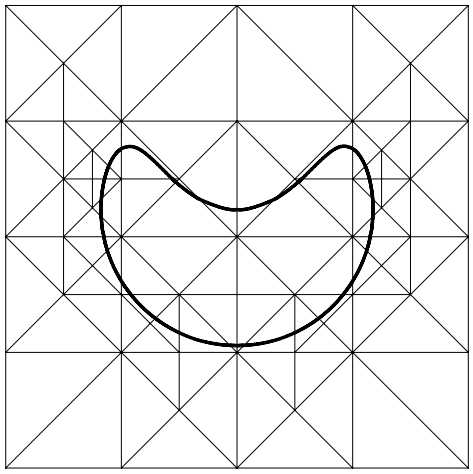
Novos Métodos Simpliciais em Computação Gráfica



Vinícius Moreira Mello

Abril de 2006

Resumo

Apresentamos uma nova representação de hipersuperfícies que combina características das representações implícita e paramétrica. A abordagem consiste em construir uma decomposição simplicial do espaço ambiente, determinar uma aproximação linear da hipersuperfície em tal decomposição e aplicar deformações em cada simplexo, os chamados *difeomorfismos simpliciais*. O assunto é tratado de forma independente da dimensão da hipersuperfície, tanto do ponto de vista matemático quanto do computacional, graças a uma análise detalhada dos requisitos sintáticos e semânticos dos conceitos da *topologia combinatória* subjacentes, análise esta subordinada aos princípios da *programação genérica*. A noção de *esquema de subdivisão estelar* é tratada de modo semelhante, a fim de formalizar a especificação de algoritmos de *multirresolução adaptativa*. Apresentamos ainda algumas aplicações que ilustram o potencial da representação.

Abstract

We present a new representation of hypersurfaces which combines features of implicit and parametric representations. The approach consists in to build a simplicial decomposition of the ambient space, to determine a linear approximation of the hypersurface in such decomposition and to apply warpings to each simplex, the so-called *simplicial diffeomorphisms*. The subject is discussed in a way independent of the hypersurface dimension, not only in the mathematical point of view, but also in the computational one, thanks to a detailed analysis of the syntactical and semantical requirements of the *combinatorial topology* underlying concepts. Such analysis obeys the *generic programming* principles. The *stellar subdivision scheme* concept is approched likewise in order to formalize the especification of *adaptive multiresolution* algorithms. We also present applications that show the potential of this new representation.

Conteúdo

Agradecimentos

> It is lack of confidence, more than anything else, that kills a civilisation. We can destroy ourselves by cynicism and disillusion, just as effectively as by bombs.
>
> Kenneth Clark

Gostaria de agradecer em primeiro lugar a Luiz Velho, que sempre confiou em mim, até quando nem mesmo eu confiava, e sempre me incentivou com seu entusiasmo inesgotável e sua paixão pela computação gráfica.

Aos professores Jonas Gomes, Paulo Cezar Pinto Carvalho, Luiz Henrique de Figueiredo e Ralph Teixeira, agradeço por terem me ensinado, com a excelência característica do IMPA, muito do que sei sobre computação gráfica.

Também aprendi muito interagindo em trabalhos de pesquisa com Paulo Roma Cavalcanti, Thomas Lewiner e Hélio Lopes, por isso deixo aqui meu sincero agradecimento.

Não poderia deixar de agradecer a todos os colegas do VISGRAF pelo companherismo, em particular a Moacyr Alvim Horta Barbosa da Silva, Margareth Prevot e Asla Sá, que mais de uma vez me deram abrigo no Rio, e a Boris Mederos e Esdras Medeiros Filho, que ajudaram na revisão da tese.

Devo também expressar minha gratidão de maneira geral ao IMPA, pela incrível estrutura, e ao CNPq, pela bolsa.

Finalmente, agradeço a Ximena Cuevas, Olga Revelles e a meus pais pelo carinho que me dedicaram nesse turbulento período de gestação de uma tese.

Capítulo 1

Introdução

Apresentamos uma nova representação de hipersuperfícies que combina características das representações implícita e paramétrica. Para dar uma idéia geral dessa nova representação, vamos considerar um problema clássico em computação gráfica, a poligonização de curvas implícitas. Considere, por exemplo, a curva C, definida implicitamente pela equação

$$C(x, y) := (y - x^2 + 1)^4 + (x^2 + y^2)^4 - 1 = 0,$$

cujo gráfico pode ser visto na figura 1.1.

Existem vários algoritmos de poligonização de curvas implícitas, mas, em geral, eles procedem assim: triangule um retângulo que contém a curva, avalie o a função $C(x, y)$ nos vértices da triangulação (figura 1.2) e aproxime $C(x, y)$ linearmente em cada triângulo (figura 1.3).

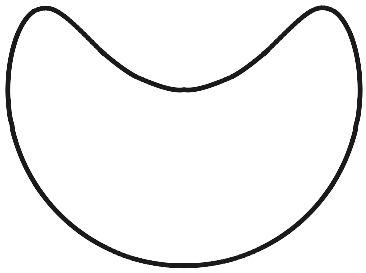


Figura 1.1: Gráfico de $C(x, y) = 0$.

As diferenças entre os diversos algoritmos de poligonização residem nas várias decisões que devem ser tomadas nos passos sumarizados acima, por exemplo, qual triangulação tomar, como aproximar a curva dentro de cada triângulo e,

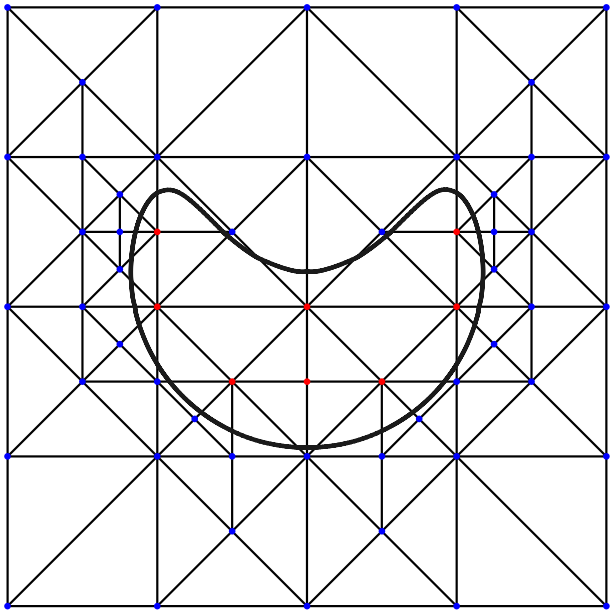


Figura 1.2: Triangulação de um retângulo envolvente à curva $C(x, y) = 0$.

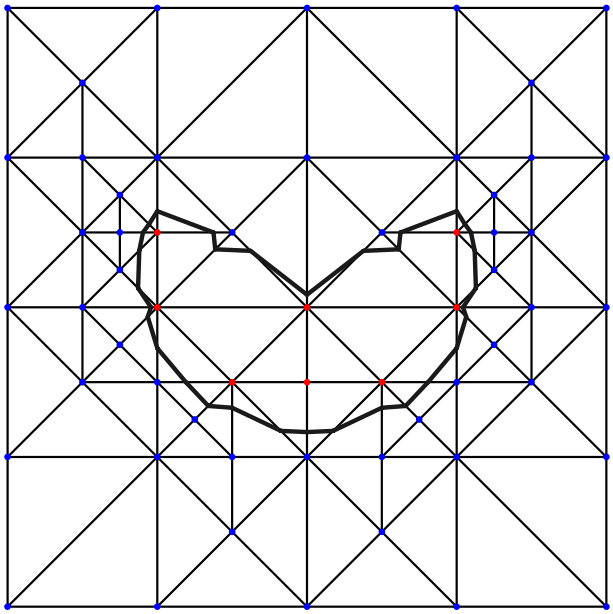


Figura 1.3: Reconstrução linear da curva $C(x, y) = 0$.

principalmente, como assegurar que a curva possui no máximo uma componente dentro de cada triângulo.

Mas vamos considerar mais detalhadamente a figura 1.3. A topologia da curva está bem descrita para essa triangulação, mas a reconstrução linear está muito distante da curva real. Nessa situação, podemos tentar algumas coisas,

tais como refinar mais a triangulação ou recorrer a algum método numérico para resolver a equação $C(x, y) = 0$ restrita a cada triângulo.

É possível ainda abordar o problema de outra forma: supor que, em cada triângulo, $C(x, y)$ pode ser aproximada por uma função polinomial de grau baixo e determinar os coeficientes do polinômio associado a fim de obter uma boa aproximação. O problema dessa abordagem é assegurar que esse modelo polinomial seja bem comportado, isto é, que ele não introduza componentes conexas espúrias ou auto-interseções.

É aí que entra a nossa idéia. Ao invés de aproximar $C(x, y)$ em cada triângulo por uma função polinomial de grau baixo, nós deformamos o espaço dentro de cada triângulo de modo que a reconstrução linear seja levada o mais próximo possível da curva $C(x, y) = 0$. Essa deformação possui três propriedades importantes: (1) ela é bijetiva e contínua dentro de cada triângulo, de modo que a topologia da curva no interior do triângulo não é alterada; (2) ela é compatível com as relações de incidência da triangulação, o que garante que a curva deformada possui a mesma topologia da reconstrução linear; e (3) ela é dada por uma composição de aplicações polinomiais de baixo grau. Em resumo, essa deformação é o que chamamos de *difeomorfismo simplical polinomial*.

Vejamos essas deformações atuando. A figura 1.4 mostra a reconstrução linear de baixa qualidade geométrica, mas topologicamente fiel, que encontramos anteriormente, juntamente com linhas coordenadas associadas às coordenadas baricêntricas de cada triângulo que é interceptado pela curva. A figura 1.5, por sua vez, mostra o efeito do difeomorfismo simplicial atuando na reconstrução linear e nas linhas coordenadas. No capítulo 6, daremos mais detalhes sobre como o difeomorfismo simplicial foi encontrado.

Até aqui, nos restringimos ao caso 1-dimensional, mas vamos mostrar que essa abordagem escala bem para dimensões superiores. Em geral, temos que nossa nova representação requer três ingredientes: um complexo simplicial K, uma função f definida nos vértices de K e um difeomorfismo simplicial X K-invariante, ou seja, que preserva as relações de incidência de K. Para entender essa nova representação deve-se estudar cada um dos objetos K, (K, f) e (K, X). Isso é o que faremos nos próximos capítulos.

Outra questão que devemos estudar é como abstrair as propriedades das tão populares malhas de triângulos para malhas simpliciais de qualquer dimensão, de tal modo que possamos especificar algoritmos de maneira independente da dimensão, ou, de outra forma, *genéricos* com relação a dimensão.

Acreditamos ter achado uma resposta interessante para essa pergunta ao unir as técnicas da *programação genérica* com os fundamentos conceituais da *topologia combinatória*. É importante notar que essa questão não surge apenas de um desejo de abstração de caráter matemático, mas de uma necessidade de especificar algoritmos eficientes de maneira elegante.

Essa resposta incorpora também outro tópico muito importante em aplicações, a noção de *multirresolução adaptativa*, que podemos observar na triangulação do exemplo. Mais precisamente, através do conceito de *esquema de subdivisão estelar*, vamos poder modelar genericamente os algoritmos de refinamento adaptativo mais freqüentes.

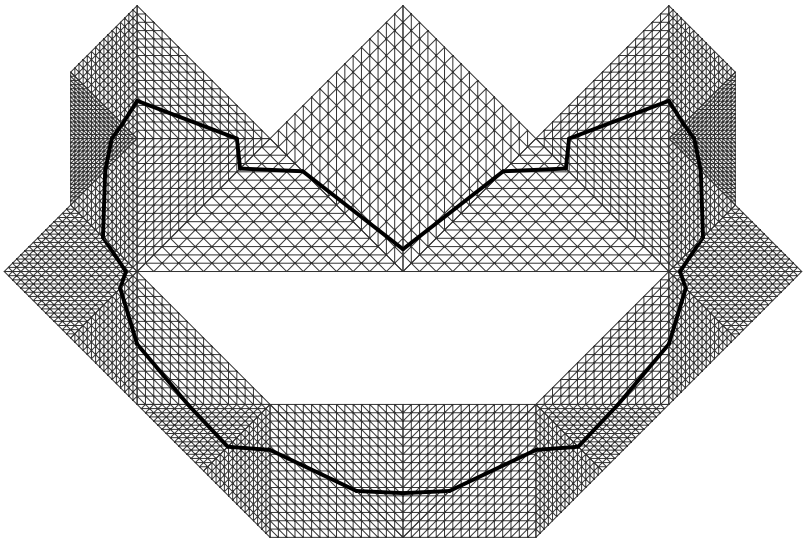


Figura 1.4: Reconstrução linear exibida com as linhas coordenadas.

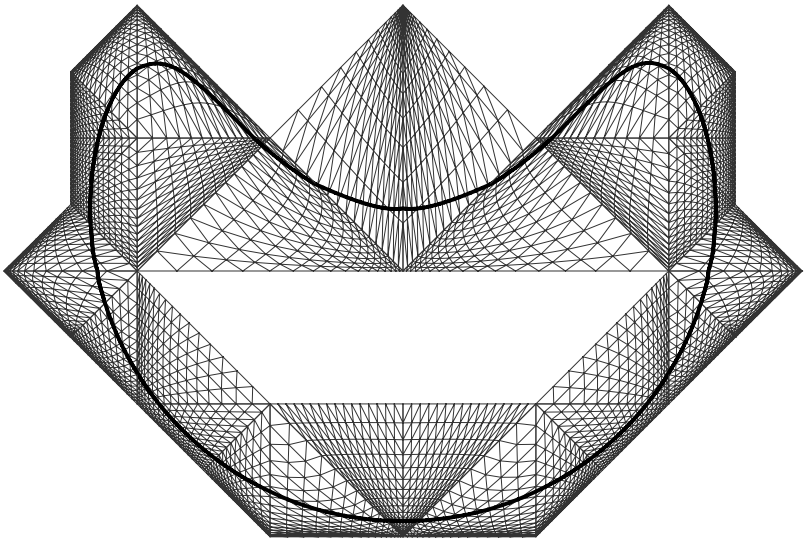


Figura 1.5: Difeomorfismo simplicial atuando sobre a reconstrução linear e as linhas coordenadas.

A tese está organizada da seguinte maneira. No capítulo 2, resumimos os trabalhos relacionados. O capítulo 3 trata de estabelecer o vínculo entre topologia combinatória e programação genérica através de certas noções, tais como complexos semi-simpliciais, pseudovariedades e isocomplexos. No capítulo 4, discutimos a noção de esquema de subdivisão estelar. O capítulo 5 é dedicado aos difeomorfismos simpliciais. Seguem dois capítulos inevitáveis, o de aplicações (Cap. 6), entre elas a de poligonização de curvas implícitas que ilustra esta introdução, e uma breve conclusão (Cap. 7).

Capítulo 2

Trabalhos Relacionados

O objetivo deste capítulo é resumir os trabalhos que, de uma forma ou outra, estão relacionados ao desta tese. A seqüência adotada para expô-los é antes cronológica do que lógica, estando mais ligada à ordem em que apareceram no curso da pesquisa, na medida em que respondiam a dúvidas ou sugeriam caminhos, do que a qualquer tentativa de sistematização.

A motivação inicial da pesquisa que levou a esta tese era criar uma estrutura de dados que representasse malhas hierárquicas de tetraedros, possuisse suporte a subdivisão e tivesse propriedades análogas as das *malhas hierárquicas 4-K* de Velho e Gomes [37]. Naquele momento, conhecíamos somente o trabalho de José Maria Ribeiro Neves sobre *triangulações retangulares* [25], que possuía alguma conexão com esse tema, mas tinha um caráter mais prático e dirigido a aplicações de visualização volumétrica.

Nosso primeiro contato com estruturas de dados espaciais foi o trabalho de Paulo Roma, em artigos como [6]. Aprendemos muito estudando seu código mas percebemos que sua estrutura era demasiadamente geral para nossas aplicações. Precisávamos representar apenas malhas de tetraedros e não subdivisões gerais do espaço.

Ao mesmo tempo, sob influência das biblioteca STL [34] e Boost Graph [33], começamos a pensar nossa estrutura como uma coleção de *containers* que armazenassem os vértices, arestas, faces e tetraedros da malha, juntamente com uma interface funcional para acesso às relações de incidência entre esses elementos. Pesquisando na literatura, chegamos ao conceito de complexo semi-simplicial [21] que se encaixou como uma luva.

Faltava, entretanto, incorporar subdivisão e hierarquia ao modelo. Como deveríamos emular as propriedades das malhas 4-K, nada mais natural que empregar subdivisões em arestas e hierarquia entre os tetraedros.

Até então, estávamos concentrados na implementação da estrutura. Mas quando foi necessário formalizar o conceito, a fim de derivar propriedades e provar a correção de algoritmos, vimos que não possuíamos a ferramenta adequada. É certo que havia o trabalho da "escola de Gênova" sobre multitriangulações [10], entretanto ainda não era exatamente o que queríamos.

Foi por acaso que descobrimos o artigo de Alexander sobre operações es-

telares [1] e imediatamente percebemos que essa era a ferramenta que necessitávamos para orientar o raciocínio, uma vez que podem ocorrer situações não intuitivas já em malhas de tetraedros.

A pesquisa já estava suficientemente madura para gerar o artigo [7], onde tentamos unir programação genérica, multitriangulações e operações estelares, mas ainda restritos ao caso tridimensional, numa aplicação em visualização volumétrica.

Ocorreu uma mudança na direção da pesquisa quando, em uma visita ao VISGRAF, Gabriel Taubin sugeriu que nossas idéias poderiam ser usadas para implementar uma versão tridimensional de um algoritmo de reconstrução de curvas dadas por pontos proposto por ele e Remi Ronfard em [35]. Na verdade, o algoritmo de reconstrução era apenas uma aplicação de um conceito original denominado *Modelos Implícitos Simpliciais*, cujo objetivo era dar uma nova formulação a idéia de objetos implícitos por partes. Nós implementamos esse algoritmo [23] e durante o processo ficamos seduzidos pela idéia de objetos implícitos por partes.

Pesquisando na bibliografia, descobrimos o notável trabalho de Bajaj *et al.* nesse assunto [3, 4]. Ele descobriu condições suficientes para que um *patch* implícito algébrico não possuísse singularidades no interior de um tetraedro. Aparentemente, isso poderia ser desalentador, pois ele de fato havia encontrado uma solução muito boa e não haveria muito mais a fazer. Mas, incomodados com a complexidade das contas envolvidas, resolvemos pensar em uma solução diferente e conceitualmente mais simples.

Foi quando nos veio a mente o trabalho de Sederberg e Parry [31] sobre deformações *free-form*, que havíamos conhecido por meio do trabalho de Coquillart [8]. A idéia era aplicar uma deformação, dada por uma aplicação polinomial, no interior de cada tetraedro e com isso deformar um *patch* linear que estivesse definido no tetraedro através de interpolação linear, como nos modelos implícitos simpliciais de Taubin e Ronfard.

Só havia um problema: a deformação deveria ser necessariamente injetiva, a fim de evitar auto-interseções e o surgimento de componentes conexas espúrias. No artigo "Preventing Self-Intersection under Free-Form Deformation" de Gain e Dodgson [11], encontramos a referência a um critério suficiente para que uma deformação fosse injetiva, devido a Meisters e Olech [22]. Conseguimos adaptar esse critério para o caso de uma função simplicialmente invariante X definida em um simplexo Δ^n, reduzindo o o problema de mostrar que X a injetiva à verificação da positividade do determinante jacobiano de X em Δ^n.

No caso polinomial que nos interessava, o problema era mostrar que certos polinômios eram positivos em Δ^n. Depois de tentar mostrar isso com auxílio de programas de computação simbólica, sem sucesso, por causa do grande número de variáveis do problema, descobrimos que este é um tema muito rico, para o qual já existiam muitos resultados [29, 30], entre os quais um teorema de Pólya, que fornecia critérios necessários e suficientes. Com algum esforço, aplicamos esse critério ao caso em questão e obtivemos um conjunto de resultados parciais que permitiram a implementação de um conceito que unia modelos implícitos simpliciais e deformações *free-form* injetivas.

Nesse ínterim, a estrutura de dados original já havia sido modificada para suportar malhas n-dimensionais. Entretanto, o processo de subdivisão ainda dependia de informações geométricas, ao passo que o esquema das malhas 4-K se baseava apenas em informações combinatórias da malha. Foi quando descobrimos o artigo de Maubach [20], no qual esse problema havia sido resolvido através da análise da triangulação CFK de um hipercubo.

Mais uma vez, incomodados pela complexidade das contas, tentamos provar nós mesmos os resultados de Maubach, seguindo a estratégia de isolar os fatos combinatórios dos geométricos, com bons resultados, pois além de obter uma nova demonstração para a correção do algoritmo de Maubach, descobrimos algumas condições que uma malha deve satisfazer para que tal algoritmo funcione.

Durante a preparação desta tese, foram publicados alguns trabalhos que utilizaram a biblioteca computacional que desenvolvemos. Lewiner *et al.* [15] empregaram-na no desenvolvimento de algoritmos para extração e compressão de malhas hierárquicas de isocomplexos. Já Marroquim *et al.* [19], trabalharam na geração de malhas de tetraedros em multirresolução, adaptadas a domínios espaciais.

Capítulo 3

Programação Genérica e Modelagem Topológica

Programação genérica é um paradigma de programação orientado a algoritmo, que consiste em considerar o algoritmo em primeiro lugar, identificando os conceitos sobre os quais ele opera. O termo "conceito" acima possui um significado preciso: um *conceito* é um conjunto de abstrações computacionais que satisfazem certos requisitos sintáticos e semânticos, conhecidos como *interface* e *propriedades*, respectivamente. Qualquer instância particular de um conceito é chamada de *modelo* e diz-se que tal modelo implementa tal conceito. A idéia básica da programação genérica é que um algoritmo que atue sobre um dado conceito deve ser especificado apenas em termos da interface e das propriedades, sendo portanto aplicável a todos os modelos que implementam esse conceito. Um algoritmo assim especificado é denominado *algoritmo genérico*. Em resumo, parafraseando N. Wirth, podemos dizer que

Generic Algorithms+Concepts=Generic Programming.

Nosso objetivo neste capítulo é aplicar as técnicas da programação genérica à modelagem topológica, termo que utilizamos em analogia à modelagem geométrica, ou seja, a modelagem topológica trata da descrição computacional de objetos topológicos. Especificamente, queremos formular um conceito que capture a noção de malha de triângulos, ou, mais geralmente, de malha n-dimensional. Mas o que entendemos por "malha n-dimensional" ?

A noção de malha em computação gráfica é propositadamente vaga, a fim de acomodar as mais diversas situações. De maneira geral, ela corresponde a noção matemática de *complexo*, a qual admite várias modalidades. *Grosso modo*, um complexo é composto de elementos mais simples colados uns aos outros de tal forma que certas regras de compatibilidade estejam asseguradas. Os diferentes complexos são nomeados segundo o tipo de elemento constituinte, podendo ser complexos simpliciais, cúbicos ou poliedrais, por exemplo, conforme os elementos sejam simplexos, cubos ou poliedros.

Cada um desses complexos tem suas aplicações preferenciais. Complexos poliedrais, por serem fechados com relação às operações de união e interseção,

são bastante empregados em aplicações de CSG. Complexos cúbicos são amplamente utilizados na discretização de EDPs. Complexos simpliciais, por sua vez, são vantajosos em aplicações que envolvam subdivisão e multirresolução. Como essas são as aplicações que nos interessam por hora, uma malha para nós corresponde a noção de complexo simplicial, com algumas restrições adicionais. Tais restrições são necessárias pois complexos simpliciais são objetos muito gerais. Na ausência de informação adicional, a recuperação da vizinhança de um dado simplexo, uma operação fundamental em muitos algoritmos, pode demandar um tempo proporcional ao número de simplexos do complexo, quando seria desejável que esse tempo fosse proporcional ao número de simplexos realmente na vizinhança.

Isso nos leva a um ponto crucial na filosofia da programação genérica: eficiência. Um dos ingredientes da especificação de um conceito é o custo computacional de cada componente da interface, de modo que a complexidade de um algoritmo genérico possa ser calculada de antemão e seja válida para qualquer modelo que implemente esse conceito. O que faremos a seguir é propor um refinamento do conceito de complexo que possui propriedades ótimas no quesito eficiência.

