmente momento magnético. Quando são considerados vários prótons tem-se o momento magnético líquido, que é o vetor soma dos momentos magnéticos de cada um dos prótons. Na figura 3.1 o momento magnético do próton é representado pela seta designada por μ.



[Figura 3.1] Próton visto como um dipolo magnético. O próton carregado positivamente se comporta como um ímã com pólo norte (N) e pólo sul (S). A intensidade e orientação deste ímã são dadas pelo vetor momento magnético, μ (Figura retirada de [Smith e Ranallo 1989]).

Quando se considera o modelo quântico simplificado os momentos magnéticos dos prótons em um campo estático têm somente duas orientações possíveis: são paralelos ou antiparalelos à direção do campo magnético estático (Figura 3.2). As duas orientações possíveis representam dois diferentes níveis de energia do próton. O próton paralelo tem menos energia que o antiparalelo, isto é comum em muitos fenômenos da natureza, onde o estado de menor energia é o preferencial. Portanto, quando certa quantidade de prótons é colocada em um campo magnético estático, os prótons paralelos, em um curto tempo, estarão presentes em maior quantidade que os prótons antiparalelos. Devido ao número excedente de prótons paralelos, surge o momento

magnético líquido que é referenciado como vetor magnetização ou simplesmente magnetização M (Figura 3.2).



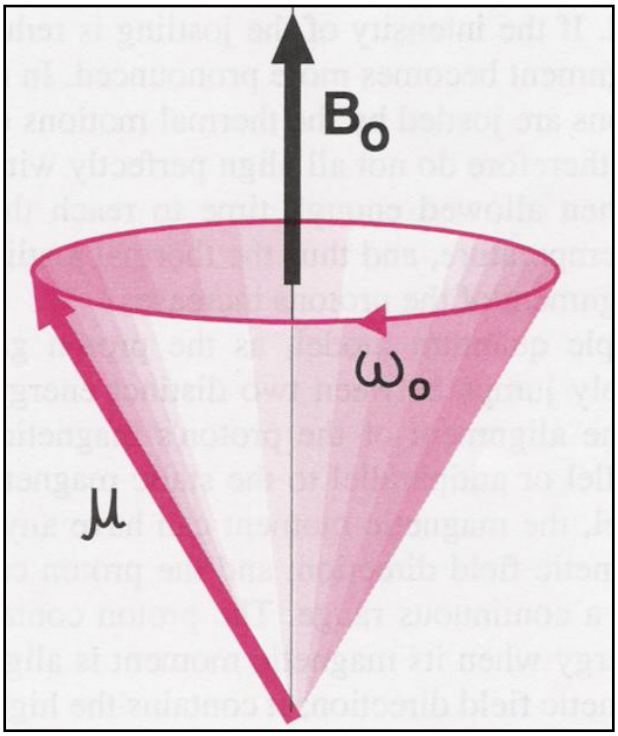
[Figura 3.2] Modelo quântico simplificado do *spin* do próton. Quando os prótons são colocados em um campo magnético estático (B_0) seus momentos magnéticos (μ) poderão estar em somente duas direções: paralelos ou antiparalelos ao campo magnético (Figura retirada de [Smith e Ranallo 1989]).

Na descrição realizada neste trabalho, para explicar os fundamentos físicos, a RMN será visualizada por um modelo clássico. O modelo quântico simplificado auxilia no entendimento das interações entre fótons e prótons, além de auxiliar na percepção do fenômeno da transição entre estados de energia. Contudo, a forma como essas mudanças de estado podem ser medidas não é bem visualizada pela descrição quântica. Para isso, será considerado o uso de um modelo clássico simplificado.

Em um ensemble de momentos magnéticos, na presença de um campo magnético externo, tem-se como resultado um vetor magnetização associado e uma propriedade de spin. Supondo, agora, que o vetor de momento magnético, μ, forma um ângulo com o vetor de campo B_0 aplicado. Essa configuração causa o aparecimento de um torque, na tentativa de alinhar μ e B_0, cuja expressão é dada por:

$$\tau = \mu \times B_0 . \qquad 3.1$$

Contudo, como o sistema exibe a propriedade de *spin*, o torque não irá alinhar os dois vetores coincidentemente, mas provocará o aparecimento de um movimento de precessão em torno do campo, como no caso de um peão girando na presença do campo gravitacional da terra (Figura 3.3).



[Figura 3.3] Diagrama ilustrativo do esquema de precessão de um vetor magnetização na presença de um campo magnético externo, B_0, aplicado (Figura retirada de [Smith e Ranallo 1989]).

Nesse sistema clássico, que possui um contínuo de energia associado, a freqüência de precessão pode ter qualquer valor, e obedece à equação de Larmor, indicando que quanto maior o campo B_0, maior será a freqüência de precessão, ω:

$$\omega = \gamma B_0 , \qquad 3.2$$

sendo γ o fator giromagnético que traduz a razão entre o momento magnético do próton e a contribuição devida a seu *spin*, distinguindo, portanto, diferentes sistemas.

A freqüência angular ω na equação 3.2 é medida em radianos por segundo. É possível reescrever a equação 3.2 em termos da freqüência de precessão (f_0) expressa em ciclos por segundo ou hertz (Hz). A relação entre ω e f_0 é:

$$\omega = 2\pi f_0 . \qquad 3.3$$

A equação 3.2 torna-se, então:

$$2\pi f_0 = \gamma B_0 \quad \text{ou} \quad f_0 = \frac{\gamma}{2\pi} B_0 . \qquad 3.4$$

Felizmente, pode-se tratar esse sistema de momentos magnéticos como exibindo uma magnetização global *M*. Quando *M* está na presença de um campo magnético, ele também exibirá um movimento de precessão análogo, que será governado pela equação:

$$\frac{dM}{dt} = \gamma M \times B_0 . \qquad 3.5$$

Para alterar a orientação da magnetização é necessária a aplicação de um segundo campo magnético (B_1), este circulante polarizado e perpendicular a B_0. A aplicação deste campo provoca a alteração do estado de magnetização do sistema. A rotação induzida no vetor magnetização depende da intensidade do campo B_1 e do tempo de aplicação deste campo. Como na prática, este tempo é relativamente curto, o evento de aplicação de um campo B_1 em um tempo t é definido como um pulso magnético ou pulso de radio-freqüência.

3.1.1 Processos de Relaxação

O modelo clássico identifica a possibilidade de se tirar o vetor magnetização da sua posição de equilíbrio (M_0), paralela ao campo B_0, alterando sua inclinação por um ângulo θ, pela aplicação de um pulso de radio-frequência (RF) e levando-o para configurações de mais alta energia.

Na maioria dos sistemas físicos, a configuração de equilíbrio corresponde a mais estável, de menor energia. Sendo assim, o distúrbio provocado pelo pulso de RF tende, naturalmente, ao retorno para a configuração inicial. Esse processo envolve a troca de energia entre o sistema de prótons e seus vizinhos, tendo dois tempos característicos intimamente correlacionados: T1 e T2. O tempo T1, conhecido como tempo de relaxação *spin*-rede, envolve as interações entre o sistema de *spin* e a rede associada a ele. Já o tempo de relaxação T2 reflete a troca de energia entre os próprios *spins*, sendo denominado tempo de relaxação *spin-spin*. A seguir, serão abordados os principais detalhes desses processos de relaxação.

3.1.1.1 Tempo de Relaxação T1

O processo de relaxação *spin*-rede, ou relaxação longitudinal, ocorrem em progressão exponencial, com um tempo T1. No caso dos prótons, o tempo T1, assume valores que podem variar de meio segundo até vários segundos, dependendo da intensidade do campo magnético e de outras condições. A existência de valores específicos de T1 para cada tipo de tecido biológico é muito explorada na técnica de RMN para aumentar o contraste das imagens.

Todos os prótons em um material estão sujeitos à presença de campos magnéticos locais de baixa intensidade, produzidos por prótons que formam a sua

vizinhança. As interações com a sua vizinhança fazem aparecer perturbações de energia com uma faixa de freqüência bastante ampla. Devido ao movimento relativo dos dipolos, esses campos locais flutuam, gerando um ruído magnético local. Tais flutuações, podem induzir uma transição de estados, favorecendo, assim, os processos de relaxação, isto é, de retorno à configuração original. Quanto mais intensa for essa interação, maior será a influência sobre o processo de relaxação, levando, conseqüentemente, a um tempo de relaxação T1 mais curto. Vários são os mecanismos físicos que levam ao aparecimento dessas flutuações de campo. Aqui serão citados alguns mais importantes.

A principal contribuição é a agitação térmica das moléculas cujos dipolos encontram-se em estados instáveis de energia. No caso da água, a maior contribuição provém do outro próton de hidrogênio constituinte da molécula. Nesse caso, a interação entre esses dois dipolos magnéticos provoca a transição para o equilíbrio.

Um segundo processo, menos intenso, diz respeito ao movimento das moléculas. Por se tratar de um processo que depende fundamentalmente da faixa de freqüência da perturbação, as características físicas da vizinhança determinam a magnitude da interferência. Assim, por exemplo, dipolos magnéticos de pouca mobilidade, como aqueles presentes em ossos ou cartilagens, cuja faixa de freqüência está em uma porção de valores baixos do espectro, produzem pouca influência sobre os tempos de relaxação T1. Nesses casos, os tempos de relaxação T1 característicos são habitualmente longos.

Em resumo, o processo de decaimento T1 é uma conseqüência da translação e rotação de núcleos vizinhos que fazem surgir flutuações locais de campo magnético.

3.1.1.2 Tempo de Relaxação T2

A magnetização transversal retorna exponencialmente ao seu valor de equilíbrio com um tempo característico T2, o que faz com que a amplitude do sinal por ela induzido decaia a zero. O sinal resultante é bastante efêmero e recebe o nome de decaimento de indução livre, ou *fid,* do inglês "*free induction decay*". Tal decaimento reflete o efeito das interações entre os *spins* nucleares. O sinal de resposta *fid* é medido por uma bobina de indução colocada ao redor do objeto que está sendo mapeado. Esta medida é processada ou reconstruída para obter a IRM.

A alteração do campo local, B_0, induz uma alteração na freqüência de precessão. Se essa alteração for devida à flutuação local, tem-se como resultado uma pequena alteração da freqüência de Larmor.

Do ponto de vista físico, essas variações locais do campo estático B_0, são provocadas por processos semelhantes aos descritos anteriormente para T1. Contudo, somente flutuações de baixa freqüência têm efeito preponderante sobre T2. Além disso, fica claro que flutuações em torno da freqüência de Larmor, que induzem T1, também resultam numa perda da magnetização, provocando, indiretamente, um encurtamento de T2. Por esse motivo, T2 é sempre menor ou igual a T1.

Na realidade, um outro processo que também contribui para a diminuição de T2 é a inomogeneidade do campo estático, B_0,, devido às imperfeições instrumentais. Muito embora o efeito de ressonância dependa de um campo B_0, altamente homogêneo, não existe bobina perfeita. Essa imperfeição causa o aparecimento de pequenas perturbações locais, e depende da qualidade do equipamento de medida. Na prática, portanto, é mais freqüente que o decaimento ocorra num tempo menor, T2*, que reflete as variações devidas aos dois processos independentes: inomogeneidade de campo e flutuações locais devidas à vizinhança.

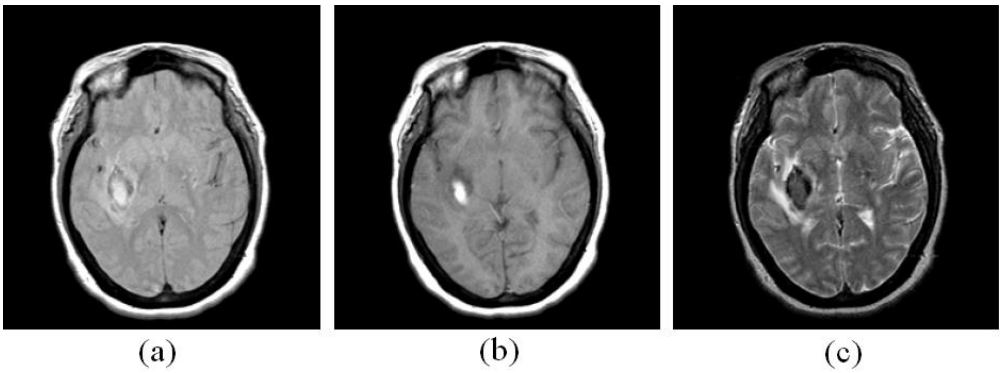
3.1.2 Produção de imagens

Para produzir imagens, o sinal de ressonância deve ser codificado para cada dimensão. A codificação na direção axial, direção de B_0, é feita adicionando-se um gradiente de campo magnético para B_0. Gradiente de campo magnético é definido como a medida da variação de sua intensidade ao longo de uma direção espacial. Este gradiente gera uma mudança linear na freqüência de Larmor na direção axial. Assim, uma fatia axial pode ser selecionada escolhendo a freqüência do pulso de RF correspondente a freqüência de Larmor desta fatia. A reconstrução espacial 2D em cada fatia é feita usando a codificação de fase e freqüência. Um gradiente, G_y, é aplicado causando freqüências ressonantes dos núcleos que variam de acordo com sua posição na direção y. Depois, G_y é removido e outro gradiente, G_x, é aplicado perpendicularmente à G_y. Como um resultado, as freqüências ressonantes dos núcleos variam na direção x devido à G_x e tem uma variação de fase na direção y devido ao gradiente anteriormente aplicado, G_y. Assim, a amostra na direção x é codificada pela freqüência e a amostra na direção y é codificada pela fase.

3.2 Aplicações Clínicas

As aplicações clínicas das IRM são bem diferenciadas. Nesta seção pretende-se dar apenas alguns exemplos de como estas são úteis no diagnóstico e caracterização de diversas doenças, sem a pretensão de enumerar exaustivamente as suas potencialidades.

A neurologia é uma das áreas em que as IRM são mais utilizadas. O interesse de sua utilização nesta área está relacionado com o diagnóstico de tumores, doenças degenerativas inflamatórias ou cardiovasculares. A Figura 3.4 ilustra um caso de aplicação de IRM em neurologia na identificação de uma lesão.



[Figura 3.4] IRM de uma hemorragia cerebral com contraste em: (a) densidade protônica, (b) T1 e (c) T2 (Imagem retirada de [Ualg 2007]).

No que diz respeito à estrutura óssea, as IRM em estudos tumorais são muito relevantes, uma vez que, geralmente, permitem um grande contraste entre o tecido normal e o tumoral, além de diferenciar bem os tecidos ósseos, musculares e articulações.

Em resumo, pode-se concluir que a IRM permite a obtenção de imagens muito nítidas de praticamente todos os tecidos humanos, sendo de extrema utilidade quando se exige detalhe e nitidez de qualquer órgão.

3.3 O Padrão DICOM

Existem vários fabricantes de *scanners* (Toshiba, Philips, GE etc.), cada qual com seu formato próprio para as imagens, fato que inviabiliza a troca de dados médicos entre hospitais ou centros de diagnóstico que possuam equipamentos de marcas distintas. Em 1983, visando padronizar a transferência de informações médicas o *American College Radiology* (ACR) e a *National Electrical Manufacturers Associations* (NEMA) formaram um comitê e desenvolveram o padrão *Digital Imaging and Communications in Medicine* (DICOM) [Nema 2007]. Este padrão digital é mais que um formato de arquivo, é uma estrutura de comunicação para intercâmbio de imagens

médicas. O padrão DICOM contém informações como: nome do paciente, tipo de exame, dimensões da imagem, além de dados da imagem. Em resumo, este padrão foi criado para cumprir os seguintes objetivos:

- promover a comunicação de informações de imagens médicas digitais, independente do tipo de equipamento;
- facilitar o desenvolvimento e a expansão de sistemas de arquivamento e comunicação de imagens, preparando-os também para interação com outros sistemas hospitalares de informação;
- permitir a criação e o compartilhamento de bancos de dados com informações de diagnósticos.

Excede o escopo desse trabalho, entender o padrão DICOM em toda sua extensão, porém é necessária uma visão geral de suas características, visto que as IRM utilizadas neste trabalho estavam neste formato, e, através das informações fornecidas por este padrão foi possível ter uma descrição detalhada das imagens utilizadas.

Capítulo 4

Transformações Geométricas

A programação de análise de imagens médicas envolve, na maioria dos casos, três tipos de objetos: (i) imagens, (ii) superfícies e (iii) transformações. Transformações são geralmente calculadas em algoritmos de registro e também podem ser usadas para transformar uma superfície ou reamostrar uma imagem.

As transformações geométricas têm um papel importante no processamento de imagens biomédicas. Translações de imagens, rotações e ou escalonamentos são necessárias para visualização de dados, reamostragem de um conjunto de dados volumétricos (IRM e PET) e registro de imagens.

As transformações geométricas são comuns em computação gráfica e na análise de imagens, pois elas permitem a eliminação de distorções geométricas que podem ocorrer quando a imagem é adquirida.

Estas podem ser divididas em duas classes: rígidas e não-rígidas. O texto a seguir sobre transformações geométricas foi escrito baseado na apostila de computação gráfica de [Traina e Oliveira 2006].

4.1 Transformações Rígidas

Em processamento de imagens, as transformações geométricas consistem em uma transformação espacial, que pode ser subdividida nas seguintes operações: translação, escala e rotação. Estas operações definirão o novo posicionamento dos *pixels* no plano da imagem.

Transformações rígidas são definidas como transformações geométricas que preservam todas as distâncias e ângulos. Estas transformações são as combinações de rotações e translações.

4.1.1 Transformações Espaciais em 2D

A translação de pontos (*x*,*y*) pode ser efetuada adicionando-se quantidades inteiras as suas coordenadas. Assim, cada ponto P(x, y) pode ser movido por d_x unidades em relação ao eixo *x*, e por d_y em relação ao eixo *y*. Logo, o ponto P(x', y'), pode ser escrito com

$$x' = x + d_x$$
$$y' = y + d_y . \qquad 4.1$$

E, definindo-se os vetores colunas,

$$P = \begin{pmatrix} x \\ y \end{pmatrix},\ P' = \begin{pmatrix} x' \\ y' \end{pmatrix} \text{ e } T = \begin{pmatrix} d_x \\ d_y \end{pmatrix}. \qquad 4.2$$

Então, a equação para representar a translação, pode ser expressa como

$$P' = P + T . \qquad 4.3$$

Podem-se efetuar mudanças de escala de um ponto pelo eixo *x* (*sx*), ou pelo eixo *y* (*sy*), através das multiplicações

$$x' = sx * x \text{ e } y' = sy * y . \qquad 4.4$$

A escala é feita em relação à origem dos pontos. Se os fatores de escala forem diferentes para *x* e para *y*, o objeto sofrerá uma deformação, mas caso *sx* e *sy* sejam iguais, a proporção do objeto será mantida.

A rotação de pontos através de um ângulo θ também é feita a partir da origem, sendo definida matematicamente por

$$x' = x \cdot \cos(\theta) - y \cdot \sin(\theta) \text{ e } y' = x \cdot \sin(\theta) + y \cdot \cos(\theta), \quad 4.5$$

e matricialmente como,

$$P' = R \cdot P \Rightarrow \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix}. \quad 4.6$$

Os ângulos positivos são definidos quando a rotação é feita no sentido anti-horário, e os ângulos negativos quando a rotação é feita no sentido horário.

4.1.2 Coordenadas Homogêneas

Para a translação, escala e rotação, as matrizes de transformação são respectivamente

$$P' = T + P, \quad 4.7$$

$$P' = S \cdot P, \quad 4.8$$

e

$$P' = R \cdot P. \quad 4.9$$

Infelizmente, a translação é tratada de forma diferente (como uma soma) em relação às outras transformações - rotação e escala - que são tratadas através de multiplicações. Para poder combinar facilmente estas transformações, deve-se tratar as três transformações de um mesmo modo e de maneira consistente.

Uma forma de se obter isso é tratar os pontos como coordenadas homogêneas, pois desta forma todas as três transformações podem ser tratadas como multiplicações.

Para a utilização de coordenadas homogêneas é necessário adicionar uma terceira coordenada ao ponto. Assim, cada ponto (x, y) será representado pela tripla (x, y, W). Desde que W seja uma coordenada diferente de zero, basta dividir a tripla por W e será obtido o mesmo ponto com as coordenadas $\left(\frac{x}{W}, \frac{y}{W}, 1\right)$. Onde os números $\frac{x}{W}$ e $\frac{y}{W}$ são chamados de coordenadas cartesianas do ponto homogêneo (x, y, W) [Lay 1997]. Geralmente as triplas são utilizadas para representar pontos em 3D, porém neste caso estão representando pontos em 2D. Os pontos são vetores de três elementos, por isso as matrizes de transformação que multiplicam um ponto por outro também devem ser 3x3. Sendo assim, a equação de translação 4.1 para coordenadas homogêneas fica:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & d_x \\ 0 & 1 & d_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}. \qquad 4.10$$

Assim, a equação 4.10 pode ser representada como

$$P' = T(d_x, d_y) \cdot P, \qquad 4.11$$

sendo,

$$T(d_x, d_y) = \begin{bmatrix} 1 & 0 & d_x \\ 0 & 1 & d_y \\ 0 & 0 & 1 \end{bmatrix}. \qquad 4.12$$

Fazendo uso das coordenadas homogêneas, a equação de escala fica sendo igual a

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \qquad 4.13$$

e a matriz 3x3 da escala pode ser representada por

$$S(s_x, s_y) = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}. \qquad 4.14$$

As equações de rotação são definidas como sendo iguais a

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \qquad 4.15$$

e a matriz rotação 3x3 pode ser representada por

$$R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}. \qquad 4.16$$

As coordenadas homogêneas de pontos não podem ser somadas ou multiplicadas por escalares, mas podem ser transformadas através da multiplicação por matrizes 3x3 [Lay 1997]. Com o uso de coordenadas homogêneas, as transformações de geometria métrica podem ser representadas por matrizes 3x3, o que permite usar o operador produto para encontrar a transformação que resulta da concatenação arbitrária de translações e rotações.

O objetivo de se utilizar coordenadas homogêneas é possibilitar a combinação das transformações geométricas, de maneira a reduzir a quantidade de cálculos e, consequentemente, o tempo computacional.

4.1.3 Transformações Espaciais 3D

A capacidade de representar e visualizar um objeto em três dimensões é fundamental para a percepção de sua forma. Porém, em muitas situações é necessário mais do que isto, ou seja, poder movimentar o objeto através de rotações e translações.

No caso de transformações espaciais 2D as matrizes de transformação são 3x3. Generalizando este conceito, as matrizes para 3D serão 4x4. De modo análogo ao caso em 2D, um ponto P de coordenadas (x, y, z) será representado por (x, y, z, W). Em coordenadas homogêneas, este ponto é $\left(\frac{x}{W}, \frac{y}{W}, \frac{z}{W}, 1\right)$, com W diferente de zero.

A translação em 3D pode ser vista como uma extensão do caso 2D, ou seja,

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & d_x \\ 0 & 1 & 0 & d_y \\ 0 & 0 & 0 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}. \qquad 4.17$$

Assim, a equação 4.17 pode ser representada como

$$P' = T(d_x, d_y, d_z) \cdot P. \qquad 4.18$$

Similarmente, a escala em 3D fica na forma

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_y & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}. \qquad 4.19$$

Deste modo, a equação 4.19 pode ser escrita como

$$P' = S(s_x, s_y, s_z) \cdot P. \qquad 4.20$$

As matrizes rotações ao redor do eixo *z*, *y* e *x* em 3D são, respectivamente,

$$R_z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad 4.21$$

$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad 4.22$$

e

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \qquad 4.23$$

E a forma geral para rotação torna-se

$$P' = R(\theta) \cdot P\,. \qquad 4.24$$

sendo R o eixo de rotação e θ o seu ângulo.

Os vetores compostos pelas linhas e colunas da submatriz 3x3 do canto superior esquerdo de $R_{x,y,z}(\theta)$ são mutuamente perpendiculares, e além disso, esta submatriz tem determinante igual a 1, o que significa que as três matrizes são chamadas de Ortogonais Especiais.

4.1.4 Transformações Compostas

O movimento de uma imagem na tela de um computador requer, muitas vezes, duas ou mais transformações básicas. A composição dessas transformações corresponde à multiplicação de matrizes quando as coordenadas homogêneas são usadas.

O propósito da composição das transformações é o ganho de eficiência que se obtém ao aplicar-se uma transformação composta a um ponto em vez de aplicar-lhe uma série de transformações, uma após a outra.

