UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

DENNIS GIOVANI BALREIRA

Sistema de Visualização do Interior de Objetos com Estruturas Internas para Simulação de Cirurgias

Dissertação apresentada como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação

Orientador: Prof. Dr. Marcelo Walter
Co-orientador: Prof. Dr. Anderson Maciel

Porto Alegre
2015

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Balreira, Dennis Giovani

Sistema de Visualização do Interior de Objetos com Estruturas Internas para Simulação de Cirurgias / Dennis Giovani Balreira. – Porto Alegre: PPGC da UFRGS, 2015.

67 f.: il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2015. Orientador: Marcelo Walter; Co-orientador: Anderson Maciel.

1. Computação gráfica. 2. Sistema de visualização. 3. Simuladores cirúrgicos. 4. Estruturas internas. I. Walter, Marcelo. II. Maciel, Anderson. III. Título.

“Não basta saber, é preciso também aplicar;
não basta querer, é preciso também agir.”
— JOHANN WOLFGANG VON GOETHE

AGRADECIMENTOS

À minha família, principalmente à minha mãe Maria José, por toda a força que tive desde sempre, em especial durante esse período, essencial para conquistar mais essa etapa.

Aos meus amigos pela presença mesmo quando o tempo era contado e aos colegas do grupo de Computação Gráfica pela convivência diária nas dependências do instituto.

Ao professor orientador Marcelo Walter por todos os conselhos, motivações e revisões ao longo de todo esse período e ao professor co-orientador Anderson Maciel pelo apoio e tempo dedicados ao trabalho.

Aos professores do grupo de computação gráfica pela expansão do conhecimento através das disciplinas cursadas, em especial ao professor Manuel Menezes de Oliveira Neto e à professora Carla Maria Dal Sasso Freitas pela revisão durante o seminário de andamento.

À UFRGS e ao Instituto de Informática pelo ensino de qualidade desde a graduação, possibilitando expandir meu conhecimento ainda mais durante o mestrado.

Ao PPGC pela oportunidade e ao CNPQ pelo financiamento da bolsa de pesquisa.

RESUMO

Simuladores cirúrgicos possibilitam a médicos e estudantes a visualização de órgãos humanos representados com alto grau de realismo. Trata-se de um grande avanço que reduz o tempo e os custos na formação desses profissionais, oferecendo menor risco aos pacientes. Entretanto, esses sistemas comerciais de simuladores têm, em geral, suas representações voltadas apenas para a superfície dos órgãos. Assim, quando ocorre o corte de um órgão com estruturas internas, o seu interior aparece vazio, não mostrando, por exemplo, estruturas anatômicas como vasos sanguíneos no seu interior. Trabalhos na área de textura sólida também não lidam com objetos que possuem estruturas internas modeladas geometricamente. Nessa dissertação é proposto um sistema gráfico interativo que permite cortes arbitrários em estruturas anatômicas, reconstruindo a textura nas superfícies da zona de corte de forma a respeitar as estruturas internas. O sistema protótipo desenvolvido propõe que cortes definidos pelo usuário alterem modelos tridimensionais, retriangulando apropriadamente a malha nas regiões de corte. Como estudo de caso, foi selecionado um modelo de fígado humano com vasos, apresentando como resultado a visualização interna em tempo real desse objeto para quaisquer planos. Essa abordagem pode ser considerada como um passo para os simuladores se tornarem ainda mais realistas. Extensões do trabalho envolvem a integração com simuladores médicos existentes, bem como uma validação do sistema por parte dos profissionais da Medicina.

Palavras-chave: Computação gráfica. Sistema de visualização. Simuladores cirúrgicos. Estruturas internas.

Visualization System for the Interior of Objects with Internal Structures for Surgery Simulation

ABSTRACT

Surgical simulators help doctors and students the visualization of human organs with high realism. Simulators are an improvement which reduce time and cost spend by professionals, offering less risk to the patients. Besides, studies show that the amount of realism seen in the simulators is related to the engaging of students in learning. However, these commercial simulator systems represent only the surface of organs. Thus, when a cut of an organ with internal structures occurs, its interior remains empty, without showing anatomical structures such as blood vessels inside. Research in the area of solid texture typically do not deal with objects which have internal structures geometrically modeled. In this dissertation, we propose a graphical interactive system that allows arbitrary cuts in anatomical structures, reconstructing the texture in the cutting zone's surface according to its internal structures. With the developed application, cuts defined by the user transform three-dimensional models, triangulating properly the mesh in the cutting area. As a case study we selected a human liver model with vessels, presenting as result the internal visualization of the object in real time for any cut planes. We consider this approach as a step in order for simulators to become more realistic. Extensions of the work include the integration with current medical simulators, as well as a validation of the system by the Medicine professionals.

Keywords: Computer graphics, Visualization system, Surgical simulators, Internal structures.

LISTA DE ABREVIATURAS E SIGLAS

3D	Três Dimensões
2D	Duas Dimensões
CG	Computação Gráfica
HMD	Head Mounted Display
FFT	Fast Fourier Transform
FVR	Full Volume Rendering
CT	Computed Tomography
CAD	Computer Aided Design
VTK	The Visualization Toolkit
GPU	Graphics Processing Unit
BSP	Binary Space Partitioning
ESS	Endoscopic Sinus Surgery
VLSI	Very-large-scale integration
MRI	Magnetic Resonance Imaging
API	Application Programming Interface

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

A simulação de procedimentos médicos é uma forma segura de educar e treinar o graduando de Medicina, pois propicia, além do conhecimento, a repetição de procedimentos, algo impossível com pacientes. Substituir o uso de órgãos reais de pessoas ou animais por modelos virtuais também é uma forma de atenuar questões ético-religiosas e legais que envolvem o assunto e minorar os transtornos que costumam ocorrer com os futuros médicos nas aulas de anatomia.

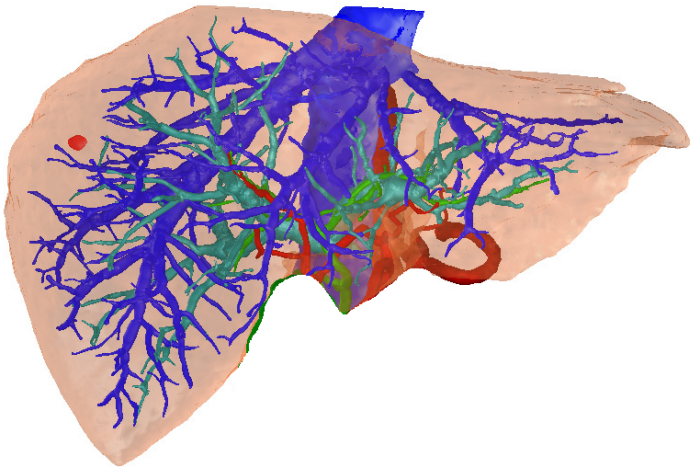
Durante anos, a prática de cirurgias era feita diretamente nos pacientes, fazendo com que o aluno aprendesse sem etapas intermediárias. Recentemente, a utilização de simuladores cirúrgicos (TAGAWA et al., Jan. 2013) (ECHEGARAY et al., May/June 2014) (VARSHNEY et al., July/Aug. 2014) vem ganhando espaço no ensino e se tornando cada vez mais uma ferramenta indispensável da medicina. Pesquisas apontam que o uso de simuladores apresenta diversos benefícios quando incorporados ao treinamento médico (KUNKLER, Sep. 2006), como melhoramento da experiência educacional, aumento da segurança dos pacientes, eficiência de custo e aprimoramento de novas técnicas. Um dos trabalhos (DAWSON; KAUFMAN, Mar. 1998) argumenta que simuladores possibilitam não só o treinamento de novos cirurgiões, mas também o aperfeiçoamento de médicos experientes com técnicas desenvolvidas depois que o aprendizado se completou. Adicionalmente, em estudo publicado (YOUNGBLOOD et al., Apr. 2005), usuários que fizeram previamente o treino com o simulador virtual *LapSim* apresentaram em geral um melhor desempenho cirúrgico posterior do que os usuários que praticaram com o simulador mecânico *Tower Trainer* e dos que não fizeram este tipo de aprendizado. Recentemente, outro estudo (YIANNAKOPOULOU et al., Jan. 2015) confirmou a utilidade dos simuladores de realidade virtual para cirurgias de laparoscopia, mas ressaltou a falta de estudo na performance para procedimentos cirúrgicos avançados.

A fim de aproximar esses dispositivos da realidade, os instrumentos de interação usados se tornam cada vez mais precisos, não apenas se assemelhando aos reais, como também fornecendo um retorno háptico adequado. Da mesma forma, as malhas poligonais do sistema são construídas respeitando algoritmos de colisão e deformação. Além disso, avanços vem contribuindo significativamente para também aprimorar a experiência visual, utilizando técnicas de iluminação em tempo real para que os ambientes virtuais dos simuladores se pareçam o mais possível com os de uma cirurgia (NUNES; WALTER; MACIEL, ago. 2012).

Entretanto, ocorre atualmente nesses simuladores que objetos virtuais em geral não costumam ser modelados como um todo. Na maioria dos casos a representação apenas das superfícies dos órgãos demonstra ser suficientes para grande parte das aplicações, uma vez que não

há necessidade de criar tridimensionalmente estruturas internas de objetos se não houver visualização ou interação com seu interior. Embora a maioria dos órgãos humanos contenha veias e outras estruturas internas, durante a simulação virtual esses modelos aparecem simplificados, sendo constituídos apenas por sua superfície, definida como uma malha poligonal. A Figura 1.1 apresenta um fígado com suas veias representando as estruturas internas.

Figura 1.1 – Fígado com suas estruturas internas.



Fonte: Prof. Dr. Hans-Peter Meinzer, Deutsches Krebsforschungszentrum | © dkfz.de

1.1 Motivação

Com os avanços na área de GPU (*Graphics Processing Unit* - Unidade de Processamento Gráfico), cada vez mais aplicações que envolvem Computação Gráfica apresentam uma crescente necessidade de processamento gráfico de polígonos em tempo real, possibilitando que mais informações sejam tratadas. Por esse motivo, apresentar objetos que contenham dados no interior é uma extensão natural e intuitiva de aplicações que necessitem dessas referências. Alguns procedimentos que envolvem resseções de órgãos maciços, como uma hepatectomia (CLAVIEN et al., Apr. 2007), por exemplo, acabam expondo justamente partes internas dos órgãos. Assim, cirurgiões e professores da área médica demandam técnicas que auxiliem na construção de modelos para esses tipos de simuladores virtuais.

Em geral, quando um modelo é cortado em simuladores virtuais comerciais de cirurgia atuais, como o *LAP-MENTOR*, *GI-MENTOR* e *BRONC-MENTOR*, por exemplo, a superfície da malha é remodelada e nada aparece em seu interior. Mesmo que nenhum deles aborde cirurgias mais complexas, que envolvem a visualização interna completa de um órgão, seu

interior aparece vazio quando este é perfurado. Trata-se de um lapso, pois o órgão com seus vasos sanguíneos e a região maciça que o preenche não podem ser visualizados.

Diversos trabalhos pesquisados relacionados com simuladores cirúrgicos que envolvem operações com estruturas internas (DELINGETTE; AYACHE, Feb. 2005) (ECHEGARAY et al., May/June 2014) (ENDO et al., Aug. 2014) não apresentam foco nessa área. Outras pesquisas recentes se concentram em diferentes campos, como na reconstrução volumétrica dos órgãos para auxílio de cirurgias (SHINDO et al., Dec. 2011) (MISE et al., Feb. 2013) (YANG et al., 2012), não visando um *rendering* realístico. Especificamente na CG, trabalhos recentes na área de textura sólida, que garantem uma coloração tridimensional interna para modelos, também não abordam métodos que produzam bons resultados e que sejam específicos para esses tipos de estruturas (PIETRONI et al., Sep. 2007) (TAKAYAMA et al., Dec. 2010) (DU; HU; MARTIN, Mar. 2013).

1.2 Objetivo

O objetivo desse trabalho é propor uma nova técnica que permita cortes arbitrários em estruturas anatômicas, reconstruindo a textura das superfícies da zona de corte levando em conta as estruturas do interior. Assim, espera-se que esse trabalho possa contribuir com o realismo de simuladores cirúrgicos.

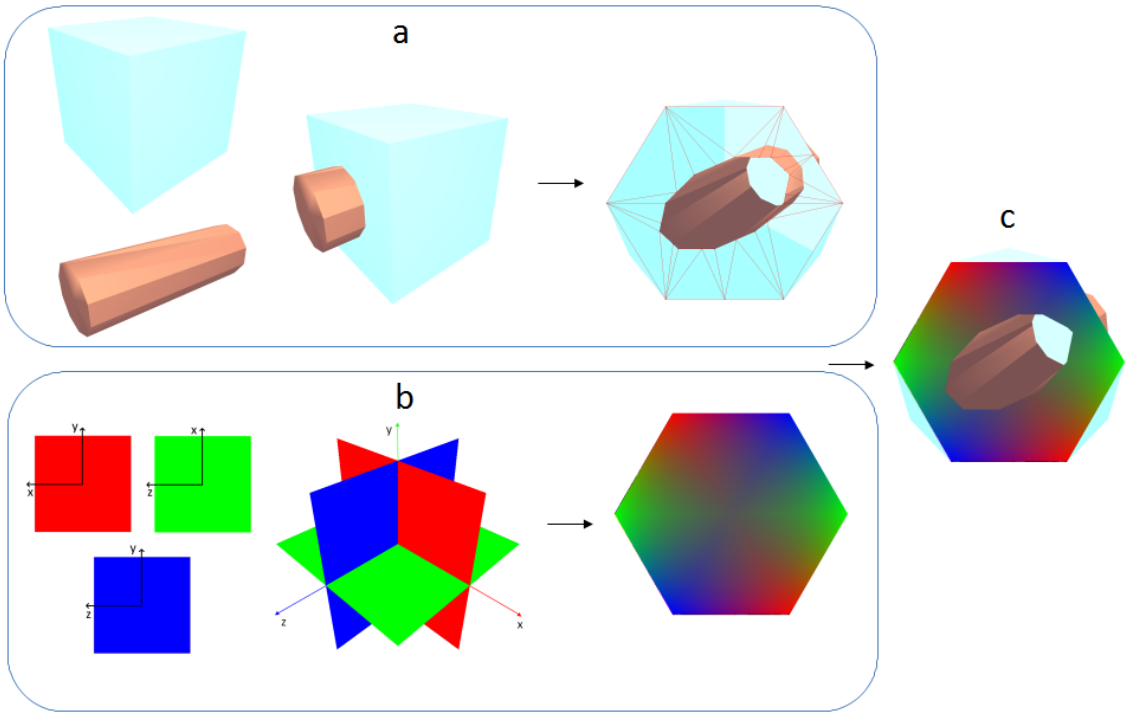
1.3 Visão geral

Este trabalho visa construir um sistema que lide com objetos com estruturas internas dada uma superfície de corte. O algoritmo proposto recebe como entrada um conjunto de imagens que representam a textura do interior do objeto, um conjunto de malhas poligonais tridimensionais e um corte, retornando a textura sobre esse plano especificado. A técnica consiste em duas etapas distintas. Na primeira, é trabalhada a parte de recorte e retriangulação dos modelos geométricos sobre os cortes. Na segunda fase, a textura é gerada por meio de uma interpolação, cujo cálculo é realizado a partir de uma função que relaciona a distância dos pontos do plano com as cores de pixels das imagens recebidas como entrada. O *pipeline* do trabalho com as etapas em alto nível se encontra na Figura 1.2.

O trabalho está organizado em cinco capítulos. O próximo capítulo apresenta, além de conceitos básicos sobre algoritmos geométricos e textura sólida necessários para o entendimento da

técnica, também trabalhos selecionados que envolvem simuladores cirúrgicos e textura sólida. Detalhes sobre o desenvolvimento do sistema, dividido em duas etapas, estão descritos no Capítulo 3. Um estudo de caso utilizando o sistema para efetuar cortes em fígados é apresentado no Capítulo 4. Por fim, o Capítulo 5 mostra as conclusões juntamente com as limitações e ideias para trabalhos futuros.

Figura 1.2 – *Pipeline* da abordagem. Dados um conjunto de malhas poligonais 3D (a) e um conjunto de imagens (b), retorna a textura sobre um plano de corte (c).



Fonte: Compilado pelo autor.

2 CONCEITOS BÁSICOS E TRABALHOS RELACIONADOS

Este capítulo está organizado em quatro seções. A primeira mostra conceitos básicos relacionados à área de textura sólida e algoritmos geométricos. Uma visão geral dentro do campo de pesquisa de textura sólida é apresentada na segunda seção. A terceira seção aborda trabalhos selecionados de sistemas que envolvem simuladores cirúrgicos virtuais. Finalmente, a quarta seção relaciona o material apresentado com aquele efetivamente explorado no método utilizado.

2.1 Conceitos Básicos

Nesta seção são apresentados conceitos fundamentais divididos em duas áreas de grande importância que constituem o domínio do trabalho. Primeiramente, são expostos o conceito de algoritmos geométricos e algumas definições importantes para a compreensão das etapas da técnica desenvolvida. Na subseção seguinte, o conceito de textura sólida é definido, revisando inicialmente o mapeamento de texturas clássicas bidimensionais e a motivação pela busca de novos tipos de texturização em 3D.

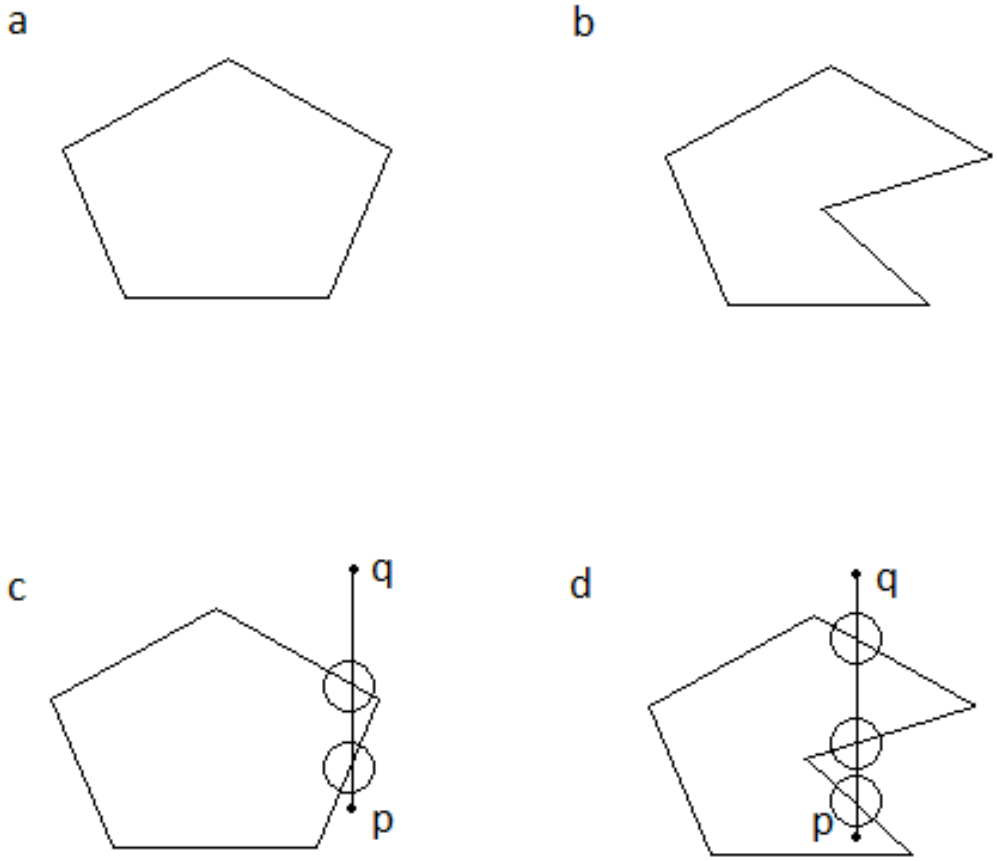
2.1.1 Algoritmos Geométricos

O ramo de algoritmos geométricos e geometria computacional teve início na década de 70, tendo por objetivo estudar problemas geométricos a fim de encontrar algoritmos e estruturas de dados que apresentem um bom desempenho à medida que problemas escalam de tamanho. O interesse é resolver problemas de tal maneira que o menor número possível de operações elementares seja aplicado, melhorando o desempenho durante o cálculo das soluções. Em geral, problemas dessa área recebem objetos geométricos como entrada e precisam resultar em um objeto geométrico modificado de acordo com um ou mais critérios definidos. Além de CG, outras áreas da computação também lançam mão de algoritmos geométricos, como processamento de imagens, visão computacional, robótica, VLSI e CAD, por exemplo.

Um polígono $P \in R^2$ no espaço euclidiano é dito convexo quando, para cada par de pontos p_i, p_j do interior de P, o segmento de reta $r(p_i, p_j)$ também está dentro de P. São ditos não-convexos ou côncavos polígonos que não satisfazem a propriedade acima. *Point-in-polygon* é uma técnica usada para detectar se um ponto se encontra dentro de um polígono. Seja um polígono P com um conjunto V de vértices e A de arestas formando o seu contorno, p um ponto

qualquer e q um ponto suficientemente distante de P. Um ponto p está dentro de P se o número de intersecções do segmento de reta $r(p, q)$ com cada aresta $a \in A$ é ímpar. Caso contrário, o ponto se encontra fora de P. O algoritmo funciona tanto para polígonos convexos quanto para côncavos. A Figura 2.1 mostra um exemplo de polígonos convexos e não-convexos, seguido de uma ilustração da técnica de *point-in-polygon*, com destaque para as intersecções dado um segmento de reta.

Figura 2.1 – Exemplo de um polígono convexo (a) e não-convexo (b). Ilustração da técnica de *point-in-polygon* para polígonos convexos (c) e não-convexos (d).

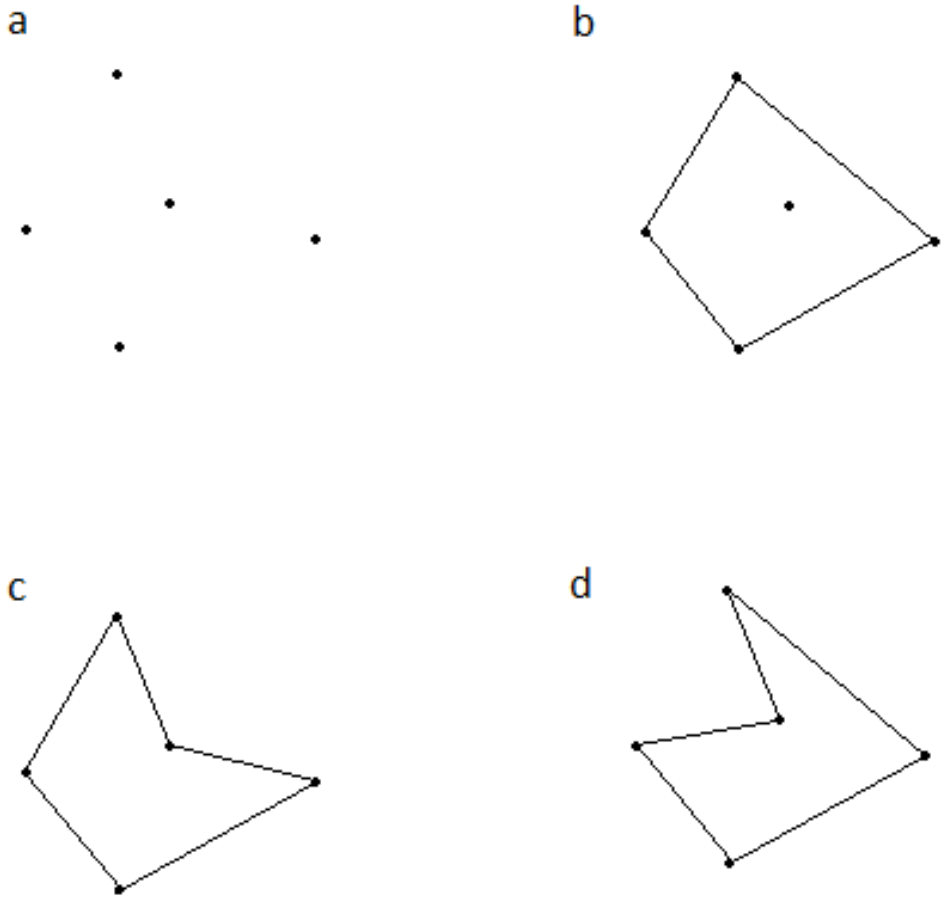


Fonte: Compilado pelo autor.

