

**Guilherme Rossetti Anzollin**

***Interpolação de Curvas com Mudança Topológica***

Joinville / SC

Dezembro de 2006

**Guilherme Rossetti Anzollin**

***Interpolação de Curvas com Mudança Topológica***

Orientador: Prof. Marcelo da Silva Hounsell, PhD

UDESC - UNIVERSIDADE DO ESTADO DE SANTA CATARINA  
CENTRO DE CIÊNCIAS TECNOLÓGICAS  
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

Joinville / SC

Dezembro de 2006

Monografia sob o título “*Interpolação de Curvas com Mudança Topológica*”, defendida por Guilherme Rossetti Anzollin e aprovada em dezembro de 2006, em Joinville, Santa Catarina, pela banca examinadora constituída pelos professores:

---

Prof. Marcelo da Silva Hounsell, PhD  
Departamento de Ciência da Computação - UDESC  
Orientador

---

Prof. Alexandre Gonçalves Silva, MSc  
Departamento de Ciência da Computação - UDESC

---

Prof. Roberto Silvio Ubertino Rosso Jr, PhD  
Departamento de Ciência da Computação - UDESC

---

Prof. Rogério Eduardo da Silva, MSc  
Departamento de Ciência da Computação - UDESC

---

Prof<sup>a</sup>. Tania Martins Preto, MEng  
UTFPR

*Inicialmente à Deus por tudo que tenho.*

*Aos meus pais Sandra e Geraldo e minha irmã Gabriela,  
dos quais sempre recebi tanto carinho e incentivo.*

*À minha namorada Vanessa por todo amor e apoio que recebi  
durante toda esta jornada.*

## *Agradecimentos*

- À meu orientador por seu apoio em todo momento;
- Aos professores Luis Gustavo Nonato e Alex Jesus Cuadros-Vargas pela atenção e pelos materiais enviados para estudo;
- À professora Tania Martins Preto pelo interesse e contribuição no trabalho;
- Aos meus caros amigos que me apoiaram durante todo o curso;
- Aos demais professores que contribuíram para o desenvolvimento desta monografia.

*“A mente que se abre a uma nova idéia jamais volta ao seu tamanho original”*

***Albert Einstein***

# *Resumo*

Neste trabalho, uma técnica de reconstrução tridimensional baseada na interpolação de curvas com mudança de topologia é apresentada. Para resolver este problema com aplicações na área médica e na representação computacional de terrenos, conceitos e técnicas de modelagem geométrica foram estudados, como a criação e representação de modelos tridimensionais a partir de fatias bidimensionais. Uma aplicação foi implementada para demonstrar a solução proposta para este problema de maneira eficiente, considerando conceitos de reconstrução tridimensional e realizando testes para a validação do trabalho.

# *Abstract*

In this work a three-dimensional reconstruction technique based on interpolation of curves with changing topology is presented. To solve this problem, that has applications in the medical area and in the computational representation of terrains, concepts of geometric modeling was studied, as the synthesis and representation of three-dimensional models through the bidimensional slices. An application was implemented to demonstrate the propose solution for this problem in an efficient way, considering concepts of three-dimensional reconstruction and tests for the validation of the work.



# *Sumário*

## **Lista de Figuras**

<b>1</b>	<b>Introdução</b>	<b>14</b>
1.1	Objetivos . . . . .	16
1.2	Estrutura do Trabalho . . . . .	17
<b>2</b>	<b>Fundamentação Teórica</b>	<b>18</b>
2.1	Curvas . . . . .	18
2.1.1	Representação de curvas . . . . .	18
2.1.2	Centróide . . . . .	19
2.1.3	Tipos de curvas . . . . .	20
2.2	Criação e Representação de Sólidos . . . . .	21
2.2.1	Representação B-rep . . . . .	21
2.3	VRML - <i>Virtual Reality Modeling Language</i> . . . . .	23
2.4	XML - <i>eXtensible Markup Language</i> . . . . .	25
<b>3</b>	<b>Reconstrução 3D</b>	<b>27</b>
3.1	Correspondência . . . . .	27
3.2	Bifurcação . . . . .	29

3.3	Geração de Malha . . . . .	30
3.4	Abordagens da Reconstrução 3D . . . . .	31
<b>4</b>	<b>Trabalhos Relacionados</b>	<b>37</b>
4.1	Algoritmo de Barequet e Sharir (1996) . . . . .	37
4.2	Algoritmo de Treece, Prager, Gee e Berman (1999) . . . . .	38
4.3	Algoritmo $\beta$ -Connection (2001) . . . . .	41
<b>5</b>	<b>Algoritmo <math>\Delta</math>-connection</b>	<b>44</b>
5.1	Correspondência no $\Delta$ -connection . . . . .	46
5.2	Geração de Malha no $\Delta$ -connection . . . . .	48
5.3	Bifurcação no $\Delta$ -connection . . . . .	50
<b>6</b>	<b>Projeto, Implementação e Melhoramentos</b>	<b>51</b>
6.1	Arquitetura do $\Delta$ -connection . . . . .	51
6.2	Interface . . . . .	52
6.3	Armazenamento de dados . . . . .	54
6.4	Interpolação: $\Delta$ -connection . . . . .	56
6.5	Melhoramentos . . . . .	58
6.5.1	Definição do Ponto Inicial . . . . .	60
6.5.2	Balanceamento de Pontos . . . . .	61
<b>7</b>	<b>Resultados</b>	<b>63</b>
7.1	Dados Editados . . . . .	63
7.2	Dados Reais . . . . .	67

7.3	Discussão . . . . .	72
<b>8</b>	<b>Conclusão</b>	<b>75</b>
8.1	Trabalhos Futuros . . . . .	76
	<b>Referências</b>	<b>77</b>
	<b>Apêndice A – Dados de fontes Reais</b>	<b>79</b>
	<b>Apêndice B – Dados Reais adaptados para XML</b>	<b>81</b>
	<b>Apêndice C – Algoritmos utilizados para o cálculo de centróides e distâncias</b>	<b>82</b>
C.1	Cálculo dos centróides . . . . .	82
C.2	Cálculo da matriz de distâncias . . . . .	83

## *Lista de Figuras*

1.1	Reconstrução de um crânio. Fonte: Bajaj et al (1996) . . . . .	15
1.2	Reconstrução de vasos sanguíneos. Fonte: Nonato et al (2005) . . . . .	15
1.3	Interpolação de curvas . . . . .	16
2.1	Representação Matricial e Poligonal. Fonte: Gatass e Peixoto (2000) . . . . .	19
2.2	Diferentes cálculos de centróide . . . . .	20
2.3	Curvas fechadas convexa e côncava . . . . .	21
2.4	Curvas intersectantes e auto-interceptante . . . . .	21
2.5	Representação por superfícies. Fonte: Castelo Filho (1998) . . . . .	22
2.6	Geometria e Topologia . . . . .	23
2.7	Documento VRML . . . . .	24
2.8	Documento XML . . . . .	25
3.1	Diferentes Correspondências . . . . .	28
3.2	Bifurcações - <i>Branching</i> . Fonte: Gatas e Peixoto (2000) . . . . .	29
3.3	Triangulação - <i>Tiling</i> . Fonte: Meyers et al (1994) . . . . .	30
3.4	Diferentes Malhas - <i>Tiling</i> . Fonte: Bajaj et al (1996) . . . . .	31
3.5	Reconstrução a partir de deformações. Fonte: McInerney e Terzopoulos (1996) . . . . .	32
3.6	Reconstrução a partir de abordagem implícita . . . . .	33
3.7	Geração de malha auxiliada por um Grafo. Fonte: Meyers et al (1994) . . . . .	34

3.8	Superfícies geradas por algoritmos diferentes. Fonte: Meyers et al (1994) . . . . .	34
4.1	Definição da correspondência . . . . .	38
4.2	Objeto reconstruído sem bifurcação. Fonte: Gatass e Peixoto (2000) . . . . .	38
4.3	Definição dos discos para as curvas. Fonte: Treece et al (1999) . . . . .	39
4.4	Comparação de distâncias entre centróides. Fonte: Treece et al (1999) . . . . .	40
4.5	Resultado da correspondência e objeto final. Fonte: Treece et al (1999) . . . . .	40
4.6	Diferentes valores para $\beta$ . Fonte: Vargas (2001) . . . . .	42
4.7	Classificação e eliminação de tetraedros. Fonte: Vargas (2001) . . . . .	42
5.1	Diferentes abordagens para reconstrução. Fonte: Meyers et al (1992) . . . . .	45
5.2	Matriz de distâncias . . . . .	47
5.3	Curva sem nenhuma correspondência . . . . .	48
5.4	Representação <i>B-rep em VRML</i> . . . . .	49
5.5	Problemas ocorridos com a geração de malha . . . . .	50
5.6	Local da bifurcação - <i>Branching</i> . . . . .	50
6.1	Arquitetura do $\Delta$ -connection . . . . .	52
6.2	Interface . . . . .	53
6.3	Armazenamento em XML . . . . .	55
6.4	Algoritmo $\Delta$ -connection . . . . .	56
6.5	Diagrama de classes da aplicação desenvolvida . . . . .	58
6.6	Possível solução de <i>tiling</i> . . . . .	59
6.7	Possível problema durante o <i>tiling</i> . . . . .	59
6.8	Definição do ponto inicial . . . . .	60

6.9	Balanceamento de pontos . . . . .	61
6.10	Triangulação para geração de malha . . . . .	62
7.1	Curvas convexas e côncavas . . . . .	64
7.2	Curvas com formas distintas . . . . .	64
7.3	Mudança de topologia . . . . .	65
7.4	Diferentes valores para $\Delta$ . . . . .	66
7.5	Canais . . . . .	67
7.6	Reconstrução de um fêmur . . . . .	68
7.7	Reconstrução de pulmões . . . . .	69
7.8	Reconstrução de canais (vasos sanguíneos) . . . . .	70
7.9	Reconstrução de um coração . . . . .	71
7.10	Análise da variação do $\Delta$ . . . . .	72
7.11	Análise do tempo de reconstrução . . . . .	73
7.12	Site do trabalho . . . . .	74

# *1 Introdução*

A Computação Gráfica é a área da computação destinada à geração de imagens em geral, tanto na forma de representação de dados quanto na forma de recriação do mundo real. A área de aplicação é enorme e vem crescendo a cada dia. Uma das áreas que evoluiu foi a Visualização Científica, que tem por objetivo resumir dados científicos (fluxo de fluidos, relatividade, reações químicas e nucleares, etc) e identificá-los através de representações gráficas a fim de facilitar a interpretação desses dados, que geralmente são em grande quantidade [Traina e Oliveira, 2004].

Segundo Vargas (2001), a aparição de dispositivos de medição não invasivos, como Tomografia por Ressonância Magnética e Tomografia Computadorizada, possibilitou a visualização de uma sequência de seções planares de objetos tridimensionais. Este fato motivou o desenvolvimento de várias técnicas de Reconstrução Tridimensional.

A Reconstrução Tridimensional então passou a ser uma técnica de bastante interesse e pesquisa, pois a partir de imagens bidimensionais extraídas por tais dispositivos é possível determinar o modelo 3D do objeto que foi analisado. A Figura 1.1 ilustra uma das fatias obtidas por Tomografia Computadorizada (a); o processo de reconstrução executado sobre duas dessas fatias (b) e o processo finalizado (c) de reconstrução de um crânio.

Um outro exemplo de reconstrução a partir de fatias é a representação de canais, onde a visualização do modelo como um todo é mais importante que a visualização de detalhes entre as fatias. A Figura 1.2 mostra seções bidimensionais que representam vasos sanguíneos (a) e em seguida a interpolação destas fatias gerando todos os canais (b).

Neste trabalho são estudadas técnicas de modelagem geométrica para reconstrução tridi-

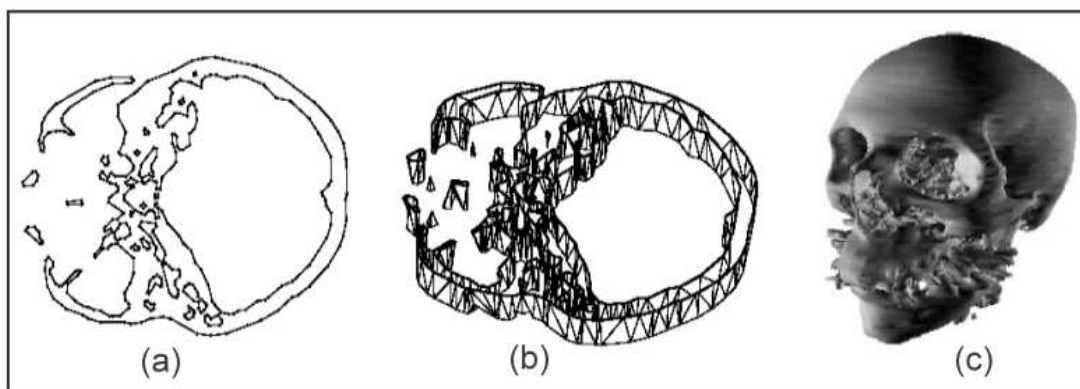


Figura 1.1: Reconstrução de um crânio. Fonte: Bajaj et al (1996)

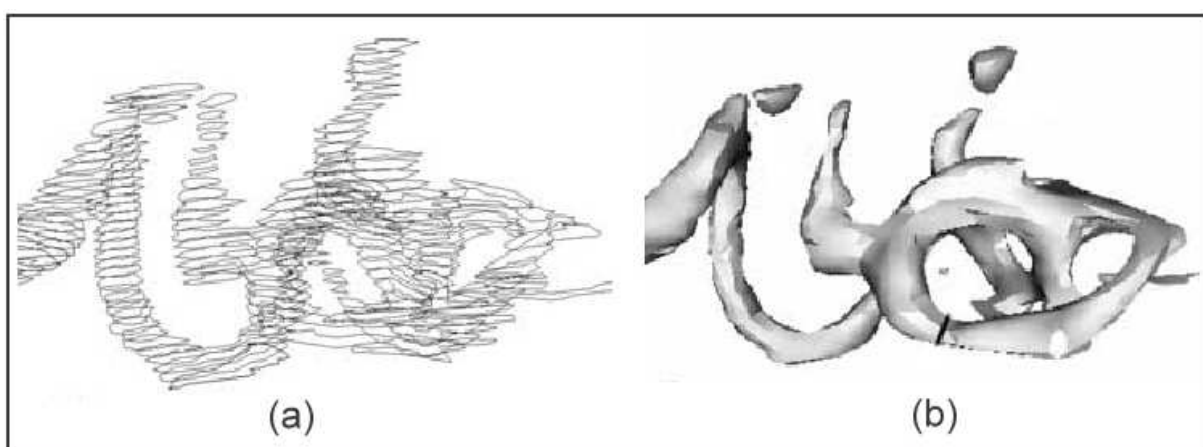


Figura 1.2: Reconstrução de vasos sanguíneos. Fonte: Nonato et al (2005)

dimensional através da interpolação de curvas considerando mudança de topologia.

Uma definição formal para o problema seria:

Dado um conjunto de curvas  $c_{ij}$ , onde  $i=1, 2, \dots, n$  e  $n$  é o número de curvas diferentes no plano  $x_j$ , criar um objeto  $O$  sendo que as curvas  $c_{ij}$  sejam a interseção de  $O$  com os planos  $x_j$ . Cada curva  $c_{ij}$  é representada por uma sequência de  $p_i$  pontos.

No que diz respeito à mudança topológica, é o fato de uma curva, em um plano, se conectar com mais de uma no plano seguinte deixando a idéia de que a curva se transformou em duas ou mais, daí a mudança topológica de uma para duas ou mais sequências de pontos. A Figura 1.3 ajuda a ilustrar o problema, mostrando em (a) as curvas fechadas em seus respectivos planos, e em (b) a superfície do objeto gerado a partir da interpolação.

Como a Figura 1.3 pode mostrar, uma fatia  $c_i$  pode conter uma ou mais curvas fechadas



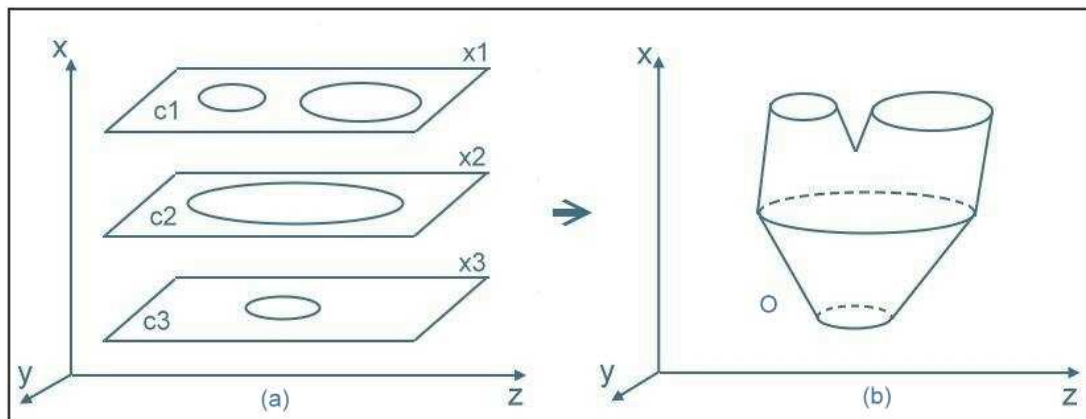


Figura 1.3: Interpolação de curvas

independentes no mesmo plano, gerando assim uma mudança topológica na interpolação entre as fatias  $c_i$  e  $c_{i+1}$ . Como nada é definido para o comportamento das curvas no entre-planos, percebe-se que podem existir vários objetos que satisfaçam ao enunciado, ou seja, através de seções com mudança topológica é possível realizar diferentes interpolações, gerando diferentes objetos. Este trabalho visa obter uma solução algorítmica para o caso de interpolações lineares, a fim de estabelecer critérios para a interpolação quando ocorrer mudança topológica entre as seções.

## 1.1 Objetivos

O objetivo geral deste Trabalho de Conclusão de Curso (TCC) é realizar a reconstrução tridimensional através do desenvolvimento de uma solução eficiente para a interpolação entre curvas com mudança topológica. Os objetivos específicos são os seguintes:

- Entender os conceitos e técnicas de modelagem geométrica para a criação e representação de objetos;
- Identificar técnicas existentes na literatura que sejam apropriadas à interpolação de curvas (reconstrução 3D);
- Estabelecer uma solução para a interpolação de curvas considerando critérios de eficiência (a serem levantados e estabelecidos);

- Projetar e implementar a solução proposta.

Ao final deste TCC será obtido um melhor entendimento do problema da interpolação de curvas e reconstrução tridimensional; será apresentada a descrição detalhada de uma solução; estará disponível um *software* que demonstre a solução adotada gerando objetos exportados para VRML<sup>1</sup> (*Virtual Reality Modeling Language*) e haverá um *site* com a descrição dos principais aspectos do projeto, o *software*, exemplos de objetos gerados e referências.

## 1.2 Estrutura do Trabalho

Este Trabalho de Conclusão de Curso foi organizado da seguinte maneira:

- O capítulo 2 apresenta inicialmente alguns conceitos sobre curvas e a técnica de representação de sólidos utilizada. Posteriormente são descritos alguns fundamentos em relação às tecnologias escolhidas para a modelagem dos objetos e para o armazenamento de dados;
- O capítulo 3 apresenta um estudo feito sobre reconstrução 3D. São descritos alguns fundamentos e abordagens utilizadas para a reconstrução 3D, seguidas de um comparativo.
- No capítulo 4 são apresentados, em ordem cronológica, três trabalhos correlatos ao tema proposto, desenvolvidos a partir de uma mesma abordagem de reconstrução;
- O capítulo 5 detalha a solução proposta para este trabalho, apresentando as etapas de reconstrução que foram desenvolvidas;
- O capítulo 6 apresenta o projeto da implementação, mostrando a arquitetura, interface, um pseudo-código do algoritmo, o diagrama de classes e as modificações realizadas;
- O capítulo 7 mostra os resultados e discussões do trabalho;
- No capítulo 8 são apresentadas as conclusões e propostos alguns trabalhos futuros.

---

<sup>1</sup>Linguagem que permite descrever elementos geométricos e ambientes virtuais

## 2 *Fundamentação Teórica*

Os objetos reconstruídos devem ser armazenados de forma a atenderem requisitos de Modelagem Geométrica. A Modelagem Geométrica é uma área da Computação Gráfica que estuda, através de métodos matemáticos, mecanismos de criação de modelos geométricos, tratando da síntese, manipulação e topologia dos objetos gráficos no computador [Dieguez, 1989; Foley et al, 1996].

### 2.1 **Curvas**

Na Modelagem Geométrica, a representação de curvas é a base para a descrição de vários modelos, desde formas simples até objetos complexos, e tem um papel importante também para a visualização de fenômenos científicos [Azevedo e Conci, 2003]. A seguir serão apresentados alguns conceitos de curvas que serão utilizados neste trabalho.

#### 2.1.1 **Representação de curvas**

Existem basicamente duas maneiras de representar as curvas (contornos): a representação matricial e a representação poligonal. A Figura 2.1 apresenta uma mesma curva sendo representada das duas maneiras.

Na representação matricial, a curva é descrita por cada ponto que é armazenado em um arranjo bidimensional (matriz) que é formada por  $i \times j$  elementos, esses elementos são retângulos denominados *pixels*. Um elemento  $(i,j)$  pode estar sobre, dentro ou fora do contorno que representa a curva, Os contornos são definidos por todos os elementos (*pixels*) situados sobre

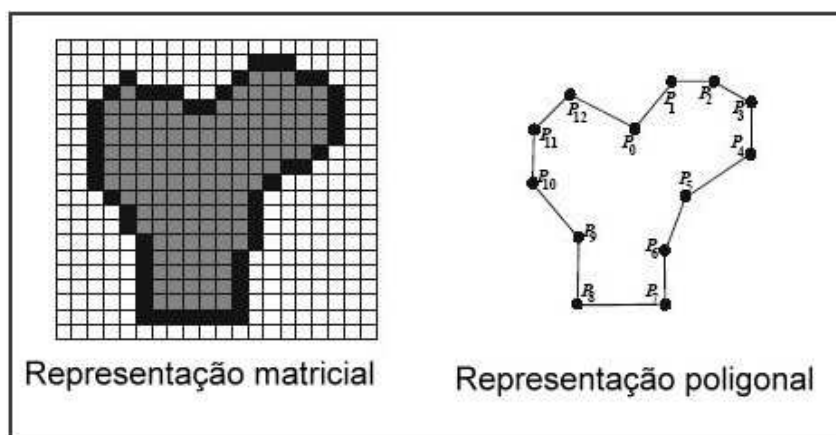


Figura 2.1: Representação Matricial e Poligonal. Fonte: Gattass e Peixoto (2000)

eles. A forma matricial de representação é utilizada para formar as imagens nas telas da maioria dos dispositivos de saída gráficos, como impressora e vídeos [Gattass e Peixoto, 2000]. Na representação poligonal, também denominada vetorial, a curva pode ser representada por uma seqüência de retas, sendo que cada reta é definida pelas coordenadas de seus pontos extremos. Assim a curva é descrita pelas coordenadas de seus respectivos vértices [Azevedo e Conci, 2003].