Uma composição genérica de transformações R, S e T, produz uma matriz da seguinte forma

$$\begin{bmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}. \qquad 4.25$$

A submatriz 2x2 superior esquerda da equação 4.25 é uma composição das matrizes de rotação e escala, enquanto t_x e t_y são obtidos por influência da translação. Quando $M \cdot P$ (um vetor de três elementos multiplicado por uma matriz 3x3) é calculado, pode-se verificar que são necessárias nove multiplicações e seis adições. Mas, como a última linha da matriz é fixa os cálculos efetivamente necessários são:

$$x' = x \cdot r_{11} + y \cdot r_{12} + t_x$$
$$y' = x \cdot r_{21} + y \cdot r_{22} + t_y, \qquad 4.26$$

o que reduz o processo a quatro multiplicações e duas adições, produzindo um ganho significante em termos de eficiência, especialmente em casos em que esta operação deve ser aplicada a centenas ou mesmo milhares de pontos por imagem.

4.2 Transformações Não- Rígidas

As transformações não-rígidas são importantes não somente para os casos de anatomia não rígida, mas também quando se deseja comparar imagens de pacientes diferentes ou comparar imagens do mesmo paciente, no caso de existirem distorções no procedimento de aquisição das imagens. Em todos os casos, é preferível escolher transformações que tem significado físico, mas em alguns casos a escolha é feita com base nas propriedades matemáticas convenientes.

Várias técnicas podem caracterizar as transformações não-rígidas, e estas podem incluir uso de combinações lineares de funções polinomiais [Woods *et al.* 1998], funções trigonométricas básicas [Ashburner e Friston 1999], “Free Form Deformation” (FFD) usando “*Splines*” [Rueckert *et al*. 1999 e Schnabel *et al*. 2001], e outros métodos. Cabe neste documento, discutir apenas do tipo mais comum de transformações não-rígidas, que são as que utilizam *splines*.

Normalmente, quando se utilizam transformações com *splines*, são necessários pontos de controle, ou seja, pontos correspondentes nas imagens que serão comparadas. Estes pontos correspondentes definem vetores de deslocamento em suas posições. Os vetores de deslocamento entre os pontos de controle são calculados pela interpolação ou aproximação destes pontos.

Por exemplo, "*Thin plate spline*" interpola os pontos de controle e, em geral, tem uma base grande, pois cada ponto de controle tem uma influência global nas transformações.

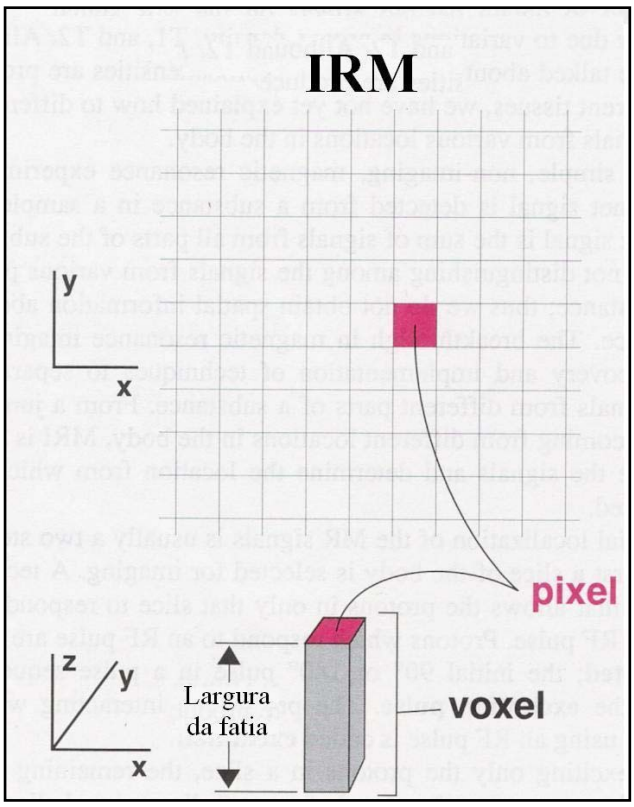
Em contraste ao "*thin plate spline*", "*B-splines*" aproximam-se nos pontos de controle e tem uma base limitada. Por exemplo, a mudança na posição de um ponto de controle influencia somente nas transformações dos vizinhos deste ponto. Posto que a base limitada reduz o tempo computacional, um grande número de graus de liberdade pode ser utilizado para este tipo de transformação.

Capítulo 5

Visualização Volumétrica em Medicina

Imagens tridimensionais estão se tornando uma modalidade de visualização muito acessível devido ao desenvolvimento contínuo da instrumentação [Russ 2002]. Gerar uma imagem de um tecido significa, essencialmente, mapear alguma característica física de diferentes porções dele. Por sua vez, qualquer imagem digital é formada por um número finito de elementos, denominados *pixels*. Assim, pode-se representá-las por redes bidimensionais (matrizes) em que o número de linhas e de colunas indica o tamanho da imagem. No caso de uma representação volumétrica, 3D, os elementos fundamentais constituintes da imagem são denominados *voxels*. Na IRM, por exemplo, cada *pixel* tem um *voxel* associado (Figura 5.1).

Visualização de volume é um conjunto de técnicas que permitem visualizar a estrutura tridimensional de um objeto a partir de um conjunto de dados de volume [Lichtenbelt *et al*. 1998]. Para a visualização volumétrica de imagens médicas, tipicamente, utiliza-se uma série de imagens 2D em seqüência, adquiridas por uma ou mais técnicas de aquisição de imagens, como ITC, IUS e IRM. Estas imagens são então "empilhadas" para recriar a estrutura 3D.



[Figura 5.1] A IRM é dividida em um número finito de elementos chamados *pixels*. Cada *pixel* é uma área retangular na imagem. O brilho de um *pixel* indica a intensidade do sinal de RM que vem de um volume de tecido associado. Este volume é chamado de *voxel* (Figura modificada de [Smith e Ranallo 1989]).

O processo de visualização de dados médicos possui várias etapas. O passo inicial consiste na aquisição de dados, que é seguida pelo seu armazenamento. Após, deve-se processar as imagens para obter uma boa distribuição de valores livres de ruído. Considerando que o usuário esteja trabalhando com fatias de dados, o próximo passo consiste na reconstrução volumétrica [Manssour 1998]. A Figura 5.2 ilustra o processo de reconstrução e visualização 3D.

[Figura 5.2] Etapas de reconstrução e visualização volumétrica.

Os algoritmos de visualização volumétrica podem ser classificados em: algoritmos de *renderização de superfície* e algoritmos de *renderização de volume*. Nos algoritmos de renderização de superfície, os dados volumétricos são convertidos para uma representação geométrica de superfície. Neste caso, métodos tradicionais de renderização de polígonos são usados para visualizar as superfícies de volume. Nos algoritmos de renderização de volume, a visualização é realizada a partir dos dados volumétricos, sem nenhuma representação intermediária.

5.1 Renderização de Superfície

Os algoritmos de extração de superfície tipicamente ajustam uma superfície, discretizada em polígonos, a pontos de mesmo valor dentro dos dados volumétricos. O processo se inicia com a escolha por parte do usuário de um valor de limiarização (*thresholding*). Então, primitivas geométricas são ajustadas automaticamente aos pontos por onde deveria passar a superfície de valor especificado. Após a definição da isosuperfície, são utilizados os métodos tradicionais de *rendering* de polígonos para realizar a visualização do volume; isto possibilita a utilização de métodos rápidos já conhecidos, assim como de *hardware* especificamente desenvolvido para este fim.

Uma desvantagem do uso de isosuperfícies está no fato de ser visualizado somente um subconjunto dos dados, determinado pelo valor de limiarização. Isto pode ser contornado com o uso de isosuperfícies transparentes, uma para cada valor de *limiarização*. No entanto, a compreensão da cena diminui à medida que são incluídas mais isosuperfícies. Em geral estes algoritmos geram um grande número de primitivas geométricas, o que pode se tornar um problema para a visualização.

Outro problema sério desses algoritmos é a inclusão errônea de artefatos na imagem, resultantes de classificações erradas da superfície (i.e. devido a problemas de aproximação) e a dificuldade de visualizar detalhes do volume. A geração de isosuperfícies é a forma mais antiga e mais bem conhecida de se estudar o conteúdo dos conjuntos de dados volumétricos.

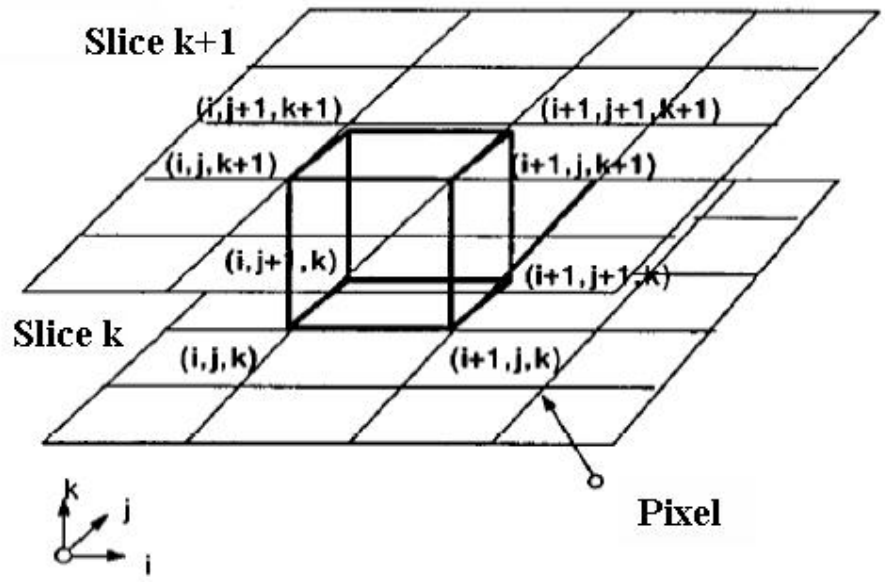
Um algoritmo de renderização de superfície bastante utilizado é o *Marching Cubes* [Levoy 1988], ele demonstra o funcionamento de técnicas de extração de superfícies contidas em volumes de dados. Seu intuito é gerar malhas triangulares de superfícies com densidade constante (isosuperfícies) a partir de dados volumétricos.

O algoritmo *Marching Cubes* proposto por Lorensen e Cline [Lorensen e Cline 1987], é uma das técnicas mais utilizadas para a visualização de dados amostrados. Essa classe de dados ocorre com freqüência em exames de TC e RM, o que a torna propícia para a aplicação em imagens médicas. Este algoritmo baseia-se em dois passos:

1. Localização da superfície correspondente ao valor especificado como parâmetro;

2. Cálculo das normais nos vértices dos triângulos, a fim de criar uma superfície de alta qualidade visual.

O *Marching Cubes* utiliza a técnica de "divisão e conquista" para localizar a superfície através de um cubo lógico, formado por oito *pixels* (quatro para cada fatia adjacente), como mostra a Figura 5.3. Cada *pixel* é representado por um vértice do cubo.

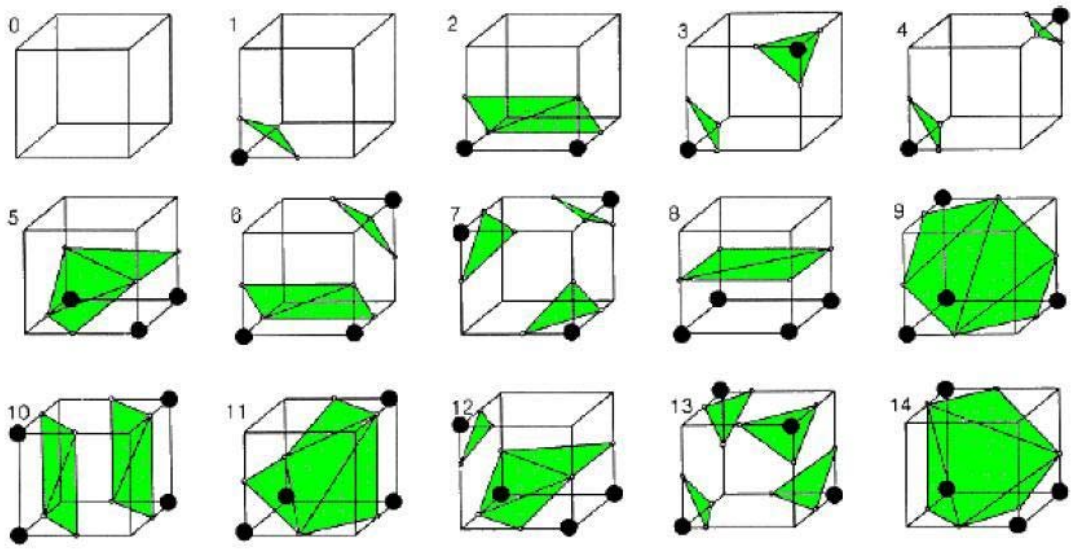


[Figura 5.3] Cubo lógico utilizado pelo algoritmo de *Marching Cube* (Figura modificada de [Lorensen e Cline 1987]).

O algoritmo determina o modo como a superfície intersecciona o cubo, movendo-se (ou "marchando") então para o próximo cubo. Para verificar se a superfície intersecciona o cubo, o algoritmo determina se o valor do vértice escolhido excede ou equivale ao valor da superfície a ser construída. Caso seja verdadeiro, o vértice recebe valor um, indicando que está dentro da superfície. Caso contrário, o vértice recebe valor zero, indicando que está fora. Assim, através da localização dessas intersecções, é possível determinar a topologia de uma superfície dentro de um cubo por meio de triangulações.

Como o cubo apresenta oito vértices, onde cada vértice possui dois estados, uma superfície pode ser interseccionada de 2^8 formas, ou seja, 256 combinações diferentes.

Assim, é possível criar uma tabela com as intersecções através dos vértices do cubo. Apesar da triangulação dos 256 casos ser possível, ela possui alto custo computacional e é suscetível a erros. Analisando possíveis configurações de intersecção, através de operações geométricas de rotação simétrica e reflexão, os 256 casos foram reduzidos para 15, como mostra a Figura 5.4 [Lorensen e Cline 1987].



[Figura 5.4] Os 15 casos de intersecção do algoritmo *Marching Cubes* (Figura modificada de [Lorensen e Cline 1987]).

O caso 0 (zero) ocorre quando todos os vértices possuem valores acima (ou abaixo) do valor da superfície, não gerando nenhum triângulo. O caso 1 (um) ocorre quando a superfície separa um vértice dos outros oito vértices, resultando em um triângulo definido por três intersecções. Os outros 13 casos produzem diferentes formas de intersecção, gerando diversas combinações de triangulação.

Em suma, a técnica de visualização de volume através de superfície utiliza primitivas geométricas (malhas de polígonos ou linhas de contorno) para apresentar o volume, ou seja, sua representação básica é o contorno. Estas primitivas normalmente são isosuperfícies extraídas automaticamente do volume. O algoritmo mais comum que

utiliza esta abordagem é o *Marching Cubes* [Lorensen e Cline 1987], como dito anteriormente . Alguns dos problemas encontrados neste tipo de algoritmo são a geração ocasional de pedaços de superfícies falsos, a manipulação incorreta de pequenas características dos dados e a dificuldade da representação da superfície para algumas estruturas do corpo humano. As grandes vantagens desta técnica, entretanto são a velocidade e pouco espaço de armazenamento requerido.

5.2 Renderização de Volume

Com a renderização de volume, imagens são criadas diretamente dos dados de volume sem a necessidade de extração de geometria intermediária. Todas as informações de níveis de cinza originalmente adquiridas no processo de aquisição das imagens são conservadas. A renderização de volume direta é uma técnica ideal para explorar dados interativamente [John e Mccloy 2004].

Esta seção tem como objetivo apresentar os algoritmos básicos de renderização de volume. Esses algoritmos têm em comum a não utilização de nenhuma representação intermediária para gerar a imagem do volume. Porém, em geral, têm um custo computacional maior que os algoritmos de renderização de superfície.

As principais fases do algoritmo de renderização de volume são: classificação de dados e projeção do volume. A fase de classificação é um processo que permite selecionar a estrutura do volume a ser visualizada. A fase de projeção do volume envolve o lançamento de raios da posição do observador em direção ao volume e cálculo de composição dos *voxels* interceptados pelo raio, para determinar a cor dos *pixels* no plano de visualização.

Como exemplo de algoritmos de renderização, pode-se citar: *Ray Casting* [Levoy 1988] e *Splatting* [Westover 1990]. Esses algoritmos realizam a projeção de

volume diretamente no plano da imagem e são os mais indicados para visualizar volumes que representam objetos amorfos.

Pode-se ainda subdividir os algoritmos de renderização direto em duas categorias principais: espaço dos objetos e espaço da imagem. Na primeira situação (espaço dos objetos), é utilizado o mapeamento direto (*forward mapping*) do volume sobre o plano da imagem, ou seja, os *voxels* são projetados na tela em uma determinada ordem. Como exemplo pode-se citar o algoritmo de *Splatting*. No outro caso (espaço da imagem), é utilizado o mapeamento reverso (*backward mapping*), onde são lançados raios para cada *pixel* da imagem em direção ao volume determinado. Como exemplo pode-se citar o algoritmo de *Ray Casting*.

A seguir serão apresentadas as principais características dos algoritmos mais importantes de renderização direta: o traçado de raios (*Ray Casting*), que opera no espaço da imagem e o *Splatting*, que opera no espaço do objeto.

5.2.1 Ray Casting

O algoritmo de *Ray Casting* permite a geração de imagens de alta qualidade através da renderização direta, operando no espaço da imagem [Levoy 1988]. Este algoritmo resolve algumas limitações da renderização de superfície, como perda de algumas propriedades do volume. Muitas vezes, os dados presentes no interior de um *voxel* contêm misturas de diversos materiais. Superfícies entre os materiais, variações locais nas propriedades volumétricas, tais como absorção ou emissão de luz, são perdidas, se o volume é reduzido a uma superfície [Drebin *et al*. 1988].

Os algoritmos tradicionais de *Ray Casting* varrem o plano de projeção e lançam raios na direção da cena. O plano de projeção corresponde à janela de seleção da imagem, e os raios por sua vez, são associados aos *pixels* da janela de exibição. Para a

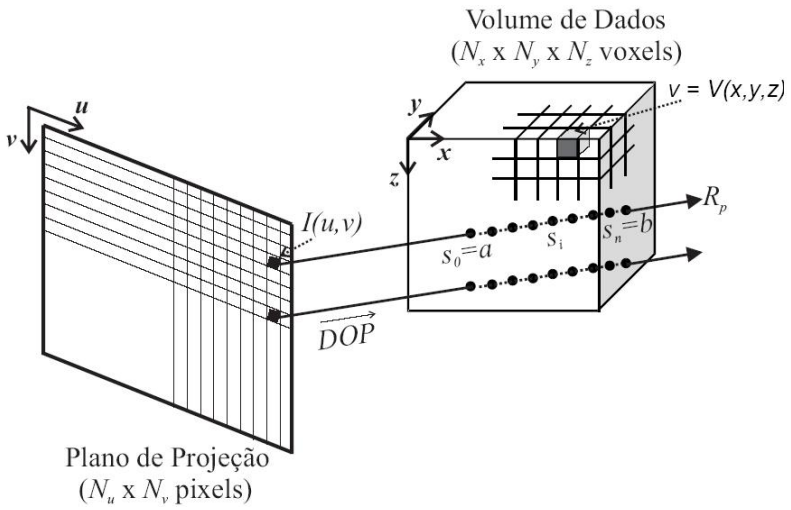
visualização de volume, normalmente são usados raios paralelos lançados ortogonalmente ao plano de projeção.

O mecanismo básico do algoritmo de *Ray Casting* consiste em traçar raios paralelos a partir da posição do observador através dos dados. Assume-se que a imagem é um retângulo medindo N*x*M *pixels* (plano de projeção) e que o raio é lançado para cada um deles. Para cada raio, então, são obtidas amostras, com um espaçamento constante, onde a cor e opacidade para cada *voxel* são calculadas e acumuladas para no final ser determinada a cor do *pixel*.

Resumidamente, o funcionamento do *Ray Casting* consiste na varredura dos *pixels* da janela de visualização e no lançamento de raios, a partir desses *pixels*, na direção do volume de dados [Levoy 1988]. Sendo assim, os principais elementos deste algoritmo são: o volume de dados e suas propriedades, os parâmetros de visualização, a janela de visualização e os raios ao longo dos quais são processados os pontos de amostragem.

O volume de dados *V(x)*, mostrado na Figura 5.5, refere-se a valores v associados a posições de amostragem discretas $X = [x, y, z]$, em geral regularmente espaçadas no domínio 3D, sendo $x \in [1, N_x]$, $y \in [1, N_y]$ e $z \in [1, N_z]$. A existência de um ou mais valores de v associados a uma posição de amostragem depende das propriedades dos dados que o volume *V(x)* representa.

Os parâmetros de visualização tradicionalmente especificam a posição do observador, o tipo, a direção, o plano de projeção e os planos de corte 3D. No contexto da visualização direta do volume, a direção de projeção $\overrightarrow{DOP}$ determina a orientação dos raios lançados no volume. O plano de projeção é mapeado por uma janela de visualização I no plano da imagem, que se refere aos valores de intensidade associados às posições dos *pixels* (Figura 5.5).

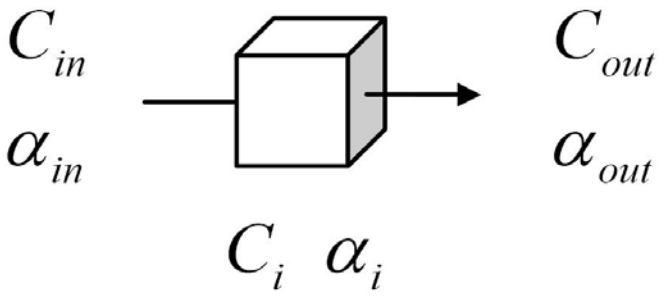


[Figura 5.5] Esquema genérico do algoritmo de *Ray Casting* (Figura retirada de [Manssour e Freitas 2002]).

A intensidade para cada *pixel* da imagem $I(u,v)$ é determinada pelo lançamento de um raio R_P deste *pixel* ao longo da direção de visualização $\overrightarrow{DOP}$. Ao longo de cada raio são processadas amostras s, em geral, com espaçamento constante. O processamento de uma amostra corresponde ao cálculo de cor e opacidade dependentes de v.

Na Figura 5.5 é utilizada a noção paramétrica para representar o raio. Sendo assim, s_i denota o i-ésimo ponto de amostragem usado para navegação ao longo de R_P, que vai do *pixel* $I(u)$ em direção ao volume de dados, com $i \in [0,n]$ e sendo $n+1$ o número de amostras [Manssour e Freitas 2002].

No algoritmo de *Ray Casting*, todas as contribuições de cor C_i e opacidade α_i, de cada *voxel* i interceptado por um raio, são acumuladas para determinar a cor, C, do *pixel*, como dito anteriormente. A cor C_{out} e a opacidade α_{out} de um raio após atravessar um *voxel* i são calculadas em função da cor C_{in} e da opacidade α_{in}, antes de atravessar o *voxel*, e da cor C_i e opacidade α_i, do próprio *voxel* (Figura 5.6)



[Figura 5.6] Cor e opacidade após atravessar um *voxel.*

O cálculo de cor e opacidade de um raio após atravessar um *voxel* i pode ser expresso por [Levoy 1990]

$$\begin{aligned} \hat{C}_{out} &= \hat{C}_{in} + \hat{C}_i \cdot [1 - \alpha_{in}] \\ \alpha_{out} &= \alpha_{in} + \alpha_i \cdot [1 - \alpha_{in}], \end{aligned} \qquad 5.1$$

sendo

$$\begin{aligned} \hat{C}_{in} &= C_{in} \cdot \alpha_{in} \\ \hat{C}_{out} &= C_{out} \cdot \alpha_{out} \\ \hat{C}_i &= C_i \cdot \alpha_i . \end{aligned} \qquad 5.2$$

Depois de calcular todas as N contribuições dos *voxels*, com k =1,...,N, a cor final C do *pixel* é dada por

$$C = \frac{\hat{C}_{out}}{\alpha_{out}} . \qquad 5.3$$

A equação de composição pode ser expressa por

$$\begin{aligned} &C = C(1) + C(2) \cdot [1 - \alpha(1)] + ... + \\ &C(N).[1 - \alpha(1)] \cdot ... \cdot [1 - \alpha(N-1)] = \\ &\sum_{k=1}^{N} \left(C(k) \cdot \prod_{i=1}^{k-1} [1 - \alpha(i)] \right). \end{aligned} \qquad 5.4$$

Em resumo, a técnica *Ray Casting* é a mais utilizada para a visualização volumétrica. Esta técnica permite a visualização de pequenos detalhes internos do volume, tem um bom controle de transparência removendo trivialmente as partes escondidas atrás de partes definidas como opacas, e visualiza o volume a partir de qualquer direção.

