
Instituto de Computação
Universidade Estadual de Campinas

Modelagem de Deformação do Espaço 2.5D para Estruturas Biológicas

Elisa de Cássia Silva Rodrigues

Este exemplar corresponde à redação final da
Dissertação devidamente corrigida e defendida
por Elisa de Cássia Silva Rodrigues e aprovada
pela Banca Examinadora.

Campinas, 03 de Outubro de 2011.


Anamaria Gomide (Orientadora)


Jorge Stolfi (Co-orientador)

Dissertação apresentada ao Instituto de Com-
putação, UNICAMP, como requisito parcial para
a obtenção do título de Mestre em Ciência da
Computação.

FICHA CATALOGRÁFICA ELABORADA POR ANA REGINA MACHADO – CRB8/5467
BIBLIOTECA DO INSTITUTO DE MATEMÁTICA, ESTATÍSTICA E
COMPUTAÇÃO CIENTÍFICA – UNICAMP

R618m Rodrigues, Elisa de Cássia Silva, 1984-
Modelagem de deformação do espaço 2.5D para
estruturas biológicas / Elisa de Cássia Silva Rodrigues. –
Campinas, SP : [s.n.], 2011.

Orientador: Anamaria Gomide.
Coorientador: Jorge Stolfi.
Dissertação (mestrado) - Universidade Estadual de
Campinas, Instituto de Computação.

1. Spline, Teoria do. 2. Computação gráfica.
3. Biologia - Simulação por computador. 4. Modelagem
geométrica. I. Gomide, Anamaria, 1949-. II. Stolfi, Jorge,
1950-. III. Universidade Estadual de Campinas. Instituto de
Computação. IV. Título.

Informações para Biblioteca Digital

Título em inglês: 2.5D space deformation modeling for biological structures

Palavras-chave em inglês:

Spline theory

Computer graphics

Biology - Computer simulation

Geometric modeling

Área de concentração: Ciência da Computação

Titulação: Mestre em Ciência da Computação

Banca examinadora:

Anamaria Gomide [Orientador]

Marcus Vinícius Alvim Andrade

Hélio Pedrini

Data da defesa: 03-10-2011

Programa de Pós-Graduação: Ciência da Computação

TERMO DE APROVAÇÃO

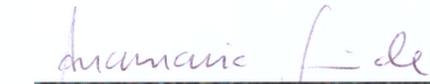
Dissertação Defendida e Aprovada em 03 de outubro de 2011, pela Banca examinadora composta pelos Professores Doutores:



Prof. Dr. Marcus Vinícius Alvim Andrade
CCE / UFV



Prof. Dr. Hélio Pedrini
IC / UNICAMP



Prof^a. Dr^a. Anamaria Gomide
IC / UNICAMP

Modelagem de Deformação do Espaço 2.5D para Estruturas Biológicas

Elisa de Cássia Silva Rodrigues¹

Outubro de 2011

Banca Examinadora:

- Anamaria Gomide (Orientadora)
- Marcus Vinícius Alvim Andrade
Departamento de Informática - UFV
- Hélio Pedrini
Instituto de Computação - UNICAMP
- Sônia Maria Gomes (Suplente externo)
Departamento de Matemática Aplicada - IMECC - UNICAMP
- Pedro Jussieu de Rezende (Suplente interno)
Instituto de Computação - UNICAMP

¹Suporte financeiro de: Bolsa CAPES (processo 01 P-04388-2010) 2009–2011

Resumo

Métodos de deformação são importantes em áreas como modelagem geométrica e animação computacional. Na biologia, a modelagem de forma, crescimento, movimento e patologias de organismos microscópicos vivos ou células requerem deformações suaves, as quais são essencialmente 2D com poucas mudanças de profundidade. Nesta dissertação, apresentamos um método de deformação do espaço 2.5D suave. O modelo 3D do organismo é modificado deformando uma grade de controle formada por prismas que o envolve. A técnica de interpolação spline é usada para satisfazer o requisito de suavidade (C^1). Implementamos esse método em um editor que torna possível definir e modificar a deformação de uma forma amigável usando o mouse. Os resultados experimentais mostram que o método é simples e efetivo.

Abstract

Shape deformation methods are important in such fields as geometric modeling and computer animation. In biology, the modeling of shape, growth, movement and pathologies of living microscopic organisms or cells requires smooth deformations, which are essentially 2D with little change in depth. In this master thesis, we present a smooth 2.5D space deformation method. The 3D model of the organism is modified by deforming an enclosing control grid of prisms. Spline interpolation is used to satisfy the smoothness (C^1) requirement. We implemented this method in an editor which makes it possible to define and modify the deformation with the mouse in a user-friendly way. The experimental results show that the method is simple and effective.

Agradecimentos

Antes de tudo, agradeço a Deus por me conceder saúde e perseverança para concluir mais uma etapa da minha vida e por colocar pessoas especiais no meu caminho, nas quais posso confiar.

À minha mãe, Maria Isabel da Silva Rodrigues, e ao meu pai, Flávio Galvão Fortunato Rodrigues, pelo apoio psicológico e financeiro e, principalmente, pelo incentivo, compreensão e amor incondicional oferecido durante toda minha vida, em especial, nesta etapa. Ao meu irmão, Eduardo Luis da Silva Rodrigues, e aos meus familiares pelo apoio em todos os momentos.

Ao Ricardo Dutra da Silva pelo suporte e apoio durante todo o mestrado. Obrigada pelo incentivo, atenção, dedicação, paciência, amizade, por estar sempre ao meu lado, nos bons momentos e nas fases difíceis, e, principalmente, por seu amor e carinho.

Aos amigos e colegas de disciplinas que colaboraram, direta ou indiretamente, com a realização e conclusão deste trabalho, compartilhando experiências, oferecendo suporte e comemorando conquistas. Em especial, ao meu amigo Roberto Pereira que sempre esteve ao meu lado, principalmente nos momentos difíceis, por saber oferecer sua amizade sincera, aconselhar e apoiar sempre.

Aos professores Dr^a Anamaria Gomide e Dr. Jorge Stolfi pela orientação, dedicação, confiança, paciência e profissionalismo oferecidos.

À Universidade Estadual de Campinas (UNICAMP), ao Instituto de Computação (IC) e ao Laboratório de Informática Visual (LIV) pela oportunidade e infraestrutura disponibilizada durante o mestrado. Aos professores e funcionários pelo suporte e orientação.

À Capes pelo apoio financeiro.

Sumário

Resumo	v
Abstract	vi
Agradecimentos	vii
1 Introdução	1
1.1 Motivação e objetivos	1
1.2 Contribuição	2
1.3 Organização	3
2 Trabalhos Relacionados	4
2.1 Métodos de deformação para modelos 3D	4
2.1.1 Métodos de deformação do espaço	5
2.1.2 Técnicas de interpolação	6
3 Splines Polinomiais	8
3.1 Triangulações	8
3.2 Funções polinomiais em \mathbb{R}^n	9
3.3 Splines polinomiais	9
3.3.1 Coordenadas baricêntricas	10
3.3.2 Representação Bernstein-Bézier para polinômios	11
3.3.3 Coeficientes de Bézier	11
3.4 Modelagem de deformação com splines	12
4 Restrições de Continuidade em Splines	14
4.1 Assegurando continuidade C^0	15
4.2 Assegurando continuidade C^1	15
4.3 Assegurando controle local	18

5	Método de Deformação do Espaço 2.5D	19
5.1	Modelo 3D e construção da grade domínio	19
5.1.1	Definição das coordenadas baricêntricas	20
5.2	Deformação da grade domínio	21
5.2.1	Deformação de uma spline	22
5.2.2	Assegurando a continuidade da spline	23
6	Representação das Splines	24
6.1	Notações adicionais	24
6.1.1	Rotulação e orientação das arestas	24
6.1.2	Rotulação e orientação das condições de quadrilátero	25
6.2	Estruturas de dados	25
6.2.1	Representação de uma triangulação	25
6.2.2	Representação de uma spline	26
6.2.3	Representação das restrições C^1	26
6.2.4	Consistência da orientação ao redor de um vértice	27
6.2.5	Sobreposição de quadriláteros	28
7	Edição dos Pontos de Controle	30
7.1	Classificação dos pontos de controle	30
7.2	Quadriláteros que usam um ponto de controle	31
7.3	Algoritmo de seleção automática dos pontos móveis	33
7.3.1	Edição de um ponto de controle <i>corner</i>	33
7.3.2	Edição de um ponto de controle <i>edge corner</i>	34
7.3.3	Edição de um ponto de controle <i>inner corner</i>	35
7.3.4	Edição de um ponto de controle <i>edge</i>	36
7.3.5	Edição de um ponto de controle <i>inner edge</i>	36
8	Atualização dos Pontos de Controle	37
8.1	Cálculo das coordenadas baricêntricas	38
8.2	Forma matricial das equações de continuidade	38
8.3	Resolvendo o sistema indeterminado	39
9	Experimentos e Resultados	44
9.1	Construção do modelo 3D e da grade	44
9.2	Resultados	45
10	Conclusões e Trabalhos Futuros	49

A Interface do Editor	51
A.1 Interface inicial do editor	51
A.2 Modos de visualização, seleção e edição	52
Referências Bibliográficas	56

Lista de Tabelas

2.1	Comparação entre células de malhas formadas por tetraedros, hexaedros e prismas.	7
9.1	Parâmetros dos modelos usados nos testes.	45
9.2	Parâmetros das splines definidas sobre as grades dos testes.	45

Lista de Figuras

1.1	Imagens microscópicas do <i>Caenorhabditis elegans</i> (a) [28] e do <i>Dileptus anser</i> (b) [14] deformados.	1
1.2	Morfologia do <i>Caenorhabditis elegans</i> (a) [7], <i>Dileptus anser</i> (c) [36] e os respectivos modelos 3D de suas formas básicas (b) e (d).	2
1.3	Uma comparação entre deformações do espaço com continuidade C^0 (a) e C^1 (b).	2
3.1	Uma triangulação válida (a) e uma triangulação inválida (b).	9
3.2	Uma função spline de \mathbb{R}^2 em \mathbb{R}	10
3.3	Sinais das coordenadas baricêntricas com relação ao triângulo u	10
3.4	Posições nominais u_{ijk} dos coeficientes de Bézier c_{ijk} para uma parte f^u de grau 5.	11
3.5	Superfície spline triangular de grau $d = 5$ em \mathbb{R}^2 mostrando o conjunto $C_{5,T}$ de posições nominais (pontos pretos).	12
3.6	Uma deformação de \mathbb{R}^2 da grade domínio T (a) na grade deformada $\phi(T)$ (b).	13
3.7	Pontos de controle de Bézier q_{ijk}^u (b) de um retalho ϕ^u de grau 3 de $\mathbb{R}^2 \rightarrow \mathbb{R}^2$ e suas posições nominais u_{ijk} no triângulo de domínio u (a). O triângulo de arestas curvas (b) é a imagem de u sob a deformação ϕ^u	13
4.1	Posições nominais dos pontos de controle envolvidos nas condições C^0 (dentro do retângulo preto) e C^1 (dentro do retângulo pontilhado) entre duas partes adjacentes de uma spline [19].	14
4.2	Pontos de controle de Bézier (b) de uma spline ϕ de grau 3 de $\mathbb{R}^2 \rightarrow \mathbb{R}^2$ que satisfazem restrições de continuidade C^0 , e suas posições nominais (a).	15
4.3	Pontos de controle de Bézier (b) de uma spline ϕ de grau 3 de $\mathbb{R}^2 \rightarrow \mathbb{R}^2$ que satisfazem as restrições de continuidade C^1 , e suas posições nominais (a).	17

4.4	A grade domínio para uma deformação spline de grau 5 (esquerda), e a grade deformada (direita), mostrando os pontos de controle e as condições de quadrilátero.	18
5.1	Um modelo biológico 3D M envolvido por uma grade domínio 3D P , e a correspondente grade domínio 2D T	20
5.2	As coordenadas baricêntricas $\alpha_0, \alpha_1, \beta_0, \beta_1$ e β_2 do ponto p do modelo 3D relacionado ao prisma U	20
5.3	Um modelo biológico 3D deformado $\psi(M)$ envolvido por uma grade deformada $\psi(P)$	21
5.4	Visão da grade deformada P no modo-xy, mostrando os pontos de controle da spline ϕ e suas relações de adjacência (linhas pontilhadas) na grade domínio.	22
5.5	Visão da grade deformada P no modo-z0 (a), mostrando os pontos de controle da spline σ_0 e no modo-z1 (b), mostrando os pontos de controle da spline σ_1	23
6.1	Rótulos dos triângulos que compartilham a aresta orientada e , e de seus vértices.	24
6.2	Rótulos dos pontos de controle que formam as condições de quadrilátero sobre a aresta compartilhada e para grau $d = 5$	25
6.3	Orientação consistente nas regiões ao redor dos vértices.	27
6.4	Rótulos dos pontos de controle que tornam a orientação dos quadriláteros consistente ao redor dos vértices.	27
6.5	Rotulações de dois quadriláteros que compartilham três pontos de controle.	28
6.6	Rotulações de dois quadriláteros que compartilham um membro medial com mesma orientação (a) e com orientações diferentes (b).	28
6.7	Rotulações de dois quadriláteros que compartilham um membro extremo.	29
7.1	Classificação dos pontos de controle em um triângulo de Bézier.	30
7.2	Quadrilátero $p.qr$ que usa um ponto do tipo <i>corner</i>	31
7.3	Quadriláteros $p.ql$, $p.qr$, $p.qp$ e $p.qn$ que compartilham um ponto do tipo <i>edge corner</i> . Em $p.ql$, $r_i^e = p$; em $p.qr$, $l_{i+1}^e = p$; em $p.qp$, $n_i^{e''} = p$ e em $p.qn$, $p_i^{e'} = p$	31
7.4	Quadriláteros $p.ql$ e $p.qr$ que compartilham um ponto do tipo <i>edge</i>	32
7.5	Quadriláteros $p.qp$ e $p.qn$ que compartilham um ponto do tipo <i>inner corner</i>	32
7.6	Quadrilátero $p.qp$ ou $p.qn$ que compartilham um ponto do tipo <i>inner edge</i>	32
7.7	Edição de um ponto de controle p	33

7.8	Ponto p do tipo <i>corner</i> editado pelo usuário, quadriláteros relevantes (cinzas), quadriláteros redundantes (pontilhados), pontos móveis (pretos) e pontos fixos (brancos).	34
7.9	Ponto p do tipo <i>edge corner</i> editado pelo usuário.	35
7.10	Ponto p do tipo <i>inner corner</i> editado pelo usuário.	35
7.11	Ponto p do tipo <i>edge</i> editado pelo usuário.	36
7.12	Ponto p do tipo <i>inner edge</i> editado pelo usuário.	36
8.1	Exemplo de edição do ponto de controle p de tipo <i>edge</i>	37
8.2	Exemplo de edição do ponto de controle p de tipo <i>inner corner</i>	39
8.3	Exemplo de edição do ponto de controle p de tipo <i>inner edge</i>	39
8.4	Exemplo de edição do ponto de controle p de tipo <i>corner</i>	42
8.5	Exemplo de edição do ponto de controle p de tipo <i>edge corner</i>	43
8.6	Exemplo de edição do ponto de controle p de tipo <i>edge</i>	43
9.1	Visão 2D (a) e visão 3D (b) do modelo 3D do <i>Caenorhabditis elegans</i> e sua grade de prismas.	44
9.2	Visão 2D (a) e visão 3D (b) do modelo 3D do <i>Dileptus anser</i> e sua grade de prismas.	45
9.3	Quatro imagens reais do <i>C. elegans</i> (esquerda) e visão 2D dos modelos deformados manualmente no modo-xy (direita).	46
9.4	Quatro imagens reais do <i>Dileptus</i> (esquerda) e visão 2D dos modelos deformados manualmente no modo-xy (direita).	47
9.5	Imagem real do <i>C. elegans</i> (a), modelo deformado manualmente no modo-xy, modo-z0 e modo-z1, visão 2D (b) e visões 3D do mesmo (c), (d) e (e).	48
A.1	Interface inicial do editor de deformação para modelos 3D.	51
A.2	Barra de controles.	52
A.3	Janela de edição no modo de visualização.	53
A.4	Janela de edição no modo de seleção, com modo-xy selecionado anteriormente.	53
A.5	Janela de edição no modo de seleção, com modo-z0 selecionado anteriormente.	54
A.6	Janela de edição no modo-xy de edição.	54
A.7	Janela de edição no modo-z0 de edição.	55
A.8	Janela de edição no modo-z1 de edição.	55

Capítulo 1

Introdução

1.1 Motivação e objetivos

Métodos de deformação são importantes em áreas como modelagem geométrica e animação computacional. Uma deformação suave interativa de complexidade arbitrária é necessária em muitas aplicações. Um exemplo é a modelagem de forma, crescimento, movimento e patologias de organismos microscópicos vivos ou células. Usamos a modelagem dessas deformações para ilustrar e validar este trabalho. Nosso objetivo é definir ferramentas matemáticas e de software para descrever tais deformações de forma amigável.

Geralmente, em um determinado estágio de sua vida, cada espécie de organismo tem uma morfologia bem definida, o que torna possível modelar sua forma básica. No entanto, muitas estruturas biológicas têm consistência elástica ou gelatinosa e podem sofrer deformações consideráveis devido ao movimento ou mesmo a patologias. Veja a Figura 1.1. Uma modelagem realística dessas deformações é crucial em tarefas como a geração de imagens sintéticas e vídeos, para estudo, ensino e reconhecimento automático de organismos em análises clínicas.



(a)



(b)

Figura 1.1: Imagens microscópicas do *Caenorhabditis elegans* (a) [28] e do *Dileptus anser* (b) [14] deformados.

Neste trabalho, consideramos que a forma básica do organismo é fornecida como uma malha triangular densa com milhares de triângulos. Veja a Figura 1.2.

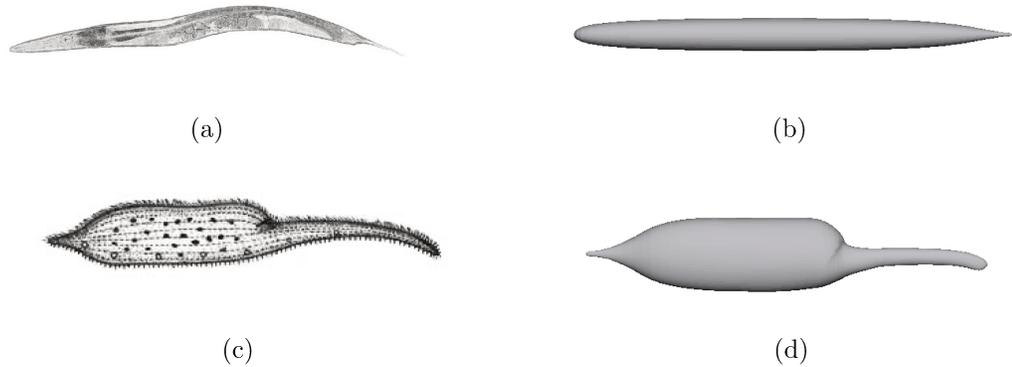


Figura 1.2: Morfologia do *Caenorhabditis elegans* (a) [7], *Dileptus anser* (c) [36] e os respectivos modelos 3D de suas formas básicas (b) e (d).

Para deformar o modelo, utilizamos uma deformação do espaço no qual ele está embutido. Esta abordagem é altamente interativa e intuitiva [17]. Geralmente, os métodos de deformação do espaço usam uma malha 3D grosseira, chamada *grade de controle* que envolve o modelo do organismo. A forma embutida pode ser modificada manipulando-se a grade de controle. Usando técnicas de interpolação, a deformação da grade de controle é estendida para todo o espaço dentro dela, e portanto para o modelo.

Muitas técnicas de deformação do espaço existentes são contínuas (C^0) mas não suaves (C^1), isto é, elas frequentemente introduzem quinas ou vincos no modelo inicial. Veja a Figura 1.3.