Uma envoltória convexa (*convex hull*) de um conjunto P de pontos no espaço euclidiano é definida como o menor conjunto convexo que contém P. Analogamente, uma envoltória côncava (*concave hull*) de um conjunto P de pontos no espaço euclidiano é descrita como o menor conjunto côncavo que contém P. Informalmente, o conjunto de pontos resultante de uma envoltória convexa é formado pelos pontos mais externos, de forma que envolvam todos os demais em seu interior. Para uma envoltória côncava o algoritmo não é absoluto, uma vez que não há, a priori, informações do objeto côncavo desejado. A Figura 2.2 mostra exemplos

de envoltórias convexas e côncavas.

Figura 2.2 – Conjunto de pontos de entrada (a), envoltória convexa de a (b), um exemplo de envoltória côncava para a (c) e outro exemplo de envoltória côncava para a (d).



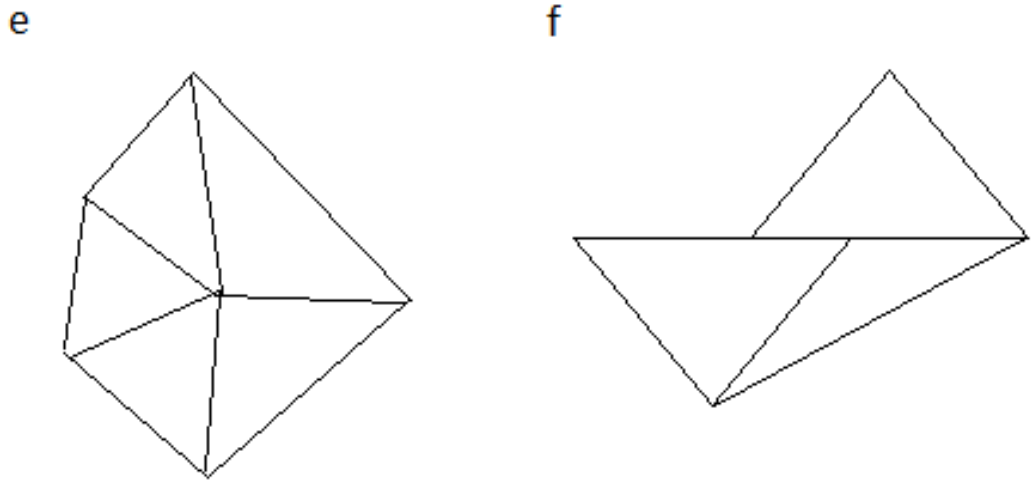
Fonte: Compilado pelo autor.

Modelos tridimensionais em CG geralmente são compostos por malhas triangulares, cuja formação é a primitiva mais simples, o triângulo. Por isso, algoritmos que lidam com triângulos podem ser otimizados, como o uso de interpolação linear durante a rasterização. Caso o objeto seja composto por outros tipos de primitivas, o hardware gráfico se encarrega de transformar cada um dos polígonos em triângulos para o processamento, já que qualquer objeto pode ser subdividido nessas figuras geométricas.

Uma diagonal D de um polígono P constituído por vértices V é definida como um segmento de reta $r(V_i, V_j)$ que pertence ao interior de P.

Genericamente, uma triangulação consiste, portanto, em dividir um plano em simplexos, que são extensões de triângulos em outras dimensões. Para o caso tridimensional, triangulação é a decomposição de P em triângulos por um conjunto maximal de diagonais não-interceptáveis. A Figura 2.3 mostra um exemplo de triangulação e não-triangulação, respectivamente.

Figura 2.3 – Exemplo de um polígono constituindo uma triangulação (a) e uma coleção qualquer (b).



Fonte: Compilado pelo autor.

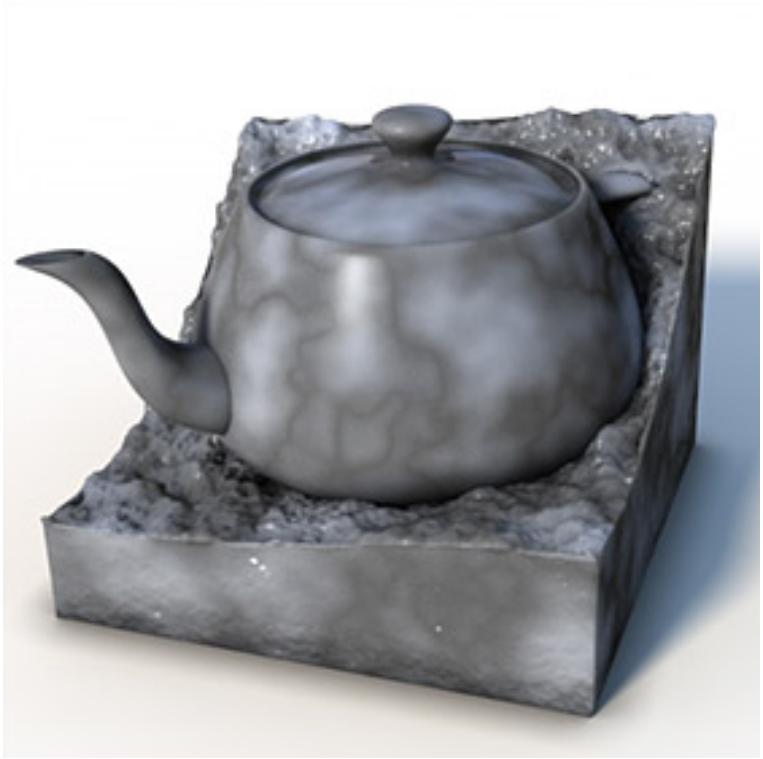
Existem diversas técnicas de triangulação, dependendo do objetivo que se deseja alcançar. A quantidade de triângulos está diretamente associada ao número de vértices presentes na sua envoltória convexa. Seja t o número de triângulos de uma triangulação T composta por n vértices e m vértices da sua respectiva evoltória convexa, o número de triângulos t é dado por $t = 2n - m - 2$. A qualidade de triangulações está intimamente ligada aos ângulos de cada um dos triângulos gerados. Dado um vetor de ângulos $\alpha(T)$ de uma triangulação T em ordem crescente, pode-se dizer que, de maneira geral, uma triangulação $T1$ é melhor que $T2$ se $\alpha(T1) > \alpha(T2)$. Portanto, triangulações melhores tendem a apresentar alguma heurística com a finalidade de tornar maiores os ângulos de cada triângulo gerado pela triangulação.

2.1.2 Textura Sólida

Em CG, texturas são imagens aplicadas sobre modelos 3D a fim de contribuir com informações de sua aparência, separando-as de sua forma geométrica. As primeiras aplicações tridimensionais foram introduzidas por Edwin Catmull em 1974 (CATMULL, 1974), consistindo em um método para mapear pixels em uma superfície tridimensional a fim de tornar os resultados mais realísticos. Com o passar do tempo, novos tipos de mapeamento surgiram para simular um fotorrealismo em tempo real, evitando cálculos custosos de iluminação e reduzindo o número de polígonos. Entretanto, esse mapeamento entre objetos 3D e texturas 2D pode gerar uma distorção dependendo da forma e topologia do modelo, já que essa correspondência entre diferentes dimensões não é direta. Consequentemente, uma parametrização planar ótima, adaptável para qualquer objeto, ainda é uma questão a ser estudada.

Textura sólida pode ser definida como um procedimento em que a função de geração de textura é aplicada diretamente sobre o espaço R^3, fazendo com que cada ponto do interior do modelo tenha um mapeamento direto com a textura criada. Esse conceito foi inicialmente definido em 1985 (PERLIN, Jul. 1985) (PEACHEY, Jul. 1985). Com essa técnica, a cor de cada ponto do exterior ou interior do objeto é calculada por uma função que texturiza aquele ponto diretamente de acordo com a posição 3D espacial, sem sofrer eventuais distorções ocorridas de acordo com a topologia e a forma dos objetos. Em casos gerais, portanto, uma mesma textura sólida pode ser utilizada para gerar cores de qualquer objeto. Entretanto, assegurar uma textura coerente em qualquer sentido do interior do objeto não é uma tarefa simples. A Figura 2.4 mostra abstratamente o conceito de textura sólida.

Figura 2.4 – Exemplificação do conceito de textura sólida, onde um *teapot* é esculpido a partir de um bloco de textura 3D.



Fonte: http://modo.docs.thefoundry.co.uk.

2.2 Trabalhos Relacionados em Textura Sólida

Esta seção revisa alguns trabalhos selecionados sobre textura sólida, sem ter por objetivo realizar um levantamento completo dessa ampla área de pesquisa. Recentemente, foi realizado um survey (PIETRONI et al., Jan. 2010) abordando esse tema. Esse trabalho propôs uma classificaçao de técnicas de textura sólida como *boundary-independent* e *boundary-dependent*. A

primeira classificação está fundamentada na sintetização do volume de um cubo colorido sem limitar-se à forma do objeto durante o mapeamento de cores. Nestes casos, a cor de cada ponto é computada a partir de uma função ou síntese sobre texturas, com cada ponto não limitado à forma do objeto. O *rendering* de cortes arbitrários parece plausível, como se o objeto estivesse encravado no espaço de texturização 3D, sendo esculpido como um todo. Essa classe de soluções apresenta dificuldades para objetos não homogêneos, já que eles requerem alguns tratamentos específicos para gerar suas estruturas internas. Por outro lado, a segunda classificação leva em conta o volume do objeto como domínio, criando uma associação com o interior e a borda para orientar a textura de acordo com a forma da superfície. A textura, portanto, segue a forma do objeto, usando essa informação para guiar o processo.

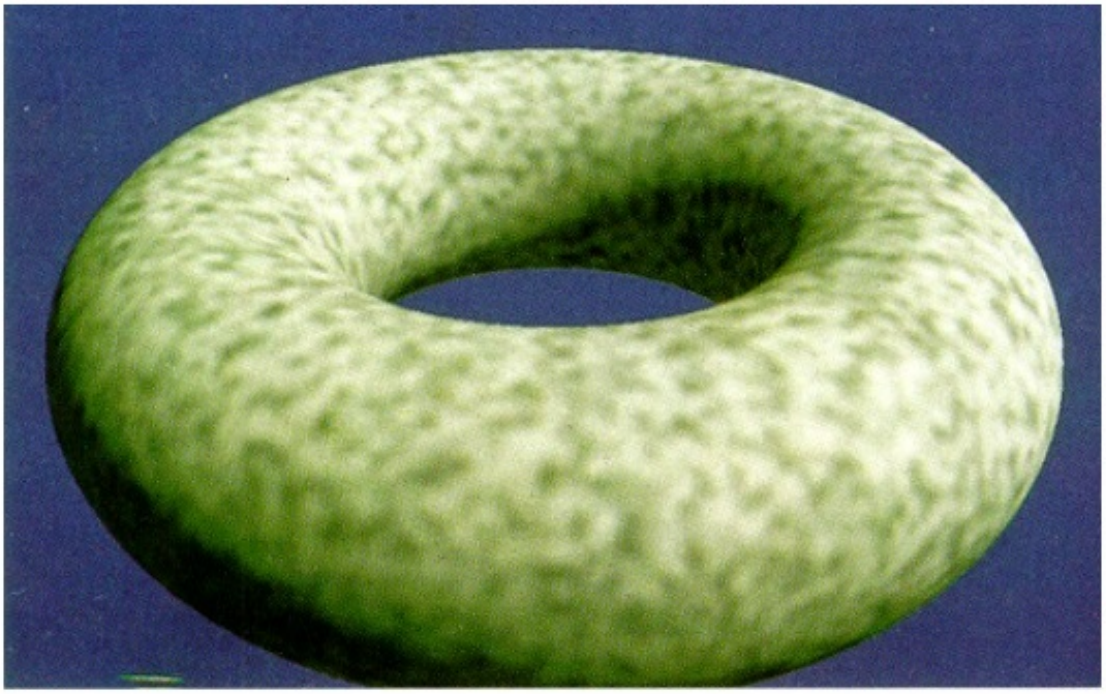
2.2.1 *Técnicas de Textura Sólida Boundary-independent*

O conceito de textura sólida (PERLIN, Jul. 1985) (PEACHEY, Jul. 1985), foi primeiramente introduzido como uma nova forma de extrapolar a representação, até aquele momento focada no rendering externo de objetos, para as estruturas internas. É nomeada *space function* (PERLIN, Jul. 1985) qualquer função cujo domínio pertence inteiramente ao espaço tridimensional (x, y, z). Nesse contexto, a cor resultante vem de uma função baseada na posição 3D e em um conjunto de parâmetros definidos. A Figura 2.5 apresenta exemplo de uma imagem produzida usando essa técnica aplicada a um *torus*. Outro trabalho (PEACHEY, Jul. 1985) também introduziu o conceito de textura sólida.

A ideia de obter informação interna de objetos teve seguimento também na área de análise espectral (GHAZANFARPOUR; DISCHLER, May/June 1995). Dada uma textura de entrada, suas informações espectrais são extraídas a partir de uma FFT (*Fast Fourier Transform*), obtendo as chamadas funções *basis* e *noise* e gerando, a partir daí, a textura sólida proceduralmente. O trabalho mostra resultados direcionados à simulação de texturas naturais, semelhantes às abordadas por Perlin. No ano seguinte, uma extensão da pesquisa que utiliza duas ou três imagens correspondendo a diferentes fatias de um bloco de textura 3D foi proposta pelos mesmos autores (GHAZANFARPOUR; DISCHLER, Aug. 1996). Mais tarde, outro trabalho obteve informação interna de objetos contendo a síntese de texturas aplicadas em visualizações 2D ortogonais (DISCHLER; GHAZANFARPOUR; FREYDIER, Sep. 1998). A técnica utiliza uma abordagem híbrida com análise do espectro e do histograma para as imagens de entrada, representando diferentes pontos de vistas ortogonais das texturas.

Um método adaptou um processo de síntese de texturas 2D chamado de *optimization-based*

Figura 2.5 – Textura sólida aplicada sobre uma superfície gerada por uma *noise function* (PERLIN, Jul. 1985).



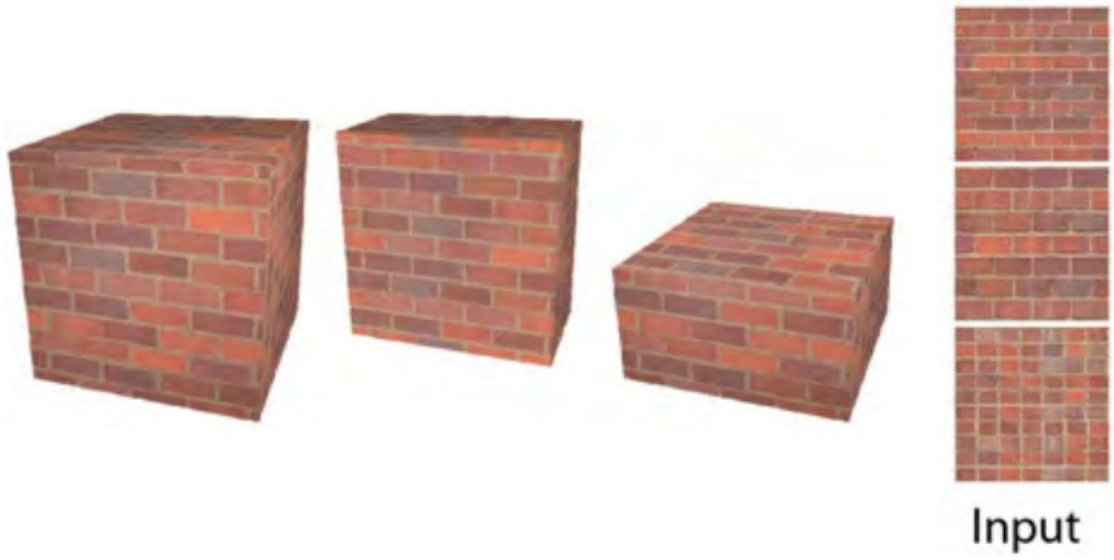
Fonte: (PERLIN, Jul. 1985).

(KOPF et al., July, 2007). A ideia principal é estender técnicas de otimização de texturas para sintetizar texturas sólidas e integrá-las a abordagens não paramétricas com *histogram matching*, fazendo com que o sólido sintetizado se pareça com a imagem 2D pela minimização global de sua função de energia. Outro procedimento desenvolvido (DONG et al., June 2008) introduz um novo algoritmo para sintetizar texturas sólidas com a habilidade de restringir a síntese para um subconjunto de voxels, forçando um determinismo espacial. A ideia principal é pré-computar todos os candidatos tridimensionais em um pré-processamento por meio de uma intercalação de três vizinhos bidimensionais das imagens de entrada. Usando uma seleção apropriada, os candidatos pré-computados melhoraram a eficiência da síntese, reduzindo o número de interações. A localidade provida pelo algoritmo permite sintetizar um conjunto de voxels próximos de uma superfície visível, em vez de sintetizar todo o volume como outros métodos, melhorando consequentemente o desempenho. Uma implementação que utiliza paralelismo diretamente em GPU foi proposta pelos autores, provendo uma síntese em tempo real de objetos por meio de cortes em simulações interativas de cortes e fragmentações.

Recentemente, Du e colegas (DU; HU; MARTIN, Mar. 2013) introduziram uma nova técnica para sintetizar um tipo específico de textura sólida, consistindo de padrões regulares ou semi-regulares. Esses padrões podem ser observados não apenas em materiais da natureza,

como folhas, mas também nos criados pelo homem, como uma parede de tijolos. Dadas imagens 2D das quais provém planos de cortes do volume de textura desejado, novas texturas 3D são sintetizadas. A colocação espacial de partículas 3D são obtidas por informações das imagens de entrada. O método inicia a partir de três imagens ortogonais, com os planos de corte correspondendo à textura desejada. As texturas devem consistir de partículas discretas como em uma matriz, seguindo um padrão regular ou semi-regular. Juntamente com cada imagem, uma máscara binária indica o lugar dos cortes das partículas a fim de serem reconstruídas, estimando as relações de vizinhança. Por fim, as partículas são adicionadas iterativamente ao volume texturizado, produzindo um interior como saída. Como contraponto, a técnica se restringe a imagens regulares, além de requerer intervenção do usuário para padrões semi-regulares. A Figura 2.6 mostra resultados obtidos com essa técnica.

Figura 2.6 – Resultado da técnica (DU; HU; MARTIN, Mar. 2013) em diferentes pontos de vista dadas três imagens de entrada.



Fonte: (DU; HU; MARTIN, Mar. 2013).

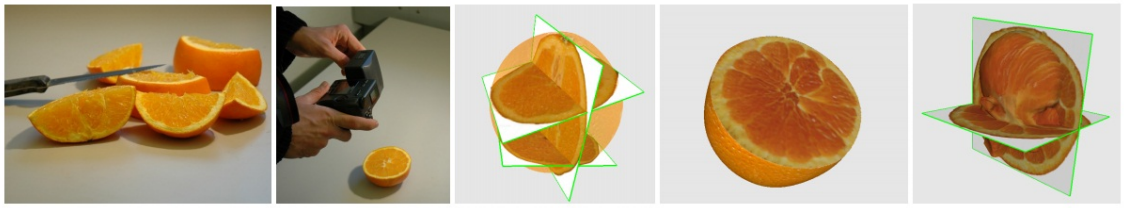
2.2.2 *Técnicas de Textura Sólida Boundary-dependent*

A técnica introduzida por Cutler et al (CUTLER et al., July 2002) é considerada a primeira abordagem sobre textura sólida classificadas como *boundary-dependent*. O interior de um modelo é definido por uma linguagem script, possibilitando a criação de texturas em camadas. A contribuição principal do método é uma abordagem procedural para continuamente construir e alterar geometrias sólidas complexas. Entretanto, poucos exemplos foram dados, sem grande variação no interior.

Owada e colegas (OWADA et al., Aug. 2004) propuseram um novo método que consiste na especificação do interior de um objeto pelo uso de uma *browsing interface* e *modeling interface*. A primeira interface é um visualizador de modelos, permitindo ao usuário observar as estruturas internas de um dado objeto. Desenhando linhas 2D no sistema de modo a setar a direção que o objeto deve dividir, o usuário gera uma projeção destes caminhos na malha 3D, definindo os planos de corte. Após, a superfície interna é retriangulada de acordo com a direção definida. Por outro lado, a *modeling interface* provê uma forma de especificar uma estrutura interna adequada para o modelo, escolhendo algumas imagens de referência, de acordo com determinados tipos de texturas. As *isotropic textures* não dependem da superfície, sendo sintetizadas no espaço paramétrico de um plano de corte. As *layered textures* requerem um campo de distância 2D para ser calculado em cada secção de corte, usando um algoritmo de síntese de texturas 2D com variações de acordo com um campo de distância. Por fim, as *oriented textures* seguem uma nova orientação definida pelo usuário, sendo sintetizadas pela translação da imagem de textura ao longo da direção do eixo y, aplicando uma técnica de comparação entre vizinhos para sintetizar as cores. Este método produz bons resultados, mas requer trabalho excessivo do usuário, além de possuir um poder limitado de criação.

Pietroni e colegas (PIETRONI et al., Sep. 2007) criaram uma nova técnica que consiste em usar poucas imagens de referência alocadas em uma estrutura 3D em torno do objeto, do qual provê *rendering* de cortes arbitrários. As imagens são deformadas para que se ajustem aos limites do objeto e a síntese é feita em tempo real usando uma técnica de morphing entre as secções de corte. Para armazenar e procurar pelos planos mais próximos, uma árvore BSP é usada, dividindo o objeto entre regiões delimitadas pela árvore. Finalmente, cada voxel a ser sintetizado é projetado sobre a região correspondente da BSP, gerando uma cor de saída obtida pelo algoritmo de *morphing* utilizado. O usuário precisa definir a curva de interpolação na hora de buscar informações do pixel correspondente, necessitando de um conhecimento prévio sobre os modelos tridimensionais usados. A Figura 2.7 apresenta uma visão geral do trabalho proposto.

Figura 2.7 – Visão geral da técnica apresentada por Pietroni e colegas (PIETRONI et al., Sep. 2007).



Fonte: (PIETRONI et al., Sep. 2007).