### 2.1.2 Centróide

A partir da representação poligonal de curvas pode-se obter algumas características como, por exemplo, a região central aproximada da mesma, que é calculada com base nas coordenadas dos vértices, e é uma informação necessária em muitas situações práticas como, por exemplo, para definir o ponto de lançamento automático de textos gráficos para identificação de elementos em tela. Esta região central é também chamada na literatura de “centróide” e pode ser calculada de diferentes maneiras, como através do centro de um círculo inscrito ou circunscrito à curva ou pela média aritmética de todos os vértices da curva (que é uma forma freqüentemente usada) [Figueiredo e Carvalho, 1991]. A Figura 2.2. apresenta três resultados de cálculo do centróide para uma mesma curva fechada: em (a) o centróide é calculado pela média aritmética dos pontos da curvas; em (b) é feita a média aritmética apenas dos pontos x e y extremos da curva; e em (c) é utilizado um círculo inscrito a partir dos pontos da curva e então calculado o centro deste

círculo.

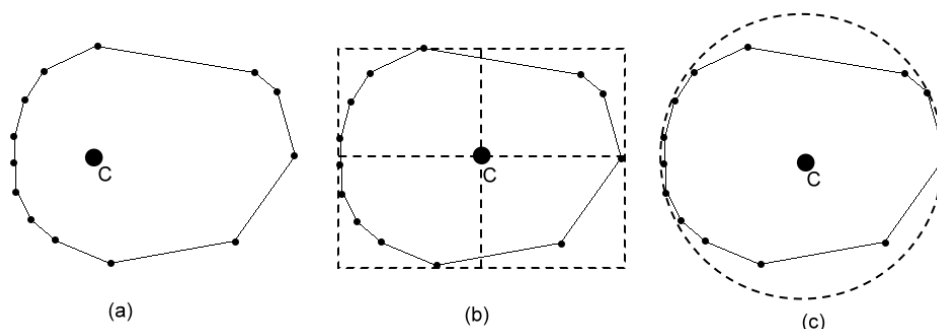


Figura 2.2: Diferentes cálculos de centróide

Outra técnica para calcular o centróide de uma curva é através do cálculo do “centro de gravidade”, esta técnica consiste em dividir a curva em triângulos e fazer uma relação do centro de gravidade de todos eles, já que o centro de gravidade de um triângulo pode ser obtido facilmente pela média aritmética das coordenadas de seus vértices, esta técnica é precisa, porém de grande custo computacional. O cálculo do centróide a partir da média dos vértices da curva, ilustrado em (a) na Figura 2.2, é uma solução de baixo custo computacional, porém pode ter o resultado afetado por características do objeto, por exemplo, a concentração de vértices em uma região da curva causa um deslocamento indesejável no centróide em direção a essa região [Figueiredo e Carvalho, 1991], é o que pode ser observado pelo posicionamento do centróide na Figura 2.2 (a) em relação à (b) e (c). Por outro lado, a técnica representada em (b) na Figura 2.2, onde é feita a média dos pontos extremos da curva (técnica de *Bounding Box*), é de baixo custo computacional e não depende da distribuição dos pontos na curva, ou seja, resolve o cálculo do centróide de maneira satisfatória.

### 2.1.3 Tipos de curvas

Este trabalho trata de curvas fechadas para a interpolação. A curva poderá ser convexa ou côncava, a Figura 2.3 ilustra essas duas situações, sendo que a diferença está que nas curvas côncavas um segmento de reta traçado a partir de dois pontos quaisquer ( $p1$ ,  $p2$ ) do interior da curva pode passar por fora desta mesma curva.

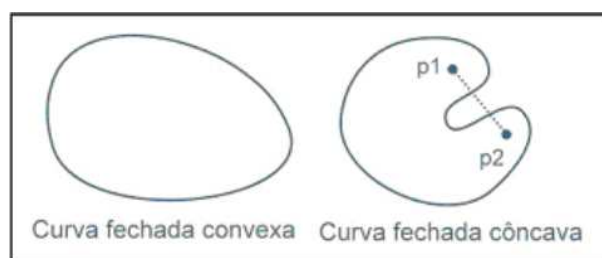


Figura 2.3: Curvas fechadas convexa e côncava

Existem casos em que duas ou mais curvas do mesmo plano possuam pontos de seu interior em comum, tais curvas são denominadas intersectantes. Outro caso que merece ser mencionado é se uma curva possuir pontos de seu contorno em comum, neste caso ela é denominada auto-interceptante. A Figura 2.4 ilustra estes casos demonstrando os pontos  $p$  que identificam a intersecção e a auto-interceptação.

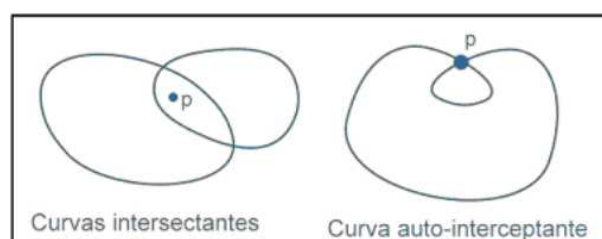


Figura 2.4: Curvas intersectantes e auto-interceptante

Neste trabalho são tratadas para a interpolação as curvas fechadas côncavas e convexas, sendo elas não intersectantes nem auto-interceptantes

## 2.2 Criação e Representação de Sólidos

Os objetos 3D reconstruídos são armazenados utilizando-se uma técnica de representação de sólidos denominada *B-rep* em arquivo VRML, para possibilitar a sua visualização posteriormente. A seguir é descrita a técnica *B-rep* de representação de sólidos.

### 2.2.1 Representação B-rep

Através de superfícies, podem ser representados objetos 3D, esta é uma forma de representação denominada *Boundary Representation (B-rep)*, e consiste na descrição de um objeto

em termos de seus limites de superfícies: vértices, arestas e faces [Foley et al, 1996]. Então, nesta técnica representação, o sólido é descrito como um conjunto de faces, estas limitadas por arestas, que por fim são limitadas por vértices, como representado na Figura 2.5. É através desta relação que é definida a topologia que, juntamente com informações geométricas (coordenadas dos vértices), representa o objeto final [Castelo Filho, 1998].

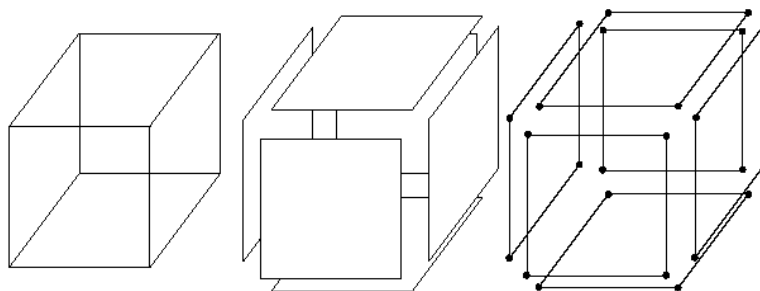


Figura 2.5: Representação por superfícies. Fonte: Castelo Filho (1998)

A representação de sólidos por superfícies possui vantagens sobre a representação por volume, que descreve o objeto varrendo todo o volume de dados que o representam. A representação volumétrica de modelos 3D é muito útil para a representação de modelos que não tem forma definida (como gases ou líquidos), pois não seria possível definir com facilidade sua superfície. A principal vantagem na utilização da técnica de representação de superfícies, em relação a métodos de representação volumétrica é a quantidade de informação armazenada, pois para gerar as faces (superfícies) de um objeto são necessárias bem menos informações do que para gerar seu volume [Gattass e Peixoto, 2000].

Existem diferentes estruturas de dados para a representação por superfícies, sendo que a principal delas, que é utilizada neste trabalho, é a baseada em arestas. Neste tipo de estrutura as faces são descritas através de um ciclo de arestas, determinando a topologia do objeto, e os vértices são obtidos indiretamente a partir dessas arestas, definindo a geometria. A geometria descreve a posição (coordenadas) de cada vértice do objeto no espaço. A topologia representa a relação dos vértices, ou seja, a sequência em que eles se conectam formando então as faces do objeto, sempre descritos em um mesmo sentido (horário ou anti-horário) para todas as faces [Azevedo e Conci, 2003]. A Figura 2.6 ilustra a definição da geometria (as coordenadas dos

quatro vértices) e a topologia (seqüência horária, representada pelas setas, dos vértices definindo as quatro faces) de um tetraedro.

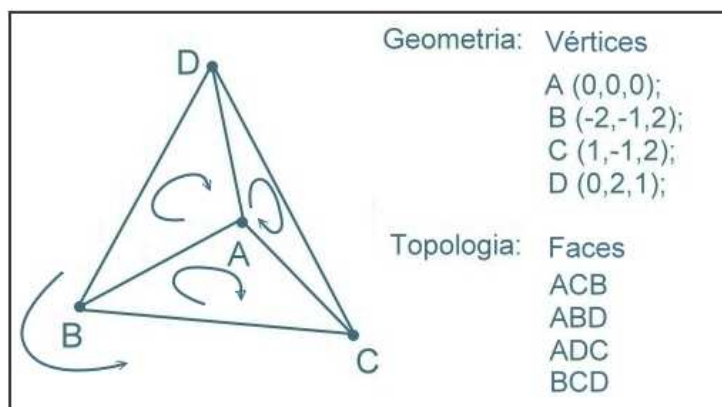


Figura 2.6: Geometria e Topologia

Segundo Speck (2001), a técnica *B-rep* possui vantagens sobre os outros métodos de modelagem, principalmente na geração de modelos complexos e na velocidade de verificação de relações topológicas. Isto ocorre pela maneira como o *B-rep* registra as informações do modelo, armazenando os parâmetros das arestas de forma explícita. Uma vez construída a superfície, aplica-se então técnicas de *rendering* conhecidas para visualizar o objeto.

## 2.3 VRML - Virtual Reality Modeling Language

VRML é uma linguagem de marcação para modelagem em Realidade Virtual e serve para a descrição de objetos e mundos tridimensionais interativos. A versão atual da linguagem (VRML 2.0) trabalha principalmente com a geometria 3D dos objetos que são elaborados a partir de primitivas geométricas (cones, esferas, cilindros e esferas) ou da definição de faces. Também é permitido o uso de objetos multimídia, como sons e filmes, para a interação com o usuário e a programação de eventos para gerar animações. Para a visualização de arquivos VRML é necessário configurar o navegador para que ele interprete o conteúdo do arquivo que está recebendo. Adiciona-se então um *plug-in*, que é um programa que adapta o navegador para que este suporte arquivos de tipos diferenciados. A linguagem VRML utiliza o sistema de coordenadas cartesiano para a modelagem, ou seja, um sistema baseado em três eixos de deslocamento



vetorial, usando a unidade de medidas “metros”, por integrar facilmente os ambientes virtuais escritos por usuários diferentes sem problemas de proporção [Carey e Bell, 1997].

O Arquivo VRML tem a extensão *.wrl* e pode ser editado por qualquer editor de texto. A estrutura do arquivo é formada basicamente pelo cabeçalho VRML e os nós onde são descritos os modelos geométricos. O cabeçalho é obrigatório e representa a versão da linguagem e o conjunto de caracteres utilizado. Os nós são estruturas sintáticas utilizadas para representação dos objetos, a composição de um nó contém o tipo do nó, um par de chaves e um conjunto de atributos do objeto que esta sendo descrito. O arquivo VRML pode contar também com comentários, um comentário de linha é iniciado pelo símbolo “#”.

A Figura 2.7 apresenta um exemplo de código VRML: a primeira linha é o cabeçalho e depois é descrito um nó raiz chamado *shape*, utilizado para descrever os objetos modelados e formado por dois nós internos: o nó *appearance* e o nó *geometry*. O nó *appearance* têm a função de descrever a aparência do objeto a ele associado, o campo *material* define os atributos de aparência do material que compõe o objeto, no caso da Figura 2.7, é aplicada a cor verde através do parâmetro *diffuseColor*. O nó *geometry* define a forma tridimensional do objeto, normalmente ele define uma das primitivas geométricas, como na Figura 2.7, onde é desenhado um cubo através da primitiva *Box*, que define um prisma regular de quatro lados definidos pelo parâmetro *size*.

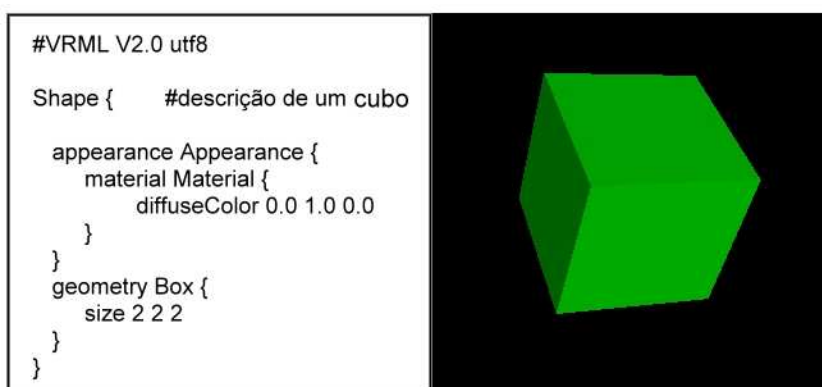


Figura 2.7: Documento VRML

O VRML permite, além de instanciar primitivas geométricas para modelagem, definir a geometria de qualquer tipo de objeto, descrevendo as faces do mesmo. Este trabalho utiliza

deste recurso que é descrito no capítulo 4, onde a proposta é detalhada.

## 2.4 XML - eXtensible Markup Language

XML é uma linguagem de marcação desenvolvida pela W3C (*World Wide Web Consortium*) que permite ao programador estruturar dados de uma maneira flexível adaptando de acordo com a aplicação. Um documento XML pode ser editado e visualizado por qualquer editor de texto simples. É formado basicamente por *tags* e dados, os *tags* são *strings* limitadas pelos caracteres “<” e “>” e os dados são as *strings* limitadas pelos *tags* que representam o conteúdo armazenado no documento. Cada elemento pertencente ao documento do XML começa com uma *tag* de abertura e termina com uma de fechamento, a *tag* de fechamento é diferenciada da de abertura por uma barra “/” que aparece antes do seu nome. Esta marcação dos dados permite introduzir informações sobre os dados, pois é o programador que define as *tags* que irá utilizar em seu documento [Graves, 2003].

A Figura 2.8 demonstra um exemplo de arquivo XML, formado inicialmente por uma declaração que especifica a versão do XML e o conjunto de caracteres utilizado, na segunda linha é escrito o primeiro elemento do documento, chamado também de elemento raiz, que termina na última linha com o *tag* de fechamento. Os demais elementos são descritos dentro do elemento raiz e podem ser formados por sub-elementos ou então pelo conteúdo que está sendo armazenado.

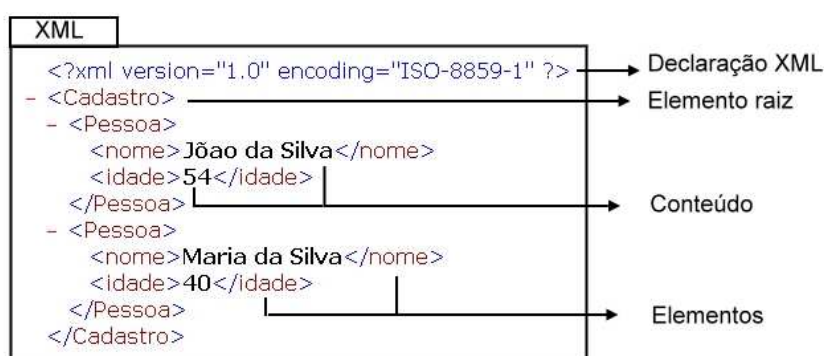


Figura 2.8: Documento XML

Para a organização da estrutura do documento XML podem ser utilizados recursos como o DTD (*Document Type Definition*), que é uma especificação para definir a estrutura do documento (elementos, atributos e notações que serão permitidos), ou o XSD (*XML Schema Definition*), recurso mais recente que funciona de forma análoga ao DTD, porém contando com algumas características a mais para tratar alguns problemas apresentados pelo DTD. Através destes recursos é possível, então, definir um tipo de estrutura, que deverá ser respeitada, para organizar os elementos de um documento XML [Décio, 2000].

A XML é muito útil para transferência de dados entre banco de dados e aplicativos, e tem sido utilizado muito também para o armazenamento de dados. Uma das principais vantagens da utilização de XML como banco de dados é pelo fato de sua estrutura ser mais expressiva do que a estrutura de bancos de dados relacionais, isso por que ela pode conter diferentes elementos, variar a ordem e a quantidade de atributos e possuir elementos iguais, o que torna mais fácil de representar dados complexos [Graves, 2003].

Neste trabalho será utilizado XML para o armazenamento das informações relativas às curvas que serão geradas. A forma como será utilizada a tecnologia para o armazenamento desses dados, sem a utilização de recursos para a definição de estruturas (DTD ou XSD), é apresentada no capítulo 4.

## 3 *Reconstrução 3D*

O problema de reconstrução tridimensional através de seções bidimensionais conta com alguns algoritmos que foram desenvolvidos no decorrer de anos. Diante dos estudos levantados, nota-se que todos esses algoritmos levam em consideração as seguintes etapas para gerar o modelo geométrico desejado: a correspondência, a geração de malha e a ramificação, mais conhecida na literatura por bifurcação. A seguir serão explicadas e exemplificadas essas etapas.

### 3.1 Correspondência

O problema de correspondência (*correspondence*) acontece quando existe mais de uma curva fechada em um dos planos paralelos, ou nos dois, o que pode gerar uma mudança topológica. A etapa de definição da correspondência é considerada o principal problema da reconstrução 3D a partir de seções planares [Vargas, 2001]. Neste caso é preciso decidir qual será a correspondência das curvas entre os planos, ou seja, quais curvas de um plano que vão ser conectadas com quais outras do outro plano. A Figura 3.1 apresenta três das várias soluções de correspondência que podem ser geradas a partir das duas seções iniciais ilustradas no canto superior esquerdo. Em um dos casos cada curva de um plano é conectada com uma única mais próxima do plano seguinte (a), em um outro caso uma das curvas do plano inferior conecta-se com as duas do plano superior, enquanto a outra conecta-se apenas com a mais próxima (b) e na última ilustração apenas uma das curvas do plano inferior se conecta com as curvas do plano superior (c).

O problema de correspondência só se torna trivial quando a relação entre as curvas nos

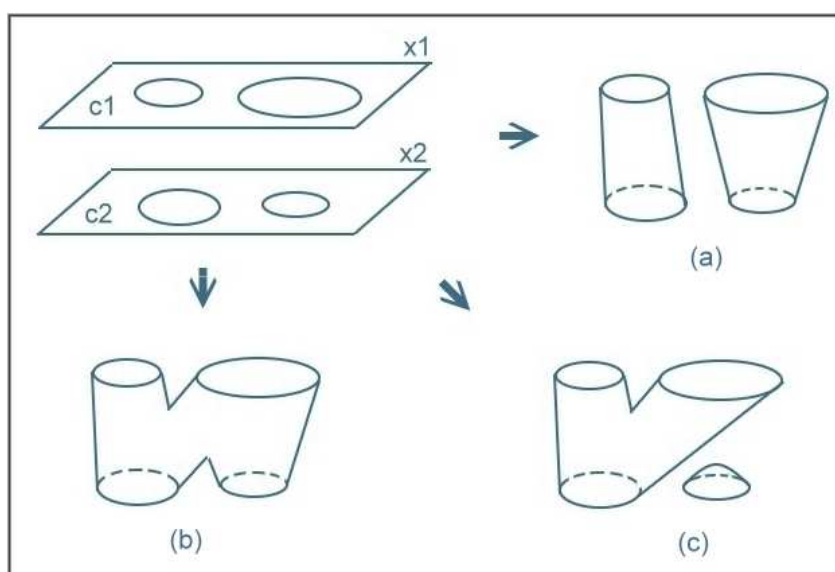


Figura 3.1: Diferentes Correspondências

planos é de 1 para 1 ( $1 \leftrightarrow 1$ ), ou seja, uma curva em cada plano, onde as duas curvas se conectam (superfície contínua) ou não (superfície descontínua). No caso de um dos planos conter uma curva e o outro mais de uma ( $1 \leftrightarrow n$ ) o problema de correspondência já deve levar em consideração algum critério para a conexão, e o mesmo acontece quando são tratadas mais de uma curva nos dois planos ( $m \leftrightarrow n$ ), como foi ilustrado na Figura 3.1.

Segundo Gattass e Peixoto (2000), a decisão da correspondência pode ser tomada com base nas informações das distâncias entre as curvas. De um modo geral, se os contornos se encontram muito distantes entre si, não há conexão entre eles. Mas se o espaçamento entre fatias adjacentes não é muito grande, é possível estabelecer a conexão destes contornos.

Porém uma solução somente baseada em distância pode não gerar todos os objetos desejados a partir de um determinado conjunto de fatias. Para escolha da correspondência, então, seria interessante contar com técnicas flexíveis permitindo que as várias soluções de conexão possam ser realizadas.

## 3.2 Bifurcação

O problema da bifurcação (*branching*) ocorre quando a relação entre as curvas no entre-planos é de  $1 \leftrightarrow n$  ou  $m \leftrightarrow n$ , ou seja, quando existe mudança topológica entre as duas seções. As bifurcações representam os pontos de sela na superfície gerada, e torna o problema de geração de malha mais delicado, pois a definição do local em que será feita a bifurcação não é uma solução trivial. A partir de duas seções com mudança topológica podem ser feitas bifurcações em diferentes regiões do entre-planos. A Figura 3.2 apresenta soluções diferentes de *branching* dados dois conjuntos de curvas iniciais (a).