Uma inovação desta tese é a utilização de um objeto conhecido tecnicamente como complexo semi-simplicial na representação de complexos simpliciais. Pretendemos mostrar que tal decisão é a mais adequada ao espírito da programação genérica.

Outro ponto a se destacar deste capítulo é que vamos estudar um certo tipo de objeto muito freqüente em diversas aplicações, mas que até aqui, pelo menos em nosso conhecimento, não havia sido batizado. Trata-se do análogo simplicial das hipersuperfícies e que, por isso mesmo, está sempre presente nos algoritmos de aproximação simplicial de hipersuperfícies, ainda que disfarçado.

O capítulo está dividido em três seções. A primeira trata dos conceitos geométricos, a segunda dos conceitos combinatórios relacionados e a terceira é uma descrição da interface necessária para operar sobre esses conceitos.

3.1 Conceitos Geométricos

3.1.1 Complexos Simpliciais

Um conjunto C de pontos no espaço $\mathbb{R}^m$ está em *posição geral* se os vetores $p_1 - p_0, \ldots, p_n - p_0$ são linearmente independentes, para qualquer seqüência $p_0, p_1, \ldots, p_n$ de pontos em C, com $n \leq m$. O fecho convexo de $n+1$ pontos $p_0, p_1, \ldots, p_n$ em posição geral é denominado *simplexo n-dimensional*, ou *n-simplexo*, e é denotado por $\langle p_0, \ldots, p_n \rangle$. Por convenção, o simplexo vazio tem dimensão -1. Os pontos p_i são chamados *vértices* do simplexo.

Se um ponto p pertence a um simplexo $\sigma = \langle p_0, \ldots, p_n \rangle$, segue diretamente da definição que p pode ser escrito unicamente da forma

$$p = w_0 p_0 + w_1 p_1 + \ldots + w_n p_n,$$

onde $w_i \geq 0$ e $\sum w_i = 1$. Os coeficientes w_i são conhecidos como *coordenadas baricêntricas* de p com relação ao simplexo σ. Dizendo a mesma coisa de outra forma, as coordenadas baricêntricas fornecem uma bijeção entre um n-simplexo σ qualquer e o *simplexo n-dimensional padrão*

$$\Delta^n = \{(w^0, \ldots, w^n) \in \mathbb{R}^{n+1} | w^i \geq 0, \sum_i w^i = 1\},$$

que leva os vértices p_i de σ nos pontos $e_i = (e_i^0, e_i^1, \ldots, e_i^n)$, com $e_i^j = \delta_{ij}$. Dessa caracterização, fica fácil ver que um simplexo é determinado completamente pelos seus vértices, a menos de uma permutação, ou seja, se

$$\langle p_0, \ldots, p_n \rangle = \langle q_0, \ldots, q_n \rangle,$$

então $q_i = p_{\pi(i)}$, para alguma permutação π dos inteiros entre 0 e n.

O fecho convexo de quaisquer $r+1$ vértices de um simplexo $\sigma = \langle p_0, \ldots, p_n \rangle$ é um r-simplexo δ. Dizemos que δ é uma *face* de σ, ou $\delta < \sigma$, em símbolos. Dizemos também que σ e δ são *incidentes*. Dois simplexos σ_1 e σ_2 que se interceptam em uma face comum δ, com $\sigma_1 \neq \delta \neq \sigma_2$, são ditos *adjacentes*.

Estamos prontos para uma importante definição.

Definição 1. *Um* complexo simplicial *é um conjunto K de simplexos do $\mathbb{R}^m$ satisfazendo a duas condições:*

1. *As faces de um simplexo em K também pertencem a K;*
2. *A interseção de dois simplexos de K ou é vazia ou é uma face comum a ambos.*

A *realização geométrica* de um complexo simplicial K, denotada por $|K|$, é o conjunto dos pontos do $\mathbb{R}^m$ que pertencem a ao menos um simplexo de K. Dizemos que K *triangula* um conjunto $C \subset \mathbb{R}^m$ se $C = |K|$. Claramente, um mesmo conjunto pode ter várias triangulações.

A dimensão de um complexo é, por definição, igual a maior das dimensões de seus simplexos.

3.1.2 Isocomplexos

Em computação gráfica, complexos simpliciais surgem essencialmente em duas situações: ou na modelagem direta de objetos, ou na triangulação do domínio de uma função. Uma vez que uma função $f: D \subset \mathbb{R}^m \to \mathbb{R}$ tenha seu domínio triangulado, é possível obter uma aproximação linear por partes $\overline{f}$, a partir da interpolação linear do valor de f nos vértices da triangulação, fornecendo assim uma representação bastante compacta de f. Isosuperfícies de $\overline{f}$, ou seja, conjuntos da forma $\{p \in D | \overline{f}(p) = k\}$, constituem, por sua vez, uma maneira indireta de se modelar objetos.

Vemos assim que complexos simpliciais estão presentes nos dois tipos mais básicos de representação de objetos, a paramétrica ("direta") e a implícita ("indireta"). Para aumentar o paralelismo entre essas representações, vamos mostrar

a seguir que as "isosuperfícies de funções lineares por partes" não são meros conjuntos de pontos, mas, ao contrário, podem ser dotadas de uma estrutura similar a de um complexo simplicial.

Seja $\sigma = \langle p_0, \ldots, p_n \rangle \subset \mathbb{R}^m$ um n-simplexo e $(f_0, \ldots, f_n) \in \mathbb{R}^{n+1}$, com cada $f_i \neq 0$. Ao conjunto

$$\theta = \langle (p_0, f_0), \ldots, (p_n, f_n) \rangle := \{ p \in \sigma | f_0 w_0 + f_1 w_1 + \ldots + f_n w_n = 0 \},$$

onde os w_i são as coordenadas baricêntricas de p com relação a σ, denominamos *isocélula de suporte σ e valores $(f_0, \ldots, f_n)$*. Notamos que uma mesma isocélula pode ter representações distintas. Por exemplo, substituindo-se cada f_i pelo múltiplo αf_i, $\alpha \neq 0$, o conjunto θ não se altera. Também é possível obter outras representações através da alteração do simplexo suporte. Por isso, sempre que nos referirmos a uma isocélula, indicaremos seu suporte e seus valores.

Vamos estudar mais atentamente o conjunto θ. Se os valores f_i são todos maiores que zero, ou todos menores que zero, θ é vazio. Caso contrário, vamos supor que existam $k+1$ valores positivos e $l+1$ valores negativos. Sem perda de generalidade, vamos supor também que os primeiros valores sejam positivos, de modo que podemos definir $f_i^+ := f_i$ e $p_i^+ := p_i$, para $i = 0, \ldots, k$ e $f_i^- := f_{k+1+i}$ e $p_i^- := p_{k+1+i}$, para $i = 0, \ldots, l$, com $f_i^+ > 0$ e $f_i^- < 0$. Ou seja, separamos as partes positivas e negativas do suporte e dos valores da isocélula θ.

Não é difícil mostrar que a aplicação $I: \Delta^k \times \Delta^l \to \sigma$ dada por

$$((w_0^+, \ldots, w_k^+), (w_0^-, \ldots, w_l^-)) \mapsto \alpha \sum_{i=0}^{k} w_i^+ p_i^+ + \beta \sum_{i=0}^{l} w_i^- p_i^-, \qquad (3.1)$$

onde

$$\alpha = \frac{-\sum_{i=0}^{l} f_i^- w_i^-}{\sum_{i=0}^{k} f_i^+ w_i^+ - \sum_{i=0}^{l} f_i^- w_i^-} \text{ e } \beta = \frac{\sum_{i=0}^{k} f_i^+ w_i^+}{\sum_{i=0}^{k} f_i^+ w_i^+ - \sum_{i=0}^{l} f_i^- w_i^-},$$

mapeia o produto $\Delta^k \times \Delta^l$ bijetivamente em θ. Da definição resulta também que θ é convexo e que está contido em um subespaço afim de dimensão $k+l = n-1$ do $\mathbb{R}^m$. Podemos, portanto, definir a *dimensão de uma isocélula θ* como a soma $k+l$, exceto se $k = n$ ou $l = n$, quando convecionamos que θ tem dimensão -1.

As *faces q-dimensionais de uma isocélula $\theta = \langle (p_0, f_0), \ldots, (p_n, f_n) \rangle$* são, por definição, isocélulas

$$\omega = \langle (p_{i_0}, f_{i_0}), \ldots, (p_{i_q}, f_{i_q}) \rangle,$$

onde $\{i_0, \ldots, i_q\} \subset \{0, \ldots, n\}$. A noção de face era o que faltava para a seguinte definição:

Definição 2. *Um* isocomplexo *é um conjunto O de isocélulas do $\mathbb{R}^m$ satisfazendo a duas condições:*

1. *As faces de uma isocélula em O também pertencem a O.*

2. *A interseção de duas isocélulas de O ou é vazia ou é uma face comum a ambas.*

Comparando as definições 1 e 2, vemos que elas são praticamente idênticas, a única diferença sendo o tipo de "célula" usada para estruturar o complexo.

É fácil ver que um isocomplexo O é definido unicamente pelo complexo simplicial K formado pelos suportes das isocélulas em O e por uma função f definida nos vértices de K com valores não nulos. Reciprocamente, o par (K, f) define um isocomplexo. Sendo assim, eventualmente usaremos tal correspondência para nos referir a isocomplexos.

Completamos assim a definição dos dois conceitos geométricos principais com os quais trabalharemos, enfatizando sua semelhança formal. Na próxima seção, vamos definir os análogos combinatórios desses conceitos, ou seja, vamos abstrair as propriedades geométricas desses conceitos e nos concentrar nas noções de incidência e adjacência.

3.2 Conceitos Combinatórios

Nesta seção, em que discutiremos conceitos combinatórios relacionados aos complexos simpliciais e isocomplexos, muitas vezes definiremos objetos no contexto combinatório que podem ser imediatamente associados a objetos geométricos. A fim de abreviar a discussão, deixaremos ao leitor o encargo de realizar tais correspondências.

3.2.1 Complexos Simpliciais Abstratos

Abstraindo as informações geométricas dos complexos simpliciais, chegamos a seguinte definição:

Definição 3. *Dado um conjunto finito V, um* complexo simplicial abstrato *sobre V é um conjunto K de subconjuntos de V satisfazendo as seguintes propriedades:*

1. *$v \in V$ se, e somente se, $\{v\} \in K$;*
2. *Se $\sigma \in K$ e $\delta \subset \sigma$, então $\delta \in K$. Um elemento σ de K é chamado* simplexo *e os subconjuntos próprios de σ são chamados* faces*;*
3. *Existe uma ordem parcial $<$ em V tal que para quaisquer $\sigma \in K$ e $v, v' \in \sigma$, ou $v < v'$ ou $v > v'$, ou seja, elementos de um mesmo simplexo são comparáveis.* [1]

É fácil ver que sempre podemos obter um complexo simplicial abstrato K' a partir de um complexo simplicial geométrico K, basta fixar uma ordem total no conjunto de vértices de K e associar a cada $\sigma = \langle p_0, \ldots, p_n \rangle$ em K o simplexo

[1] O item 3 da definição pode parecer desnecessário, e de fato não é facilmente encontrado na literatura, mas tem uma grande importância na nossa abordagem, pois permite a definição subseqüente de operador de face.

$\sigma' = \{p_0, \ldots, p_n\}$ em K'. Veremos que as relações de incidência são preservadas por esta associação, apenas os aspectos geométricos são "esquecidos". Reciprocamente, é possível mostrar que todo complexo simplicial abstrato pode ser mergulhado em algum $\mathbb{R}^m$, para m suficientemente grande.

Dizemos que $L \subset K$ é um *subcomplexo* de K se L é um complexo sobre os elementos de K em L, com a mesma ordem presente em K.

Aproveitando a notação do caso geométrico, denotaremos um simplexo $\sigma = \{v_0, v_1, \ldots, v_n\}$ em K, com $v_0 < v_1 < \ldots < v_n$, por $\langle v_0, v_1, \ldots, v_n \rangle$.

No que segue, por vezes omitiremos o adjetivo "abstrato", mas sempre estará claro, a partir do contexto, a qual tipo de complexo queremos nos referir.

Um elemento σ de K de cardinalidade $n+1$ é denominado um *n-simplexo* e dizemos que σ tem *dimensão* n, ou, em símbolos, que $d(\sigma) = n$. Simplexos zero e um dimensionais são denominados *vértices* e *arestas*, respectivamente. A dimensão de K é o máximo das dimensões de seus simplexos. Por convenção, o elemento $\emptyset$ de K tem dimensão -1. Adicionalmente, se K tem dimensão m, chamaremos os m-simplexos de *células* e os $(m-1)$-simplexos de *facetas*. Um complexo K é *puro* se todo simplexo σ está contido em alguma célula.

Para cada n, podemos associar um conjunto K_n consistindo dos simplexos n-dimensionais de K. Podemos também definir operadores $\partial_i^n \colon K_n \to K_{n-1}$, $0 \leq i \leq n$, denominados *operadores de face*, da seguinte maneira: seja $\sigma = \langle v_0, v_1, \ldots, v_n \rangle$ um n-simplexo . Então $\partial_i^n(\sigma) = \langle v_0, , \ldots, v_{i-1}, v_{i+1}, \ldots, v_n \rangle$, ou seja, $\partial_i^n(\sigma)$ é obtido eliminando-se o i-ésimo vértice de σ. Quando não houver perigo de confusão, omitiremos o superscrito no símbolo de operador de face. É fácil ver que os operadores de face satisfazem a relação

$$\partial_j^{n-1} \partial_i^n = \partial_{i-1}^{n-1} \partial_j^n, \text{ para } 0 \leq j < i \leq n. \tag{3.2}$$

É possível inverter o raciocínio e tomar a relação (3.2) como base para a definição de um novo objeto:

Definição 4. *Um* complexo semi-simplicial *m-dimensional K é uma seqüência de conjuntos finitos $K_0, K_1, \ldots, K_m$, com K_m não vazio, dotados de operadores $\partial_i^n \colon K_n \to K_{n-1}$, satisfazendo a relação (3.2).*

Um complexo semi-simplicial é um objeto mais geral que um complexo simplicial abstrato, pois todo complexo simplicial abstrato dá origem a um complexo semi-simplicial, como se depreende da discussão acima, mas um complexo semi-simplicial permite simplexos degenerados.

A grande vantagem dos complexos semi-simpliciais é que a natureza dos elementos dos conjuntos K_n não está especificada. Não precisamos pensar neles como conjuntos de vértices. Do ponto de vista da programação genérica, elementos de conjuntos K_n distintos podem implementar conceitos diferentes e podem estar armazenados em diferentes tipos de *containers*.

Nossa estratégia é utilizar complexos semi-simpliciais para representar complexos simpliciais abstratos. Para tanto precisamos apenas transportar a noção de face para o contexto de complexos semi-simpliciais. Podemos fazê-lo indutivamente: seja δ um n-simplexo e σ um m-simplexo com $n < m$. Se $m = n+1$,

δ é uma face de σ se $\partial_i(\sigma) = \delta$, para algum i; caso contrário, δ é uma face de σ se existir um $\tau \in K_{n+1}$ face de σ tal que $\partial_i(\tau) = \delta$, para algum i.

Podemos agora definir a relação fundamental entre simplexos de um complexo, seja ele simplicial ou semi-simplicial. Dois simplexos δ e σ são *incidentes* se δ é uma face de σ ou se σ é uma face de δ.

Portanto, as relações de incidência em um complexo estarão completamente definidas se pudermos responder para cada simplexo σ estas duas perguntas:

1. quais são as faces de σ ?

2. de quais simplexos σ é face ?

Naturalmente estamos interessados em algoritmos que respondam essas perguntas de maneira eficiente. A seguir vamos considerar complexos simpliciais representados por complexos semi-simpliciais.

A primeira pergunta é mais fácil e a definição indutiva de face dada acima já fornece um algoritmo. Se σ é um m-simplexo, todos os n-simplexos que são faces de σ são obtidos através da composição sucessiva de $m-n$ operadores de face aplicadas a σ,

$$\partial_{j_1}^{n+1}\partial_{j_2}^{n+2}\ldots\partial_{j_{m-n}}^{m}\sigma, \tag{3.3}$$

para todas as escolhas possíveis de j_i. Mas algumas dessas escolhas são redundantes. Aplicando repetidamente a relação (3.2) a pares consecutivos de operadores em (3.3) podemos nos limitar a índices j_i que satisfazem

$$0 \leq j_1 < j_2 < \ldots < j_{m-n} \leq m.$$

Vamos denotar por $F(m,n)$ o conjunto das $(m-n)$-uplas de inteiros que satisfazem as desigualdades acima. Observe que percorrer os elementos de $F(m,n)$ é equivalente a percorrer o conjunto das combinações de $m+1$ objetos em grupos de $m-n$, problema extensivamente estudado (ver [14], por exemplo).

Portanto, para enumerar as faces de um m-simplexo σ é suficiente percorrer os elementos de $F(m,n)$ para $0 \leq n < m$ e aplicá-los via (3.3), respondendo assim a primeira pergunta. Antes de passar à segunda pergunta, vejamos um exemplo.

Seja $\sigma = \langle v_0, v_1, v_2, v_3, v_4\rangle$. Vamos aplicar o método descrito acima para percorrer as faces de dimensão 2 e 3 de σ:

$$\begin{array}{ll}
\partial_2\partial_3\partial_4\sigma = \langle v_0, v_1\rangle & \partial_0\partial_1\sigma = \langle v_2, v_3, v_4\rangle \\
\partial_1\partial_3\partial_4\sigma = \langle v_0, v_2\rangle & \partial_0\partial_2\sigma = \langle v_1, v_3, v_4\rangle \\
\partial_1\partial_2\partial_4\sigma = \langle v_0, v_3\rangle & \partial_0\partial_3\sigma = \langle v_1, v_2, v_4\rangle \\
\partial_1\partial_2\partial_3\sigma = \langle v_0, v_4\rangle & \partial_0\partial_4\sigma = \langle v_1, v_2, v_3\rangle \\
\partial_0\partial_3\partial_4\sigma = \langle v_1, v_2\rangle & \partial_1\partial_2\sigma = \langle v_0, v_3, v_4\rangle \\
\partial_0\partial_2\partial_4\sigma = \langle v_1, v_3\rangle & \partial_1\partial_3\sigma = \langle v_0, v_2, v_4\rangle \\
\partial_0\partial_2\partial_3\sigma = \langle v_1, v_4\rangle & \partial_1\partial_4\sigma = \langle v_0, v_2, v_3\rangle \\
\partial_0\partial_1\partial_4\sigma = \langle v_2, v_3\rangle & \partial_2\partial_3\sigma = \langle v_0, v_1, v_4\rangle \\
\partial_0\partial_1\partial_3\sigma = \langle v_2, v_4\rangle & \partial_2\partial_4\sigma = \langle v_0, v_1, v_3\rangle \\
\partial_0\partial_1\partial_2\sigma = \langle v_3, v_4\rangle & \partial_3\partial_4\sigma = \langle v_0, v_1, v_2\rangle
\end{array}$$

Ao analizarmos esse resultado, inferimos um padrão interessante. Está claro que para obter a face $\delta = \langle v_{i_0}, v_{i_1}, \ldots, v_{i_n} \rangle$ de um simplexo $\sigma = \langle v_0, v_1, \ldots, v_m \rangle$, basta considerar o conjunto complementar

$$\{j_1, j_2, \ldots, j_{m-n}\} = \{0, 1, \ldots, m\} \setminus \{i_0, i_1, \ldots, i_n\},$$

com $j_1 < j_2 < \ldots < j_{m-n}$, e aplicar a σ o *operador índice*

$$\sigma[i_0, i_1, \ldots, i_n] := \partial_{j_1} \partial_{j_2} \ldots \partial_{j_{m-n}} \sigma,$$

para o qual adotamos uma notação pós-fixa.

Já para responder à segunda pergunta podemos simplesmente enumerar as faces de cada simplexo τ de dimensão maior que a de σ e verificar se σ está entre elas. Este método é claramente insatisfatório, mas é o único possível na ausência de informação adicional. A seguir vamos obter refinamentos do conceito de complexo simplicial que permitem responder a pergunta 2 de maneira mais eficiente.

3.2.2 Pseudovariedades e Malhas Conformes

Algumas definições são bem-vindas. Dois simplexos σ_1 e σ_2 em um complexo simplicial abstrato K são *independentes* se $\sigma_1 \cap \sigma_2 = \emptyset$. A *junção* $\sigma_1 \star \sigma_2$ de dois simplexos independentes σ_1 e σ_2 é o conjunto $\sigma_1 \cup \sigma_2$. A junção de dois subcomplexos L e L', denotado por $L \star L'$, é o conjunto $\{\sigma \star \tau : \sigma \in L, \tau \in L'\}$. O *link* de um simplexo $\sigma \in K$ é definido por

$$\text{link}(\sigma, K) = \{\tau \in K : \sigma \star \tau \in K\}.$$

Finalmente, a estrela de σ em K, denotado por $\text{st}(\sigma, K)$, é a junção $\sigma \star \text{link}(\sigma, K)$. Diante dessas definições, vemos que uma maneira equivalente de colocar a pergunta 2 é: dado um simplexo σ, como determinar sua estrela ? A definição a seguir é uma restrição natural e útil para responder essa pergunta.

Definição 5. *Um complexo simplicial m-dimensional K é uma* pseudovariedade *se:*

1. *K é um complexo puro, ou seja, todo n-simplexo, com $n < m$, é face de alguma célula;*
2. *toda faceta é incidente a no máximo duas células.*

O item 1 acima implica que cada n-simplexo é face de ao menos um $(n+1)$-simplexo e podemos portanto utilizar essa informação para determinar uma célula inicial a partir da qual uma busca aos outros simplexos incidentes pode ser efetuada. Já o item 2 permite associar à pseudovariedade K um grafo $G(K)$, chamado *grafo de vizinhança de K*, ou *grafo dual*, cujos vértices são as células de K e cujas arestas são as facetas de K com exatamente duas células incidentes. A idéia é utilizar esse grafo para prosseguir a busca dos simplexos incidentes. Mas algumas restrições adicionais devem ser impostas à pseudovariedade para limitar essa busca apenas à estrela do simplexo.

Dizemos que uma pseudovariedade K é *fortemente conexa* se $G(K)$ é conexo. Para recuperar a estrela de um simplexo de maneira eficiente, entretanto, é necessária uma condição mais forte sobre K.

Definição 6. *Dizemos que uma pseudovariedade fortemente conexa K é* hereditária *se a estrela de cada vértice é fortemente conexa, isto é, se $G(\mathrm{st}(v, K))$ é conexo para todo vértice $v \in K_0$. Uma pseudovariedade hereditária é denominada* malha conforme.

É possível provar que se K é uma malha conforme, então $G(\mathrm{st}(\sigma, K))$ é conexo para qualquer n-simplexo de K, para $n \geq 0$ ([1], Teorema [8:2]).

Em resumo, nossa abordagem para responder a pergunta 2 consiste em impor um mínimo de restrições ao conceito de complexo simplicial que sejam suficientes para recuperar a estrela de cada simplexo, através do percorrimento do grafo $G(\mathrm{st}(\sigma, K))$. Vale ressaltar que variedades combinatórias são casos particulares de malhas conformes.

3.2.3 Pseudovariedades Orientadas

Outro conceito importante que devemos modelar é o de orientação. Seja K um complexo simplicial m-dimensional puro. Uma *orientação* de K é uma aplicação $o\colon K_m \to \{-1, +1\}$, ou seja, uma atribuição de um sinal a cada célula. Dizemos que uma pseudovariedade K é *orientável* se existe uma orientação o satisfazendo

$$o(\sigma_1)(-1)^i = -o(\sigma_2)(-1)^j \tag{3.4}$$

para todos os pares (σ_1, σ_2) tais que $\partial_i \sigma_1 = \partial_j \sigma_2$, isto é, células adjacentes induzem orientações opostas em suas facetas comuns. Nesse caso, dizemos que o par (K, o), ou simplesmente K quando a orientação estiver subentendida, é uma *pseudovariedade orientada*.