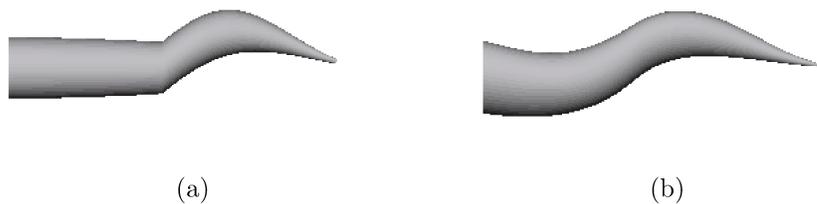


Figura 1.3: Uma comparação entre deformações do espaço com continuidade C^0 (a) e C^1 (b).

1.2 Contribuição

Neste trabalho, focamos em técnicas de modelagem de deformação que são C^1 e permitem uma edição conveniente da grade de controle pelo usuário.

Muitos métodos de deformação usam grades formadas por tetraedros; porém para deformações C^1 essas grades tem um grande número de parâmetros que devem ser ajustados, o que dificulta o trabalho de edição pelo usuário.

Para alcançar o objetivo de modelar deformações C^1 de forma amigável, propomos um subconjunto especial de deformações do espaço que podem ser descritas por uma grade mais simples formada por uma camada de prismas triangulares com faces superior e inferior curvas. A deformação resultante é a combinação de uma deformação 2D nas direções x e y com uma deformação 1D na direção z que depende de x e y . Podemos portanto dizer que é um método de deformação do espaço “2.5D”. Embora os modelos de micro-organismo considerados na aplicação sejam 3D, as deformações são essencialmente 2D, com poucas mudanças de profundidade; mesmo porque a terceira dimensão não pode ser percebida com facilidade através do microscópio.

Implementamos o método em um editor que torna possível definir e modificar a deformação de uma forma amigável usando o mouse. A escolha de células prismáticas reduz o número de pontos da grade de controle, tornando a deformação mais fácil de editar. Os resultados experimentais mostram que o método proposto é simples, efetivo e permite uma edição conveniente das deformações. Nosso editor assegura que a deformação do espaço obtida pela deformação da grade é C^1 .

1.3 Organização

Este trabalho está estruturado em dez capítulos e um apêndice. Uma visão geral do problema é apresentada no Capítulo 2, onde discutimos algumas técnicas de modelagem de deformação para modelos 3D, com foco em deformação do espaço e técnicas de interpolação. No Capítulo 3 descrevemos a fundamentação matemática sobre splines e mostramos como elas podem ser utilizadas na modelagem de deformação. No Capítulo 4 apresentamos os conceitos matemáticos e a representação geométrica das restrições de continuidade e suavidade das splines.

Abordamos, em seguida, a construção do método de deformação do espaço 2.5D para modelos 3D baseado na grade de controle formada por prismas. No Capítulo 5 temos uma visão geral desse método. As formas de representação e as estruturas de dados da triangulação, das funções spline e das restrições de continuidade são definidas no Capítulo 6. O método desenvolvido para edição da deformação de uma spline é apresentado no Capítulo 7 e a forma de atualização dessa spline após a edição, no Capítulo 8.

No Capítulo 9 apresentamos os experimentos. Por fim, as conclusões do trabalho e as possíveis pesquisas futuras são discutidas no Capítulo 10. No Apêndice A apresentamos a interface e as funções do editor implementado para testar o método que nós desenvolvemos.

Capítulo 2

Trabalhos Relacionados

Neste capítulo apresentamos uma revisão bibliográfica dos métodos de deformação de modelos 3D. Mais especificamente, abordagens que usam grades volumétricas como ferramenta de controle para deformar o espaço que envolve o modelo. Apresentamos também, algumas técnicas de interpolação usadas para transferir a deformação da grade para o modelo.

2.1 Métodos de deformação para modelos 3D

Os métodos de deformação podem ser classificados em deformações baseadas na física [33] ou na geometria. Na primeira, o objetivo é simular o comportamento físico de objetos que estão sob o efeito de forças internas e externas. Esses métodos são necessários na simulação de interações realísticas, porém não são adequados para aplicações totalmente interativas ou simulações em tempo-real devido ao alto custo computacional exigido. As técnicas geométricas são mais simples, mais rápidas e relativamente mais fáceis de implementar. Elas permitem manipulações arbitrárias na forma geométrica do objeto, sem levar em conta as leis da física. Dessa forma, o foco dessas técnicas está em manter a suavidade do modelo durante a deformação e permitir a reprodução de deformações reais (sem excluir deformações fisicamente impossíveis) [5].

As técnicas baseadas na geometria podem ser classificadas em métodos de deformação de superfície, que modificam a forma através da manipulação direta dos vértices do modelo; ou em métodos de deformação do espaço, que modificam a forma através da manipulação de pontos de controle que envolvem o modelo.

Em geral, os métodos de deformação de superfície são empregados quando a deformação deve preservar detalhes da superfície do modelo. Botsch e Sorkine, em [8], discutem técnicas lineares de deformação de superfície, que alcançam bons resultados para pequenas deformações. Entretanto, deformações em larga escala podem gerar ar-

tefatos indesejados devido a erros de linearização. Por essa razão, métodos não lineares tornaram-se importantes [42]. As técnicas de deformação de superfície fornecem um controle completo das deformações. Entretanto, para modelos formados por milhares de vértices, essas técnicas são impraticáveis devido ao grande número de pontos que precisam ser manipulados para obter deformações simples, e a dificuldade de manter a suavidade da superfície.

Nos métodos de deformação do espaço, o usuário manipula pontos de controle que representam o espaço que envolve o modelo, o qual é deformado através de uma técnica de interpolação adequada. Esses métodos podem ter substancialmente menos parâmetros de liberdade que o modelo em si, reduzindo bastante o trabalho de edição da deformação. Geralmente, são empregados em aplicações interativas onde o modelo a ser deformado possui milhares de vértices. Outra vantagem desse método é que são independentes da representação do modelo embutido.

2.1.1 Métodos de deformação do espaço

Na deformação do espaço, o usuário manipula um objeto geométrico de controle. De acordo com a dimensão desse objeto, Gain and Bechmann [17] classificam os métodos de deformação do espaço em:

- **controle por pontos (0D):** o usuário manipula pontos de controle posicionados livremente sobre a região que envolve o modelo. Cada ponto de controle possui uma região de influência dentro da qual o movimento do ponto causa deslocamentos dos pontos do modelo. Alguns métodos desta classe são: deformação baseada em restrições [3], deformação de forma livre manipulada diretamente [17], deformação de forma livre Dirichlet [17] e deformação radial simples [1, 2];
- **controle por curvas (1D):** o usuário manipula uma ou mais curvas ou eixos de controle que afetam a deformação do modelo local ou globalmente. São exemplos desses métodos: deformações globais regulares [17], deformação generalizada de Casteljaou [17], deformação de eixo [17], *Wires* [17], *Bender* [17] e *Skinning Frameworks* [4, 31];
- **controle por superfícies (2D):** o usuário manipula uma ou mais superfícies de controle que afetam a deformação do modelo local ou globalmente. São métodos de deformação controlados por superfície: retalhos paramétricos [12, 13], *Star-Convex* [10], malhas triangulares [24, 25] e ferramentas baseadas em campo vetorial [39, 40];

- **controle por malhas volumétricas (3D):** o usuário manipula uma grade grossa que envolve o modelo. Sederberg e Parry [35] introduziram as abordagens de deformação do espaço de modelos utilizando grades volumétricas formadas por hexaedros: a deformação de forma livre de Bézier (*Free form deformation (FFD)*). Em seguida vieram métodos como: FFD com controle local [27, 21] e FFD com grade de topologia arbitrária [6, 9, 23, 32, 43].

Neste trabalho, optamos por usar malhas de controle volumétricas. Aparentemente, elas são as mais usadas em aplicações que requerem deformações de complexidade arbitrária de modelos 3D.

A maioria dos métodos de deformação do espaço com grades volumétricas descritos na literatura usam malhas de controle hexaedrais [27, 35] ou tetraedrais [6, 23, 43]. Como observamos na Seção 1.2, optamos por usar uma grade formada por uma camada única de prismas triangulares de paredes verticais. Portanto nosso método pode ser visto como “2.5D” na classificação de Gain and Bechmann [17].

2.1.2 Técnicas de interpolação

As técnicas de interpolação são utilizadas para transferir a deformação da grade de controle para o modelo embutido no espaço deformado. Algumas dessas técnicas de interpolação podem preservar a continuidade e a suavidade do modelo 3D embutido.

Uma interpolação suave pode ser obtida, por exemplo, usando a técnica de coordenada de valor médio [22, 15, 16], porém existem exceções em que a suavidade não é mantida. Outra possibilidade é usar coordenadas de *Green* [30] que oferecem bons resultados de suavidade e preservação da forma original, entretanto, não proporciona controle local da deformação, ou seja, qualquer movimento na grade de controle se propaga por todo o modelo.

Uma técnica mais simples e largamente adotada em aplicações gráficas que proporcionam controle local são as funções spline, definidas no Capítulo 3. Os métodos de deformação do espaço com grades volumétricas hexaedrais e tetraedrais, descritos em [35] e [6], utilizam interpolação spline.

Entretanto, o requisito de continuidade C^1 força o uso de splines de grau elevado, que tem um número muito grande de parâmetros. Por exemplo, com uma malha de tetraedros, o grau de interpolação da spline deve ser no mínimo 5 para permitir edição local da grade. Para especificar cada tetraedro na malha, o usuário deve especificar a posição dos 56 pontos de controle ($56 \times 3 (x, y, z) = 168$ parâmetros reais) [6]. Com uma malha hexaedral pode-se ter splines C^1 de grau 3, onde cada célula requer 64 pontos de controle (192 parâmetros) [17].

O elevado número de pontos de controle dificulta muito a edição gráfica da deformação. É difícil identificar visivelmente e selecionar os pontos de controle internos de cada célula. Além disso, para garantir as restrições de continuidade C^1 , o software de edição pode ter que mover muitos pontos de controle automaticamente, causando confusão ao usuário durante a edição [6, 17].

Em comparação, o método que escolhemos exige apenas 42 pontos de controle, todos os pontos estão no exterior da grade de controle e suas coordenadas são restritas de tal modo que há apenas 84 parâmetros reais livres ($21 \times 2 (x, y) + 42 \times 1 (z)$). Além disso, as coordenadas podem ser editadas separadamente de modo que em cada momento o usuário só precisa editar no máximo 21 pontos por célula. Veja a Tabela 2.1.

	Grau	Nº pontos	Pontos internos	Parâmetros reais
Tetraedro	5	56	sim	$56 \times 3(x,y,z) = 168$
Hexaedro	3	64	sim	$64 \times 3(x,y,z) = 192$
Prisma	5	42	não	$21 \times 2(x,y) + 42 \times 1(z) = 84$

Tabela 2.1: Comparação entre células de malhas formadas por tetraedros, hexaedros e prismas.

Nossa abordagem para edição da deformação bidimensional tem semelhanças com a da tese de doutorado de Xu [41]. Em ambos os trabalhos, a edição de um ponto de controle de Bézier causa o ajuste automático de vários outros pontos próximos, a fim de preservar a suavidade da spline. Entretanto, Xu determina previamente o conjunto de pontos a ajustar para cada ponto editado, por uma análise exaustiva de casos. Na nossa implementação o conjunto de pontos a ajustar é determinado no momento da edição, por um algoritmo mais simples e mais genérico. Além disso, Xu sempre procura obter o menos conjunto possível de pontos a ajustar, que faz com que o ajuste seja único; enquanto que nosso algoritmo pode escolher um conjunto maior de pontos, e usa um critério de mínimos quadrados ponderado para fazer o ajuste. Por conta desta diferença, os ajustes feito pelo nosso algoritmo são mais simétricos e naturais dos que os obtidos por Xu.

Capítulo 3

Splines Polinomiais

Neste capítulo apresentamos a fundamentação matemática e a representação geométrica de splines polinomiais. Mostramos também como podem ser utilizadas na modelagem de deformação.

3.1 Triangulações

Uma triangulação T é um conjunto de triângulos contidos no plano \mathbb{R}^2 . A união de todos os triângulos de T é igual ao domínio Ω da triangulação. Neste trabalho, toda triangulação deve satisfazer as seguintes restrições:

- Os vértices de cada triângulo não podem ser colineares.
- Não deve existir intersecção entre os interiores de quaisquer dois triângulos de T .
- Entre dois triângulos, pode haver apenas uma aresta inteira ou um vértice em comum.
- O domínio Ω deve ser conexo.
- O número de vértices da fronteira é igual ao número de arestas da fronteira.

A Figura 3.1(a) mostra uma triangulação que respeita essas restrições. A última condição exclui triangulações cuja conectividade pode ser alterada removendo um único vértice. Veja a Figura 3.1(b).

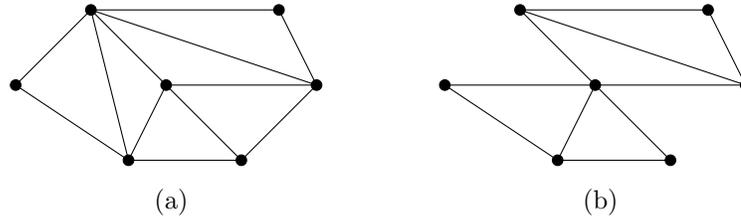


Figura 3.1: Uma triangulação válida (a) e uma triangulação inválida (b).

3.2 Funções polinomiais em \mathbb{R}^n

Seja d um inteiro não negativo e $x = (x_1, x_2, \dots, x_n) \in \mathbb{R}^n$. O espaço $P_{d,n}$ dos polinômios de grau $\leq d$ é o espaço de todas as funções da forma

$$p(x) = \sum_{0 \leq i_1 + i_2 + \dots + i_n \leq d} c_{i_1 i_2 \dots i_n} x_1^{i_1} x_2^{i_2} \dots x_n^{i_n}, \quad (3.1)$$

onde $c_{i_1 i_2 \dots i_n}$ são números reais e i, j, k são inteiros não negativos.

Um polinômio p do espaço $P_{d,n}$ é dito *homogêneo de grau d* se todos os monômios com coeficientes não nulos têm grau total d . Seja $H_{d,n}$ o conjunto destes polinômios e g uma função pertencente a $H_{d,n}$, então

$$g(p) = \sum_{i_1 + i_2 + \dots + i_n = d} c_{i_1 i_2 \dots i_n} p_1^{i_1} p_2^{i_2} \dots p_n^{i_n}. \quad (3.2)$$

O espaço $P_{d,n}$ é um espaço vetorial de dimensão $\binom{d+n}{n}$ e $H_{d,n}$ um subespaço vetorial de dimensão $\binom{d+n-1}{n-1}$ [19, 26].

3.3 Splines polinomiais

Uma *spline polinomial* é uma função definida por partes cujas partes são polinômios [11]. Mais precisamente, uma *spline sobre uma malha T em \mathbb{R}^n* é uma função $f : \mathbb{R}^n \rightarrow \mathbb{R}$ tal que a restrição f^u de f para cada parte u da malha é um polinômio nas coordenadas do argumento. Veja a Figura 3.2.

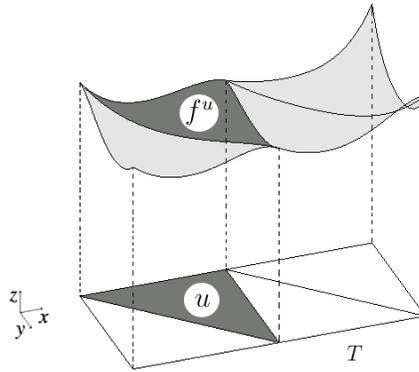


Figura 3.2: Uma função spline de \mathbb{R}^2 em \mathbb{R} .

Splines são amplamente usadas para modelar funções de complexidade ilimitada com controle local [11, 26].

3.3.1 Coordenadas baricêntricas

Um ponto no espaço pode ser representado através de uma combinação linear de outros pontos no mesmo espaço. Seja $u = (u_0, u_1, u_2)$ um triângulo em \mathbb{R}^2 , onde $u_i = (x_i, y_i)$. Então, qualquer ponto $p = (x, y) \in \mathbb{R}^2$ tem uma representação única na forma

$$p = \beta_0 u_0 + \beta_1 u_1 + \beta_2 u_2, \tag{3.3}$$

com $\beta_0 + \beta_1 + \beta_2 = 1$.

Os coeficientes β_0 , β_1 e β_2 são chamados de *coordenadas baricêntricas de p relativas ao triângulo u*. A posição do ponto $p \in \mathbb{R}^2$, com relação ao triângulo u , é determinada pelos sinais desses coeficientes [26]. Veja a Figura 3.3.

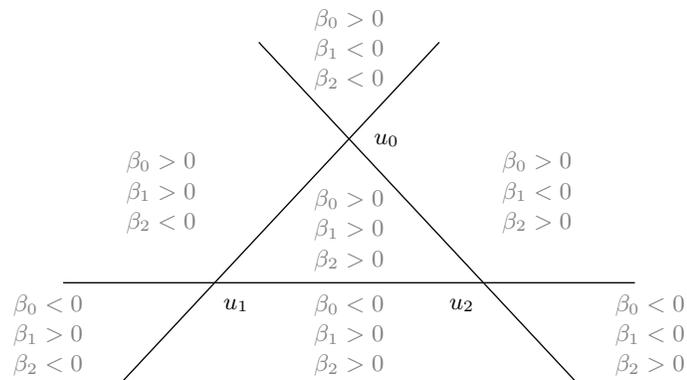


Figura 3.3: Sinais das coordenadas baricêntricas com relação ao triângulo u .

3.3.2 Representação Bernstein-Bézier para polinômios

Se $f : \mathbb{R}^2 \rightarrow \mathbb{R}$ é uma spline polinomial, então as células da malha T são triângulos. Logo cada polinômio f^u pode ser convenientemente expresso como uma combinação linear dos *polinômios de Bernstein-Bézier* do triângulo correspondente u [26].

Sejam p um ponto de \mathbb{R}^2 e β_0, β_1 e β_2 as coordenadas baricêntricas de p relativas aos vértices de um triângulo u . Sejam d, i, j, k inteiros não negativos tal que $i + j + k = d$. Então, um polinômio Bernstein-Bézier bidimensional de grau d é definido como

$$B_{ijk}(p) = \frac{d!}{i!j!k!} \beta_0^i \beta_1^j \beta_2^k. \quad (3.4)$$

Note que B_{ijk} é uma função polinomial homogênea com grau total d nas coordenadas baricêntricas β_0, β_1 e β_2 . O conjunto de todos os polinômios B_{ijk} com $i + j + k = d$ é uma base para os polinômios de grau d definido em \mathbb{R}^2 . Isto é, todo polinômio g de grau d definido em \mathbb{R}^2 pode ser escrito de forma única como

$$g(p) = \sum_{i+j+k=d} c_{ijk} B_{ijk}(p), \quad (3.5)$$

para todo $p \in \mathbb{R}^2$ onde os coeficientes c_{ijk} dependem do polinômio g e do triângulo u .

Os números c_{ijk} são chamados *coeficientes de Bézier do polinômio g* . Qualquer polinômio g na forma da Equação (3.5) pertence ao subespaço vetorial $H_{d,2}$.

3.3.3 Coeficientes de Bézier

Cada coeficiente c_{ijk} , na Equação (3.5), pode ser associado a uma posição nominal u_{ijk} relativa aos vértices u_0, u_1 e u_2 do triângulo u [26]. Veja a Figura 3.4.

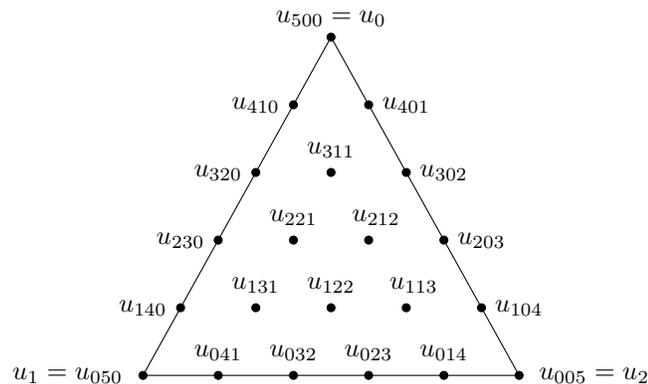


Figura 3.4: Posições nominais u_{ijk} dos coeficientes de Bézier c_{ijk} para uma parte f^u de grau 5.