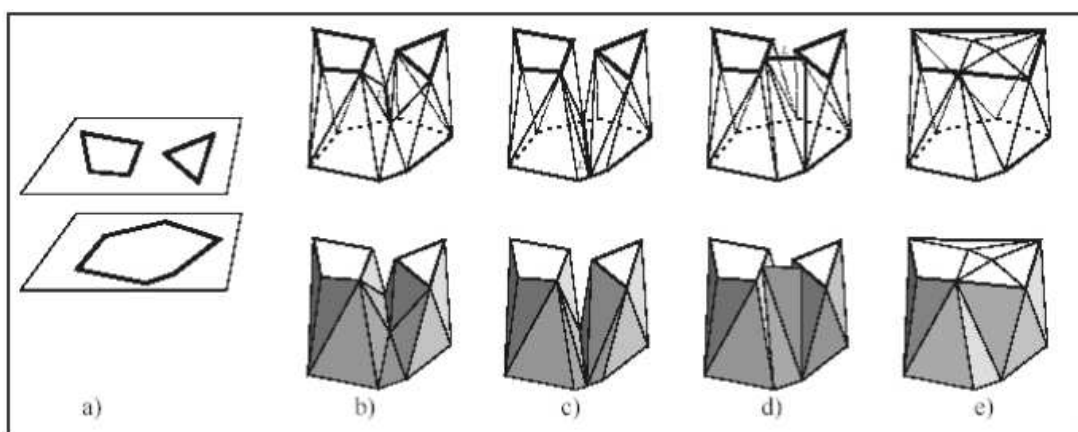


Figura 3.2: Bifurcações - *Branching*. Fonte: Gatas e Peixoto (2000)

Das diferentes soluções ilustradas na Figura 3.2, cada uma representa a bifurcação em um nível do entre-planos. Uma considera a bifurcação justamente no ponto médio entre um plano e o outro (b), o que seria a solução mais próxima do real. Outra solução divide o contorno que não sofre mudança topológica em duas regiões e então efetua o *branching* de cada região com o contorno correspondente (c). A solução (d) efetua o *branching* logo que começa o entre-planos, unindo os contornos com mudança topológica a fim de aumentar a superfície do sólido final. E a ultima solução (e) simplesmente não considera a bifurcação, gerando um contorno que abrange as duas curvas do plano superior, e realizando a interpolação de  $1 \leftrightarrow 1$  com a curva do plano inferior.

Alguns algoritmos trazem soluções eficientes para resolver a etapa de *branching* através de reconstrução volumétrica, outros a consideram apenas como uma consequência da interpolação

com mudança topológica, não tratando detalhadamente a bifurcação. No capítulo 4 serão demonstrados alguns desses algoritmos que trataram desta etapa.

### 3.3 Geração de Malha

A geração de malha *tiling* é o processo de interpolação entre as curvas correspondentes. É nesta etapa que são definidos quais vértices de uma curva serão conectados a quais vértices da curva “correspondente”, gerando assim uma malha triangular que representa a superfície do objeto final.

Com as curvas representadas na forma poligonal o processo de geração de malha consiste na conexão de todos os vértices das curvas fechadas. Tal processo é tratado na literatura como triangulação [Bajaj et al, 1996; Gattass e Peixoto, 2000; Vargas, 2001], pois cada face da superfície final é composta por 3 vértices conectados, sendo dois destes vértices pertencentes à uma curva e um deles à outra. A Figura 3.3 representa o processo de triangulação entre duas curvas, uma representada pelos pontos Q e outra pelos pontos P, e uma das faces formada pelos pontos  $P_1$ ,  $Q_1$  e  $P_2$  da superfície que será gerada.

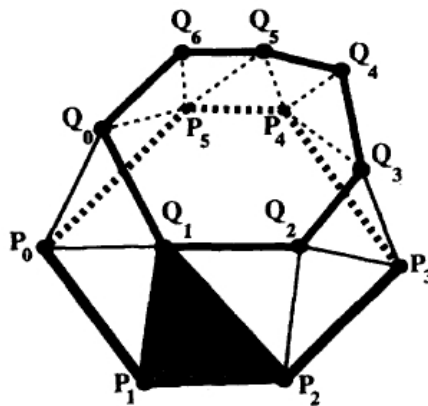


Figura 3.3: Triangulação - *Tiling*. Fonte: Meyers et al (1994)

A geração de malha nem sempre conta com uma solução trivial. A Figura 3.4 ilustra uma projeção de duas curvas (a) estando cada uma em um plano e o processo de geração de malha resultando em duas soluções diferentes. Em uma das soluções (b) a geração de malha é baseada na construção de uma superfície mínima, sem interseção, conectando os pontos mais próximos

de cada curva, na outra solução (c) é feita a geração de malha sem levar em consideração a relação espacial dos pontos das duas curvas, ou seja, um ponto da extrema esquerda da curva superior pode se conectar com um ponto da direita da curva inferior, construindo uma superfície intersectante.

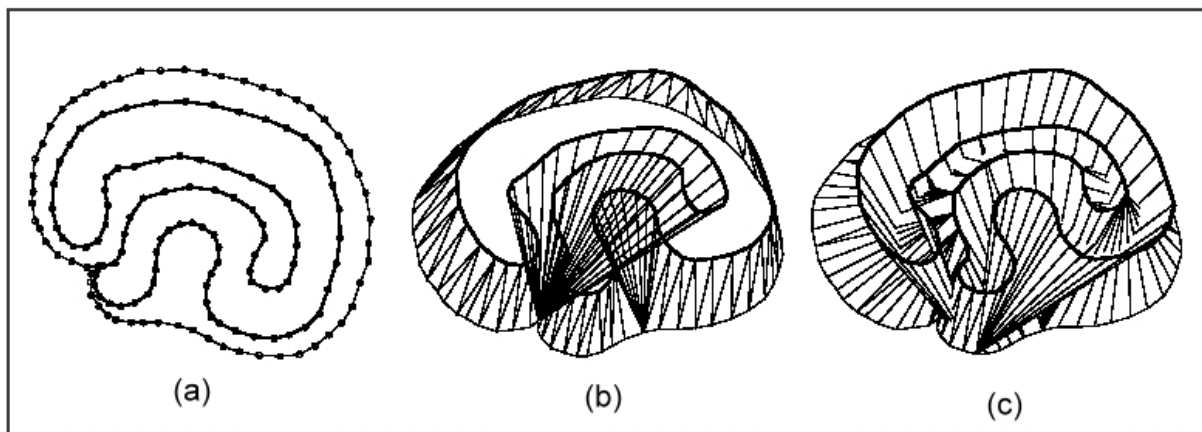


Figura 3.4: Diferentes Malhas - *Tiling*. Fonte: Bajaj et al (1996)

A consideração da topologia dos contornos é essencial para gerar uma boa triangulação. Se entre as duas seções a topologia for muito diferente e não houver uma relação entre os pontos das curvas que serão interpoladas, a superfície gerada pode não ser um dos resultados esperados, como no exemplo (c) da Figura 3.4.

### 3.4 Abordagens da Reconstrução 3D

Tendo em vista que o problema de reconstrução tridimensional a partir de seções planares tem inúmeras aplicações e a área de pesquisa é grande para o desenvolvimento de algoritmos que tratem da reconstrução, várias técnicas já foram desenvolvidas para tratar da interpolação de curvas. As diversas técnicas podem ser classificadas conforme o modo como abordam os problemas de reconstrução citados anteriormente (correspondência, bifurcação e geração de malha). Os quatro tipos de abordagens básicas de reconstrução são descritos a seguir.

A abordagem de **modelos deformáveis** utiliza geometria, física e teoria da aproximação para a reconstrução. A geometria é utilizada para representar a forma do objeto, a física impõe confinamentos em como a forma pode variar no espaço e tempo, e a teoria da aproximação



provê mecanismos e técnicas para aproximar os modelos reconstruídos aos dados originais medidos. Neste método são feitas deformações em um modelo inicial, que no caso são as fatias bidimensionais, para chegar ao objeto final. McInerney e Terzopoulos (1996) apresentaram um trabalho de reconstrução aplicado à medicina que utiliza modelos deformáveis, e provaram ser eficaz em segmentar, combinar, e representar estruturas anatômicas. A Figura 3.5 mostra um processo de reconstrução através de modelos deformáveis onde a partir de uma esfera (a) cortada por três planos são feitas deformações e aproximações até se chegar a um modelo desejado (b).

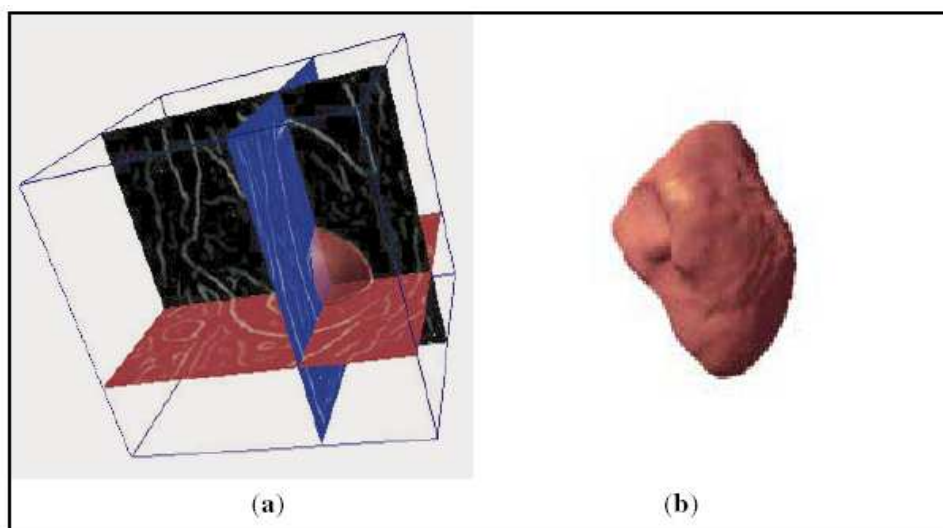


Figura 3.5: Reconstrução a partir de deformações. Fonte: McInerney e Terzopoulos (1996)

McInerney e Terzopoulos (1996) afirmam que os modelos deformáveis superam muitas das limitações de técnicas de baixo nível de processamento de imagens, fornecendo representações compactas e analíticas da forma do objeto. Uma vantagem desta técnica é que o processo de segmentação de imagens, onde a partir da imagem original é efetuada a representação poligonal das curvas, faz parte do processo de reconstrução, porém as etapas de reconstrução não são detalhadas. Pode-se perceber então que a técnica de reconstrução através de modelos deformáveis utiliza mais de conceitos de processamento de imagens, do que de modelagem geométrica.

Nas abordagens **implícitas** é utilizada uma função implícita para interpolar as curvas e gerar o objeto, de modo que a superfície do objeto (borda do objeto) que se quer reconstruir está no conjunto zero desta função, ou seja, em  $f(x, y)=0$ . Esta função é determinada a partir da

interpolação das funções de cada seção paralela (fatia) que contem as curvas a serem interpoladas. Gatass e Peixoto (2000), descrevem as abordagens implícitas através de duas etapas: a definição das funções que representam as curvas das fatias, chamadas de *field functions*, e a interpolação dessas funções para formar a função implícita que representará a superfície do objeto final. A Figura 3.6 ilustra a relação das funções com o objeto resultante.

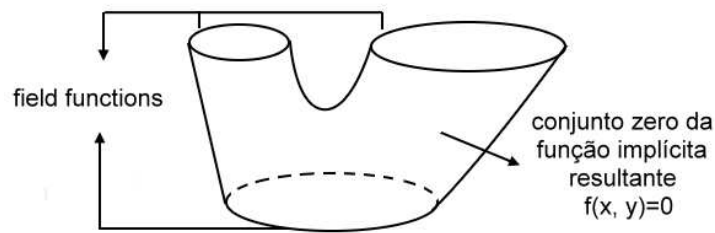


Figura 3.6: Reconstrução a partir de abordagem implícita

Para esta abordagem a representação matricial de curvas é a mais adequada, pois existe uma correspondência natural entre a representação matricial e a função implícita, ou seja, uma curva representada matricialmente pode ser definida como o conjunto de pontos  $(x, y)$  da fatia, tal que  $f(x, y)$  representa cada *field function* utilizada para gerar a função implícita. As etapas de definição das correspondências e tratamento de bifurcação não contam com muita flexibilidade nas abordagens implícitas, pois são automaticamente definidas pela função, o que resulta em uma solução única de interpolação para um determinado conjunto de curvas inicial [Gatass e Peixoto, 2000]. Pode-se concluir que as abordagens implícitas tratam o problema de reconstrução de uma forma automática, porém não gera todos os modelos possíveis a partir de um conjunto de curvas.

As abordagens de **otimização** são baseadas em buscas ótimas em estruturas de dados definidas a partir das curvas iniciais para a determinação de critérios para reconstrução, como o cálculo da distância mínima entre os vértices de uma curva para a conexão dos mesmos. Existem vários trabalhos que utilizam otimização, pois a maioria dos algoritmos sempre procura gerar um modelo ótimo final. O trabalho de Meyers et al (1994) utiliza esse método para a reconstrução a partir de fatias, onde o problema da geração de malha entre duas curvas é expresso a partir de uma busca em uma estrutura de dados onde são mapeados os vértices das

duas curvas gerando um grafo que representa o caminho mínimo de conexão entre os vértices. A Figura 3.7 ilustra este processo: em (a) as duas curvas e seus respectivos vértices e em (b) a estrutura de dados utilizada para o mapeamento dos vértices e o grafo gerado representando o ciclo de menor caminho entre todos os vértices.

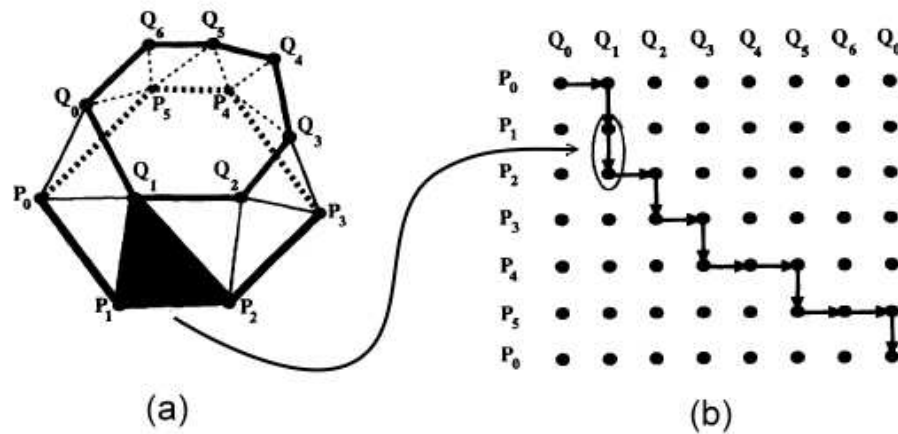


Figura 3.7: Geração de malha auxiliada por um Grafo. Fonte: Meyers et al (1994)

Um fato que deve ser mencionado quanto às abordagens de otimização é que nem sempre a superfície gerada a partir de um caminho ótimo entre os vértices das curvas interpoladas é a superfície final desejada [Meyers et al, 1994]. A Figura 3.8 mostra dois modelos gerados a partir de um mesmo conjunto inicial de curvas, em (a) a superfície do objeto é gerada por um algoritmo de otimização que calcula a superfície mínima entre as curvas, e em (b) o objeto é construído através de um algoritmo com regras definidas para a reconstrução.

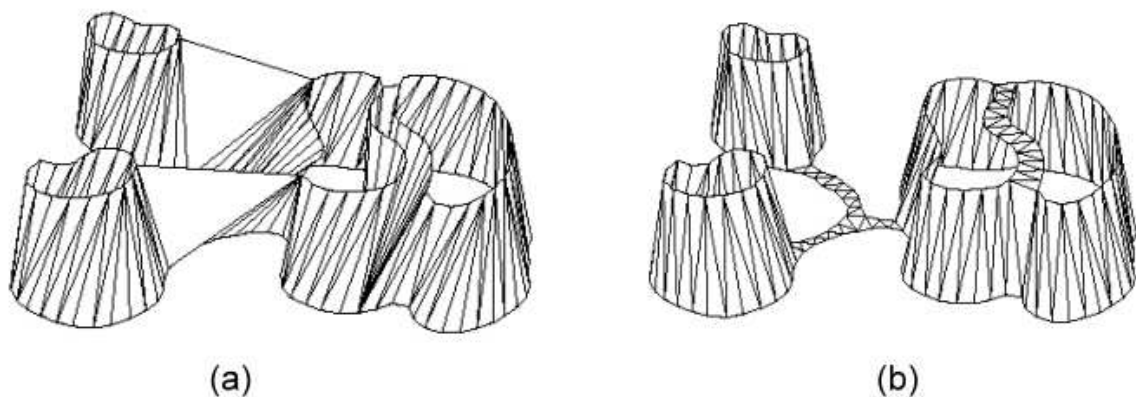


Figura 3.8: Superfícies geradas por algoritmos diferentes. Fonte: Meyers et al (1994)

Pela Figura 3.8 pode-se notar que algoritmos que consideram apenas critérios de otimização podem gerar superfícies indesejáveis. As abordagens de otimização podem tratar com eficiência do critério de geração de malha (*tiling*), mas é preciso alguma heurística para tratar o problema de bifurcação (*branching*), e também de correspondência (*correspondence*) [Vargas, 2001].

Abordagens que utilizam **heurísticas** tratam o critério de correspondência com flexibilidade. Nessas abordagens a reconstrução geralmente é realizada considerando as três etapas (*correspondence*, *branching* e *tiling*) contando com um conjunto de regras definidas (heurísticas), que são especificadas no algoritmo de reconstrução e decidem a geometria e topologia do modelo final. Bajaj et al (1999) demonstram que a maioria das técnicas de reconstrução que não utilizam heurísticas também não tratam das três etapas de reconstrução separadamente. Então é proposto um algoritmo, onde um conjunto de regras é definido e, a partir dessas regras, as três etapas de reconstrução 3D são realizadas para a construção do modelo final. Um exemplo de heurística utilizada no trabalho de Bajaj et al (1999) é na etapa de geração de malha (*tiling*) que conta com um parâmetro  $w$ , o qual influencia no cálculo da superfície resultante e, de acordo com esse parâmetro, o objeto final pode ter uma superfície regular ou irregular. Além do parâmetro  $w$  visto a seguir, a heurística também utiliza os parâmetros  $D$ , que expressa uma distância entre os pontos de duas curvas paralelas, e  $H$ , que expressa a distância entre pontos de uma mesma curva. Então são definidas as seguintes regras para o cálculo deste parâmetro a partir de outros parâmetros já definidos:

- Regra 1: Se o parâmetro inicial  $D$  for menor ou igual a  $0.5H$ , então  $w$  assume o valor de  $2(1 - D/H)$ ;
- Regra 2: Se  $D$  for maior que  $0.5H$  e menor que  $H$ , então  $w$  assume o valor 1;
- Regra 3: Se  $D$  for maior ou igual a  $H$ , então  $w$  assume o valor  $H/D$ .

A partir dessas regras é definido então o valor de  $w$ , e conseqüentemente a superfície do objeto final. Assim que funciona a abordagem heurística para a reconstrução 3D, as regras definidas influenciam diretamente no objeto reconstruído.

Com base no estudo das principais técnicas de reconstrução 3D foi feito um comparativo para decidir qual técnica será utilizada para a solução proposta neste trabalho, a Tabela 3.1 apresenta este comparativo citando as quatro abordagens estudadas.

Tabela 3.1: Comparativo entre as abordagens de reconstrução 3D

Abordagem	Do que utilizam	Como consideram as três etapas de Reconstrução
<b>Modelos Deformáveis</b>	Processamento de Imagens, teoria de aproximação e física.	Não detalha nenhuma das três etapas
<b>Implícitas</b>	Funções implícitas	A correspondência e bifurcação não são tratadas, são geradas automaticamente.
<b>Otimização</b>	Busca ótima em grafos	A geração de malha é tratada com eficiência, mas a correspondência e bifurcação não são detalhadas.
<b>Heurísticas</b>	Definição de regras	Geralmente as três etapas são tratadas com flexibilidade, principalmente a definição da correspondência.

Considerando o comparativo da Tabela 3.1, nota-se que das abordagens estudadas a única que pode tratar das três etapas de reconstrução separadamente é a que utiliza de heurísticas, ou seja, que partem de regras definidas. Este tipo de abordagem também torna o algoritmo de reconstrução mais flexível, ou seja, o modo como é realizada a reconstrução pode ser controlado manipulando apenas as regras que foram definidas. Este trabalho utilizará essa abordagem para a criação do algoritmo que será proposto. A seção seguinte mostrará alguns trabalhos que utilizam a abordagem heurística.

## 4 *Trabalhos Relacionados*

Foi visto na seção anterior que a reconstrução tridimensional através da interpolação de curvas com mudança topológica pode gerar diferentes modelos geométricos. Para o modelo final ser próximo ao desejado, é necessário que durante a reconstrução sejam pré-estabelecidos critérios, ou seja, heurísticas que decidirão o processo. Como foi já mencionado neste trabalho, a principal etapa de reconstrução é a definição da correspondência entre as curvas quando existe mudança de topologia, e é justamente nesta etapa que é necessária a definição de regras para decidir quais curvas irão se conectar e como será feita essa conexão. A seguir serão descritos alguns trabalhos que utilizaram heurísticas, focando a etapa de correspondência.

### 4.1 **Algoritmo de Barequet e Sharir (1996)**

No trabalho descrito por Barequet e Sharir (1996), são tratados os planos com curvas com mudança topológica, isto é, mais de uma curva em um plano ou nos dois. Para decidir sobre a correspondência das curvas é feita uma projeção  $xy$  de dois planos consecutivos. Nesta projeção são analisadas as curvas que se sobrepõem e para elas é definida a conexão. A heurística é a seguinte: se na projeção, houver intersecção da área das curvas, elas são conectadas, caso contrário não. A Figura 4.1 demonstra as fatias com as curvas (a), a projeção  $xy$  dos dois planos, a intersecção das curvas para definir a correspondência (b), e a correspondência resultante (c).