Uma pseudovariedade m-dimensional K mergulhada em $\mathbb{R}^m$ é sempre orientável, basta escolher a *orientação padrão*

$$o(\sigma) = \operatorname{sgn} \det P_\sigma,$$

onde $\sigma = \langle p_0, p_1, \ldots, p_m \rangle \in K$ e

$$P_\sigma = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ p_0^1 & p_1^1 & \cdots & p_m^1 \\ \vdots & \vdots & \ddots & \vdots \\ p_0^m & p_1^m & \cdots & p_m^m \end{pmatrix}.$$

Para ver isso, suponha que $\sigma_1, \sigma_2 \in K$ sejam tais que $\partial_i \sigma_1 = \partial_j \sigma_2$. Então

$$\det P_{\sigma_1} = (-1)^i \begin{vmatrix} 1 & 1 & \cdots & 1 \\ a^1 & q_1^1 & \cdots & q_m^1 \\ \vdots & \vdots & \ddots & \vdots \\ a^m & q_1^m & \cdots & q_m^m \end{vmatrix}$$

e

$$\det P_{\sigma_2} = (-1)^j \begin{vmatrix} 1 & 1 & \cdots & 1 \\ b^1 & q_1^1 & \cdots & q_m^1 \\ \vdots & \vdots & \ddots & \vdots \\ b^m & q_1^m & \cdots & q_m^m \end{vmatrix},$$

de modo que a e b estão em lados opostos do hiperplano determinado pelos pontos $q_1, \ldots, q_m$. Aplicando uma mudança de coordenadas, se necessário, podemos supor que $a^1 > 0$, $b^1 < 0$ e $q_i^1 = 0$. Efetuando a expansão de Laplace na segunda linha dos determinantes acima, obtemos (3.4).

Uma pseudovariedade orientada fica completamente caracterizada, portanto, pela lista de suas células, acompanhadas do respectivo sinal de orientação. Assim, adotaremos na seqüência uma notação "algébrica", na qual uma pseudovariedade orientada m-dimensional (K, o) é representada pela "soma algébrica" de suas células. Por exemplo, se $m = 3$ e

$$K_3 = \{\sigma_1 = \langle v_0, v_1, v_2, v_3\rangle, \sigma_2 = \langle v_0, v_1, v_2, v_4\rangle\},$$

com $o(\sigma_1) = +1$ e $o(\sigma_2) = -1$, denotaremos

$$K = +\langle v_0, v_1, v_2, v_3\rangle - \langle v_0, v_1, v_2, v_4\rangle.$$

3.2.4 Isocomplexos Abstratos

Finalmente, vamos definir o conceito de isocomplexo abstratamente.

Definição 7. *Seja K um complexo simplicial abstrato e s:$V \to \{+1, -1\}$ uma função definida nos vértices de K. Ao complexo simplicial abstrato O sobre o conjunto $V' = \{(v, s(v))|v \in K\}$, tal que*

$$\langle v_0, \ldots, v_n\rangle \in K \Leftrightarrow \langle (v_0, s(v_0)), \ldots, (v_n, s(v_n))\rangle \in O,$$

denominamos isocomplexo abstrato *com* suporte K *e* valores s.

Um elemento $\theta = \langle (v_0, s(v_0)), \ldots, (v_n, s(v_n))\rangle \in O$ é denominado *isocélula* com suporte $\langle v_0, \ldots, v_n\rangle$ e valores $(s(v_0), \ldots, s(v_n))$, e será denotado por $\langle v_0^{\pm}, \ldots, v_n^{\pm}\rangle$, onde a escolha do sinal $+$ ou $-$ superescrito a v_i dependerá do sinal de $s(v_i)$. Por exemplo, representaremos $\theta = \{(v_0, +1), (v_1, +1), (v_2, -1)\}$ por $\langle v_0^+, v_1^+, v_2^-\rangle$.

A fim de introduzir a noção de dimensão de uma isocélula, será útil definir o par de funções $d^+, d^- : O \to \mathbb{Z}$ dadas por $d^+(\theta) = k$ e $d^-(\theta) = l$, se θ é uma isocélula com $s_i > 0$ para $k + 1$ elementos e $s_i < 0$ para $l + 1$ elementos. A *dimensão* de uma isocélula θ é, por definição, igual a $d^+(\theta) + d^-(\theta)$, se $d^+(\theta) \geq 0$ e $d^-(\theta) \geq 0$, ou igual a -1, caso contrário.

Definida a dimensão de uma isocélula, podemos aproveitar das seções anteriores, *mutatis mutandis*, as definições de vértice, aresta, célula, faceta, complexo puro, pseudovariede e malha conforme. Conseqüentemente, um vértice em O tem a forma $\langle v_0^+, v_1^-\rangle$ ou $\langle v_0^-, v_1^+\rangle$, por exemplo.

Analogamente ao caso simplicial, podemos também definir operadores de face para isocélulas, da seguinte forma: $\partial_i^+\theta = \omega$, onde ω é a face obtida eliminando-se o i-ésimo elemento positivo de θ, e $\partial_i^-\theta = \omega'$ é a face obtida eliminando-se o i-ésimo elemento negativo. Por exemplo, para

$$\theta = \langle v_0^+, v_1^+, v_2^-, v_3^+, v_4^- \rangle,$$

$\partial_0^+\theta = \langle v_1^+, v_2^-, v_3^+, v_4^- \rangle$ e $\partial_1^-\theta = \langle v_0^+, v_1^+, v_2^-, v_3^+ \rangle$. É possível verificar que as seguintes relações são válidas:

$$\partial_j^+\partial_i^+(\theta) = \partial_{i-1}^+\partial_j^+(\theta), \text{ para } 0 \leq j < i \leq d^+(\theta),$$

$$\partial_j^-\partial_i^-(\theta) = \partial_{i-1}^-\partial_j^-(\theta), \text{ para } 0 \leq j < i \leq d^-(\theta) \text{ e}$$
$$\partial_j^-\partial_i^+(\theta) = \partial_i^+\partial_j^-(\theta), \text{ para } 0 \leq i \leq d^+(\theta) \text{ e } 0 \leq j \leq d^-(\theta).$$

O operador índice no caso de isocélulas, por sua vez, é definido por

$$\theta[i_0, i_1, \ldots, i_p]_+[k_0, k_1, \ldots, k_q]_- := \partial_{j_1}^+\partial_{j_2}^+ \ldots \partial_{j_{(d^+(\theta)-p)}}^+ \partial_{l_1}^-\partial_{l_2}^- \ldots \partial_{l_{(d^-(\theta)-q)}}^- \theta,$$

onde

$$\{j_1, j_2, \ldots, j_{(d^+(\theta)-p)}\} = \{0, 1, \ldots, d^+(\theta)\} \setminus \{i_0, i_1, \ldots, i_p\},$$

com $j_1 < j_2 < \ldots < j_{(d^+(\theta)-p)}$, e

$$\{l_1, l_2, \ldots, l_{(d^-(\theta)-q)}\} = \{0, 1, \ldots, d^-(\theta)\} \setminus \{k_0, k_1, \ldots, k_q\},$$

com $l_1 < l_2 < \ldots < l_{(d^-(\theta)-q)}$. Para enumerar os vértices de uma isocélula θ, por exemplo, é suficiente aplicar o operador $[i]_+[k]_-$ a θ, para $0 \leq i \leq d^+(\theta)$ e $0 \leq k \leq d^-(\theta)$, totalizando $(d^+(\theta)+1)(d^-(\theta)+1)$ vértices.

3.2.5 Triangulação de isocélulas

A idéia básica por trás da definição de isocomplexo abstrato é capturar as informações de incidência de um dado isocomplexo geométrico O. Uma alternativa seria triangular O, transformando-o em um complexo simplicial $\mathrm{Tr}(O)$, e a partir daí obter as informações de incidência. Mas essa conversão de representações pode ser custosa, portanto devemos efetuá-la somente quando necessário. A seguir, vamos descrever um procedimento para a triangulação de isocélulas, cuja validade é demonstrada em [13].

Seja $\theta = \langle (p_0, f_0), \ldots, (p_n, f_n) \rangle$ uma isocélula geométrico tal que $d^+(\theta) = k$ e $d^-(\theta) = l$. Seja

$$V = \{\theta[i]_+[j]_- | 0 \leq i \leq k \text{ e } 0 \leq j \leq l\},$$

o conjunto de vértices de θ, o qual ordenamos lexicograficamente, isto é,

$$\theta[i]_+[j]_- < \theta[i']_+[j']_- \Leftrightarrow (i,j) <_{\text{lex}} (i',j').$$

Considere agora o conjunto $\mathcal{L}$ de todos os *caminhos monónotonos* $L = L_0L_1 \ldots L_{k+l}$ num reticulado de $(k+1)\times(l+1)$ pontos, ou seja, cada $L_q = (i_q, j_q)$

é tal que $L_0 = (0,0)$, $L_{k+l} = (k,l)$ e $L_{q+1} = L_q + (1,0)$ ou $L_{q+1} = L_q + (0,1)$. Note que podemos indentificar cada caminho em $\mathcal{L}$ por uma permutação do multiconjunto $\{k \cdot 0, l \cdot 1\}$, onde cada 0 significa um passo "à direita" (incremento da primeira coordenada) e cada 1, um passo "acima" (incremento da segunda). Assim,

$$L_{0110} = (0,0)(1,0)(1,1)(1,2)(2,2),$$

por exemplo. Essa correspondência mostra claramente que existem $\frac{(k+l)!}{k!l!}$ caminhos distintos em $\mathcal{L}$.

O complexo $\mathrm{Tr}(\theta)$ é definido sobre V de modo que exista uma bijeção entre os caminhos de $\mathcal{L}$ e as células de $\mathrm{Tr}(\theta)$, bijeção esta que associa o caminho $(i_0,j_0)(i_1,j_1)\dots(i_{k+l},j_{k+l}) \in \mathcal{L}$ à célula

$$\langle\, \theta[i_0]_+[j_0]_-, \theta[i_1]_+[j_1]_-, \dots, \theta[i_{k+l}]_+[j_{k+l}]_- \,\rangle \in \mathrm{Tr}(\theta).$$

Este complexo $\mathrm{Tr}(\theta)$, então, representa a *triangulação canônica* da isocélula θ.

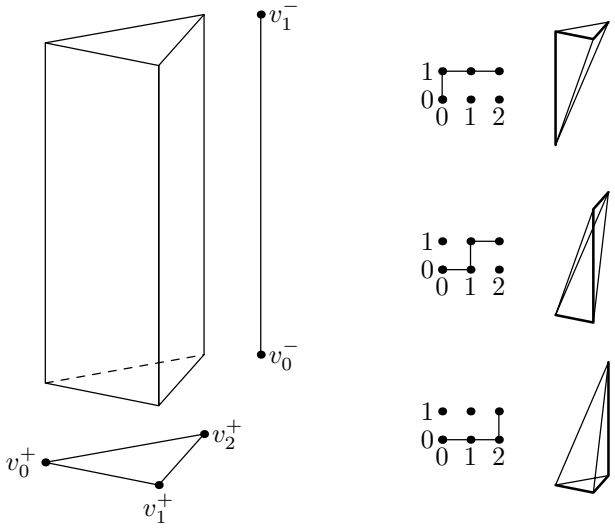


Figura 3.1: Triangulação canônica da isocélula $\theta = \langle v_0^+, v_1^+, v_2^+, v_0^-, v_1^- \rangle$. A cada caminho monótono no reticulado corresponde uma célula de $\mathrm{Tr}(\theta)$.

Em geral, se O é um isocomplexo, $\mathrm{Tr}(O) := \cup_{\theta \in O} \mathrm{Tr}(\theta)$ é uma triangulação de O. Isso segue da compatibilidade da ordenação dos vértices entre células adjacentes.

Resta um detalhe para obtermos uma descrição completa da triangulação $\mathrm{Tr}(O)$ de um isocomplexo abstrato O: a orientação que devemos atribuir às células de $\mathrm{Tr}(O)$, caso o suporte K de O esteja orientado. A fim de chegar a uma formulação combinatória dessa questão, vamos inicialmente analizar esse problema no contexto geométrico.

Seja $\theta = \langle (p_0, f_0), \dots, (p_m, f_m) \rangle$ uma isocélula mergulhada no $\mathbb{R}^m$ com $\sigma = \langle p_0, \dots, p_m \rangle$ como suporte. As arestas de σ são interceptadas por θ em pontos

$$q_{ij} = (1 - w_{ij})p_i + w_{ij}p_j,$$

com

$$w_{ij} = \frac{f_i}{f_i - f_j},$$

onde i e j são tais que f_i e f_j têm sinais opostos. Suponha que alguma triangulação de θ contenha a célula $\delta = \langle q_{i_0 j_0}, q_{i_1 j_1}, \ldots, q_{i_{m-1} j_{m-1}} \rangle$. Os vértices de δ determinam um hiperplano H_δ, que contém θ, cujos pontos satisfazem a equação $\det H_\delta(x) = 0$, onde

$$H_\delta(x) = -\begin{pmatrix} 1 & 1 & \cdots & 1 \\ x^1 & q^1_{i_0 j_0} & \cdots & q^1_{i_{m-1} j_{m-1}} \\ \vdots & \vdots & \ddots & \vdots \\ x^m & q^m_{i_0 j_0} & \cdots & q^m_{i_{m-1} j_{m-1}} \end{pmatrix}.$$

Assim, podemos convencionar que δ tem orientação positiva se os vértices de σ com valores positivos estão no semi-espaço positivo determinado pelo plano H_δ, e negativa caso contrário. É suficiente verificar isso para um único vértice, pois todos os vértices com valores positivos estão, por construção, num mesmo semi-espaço. Obtemos, portanto, a orientação o dada por

$$o(\delta) = (\operatorname{sgn} f_0)(\operatorname{sgn} \det H_\delta(p_0)). \tag{3.5}$$

Podemos simplificar a expressão (3.5) notando que

$$H_\delta(p_0) = -P_\sigma D_\delta,$$

onde

$$D_\delta = \begin{pmatrix} 1 & d_{00} & \cdots & d_{m-1,0} \\ 0 & d_{01} & \cdots & d_{m-1,1} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & d_{0,m} & \cdots & d_{m-1,m} \end{pmatrix},$$

com d_{kl} diferente de zero apenas quando $(k,l) = (k, i_k)$ ou $(k,l) = (k, j_k)$; neste caso $d_{k,i_k} = (1 - w_{k,i_k})$ e $d_{k,j_k} = w_{k,j_k}$. Pelo corolário 20 do capítulo 5, temos que

$$\operatorname{sgn} \det D_\delta = \operatorname{sgn} \det E_\delta,$$

onde E_δ é definida do mesmo modo que D_δ, mas com as entradas positivas substituídas pelo escalar 1. A matriz E_δ é, portanto, uma espécie de matriz de incidência de δ, onde as colunas, com exceção da primeira, representam as arestas de σ interceptadas por δ e as linhas representam os vértices de σ.

Supondo que foi adotada a orientação padrão para σ, temos que

$$o(\delta) = (\operatorname{sgn} f_0)(-\operatorname{sgn} \det P_\sigma)(\operatorname{sgn} \det D_\delta) = -(\operatorname{sgn} f_0) o(\sigma)(\operatorname{sgn} \det E_\delta).$$

A equação acima envolve apenas dados combinatórios de θ e σ e por isso servirá de base para a seguinte definição de orientação de células de uma triangulação de isocélulas abstratas:

$$o(\delta) := -s(\sigma[0]) o(\sigma)(\operatorname{sgn} \det E_\delta). \tag{3.6}$$

No caso particular de um simplexo δ pertencente a triangulação canônica $\mathrm{Tr}(\theta)$ de uma isocélula θ (que, sem perda de generalidade, podemos considerar tal que os vértices positivos do suporte precedem os negativos), é possível encontrar uma expressão simples para o determinante de E_δ. Com efeito, como mencionado anteriormente, a cada simplexo $\delta \in \mathrm{Tr}(\theta)$ corresponde uma permutação $A = (a_1, a_2, \ldots, a_{k+l})$ do multiconjunto $\{k \cdot 0, l \cdot 1\}$, de modo que a cada matriz E_δ corresponde uma matriz E_A. Pela definição de E_A, temos que a i-ésima linha de E_A possui todas as entradas nulas, com exceção da última, onde $i = k + 1$, caso a_{k+l} seja igual a 0, e $i = (k + 1) + (l + 1)$, caso a_{k+l} seja igual a 1. Assim, efetuando a expansão de Laplace na i-ésima linha, verifica-se que

$$\det E_{(a_1,\ldots,a_{k+l})} = \det E_{(a_1,\ldots,a_{k+l-1})}(-1)^s$$

onde

$$s = ((k+1+l+1)+(k+1+l+1))[a_{k+l}=1]+$$
$$((k+1+l+1)+k+1)[a_{k+l}=0]$$

donde resulta que

$$\det E_{(a_1,\ldots,a_{k+l})} = \det E_{(a_1,\ldots,a_{k+l-1})}(-1)^{(l+1)[a_{k+l}=0]} =$$

$$\det E_{(a_1,\ldots,a_{k+l-1})}(-1)^{(\sum_{i<k+l}[a_i=1])[a_{k+l}=0]}(-1)^{[a_{k+l}=0]}.$$

Aplicando essa equação recursivamente, conclui-se que

$$\det E_A = (-1)^{\mathrm{inv}(A)}(-1)^k,$$

onde $\mathrm{inv}(A)$ denota o número de inversões da permutação A.

Para finalizar esta seção, vamos dar um exemplo completo de triangulação de um isocomplexo. Considere o isocomplexo O mostrado na figura 3.2. Ele é composto de duas células

$$\theta = \langle (p_0, -1), (p_1, +3), (p_2, -6), (p_3, +3) \rangle$$
e
$$\theta' = \langle (p_{0'}, -3), (p_1, +3), (p_2, -6), (p_3, +3) \rangle,$$

com suportes $\sigma = \langle p_0, p_1, p_2, p_3 \rangle$ e $\sigma' = \langle p_{0'}, p_1, p_2, p_3 \rangle$, respectivamente. Os seis vértices q_{ij} são obtidos por interpolação linear, ou seja,

$$q_{ij} = \frac{-f_j}{f_i - f_j} p_i + \frac{f_i}{f_i - f_j} p_j,$$

para os i e j, com $i < j$ tais que $f_i f_j < 0$. Pela regra da mão direta, $o(\sigma) = +1$ e $o(\sigma') = -1$.

Abstratamente, substituindo os pontos p_i por vértices v_i e definindo $s(v_i) = \operatorname{sgn} f_i$, temos que $\theta = \langle v_0^-, v_1^+, v_2^-, v_3^+ \rangle$, $\theta' = \langle v_{0'}^-, v_1^+, v_2^-, v_3^+ \rangle$. Aplicando o algoritmo de triangulação, obtemos as seguintes células para $\mathrm{Tr}(O)$:

$$\begin{array}{rcll}
\delta_1 &=& \langle \theta[0]_+[0]_-, \theta[1]_+[0]_-, \theta[1]_+[1]_- \rangle &= \langle \langle v_0^-, v_1^+ \rangle, \langle v_0^-, v_3^+ \rangle, \langle v_2^-, v_3^+ \rangle \rangle \\
\delta_2 &=& \langle \theta[0]_+[0]_-, \theta[0]_+[1]_-, \theta[1]_+[1]_- \rangle &= \langle \langle v_0^-, v_1^+ \rangle, \langle v_1^+, v_2^- \rangle, \langle v_2^-, v_3^+ \rangle \rangle \\
\delta_1' &=& \langle \theta'[0]_+[0]_-, \theta'[1]_+[0]_-, \theta'[1]_+[1]_- \rangle &= \langle \langle v_{0'}^-, v_1^+ \rangle, \langle v_{0'}^-, v_3^+ \rangle, \langle v_2^-, v_3^+ \rangle \rangle \\
\delta_2' &=& \langle \theta'[0]_+[0]_-, \theta'[0]_+[1]_-, \theta'[1]_+[1]_- \rangle &= \langle \langle v_{0'}^-, v_1^+ \rangle, \langle v_1^+, v_2^- \rangle, \langle v_2^-, v_3^+ \rangle \rangle
\end{array}$$

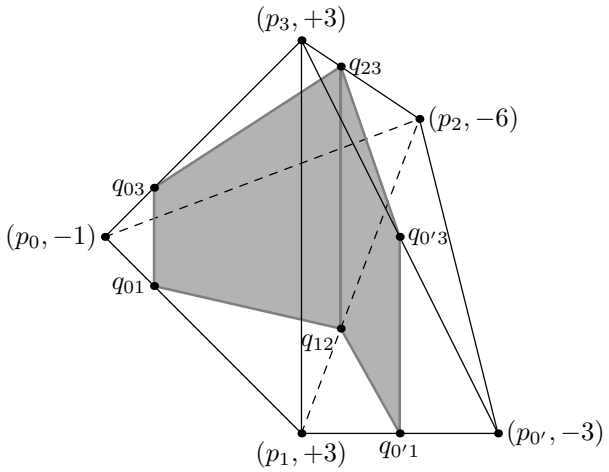


Figura 3.2: Isocomplexo O.

Com respeito à orientação, temos que

$$E_{\delta_1} = E_{\delta_1'} = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix} \quad \text{e} \quad E_{\delta_2} = E_{\delta_2'} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Logo, $o(\delta_1) = -1$, $o(\delta_2) = +1$, $o(\delta_1') = +1$ e $o(\delta_2') = -1$, resultados que são coerentes com a geometria exibida na figura 3.3.

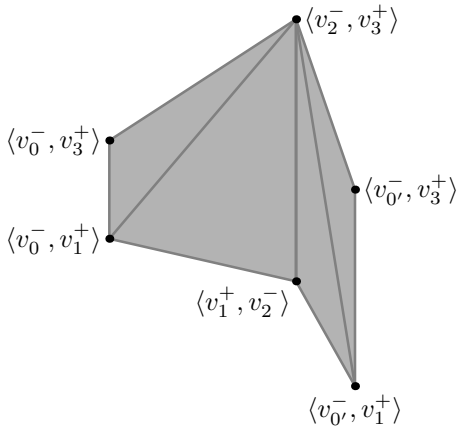


Figura 3.3: Triangulação $\mathrm{Tr}(O)$.

3.3 Interface

Vamos descrever agora a interface computacional que utilizamos para operar com os objetos definidos nas seções anteriores. Antes, porém, vamos discorrer

rapidamente sobre um recurso da linguagem C++ que usamos extensivamente e que, de certa forma, é responsável pela uniformidade da interface com relação a dimensão dos objetos representados.

É bem conhecido por todos os usuários da biblioteca STL [34] que a linguagem C++ suporta *tipos parametrizados*. É possível, por exemplo, criar vetores de inteiros (vector<int>) ou vetores de *doubles* (vector<double>), onde os tipos int e double funcionam como parâmetros do *container* vector. Neste caso, os parâmetros simplesmente determinam o tipo de dado que é armazenado no *container*. Mas os parâmetros podem ter muitas outras finalidades, como se pode ver nas bibliotecas BOOST [33] e LOKI [2].

O que talvez não seja bem conhecido é que tipos podem ser parametrizados não só por outros tipos, como também por valores. Um exemplo é o *container* array da biblioteca BOOST que possui a seguinte definição:

```
template<class T, std::size_t N>
class array {
  public:
  T elems[N];      // fixed-size array of elements of type T
  ...
};
```

Assim, array<int,10> declara um vetor de inteiros que possui exatamente 10 entradas. O valor de N é usado internamente para realizar verificação de faixa (*range checking*). Nossa idéia é usar uma técnica semelhante para parametrizar a dimensão de simplexos e malhas.

Da mesma forma que iteradores são usados na biblioteca STL para isolar os algoritmos de detalhes da implementação dos *containers*, empregamos duas famílias de *tipos associados* a uma malha T, como pode ser visto na listagem 3.1, para este mesmo fim. Ambas famílias são parametrizdas pelo tipo da malha e por um inteiro, da mesma forma que o *container* array. O tipo simplex_iterator<T,d> representa um iterador para o conjunto de simplexos d-dimensionais de T. Quando um simplex_iterator<T,d> é dereferenciado, o resultado é um simplex_descriptor<T,d> que representa um simplexo d-dimensional de T. Além das operações requeridas nas interfaces exibidas na listagem, é assumido que variáveis desses tipos possam ser passadas por valor sem *overhead*.

Outras informações sobre o tipo T, necessárias em tempo de compilação, são armazenados na estrutura sc_traits<T>, entre elas a dimensão e a categoria da malha T, ou seja, se T modela um complexo semi-simplicial, um complexo puro, uma pseudovariedade ou uma multitriangulação. Daremos mais detalhes sobre isso no próximo capítulo.