Este triângulo é chamado *triângulo de Bézier*. Cada triângulo de grau d possui um conjunto $C_{d,u}$ com $(d+1)(d+2)/2$ posições nominais u_{ijk} , com coordenadas baricêntricas $(i/d, j/d, k/d)$, isto é

$$C_{d,u} = \{u_{ijk} | u_{ijk} = (i/d)u_0 + (j/d)u_1 + (k/d)u_2\}_{i+j+k=d}. \quad (3.6)$$

A união de triângulos de Bézier forma uma *superfície spline triangular*. O conjunto de posições nominais de uma superfície spline triangular é $C_{d,T}$, onde

$$C_{d,T} = \bigcup_{u \in T} C_{d,u}. \quad (3.7)$$

O conjunto $C_{d,T}$ não possui elementos repetidos, isto é, cada ponto de uma aresta compartilhada por dois triângulos de Bézier é incluído uma única vez no conjunto. Veja a Figura 3.5.

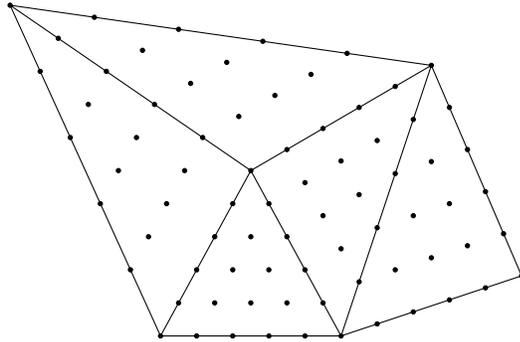


Figura 3.5: Superfície spline triangular de grau $d = 5$ em \mathbb{R}^2 mostrando o conjunto $C_{5,T}$ de posições nominais (pontos pretos).

3.4 Modelagem de deformação com splines

Uma deformação do espaço pode ser definida como uma função spline $\phi : \mathbb{R}^n \rightarrow \mathbb{R}^n$. Uma forma conveniente de modelar tais funções é escolher para cada coordenada de ϕ uma função spline ϕ_r , de tal forma que todas estas splines tenham o mesmo grau e são definidas na mesma malha T . A função ϕ deforma T , a *grade domínio*, em uma nova malha $\phi(T)$ com arestas curvas, a *grade deformada*, como mostra a Figura 3.6.

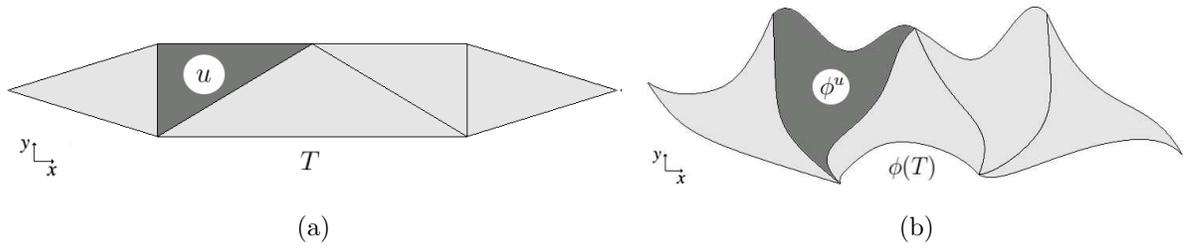


Figura 3.6: Uma deformação de \mathbb{R}^2 da grade domínio T (a) na grade deformada $\phi(T)$ (b).

A representação Bernstein-Bézier pode ser usada para descrever uma deformação $\phi : \mathbb{R}^n \rightarrow \mathbb{R}^n$. Seja u um triângulo de uma triangulação T e ϕ^u a parte de ϕ com domínio u . Para cada coordenada r (0 para x ou 1 para y), o coeficiente de Bézier $c_{ijk;r}^u$ de ϕ_r^u pode ser visto como coordenada r do ponto q_{ijk}^u , o *ponto de controle de Bézier* de ϕ^u com índices i, j e k . A Figura 3.7 mostra como uma função ϕ pode ser modificada movendo os pontos q_{ijk}^u .

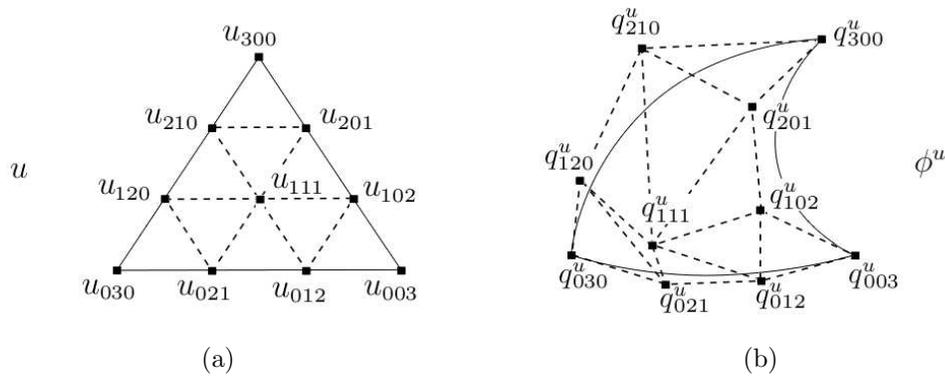


Figura 3.7: Pontos de controle de Bézier q_{ijk}^u (b) de um retalho ϕ^u de grau 3 de $\mathbb{R}^2 \rightarrow \mathbb{R}^2$ e suas posições nominais u_{ijk} no triângulo de domínio u (a). O triângulo de arestas curvas (b) é a imagem de u sob a deformação ϕ^u .

Note que os pontos de controle q_{ijk}^u são distintos de suas posições nominais u_{ijk} . Eles também são distintos da imagem $\phi^u(u_{ijk})$ das posições nominais, exceto nos vértices, isto é, $\phi^u(u_{d00}) = q_{d00}^u$, $\phi^u(u_{0d0}) = q_{0d0}^u$ e $\phi^u(u_{00d}) = q_{00d}^u$.

Capítulo 4

Restrições de Continuidade em Splines

Splines polinomiais podem estar sujeitas a restrições lineares adicionais, como as restrições de continuidade estabelecidas nas fronteiras entre as partes da spline. A continuidade da spline pode ser de qualquer ordem r (C^r). Para isso as derivadas até a ordem r das funções que se encontram devem ser contínuas [26].

É possível garantir que polinômios tenham continuidade C^1 ao longo de uma aresta da triangulação impondo condições a alguns coeficientes de Bézier, como mostra a Figura 4.1.

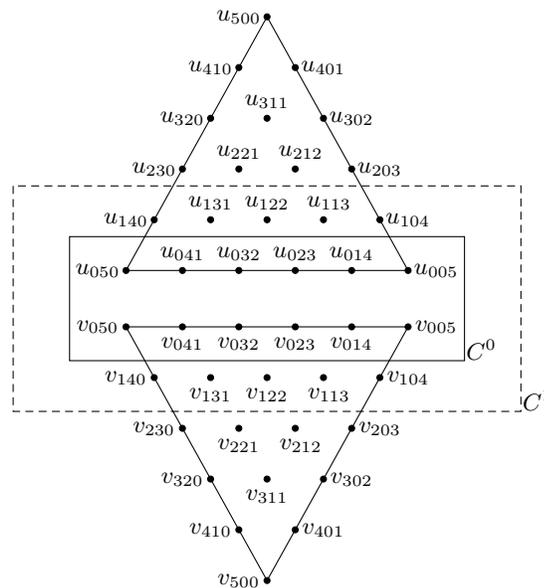


Figura 4.1: Posições nominais dos pontos de controle envolvidos nas condições C^0 (dentro do retângulo preto) e C^1 (dentro do retângulo pontilhado) entre duas partes adjacentes de uma spline [19].

4.1 Assegurando continuidade C^0

Uma função spline tem continuidade de ordem 0 (C^0) quando não existem descontinuidades através das fronteiras entre as partes. Para splines definidas sobre uma triangulação, a condição C^0 pode ser facilmente expressa em termos dos coeficientes de Bézier. Mais precisamente, seja $f : \mathbb{R}^2 \rightarrow \mathbb{R}$ uma função spline definida sobre uma triangulação T . Sejam u e v dois triângulos adjacentes de T com coeficientes de Bézier c_{ijk}^u e $c_{i'j'k'}^v$. A condição para f ser contínua através da aresta comum de u e v é que $c_{ijk}^u = c_{i'j'k'}^v$ para todo i, j, k, i', j', k' tal que as posições nominais coincidam, isto é, tal que $u_{ijk} = v_{i'j'k'}$.

O mesmo critério determina quando uma deformação $\phi : \mathbb{R}^2 \rightarrow \mathbb{R}^2$ é contínua de ordem 0. Ou seja, nós devemos ter $q_{ijk}^u = q_{i'j'k'}^v$ sempre que $u_{ijk} = v_{i'j'k'}$. Veja a Figura 4.2.

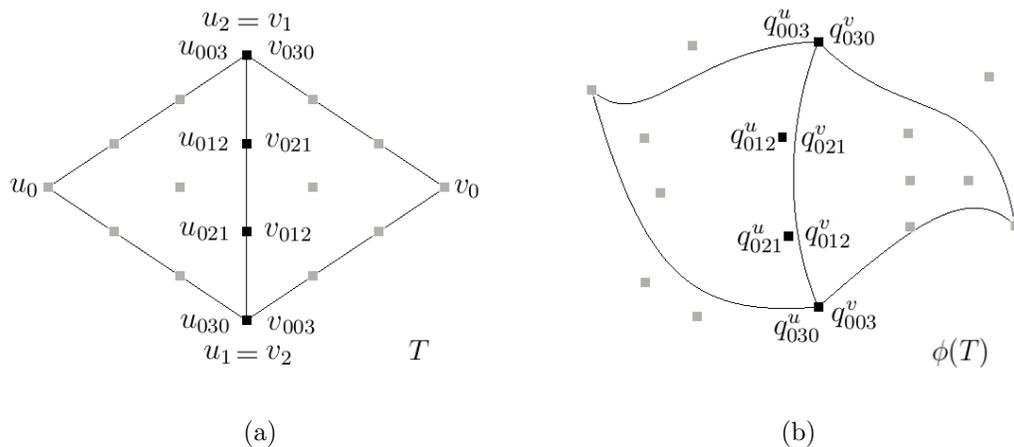


Figura 4.2: Pontos de controle de Bézier (b) de uma spline ϕ de grau 3 de $\mathbb{R}^2 \rightarrow \mathbb{R}^2$ que satisfazem restrições de continuidade C^0 , e suas posições nominais (a).

4.2 Assegurando continuidade C^1

A condição C^0 não garante a suavidade da função spline. Uma spline é suave quando tem pelo menos continuidade de ordem 1 (C^1) no encontro de suas partes, isto é, dados os triângulos u e v , as derivadas de primeira ordem dos polinômios correspondentes f^u e f^v são iguais nas fronteiras entre u e v . O Teorema 1 expressa esta condição em termos dos coeficientes de Bézier de f^u e f^v [26].

Teorema 1. *Sejam u e v dois triângulos adjacentes com vértices u_0, u_1, u_2 e v_0, v_1, v_2 , respectivamente, onde $u_1 = v_1$ e $u_2 = v_2$ pertencem à aresta compartilhada e. Sejam f^u e f^v polinômios de grau d com coeficientes de Bézier c_{ijk}^u e c_{ijk}^v relativos a u e v ,*

respectivamente, isto é

$$f^u(p) = \sum_{i+j+k=d} c_{ijk}^u B_{ijk}^u(p) \quad (4.1)$$

$$f^v(p) = \sum_{i+j+k=d} c_{ijk}^v B_{ijk}^v(p). \quad (4.2)$$

Então, f^u e f^v e todas suas derivadas parciais até a ordem r são unidos suavemente na aresta e , se e somente se,

$$c_{ijk}^v = \sum_{i'+j'+k'=i} c_{i',j',k+k'}^u B_{i'j'k'}^u(v_0), \quad (4.3)$$

para todo $i = 0, \dots, r$ e todo j, k tal que $j + k = d - i$.

Corolário 1. Em particular, os polinômios f^u e f^v do Teorema 1 são contínuos de primeira ordem ao longo da aresta e , se e somente se,

$$c_{0jk}^v = c_{0jk}^u, \quad (4.4)$$

para todo j, k tal que $j + k = d$ e

$$c_{1jk}^v = \beta_0 c_{1,j,k}^u + \beta_1 c_{0,j+1,k}^u + \beta_2 c_{0,j,k+1}^u, \quad (4.5)$$

para todo j, k tal que $j + k = d - 1$, onde β_0, β_1 e β_2 são coordenadas baricêntricas de v_0 relativas a u_0, u_1 e u_2 .

Suponha, por exemplo, dois polinômios de grau 5 relativos aos triângulos u e v . Então, as condições de continuidade de ordem zero são

$$\begin{aligned} c_{005}^u &= c_{005}^v \\ c_{014}^u &= c_{014}^v \\ c_{023}^u &= c_{023}^v \\ c_{032}^u &= c_{032}^v \\ c_{041}^u &= c_{041}^v \\ c_{050}^u &= c_{050}^v; \end{aligned}$$

e as condições adicionais para continuidade de ordem 1 são

$$\begin{aligned} c_{104}^u &= c_{104}^v \beta_0 + c_{014}^v \beta_1 + c_{005}^v \beta_2 \\ c_{113}^u &= c_{113}^v \beta_0 + c_{023}^v \beta_1 + c_{014}^v \beta_2 \\ c_{122}^u &= c_{122}^v \beta_0 + c_{032}^v \beta_1 + c_{023}^v \beta_2 \\ c_{131}^u &= c_{131}^v \beta_0 + c_{041}^v \beta_1 + c_{032}^v \beta_2 \\ c_{140}^u &= c_{140}^v \beta_0 + c_{050}^v \beta_1 + c_{041}^v \beta_2. \end{aligned}$$

Para que uma deformação $\phi(T) : \Omega \rightarrow \mathbb{R}^2$ definida por splines seja contínua de ordem 1, valem as mesmas condições vistas para funções splines. Ou seja, valem as Equações (4.4) e (4.5) com os pontos de controle q_{ijk}^u no lugar dos coeficientes c_{ijk}^u . Isto é, sejam u e v dois triângulos de T que compartilham os vértices $u_1 = v_1$ e $u_2 = v_2$. Então, a deformação ϕ é suave através da aresta compartilhada, se e somente se,

$$q_{0jk}^v = q_{0jk}^u \quad (4.6)$$

e

$$q_{1jk}^v = \beta_0 q_{1,j,k}^u + \beta_1 q_{0,j+1,k}^u + \beta_2 q_{0,j,k+1}^u, \quad (4.7)$$

onde $i, j, k, \beta_0, \beta_1$ e β_2 são como descritos acima.

Note que as coordenadas baricêntricas de v_{1jk} relativas a $u_{1,j,k}, u_{0,j+1,k}$ e $u_{0,j,k+1}$ são as mesmas que aquelas de v_0 relativas a u_0, u_1 e u_2 . Ou seja, estas quatro posições nominais formam um quadrilátero no domínio de ϕ que é similar ao quadrilátero v_0, u_0, u_1, u_2 . A Equação (4.7) diz que o quadrilátero correspondente formado pelos pontos de controle $q_{1jk}^v, q_{1jk}^u, q_{0,j+1,k}^u, q_{0,j,k+1}^u$ devem ser uma imagem afim daquele quadrilátero do domínio. Veja a Figura 4.3.

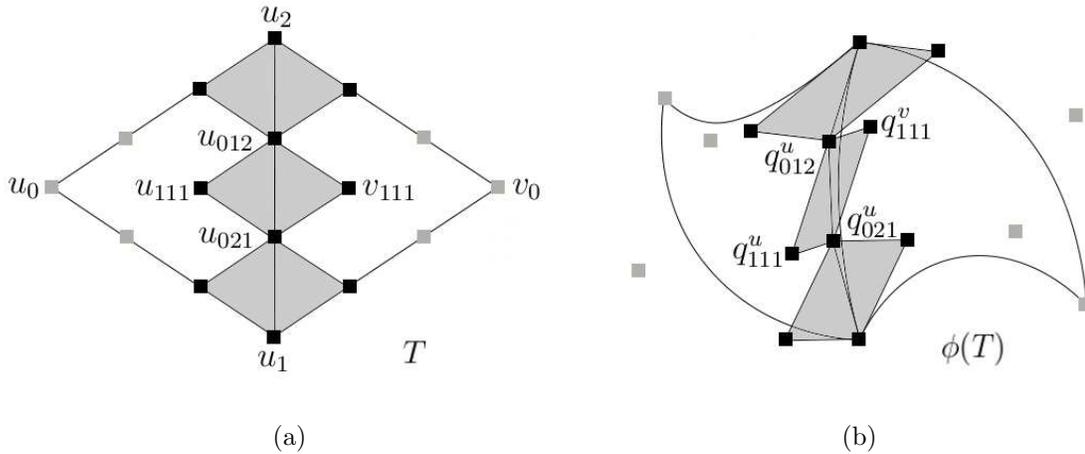


Figura 4.3: Pontos de controle de Bézier (b) de uma spline ϕ de grau 3 de $\mathbb{R}^2 \rightarrow \mathbb{R}^2$ que satisfazem as restrições de continuidade C^1 , e suas posições nominais (a).

O Teorema 1 e suas consequências também valem com índices permutados quando os dois triângulos de domínio compartilham outras arestas (por exemplo, quando $u_0 = v_1$ e $u_1 = v_2$, etc.).

Chamamos a Equação (4.7) de *condição de quadrilátero* naqueles quatro pontos de controle. Dizemos que os pontos q_{1jk}^v e q_{1jk}^u são os membros *extremos* da condição, e $q_{0,j+1,k}^u$ e $q_{0,j,k+1}^u$ são os membros *mediais*.

4.3 Assegurando controle local

A principal vantagem das splines sobre outras funções de métodos de aproximação é que elas permitem controle local. Ou seja, quando apenas um ponto de controle é modificado, a spline muda apenas dentro do triângulo de domínio correspondente e talvez outros poucos triângulos ao redor dele.

No entanto, frequentemente é preciso mudar dois ou mais pontos de controle ao mesmo tempo para manter as condições de continuidade C^0 e C^1 . Se o grau d é muito baixo, estas condições podem estar interconectadas de forma que as mudanças se propaguem de triângulo a triângulo sobre toda a grade de domínio, assim o controle local não é possível.

Em particular, para grades triangulares em \mathbb{R}^2 , pode-se facilmente obter controle local e continuidade C^1 com splines de grau $d \geq 5$. Para $d \leq 4$ estas características não podem ser obtidas simultaneamente [26].

Portanto, para a aplicação deste trabalho, que necessita de controle local, são usadas geralmente splines polinomiais de grau pelo menos 5. Assim, cada triângulo da spline tem pelo menos 21 pontos de controle. Veja a Figura 4.4.

