ImageLab: Um sistema Multi-Orientado na Visualização e Processamento de Imagens Médicas

Paulo Gustavo Portella Ziemer
Marcelo Collares
Eduardo Camargo
Igor Castellani de Freitas
Pablo Javier Blanco
Rául Antonino Feijóo
ziemer@lncc.br
collares@lncc.br
camargo@lncc.br
igor@lncc.br
pablo@lncc.br
feij@lncc.br
HeMoLab - Laboratório de Modelagem em Hemodinâmica
Laboratório Nacional de Computação Científica,
Av. Getúlio Vargas, 25651-075, Petrópolis, RJ, Brasil
Rodrigo Luis de Souza Silva
rodrigoluis@ice.ufjf.br
Universidade Federal de Juiz de Fora,
Rua José Lourenço Kelmer, 36036-330, Juiz de Fora, MG, Brasil

Resumo. *O ImageLab é um sistema projetado para atuar como laboratório na implementação de novos algoritmos e métodos na área de processamento e visualização de imagens médicas. Embora o caráter de pesquisa possua uma maior significância no uso da ferramenta, esta pode também ser utilizada diretamente pelo profissional Médico, uma vez que são oferecidas funcionalidades específicas utilizadas na análise e diagnóstico, focando atualmente em imagens cardiovasculares. Outro aspecto do software a ser destacado é a possibilidade de permitir a obtenção da geometria de modelos de simulação (interação com a ferramenta de modelagem e simulação HeMoLab). Com relação a implementação, o sistema utiliza bibliotecas de código aberto (ITK, VTK e Qt) e conta com arquitetura modular que visa facilitar a inclusão de novos filtros e métodos de processamento de imagem. Inicialmente, foram incorporados ao sistema filtros de processamento de imagem tradicionais e métodos de visualização 3D como geração de superfícies e Volume Rendering, entre outras. Técnicas recentes de processamento de imagens, como a segmentação através da derivada topológica, as quais estão ainda em fase de pesquisa, foram incorporadas ao software. Neste trabalho é descrita a estrutura do ImageLab, principais funcionalidades, aspectos relativos à arquitetura de software e, ao final, é mostrado uma aplicação de potencial interesse na área médica.*

Keywords: *Processamento de Imagens, Imagens Médicas, arquitetura de software*

1. Introdução

Um dos grandes avanços da medicina foi a utilização de equipamentos e exames capazes de fornecer imagens detalhadas das estruturas que compõem o corpo humano de forma invasiva e não-invasiva. Exames de Tomografia Computadorizada (TC), Ressonância Magnética, Ultra-som, Angiografia e etc. podem ser destacados por serem extremamente utilizados pela comunidade médica. *Softwares* de imagens auxiliam o Médico no diagnóstico e outras etapas de pesquisa e tratamento. Através de técnicas de processamento de imagem doenças podem ser detectadas em seu estágio inicial, patologias podem ser melhor estudadas e caracterizadas, procedimentos cirúrgicos podem serem detalhados com maior precisão e etc.

De forma adicional, o manuseio de imagens médicas constitui uma parte fundamental da pesquisa em modelagem e simulação computacional de sistemas fisiológicos complexos, já que muitas das informações usadas na criação de modelos devem ser inferidas a partir de imagens médicas obtidas de pacientes específicos (Oshima, 2004). O caso mais exemplar é o da reconstrução de geometrias de vasos sanguíneos para realizar a modelagem do escoamento sanguíneo na região de interesse (Avolio, 1980; Urquiza et al., 2006; Stergiopulos et al., 1992).

Devido à adoção do formato DICOM (*Digital Imaging and Communications in Medicine*) (Standard, 2009; Pianykh, 2008) como padrão na qual as imagens médicas são armazenadas, transmitidas, visualizadas e etc., foi possível a profusão de programas e bibliotecas específicas para a manipulação de imagens DICOM. Na manipulação dessas imagens, médicos e pesquisadores usualmente utilizam *softwares* desenvolvidos por empresas fornecedoras das máquinas de aquisição de imagens. Entretanto, para a pesquisa médica, isso dificulta a inclusão de novas funcionalidades, algoritmos e métodos numéricos, muitos dos quais são essenciais na obtenção de dados para alimentar modelos computacionais. Um claro exemplo disto é o desenvolvimento de técnicas de modelagem baseadas em imagens para inferir propriedades físicas dos tecidos via a resolução de um problema inverso. Sendo assim, e com a finalidade principal de auxiliar a pesquisa nas áreas da medicina e da modelagem computacional através da inclusão de funcionalidades customizadas pelo pesquisador, está em desenvolvimento o *software* de processamento de imagens médicas *ImageLab*. Este sistema possui diversos filtros de processamento de imagens e módulos de reconstrução e visualização 3D, constituindo-se no ponto de partida para o desenvolvimento de uma ferramenta com as características salientadas acima. Nele são utilizadas bibliotecas de código aberto como o ITK (ITK - Insight Toolkit, 2009), que é responsável pelos algoritmos de processamento dos filtros de imagens, o VTK (VTK - Visualization Toolkit, 2009) que é utilizado na representação gráfica das imagens 2D e 3D e a *interface* gráfica do sistema foi desenvolvida em Qt (Blanchette and Summerfield, 2008).

Este artigo apresenta as principais características e diferenciais, bem como as perspectivas de progresso do *software* ImageLab que está em desenvolvimento no projeto de Modelagem e Simulação Computacional do Sistema Cardiovascular Humano conduzido pelo grupo de pesquisa e desenvolvimento PhySys (HeMoLab) do LNCC. Por sua vez, estas atividades encontram-se inseridas dentro do contexto do Instituto Nacional de Ciência e Tecnologia em Medicina Assistida por Computação Científica (Projeto MACC, 2009).