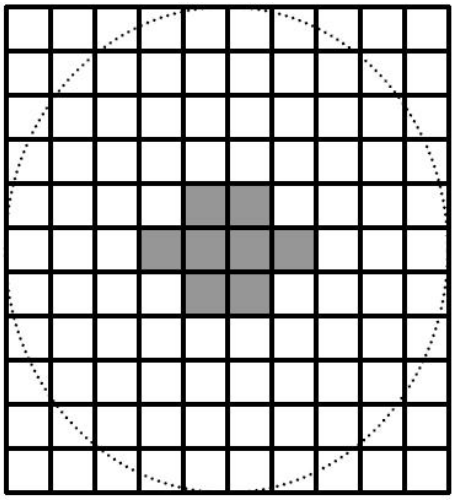
5.2.2 Splatting

O algoritmo *Splatting* [Westover 1990] trabalha no espaço dos objetos e procura projetar cada *voxel* do volume no plano da imagem. Essa projeção normalmente é realizada a partir dos *voxels* mais próximos do observador até o mais distante. A idéia central é "arremessar" cada *voxel* sobre o plano da imagem, o que deixa uma "marca" característica do *voxel*, daí o nome do algoritmo.

O algoritmo é definido em quatro etapas principais: transformação, tonalidade, reconstrução e visibilidade. Inicialmente, a amostra é processada através de sua transformação de coordenadas do volume para o espaço da tela. Em seguida é feita a tonalização da amostra, que engloba a classificação e iluminação, usando apenas informações locais.

A reconstrução é realizada para todas as amostras em um plano do volume de dados, definido como um plano paralelo ao plano da imagem. Após, a determinação da porção da imagem que a amostra influencia, através da função *footprint*, sua contribuição é adicionada em um acumulador do plano. Quando todas as amostras de um plano de volume de dados são processadas, o conteúdo do acumulador do plano é combinado na imagem através de um operador de composição. Depois que todas as amostras forem processadas, é então obtida a imagem final.

A função *footprint* é calculada para cada posição de observação do volume de dados. Como cada *voxel* projetado (*splat*) é representado por um núcleo (*kernel*) de interpolação simétrica, primeiro é calculada a extensão do espaço da tela onde o *kernel* será projetado. Esta projeção é então chamada de um *footprint*. Para uma projeção ortográfica, o núcleo mais comum é um Gaussiano esférico [Binotto 2003], ilustrado na Figura 5.7.



[Figura 5.7] *Footprint,* onde as áreas escuras indicam maior intensidade (Figura retirada de [Binotto 2003]).

Todos os *pixels* cujos centros se encontrão na região onde o *kernel* é projetado podem ser afetados pela amostra. A tabela que representa a função *footprint* é determinada em uma etapa de pré-processamento, acelerando o método [Westover 1990].

Capítulo 6

Visualização Científica

Neste capítulo serão descritas algumas ferramentas de visualização científica juntamente com suas funções e aplicações.

6.1 Conceito

Visualização científica (VC) é a área da computação dedicada à visualização de dados físicos, ou científicos, geralmente provenientes de medições ou simulações numéricas, fazendo uso de processamento de imagens e computação gráfica. Algumas técnicas de VC, cujos registros datam do século XII, são utilizadas até os dias de hoje, para visualização de um grande conjunto de dados complexos, e são implementadas em muitas ferramentas computacionais para visualização científica disponíveis atualmente.

Segundo [Schoeder *et al*. 2004], visualização é a transformação de dados ou informações em imagens, estimulando assim, o principal sentido humano, *a visão*. VC é a aplicação deste processo na visualização de dados científicos. O objetivo da VC é promover um nível mais profundo de entendimento dos dados sob investigação, confiando na habilidade poderosa dos humanos de visualizar. Assim, técnicas e ferramentas de visualização têm sido usadas para analisar e mostrar grandes volumes de dados multidimensionais, freqüentemente variantes no tempo, de modo a permitir ao usuário extrair características e resultados de maneira rápida e fácil.

Encontram-se aplicações da VC em muitas áreas do conhecimento: medicina, astronomia, meteorologia, engenharias e ciências em geral. Como exemplo, tem-se a visualização de dados provenientes de ITC e IRM, animações de tornados, e pós-

processamento de análises numéricas. Um caso de aplicação de VC em engenharia é a visualização de resultados obtidos por métodos numéricos (Método dos Elementos Finitos, Métodos dos Volumes Finitos, Métodos dos Elementos de Contorno, etc).

O processo de visualização de dados passa necessariamente por três passos fundamentais, que são: a aquisição dos dados, a transformação em uma forma apropriada para representação e a "renderização" (*rendering*) [Schroeder *et al*.2004]. As técnicas de visualização envolvem, portanto, algoritmos de processamento de dados que extraem os dados de interesse da amostra e os convertem em uma forma adequada para representação.

6.2 Ferramentas de Visualização Científica

O processo de visualização por computador envolve etapas distintas e algumas delas podem ser agilizadas com o uso de bibliotecas gráficas e/ou ferramentas (*toolkits*) de visualização. Bibliotecas gráficas fornecem funções que, fazendo interface com o *hardware* gráfico, facilitam a tarefa de geração das imagens no monitor ou em outro dispositivo. As ferramentas de VC fornecem um conjunto de algoritmos de visualização que fazem interface com a biblioteca gráfica utilizada, auxiliando na tarefa de transformar os dados e também apresentá-los no dispositivo de saída.

O desenvolvimento de ferramentas de VC pode abordar as etapas envolvidas no processo de visualização por meio de diferentes métodos. Um método que tem sido comumente utilizado, tornando-se um paradigma, é o método chamado de modelo de fluxo de dados [Earnshaw e Watson 1993].

6.2.1 Modelo de Fluxo de Dados

O modelo de processo de visualização chamado “fluxo de dados” (*dataflow*) influenciou o desenvolvimento de diversos sistemas de visualização. Nesse modelo o sistema oferece uma variedade de módulos que recebem os dados (*input*), os transformam e reescrevem (*output*). Módulos podem ser selecionados e combinados para formar a visualização desejada dos dados.

O modelo de fluxo de dados teve origem no trabalho de Upson e Keeler [Upson e Keeler 1988], que definiu para o processo de visualização um modelo orientado aos dados, em que os dados são transformados por meio de passos lógicos até a representação final, conforme ilustra a Figura 6.1.

[Figura 6.1] Modelo de fluxo de dados (Figura modificada de [Schroeder *et al.* 2004]).

Baseado na Figura 6.1, pode-se definir o filtro como sendo o passo onde os dados de interesse são extraídos (planos de corte, isocontornos), o mapeador (*mapper*) como o passo onde são construídas as geometrias para representar os dados extraídos, e o último passo, o renderizador (*render*) como sendo o passo onde a geometria é convertida em uma imagem e mostrada em um monitor ou em outro *display*.

6.2.2 Modelo Orientado a objetos

No modelo orientado a objetos, a metodologia de programação é utilizada de forma a explorar propriedades comuns e hierarquias. Um exemplo de ferramenta que usa o modelo orientado a objetos é o VTK [Avila *et al*. 2004].

Sistemas orientados a objetos oferecem a possibilidade de criar sistemas por meio da reutilização de componentes de *software*. Assim, várias bibliotecas de classes são disponibilizadas, desde estruturas padrões de dados até *solvers* matemáticos para resolução de equações e sistemas. No "desenho" usual de sistemas orientados a objetos, as estruturas de dados e os métodos são encapsulados em objetos. Algoritmos (ou métodos) e conjuntos de dados (ou estruturas de dados) são encapsulados separadamente [Schroeder *et al*.2004].

As ferramentas orientadas a objetos, como VTK, oferecem modelos de dados que são definidos de forma extensiva como bibliotecas de classes, onde cada classe implementa uma estrutura de dados.

6.2.3 Exemplos de *Softwares* e Ferramentas de Visualização Científica

Atualmente, muitos *softwares* e ferramentas para VC podem ser encontrados. Alguns destes *softwares* e ferramentas são: *HoloDraw* [Holodraw 2006], *The Visualuzation Toolkit* [Kitware 2006], *Amira* [Amira 2006], *ParaView* [Paraview, 2006], *VisAD* [Visad 2006], *VMD* [VMD 2006*], Matlab* [Matlab 2006], *Scilab* [Scilab 2006], *Spotfire* [Spotfire 2006], *VolView* [Volview 2006] e *Osiris* [Ligier *et al* 1994], entre outros.

Cada um destes *softwares* tem suas características próprias. A seguir, são descritas resumidamente características de algumas destas ferramentas, mas sem a pretensão de trazer um entendimento completo, pois isto foge do objetivo deste trabalho.

O *software Matlab* é um sistema interativo cujo elemento de dados básico é uma matriz que não requer dimensão. Isto permite a formulação de soluções para muitos problemas de computação, especialmente aqueles envolvendo representações de matrizes [Gonzales *et al*. 2004]. A ferramenta *Matlab*, por exemplo, pode ser considerada uma linguagem de programação interpretada, uma vez que a concatenação dos comandos dados no espaço de trabalho gera um "programa", que pode ser armazenado em um arquivo e "carregado" quando desejado [Roqueiro 2007].

O *software VisAD* consiste em uma biblioteca de componentes Java para visualização interativa e colaborativa de dados provenientes de análises numéricas [Visad 2006].

O *Amira* é um *software* profissional para visualização de dados 3D [Amira 2006]. Com ele, malhas volumétricas tetraédricas podem ser geradas, sendo adequado para visualização de resultados de análises por elementos finitos.

O *VolView* [Volview 2006] é um sistema comercial desenvolvido pelos autores do VTK. Apesar de ser um sistema interativo de visualização e exploração de imagens médicas, ele possui um custo muito elevado.

O *software Osiris* [Ligier *et al*. 1994] foi projetado para ser uma ferramenta genérica de manipulação de imagens médicas. Suas principais vantagens são a presença de filtros e recursos de análises quantitativas.

Depois da análise das diversas ferramentas de VC e *softwares*, optou-se pelo uso do VTK. Para esta escolha foram levados em conta os seguintes requisitos: a

necessidade de um sistema interativo de visualização de imagens, possibilidade de interação de um *software* com um transdutor de posição e o fato de VTK estar sendo atualmente muito usado em aplicações médicas [Schroeder *et al*. 2004].

Na próxima seção serão apresentadas algumas características, funcionalidades e usos da ferramenta de visualização científica VTK.

6.3 Visualization Toolkit

Entre as ferramentas de Visualização Científica que existem, uma se destaca por ser amplamente utilizada por cientistas do mundo todo, inclusive no desenvolvimento de aplicativos comerciais em várias áreas, como a área de processamento de imagens médicas. Essa ferramenta é o *Visualization ToolKit*, ou VTK. Esta seção dedica-se a esta ferramenta.

6.3.1 Arquitetura

O VTK é uma ferramenta de visualização científica disponibilizada gratuitamente pela *Kitware Inc*. [Avila *et al*. 2004]. O VTK possui código fonte aberto e é totalmente portável. Ele consiste em uma biblioteca de classes implementadas em C++ que possui diversos níveis de interface para linguagens interpretadas, incluindo Tcl/Tk, Java e Python (Figura 6.2). Estas classes implementadas em C++ são largamente usadas para o processamento de imagens e visualização científica.

A arquitetura do VTK permite a criação de algoritmos eficientes (tanto com relação à memória quanto ao processamento) na linguagem C++, e em outras linguagens. O núcleo corresponde a estruturas de dados, algoritmos e funções do sistema, e proporciona eficiência e velocidade adequadas. A camada interpretada

oferece flexibilidade e extensibilidade. Por exemplo, a utilização de ferramentas de GUI (*Graphic User Interface*, ou interface gráfica do usuário), tais como Tcl/Tk, Python/Tk e Java AWT, permitem rápido desenvolvimento de aplicações profissionais [Avila *et al*. 2004].

[Figura 6.2] Esquema da arquitetura do VTK. O VTK consiste de um núcleo (C++) compilado envolto com várias linguagens interpretadas (Java, Tcl, Python) (Figura modificada de [Avila *et al*. 2004]).

O VTK é desenvolvido segundo a abordagem orientada a objetos. Utilizando-se desta abordagem, um sistema modela *classes* de entidades, para todas as entidades presentes no sistema. Tais entidades possuem significado, atributos e métodos, de acordo com o seu comportamento no mundo real. Seu significado representa o papel que desempenha no funcionamento do sistema. Seus *atributos* são propriedades ou parâmetros das entidades, e os *métodos* alteram seus atributos e determinam suas ações.

A programação orientada a objetos difere da programação convencional, em que a execução de uma tarefa implica na execução de um método de uma entidade relacionada com aquela tarefa. A criação de objetos é feita por instanciação das classes definidas, de maneira que cada novo objeto tem seus próprios métodos, que podem ser

encarados como cópias dos métodos da classe original. O VTK possui dois subsistemas principais, o modelo gráfico e o de visualização.

6.3.1.1 Modelo Gráfico

O Modelo Gráfico do VTK cria uma camada abstrata sobre a linguagem gráfica, como por exemplo, *OpenGL*, para assegurar a portabilidade entre plataformas [Schroeder *et al*. 2004].

Existem sete objetos básicos utilizados no VTK para a renderização de uma cena, embora existam muitos outros por trás de todo o processo. Estes objetos são [Avila *et al*. 2004]:

a) ***vtkRenderWindow***: É a janela na qual ocorre a renderização;

b) ***vtkRenderer***: Coordena o processo de renderização, envolvendo luz, câmeras e atores;

c) ***vtkLight*:** São as fontes de luz para iluminação da cena;

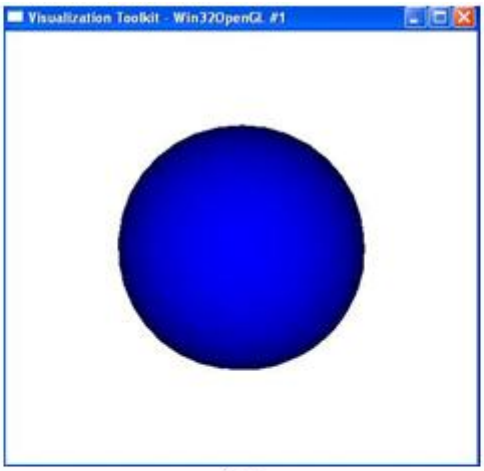
d) ***vtkCamera***: A câmera da cena;

e) ***vtkActor*:** Representa um objeto renderizado na cena, suas propriedades e posição;

f) ***vtkProperty***: Define as propriedades de aparência de um ator, tais como cor e transparência;

g) ***vtkMapper***: Define a representação geométrica para um ator.

A Figura 6.3 exemplifica uma imagem renderizada utilizando VTK, juntamente com o código-fonte que produziu esta imagem.



(a)

```
import vtk
# Cria a geometria da esfera
sphere=vtk.vtkSphereSource()
sphere.SetRadius (1.0)
sphere.SetThetaResolution (18)
sphere.SetPhiResolution (18)
# mapper da esfera
map=vtk.vtkPolyDataMapper()
map.SetInput(sphere.GetOutput())
# ator da esfera
aSphere=vtk.vtkActor()
aSphere.SetMapper (map)
aSphere.GetProperty().SetColor (0 ,0 ,1)
```

```
# cria a janela para renderização
renWin=vtk.vtkRenderWindow()
ren1=vtk.vtkRenderer()
renWin.AddRenderer (ren1)
# cria a interação
iren=vtk.vtkRenderWindowInteractor()
iren.SetRenderWindow (renWin)
#adição dos atores
ren1.AddActor (aSphere)
ren1.SetBackground (1, 1, 1)
#renderização da cena
renWin.Render()
#inicia a interação com o mouse
iren.Start()
```

(b)

[Figura 6.3] Exemplo de renderização com VTK. (a) Figura renderizada. (b) Código-fonte do algoritmo que produziu a figura.

A classe *vtkRenderWindow* junta todo o processo de renderização. Ela é responsável pelo controle de uma janela no dispositivo de vídeo. No VTK, as instâncias de *vtkRenderWindow* são independentes do *hardware* ou sistema operacional utilizados. Sendo assim, elas se adaptarão automaticamente à configuração do computador quando criadas.

A classe *vtkRenderer* é responsável pela junção de informações sobre luzes, câmera, e atores, utilizados na produção da imagem. Cada instância mantém uma lista ativa dessas entidades em uma cena particular. Pelo menos um ator precisa ser definido. Se as luzes e câmeras não forem definidas, eles serão criados automaticamente pelo *vtkRenderer*. É necessário que o *vtkRenderer* esteja associado a um objeto da classe *vtkRenderWindow* no qual haverá o desenho da cena.

Instâncias da classe *vtkLight* são responsáveis pela iluminação da cena. As luzes no VTK podem ser posicionadas, com um cone de iluminação e fatores de atenuação associados, ou posicionadas no infinito, com raios paralelos entre si.

As câmeras, criadas por *vtkCamera*, servem para fornecer métodos de posicionamento e orientação do plano de visualização. Elas possuem parâmetros como: posição, ponto focal, localização dos planos que determinam o volume e campo de visão.

A classe vtkActor representa os objetos na cena. Em particular, ela combina propriedades do objeto, como cor, tipo de sombra, transparência, definição geométrica, e orientação no sistema de coordenadas global. Tudo isso é implementado pela criação automática de diversos outros objetos como, por exemplo, *vtkProperty* e *vtkTransform*. Existem, ainda, diversas outras classes de atores com comportamento especializado, implementados como subclasses da classe *vtkActor*.

Instâncias de *vtkProperty* afetam a aparência de renderização de um ator. Quando os atores são criados, uma instância de suas propriedades é automaticamente criada com ele. Entretanto, também é possível criar objetos de propriedade diretamente e, então, associá-lo a um ou mais atores. Desta forma, dois atores podem compartilhar das mesmas propriedades.

Existem ainda outras classes muito úteis no processo de renderização, como por exemplo, *vtkRenderWindowInteractor*, que permite a interação do usuário com a cena de maneira simples e com muitos recursos.

6.3.1.2 Modelo de Visualização

O processo de visualização (*pipeline* de visualização) é responsável por construir a representação geométrica que é então renderizada pelo *pipeline* gráfico.

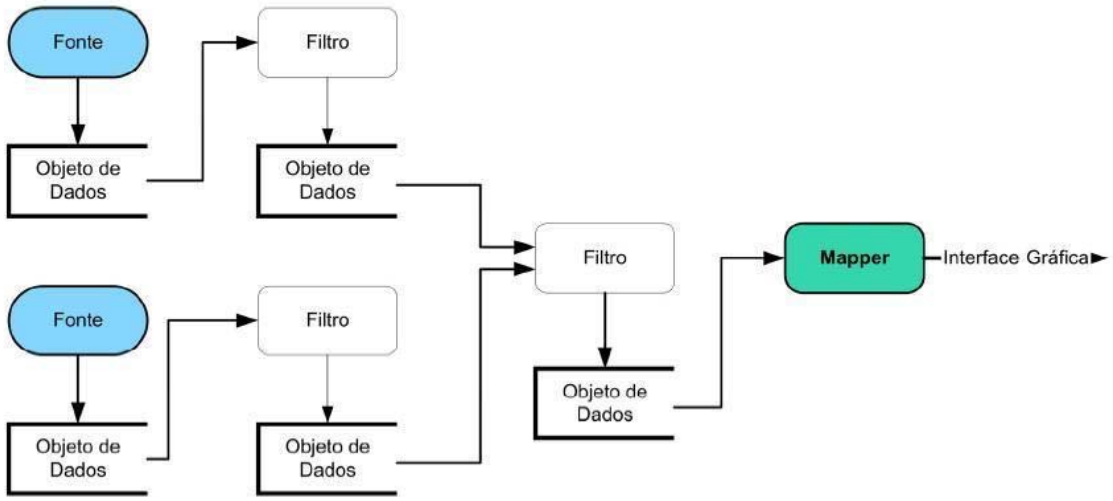
O VTK usa um fluxo de dados para transformar informação em dados gráficos. Existem dois tipos básicos de objetos envolvidos nesse fluxo: *vtkDataObject* e *vtkProcessObject* [Avila *et al*. 2004].

O primeiro corresponde aos objetos de dados, que representam diversos tipos de dados. Vários conjuntos de dados podem ser representados no VTK com uma estrutura formal. Esses objetos consistem de estrutura geométrica e topológica, e de atributos de dados associados, como por exemplo, valores escalares e vetoriais.

Já o segundo corresponde aos objetos de processo, ou, mais genericamente denominados, filtros. Eles operam nos objetos de dados para produzir novos objetos de dados, e representam ainda os algoritmos do sistema.

Os objetos de processo e de dados são conectados para formar o *pipeline* de visualização, ou seja, a rede de fluxo de dados, como mostrado na Figura 6.4. Os filtros recebem um ou mais objetos de dados e geram, também, um ou mais objetos de dados,

que são então transformados em dados gráficos pelos *mapeadores (mappers)* e renderizadores.

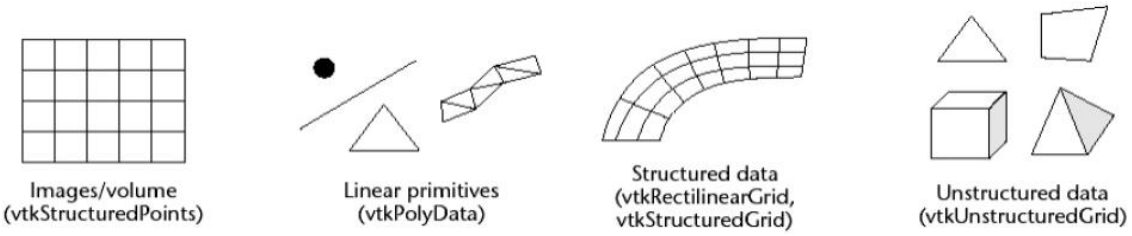


[Figura 6.4] *Pipeline* de visualização do VTK (Figura modificada de [Schroeder *et al*. 2004]).

6.3.1.3 Pipeline de Execução

O *pipeline* de visualização do VTK compreende os objetos que podem ser conectados de maneira a gerar a visualização desejada.

O *pipeline* é construído pela conexão de objetos de dados e filtros (*SetInput/ GetInput*). Os objetos de dados fornecem acesso aos dados e os filtros realizam operação sobre esses dados. Os objetos de dados suportados pelo VTK são mostrados na Figura 6.5.



[Figura 6.5] Objetos de dados do VTK (Figura retirada de [Avila *et al*. 2004]).

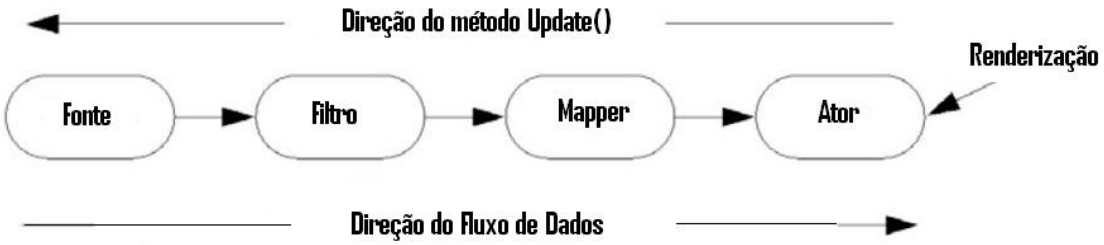
Os *pipelines* de visualização do VTK só são executados quando algum dado relacionado é requisitado. A Figura 6.6 ilustra em alto nível de abstração o *pipeline* de execução. Dependendo de quais partes do *pipeline* estão desatualizados, os filtros podem ser re-executados, para trazer os dados atualizados ao fim da cadeia de fluxo, que podem então ser renderizados por um ator.

Os elementos de um *pipeline* incluem três tipos de objetos:

Sources (fontes)- Geram os dados. Fazem leitura de dados e geram pontos a partir de funções implícitas.

Filters (filtros)- Transformam dados. Extraem informações e dados a partir de outros dados.