Para abreviar o código, definimos uma série de macros, apresentadas na listagem 3.2.

Vamos agora detalhar a interface de cada conceito. Devemos notar que todos os requisitos a seguir são basicamente operações de acesso imediato, portanto a *complexidade de cada uma dessas operações deve ser constante*.

A listagem 3.3 resume os requerimentos de um complexo semi-simplicial. A função face_op nada mais é que a versão computacional do operador ∂_i, en-

```
namespace vgtl {

  // Dimension type
  typedef std::size_t dim_t;

  // Simplex descriptor type
  template <class T, dim_t d>
  struct simplex_descriptor {
    simplex_descriptor();
    simplex_descriptor&
      operator=(const simplex_descriptor& s);
    bool operator<(const simplex_descriptor& s) const;
    bool operator==(const simplex_descriptor& s) const;
  };

  // Simplex iterator type
  template <class T, dim_t d>
  struct simplex_iterator {
    simplex_iterator();
    simplex_iterator&
      operator=(const simplex_iterator& s);
    simplex_iterator& operator++();
    simplex_descriptor<T,d> operator*();
    bool operator==(const simplex_iterator& i) const;
  };

  // Simplicial Complex trait class
  template <class T>
  struct sc_traits {
    static const dim_t dim=T::dim;
    static const dim_t split_dim=T::split_dim;
    typedef typename T::sc_category sc_category;
  };
}
```

Listagem 3.1: Tipos associados.

```
#define Dim(T) vgtl::sc_traits<T>::dim
#define Simplex(T,N) vgtl::simplex_descriptor<T,N>
#define Vertex(T) Simplex(T,0)
#define Edge(T) Simplex(T,1)
#define Facet(T) Simplex(T,Dim(T)-1)
#define Cell(T) Simplex(T,Dim(T))
#define Simplex_it(T,N) vgtl::simplex_iterator<T,N>
#define Vertex_it(T) Simplex_it(T,0)
#define Edge_it(T) Simplex_it(T,1)
#define Facet_it(T) Simplex_it(T,Dim(T)-1)
#define Cell_it(T) Simplex_it(T,Dim(T))
#define SplitDim(T) vgtl::sc_traits<T>::split_dim
#define SplitSimplex(T) Simplex(T,SplitDim(T))
```

Listagem 3.2: Macros auxiliares.

```
// SimplicialComplexConcept

  Simplex(T,k-1)
  face_op(const T& t, Simplex(T,k) s, int i);
// (0 < k ≤ Dim(T) )
// Retorna ∂_i s.

  void simplices(const T& t,
                 Simplex_it(T,k)& sbegin,
                 Simplex_it(T,k)& send);
// Atualiza por referência o par de iteradores
// (sbegin, send) para os k-simplexos de T.

  bool empty(const T& t, Simplex(T,k) s);
// Testa se s representa um simplexo vazio.
```

Listagem 3.3: Conceito de Complexo Simplicial.

quanto a função simplices retorna iteradores para os simplexos. A função empty apenas testa se um descritor de simplexo representa um simplexo vazio. Como isso é feito é uma questão de implementação, mas, tipicamente, um descritor é representado internamente por um ponteiro ou por um índice inteiro. Sendo assim, é razoável que um simplexo vazio seja representado por um ponteiro nulo ou por um índice inválido, respectivamente.

Com esses requerimentos, já é possível especificar algoritmos que recuperam as relações de incidência em um complexo. Um primeiro passo é sobrecarregar a função face_op, a fim de permitir o acesso as faces de qualquer dimensão, como se vê na listagem 3.4.

O algoritmo para teste de pertinência exibido na listagem 3.5 simplesmente percorre todas as combinações válidas de trás para frente até encontrar uma face do simplexo sg que seja igual a ss. Note como a dimensão dos simplexos é usada em tempo de compilação para o tratamento de casos específicos.

```
template <class T, dim_t k, dim_t l>
Simplex(T,k-l)
face_op(const T& t,
        Simplex(T,k) s,
        const array<dim_t,l>& a) {
  const array<dim_t,l-1>& b=
    reinterpret_cast<const array<dim_t,l-1>&>(a[1]);
  return face_op(t,face_op_(t,s,b),a[0]);
}

template <class T, dim_t k>
Simplex(T,k-1)
face_op(const T& t,
        Simplex(T,k) s,
        const array<dim_t,1>& a) {
  return face_op(t,s,a[0]);
}
```

Listagem 3.4: Composição de operadores de face.

```
  template <dim_t k, dim_t l, class T>
  typename enable_if<greater_c<k,l>,bool>::type
  in(const T& t,
     Simplex(T,k) sg,
     Simplex(T,l) ss) {
    array<int,k-l> v;
    for(int i=(l+1); i<(k+1); ++i) v[i-l-1]=i;
    do {
      if(face_op(t,sg,v)==ss) return true;
    } while(prev_combination(k+1,v.begin(),v.end()));
    return false;
  }

  template <dim_t k, dim_t l, class T>
  typename enable_if<lesser_c<k,l>,bool>::type
  in(const T& t,
     Simplex(T,k) sg,
     Simplex(T,l) ss) {
    return false;
  }

  template <dim_t k, dim_t l, class T>
  typename enable_if<equal_c<k,l>,bool>::type
  in(const T& t,
     Simplex(T,k) sg,
     Simplex(T,l) ss) {
    return sg==ss;
  }
```

Listagem 3.5: Incidência.

```
// PureSimplicialComplexConcept
// Refina SimplicialComplexConcept

  Simplex(T,k+1)
  up_simplex(const T& t, Simplex(T,k) s);
// (0 ≤ k < Dim(T) )
// Retorna um k+1-simplexo u, tal que ∂_i u = s, para
// algum i.

  bool is_current(const T& t, Cell(T) c);
// Testa se c pertence a triangulação corrente.
```

Listagem 3.6: Conceito de Complexo Simplicial Puro.

```
template <class T, dim_t k>
typename disable_if<cell_c<T,k>,Cell(T)>::type
top_cell(const T& t, Simplex(T,k) s) {
  return top_cell(t,up_simplex(t,s));
}

template <class T, dim_t k>
typename enable_if<cell_c<T,k>,Cell(T)>::type
top_cell(const T& t, Simplex(T,k) s) {
  return s;
}
```

Listagem 3.7: Top cell.

A interface para complexos puros, listagem 3.6, adiciona dois novos requisitos. A função up_simplex(t,s) retorna um simplexo u tal que $\partial_i u = s$. Aplicando recursivamente essa função, obtemos o algoritmo top_cell, listado em 3.7. Note como a informação da dimensão é usada para parar a recursão. Já a função is_current testa se uma célula pertence a malha corrente, conceito útil em várias situações.

O conceito de pseudovariedade, exibido na listagem 3.8, refina o de complexo puro, basicamente com a introdução do requisito cells(t,f), que retorna o par de células incidentes à faceta f. Se a faceta f é incidente a apenas uma célula, ou seja, se f é uma *faceta de bordo*, convecionamos que os dois componentes do par são iguais a célula incidente. A função orientation(t,c) retorna a orientação da célula c e as funções mark e mark_set são úteis em algoritmos de percorrimento do grafo de vizinhança de uma pseudovariedade. Para tornar o código de tais algoritmos mais legível, definimos algumas funções auxiliares, constantes da listagem 3.9.

Outra operação bastante comum em uma pseudovariedade é a que retorna para cada célula σ, a célula $\mathrm{adj}_i\,\sigma$, isto é, a única célula da triangulação que compartilha a faceta $\partial_i\sigma$ com σ. O algoritmo que implementa essa operação aparece na listagem 3.10.

```
// PseudomanifoldConcept
// Refina PureSimplicialComplexConcept
  pair<Cell(T),Cell(T)>
  cells(const T& t, Facet(T) f);
// Retorna as células incidentes a f. Se f é uma faceta
// de bordo, (c1,c2)=cells(t,f) ⇒ c1==c2.

  int orientation(const T& t, Cell(T) c);
// Retorna a orientação de c.

  enum mark_type {white_mark, black_mark, gray_mark};

  mark_type mark(const T& t, Cell(T) c);
// Retorna a cor da célula c

  void mark_set(const T& t, Cell(T) c, mark_type m);
// Atribui a cor m à célula c
```

Listagem 3.8: Conceito de Pseudovariedade.

```
  template <class T>
  void visit(const T& t, Cell(T) ce) {
    mark_set(t,ce,black_mark);
  }

  template <class T>
  void unvisit(const T& t, Cell(T) ce) {
    mark_set(t,ce,white_mark);
  }

  template <class T>
  bool visited(const T& t, Cell(T) ce) {
    return (mark(t,ce)!=white_mark);
  }
```

Listagem 3.9: Visit.

```
  template <class T>
  Cell(T) adjacent(const T& t, Cell(T) ce, Facet(T) f) {
    pair<Cell(T),Cell(T)> ces=cells(t,f);
    if(ces.first==ce) return ces.second;
    else return ces.first;
  }

  template <class T>
  Cell(T) adjacent(const T& t, Cell(T) ce, int i) {
    return adjacent(t,ce,face_op(t,ce,i));
  }
```

Listagem 3.10: Adjacent.

```
template <class T, dim_t k, class Output>
typename disable_if<facet_or_cell_c<T,k>,void >::type
star(const T& t, Simplex(T,k) s, Output o) {
  queue<Cell(T)> st;
  queue<Cell(T)> to_visit;
  to_visit.push(top_cell(t,s));
  while(!to_visit.empty()) {
    Cell(T) c=to_visit.front();
    to_visit.pop();
    if(!visited(t,c)) {
      st.push(c);
      visit(t,c);
      for(int i=0;i<=Dim(T);++i) {
        Facet(T) f=face_op(t,c,i);
        Cell(T) ca=adjacent(t,c,f);
        if((!visited(t,ca))&&in(t,f,s)) to_visit.push(ca);
      }
    }
  }
  while(!st.empty()) {
    Cell(T) c=st.front();
    st.pop();
    unvisit(t,c);
    *o=c; ++o;
  }
}
```

Listagem 3.11: Estrela.

Um exemplo mais complexo de algoritmo que faz sentido em pseudovariedades, mais precisamente em malhas conformes, é o que recupera a estrela de um dado simplexo (listagem 3.11). Dado um simplexo s, o algoritmo star executa uma busca em largura no grafo $G(\mathrm{st}(s,t))$. O fato deste grafo ser conexo garante que o algoritmo retorna todas as células incidentes a s.

Por fim, os atributos geométricos básicos dos complexos simplicias e dos isocomplexos podem ser acessados através da interface exibida na listagem 3.12.

Além dos algoritmos já descritos, implementamos muitos outros, inclusive algoritmos para criação e alteração de malhas, em particular um algoritmo genérico para triangulação de isocomplexos. Tais algoritmos necessitam que o tipo T implemente outras funções que são responsáveis pela criação de elementos e pela atualização de dados. A descrição destes requisitos tomaria muito tempo, mas não há nenhum segredo. Quem estiver interessado pode consultar a documentação específica da biblioteca.

Por fim, vamos descrever brevemente como criar uma estrutura vc parametrizável pela dimensão e que implemente todos os requisitos acima. A idéia básica está descrita na listagem 3.13. Quando o tipo bs<T,k> (o nome vem de *basic simplex*) vai ser instanciado, uma série de testes envolvendo k e Dim(T) é efetuada em tempo de compilação, a fim de determinar que campos estarão

```
// Geometrical Requirements
  point<Dim(T),double>
  euclidean_point(const T& t, Vertex(T) v);
// Retorna as coordenadas do vértice v.

  int signal(const T& t, Vertex(T) v);
// Retorna o signal associado ao vértice v.

  double scalar_value(const T& t, Vertex(T) v);
// Retorna o valor escalar associado ao vértice v.
```

Listagem 3.12: Requisitos geométricos.

presentes. Por exemplo, suponha que Dim(T)==3 e k==2. Como k é diferente de 0, bs<T,2> contém o campo fop; como k é igual a Dim(T)−1, bs<T,2> contém o campo cov. Em resumo, bs<T,2> tem a seguinte forma:

```
  struct bs<T,2,Data> : Data<2> {
    Simplex(T,1) fop[3];
    Simplex(T,3) cov[2];
  };
```

Ou seja, um bs<T,2> contém referências para suas três arestas e para as duas células incidentes, além dos atributos presentes na estrutura Data<2>.

A classe vc pode agora ser definida de modo a conter uma seqüência de vetores de simplexos (listagem 3.14). Quando a classe vc<d> vai ser instanciada, recursivamente são instanciadas as classes vc_rep, que contêm vetores de simplexos básicos. Para d==3, obtemos a seguinte estrutura:

```
  struct vc<3,Data> : Data<4> {
    vector<bs<vc<3,Data>,3,Data> > vc_rep<3,3,Data>::a;
    vector<bs<vc<3,Data>,2,Data> > vc_rep<2,3,Data>::a;
    vector<bs<vc<3,Data>,1,Data> > vc_rep<1,3,Data>::a;
    vector<bs<vc<3,Data>,0,Data> > vc_rep<0,3,Data>::a;
  };
```

Neste caso, utilizamos o *containter* vector para armazenar os simplexos, mas poderíamos ter usado qualquer outro. O *container* list, por exemplo, seria mais interessante caso a aplicação de interesse envolvesse deleções de simplexos. Ou seja, a escolha do *container* depende da aplicação, o que importa é dar ao usuário liberdade de escolha, mantendo a interface básica.

3.4 Conclusão

Embora tenhamos adotado uma ordem lógica para a apresentação do conteúdo deste capítulo, na qual percorremos o caminho que desce dos conceitos matemáticos abstratos para o concreto mundo dos algoritmos e estruturas de dados, temos que admitir que na realidade o caminho aconteceu quase inteiramente no sentido contrário. A motivação inicial era a implementação de uma aplicação

```
template <class T, dim_t k>
struct bs_faces {
  Simplex(T,k-1) fop[k+1];
};

template <class T>
struct bs_faces<T,0> {
};

template <class T, dim_t k, bool t=(k<(Dim(T)-1))>
struct bs_up {
  Simplex(T,k+1) up;
};

template <class T, dim_t k>
struct bs_up<T,k,false> {
};

template <class T, dim_t k, bool t=(k!=Dim(T))>
struct bs_cell {
};

template <class T, dim_t k>
struct bs_cell<T,k,false> {
  bool cur : 1;
  bool ori : 1;
  mark_type mk : 2;
  short lvl: 16;
  bs_cell() : cur(true), mk(white_mark),
    ori(false), lvl(0) {}
};

template <class T, dim_t k, bool t=(k!=(Dim(T)-1))>
struct bs_facet {
};

template <class T, dim_t k>
struct bs_facet<T,k,false> {
  Cell(T) cov[2];
  bs_facet() {};
};

template <class T, dim_t k,
          template <dim_t> class Data=extra_data>
struct bs : Data<k>, bs_faces<T,k>,
    bs_cell<T,k>, bs_facet<T,k>, bs_up<T,k> {
};
```

Listagem 3.13: Implementação.

```
template <dim_t d,
          template <dim_t> class Data>
struct vc;

template <dim_t k, dim_t d,
          template <dim_t> class Data>
struct vc_rep : vc_rep <k-1,d,Data> {
  vector<bs<vc<d,Data>,k,Data> > a;
};

template <dim_t d,
          template <dim_t> class Data>
struct vc_rep<0,d,Data> {
  vector<bs<vc<d,Data>,0,Data> > a;
};

template <dim_t d,
          template <dim_t> class Data=extra_data>
struct vc : vc_rep<d,d,Data>, Data<d+1> {
};
```

Listagem 3.14: Implementação.

para visualização de malhas de tetraedros. Duas influências guiaram a implementação: de uma parte, a biblioteca STL nos fez pensar em uma malha como um conjunto de *containers* no qual são armazenados os atributos dos simplexos; da outra parte, a maneira como a biblioteca BOOST GRAPH relaciona as arestas e vértices de um grafo por meio de uma interface funcional, nos inspirou a fazer o mesmo para os simplexos da malha. Foi pesquisando a maneira correta de fazer isso que descobrimos na literatura o conceito de complexo semi-simplicial, que se encaixou como uma luva. Com um pouco mais de intimidade com as técnicas da programação genérica, chegamos a estrutura de dados exibida na listagem 3.13, que é a base para a implementação de complexos semi-simpliciais n-dimensionais.

A inspiração para o conceito de isocomplexo veio do trabalho de Taubin e Ronfard sobre *Modelos Implícitos Simpliciais* [35]. Nós apenas enfatizamos a semelhança formal entre complexos simpliciais e isocomplexos, além de dar um tratamento mais abstrato. Já o algoritmo de triangulação de isocomplexos foi programado inicialmente com uma construção recursiva mas, posteriormente, na tentativa de justificar esta construção, descobrimos em [13] a construção baseada em reticulados.

Entre os trabalhos anteriores sobre modelagem baseada em complexos simpliciais em dimensão arbitrária destacamos [28]. Diferentemente da nossa abordagem, entretanto, a estrutura de dados apresentada nesse artigo não codifica todas relações de incidência, mas somente as relações de adjacência entre células.

Fora do mundo simplicial, existem muitas estruturas de dados para a modelagem de objetos topológicos, dentre as quais citamos a *cell-tuple* de Brisson

[5] e os *G-Maps* de Lienhardt [17]. Seria interessante, em um trabalho futuro, aplicar algumas técnicas de programação genérica que utilizamos aqui a estas estruturas de dados.

A principal objeção com respeito a utilização de complexos simpliciais para a representação de objetos topológicos reside em sua "verbosidade", ou seja, um grande número de simplexos é necessário para se representar certos objetos, mesmo em casos bem simples. Isso justifica a existência das várias estruturas de dados para complexos não simpliciais. Enquanto não seja possível refutar completamente essa objeção, pretendemos mostrar que, no caso de isocomplexos, este problema pode ser amenizado com a introdução de "deformações" associadas ao suporte de cada isocélula, dando origem ao conceito de isocomplexo curvilinear. Mas antes de tratar deste assunto, vamos discutir no próximo capítulo duas razões que explicam por que os complexos simpliciais são ainda bastante usados, apesar de todas as objeções: subdivisão e multirresolução.

Capítulo 4

Esquemas de Subdivisão Estelar

Freqüentemente, seja no estudo das propriedades topológicas das malhas, seja em aplicações, é necessário aplicar modificações nas malhas que diminuam ou aumentem sua complexidade, isto é, a quantidade total de simplexos, mas que não alterem determinadas propriedades. Um exemplo de tais modificações são as *operações estelares*. De fato, muitos dos conceitos da topologia combinatória estão fundamentados nas operações estelares [16]. Seja K um complexo sobre o conjunto de vértices V, K' um complexo sobre $V' = V \cup \{v\}$ e σ um simplexo de K. A operação que transforma K em K', removendo $\mathrm{st}(\sigma, K)$ e substituindo-o por $v \star \partial\sigma \star \mathrm{link}(\sigma, K)$, onde $\partial\sigma$ denota o *bordo* de σ, ou seja, o complexo formado pelas faces próprias de σ, é denominada *subdivisão estelar* e é denotada por $K' = K(\sigma, v)$. A operação inversa $(\sigma, v)^{-1}$ que transforma K' em K é chamada *fusão estelar*. Estas operações estão representadas na figura 4.1.

Vejamos rapidamente como operações estelares são utilizadas em topologia combinatória. Dois complexos são *estelar equivalentes* se estão relacionados por uma seqüência de operações e stelares. Uma *n-bola estelar* é um complexo estelar equivalente a um n-simplexo e uma *n-esfera estelar* é um complexo estelar equivalente ao bordo de um $(n+1)$-simplexo. Um complexo simplicial M é uma *variedade estelar n-dimensional* se para cada vértice $v \in M$, $\mathrm{link}(v, M)$ é uma $(n-1)$-esfera estelar ou uma $(n-1)$-bola estelar. Um resultado básico da teoria estelar, o teorema de Newman, afirma que se existe um homeomorfismo linear por partes entre dois complexos simpliciais, então eles são estelar equivalentes, e vice-versa, estabelecendo assim uma correpondência entre a teoria estelar (mais combinatória) e a linear por partes (mais geométrica), a qual permite o trânsito entre essas duas abordagens, de acordo com a conveniência. Conseqüentemente, o conceito de variedades estelar é essencialmente equivalente ao conceito de variedade linear por partes.

É possível mostrar também que as propriedades topológicas de um complexo K são preservadas por operações estelares. Em particular, se K é uma malha conforme, K' também o é.

Esse foi um pequeno resumo da teoria estelar. O nosso objetivo, entretanto, é mais prático que teórico. Queremos definir esquemas de subdivisão combinatórios, ou seja, métodos "automáticos" de se propagar subdivisões em uma

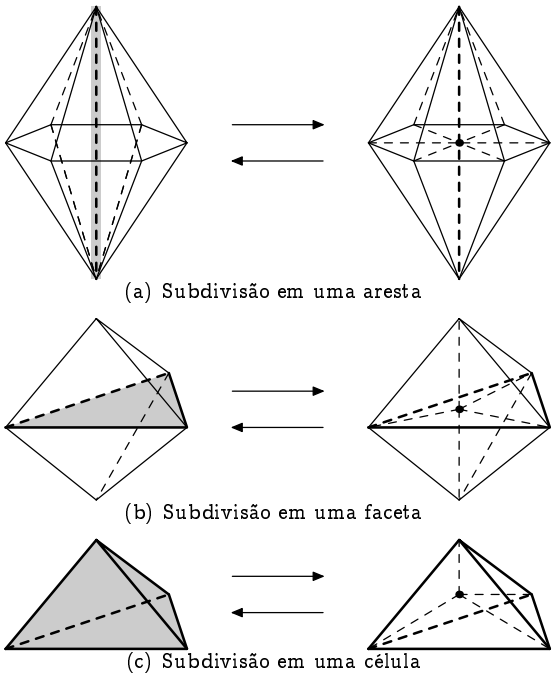


Figura 4.1: Operações estelares aplicadas em faces de um tetraedro. ($\rightarrow$) indica subdivisão e ($\leftarrow$), fusão.

malha que se baseiem apenas em seus dados combinatórios. A idéia é usar a ordem parcial definida no conjunto de vértices da malha para guiar o processo de subdivisão. Para tanto, faz-se necessário, primeiramente, compatibilizar o item 3 da definição de complexo simplicial abstrato (definição 3) com a notação para subdivisão estelar, o que faremos na próxima seção.

4.1 Subdivisões estelares ordenadas

Com relação a notação empregada para representar subdivisões estelares, ainda que esteja implícito que a ordem parcial entre os elementos de V' distintos de v seja a mesma que está presente entre os elementos de V, não fica claro como o elemento adicionado v compara-se aos demais. Para resolver esta questão, vamos escrever $K' = K(\delta, v)_i$ quando for necessário explicitar que a ordem dos elementos em V' é tal que v é o i-ésimo elemento em cada uma das células de $\mathrm{st}(v, K')$. Essa operação é denominada *subdivisão estelar ordenada*.

Vamos considerar o efeito de uma subdivisão estelar ordenada sobre o complexo K gerado por uma única célula m-dimensional σ. Seja δ uma face n-dimensional de σ tal que

$$\sigma[j_0, j_1, \ldots, j_n] = \delta.$$

Da definição de subdivisão estelar, segue que o complexo $K' = K(\delta, v)_l$ consiste

de $n+1$ células

$$c_0\sigma, c_1\sigma, \ldots, c_n\sigma,$$

que denominamos *células filhas* da célula σ, uma vez que cada face de dimensão $n-1$ de $\partial\delta$ dá origem a uma célula de K'. Como v ocupa a posição l em todos as células $c_i\sigma$ e cada faceta $\partial_{j_k}\sigma$ pertence a um único $c_i\sigma$, podemos definir as células filhas de tal modo que a relação

$$\partial_l c_i\sigma = \partial_{j_{n-i}}\sigma$$

seja satisfeita.