2. Bibliotecas utilizadas

O ImageLab está sendo desenvolvido na linguagem C++ e utiliza três componentes principais em sua arquitetura: ITK, VTK e Qt. O ITK é uma biblioteca de código aberto, desenvolvida em C++, que surgiu como suporte para o projeto *The Visible Human Project* (Ackerman, 1995). Ela disponibiliza diversos filtros de Segmentação e Registro de imagens, sendo a segmentação o processo de identificação e classificação de dados que estão em representação

digital e o processo de Registro método que retorna a transformada espacial que mapeia pontos de uma imagem aos pontos correspondentes em outra imagem. O registro pode ser utilizado na aquisição repetida de imagens e é geralmente usado para obter séries temporais e dessa forma capturar o desenvolvimento de doenças, progresso de tratamento, movimento de órgãos e tecidos e etc. (Yoo, 2004). Como exemplos da aplicação do processo de segmentação pode-se citar a localização e monitoramento de patologias, estudo de estruturas anatômicas, medição de volume, cirurgia guiada por imagens e etc. (O'Donnell, 2001). Como o ITK apresenta as funções de leitura, escrita e processamento de imagem, utilizou-se a biblioteca VTK, também gratuita, que desempenha papel na renderização, visualização e interação com as imagens. Alguns filtros do ImageLab também utilizam o VTK na parte de processamento. Este é desenvolvido em C++ e utiliza premissas OpenGL na renderização de dados. Para a *interface* foi escolhida a biblioteca Qt, desenvolvida pela empresa Trolltech, por permitir um rápido desenvolvimento dos componentes de *interface* com excelente produtividade.

3. ImageLab: Principais funcionalidades

Inicialmente, o ImageLab surgiu da necessidade de desacoplar as funcionalidades de processamento de imagens que foram originalmente desenvolvidas para o HeMoLab (Larrabide and Feijóo, 2006). Leitor de imagens DICOM, seleção de sub-espaço de trabalho e filtros de suavização e segmentação são exemplos de funcionalidades incluidas neste novo *software*. Porém, na medida em que a ferramenta era desenvolvida notou-se que o ImageLab também poderia ser desenvolvido para utilização na pesquisa acadêmica e também como ferramenta auxiliar na prática médica.

O sistema desenvolvido possui em sua *interface* inicialmente três janelas para a visualização do volume de imagens nos planos Sagittal, Axial e Coronal e uma quarta janela onde é exibida uma das opções de visualização em 3D como mostra a Figura 1.

A ferramenta conta com a possibilidade de navegação pela execução dos filtros através de operações de *undo*/*redo*, ou seja, na medida em que são aplicados filtros e novas imagens são geradas, as anteriores são armazenadas em memória em uma estrutura de *buffer* circular e assim as instâncias mais antigas não são removidas da memória. Dessa maneira, ao retornar para um estado anterior, nenhum processamento adicional é realizado. O número de instâncias armazenadas em memória pode ser configurado permitindo que a aplicação se ajuste à quantidade de memória da máquina em que está sendo executada a aplicação.

Preocupou-se em criar um sistema customizável, pois essa característica é essencial na facilidade de uso e oferecimento de uma *interface* amigável ao usuário, contribuindo de forma predominante na adoção da ferramenta por usuário. Dessa forma, é possível, por exemplo, configurar a ação que é associada ao botão direito do *mouse* (operações de *zoom, rotate, pan*), controle de visualização de dados de identificação na tela, opções de visualização das imagens e etc.

O ImageLab incorpora diversos filtros de processamento de imagens, sendo a maioria do ITK. Estes estão classificados nas seguites categorias: filtros de **filtragem por tom de cinza** (*thresholding*) que mudam ou identificam o valor do *pixel* baseados em um ou faixa de valores especificados no parâmetro do filtro; filtros de **suavização** (*smoothing*) que desempenham importante função no melhoramento da qualidade da imagem através da redução de ruído e visam, tipicamente, uma posterior etapa de segmentação; na categoria de filtros de **detecção de borda** (*edge Detection*) o ImageLab conta com o filtro (*CannyEdge Detection*) que é capaz de detectar e ressaltar contornos; filtros de **gradiente** ajudam a determinar os contornos dos objetos e na separação de regiões homogêneas; filtros de **vizinhança** realizam convolução de *pixels* basedos nos valores dos vizinhos e incluem redução de ruído; filtros de **segmentação**