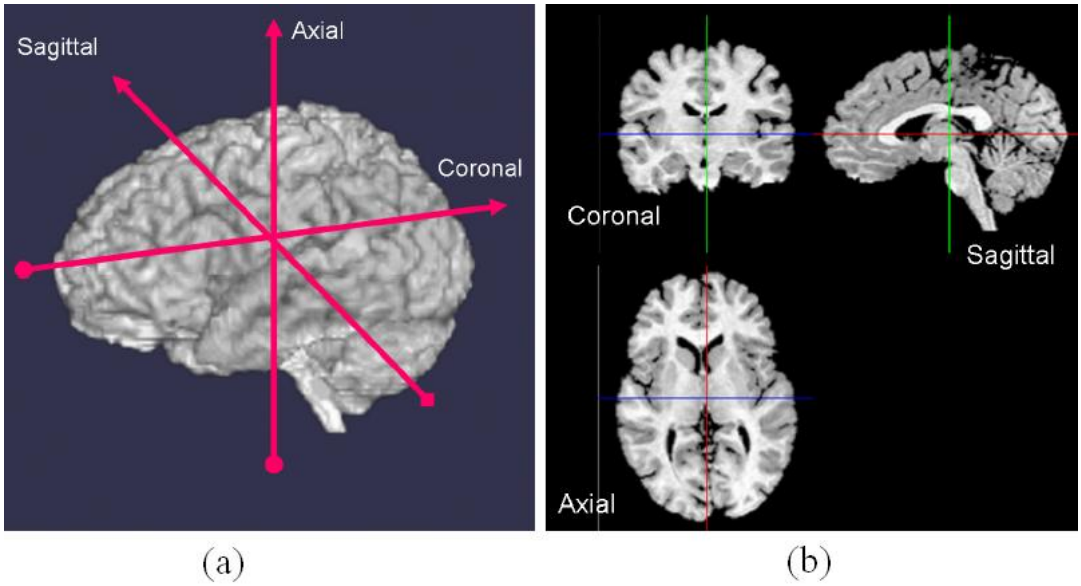
Mappers (mapeadores)- Representam a parte final do fluxo de visualização. São usados para converter dados em primitivas gráficas ou escrever gráficos em arquivos.



[Figura 6.6] *Pipeline* de execução (Figura modificada de [Avila *et al*. 2004]).

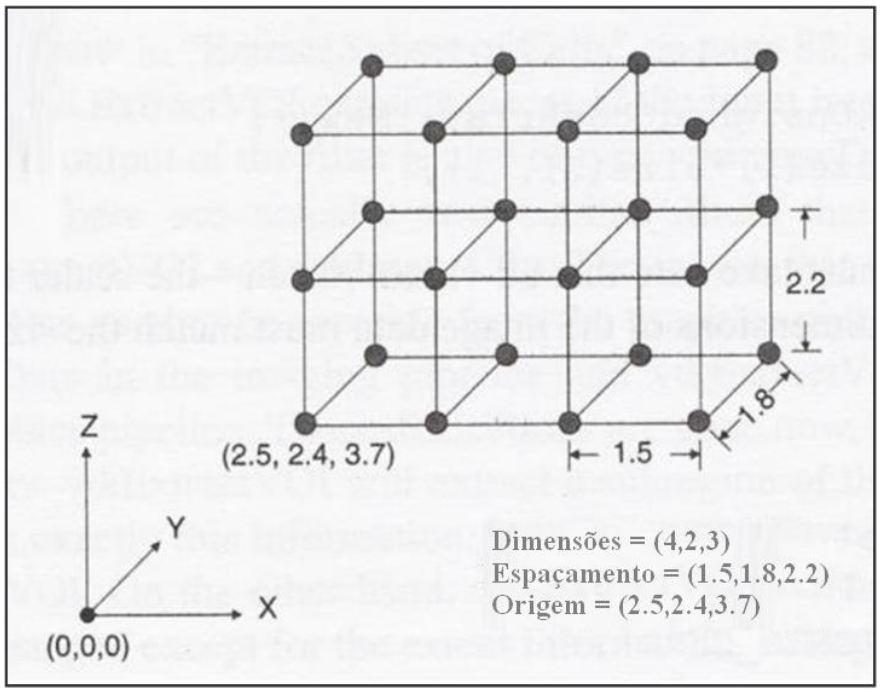
6.3.2 *Imagens em VTK*

Em geral, em imagens médicas, é necessário não somente manter suas intensidades, mas também seus atributos adicionais como: as dimensões dos *voxels*, a orientação da imagem (a relação da orientação dos eixos x,y e z em relação ao corpo humano- Figura 6.7), a posição dos *voxels*, etc.



[Figura 6.7] (a) Orientação padrão de aquisição de imagens: sagital, axial e coronal. (b) Fatias coronal, sagital e axial (Figura retirada de [Papademetris 2006]).

Em VTK, as imagens são armazenadas usando a classe *vtkImageData*. A classe *vtkImageData* é definida pelas dimensões, espaçamento e origem [Avila *et al*. 2004]. As dimensões são o número de *voxels* ou *pixels* ao longo de cada um dos eixos x, y e z. A origem é a posição da coordenada real do canto esquerdo inferior da primeira fatia de dados. O espaçamento é a distância entre os *pixels* ao longo dos eixos x, y e z. A Figura 6.8 exemplifica os parâmetros da estrutura *vtkImageData*.



[Figura 6.8] Funções que definem a classe *vtkImageData* (Figura modificada de [Avila *et al*. 2004]).

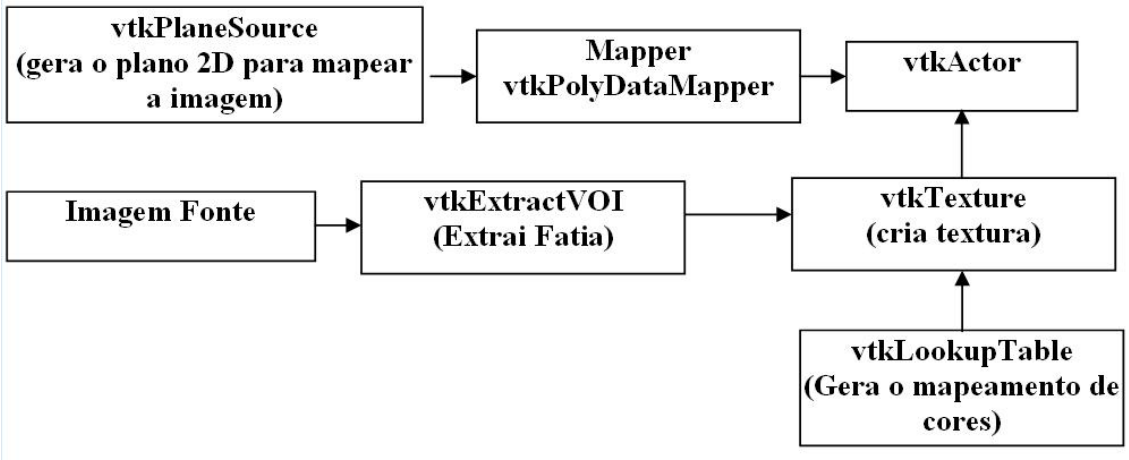
Quando imagens 3D são mostradas, algumas dificuldades são encontradas. Isto se deve ao fato destas imagens serem “espessas” e o seu interior ser mais importante que as bordas que as delimitam.

Existem três maneiras de mostrar imagens usando VTK [Papademetris 2006]: (i) usando *vtkImageViewer*; (ii) usando mapeamento de textura sobre planos e (iii) renderização de volume.

A classe *vtkImageViewer* é o modo mais conveniente de mostrar imagens 2D em VTK. Entre as várias funções que se pode ajustar usando esta classe, é importante citar: ajuste do tamanho e cor da janela em que a imagem será mostrada e alteração do mapa de cores da imagem.

Outra maneira de visualizar imagens usando VTK é pela utilização da textura. Textura, em processamento de imagens, é um atributo representando o arranjo espacial dos níveis de cinza dos *pixels* em uma região [Russ 2002]. O *pipeline* usado para

mostrar fatias como texturas mapeadas dentro de planos está esquematizado na Figura 6.9. Em geral existem três partes: (1) Desenha-se a “geometria”, colocando o plano em posição apropriada; (2) A imagem fonte é gerada extraindo uma fatia 2D de uma imagem 3D e (3) A textura é criada combinando a fatia 2D da imagem com o mapa de cores.



[Figura 6.9] *Pipeline* para mostrar uma fatia de uma imagem como uma textura mapeada sobre um plano retangular. A primeira linha do esquema é o *pipeline* da geometria, que simplesmente mostra a limitação do plano. A linha do meio é o *pipeline* da textura para gerar a aparência do retângulo. O mapa de cores (última linha) é adicionado para fazer o mapeamento da intensidade da imagem para cores (Figura retirada de [Papademetris 2006]).

A renderização de volume é uma ferramenta poderosa muito utilizada em diversos estudos. Existem três técnicas principais para renderização volumétrica implementadas no VTK: *ray casting*, mapeamento de textura 2D, e suporte para o *hardware* de renderização volumétrica, o *VolumePro*.

- **Ray Casting**: É a técnica mais empregada e a que apresenta melhor qualidade. Mais detalhes desta técnica serão abordados a seguir.

- **Mapeamento de Textura 2D**: Como uma alternativa ao *ray casting*, a renderização de volume pode ser feita mapeando o volume sobre um polígono. A classe do VTK envolvida neste processo é *vtkVolumeTextureMapper2D*, que é a classe para *mapeadores* de volume que emprega *hardware* de mapeamento de textura como técnica de renderização.
- **VolumePro**: *Hardware* dedicado para renderização de volume. A classe de mapeador de volume para esta técnica é *vtkVolumeProMapper*.

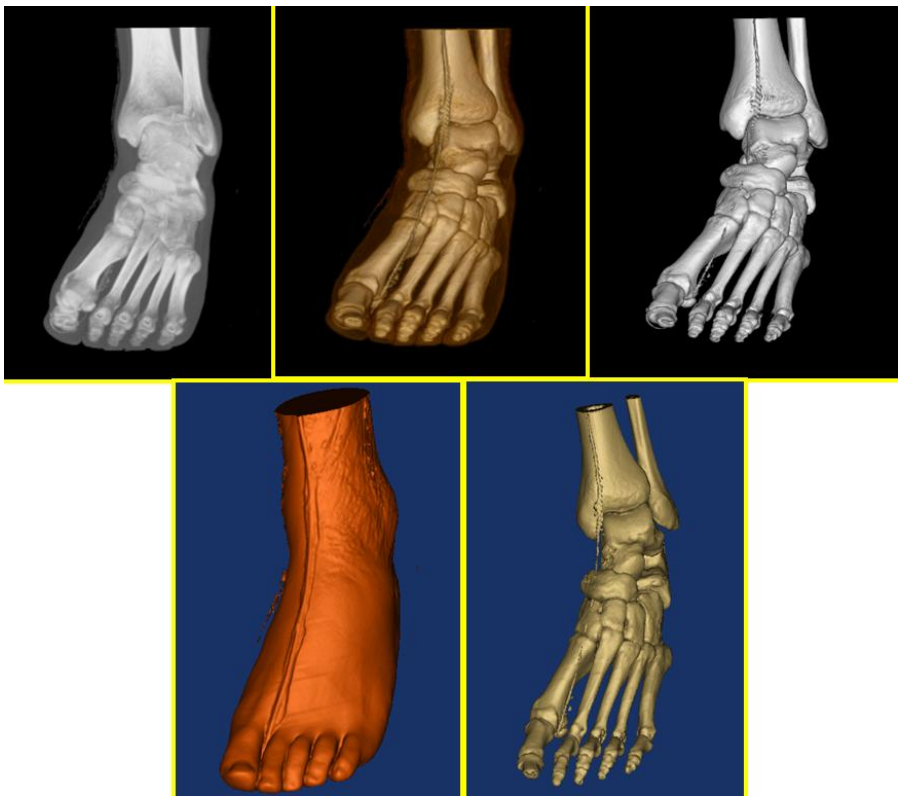
Para cumprir os objetivos deste trabalho, optou-se pelo uso da técnica de *ray casting*, que mostrou-se bastante oportuna devido, principalmente, ao fato de ser mais precisa do que as demais técnicas citadas anteriormente. Portanto, o texto a seguir refere-se a uma análise mais detalhada da utilização de *Ray Casting* no VTK.

A classe *vtkVolumeRayCastMapper* é o *mapeador* de volume que emprega o algoritmo de *ray casting* para fazer a renderização. Existem alguns parâmetros que especificam o volume renderizado pelo algoritmo *ray casting*. Aqui será citado apenas da função *ray cast*, utilizada neste trabalho, que deve ser estabelecida no *mapeador*. Esta função faz a composição do volume de acordo com as propriedades definidas na classe *vtkVolumeProperty*. Existem três subclasses suportadas pela *vtkVolumeRayCastFunction*: a *vtkVolumeRayCastIsosurfaceFunction* que pode ser usada para renderizar isosuperfícies dentro de dados volumétricos, a classe *vtkVolumeRayCastMIPFunction* que pode ser usada para gerar projeções de intensidade máxima e *vtkVolumeRayCastCompositeFunction* que é usada para renderizar o volume fazendo a composição de acordo com as propriedades definidas na classe

vtkVolumeProperty. Esta por sua vez tem o propósito de armazenar os parâmetros necessários para renderização do volume. Alguns deles são:

- Coeficiente de reflexão ambiente, difusa e especular;
- Funções de transferência;
- Tipo de interpolação para reamostragem.

Um exemplo de imagens que podem ser geradas usando as subclasses suportadas por *vtkVolumeRayCastFunction* pode ser vista na Figura 6.10. A primeira imagem do canto superior esquerdo foi gerada usando projeção de intensidade máxima. As outras duas imagens superiores foram geradas usando composição, enquanto as imagens inferiores foram geradas usando uma função de isosuperfície.



[Figura 6.10] Renderização de volume usando o algoritmo *ray casting* (Figura retirada de [Avila *et al*. 2004]).

Capítulo 7

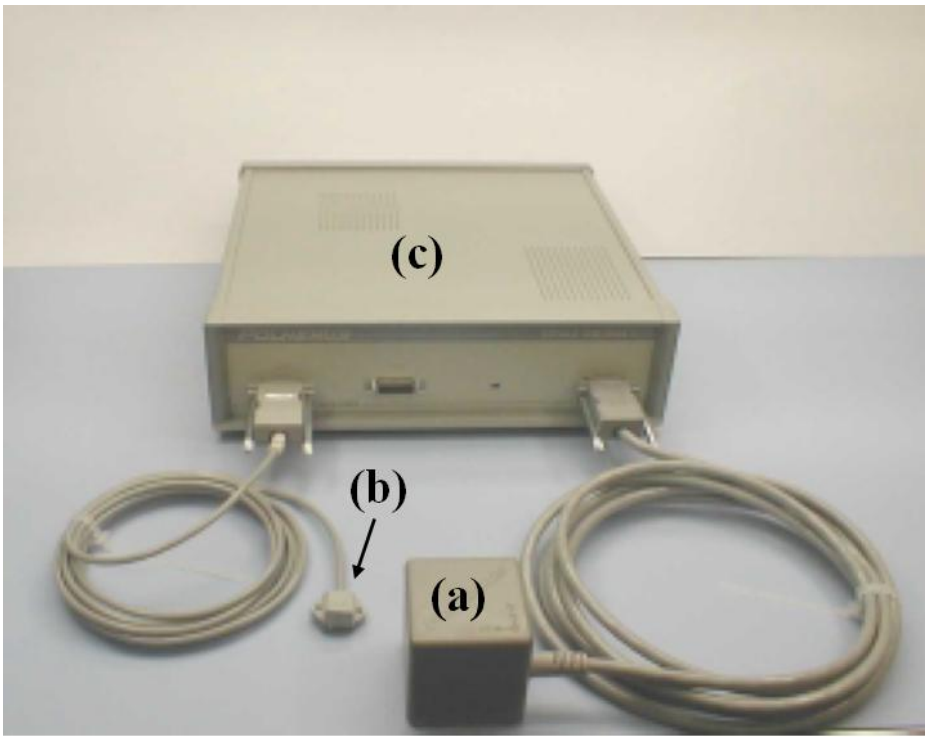
Transdutores de Posição

Transdutores de posição 3D (*3D position trackers*) são rastreadores espaciais que medem, em tempo real, a posição e orientação angular de objetos no espaço. Um objeto movendo-se no espaço 3D tem seis graus de liberdade, sendo três translações e três rotações. Em aplicações biomédicas, esses transdutores são usados para medir os movimentos da cabeça, mãos e, eventualmente, dos membros do usuário, com a proposta de controlar a visão, locomoção e manipulação de objetos.

7.1 Transdutores de Posição Magnéticos

O princípio de funcionamento do transdutor magnético de posição é baseado na técnica de indução magnética, usando bobinas sensoras e excitadoras. Um exemplo deste transdutor de posicionamento 3D é o *Polhemus*® (Figura 7.1). As bobinas sensoras e transmissoras do *Polhemus*® consistem de 3 bobinas quadradas e ortogonais [Polhemus, 2004]. A precisão do *Polhemus*® é da ordem de 2 mm [Polhemus 2004].

O *Polhemus*® consiste de um transmissor, um receptor e uma unidade de controle (Figura 7.1). A unidade de controle é conectada ao computador através da porta serial ou USB. O transmissor permanece estacionário (sistema de coordenada de referência), enquanto o receptor é acoplado ao objeto móvel. Quando o receptor se move com o objeto, a unidade de controle calcula a transformação de corpo rígido que associa os sistemas de coordenadas do transmissor e receptor [Pagoulatos *et al*. 1999]. Este cálculo é feito baseado na variação do campo magnético induzido na bobina receptora, quando sua distância relativa à bobina transmissora é modificada.



[Figura 7.1] Transdutor de posicionamento 3D *Polhemus*®. É possível observar em (a) o transmissor, em (b) o receptor e em (c) a unidade de controle.

7.2 Transdutores de Posição Ópticos

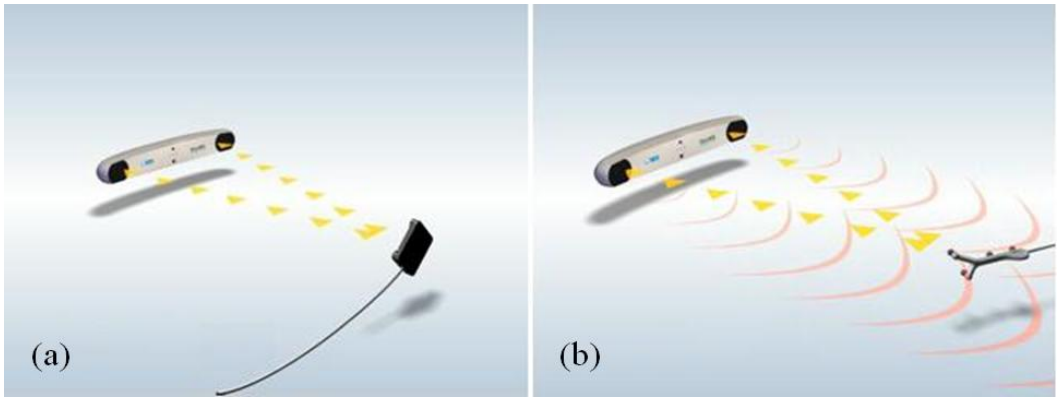
O princípio de funcionamento destes transdutores está baseado na análise da projeção bidimensional de uma imagem, ou na determinação dos ângulos de feixes da varredura, para calcular a posição e orientação de um dado objeto. Os transdutores ópticos são geralmente câmeras.

Com a utilização de câmeras, técnicas de visão computacional devem ser utilizadas para determinar a posição do objeto. Se somente uma câmera for utilizada, é possível determinar um segmento de reta que passa pelo objeto detectado e pelo centro de projeção da câmera. Usando mais de uma câmera, pode-se determinar a posição e orientação do objeto.

Em geral, os transdutores ópticos oferecem uma solução de rastreamento simples com precisão da ordem de 0.1-0.5 mm [Simon 1997]. Quanto à interferência do meio, o laser e outros emissores podem refletir em objetos próximos, atrapalhando a medição.

O objetivo de se utilizar um transdutor óptico é calcular a localização e orientação de um objeto ou ferramenta dentro de um sistema de coordenadas. Rastreadores ópticos usam um transdutor de posição para detectar a emissão de infravermelho ou marcadores fixados na ferramenta ou objeto, que refletem luz infravermelha. O transdutor de posição calcula a posição e orientação da ferramenta baseado na informação que este transdutor recebe dos marcadores. Existem dois tipos de marcadores que podem ser usados no rastreamento óptico (Figura 7.2):

- **Marcadores Ativos**: são marcadores emissores de infravermelho que são ativados por um sinal elétrico;
- **Marcadores Passivos**: são marcadores esféricos que refletem luz infravermelha, emitida por iluminadores sobre o transdutor de posição.



[Figura 7.2] (a) Marcador Ativo e (b) Marcador Passivo (Imagem Retirada [Ndigital 2007a]).

Durante o procedimento cirúrgico, os transdutores ópticos funcionam rastreando a posição dos marcadores infravermelhos sobre a ferramenta cirúrgica. A Figura 7.3

mostra um tipo de rastreador óptico que pode ser usado em cirurgia. Em geral, transdutores ópticos oferecem uma solução de rastreamento simples com boa precisão, porém eles apresentam certas desvantagens quando usados em cirurgia, devido principalmente ao fato de serem volumosos e da câmera ter de ficar sempre enxergando o objeto que está sendo rastreado. Estas desvantagens são bem críticas, quando consideramos o espaço restrito de uma sala cirúrgica, já que a mesma possui vários equipamentos e uma equipe de médicos e enfermeiros.

[Figura 7.3] Exemplo de sistema de rastreamento óptico (a) *Polaris Spectra*® e (b) *Polaris Vicra*® (Imagem retirada de [Polaris 2007]).

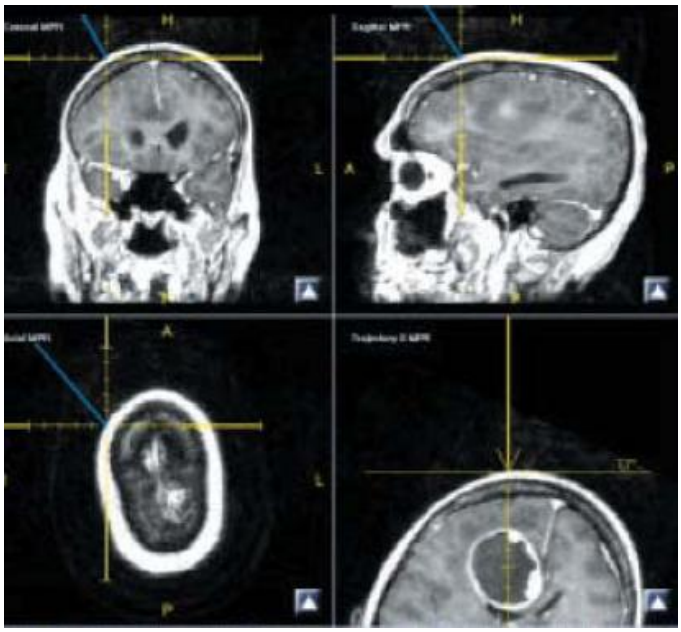
Os transdutores mostrados na Figura 7.3 são bastante usados em cirurgia guiada por imagem, embora os mesmos apresentem algumas restrições, salientadas no parágrafo anterior.

O transdutor de posição *Polaris*® é montado na sala cirúrgica, estrategicamente posicionado para maximizar a visibilidade dos instrumentos por ele guiados, durante o procedimento cirúrgico. A cabeça do paciente é fixada na mesa como mostra a Figura 7.4.

[Figura 7.4] Imagem mostrando o posicionamento do transdutor e da estrutura que fixa a cabeça do paciente (Imagem retirada de Ndigital 2007b).

Depois da anestesia, a posição física do paciente é “registrada” com os dados de IRM pré-operatórios, fazendo uma amostragem de pontos sobre a superfície da pele com o sistema de rastreamento óptico, e calculando a transformação que melhor alinha estes pontos, ou através do uso de marcadores externos ou anatômicos que são identificados no paciente e nas imagens pré-operatórias. A identificação destes marcadores serve para encontrar a transformação rígida que relaciona o espaço do paciente com o espaço das imagens [Comeau *et al*. 2000].