A orientação das células filhas resulta dessa relação, uma vez que, para a orientação ser consistente, é necessário que

$$o(c_i\sigma)(-1)^l = o(\sigma)(-1)^{j_{n-i}}.$$

Assim podemos definir

$$o(c_i\sigma) := o(\sigma)(-1)^{l+j_{n-i}}.$$

Antes de prosseguir, vamos a um exemplo concreto. Considere o complexo

$$K = \langle v_0, v_1, v_2, v_3\rangle + \langle v_1, v_2, v_3, v_4\rangle,$$

composto de duas células $\sigma_1 = \langle v_0, v_1, v_2, v_3\rangle$ e $\sigma_2 = \langle v_1, v_2, v_3, v_4\rangle$. Vamos calcular o complexo $K' = K(\delta, v)_2$, com $\delta = \langle v_1, v_3\rangle$. Temos que

$$\sigma_1[1,3] = \delta \text{ e } \sigma_2[0,2] = \delta,$$

portanto

$$\begin{array}{ll} c_0\sigma_1 = \langle v_0, v_1, v, v_2\rangle & c_1\sigma_1 = \langle v_0, v_2, v, v_3\rangle \\ c_0\sigma_2 = \langle v_1, v_2, v, v_4\rangle & c_1\sigma_2 = \langle v_2, v_3, v, v_4\rangle \end{array}$$

e

$$\begin{array}{ll} o(c_0\sigma_1) = (-1)^{2+3} = -1 & o(c_1\sigma_1) = (-1)^{2+1} = -1 \\ o(c_0\sigma_2) = (-1)^{2+2} = +1 & o(c_1\sigma_2) = (-1)^{2+0} = +1 \end{array},$$

logo

$$K' = -\langle v_0, v_1, v, v_2\rangle - \langle v_0, v_2, v, v_3\rangle + \langle v_1, v_2, v, v_4\rangle + \langle v_2, v_3, v, v_4\rangle,$$

resultado coerente com a figura 4.2.

Da discussão acima resulta que uma seqüência de operações de subdivisão $\tau = (\delta_1, v_1)_{l_1}(\delta_2, v_2)_{l_2}\ldots(\delta_q, v_q)_{l_q}$ sobre uma malha inicial K, impõe uma hierarquia entre as células dos complexos intermediários, hierarquia que relaciona as células subdivididas com suas filhas. Podemos assim associar a uma malha inicial K e a uma seqüência de subdivisões τ, uma *função nível* level(σ) que mede a profundidade de σ na hierarquia. A função level deve claramente satisfazer as propriedades

$$\text{level}(\sigma) = 0 \Leftrightarrow \sigma \in K \text{ e } \text{level}(c_i\sigma) = \text{level}(\sigma) + 1.$$

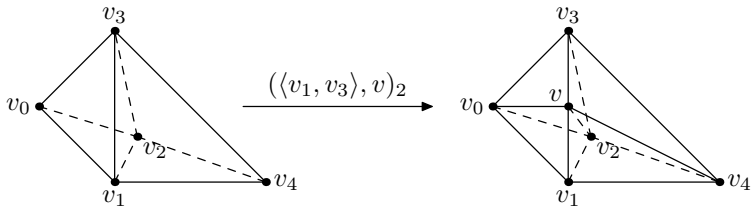


Figura 4.2: Representação gráfica da operação $K' = K(\delta, v)_2$.

Se σ é tal que level(σ) > 0, então existe uma única célula $p(\sigma)$ tal que $c_i p(\sigma) = \sigma$, para algum i, a qual denominamos *célula pai* de σ. Se level(σ) $= 0$, vamos convencionar que $p(\sigma) = \emptyset$.

Da mesma forma, level(σ) > 0 implica que existe um único vértice weld(σ) $\in \sigma$ tal que weld(σ) $\notin p(\sigma)$. O vértice weld(σ) é denomidado *vértice de fusão* de σ, pois se quisermos desfazer a operação que subdividiu σ, este é o vértice que deve ser removido da malha. Também vamos convencionar que weld(σ) $= \emptyset$, se level(σ) $= 0$.

Agora estamos prontos para definir os esquemas de subdivisão estelar.

4.2 Esquemas de Subdivisão Estelar

Um esquema de subdivisão estelar é composto de dois ingredientes: um conjunto de *invariantes combinatórios* $\mathcal{I}$ e um algoritmo de subdivisão SUBDIVIDE. Dada uma malha conforme K satisfazendo a $\mathcal{I}$, possivelmente obtida a partir de uma seqüência de subdivisões τ, e uma célula $\sigma \in K$, SUBDIVIDE(K,σ) efetua uma seqüência de subdivisões estelares $(\delta_1, v_1)_{l_1}(\delta_2, v_2)_{l_2} \ldots (\delta_q, v_q)_{l_q}$ em K, onde somente δ_q é face de σ, a chamada *face de subdivisão* de σ, de tal modo que a malha resultante

$$K' = K(\delta_1, v_1)_{l_1}(\delta_2, v_2)_{l_2} \ldots (\delta_q, v_q)_{l_q},$$

bem como cada uma das malhas intermediárias, também satisfaça a $\mathcal{I}$. Em resumo, o algoritmo SUBDIVIDE, a fim de manter certas propriedades da malha invariantes, propaga uma seqüência de subdivisões antes de subdividir σ.

Como o próprio nome indica, o conjunto de invariantes $\mathcal{I}$ deve referir-se apenas a dados combinatórios da malha, tais como a ordem parcial entre os vértices e as relações de adjacência e de hierarquia entre as células. O conjunto $\mathcal{I}$ também pode conter relações entre "rótulos" ou "cores" associadas as células, desde que existam regras definidas sobre como tais objetos se comportam com respeito à subdivisão estelar.

Um exemplo típico de invariante é o que estabelece que células adjacentes diferem em um nível no máximo, isto é,

$$\partial_i \sigma = \partial_j \omega \Rightarrow |\,\text{level}(\sigma) - \text{level}(\omega)| \leq 1. \qquad \text{(OneLevel)}$$

Esse invariante é importante pois assegura que atributos físicos ou geométricos associados as células variam "suavemente" na malha.

É óbvio que quanto mais "frouxos" forem os invariantes, mais livres estarão os algoritmos para efetuarem subdivisões. Inversamente, invariantes rígidos impõe restrições quanto a seqüência de subdivisões aplicadas à malha que, em última análise, resultam em malhas dotadas de regularidade e simetria.

Vamos examinar algumas restrições desejáveis aos algoritmos de subdivisão. Dizemos que SUBDIVIDE é *consistente* se existe uma função split definida na malha que associa a cada célula σ' uma face $\text{split}(\sigma') < \sigma'$ tal que, para qualquer subdivisão $K'(\delta, v)$ que venha a ser efetuada durante a execução de SUBDIVIDE(K, σ),

$$\sigma' \in \text{st}(\delta, K') \Rightarrow \text{split}(\sigma') = \delta,$$

ou seja, todas as células a serem subdivididas em uma mesma operação estelar concordam sobre a face de subdivisão. Graças a essa propriedade, é possível projetar um *algoritmo de fusão* genérico WELD(K, v) que remove v de K, desfazendo aquelas operações subseqüentes a que inseriu v na malha e que dependem de v.

Um algoritmo consistente é *alternante* se $\text{weld}(\sigma) \not\subset \text{split}(\sigma)$ para toda célula σ. Isso significa que faces recém criadas não podem ser faces de subdivisão, forçando a propagação da subdivisão para fora da estrela de $\text{split}(\sigma)$.

Com relação a hierarquia entre as células, dizemos que um algoritmo de subdivisão é *balanceado* se, em qualquer subdivisão $K'(\delta, v)$ que venha a ser efetuada durante a execução de SUBDIVIDE(K, σ),

$$\sigma', \sigma'' \in \text{st}(\delta, K') \Rightarrow \text{level}(\sigma') = \text{level}(\sigma''),$$

isto é, todas as células a serem subdivididas estão no mesmo nível hierárquico. Se interpretarmos que a função $\text{level}(\sigma)$ mede a qualidade da aproximação de atributos geométricos associados as células, verificamos que a propriedade acima está relacionada a homogeneidade dos atributos calculados durante a propagação da subdivisão.

Finalmente, um algoritmo de subdivisão é *baseado em n-simplexo* se a dimensão da face de subdivisão é igual a n. A seguir, vamos examinar dois esquemas de subdivisão estelar, cujos algoritmos de subdivisão possuem as propriedades acima, ou seja, são consistentes, alternantes e balanceados, sendo que um deles, o *esquema de Mitchell* [24], é baseado em faceta e o outro, o *esquema de Maubach* [20], baseado em aresta. O esquema de Mitchell é mais fácil de se entender, enquanto o de Maubach é mais sutil. Por isso adotaremos a estratégia de descrever o esquema de Mitchell com mais detalhe do que o necessário, a fim de identificar exatamente as etapas críticas da demonstração da correção do esquema, aproveitando em seguida a mesma estrutura dedutiva para o caso do esquema de Maubach.

4.2.1 Esquema de Mitchell

A idéia básica do esquema de Mitchell é "divida sempre a face oposta ao 'primeiro' vértice". Reinterpretada em termos de invariantes da malha, essa idéia resulta na seguinte definição:

```
1  template <class T>
2  void
3  mitchell_subdivide(T& t, Cell(T) c) {
4    Cell(T) ca=adjacent(t,c,0);
5    if(level(t,ca)<level(t,c)) mitchell_subdivide(t,ca);
6    mitchell_facet_split(t,face_op(t,c,0));
7  }
```

Listagem 4.1: Algoritmo de Mitchell

Definição 8. *Uma malha m-dimensional K satisfaz os* invariantes de Mitchell *se, além da propriedade expressa em (OneLevel), para quaisquer células adjacentes σ e ω, com $\partial_i\sigma = \partial_j\omega$, valem as seguintes propriedades:*

$$\text{level}(\sigma) = \text{level}(\omega) \Rightarrow \begin{cases} i = 0 = j \\ \quad ou \\ i \neq 0 \neq j \end{cases} \tag{MitchellA}$$

$$\text{level}(\sigma) = \text{level}(\omega) + 1 \Rightarrow i = 0 \neq j \tag{MitchellB}$$

Essas condições possuem uma significado bem simples. Se definirmos

$$\text{split}(\sigma) := \partial_0\sigma = \sigma[1, 2, \ldots, m],$$

vemos que a primeira condição diz que para duas células σ e ω adjacentes de mesmo nível, ou $\text{split}(\sigma) = \text{split}(\omega)$, ou $\text{split}(\sigma) \notin \omega$ e $\text{split}(\omega) \notin \sigma$. Já a segunda condição diz, como veremos abaixo, que se a célula de menor nível ω for subdividida em $\text{split}(\omega)$, uma de suas células filhas compartilhará a face de subdivisão com σ.

É fácil dar exemplos de malhas que satisfazem os invariantes de Mitchell. De fato, dada uma malha K qualquer, é possível efetuar um número finito de subdivisões estelares em K, de modo a obter uma malha K' que satisfaça os invariantes de Mitchell. Basta, por exemplo, aplicar a operação $(\sigma, v_\sigma)_0$ a todas as células $\sigma \in K$ e colocar $\text{level}(\sigma') = 0$, para todas as células σ' da malha resultante K'. Mas, dependendo da malha K, pode-se obter o mesmo efeito com menos subdivisões.

O algoritmo de subdivisão que explora esses invariantes está descrito na listagem 4.1. A idéia é propagar subdivisões da forma $(\text{split}(\sigma), v)_0$ mantendo os invariantes. A seguir, vamos provar três lemas que serão necessários na demonstração da correção do algoritmo 4.1.

Lema 1. *Seja K uma malha que satisfaz os invariantes de Mitchell e $\sigma, \omega \in K$ duas células adjacentes com $\partial_i\sigma = \partial_j\omega$. Então*

$$\begin{cases} \text{level}(\omega) \leq \text{level}(\sigma) & , \text{ se } i = 0 \\ \text{level}(\omega) \geq \text{level}(\sigma) & , \text{ se } i \neq 0\,. \end{cases}$$

Demonstração. Se level(ω) $>$ level(σ), então, por (OneLevel), level(ω) = level(σ) + 1. Segue de (MitchellB) que $i \neq 0$, o que prova o primeiro caso. Se level(ω) $<$ level(σ), então, por (OneLevel), level(σ) = level(ω) + 1. Segue de (MitchellB) que $i = 0$, o que prova o segundo caso. □

Lema 2. *Seja K uma malha composta de uma única célula m-dimensional ω e $K' = K(w[1,2,\ldots,m],v)_0$. Então as células $c_i\omega$ de K', $i = 0,\ldots,m-1$, satisfazem as relações $\partial_{m-i}c_i\omega = \partial_{m-i}c_{i+1}\omega$, para $i = 0,\ldots,m-2$, e $\partial_m c_{m-1}\omega = \partial_2 c_0\omega$.*

Demonstração. Seja $w = \langle x, v_1, v_2, \ldots, v_{m-2}, v_{m-1}, v_m\rangle$. Aplicando a operação $(w[1,2,\ldots,m],v)_0$, obtemos as seguintes células e relações em K':

$$\begin{array}{r|l}
c_0\omega = \langle v, x, v_1, v_2, \ldots, v_{m-2}, v_{m-1}\rangle & \\
c_1\omega = \langle v, x, v_1, v_2, \ldots, v_{m-2}, v_m\rangle & \partial_m c_0 w = \partial_m c_1 w \\
c_2\omega = \langle v, x, v_1, v_2, \ldots, v_{m-1}, v_m\rangle & \partial_{m-1} c_1 w = \partial_{m-1} c_2 w \\
\vdots & \vdots \\
c_{m-2}\omega = \langle v, x, v_1, v_3, \ldots, v_{m-1}, v_m\rangle & \partial_2 c_{m-2} w = \partial_2 c_{m-1} w \\
c_{m-1}\omega = \langle v, x, v_2, v_3, \ldots, v_{m-1}, v_m\rangle & \partial_m c_{m-1} w = \partial_2 c_0 w
\end{array}$$

e o resultado segue, pois nenhuma outra relação de adjacência é possível. □

Lema 3. *Seja K uma malha composta de duas células m-dimensionais adjacentes σ e ω, com $\partial_i\sigma = \partial_j\omega$, tais que*

$$0 \leq \text{level}(\sigma) - \text{level}(\omega) \leq 1.$$

Então, se K satisfaz os invariantes de Mitchell, $K' = K(w[1,2,\ldots,m],v)_0$ também os satisfaz.

Demonstração. São três os casos possíveis:

1. level(σ) = level(ω) e $i = 0 = j$

   Ambas células são divididas, dando origem às células $c_k\sigma$ e $c_k\omega$, todas com mesmo nível, que, pelo lema anterior, satisfazem as relações $\partial_{m-k}c_k\omega = \partial_{m-k}c_{k+1}\omega$, $\partial_{m-k}c_k\sigma = \partial_{m-k}c_{k+1}\sigma$, para $k = 0,\ldots,m-2$, $\partial_m c_{m-1}\omega = \partial_2 c_0\omega$, $\partial_m c_{m-1}\sigma = \partial_2 c_0\sigma$. Além dessas relações, claramente temos que $\partial_1 c_k\sigma = \partial_1 c_k\omega$, para $k = 0,\ldots,m-1$. Todas essas relações satisfazem a (MitchellA).

2. level(σ) = level(ω) e $i \neq 0 \neq j$

   ω é dividida, dando origem às células $c_k\omega$, que, pelo lema anterior, satisfazem as relações $\partial_{m-k}c_k\omega = \partial_{m-k}c_{k+1}\omega$, , para $k = 0,\ldots,m-2$ e $\partial_m c_{m-1}\omega = \partial_2 c_0\omega$, que satisfazem (MitchellA). Temos também que $\partial_i\sigma = \partial_0 c_{m-j}\omega$, que satisfaz a (MitchellB).

3. level(σ) = level(ω) + 1 e $i = 0 \neq j$

   ω é dividida, dando origem às células $c_k\omega$, que, pelo lema anterior, satisfazem as relações $\partial_{m-k}c_k\omega = \partial_{m-k}c_{k+1}\omega$, , para $k = 0,\ldots,m-2$ e $\partial_m c_{m-1}\omega = \partial_2 c_0\omega$, que satisfazem (MitchellA). Temos também que $\partial_i\sigma = \partial_0 c_{m-j}\omega$, que satisfaz a (MitchellA).

Em todos os casos, K' satisfaz os invariantes de Mitchell. □

Recapitulando, o lema 1 diz basicamente que, para cada célula σ em uma malha Mitchell, existe no máximo uma célula adjacente de nível menor do que σ, justamente a célula $\mathrm{adj}_0\,\sigma$. O lema 2 diz que as células filhas de uma célula σ satisfazem os invariantes de Mitchell após uma subdivisão. Finalmente, o lema 3, mostra que a invariância persiste depois de uma subdivisão, no caso de duas células adjacentes. Como os invariantes de Mitchell estão descritos em termos de pares de células adjacentes, podemos usar esses resultados para provar que o algoritmo de Mitchell funciona corretamente.

Teorema 4. *Sejam K uma malha m-dimensional que satisfaz os invariantes de Mitchell e σ uma célula de K. Então a malha K' resultante da aplicação do algoritmo (4.1) a (K,σ) também satisfaz os invariantes de Mitchell. Além disso, o algoritmo é consistente, alternante e balanceado.*

Demonstração. Por indução no nível de σ. Seja l o menor valor assumido pela função level em K. Se $\mathrm{level}(\sigma) = l$, segue do lema 1 que $\mathrm{level}(\omega) = l$, com $\omega = \mathrm{adj}_0\,\sigma$. Portanto, a recursão na linha 5 não é efetuada. De (MitchellA) resulta que $\partial_0\sigma = \partial_0\omega$, ou seja, $\mathrm{split}(\sigma) = \mathrm{split}(\omega)$. Aplicando o lema 3 a todos os pares de células adjacentes (σ',ω'), com $\omega' \in \mathrm{st}(\mathrm{split}(\sigma), K)$, temos que a subdivisão efetuada na linha 6 resulta em uma malha K' que satisfaz a tese. Se $\mathrm{level}(\sigma) > l$, podemos utilizar o lema 1 e a hipótese de indução na linha 5 para recair a um caso semelhante ao inicial. □

4.2.2 Esquema de Maubach

O esquema de Maubach se originou do estudo da processo de subdivisão da triangulação canônica (também conhecida como Coxeter-Freudenthal-Kuhn [12]) de um cubo "pela maior aresta" (figura 4.3). A idéia é codificar a maior aresta combinatorialmente.

Definição 9. *Uma malha m-dimensional K satisfaz os* invariantes de Maubach *se, além da propriedade expressa em (OneLevel), para quaisquer células adjacentes σ e ω, com $\partial_i\sigma = \partial_j\omega$ e definindo* $\mathrm{type}(\sigma) := m - (\mathrm{level}(\sigma) \bmod m)$*, valem as seguintes propriedades:*

$$\mathrm{level}(\sigma) = \mathrm{level}(\omega) \Rightarrow \begin{cases} i = j \\ ou \\ \{i,j\} = \{0, \mathrm{type}(\omega)\} \end{cases} \qquad \text{(MaubachA)}$$

$$\mathrm{level}(\sigma) = \mathrm{level}(\omega) + 1 \Rightarrow i = \mathrm{type}(\omega) \;\; e \;\; j \in \{0, \mathrm{type}(\omega)\} \qquad \text{(MaubachB)}$$

Analogamente ao esquema de Mitchell, a interpretação dessas condições está relacionada ao compartilhamento das faces de subdivisão, no caso *arestas de subdivisão*, entre células adjacentes. Se definirmos

$$\mathrm{split}(\sigma) := \sigma[0, \mathrm{type}(\sigma)],$$

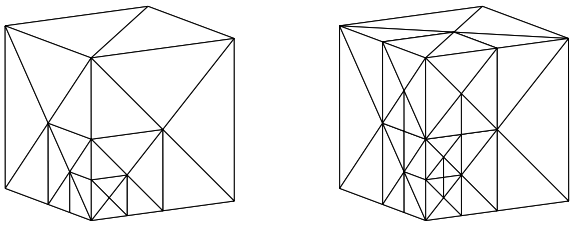


Figura 4.3: Propagação da subdivisão segundo a "maior aresta".

vemos que a primeira condição diz que para duas células σ e ω adjacentes de mesmo nível, ou $\text{split}(\sigma) = \text{split}(\omega)$, ou $\text{split}(\sigma) \notin \omega$ e $\text{split}(\omega) \notin \sigma$. Com efeito, se $\text{split}(\sigma)$ é aresta de ω, então $\partial_i\sigma = \partial_j\omega$ com $i \notin \{0, \text{type}(\sigma)\}$. Portanto, da condição (MaubachA), resulta que $i = j$, já que $\text{type}(\sigma) = \text{type}(\omega)$. Assim, σ e ω são duas células que diferem apenas no i-ésimo vértice, com $i \notin \{0, \text{type}(\omega)\}$, logo $\text{split}(\sigma) = \text{split}(\omega)$. Já a segunda condição diz, como veremos abaixo, que se a célula de menor nível ω for subdividida em $\text{split}(\omega)$, uma de suas células filhas compartilhará a face de subdivisão com σ.

Como exemplo de malhas que satisfazem os invariantes de Maubach, temos o seguinte resultado, cuja demonstração postergaremos para o final desta seção: a triangulação canônica de um produto cartesiano de um número arbitrário de simplexos satisfaz os invariantes de Maubach.

Dada uma malha m-dimensional K qualquer, é possível, de maneira análoga ao esquema de Mitchell, efetuar um número finito de subdivisões estelares em K, de modo a obter uma malha K' que satisfaça os invariantes de Maubach. Uma maneira de fazer isso é aplicar inicialmente a operação $(\sigma, v_\sigma)_m$ a todas as células $\sigma \in K$, depois a operação $(\delta, v_\delta)_{m-1}$ a todas as facetas $\delta \in K$, e assim sucessivamente até aplicar a operação $(\epsilon, v_\epsilon)_1$ a todas as arestas $\epsilon \in K$ e colocar $\text{level}(\sigma') = 0$, para todas as células σ' da malha resultante K'. Daremos mais detalhes sobre essa construção no fim da seção.

O algoritmo de subdivisão de Maubach, exibido na listagem 4.2, é um pouco mais complexo que o de Mitchell, mas os passos necessários para provar sua correção são bastante similares, como veremos a seguir.

Lema 5. *Seja K uma malha que satisfaz os invariantes de Maubach e $\sigma, \omega \in K$ duas células adjacentes com $\partial_i\sigma = \partial_j\omega$. Então*

$$\begin{cases} \text{level}(\omega) \leq \text{level}(\sigma) & , \text{ se } i \notin \{0, \text{type}(\sigma)\} \\ \text{level}(\omega) \geq \text{level}(\sigma) & , \text{ se } i \in \{0, \text{type}(\sigma)\}. \end{cases}$$

Além disso, se $\text{level}(\omega) < \text{level}(\sigma)$, *então* $i = \text{type}(\omega)$.

Demonstração. Se $\text{level}(\omega) > \text{level}(\sigma)$, então, por (OneLevel), $\text{level}(\omega) = \text{level}(\sigma) + 1$. Segue de (MaubachB) que $i \in \{0, \text{type}(\sigma)\}$, o que prova o primeiro caso. Se $\text{level}(\omega) < \text{level}(\sigma)$, então, por (OneLevel), $\text{level}(\sigma) = \text{level}(\omega) + 1$.

```
1  template <class T>
2  void
3  maubach_subdivide(T& t, Cell(T) ce) {
4    dim_t k=Dim(T)-level(t,ce)%Dim(T);
5    list<Cell(T)> st;
6    queue<Cell(T)> to_visit;
7    to_visit.push(ce);
8    while(!to_visit.empty()) {
9      Cell(T) c=to_visit.front();
10     to_visit.pop();
11     if(!visited(t,c)) {
12       visit(t,c);
13       st.push_back(c);
14       for(int i=0; i<=Dim(T);++i) {
15         if((i==0)||(i==k)) continue;
16         Cell(T) ca=adjacent(t,c,i);
17         if(!visited(t,ca)) {
18           if(level(t,ca)<level(t,c)) {
19             maubach_subdivide(t,ca);
20             ca=adjacent(t,c,i);
21           }
22           to_visit.push(ca);
23         }
24       }
25     }
26   }
27   Edge(T) e=face_ind(t,ce,_<dim_t>(0,k));
28   maubach_edge_split(t,e,k,st.begin(),st.end());
29 }
```

Listagem 4.2: Algoritmo de Maubach

Segue de (MaubachB) que $i = \text{type}(\omega)$, logo $i \notin \{0, \text{type}(\sigma)\}$, o que prova a observação e o segundo caso. □

Lema 6. *Seja K uma malha composta de uma única célula m-dimensional ω e $K' = K(w[0,l],v)_l$, com $l > 0$. Então as células $c_0\omega$ e $c_1\omega$ de K' satisfazem a relação $\partial_0 c_0\omega = \partial_{l-1} c_1\omega$.*

Demonstração. Seja $w = \langle v_0, v_1, \ldots, v_m\rangle$. Aplicando a operação $(w[0,l],v)_l$, obtemos as seguintes células e relações em K':

$$\left.\begin{array}{c} c_0\omega = \langle v_0, v_1, \ldots, v_{l-1}, v, v_{l+1}, \ldots, v_m\rangle \\ c_1\omega = \langle v_1, v_2, \ldots, v_l, v, v_{l+1}, \ldots, v_m\rangle \end{array}\right| \partial_0 c_0 w = \partial_{l-1} c_1 w$$

e o resultado segue. □

Lema 7. *Seja K uma malha composta de duas células m-dimensionais adjacentes σ e ω, com $\partial_i\sigma = \partial_j\omega$, tais que*

$$0 \leq \text{level}(\sigma) - \text{level}(\omega) \leq 1.$$

Então, se K satisfaz os invariantes de Maubach, $K' = K(w[0,l],v)_l$, com $l = \text{type}(\omega)$, também os satisfaz.