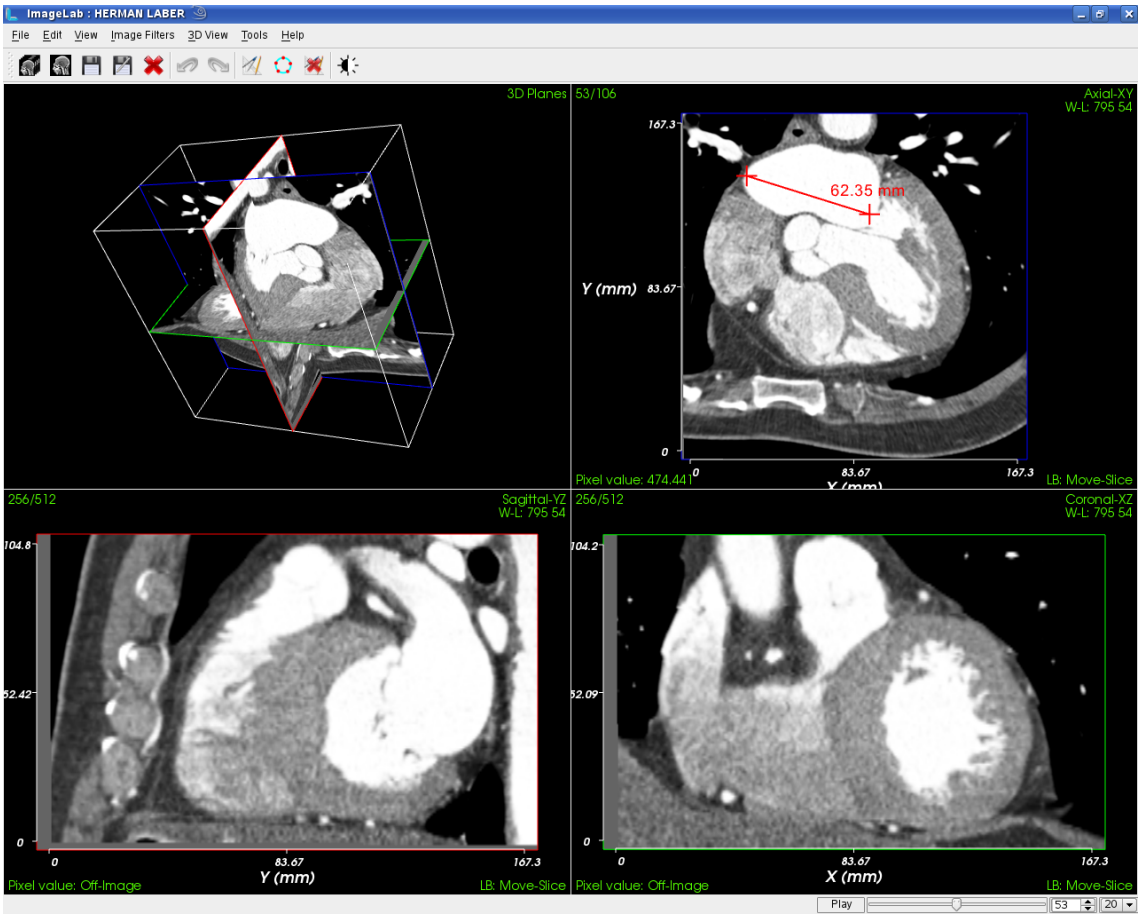


Figura 1: Janela principal do ImageLab.

visam a identificação de determinadas estruturas presentes na imagem. Como o processo de segmentação é dependente do tipo de imagem, o ITK oferece métodos de segmentação baseados em diversas técnicas como, por exemplo: crescimento de região, segmentação baseada em *watersheeds*, *level-set* e etc. Além de filtros de segmentação próprios do ITK, o ImageLab incorpora também o filtro de derivada topológica (Larrabide, 2007) e *Voxel Grown* (Larrabide and Fiorentini, 2003), implementados originalmente no HeMoLab.

Imagens DICOM podem apresentar um grande número de *slices*. Sendo assim, a aplicação de um filtro no volume de imagens inteiro pode demandar alta carga de processamento e consequentemente atrasar o processo de "aplicação de filtro e visualização de resultado". Com intuito de dinamizar este processo, a função chamada *quick apply* foi desenvolvida. Esta se baseia na aplicação do filtro, com os parâmetros escolhidos pelo usuário somente no *slice* visualizado no momento e assim o processamento é acelerado de forma significativa. Desta forma, o usuário tem a possibilidade de determinar os parâmetros ideais para aquele tipo de filtro, podendo então iniciar o processamento no volume completo de imagens. É necessário ressaltar que este procedimento de aceleração funciona integralmente em filtros que sejam locais, no sentido de não precisar de informações de *slices* vizinhos, ou seja, em filtros que operem somente no *slice* atual. Para a maioria dos filtros de segmentação o resultado do *quick apply* diverge do resultado da aplicação do filtro no volume inteiro devido a estas razões.

Além de processamento de imagens médicas, o ImageLab também foi projetado para ser utilizado como etapa inicial do processo de modelagem e simulação computacional do sistema cardiovascular. As imagens médicas são utilizadas pois fornecem a geometria 3D dos distri-

tos arteriais, onde a simulação do escoamento sanguíneo será posteriormente realizada. Para a obtenção da geometria, as estruturas de interesse devem ser inicialmente segmentadas (por exemplo, obtenção de imagem binária). Depois, aplica-se filtro de geração de iso-superfícies e obtem-se a malha de triângulos que descreve a geometria. Finalizando, o ImageLab permite escrever a malha em arquivo (formato vtk ou formato próprio do HeMoLab) e esta poder ser lida pelo HeMoLab permitindo que o processo de modelagem (refinamento e melhoramento de malha, inclusão de parâmetros materiais, definição de condições de contorno e etc.) seja continuado.

Como ferramentas básicas de trabalho, o *software* apresenta módulo de medição de área e comprimento, apresentação do valor de *pixel* da imagem (tom de cinza), seleção de sub-volume (função *extract-grid*), variação de parâmetros de visualização *window-level*, caminhamento entre *slices* de forma automática (função cine) e futuramente visualização do histograma da imagem. Também foi implementado um módulo de comparação de arquivos DICOM, onde dois conjuntos de imagens podem ser visualmente contrastados. Esta característica é particularmente útil em situações onde se deseja analisar uma mesma imagem em estados diferentes, duas imagens de fontes diferentes (pacientes diferentes) e etc. (Figura 2).