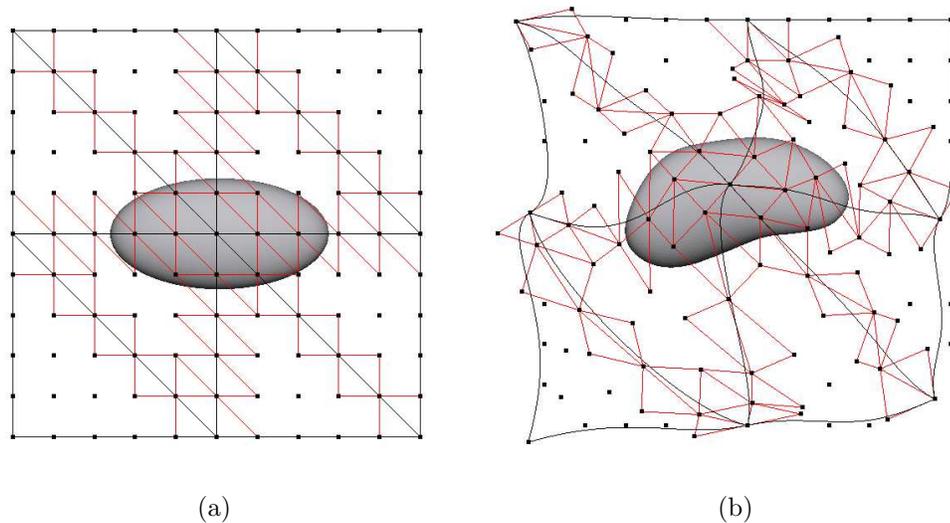


Figura 4.4: A grade domínio para uma deformação spline de grau 5 (esquerda), e a grade deformada (direita), mostrando os pontos de controle e as condições de quadrilátero.

Capítulo 5

Método de Deformação do Espaço 2.5D

Neste capítulo descrevemos nosso método de deformação do espaço 2.5D no contexto de edição interativa. Apresentamos a sequência de passos realizados pelo usuário para efetuar uma deformação. Isso envolve o carregamento de um modelo 3D, a construção da grade domínio, o relacionamento entre o modelo e a grade, a deformação da grade domínio através da deformação das splines definidas com restrições de suavidade.

5.1 Modelo 3D e construção da grade domínio

Inicialmente, o usuário escolhe um arquivo de malha triangular e, a partir deste arquivo, o programa carrega o modelo tridimensional M a ser deformado. Então, o usuário escolhe um grau $d \geq 5$ e define uma grade domínio P em \mathbb{R}^3 que envolve M . A grade domínio consiste de uma coleção de prismas em \mathbb{R}^3 , cujas projeções no plano xy são uma triangulação T de \mathbb{R}^2 , e a projeção no eixo z é um intervalo $[a, b]$, como mostra a Figura 5.1.

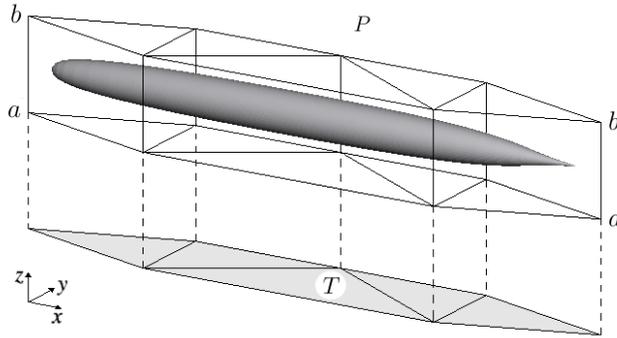


Figura 5.1: Um modelo biológico 3D M envolvido por uma grade domínio 3D P , e a correspondente grade domínio 2D T .

A grade domínio P é criada a partir da definição, pelo usuário, da triangulação T e do intervalo $[a, b]$.

5.1.1 Definição das coordenadas baricêntricas

Com a grade domínio P construída, o programa encontra o prisma U de P que contém cada vértice p do modelo, e computa as coordenadas baricêntricas β_0 , β_1 e β_2 de p com relação ao triângulo u da triangulação T que corresponde ao prisma U . O programa também computa a posição vertical de p relativa às duas faces triangulares u^0 e u^1 de U , ou seja, os dois números $\alpha_0 = (b - z)/(b - a)$ e $\alpha_1 = (z - a)/(b - a)$, como mostra a Figura 5.2.

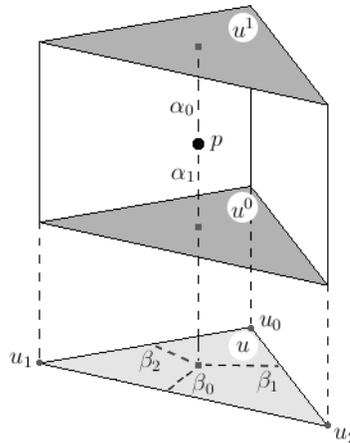


Figura 5.2: As coordenadas baricêntricas α_0 , α_1 , β_0 , β_1 e β_2 do ponto p do modelo 3D relacionado ao prisma U .

5.2 Deformação da grade domínio

A grade deformada $\psi(P)$ consiste de outra coleção de prismas com paredes verticais, cujas faces superior e inferior são triângulos curvos. Estes triângulos curvos de grau escolhido d constituem duas superfícies spline, as quais são as superfícies superior e inferior da grade deformada. Ambas superfícies são baseadas na mesma triangulação domínio T em \mathbb{R}^2 , como mostra a Figura 5.3.

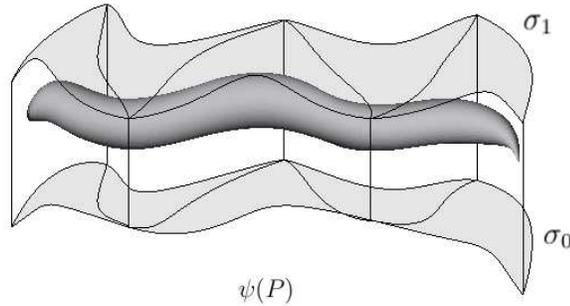


Figura 5.3: Um modelo biológico 3D deformado $\psi(M)$ envolvido por uma grade deformada $\psi(P)$.

A deformação ψ do espaço é uma função do interior da grade domínio P para o interior da grade deformada $\psi(P)$. O modelo deformado $\psi(M)$ é outra malha triangular densa com a mesma topologia de M obtida pelo mapeamento de todos vértices de M através da função ψ .

Uma vez que as paredes de ψ são sempre verticais, ψ pode ser separada em uma deformação 2D, $\phi : T \rightarrow \mathbb{R}^2$, e duas funções spline $\sigma_0 : T \rightarrow \mathbb{R}$ e $\sigma_1 : T \rightarrow \mathbb{R}$. Ou seja,

$$\psi(p) = (\psi(p).x; \psi(p).y; \psi(p).z), \quad (5.1)$$

onde $p = (x, y, z) \in \mathbb{R}^3$ e

$$\begin{aligned} \psi(x, y, z).x &= \phi(x, y).x \\ \psi(x, y, z).y &= \phi(x, y).y \\ \psi(x, y, z).z &= \alpha_0(x, y)\sigma_0(x, y) + \alpha_1(x, y)\sigma_1(x, y). \end{aligned}$$

Note que, para cada posição p dentro de P , a coordenada z do ponto $\psi(p)$ varia entre $\sigma_0(x, y)$ e $\sigma_1(x, y)$.

5.2.1 Deformação de uma spline

As splines σ_0 , σ_1 e ϕ , descritas na Seção 5.2, são definidas por seus pontos de controle de Bézier. O usuário pode modificar a deformação movendo cada ponto de controle com o mouse. Nesta fase, o usuário pode alterar entre três modos de edição: *modo-xy*, *modo-z0* e *modo-z1*.

No modo-xy o usuário tem uma visão do topo da grade deformada, isto é, uma visão da triangulação $\phi(T)$ e pode apenas modificar as coordenadas x e y de cada ponto de controle q_{ijk}^u de ϕ .

Cada triângulo tem $(d+1)(d+2)/2$ pontos de controle; exceto dois ou mais pontos de controle, com as mesmas posições nominais, ao longo das arestas compartilhadas, os quais são apresentados e editados como um ponto simples, como mostra a Figura 5.4.

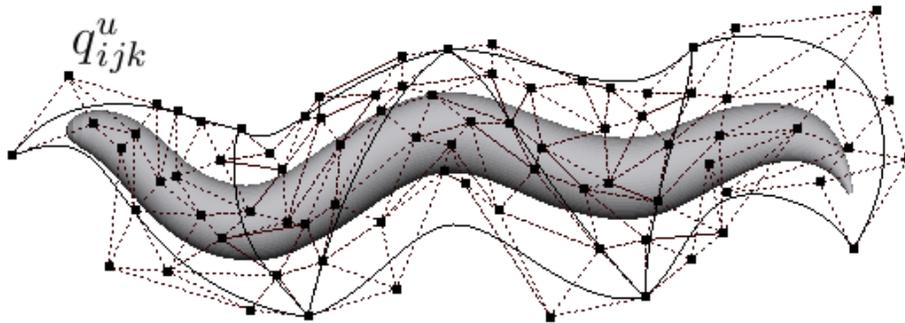


Figura 5.4: Visão da grade deformada P no modo-xy, mostrando os pontos de controle da spline ϕ e suas relações de adjacência (linhas pontilhadas) na grade domínio.

Quando a edição ocorre no modo-z0 ou no modo-z1, o usuário pode editar os coeficientes $c_{ijk;r}^u$ da função spline σ_0 ou σ_1 , respectivamente. Nesses modos, o usuário tem uma visão oblíqua da grade deformada $\psi(P)$. Para cada triângulo $u \in T$ e cada conjunto de índices i , j e k com $i + j + k = d$ existem dois coeficientes: $c_{0;ijk}^u$ para a superfície inferior e $c_{1;ijk}^u$ para a superfície superior. Portanto, existem $2(d+1)(d+2)/2 = (d+1)(d+2)$ pontos de controle para cada prisma de P mas apenas a coordenada $z0$ ou $z1$ pode ser alterada pelo usuário. Cada uma das duas splines σ_0 e σ_1 é editada independentemente, como mostra a Figura 5.5.

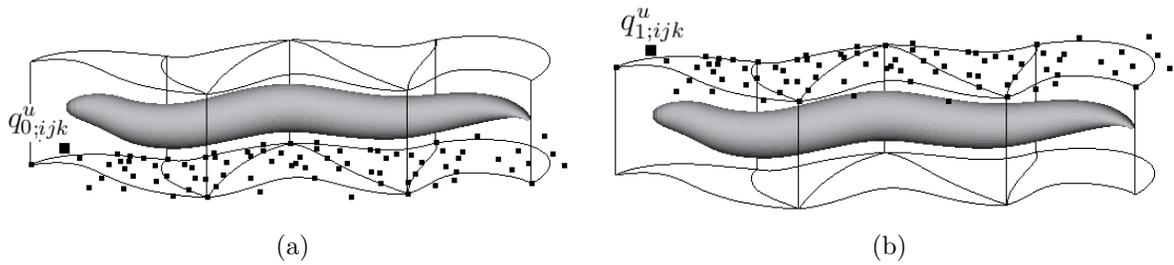


Figura 5.5: Visão da grade deformada P no modo-z0 (a), mostrando os pontos de controle da spline σ_0 e no modo-z1 (b), mostrando os pontos de controle da spline σ_1 .

5.2.2 Assegurando a continuidade da spline

Como observado no Seção 4.1, a condição de continuidade C^0 é assegurada, pois apenas um ponto de controle é disponibilizado para duas ou mais posições nominais que coincidem, ou seja, posições nominais que ficam sobre arestas ou vértices compartilhados por dois ou mais triângulos da triangulação T .

Da mesma forma, como explicado na Seção 4.2, a restrição de continuidade C^1 é representada por um número de condições de quadrilátero (que o programa pode mostrar opcionalmente, como na Figura 4.4). Para assegurar a continuidade C^1 , sempre que o usuário modifica um ponto de controle, o programa determina um ou mais pontos de controle adicionais que devem ser modificados para preservar as condições de quadrilátero, e automaticamente aplica a eles os ajustes necessários, conforme descrito no Capítulo 7 e 8.

Esta computação é feita independentemente para cada spline de acordo com o modo de edição escolhido pelo usuário, ou seja, no modo-xy é repetido para ambas splines, enquanto no modo-z0 isso afeta apenas a spline σ_0 e no modo-z1 apenas a spline σ_1 .

Capítulo 6

Representação das Splines

Neste capítulo definimos a forma de representação e a estrutura de dados para uma triangulação, para uma spline e para suas restrições de continuidade. Essa definição é importante para assegurar a consistência dos dados armazenados na estrutura, os quais servem como base para o algoritmo de edição da spline.

6.1 Notações adicionais

6.1.1 Rotulação e orientação das arestas

Dada uma triangulação T , para cada aresta orientada e denotamos por l^e e r^e a origem e o destino da mesma, de modo que podemos escrever $e = (l^e, r^e)$. Além disso, usamos as notações u^e e v^e , para os dois triângulos adjacentes à aresta e , à esquerda e à direita, se existem. Usamos também a notação p^e e n^e para rotular os outros vértices desses dois triângulos, como mostra a Figura 6.1.

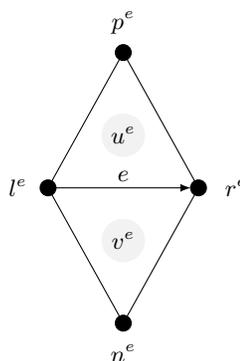


Figura 6.1: Rótulos dos triângulos que compartilham a aresta orientada e , e de seus vértices.

Observe que quando consideramos a mesma aresta no sentido oposto, os triângulos u^e e v^e trocam de posição, o rótulo l^e troca com r^e e p^e troca com n^e .

6.1.2 Rotulação e orientação das condições de quadrilátero

Para cada aresta orientada e existem d condições de quadrilátero, numeradas de 0 a $d - 1$ no sentido da aresta. Para cada uma dessas condições de quadrilátero, denotamos por l_i^e , r_i^e , p_i^e e n_i^e os quatro pontos de controle daquela condição, como mostrado na Figura 6.2. Dizemos que l_i^e e r_i^e são membros mediais *esquerdo* e *direito*; e p_i^e e n_i^e são membros extremos *anterior* e *posterior* das condições de quadrilátero i , respectivamente.

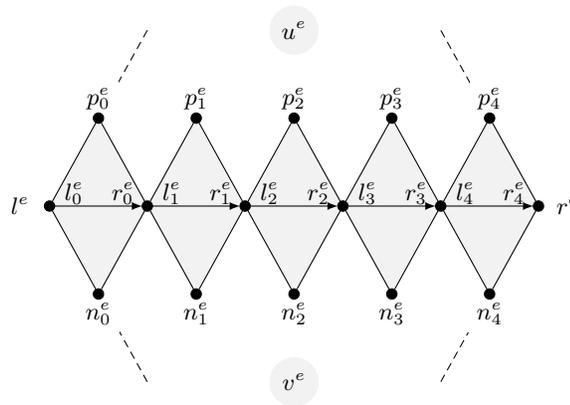


Figura 6.2: Rótulos dos pontos de controle que formam as condições de quadrilátero sobre a aresta compartilhada e para grau $d = 5$.

Existem duas rotulações possíveis para os pontos de uma condição de quadrilátero, em relação à orientação da aresta e . Se e' é a aresta e tomada no sentido inverso, então temos que Q_i^e coincide com $Q_{d-i-1}^{e'}$, mas $p_i^e = n_{d-i-1}^{e'}$ e $l_i^e = r_{d-i-1}^{e'}$.

6.2 Estruturas de dados

6.2.1 Representação de uma triangulação

Uma triangulação T é representada no nosso programa por uma lista de seus vértices, uma lista de suas arestas e uma lista de suas faces. Para cada vértice, há um registro do tipo `Point`, para cada aresta há um registro do tipo `Edge` e para cada face há um registro do tipo `Face`.

- **Point:** possui as coordenadas x e y na triangulação T (imutáveis) e as coordenadas correntes x , y , $z0$ e $z1$ no \mathbb{R}^3 dos dois pontos correspondentes da grade deformada P .

- **Edge:** possui os ponteiros `l` e `r`, para o vértice inicial e o vértice final da aresta, respectivamente. A orientação da aresta é escolhida arbitrariamente.
- **Face:** possui os ponteiros `p0`, `p1` e `p2` para seus vértices e `e0`, `e1` e `e2` para suas arestas. Os vértices e arestas são numerados no sentido anti-horário, a partir de um vértice arbitrário e p_i é a origem de e_i para cada i .

6.2.2 Representação de uma spline

Para definir uma spline sobre a triangulação T , o programa utiliza três tipos adicionais de registro. Para cada ponto de controle, há um registro do tipo `ControlPoint`. Para cada aresta compartilhada de T , há um registro do tipo `SharedEdge` e dois registros do tipo `BezierTriangle`.

- **ControlPoint:** possui um ponteiro para um registro do tipo `Point` e um inteiro `type` que indica sua posição no triângulo de Bézier.
- **BezierTriangle:** possui um ponteiro `f`, para a face da triangulação a que se refere, e um vetor `Cp` de ponteiros, para seus $(d+1)(d+2)/2$ pontos de controle q_{ijk}^f . Esses pontos são armazenados em ordem lexicográfica dos índices ijk . Note que dois ou mais pontos de controle que devem ser identificados para obter continuidade C^1 são representados por um único registro de tipo `ControlPoint`. Em particular, os ponteiros q_{d00}^f , q_{0d0}^f e q_{00d}^f apontam para registros `ControlPoint` que apontam para os registros `Point` que são os vértices da face.
- **SharedEdge:** possui um ponteiro `e` para o registro `Edge` correspondente, mais dois ponteiros `p` e `n`, para os outros dois vértices dos triângulos que compartilham a aresta, conforme a Figura 6.1. Possui também ponteiros para os dois registros `BezierTriangle` `u` e `v` e um vetor `Qc[0...d-1]` de ponteiros para os d registros do tipo `Quadrilateral` que representam as condições de continuidade C^1 associadas a essa aresta. Veja a Seção 6.2.3.

6.2.3 Representação das restrições C^1

Cada condição de quadrilátero é representada por um registro de seguinte tipo:

- **Quadrilateral:** possui quatro ponteiros `p`, `l`, `n` e `r`, para os quatro pontos de controle que entram na condição (Veja a Seção 6.1.2). Possui também um inteiro `orientation` que indica a orientação do quadrilátero com relação a orientação da aresta compartilhada (0 - mesma orientação; 1 - orientação oposta).

6.2.4 Consistência da orientação ao redor de um vértice

Na estrutura de dados, armazenamos cada condição de quadrilátero apenas uma vez; ou seja, armazenamos no campo $Qc[i]$ de cada aresta e o quadrilátero Q_i^e ou o quadrilátero $Q_{d-i-1}^{e'}$ onde e' é a aresta e na orientação oposta.

A razão para armazenar quadriláteros com orientação contrária à aresta e é que precisamos garantir que os quadriláteros que cercam um vértice tenham orientações consistentes entre si. Veja a Figura 6.3.

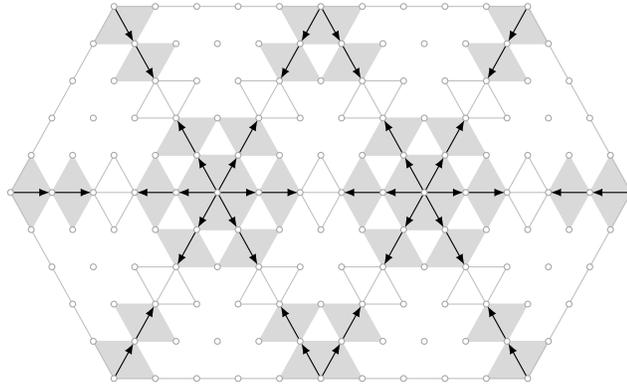


Figura 6.3: Orientação consistente nas regiões ao redor dos vértices.

Para tanto, devemos escolher para $Qc[i]$, a orientação coerente com a aresta e , ou seja, Q_i^e , quando $i = 0$ ou $i = 1$; e a orientação oposta, ou seja, $Q_i^{e'}$, quando $i = d - 1$ ou $i = d - 2$. Para os demais quadriláteros ($2 \leq i \leq d - 3$), podemos escolher qualquer uma das duas orientações. Nós escolhemos a mesma orientação da aresta e . Veja a Figura 6.4.