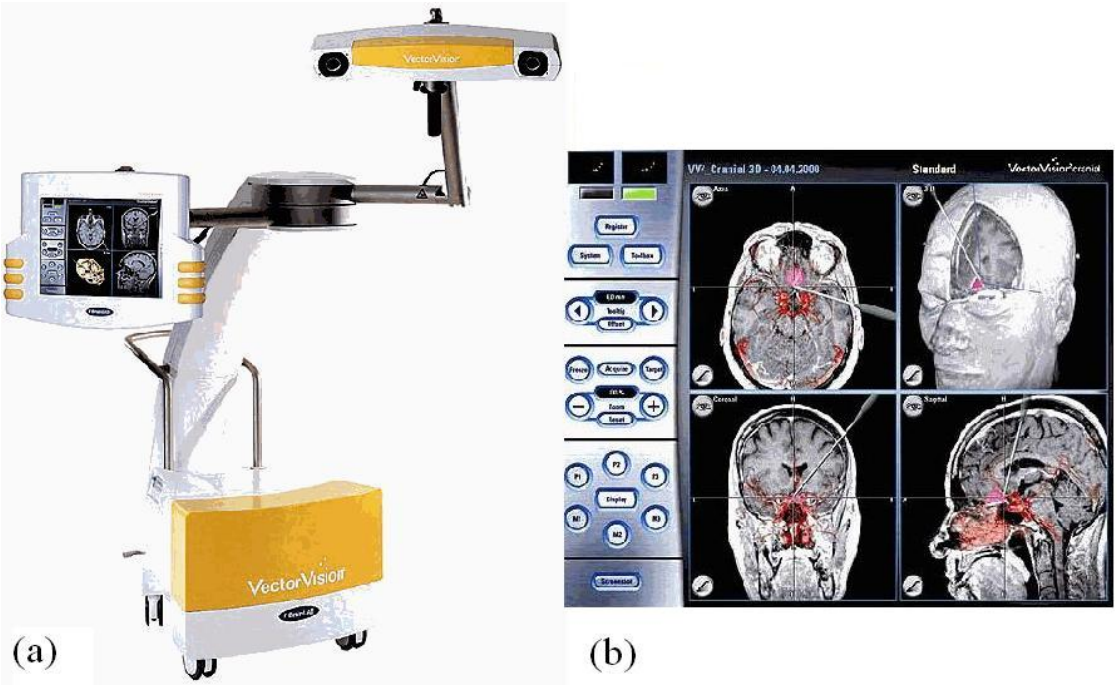
O cirurgião usa o sistema de terapia assistido por computador para fazer o planejamento e orientação da craniotomia. Um exemplo de interface de *software* usado para terapia assistida por computador pode ser vista na Figura 7.5. Os limites médio, lateral, anterior e posterior da craniotomia são mapeados usando o rastreador e os cortes ortogonais.



[Figura 7.5] Sistema de Navegação produzido pelo software de aplicação IGS Scout SNS (Imagem retirada de [Ndigital 2007b]).

O cirurgião faz a craniotomia, dissecando a dura-máter e cortando o tumor utilizando imagens pré-operatórias. Porém, os cirurgiões percebem que o cérebro se desloca uma vez que uma janela no crânio é aberta, tornando o uso do sistema de terapia assistido por computador limitado, visto que as imagens utilizadas são estacionárias e obtidas antes do início da cirurgia, e não podem detectar o deslocamento ocorrido após a craniotomia [Ndigital 2007b].

Existem alguns sistemas de neuronavegação comerciais, sendo um destes o sistema *BrainLAB VectorVision*® (Figura 7.6). Neste sistema, o rastreamento de ferramentas cirúrgicas é através do uso de transdutores de posição óptico.



[Figura 7.6] (a)Sistema comercial de neuronavegação *BrainLAB VectorVision*® e (b) Interface do software de neuronavegação (Imagens retiradas de [BrainLab 2007]).

O sistema *BrainLAB VectorVision*® é um sistema de câmeras que trabalha passivamente, isto é, luz infravermelha é emitida pelos transmissores instalados próximos a câmeras receptoras, e é refletida por marcadores (Figura 7.2(b)). Estas reflexões são detectadas e usadas pelo computador para estabelecer a posição da cabeça do paciente e dos instrumentos cirúrgicos [Stelter *et al*. 2006]. Este sistema de neuronavegação correlaciona o espaço da ferramenta cirúrgica, rastreada por um marcador passivo, com o espaço de imagens pré-operatórias do paciente, inicialmente processadas pela *workstation VectorVision*®.

7.3 Transdutores de Posição Acústicos

O princípio de funcionamento dos rastreadores acústicos utiliza, tipicamente, ondas sonoras ultra-sônicas para medir distâncias [Shuxiang *et al*. 2003]. Os métodos

mais usados são o cálculo do tempo de vôo e a coerência de fase. Em ambos, o objetivo é converter tempo em distância. Um único par transmissor/receptor fornece a distância do objeto em relação a um ponto fixo. Para estimar a posição são necessários um transmissor e três receptores ou um receptor e três transmissores. A configuração do sistema não é cara, pois o equipamento necessário é composto de microfones, alto-falantes e um computador. A precisão deste tipo de transdutor é da ordem de 1mm [Simon 1997].

Quanto à interferência do meio, as propriedades do som limitam esse método. O desempenho é degradado na presença de um ambiente ruidoso ou devido à geração de ecos. O som deve percorrer um caminho sem obstrução entre os alto-falantes e os microfones.

Devido às restrições de interferência, a distância média entre receptor e transmissor são de alguns metros, o que pode tornar seu uso em cirurgias limitado. A Figura 7.7 mostra um exemplo de transdutor acústico de posição da marca *Logitech Tracker*®.

[Figura 7.7] Dispositivo acústico *Logitech Tracker*® (Imagem retirada de [Vrealities 2007]).

Capítulo 8

Materiais e Métodos

Neste capítulo serão descritos os equipamentos, ferramentas e métodos empregados para o desenvolvimento da pesquisa apresentada.

8.1 Estrutura Básica de Desenvolvimento

Os algoritmos relacionados à visualização de dados volumétricos possuem um alto custo computacional, necessitando não só de um equipamento robusto como também de um sistema operacional que garanta um bom desempenho.

O objetivo do desenvolvimento do programa proposto nesta pesquisa é que o usuário tenha uma resposta interativa rápida resultante de sua ação. Para isso, buscou-se otimizar ao máximo o programa, para que o mesmo execute apenas as tarefas que são necessárias para o desenvolvimento do trabalho. Outro comprometimento é que o programa forneça um sistema de visualização de imagem 3D para o planejamento cirúrgico, além de possibilitar a extração de planos do volume de IRM previamente construído, durante procedimentos cirúrgicos, com auxílio de um transdutor de posição.

O sistema operacional escolhido para a realização deste trabalho foi o *Microsoft Windows*®, pois o mesmo permitiu um bom desempenho na realização das tarefas propostas neste trabalho.

8.2 Linguagem de Desenvolvimento

O desenvolvimento de um *software* confiável, que seja consistente, estável, rápido e robusto para manipular imagens depende basicamente da linguagem que será empregada.

Para atender as características descritas no parágrafo anterior, a linguagem *Python* foi escolhida como a mais adequada, pois além de oferecer pleno suporte à programação orientada a objetos, é uma linguagem em que o VTK, utilizado neste trabalho, pode ser programado. Por sua flexibilidade e eficiência esta linguagem de programação ganhou credibilidade de instituições como a NASA (*National Aeronautics and Space Administration*) e Google, que passaram a empregá-la no desenvolvimento de seus *softwares*.

O computador utilizado foi um microcomputador *Pentium* IV com 1.8GHz de velocidade e 512MB de memória RAM.

8.3 Base de Imagens para Validação dos Programas

As IRM foram adquiridas no equipamento da *Siemens Magneton Vision*® de 1.5T, com bobinas gradiente de 25mT/m e uma bobina principal emissora/receptora de polarização circular, comercialmente disponível, usando o *software* fornecido pelo fabricante. As IRM foram adquiridas obedecendo um protocolo clínico padrão para epilepsia no centro cirúrgico do Hospital das Clínicas de Ribeirão Preto (HCRP).

A seqüência utilizada para aquisição das IRM foi gradiente eco T1 com espessura de corte de 1mm contíguo, matriz quadrada de 256x256, campo de visão (FOV) de 25.6cm , TR (tempo de repetição): 9.7ms e TE (tempo de eco): 4.0ms.

8.4 Transdutor de Posição Utilizado

A aplicação dada ao transdutor de posição neste trabalho é atualizar (ou renderizar) o volume de IRM conforme a visão do usuário, com a finalidade de extrair fatias deste volume. Durante o desenvolvimento do trabalho, foram feitos alguns testes para detectar a interferência do centro cirúrgico sobre o transdutor de posição. Estes testes foram realizados fazendo-se deslocamentos pré-estabelecidos no elemento receptor do *Polhemus*®. Como resultado, pôde-se perceber que o transdutor magnético mostrou-se totalmente estável no centro cirúrgico.

Observando as características dos transdutores de posição existentes, como visto no capítulo 7, optou-se pelo uso do transdutor magnético *Polhemus*®. A escolha deve-se ao fato de o mesmo apresentar algumas características essenciais para sua utilização durante a cirurgia. Estas características são:

1. **Precisão/velocidade**: estes sistemas são bastante precisos, cerca de 2 mm para a posição absoluta e 0.1º para orientação. A velocidade de captura dos dados é de 100 a 200 medidas/segundo. A precisão na medida dos deslocamentos relativos e de curto alcance é menor que 1 mm.

2. **Interferência**: não apresentou interferência em ambiente cirúrgico, e diferentemente dos demais transdutores, este não é interferido pela presença dos cirurgiões durante o ato cirúrgico.

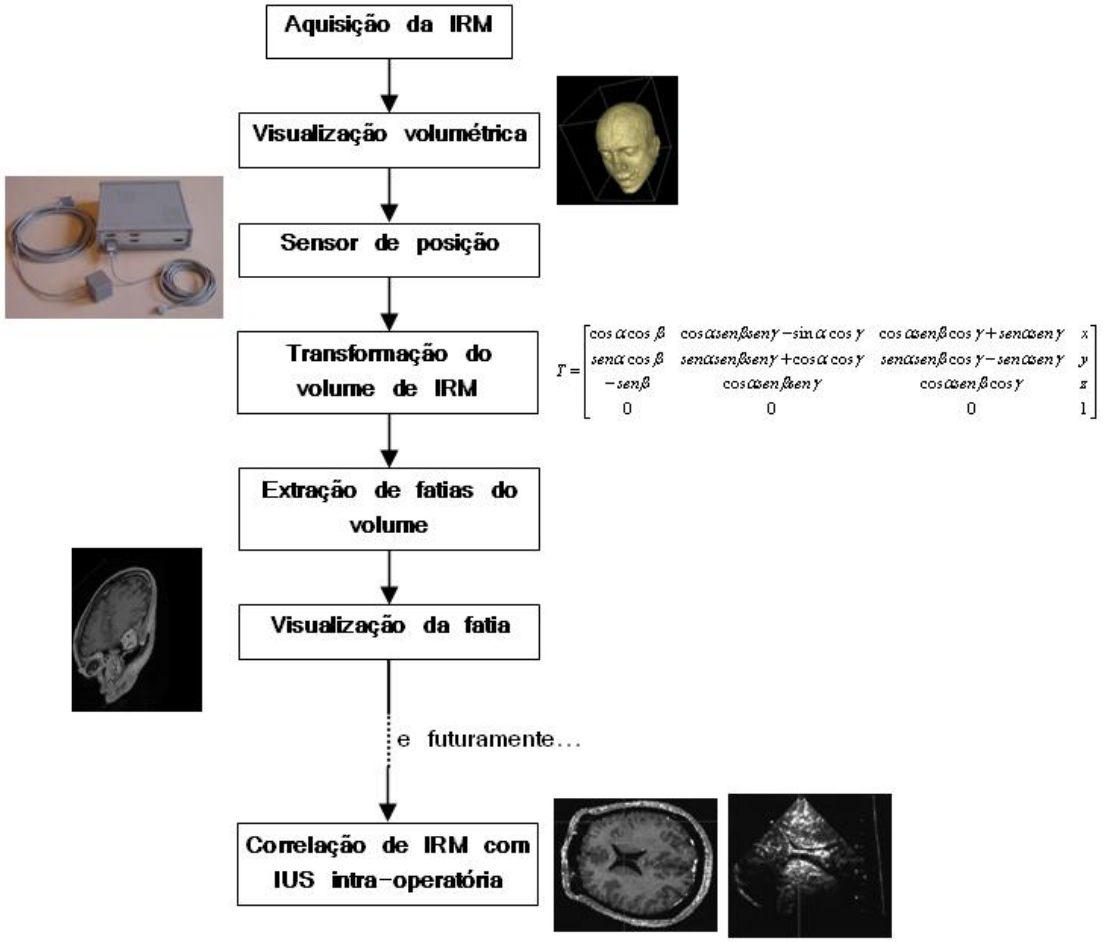
3. **Uso**: é fácil de usar em sala cirúrgica, visto que seu receptor é relativamente pequeno;

Além de todas estas características, bastante pertinentes para uso deste tipo de transdutor de posição em cirurgias, pode-se adicionar a estas o fato do mesmo ainda não ser utilizado nas implementações de produtos comerciais, o que proporciona ao trabalho desenvolvido um diferencial adicional e inovador.

8.5 Metodologia

A partir de agora, será tratada a metodologia usada para fazer esta pesquisa, ou seja, como foram implementados os programas para solucionar o objetivo do trabalho proposto.

Antes de iniciar a abordagem da implementação, são mostradas na forma de um diagrama esquemático (Figura 8.1), as várias etapas desenvolvidas neste trabalho e a ligação entre as mesmas. Cada uma destas etapas, mostradas no diagrama, será, a seguir, explicada detalhadamente.



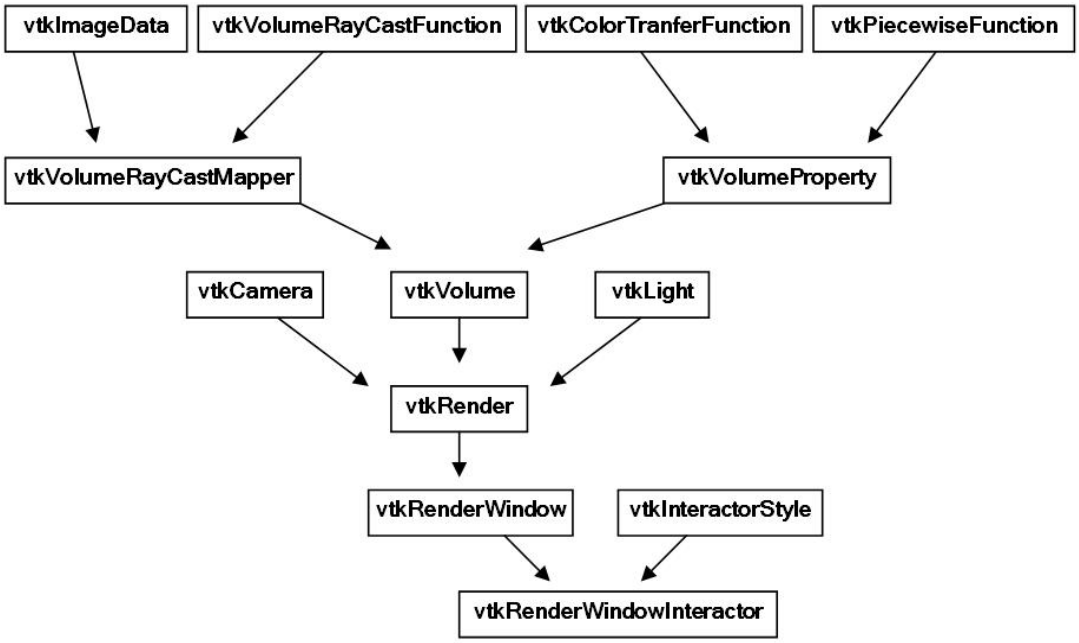
[Figura 8.1] Diagrama esquemático das etapas desenvolvidas neste trabalho.

8.5.1 Implementação do Programa para Planejamento Pré-Cirúrgico

Para desenvolver um sistema flexível, portátil e fácil de estender, optou-se pela utilização de *softwares* livres: linguagem de programação *Python* e a biblioteca gráfica VTK 5.0.

O formato dos arquivos suportados pelo programa implementado são do tipo DICOM, sendo que o programa pode sofrer pequenas modificações para suportar outros tipos de arquivos.

A arquitetura desenvolvida inclui várias classes do VTK. Nos próximos parágrafos, serão abordadas as classes e como as mesmas foram inseridas durante a implementação do programa. A Figura 8.2 mostra um esquema das principais classes do VTK para renderização de volume, e como uma classe serve de entrada para outra. A seta indica a seguinte orientação: $A \rightarrow B$, neste caso A é a entrada de B.



[Figura 8.2] Esquema das principais classes do VTK para renderização de volume.

Primeiramente, as séries de fatias 2D de IRM que compõem o volume foram lidas através da classe *vtkDICOMImageReader*. Esta classe permite que seja indicado o diretório que as IRM se encontram.

Após as IRM serem lidas, as mesmas foram armazenadas usando a classe *vtkImageData*. Além disso, a utilização desta classe também permitiu que fossem definidas as dimensões, espaçamento e origem do volume de IRM.

Um dos mais importantes benefícios de renderização do volume de imagens médicas é a habilidade de distinguir entre diferentes tecidos no volume. Isto é chamado de classificação, e é efetuada pela função de transferência, que mapeia as propriedades dos dados extraídos ou calculados de um dado ponto no volume, para propriedades ópticas como cor e opacidade. O próximo passo no programa foi mapear os dados, usando uma função de transferência, que no VTK, foi implementado com as seguintes classes: *vtkColorTransferFunction* e *vtkPiecewiseFunction*. A tarefa de especificar as funções de transferência que gerem imagens de qualidade e que transmitam as informações requeridas não é trivial, e tem sido amplamente discutida. O modo usado para encontrar a função de transferência de cor ou opacidade é através de “tentativa e erro”.

Na seqüência, foi usada a classe *vtkVolumeRayCastCompositeFunction* que foi inserida dentro da classe *vtkVolumeRayCastMapper*. O objetivo de usar esta função foi fazer a composição do volume de acordo com as propriedades definidas na classe *vtkVolumeProperty*. Esta, por sua vez, tem o propósito de armazenar os parâmetros necessários à renderização de volume, como: funções de transferência e tipo de interpolação para reamostragem.

Depois que se obteve o volume renderizado, foram inseridos três planos sobre este volume. A classe usada para esta finalidade foi *vtkImagePlaneWidget*, que permite

colocar interativamente um plano sobre um volume de imagens. Cada um dos planos foi especificado com uma origem e dois pontos que, juntos, definem dois eixos para o plano.

Para mostrar as estruturas internas do volume renderizado, foram criadas novas janelas, permitindo visualizar a imagem que está sendo mostrada no plano sobre o volume. Esta etapa foi feita exportando a textura do *plane widget* para um plano que foi criado com a classe *vtkPlaneSource*. A textura foi transferida para o plano usando a classe *vtkTextureMapToPlane*. Para se ter um melhor controle e deixá-lo com o mesmo tamanho do *plane widget*, o plano criado com *vtkPlaneSource* foi especificado com os três pontos que foram determinados para o *plane widget*. Estas especificações foram feitas dentro da classe *vtkPlaneSource.*

Muitas aplicações em VTK necessitam explicitamente criar a classe *vtkLight* ou *vtkCamera*. A classe *vtkCamera* é a câmera virtual para renderização 3D. Uma vez que uma câmera é criada, pode-se acessar o volume renderizado, ajustando alguns parâmetros como posição, ponto focal e campo de visão, para que a primeira cena que apareça na tela seja uma cena pré-determinada. A câmera contém alguns métodos convenientes de rotação ao redor de uma posição ou ponto focal como elevação, giro, etc.

Para visualizar os dados, foi necessário abrir uma janela sobre a tela do computador. Para isto, foi utilizada a classe *vtkRenderWindow*. A funcionalidade desta janela foi permitir a visualização das IRM no espaço 3D bem como a interação direta com as mesmas sobre a cena. A interação padrão do *vtkRenderWindow* é a interação com o mouse. Isto significa que, com o mouse, o usuário pode rotacionar e transladar as IRM.

8.5.2 Implementação do Programa para Extração de Fatias do Volume de IRM

O principal objetivo deste programa é extrair fatias de um volume de IRM. A fatia extraída deste volume servirá para correlacioná-la com IUS intra-operatórias, futuramente, no sistema de neuronavegação que está sendo desenvolvido. Este programa conta basicamente com a leitura do volume de IRM, leitura dos dados do transdutor de posição, transformação geométrica para transformar o volume de IRM e a extração propriamente dita. Para implementar este programa, foram usadas várias classes do VTK, as principais foram: *vtkDICOMImageReader*, *vtkMatrix4x4*, *vtkImageChangeInformation*, *vtkImageReslice* e *vtkImageViewer*. O próximo parágrafo trará mais informações sobre estas classes e o modo como estas foram inseridas no programa.

A leitura do volume de IRM foi feita com o uso da classe *vtkDICOMImageReader*. Os dados do *Polhemus*® foram lidos através do uso de uma biblioteca destinada a fornecer as coordenadas e os ângulos de Euler. Os dados lidos pelo transdutor de posição, foram inseridos no programa de extração de fatias para que os mesmos fossem incluídos na matriz transformação.

As transformações lineares em VTK são representadas internamente como matrizes 4x4. Isto permite o uso de uma única operação para capturar a combinação de translação/rotação/escalonamento [Papademetris 2006]. De forma geral, pode-se escrever a transformação em duas partes como

$$y = Ax + b\,, \tag{8.1}$$

sendo *A* uma matriz 3x3, que combina rotação e escalonamento e *b* um vetor 3x1, que é a translação.

Fazer uso das transformações sem combiná-las demanda muito tempo computacional, por isso neste trabalho as transformações foram combinadas em uma única matriz, usando coordenadas homogêneas, como visto no capítulo 4.

A matriz transformação de corpo rígido genérica, adotada neste trabalho, é dada por [Varandas *et al*. 2004]

$$T = \begin{bmatrix} \cos\alpha\cos\beta & \cos\alpha sen\beta sen\gamma - \sin\alpha\cos\gamma & \cos\alpha sen\beta\cos\gamma + sen\alpha sen\gamma & x \\ sen\alpha\cos\beta & sen\alpha sen\beta sen\gamma + \cos\alpha\cos\gamma & sen\alpha sen\beta\cos\gamma - sen\alpha sen\gamma & y \\ -sen\beta & \cos\alpha sen\beta sen\gamma & \cos\alpha sen\beta\cos\gamma & z \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad 8.2$$

sendo α, β e γ os graus de liberdade referentes à rotação e x, y e z os graus de liberdade referentes à translação. Os ângulos α, β e γ são as rotações ao redor dos eixos x, y e z respectivamente.

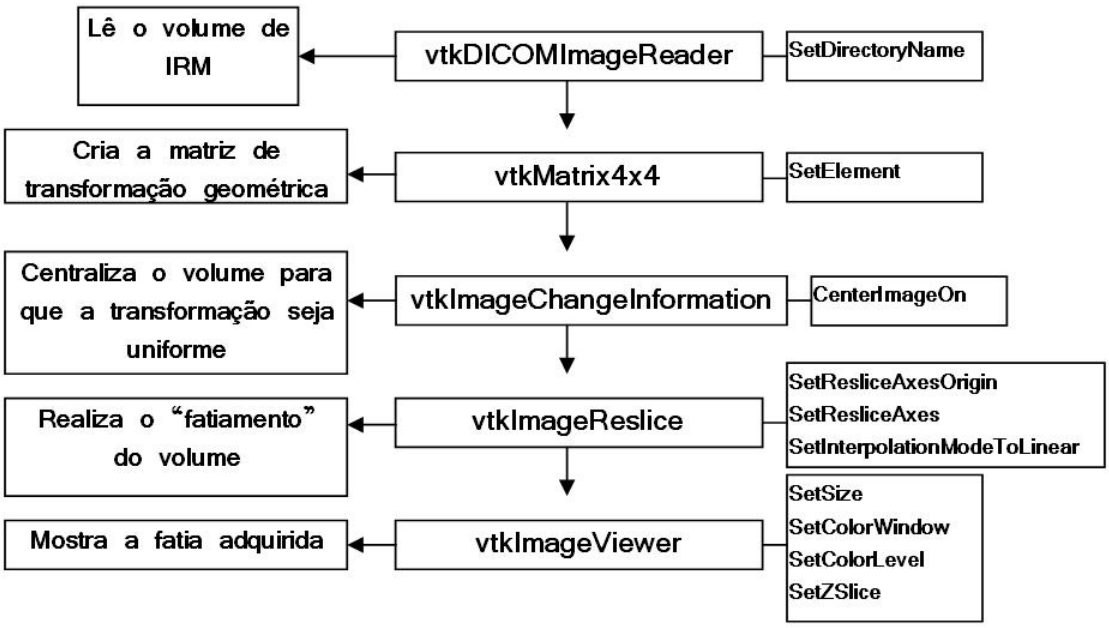
A transformação dada pela equação 8.2 foi gerada e armazenada com a classe *vtkMatrix4x4*. Esta classe do VTK fornece um meio de armazenar matrizes 4x4, além de possuir vários métodos que podem ser empregados para manipulação de matrizes. O método utilizado neste trabalho foi o *SetElement*, que permite escrever os elementos da matriz. Os elementos desta matriz foram calculados mediante os dados fornecidos pelo *Polhemus*®.