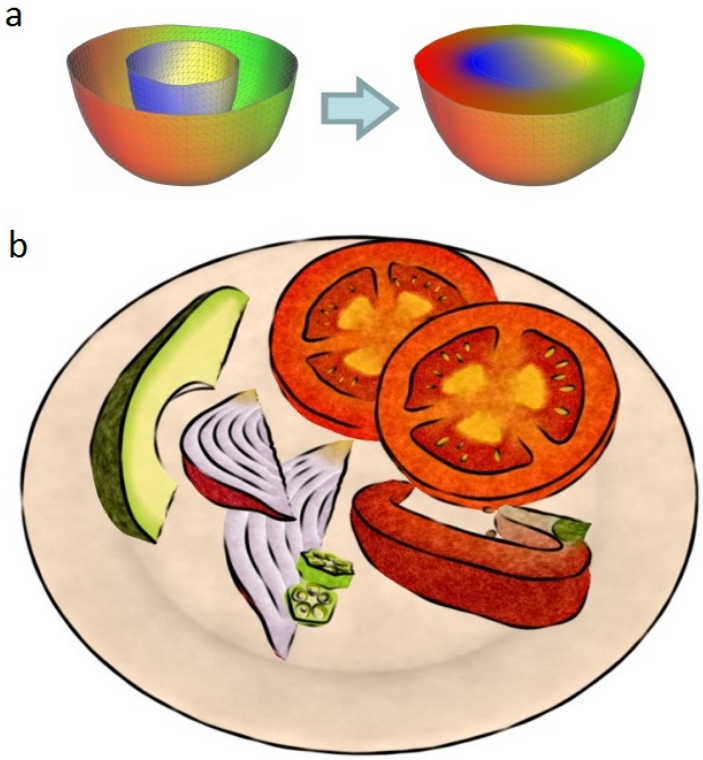
Um conceito antigo chamado *lapped textures* (PRAUN; FINKELSTEIN; HOPPE, July

2000) foi adaptado para o contexto de texturas sólidas (TAKAYAMA et al., Aug. 2008). A ideia principal da técnica é localizar regiões interessantes da imagem 2D e colocá-las sobre a superfície, cobrindo toda a parte externa do modelo. A coleção de sobreposições de pedaços dessa textura é chamada de *lapped texture*. Takayama e colegas ampliaram o conceito para preencher volumes em vez de cobrir apenas a superfície do modelo. O sistema coloca pedaços de textura de acordo com um campo tensorial suave como três campos de vetores ortogonais pelo volume para organizar os pedaços sólidos, requerendo apenas as coordenadas da textura 3D de cada vértice da malha para serem armazenadas. O método também expande a original levando em conta texturas não homogêneas durante a criação de modelos sólidos.

No mesmo ano, foi proposta uma nova técnica chamada *volume painter* (OWADA et al., 2008), um sistema baseado em esboços para projetar volumes a partir de desenhos. Uma superfície modelo que pode ser dividida pelo usuário possibilita uma interação usando pinturas para preencher volumetricamente o espaço 3D. O usuário divide o espaço em regiões, onde cada uma corresponde a uma textura uniforme. As texturas são geradas pela distribuição de partículas nas regiões, limitadas pelo objeto. Esse trabalho apresenta um novo editor de modelos sólidos, porém requer muita intervenção do usuário, surgindo como uma ferramenta de ajuda ao *design* de estruturas internas de objetos volumétricos não fotorrealistas.

Mais recentemente, uma abordagem trouxe um novo conceito chamado *diffusion surfaces* (TAKAYAMA et al., Dec. 2010) em resposta à dificuldade de modelar estruturas que requerem uma variação suave entre as cores da estrutura interna, como frutas e vegetais. *Diffusion surfaces* são superfícies tridimensionais com cores definidas na frente e atrás dessas superfícies, fazendo com que as cores internas geradas dentro desse volume sejam obtidas pela difusão das cores de superfícies vizinhas. Em vez de usar uma forma conhecida para computar as cores difusas pela resolução de uma equação de Poisson, o que seria custoso, o método escolhido é uma interpolação local em regiões definidas pelo usuário por meio de secções de corte. Para a interpolação é usada uma versão modificada do algoritmo *positive mean value coordinates*, evitando malhas volumétricas. Os resultados focaram em objetos que possuem algum tipo de simetria rotacional, habilitando algumas variações randômicas de população por tipos de modelos. A técnica requer ainda muita intervenção do usuário para criação de modelos. A Figura 2.8 ilustra o conceito de *diffusion surfaces*, além de alguns resultados.

Figura 2.8 – Ilustração de *diffusion surfaces* (a), seguida de alguns resultados da técnica (b).



Fonte: (TAKAYAMA et al., Dec. 2010).

2.3 Trabalhos Relacionados em Simuladores Cirúrgicos Virtuais

Esta seção aborda alguns trabalhos dentro da vasta área de simuladores cirúrgicos virtuais, organizados ascendentemente conforme ano de publicação. Estes sistemas ganharam força principalmente na década de 90, com pesquisas ampliando as áreas de simulação física, interação e realidade virtual. Utilizando *solid modeling technology* (SATOH et al., May. 1992), um novo sistema de simulação cirúrgica da área de operação craniofacial foi proposto, permitindo construir modelos usando essa nova abordagem em relação à modelagem por *voxels* empregada em trabalhos anteriores. Essa característica permite representar mais facilmente curvas suavizadas, gerando consequentemente modelos com formas mais realistas para a aplicação médica.

Em 1993, foi apresentado um simulador cirúrgico (SATAVA, May/June 1993) que utiliza realidade virtual com um HMD (*Head Mounted Display*) para visualização e *Data Glove* para interação. O sistema em si é considerado tanto um visualizador quanto um simulador de órgãos da região do abdômen. Apesar de dispor de um *rendering* não realístico e uma simulação simplificada, sem qualquer tipo de retorno háptico, esse trabalho serviu de base para que sistemas posteriores pudessem estender a ideia. A fim de apresentar um simulador cirúrgico relacionado à área craniofacial, no ano seguinte foi proposto um sistema (DELINGETTE et al., Sep. 1994)

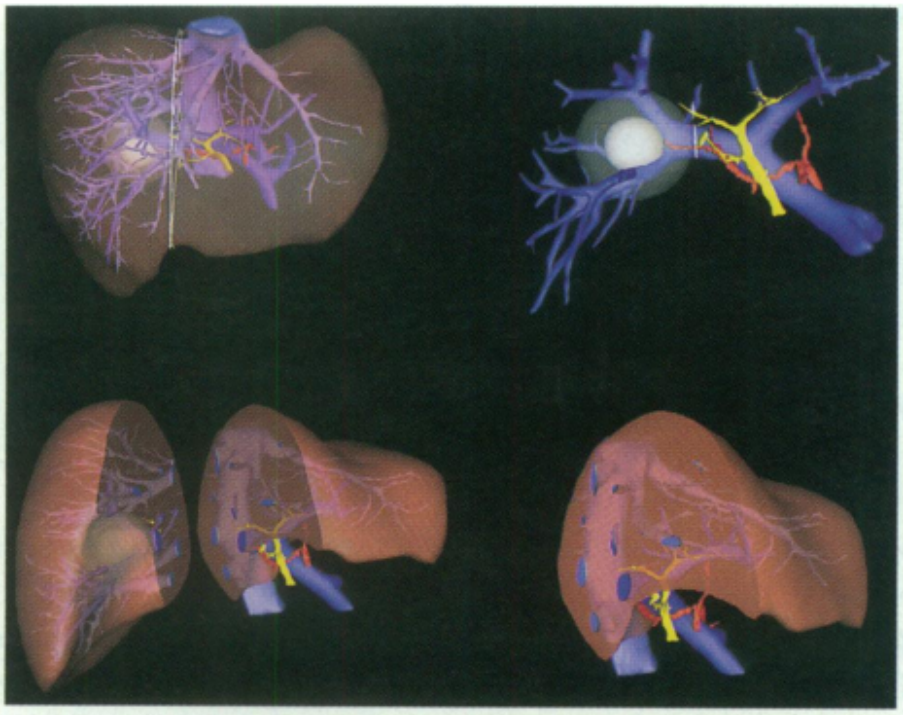
que utiliza uma representação chamada *simplex meshes* para superfícies e volumes. Essa nova abordagem permite que sejam feitas mudanças locais na conectividade das malhas, ponto fundamental para facilitar a deformação e suavização dos modelos, característica importante para simular esse tipo de cirurgia.

Com a ampliação de simuladores, surgiu a necessidade de uma formalização mais teórica sobre as características que esses sistemas deveriam possuir. Dawson e Kaufman (DAWSON; KAUFMAN, Mar. 1998) definiram três critérios fundamentais para que simuladores cirúrgicos possam ser considerados como uma ferramenta: devem ser validados, realísticos e acessíveis. A validação envolve uma certificação de que todo o retorno oferecido pela simulação esteja o mais próximo possível do procedimento cirúrgico, enquanto a realidade está mais relacionada com a aparência, referindo-se ao nível de realismo dos objetos e das interações em questão. Por fim, a acessibilidade preocupa-se em fazer com que o custo dos dispositivos permita que eles sejam difundidos. Portanto, segundo os autores, sistemas devem levar em conta esses três critérios durante seu desenvolvimento para que sejam válidos.

Também neste mesmo ano, um novo trabalho (MARESCAUX et al., Nov. 1998) propôs um simulador para cirurgias do fígado. A anatomia interna de um fígado foi reconstruída, apresentando os vasos, tumores e o tecido externo. Um local de corte de acordo com a região em que o tumor é encontrado permite dividir o fígado em duas partes, além de demonstrar uma deformação na malha com retorno de força. Entretanto, o método teve seu maior foco na visualização para aprendizagem cirúrgica, sem visar um nível de realismo esperado de um simulador. A Figura 2.9 apresenta diferentes imagens na visualização desse sistema, simulando uma hepatectomia. Técnicas de corte e deformação em tempo real também foram direcionadas para a treino e simulação de cirurgias (DELINGETTE; COTIN; AYACHE, May 1999). Nesta abordagem, três modelos baseados em elementos finitos e elasticidade linear foram propostos, onde o terceiro é constituído por uma abordagem híbrida dos dois primeiros, procurando equilibrar o balanço entre tempo de computação e realismo visual. Os resultados mostram uma interação de um instrumento cirúrgico virtual com uma malha triangular de um fígado. Em outra pesquisa de Dawson (DAWSON et al., Dec. 2000) foi proposto um novo simulador para treinamento de cardiologia intervencionista, um ramo da cardiologia que realiza tratamentos e diagnósticos por meio de catéteres no sistema cardiovascular. Esse método apresenta sistema háptico, modelagem física das estruturas tridimensionais e um feedback visual semelhante ao que o cardiologista observa no monitor durante um procedimento real.

Um novo simulador para cirurgias do intestino que permite colisão e visualização em tempo real (RAGHUPATHI et al., Nov./Dec. 2004) lançou mão de uma técnica de *skinning* adaptada

Figura 2.9 – Hepatectomia virtual direita visualizada a partir de um plano de corte baseado na localização do tumor (MARESCAUX et al., Nov. 1998).



Fonte: (MARESCAUX et al., Nov. 1998).

para cilindros. Pelo fato de o intestino poder ser aproximado por um cilindro, uma nova técnica de colisão foi desenvolvida para esta primitiva geométrica, já que o órgão interage não só com instrumentos, mas também com ele mesmo durante uma cirurgia real. Outro trabalho específico para cirurgias hepáticas foi apresentado em 2005 (DELINGETTE; AYACHE, Feb. 2005), detalhando diversas etapas que esses simuladores devem abordar. Particularmente, o modelo geométrico é obtido a partir da adaptação de uma malha genérica de fígado com imagens de tomografia computadorizada, incluindo veias e possíveis lesões. Na parte de renderização, foram aplicados métodos para tornar a texturização e iluminação mais fiéis, além de tentar reproduzir as marcas causadas por um bisturi elétrico, instrumento usado nessas cirurgias. Em 2007, um novo trabalho (LIM; JIN; DE, Dec. 2007) apresentou técnicas para melhorar o realismo dos cenários nas simulações a partir de imagens e algoritmos para cortes cirúrgicos. Essa metodologia para corte de malhas consegue simular cortes progressivos com uma técnica chamada *node-snapping*, aumentando minimamente o número de novas primitivas criadas. Além disso, experimentos aplicados a animais foram feitos para verificar características dos cortes. Outra técnica envolvendo deformação e cortes de tecidos em tempo real exibiu resultados dentro da área médica, utilizando implementações em GPU para melhora de desempenho (COURTECUISSE et al., Sep. 2010). O procedimento de corte do objeto envolve modificar a malha e atualizar o sistema, juntamente com as matrizes usadas no algoritmo. As mudanças topológicas

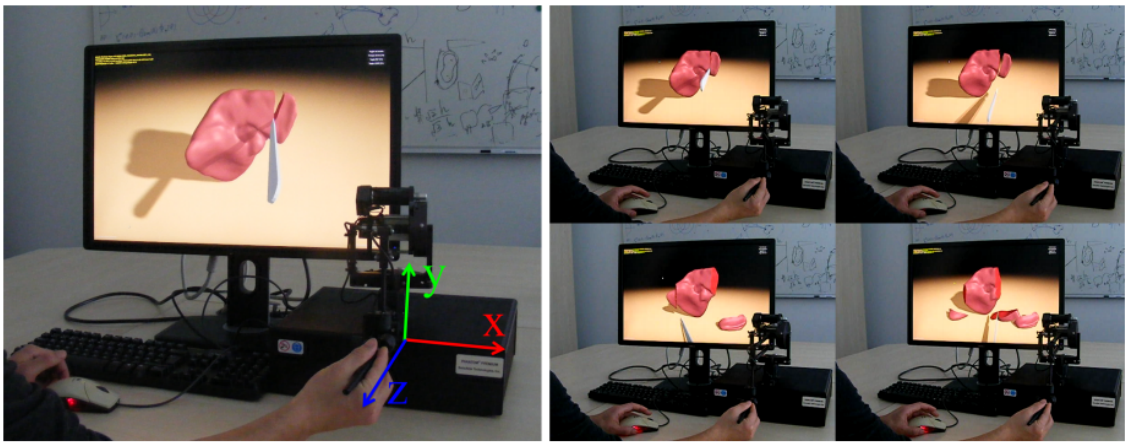
que precisam ser realizadas consistem em remoção, subdivisão e adição dos elementos. Por fim, um sistema de simulação de hepatectomia foi desenvolvido, contendo essas técnicas.

Trabalhos mais recentes compreendem também o desenvolvimento e aperfeiçoamento de simuladores. Concebido para intervenção intracardíaca utilizando realidade virtual (CHIANG et al., Apr. 2012), pela primeira vez é abordado um simulador para mapeamento do miocárdio e injeção intracardíaca. O trabalho incorpora elementos de visualização, modelagem e interação dinâmica visto em outros simuladores. Os objetos modelados foram extraídos a partir de dados volumétricos obtidos por meio de uma tomografia computadorizada, deformada posteriormente usando *free-form deformation* para sensação de movimento. A fim de tornar a simulação em tempo real, os modelos são subdivididos em *voxels* para que o sistema possa ser paralelizado na renderização do volume, além de simplificar o procedimento de detecção de colisão do cateter com toda a estrutura do órgão. A reconstrução da superfície do coração é feita à medida que o miocárdio vai sendo mapeado, por meio de pontos de contato. Apesar de incorporar uma referência bem mais completa das estruturas internas do sistema cardíaco, o foco é dado ao sistema de colisão, assim como ocorre no procedimento cirúrgico. Um novo sistema para treinamento de laparoscopia foi proposto em 2013, utilizando informações obtidas por um especialista para orientar usuários na simulação (TAGAWA et al., Jan. 2013). Durante o treinamento, dois monitores são dispostos verticalmente adjacentes, onde um deles apresenta o simulador e outro a visão em primeira pessoa do procedimento feito pelo instrutor no mesmo sistema. Além disso, um retorno de força é disposto para guiar os estudantes, baseado no que foi usado pelo especialista. Apenas uma primeira etapa da cirurgia foi escolhida para o experimento. Esse sistema utilizou um modelo de fígado e um sistema massa-mola para deformação, além de usar computação paralela em GPU.

Com características semelhantes aos trabalhos anteriores, um simulador de cirurgias para o cérebro humano (ECHEGARAY et al., May/June 2014) foi recentemente apresentado. Nesse sistema, tanto a região encefálica quanto o crânio são abordados. Os modelos usados são obtidos a partir de informações extraídas de CT (*Computed Tomography*) e MRI (*Magnetic Resonance Imaging*). O *rendering* do crânio é feito utilizando *volumetric isosurface*, provendo alta qualidade sem requerer pré-processamento, enquanto o do cérebro usa FVR (*Full Volume Rendering*). Envolvendo um tipo específico de cirurgia chamado ESS (*Endoscopic Sinus Surgery*), outro simulador virtual foi abordado recentemente (VARSHNEY et al., July/Aug. 2014). O sistema permite ao usuário aplicar alguns procedimentos dessa cirurgia em um modelo nasal. Desenvolvido por especialistas e contendo grande parte das características que os outros também possuem, este pode ser considerado o simulador mais completo dessa área até então.

Voltando à área de cortes de malhas, um novo trabalho (WU; WESTERMANN; DICK, Feb. 2014) propôs uma abordagem fisicamente precisa na simulação de cortes em tempo real, apresentando efeitos na área de simuladores cirúrgicos. O método aumenta a resolução de elementos finitos, produzindo melhores efeitos e mantendo tanto suavidade quanto retorno háptico. No entanto, não é detalhado o modo como é feita a textura interna dos cortes. Foi desenvolvido então um simulador que permite ao usuário cortar um fígado com um bisturi interativamente, conforme mostra a Figura 2.10. Ainda em 2014, os mesmos autores publicaram nessa área de pesquisa um *survey* (WU; WESTERMANN; DICK, Apr. 2014) contendo diversos trabalhos relacionados a esse assunto.

Figura 2.10 – *Setup* do experimento (esquerda) e sequência de cortes no ambiente virtual pelo usuário (direita) (WU; WESTERMANN; DICK, Feb. 2014).



Fonte: (WU; WESTERMANN; DICK, Feb. 2014).

2.4 Discussão

A primeira parte deste capítulo abordou alguns conceitos fundamentais da área de algoritmos geométricos necessários para o entendimento da técnica implementada, além de mostrar algumas definições básicas relacionadas à texturização tradicional em CG e como foi feita a evolução para texturas sólidas. Na sequência, foram revisados trabalhos de texturas sólidas julgados relevantes para revisão bibliográfica, separados em dois grupos. Um subconjunto da vasta área de pesquisas relacionadas com simuladores cirúrgicos foi também abordado nesse capítulo.

O trabalho aqui apresentado enquadra-se como um sistema de visualização envolvendo técnicas de algoritmos geométricos para cortes em malhas tridimensionais com estruturas internas, preenchendo o interior com cores por meio de uma técnica de textura sólida desenvolvida, que

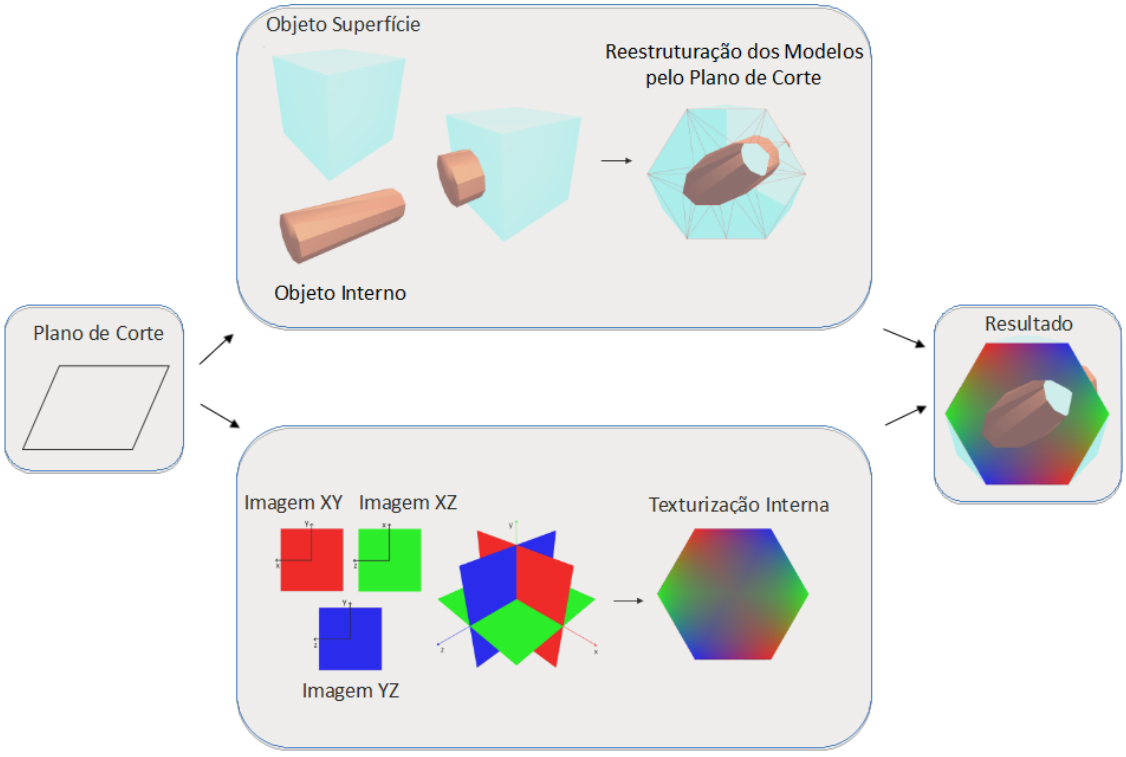
se enquadra na classificação *boundary-independent*. O estudo de caso envolve cortes aplicados em um órgão humano, justificando uma revisão também em trabalhos relacionados com simuladores cirúrgicos virtuais.

3 METODOLOGIA

Este capítulo dedica-se a apresentar a técnica utilizada para construir o sistema de visualização do interior de objetos com estruturas internas proposto nesse trabalho. Dados modelos tridimensionais, imagens e um plano de corte, busca-se retornar a visualização realista do corte realizado, levando em consideração as estruturas internas e a textura. Assim, a intersecção do plano com os objetos que representam as superfícies permite que a textura sólida seja calculada por uma interpolação a partir das imagens de entrada. Caso o plano intersecte com os objetos que descrevem as estruturas internas, toda a zona de intersecção é definida como *hole* e a textura nessas partes é desconsiderada.

O capítulo está organizado em duas seções. A primeira descreve uma série de passos para reestruturar os modelos a partir do corte definido pelo plano, enquanto a segunda expõe o método de texturização para a área definida na primeira seção. A Figura 3.1 apresenta o pipeline geral da técnica. A partir dos modelos e das imagens de entrada, o plano de corte pode ser manipulado iterativamente, gerando novas saídas de acordo com a orientação e localização espacial.

Figura 3.1 – Pipeline do método proposto.

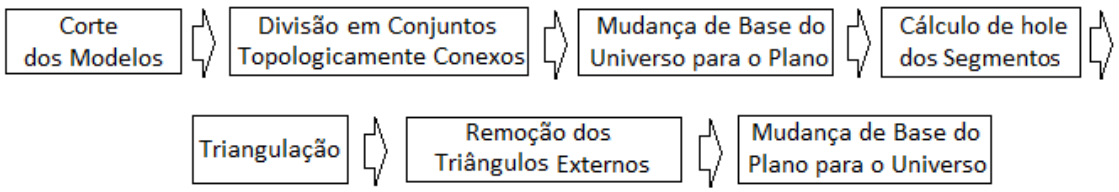


Fonte: Compilado pelo autor.

3.1 Reestruturação dos Modelos pelo Plano de Corte

Esta seção descreve a sequência de passos para a reestruturação dos objetos recebidos como entrada dado um plano de corte. No final do processamento deste módulo, cada um dos objetos é seccionado sobre a intersecção do modelo com o plano de corte, retriangulando os modelos que representam a superfície e definindo os modelos que descrevem as estruturas internas como *hole*. Esse algoritmo adapta-se a diferentes tipos de objetos formados por malhas triangulares, lidando com modelos convexos, côncavos e até mesmo desconexos. Os passos desse módulo estão esquematizados na Figura 3.2.