Para a etapa de geração de malha (*tiling*), é realizada uma triangulação 2D a partir da projeção feita e depois feita a triangulação 3D. A etapa de tratamento de bifurcações (*branching*) não é detalhada nesta técnica, e considerando um exemplo em que existe mudança de topologia

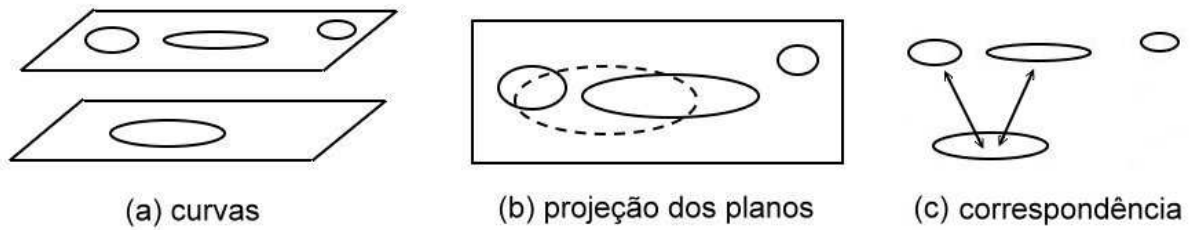


Figura 4.1: Definição da correspondência

entre as curvas, o algoritmo apenas efetua a união das curvas com mudança topológica no mesmo plano, gerando uma só curva que é então interpolada com a curva correspondente do plano seguinte. A Figura 4.2 ilustra como seria o resultado final de uma reconstrução através deste trabalho, onde na fatia superior existisse mudança topológica.

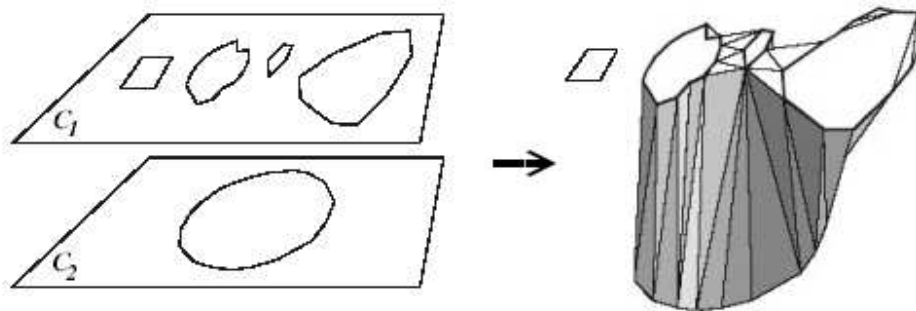


Figura 4.2: Objeto reconstruído sem bifurcação. Fonte: Gatass e Peixoto (2000)

No trabalho de Barequet e Sharir (1996) a heurística só é utilizada pra definir a correspondência das curvas e o tratamento de bifurcações não é considerado, sendo um dos poucos trabalhos que utilizam heurísticas e não consideram uma das etapas de reconstrução 3D.

## 4.2 Algoritmo de Treece, Prager, Gee e Berman (1999)

Outro trabalho que também trata de reconstrução a partir seções planares considerando heurísticas foi desenvolvido por Treece et al (1999) e é baseado no cálculo da distância entre regiões das curvas. Para cada plano contendo as curvas são relacionados discos, que podem ser internos ou externos as curvas, os discos externos são utilizados para representar regiões entre

as curvas do mesmo plano, enquanto os discos internos são utilizados para representar cada região interna de cada curva. A Figura 4.3 apresenta em (a) o conjunto inicial de curvas, e em (b) os discos internos e externos que são atribuídos a cada plano aos quais as curvas pertencem.

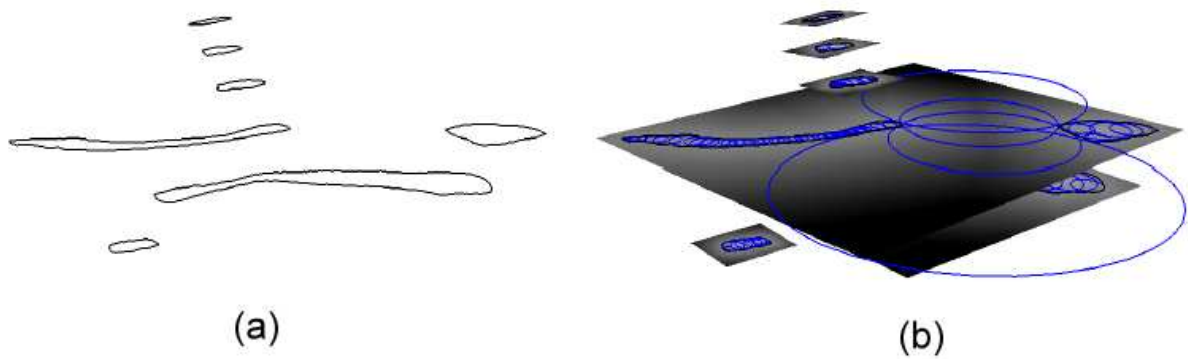


Figura 4.3: Definição dos discos para as curvas. Fonte: Treece et al (1999)

Para cada disco é calculado seu centro, denominado centróide, que será utilizado para calcular a distância entre cada par de discos de dois planos consecutivos. O cálculo da probabilidade de correspondência baseado na distância de cada par de discos é a etapa mais importante neste processo. A heurística definida neste algoritmo é a seguinte: os discos de dois planos consecutivos com menor distâncias entre si serão correspondentes. É feita então, para cada dois planos consecutivos, uma comparação de distâncias entre os centróides de cada par de discos. A correspondência, neste caso, não leva em consideração toda a área da curva e sim cada região representada por um disco.

A Figura 4.4 apresenta esse processo ilustrando uma projeção de dois planos consecutivos A e B e três curvas, uma composta pelos discos a1, a2 e a3 e outra pelo disco a4, ambas localizadas no plano A, e a relação delas com uma outra curva representada pelo disco b1 localizada no plano B. Em (a) é comparada a distância do disco b1 com o disco a4, em (b) a distância do disco b1 com o disco a1, em (c) a distância do disco b1 com o disco a2, e finalmente em (d) a distância do disco b1 com o disco a3, resultando na conexão das regiões destes discos, pelo fato de terem a menor distância.

Feita a correspondência de todos os discos, as curvas que não tiverem nem um par de



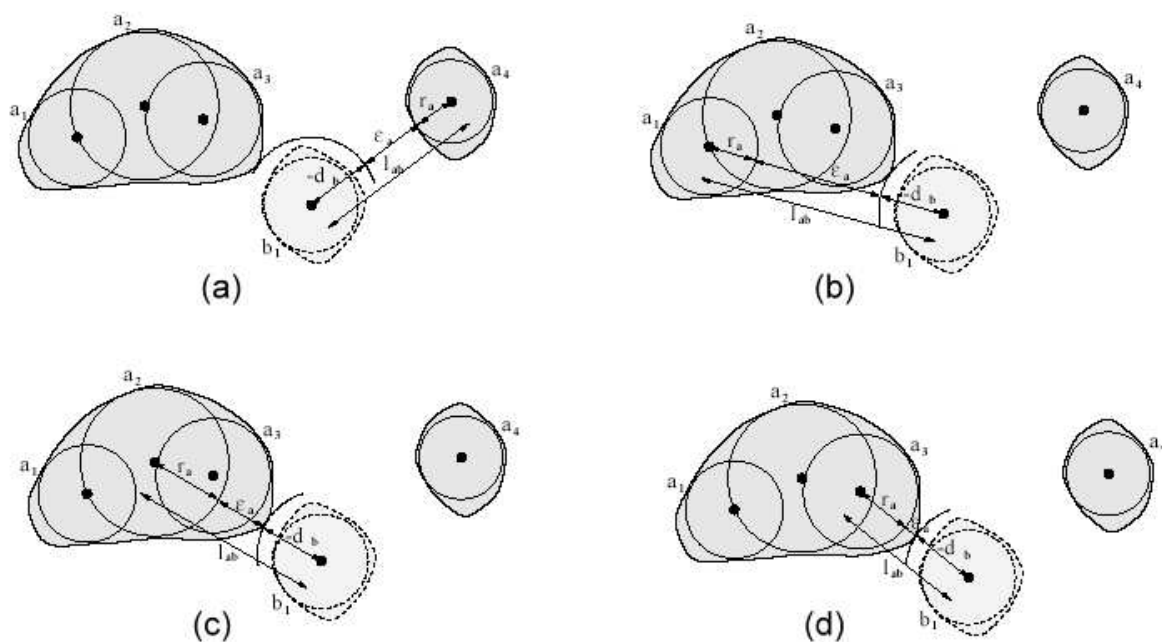


Figura 4.4: Comparação de distâncias entre centróides. Fonte: Treece et al (1999)

discos conectados não serão conectadas. Então, depois de decidida a correspondência de todas as curvas agora, além da heurística utilizada para esta etapa, é utilizada uma função implícita para gerar a superfície do objeto resultante. A Figura 4.5 representa em (a) o resultado da correspondência dos discos da Figura 4.3, e em (b) a superfície do objeto final gerada a partir de uma função implícita, método discutido no capítulo 2 deste trabalho.

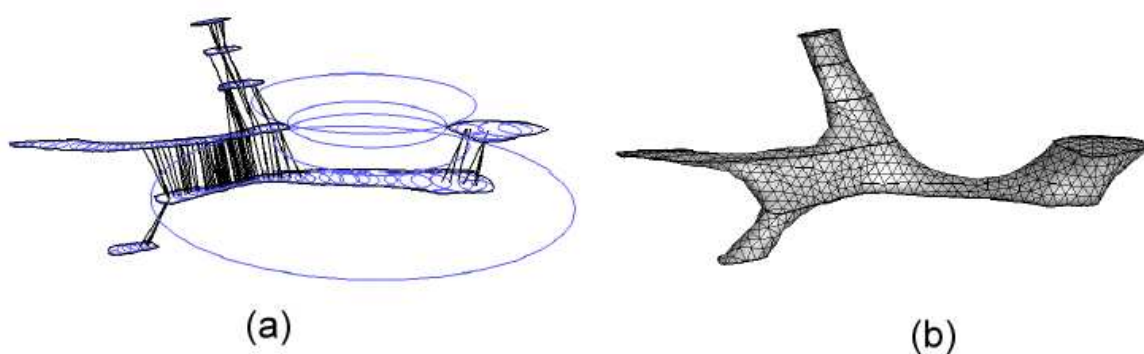


Figura 4.5: Resultado da correspondência e objeto final. Fonte: Treece et al (1999)

Algumas características podem ser analisadas do trabalho de Treece et al (1999). Uma delas é que as regiões representadas pelos discos só vão conectar com as outras regiões mais próximas de cada plano consecutivo se essa relação for recíproca, isso permite deixar regiões

sem conexão se for preciso. Uma característica em relação a proposta de Barequet e Sharir (1996) discutida anteriormente neste capítulo é que não é necessário duas curvas se sobreponem para se conectarem. E apesar de utilizar heurística para a definição da correspondência, o trabalho de Treece et al (1999) também utiliza funções implícitas para a geração de malha do objeto final.

### 4.3 Algoritmo $\beta$ -Connection (2001)

Uma outra técnica que utiliza heurística é proposta por Vargas (2001). Trata-se de uma estratégia de reconstrução volumétrica chamada de  $\beta$ -Connection que tem a flexibilidade para produzir uma família de objetos construídos a partir de um mesmo conjunto de seções planares, tornando possível múltiplas opções de um objeto final. Para resolver o problema de correspondência (*correspondence*) o algoritmo efetua um cálculo de distância e recebe um parâmetro  $\beta$  do usuário pra decidir quais curvas vão se conectar. Este parâmetro é um inteiro positivo que expressa uma distância entre as curvas para a conexão. A heurística definida é a seguinte: Se a distância entre duas curvas quaisquer for menor que o valor do parâmetro  $\beta$  definido pelo usuário, então estas curvas são conectadas. Isto é, para um  $\beta$  definido pelo usuário, todas as curvas que estejam a uma distância menor ou igual a  $\beta$  ficarão conectadas.

A Figura 4.6 apresenta diferentes soluções de conexão resultantes da estratégia proposta por Vargas (2001). A partir das curvas situadas em seções paralelas (a), na primeira solução (b) o parâmetro  $\beta$  definido é menor que todas as distâncias entre as curvas de dois planos consecutivos, não acontecendo nenhuma conexão, na segunda solução (c) o valor do parâmetro  $\beta$  vale 3, então todas as curvas de dois planos consecutivos com distância entre elas menor ou igual a 3 são conectadas, e na ultima ilustração (d) é atribuído um valor grande para  $\beta$  sendo maior que todas as distâncias entre as curvas de dois planos consecutivos, neste caso todas elas se conectam.

O trabalho de Vargas (2001) não constrói somente a superfície do objeto final, é feita uma triangulação entre todos os vértices das curvas iniciais e então são gerados tetraedros para re-

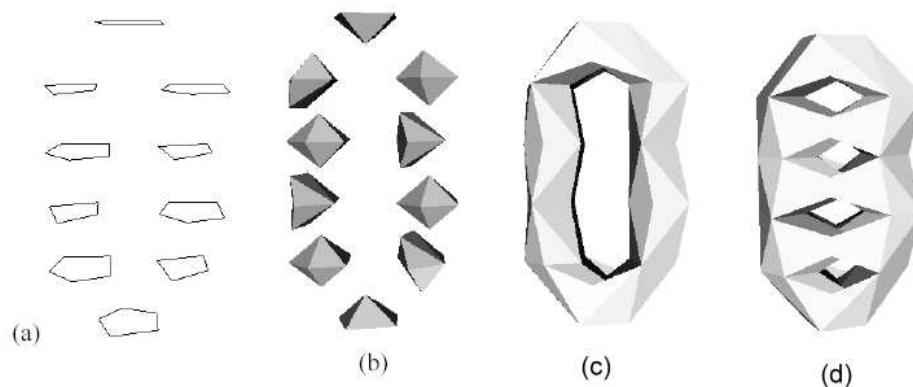


Figura 4.6: Diferentes valores para  $\beta$ . Fonte: Vargas (2001)

presentar todo o volume do objeto final. A etapa de bifurcação (*branching*) é tratada detalhadamente por duas etapas de seu algoritmo: a classificação de todos os tetraedros do objeto e a eliminação dos tetraedros que não pertencerão ao objeto final. A classificação dos tetraedros é feita analisando se as arestas dos mesmos têm alguma relação com as arestas das curvas iniciais, assim são identificados os tetraedros internos e externos ao modelo final. A Figura 4.7 ilustra este processo, onde partindo das curvas (a) são gerados tetraedros a partir de todos os vértices das mesmas, gerando em (b) o volume total formado por estes tetraedros, é feita então a classificação dos mesmos (c) com a finalidade de eliminar os necessários para representar a bifurcação, e gerar finalmente o modelo final do objeto reconstruído (d), com os tetraedros externos eliminados.

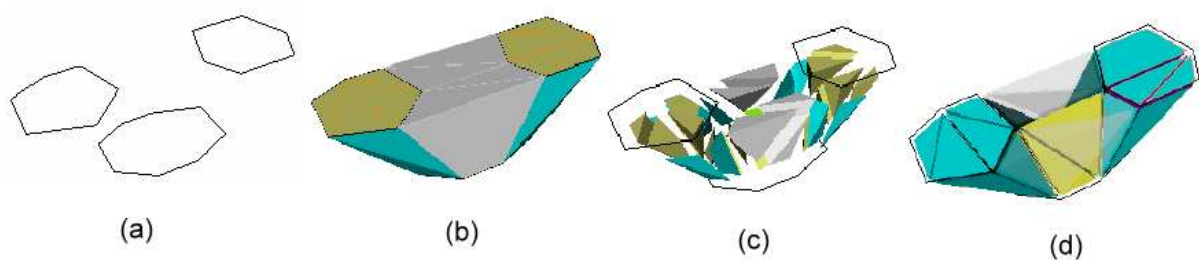


Figura 4.7: Classificação e eliminação de tetraedros. Fonte: Vargas (2001)

Segundo Vargas, (2001) a técnica de reconstrução *Beta-Connection* oferece mais flexibilidade na escolha das componentes conexas, pois partindo de um mesmo conjunto de seções planares pode-se chegar a diferentes formas de objetos, o que é difícil através de outros algoritmos da literatura, resolvendo os problemas de ramificação e geração de malha de forma

satisfatória.

O algoritmo proposto neste trabalho é construído com base nas contribuições descritas neste capítulo, partindo da idéia do  $\beta$ -Connection [Vargas, 2001] onde se tem flexibilidade para a etapa de decisão de correspondência através de um parâmetro. Como  $\beta$ -Connection é uma técnica de reconstrução volumétrica, e neste trabalho está se tratando da reconstrução de superfícies, não serão utilizados tetraedros para o cálculo de distância entre as curvas e sim através da projeção dos planos [Barequet e Sharir, 1996], serão calculadas as distâncias entre os centróides das curvas [Treece et al, 1999] a fim de compará-las com o parâmetro definido pelo usuário para decidir a correspondência. Então, a solução apresentada no capítulo a seguir incorpora características das três abordagens anteriores.

## 5 *Algoritmo $\Delta$ -connection*

Com base nos trabalhos relatados nos capítulos anteriores, é possível perceber que se faz reconstrução 3D para obter diversas informações sobre o modelo original, basicamente por dois motivos:

- Estudar a estrutura do modelo, a relação entre as partes, o entendimento do todo. Existência ou não de certas formações (bolsões, conexões, etc.) e;
- Obter características mensuráveis do modelo (volume, área, comprimento, etc.).

Percebe-se que esta divisão leva a duas possíveis abordagens quanto às aplicações das técnicas de reconstrução 3D, uma voltada à visualização do objeto como um todo, outra voltada para a precisão das partes do objeto. Ainda pode-se perceber variações na reconstrução no que diz respeito ao tipo do modelo a ser reconstruído, que pode ser na forma de canais, onde a ocorrência de detalhes na superfície é menos importante que a visualização da estrutura do modelo, ou de corpos onde o local da bifurcação e os detalhes da superfície tornam-se mais aparente. A Figura 5.1 ilustra duas abordagens diferentes, onde em (a) a reconstrução é voltada para a visualização de canais e pode ser feita de uma forma mais simples, sem levar em consideração todos os detalhes do objeto final, por exemplo, o local que irá ocorrer cada bifurcação, realizando uma reconstrução preocupada principalmente com a visualização do modelo como um todo, já em (b), é ilustrada a estrutura de um corpo considerando cada detalhe das curvas, se a reconstrução não tratar com precisão a bifurcação (*branching*) que vai ser gerada, será visível qualquer má formação na superfície do modelo final.

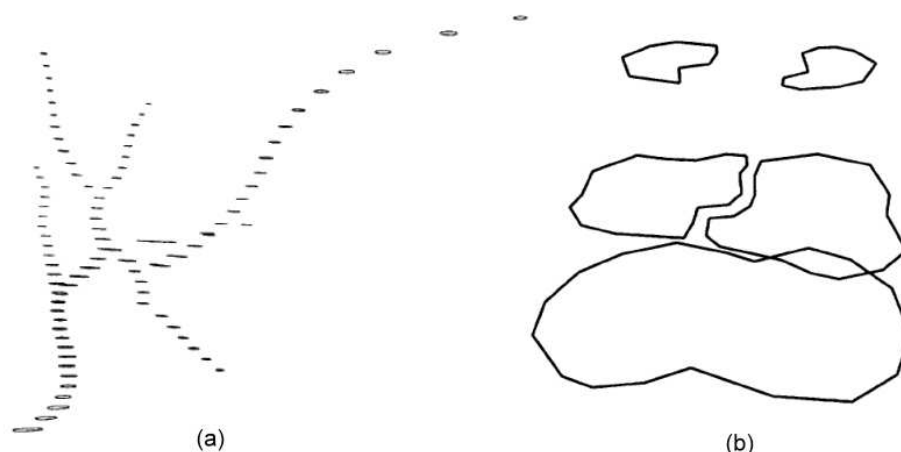


Figura 5.1: Diferentes abordagens para reconstrução. Fonte: Meyers et al (1992)

A reconstrução voltada para visualização da estrutura sem necessariamente ter-se precisão já encontra grande utilidade para várias aplicações como a identificação de adensamento de capilaridades (várias ramificações agrupadas) ou uma má formação congênita (aparecimento de uma geometria estranha). Estas aplicações se valem da vantagem que estas técnicas trazem de poder tornar visível externamente uma formação cuja visualização, de outra forma, seria invasiva. Para este tipo de abordagem a flexibilidade e as heurísticas de identificação de correspondências são de grande importância aliadas ao fato de que se busca uma resposta rápida à esta abordagem

A proposta deste trabalho é voltada para esta abordagem, ou seja, para a visualização de canais, ilustrada em (a) na Figura 5.1. A etapa de definição de correspondência (*correspondence*) terá ênfase, e não será tratado com precisão cada local de bifurcação (*branching*), sendo resultado direto e impreciso, obtido da etapa de geração de malha (*tiling*).

Na abordagem adotada aqui, a interpolação quando existe mudança de topologia resulta numa sobreposição de superfícies que proporciona um efeito visual da bifurcação (*branching*) sem cálculos adicionais e sem processamento diferenciado para a geração de malha (*tiling*). Assim, economizam-se tempo e processamento para estes dois detalhes. Vale ressaltar que efetuar o cálculo da intersecção das correspondências, em média não altera a contagem de faces do módulo, apenas sua precisão. Sendo assim não influencia no tempo de *rendering* do modelo

com esta abordagem. A seguir será descrito como cada uma das etapas de reconstrução será tratada neste trabalho.

## 5.1 Correspondência no $\Delta$ -connection

A etapa de definição de correspondências (*correspondence*) será realizada em três passos. Primeiramente será calculado um centróide para cada curva. O centróide será calculado através da média aritmética dos pontos  $x$  e  $y$  extremos da curva.

Depois de calculados os centróides, são calculadas as distâncias entre eles. Uma pequena otimização será feita no cálculo das distâncias no sentido de que estas serão calculadas no plano  $XY$ , ou seja, desconsiderando a distância  $Z$  entre os planos (que será sempre a mesma), e ainda será evitada a extração da raiz quadrada, que é um cálculo custoso. Então o cálculo representado pela equação 5.1 será resumido pelo apresentado na equação 5.2.

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2} \quad (5.1)$$

$$d^2 = (x_1 - x_2)^2 + (y_1 - y_2)^2 \quad (5.2)$$

Todas as distâncias são armazenadas em uma matriz ( $d[i][j]$ ), a Figura 5.2 ilustra, a partir de uma projeção dos dois planos, as distâncias entre os centróides das curvas e a matriz onde elas são armazenadas, tendo como linhas os centróides das curvas de um plano ( $\text{CentroCurvaP1}[i]$ ) e como colunas os centróides das curvas do plano seguinte ( $\text{CentroCurvaP2}[j]$ ). As curvas desenhadas em pontilhado pertencem ao plano projetado para o cálculo as distâncias.