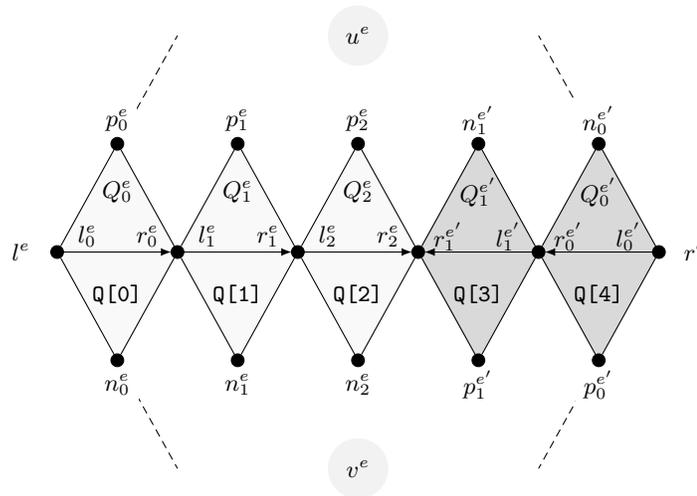


Figura 6.4: Rótulos dos pontos de controle que tornam a orientação dos quadriláteros consistente ao redor dos vértices.

6.2.5 Sobreposição de quadriláteros

Como consequência das escolhas acima, há restrições sobre a maneira em que dois ou mais quadriláteros podem se encaixar uns nos outros.

Para quaisquer duas condições de quadrilátero $Q_0^{e'}$ e $Q_0^{e''}$ armazenadas na estrutura, que compartilham três pontos de controle, um dos quais é um vértice de T , esta convenção implica que e' e e'' saem desse vértice, e portanto valem as seguintes identidades: $l_0^{e'} = l_0^{e''}$; $p_0^{e'} = r_0^{e''}$; e $r_0^{e'} = n_0^{e''}$, como mostra a Figura 6.5.

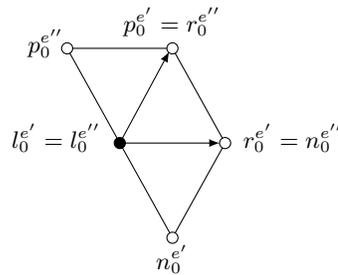


Figura 6.5: Rotulações de dois quadriláteros que compartilham três pontos de controle.

Duas condições de quadrilátero $Q_c[i]$ e $Q_c[i+1]$ armazenadas para uma aresta compartilhada e , que compartilham um ponto de controle medial, podem se enquadrar em dois casos. No primeiro caso, os dois quadriláteros têm a mesma orientação, ou seja, são $Q_i^{e''}$ e $Q_{i+1}^{e''}$ onde e'' pode ser a aresta e ou a aresta oposta e' , e satisfazem a restrição $r_i^{e''} = l_{i+1}^{e''}$. Esse caso é obrigatório se um dos pontos mediais é um vértice de T , o qual deve ser $l_0^{e''}$. Veja a Figura 6.6(a). No outro caso, os quadriláteros $Q_c[i]$ e $Q_c[i+1]$ têm orientações diferentes de modo que o primeiro concorda com e e o segundo concorda com e' , ou seja, são $Q_i^{e''}$ e $Q_{i+1}^{e'}$ e satisfazem a restrição $r_i^{e''} = r_{i+1}^{e'}$. Veja a Figura 6.6(b).

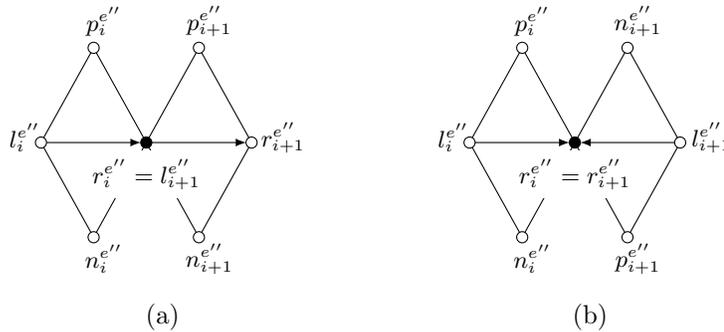


Figura 6.6: Rotulações de dois quadriláteros que compartilham um membro medial com mesma orientação (a) e com orientações diferentes (b).

Finalmente, quaisquer duas condições de quadrilátero $Q_1^{e'}$ e $Q_1^{e''}$ que compartilham um ponto de controle extremo devem estar orientados de modo a satisfazer a restrição $n_1^{e'} = p_1^{e''}$, como mostra a Figura 6.7.

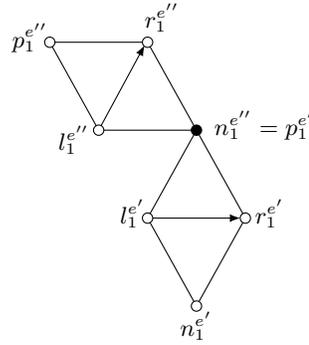


Figura 6.7: Rotulações de dois quadriláteros que compartilham um membro extremo.

Capítulo 7

Edição dos Pontos de Controle

Neste capítulo, discutiremos particularidades do método desenvolvido para manter a continuidade C^1 de uma spline triangular, durante uma deformação. Esse processo é dividido em três etapas: classificação dos pontos de controle, definição da vizinhança de acordo com o tipo de cada ponto e edição de um ponto de controle, a qual envolve a seleção das condições de continuidade e dos pontos de controle que devem ser atualizados quando a posição de um determinado ponto é modificada pelo usuário.

7.1 Classificação dos pontos de controle

Durante a criação de cada ponto de controle q_{ijk}^u , ele é classificado em seis tipos de acordo com sua posição nominal u_{ijk} no triângulo u , como mostrado na Figura 7.1.

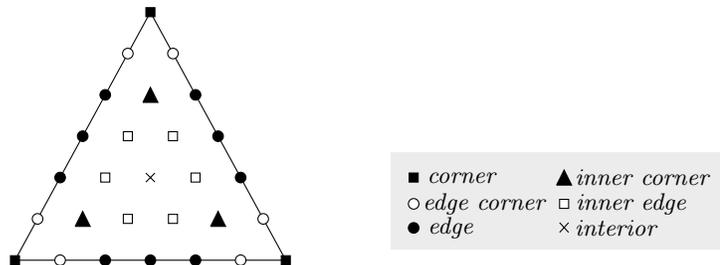


Figura 7.1: Classificação dos pontos de controle em um triângulo de Bézier.

São definidos os seguintes tipos de ponto de controle:

- *0 - corner*: u_{ijk} , tal que $i = d$, $j = d$ ou $k = d$, ou seja, o ponto é um vértice da grade domínio;
- *1 - edge corner*: u_{ijk} , tal que $i = 0$ e ($j = 1$ ou $k = 1$), $j = 0$ e ($i = 1$ ou $k = 1$) ou $k = 0$ e ($i = 1$ ou $j = 1$);

- 2 - *edge*: u_{ijk} , tal que $i = 0$, $j = 0$ ou $k = 0$, ou seja, o ponto é definido sobre uma aresta porém não é vértice;
- 3 - *inner corner*: u_{ijk} , tal que $i = j = 1$, $i = k = 1$ ou $j = k = 1$;
- 4 - *inner edge*: u_{ijk} , tal que $i = 1$, $j = 1$ ou $k = 1$, e não é nenhum dos anteriores;
- 5 - *interior*: u_{ijk} , tal que $i \geq 2$, $j \geq 2$ e $k \geq 2$.

7.2 Quadriláteros que usam um ponto de controle

Um ponto de controle p pode pertencer a uma ou mais condições de quadrilátero, dependendo de seu tipo. Para poder encontrar essas condições, o programa acrescenta ao registro `ControlPoint` de p quatro ponteiros `p.qr`, `p ql`, `p.qp` e `p.qn` para registros de tipo `Quadrilateral`. Conforme o tipo do ponto, apenas alguns desses ponteiros são usados e os demais ficam nulos.

Se p for do tipo *corner*, p é membro medial esquerdo (l_0^e) de uma ou mais condições de quadrilátero. Neste caso, o programa usa um único ponteiro do registro de p , o `p.qr`, para apontar um desses quadriláteros. Veja a Figura 7.2.

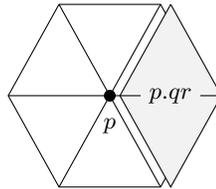


Figura 7.2: Quadrilátero `p.qr` que usa um ponto do tipo *corner*.

Se p for do tipo *edge corner*, p pode ser membro medial esquerdo (l_{i+1}^e) ou direito (r_i^e) de até duas condições de quadrilátero, e ainda pode ser membro extremo anterior ($p_i^{e'}$) ou posterior ($n_i^{e''}$) de até duas condições de quadrilátero. Neste caso, o programa usa os quatro ponteiros, `p ql`, `p.qr`, `p.qp` e `p.qn`, para apontar esses quadriláteros. Veja a Figura 7.3.

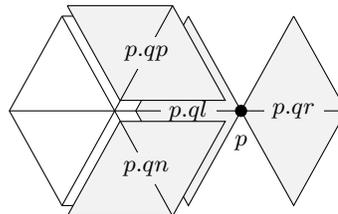


Figura 7.3: Quadriláteros `p ql`, `p.qr`, `p.qp` e `p.qn` que compartilham um ponto do tipo *edge corner*. Em `p ql`, $r_i^e = p$; em `p.qr`, $l_{i+1}^e = p$; em `p.qp`, $n_i^{e''} = p$ e em `p.qn`, $p_i^{e'} = p$.

Se p for do tipo *edge*, p pode ser membro medial direito (r_i^e) e esquerdo (l_{i+1}^e) de duas condições de quadrilátero. Neste caso, o programa usa os dois ponteiros, $p.ql$ e $p.qr$, para indicar esses dois quadriláteros. Veja a Figura 7.4. Note que $p.ql$ aponta para o quadrilátero à esquerda de p , ou seja, $p = r_i^e$, e que $p.qr$ aponta para o quadrilátero à direita de p , ou seja, $p = l_{i+1}^e$. Exceto para o caso em que $p.qr$ tem orientação oposta. Neste caso, $p = r_{i+1}^e$.

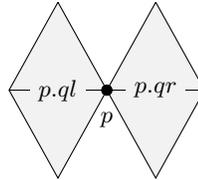


Figura 7.4: Quadriláteros $p.ql$ e $p.qr$ que compartilham um ponto do tipo *edge*.

Se p for do tipo *inner corner*, p é membro extremo anterior ($p_i^{e'}$) ou posterior ($n_i^{e''}$) de uma ou duas condições de quadrilátero. Neste caso, o programa usa os dois ponteiros, $p.qp$ e $p.qn$, para apontar esses quadriláteros. Veja a Figura 7.5. Note que em $p.qp$, $n_i^{e''} = p$ e em $p.qn$, $p_i^{e'} = p$.

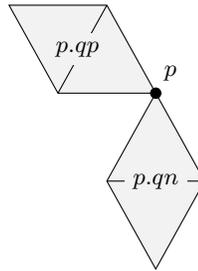


Figura 7.5: Quadriláteros $p.qp$ e $p.qn$ que compartilham um ponto do tipo *inner corner*.

Se p for do tipo *inner edge*, p é membro extremo anterior (p_i^e) ou posterior (n_i^e) de apenas uma condição de quadrilátero. Neste caso, o programa usa apenas um dos dois ponteiros de p , $p.qp$ ou $p.qn$. Veja a Figura 7.6. Note que em $p.qp$, $n_i^e = p$ e em $p.qn$, $p_i^e = p$.

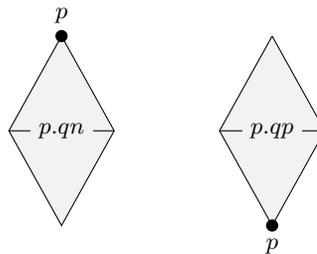


Figura 7.6: Quadrilátero $p.qp$ ou $p.qn$ que compartilham um ponto do tipo *inner edge*.

7.3 Algoritmo de seleção automática dos pontos móveis

A edição de um ponto de controle geralmente requer alterações em outros pontos para preservar as condições de continuidade envolvidas na edição. Veja a Figura 7.7.

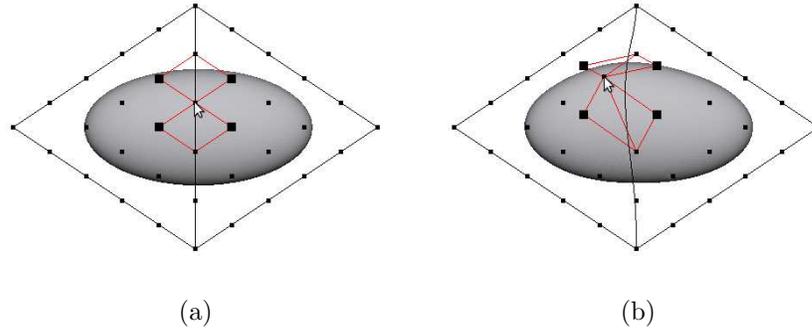


Figura 7.7: Edição de um ponto de controle p .

Quando o usuário modifica a posição de um ponto p , o programa encontra todos os quadriláteros que envolvem p e seleciona, entre os pontos desses quadriláteros, um subconjunto de *pontos móveis* que devem ser atualizados. Se um desses pontos móveis entra em um quadrilátero adicional, o processo é iterado até obter um conjunto de quadriláteros e pontos móveis que satisfaz as seguintes condições:

- Todo quadrilátero que envolve p está incluído;
- É possível satisfazer todos os quadriláteros alterando apenas os pontos móveis.

Nas condições mais favoráveis, o subconjunto de pontos móveis incluirá apenas pontos do próprio triângulo que contém p . Isto ocorre quando p é do tipo *inner corner*. Nos outros casos, podem ser incluídos pontos de alguns triângulos adjacentes àquele que contém o ponto editado.

O método implementado para seleção automática dos pontos móveis baseia-se na estrutura de vizinhança descrita na Seção 7.2, ao invés de utilizar diretamente as informações padrões de vizinhança entre os retalhos da spline. Este método é descrito de forma mais detalhada nas seções seguintes.

7.3.1 Edição de um ponto de controle *corner*

Quando um ponto p do tipo *corner* é editado pelo usuário, o programa seleciona inicialmente os quadriláteros que têm este ponto como membro medial e os quadriláteros vizinhos a eles. Veja a Figura 7.8. No caso de se formar um ciclo ao redor do vértice, dois

dos quadriláteros (tracejados) são excluídos pois são redundantes, ou seja, suas equações são combinações lineares das demais.

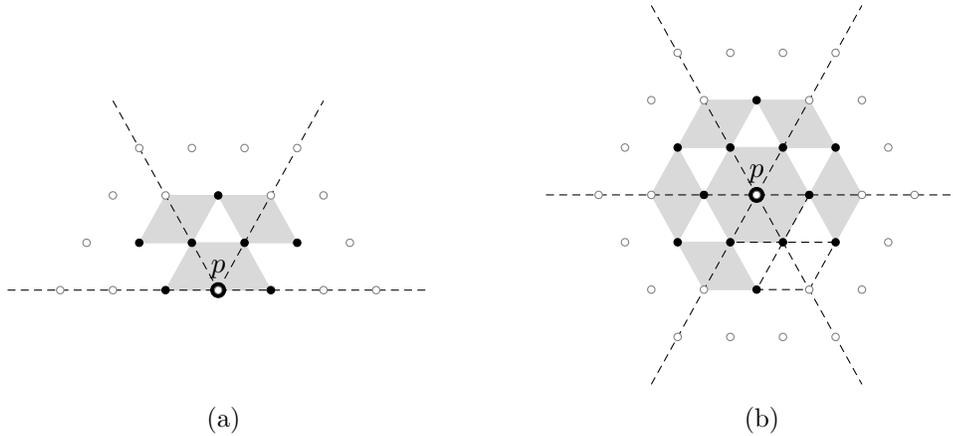


Figura 7.8: Ponto p do tipo *corner* editado pelo usuário, quadriláteros relevantes (cinzas), quadriláteros redundantes (pontilhados), pontos móveis (pretos) e pontos fixos (brancos).

O conjunto de pontos móveis selecionados inclui todos os pontos de tipos *edge corner* e *inner corner* pertencentes aos quadriláteros selecionados. Note que o ponto p editado é sempre considerado um ponto fixo.

Para um vértice com $g \geq 2$ triângulos, este critério resulta em $2g - 2$ quadriláteros relevantes. O número de pontos móveis é $2g$, caso p seja um vértice interno da triangulação T e $2g + 1$, caso contrário. Por exemplo, na Figura 7.8(a), $g = 3$, então temos 4 quadriláteros e 7 pontos móveis. Na Figura 7.8(b), $g = 6$, então temos 10 quadriláteros e 12 pontos móveis.

Se $g = 1$, não há condições a serem satisfeitas, então basta alterar o ponto p .

7.3.2 Edição de um ponto de controle *edge corner*

Quando um ponto p do tipo *edge corner* sob uma aresta compartilhada é editado pelo usuário, o programa seleciona os quadriláteros e os pontos móveis em torno do vértice de T mais próximo, conforme indicado pela Figura 7.9. O processo é análogo ao descrito na seção anterior.

Os pontos móveis são os de tipo *edge corner* e *inner corner*, exceto o ponto p .

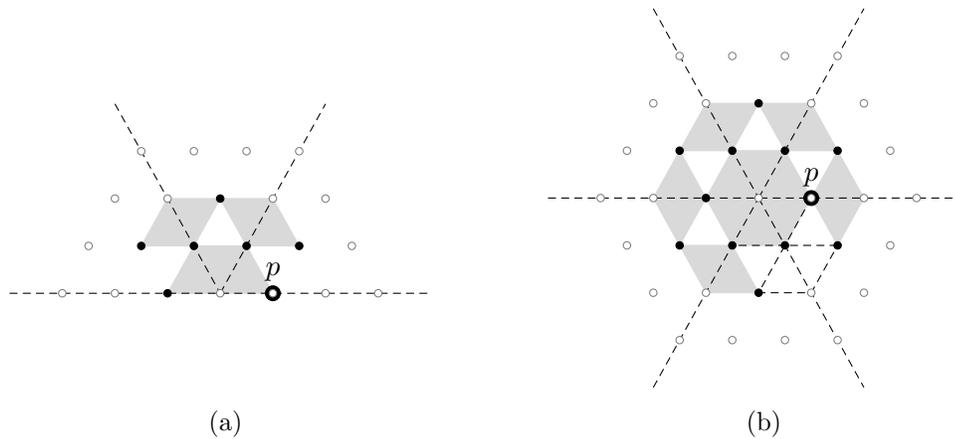


Figura 7.9: Ponto p do tipo *edge corner* editado pelo usuário.

Para um vértice com $g \geq 2$ triângulos, o número de quadriláteros é $2g - 2$ e o número de pontos móveis é $2g - 1$ se o vértice for interno e $2g$ se for externo. Na Figura 7.9(a), $g = 3$ e são selecionados 4 quadriláteros e 6 pontos móveis. Na Figura 7.9(b), $g = 6$ e são selecionados 10 quadriláteros e 11 pontos móveis.

7.3.3 Edição de um ponto de controle *inner corner*

Quando um ponto p do tipo *inner corner* é editado pelo usuário, o programa seleciona os dois quadriláteros que incluem p como membro extremo. Para cada um desses quadriláteros, também é selecionado um outro quadrilátero vizinho na mesma aresta, que compartilha com ele um ponto medial e não é adjacente a um vértice de T , conforme indicado na Figura 7.10(a) e 7.10(b). Os pontos móveis são escolhidos como indicado na figura.

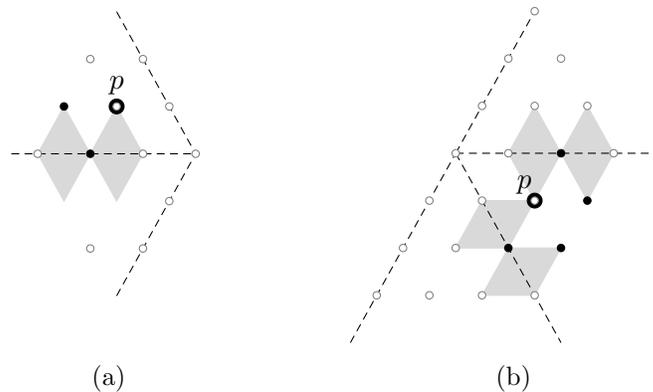


Figura 7.10: Ponto p do tipo *inner corner* editado pelo usuário.