Figura 3.2 – Pipeline do módulo de reestruturação dos modelos pelo plano de corte.



Fonte: Compilado pelo autor.

3.1.1 Corte dos Modelos

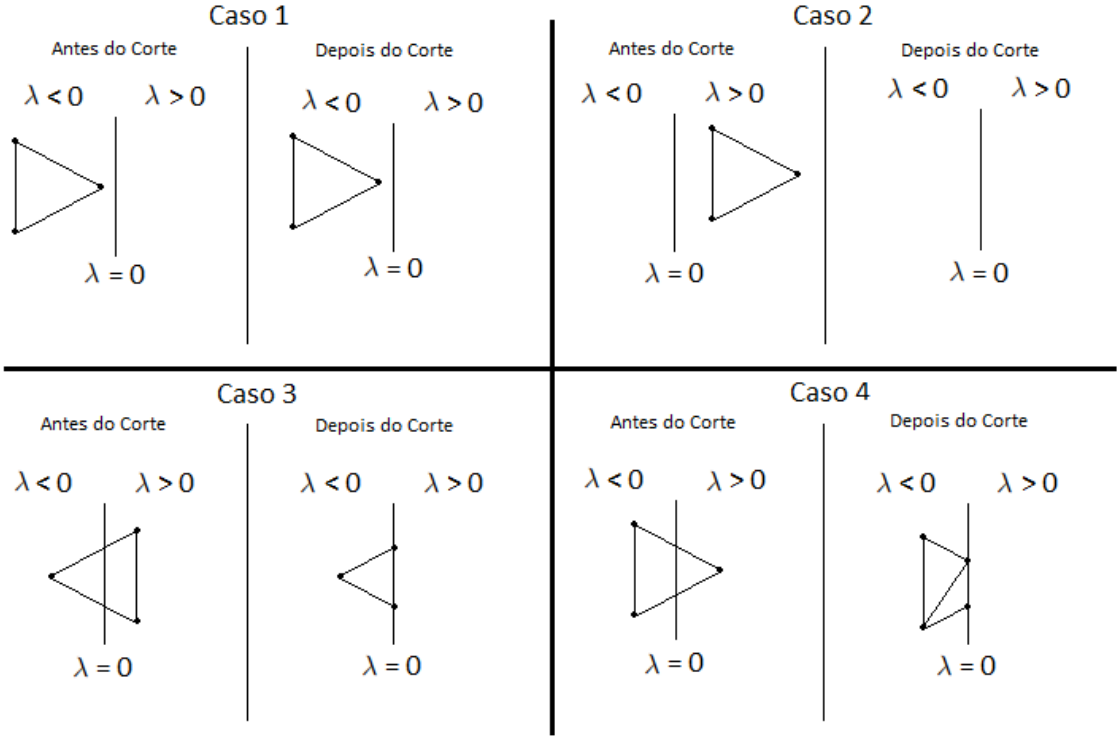
De modo geral, a primeira parte do algoritmo busca segmentar os modelos dado um plano de corte, alterando a posição dos triângulos que formam a malha. O processo consiste em deixar intactos os triângulos que se encontram em um dos lados do plano, remover completamente os que se dispõem do outro lado e alterar aqueles cujas arestas o intersectam. Inicialmente, para cada conjunto de vértices dos triângulos da malha, verifica-se de que lado do plano cada um dos vértices se encontra, substituindo cada vértice na equação do plano e obtendo um valor λ correspondente. Esse número está relacionado aos quatro possíveis casos associados à mudança dos triângulos apresentados a seguir, desconsiderando a permutação dos vértices de cada triângulo.

No primeiro caso, se o valor λ dos três vértices de um triângulo satisfaz $\lambda \leq 0$, esse triângulo permanece inalterado (Figura 3.3 - Caso 1). No segundo caso, se o valor λ de todos os vértices de um triângulo satisfaz $\lambda > 0$, o triângulo é removido do objeto (Figura 3.3 - Caso 2). Havendo diferença entre os sinais dos três valores λ obtidos para cada vértice, significa que o triângulo

é interseccionado pelo plano, torna-se necessário recalcular os vértices cujos valores respeitam $\lambda > 0$. Diferentes soluções devem ser aplicadas dependendo da localização espacial dos vértices em relação ao plano, gerando os demais casos.

No terceiro caso, se apenas um dos três valores λ satisfaz $\lambda \leq 0$, então os dois vértices restantes cujos λ satisfazem $\lambda > 0$ são reposicionados para o ponto de intersecção do plano com a respectiva aresta de cada ponto (Figura 3.3 - Caso 3). No quarto caso, se dois dos três valores de λ respeitam $\lambda \leq 0$, então o vértice restante que respeita $\lambda > 0$ é alterado analogamente para um dos pontos de intersecção do plano com uma das arestas que liga esse vértice aos demais. Como existem dois pontos de intersecção para apenas um ponto a ser deslocado, neste caso um novo triângulo é criado, formado pelos dois pontos de intersecção e pelo outro vértice em que $\lambda \leq 0$ de forma a não provocar sobreposição com o outro triângulo criado (Figura 3.3 - Caso 4). Se os vértices alterados possuem normais, estas podem ser calculadas por uma média ponderada levando em conta a distância do ponto de intersecção aos vértices que formam a aresta em questão ou simplesmente copiadas do vértice transladado.

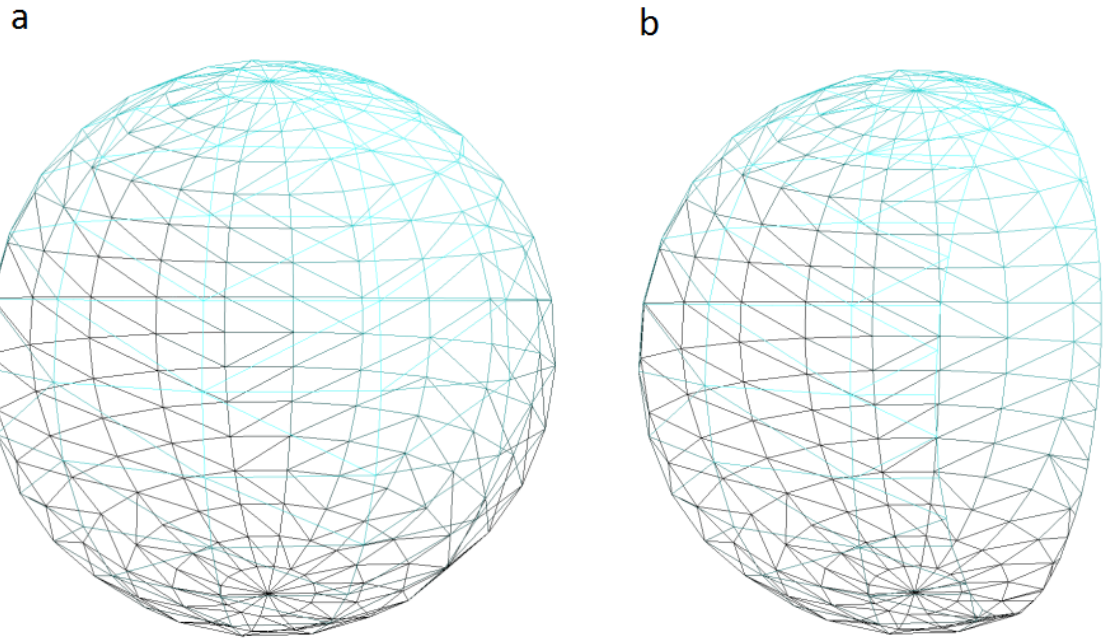
Figura 3.3 – Tipos de cortes aplicados a cada triângulo da malha do ponto de vista 2D.



Fonte: Compilado pelo autor.

Ao final dessa etapa os modelos são segmentados, reconstruindo as ligações dos triângulos da superfície da malha que cruzavam o plano. A Figura 3.4 apresenta um exemplo de triângulos antes e depois da segmentação para um modelo em 3D. O passo seguinte visa mostrar ciclos para esses pontos que se encontram sobre o plano.

Figura 3.4 – Exemplo do modelo de uma esfera antes (a) e depois (b) do corte.



Fonte: Compilado pelo autor.

3.1.2 Divisão em Conjuntos Topologicamente Conexos

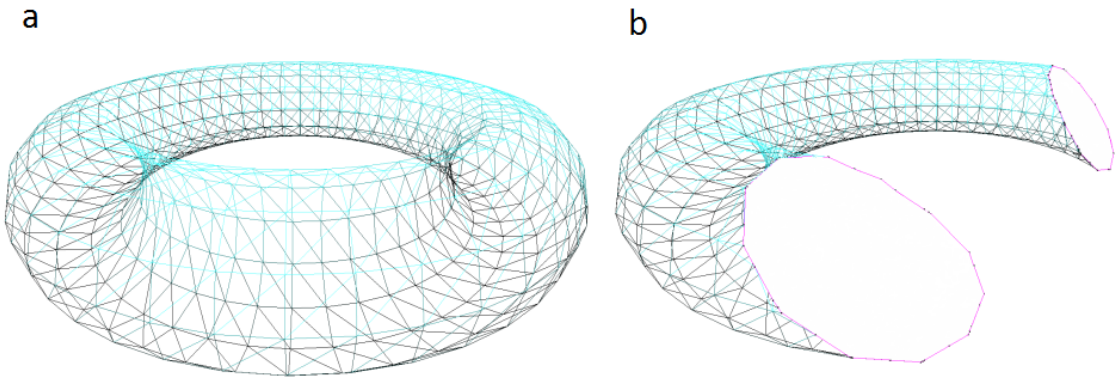
Tendo em vista que o sistema proposto abrange objetos formados por uma malha triangular, essa fase busca separar em grupos todos os vértices que estão conectados entre si sobre a equação do plano, abrangendo também casos onde o corte provoca malhas deconexas na superfície do plano. A ideia consiste, portanto, em fazer uma pesquisa em profundidade ao longo dos vértices que satisfazem a equação do plano, percorrendo vizinhos nas mesmas condições até que um ciclo seja fechado, indicando um conjunto topologicamente conexo e obtendo também a ordenação dos vértices.

Escolhe-se arbitrariamente um dentre os vértices que estão sobre o plano, obtidos na subseção anterior. A partir desse vértice, verifica-se a ligação com seus vizinhos que também se encontram sobre o plano por meio das arestas incidentes, até que seja descoberto o primeiro vértice pesquisado, formando um ciclo. Todo o processo é repetido enquanto existirem vértices que ainda não foram atingidos, já que pode haver qualquer quantidade de conjuntos de vértices desconexos. Cada ciclo de vértices sobre o plano encontrado é denominado segmento. Assume-se que os objetos aqui representados possuem volumes formando pelo menos um segmento independentemente do plano de corte escolhido para que possa ser dada continuidade à abordagem. Entretanto, vale observar que este é um requisito para que objetos possuam um *rendering* interno.

Portanto, o algoritmo encontra as envoltórias côncavas bidimensionais corretas se projetadas

sobre o plano utilizando a informação da vizinhança em 3D por meio da organização topológica dos vértices, onde cada segmento está separado e com os vértices ordenados. A Figura 3.5 mostra um exemplo em que dois conjuntos conexos são detectados para um objeto *torus*. A etapa seguinte apresenta uma mudança de base para que esses segmentos sejam definidos no sistema de coordenadas do plano.

Figura 3.5 – Exemplo do modelo de um *torus* dado um plano antes (a) e depois (b) do corte, mostrando dois conjuntos conexos distintos.



Fonte: Compilado pelo autor.

3.1.3 Mudança de Base do Universo para o Plano

Todos os pontos gerados pela superfície do modelo já segmentados em grupos conexos se encontram em coordenadas do universo. Com a finalidade de facilitar e aplicar os algoritmos subsequentes, é realizada uma mudança no sistema de coordenadas, uma vez que todos os pontos alterados já se encontram sobre o plano. Assim, esses pontos são definidos em uma nova base vetorial formada pelos eixos u, v e n representando o sistema de coordenadas do plano, permitindo a aplicação de algoritmos em duas dimensões.

Seja $\vec{n}$ o vetor da normal do plano de corte, t um ponto no espaço 3D pertencente ao plano e $\vec{a}$ um vetor arbitrário. Uma nova base vetorial é definida por meio de três vetores ortogonais entre si: $\vec{n}$ e dois novos vetores $\vec{v} = |\vec{n}|X|\vec{a}|$ e $\vec{u} = |\vec{v}|X|\vec{n}|$. Considerando $\vec{n}$ já normalizado, uma matriz $B_{U\to P}$ de mudança de base de pontos do universo para pontos do plano pode ser definida por:

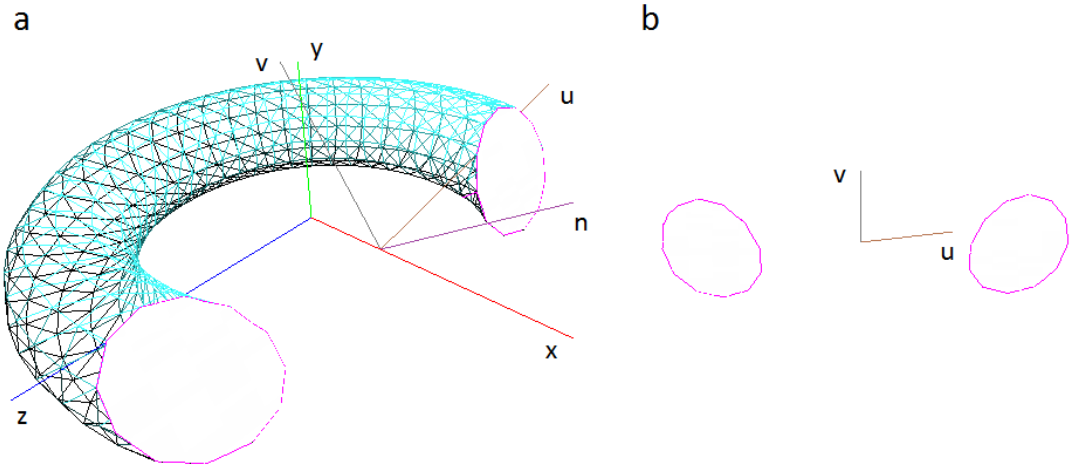
$$B_{U\to P} = \begin{pmatrix} u_x & u_y & u_z & -t_x \cdot u \\ v_x & v_y & v_z & -t_y \cdot v \\ n_x & n_y & n_z & -t_z \cdot n \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Portanto, para cada vértice dos segmentos sobre o plano de corte encontrados no universo, sua respectiva coordenada local é dada por:

$$\begin{pmatrix} x_P \\ y_P \\ z_P \\ 1 \end{pmatrix} = B_{U \to P} \cdot \begin{pmatrix} x_U \\ y_U \\ z_U \\ 1 \end{pmatrix}$$

Logo, esses vértices também estão definidos no sistema de coordenadas do plano. A Figura 3.6 apresenta os conjuntos de vértices definidos nas coordenadas do universo e do plano após mudança de base. A seção seguinte demonstra o cálculo dos chamados *holes*, necessários para que os objetos de entrada definidos como estruturas internas sejam desconsiderados na etapa de triangulação mostrada adiante.

Figura 3.6 – Exemplo do modelo de um *torus* com pontos no espaço universo (a) e transformados (b).



Fonte: Compilado pelo autor.

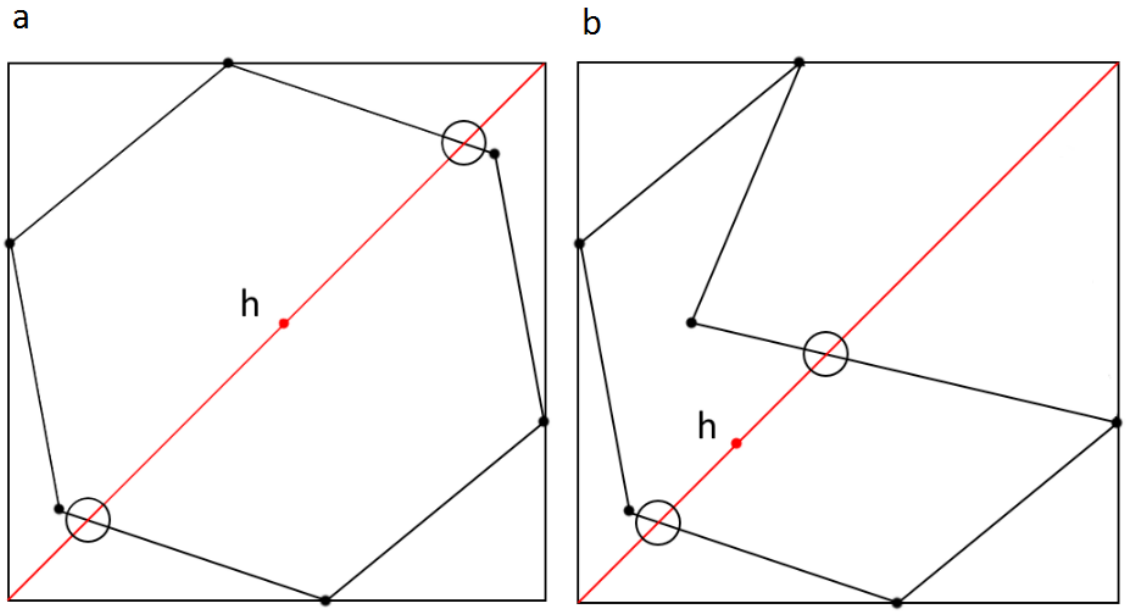
3.1.4 Cálculo de *Hole* dos Segmentos

Para que a técnica de triangulação utilizada na seção seguinte funcione de forma a não triangular determinadas partes de objetos que representam estruturas internas, além da sequência de cada um dos segmentos calculados, também são requeridos *holes*. Um *hole* é, portanto, definido como um ponto localizado dentro de um polígono bidimensional. A técnica usada para descobrir um ponto para cada segmento que certamente está na sua parte interna consiste em traçar duas linhas diagonais de um extremo a outro para cada segmento. Logo após, cada uma das linhas é percorrida em um sentido, calculando todas as intersecções na ordem em que são

encontradas. O ponto médio dos dois primeiros pontos de intersecção da diagonal com as arestas do polígono encontrados que satisfizerem o algoritmo de *point-in-polygon* é definido como *hole*.

Para cada segmento com seus vértices já definidos em coordenadas do plano, a *bounding box* bidimensional contendo quatro pontos circunscritos é encontrada. Os pontos de intersecção de uma das diagonais da *bounding box* com todas as arestas de cada um dos segmentos são calculados e ordenados por distância com uma das extremidades da diagonal. O novo ponto candidato a *hole* é calculado pela média aritmética entre dois pontos de intersecção consecutivos (Figura 3.7), verificados pela técnica *point-in-polygon*. Se nenhum dos candidatos estiver dentro do polígono (Figura 3.8-a), outra diagonal é selecionada e o mesmo processo é repetido (Figura 3.8-b).

Figura 3.7 – Exemplo do cálculo de *holes* para diversos segmentos obtidos a partir de um objeto representando furos.



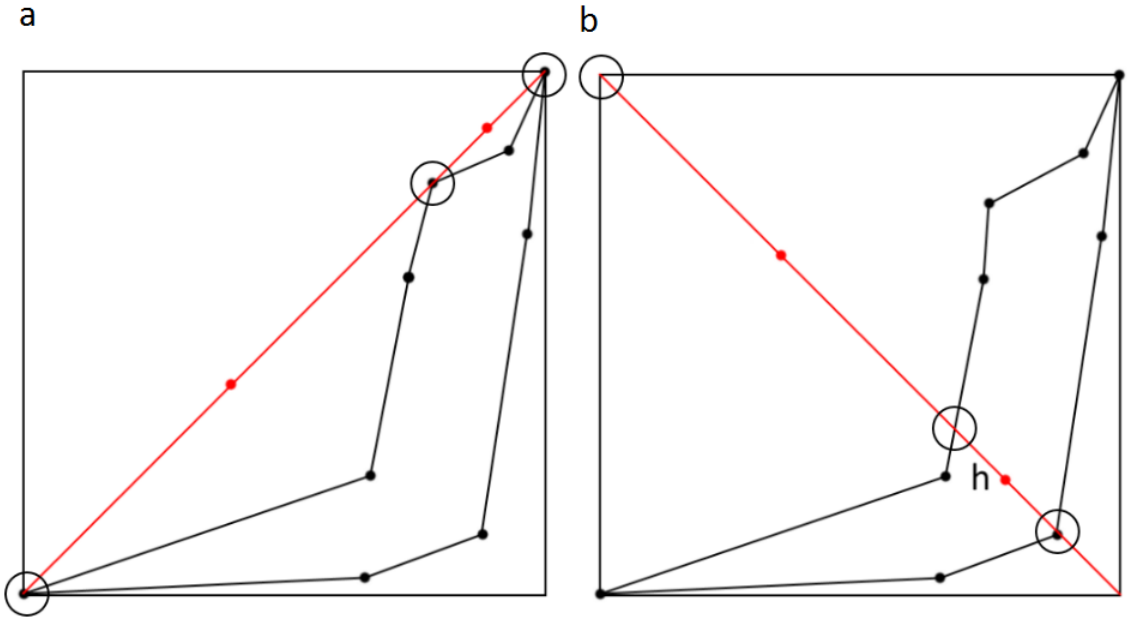
Fonte: Compilado pelo autor.

Assim, cada um dos segmentos que fazem parte dos objetos representando estruturas internas possui um *hole* associado, necessário para o procedimento de triangulação da seção seguinte.

3.1.5 Triangulação

O passo atual consiste em encontrar uma triangulação para os pontos que se interseccionam com o plano. A triangulação escolhida utilizou a técnica de *Shewchuk* (SHEWCHUK, May

Figura 3.8 – Exemplo de caso especial do cálculo de *holes* onde a técnica sem extensões não funciona (a) e resolução após extensão (b).

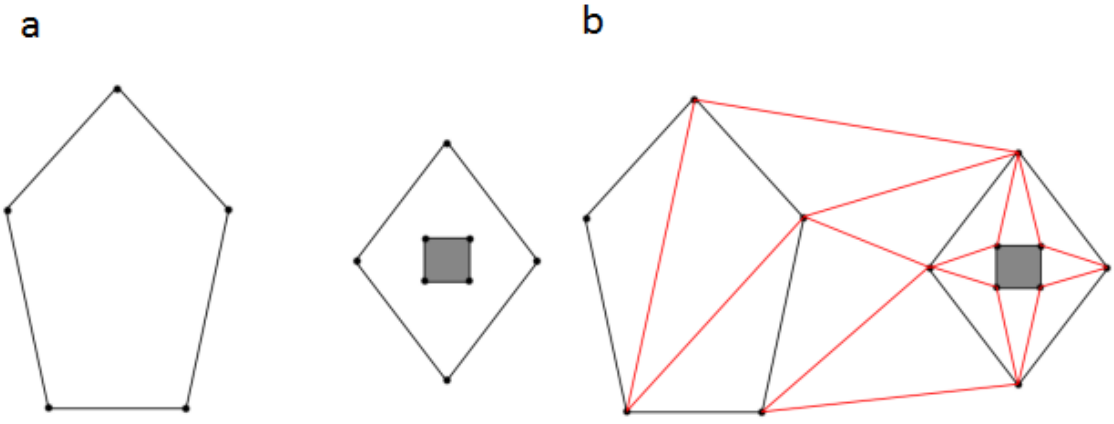


Fonte: Compilado pelo autor.

1996), onde novas arestas são criadas sobre os pontos sem alterar suas coordenadas. A abordagem, dados pontos em um espaço bidimensional e, opcionalmente, segmentos e *holes*, gera conexões entre os vértices de acordo com as restrições da técnica.