Definida a matriz de distâncias, são calculadas também as distâncias mínimas e máximas da matriz com o objetivo de informar ao usuário qual o intervalo de distâncias entre todas as curvas. O algoritmo recebe um parâmetro  $\Delta$ , definido pelo usuário, que é comparado com cada distância calculada, ou seja, cada elemento da matriz de distâncias, afim de decidir se duas determinadas curvas serão conectadas ou não. A heurística definida é a seguinte:

- Se  $\Delta$  for menor que a distância mínima da matriz, nenhuma curva é interpolada;

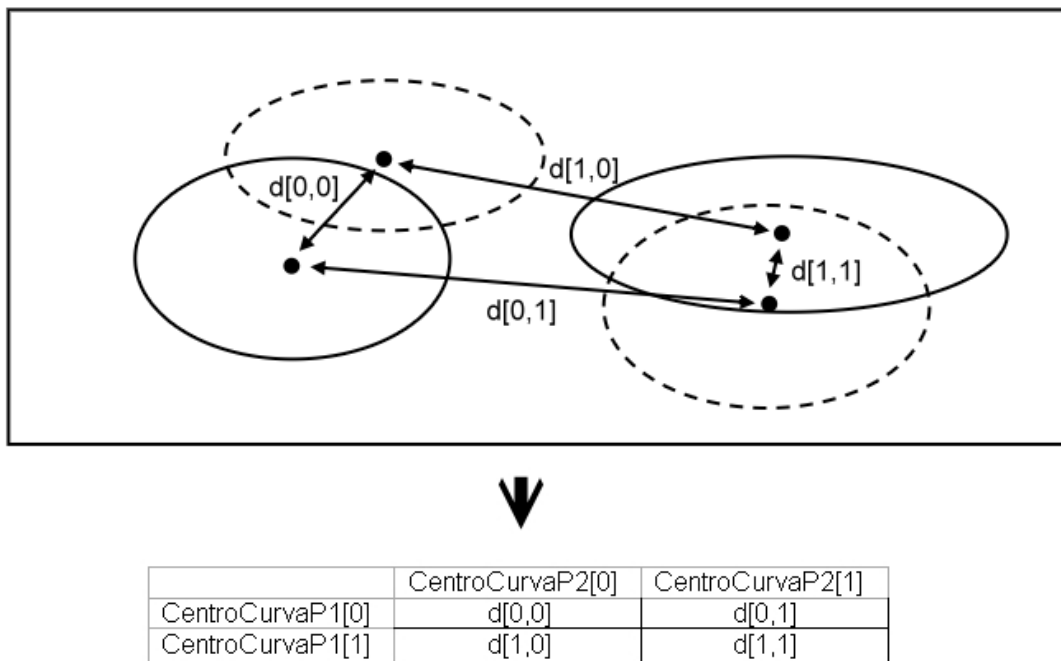


Figura 5.2: Matriz de distâncias

- Se  $\Delta$  for maior ou igual que a distância máxima da matriz, todas as curvas são interpoladas;
- Se  $\Delta$  estiver no intervalo entre a distância mínima e máxima, é feita a seguinte regra para cada uma das distâncias:
  - Se a distância entre duas curvas for menor ou igual a  $\Delta$ , então essas curvas são conectadas;
  - Se a distância for maior que  $\Delta$ , então não ocorrerá conexão dessas duas curvas.

Para cada duas curvas que forem conectadas a partir da heurística definida, é realizada a próxima etapa do algoritmo que trata da geração de malha (tiling). As curvas que não tiverem nenhuma conexão não serão interpoladas, como apresentada pela seta na Figura 5.3 (b).

Como a heurística apresentada aqui depende da distância “d” (em grego delta) o algoritmo é denominado de  $\Delta$ -connection, a exemplo do  $\beta$ -connection proposto por Vargas (2001). A seguir será descrita a etapa de geração de malha para as curvas correspondentes.



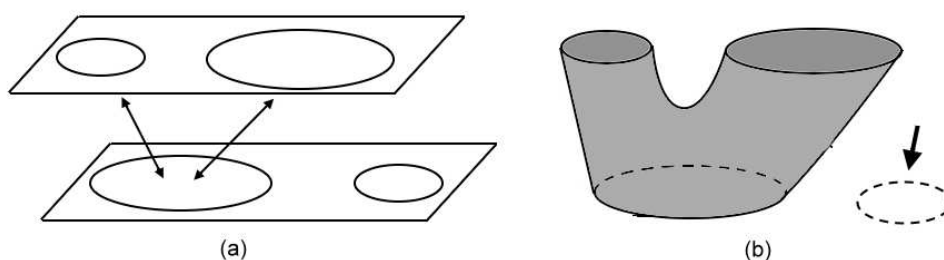


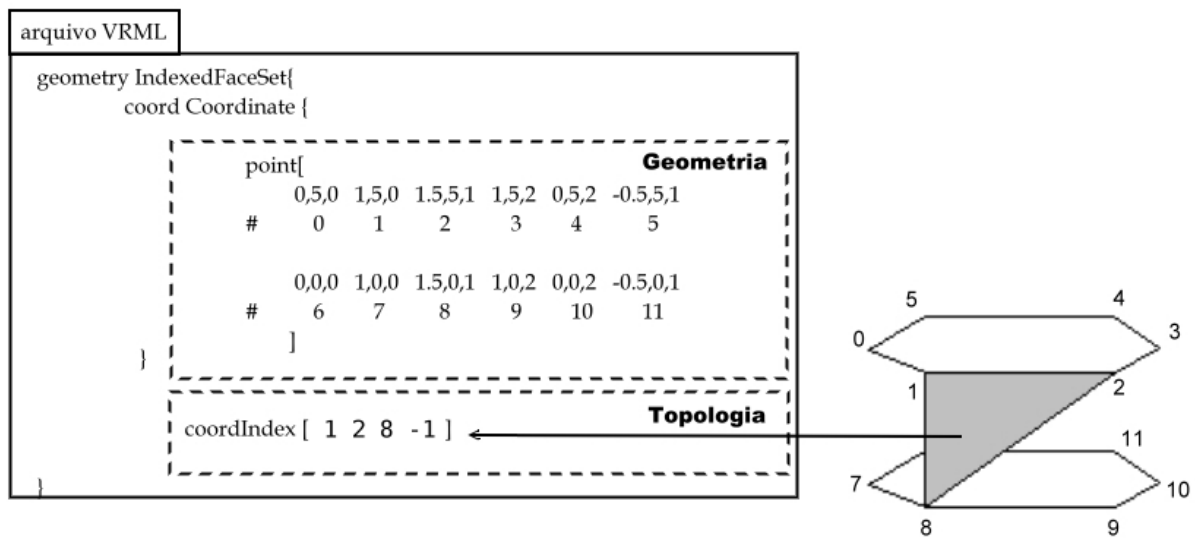
Figura 5.3: Curva sem nenhuma correspondência

## 5.2 Geração de Malha no $\Delta$ -connection

Uma vez identificadas as correspondências, a geração de malha (*tiling*) será feita para cada par de curvas correspondentes. Nesta etapa será usado o modelo *B-rep* poliédrico, explicado na seção 2.2 deste trabalho, para representar o objeto a partir da sua geometria e topologia.

O VRML conta com um recurso para modelar objetos que trata-se do nó *IndexedFaceSet* o qual permite descrever as coordenadas e a topologia dos vértices, definindo cada face do objeto final através dos nós *Coordinate* e *coordIndex*. No nó *Coordinate* estão listadas as coordenadas de todos os vértices no espaço 3D a partir do nó *point* que guarda uma lista de pontos, aqui é definida a geometria das faces que serão definidas. O nó *coordIndex* descreve uma lista de faces a partir dos índices do nó *Coordinate*, ou seja, a partir da geometria é agora definida a topologia das faces, sempre em um mesmo sentido (horário). A Figura 5.4 apresenta como é feita a descrição das faces de um objeto em VRML a partir do nó *IndexedFaceSet*, em pontilhado foram destacadas duas regiões do código: a região do nó *point* onde é definida a geometria, ou seja, as coordenadas de cada ponto das duas curvas que serão interpoladas, e a região do nó *coordIndex*, onde é definida a topologia, ou seja, a ligação dos pontos definidos para formar cada face, ilustrada no objeto ao lado, da malha a ser gerada.

Então para cada par de curvas correspondentes, a geração de malha será feita da seguinte forma: primeiro serão gerados os vértices das duas curvas (geometria) através do nó *Coordinate* e depois será feita a conexão destes vértices (topologia) através do nó *coordIndex*, gerando todas as faces do objeto final. A geração de malha não pode ser feita por uma simples triangulação entre os vértices das duas curvas devido a possível diferença do número de vértices entre as

Figura 5.4: Representação *B-rep* em VRML

mesmas e o fato de os vértices iniciais de cada curva nem sempre estarem próximos.

A Figura 5.5 representa em (a) o problema que pode ocorrer quanto existe a diferença de número de vértices entre as curvas de forma bem simples, onde é realizada a geração de malha entre uma curva com quatro vértices, representada pelos pontos P (P1, P2, P3 e P4) e uma curva com seis, representados pelos pontos Q (Q1, Q2, Q3, Q4, Q5 e Q6). É realizada uma simples triangulação, ou seja, cada vértice de uma curva é conectado com o próximo vértice da curva seguinte, e como o número de vértices não é igual, a geração de malha não conecta todos eles (Q6 fica desconectado). Em (b) é ilustrado o problema dos vértices iniciais de cada curva não estarem próximos no espaço. Nota-se que mesmo entre duas curvas com o mesmo número de vértices, a malha que estará sendo gerada ficará retorcida pelo fato dos vértices correspondentes não estarem próximos, isto ocorre devido os vértices iniciais de cada curva (P1 e Q1), por onde começa triangulação, não pertencerem a uma mesma região em comum das duas curvas.

Para resolver tais problemas, o algoritmo deverá contar com algumas etapas realizadas antes de efetuar a triangulação para gerar a malha, que serão detalhadas no próximo capítulo.

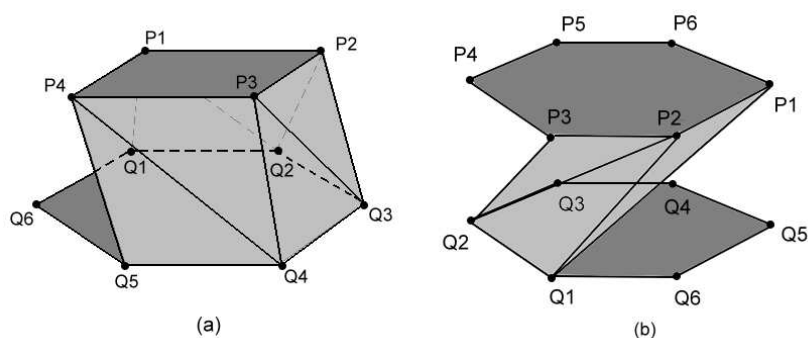


Figura 5.5: Problemas ocorridos com a geração de malha

### 5.3 Bifurcação no $\Delta$ -connection

A etapa de tratamento de bifurcações (*branching*) será resultado direto da geração de malha. Não será decidido o local em que ocorrerá a bifurcação, sendo ele resultado da malha gerada entre as curvas, sem nenhum cálculo de intersecção de superfícies. A Figura 5.6 demonstra como o algoritmo tratará das bifurcações quando existir mudança de topologias nas curvas, ou seja, onde uma curva se divide em duas ou vice-versa.

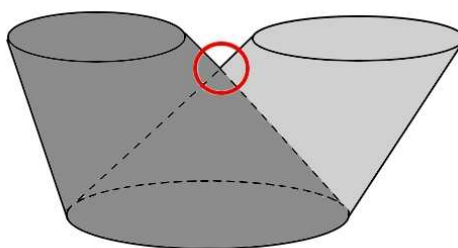


Figura 5.6: Local da bifurcação - *Branching*

É possível notar na Figura 5.6 que a bifurcação é resultado direto das duas superfícies geradas, sendo uma superfície o resultado da conexão da primeira curva do plano superior com a curva do plano inferior e outra o resultado da segunda curva do plano superior conectada com a curva do plano inferior. Esta solução não trata do local onde ocorre a bifurcação, mas é uma solução automática e não exige um custo computacional alto, o que se torna interessante para o método de reconstrução adotado, que é voltado para a visualização externa do sólido como um todo e não para uma reconstrução precisa de cada parte da superfície. Compare o conceito mostrado com o resultado obtido na Figura 7.6.

## 6 *Projeto, Implementação e Melhoramentos*

Este capítulo descreve principalmente a etapa de implementação da solução proposta no capítulo anterior, iniciando pela listagem das características e requisitos do algoritmo proposto, descritos a seguir:

- Centróides calculados pela média dos pontos extremos;
- Arquivo de representação das curvas em XML;
- Algoritmo com parâmetro de controle de flexibilidade;
- Algoritmo simétrico, ou seja, a reconstrução de cima para baixo (*Top-down*) é igual a reconstrução de baixo para cima (*Bottom-up*);
- Objeto representado em *B-rep*, usando VRML com faces orientadas no sentido horário;
- A decisão do local de bifurcação (*branching*) não é tratada;
- Aplicação preferencial: visualização de estruturas (canais).

A seção a seguir vai apresentar o projeto da arquitetura definida para o processo de reconstrução que envolve o algoritmo de interpolação.

### 6.1 **Arquitetura do $\Delta$ -connection**

O processo de reconstrução do  $\Delta$ -connection é dividido em módulos, desde a criação das curvas, que pode ser feita na aplicação desenvolvida, até a visualização do objeto final. A Figura

6.1 representa arquitetura da proposta e os módulos envolvidos.

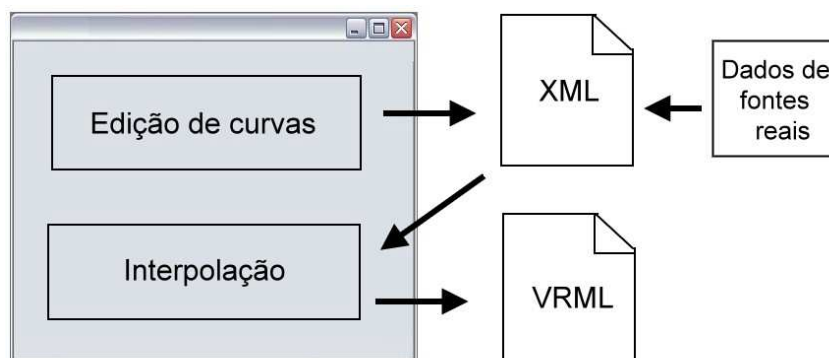


Figura 6.1: Arquitetura do  $\Delta$ -connection

A aplicação implementada permite a edição das curvas a serem interpoladas. Toda informação relativa ao conjunto de curvas que foram editadas é salvo em um arquivo XML que serve de entrada para o algoritmo de reconstrução, o qual realiza a interpolação e escreve um arquivo VRML com as informações geométricas do objeto gerado que pode ser visualizado logo após este processo no *browser* que é chamado pelo programa. O arquivo XML utilizado pelo algoritmo de interpolação não precisa necessariamente ser editado pela aplicação, podendo receber dados de fontes reais editados em qualquer outro lugar e adaptados ao modelo XML descrito da seção 6.3. A seguir será mostrada a interface da aplicação desenvolvida para a edição das curvas e execução do algoritmo.

## 6.2 Interface

A aplicação foi desenvolvida em JAVA e permite a edição de curvas e a execução do algoritmo  $\Delta$ -connection. Para a edição das curvas foram utilizados alguns recursos nativos da linguagem como as classes *java.awt.Canvas* e *java.awt.Point*.

Uma primeira versão da interface foi desenvolvida para os testes iniciais durante a implementação. Esta versão permitia a edição das curvas de maneira simples, sem apresentar ao usuário todas as informações relativas às curvas editadas e a área de edição ainda não utilizava o sistema de coordenadas cartesiano. Após uma reestruturação foi definida a interface definitiva

para a aplicação, ilustrada na Figura 6.2 onde é possível visualizar duas curvas no plano que está sendo editado e uma terceira curva, desenhada em cinza, que pertence ao plano anterior.

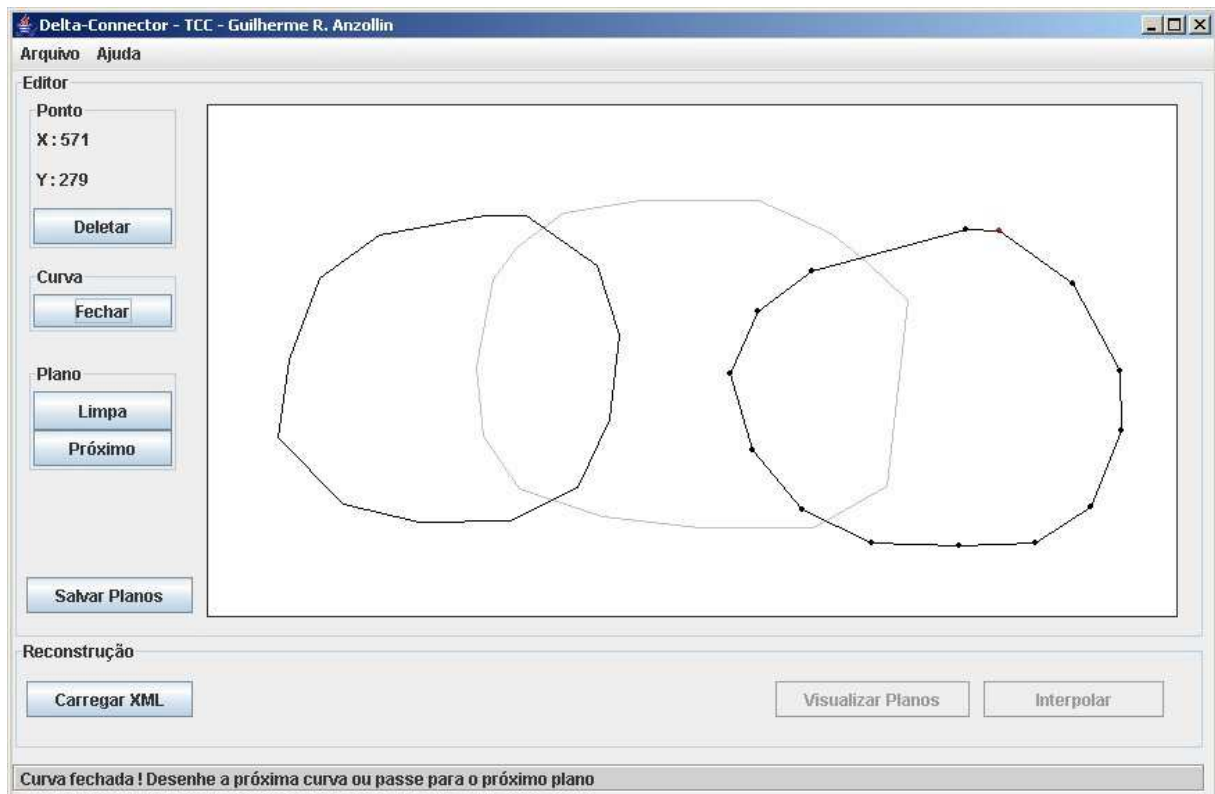


Figura 6.2: Interface

Quanto aos componentes da interface apresentada, a função de cada um é a seguinte:

- Painel “Editor”
  - Painel “Ponto”: exibe as coordenadas do ponto atual e permite deletá-lo;
  - Painel “Curva”: permite fechar a curva atual;
  - Painel “Plano”: permite limpar o plano atual e habilitar o próximo plano a ser editado se houver pelo menos uma curva desenhada;
  - Botão “Salvar Planos”: solicita ao usuário a distância entre os planos e salva o que foi editado em XML;
- Painel “Reconstrução”

- Botão “Carregar XML”: permite procurar um arquivo XML no sistema para ser carregado para a interpolação;
- Botão “Visualizar Planos”: permite visualizar em um *browser* que será chamado os planos contidos no arquivo XML, se este for carregado;
- Botão “Interpolar”: solicita o parâmetro  $\Delta$  ao usuário e executa o algoritmo de interpolação chamando um *browser* para a visualização do objeto gerado.

A interface conta ainda com a opção de iniciar uma nova edição de planos no menu “Arquivo”, informações básicas da aplicação no menu “Ajuda” e um *statusbar* na região inferior para retornar informações ao usuário à medida que o mesmo for interagindo com a aplicação.

## 6.3 Armazenamento de dados

Para o armazenamento dos dados relativos às curvas foi definida inicialmente uma estrutura simples de documento XML como uma primeira abordagem para validar os dados editados e o uso do algoritmo de reconstrução, contendo as informações como o número de planos, a distância entre planos e as coordenadas das curvas. Porém ao decorrer do trabalho foram pesquisadas estruturas de dados utilizadas para armazenamento de curvas e o modelo inicial sofreu alterações.

Foram pesquisados alguns padrões, como o ISO 10303-28 (2002) que especifica o uso da XML para representar esquemas relativos a dados geométricos como sólidos e curvas, mas pelo fato da estrutura gerada a partir deste padrão conter uma grande quantidade de detalhes, os quais não seriam utilizados pelo algoritmo de reconstrução e tornaria a estrutura muito grande ao tratar de dados reais relativos a reconstrução 3D, esta foi descartada.

Durante a fase de implementação deste trabalho foram analisados alguns arquivos contendo curvas de dados reais. Um esboço do conteúdo de um documento contendo estes dados está disponível no apêndice A [Barequet, 2006]. Ainda foram analisados dados já vetorizados e disponíveis no formato VTK [VTK, 2006], porém este dividia a representação de um objeto em

vários arquivos cada um contendo um respectivo plano, o que contrariava a idéia inicial de se armazenar em um único arquivo todo o conjunto de planos referentes a um objeto.