Note que o resultado pode incluir dois quadriláteros e dois pontos de controle ou quatro quadriláteros e quatro pontos de controle.

7.3.4 Edição de um ponto de controle *edge*

Se um ponto p do tipo *edge* é editado, o programa seleciona os dois quadriláteros que incluem p como membro medial. Veja a Figura 7.11(a). Se esses quadriláteros incluem pontos do tipo *inner corner*, então o método é iterado a partir desses pontos. Veja a Figura 7.11(b).

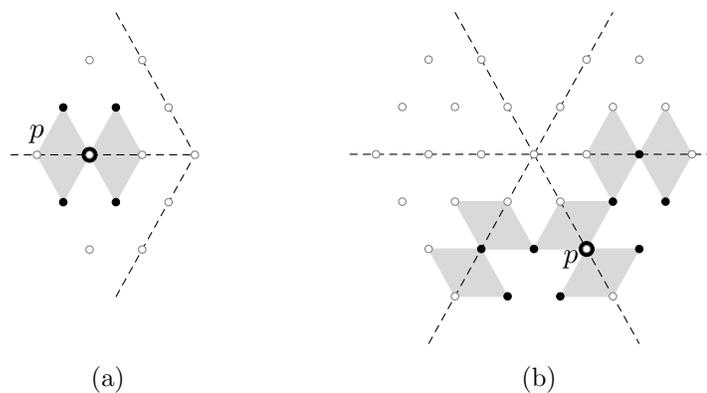


Figura 7.11: Ponto p do tipo *edge* editado pelo usuário.

Note que o resultado pode incluir $n = 2$ ou $n = 6$ quadriláteros e $n + 2$ pontos de controle em cada caso.

7.3.5 Edição de um ponto de controle *inner edge*

Se um ponto p do tipo *inner edge* é editado pelo usuário, o programa seleciona o quadrilátero que inclui p como membro extremo e seleciona como ponto móvel o outro membro extremo deste quadrilátero. Veja a Figura 7.12.

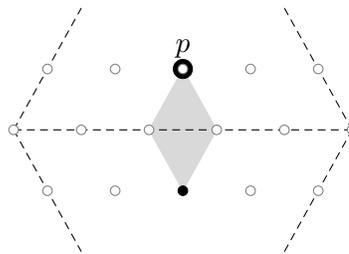


Figura 7.12: Ponto p do tipo *inner edge* editado pelo usuário.

Capítulo 8

Atualização dos Pontos de Controle

Para cada ponto editado, uma vez escolhidos os pontos móveis e as restrições relevantes, o programa gera as equações de continuidade C^1 para gerar o sistema de equações. A solução desse sistema atualiza a posição de cada ponto móvel de acordo com a nova posição do ponto editado pelo usuário.

Em uma spline de grau 5, se o ponto de controle p do tipo *edge* é editado, o algoritmo de seleção inclui dois quadriláteros $p.q1$ e $p.qr$. Dentre os pontos desses quadriláteros, é selecionado o subconjunto $\pi = \{\pi_1, \pi_2, \pi_3, \pi_4\}$ de 4 pontos móveis. Os outros três pontos dos quadriláteros formam o subconjunto de pontos fixos $\varphi = \{\varphi_1, \varphi_2, \varphi_3\}$, onde $\varphi_2 = p$ é o ponto editado. Veja a Figura 8.1(a).

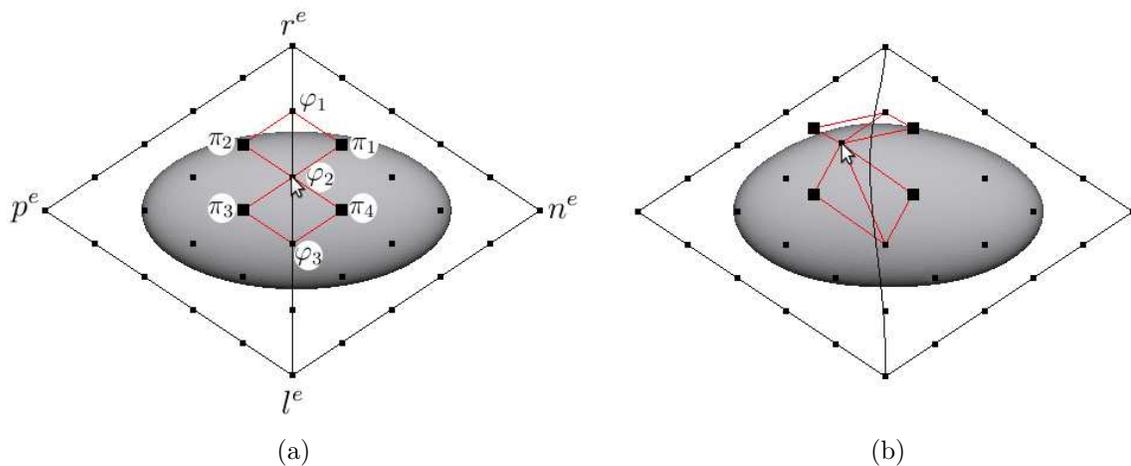


Figura 8.1: Exemplo de edição do ponto de controle p de tipo *edge*.

8.1 Cálculo das coordenadas baricêntricas

Cada quadrilátero selecionado gera uma equação de continuidade C^1 na forma

$$p_i^e = \beta_0 n_i^e + \beta_1 l_i^e + \beta_2 r_i^e, \quad (8.1)$$

ou seja,

$$\beta_0 n_i^e + \beta_1 l_i^e + \beta_2 r_i^e - p_i^e = 0, \quad (8.2)$$

onde β_0 , β_1 e β_2 são as coordenadas baricêntricas de p^e com relação a n^e , l^e e r^e na triangulação T

$$p^e = \beta_0 n^e + \beta_1 l^e + \beta_2 r^e. \quad (8.3)$$

Note que quando o quadrilátero está invertido, β_1 é a coordenada baricêntrica de p^e com relação a r^e e β_2 é a coordenada baricêntrica de p^e com relação a l^e .

No exemplo da Figura 8.1, para os dois quadriláteros são geradas as seguintes equações de continuidade C^1

$$(-1)\pi_1 + (\beta_0)\pi_2 + (\beta_1)\varphi_1 + (\beta_2)\varphi_2 = 0 \quad (8.4)$$

$$(-1)\pi_3 + (\beta_0)\pi_4 + (\beta_2)\varphi_2 + (\beta_1)\varphi_3 = 0. \quad (8.5)$$

8.2 Forma matricial das equações de continuidade

O algoritmo de seleção de pontos móveis garante que o número n de pontos de controle escolhidos é maior ou igual ao número m de quadriláteros e que em cada quadrilátero escolhido há pelo menos um ponto móvel. Podemos então escrever as equações de continuidade geradas na forma de um sistema de equações lineares

$$A\pi = -B\varphi, \quad (8.6)$$

onde π é o vetor de coordenadas dos n pontos móveis, φ é o vetor de coordenadas dos s pontos fixos, A é a matriz $m \times n$ dos coeficientes dos pontos móveis, e B a matriz $m \times s$ dos coeficientes dos pontos fixos. As coordenadas dos pontos móveis e dos pontos fixos são x e y para edição no modo-xy, z_0 no modo-z0 e z_1 no modo-z1.

No exemplo da Figura 8.1, as matrizes de coeficientes são

$$A = \begin{bmatrix} -1 & \beta_0 & 0 & 0 \\ 0 & 0 & -1 & \beta_0 \end{bmatrix} \text{ e } B = \begin{bmatrix} \beta_1 & \beta_2 & 0 \\ 0 & \beta_2 & \beta_1 \end{bmatrix}. \quad (8.7)$$

Quando o número de pontos móveis é igual ao número de equações, podemos resolver o sistema da Equação (8.6) diretamente, obtendo uma única solução. Pelo algoritmo de

seleção, apresentado no Capítulo 7, isso ocorre se e somente se o ponto editado é do tipo *inner corner* ou *inner edge*. Veja as Figuras 8.2 e 8.3.

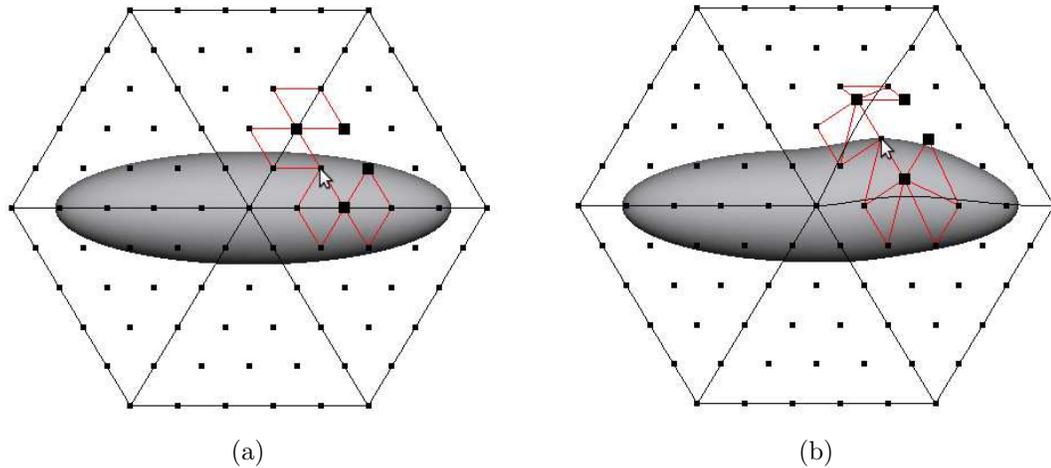


Figura 8.2: Exemplo de edição do ponto de controle p de tipo *inner corner*.

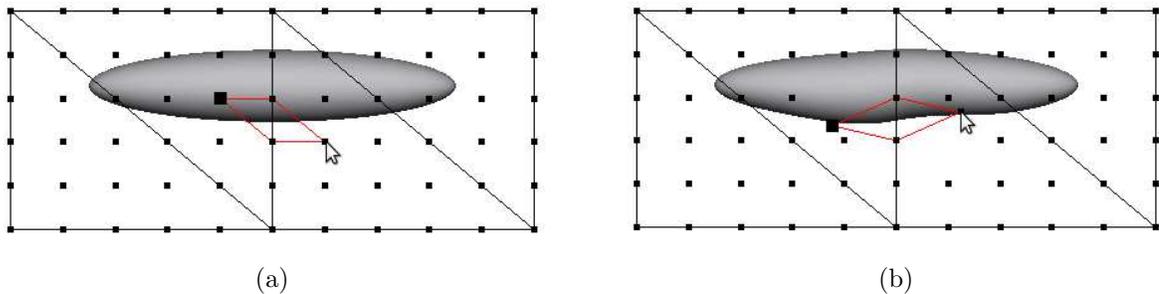


Figura 8.3: Exemplo de edição do ponto de controle p de tipo *inner edge*.

8.3 Resolvendo o sistema indeterminado

Na edição de um ponto de tipo *corner*, *edge corner* ou *edge*, o número de pontos móveis é maior que o número de quadriláteros, como no exemplo da Figura 8.1. Nesse caso, o sistema é indeterminado, ou seja, tem infinitas soluções. Para superar esse obstáculo, procuramos a solução de *mínimos quadrados*, que tenta deslocar os pontos móveis o mínimo possível de modo a satisfazer as equações. Mais precisamente, para cada coordenada considerada (x , y , z_0 ou z_1) queremos minimizar o valor

$$S(\pi) = \sum_{j=1}^n w_j (\pi_j - \pi_j^0)^2, \quad (8.8)$$

onde π_j^0 é a coordenada atual do ponto móvel, π_j a nova coordenada e w_j é um *peso* atribuído a esse ponto.

O objetivo dos pesos w_j é controlar a magnitude dos deslocamentos: quanto maior é o peso w_j em relação aos demais pesos, menos o ponto π_j se move. No nosso editor, atribuímos o peso w_j de acordo com a distância topológica do ponto móvel π_j até o ponto editado pelo usuário. A distância topológica entre dois pontos de controle é o número mínimo de quadriláteros escolhidos que relacionam um com o outro.

Para minimizar S , respeitando as restrições $A\pi = -B\varphi$, é necessário que seu gradiente ∇S seja perpendicular ao conjunto-solução dessas restrições. Matematicamente, o gradiente de S é dado por

$$(\nabla S)_j = \frac{\partial S}{\partial \pi_j} = 2w_j(\pi_j - \pi_j^0) \quad (8.9)$$

para $j = 1, 2, \dots, n$.

Um vetor de \mathbb{R}^n é perpendicular ao conjunto-solução de $A\pi = -B\varphi$ se é da forma $-A^{tr}\lambda$ para algum vetor coluna $\lambda \in \mathbb{R}^m$ onde A^{tr} é a matriz transposta de A . Impondo essa condição ao gradiente de S , temos que

$$2w_j(\pi_j - \pi_j^0) = -\sum_{k=1}^n \lambda_k A_{kj} \quad (8.10)$$

ou seja,

$$2w_j\pi_j + \sum_{k=1}^n \lambda_k A_{kj} = 2w_j\pi_j^0. \quad (8.11)$$

A Equação (8.11) pode ser escrita na forma matricial

$$2W\pi + A^{tr}\lambda = 2W\pi^0. \quad (8.12)$$

onde W é a matriz diagonal dos pesos.

Podemos combinar a Equação (8.12) com as restrições de continuidade C^1 da Equação (8.6) obtendo o sistema linear $Mu = v$, onde M é uma matriz quadrada $(m+n) \times (m+n)$, u é a concatenação dos vetores de incógnitas π e λ , e v é a concatenação dos dois vetores $2W\pi^0$ e $-B\varphi$. Ou seja,

$$\begin{bmatrix} 2W & A^{tr} \\ A & 0 \end{bmatrix} \begin{bmatrix} \pi \\ \lambda \end{bmatrix} = \begin{bmatrix} 2W\pi^0 \\ -B\varphi \end{bmatrix}, \quad (8.13)$$

ou seja,

e

$$b = B\phi = \begin{bmatrix} -1 & -1 & 0 \\ 0 & -1 & -1 \end{bmatrix} \begin{bmatrix} 0 & 0.480 \\ -0.181 & 0.336 \\ 0.000 & -0.160 \end{bmatrix} = \begin{bmatrix} 0.181 & -0.816 \\ 0.181 & -0.176 \end{bmatrix}. \quad (8.18)$$

Note que a segunda linha da matriz φ são as novas coordenadas do ponto editado $p = \varphi_2$. Portanto, o sistema gerado $Mu = x$ é

$$\begin{bmatrix} 2 & 0 & 0 & 0 & -1 & 0 \\ 0 & 2 & 0 & 0 & -1 & 0 \\ 0 & 0 & 2 & 0 & 0 & -1 \\ 0 & 0 & 0 & 2 & 0 & -1 \\ -1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & -1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \pi_1 \\ \pi_2 \\ \pi_3 \\ \pi_4 \\ \lambda_1 \\ \lambda_2 \end{bmatrix} = \begin{bmatrix} 0.480 & 0.640 \\ -0.480 & 0.640 \\ -0.480 & 0.000 \\ 0.480 & 0.000 \\ 0.181 & -0.816 \\ 0.181 & -0.176 \end{bmatrix}. \quad (8.19)$$

Resolvendo-se o sistema, obtemos as novas posições (x, y) dos pontos móveis, ou seja, temos o vetor π

$$\pi = \begin{bmatrix} 0.150 & 0.408 \\ -0.330 & 0.408 \\ -0.330 & 0.088 \\ 0.150 & 0.088 \end{bmatrix}. \quad (8.20)$$

Os pontos móveis reposicionados de acordo com estas coordenadas estão apresentadas na Figura 8.1(b).

Nas Figuras 8.4, 8.5 e 8.6, apresentamos mais alguns exemplos de edição onde o número de pontos móveis é maior que o número de quadriláteros selecionados.

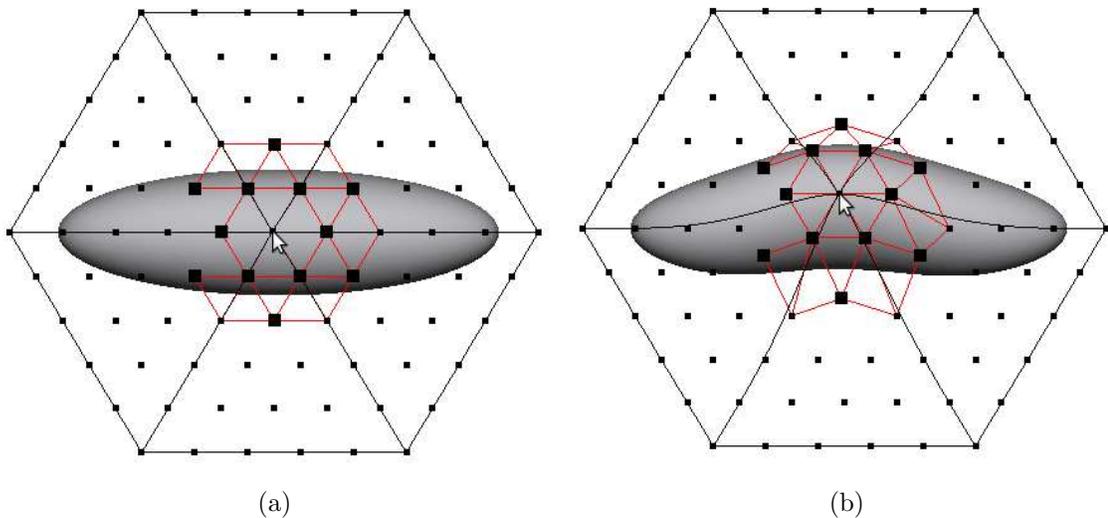


Figura 8.4: Exemplo de edição do ponto de controle p de tipo *corner*.

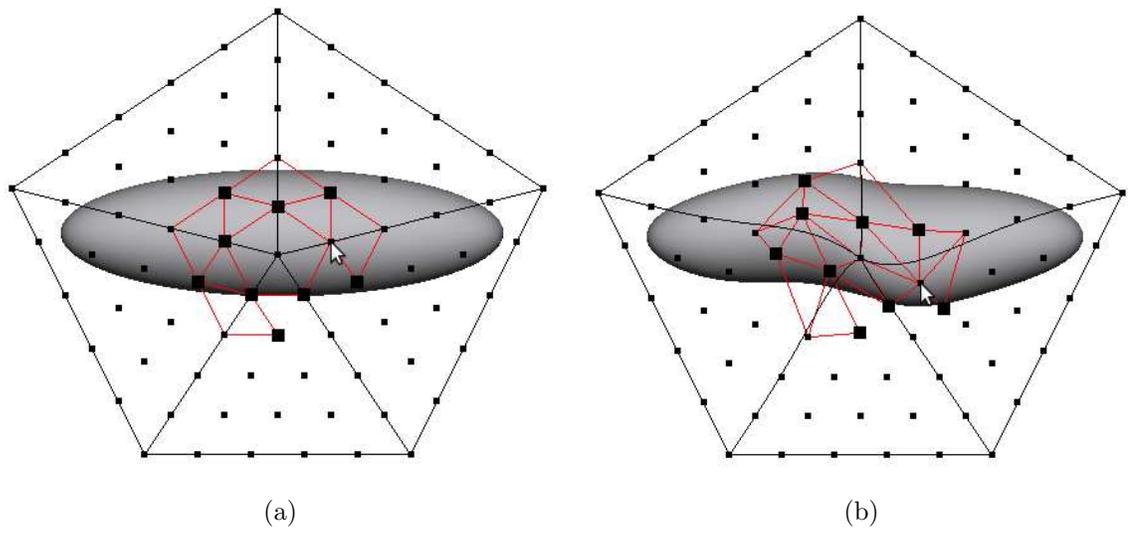


Figura 8.5: Exemplo de edição do ponto de controle p de tipo *edge corner*.