Para centralizar o volume e permitir que a transformação fosse uniforme, utilizou-se a classe *vtkImageChangeInformation.* Uma das configurações usadas para esta finalidade foi *CenterImageOn*.

A classe fundamental para extração de fatias de um volume de imagens é a *vtkImageReslice*. Esta classe desempenha várias funções, sendo as principais: aplicação de rotações, translações e escalonamentos sobre uma imagem e a extração de fatias de um volume de imagens. A partir do uso desta classe, pôde-se configurar alguns fatores referentes à transformação aplicada ao volume e à extração de fatias. As principais

configurações usadas na implementação apresentada foram: *SetResliceAxesOrigin*, *SetResliceAxes* e *SetInterpolateMode*.

Depois que o volume de IRM foi transformado e fatiado, a imagem da fatia foi mostrada usando a classe *vtkImageViewer*, que além de permitir a visualização das imagens, proporcionou também configurar o tamanho e cor da janela onde a imagem era mostrada, além do nível de cinzas das mesmas. A Figura 8.3 mostra um esquema das classes principais usadas para o procedimento de extração de fatias.



[Figura 8.3] Esquema das principais classes do VTK para extração de fatias de um volume.

É interessante entender como ocorre a aplicação de uma transformação geométrica sobre o volume, com a utilização do VTK. A explicação seguinte se refere ao funcionamento interno das classes do VTK que permitem esta transformação.

A transformação, de corpo rígido T (equação 8.2), é aplicada ao volume de dados V. A transformação T é uma matriz 4x4, contendo uma combinação de rotações e translações.

Como mostra a equação 8.3, o volume a ser transformado, V, tem *K* fatias, cada uma compreendendo *J* linhas de *I voxels* [Avila *et al*. 2004].

$$T \in \Re^{4x4}$$
$$V = \{v_{ijk}\} \quad \begin{matrix} i=1,..,.I \\ j=1,...,J \\ k=1,...,K. \end{matrix} \qquad 8.3$$

As considerações feitas em VTK são de que a origem do volume V está localizada no canto inferior esquerdo do conjunto de dados, e que o mesmo tem um espaçamento regular de 1.0mm, como mostra a equação 8.4.

$$p_{ijk} = \begin{pmatrix} x_{pijk} \\ y_{pijk} \\ z_{pijk} \\ w_{pijk} \end{pmatrix} = \begin{pmatrix} i-1 \\ j-1 \\ k-1 \\ 1 \end{pmatrix} \qquad 8.4$$

Na equação 8.4, p_{ijk} denota a posição do *voxel* v_{ijk} nas coordenadas do mundo real. Também é possível notar, nesta equação, a utilização de coordenadas homogêneas, w_{pijk}, relatadas no capítulo 4.

A ordem de transformar o volume é processada sequencialmente, sendo as fatias distribuídas de frente para trás. Em cada fatia, primeiro as linhas do *voxel* são consideradas.

Cada posição do *voxel* (*i*,*j*,*k*) que é gerada no processo, é transformada de acordo com a matriz transformação. Isto é feito por uma simples multiplicação entre uma matriz e um vetor.

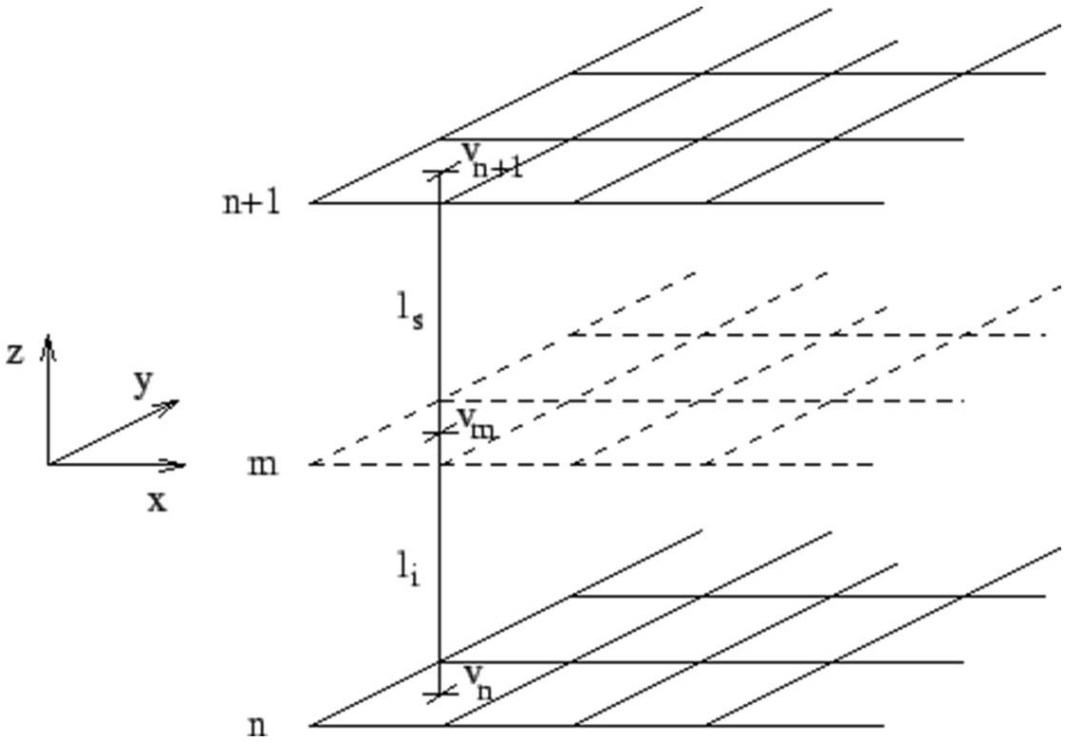
$$\overline{p_{ijk}} = T \cdot p_{ijk} = T \cdot \begin{pmatrix} i-1 \\ j-1 \\ k-1 \\ 1 \end{pmatrix}. \qquad 8.5$$

A equação 8.5 mostra a transformação da posição do *voxel* para a posição do *voxel* transformado $\overline{p_{ijk}}$. É importante notar que esta transformação é aplicada para todos os *voxels* do volume.

Depois que a transformação é aplicada aos *voxels*, é utilizado um método de interpolação para aproximar os valores dos *voxels*. O método mais simples é a interpolação linear, que foi usada para fazer este trabalho.

Na interpolação linear assume-se que a variação de intensidade dos *voxels* é linear na direção ***z***. A Figura 8.4 mostra como é feita a interpolação linear de um *voxel* v_m, considerando seus vizinhos v_n e v_{n+1} na direção ***z***. A intensidade associada ao novo *voxel* v_m é dada pela equação 8.6.

$$i(v_m) = i(v_n) + \frac{(i(v_{n+1}) - i(v_n))l_i}{l_s + l_i} \qquad 8.6$$



[Figura 8.4] Esquema da interpolação linear (Figura retirada de [Dcc 2007]).

8.6 Validação do Programa de Extração de Fatias de IRM

Após a implementação do programa, foi proposto um experimento simples para fazer alguns testes qualitativos, e verificar o seu funcionamento com base na aplicação proposta.

Primeiramente, foi colocado um protótipo de cabeça e o transmissor do *Polhemus*® sobre uma mesa, tomando-se cuidado para que os mesmos ficassem alinhados e fixos (Figura 8.5). O elemento receptor do *Polhemus*® foi acoplado a um transdutor ultrassônico microconvexo. Este transdutor ultrassônico é o mesmo utilizado para obtenção das IUS intra-operatórias, no acompanhamento da remoção da lesão, durante a cirurgia.

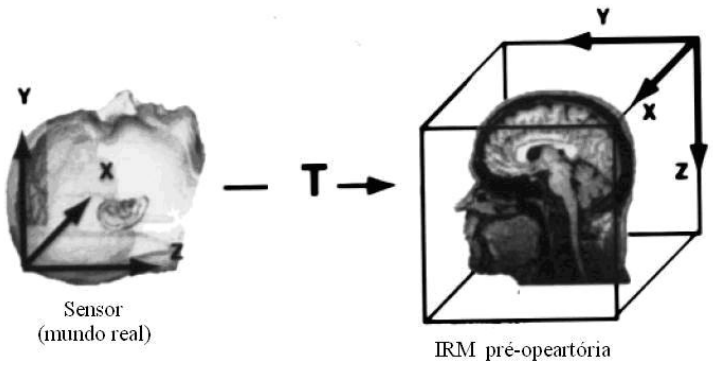
[Figura 8.5] Montagem do experimento proposto.

O procedimento de calibração feito aqui, foi bem simplificado. Foram amostrados três pontos, um sobre o nariz, outro no topo e um na parte posterior do protótipo. Estes pontos serviram de referência, para certificar que as fatias selecionadas

através do uso do programa estavam coerentes com o que se estava observando no protótipo.

O volume de IRM usado para extração de fatias não era do protótipo, porém o mesmo permitiu a possibilidade de avaliar qualitativamente a performance do programa. Os resultados e discussões obtidos, usando as aproximações utilizadas, serão abordados no próximo capítulo deste documento.

Através do experimento mostrado na Figura 8.5, buscou-se fazer uma aproximação para transformar o espaço do paciente para o espaço do volume de IRM (Figura 8.6). Mais uma vez, vale lembrar que a matriz transformação utilizada para transformar estes espaços, foi apenas uma aproximação, através do uso de uma transformação genérica. Esta aproximação foi necessária para validar o programa desenvolvido. Resultados melhores e mais precisos poderão ser obtidos com a utilização de um *phantom* de cabeça com marcadores, que já está sendo desenvolvido, e tem a finalidade de fornecer a melhor transformação para relacionar as IUS, obtidas durante o procedimento cirúrgico, com as IRM obtidas previamente durante o planejamento pré-cirúrgico. As modificações que o *software* sofrerá, após o procedimento de calibração, são mostradas no Anexo A.



[Figura 8.6] Ilustração da transformação de coordenadas do paciente para o sistema de coordenadas do volume de IRM (Figura modificada de [Comeau *et al*. 2000]).

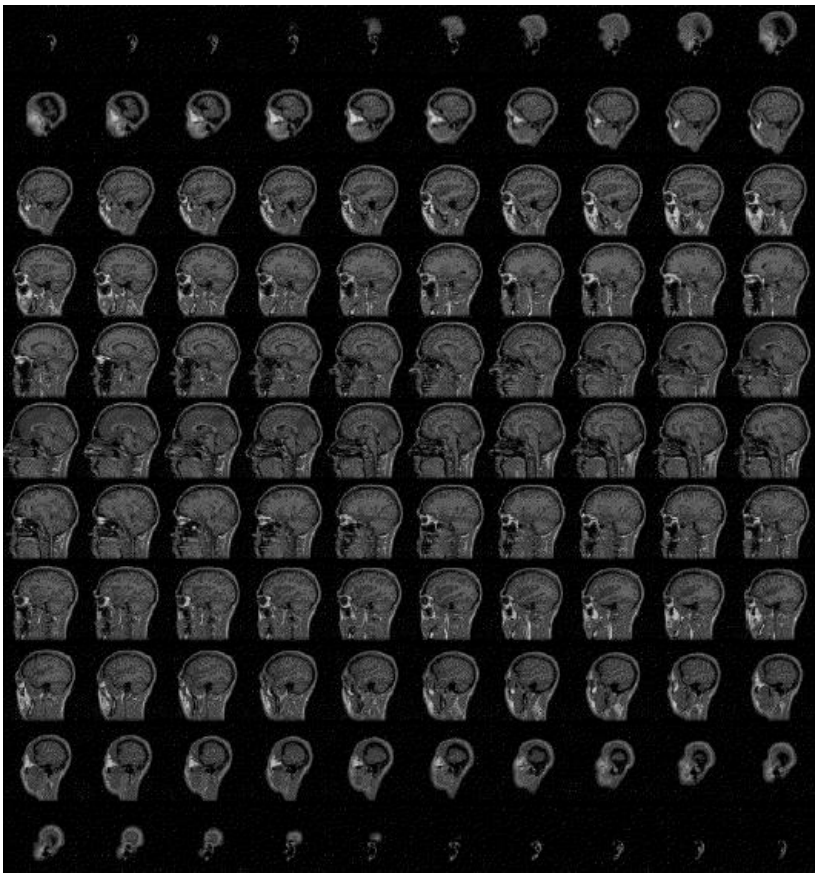
Capítulo 9

Resultados e Discussão

Neste capítulo serão abordados os resultados obtidos com os programas desenvolvidos, juntamente com uma discussão dos mesmos.

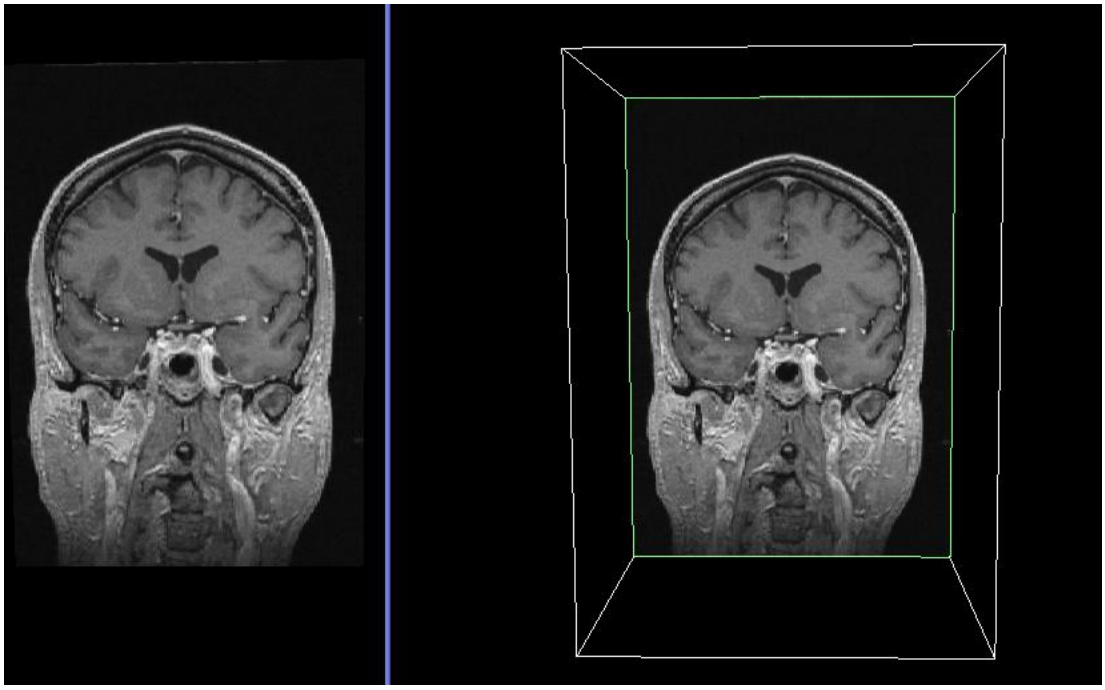
9.1 Visualização Volumétrica

O volume de imagens médicas é armazenado como uma pilha de imagens 2D, mostrado na Figura 9.1. Para atender a necessidade de visualizar esta pilha de imagens de um modo interativo e rápido foram desenvolvidos, neste trabalho, *softwares* que permitem uma visualização 3D destas imagens.

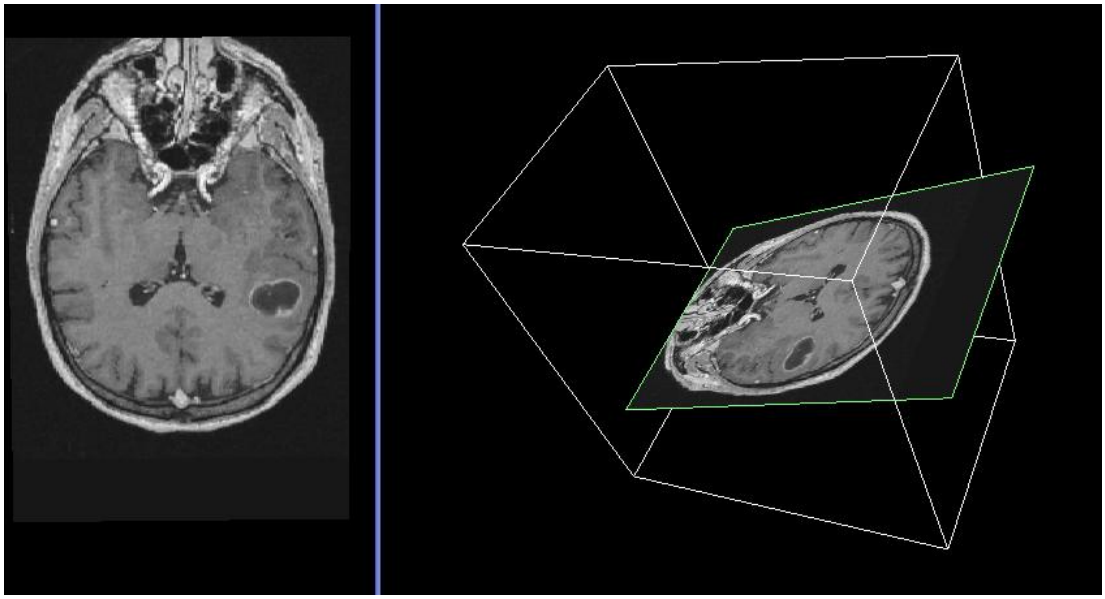


[Figura 9.1] Volume de IRM armazenado como um conjunto de imagens 2D.

Um dos programas de visualização, inicialmente desenvolvido, foi feito através do uso da técnica *multilpanar reformation* (MPR) que é o modo de visualização mais popular na prática clínica [Vidal *et al*. 2006]. Este modo de visualizar volumes é feito através da visualização de uma série de fatias, cada uma paralela a uma das faces do volume (Figura 9.2), ou em uma direção oblíqua (Figura 9.3). O programa que faz uso da técnica de MPR, permite ao usuário mover as fatias 2D de modo que ele possa reconstruir um modelo mental 3D da anatomia.

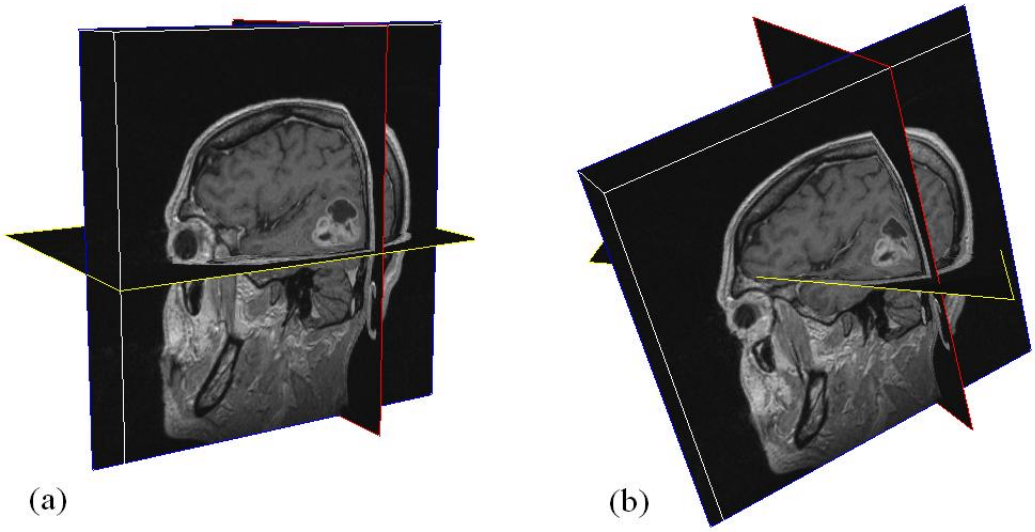


[Figura 9.2] Visualização de uma série de fatias paralelas a uma face do volume.



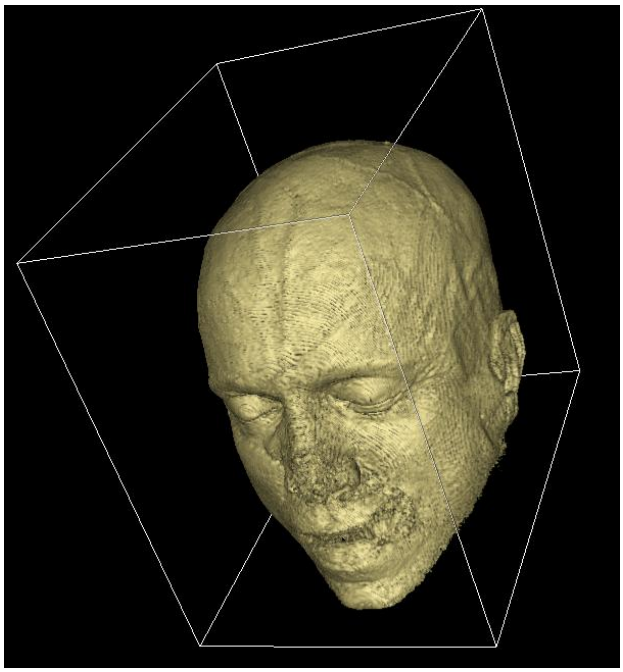
[Figura 9.3] Visualização de uma série de fatias em uma direção oblíqua a uma face do volume.

Fazendo uso desta técnica, o usuário pode iniciar a visualização com uma única fatia em uma orientação padrão ou com as três fatias (sagital, coronal e axial), como mostra a Figura 9.4(a). O usuário também pode rotacionar e transladar estas fatias, usando o botão esquerdo do mouse, para focalizar uma estrutura anatômica específica (Figura 9.4(b)). Neste programa, também é possível escolher o contraste no momento da visualização das imagens, usando o botão de rolagem do mouse.



[Figura 9.4] (a) Visualização das três fatias com orientação padrão e (b) visualização das fatias oblíquas, após a rotação e translação dos planos.

Após a visualização do volume, através de fatias paralelas ou oblíquas a um plano do volume, o programa foi modificado para mostrar o volume inteiro renderizado, sobre a tela do computador (Figura 9.5). Com este programa, é possível mover o volume de IRM com o mouse e mudar a cor e opacidade através da função de transferência, estabelecida no código-fonte. Utilizando o botão direito do mouse, também é possível mover este volume e aplicar transformações ao mesmo. A Figura 9.5 mostra um volume de IRM com cor e opacidade definidos por uma função de transferência. O código-fonte desta função de transferência é mostrado na Figura 9.6. As IRM usadas para reconstruir os volumes tinham dimensões de 256x256x160 *voxels*.

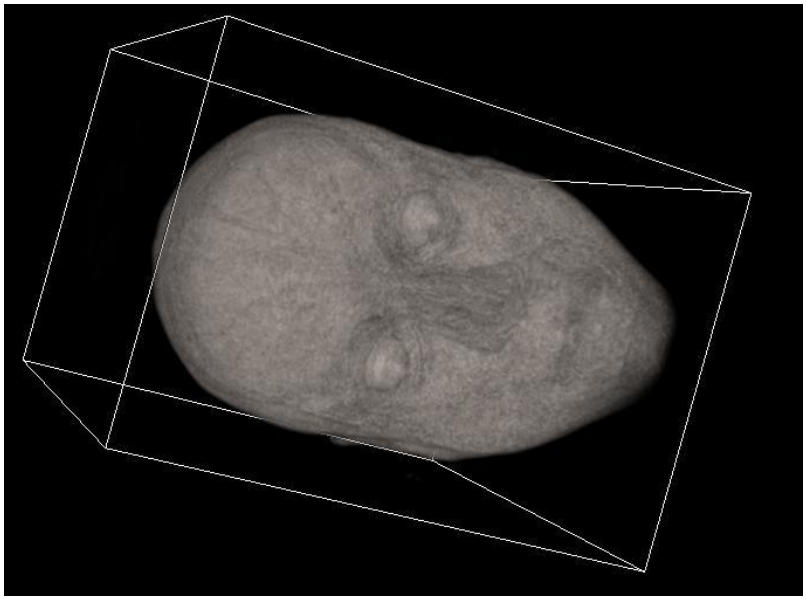


[Figura 9.5] Visualização do volume renderizado.

```
tfun = vtk.vtkPiecewiseFunction()
tfun.AddPoint(70.0, 0.0)
ctfun = vtk.vtkColorTransferFunction()
ctfun.AddRGBPoint(1, 1, .9412,0.6)
```

[Figura 9.6] Código-fonte para definição da função de transferência.

A Figura 9.7 mostra o mesmo volume de IRM renderizado na Figura 9.5, porém com a função de transferência e o plano de visão, modificados. O código-fonte desta função de transferência é mostrado na Figura 9.8.