Vale ressaltar que o resultado da abordagem como foi utilizada considera a envoltória convexa, mesmo que o objeto seja côncavo ou esteja separado em diversos segmentos na região do corte. Esse problema é resolvido na seção seguinte. A Figura 3.9 mostra um exemplo da triangulação aplicada para dois segmentos, onde um deles contém um *hole*.

Figura 3.9 – Exemplo de triangulação para um conjunto com dois segmentos e um *hole* (área hachurada) (a) e sua respectiva triangulação obtida (b), com novas arestas em vermelho.



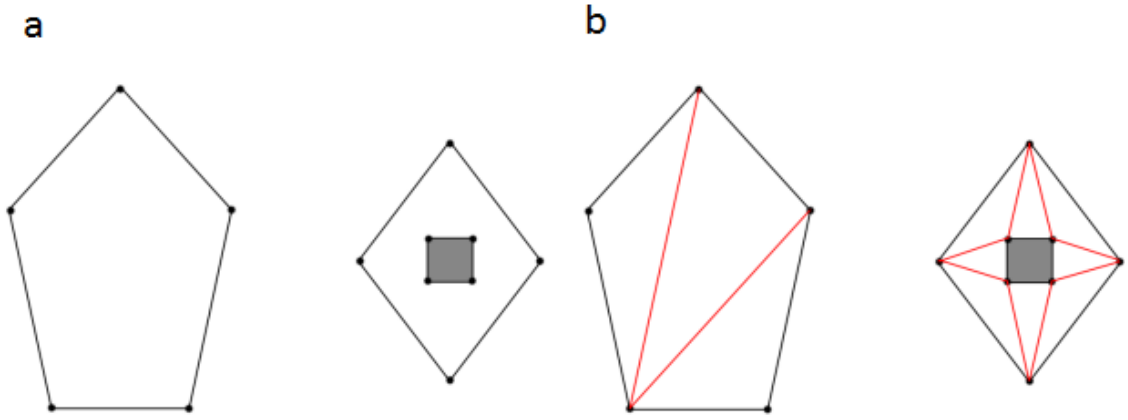
Fonte: Compilado pelo autor.

3.1.6 Remoção dos Triângulos Externos

Essa etapa consiste em retirar todos os triângulos externos aos segmentos gerados pela técnica de triangulação usada, podendo corresponder a ligações que erroneamente transformam um polígono côncavo em convexo ou que conectam dois segmentos desconexos entre si. Para cada triângulo gerado no passo anterior é calculado seu baricentro e em seguida verificado se esse ponto se encontra dentro de algum dos segmentos pelo algoritmo de *point-in-polygon*. Todos os triângulos que se encontram fora dos segmentos são desconsiderados para as próximas etapas.

A Figura 3.10 mostra o resultado final da triangulação usando o mesmo exemplo apresentado na Figura 3.9. A etapa seguinte corresponde à última fase do capítulo, envolvendo a mudança de base novamente para coordenadas do universo.

Figura 3.10 – Exemplo de triangulação para um conjunto com dois segmentos e um *hole* (área hachurada) (a) e sua respectiva triangulação obtida após remoção dos triângulos externos (b), com novas arestas em vermelho.



Fonte: Compilado pelo autor.

3.1.7 Mudança de Base do Plano para o Universo

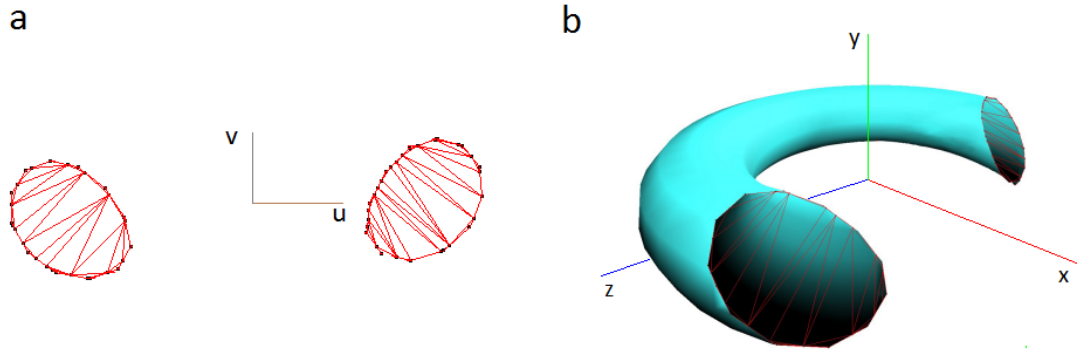
Tendo em vista a mudança de base representada na Subseção 3.1.3, é preciso que os triângulos gerados estejam definidos sobre as coordenadas do universo, para que o *rendering* seja realizado corretamente, processo detalhado na seção seguinte. Para isso, cada vértice de cada triângulo encontrado deve ser multiplicado pela matriz inversa de $B_{U \to P}$.

Seja a matriz $B_{P \to U}$ de mudança de base, inversa de $B_{U \to P}$. Então, para cada vértice de cada triângulo obtido nas etapas anteriores sua coordenada no universo é obtida por meio de:

$$\begin{pmatrix} x_U \\ y_U \\ z_U \\ 1 \end{pmatrix} = B_{P \to U} \cdot \begin{pmatrix} x_P \\ y_P \\ z_P \\ 1 \end{pmatrix}$$

A Figura 3.11 mostra uma possível triangulação na base $B_{P \to U}$ projetada em um plano bidimensional e seu resultado na base $B_{U \to P}$. Esta seção finaliza o processo de reestruturação dos modelos pelo plano de corte, representando o modelo original segmentado e devidamente retriangulado sobre o plano definido.

Figura 3.11 – Exemplo de triangulação sobre a base $B_{P \to U}$ projetada em no plano (a) e seu resultado na base $B_{U \to P}$ (b).

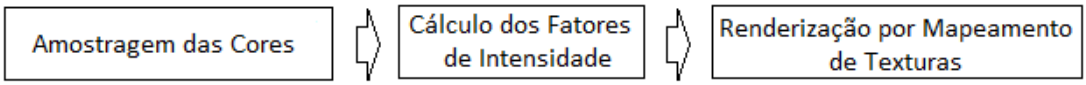


Fonte: Compilado pelo autor.

3.2 Texturização Interna

Esta seção abrange as fases para obter o *rendering* interno do objeto no plano de corte, a partir de um conjunto de pontos utilizando uma técnica simples de textura sólida baseada na interpolação de imagens. Essa etapa pode ser usada separadamente para colorir quaisquer pontos de um volume, entretanto para o contexto do sistema desenvolvido deve ser explorada de forma conjunta com os passos previamente descritos. Dados os triângulos obtidos na seção anterior, três imagens de entrada representando texturas internas são usadas para determinar a cor da área do plano que intersecta o objeto, obtida por meio de uma função de interpolação entre as imagens. Para gerar o resultado sobre o plano, um mapa de texturas é amostrado, colorido e interpolado sobre os triângulos em questão. A Figura 3.12 apresenta o pipeline desse módulo.

Figura 3.12 – Pipeline do módulo de *texturização* interna.



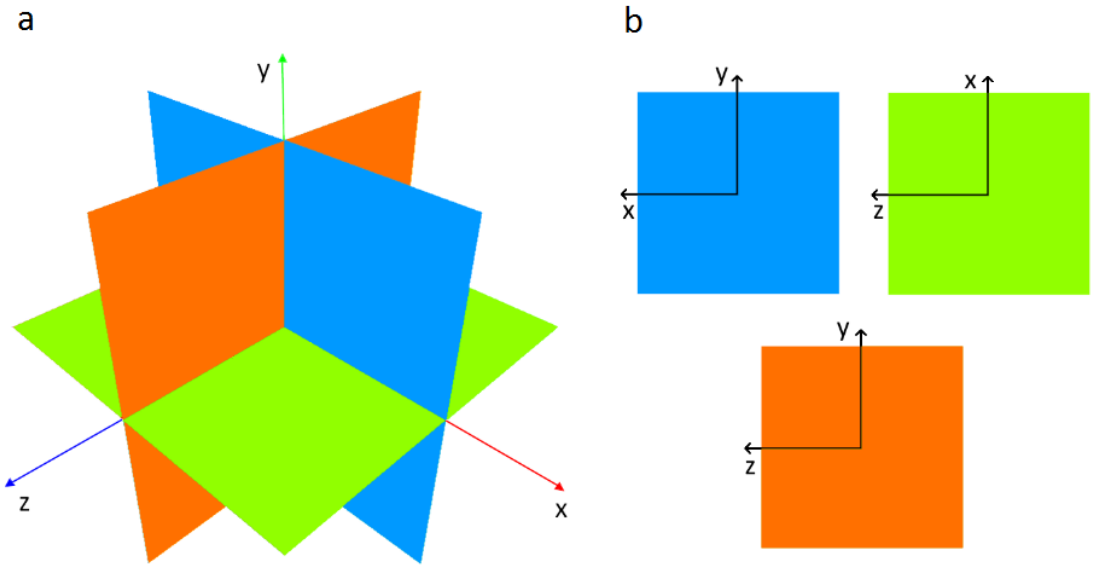
Fonte: Compilado pelo autor.

3.2.1 Amostragem das Cores

Essa etapa inicial consiste em primeiramente criar um mapeamento das imagens como planos ortogonais entre si centrados na origem do objeto superfície, limitados de acordo com a *bounding box* do modelo. Na sequência, cada ponto do plano a ser texturizado recebe uma cor de cada uma das imagens de acordo com sua localização espacial. Essa etapa permite encontrar as três cores de qualquer ponto interno ao volume definido pela *bounding box*.

Seja I uma imagem formada por uma matriz I_w x I_h de pixels, acessados pelas coordenadas de linhas i e colunas j $I_{i,j}$. As imagens $I1$, $I2$ e $I3$ com dimensões associadas aos planos $z = 0$, $y = 0$ e $x = 0$ são dispostas nos eixos verticais e horizontais (y, x), (x, z), (y, z), respectivamente. Os maiores e menores valores de coordenadas dos objetos representando a superfície de entrada são denotados por $max = (max_x, max_y, max_z)$ e $min = (min_x, min_y, min_z)$. Intuitivamente, de acordo com a fórmula adiante, as imagens graficamente estão colocadas sobre o universo, sendo redimensionadas de acordo com a *bounding box* do objeto. A Figura 3.13 apresenta essa disposição.

Figura 3.13 – Disposição das imagens sobre as coordenadas do universo (a) e seus respectivos eixos em 2D (b).



Fonte: Compilado pelo autor.

Sejam $c1$, $c2$ e $c3$ três cores que correspondem a um determinado pixel de cada uma das imagens a ser calculado e um ponto p, definido dentro dos limites da *bounding box*. As posições m_i e m_j de cada pixel correspondente a cada uma das imagens associadas ao ponto p podem ser calculadas por meio das fórmulas:

$$m1_{i,j} = \left\{ \left\lfloor \frac{(max_y - p_y)}{(max_y - min_y)} I1_h \right\rceil, \left\lfloor \frac{(max_x - p_x)}{(max_x - min_x)} I1_w \right\rceil \right\}$$

$$m2_{i,j} = \left\{ \left\lfloor \frac{(max_x - p_x)}{(max_x - min_x)} I2_h \right\rceil, \left\lfloor \frac{(max_z - p_z)}{(max_z - min_z)} I2_w \right\rceil \right\}$$

$$m3_{i,j} = \left\{ \left\lfloor \frac{(max_y - p_y)}{(max_y - min_y)} I3_h \right\rceil, \left\lfloor \frac{(max_z - p_z)}{(max_z - min_z)} I3_w \right\rceil \right\}$$

Portanto, cada uma das cores c associada ao ponto p pode ser obtida por:

$$c1 = I1_{m1_i, m1_j}$$

$$c2 = I2_{m2_i, m2_j}$$

$$c3 = I3_{m3_i, m3_j}$$

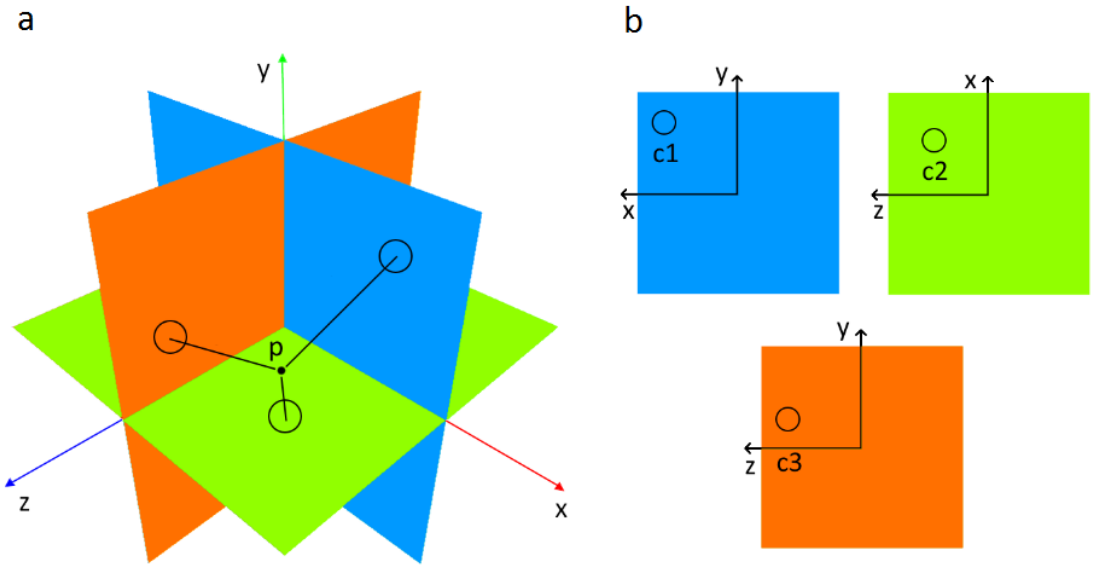
Na prática, a amostragem realizada consiste em descobrir a cor traçando-se uma linha reta do ponto aos três planos que contém as imagens. A Figura 3.14 ilustra esse processo, definindo qual pixel deve ser escolhido de acordo com a posição traçada do ponto no espaço à imagem também no espaço. Esse processo pode ser calculado para todos os pontos limitados pelo volume da *bounding box* dos quais se deseja obter informações das cores. No contexto do trabalho, deve ser feito para um conjunto de pontos definidos sobre o plano, dependendo da resolução desejada.

3.2.2 Cálculo dos Fatores de Intensidade

Após a obtenção das cores de cada uma das imagens para cada ponto que se deseja amostrar, resta definir um peso para cada uma delas, posteriormente interpolando-as a fim de gerar uma única cor para os pontos. A estratégia usada baseia-se na distância entre o ponto e as imagens, adquirindo uma maior contribuição de imagens mais próximas e uma menor de imagens mais afastadas. Essa subseção calcula os chamados fatores de intensidade que, quando utilizados juntamente com as cores obtidas na seção anterior, fornecem o resultado da cor final, detalhada na subseção seguinte.

Seja f um fator de intensidade, que corresponde a uma porcentagem de contribuição de uma

Figura 3.14 – Amostragem para um ponto p sobre as três imagens no espaço 3D (a) e suas respectivas cores c para cada imagem em 2D (b).



Fonte: Compilado pelo autor.

imagem I sobre o ponto p. Os fatores auxiliares f_x, f_y e f_z são calculados por:

$$f_x = 1 - \left| 1 - \frac{(max_x - p_x)}{\frac{max_x - min_x}{2}} \right|$$

$$f_y = 1 - \left| 1 - \frac{(max_y - p_y)}{\frac{max_y - min_y}{2}} \right|$$

$$f_z = 1 - \left| 1 - \frac{(max_z - p_z)}{\frac{max_z - min_z}{2}} \right|$$

Dados os fatores $f1$, $f2$ e $f3$, correspondentes às imagens $I1$, $I2$ e $I3$, respectivamente, tem-se:

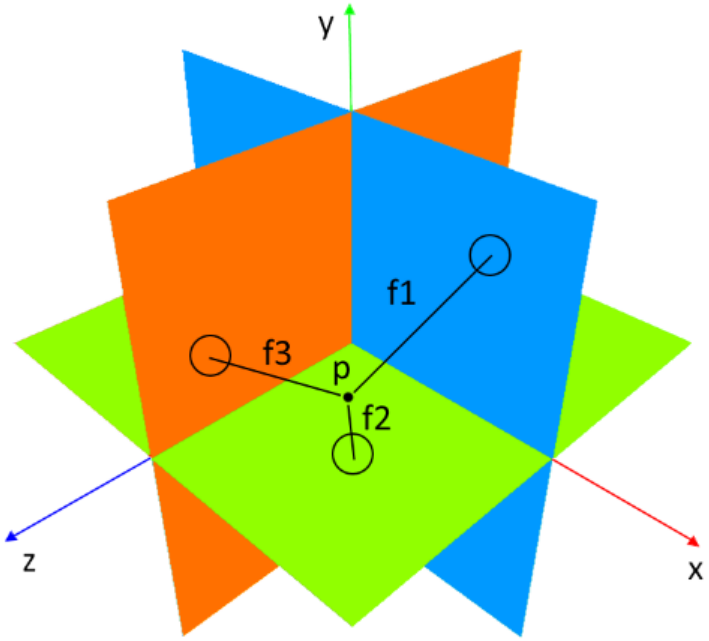
$$f1 = \frac{f_z}{f_x + f_y + f_z}$$

$$f2 = \frac{f_y}{f_x + f_y + f_z}$$

$$f3 = \frac{f_x}{f_x + f_y + f_z}$$

Essa restrição das fórmulas para descoberta dos fatores de intensidade garante que a soma das contribuições de cada cor final não passa de um, fazendo com que a cor do ponto não seja saturada. Uma ilustração da técnica com uma representação abstrata dos fatores de intensidade associados a cada imagem pode ser vista na Figura 3.15.

Figura 3.15 – Esquema intuitivo apresentando uma noção tridimensional aproximada dos fatores de intensidade para cada uma das imagens dado um ponto p.



Fonte: Compilado pelo autor.

3.2.3 Renderização por Mapeamento de Texturas

A subseção atual mostra primeiramente o cálculo das cores para um determinado ponto p, juntando os cálculos expostos nas subseções anteriores e concluindo o procedimento genérico para coloração de pontos limitados pela área de definição das texturas. Entretanto, o escopo do trabalho não se limita a encontrar pontos coloridos no espaço 3D, mas sim a gerar uma textura sólida que interaja com estruturas internas. Para isso, dados os triângulos calculados na Seção 3.1, busca-se renderizá-los por meio da criação e posterior mapeamento de uma textura criada por amostragem regular de pontos sobre a *bounding box* bidimensional desses triângulos definidos em coordenadas do plano.

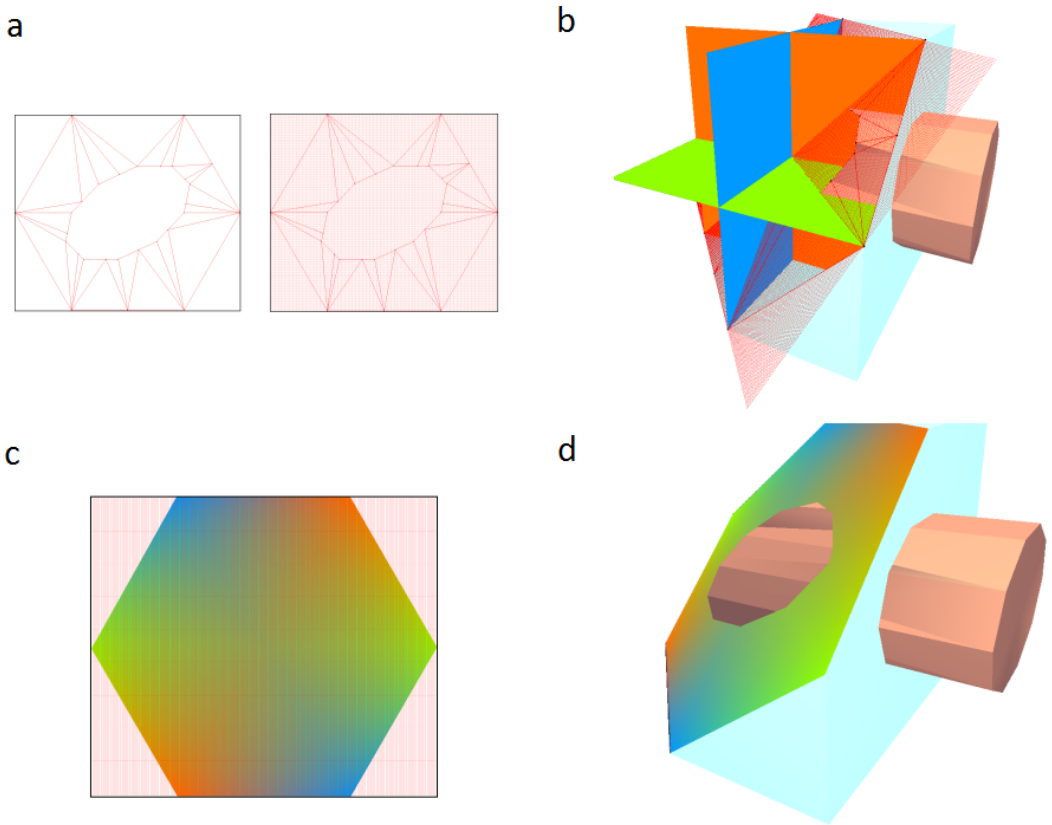
O processo de geração da cor c de cada ponto p pode ser obtida por:

$$c = c1f1 + c2f2 + c3f3 \tag{3.1}$$

Para que a imagem renderizada leve em consideração apenas pontos do plano, um *grid* regular bidimensional formado por pontos sobre o plano de corte é definido. Esse grid com n linhas e m colunas se encontra ajustado sobre a *bounding box* bidimensional, calculada de acordo com os vértices dos triângulos selecionados a serem renderizados (Figura 3.16-a), descritos em coordenadas do plano.

Para que os pontos sejam coloridos, novamente a matriz $B_{P \to U}$ é aplicada, desta vez sobre cada um dos pontos do *grid*, representando-os sobre o espaço do universo (Figura 3.16-b). Então, as cores desses pontos são obtidas por meio da equação 3.1. Associando as cores dos pontos no espaço do universo com o espaço do plano, onde foram originalmente definidos, é possível mapear diretamente essas cores encontradas para os pixels de uma nova imagem I, com dimensões $I_w = m$ e $I_h = n$. Essa imagem é definida como uma textura, que possui coordenadas u e v normalizadas pelos valores da *bounding box* (Figura 3.16-c). Assim, as coordenadas de textura possuem um mapeamento direto com cada vértice dos triângulos definidos sobre coordenadas do plano, bastando associá-las antes dos triângulos serem renderizados no universo (Figura 3.16-d).

Figura 3.16 – Processo de coloração e mapeamento de texturas. Dada uma triangulação e os pontos sobre o grid (a), os pontos são representados no espaço universo por mudança de base (b) e as cores interpoladas por cada ponto são calculadas, gerando uma imagem de textura (c) e, por fim, a textura é mapeada para os triângulos a serem renderizados pelo modelo (d).



Fonte: Compilado pelo autor.

Como os possíveis *holes* encontrados não fazem parte desses triângulos marcados como renderizáveis, nenhuma cor será colocada neles, já que a textura não será mapeada.

3.3 Discussão

Esse capitulo apresentou o sistema desenvolvido, procurando mostrar os conceitos primeiramente de forma genérica e depois contextualizados ao escopo do trabalho. A primeira parte detalhou a sequência de passos para retriangular os objetos de entrada, gerando como saída triângulos a serem renderizados, já considerando possíveis *holes* como áreas sem cores internas. Na seção anterior, ao longo das primeiras duas subseções, foi descrita uma técnica da área de textura sólida, envolvendo uma interpolação linear entre as imagens, gerando cores para qualquer ponto dentro do paralelepípedo formado pelas três imagens ortogonais entre si. Na última subseção foi especificada a fórmula final para as cores dos pontos, além de mostrar o *rendering* feito por meio do mapeamento de texturas sobre os triângulos gerados.