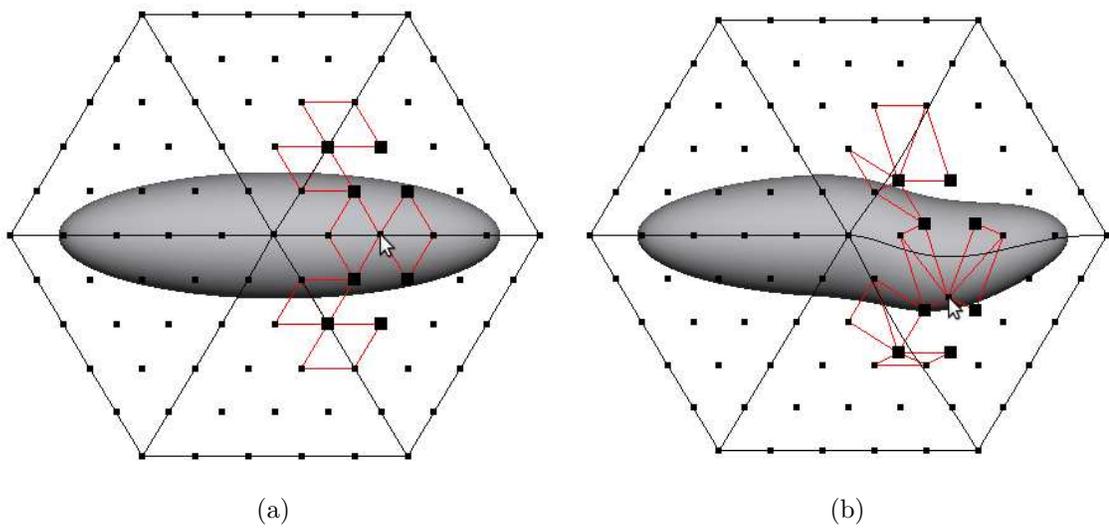


Figura 8.6: Exemplo de edição do ponto de controle p de tipo *edge*.

Capítulo 9

Experimentos e Resultados

Neste capítulo apresentamos os experimentos e imagens dos resultados obtidos. Nossos experimentos testaram a adequação da formulação construída, usando o editor implementado para reproduzir algumas deformações de micro-organismos, observadas em imagens microscópicas reais.

Foram utilizados dois organismos nos experimentos, o nematoide *Caenorhabditis elegans* e o protozoário *Dileptus anser*.

9.1 Construção do modelo 3D e da grade

Para cada organismo, construímos um modelo usando o editor Blender 3D [37]. O modelo é uma malha triangular densa da forma básica (não deformada) do organismo. Usamos, em seguida, nosso editor para construir uma grade de controle apropriada a cada modelo, a partir da triangulação domínio T fornecida na forma de um arquivo de texto e dois valores de altura. Veja as Figuras 9.1 e 9.2.

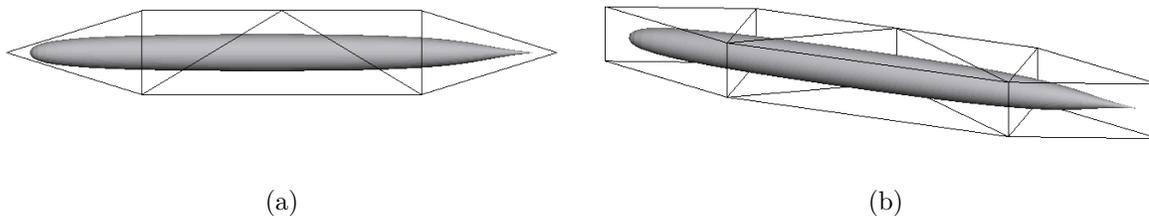


Figura 9.1: Visão 2D (a) e visão 3D (b) do modelo 3D do *Caenorhabditis elegans* e sua grade de prismas.

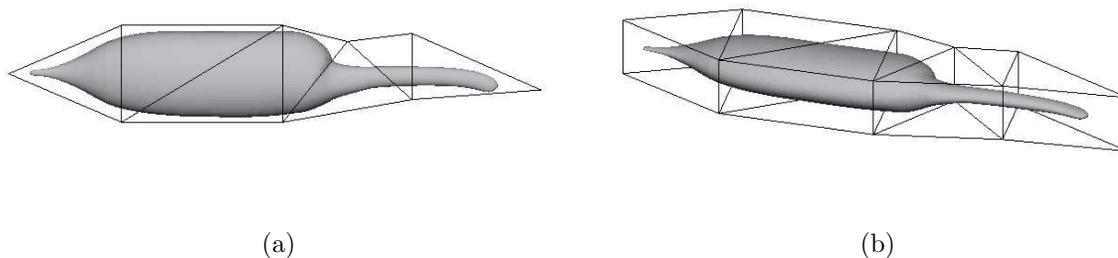


Figura 9.2: Visão 2D (a) e visão 3D (b) do modelo 3D do *Dileptus anser* e sua grade de prismas.

As Tabelas 9.1 e 9.2 resumem os parâmetros dos modelos e das splines definidas sobre as grades de controle.

	<i>C. elegans</i>	<i>Dileptus anser</i>
<i>Número de vértices</i>	10425	18967
<i>Número de faces</i>	20830	37930

Tabela 9.1: Parâmetros dos modelos usados nos testes.

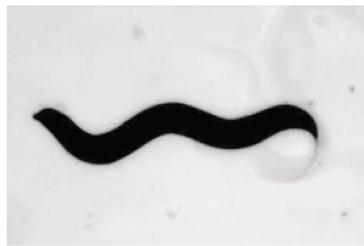
	<i>C. elegans</i>	<i>Dileptus anser</i>
<i>Grau da spline</i>	5	5
<i>Número de vértices</i>	7	9
<i>Número de faces</i>	5	7
<i>Número de arestas</i>	11	15
<i>Número de arestas compartilhadas</i>	4	6
<i>Número de pontos de controle</i>	81	111
<i>Número de condições de quadrilátero</i>	20	30

Tabela 9.2: Parâmetros das splines definidas sobre as grades dos testes.

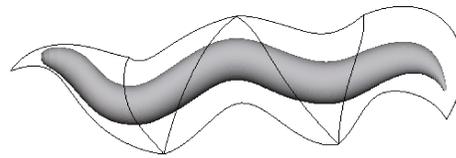
9.2 Resultados

Obtivemos da Internet algumas imagens microscópicas reais desses organismos em posições variadas. Usamos então o nosso editor para deformar de forma interativa o modelo 3D da forma básica do organismo até o casamento com as imagens reais. As imagens reais e os resultados obtidos são mostrados nas figuras 9.3 a 9.5.

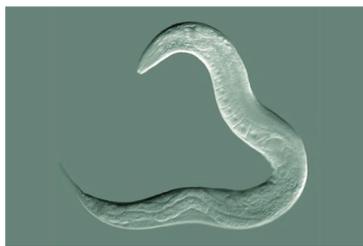
Nas figuras 9.3 e 9.4, foi suficiente deformar o organismo no modo-xy. Na Figura 9.5, onde o organismo está dobrado sobre si mesmo, também foram usados o modo-z0 e o modo-z1 para evitar autointersecções.



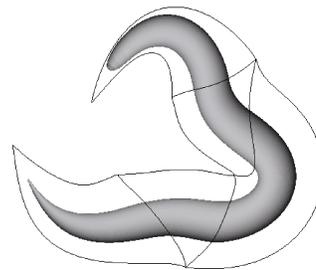
(a) [20].



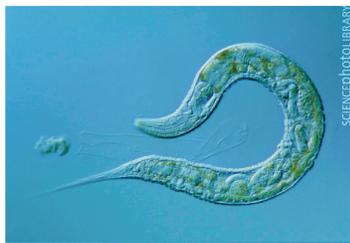
(b)



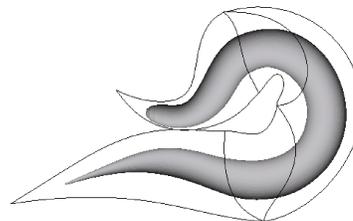
(c) [18].



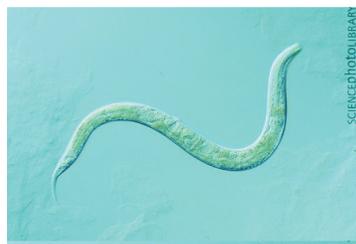
(d)



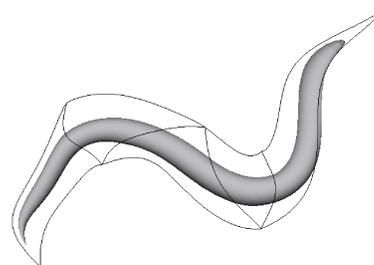
(e) [28].



(f)



(g) [29].

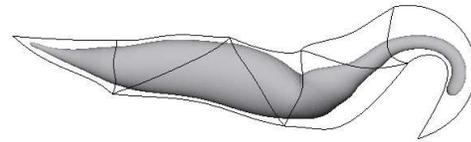


(h)

Figura 9.3: Quatro imagens reais do *C. elegans* (esquerda) e visão 2D dos modelos deformados manualmente no modo-xy (direita).



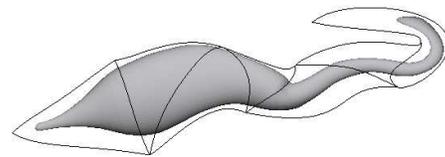
(a) [14].



(b)



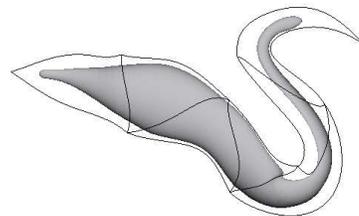
(c) [14].



(d)



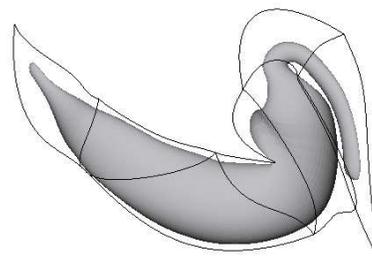
(e) [14].



(f)

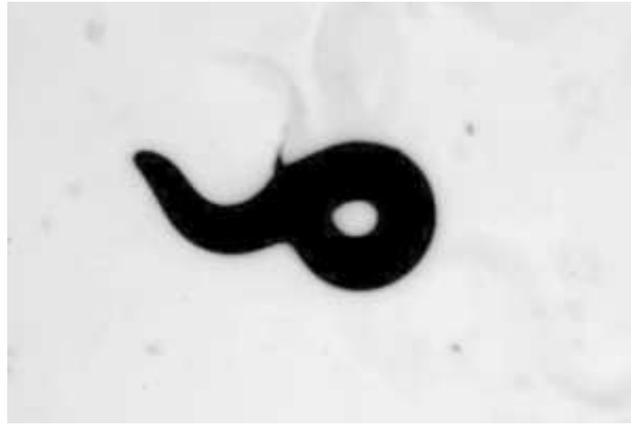


(g) [38].

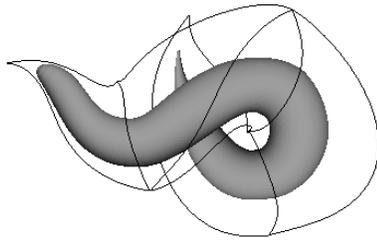


(h)

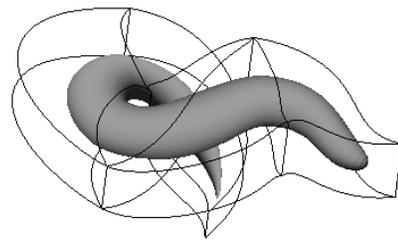
Figura 9.4: Quatro imagens reais do *Dileptus* (esquerda) e visão 2D dos modelos deformados manualmente no modo-xy (direita).



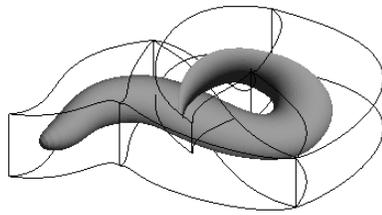
(a) [20].



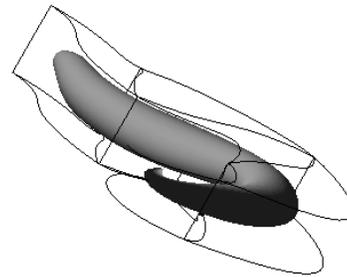
(b)



(c)



(d)



(e)

Figura 9.5: Imagem real do *C. elegans* (a), modelo deformado manualmente no modo-xy, modo-z0 e modo-z1, visão 2D (b) e visões 3D do mesmo (c), (d) e (e).

Capítulo 10

Conclusões e Trabalhos Futuros

Nesta dissertação estudamos a deformação do espaço para modelos 3D de micro-organismos utilizando grades de controle volumétricas e splines como técnica de interpolação.

Implementamos um método de deformação do espaço 2.5D, baseado em uma grade de controle formada por uma camada simples de prismas triangulares. Esta abordagem permite a edição separada das três splines que compõem a grade em três modos distintos - modo-xy, modo-z0 e modo-z1. A deformação da grade é obtida através da deformação dessas splines. Uma deformação $\phi : \mathbb{R}^2 \rightarrow \mathbb{R}^2$ do plano xy e duas funções $\sigma_0, \sigma_1 : \mathbb{R}^2 \rightarrow \mathbb{R}$ que deformam a superfície superior e inferior da grade. Essas splines são definidas sobre um triangulação plana T e possuem o mesmo grau $d \geq 5$. No modo-xy, a deformação é aplicada a ambas superfícies da grade; no modo-z0 e no modo-z1, ela é aplicada separadamente a cada superfície. A interpolação implementada garante deformações locais e que essas sejam transferidas de forma consistente para o modelo embutido na grade, garantindo a suavidade da sua superfície.

A definição da grade de prismas reduziu o número de pontos de controle comparado a grades hexaedrais e tetraedrais, e conseqüentemente, facilitou a manipulação da deformação. Uma vez que, na aplicação considerada, ocorrem poucas deformações na terceira dimensão, este método 2.5D permitiu obter resultados satisfatórios com muito menos trabalho de edição.

Para representar internamente estas splines, construímos uma estrutura de vizinhança baseada nos pontos de controle de Bézier e nas condições de quadrilátero que representam as restrições de continuidade C^1 . Essa estrutura facilitou a implementação dos algoritmos de edição da spline já que dispensa consultas às informações da triangulação durante a deformação.

Os resultados experimentais demonstraram que é possível simular vários tipos de deformações que ocorrem em estruturas biológicas. Os testes realizados, com dois organismos altamente deformáveis, mostram que o método satisfaz os objetivos iniciais do projeto.

Podemos citar como possíveis trabalhos futuros os seguintes tópicos:

- **Técnicas de edição da malha de controle:** é possível facilitar a aquisição de algumas deformações grosseiras dos micro-organismos, que ocorrem com frequência, implementando técnicas mais elaboradas de edição da malha de controle. Uma ideia é utilizar uma abordagem multiescala com grades de controle de diferentes resoluções. Outra possibilidade seria implementar um método de deformação do espaço baseado em curvas, onde cada ponto da curva representa um grupo de pontos de controle da grade de prismas do nosso método.
- **Restrições para preservar volume:** um dos motivos para se implementar um método de deformação para modelos 3D, mesmo que as deformações dos micro-organismos sejam essencialmente 2D, é que assim é possível manter o volume dos modelos durante uma deformação. Podemos incorporar restrições para preservação do volume, tanto localmente (para organismos sólidos) quanto do modelo todo (para organismos com interior fluido).
- **Restrições para evitar autointersecções:** esse tipo de restrição é interessante para a aplicação pois, como os micro-organismos habitam em ambientes líquidos ou gelatinosos, é possível que eles passem por cima ou por baixo de si mesmos. Nesse caso, para evitar que o nosso método produza resultados irreais, o usuário precisa fazer alterações na terceira dimensão manualmente, o que exige do usuário um pouco mais de trabalho na edição. Podemos incorporar essa restrição no método implementado de forma a identificar os casos em que o modelo colide consigo mesmo e aplicar mudanças automáticas na terceira dimensão da grade de controle de forma que facilite a edição da deformação e não gere autointersecções.

Apêndice A

Interface do Editor

Este capítulo apresenta a interface do nosso editor de deformação para modelos 3D, implementado em Qt/C++ e OpenGL.

A.1 Interface inicial do editor

O editor apresenta a interface inicial mostrada na Figura A.1. Esta interface possui uma barra de menus, uma barra de controles e uma janela de edição.

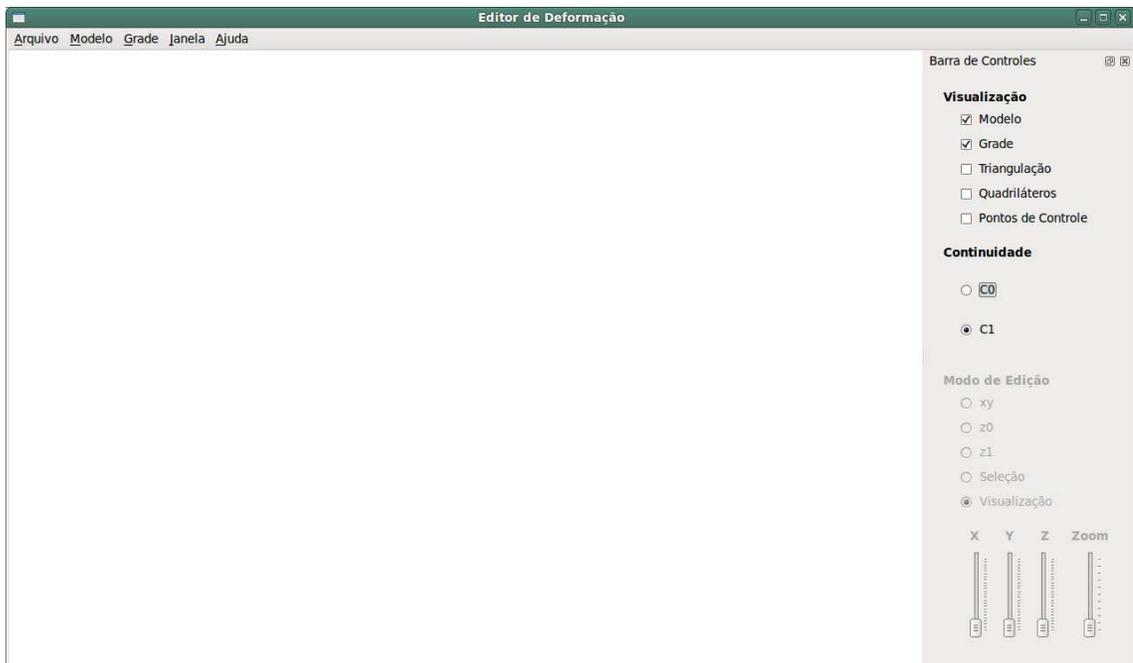


Figura A.1: Interface inicial do editor de deformação para modelos 3D.

Na barra de menus existem os menus para carregar um modelo 3D, criar ou carregar uma grade de prismas. Também possui opções para salvar e limpar tanto o modelo quanto a grade exibidos na janela de edição.

O modelo 3D é carregado a partir de um arquivo no formato `.obj`, que contém uma malha triangular. A grade também pode ser carregada a partir de um arquivo no formato `“.obj”` do Blender [37], que contém uma malha triangular 2D plana, cujos vértices tem todos $z = 0$. Se o arquivo não existir é possível criar uma triangulação 2D no próprio editor a partir da definição de seus vértices e arestas. Em seguida, o usuário define a posição superior e inferior da grade de prismas. O programa então se encarrega da construção das splines e de criar o relacionamento entre modelo e grade.

Na barra de controles existem opções que o usuário pode alterar em tempo de execução. Entre elas é possível escolher o que será exibido na janela de edição dentre as seguintes opções: modelo 3D, grade de primas, pontos de controle, triangulação sobre os pontos de controle e quadriláteros das restrições de suavidade. O usuário também pode escolher entre uma deformação com continuidade C^0 ou C^1 , sendo C^1 a opção padrão. Veja Figura A.2(a). Na barra de controles, o usuário também pode escolher entre o modo de visualização, o modo de seleção e os modos de edição. Veja Figura A.2(b).