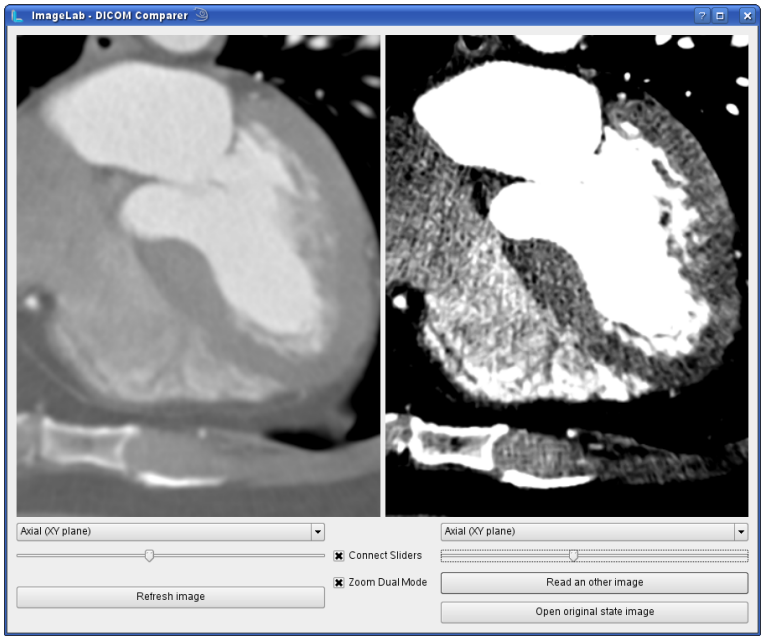
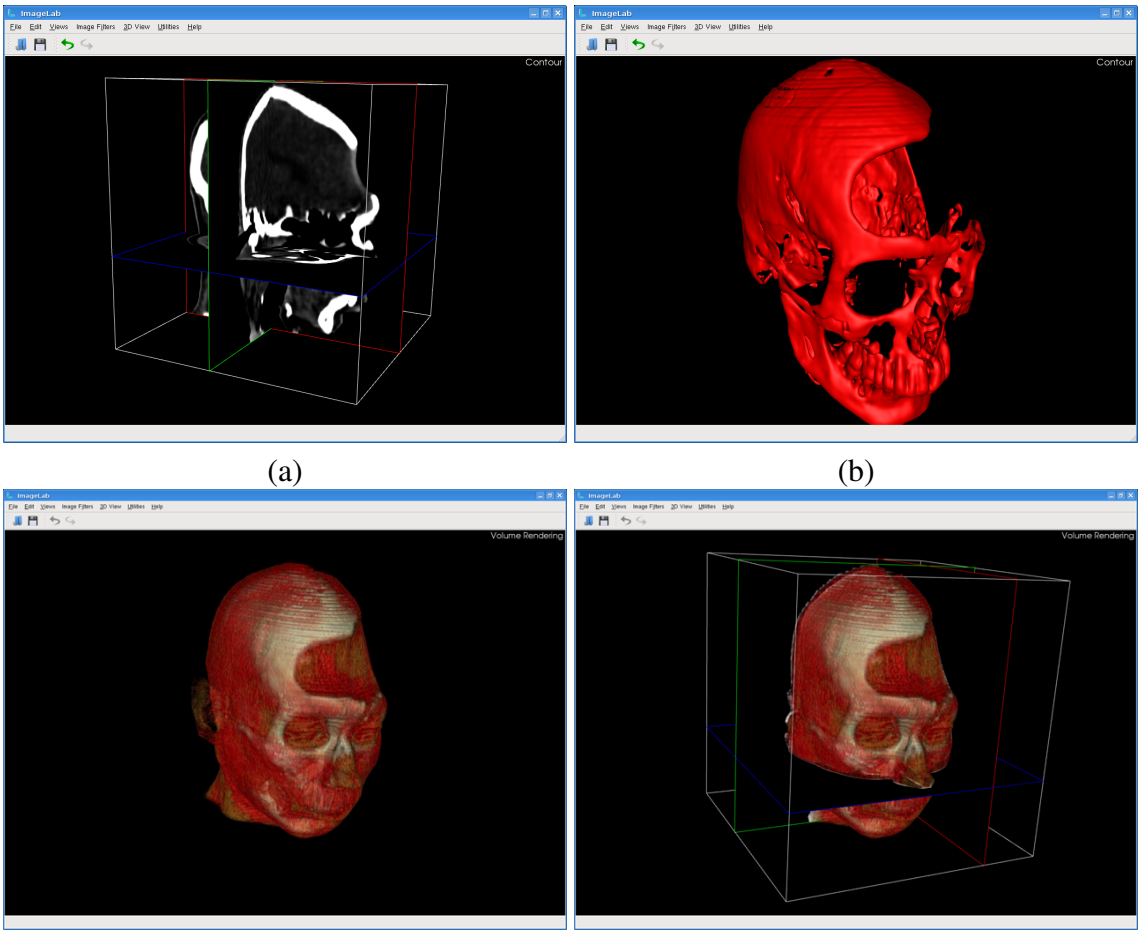


Figura 2: Janela de comparação de imagens médicas.

O sistema desenvolvido possui diferentes possibilidades de visualização em 3D como, por exemplo, a representação interativa dos planos Sagittal, Axial e Coronal em um mesmo volume (Figura 3.a). Os planos podem ser rotacionados e assim ficarem oblíquos entre si. Este tipo de funcionalidade é bastante utilizado na geração de imagens com corte transversal à estruturas de interesse. Por exemplo, na etapa de diagnóstico é possível avaliar se uma determinada artéria apresenta estenose (estreitamento por acúmulo de tecido) através da obtenção manual da sequência de imagens transversais à esta. A ferramenta já conta com obtenção de planos oblíquos e atualmente trabalha-se na criação de uma *interface* mais amigável para esta funcionalidade. Como demais opções de visualização 3D incluem-se a geração de superfícies (Figura 3.b), *VolumeRendering* (Figura 3.c) ou uma composição entre esses métodos (Figura 3.d).



(a) (b)

(c) (d)

Figura 3: Exemplos de modos de visualização em 3D disponíveis no ImageLab. Pode-se visualizar através de planos (a), Superfícies (b), *Volume Rendering* (c) ou uma composição entre os métodos anteriores (d).

3.1 Aspectos de arquitetura do sistema

Nesta sessão serão apresentados os principais aspectos relacionados à engenharia de *software*, descrevendo a modelagem adotada no desenvolvimento de cada módulo do sistema e a interação entre eles. Utilizamos como base para a documentação do *software*, a linguagem *Unified Modeling Language* (UML) que é centrada em arquitetura (diagramas em geral), é iterativa (liberação de versões do sistema) e incremental (adição de funcionalidade no sistema) (Medeiros, 2004).

Na Figura 4, temos a visão geral do sistema. As classes foram divididas em pacotes para um melhor entendimento de como funcionam os relacionamentos entre os módulos do sistema. O empacotamento das classes foi feito de acordo com suas heranças. Desta forma, mantém-se agrupadas as classes que possuem a mesma árvore de herança. Neste diagrama, é possível perceber como as principais classes de cada pacote interagem com as principais classes de outros pacotes. Foram mantidas apenas as interações mais relevantes para simplificar ao máximo o diagrama.