Demonstração. São sete os casos possíveis:

1. $\text{level}(\sigma) = \text{level}(\omega)$ e $i = j = 0$

   ω é subdividida resultando nas células $c_0\omega$ e $c_1\omega$, com $\partial_0 c_0\omega = \partial_{l-1} c_1\omega$, que satisfaz a (MaubachA). Temos também que $\partial_0\sigma = \partial_l c_1\omega$, que satisfaz a (MaubachB).

2. $\text{level}(\sigma) = \text{level}(\omega)$ e $i = j = l$

   ω é subdividida resultando nas células $c_0\omega$ e $c_1\omega$, com $\partial_0 c_0\omega = \partial_{l-1} c_1\omega$, que satisfaz a (MaubachA). Temos também que $\partial_l\sigma = \partial_l c_0\omega$, que satisfaz a (MaubachB).

3. $\text{level}(\sigma) = \text{level}(\omega)$ e $i = j = k \notin \{0, l\}$

   σ e ω são subdivididas resultando nas células $c_0\sigma$ e $c_1\sigma$, com $\partial_0 c_0\sigma = \partial_{l-1} c_1\sigma$, e $c_0\omega$ e $c_1\omega$, com $\partial_0 c_0\omega = \partial_{l-1} c_1\omega$, que satisfazem a (MaubachA). Temos também que $\partial_k c_0\sigma = \partial_k c_0\omega$ e $\partial_{k-1} c_1\sigma = \partial_{k-1} c_1\omega$, que satisfazem a (MaubachA).

4. $\text{level}(\sigma) = \text{level}(\omega)$, $i = 0$ e $j = l$

   ω é subdividida resultando nas células $c_0\omega$ e $c_1\omega$, com $\partial_0 c_0\omega = \partial_{l-1} c_1\omega$, que satisfaz a (MaubachA). Temos também que $\partial_0\sigma = \partial_l c_0\omega$, que satisfaz a (MaubachB).

5. $\text{level}(\sigma) = \text{level}(\omega)$, $i = l$ e $j = 0$

   ω é subdividida resultando nas células $c_0\omega$ e $c_1\omega$, com $\partial_0 c_0\omega = \partial_{l-1} c_1\omega$, que satisfaz a (MaubachA). Temos também que $\partial_l\sigma = \partial_l c_1\omega$, que satisfaz a (MaubachB).

6. level(σ) = level(ω) + 1, $i = l$ e $j = 0$

   ω é subdividida resultando nas células $c_0\omega$ e $c_1\omega$, com $\partial_0 c_0\omega = \partial_{l-1} c_1\omega$, que satisfaz a (MaubachA). Temos também que $\partial_l \sigma = \partial_l c_1\omega$, que satisfaz a (MaubachA).

7. level(σ) = level(ω) + 1, $i = l$ e $j = l$

   ω é subdividida resultando nas células $c_0\omega$ e $c_1\omega$, com $\partial_0 c_0\omega = \partial_{l-1} c_1\omega$, que satisfaz a (MaubachA). Temos também que $\partial_l \sigma = \partial_l c_0\omega$, que satisfaz a (MaubachA).

Em todos os casos, K' satisfaz os invariantes de Maubach. □

De modo similar ao esquema de Mitchell, podemos aplicar os lemas acima na demonstração da correção do algoritmo de subdivisão de Maubach. O único detalhe adicional é que, neste caso, a hipótese da malha K ser conforme é imprescindível, já que o algoritmo de Maubach embute implicitamente o percorrimento da estrela da aresta de subdivisão.

Teorema 8. *Sejam K uma malha m-dimensional que satisfaz os invariantes de Maubach e σ uma célula de K. Então a malha K' resultante da aplicação do algoritmo (4.2) a (K, σ) também satisfaz os invariantes de Maubach. Além disso, o algoritmo é consistente, alternante e balanceado.*

Demonstração. Por indução no nível de σ. Seja l o menor valor assumido pela função level em K. Se level(σ) = l, segue do lema 5 que células adjacentes a σ e que compartilham a aresta de subdivisão $\epsilon = \sigma[0, \text{type}(\sigma)]$ também têm nível l. Portanto, a recursão na linha 19 não é executada, e o loop entre as linhas 8 – 26 nada mais faz que percorrer a estrela de ϵ e armazená-la na lista st. Aplicando o lema 7 a todos os pares de células adjacentes (σ', ω'), com $\omega' \in \text{st}(\epsilon, K)$, temos que a subdivisão efetuada na linha 28 resulta em uma malha K' que satisfaz a tese. Se level(σ) > l, podemos utilizar o lema 5 e a hipótese de indução na linha 19 para recair a um caso semelhante ao inicial. □

Vamos retomar o tema das malhas que satisfazem os invariantes de Maubach. Antes de provar o teorema 9, vamos definir o que entendemos por triangulação canônica de um produto de simplexos. Dados n simplexos

$$\begin{aligned} \Delta^{m_1} &= \langle v_0^1, \ldots, v_{m_1}^1 \rangle, \\ \Delta^{m_2} &= \langle v_0^2, \ldots, v_{m_2}^2 \rangle, \\ &\cdots \\ \Delta^{m_n} &= \langle v_0^n, \ldots, v_{m_n}^n \rangle, \end{aligned}$$

o conjunto de vértices V do produto cartesiano $\Sigma = \Delta^{m_1} \times \cdots \times \Delta^{m_n}$, é formado pelos vértices v_L, com $L = (L^1, \ldots, L^n) \in P$, onde

$$P = \{0, \ldots, m_1\} \times \{0, \ldots, m_2\} \times \cdots \times \{0, \ldots, m_n\},$$

tal que

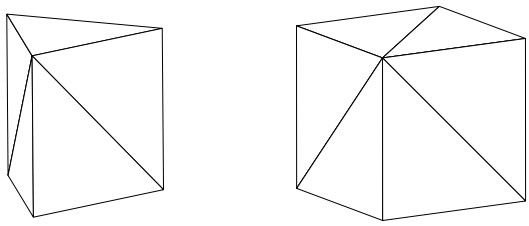
$$v_L = (v_{L^1}^1, v_{L^2}^2, \ldots, v_{L^n}^n).$$

Considere agora o conjunto $\mathcal{L}$ de todos os *caminhos monónotonos* $L = L_0 L_1 \dots L_s$ sobre o reticulado P, com $s = m_1 + \dots + m_n$, ou seja, cada $L_q = (L_q^1, \dots, L_q^n)$ é tal que $L_0 = (0, \dots, 0)$, $L_s = (m_1, \dots, m_n)$ e $L_{q+1} = L_q + J$, onde todas as entradas de J são nulas, exceto uma que vale 1.

O complexo $\mathrm{Tr}(\Sigma)$ é definido sobre V de modo que exista uma bijeção entre os caminhos de $\mathcal{L}$ e as células de $\mathrm{Tr}(\Sigma)$, bijeção esta que associa o caminho $L_0 L_1 \dots L_s \in \mathcal{L}$ à célula

$$\langle v_{L_0}, v_{L_1}, \dots, v_{L_s} \rangle \in \mathrm{Tr}(\Sigma).$$

Este complexo $\mathrm{Tr}(\Sigma)$, então, representa a *triangulação canônica* de Σ.



(a) $\Delta^1 \times \Delta^2$ (b) $\Delta^1 \times \Delta^1 \times \Delta^1$

Figura 4.4: Produtos de simplexos.

Teorema 9. *Seja* $\mathrm{Tr}(\Sigma)$ *uma malha que representa a triangulação canônica do produto de n simplexos* $\Sigma = \Delta^{m_1} \times \Delta^{m_2} \times \dots \times \Delta^{m_n}$. *Então, para quaisquer células* $\sigma, \omega \in \mathrm{Tr}(\Sigma)$,

$$\partial_i \sigma = \partial_j \omega \Rightarrow i = j.$$

Em particular, $\mathrm{Tr}(\Sigma)$ *satisfaz os invariantes de Maubach.*

Demonstração. Uma propriedade básica da construção acima é que para toda célula $\sigma \in \mathrm{Tr}(\Sigma)$,

$$\sigma[j] = v_L \Leftrightarrow |L| = j.$$

Sejam $\sigma, \omega \in \mathrm{Tr}(\Sigma)$ duas células adjacentes, com

$$\sigma = \langle v_{L_0}, \dots, v_{L_s} \rangle \text{ e } \omega = \langle v_{L'_0}, \dots, v_{L'_s} \rangle,$$

tais que $\partial_i \sigma = \langle v_{L''_0}, \dots, v_{L''_{s-1}} \rangle = \partial_j \omega$. Então existe um único $k \in \{0, \dots, s\}$ tal que $k \notin \{|L''_0|, \dots, |L''_{s-1}|\}$. Portanto, $|L_i| = k = |L'_j|$, logo $i = k = j$. Note que $k \notin \{0, s\}$, visto que para toda célula $\sigma \in \mathrm{Tr}(\Sigma)$, $\sigma[0, s] = \langle v_{(0,\dots,0)}, v_{(m_1,\dots,m_n)} \rangle$. □

O outro problema que devemos tratar é como transformar uma malha arbitrária K em uma malha que satisfaz os invariantes de Maubach. Para tanto,

vamos inicialmente considerar como obter a *subdivisão baricêntrica* de K através da aplicação de subdivisões estelares. Depois provaremos que a subdivisão baricêntrica de K assim obtida satisfaz os invariantes de Maubach.

A definição usual da subdivisão baricêntrica $B(\sigma)$ de um simplexo σ é indutiva: se $d(\sigma) = 0$, então $B(\sigma) = \sigma$; se $d(\sigma) = n$, encontre $B(\partial_i \sigma)$ para cada $0 \leq i \leq n$ e tome $B(\sigma)$ igual ao complexo $v_\sigma \star \cup_{i=0}^{n} B(\partial_i \sigma)$. Se adotarmos a ordem parcial em $B(\sigma)$ na qual v_σ é o n-ésimo vértice de qualquer célula $\delta \in B(\sigma)$, temos que

$$\delta = \langle v_{\sigma_0}, v_{\sigma_1}, \ldots, v_{\sigma_n} \rangle,$$

com $\sigma_n = \sigma$, $\sigma_{j-1} < \sigma_j$ e $d(\sigma_j) = j$. Reciprocamente, toda seqüência de faces $\sigma_0 < \sigma_1 < \ldots < \sigma_n$, com $\sigma_n = \sigma$ e $d(\sigma_j) = j$, dá origem a uma célula δ de $B(\sigma)$.

O mesmo efeito pode ser obtido por uma seqüência de subdivisões, basta tomar

$$B(\sigma) = \sigma(\sigma, v_\sigma)_n \prod_{\substack{\sigma_{n-1} < \sigma \\ d(\sigma_{n-1}) = n-1}} (\sigma_{n-1}, v_{\sigma_{n-1}})_{n-1} \cdots \prod_{\substack{\sigma_1 < \sigma \\ d(\sigma_1) = 1}} (\sigma_1, v_{\sigma_1})_1.$$

Para estender a subdivisão baricêntrica a um complexo K, é suficiente aplicar as subdivisões sobre todas os simplexos de K, na ordem correta:

$$B(K) = K \prod_{\substack{\sigma_n \in K \\ d(\sigma_n = n)}} (\sigma_n, v_{\sigma_n})_n \prod_{\substack{\sigma_{n-1} \in K \\ d(\sigma_{n-1}) = n-1}} (\sigma_{n-1}, v_{\sigma_{n-1}})_{n-1} \cdots \prod_{\substack{\sigma_1 \in K \\ d(\sigma_1) = 1}} (\sigma_1, v_{\sigma_1})_1.$$

Podemos agora enunciar o resultado desejado.

Teorema 10. *Seja K uma malha n-dimensional. Então, para quaisquer células $\sigma, \omega \in B(K)$,*

$$\partial_i \sigma = \partial_j \omega \Rightarrow i = j.$$

Em particular, $B(K)$ satisfaz os invariantes de Maubach.

Demonstração. Uma propriedade básica da construção acima é que para toda célula $\sigma \in B(K)$,

$$\sigma[j] = v_\delta \Leftrightarrow d(\delta) = j.$$

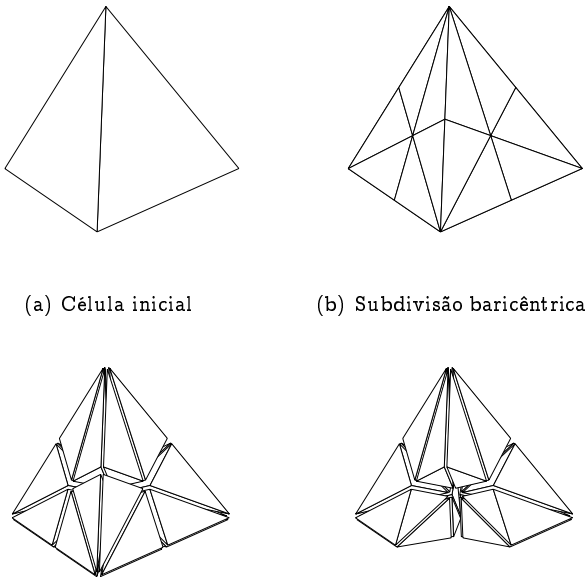
Sejam $\sigma, \omega \in B(K)$ duas células adjacentes, com

$$\sigma = \langle v_{\delta_0}, \ldots, v_{\delta_n} \rangle \text{ e } \omega = \langle v_{\delta'_0}, \ldots, v_{\delta'_n} \rangle,$$

tais que $\partial_i \sigma = \langle v_{\delta''_0}, \ldots, v_{\delta''_{n-1}} \rangle = \partial_j \omega$. Então existe um único $k \in \{0, \ldots, n\}$ tal que $k \notin \{d(\delta''_0), \ldots, d(\delta''_{n-1})\}$. Portanto, $d(\delta_i) = k = d(\delta'_j)$, logo $i = k = j$. □

4.2.3 Aspectos geométricos

O tratamento que demos aos esquemas de subdivisão estelar foi, até aqui, puramente combinatório, mas obviamente os aspectos geométricos também são relevantes. Vamos, portanto, descrever brevemente um resultado, devido a



(a) Célula inicial

(b) Subdivisão baricêntrica

(c) Encolhendo as células

(d) Removendo a estrela de um vértice

Figura 4.5: Subdivisão baricêntrica. Na figura (d) fica claro que cada célula n-dimensional fica dividida em $n+1$ hipercubos combinatórios n-dimensionais

Maubach, sobre as propriedades geométricas do esquema que leva seu nome, além de discutir alguns outras questões de caráter geométrico.

Seja $C^n = [0,1]^n$ o *hipercubo n-dimensional* e $\mathrm{Tr}(C^n)$ sua triangulação canônica. Considere uma malha M obtida a partir de um seqüência de aplicações do algoritmo de Maubach tal que as arestas são subdivididas sempre em seus pontos médios, ou seja, $M = M_m$ onde $M_0 = \mathrm{Tr}(C^n)$, cujas células possuem nível 0, e

$$M_{i+1} = \textsc{MaubachSubdivide}(M_i, \sigma_i),$$

com σ_i é uma célula qualquer de M_i.

O resultado de Maubach diz que se $\mathrm{type}(\sigma) = \mathrm{type}(\omega)$ para duas células $\sigma, \omega \in M$, então σ e ω são semelhantes, ou seja, existe uma transformação de semelhança que leva σ a ω. Resulta, portanto, que existem no máximo n tipos diferentes de células em M, a menos de semelhanças. A demonstração original de Maubach é um tanto intrincada, por isso esboçaremos a seguir uma demonstração mais simples, utilizando os resultados combinatórios da seção anterior.

Fixada uma dimensão n, vamos definir n simplexos n-dimensionais

$$\sigma_i = \langle p_{i,0}, p_{i,1}, \ldots, p_{i,n} \rangle,$$

onde $p_{i,j} \in \mathbb{R}^n$, $p_{n,0} = 0$ e $p_{n,j+1} = p_{n,j} + e_{j+1}$, e

$$p_{i-1,j} = \begin{cases} p_{i,j} & \text{,se } j \neq i \\ \frac{p_{i,j}}{2} & \text{,se } j = i\,. \end{cases}$$

Se um simplexo $\omega = \langle q_0, \ldots, q_n \rangle$ é semelhante ao simplexo σ_i de tal modo que a transformação de semelhança $S\colon \mathbb{R}^n \to \mathbb{R}^n$ satisfaz a condição $S(q_j) = p_{i,j}$, dizemos que $\mathrm{gtype}(\omega) = i$.

O fato geométrico essencial relativo aos simplexos σ_i pode ser assim resumido: se ω é um simplexo tal que $\mathrm{gtype}(\omega) = l$, então os dois simplexos $c_0\omega$ e $c_1\omega$ obtidos pela operação estelar

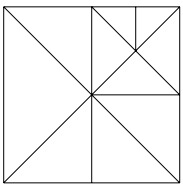
$$\omega(\omega[0,l], \frac{\omega[0] + \omega[l]}{2})_l,$$

satisfazem $\mathrm{gtype}(c_0\omega) = \mathrm{gtype}(c_1\omega) = l'$, onde $l' = l - 1$ se $l \neq 1$ e $l' = n$ se $l = 1$. A demonstração deste fato é tediosa, mas a idéia é simples. Segue facilmente da definição que $c_0\sigma_i = \sigma_{i-1}$, se $i \neq 1$ e que $c_0\sigma_1 = \frac{\sigma_n}{2}$, onde $\frac{\sigma_n}{2}$ denota um escalamento de fator $\frac{1}{2}$ de σ_n. Por outro lado, após a translação que leva o vértice inserido à origem, $c_1\sigma_i$ é levado em $c_0\sigma_i$ por uma reflexão seguida de uma permutação. Compondo as transformações na ordem correta, chega-se ao resultado desejado.

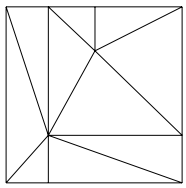
O teorema de Maubach ficará demonstrado se provarmos, para toda célula σ de uma malha M obtida da malha $\mathrm{Tr}(C^n)$ por uma seqüência de aplicações do algoritmo de Maubach, que $\mathrm{type}(\sigma) = \mathrm{gtype}(\sigma)$. Mas isso segue facilmente do fato acima por indução no número k de subdivisões. De fato, se $k = 0$, $\mathrm{type}(\sigma) = n$ para todas as células, pois $M = \mathrm{Tr}(C^n)$ é a malha inicial, e $\mathrm{gtype}(\sigma) = n$, pois uma simples permutação de coordenadas leva σ a σ_n. Se $k > 0$, temos que $M = M'(\delta, v)$ onde todas as células que serão subdivididas possuem o mesmo nível, já que o algoritmo de subdivisão é balanceado. Logo $\mathrm{type}(\sigma) = l$ para todas as células σ na estrela de δ. Aplicando a hipótese de indução, temos que $\mathrm{gtype}(\sigma)$ também é igual a l. Segue da propriedade geométrica mencionada no parágrafo anterior que a tese se verifica após a subdivisão.

A demonstração dada por Maubach é mais complexa porque ele exibe explicitamente as transformações de semelhança, informação que pode ser útil tanto na teoria quanto em aplicações. Por exemplo, é possível usar essas transformações para calcular as coordenadas dos vértices de uma célula sob demanda, ao invés de armazená-las como atributos dos vértices.

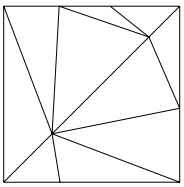
Embora o critério de subdivisões de arestas no ponto médio resulte em uma malha bastante simétrica, em algumas aplicações é necessário relaxar essa condição. Sem pretender esgotar as possibilidades, exibimos na figura 4.6 três outros critérios de subdivisão de aresta. No primeiro, uma reparametrização de coordenadas em cada eixo distorce a triangulação. No segundo, a subdivisão pode ocorrer em qualquer ponto da aresta, não apenas em seu ponto médio. No último critério, o vértice inserido pode estar em qualquer ponto, desde que a malha não se dobre sobre si mesma.



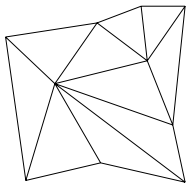
(a) Divisão ao meio



(b) Divisão ao meio, mas com reparametrização em cada eixo



(c) Divisão em qualquer ponto ao longo da aresta



(d) Divisão arbitrária

Figura 4.6: Critérios geométricos para subdivisão. Note que todas as triangulações são combinatorialmente equivalentes

4.3 Interface

Como vimos nas seções anteriores, a aplicação de subdivisões estelares induz uma hierarquia entre as células das malhas intermediárias. Em algumas aplicações, é interessante guardar toda a hierarquia, de modo que a resolução da malha possa ser ajustada segundo algum critério adaptativo. Nosso objetivo nesta seção é formular um conceito que una as idéias de multiresolução e de subdivisão estelar. Antes, contudo, precisamos de algumas definições.

Seja $(C, <)$ um conjunto parcialmente ordenado. Dados $c, c' \in C$, diremos $c \prec c'$ se $c < c'$ e não existe nenhum outro $c'' \in C$ tal que $c < c'' < c'$. Um elemento $c \in C$ é *minimal* se $c \leq c'$ para todo $c' \in C$. Um elemento minimal único é denominado *mínimo*. Elementos *maximais* e elemento *máximo* são definidos analogamente.

Definição 10. *Uma* multi-triangulação estelar n-ária *(n-MT, abreviadamente) é um conjunto parcialmente ordenado $(\mathcal{T}, <)$, onde $\mathcal{T}$ é um conjunto finito de malhas conformes e a ordem $<$ satisfaz:*

1. *Dadas $K, K' \in \mathcal{T}$, $K \prec K'$ se, e somente se, $K' = K(\delta, v)$, para algum simplexo $(n-1)$-dimensional δ de K;*

2. *Existe uma malha mínima e uma malha máxima em $\mathcal{T}$, chamadas malha inicial e malha final, respectivamente.*

A propriedade 2 diz que uma n-MT é, de fato, um *reticulado*, portanto um n-MT pode ser pensada como um grafo dirigido acíclico, com uma fonte e um dreno, cujas arestas estão rotuladas por subdivisões estelares em simplexos $(n-1)$-dimensionais.

Uma n-MT é composta de várias malhas, relacionadas entre si por operações estelares. Mas isso não significa que devamos representar cada malha separadamente, uma vez que as malhas compartilham muitos de seus simplexos. Nossa abordagem é representar todas as malhas em um único grande complexo simplicial, com um sub-complexo representando a malha corrente. Funções adicionais codificam as relações hierárquicas entre as células e as relações estelares entre o $(n-1)$-simplexo subdividido e o vértice inserido. Essas funções extras são suficientes para se implementar as transições entre malhas no reticulado.

A interface computacional exibida na listagem 4.3 resume os requisitos que devem ser acrescentados ao conceito de pseudovariedade para se lidar com multitriangulações estelares. As funções parent e child permitem a navegação pela hierarquia de células. Elas também permitem descobrir se uma célula pertence a malha inicial, caso sua ela tenha o simplexo vazio como pai, ou se pertence a malha final, caso todas suas células filhas sejam o simplexo vazio. As funções split_vertex e split_simplex registram as subdivisões estelares e, similarmente, testando-se o valor de retorno com o simplexo vazio, permitem descobrir se um vértice pertence a malha inicial, caso não tenha sido originado de nenhuma subdivisão, e se um $(n-1)$-simplexo foi ou não dividido. A função level retorna o nível da célula na hierarquia. Ela pode ser implementada através de um algoritmo que sobe na hierarquia de células, ou como uma simples função de acesso a um atributo associado às células. Finalmente, a função is_current, já descrita em outro conceito, é utilizada para determinar a malha corrente.