O arquivo XML proposto aqui, utilizado para o armazenamento das curvas editadas e como entrada para o algoritmo de reconstrução, foi organizado de maneira que ficasse clara a representação das mesmas e o mais próximo possível dos dados reais apresentados no apêndice A para manter um certo padrão e poder utilizá-los como testes. A Figura 6.3 representa a estrutura do arquivo XML que será utilizada. O apêndice B dispõe um documento XML relativo aos dados reais apresentados no apêndice A.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
- <Slices>
  <Info>Espaço reservado para informações sobre os slices</Info>
  - <x_dimension>
    <min>0</min>
    <max>700</max>
  </x_dimension>
  - <y_dimension>
    <min>0</min>
    <max>370</max>
  </y_dimension>
  <n_points>27</n_points>
  <n_curves>3</n_curves>
  <n_slices>2</n_slices>
  <dist_slices>100</dist_slices>
  - <slice>
    <n_curves>1</n_curves>
    <curve>300 77 274 96 246 140 244 181 284 224 324 238 383 231 419 207
      424 165 411 124</curve>
  </slice>
  - <slice>
    <n_curves>2</n_curves>
    <curve>289 107 252 124 257 169 297 185 327 157 334 124 326 105</curve>
    <curve>211 134 207 174 256 237 328 238 367 176 364 141 300 78 248 78
      229 99 217 124</curve>
  </slice>
</Slices>
```

Figura 6.3: Armazenamento em XML

A estrutura do arquivo é formada por um *tag* raiz denominado “Slices”. O primeiro elemento, denominado “Info”, serve para armazenar qualquer informação adicional relativa ao conjunto de planos. Os próximos dois *tags* denominados de “x\_dimension” e “y\_dimension” armazenam as coordenadas mínimas e máximas dos planos onde foram editadas as curvas. Os *tags* “n\_point”, “n\_curves” e “n\_slices” informam respectivamente o número de pontos, curvas e planos do arquivo, e em “dist\_planos” é informada a distância entre os planos armazenados.



A cada plano que estiver armazenado no arquivo é criado um elemento contendo suas informações, o tag “n\_curves” informa o número de curvas que o mesmo contém, e para cada curva são descritas as coordenadas x e y de cada um de seus vértices, separadas por espaços em branco. Percebe-se que o tag “n\_curves” é usado em dois momentos com semântica diferente, sendo na primeira ocorrência pra informar o número total de curvas do arquivo, e depois a cada plano para informar o número de curvas do mesmo

## 6.4 Interpolação: $\Delta$ -connection

O algoritmo de interpolação tem como entrada um arquivo XML e o parâmetro  $\Delta$  definido pelo usuário para decidir a correspondência das curvas. A Figura 6.4 apresenta como o algoritmo foi implementado.

```
deltaConnection()
{
    //calcula o centróide de todas as curvas
    centroides[ ][ ] = calc.centroides(planos);
    //calcula a matriz de distâncias a partir dos centróides
    distancias[ ][ ] = calc.matrizDeDistancias(centroides);
    //entrada do parametro delta para interpolação
    entradaDelta()

    //para cada 2 planos consecutivos
    for( i=0; i<(nPlanos-1); i++ )