Portanto, o sistema proposto consegue gerar informação para a visualização do interior de objetos que possuem estruturas internas, divididos entre dois tipos de modelos de entrada: superfícies e estruturas internas. As cores dos objetos superfícies são obtidas por meio de uma técnica de interpolação linear entre três imagens representando secções ortogonais entre si de acordo com a distância dos pontos com as imagens. Por outro lado, os objetos que representam as estruturas internas não recebem cores associadas, permitindo visualizar modelos mais complexos com partes do interior vazias. Assumindo que uma superfície pode ser dividida em diversos planos de corte, o sistema pode ser adaptado para esses casos também. Além disso, com o conceito proposto, cada superfície pode ser representada por um conjunto distinto de três imagens, permitindo a criação de modelos mais complexos envolvendo diferentes texturizações em camadas.

4 RESULTADOS

Esse capítulo visa mostrar a técnica apresentada no capítulo anterior por meio de três estudos de caso com o objetivo de demonstrar o sistema desenvolvido. O primeiro corresponde à Seção 3.1, mostrando objetos com estruturas internas segmentados segundo planos de corte, a fim de detalhar melhor a primeira parte do trabalho. O segundo está relacionado à Seção 3.2, focando no resultado produzido pela técnica de interpolação das texturas, apresentando um estudo direcionado à segunda parte do sistema. Por fim, o terceiro expõe o sistema completo para um modelo de fígado com vasos internos, envolvendo ambas as etapas da técnica apresentada para simuladores cirúrgicos. O ambiente de desenvolvimento utilizou OpenGL como API gráfica e os testes rodaram em um computador com a seguinte especificação:

- Processador: Intel Core i7-4770 CPU 3.40 GHz
- Memória (RAM): 16.0 GB
- Sistema Operacional: Windows 8 64 bits
- Placa Gráfica: NVIDIA GeForce GTX 770

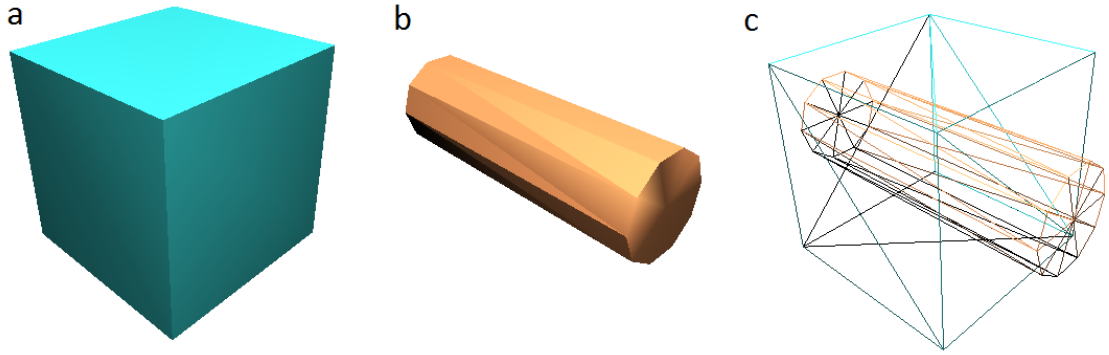
A estrutura do capítulo consiste em quatro seções, as três primeiras correspondem aos estudos de caso e a última a uma discussão geral sobre os testes e os respectivos resultados.

4.1 Estudo de Caso 1

Esta seção foca majoritariamente a primeira parte da técnica desenvolvida, que consiste na visualização do modelo segmentado a partir de um plano de corte definido. Para isso são usados dois conjuntos de modelos distintos, onde o primeiro corresponde a um cubo com um cilindro no interior (Figura 4.1) e o outro a uma esfera com dois *torus* internamente, dispostos perpendicularmente entre si (Figura 4.2). O número de vértices e triângulos de cada um dos quatro tipos de modelos utilizados se encontra na Tabela 4.1. O algoritmo roda em tempo real para qualquer plano de intersecção nesses modelos, apesar de variar o tempo de acordo com a quantidade de triângulos intersectados. As imagens utilizadas como entrada não são consideradas relevantes para esse estudo, já que o objetivo é apresentar apenas o algoritmo de corte.

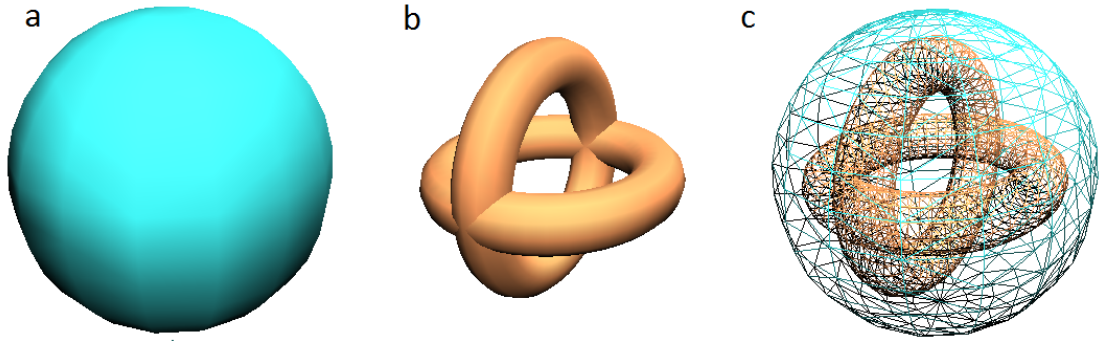
Quatro cortes foram gerados para este estudo de caso, cujos detalhes envolvendo modelo, normal e ponto do plano se encontram na Tabela 4.2. Os primeiros dois cortes foram realizados em relação ao primeiro conjunto (Figura 4.3), enquanto outros dois foram aplicados sobre o

Figura 4.1 – Modelo de um cubo (a) representando a malha da superfície e de um cilindro (b) representando a estrutura interna quando dispostos em conjunto (c).



Fonte: Compilado pelo autor.

Figura 4.2 – Modelo de uma esfera (a) representando a malha da superfície e de dois *torus* (b) representando a estrutura interna quando dispostos em conjunto (c).



Fonte: Compilado pelo autor.

segundo (Figura 4.4), procurando cobrir diferentes pontos de vista. Os triângulos gerados sobre o plano não estão representados nas figuras. Para cada secção foram obtidos dados referentes ao número de pontos de intersecção dos objetos com o plano (PI) e ao número de triângulos gerados pela triangulação sobre o plano (TP), bem como ao tempo de execução (TE) em milisegundos para a reestruturação da malha, organizados no final da Tabela 4.2.

Tabela 4.1 – Quantidade de vértices, triângulos e dimensões por modelo.

Modelo	Vértices	Triângulos	Dimensões (x, y, z)
Cubo	8	12	$(1, 1, 1)$
Cilindro	22	40	$(0.5, 0.476, 1.5)$
Esfera	242	480	$(1.5, 1.5, 1.5)$
Torus (2x)	576	1152	$(1.25, 0.25, 1.25)$

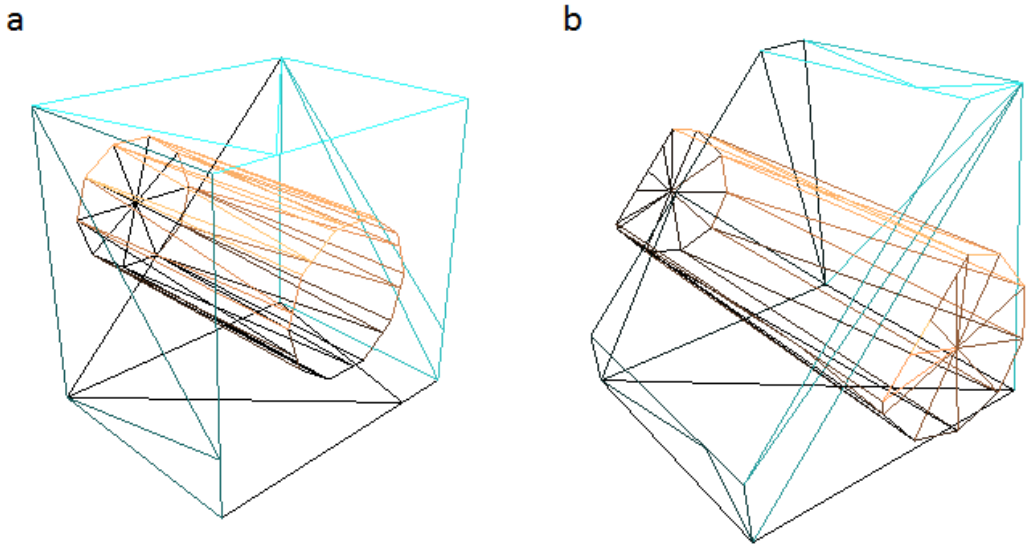
Fonte: Compilado pelo autor.

Tabela 4.2 – Associação dos cortes definidos por normal e ponto com os modelos e as respectivas informações: PI (número de pontos de insersecção dos objetos com o plano), TP (número de triângulos gerados pela triangulação sobre o plano) e TE (tempo de execução em milisegundos).

Corte	Modelo	Normal	Ponto	PI	TP	TE
1	Cubo e Cilindro	$(0, 0, 1)$	$(0, 0, 0.3)$	28	22	2.94
2	Cubo e Cilindro	$(1, -1, 0)$	$(0.2, 0, 0)$	19	14	1.96
3	Esfera e *Torus* (2x)	$(1, 0, 0)$	$(0, 0, 0)$	313	426	13.24
4	Esfera e *Torus* (2x)	$(0.75, 0.1, 0.65)$	$(-0.2, 0.3, 0.1)$	194	208	9.82

Fonte: Compilado pelo autor.

Figura 4.3 – Cortes 1 (a) e 2 (b) gerados para o primeiro conjunto formado pelos modelos de cubo e cilindro.

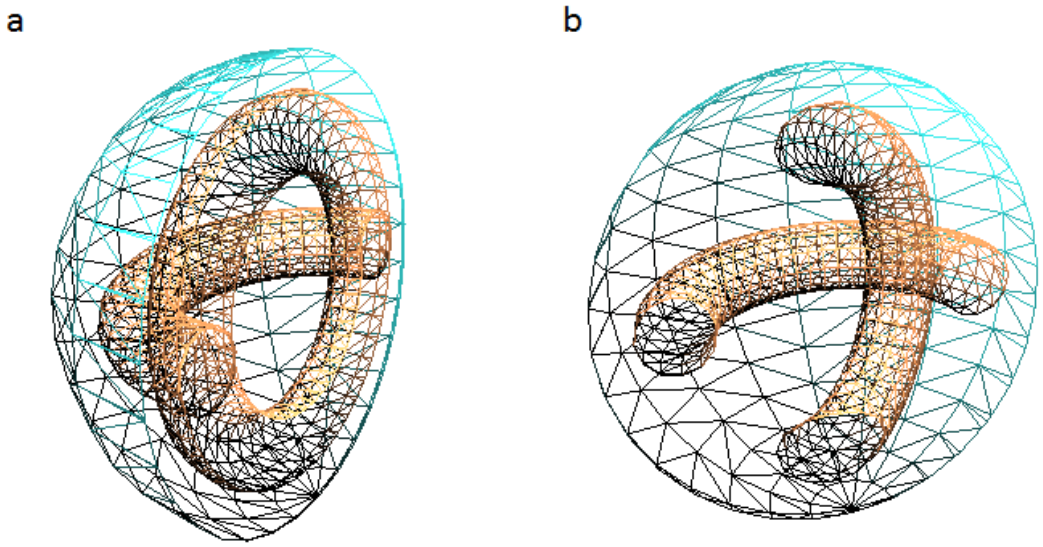


Fonte: Compilado pelo autor.

4.2 Estudo de Caso 2

O segundo estudo de caso desta seção visa mostrar os resultados obtidos com foco na técnica apresentada de textura sólida. De forma a facilitar a visibilidade, são usados modelos desconsiderando as estruturas internas. Dois diferentes objetos foram escolhidos, sendo o primeiro um cubo (Figura 4.1-a) e o segundo uma esfera (Figura 4.1-b), com informações apresentadas na Tabela 4.1. Cada um desses modelos é associado a um conjunto diferente de imagens de entrada com resolução 256 x 256. Para o cubo, duas imagens objetivam verificar a interpolação para casos simples. Para a esfera foram utilizadas três texturas irregulares distintas para cada um dos planos, extraídas do trabalho de síntese de texturas proposto por Kwatra e colegas (KWATRA et al., Jul. 2005). Em ambos os casos, para auxiliar na visualização, cada uma das imagens se encontra em planos ortogonais entre si centrados na origem. Essas informações

Figura 4.4 – Cortes 3 (a) e 4 (b) especificados na Tabela 4.2, gerados para o segundo conjunto formado pelos modelos de esfera e *torus*.

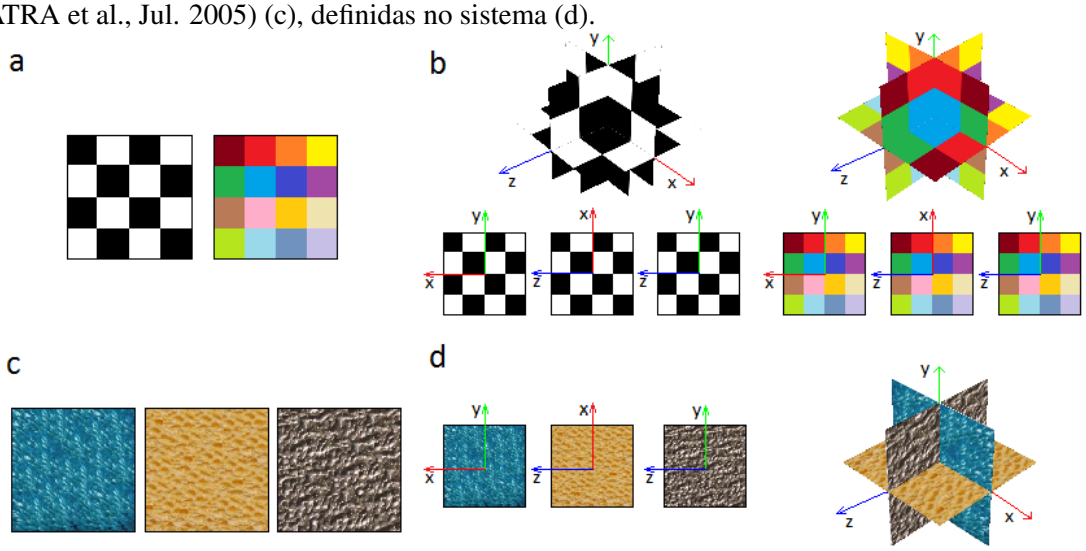


Fonte: Compilado pelo autor.

estão representadas na Figura 4.5.

Essa parte da técnica tem o tempo de execução variado de acordo com a quantidade de pontos usados para amostrar os pontos do *grid*, que, por sua vez, está relacionado com a qualidade da textura criada. A alteração do tamanho das imagens de entrada não apresenta diferenças consideráveis no desempenho, que é determinado principalmente pelo algoritmo de corte e pelo tamanho do *grid* usado para amostrar a textura.

Figura 4.5 – Texturas usadas como estudo de caso. A primeira consiste em quadrados pretos e brancos e coloridos (a), dispostas no sistema (b). As demais texturas apresentam padrões irregulares variados (KWATRA et al., Jul. 2005) (c), definidas no sistema (d).



Fonte: Compilado pelo autor.

Cinco cortes foram realizados com modelos, normal e ponto do plano apresentados na Ta-

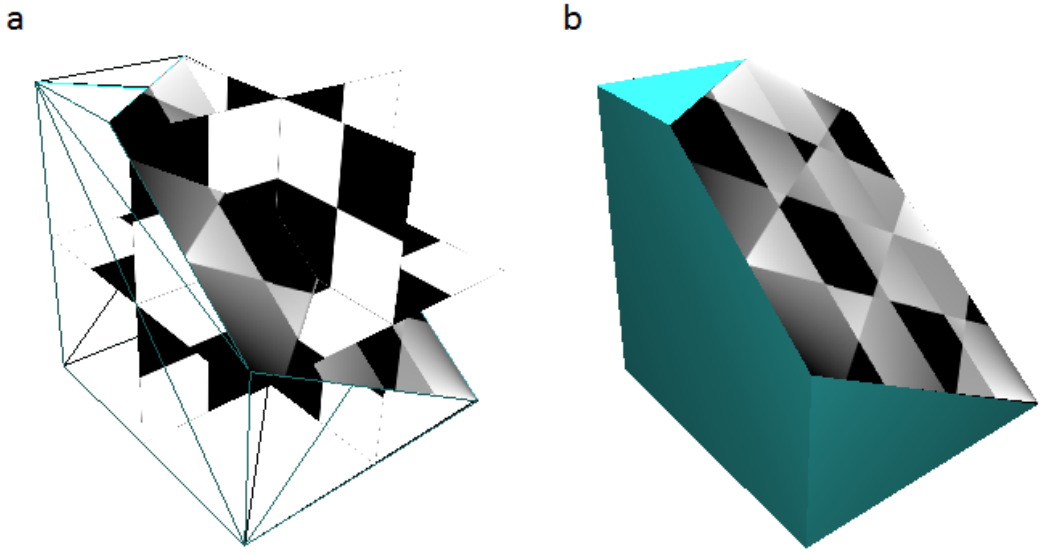
bela 4.3. Para o primeiro modelo dois cortes foram selecionados, a fim de mostrar que as cores obtidas pela técnica de interpolação seguem o padrão apresentado (Figuras 4.6 e 4.7). É possível observar uma variação dos tons de cinza e dos coloridos dentro dos próprios quadrantes, de acordo com a distância dos pontos da textura aos planos. O último corte foi realizado sobre a esfera, gerando três imagens distintas que objetivam revelar o impacto da amostragem do *grid* de pontos na resolução da textura criada (Figura 4.8). O número de pontos de intersecção dos objetos com o plano (PI) e de triângulos gerados pela triangulação sobre o plano (TP), a resolução quadrada do grid (RG) de textura do corte e o tempo de execução em milisegundos (TE) estão especificados na Tabela 4.3.

Tabela 4.3 – Associação dos cortes definidos por normal e ponto com os modelos e as respectivas informações: PI (número de pontos de insersecção dos objetos com o plano), TP (número de triângulos gerados pela triangulação sobre o plano), RG (resolução quadrada do grid de textura do corte) e TE (tempo de execução em milisegundos).

Corte	Modelo	Normal	Ponto	PI	TP	RG	TE
1	Cubo	$(0.5, 0.7, -0.4)$	$(0.2, 0, 0)$	10	7	256	16.28
2	Cubo	$(-0.3, 0.65, -0.65)$	$(-0.35, 0.2, 0.2)$	9	10	256	15.49
3	Esfera	$(1, 1, 1)$	$(0, 0, 0)$	61	60	32	13.97
4	Esfera	$(1, 1, 1)$	$(0, 0, 0)$	61	60	512	78.18

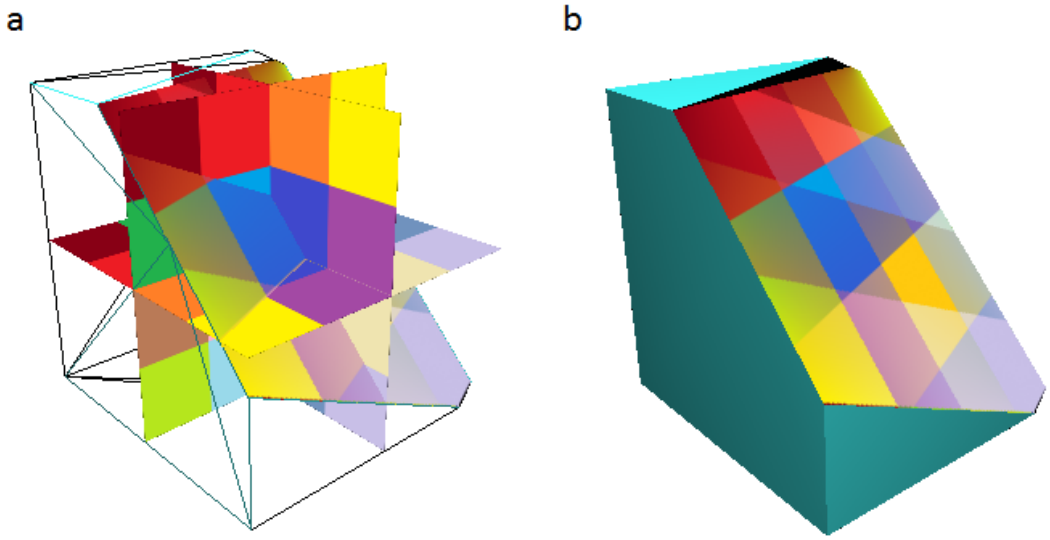
Fonte: Compilado pelo autor.

Figura 4.6 – Corte 1 especificado na Tabela 4.3, gerado para o conjunto de imagens formadas por um padrão quadriculado em preto e branco. Imagem do resultado juntamente com (a) e sem (b) a disposição das texturas de entrada.



Fonte: Compilado pelo autor.

Figura 4.7 – Corte 1 especificado na Tabela 4.3, gerado para o conjunto de imagens formadas por um padrão quadriculado colorido. Imagem do resultado juntamente com (a) e sem (b) a disposição das texturas de entrada.



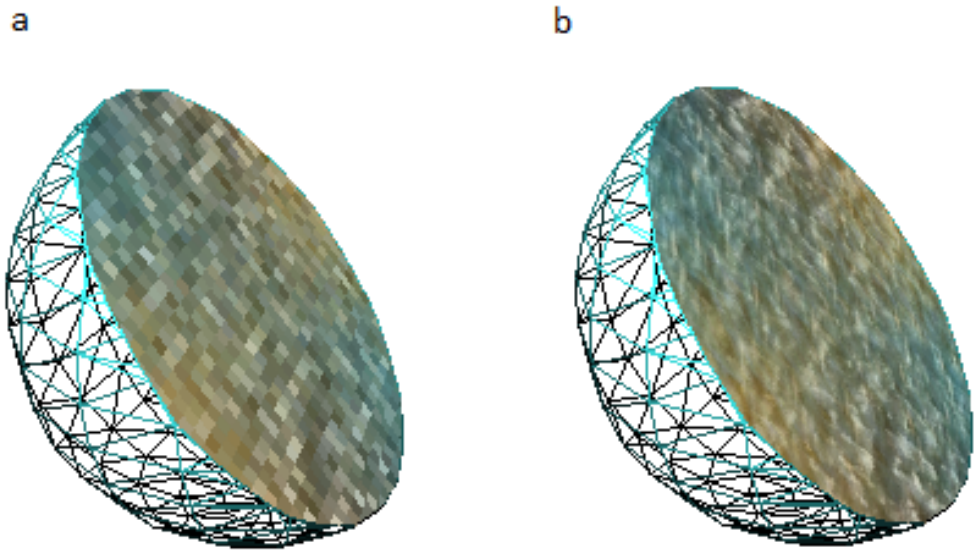
Fonte: Compilado pelo autor.