Na Figura 5, é apresentado o pacote de *interface* de forma mais detalhada. A classe imGUIMainInterface é responsável por montar a tela inicial do *software* e é responsável também por fazer a leitura e a escrita de arquivos, bem como, encaminhar-los para aplicação dos filtros. Nela é também instanciado o imGUIManager, que é responsável por todo gerenciamento

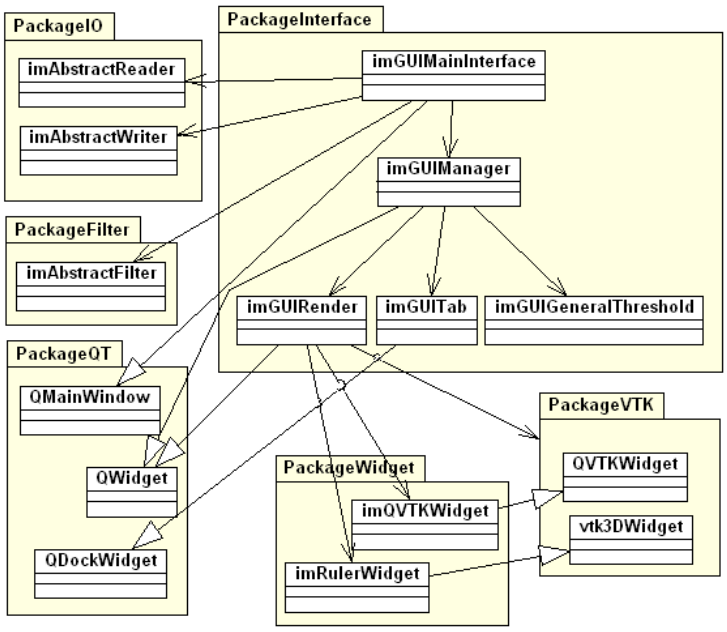


Figura 4: Arquitetura geral do sistema.

de *interfaces*, e ele por sua vez instancia, controla e promove a comunicação entre as *interfaces*. A classe imGUIRender é responsável por todas as operações de renderização 2D e 3D. A classe imGUITab é responsável por exibir as *interfaces* instanciadas pelo imGUIManager. Os arquivos com prefixo Ui (*User Interface*) são gerados pelo editor de *interface* do QT, chamado *designer*, e cada classe de *interface* herda do seu respectivo arquivo Ui. O *software* possui cerca de 40 filtros de processamento de imagem, e para simplificar o modelo, iremos utilizar apenas o filtro imGUIGeneralThreshold para representá-los. As classes de *interface* dos filtros possuem herança múltipla, pois precisam herdar de seus respectivos arquivos Ui e da classe QDockWidget. Estas características garantem que as *interfaces* dos filtros possam ser instanciadas e exibidas de forma dinâmica, de acordo com o filtro que for escolhido pelo usuário.

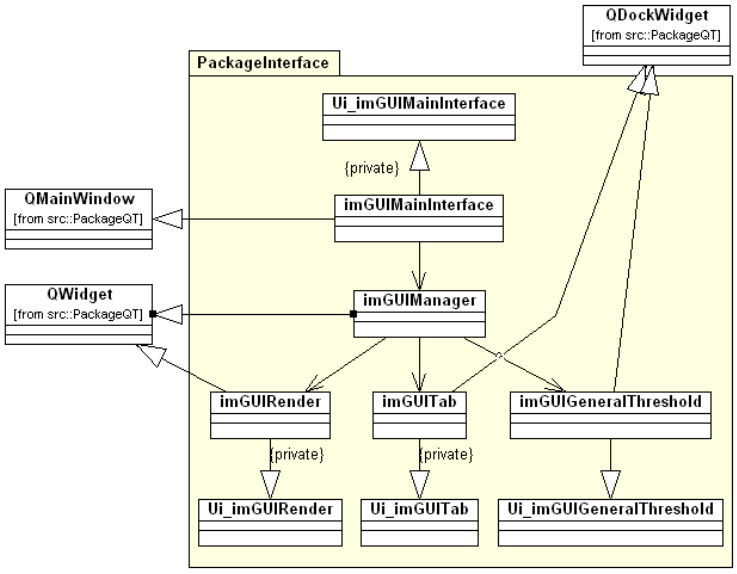
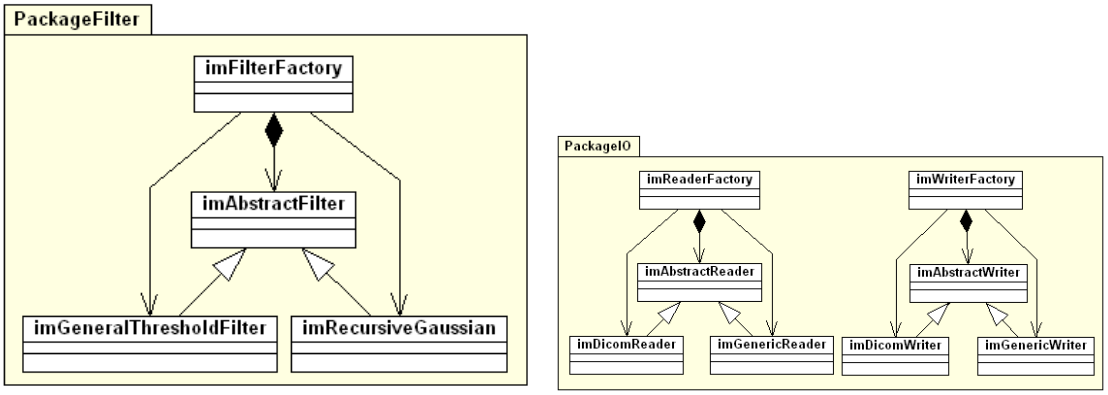


Figura 5: Pacote de *interface*.

Na Figura 6, o pacote *filter* é apresentado de forma mais detalhada. Para a criação deste modelo, usamos o padrão fábrica descrito em (Larman, 2004), para garantir que assim como as suas *interfaces*, os filtros também possam ser utilizados de forma dinâmica. Todos os filtros derivam da classe imAbstractFilter e, conforme vimos na Figura 4, a classe imGUIMainInterface possui uma referência do tipo imAbstractFilter. No momento em que o filtro é aplicado, esta referência recebe o retorno de um método estático da classe imFilterFactory que conforme, o parâmetro de entrada, retorna um filtro caracterizando uma operação polimorfa. O mesmo procedimento ocorre com o pacote IO na Figura 6.b.