[Figura 9.7] Visualização do volume renderizado.

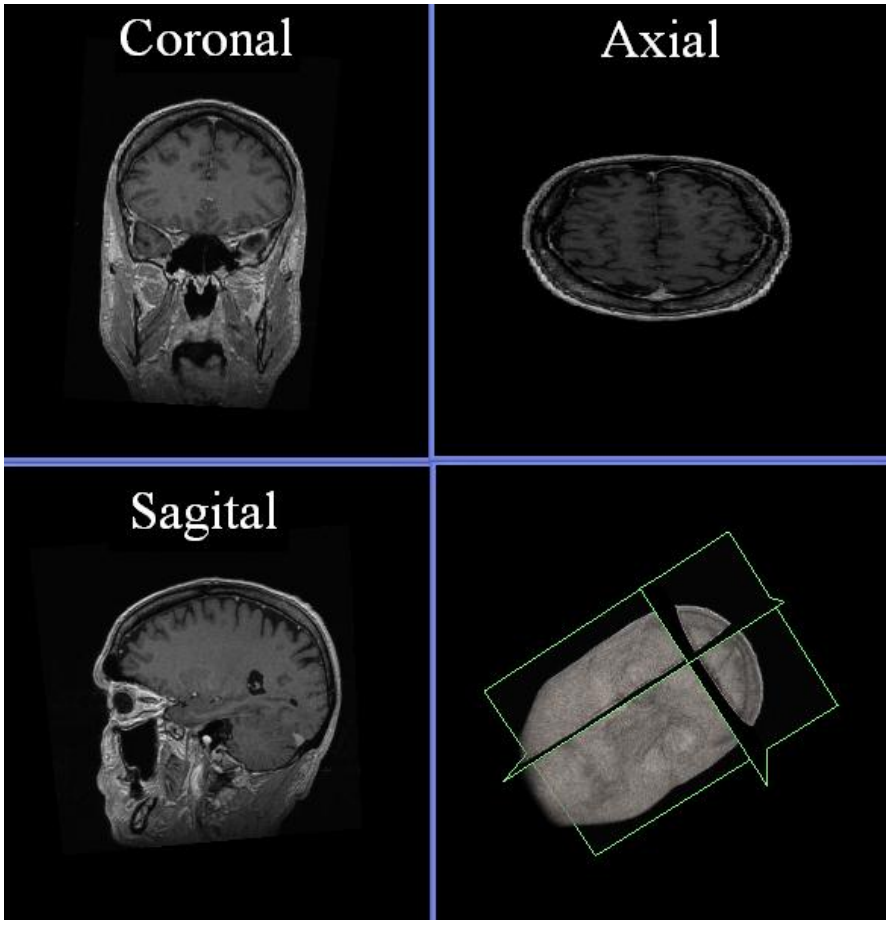
```
tfun=vtk.vtkPiecewiseFunction()
tfun.AddPoint(20, 0.0)
tfun.AddPoint(255, 0.2)
ctfun = vtk.vtkColorTransferFunction()
ctfun.AddRGBPoint(1, 1, .9412,0.6)
ctfun.AddRGBPoint(1, 1, .9412,0.6)
ctfun.AddRGBPoint(1, 1, .9412,0.9)
```

[Figura 9.8] Código-fonte para definição da função de transferência.

Estes resultados iniciais, de visualização de volume, foram feitos para serem usados no *software* de planejamento cirúrgico, visto que a interação com o conjunto de dados 3D permite ao neurocirurgião fazer o planejamento pré-cirúrgico.

9.2 Planejamento Pré-Cirúrgico

Para o planejamento pré-cirúrgico, foi implementado um programa que utiliza além da visualização 3D, planos de corte obtidos com a utilização de *widgets*, como visto no capítulo 8. Estes planos tiveram suas texturas exportadas para três janelas que permitem a visualização dos três planos da cabeça (Figura 9.9). Os planos podem ser movidos livremente, com a utilização do mouse. À medida que os planos são movidos sobre o volume, as imagens correspondentes a posição dos planos, sobre o volume renderizado, são visualizadas imediatamente nas três janelas.



[Figura 9.9] A interface do navegador pré-cirúrgico, mostrando o modelo tridimensional bem como os três planos ortogonais.

O volume renderizado, mostrado na Figura 9.9, pode ser manipulado com o mouse, possibilitando a navegação no interior do volume cerebral. Alguns parâmetros que o usuário pode modificar com a utilização do mouse, no programa desenvolvido são:

- **Rotação**: modifica o ângulo e direção da câmera virtual, criada com a classe *vtkCamera*;
- **Zoom:** aumenta ou diminui a distância da câmera virtual ao volume;
- **Translação:** move a câmera virtual de cima para baixo e da esquerda para direita;
- **Obtenção de planos oblíquos**: permite movimentar os planos sobre o volume, de modo que o usuário tenha acesso a fatias oblíquas;
- **Contraste:** permite ao usuário mudar o contraste das imagens, para melhor visualização.

Através da utilização dos parâmetros anteriormente citados, o neurocirurgião pode encontrar a melhor janela cirúrgica para o início da cirurgia. A IRM escolhida para representar a janela cirúrgica, será usada no início do procedimento de correlação com as IUS intra-operatórias, além de ser a imagem usada como referência no programa de extração de fatias.

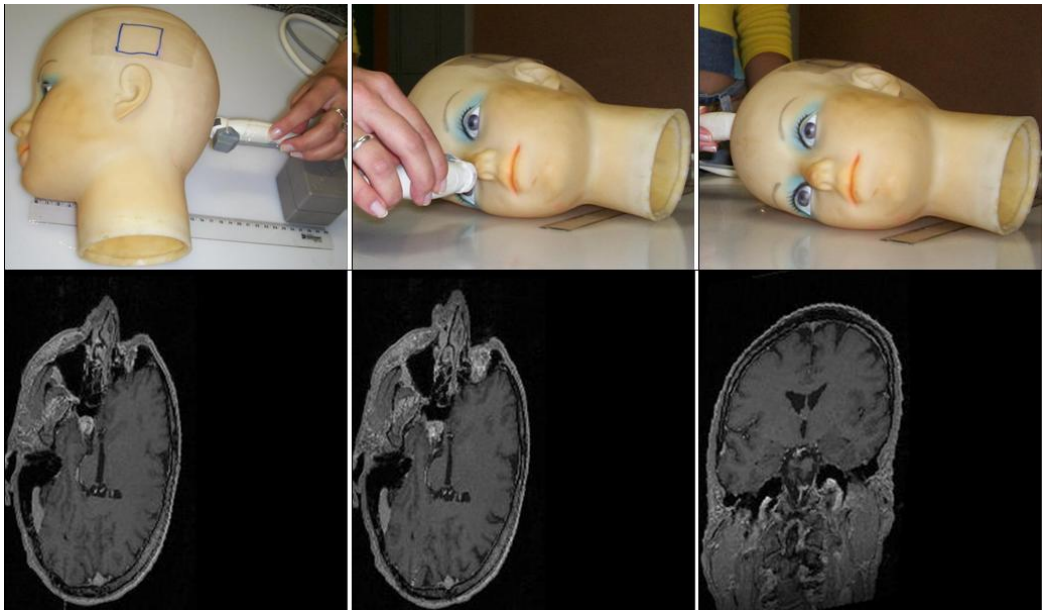
Neste momento, cabe ressaltar que já existem vários *softwares* que permitem visualização 3D e desfrutam de várias ferramentas de análise, como ANALYZE [Analyze 2007], MEDx [Medx 2007], e MNI [MNI 2007]. Porém, foi necessário para este trabalho, o desenvolvimento de um *software* em que se tivesse acesso ao código-fonte, para explorar todas as informações possíveis e também fazer modificações futuras, conforme as necessidades. Outro aspecto levado em consideração em não se utilizar *softwares* comerciais, foi o fato dos mesmos apresentarem um grande número de

ferramentas de análise de imagens, o que os tornam lentos. Além disso, estas ferramentas são desnecessárias para a elaboração deste trabalho. Por isso, o *softwar*e desenvolvido neste trabalho conta apenas com as ferramentas essenciais, para navegação e visualização do volume de IRM, de forma rápida e interativa.

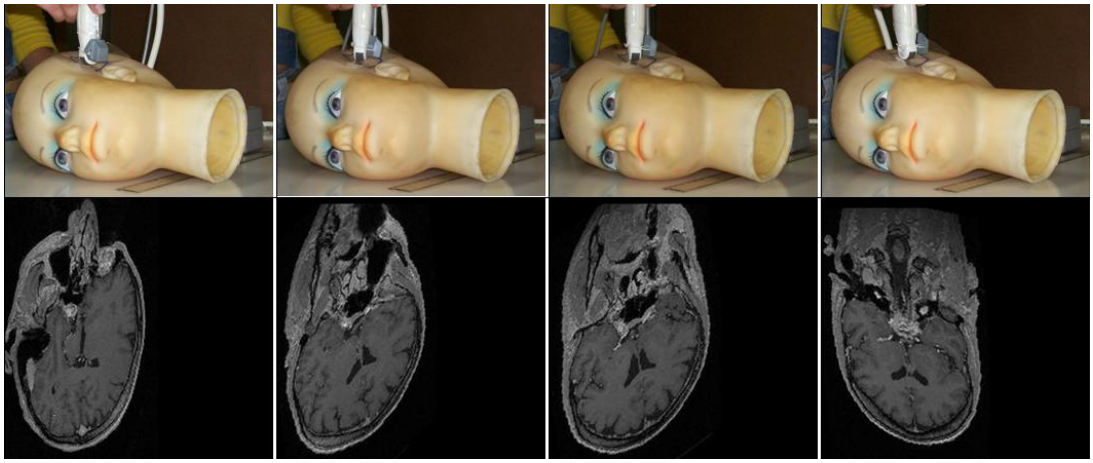
9.3 Orientação Cirúrgica

O programa para orientação cirúrgica desenvolvido, permite a extração de fatias do volume de IRM guiado pelas coordenadas do transdutor magnético 3D (x, y, z e os ângulos de Euler). A fatia extraída servirá para ser correlacionada com as IUS intra-operatórias em tempo real, futuramente.

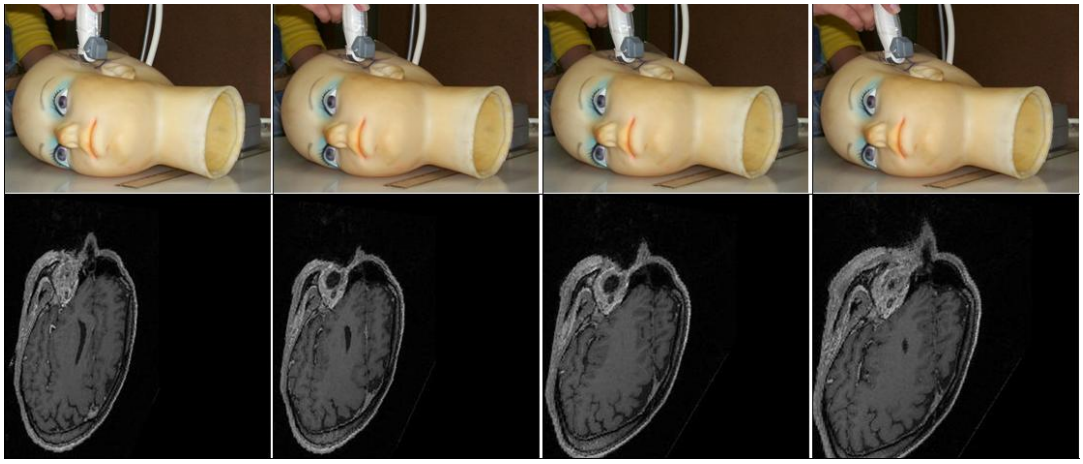
Para validar a implementação deste programa, foi proposto um experimento, explicado no capítulo 8. Os resultados obtidos podem ser vistos nas Figuras 9.10, 9.11, 9.12 e 9.13. A Figura 9.10 ilustra os pontos de referências espaciais na cabeça sendo adquirido com o transdutor 3D (*Polhemus*®) acoplado ao transdutor ultrassônico microconvexo. As Figuras 9.11, 9.12 e 9.13, mostram testes de varredura do transdutor ultrassônico sobre a janela cirúrgica, realizados nas direções dos eixos X, Y e Z, bem como as IRM extraídas do volume de imagens a partir da informação espacial 3D obtida com o transdutor *Polhemus*®. Os planos dessas IRM correspondem, ao plano do “campo de visão” do ultra-som. Lembrando que o volume de IRM é adquirido durante o planejamento pré-cirúrgico.



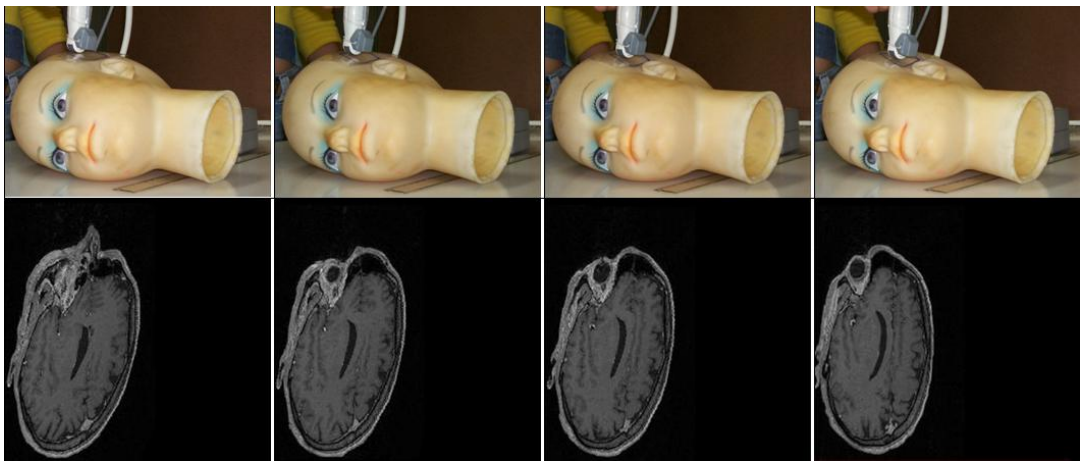
[Figura 9.10] Resultados obtidos com o programa de extração de fatias. As imagens na primeira linha mostram a localização do receptor do *Polhemus*® em alguns pontos de referência sobre o protótipo de cabeça, enquanto as imagens da linha inferior mostram as fatias de IRM correspondentes ao "campo de visão" do Ultra-Som.



[Figura 9.11] As imagens na primeira linha mostram o receptor do *Polhemus*® acoplado ao transdutor ultrassônico microconvexo e sendo rotacionado sobre a janela cirúrgica, em torno do eixo Z, enquanto as imagens da linha inferior mostram as fatias de IRM correspondentes ao "campo de visão" do Ultra-Som para diferentes posições em relação ao eixo Z.



[Figura 9.12] As imagens na primeira linha mostram o receptor do *Polhemus*® acoplado ao transdutor ultrassônico microconvexo e sendo rotacionado sobre a janela cirúrgica, em torno do eixo X, enquanto as imagens da linha inferior mostram as fatias de IRM correspondentes ao "campo de visão" do Ultra-Som para diferentes posições em relação ao eixo X.



[Figura 9.13] As imagens na primeira linha mostram o receptor do *Polhemus*® acoplado ao transdutor ultrassônico microconvexo e sendo transladado sobre a janela cirúrgica, na direção Y, enquanto as imagens da linha inferior mostram as fatias de IRM correspondentes ao "campo de visão" do Ultra-Som para diferentes posições Y.

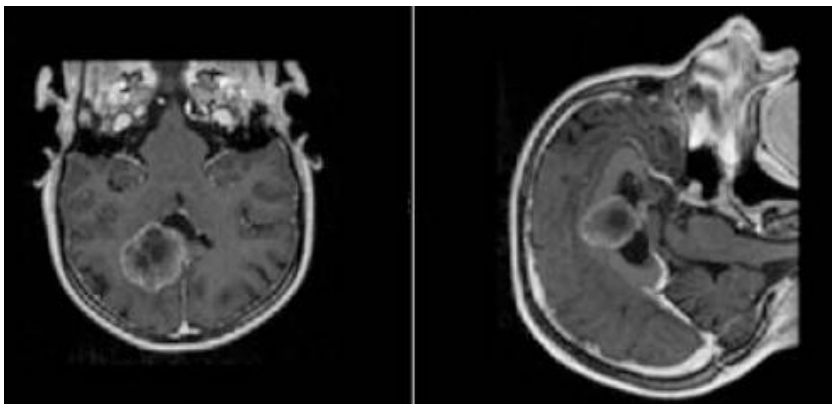
Observando os resultados mostrados nas Figuras 9.10, 9.11, 9.12 e 9.13, pode-se perceber que o programa selecionou a fatia de acordo com o corte que se estava

observando com o ultra-som (sagital, coronal, axial), ou seja, quando estava se visualizando um corte axial, por exemplo, a fatia extraída mostrou um corte axial.

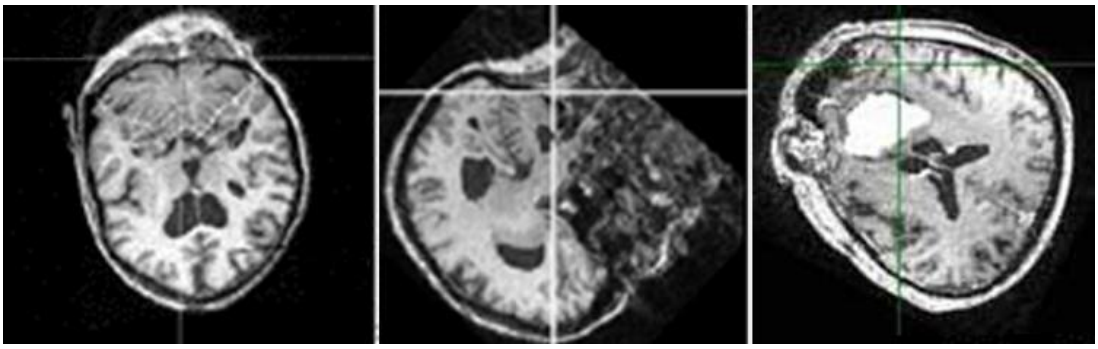
É importante salientar que as IRM não são do protótipo em análise, e por isso, não foram realizadas comparações de posições anatômicas específicas entre as imagens de ultra-som e as IRM. O que se pôde constatar, através dos resultados obtidos, é que a varredura do volume acompanha as variações de posição e angulação fornecidas pelo *Polhemus*®.

Como é possível observar na Figura 9.11, quando o receptor do *Polhemus*® se desloca em torno do eixo Z, as imagens do volume de IRM acompanham o movimento, e esta correspondência também é observada quando o receptor é deslocado em torno do eixo X (Figura 9.12) e transladado em Y (Figura 9.13). Os resultados mostraram-se satisfatórios, uma vez que forneceram fatias que acompanharam o movimento do receptor do transdutor de posição. Estes resultados serão melhorados, futuramente, com os procedimentos de calibração que serão realizados com o uso de um *phantom* com marcadores. Com esta calibração, será possível utilizar uma matriz de transformação específica para este procedimento, tornando as medidas mais precisas, e possibilitando comparações entre regiões anatômicas específicas.

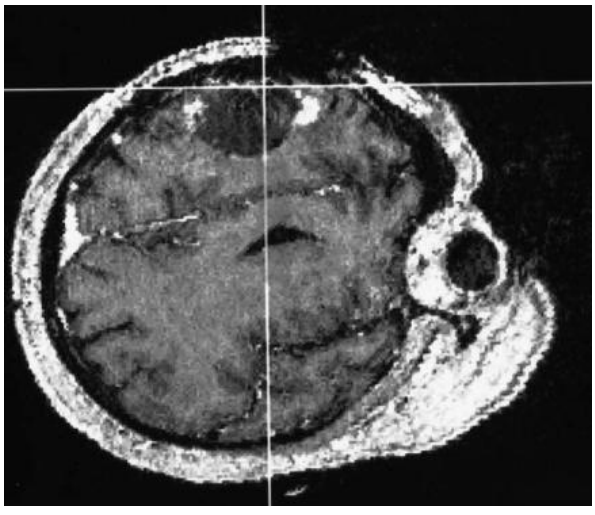
O principal objetivo em validar o programa de extração de fatias, foi provar seu funcionamento qualitativo e mostrar que a extração de fatias para posterior comparação com IUS intra-operatórias é semelhante a alguns dados encontrados na literatura, com é o caso das pesquisas de Comeau [Comeau *et al*. 2000; Comeau e Peters 1998] e Lindseth [Lindseth *et al*. 2003]. Para ilustrar os resultados encontrados na literatura, pode-se observar as Figuras 9.14, 9.15 e 9.16, que mostram fatias oblíquas extraídas do volume de IRM pré-operatório obtidas com a utilização de um transdutor de posição óptico.



[Figura 9.14] IRM oblíquas obtidas do volume de IRM (Figura retirada de [Lindseth *et al*. 2003]).



[Figura 9.15] Extração de IRM oblíquas do volume de IRM (Figura retirada de [Comeau e Peters 1998]).



[Figura 9.16] Alteração da IRM para comparação com IUS intra-operatória (Figura retirada de [Comeau *et al*. 2000]).

Nas imagens mostradas nas Figuras 9.14, 9.15 e 9.16, é possível observar os cortes oblíquos de IRM e que os mesmos se encontram alterados após a aplicação de uma transformação geométrica, obtida com o uso de um transdutor de posição óptico *Polaris*®. Esta transformação foi usada para extrair os cortes oblíquos de IRM, para sua possível visualização e comparação simultânea com IUS intra-operatória. As alterações angulares e deslocamentos, encontradas nestas imagens, também podem ser encontradas nas Figuras 9.10, 9.11, 9.12 e 9.13, obtidas com o programa implementado neste trabalho. Estas alterações devem-se ao fato de que o volume deve ser orientado para posterior extração de cortes oblíquos, com a finalidade de representar o corte do volume que o neurocirurgião está observando.

A descrição do sistema proposto neste trabalho, contará com a utilização de marcadores para correlacionar as IRM e as IUS. Na literatura são encontrados dois tipos de marcadores, que podem ser internos ou externos. Os marcadores internos incluem marcas anatômicas, intensidade de *pixels* e características geométricas (superfície da pele ou osso, ventrículos, nervos, etc). Os marcadores externos são estruturas esterotáticas usadas durante a aquisição das imagens pré-operatórias e durante a cirurgia. Em [Mountz *et al*. 1994] os autores apresentaram um par de óculos para ser colocado no paciente. Os óculos se encaixam ao canal auditivo e possuem um sistema de ajuste, que pode ser travado quando se obtém a posição correta de posicionamento. Assim, o paciente pode se deslocar com a armação sem alterar sua posição ajustada previamente.

Neste tipo de técnica, tendo-se o cuidado inicial com o posicionamento dos marcadores, aumenta-se a velocidade do cálculo das transformações geométricas, facilitando sua automatização e aumentando-se a velocidade de registro das modalidades de imagens em questão.

Baseado nos testes que serão realizados com o *phantom*, que está sendo desenvolvido, será possível escolher o melhor tipo de marcador para o sistema proposto.

O *software* foi desenvolvido para inicialmente carregar o volume de IRM, no início da cirurgia. Uma vez que este volume é carregado, são realizadas transformações, com base nos valores das coordenadas e dos ângulos de Euler adquiridos pelo transdutor de posição 3D que estará acoplado ao transdutor de ultra-som micro-convexo. Estes dados provenientes do *Polhemus*® são lidos pelo *software*, através do uso de uma biblioteca feita para aquisição dos dados deste transdutor.

A melhor transformação a ser aplicada no volume de IRM, é adquirida usando os marcadores. Estes marcadores deverão ser visualizados sobre o paciente na sala cirúrgica e nas IRM pré-operatórias. A partir da identificação de marcadores homólogos nestes dois espaços, é possível encontrar a melhor transformação de correlação entre estes. As transformações realizadas no volume, durante o procedimento cirúrgico, visam orientar o volume de IRM para obter fatias oblíquas deste. Uma vez adquirida estas fatias, estas poderão ser comparadas com as IUS intra-operatórias.

Capítulo 10
Conclusão

O sucesso da cirurgia de epilepsia depende da precisão nos procedimentos pré-cirúrgicos e da remoção completa da lesão, por isso é fundamental que a distribuição espacial da lesão seja conhecida antes e depois da craniotomia. Visando isto, foram implementados programas que pretendem ajudar no planejamento e orientação cirúrgica.