4.3 Estudo de Caso 3

O terceiro estudo de caso abrange efetivamente a técnica proposta nesse trabalho, mostrando elementos das duas seções anteriores. A fim de demonstrar a técnica descrita no Capítulo 3 e visando apresentar resultados relacionados com simuladores cirúrgicos, o modelo completo constituído pela superfície e vasos do fígado foi gerado a partir de imagens de CT do projeto *SLIVER07* (HEIMANN et al., Feb. 2009). Por meio do conjunto de imagens do arquivo referente ao *Training data, part 1*, em torno de 300 imagens de CT com resolução 512 x 512 da região abdominal com máscaras binárias para auxiliar na segmentação do fígado foram acessadas. A partir dai, com o auxílio do software *MeVisLab* (MEVIS MEDICAL SOLUTIONS AG, 2015), a segmentação das imagens entre fígado e vasos do fígado foi obtida por meio da diferença de densidade dos tecidos, sem necessidade de segmentação manual de cada fatia. O algoritmo de *marching cubes* contido no software por meio do módulo que usa VTK (*The Visualization Toolkit*) foi então aplicado para construir uma malha triangular 3D para cada objeto separadamente.

A malha triangular da superfície do fígado e dos vasos em alta resolução contém, respectivamente, 66798 vértices e 133574 triângulos e 22551 vértices e 45338 triângulos. Uma nova malha com uma menor quantidade de vértices e triângulos (Tabela 4.4) foi gerada para ambos os modelos pela técnica *decimation* presente no *software Blender* (BLENDER FOUNDATION, 2015), obtendo os objetos usados nos estudos de caso dessa seção (Figuras 4.9-a e 4.9-b).

Figura 4.8 – Cortes 3 (a) e 4 (b) especificados na Tabela 4.3, gerados para o segundo conjunto de imagens formado por três texturas distintas (KWATRA et al., Jul. 2005) dispostas conforme Figura 4.5-d. A variação mostra o impacto da amostragem dos pontos do *grid* de textura na qualidade da interpolação gerada.



Fonte: Compilado pelo autor.

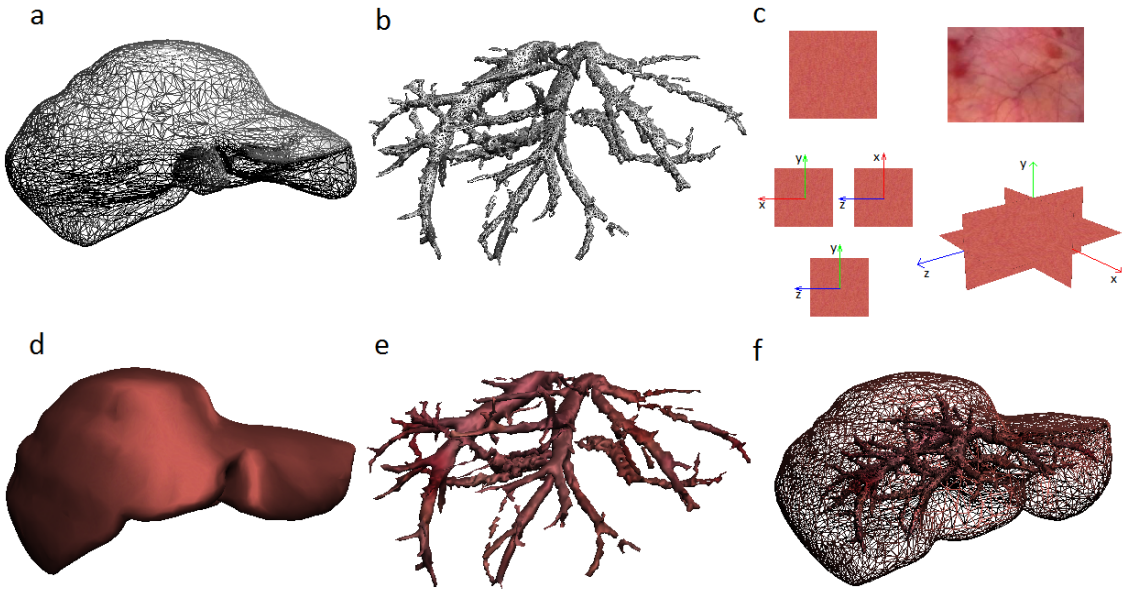
Tabela 4.4 – Quantidade de vértices, triângulos e dimensões por modelo.

Modelo	Vértices	Triângulos	Dimensões (x, y, z)
Fígado	3350	6673	$(2.44, 1.711, 3.323)$
Vasos Internos	11216	22662	$(1.86, 1.237, 2.493)$

Fonte: Compilado pelo autor.

A textura extraída de um trabalho envolvendo síntese de texturas para simuladores cirúrgicos virtuais (XUE-MEI; AI-MIN; QIN-PING, Oct. 2009) para um fígado foi tanto mapeada externamente como também usada nas três imagens ortogonais de entrada para gerar a textura interna. Essa textura foi escolhida pela falta de imagens de cortes reais que deveriam ser usadas. Mesmo com três imagens iguais como entrada, o resultado final da interpolação linear não gera a própria textura, já que, em geral, a amostragem das cores vem de pontos distintos das imagens. Para os vasos, a textura externa escolhida para o mapeamento veio de um trabalho relacionado ao *rendering* em tempo real para simuladores cirúrgicos (ELHELW et al., Aug. 2004). A primeira textura teve tamanho reajustado para 256 x 256 e a segunda para 595 x 415, estando representadas na Figura 4.9-c, juntamente com a disposição das texturas internas ajustadas sobre os planos. Assim, os modelos do fígado e dos vasos tiveram a superfície renderizada de acordo com as texturas apresentadas, conforme evidenciam as Figuras 4.9-d e 4.9-e. Uma visão geral do modelo com as duas estruturas combinadas se encontra na Figura 4.9-f.

Figura 4.9 – Visão geral dos modelos e texturas usadas. Modelos de fígado (a) e vasos internos (b) em *wireframe*. Textura interna usada para representar a superfície e o interior do fígado (XUE-MEI; AI-MIN; QIN-PING, Oct. 2009), textura externa para a superfície dos vasos (ELHELW et al., Aug. 2004) e imagens da textura interna no sistema (c). Modelos de fígado (d) e vasos internos (e) com as suas texturas externas correspondentes. Modelos dispostos juntos (f).



Fonte: Compilado pelo autor.

Diversos resultados foram obtidos a partir dos recursos de entrada apresentados, sendo expostos em uma série de cortes realizados utilizando o sistema desenvolvido. Assim como nas seções anteriores, as normais e os pontos dos planos para todos os cortes estão na Tabela 4.5, juntamente com informações sobre o número de pontos de intersecção dos objetos com o plano (PI) e dos triângulos gerados pela triangulação sobre o plano (TP). Além disso, essa mesma tabela contém para cada corte a sua resolução quadrada do grid (RG) e o seu tempo de execução (TE) em milisegundos.

Associação dos cortes definidos por normal e ponto com os modelos e as respectivas informações: PI (pontos de insersecção dos objetos com o plano), TP (número de triângulos gerados pela triangulação sobre o plano), RG (resolução quadrada do grid de textura do corte) e TE (tempo de execução em milisegundos).

As secções estão organizadas em diferentes grupos, contendo uma figura para cada objetivo proposto. O Corte 1 (Figura 4.10) envolve um plano com diversas intersecções do fígado e dos seus vasos, além de apresentar uma resolução de 256 x 256 do *grid* de texturas. Essa resolução é igual às das imagens de entrada, provendo uma amostragem de pontos coerente.

Os cortes de números 2 a 7 apresentam uma visualização de diversas fatias do fígado com a mesma inclinação do plano de corte, variando sobre um mesmo sentido (Figura 4.11). Como

Tabela 4.5 – Associação dos cortes definidos por normal e ponto com os modelos do fígado e vasos internos e as respectivas informações: PI (número de pontos de insersecção dos objetos com o plano), TP (número de triângulos gerados pela triangulação sobre o plano), RG (resolução quadrada do grid de textura do corte) e TE (tempo de execução em milisegundos).

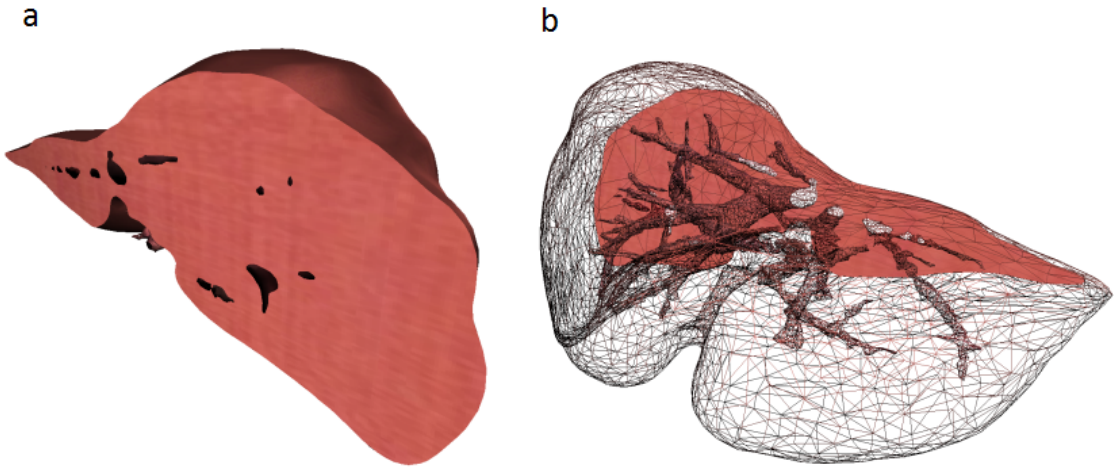
Corte	Normal	Ponto	PI	TP	RG	TE
1	$(1, 0, 0)$	$(0.2, 0, 0)$	667	657	256	136.21
2	$(1, 1, 0)$	$(1, 0, 0)$	182	178	256	41.32
3	$(1, 1, 0)$	$(0.6, 0, 0)$	586	600	256	120.86
4	$(1, 1, 0)$	$(0.2, 0, 0)$	569	597	256	118.02
5	$(1, 1, 0)$	$(-0.2, 0, 0)$	618	647	256	143.25
6	$(1, 1, 0)$	$(-0.6, 0, 0)$	476	498	256	93.86
7	$(1, 1, 0)$	$(-1, 0, 0)$	269	273	256	55.58
8	$(0.7, 0.2, -0.5)$	$(-0.2, -0.4, 0.1)$	479	492	32	37.73
9	$(0.7, 0.2, -0.5)$	$(-0.2, -0.4, 0.1)$	479	492	64	40.42
10	$(0.7, 0.2, -0.5)$	$(-0.2, -0.4, 0.1)$	479	492	128	51.06
11	$(0.7, 0.2, -0.5)$	$(-0.2, -0.4, 0.1)$	479	492	256	92.49
12	$(0.7, 0.2, -0.5)$	$(-0.2, -0.4, 0.1)$	479	492	512	255.65
13	$(0.7, 0.2, -0.5)$	$(-0.2, -0.4, 0.1)$	479	492	1024	905.47
14	$(-0.6, -0.7, -0.9)$	$(0, 0, 0.1)$	568	524	256	104.59
15	$(0, 1, 0)$	$(0, 0, 0)$	303	300	256	70.94
16	$(0.6, 0.7, 0)$	$(0.2, 0, 0)$	604	620	256	121.82
17	$(0, -0.4, -0.9)$	$(-0.1, 0, -0.1)$	614	561	256	103.77

Fonte: Compilado pelo autor.

a resolução do *grid* permanece constante, é possível observar um tempo de execução maior quanto maior a quantidade de pontos sobre o plano. Essa queda de desempenho está relacionada com a parte geométrica do sistema, envolvendo um maior número de recortes e triangulações, além do consequente aumento da quantidade de triângulos da textura quando é mapeada pelo *hardware*. A Figura 4.12-a mostra a quantidade de pontos interseccionados (PI) pelo plano com relação a cada um desses cortes e a Figura 4.12-b apresenta a variação do tempo de execução para cada um dos cortes apresentados. Observando o comportamento dos dois gráficos, é possível perceber uma forte relação entre a quantidade de pontos e o tempo de execução, com variação dos máximos e mínimos locais aproximada.

Os cortes de número 8 a 13 trazem diferentes resoluções no *grid*, correspondente à textura a ser mapeada sobre os triângulos (Figura 4.13). Além disso, um detalhe do modelo mostrando a textura foi extraído e ampliado pela própria imagem e com *zoom* pelo programa para cada resolução. Como o corte permanece sempre sobre o mesmo lugar, é possível notar uma alta perda de desempenho por conta do aumento da quantidade de pontos amostrados pela textura. A Figura 4.14 ilustra a variação crescente do tempo de execução quanto maior a resolução quadrada do grid (RG).

Figura 4.10 – Corte 1 especificado na Tabela 4.5. Apresentação de um corte do sistema renderizado (a) e em *wireframe* (b).



Fonte: Compilado pelo autor.

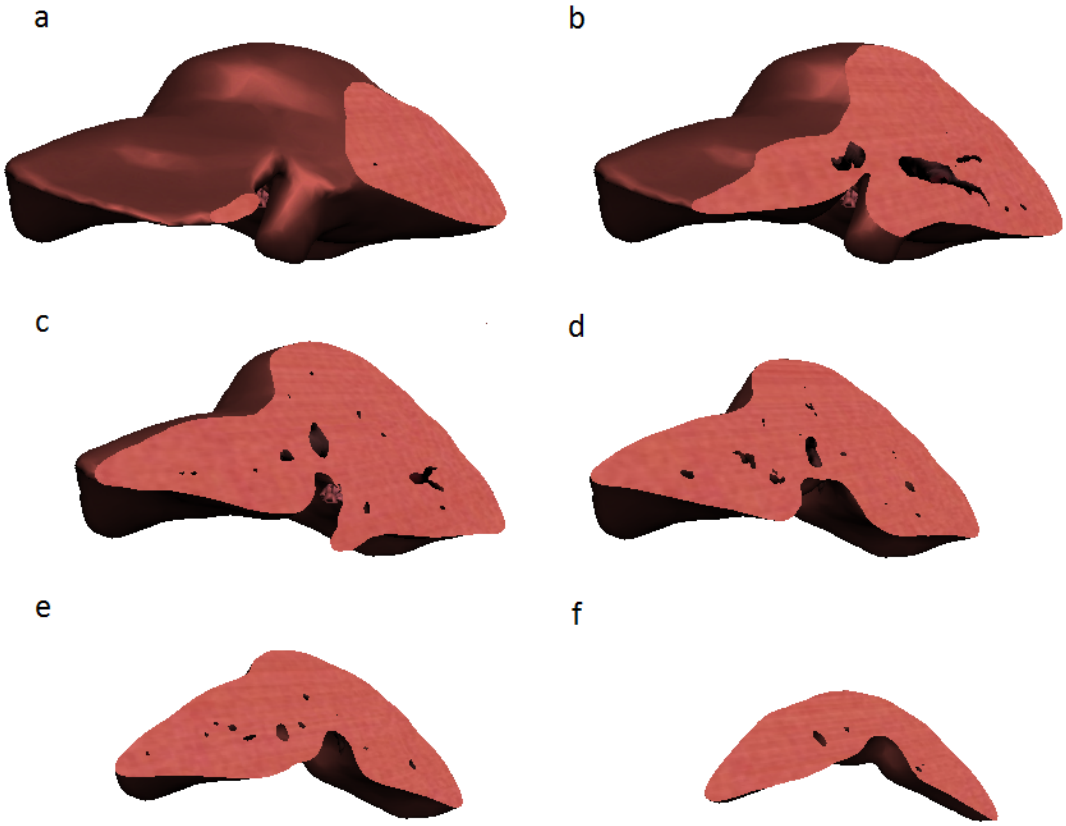
O Corte 14 secciona o fígado virtual de forma semelhante às cirurgias de hepatectomia, fazendo com que apenas uma parte do fígado permaneça no sistema. A Figura 4.15 compara o resultado obtido a um corte semelhante de um trabalho de visualização na área de cirurgia do fígado. Entretanto, o modelo do órgão e dos vasos desse trabalho não são construídos e renderizados geometricamente, obtendo apenas seus dados volumétricos.

O Corte 15 desconsidera estruturas internas do fígado, aplicando a técnica apenas para o modelo que descreve a superfície (Figura 4.16). Esse caso mostra que a técnica também funciona para colorir internamente objetos sem estruturas internas representadas por modelos geométricos.

Os Cortes 16 e 17 primeiramente comparam o resultado usando a técnica de textura sólida proposta (Figuras 4.17-a, 4.17-b) com o mapeamento direto da mesma imagem de entrada (Figuras 4.17-c, 4.17-d) sobre os triângulos encontrados. É possível notar que o simples mapeamento direto da textura de entrada produz um resultado menos borrado, uma vez que não há interpolação entre as imagens. Entretanto, a variabilidade da técnica abordada gera uma nova textura para cada alteração no plano, não sendo possível de ser alcançada pela repetição da mesma textura em diversos planos.

Além disso, os dois últimos cortes foram reaproveitados para uma nova comparação entre a técnica apresentada e o mapeamento direto. Nesse novo caso, uma das texturas de entrada contém uma pequena mancha, indicando uma possível anomalia interna do fígado, enquanto as outras permanecem iguais. O novo conjunto de texturas alocadas em seus respectivos planos se encontra na Figura 4.17-e e a suposta disposição no espaço 3D na Figura 4.17-f. Os cortes que levam em conta a nova textura para interpolação estão apresentados nas Figuras 4.17-g e

Figura 4.11 – Cortes 2 (a), 3 (b), 4 (c), 5 (d), 6 (e) e 7 (f) especificados na Tabela 4.5. Representação de segmentos mostrando uma variação de cortes sobre um mesmo sentido no fígado.



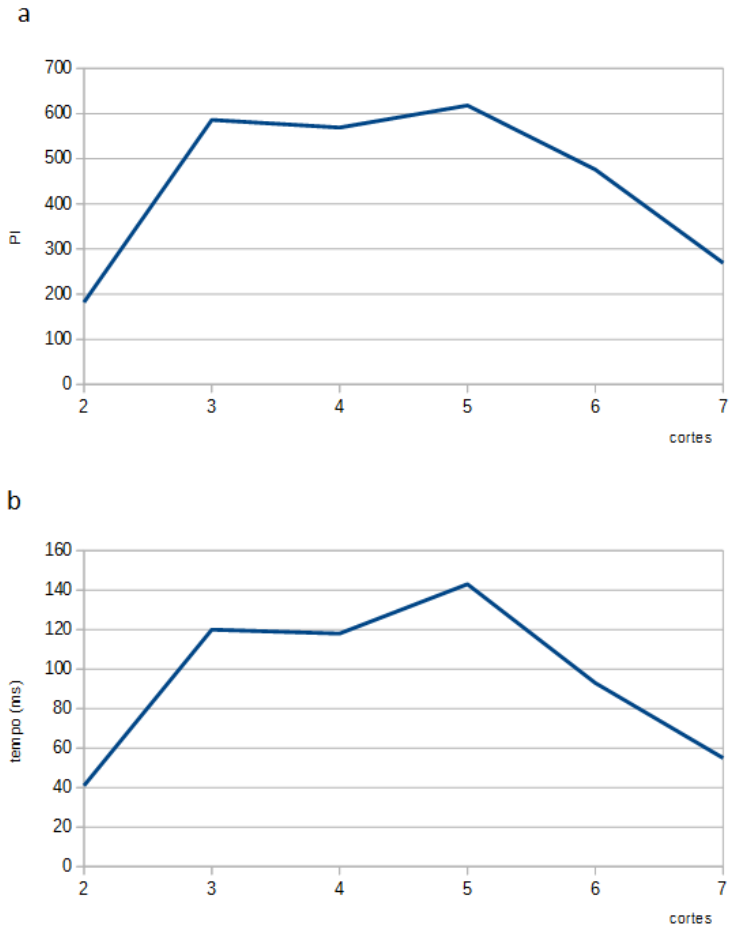
Fonte: Compilado pelo autor.

4.17-h. Nos resultados observa-se uma mancha que pode indicar uma anomalia expandida ao longo do órgão, algo difícil de ser reproduzido com o mapeamento direto da textura de entrada.

4.4 Discussão

Essa seção apresentou três estudos de caso para demonstrar o sistema desenvolvido. O primeiro teve foco na parte de reestruturação dos modelos pelo plano de corte e usou como exemplos figuras geométricas simples, definindo resultados de objetos segmentados. O Estudo de Caso 2 envolveu inicialmente texturas com padrões regulares para demonstrar situações facilmente visíveis sobre a técnica de texturização desenvolvida, utilizando para isso modelos de cubos e esferas sem estruturas internas. O terceiro e principal estudo de caso envolveu uma malha geométrica de um fígado e de seus respectivos vasos sanguíneos, obtidos a partir de diversas imagens de CT de um ser humano. Pelo fato de o trabalho não ter se baseado em imagens extraídas do interior de um fígado real, uma mesma imagem foi utilizada sobre os três planos,

Figura 4.12 – Gráficos representando a variação dos pontos de intersecção do plano (PI) (a) e a variação do tempo medido (b) para os cortes de 2 a 7.



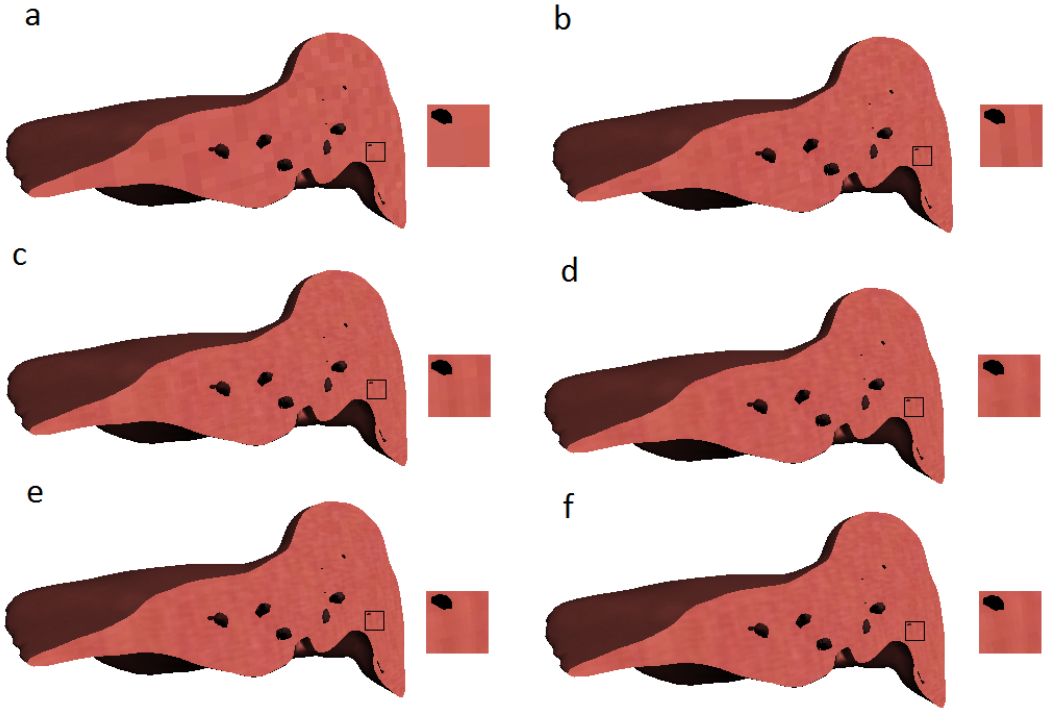
Fonte: Compilado pelo autor.

dispostas de tal forma que uma interpolação linear sobre elas não resulte na própria imagem de entrada.

Os resultados nessa seção demonstraram visualmente a técnica sob diversas perspectivas. Por meio de testes direcionados, foi possível observar que o desempenho do sistema está relacionado tanto à quantidade de triângulos interseccionados pelos modelos quanto à resolução da textura amostrada sobre o plano. Como as imagens de entrada apresentaram uma resolução fixa de 256 x 256, não foi observado um ganho visual considerável ao aumentar essa resolução. Adicionalmente, o sistema permite também coloração interna de objetos sem estruturas internas, além de apresentar um corte aproximado aos realizados em cirurgias de retirada parcial do fígado.