(a) (b)

Figura 6: Diagrama de pacote: Filtros (a) e IO (b)

Outra entidade muito importante do *software* é chamada de *InstanceManager* que foi desenvolvido utilizando o *design pattern* denominado *singleton* (que garante a existência de apenas um objeto único desta classe, mantendo um ponto global de acesso a este) (Gamma et al., 1994). A seguir, serão explicadas algumas das funcionalidades principais do *InstanceManager* no contexto do funcionamento do ImageLab. O *InstanceManager* é responsável por gerenciar os objetos de imagem, armazenando-os em uma estrutura do tipo vetor. Com isso, é possível executar operações de desfazer (*undo*) e refazer (*redo*) navegando entre as imagens produzidas pelas sequenciais aplicações dos filtros de imagem. Quando um conjunto de imagens DICOM é lido pelo sistema, este é armazenado na primeira posição do vetor de imagens. Na aplicação de algum filtro, uma nova imagem é criada e este novo objeto é armazenado no vetor na posição seguinte a do objeto usado como base para aplicação do filtro, de maneira à preservar a imagem base. Para exemplificar o funcionamento do *InstanceManager*, iremos convensionar algumas abreviações: A letra **O** representará a instância original que é o conjunto de imagens inicialmente carregado pelo ImageLab. A letra **T** representará a aplicação do filtro *General Threshold* sobre **O**. A letra **G** representará a aplicação do filtro *Recursive Gaussian* sobre **T** e por último a letra **B** representará a aplicação do filtro *Bilateral Image* sobre **G**. A seta para baixo representa o índice do vetor e indica qual imagem está sendo visualizada nas áreas de renderização (*renders*). Nos exemplos o *InstanceManager* poderá armazenar até cinco instâncias, gerando assim um vetor de cinco posições. Apresentaremos como o *InstanceManager* se comporta utilizando a aplicação de filtros após a utilização da função desfazer(*undo*). Na Figura 7.a temos um *pipeline*[1] com três filtros e **B** está sendo visualizada nas renders. Em 7.a-2 foram utilizados três vezes a função desfazer, com isso **O** está sendo mostrado nas renders. Em 7.a-3 foi utilizado o filtro **G**. O objeto corrente é utilizado como base para aplicação do filtro **G**, e conforme esse

[1]pipeline é o nome utilizado para descrever uma estrutura de elementos de processamento conectados, arranjados de forma em que a saída de um elemento é configurado como a entrada do próximo

caso, as demais instâncias foram desalocadas e **G** armazenado no vetor de imagens na posição seguinte a de **O**. Apresentaremos agora como o *InstanceManager* se comporta quando diminuimos o número de instâncias na janela de preferências. Não apresentaremos o comportamento do *InstanceManager* quando aumentamos o número de instâncias pois o redimensionamento é trivial, apenas aumentando o número posições do vetor. Na Figura 7.b-1 temos o mesmo *pipeline* da Figura 7.a-1 adicionado de uma função desfazer, o que deixa **G** como objeto corrente sendo mostrado nas renders. Em 7.b-2 reduzimos o número de instâncias de cinco para três, o que faz com que as imagens fora do limite estabelecido sejam desalocadas e o vetor seja redimensionado. Mais abaixo na Figura 7.b-3 novamente reduzimos o número de instâncias, e o vetor deverá ser redimensionado. Neste caso o objeto corrente é que está fora do limite estabelecido, e de maneira a preservá-lo, são desalocados os objetos menos recentemente utilizados (neste caso **O**) e o vetor então é redimensionado. Isto é feito para preservar ao máximo o *pipeline* e os parâmetros utilizados na aplicação dos filtros.

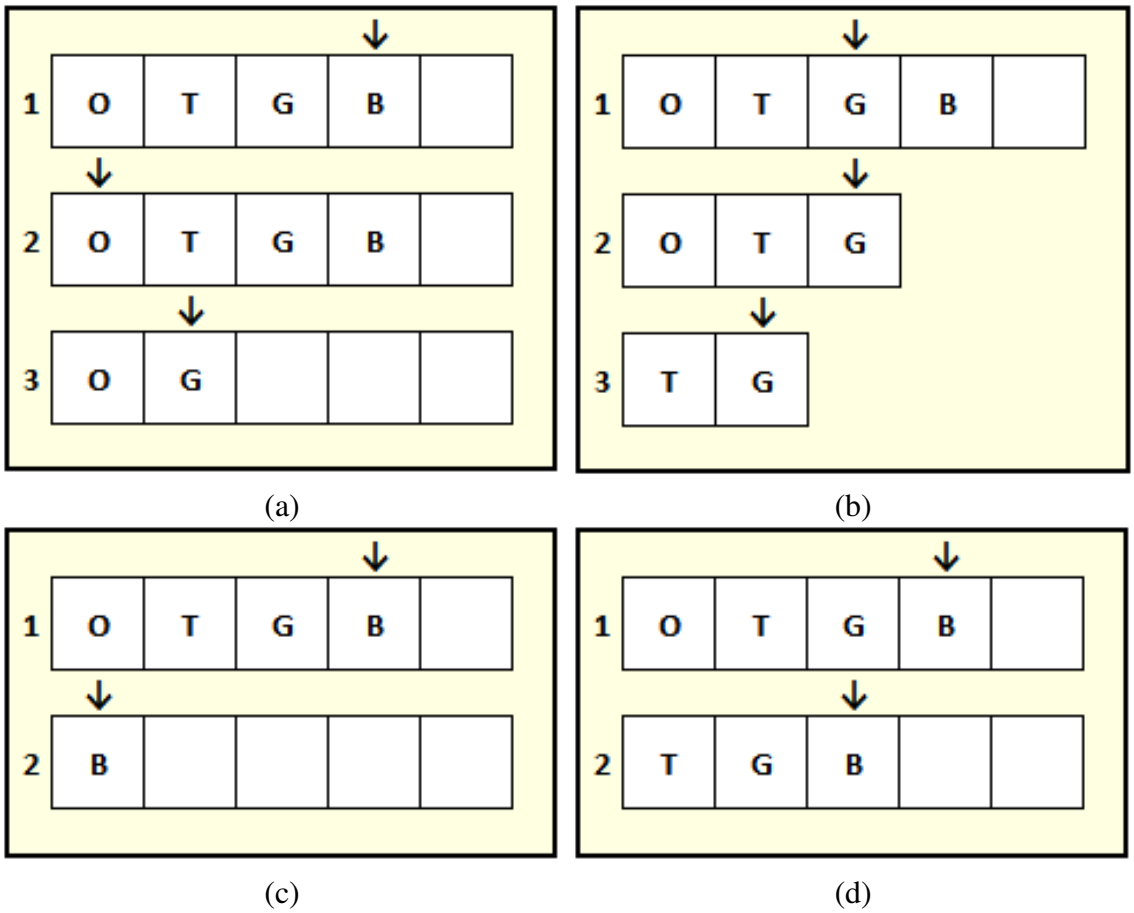


Figura 7: (a) Pipeline com quatro imagens, três operações de desfazer e aplicação de novo filtros, (b) Redimensionamento do tamanho do *InstanceManager*, (c) função *Remove Others*, (d) função *Remove Original Instance*