Neste trabalho, foi apresentada uma aplicação interativa para visualização volumétrica e extração de fatias de um volume de imagens, com a finalidade de auxiliar o neurocirurgião. Através do uso de diferentes técnicas de visualização volumétrica e, da interação com os dados volumétricos é possível realizar-se um planejamento cirúrgico.

A arquitetura do sistema desenvolvido possibilita a adição de novas classes do VTK, permitindo que novas funcionalidades sejam rapidamente integradas. Como todas as classes básicas e fundamentais já foram implementadas, agora passará a ser desenvolvida a adaptação de novas ferramentas necessárias para o melhoramento das funcionalidades, já existentes.

Com os programas desenvolvidos neste trabalho, tem-se toda a base do *software* de neuronavegação, bastando somente adaptar os resultados provenientes de procedimentos de calibração. Esta adaptação poderá ser feita com poucas linhas de comando, conforme apresentada no anexo A. Além disso, é importante ressaltar que o *software* desenvolvido pode ser usado com outros transdutores de posição, além do transdutor de posição magnético escolhido.

A utilização do *Polhemus*® como transdutor de posição na construção de um sistema de neuronavegação apresenta as seguintes vantagens em relação aos sistemas comerciais existentes:

- Baixo custo - O preço para o desenvolvimento do sistema proposto neste trabalho é em torno de R$ 15.000,00, visto que o transdutor de posição custa R$ 5.000,00 e um computador de alta performance, em média, R$ 10.000,00. Este valor é baixo, quando comparado, por exemplo, com o sistema comercial de neuronavegação *BrainLAB VectorVision*®, que possui um custo em torno de R$ 150.000,00.
- Melhor desempenho e funcionamento na sala cirúrgica - A sala cirúrgica tem um espaço limitado, visto que conta com uma equipe de médicos, enfermeiros e uma série de equipamentos presentes durante o procedimento cirúrgico. Levando este fato em consideração, percebe-se que o uso do transdutor ótico torna-se um problema, pois a câmera deste transdutor deve estar permanentemente enxergando os marcadores aderidos à ferramenta cirúrgica que está sendo orientada. Neste aspecto, o *Polhemus*® apresenta outra vantagem, já que é compacto e seu campo de visão não é distorcido pela aglomeração de pessoas e equipamentos. Além disso, testes realizados em ambiente cirúrgico mostraram que o *Polhemus*® não sofreu qualquer tipo de interferência magnética, visto que as ferramentas cirúrgicas são de aço inox, um material não-magnético.

Um outro aspecto importante deste trabalho é a sua contribuição para as linhas de pesquisa do Grupo de Inovação em Instrumentação Médica e Ultra-Som (GIIMUS). Um exemplo de sua contribuição é o apoio na geração de imagens elastográfica 3D que

serão obtidas à mão livre com um equipamento de ultra-som 2D, e alguns outros projetos, que futuramente, venham envolver realidade virtual. Além disso, este *software* também pode ser utilizado para outros tipos de cirurgia, bastando apenas realizar os procedimentos corretos de calibração.

Através do uso da metodologia empregada no desenvolvimento deste trabalho, principalmente da biblioteca gráfica VTK, puderam-se obter resultados satisfatórios. O algoritmo desenvolvido para o planejamento pré-cirúrgico gerou uma alta resolução na visualização dos cortes ortogonais e oblíquos de IRM, além se ser um programa rápido e interativo. Quanto ao programa de orientação cirúrgica, pode-se perceber que o mesmo aplicou transformações ao volume de IRM com base nos valores das coordenadas e dos ângulos de Euler adquiridos pelo transdutor de posição 3D.

De acordo, com a proposta inicial do trabalho, pode-se perceber que as metas destinadas a este, foram cumpridas, ou seja, a parte inicial de um trabalho maior já está implementada, bastando agora algumas alterações para que futuramente se consiga a implementação de um sistema de neuronavegação completo em tempo real.

10.1 Trabalhos Futuros

Nesta seção são apresentadas algumas sugestões para a continuidade deste trabalho. Tais sugestões provêm da experiência adquirida no desenvolvimento do sistema, bem como das dificuldades encontradas.

- Adaptar os dados provenientes do procedimento de calibração ao *software* de extração de fatias. Este procedimento de calibração se dará através do uso de um *phantom* com marcadores;
- Correlacionar as fatias extraídas do volume de IRM pré-operatórias com IUS intra-operatórias;

- Fazer o registro das IRM com IUS para corrigir o deslocamento do cérebro após a craniotomia;
- Adaptar uma interface gráfica ao *software*, de modo a facilitar seu uso, durante o procedimento cirúrgico.

Referências Bibliográficas

[Adams *et al*. 1996] Adams, L., Knepper, A., Meyer-Ebrecht, D., Ruger, R., van der Brug, W. *An optical navigator for brain surgery*. Computer, (29), pp. 48—54, (1996).

[Amira 2006] Amira home page. Disponível em: <http://www.amiravis.com>. Acessado em 07 de abril de 2006.

[Analyze 2007] Analyze home page. Disponível em: http://www.mayo.edu/bir/Software/Analyze/Analyze1.html. Acesso em: 1 de Março de 2007

[Ashburner e Friston 1999] Ashburner, J., Friston, K.J. *Nonlinear spatial normalization using basis functions*. Human Brain Mapping, 4, pp. 254—266 (1999).

[Avila *et al*. 2004] Avila, L.S., Barré, S., Blue, R., Geveci, B., Henderson, A., Hoffman, W.A., King, B., Law, C.C., Martin, K.M., Schroeder, W.J. *The VTK user's guide*. 3ª ed. Kitware, Inc, 2004. 332p.

[Barkovich *et al*. 1995] Barkovich, A.J., Rowlwy, H.A., Andermann, F. *MR in partial epilepsy: value of high-resolution volumetric techniques*. AJNR, 16 (2), pp. 339—343 (1995).

[Binotto 2003] Binotto, A.P.D. *Visualização em tempo real de dados volumétricos dinâmicos usando hardware gráfico*. Dissertação (mestrado). Universidade Federal do Rio Grande do Sul, UFRGS (2003).

[BrainLab 2007] BrainLab Home Page. Disponível em: <http://www.brain-surgery.com/brainlab1.gif>. Acesso em 05 de março de 2007.

[Bruton 1988] Bruton, C. J. *The neuropathology of temporal lobe epilepsy*. New York. Oxford University Press, 1988. 176p.

[Butler *et al*. 1998] Butler, W.E., Piaggio, M., Constantinou, C., NiklasonL., Gonzales, R.G., Cosgrove, G.R., Zervas N.T. *A mobile computed tomographic scanner with intraoperative and intensive care unit applications.* Neurosurgery, 42 (6), pp. 1304—1310 (1998).

[Cendes *et al*, 1993a] Cendes, F., Andermann, F., Dubeau, F., Gloor, P., Evans, A., Jones-Gotman, M., A. Olivier, Andermann, E., Robitaille, Y., Lopes-Cendes, I., Peters, T., Melanson, D. *Early childhood prolonged febrile convulsions, atrophy and sclerosis of mesial structures, and temporal lobe epilepsy*. Neurology, 43 (6), pp. 1083-1087 (1993).

[Cendes et al. 1993b] Cendes, F., Andermann, F., Gloor, P., Evans, A., Gotman, M. J., Watson, C., Melanson, D., Olivier, A., Peters, T., Lopes-Cendes, I., Leroux, G. *MRI volumetric measurement of amygdala and hippocampus in temporal lobe epilepsy*. Neurology, 43 (4), pp. 719—725 (1993).

[Cendes 2005] Cendes, F. *Mesial Temporal Lobe Epilepsy Syndrome: An Updated Overview*. J. Epilepsy Clin Neurophysiol., 11(3), pp. 141—144 (2005).

[Comeau *et al*. 1998] Comeau, R.M., Fenster, A., Peters, T.M. *Intraoperative US in interactive image-guided neurosurgery*. Radiographics, 18, pp. 1019—1027 (1998).

[Comeau e Peters 1998] Comeau, R.M e Peters, T. *Intraoperative Ultrasound Imaging in Interactive Image Guided Neurosurgery*. Tech. Rep. Disponível em: < http://www.bic.mni.mcgill.ca/ research/groups/igns/US/us_home.html>.

[Comeau *et al*. 2000] Comeau, R.M., Sadikot, A.F., Fenster, A., Peters, T.M. *Intraoperative ultrasound for guidance and tissue shift correction in image-guided neurosurgery*. Medical Physics, 27 (4), pp. 787—800 (2000).

[Dcc 2007] Dcc Home Page. Disponível em: <http://www.dcc.unicamp.br/~ cpg/materialdidaticomo815/ 9802/curso/node26>. Acesso em 07 de março de 2007.

[Drebin *et al*. 1988] Drebin, R.A., Carpenter, L., Hanrahan, P. *Volume Rendering*. Computer Graphics (Proc. SIGGRAPH), 22(4), pp. 65—74 (1988).

[Earnshaw e Watson 1993] Earnshaw, R.A., Watson, D. *Animation and scientific visualization*. London: Academic, 1993. 285p.

[Gobbi *et al*. 1999] Gobbi, D., Comeau, R.M., Peters, T. *Ultrasound probe tracking for real-time ultrasound/MRI overlay and visualization of brain shift*. Medical Image Computing and Computer Assisted Intervention -MICCAI'99. C. Taylor and A. Cholchester, eds. Lecture Notes in Computer Science, 1679, pp. 920—927 (1999).

[Gobbi *et al*.2000] Gobbi, D., Comeau, R.M., Peters, T. *Ultrasound/MRI Overlay with Image Warping for Neurosurgery*. Medical Image Computing and Computer Assisted Intervention – MICCAI 2000. Lecture Notes in Computer Science, 1935, pp. 106—114 (2000).

[Gonzales *et al*. 2004] Gonzales, R.C., Woods, R.E., Steven, L.E. *Digital image processing using matlab*. Person prentice Hall, New Jersey, 2004. 609p.

[Guerreiro e Guerreiro 1996] Guerreiro, C.A.M., Guerreiro, M.M. *Epilepsia*. 2ª edição. São Paulo. Ed. Lemos Editorial, 1996. 477p.

[Holodraw 2006] Holodraw Home Page. Disponível em: <http://simkin.asu.edu/holodraw> Acesso em 10 de abril de 2006.

[Jack 1990] Jack, C.R. *MRI-based hippocampal volume measurements in epilepsy*. Epilepsia, 35(Suppl 6), pp. S21—29 (1990).

[Janszky *et al*. 2003] Janszky, J., Schulz ,R., Ebner, A. *Clinical features and surgical outcome of medial temporal lobe epilepsy with a history of complex febrile convulsions*. Epilepsy Research, 55 pp. 1—8 (2003).

[John e Mccloy 2004] John, N.W., Mccloy, R.F. *Navigation and visualization three-dimensional data sets*. The British Journal of radiology, 77, pp. 108—113 (2004).

[Kgu 2007] Kgu home page. Disponível em: <http://www.kgu.de/bic/ images/Equipment/EEG_32ch_02.jpg>. Acesso em: 4 de março de 2007.

[Kitchen *et al*. 1993] Kitchen, N.D., Lemieuex, L., Thomas, D.G.T. *Accuracy in frame based and frameless stereotaxy.* Stereotactic and Functional Neurosurgery, 61, pp. 195—206 (1993).

[Kitware 2006] VTK home page. Disponível em: <http://www.vtk.org>. Acesso em: 1 de junho de 2006.

[Kuzniecky *et al*. 1987] Kuzniecky, R., de la Sayette, V., Ethier, R., Melsnson, D., Andermann, F., Berkovic, S., Robitaille, Y., Olivier, A., Peters, T., Feindel, W. *Magnetic resonance imaging in temporal lobe epilepsy: pathological correlations*. Ann Neurol., 22(3), pp. 341—347 (1987).

[Lay 1997] Lay, D.C. *Álgebra Linear e suas aplicações*. 2ª edição. Rio de Janeiro. Editora Livros Técnicos e Científicos S.A., 1997. 504p.

[Levoy 1988] Levoy, M. *Display of Surfaces from Volume Data*. IEEE Computer Graphics and Application, 8, pp. 29—37 (1988).

[Levoy 1990] Levoy, M. *Efficient ray tracing of volume data*. ACM Transactions on Graphics, 9(3), pp. 245—261 (1990).

[Lichtenbelt *et al*. 1998] Lichtenbelt, B., Crane, R., Naqvi, S. Introduction to volume rendering. Hewlett-Packard Professinal Books. Prentice Hall PTR, 1998.236 p.

[Ligier *et al*. 1994] Ligier, Y., Ratib, O., Logean, M., Girard, C. *Osiris: a medical image-manipulation system*. MD Comput., 11(4), pp. 212—218 (1994).

[Lindseth *et al*. 2003] Lindseth, F., Kaspersen, J.H., Ommedal, S., Lango, T, Bang, J., Hokland, J., Unsgaard, G., Hernes, T.A. *Multimodal image fusion in ultrasound-based neuronavigation: improving overview and interpretation by integrating preoperative MRI with intraoperative 3D ultrasound.* Computer Aided Surgery., 8(2), pp. 49-69 (2003).

[Lorensen e Cline 1987] Lorensen, W.E., Cline, H.E. *Marching Cubes: A high resolution 3D surface construction algorithm*. ACM SIGGRAPH Computer Graphics, 21, pp. 163—169 (1987).

[Manssour 1998] Manssour, I.H. *Visualização colaborativa de dados científicos com ênfase na área médica*. Porto Alegre. Exame de Qualificação (EQ-23) - Instituto de Informática, Universidade Federal do Rio Grande do Sul (1998).

[Manssour e Freitas 2002] Manssour, I., Freitas, C. *Visualização Volumétrica*. Revista de Informática Teórica e Aplicada, 9(2), pp. 97—126 (2002).

[Matlab 2006] Matlab Home Page. Disponível em: <http://www.mathworks.com/>. Acesso em 10 de abril de 2006.

[Medx 2007] Medx home page. Disponível em:< http://medx.sensor.com/products/medx/index.html>. Acesso em: 1 de Março de 2007.

[MNI 2007] MNI home page. Disponível em: <http://www.bic.mni.mcgill.ca/software/>. Acesso em: 1 de Março de 2007.

[Mountz *et al*. 1994] Mountz, J.M., Zhang, B., Liu H.G., Inampudi, C. *A reference method for correlation of anatomic and functional brain images: validation and clinical application.* Seminars in nuclear medicine, 24(4), pp. 256—271 (1994).

[Ndigital 2007a] Ndigital Home Page. Disponível em: <http://www.ndigital.com/medical/technology-optical.php> Acesso em: 20 de Fevereiro de 2007.

[Ndigital 2007b] Ndigital Home Page. Disponível em: <http://www.ndigital.com/medical/documents/ComputerAssistedTherapyandNDITechnology.pdf>. Acesso em: 20 de Fevereiro de 2007.

[Nema 2007] NATIONAL ELECTRICAL MANUFACTURERS ASSOCIATION. *Digital Imaging and Communications in Medicine (DICOM).* Disponível em: <http://medical.nema.org> Acesso em: 6 de março de 2007.

[Pagoulatos *et al*. 1999] Pagoulatos, N., Edwards, W.S., Haynor, R.D., Kim, Y. *Interactive 3-D registration of ultrasound and magnetic resonance images based on a magnetic position sensor*. IEEE Trans. On Information Technology in Biomedicine, 3, pp. 278—288 (1999).

[Papademetris 2006] Papademetris, X. *An introduction to programming for medical image analysis with the visualization toolkit*. Disponível em: http://www.bioimagesuite.org/vtkbook/xpvtkbook.pdf>.

Acessado em 10 de janeiro de 2007.

[Paraview 2006] Paraview Home Page. Disponível em: <http://www.paraview.org/HTML/Index.html>. Acesso em 10 de abril de 2006.

[Peters 2006] Peters, T. M. *Image-guidance for surgical procedures*. Physics in Medicine and Biology, 51, pp. 505—540 (2006).

[Polaris 2007] Polaris Home Page. Disponível em: <http://www.ndigital.com/medical/polarisfamily>. Acesso em: 05 de março de 2007.

[Polhemus 2004] Polhemus Incorporated, Colchester, Vermont E.U.A.; *Manual 3D Space® Isotrak II®*, 2004.

[Roqueiro 2007] Página eletrônica do professor Nestor Roqueiro da UFSC. Disponível em: <http://www.das.ufsc.br/~nestor/cursos/matlab /matlab/matlab.html>. Acessado em: 10 de Janeiro de 2007.

[Rueckert *et al*. 1999] Rueckert, D., Sonoda, L.I., Hayes, C., Hill, D.L., Leach, M.O., Hawkes, D.J. *Nonrigid registration using free-form deformations: application to breast MR images*. IEEE Trans Med Imaging, 18 (8), pp. 712—721 (1999).

[Russ 2002] Russ, J.C. *The Image Processing Handbook*. 4ª edição. Boca Raton, Fla. CRC Press, 2002. 732p.

[Scilab 2006] Scilab Home Page. Disponível em: <http://www.scilab.org/>. Acesso em 10 de abril de 2006.

[Schnabel *et al*. 2001] Schnabel, J.A., Tanner, C., Smith, A.D.C., Leach, M.O., Hayes, C., Degenhard, A., Hose, R., Hill, D.L., Hawkes, D.J. *Validation of Non-rigid Registration Using Finite Element Methods*. Lecture Notes in Computer Science, 2082, pp. 344—357 (2001).

[Schoroeder *et al*. 2004] Schoeder, W., Martin, K., Lorensen, B. *The Visualization Toolkit: an object-oriented approach to 3D graphics*. 3ª ed. Kitware, Inc, 2004. 474p.

[Shuxiang *et al*. 2003] Schuxinag, D., Feiming, B., Jie-Fang, L., Dwight, V. *An acoustic position sensor*. Review of Scientific Instruments, 74 (11), pp. 4863—4868 (2003).

[Simon 1997] Simon, D. *Intra-Operative Position Sensing and Tracking Devices*. Proceedings of the First Joint CVRMed / MRCAS Conference, pp. 62—64 (1997).

[Smith e Ranallo 1989] Smith, Hans-J., Ranallo,F.N. *A non-mathematical apprroach to basic MRI. Madison*. 1ª edição. Medical Physics Publishing Corporation.1989. 203p.

[Sobotta 1995] Sobotta, J. *Sobotta Atlas de Anatomia Humana*. 20ª edição. Rio de Janeiro. Editora Guanabara Koogan S.A. 1995. 401p.

[Spotfire 2006] Spotfire Home Page. Disponível em: <http://www.spotfire.com/>. Acesso em 10 de abril de 2006

[Stelter *et al*. 2006] Stelter, K., Andratschke, M., Leuning, A., Hagedorn, H. Computer-*assisted surgery of the paranasal sinuses: technical and clinical experience with 368 patients, using the Vector Vision Compact® system*. The Journal of Laryngology & Otology, 120, pp. 1026-1032 (2006).

[Traina e Oliveira 2006] Traina, A., Oliveira, M. *Apostila de Computação Gráfica* (2006). Disponível em: <http://gbdi.icmc.usp.br/documentacao/apostilas/cg/downloads/apostilas.pdf> Acesso em: 5 de abril de 2006.

[Tronnier *et al*. 1997] Tronnier, V. M., Wirtz, C.R., Knauth, M., Lenz, G., Pastyr, O., Bonsanto, M., Albert, F.K., F.K., Kuth, R., Staubert A., Schlegel W., sartor, K. *Intraoperative diagnostic and interventional magnetic resonance imaging in neurosurgery*.

Neurosurgery, 40, pp. 891—900 (1997).

[Ualg 2007] Ualg Home Page. Disponível em: <http://w3.ualg.pt/~cmsilva/documentos/RMN_Mestrado.pdf>. Acesso em: 10 de março de 2007.

[Upson e Keeler 1988] Upson, C., Keeler, M. *V-Buffer: Visible Volume Rendering*. SIGGRAPH Computer Graphics, New York, 22(4), pp. 59—64 (1988).

[Varandas *et al*. 2004] Varandas, J., Baptista, P., Santos, J., Martins. R., Dias, J. *VOLUS - A visualization system for 3D ultrasound data*. Ultrasonics, 42, pp. 689—694 (2004).

[Vidal *et al*. 2006] Vidal, F.P., Bello, F., Brodlie, K.W., John, N.W., Gould, D., Phillips, R., Avis, N.J. *Principles and applications of computer graphics in medicine*. Computer Graphics Forum, 25, pp. 113—137 (2006).

[Visad 2006] VisAD Home Page. Disponível em: <http://www.ssec.wisc.edu/~billh/visad.html>. Acesso em: 5 de abril de 2006.

[VMD 2006] VMD Home Page. Disponível em: <http://www.ks.uiuc.edu/Research/vmd/>. Acesso em 10 de abril de 2006.

[Volview 2006] Volview Home Page. Disponível em: <http://www.kitware.com/products/volview.html>. Acesso em 07 de abril de 2006.

[Vrealities 2007] Vrealities Home Page. Disponível em: <http://www.vrealities.com/logitech.html>. Acesso em: 04 de março de 2007.

[Webb *et al*. 1999] Webb, J., Guimond, A., Eldridge, P., Chadwik, D., Meunier, J., Thirion, J.P., Roberts, N. *Automatic detection of*

hippocampal atrophy on magnetic resonance images. Magnetic Resonance Imaging., 17(8), pp. 1149—1161 (1999).

[Westover 1990] Westover, L. *Footprint Evaluation for Volume Rendering*. ACM SIGGRAPH Computer Graphics, 24(4) pp. 367—376 (1990).

[Woods *et al*. 1998] Woods, R.P., Grafton, S.T., Watson, J.D.G., Sicotte, N.L., Mazziotta, J.C. *Automated Image Registration: II. Intersubject validation of linear and nonlinear models*. Journal of Computer Assisted Tomography, 22, pp. 153—165 (1998).

Apêndice A

Modificações do *software* após calibração

O programa foi desenvolvido com a intenção de que poucas linhas de seu código-fonte sejam alteradas, após o procedimento de calibração, adquirido com o *phantom* dedicado. A alteração está relacionada com a matriz transformação aplicada ao volume de IRM.

Neste trabalho, para validar o funcionamento do software, em extrair fatias de um volume, foi usada uma transformação de corpo rígido genérica. Utilizando esta matriz, foram realizados testes qualitativos, apenas com o propósito de adquirir fatias de um volume, usando dados provenientes de um transdutor de posição.

Para adquirir a melhor transformação que correlaciona o espaço do paciente com as IRM pré-operatórias, é necessário identificar três marcadores não-colineares. Uma vez que estes marcadores sejam identificados, é possível, utilizando a técnica de mínimos quadrados, determinar a transformação entre os dois conjuntos de dados adquiridos. É importante ressaltar que já existem algoritmos implementados com VTK, que permitem varrer os pontos homólogos nos dois espaços e encontrar a melhor transformação que os alinhe.

A Figura A1 mostra a parte do código-fonte que será alterada após o procedimento de calibração. Nesta figura, as variáveis thx, thy e thz estão relacionadas com as rotações, e dx1, dy1 e dz1 com as translações. O procedimento de modificar esta matriz é simples, bastando substituir seus elementos pelos dados oriundos da calibração, como pode ser observado na Figura A1.

```
transform=vtk.vtkMatrix4x4()
transform.SetElement(0,0,cos(thx)*cos(thy))
transform.SetElement(0,1,cos(thx)*sin(thz)*sin(thy)-sin(thx)*cos(thz))
transform.SetElement(0,2,cos(thx)*sin(thy)*cos(thz)+sin(thx)*sin(thz))
transform.SetElement(0,3,dx1)
transform.SetElement(1,0,sin(thx)*cos(thz))
transform.SetElement(1,1,sin(thx)*sin(thy)*sin(thz)+cos(thx)*cos(thz))
transform.SetElement(1,2,sin(thx)*sin(thy)*cos(thz)-cos(thx)*sin(thz))
transform.SetElement(1,3,dy1)
transform.SetElement(2,0,-sin(thy))
transform.SetElement(2,1,cos(thx)*sin(thy)*sin(thz))
transform.SetElement(2,2,cos(thx)*sin(thy)*cos(thz))
transform.SetElement(2,3,dz1)
transform.SetElement(3,0,0)
transform.SetElement(3,1,0)
transform.SetElement(3,2,0)
transform.SetElement(3,3,1)
```

[Figura A1]: Código-fonte mostrando o modo de inserir a matriz transformação no *software.*

Outra modificação pertinente, é o ponto que será escolhido como origem para as transformações. A princípio está origem está no centro do volume das IRM, após os procedimentos de calibração, a origem passará a ser o ponto inicial do *Polhemus*® sobre a cabeça do paciente. Isto será feito transladando o centro do volume para este ponto.