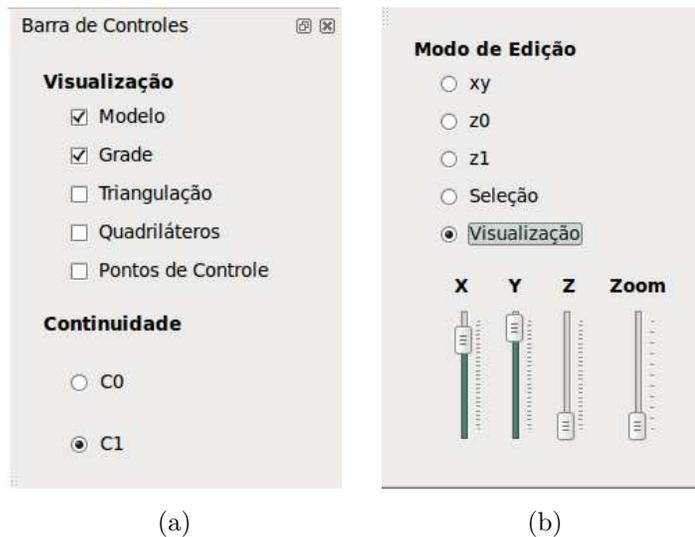


Figura A.2: Barra de controles.

A.2 Modos de visualização, seleção e edição

No modo de visualização, a definição padrão das opções de visualização, descritas na seção anterior, permitem ao usuário visualizar apenas o modelo e a grade de prismas. No

entanto, essas opções podem ser alteradas como desejado. Veja Figura A.3. Neste modo, é possível rotacionar a grade e o modelo de qualquer forma, e aproximá-los ou afastá-los, pelos controles habilitados (parte inferior da barra de controles).

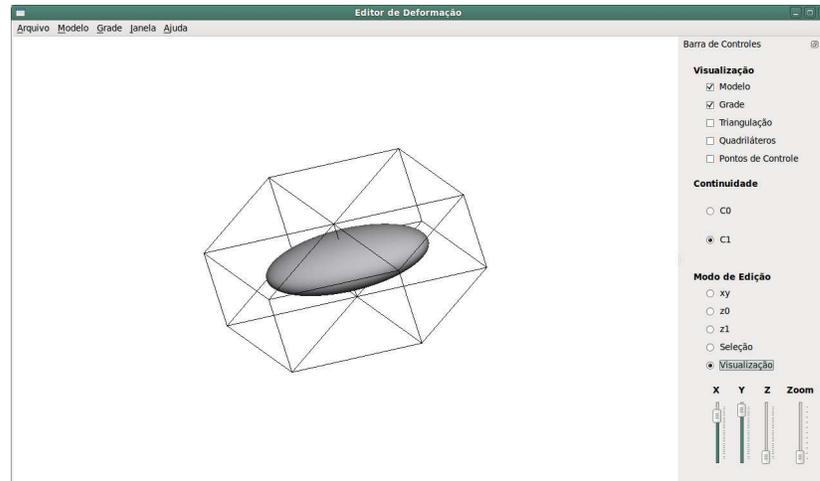


Figura A.3: Janela de edição no modo de visualização.

No modo de seleção, o usuário visualiza o modelo, a grade, os quadriláteros das restrições e os pontos de controle, por padrão. No momento da seleção, a rotação da grade será igual àquela do modo de edição anterior. Por exemplo, na Figura A.4, o usuário estava no modo-xy de edição antes de escolher o modo de seleção, enquanto na Figura A.5, o modo-z0 estava selecionado.

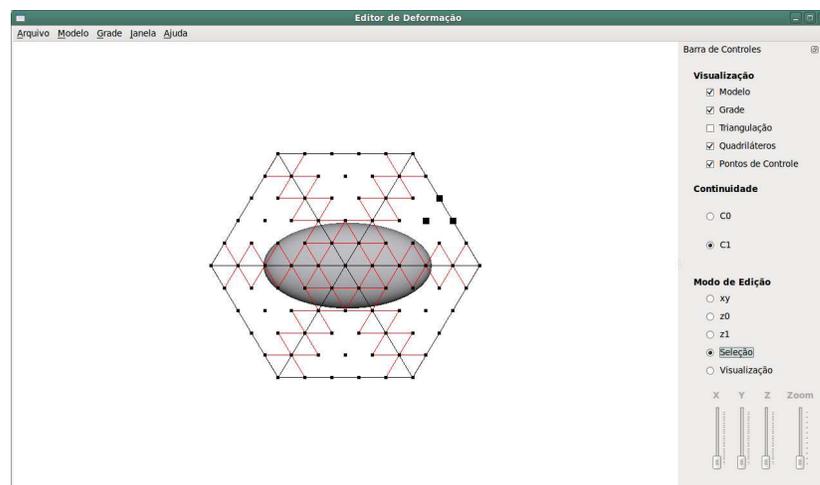


Figura A.4: Janela de edição no modo de seleção, com modo-xy selecionado anteriormente.

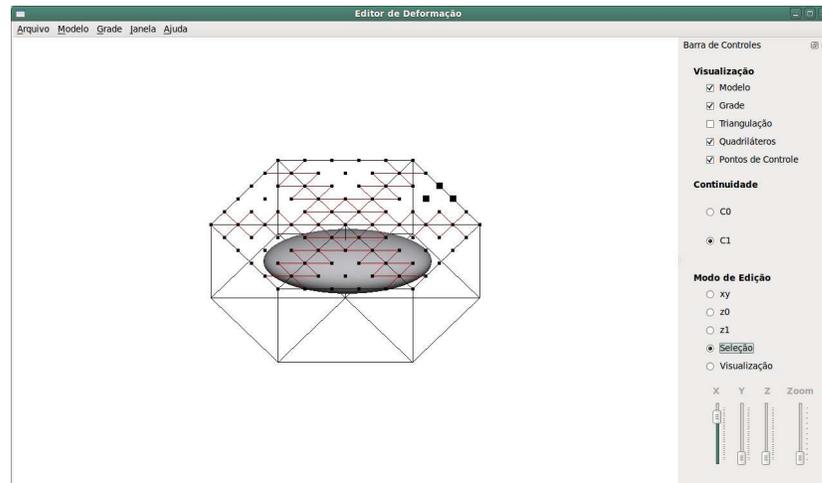


Figura A.5: Janela de edição no modo de seleção, com modo-z0 selecionado anteriormente.

Com os pontos selecionados, o usuário pode escolher um outro modo, diferente do anterior, para editá-los. Os pontos selecionados serão deslocados juntos com o ponto editado pelo usuário. Isso restringe os pontos que podem ser selecionados, já que pontos que participam das condições de continuidade podem ter deslocamentos diferentes. Os demais pontos podem ser selecionados livremente.

Conforme descrito na Seção 5.2.1, existem três modos de edição: modo-xy (veja Figura A.6); modo-z0 (veja Figura A.7); e, modo-z1 (veja Figura A.8).

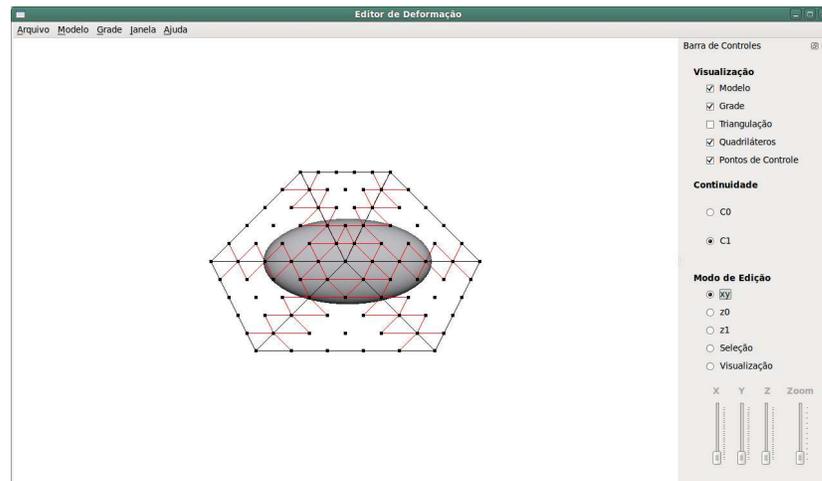


Figura A.6: Janela de edição no modo-xy de edição.

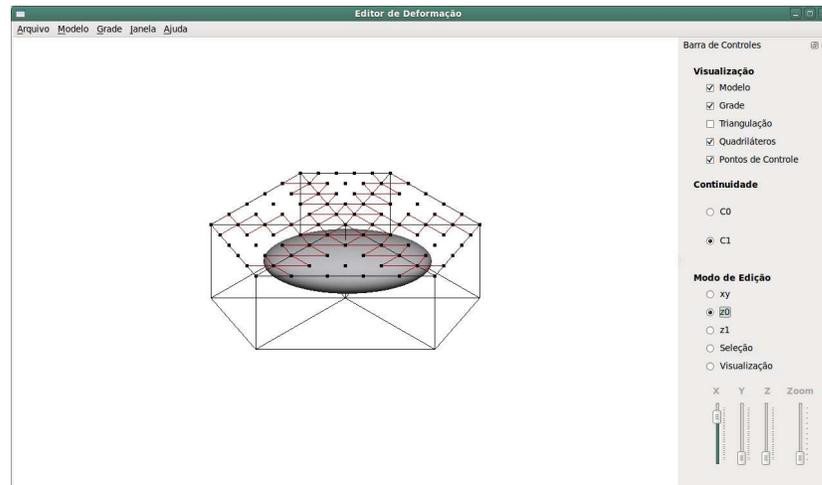


Figura A.7: Janela de edição no modo-z0 de edição.

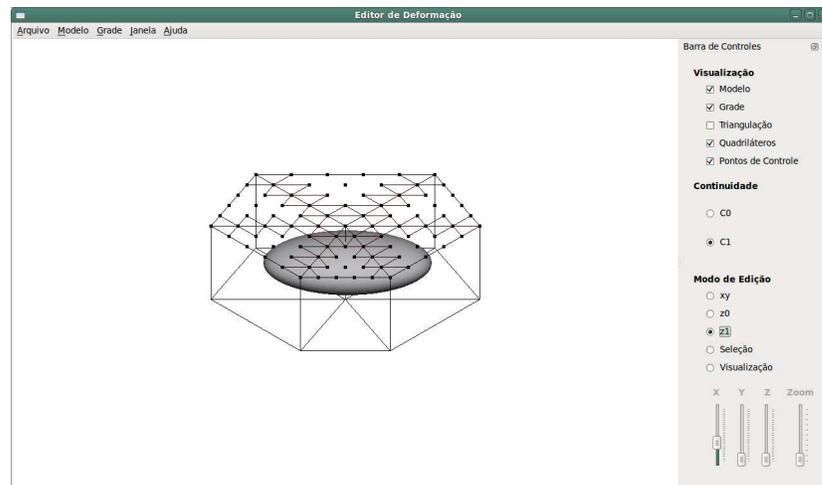


Figura A.8: Janela de edição no modo-z1 de edição.

Referências Bibliográficas

- [1] Alexis Angelidis, Marie-Paule Cani, Geoff Wyvill, and Scott King. Swirling-sweepers: Constant-volume modeling. *Graph. Models*, 68(4):324–332, 2006.
- [2] Alexis Angelidis, Geoff Wyvill, and Marie-Paule Cani. Sweepers: Swept user-defined tools for modeling by deformation. In *SMI '04: Proceedings of the Shape Modeling International 2004*, pages 63–73, Washington, DC, USA, 2004. IEEE Computer Society.
- [3] Fabrice Aubert and Dominique Bechmann. Volume-preserving space deformation. *Comput. Graph.*, 21(5):625–639, 1997.
- [4] Alan H. Barr. Global and local deformations of solid primitives. *SIGGRAPH Comput. Graph.*, 18:21–30, January 1984.
- [5] Cagatay Basdogan and Chih-Hao Ho. *Force reflecting deformable objects for virtual environments*. SIGGRAPH'99 Course Notes n. 38, SIGGRAPH-ACM publication, 1999.
- [6] Dominique Bechmann, Yves Bertrand, and Sylvain Thery. Continuous free-form deformation. In *COMPUGRAPHICS '96: Proceedings of the fifth international conference on computational graphics and visualization techniques on Visualization and graphics on the World Wide Web*, pages 1715–1725, New York, NY, USA, 1997. Elsevier Science Inc.
- [7] Mark Blaxter. Mark blaxter's teaching pages. Disponível em: <http://www.nematodes.org/teaching/tutorials/Caenorhabditis/caenorhabditis.shtml>. Acessado em: 13 Ago 2011.
- [8] Mario Botsch and Olga Sorkine. On linear variational surface deformation methods. *IEEE Transactions on Visualization and Computer Graphics*, 14(1):213–230, 2008.
- [9] Sabine Coquillart. Extended free-form deformation: A sculpturing tool for 3d geometric modeling. In *SIGGRAPH '90: Proceedings of the 17th annual conference on*

- Computer graphics and interactive techniques*, pages 187–196, New York, NY, USA, 1990. ACM.
- [10] Philippe Decaudin. Geometric deformation by merging a 3D-object with a simple shape. In *GI '96: Proceedings of the conference on Graphics interface '96*, pages 55–60, Toronto, Ont., Canada, 1996. Canadian Information Processing Society.
- [11] Gerald Farin. *Curves and surfaces for CAGD: A practical guide*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 5th edition, 2002.
- [12] Jieqing Feng, Lizhuang Ma, and Qunsheng Peng. A new free-form deformation through the control of parametric surfaces. *Computers and Graphics*, 20:531–539, 1996.
- [13] Jieqing Feng, Jin Shao, Xiaogang Jin, Qunsheng Peng, and A. Robin Forrest. Multi-resolution free-form deformation with subdivision surface of arbitrary topology. *The Visual Computer*, 22(1):28–42, 2006.
- [14] Donald L. Ferry. Vidcaps from the Lake near Mud Lake. Disponível em: <http://wolfbat359.com/crpvc.html>. Acessado em: 27 Maio 2011.
- [15] Michael S. Floater. Mean value coordinates. *Comput. Aided Geom. Des.*, 20:19–27, March 2003.
- [16] Michael S. Floater, Géza Kós, and Martin Reimers. Mean value coordinates in 3d. *Comput. Aided Geom. Des.*, 22:623–631, October 2005.
- [17] James Gain and Dominique Bechmann. A survey of spatial deformation from a user-centered perspective. *ACM Trans. Graph.*, 27(4):1–21, 2008.
- [18] Bob Goldstein. The Goldstein Lab. Disponível em: <http://www.bio.unc.edu/Faculty/Goldstein/lab/wormMO.gif>. Acessado em: 27 Maio 2011.
- [19] Anamaria Gomide. *Splines polinômiais não homogêneos na esfera*. PhD em engenharia elétrica, Faculdade de Engenharia Elétrica e de Computação - Universidade Estadual de Campinas, Philadelphia, PA, USA, 1999.
- [20] John Alex Halderman. Worm patterns. Disponível em: <http://www.youtube.com/watch?v=7WOxyVvMp8s>. Acessado em: 27 Maio 2011.

- [21] Gentaro Hirota, Renee Maheshwari, and Ming C. Lin. Fast volume-preserving free form deformation using multi-level optimization. In *SMA '99: Proceedings of the fifth ACM Symposium on Solid Modeling and Applications*, pages 234–245, New York, NY, USA, 1999. ACM.
- [22] Kai Hormann and Michael S. Floater. Mean value coordinates for arbitrary planar polygons. *ACM Transactions on Graphics*, 25:1424–1441, 2006.
- [23] Jin Huang, Lu Chen, Xinguo Liu, and Hujun Bao. Efficient mesh deformation using tetrahedron control mesh. In *SPM '08: Proceedings of the 2008 ACM Symposium on Solid and Physical Modeling*, pages 241–247, New York, NY, USA, 2008. ACM.
- [24] Tao Ju, Scott Schaefer, and Joe Warren. Mean value coordinates for closed triangular meshes. *ACM Trans. Graph.*, 24:561–566, July 2005.
- [25] Kazuya G. Kobayashi and Katsutoshi Ootsubo. t-ffd: Free-form deformation by using triangular mesh. In *Proceedings of the eighth ACM Symposium on Solid Modeling and Applications*, SM '03, pages 226–234, New York, NY, USA, 2003. ACM.
- [26] Ming-Jun Lai and Larry L. Schumaker. *Spline functions on triangulations*. Cambridge University Press, New York, NY, USA, 2007.
- [27] Henry J. Lamousin and Warren N. Waggenspack Jr. NURBS-based free-form deformations. *IEEE Comput. Graph. Appl.*, 14(6):59–65, 1994.
- [28] Science Photo Library. Lm of the nematode worm, *Caenorhabditis elegans*. Disponível em: <http://www.sciencephoto.com/media/366424/enlarge>. Acessado em: 13 Ago 2011.
- [29] Science Photo Library. Lm of the nematode worm, *Caenorhabditis elegans*. Disponível em: <http://www.sciencephoto.com/media/366425/enlarge>. Acessado em: 13 Ago 2011.
- [30] Yaron Lipman, David Levin, and Daniel Cohen-Or. Green coordinates. *ACM Trans. Graph.*, 27:78:1–78:10, August 2008.
- [31] Ignacio Llamas, Alexander Powell, Jarek Rossignac, and Chris D. Shaw. Bender: A virtual ribbon for deforming 3D shapes in biomedical and styling applications. In *Proceedings of the 2005 ACM Symposium on Solid and Physical Modeling*, SPM '05, pages 89–99, New York, NY, USA, 2005. ACM.

- [32] Ron MacCracken and Kenneth I. Joy. Free-form deformations with lattices of arbitrary topology. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 181–188, New York, NY, USA, 1996. ACM.
- [33] Andrew Nealen, Matthias Müller, Richard Keiser, Eddy Boxerman, and Mark Carlson. Physically based deformable models in computer graphics. *Comput. Graph. Forum*, 25(4):809–836, 2006.
- [34] Carl E. Pearson. *Handbook of applied mathematics: Selected results and methods*. Van Nostrand Reinhold, New York, NY, US, 2th edition, 1990.
- [35] Thomas W. Sederberg and Scott R. Parry. Free-form deformation of solid geometric models. *SIGGRAPH Comput. Graph.*, 20(4):151–160, 1986.
- [36] Zhang Z Shen YF. Modern biomonitoring techniques using freshwater microbiota. Disponível em: <http://starcentral.mbl.edu/microscope>. Acessado em: 13 Ago 2011.
- [37] The Blender Foundation. Blender. Disponível em: <http://www.blender.org>. Acessado em: 27 Maio 2011.
- [38] Y. Tsukii. Protist Images: Dileptus anser. Disponível em: <http://protist.i.hosei.ac.jp/pdb/images/Ciliophora/Dileptus/anser3.jpg>. Acessado em: 13 Ago 2011.
- [39] Wolfram von Funck, Holger Theisel, and Hans-Peter Seidel. Vector field based shape deformations. *ACM Trans. Graph.*, 25(3):1118–1125, 2006.
- [40] Wolfram von Funck, Holger Theisel, and Hans-Peter Seidel. Explicit control of vector field based shape deformations. In *PG '07: Proceedings of the 15th Pacific Conference on Computer Graphics and Applications*, pages 291–300, Washington, DC, USA, 2007. IEEE Computer Society.
- [41] Dianna Xu. *Incremental algorithms for the design of triangular-based spline surfaces*. PhD in computer and information science, Faculties of the University of Pennsylvania, Campinas, SP, Brasil, 2002.
- [42] Wei-Wei Xu and Kun Zhou. Gradient domain mesh deformation: A survey. *J. Comput. Sci. Technol.*, 24(1):6–18, 2009.
- [43] Yong Zhao, Xinguo Liu, Chunxia Xiao, and Qunsheng Peng. A unified shape editing framework based on tetrahedral control mesh. *Comput. Animat. Virtual Worlds*, 20(2-3):301–310, 2009.