Apresentaremos agora uma função chamada *Remove Others*. Esta função elimina todos os objetos com excessão do objeto corrente. Essa funcionalidade é útil na economia de memória, visto que aplicação de um filtro resulta em um novo objeto instanciado, e o mesmo é inteiramente alocado em memória. Na Figura 7.c-1 temos o mesmo *pipeline* das figuras anteriormente citadas, e como o objeto corrente é o **B** todas as outras instâncias de objetos são então desalocadas, permanecendo **B** como instância única como em 7.c-2. Por último, outra função para economia de memória, chamada de *Remove Original Instance* que remove apenas a imagem **O**,

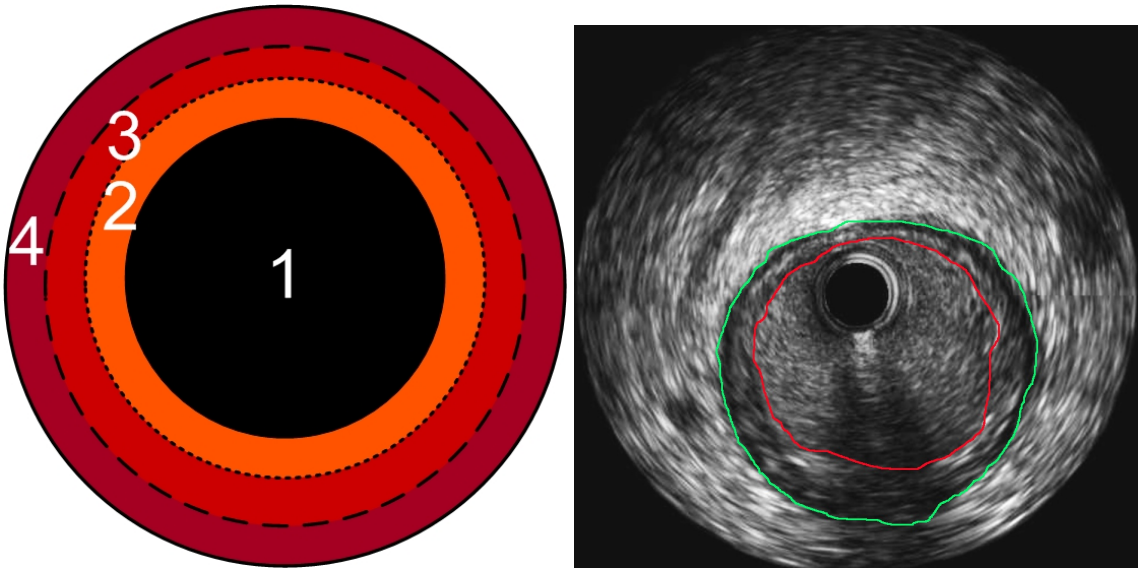
como mostra a Figura 7.d-1 e 2.

4. Perspectivas na visualização e processamento de imagens de exames *Intravascular Ultrasound* (IVUS) no ImageLab

O IVUS (*Intravascular Ultrasound*) (Nissen-MD and Yock-MD, 2001) é um exame invasivo que permite a obtenção de imagens de ultrasom das estruturas internas arteriais e é realizado geralmente em artérias coronárias. As imagens são obtidas com o auxílio de um cateter que emite ondas sonoras e captura o retorno destas e a partir da análise das intensidades de retorno é possível inferir os materiais constituintes e assim gerar imagens representativas destes. Um exemplo de imagem produzida pode ser visualizada na Figura 8.b. O conjunto de imagens é gerado na medida em que o cateter é puxado (*pullback*) com velocidade constante (geralmente 0.5mm/s) e produz imagens com uma taxa de, tipicamente, 30 quadros/seg.

Para poder entender as imagens IVUS é necessário conhecer a estrutura da parede arterial. Esta apresenta três camadas (Figura 8.a): **adventícia** (4) que é a camada mais externa e faz contato com o tecido externo circudante; **média** que é camada intermediária (3) e **íntima** (2) que é a camada que faz contato com o **lúmen** (1)(região preenchida pelo o sangue). Na íntima, placas de tecido podem se desenvolver e assim possibilitam a geração de um conjunto de patologias chamado de Arterioesclerose.

Nas imagens IVUS, além das diversas camadas da parede arterial, placas de material calcificado/ colesterol/ tecido fibroso e etc. podem também ser visualizados e assim auxiliar o Médico no diagnóstico e definição de estratégias de tratamento (exemplo de aplicação: localização e dimensionamento de *Stent*).



(a) (b)

Figura 8: (a) Camadas constituientes da parede arterial. (b) Bordas das paredes arteriais destacadas (aproximação): Borda lúmen/íntima delimitada pela linha vermelha e borda média/adventícia pela linha verde)

Imagens IVUS são obtidas através de técnicas de ultrasom e apresentam qualidade inferior à imagens de tomografia computadorizada e ressonância magnética. Devido a isso, a etapa de pré-processamento (visando a segmentação) apresenta-se fundamental para sucesso na segmentação. Espera-se que através de filtros básicos de melhoramento e remoção de ruído e filtro específicos desenvolvidos para imagens IVUS, o ImageLab possa cumprir de forma eficiente, a segmentação de imagens IVUS.