A implementação do conceito n-MT utiliza as mesmas técnicas descritas no capítulo anterior (confira a listagem 4.4). A estrutura ms, que representa um simplexo em uma n-MT, possui um parâmetro extra que indica a dimensão do simplexo de subdivisão. Os dados adicionais são justamente os necessários satisfazer os requisitos descritos no parágrafo anterior.

4.4 Conclusão

A motivação inicial por trás dos resultados deste capítulo era a extensão para malhas de tetraedros do trabalho de Velho e Gomes sobre malhas hierárquicas de triângulos [37]. Pesquisando na literatura, descobrimos que as operações estelares descritas em [1] e [16] poderiam servir de fundação para tal extensão, não só para o caso de malhas de tetraedros, como também no caso n-dimensional. Ao mesmo tempo, estudando a noção de multitriangulação da "escola de Gênova" [10], verificamos que poderíamos substituir as alterações locais, porém arbitrárias, por operações estelares, sem grande perda de expressividade, mas com grande simplificação em termos de estruturas de dados e implementação.

```
// MultiTriangulationConcept
// Refina PseudomanifoldConcept
  Cell(T)
  child(const T& t, Cell(T) c, int i);
// Retorna a i-ésima célula filha de c.

  Cell(T)
  parent(const T& t, Cell(T) c);
// Retorna a célula pai de c.

  SplitSimplex(T)
  split_simplex(const T& t, Vertex(T) v);
// Retorna o simplexo s que foi dividido pela
// operação (s, v).

  Vertex(T)
  split_vertex(const T& t, SplitSimplex(T) s);
// Retorna o vértice v que foi inserido pela
// operação (s, v).

  int level(const T& t, Cell(T) c);
// Retorna o nível de c.
```

Listagem 4.3: Multi-triangulação.

Essa pesquisa gerou como resultado o artigo [7], onde se encontra a definição de *multi-triangulação binária*, a qual generalizamos na definição 10 deste capítulo.

Mas havia ainda um detalhe que não estava claro. No trabalho de Velho e Gomes, a escolha da aresta de subdivisão obedecia ao critério da "aresta oposta ao primeiro vértice", que atribuímos a Mitchell [24], mas que, provavelmente, já fazia parte da tradição em várias áreas da matemática, sendo que o próprio Mitchell faz referências a um trabalho anterior de Sewell [32]. O fato é que esse critério puramente combinatório funciona bem para subdivisão de arestas em malhas de triângulos, mas não possuía uma generalização óbvia para malhas n-dimensionais. Foi Maubach em [20] que teve o *insight* de buscar a generalização na triangulação CFK do cubo n-dimensional.

Nossa contribuição foi simplesmente dar um tratamento mais combinatório, baseado em operações estelares ordenadas, ao critério de Maubach. Com essa separação entre geometria e combinatória, foi possível identificar certos invariantes do algoritmo de Maubach e, conseqüentemente, condições suficientes sobre as malhas de entrada para que o algoritmo funcione, que deram origem aos teoremas 9 e 10.

Uma direção de pesquisa futura é buscar novos esquemas de subdivisão estelar, seguindo a receita de Maubach, ou seja, estudar as propriedades geométricas de uma triangulação altamente simétrica e daí extrair critérios combinatórios. Outra possibilidade é permitir que as operações estelares possam ocorrer em

```
template <class T, dim_t k, bool t=(k!=Dim(T))>
struct ms_hier {
};

template <class T, dim_t k>
struct ms_hier<T,k,false> {
  Cell(T) child[SplitDim(T)+1];
  Cell(T) parent;
  ms_hier() {}
};

template <class T, dim_t k, bool t=(k!=SplitDim(T))>
struct ms_split {
};

template <class T, dim_t k>
struct ms_split<T,k,false> {
  Vertex(T) split;
  ms_split() {}
};

template <class T, dim_t k>
struct ms_vertex {
};

template <class T>
struct ms_vertex<T,0> {
  SplitSimplex(T) split;
  ms_vertex() {}
};

template <class T, dim_t k,
          template <dim_t> class Data=extra_data>
struct ms : bs<T,k,Data>, ms_hier<T,k>,
  ms_split<T,k>, ms_vertex<T,k> {
};
```

Listagem 4.4: Implementação.

simplexos de qualquer dimensão.

Esquemas de subdivisão geralmente são empregados juntamente com métodos de interpolação para se obter boas aproximações de uma função $f: U \to \mathbb{R}$ dada. A idéia é usar o esquema de subdivisão para subdividir o domínio U adaptativamente e, em seguida, construir uma função $\overline{f}: U \to \mathbb{R}$ que aproxima bem a função f através do método de interpolação. O critério de aproximação geralmente envolve alguma norma no espaço de funções de interesse. No próximo capítulo vamos estudar um método de interpolação diferente do usual, no qual não estamos interessados em minimizar o erro de aproximação, mas em garantir que a topologia das curvas de nível de f seja preservada.

Capítulo 5

Difeomorfismos Simpliciais

Complexos simpliciais fornecem apenas aproximações lineares de objetos gráficos. Mas as definições de simplexo e complexo simplicial podem ser estendidas a fim de fornecer aproximações mais suaves. Um *simplexo curvilinear p-dimensional* é um conjunto $\sigma \subset \mathbb{R}^m$ difeomorfo ao simplexo p-dimensional padrão Δ^p, ou seja, $\sigma = X(\Delta^p)$, onde X é um difeomorfismo. As faces de σ são as imagens por X das faces de Δ^m. Um *complexo simplicial curvilinear* é então definido de maneira análoga a um complexo simplicial, substituindo-se o termo "simplexo" por "simplexo curvilinear".

Existe uma vasta literatura na área de modelagem sobre complexos simpliciais curvilineares para o caso de X ser uma função polinomial ou polinomial por partes, mas em geral é difícil obter condições suficientes simples sobre os coeficientes dos polinômios (os "pontos de controle") de modo a assegurar que X seja de fato um difeomorfismo. Afinal, se por um lado não é possível provar que X é um difeomorfismo observando apenas seu comportamento local, por outro a análise global é dificultada pela liberdade que o espaço ambiente oferece para que σ se deforme.

Mas o que pode ser dito sobre os isocomplexos ? Nosso objetivo nesse capítulo é encontrar uma definição razoável para idéia de isocomplexos curvilineares. À primeira vista, poderíamos simplesmente substituir "simplexo" por "simplexo curvilinear" na definição de isocomplexo, mas isso nos levaria ao mesmo problema citado no parágrafo anterior. Nossa idéia é impor restrições às funções X de modo que seja possível provar que elas são difeomorfismos analisando apenas seu comportamento local. Surpreendentemente, essa análise para X polinomial possibilita a derivação de algumas condições suficientes simples para que X seja um difeomorfismo.

5.1 Funções simplicialmente invariantes

Nossa abordagem para definir uma versão curvilinear dos isocomplexos tem início com o estudo das aplicações de um complexo nele mesmo que preservam a relação de pertinência a um simplexo.

Definição 11. *Seja K um complexo simplicial e $X : |K| \to |K|$ uma função contínua. Dizemos que X é* simplicialmente invariante *com respeito ao complexo K, ou, mais brevemente, que X é* K-invariante*, se, para todo simplexo $\sigma \in K$, $X(\sigma) \subset \sigma$.*

Em particular, os vértices de K são deixados fixos por uma aplicação K-invariante. A função identidade em $|K|$ é um exemplo trivial de função simplicialmente invariante. Além disso, a composição de funções K-invariantes também é claramente K-invariante.

A primeira propriedade notável das funções simplicialmente invariantes é que o sinal "$\subset$" na definição 11 é de fato uma igualdade. Provaremos isso com a ajuda dos seguintes lemas.

Lema 11. *(Sperner) Seja T uma triangulação do simplexo m-dimensional padrão Δ^m, e $L : T_0 \to \{0, \ldots, m\}$ uma aplicação dos vértices de T nos inteiros de 0 a m. Dizemos que L atribui um* rótulo *$L(v)$ a cada vértice de T. Se para todo $v = (v^0, \ldots, v^m) \in T_0$,*

$$L(v) = i \Rightarrow v^i \neq 0,$$

então existe m-simplexo $\sigma = \langle v_0, \ldots, v_m \rangle \in T$ cujos vértices estão completamente rotulados*, ou seja, tal que*

$$L(\{v_0, \ldots, v_m\}) = \{0, \ldots, m\}.$$

Demonstração. Consulte O Livro[39] para uma demonstração. □

Lema 12. *Seja Δ^m o simplexo m-dimensional padrão e $X : \Delta^m \to \Delta^m$ uma função Δ^m-invariante. Então existe um ponto $x \in \Delta^m$ tal que $X(x) = \frac{1}{m+1}\mathbb{1}$, onde $\mathbb{1} := (1, 1, \ldots, 1)$. Ou seja, X leva x ao baricentro de Δ^m.*

Demonstração. Seja $(T_n)_{n\in\mathbb{N}}$ uma seqüência de triangulações de Δ^m tal que $\lim_{n\to\infty} \text{mesh}(T_n) = 0$, onde

$$\text{mesh}(T) = \max\{\text{diam}(\sigma) | \sigma \in T\}.$$

Defina uma aplicação L_n sobre os vértices T_n da seguinte maneira: se v é um vértice de T_n e $X(v) = (w^0, \ldots, w^m)$, $L_n(v) = i$, onde i é o menor inteiro tal que $w^i = \max_j\{w^j\}$. O fato de X ser Δ^m-invariante significa que, para todo $v = (v^0, \ldots, v^m) \in \Delta^m$, $X(v) = (w^0, \ldots, w^m)$ satisfaz

$$v^i = 0 \Rightarrow w^i = 0.$$

Portanto,

$$L_n(v) = i \Rightarrow w^i \neq 0 \Rightarrow v^i \neq 0,$$

ou seja, L_n satisfaz as hipóteses do Lema de Sperner, o que nos assegura a existência de uma seqüência de m-simplexos $(\sigma_n)_{n\in\mathbb{N}}$ completamente rotulados por L_n tal que $\lim_{n\to\infty} \text{diam}(\sigma_n) = 0$. Como Δ^m é um conjunto compacto, podemos assumir, passando a uma subseqüência se necessário, que a seqüência dos baricentros de σ_n converge para um ponto x. Pela continuidade de X e pela definição de L_n, segue que $X(x) = (w^0, \ldots, w^m)$, com cada $w^i = \max_j\{w^j\}$, ou seja, $X(x) = \frac{1}{m+1}\mathbb{1}$. □

Teorema 13. *Seja Δ^m o simplexo m-dimensional padrão e $X\colon \Delta^m \to \Delta^m$ uma função Δ^m-invariante. Então para todo simplexo $\sigma \in \Delta^m$, $X(\sigma) = \sigma$. Em particular, X é sobrejetiva.*

Demonstração. Por indução em m. O caso $m = 0$ é trivial. Seja $m > 0$. Pela hipótese de indução, $X(\sigma) = \sigma$ para todos os simplexos σ de Δ^m que são faces próprias. Resta provar que para todo ponto $y \in \text{int}(\Delta^m)$, ou seja, tal que $y \in \Delta^m$ e $y^i > 0$, existe um ponto $x \in \Delta^m$ tal que $X(x) = y$. Seja $Y_y\colon \Delta^m \to \Delta^m$ dada por

$$Y_y(w^0, \ldots, w^m) = \frac{1}{\sum \frac{w^i}{y^i}}\left(\frac{w^0}{y^0}, \ldots, \frac{w^m}{y^m}\right).$$

É fácil ver que Y_y é Δ^m-invariante e inversível, com inversa

$$Y_y^{-1}(w^0, \ldots, w^m) = \frac{1}{\sum w^i y^i}(w^0 y^0, \ldots, w^m y^m).$$

Aplicando o lema anterior a $Y_y X$ temos que existe um ponto x tal que

$$Y_y X(x) = \frac{1}{m+1}\mathbb{1}.$$

Segue das propriedades da função Y_y que

$$X(x) = Y_y^{-1}\left(\frac{1}{m+1}\mathbb{1}\right) = y.$$

□

Corolário 14. *Seja K um complexo simplicial e $X\colon |K| \to |K|$ uma função K-invariante. Então para todo simplexo $\sigma \in K$, $X(\sigma) = \sigma$. Em particular, X é sobrejetiva.*

Demonstração. Escreva X em relação às coordenadas baricêntricas de σ e aplique a proposição anterior. □

Agora que sabemos que funções simplicialmente invariantes são sobrejetivas em cada simplexo, cabe perguntar quando elas são injetivas. Como dissemos anteriormente, determinar se uma função é globalmente injetiva é um problema difícil em geral. Por outro lado, a injetividade local é uma propriedade mais fácil de se verificar. No caso de uma aplicação diferenciável F, por exemplo, basta que a derivada DF seja injetiva em um ponto x para que F seja injetiva numa vizinhança de x.

Num artigo clássico [22], Meisters e Olech estudaram detalhadamente algumas condições que uma função contínua e localmente injetiva deve satisfazer para ser globalmente injetiva. O que apresentamos a seguir é um adaptação dos argumentos apresentados nesse artigo para o caso das funções simplicialmente invariantes. Para tornar a exposição auto-contida, vamos inicialmente provar o seguinte lema.

Lema 15. *Seja $U \subset \mathbb{R}^m$ um conjunto aberto e $f\colon U \to \mathbb{R}^m$ uma função contínua. Se f é localmente injetiva, então $f(U)$ é um conjunto aberto.*

Demonstração. Se $f(U)$ é vazio, não há o que provar. Se $f(U)$ não é vazio, seja $y = f(x)$ um ponto de $f(U)$. Como U é aberto, existe uma vizinhança N de x contida em U tal que $f|_N$ é injetiva. Ademais, podemos obter um aberto $V \subset U$ tal que $x \in V$, com $\overline{V}$ é compacto e $\overline{V} \subset U$. Mas então f restrito a $\overline{V}$ é um homeomorfismo e como x pertence ao interior de $\overline{V}$, segue do Teorema da Invariância do Domínio de Brouwer que $f(x)$ pertence ao interior de $f(\overline{V})$ e portanto ao interior de $f(U)$. □

Estamos prontos para apresentar o resultado principal desta seção— funções simplicialmente invariantes localmente injetivas são injetivas.

Teorema 16. *Seja Δ^m o simplexo m-dimensional padrão e $X\colon \Delta^m \to \Delta^m$ uma função Δ^m-invariante. Se X é localmente injetiva, então X é injetiva.*

Demonstração. Por indução em m. O caso $m = 0$ é trivial. Seja $m > 0$. Do lema 15 combinado à proposição 13 resulta que

$$\begin{cases} X(\mathrm{bd}(\Delta^m)) = \mathrm{bd}(X(\Delta^m)) = \mathrm{bd}(\Delta^m) \\ X(\mathrm{int}(\Delta^m)) = \mathrm{int}(X(\Delta^m)) = \mathrm{int}(\Delta^m) \end{cases} \tag{5.1}$$

Pela proposição 13, para cada ponto de $y \in \Delta^m$ existe ao menos um ponto $x \in \Delta^m$ tal que $X(x) = y$. Seja A o conjunto dos $y \in \Delta^m$ tais que existe um único $x \in \Delta^m$ com $X(x) = y$ e $B = \Delta^m \setminus A$, ou seja, os pontos com mais de uma pré-imagem. Pela hipótese de indução, X restrita a $\mathrm{bd}(\Delta^m)$ é injetiva, e como por (5.1) nenhum ponto do interior de Δ^m é mapeado em $\mathrm{bd}(\Delta^m)$, segue que $\mathrm{bd}(\Delta^m) \subset A$. Nosso objetivo é mostrar que B é vazio. Para tanto, vamos supor que B não é vazio e, dessa suposição, concluir que $A \cup B$ formam uma cisão não trivial de Δ^m, o que é absurdo visto que Δ^m é conexo. Para derivar essa contradição, mostraremos a seguir que $\overline{A} \cap B = \emptyset$ e $A \cap \overline{B} = \emptyset$.

Seja y um ponto de B. Então existem dois pontos distintos x_1 e x_2 em Δ^m tais que $X(x_1) = X(x_2) = y$. Como $X(\mathrm{bd}(\Delta^m)) = \mathrm{bd}(\Delta^m) \subset A$, segue que x_1 e x_2 não pertencem ambos a $\mathrm{bd}(\Delta^m)$. Também não é possível ter $x_1 \in \mathrm{bd}(\Delta^m)$ e $x_2 \in \mathrm{int}(\Delta^m)$ (ou vice-versa), por (5.1). Portanto x_1 e x_2 pertencem a $\mathrm{int}(\Delta^m)$. Conseqüentemente, existem vizinhanças abertas de x_1 e x_2, N_1 e N_2 respectivamente, com $N_1 \cap N_2 = \emptyset$, tais que

$$N_1 \subset \mathrm{int}(\Delta^m) \text{ e } N_2 \subset \mathrm{int}(\Delta^m).$$

Pelo lema 15, $X(N_1)$ e $X(N_2)$ são conjuntos abertos. Assim,

$$N = X(N_1) \cap X(N_2)$$

é uma vizinhança aberta de y inteiramente contida em B, pois cada ponto de N possui ao menos duas pré-imagens. Logo $\overline{A} \cap B = \emptyset$.

Seja agora $(y_n)_{n\in\mathbb{N}}$ uma seqüência de pontos em B que converge para um ponto $y \in \Delta^m$. Então existem seqüências $(x_n^1)_{n\in\mathbb{N}}$ e $(x_n^2)_{n\in\mathbb{N}}$ em Δ^m tais que, para todo n,

$$y_n = X(x_n^1) = X(x_n^2) \text{ e } x_n^1 \neq x_n^2.$$

Como Δ^m é compacto, passando a subseqüências se necessário, podemos assumir que $(x_n^i)_{n\in\mathbb{N}}$ converge para um certo $x^i \in \Delta^m$, $i = 1, 2$. Pela continuidade de X, segue que

$$y = X(x^1) = X(x^2).$$

Temos que $x^1 \neq x^2$, pois caso contrário X não seria localmente injetiva em $x^1 (= x^2)$. Logo $A \cap \overline{B} = A \cap B = \emptyset$. □

Corolário 17. *Seja K um complexo simplicial e $X\colon |K| \to |K|$ uma função K-invariante. Se X é localmente injetiva, então para todo simplexo $\sigma \in K$, $X|_\sigma\colon \sigma \to \sigma$ é um homeomorfismo.*

Dito de outra maneira, toda função simplicialmente invariante localmente injetiva é um homeomorfismo.

Até aqui, a continuidade de X foi suficiente para obter todos os resultados. Mas, quando quisermos mostrar que X é localmente injetiva, a diferenciabilidade tornar-se-á muito útil pois, como mencionamos anteriormente, nesse caso a injetividade local segue da injetividade da derivada, reduzindo o problema a determinar se uma transformação linear é injetiva. Em resumo, a idéia é passar da topologia à análise e desta à álgebra.

Podemos definir, portanto, objeto de nossos estudos a partir de agora.

Definição 12. *Seja K um complexo simplicial e $X\colon |K| \to |K|$ uma função K-invariante. Dizemos que X é um* difeomorfismo simplicial *com respeito ao complexo K, ou, mais brevemente, que X é um* difeomorfismo K-invariante, *se, para todo simplexo $\sigma \in K$, X restrita a $\mathrm{int}(\sigma)$ é um difeomorfismo.*

Se denotarmos por $\mathrm{DS}(K)$ o conjunto de todos os difeomorfismos K-invariantes, temos claramente que

1. $\mathrm{Id} \in \mathrm{DS}(K)$;
2. $X \in \mathrm{DS}(K) \Rightarrow X^{-1} \in \mathrm{DS}(K)$;
3. $X_1, X_2 \in \mathrm{DS}(K) \Rightarrow X_1 X_2 \in \mathrm{DS}(K)$.

Também é claro por tudo que foi exposto que para uma função K-invariante X ser um difeomorfismo K-invariante é suficiente que, para todo $\sigma \in K$, $X|_{\mathrm{int}(\sigma)}$ seja diferenciável e que $D(X|_{\mathrm{int}(\sigma)})(p)$ seja injetiva para todo $p \in \mathrm{int}(\sigma)$. Na seqüência, vamos nos dedicar aos membros polinomiais de $\mathrm{DS}(K)$, mostrando que tais condições muitas vezes decorrem de condições mais simples.

5.2 Difeomorfismos simpliciais polinomiais

Seja $F := (F^1, \ldots, F^m)\colon \mathbb{R}^m \to \mathbb{R}^m$ uma aplicação polinomial, ou seja, uma aplicação da forma

$$x := (x^1, \ldots, x^m) \mapsto (F^1(x^1, \ldots, x^m), \ldots, F^m(x^1, \ldots, x^m))$$

com cada $F^i \in \mathbb{R}[X] := \mathbb{R}[X_1, \ldots, X_m]$, o anel dos polinômios em m variáveis sobre $\mathbb{R}$. Por $\deg(F)$ denotamos o *grau* de F, ou seja,

$$\deg(F) = \max_i \deg F^i.$$

Estamos interessados em descrever condições suficientes para que uma aplicação polinomial seja um difeomorfismo simplicial com respeito a um simplexo $\sigma = \langle p_0, p_1, \ldots, p_m \rangle \subset \mathbb{R}^m$. Resulta que essas condições são mais naturalmente formuladas em termos das coordenadas baricêntricas associadas a σ.

As coordenadas baricêntricas $w := (w^0, \ldots, w^m)$ de um ponto $x \in \mathbb{R}^m$, em relação ao simplexo σ, satisfazem a relação

$$P_\sigma \begin{pmatrix} w^0 \\ w^1 \\ \vdots \\ w^m \end{pmatrix} = \begin{pmatrix} 1 \\ x^1 \\ \vdots \\ x^m \end{pmatrix} \tag{5.2}$$

onde

$$P_\sigma = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ p_0^1 & p_1^1 & \cdots & p_m^1 \\ \vdots & \vdots & \ddots & \vdots \\ p_0^m & p_1^m & \cdots & p_m^m \end{pmatrix}.$$

Seja $A^m = \{w \in \mathbb{R}^{m+1} \mid \sum_i w^i = 1\}$ e $G := (G^0, \ldots, G^m) \colon A^m \to A^m$ uma aplicação polinomial da forma

$$(w^0, \ldots, w^m) \mapsto (G^0(w^0, \ldots, w^m), \ldots, G^m(w^1, \ldots, w^m))$$

com cada $G^i \in \mathbb{R}[W] := \mathbb{R}[W_0, \ldots, W_m]$. Dizemos que G é uma *representação baricêntrica* de F se

$$P_\sigma \begin{pmatrix} G^0(w) \\ G^1(w) \\ \vdots \\ G^m(w) \end{pmatrix} = \begin{pmatrix} 1 \\ F^1(x) \\ \vdots \\ F^m(x) \end{pmatrix} \tag{5.3}$$

vale para todos $x \in \mathbb{R}^m$ e $w \in A^m$ que satisfazem (5.2).

Note que F pode ter várias representações pois, como $\sum_i w^i = 1$, G^i assume os mesmos valores que $G^i + Q.(W_0 + \ldots + W_m - 1)$ quando avaliado em w, onde Q é um polinômio qualquer em $\mathbb{R}[W]$. Em outras palavras, se G e H são duas representações baricêntricas de F, então

$$G^i = H^i \pmod{\mathcal{M}},$$

onde $\mathcal{M} = \langle W_0 + \cdots + W_m - 1 \rangle$ é o ideal de $\mathbb{R}[W]$ gerado pelo polinômio $W_0 + \cdots + W_m - 1$.

Dizemos que uma representação baricêntrica H de F é *homogênea de grau n* se cada H^i é um polinômio homogêneo de grau n. Existe uma única representação baricêntrica de F homogênea de grau $n = \deg(F)$. Para mostrar isso,

defina polinômios G^i pela relação (5.3), de modo que

$$G^i = \sum_{|I| \leq n} g_I^i W^I,$$

onde I é um vetor de inteiros não negativos $(I^0, \ldots, I^m)$, $|I| = \sum_i I^i$ e $W^I := W_0^{I^0} \ldots W_m^{I^m}$. É fácil ver que $H := (H^0, \ldots, H^m)$, com

$$H^i = \sum_{|I| \leq n} g_I^i W^I (W_0 + \ldots + W_m)^{n-|I|}$$

é a representação desejada.