        //para todos os índices da matriz de distâncias
        for( a=0; a<distancias.length; a++ )
            for( b=0; b<distancias[a].length; b++ )
                //condição para a interpolação
                if(distancias[a][b] <= delta)
                    //interpola curva a e b
                    tiling(planos[i][a], planos[i+1][b]);
}
```

Figura 6.4: Algoritmo  $\Delta$ -connection

A primeira etapa do algoritmo é calcular os centróides dos planos e armazenar na matriz de dados “centroides[ ][ ]” para utilizar como entrada no cálculo da distância entre cada par de centróides pertencentes a curvas de planos consecutivos. Os algoritmos de cálculo dos centróides

e cálculo da matriz de distâncias estão disponíveis no apêndice C. Depois de calculada a matriz de distâncias, a menor e maior distância é informada ao usuário e então é solicitada a entrada do parâmetro  $\Delta$ .

Para cada par de planos consecutivos é analisada cada distância entre as curvas a serem interpoladas e comparada com o valor de  $\Delta$ . Através da condição já mostrada na seção 5.1 é verificado se duas determinadas curvas serão interpoladas através do método *tiling* ou não.

Outros métodos foram implementados para auxiliar no processo de reconstrução, como a leitura do arquivo XML e armazenamento das informações em uma estrutura de dados, a escrita em VRML, entre outros. A Figura 6.5 apresenta o diagrama de classes da implementação, preparado com auxílio da ferramenta Jude, listando todos os atributos e métodos que foram utilizados, desde a edição de curvas até a interpolação. As classes MyCanvas, Curvas e Ponto são responsáveis pela edição de curvas e escrita em XML, enquanto as classes XMLtoVRML e Calculos contém os métodos responsáveis pela interpolação.

Na primeira versão do algoritmo, o método *tiling* realizava a geração de malha conectando cada ponto de uma curva a ser interpolada com todos os demais pontos da outra. Esta solução gerava uma visualização externa aceitável, porém ocorriam dois fatos que impediam de ser essa a melhor solução para a geração de malha. O primeiro diz respeito a gerar um número de faces que ficariam localizadas no interior do objeto reconstruído, ou seja, o objeto final teria um grande número de faces ocultas. O segundo fato pelo qual tornou necessária a realizações de otimizações na geração de malha é por essa solução não tratar de curvas convexas, gerando faces que não poderiam estar presentes no objeto final. A seção a seguir vai apresentar as otimizações realizadas após implementada a primeira versão do algoritmo.

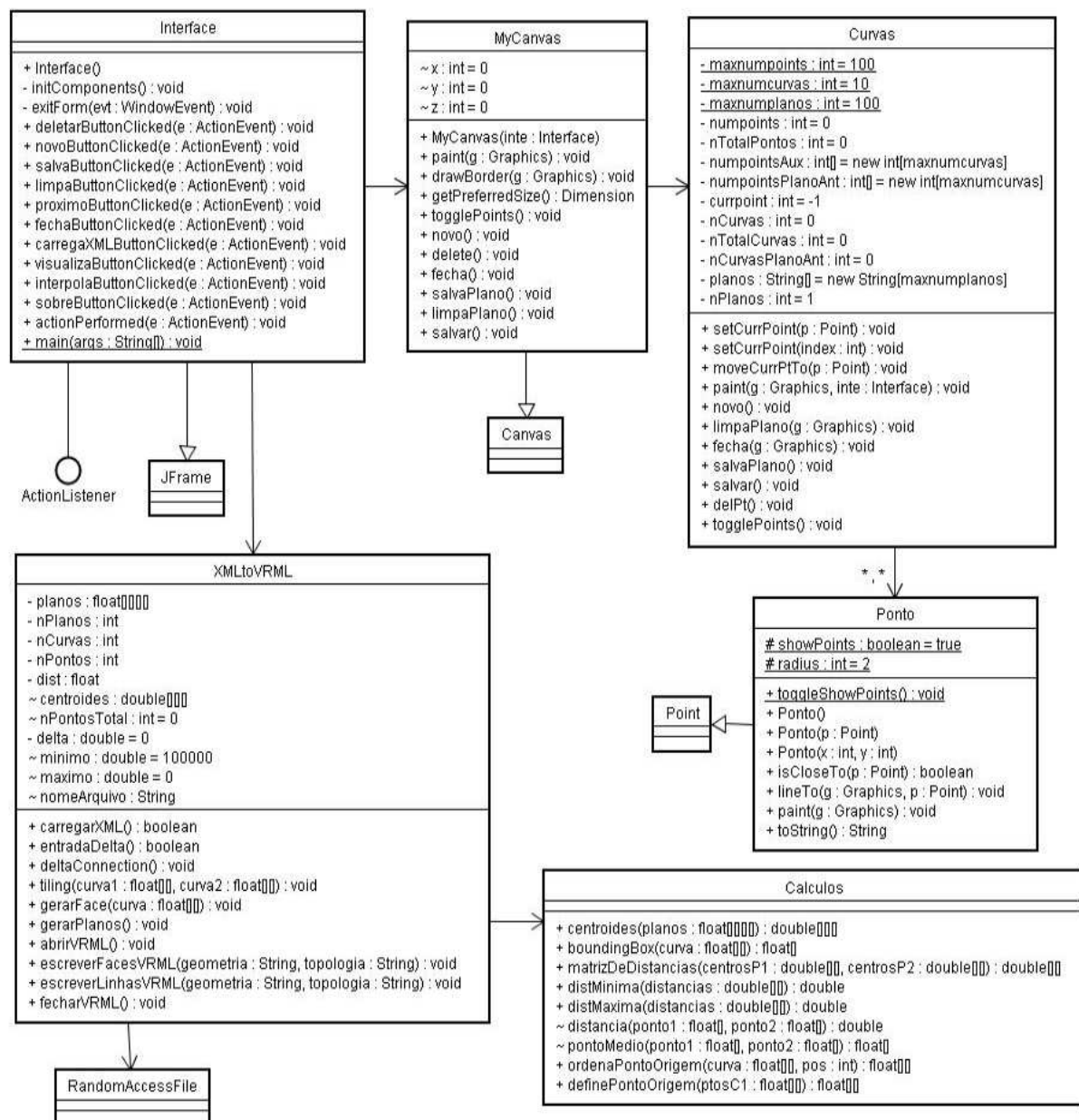


Figura 6.5: Diagrama de classes da aplicação desenvolvida

## 6.5 Melhoramentos

Foram pesquisadas e analisadas diferentes maneiras de realizar a geração de malha de forma que a superfície final fosse gerada com uma aparência aceitável e considerando também o tempo que o algoritmo levaria para realizar a reconstrução. Uma maneira que foi discutida e considerada ótima para os casos onde não existem concavidades funciona da maneira como apresentado na Figura 6.6: dadas duas curvas a serem interpoladas (a), primeiro efetua-se uma translação

para que as duas fiquem concêntricas (b), depois conecta-se cada vértice de uma curva com o vértice mais próximo da curva seguinte (c), e por último translada-se novamente as curvas para seus respectivos pontos iniciais.

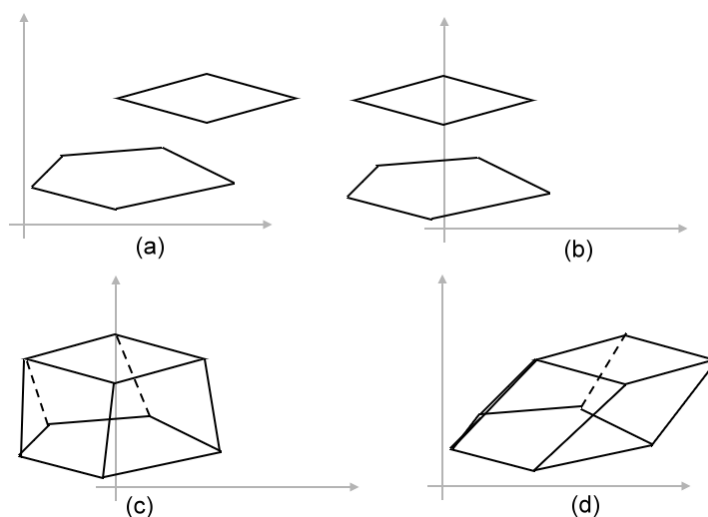


Figura 6.6: Possível solução de *tiling*

Para curvas com concavidades este método pode não funcionar pelo fato de um vértice poder estar mais próximo de outro vértice, o qual não é interessante a conexão, como ilustrado na Figura 6.7, desta feita torcendo a malha resultante.

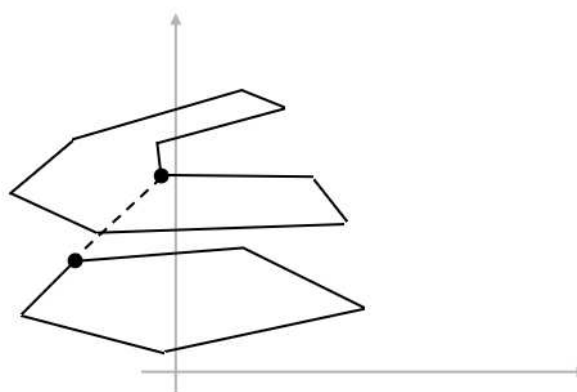


Figura 6.7: Possível problema durante o *tiling*

Foi então proposto um método de geração de malha onde eram adicionados pontos para cada par de curvas para que as mesmas contassem com um mesmo número de pontos e então fosse possível uma triangulação ao redor das curvas a fim de se chegar à malha final. Outro problema a ser resolvido neste caso foi o das curvas possuírem pontos iniciais em regiões di-

ferentes, o que resultaria em uma malha retorcida, como apresentada na Figura 5.5 do capítulo anterior. A seguir será detalhado como foram realizados os passos para se chegar em uma geração de malha de forma eficiente.

### 6.5.1 Definição do Ponto Inicial

Esta etapa visa reordenar os pontos das curvas a serem interpoladas de modo que os pontos iniciais de cada uma fiquem próximos no espaço. A técnica funciona da seguinte maneira: dadas duas curvas a serem interpoladas a partir de seus pontos iniciais (P1 e Q1) como mostrado em (a) na Figura 6.8, o primeiro passo é definir um retângulo que envolva toda a área de cada uma das curvas e localizar um dos vértices desse retângulo, como ilustrado em (b). A partir desse vértice, é localizado qual o ponto de cada curva é o mais próximo dele (c), então as curvas são reordenadas tendo este ponto como inicial (d), que localiza-se em uma região em comum nas duas curvas.

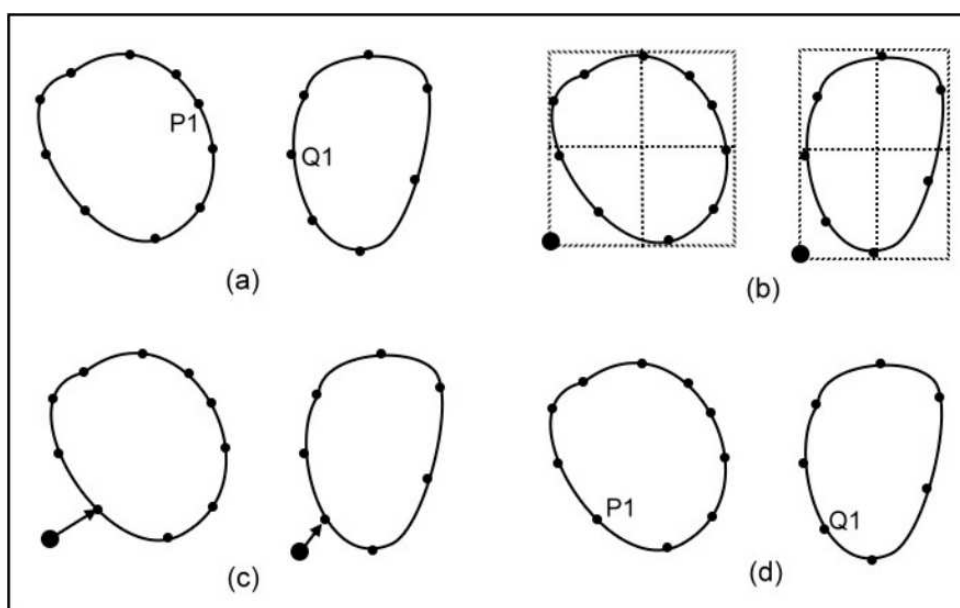


Figura 6.8: Definição do ponto inicial

Após a definição do ponto inicial é realizada a etapa de balanceamento de pontos para as curvas, como descrito a seguir.

### 6.5.2 Balanceamento de Pontos

Esta etapa é executada para garantir que as duas curvas possuam o mesmo número de pontos antes de serem interpoladas. Isto facilita a triangulação e permite que a malha gerada seja mais suave evitando inclusive intersecções entre as faces.

Para cada par de curvas a serem interpoladas, é analisado o número de pontos que cada uma possui. Cada curva passará a ter seus pontos originais mais um número de pontos da curva correspondente, ou seja, a soma dos dois conjuntos de pontos, tendo em vista que apesar de aumentar a estrutura de dados do objeto que será gerado, quanto mais pontos forem distribuídos na curva, maior a suavidade e perfeição da malha.

A atribuição de pontos na curva não é feita de forma arbitrária, a cada dois pontos consecutivos é analisada a distância entre eles e cada novo ponto é adicionado entre os pontos que tiverem a maior distância entre si, proporcionando assim um balanceamento de distâncias entre todos os pontos de cada curva e não afetando a topologia da mesma. A figura 6.9 mostra em (a) duas curvas que serão interpoladas, uma com quatro e outra com seis pontos. Depois da etapa de atribuição de pontos (b) cada uma contém 12 pontos distribuídos de acordo com a técnica apresentada.

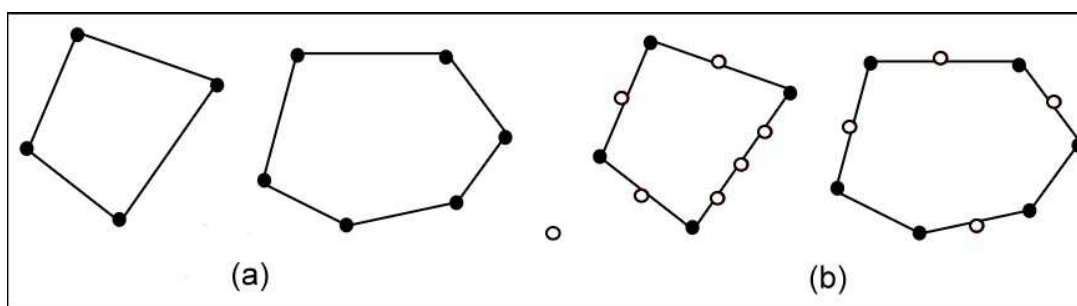


Figura 6.9: Balanceamento de pontos

Foi testada a idéia de se atribuir somente a diferença de pontos entre as duas curvas para a curva com menor pontos, porém a malha resultante ainda contava com uma superfície distorcida. Então apesar da técnica de atribuir a soma dos dois conjuntos de pontos para cada curva gerar muito mais pontos que a técnica de atribuir somente a diferença, ela permite um melhor balanceamento de pontos entre as duas curvas.

Após as etapas de definição do ponto inicial e balanceamento de pontos, basta ser feita uma triangulação entre os vértices das duas curvas. Para a triangulação é gerada uma face (triângulo) para cada dois pontos consecutivos de primeira curva e um da segunda, como ilustrado na Figura 6.10 onde temos as faces “P1Q1P2” e “Q1P2Q2”, respectivamente. Depois de geradas as faces é escrito em VRML a geometria e topologia de cada face pertencente a superfície final do objeto, como apresentado na Figura 5.4 do capítulo anterior.

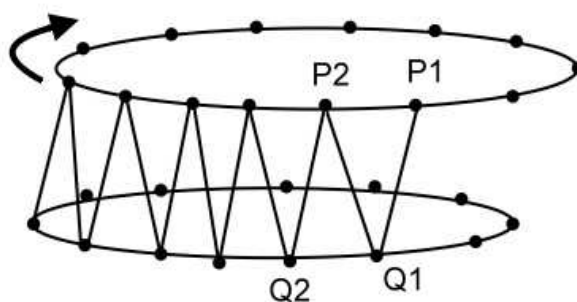


Figura 6.10: Triangulação para geração de malha

Depois de finalizada a implementação foram realizados testes para a validação da proposta, apresentados no capítulo a seguir.

## 7 *Resultados*

Neste capítulo serão apresentados os resultados obtidos do algoritmo de interpolação desenvolvido. Com relação aos dados utilizados como entrada para o algoritmo, alguns deles foram editados na aplicação desenvolvida e outros foram cedidos por autores da área. Estes últimos tratam-se de dados reais relativos a partes do corpo humano obtidos através de aparelhos de medição.

### 7.1 **Dados Editados**

Durante o período de implementação foram realizados vários testes com dados editados na aplicação desenvolvida para desenhar as curvas, o que foi de grande importância para testes e otimizações no algoritmo até se chegar em uma solução eficiente. A Figura 7.1 apresenta os resultados obtidos para três diferentes situações de interpolação sem mudança topológica, onde são ilustradas para cada situação: as curvas, o modelo em *wireframe* e o modelo final. Em (a) é mostrado um caso com duas curvas convexas, em (b) uma curva convexa é interpolada com uma côncava, e em (c) é feita a interpolação de duas curvas côncavas.



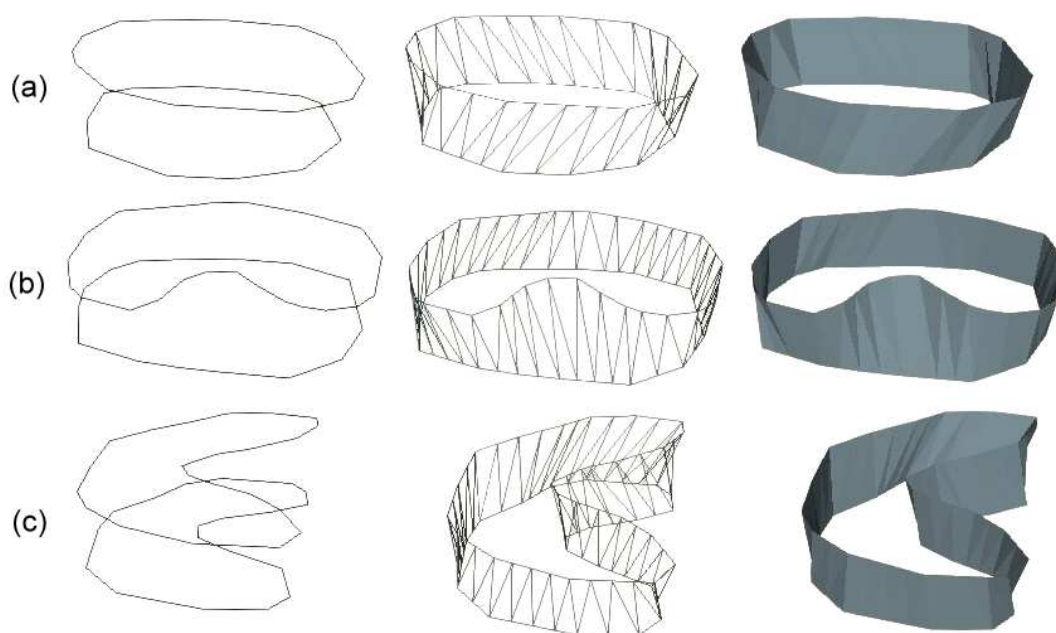


Figura 7.1: Curvas convexas e côncavas

Nota-se que para os três casos ilustrados a superfície foi gerada como o esperado, sem intersecções, conectando os pontos de modo que a malha não fique retorcida. O algoritmo também foi testado para casos onde a forma das duas curvas a serem interpoladas não é semelhante. A Figura 7.2 apresenta um caso de interpolação de duas curvas com aparência bastante distintas, onde a malha final gerada também não possui nenhuma intersecção proporcionando à superfície uma aparência de acordo com o que seria esperado. Neste exemplo, o conjunto de curvas inicial possuía 33 pontos e foram atribuídos mais 33 na etapa de geração de malha.

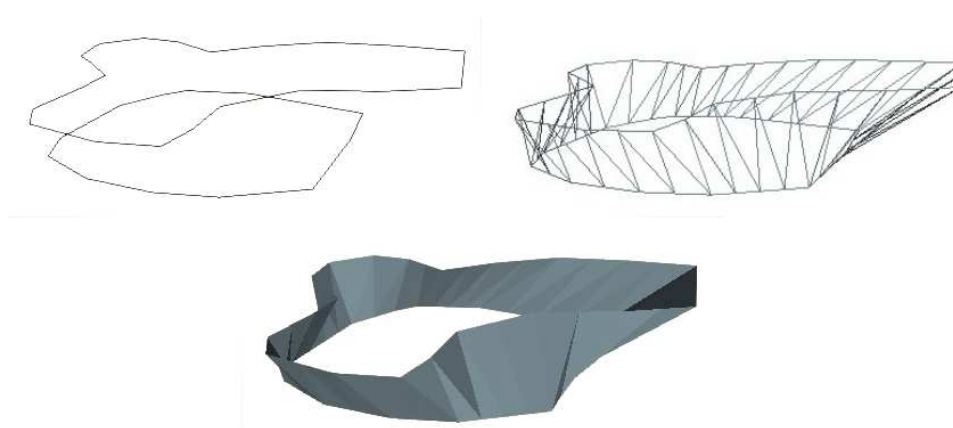


Figura 7.2: Curvas com formas distintas

Visto que a interpolação em casos que não ocorrem mudança de topologia ocorreu de ma-

neira desejável, são apresentados agora alguns exemplos de casos onde a mudança acontece, ou seja, quando uma curva é interpolada com mais de uma outra. A Figura 7.3 apresenta um resultado de interpolação de uma curva com outras duas, onde o conjunto inicial de curvas contava com 24 pontos e foram atribuídos mais 42 pontos para a geração de malha.

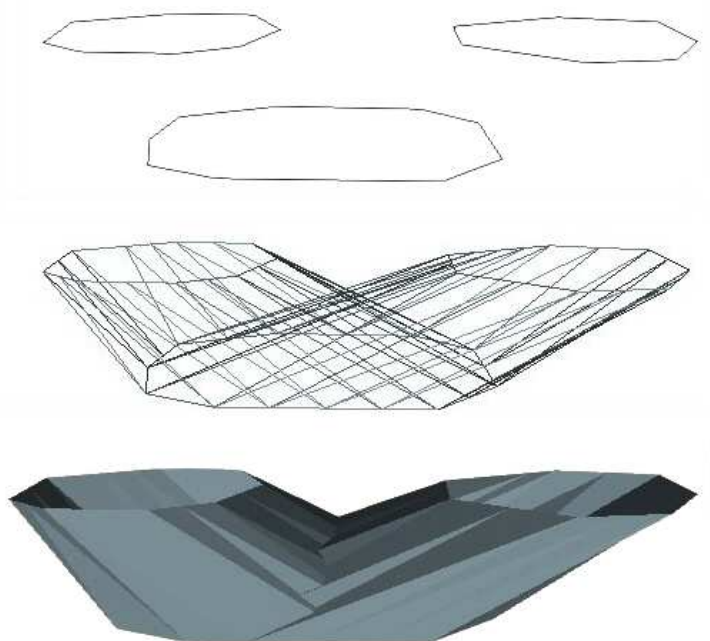


Figura 7.3: Mudança de topologia

Quando acontece a mudança de topologia, ocorre uma sobreposição malha que proporciona o efeito visual da bifurcação do objeto (*branching*). Como já explicado na seção 5.3 a bifurcação ocorre sem nenhum cálculo adicional de intersecção de superfícies, sendo resultado direto da geração de malha não afetando na visualização externa do objeto que é o foco deste trabalho.

O principal fato a ser mencionado quando trata-se de mudança de topologia é a definição da correspondência entre as curvas, que neste algoritmo conta com o parâmetro  $\Delta$  definido pelo usuário. A Figura 7.4 apresenta diferentes resultados gerados pelo algoritmo a partir de um conjunto inicial de curvas com mudança topológica.

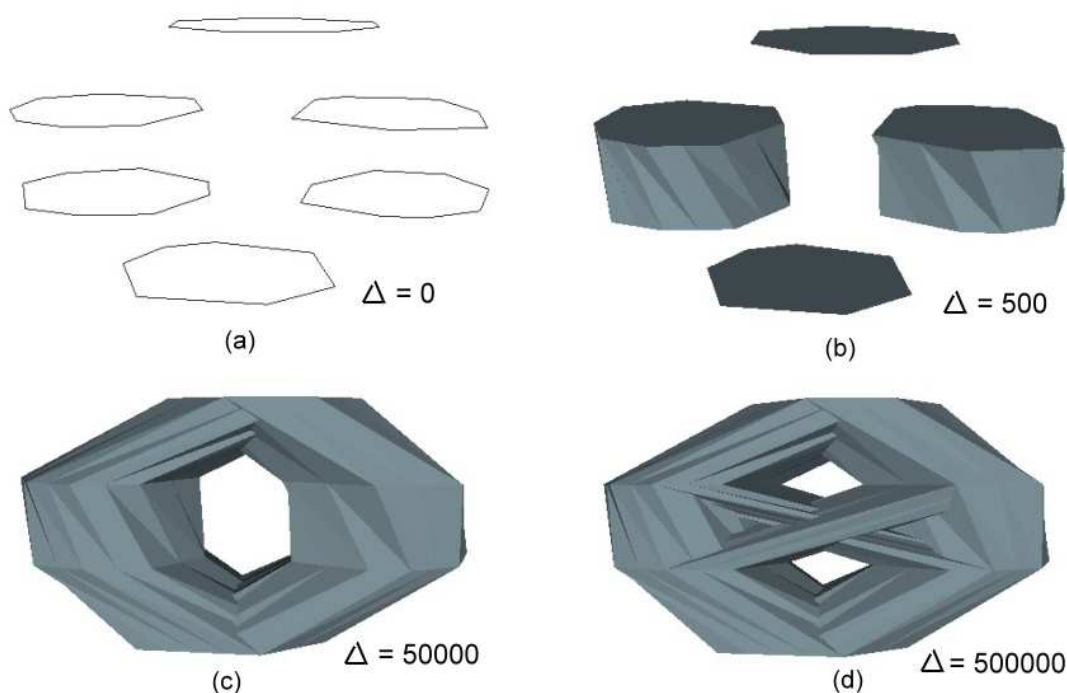


Figura 7.4: Diferentes valores para  $\Delta$

Para o conjunto de curvas apresentado, o algoritmo calculou a distância mínima (400) e máxima (500000) entre os centróides para cada par de planos. Satisfazendo a heurística já explicada na seção 5.1 deste trabalho, para o valor zero atribuído à  $\Delta$  (a) nenhuma curva é interpolada enquanto para o valor igual a distância máxima (d), todas as curvas são interpoladas com as curvas do plano consecutivo. É importante mencionar que a flexibilidade na escolha das correspondências entre as curvas oferecida pelo  $\beta$ -connection no trabalho de Vargas (2001), ilustrado na Figura 4.6, também foi alcançada neste trabalho.

O último teste editado a ser apresentado é o de um conjunto de curvas que representa a estrutura de canais, aplicação preferencial do algoritmo. A Figura 7.5 apresenta um conjunto inicial contendo 124 pontos em 12 curvas distribuídas em 8 planos. Neste exemplo o conjunto de curvas além de contar com mudança de topologia também possui a presença de topo e base, que são casos onde curvas presentes no meio do conjunto representam o fim ou o início de um caminho de interpolação, caso que pode ocorrer em estruturas de canais.

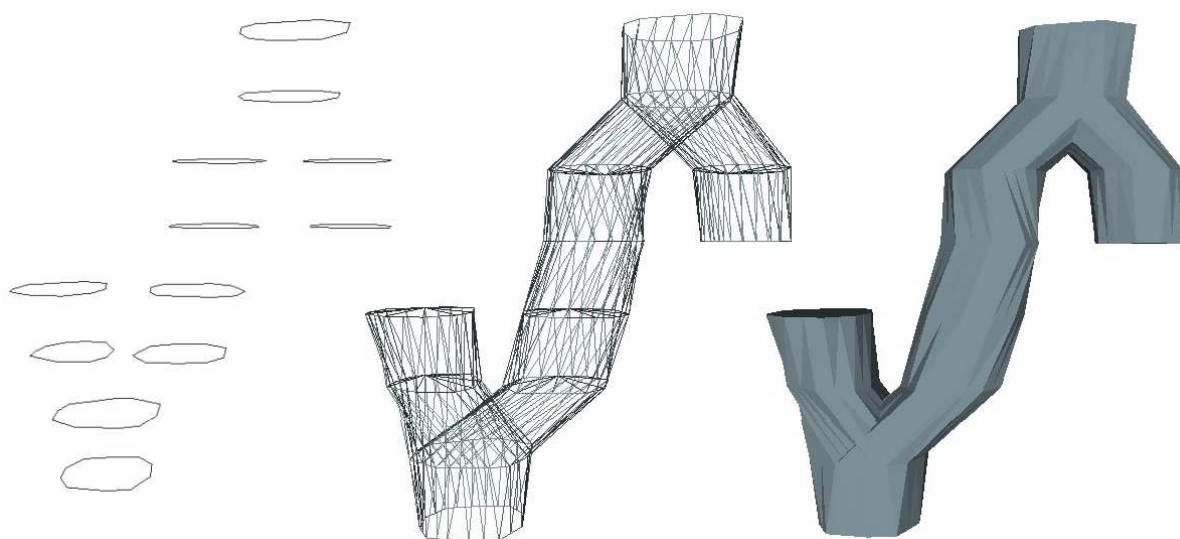


Figura 7.5: Canais

## 7.2 Dados Reais

Depois de validado o algoritmo, a partir dos testes feitos com os dados editados na aplicação, foram utilizados dados reais relativos a imagens médicas para verificar a eficiência do algoritmo. Estes dados foram recebidos no formato apresentado no apêndice A [Barequet, 2006] e adaptados para a estrutura em XML proposta.

Os documentos que contém os dados de cada exemplo a seguir foram disponibilizados no *site* que será apresentado no final deste capítulo. O primeiro exemplo ilustrado na Figura 7.6 apresenta a reconstrução de um fêmur a partir de um conjunto de curvas contendo o total de 1606 pontos distribuídos em 33 planos.

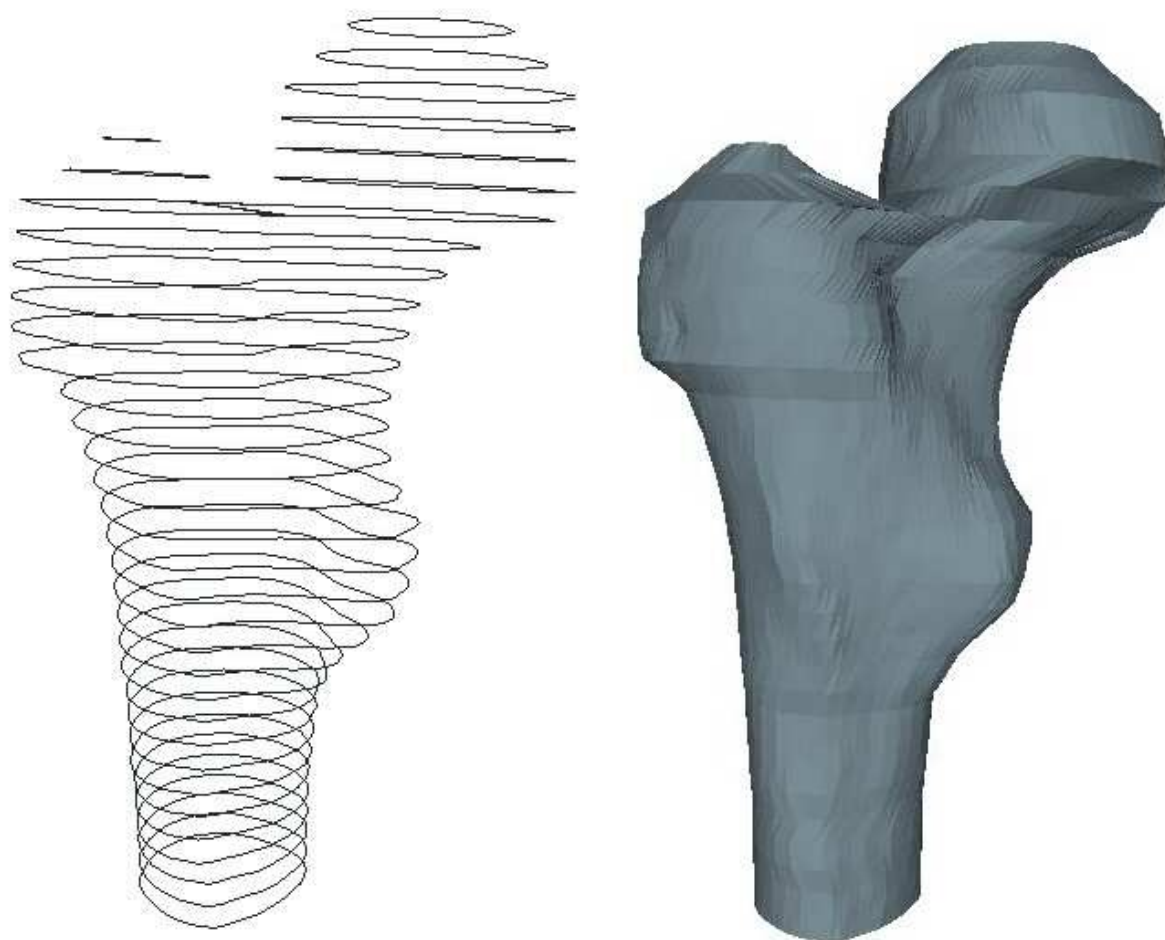


Figura 7.6: Reconstrução de um fêmur

O exemplo da Figura 7.6, apesar de não tratar de estruturas de canais, teve um bom resultado devido ao fato da estrutura não ser muito complexa, contendo apenas um caso de mudança de topologia. Ao final da reconstrução foram adicionados 5108 pontos.

Outro exemplo reconstruído foi o de pulmões a partir de um conjunto de curvas contendo 5210 pontos distribuídos em 20 planos, como ilustrado na figura 7.7, onde também ocorre mudança de topologia quando o canal do centro se divide em dois, e também quando se conecta com os dois pulmões.

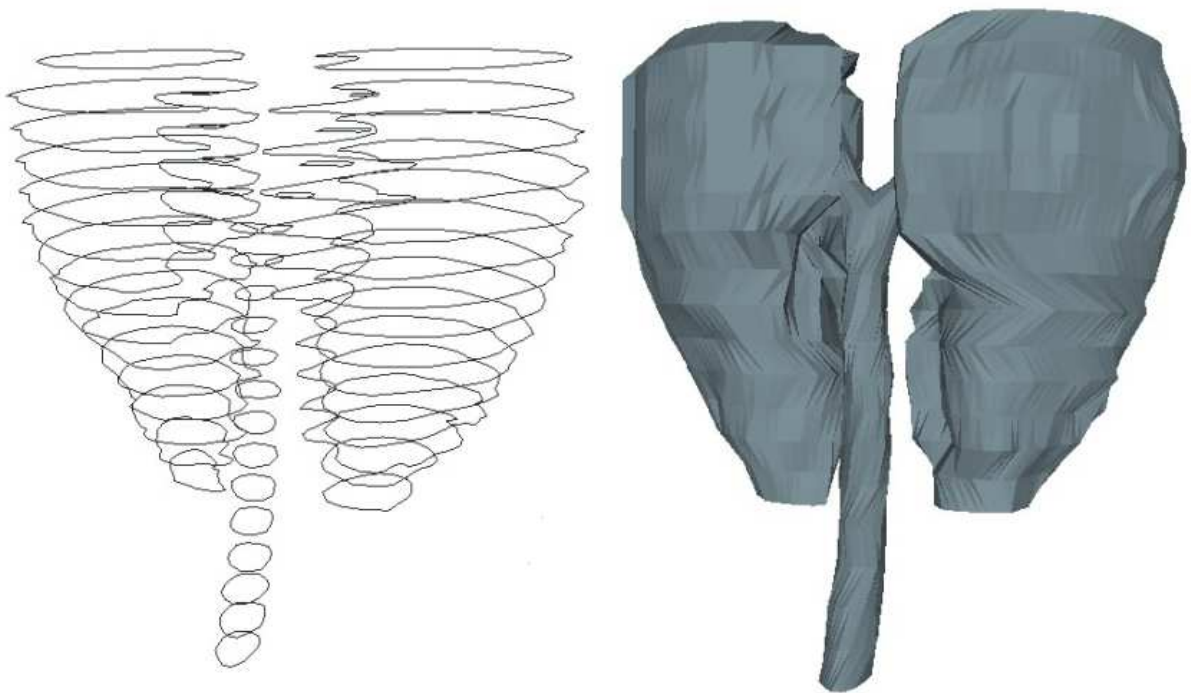


Figura 7.7: Reconstrução de pulmões

A estrutura do exemplo da Figura 7.7 é um pouco mais complexa em relação a estrutura do fêmur apresentado anteriormente, mesmo assim a superfície final gerada também teve um bom resultado, pois o modelo é formado por 3 partes (dois pulmões e um canal) com estruturas sem muitos detalhes. Ao final da reconstrução foram adicionados 6512 pontos.



Como já mencionado, o algoritmo implementado tem como aplicação preferencial a visualização de canais, onde é raro existir detalhes no conjunto que será interpolado. A Figura 7.8 ilustra a reconstrução de canais (vasos sanguíneos) através de um conjunto de 4648 pontos distribuídos em 24 planos, com duas diferentes visões do mesmo objeto reconstruído.

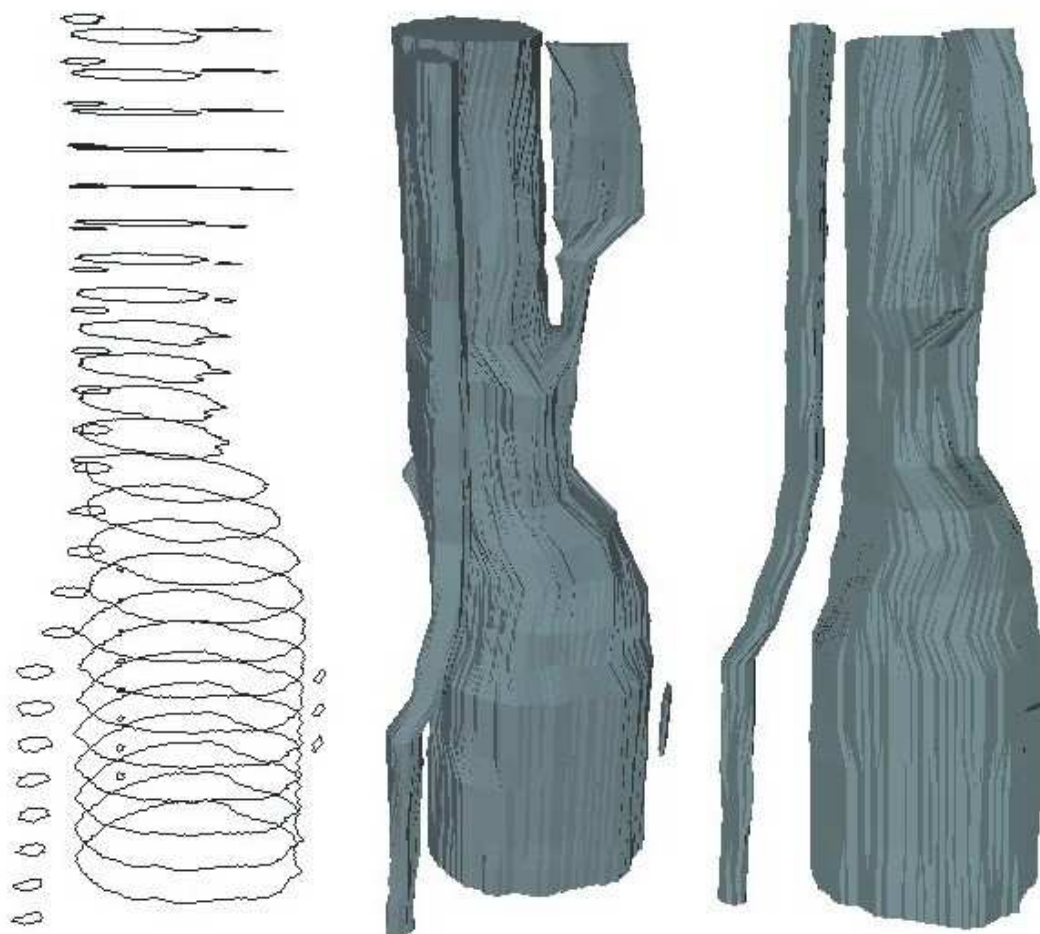


Figura 7.8: Reconstrução de canais (vasos sanguíneos)

Ao final da reconstrução foram adicionados 16622 pontos. O exemplo da Figura 7.8 foi o que mais teve pontos adicionados pelo fato das curvas que formam a maior estrutura do conjunto, que foi interpolada do primeiro ao último plano, contar com um grande número inicial de pontos (aproximadamente 100 pontos por curva). É importante mencionar que quanto maior a distribuição de pontos para as curvas, maior a suavidade da malha final, então apesar de muitos pontos serem adicionados, neste exemplo, a superfície gerada contou com uma visualização de acordo com o esperado.

O caso que mais chamou a atenção durante os testes foi a reconstrução de um coração, apresentado na Figura 7.9 formado por um conjunto de 1285 pontos distribuídos em 30 planos. Neste exemplo nota-se que a região superior do conjunto possui curvas pequenas e mais próximas entre si em relação ao resto, ou seja, existe uma maior concentração de curvas nesta região.

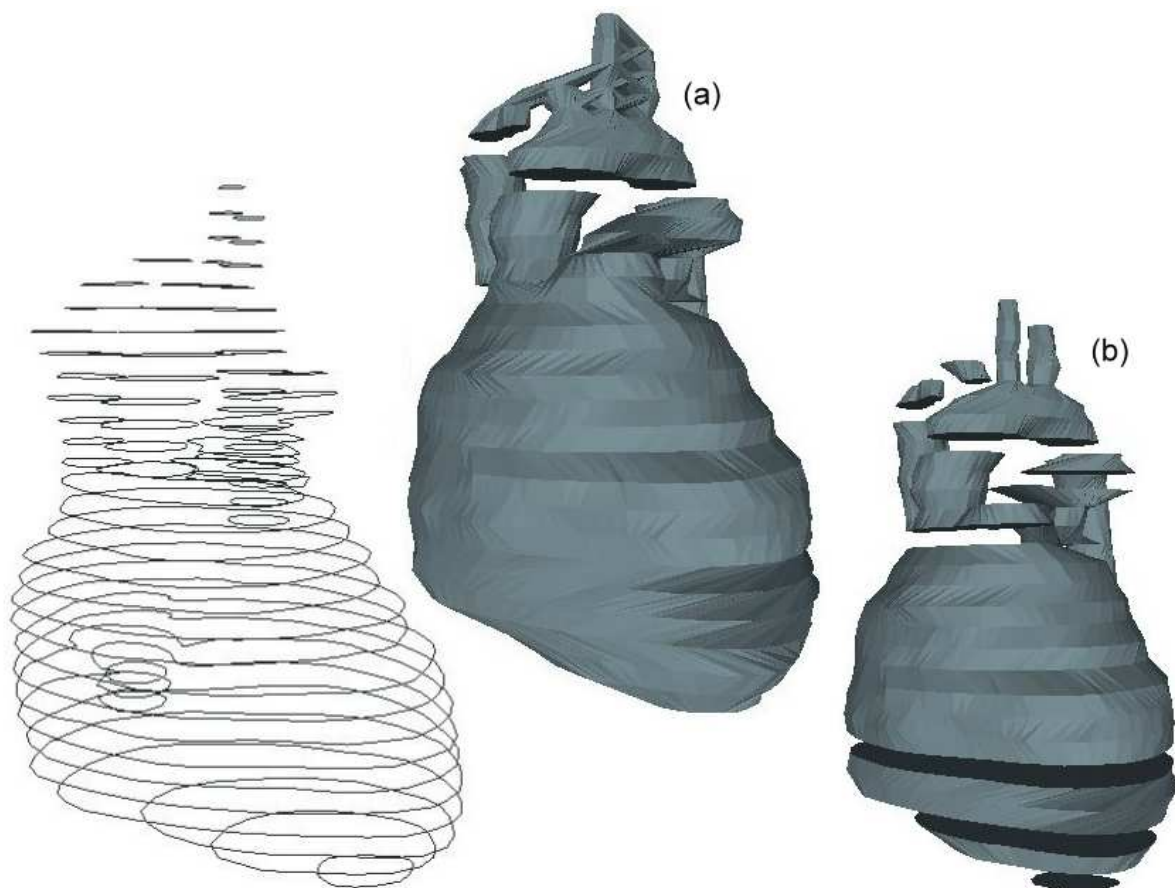


Figura 7.9: Reconstrução de um coração

Os conjuntos de curvas com esta característica podem ter um resultado não desejável pelo fato de todas as curvas da região mais concentrada se conectarem por terem como distância entre elas um valor muito pequeno em relação ao resto do conjunto, como apresentado em (a) na Figura 7.9. Se for definido um valor para  $\Delta$  muito pequeno para que estas curvas não se conectem, como apresentado em (b) na Figura 7.9, as demais curvas que deveriam ser interpoladas podem acabar não se conectando por ter uma distância maior que o  $\Delta$ .



## 7.3 Discussão

Foi feita uma análise e discussão em relação aos resultados referentes aos testes com dados reais. A primeira análise é em relação ao parâmetro de correspondência  $\Delta$  utilizado para gerar cada um dos exemplos ilustrados como apresentado na Figura 7.10.

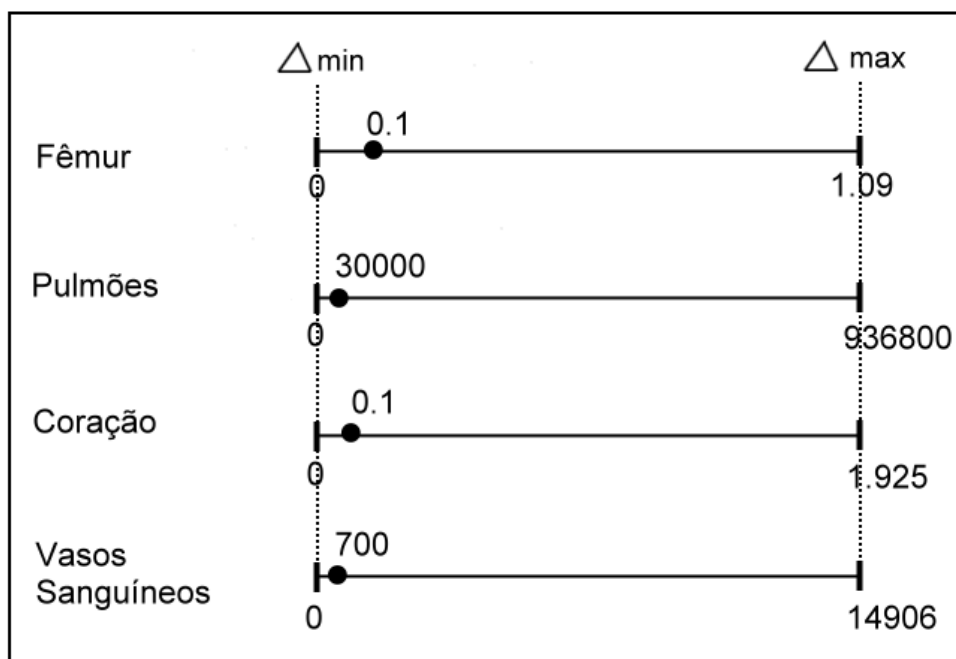


Figura 7.10: Análise da variação do  $\Delta$

Pode-se perceber que para todos os casos o valor atribuído a  $\Delta$  é pequeno em relação ao intervalo de distâncias  $[\Delta_{\min}, \Delta_{\max}]$ . Isto ocorre pelo fato de que para os testes realizados era interessante a conexão somente das curvas que estivessem próximas. O caso em que  $\Delta$  ficou mais longe do valor mínimo de distância foi na reconstrução do fêmur, onde só ocorre uma vez a mudança de topologia e a maioria dos planos possuem somente uma curva, que sempre é interpolada com a curva do plano seguinte. É possível concluir desta análise que como geralmente se deseja interpolar somente as curvas próximas entre si, o valor de  $\Delta$  será sempre pequeno em relação ao intervalo de distâncias.

Uma segunda análise foi feita em relação ao tempo de reconstrução, o gráfico ilustrado na Figura 7.11 apresenta os de tempo registrados em função da quantidade de pontos gerados a partir dos testes feitos com dados reais.

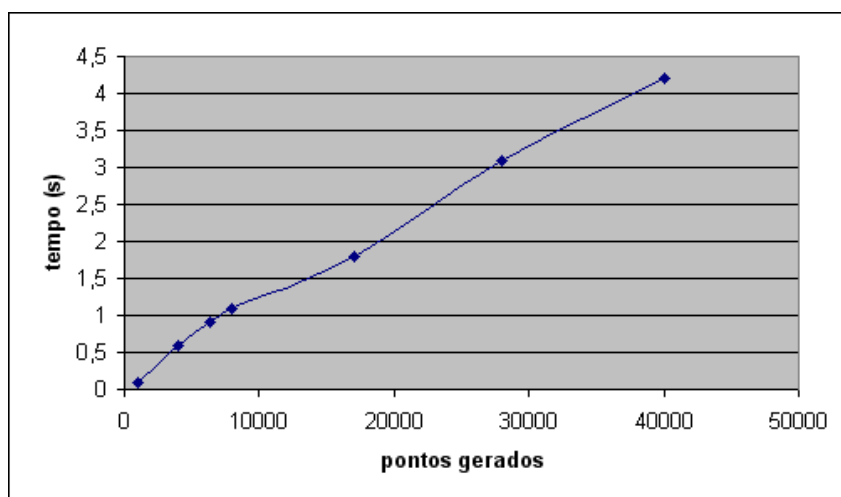



Figura 7.11: Análise do tempo de reconstrução

Estes dados foram registrados em uma máquina com processador AMD Athlon 1.3 GHz com 512 MB de memória, sem placa aceleradora 3D. Os resultados da Figura 7.11 mostram que a medida em que mais pontos são gerados para o balanceamento do número de pontos entre as curvas, aumenta o tempo total de reconstrução de um modelo. Em um exemplo onde foram gerados em torno de 1000 pontos o tempo de reconstrução foi de 0.1 segundo enquanto em outro exemplo onde foram gerados 16622 pontos o tempo ficou em torno de 1.8 segundo. A fim de obter um número maior de experimentos foram testados valores altos para  $\Delta$  gerando até 40000 pontos, que é um caso raro de acordo com os testes realizados, resultando tempo de reconstrução em torno de 4.2 segundos. A curva gerada no gráfico da Figura 7.11 tem uma aparência linear, oque permite afirmar que o tempo de reconstrução será diretamente proporcional ao número de pontos gerados.

Todos os dados referentes aos resultados obtidos, bem como a aplicação desenvolvida e o código fonte do algoritmo estão disponíveis no *site*, ilustrado pela Figura 7.12, que foi desenvolvido durante o trabalho no endereço [www2.joinville.udesc.br/~larva/deltaconnection](http://www2.joinville.udesc.br/~larva/deltaconnection).



# -connection

Interpolação de curvas com mudança topológica

## O Projeto

Neste trabalho, uma técnica de reconstrução tridimensional baseada na interpolação de curvas com mudança de topologia foi proposta. Para resolver este problema, que tem, por exemplo, aplicações na área médica e na representação computacional de terrenos, conceitos e técnicas de modelagem geométrica foram pesquisados, como a criação e representação de modelos tridimensionais a partir de fatias bidimensionais. Uma aplicação foi implementada para demonstrar a solução proposta para este problema de maneira eficiente.

Autor: Guilherme Rossetti Anzollin  
Orientador: Marcelo da Silva Hounsell

## Resultados

Para visualizar os resultados é necessário instalar um [plugin](#) e selecionar a opção "fit" do browser que será instalado.

Reconstrução de um osso	<a href="#">wrl (visualização)</a>	249KB	<a href="#">xml</a>	25KB
Reconstrução de pulmões	<a href="#">wrl (visualização)</a>	279KB	<a href="#">xml</a>	16KB
Reconstrução de um coração	<a href="#">wrl (visualização)</a>	269KB	<a href="#">xml</a>	20KB
Reconstrução de vasos sanguíneos	<a href="#">wrl (visualização)</a>	704KB	<a href="#">xml</a>	27KB

## Downloads

<a href="#">Monografia</a>	<a href="#">Programa</a>	<a href="#">Código Fonte</a>
----------------------------	--------------------------	------------------------------

© Copyright Universidade do Estado de Santa Catarina - Todos os direitos reservados  
Última atualização em 14 de novembro de 2006

Figura 7.12: Site do trabalho

## 8 *Conclusão*

A busca por um método de interpolação de curvas com mudança topológica que seja visualmente interessante e rápido exigiu um maior entendimento de conceitos relacionados à reconstrução 3D. Posteriormente, a identificação da abordagem heurística propiciou o caminho que parece se adequar melhor aos objetivos traçados para este trabalho.

Uma solução para o problema de interpolação de curvas com mudança topológica baseado em abordagens heurísticas de reconstrução tridimensional foi apresentada. Esta solução tem como critérios de eficiência principalmente a rapidez e simplicidade no cálculo das etapas de reconstrução.

O algoritmo que realiza a interpolação das curvas foi desenvolvido com ênfase nas etapas de definição de correspondências e geração de malha. Para a etapa de correspondência o algoritmo conta com um parâmetro de entrada, que o torna flexível para decidir no resultado do objeto reconstruído. Para a etapa de geração de malha são adicionados pontos às curvas sem alterar a topologia das mesmas com objetivo de balancear os pontos entre as curvas e suavizar a superfície gerada.

O local de bifurcações, que ocorrem quando existe mudança de topologia entre as curvas, é decidido automaticamente, sendo assim, esta etapa de reconstrução não foi detalhada no algoritmo. Então, a reconstrução realizada será voltada para a visualização de toda estrutura do objeto e não para a precisão, sendo que a área de aplicação recomendada é para a visualização exterior de canais.

Os testes foram realizados com dados gerados pelo editor de curvas implementado e também

com dados reais relativos à imagens médicas cedidos por autores da área. Os resultados apresentados a partir dos testes permitiram a validação da proposta mostrando que o presente trabalho alcançou seus objetivos traçados para o período com eficiente detalhamento.

## 8.1 Trabalhos Futuros

Ficam aqui algumas sugestões de trabalhos futuros para acrescentar ao algoritmo de interpolação ou então ao editor de curvas desenvolvido. Em relação ao editor de curvas, podem ser realizados trabalhos como por exemplo:

- A adição de diferentes formas de desenho, por exemplo, formas pré-definidas (retângulos, círculos, elipses);
- A visualização 3D em tempo real do conjunto de planos que está sendo editados.

Em relação ao algoritmo, segue abaixo algumas sugestões para trabalhos mais elaborados:

- Definir métricas para avaliar a superfície final do objeto e através dessas otimizar a malha a fim de atingir uma superfície com o mínimo de torção possível, uma dessas métricas pode ser o comprimento das arestas que formam cada face da superfície final do objeto. Quanto menor o comprimento destas arestas, maior a probabilidade de a superfície não ter torções, visto que uma superfície torcida é formada por arestas maiores;
- Acrescentar ao algoritmo o tratamento do local de bifurcação, identificando quando irá ocorrer uma mudança de topologia e tratando esta mudança. Uma possível maneira de identificar quando uma curva irá se dividir em duas é identificando a concavidade que a mesma vai formando até a divisão;
- Acrescentar ao algoritmo diferentes opções de cálculo de centróides e diferentes técnicas de geração de malha *tiling*;
- Acrescentar ao processo de reconstrução a opção de definir valores diferentes para o parâmetro  $\Delta$  a cada par de planos consecutivos.

## *Referências*

- AZEVEDO, E.; CONCI, A. *Computação Gráfica: Teoria e Prática*. [S.l.]: Editora Campus, 2003. 353 p.
- BAJAJ, C. L.; COYLE, E. J.; LIN, K.-N. Arbitrary topology shape reconstruction from planar cross sections. *Graphical Models and Image Processing*, v. 58, n. 6, p. 524–543, 1996.
- BAJAJ, C. L.; COYLE, E. J.; LIN, K.-N. Tetrahedral meshes from planar cross sections. *Computer Methods in Applied Mechanics and Engineering*, v. 179, p. 31–52, 1999.
- BAREQUET. *Publicly-Available Resources*. Disponível em: <ftp://ftp.cs.technion.ac.il/pub/barequet/psdb>. Acessado em: 16 nov. 2006.
- BAREQUET, G.; SHARIR, M. Piecewise-linear interpolation between polygonal slices. *Computer Methods in Applied Mechanics and Engineering*, v. 63, n. 2, p. 251–272, 1996.
- CAREY, R.; BELL, G. *The annotated VRML 2.0 reference manual*. [S.l.]: Reading, Massachusetts: Addison-Wesley, 1997. 504 p.
- DECIO, O. C. *XML: Guia de Consulta Rápida*. [S.l.]: São Paulo. Novatec Editora Ltda, 2000. 96 p.
- DIEGUEZ, J. P. P. *Modelagem Geométrica para Computação Gráfica*. [S.l.]: Achiamé, 1989. 165 p.
- FIGUEIREDO, L. H.; CARVALHO, P. C. P. *Introdução à Geometria Computacional*. [S.l.]: Rio de Janeiro: Instituto de Matemática Pura e Aplicada, 18º Colóquio Brasileiro de Matemática, 1991. 108 p. Disponível em <http://www.inf.ufpr.br/andre/geom/geom.html>. Acessado em 12 mai. 2006.
- FILHO, A. C. *Modelagem Geométrica: Representação e Manipulação de Objetos Geométricos Utilizando o Computador*. [S.l.]: São José Do Rio Preto: Sociedade Brasileira de Matemática Aplicada e Computacional, 1998. 30 p.
- FOLEY, J. D. et al. *Computer Graphics: Principles and Practice*. 2. ed. [S.l.]: Addison-Wesley Publishing Company, 1996. 1174 p.
- GATTASS, M.; PEIXOTO, A. Reconstrução de superfícies a partir de seções bidimensionais. *Technical Report MCC 28/00, PUC-Rio*, Julho 2000. Disponível em [http://www.visgraf.impa.br/Data/RefBib/PS\\_PDF/peixoto-gattass-jul00/SurfaceReconstruction.pdf](http://www.visgraf.impa.br/Data/RefBib/PS_PDF/peixoto-gattass-jul00/SurfaceReconstruction.pdf).
- GRAVES, M. *Projeto de Banco de Dados em XML*. [S.l.]: São Paulo. Makron Books, 2003. 518 p.

ISO - INTERNATIONAL ORGANIZATION FOR STANDARDIZATION. *Technical Specification - ISO/TS 10303-28 -Industrial Automation Systems and Integration - Product Data Representation and Exchange - Part 28: Implementation Methods: XML Implementation of EXPRESS Schemas and Data*. [S.l.], 2002.

MCINERNEY, T.; TERZOPOULOS, D. Deformable models in medical image analysis: A survey. *Medical Image Analysis*, v. 1, n. 2, p. 91–108, 1996.

MEYERS, D.; SKINNER, S.; SLOAN, K. Surface from contours. *ACM Trans. On Graphics*, v. 11, n. 3, p. 228–258, 1992.

NONATO, L. G. et al. Beta-connection: Generating a family of models from planar cross sections. *ACM Transactions on Graphics*, v. 24, n. 4, p. 1–20, Outubro 2005.

ROMEU, R. K. et al. Impacto dos sistemas de visualização 3d e de realidade virtual nos estudos integrados de reservatórios. *Rio Oil and Gas Expo and Conference, Rio de Janeiro*, 2000.

SPECK, J. H. *Avaliação Comparativa das Metodologias Utilizadas em Programas de Modelagem Sólida*. 185 p. Dissertação (Mestrado) — PPGE/UFSC, 2001.

TRAINA, A. J. M.; OLIVEIRA, M. C. F. *Apostila de Computação Gráfica*. [S.l.], Setembro 2004. Disponível em <<http://infovis.ucpel.tche.br/luzzardi/ComputerGraphics.pdf>>. Acessado em 10 abr. 2006.

TREECE, G. M. et al. Surface interpolation from sparse cross-sections using region correspondence. *Technical Report CUED/F-INFENG/TR 342*, v. 19, n. 11, p. 1106–1114.

VARGAS, A. J. C. *Beta-Conexão: Uma Família de Objetos Tridimensionais Reconstruídos a partir de Seções Planares*. 73 p. Dissertação (Mestrado) — Instituto de Ciências Matemáticas e de Computação, USP, São Carlos, 2001.

VTK. *Visualization Toolkit*. Disponível em: <<http://public.kitware.com/VTK/>>. Acessado em: 16 nov. 2006.

## ***APÊNDICE A – Dados de fontes Reais***