O ImageLab pode ler imagens IVUS (geralmente armazenados como DICOMs *multi-frame*) e pretende-se oferecer a possibilidade ao usuário de aplicar filtros de segmentação que sejam, inicialmente, capazes de corretamente identificar as bordas lúmen/íntima e borda média/adventícia (Giannoglou et al., 2006) (Figura 8.b). Em um segundo estágio, estruturas, como por exemplo, placas de cálcio, também seriam identificadas. Através da obtenção da linha central de caminhamento da artéria (através de imagens de Angiografia biplanar ou imagens de Tomografia), a malha de superfície, que descreve a geometria da artéria, pode ser gerada permitindo a composição de modelos de simulação mais completos compostos de parede interna, externa e grupos com propriedades materiais distintas (modelos atualmente utilizados descrevem somente a parede externa). Uma vez as paredes identificadas, é possível também realizar a representação dos *frames* IVUS no espaço e através da utilização de técnicas de computação gráfica, o *software* poderia marcar áreas, de por exemplo, estreitamento (visando auxiliar o Médico no processo de diagnóstico) ou outros fatores dignos de serem ressaltados.

5. Conclusões e perspectivas futuras

No presente artigo foi apresentado o sistema ImageLab que tem o objetivo de ser um ambiente para manipulação, processamento e visualização de imagens médicas em formato DICOM e imagens em formatos convencionais. O ImageLab foi projetado para ser um ambiente base para a inclusão e validação de novos métodos de imagem, permitindo assim a realização de estudos mais específicos nas diversas áreas de processamento de imagens médicas.

O *Software* contém diversos filtros de processamento, disponibilizados pela biblioteca ITK, bem como outros filtros, desenvolvidos para o ambiente HeMoLab, utilizados na segmentação de imagens médicas. A biblioteca VTK é utilizada para visualização das imagens e interação com o usuário, enquanto na *interface* foi utilizado o *toolkit* Qt. Técnicas de visualização 3D, como planos em 3D, *Volume Rendering* e Superfícies estão disponíveis dentro do ambiente.

O uso clínico da ferramenta, por exemplo no processo de diagnóstico, também é uma dos possíveis aplicações do *software*, uma vez que este oferece as funcionalidades básicas para este tipo de procedimento. Uma importante inovação do *software* será a capacidade de oferecer técnicas de processamento e visualização de imagens IVUS. Estas apresentam-se de extrema importância no contexto da Modelagem Computacional e também na análise clínica. Espera-se que seja possível realizar o estudo das propriedades mecânicas das estruturas que compõem as artérias (através de técnicas de problema inverso) permitindo, dessa forma, a composição de modelos mais complexos que englobem informação sobre as camadas na parede arterial, tecidos com propriedades mecânicas diferentes, etc.

É importante destacar que a ferramenta será testada e validada por instituições parceiras que atuam no campo de imagens médicas e assim estas poderão fornecer informações valiosas para a correção de falhas e inclusão de novas funcionalidades mais específicas à determinadas especialidades.

Referências

Ackerman, M. J., 1995. Accessing the visible human project. *D-Lib Magazine*, vol. 1.

Avolio, A., 1980. Multi–branched model of the human arterial system. *Med. Biol. Engrg. Comp.*, vol. 18, pp. 709–718.

Blanchette, J. & Summerfield, M., 2008. *C++ GUI Programming with Qt 4*. Prentice Hall.

Gamma, E., Helm, R., Johnson, R., & Vlissides, J., 1994. *Design Patterns Elements of Reusable Object-Oriented Software*. Addison Wesley Professional Computing Series.

Giannoglou, G. D., Chatzizisis, Y. S., Sianos, G., Tsikaderis, D., Matakos, A., Koutkias, V., Diamantopoulos, P., Maglaveras, N., Parcharidis, G. E., & Louridas, G. E., 2006. In-vivo validation of spatially correct three-dimensional reconstruction of human coronary arteries by integrating intravascular ultrasound and biplane angiography. *Coronary Artery Disease*, vol. 17, n. 6, pp. 533–543.

ITK - Insight Toolkit, 2009. http://www.itk.org.

Larman, C., 2004. *Utilizando UML e Padres*. Bookman.

Larrabide, I., 2007. *Processamento de imagens via derivada topológica e suas aplicações na modelagem e simulação do sistema cardiovascular humano*. PhD thesis, National Laboratory for Scientific Computation.

Larrabide, I. & Feijóo, R., 2006. HeMoLab: Laboratório de Modelagem em Hemodinâmica. Technical Report 13/2006, National Laboratory for Scientific Computing.

Larrabide, I. & Fiorentini, S., 2003. Voxel grow – a region growing segmentation technique. *International Conference on Computer Science, Software Engineering, Information Technology, e-Business, and Applications*.

Medeiros, E. S., 2004. *Desenvolvendo Software com UML 2.0*. Pearson Makron Books.

Nissen-MD, S. E. & Yock-MD, P., 2001. Intravascular ultrasound novel pathophysiological insights and current clinical applications. *Circulation*, vol. 103, n. 4, pp. 604–616.

O'Donnell, L., 2001. Semi-automatic medical image segmentation. Master's thesis, Massachusetts Institute of Technology.

Oshima, M., 2004. A New Approach to Cerebral Hemodynamics: Patient-Specific Modeling and Numerical Simulation of Blood Flow and Arterial Wall Interaction. *Bulletin for The International Association for Computational Mechanics*, vol. 14, pp. 4–9.

Pianykh, O. S., 2008. *Digital Imaging and Communications in Medice (DICOM)*. Springer.

Projeto MACC, 2009. https://macc.lncc.br.

Standard, T. D., 2009. http://medical.nema.org/.

Stergiopulos, N., D.F. Young, & T.R. Rogge, 1992. Computer Simulation of Arterial Flow with Applications to Arterial and Aortic Stenoses. In *Journal of Biomechanics*, volume 25, pp. 1477–1488.

Urquiza, S., Blanco, P., Vénere, M., & Feijóo, R., 2006. Multidimensional modeling for the carotid blood flow. *Comput. Meth. Appl. Mech. Engrg.*, vol. 195, pp. 4002–4017.

VTK - Visualization Toolkit, 2009. http://www.vtk.org.

Yoo, T. S., 2004. *Insight into Images: Principles and Practice for Segmentation, Registration, and Image Analysis*. AK Peters Ltd.