Por fim, uma comparação da técnica de textura sólida do sistema com texturização bidimensional direta sobre o plano de corte utilizando a mesma textura mostrou um resultado com menos ruído provocado pela interpolação. Entretanto, qualquer variação causada pelo plano

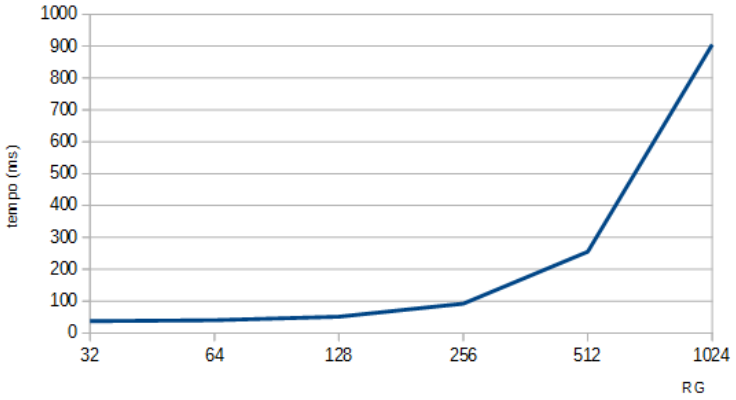
Figura 4.13 – Cortes 8 (a), 9 (b), 10 (c), 11 (d), 12 (e) e 13 (f) especificados na Tabela 4.5. Representação de diversos cortes sobre o mesmo local variando a resolução da textura criada. Os detalhes superiores das imagens representam uma ampliação da região marcada diretamente da imagem correspondente e os interiores por meio de *zoom* pelo programa.



Fonte: Compilado pelo autor.

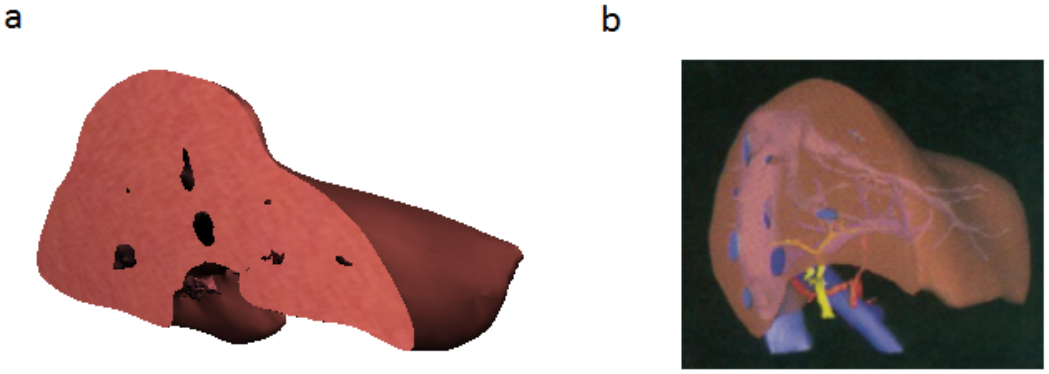
gera uma nova textura interna, evitando uma textura estática para todo o fígado. Além disso, testes mostraram que eventuais anomalias podem ser colocadas nas texturas, alterando partes internas específicas do órgão pela interpolação e simulando uma possível situação em que pacientes desenvolvem tumores, por exemplo. Esse caso se torna mais difícil de ser resolvido com o mapeamento direto de texturas.

Figura 4.14 – Gráfico representando a variação do tempo de execução com a resolução regular do grid (RG) para os cortes de 8 a 13.



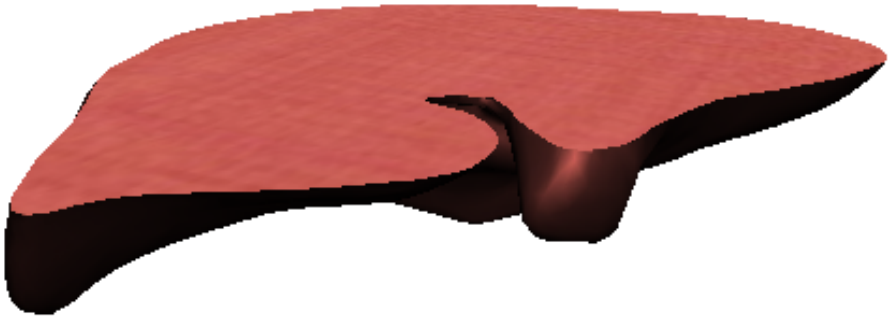
Fonte: Compilado pelo autor.

Figura 4.15 – Corte 14 especificado na Tabela 4.5. Corte semelhante ao feito para uma hepatectomia direita e comparação com visualizador desenvolvido mostrando essa cirurgia (MARESCAUX et al., Nov. 1998).



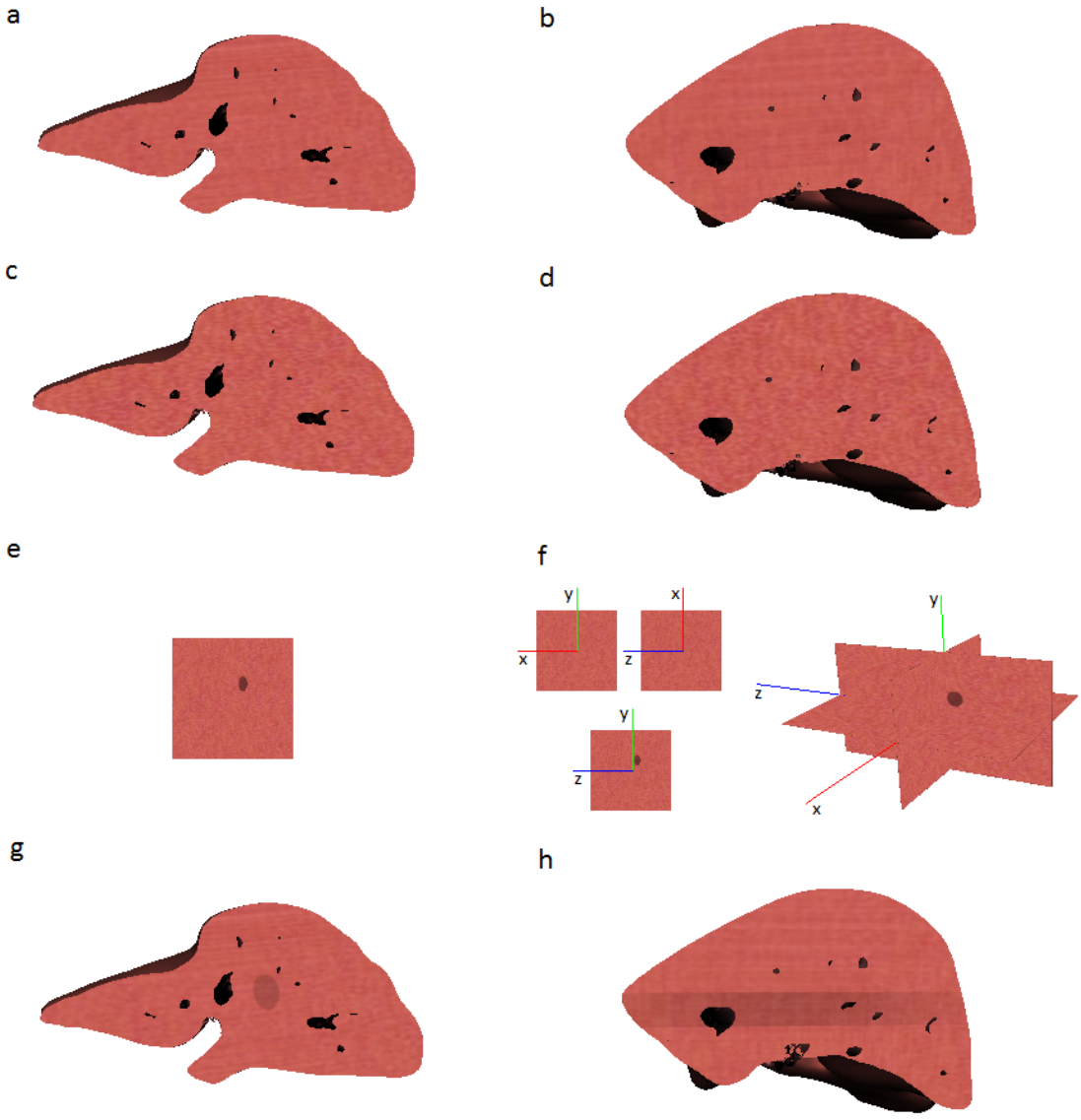
Fonte: Compilado pelo autor.

Figura 4.16 – Corte 15 especificado na Tabela 4.5. Apresenta uma coloração do interior do órgão ignorando os vasos internos por meio de um corte na região axial.



Fonte: Compilado pelo autor.

Figura 4.17 – Cortes 16(a),(c),(g) e 17(b)(d)(h), especificados na Tabela 4.5. Comparação entre o resultado utilizando a textura sólida desenvolvida para cortes distintos (a)(c) e um mapeamento direto da textura de entrada (b)(d). Uma nova imagem de entrada forma um novo conjunto (e), apresentado sobre os planos ortogonais (f). Utilizando o novo conjunto e os mesmos cortes(g)(h), uma nova textura é gerada sobre o plano.



Fonte: Compilado pelo autor.

5 CONCLUSÃO

Essa dissertação apresentou um sistema de visualização do interior de modelos com estruturas internas para simulação de cirurgias. A partir de um conjunto de malhas 3D, imagens e uma superfície, o algoritmo proposto gera uma coloração interna para o corte especificado. O trabalho envolveu uma revisão de conceitos básicos e pesquisas relacionadas e detalhou a abordagem implementada, utilizando estudos de caso para ilustrar o procedimento. Este capítulo está dividido em três seções. A primeira destaca as contribuições geradas pelo método. A segunda expõe as limitações encontradas, discutindo possíveis soluções. A última seção mostra pontos que podem ser expandidos em trabalhos futuros.

5.1 Contribuições

A Computação Gráfica procura tornar as informações virtuais mais próximas da realidade, sendo amplamente usada atualmente em simuladores cirúrgicos. Pesquisas envolvendo esses sistemas, em geral, não se detiveram na reconstrução interna dos órgãos para esse objetivo. Procurando suprir essa falta, o trabalho descreveu um sistema de cortes para malhas com estruturas internas, comumente usadas em simuladores. Como diversos trabalhos direcionam o foco para outras áreas, como métodos de colisão e retorno háptico, esse modo de reconstrução interna pode ser facilmente adaptado a fim de construir um simulador mais completo. Assim sendo, este pode ser considerado um impulso para que simuladores aumentem seu nível de realismo.

Embora o procedimento tenha sido construído para cortes, a ideia pode ser adaptada também para superfícies, desde que subdivididas em planos de corte. É importante ressaltar que, apesar da técnica de coloração interna dos modelos ser mapeada por textura sobre os triângulos, é possível utilizá-la para colorir qualquer ponto no espaço, ampliando ainda mais a sua utilização. Além disso, o método gera variabilidade da texturização dentro do modelo se comparado à utilização de uma textura diretamente colocada sobre os planos.

5.2 Limitações

Durante a realização desse sistema foram detectadas limitações. Uma delas é que a técnica de texturização utilizada é bem simples, baseando-se apenas em uma interpolação sobre as

imagens para coloração interna. Outro aspecto a ser mencionado é o estudo de caso envolvendo o fígado. Não foi possível obter fotografias reais de um fígado humano com corte nas regiões definidas, tendo sido utilizadas três imagens iguais de uma textura da superfície do órgão. Além disso, os tempos de execução da técnica não apresentaram melhores resultados, uma vez que não foi utilizada programação em GPU pelo grau de complexidade envolvido.

5.3 Trabalhos Futuros

Novos ramos de pesquisa nessa área envolvem primeiramente melhorar a técnica de texturização interna utilizada, procurando estudar maneiras que produzam resultados mais interessantes para esse tipo específico de objetos. Além disso, o acesso a imagens internas do fígado poderá ser feito por intercâmbio com pesquisadores da área médica. O tempo de execução poderá ser melhorado com uma implementação em GPU, além de aperfeiçoar a visualização dos cortes produzidos com técnicas de *bump mapping*. Espera-se que o sistema também possa ter como novos estudos de caso outros órgãos humanos, além de integrar o método desenvolvido com simuladores virtuais.

REFERÊNCIAS

BLENDER FOUNDATION. **Blender**. Amsterdam, Netherlands: Blender Institute, 2015. Disponível em: <http://www.blender.org>. Acesso em: 12 jan. 2015.

CATMULL, E. **A Subdivision Algorithm for Computer Display of Curved Surfaces**. 1st. ed. Salt Lake City, USA: Computer Science, 1974. 136 p. (University of Utah. Computer Science. UTEC-CSc [series]).

CHIANG, P. et al. A vr simulator for intracardiac intervention. **IEEE computer graphics and applications**, Los Alamitos, USA, v. 33, n. 1, p. 44-57, Apr. 2012.

CLAVIEN, P. A. et al. Strategies for safer liver surgery and partial liver transplantation. **New England Journal of Medicine**, Boston, USA, v. 356, n. 15, p. 1545-1559, Apr. 2007.

COURTECUISSE, H. et al. Gpu-based real-time soft tissue deformation with cutting and haptic feedback. **Progress in Biophysics and Molecular Biology**, Amsterdam, Netherlands, v. 103, n. 2, p. 159-168, Sep. 2010.

CUTLER, B. et al. A procedural approach to authoring solid models. **ACM Transactions on Graphics**, New York, USA, v. 21, n. 3, p. 302-311, July 2002.

DAWSON, S. L. et al. Equipment and technology-designing a computer-based simulator for interventional cardiology training. **Catheterization and Cardiovascular Interventions**, New York, USA, v. 51, n. 4, p. 522-527, Dec. 2000.

DAWSON, S. L.; KAUFMAN, J. A. The imperative for medical simulation. **Proceedings...**, Los Alamitos, USA, v. 86, n. 3, p. 479-483, Mar. 1998.

DELINGETTE, H.; AYACHE, N. Hepatic surgery simulation. **Communications of the ACM**, New York, USA, v. 48, n. 2, p. 31-36, Feb. 2005.

DELINGETTE, H.; COTIN, S.; AYACHE, N. A hybrid elastic model allowing real-time cutting, deformations and force-feedback for surgery training and simulation. In: IEEE. **Proceedings...** Los Alamitos, USA, May 1999. p. 70-81.

DELINGETTE, H. et al. Craniofacial surgery simulation testbed. In: **Visualization in Biomedical Computing 1994**. Bellingham, USA: [s.n.], Sep. 1994. p. 607-618.

DISCHLER, J. M.; GHAZANFARPOUR, D.; FREYDIER, R. Anisotropic solid texture synthesis using orthogonal 2d views. In: WILEY ONLINE LIBRARY. **Computer Graphics Forum**. Oxford, UK, Sep. 1998. v. 17, n. 3, p. 87-95.

DONG, Y. et al. Lazy solid texture synthesis. In: EUROGRAPHICS ASSOCIATION. **Proceedings...** Aire-la-Ville, Switzerland, June 2008. p. 1165-1174.

DU, S.-P.; HU, S.-M.; MARTIN, R. R. Semiregular solid texturing from 2d image exemplars. **IEEE Transactions on Visualization and Computer Graphics**, Los Alamitos, USA, v. 19, n. 3, p. 460–469, Mar. 2013. ISSN 1077-2626.

ECHEGARAY, G. et al. A brain surgery simulator. **Computer Graphics and Applications, IEEE**, Los Alamitos, USA, v. 34, n. 3, p. 12-18, May/June 2014.

ELHELW, M. A. et al. Real-time photo-realistic rendering for surgical simulations with graphics hardware. In: **Medical Imaging and Augmented Reality**. New York, USA: Springer, Aug. 2004. p. 346-352.

ENDO, K. et al. A patient-specific surgical simulator using preoperative imaging data: an interactive simulator using a three-dimensional tactile mouse. **Journal of Computational Surgery**, New York, USA, v. 3, n. 1, p. 1-8, Aug. 2014.

GHAZANFARPOUR, D.; DISCHLER, J.-M. Generation of 3d texture using multiple 2d models analysis. In: WILEY ONLINE LIBRARY. **Computer Graphics Forum**. Oxford, UK, Aug. 1996. v. 15, n. 3, p. 311-323.

GHAZANFARPOUR, D.; DISCHLER, J.-M. Spectral analysis for automatic 3-d texture generation. **Computers & Graphics**, Amsterdam, Netherlands, v. 19, n. 3, p. 413-422, May/June 1995.

HEIMANN, T. et al. Comparison and evaluation of methods for liver segmentation from ct datasets. **Medical Imaging, IEEE Transactions on**, Los Alamitos, USA, v. 28, n. 8, p. 1251-1265, Feb. 2009.

KOPF, J. et al. Solid texture synthesis from 2d exemplars. **ACM Transactions on Graphics**, New York, USA, v. 26, n. 3, p. 2, July, 2007.

KUNKLER, K. The role of medical simulation: an overview. **The International Journal of Medical Robotics and Computer Assisted Surgery**, Oxford, UK, v. 2, n. 3, p. 203-210, Sep. 2006.

KWATRA, V. et al. Texture optimization for example-based synthesis. In: **ACM Transactions on Graphics**. New York, USA: [s.n.], Jul. 2005. v. 24, n. 3, p. 795-802.

LIM, Y.; JIN, W.; DE, S. On some recent advances in multimodal surgery simulation: A hybrid approach to surgical cutting and the use of video images for enhanced realism. **Presence**, Cambridge, USA, v. 16, n. 6, p. 563-583, Dec. 2007.

MARESCAUX, J. et al. Virtual reality applied to hepatic surgery simulation: the next revolution. **Annals of surgery**, Philadelphia, USA, v. 228, n. 5, p. 627, Nov. 1998.

MEVIS MEDICAL SOLUTIONS AG. **MeVisLab**. Bremen, Germany: MeVis Medical Solutions AG, 2015. Disponível em: <http://www.mevislab.de>. Accesso em: 12 jan. 2015.

MISE, Y. et al. Virtual liver resection: computer-assisted operation planning using a three-dimensional liver representation. **Journal of hepato-biliary-pancreatic sciences**, Oxford, UK, v. 20, n. 2, p. 157-164, Feb. 2013.

NUNES, A. L. P.; WALTER, M.; MACIEL, A. Proposta de um modelo visualmente realístico para simulação virtual de laparoscopia orientada a dados médicos. In: **SIBGRAPI 2012 - Conference on Graphics, Patterns and Images - Workshop of Theses and Dissertations**. [S.l.: s.n.], ago. 2012.

OWADA, S. et al. Volume painter: Geometry-guided volume modeling by sketching on the cross-section. In: EUROGRAPHICS ASSOCIATION. **Proceedings...** Aire-la-Ville, Switzerland, 2008. p. 9-16.

OWADA, S. et al. Volumetric illustration: designing 3d models with internal textures. **ACM Transactions on Graphics**, New York, USA, v. 23, n. 3, p. 322-328, Aug. 2004.

PEACHEY, D. R. Solid texturing of complex surfaces. In: ACM. **ACM SIGGRAPH Computer Graphics**. New York, USA, Jul. 1985. v. 19, n. 3, p. 279-286.

PERLIN, K. An image synthesizer. **SIGGRAPH Comput. Graph.**, New York, USA, v. 19, n. 3, p. 287-296, Jul. 1985.

PIETRONI, N. et al. Solid-texture synthesis: a survey. **Computer Graphics and Applications, IEEE**, Los Alamitos, USA, v. 30, n. 4, p. 74-89, Jan. 2010.

PIETRONI, N. et al. Texturing internal surfaces from a few cross sections. In: WILEY ONLINE LIBRARY. **Computer Graphics Forum**. Oxford, UK, Sep. 2007. v. 26, n. 3, p. 637-644.

PRAUN, E.; FINKELSTEIN, A.; HOPPE, H. Lapped textures. In: ACM PRESS/ADDISON-WESLEY PUBLISHING CO. **Proceedings...** New York, USA, July 2000. p. 465-470.

RAGHUPATHI, L. et al. An intestinal surgery simulator: Real-time collision processing and visualization. **Visualization and Computer Graphics, IEEE Transactions on**, Los Alamitos, USA, v. 10, n. 6, p. 708-718, Nov./Dec. 2004.

SATAVA, R. M. Virtual reality surgical simulator. **Surgical endoscopy**, New York, USA, v. 7, n. 3, p. 203-205, May/June 1993.

SATOH, J. et al. Simulation of surgical operations based on solid modeling. In: **Visual Computing**. Tokio, Japan: Springer, May. 1992. p. 907-916.

SHEWCHUK, J. R. Triangle: Engineering a 2d quality mesh generator and delaunay triangulator. In: **Applied computational geometry towards geometric engineering**. Berlin, Germany: Springer, May 1996. p. 203-222.

SHINDO, T. et al. 3d visualization of liver and its vascular structures and surgical planning system-surgical simulation. In: IEEE. **Computer Sciences and Convergence Information Technology (ICCIT), 2011 6th International Conference on**. Los Alamitos, USA, Dec. 2011. p. 939-944.

TAGAWA, K. et al. Laparoscopic surgery simulator using first person view and guidance force. In: **MMVR**. San Diego, USA: [s.n.], Jan. 2013. p. 431-435.

TAKAYAMA, K. et al. Lapped solid textures: filling a model with anisotropic textures. In: ACM. **ACM Transactions on Graphics**. New York, USA, Aug. 2008. v. 27, n. 3, p. 53.

TAKAYAMA, K. et al. Volumetric modeling with diffusion surfaces. **ACM Transactions on Graphics**, New York, USA, v. 29, n. 6, p. 180, Dec. 2010.

VARSHNEY, R. et al. Development of the mcgill simulator for endoscopic sinus surgery: A new high-fidelity virtual reality simulator for endoscopic sinus surgery. **American journal of rhinology & allergy**, East Providence, USA, v. 28, n. 4, p. 330-334, July/Aug. 2014.

WU, J.; WESTERMANN, R.; DICK, C. Physically-based simulation of cuts in deformable bodies: A survey. In: **Eurographics 2014-State of the Art Reports**. Strasbourg, France: [s.n.], Apr. 2014. p. 1-19.

WU, J.; WESTERMANN, R.; DICK, C. Real-time haptic cutting of high-resolution soft tissues. **Studies in Health Technology and Informatics**, Manhattan Beach, USA, v. 196, p. 469-475, Feb. 2014.

XUE-MEI, L.; AI-MIN, H.; QIN-PING, Z. Organ texture synthesis for virtual reality-based surgical simulators. In: IEEE. **Computer Science and Engineering, 2009. WCSE'09. Second International Workshop on**. Quingdao, China, Oct. 2009. v. 1, p. 238-241.

YANG, X. et al. Development of a user-centered virtual liver surgery planning system. In: HUMAN FACTORS AND ERGONOMICS SOCIETY. **Proceedings...** [S.l.], 2012. v. 56, n. 1, p. 772-776.

YIANNAKOPOULOU, E. et al. Virtual reality simulators and training in laparoscopic surgery. **International Journal of Surgery**, Amsterdam, Netherlands, v. 13, p. 60-64, Jan. 2015.

YOUNGBLOOD, P. L. et al. Comparison of training on two laparoscopic simulators and assessment of skills transfer to surgical performance. **Journal of the American College of Surgeons**, Amsterdam, Netherlands, v. 200, n. 4, p. 546-551, Apr. 2005.