```

! This file is taken from the database of:  Gill Barequet
!                                           Dept. of Computer Science
!                                           Tel Aviv University
!                                           barequet@math.tau.ac.il

general

  organ      Heart
  sex        ?
  age        ?
  illness    ?
  date       ?
  scanner    MRI
  contributor ?

address

  Inst._Biokybernetik_und_Biomedizinische_messtechnik/_Univ._Karlsruhe

units      MM
scale_x    1.0
scale_y    1.0
scale_z    1.0
comment    ?

end_general

slices

  30 65 1285

```



```

1 3.787 1
15      0.400  0.576      0.431  0.543      0.512  0.526      0.595  0.524
      0.661  0.526      0.739  0.536      0.782  0.562      0.789  0.604
      0.758  0.678      0.716  0.730      0.623  0.761      0.524  0.775
      0.427  0.761      0.377  0.697      0.374  0.630
. . .
30 0.668 1
      7      0.052  0.081      0.069  0.123      0.040  0.156      -0.007  0.164
      -0.036  0.126      -0.047  0.066      -0.002  0.055
end_slices

```

## ***APÊNDICE B – Dados Reais adaptados para XML***

```

<?xml version="1.0" encoding="ISO-8859-1"?>
<Slices>
  <Info>Slices para a reconstrução de um coração</Info>
  <x_dimension><min>0</min><max>700</max></x_dimension>
  <y_dimension><min>0</min><max>370</max></y_dimension>
  <n_points>1285</n_points>
  <n_curves>65</n_curves>
  <n_slices>30</n_slices>
  <dist_slices>0.1</dist_slices>
  <slice>
    <n_curves>1</n_curves>
    <curve>0.400 0.576  0.431 0.543  0.512 0.526  0.595 0.524
            0.661 0.526  0.739 0.536  0.782 0.562  0.789 0.604
            0.758 0.678  0.716 0.730  0.623 0.761  0.524 0.775
            0.427 0.761  0.377 0.697  0.374 0.630</curve>
  </slice>
  . . .
  <slice>
    <n_curves>1</n_curves>
    <curve>0.052 0.081  0.069 0.123  0.040 0.156  -0.007 0.164
            -0.036 0.126  -0.047 0.066  -0.002 0.055</curve>
  </slice></Slices>

```

## ***APÊNDICE C – Algoritmos utilizados para o cálculo de centróides e distâncias***

### **C.1 Cálculo dos centróides**

```

public float[] boundingBox (float[] [] curva)
{
    float min[] = {curva[0][0],curva[0][1]};
    float max[] = {curva[0][0],curva[0][1]};
    float centroide[] = new float[3];
    for(int i=0; i<curva.length; i++)
    {
        if(min[0] > curva[i][0]) min[0] = curva[i][0];
        if(max[0] < curva[i][0]) max[0] = curva[i][0];
        if(min[1] > curva[i][1]) min[1] = curva[i][1];
        if(max[1] < curva[i][1]) max[1] = curva[i][1];
    }
    centroide[0] = (min[0]+max[0])/2;
    centroide[1] = (min[1]+max[1])/2;
    centroide[2] = (curva[0][2]); // distância entre planos
    return centroide;
}

```

## C.2 Cálculo da matriz de distâncias

```
public double[] [] matrizDeDistancias(double[] [] centrosP1, double[] [] centrosP2)
{
    double[] [] dist = new double[centrosP1.length][centrosP2.length];
    for(int i=0; i<centrosP1.length; i++)
        for(int j=0; j<centrosP2.length; j++)
            dist[i][j]= Math.pow((centrosP1[i][0]-centrosP2[j][0]),2)+
                Math.pow((centrosP1[i][1]-centrosP2[j][1]),2);
    return dist;
}
```