Alternativamente, H pode ser escrita da forma

$$H = \sum_{|I|=n} b_I B_I, \tag{5.4}$$

onde os $b_I \in A^m$ são chamados *pontos de controle de* H e $B_I := \binom{n}{I} W^I$ são os *polinômios de Bézier-Bernstein* [9]. Para ver que de fato $b_I \in A^m$, note que por (5.2) e (5.3)

$$\sum_i H^i \left(\frac{W_0}{W_0 + \ldots + W_m}, \ldots, \frac{W_m}{W_0 + \ldots + W_m}\right) = 1.$$

Como H^i é homogêneo de grau n,

$$\sum_i H^i = (W_0 + \ldots + W_m)^n. \tag{5.5}$$

Portanto,

$$\sum_i \sum_{|I|=n} b_I^i B_I = (W_0 + \ldots + W_m)^n$$

$$\sum_{|I|=n} \left(\sum_i b_I^i\right) B_I = \sum_{|I|=n} B_I$$

donde

$$\sum_i b_I^i = 1,$$

ou seja, $b_I \in A^m$.

Podemos definir pontos $c_I \in \mathbb{R}^m$, relacionados aos b_I pela equação

$$P_\sigma b_I = \begin{pmatrix} 1 \\ c_I^1 \\ \vdots \\ c_I^m \end{pmatrix},$$

chamados *pontos de controle de* F com relação ao simplexo σ. Tais pontos determinam a função F completamente e nosso objetivo agora é encontrar

condições suficientes sobre os pontos c_I para que F seja um difeomorfismo σ-invariante. Na verdade, as condições serão dadas sobre os pontos b_I de representações homogêneas na forma (5.4), de modo a torná-las independentes do particular simplexo σ.

Vamos inicialmente derivar uma condição suficiente para que H seja Δ^m-invariante. Seja χ a função que a cada n-upla $(x^1, \ldots, x^n)$ faz corresponder uma n-upla $(y^1, \ldots, y^n)$ tal que

$$y^i = \begin{cases} -1, & \text{se } x^i < 0 \\ 0, & \text{se } x^i = 0 \\ 1, & \text{se } x^i > 0\,. \end{cases}$$

Definição 13. *Dizemos que uma aplicação polinomial homogênea de grau n H está* ajustada *se, quando posta na forma (5.4), $\chi(b_I) = \chi(I)$ para todo I com $|I| = n$.*

A idéia intuitiva por trás dessa definição é a seguinte: se p é um ponto de Δ^m, $\chi(p)$ é um código binário que identifica a face de menor dimensão de Δ^m que contém p. Isso significa que cada ponto de controle b_I pertence a face de código $\chi(I)$.

Proposição 1. *Se H está ajustada, então H é Δ^m-invariante.*

Demonstração. Devemos mostrar que se $w^i = 0$ então $H^i(w) = 0$. Como H está ajustada, $I^i = 0$ implica que $b_I^i = 0$. Portanto

$$w^i = 0 \Rightarrow H^i(w) = \sum_{|I|=n} b_I^i B_I(w) = \sum_{|I|=n, I^i=0} b_I^i B_I(w) = 0.$$

□

Tendo em vista o exposto na seção anterior, resta agora encontrar condições suficientes para que a aplicação H seja localmente injetiva. Antes, porém, vamos explicitar a relação entre os determinantes jacobianos de F e de sua representação homogênea H. Seja $F' := \det(JF)$ e $H' := \det(JH)$ onde

$$JF = \left(\frac{\partial F^i}{\partial X_j}\right) \quad \text{e} \quad JH = \left(\frac{\partial H^i}{\partial W_j}\right).$$

Proposição 2. *Seja H a representação homogênea de F de grau $n = \deg(F)$. Se $p \in \mathbb{R}^m$ e $w \in A^m$ satisfazem a relação (5.2), então*

$$nF'(p) = H'(w).$$

Demonstração. Por (5.2), (5.3) e pela regra da cadeia, temos que a matriz

$$L = P_\sigma JH(w) P_\sigma^{-1}$$

possui a seguinte decomposição em blocos

$$\begin{pmatrix} u & v \\ q & JF(p) \end{pmatrix}.$$

Pela equação (5.5), temos que

$$\sum_i \partial_j H^i(W_0, \ldots, W_m) = n(W_0 + \ldots + W_m)^{n-1}$$

para $j = 0, \ldots, m$. Segue que o produto $P_\sigma JH(w)$ pode ser escrito em blocos como

$$\begin{pmatrix} n\mathbb{1} \\ Q \end{pmatrix}.$$

Assim, a primeira linha da matriz L satisfaz a relação

$$\begin{pmatrix} u & v \end{pmatrix} P_\sigma = n\mathbb{1}.$$

Resolvendo esse sistema, concluimos que $v = 0$ e $u = n$. Portanto,

$$H'(w) = \det P_\sigma H'(w) \det P_\sigma^{-1} = \det L = u \det JF(p) = nF'(p).$$

□

A proposição acima mostra em particular que se H' é positiva em Δ^m, então F' é positiva em σ. O teorema a seguir, por sua vez, nos fornece condições necessárias e suficientes para a positividade de H'.

Teorema 18. *(Pólya) Seja $H \in \mathbb{R}[W]$ um polinômio homogêneo de grau n. Então $H(w) > 0$ para todo $w \in \Delta^m$ se e somente se $H \cdot (W^0 + \ldots + W^m)^N$ tem para algum $N \in \mathbb{N}$ a forma*

$$\sum_{|K|=n+N} a_K W^K$$

com $a_K > 0$.

Demonstração. Uma demonstração encontra-se em [29] (*Satz 3.6*). Esse resultado não é desconhecido na literatura de modelagem geométrica: no fundo trata-se de uma outra formulação da propriedade aproximativa da operação de elevação de grau ("degree elevation"), confira em [9]. □

Mais detalhes sobre a importância desse teorema podem ser encontrados em [30]. Para aplicá-lo no nosso contexto, devemos escrever H' de forma conveniente. A fim de abreviar as contas, vamos introduzir algumas notações.

Dadas duas matrizes $A = (a_{ij}), B = (b_{ij}) \in \mathbb{R}^{m \times m}$, definimos $A \odot B := (a_{ij} b_{ij})$, ou seja, cada entrada de $A \odot B$ é o produto das entradas de A e B. Em geral, o determinante de $A \odot B$ não possui nenhuma relação com os determinantes de A e B. Mas, se existe uma constante $c \in \mathbb{R}$ tal que $\prod_{i=1}^m b_{i\sigma(i)} = c$ para toda permutação $\sigma \in S_m$, é fácil ver que $\det(A \odot B) = c \det(A)$.

Proposição 3.

$$H' = \sum_{|K|=(m+1)(n-1)} a_K W^K$$

onde

$$a_K = \sum_{\substack{I_0 > \ldots > I_m \\ I_0 + \ldots + I_m = K+1}} \det(b_{I_0}, \ldots, b_{I_m}) \det(I_0, \ldots, I_m) \prod_{i=0}^m \binom{n}{I_i}.$$

Demonstração. Podemos escrever H como um produto de matrizes

$$H = (b_I^i)_{(m+1)\times p}(B_I)_{p\times 1},$$

onde $p = \#\{I = (I^0, \ldots, I^m) \in \mathbb{N}^{m+1} \,||I| = n\}$. Portanto

$$JH = (b_I^i)(\nabla B_I) = (b_I^i)\left((I^j) \odot (\binom{n}{I} W^{I-e_j})\right).$$

Pela fórmula de Cauchy-Binet e pela observação acima,

$$H' = \sum_{I_0>\ldots>I_m} \det(b_{I_0}, \ldots, b_{I_m})\left(\det(I_0, \ldots, I_m)\prod_{i=0}^{m}\binom{n}{I_i} W^{\sum_{i=0}^m I_i - \mathbb{1}}\right),$$

logo

$$H' = \sum_{|K|=(m+1)(n-1)} a_K W^K$$

com

$$a_K = \sum_{\substack{I_0>\ldots>I_m \\ I_0+\ldots+I_m=K+1}} \det(b_{I_0}, \ldots, b_{I_m}) \det(I_0, \ldots, I_m) \prod_{i=0}^{m}\binom{n}{I_i}.$$

□

Na seqüência, vamos determinar condições suficientes para uma aplicação homogênea H de grau n ser um difeomorfismo Δ^m-invariante, para algumas combinações de n e m.

5.2.1 Caso unidimensional ($m = 1$)

O caso mais simples a se considerar é o unidimensional. Primeiramente, devemos notar que a ordem lexicográfica aplicada aos pontos de Δ^1 fornece a mesma informação que a ordem usual no intervalo $[0, 1]$, ou seja, se $b_I >_{\text{lex}} b_J$, com $b_I, b_J \in \Delta^1$, então b_I está "à direita" de b_J. A proposição a seguir mostra que se os pontos de controle guardarem suas posições relativas, então H é um difeomorfismo simplicial (figura 5.1).

Proposição 4. *Seja $H = \sum_{|I|=n} b_I B_I$ uma aplicação polinomial ajustada. Se os pontos de controle estão ordenados, ou seja, se $I >_{lex} J \Rightarrow b_I >_{lex} b_J$, então $H|_{\Delta^1}$ é um difeomorfismo Δ^1-invariante.*

Demonstração. Basta notar que

$$b_I >_{\text{lex}} b_J \Leftrightarrow \det(b_I, b_J) > 0.$$

O resultado segue do Teorema de Pólya e da proposição (3). □

A proposição acima é válida para aplicações unidimensionais de qualquer grau n, mas é possível obter melhores resultados em casos particulares. Para $n = 2$, é fácil ver que se H está ajustada então os pontos de controle estão

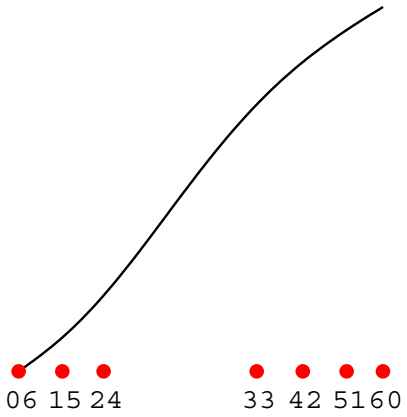


Figura 5.1: Uma difeomorfismo de grau 6. Note que os pontos de controle estão ordenados.

ordenados. Mais interessante é o caso $n = 3$, onde é suficiente que H esteja ajustada, pois suponha que $b_{21} < b_{12}$. Substituindo as relações

$$\det(b_{12}, b_{03}) = \det(b_{21}, b_{03}) + \det(b_{12}, b_{21})$$

$$\det(b_{30}, b_{03}) = \det(b_{21}, b_{03}) + \det(b_{12}, b_{21}) + \det(b_{30}, b_{12})$$

$$\det(b_{30}, b_{21}) = \det(b_{12}, b_{21}) + \det(b_{30}, b_{12})$$

na expressão de H', obtemos

$$H' = 9(\det(b_{21}, b_{03})(W^{22} + 2W^{13} + W^{04}) +$$

$$\det(b_{12}, b_{21})(W^{40} - 2W^{22} + W^{04}) + \det(b_{30}, b_{12})(W^{40} + 2W^{31} + W^{22})).$$

Não é difícil ver que H' é positiva em Δ^1, uma vez que a expressão

$$(W^{40} - 2W^{22} + W^{04})$$

é positiva em Δ^1, exceto no ponto $(\frac{1}{2}, \frac{1}{2})$, onde é zero. Mas nesse ponto os outros termos são positivos.

Já no caso $n = 4$, o contra-exemplo exibido na figura 5.2 mostra que H estar ajustada não é suficiente para que seja um difeomorfismo Δ^1-invariante. Note que esse resultado se aplica igualmente para dimensões superiores.

5.2.2 Caso quadrático ($n = 2$)

A fim de mostrar que toda aplicação polinomial H de grau dois ajustada é um difeomorfismo Δ^m-invariante, para qualquer $m > 0$, vamos provar alguns lemas de caráter combinatório.

Lema 19. *Seja $A = (a_{ij}) \in \mathbb{R}^{m \times m}$ uma matriz de zeros e uns cujas linhas e colunas possuem exatamente dois uns tal que* $\det(A) \neq 0$. *Então existem*

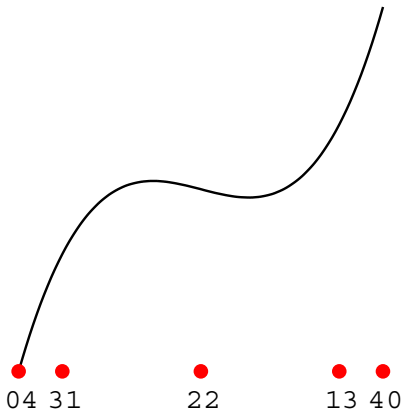


Figura 5.2: Uma função ajustada de grau 4 que não é injetiva. Note que os pontos de controle b_{13} e b_{31} trocaram de lugar.

permutações $\pi, \gamma \in S_m$*, às quais correspondem matrizes* $P = (p_{ij})$ *e* $C = (c_{ij})$*, com* $p_{ij} = \delta_{\pi(i),j}$ *e* $c_{ij} = \delta_{\gamma(i),j}$*, tais que*

1. $PA = \mathrm{Id} + C$*;*
2. γ *não tem pontos fixos;*
3. *Se* $\gamma = \gamma_1 \gamma_2 \dots \gamma_k$*, onde os* γ_i *são ciclos disjuntos, cada* γ_i *é uma permutação par e* $\det(\mathrm{Id} + C) = 2^k$*;*
4. *Se* $B = (b_{ij}) \in \mathbb{R}^{m \times m}$ *é uma matriz tal que* $b_{ij} \geq 0$ *e* $b_{ij} = 0 \Leftrightarrow a_{ij} = 0$*, então* $\det(A)\det(B) > 0$*.*

Demonstração.

1. Existe uma matriz P tal que PA possui apenas uns na diagonal, pois caso contrário, pela própria definição de determinante, $\det(A)$ seria igual a zero. Defina, portanto, $C = PA - \mathrm{Id}$. Como A tem exatamente dois uns em cada linha e coluna, C tem exatamente um um em cada linha e coluna, logo é uma matriz de permutação;
2. Como a diagonal de C possui apenas zeros, γ não tem pontos fixos;
3. Seja $D = \mathrm{Id} + C$. Temos que

$$d_{ij} = [i = j] + [\gamma(i) = j].$$

Pela definição de determinante,

$$\det(D) = \sum_{\sigma \in S_m} \mathrm{sgn}(\sigma) \Pi_i ([i = \sigma(i)] + [\gamma(i) = \sigma(i)]).$$

Mas

$$\forall i, [i = \sigma(i)] + [\gamma(i) = \sigma(i)] = 1 \Leftrightarrow \sigma = \gamma_{k_1} \dots \gamma_{k_j},$$

onde $\{k_1, \ldots, k_j\} \subset \{1, \ldots, k\}$. Segue que

$$\det(D) = \sum_{\{k_1,\ldots,k_j\}\subset\{1,\ldots,k\}} \operatorname{sgn}(\gamma_{k_1} \ldots \gamma_{k_j}) = \Pi_{i=1}^{k}(1 + \operatorname{sgn}(\gamma_i)).$$

Portanto cada γ_i é uma permutação par, pois caso contrário $\det(D)$ seria zero, e $\det(D) = 2^k$.

4.
$$\det(A)\det(B) = \det(P)\det(A)\det(P)\det(B) = \det(D)\det(E)$$

onde $E = PB$ é tal que $e_{ij} > 0 \Leftrightarrow d_{ij} > 0$. Assim as permutações σ tais que $\Pi_{i=1}^{m} e_{i\sigma(i)} \neq 0$ são exatamente aquelas da forma $\sigma = \gamma_{k_1} \ldots \gamma_{k_j}$. Como cada γ_i é par, $\det(E) > 0$, e portanto $\det(A)\det(B) > 0$.

□

Corolário 20. *Seja $A = (a_{ij}) \in \mathbb{R}^{m\times m}$ uma matriz de zeros e uns cujas linhas possuem no máximo dois uns e tal que $\det(A) \neq 0$. Se $B = (b_{ij}) \in \mathbb{R}^{m\times m}$ é uma matriz tal que $b_{ij} \geq 0$ e $b_{ij} = 0 \Leftrightarrow a_{ij} = 0$, então $\det(A)\det(B) > 0$.*

Demonstração. Se A possui exatamente dois uns por linha e coluna, podemos aplicar o lema anterior. Caso contrário, podemos permutar linhas e colunas de A de modo que

$$P_1AQ_1 = \begin{pmatrix} 1 & 0 \\ q & A_1 \end{pmatrix} \quad \text{ou} \quad P_1AQ_1 = \begin{pmatrix} 1 & q \\ 0 & A_1 \end{pmatrix}$$

para certas matrizes de permutação P_1 e Q_1. Note que $\det(A) = s_1 \det(A_1)$, com $s_1 \in \{-1, 1\}$. Procedendo indutivamente, chegamos a uma matriz A_n que possui exatamente dois uns por linha e coluna, tal que

$$\det(A) = s_n \det(A_n).$$

Aplicando o mesmo procedimento a matriz B, concluimos que

$$\det(A)\det(B) = (s_n \det(A_n))(s_n b_1 \ldots b_k \det(B_n)) =$$

$$s_n^2(b_1 \ldots b_k)(\det(A_n)\det(B_n)) > 0$$

onde os b_i são certos elementos da matriz B. □

Lema 21. *Dado um vetor $K = (K^0, \ldots, K^m) \in \mathbb{N}^{m+1}$, com $|K| = m + 1$, sempre é possível encontrar vetores $I_0, \ldots, I_m \in \mathbb{N}^{m+1}$, com $|I_i| = 2$, tais que*

$$I_0 + \ldots + I_m = K + \mathbb{1}$$

e $\det(I_0, \ldots, I_m) \neq 0$.

Demonstração. Por indução em m. O caso $m = 0$ é trivial. Podemos supor sem perda de generalidade que $K^i \geq K^j$ se $i < j$. Temos dois casos possíveis: ou $K^m = 0$ ou $K^m = 1$. Se $K^m = 1$, colocamos $I_j^i = 2\delta_{ij}$, e $\det(I_0, \ldots, I_m) = 2^{m+1} > 0$. Se $K^m = 0$, definimos $K' = (K^0 - 1, \ldots, K^{m-1})$

e, aplicando a hipótese de indução, obtemos vetores $I'_j \in \mathbb{N}^m$, com $|I'_j| = 2$ e $\det(I'_0, \ldots, I'_{m-1}) > 0$. Agora basta colocar $I_j = (I'_j, 0)$ para $j = 0, \ldots, m-1$ e I_m satisfazendo $I^i_m = [i = 0 \text{ ou } i = m]$. Utilizando a expansão de Laplace, concluimos $\det(I_0, \ldots, I_m) = \det(I'_0, \ldots, I'_{m-1}) > 0$. □

Proposição 5. *Seja $H = \sum_{|I|=2} b_I B_I$ uma aplicação de grau 2 ajustada. Então $H|_{\Delta^m}$ é um difeomorfismo Δ^m-invariante.*

Demonstração. A idéia é mostrar que o coeficiente a_K da proposição 3 é positivo. Inicialmente vamos provar que $a_K \geq 0$. Note que

$$\det(I_0, \ldots, I_m) \prod_{i=0}^{m} \binom{2}{I_i} = \det(\binom{2}{I_0} I_0, \ldots, \binom{2}{I_m} I_m) = 2^{m+1} \det(A),$$

onde A é uma matriz de zeros e uns com no máximo dois uns por linha. Segue do corolário 20 que $a_K \geq 0$. Mas o lema 21 implica que ao menos um termo de a_K é positivo. Logo, $a_K > 0$ e portanto $H|_{\Delta^m}$ é um difeomorfismo Δ^m-invariante. □

5.2.3 Um ponto de controle livre

Vamos agora estudar um caso bem interessante e útil. O que acontece quando, intuitivamente, permitimos que apenas um ponto de controle associado a uma face de Δ^m se mova ? Será necessário provar o seguinte lema:

Lema 22. *Se $M = \mathrm{Id} + uv^T + ab^T$, com $v^T a = 0$, então*

$$\det M = (1 + v^T u)(1 + b^T a).$$

Demonstração. A identidade

$$\det(\mathrm{Id} + uv^T) = 1 + v^T u$$

é válida em geral, donde se conclui que

$$\det(N + ab^T) = \det(N^{-1}) \det(N + ab^T) \det(N) =$$

$$\det(\mathrm{Id} + N^{-1} ab^T) \det(N) = \det(N) + b^T(\det(N) N^{-1}) a,$$

para N inversível. Pondo $N = \mathrm{Id} + uv^T$, temos que

$$N^{-1} = \mathrm{Id} - \frac{1}{1 + v^T u} uv^T,$$

e

$$\det(M) = \det(N + ab^T) = (1 + v^T u) + b^T((1 + v^T u)\,\mathrm{Id} - uv^T)a =$$
$$= (1 + v^T u)(1 + b^T a).$$

□

Proposição 6. *Seja $H = \sum_{|I|=n} b_I B_I$ uma aplicação de grau n ajustada e $J = (J^0, \ldots, J^m) \in \mathbb{N}^{m+1}$ com $|J| = n$ e $J^i \in \{0, 1\}$. Se $b_I = I/n$ para $I \neq J$, então $H|_{\Delta^m}$ é um difeomorfismo Δ^m-invariante.*

Demonstração. Temos que

$$H = \sum_{|I|=n} \frac{I}{n} B_I + b_J B_J - \frac{J}{n} B_J =$$

$$(W_0 + \ldots + W_m)^{n-1} \begin{pmatrix} W_0 \\ \vdots \\ W_m \end{pmatrix} + (b_J - \frac{J}{n}) B_J.$$

Segue que

$$JH = (W_0 + \ldots + W_m)^{n-1} (\mathrm{Id} + uv^T + ab^T),$$

com

$$u = \frac{n-1}{W_0 + \ldots + W_m} \begin{pmatrix} W_0 \\ \vdots \\ W_m \end{pmatrix},$$

$$v^T = \begin{pmatrix} 1 & \cdots & 1 \end{pmatrix},$$

$$a = \frac{n}{(W_0 + \ldots + W_m)^{n-1}} (b_J - J/n) \text{ e}$$

$$b^T = \begin{pmatrix} J^0 B_{J-e_0} & \cdots & J^m B_{J-e_m} \end{pmatrix}.$$

Do lema resulta que

$$H' = n(W_0 + \ldots + W_m)^{(m+1)(n-1)} (1 + b^T \frac{n}{(W_0 + \ldots + W_m)^{n-1}} (b_J - J/n)) =$$

$$n(W_0 + \ldots + W_m)^{m(n-1)-1} ((W_0 + \ldots + W_m)^n + n(W_0 + \ldots + W_m) b^T (b_J - J/n)) =$$

$$n(W_0 + \ldots + W_m)^{m(n-1)-1} E,$$

onde

$$E = \sum_{|I|=n} B_I + D(b_J - J/n) \text{ e}$$

$$D = \begin{pmatrix} \sum_i J^0 (1 - \delta_{i0} + J^i) B_{J-e_0+e_i} & \cdots & \sum_i J^m (1 - \delta_{im} + J^i) B_{J-e_m+e_i} \end{pmatrix}.$$

Ou seja,

$$E = \sum_{|I|=n} B_I + \sum_j \sum_i J^j (1 - \delta_{ij} + J^i) B_{J-e_j+e_i} (b_J^j - J^j/n) =$$

$$\sum_{|I|=n} B_I + \sum_j \sum_{i \neq j} J^j (1 + J^i) B_{J-e_j+e_i} (b_J^j - J^j/n) + \sum_j (J^j)^2 B_J (b_J^j - J^j/n) =$$

$$\sum_{|I|=n} B_I + \sum_j \sum_{i \neq j} J^j (1 + J^i) B_{J-e_j+e_i} (b_J^j - J^j/n) + B_J \sum_j (b_J^j - J^j/n) =$$

$$\sum_{|I|=n} B_I + \sum_j \sum_{i \neq j} J^j (1 + J^i) B_{J-e_j+e_i} (b_J^j - J^j/n).$$

Analisando os coeficientes de E, verificamos que, para $i \neq j$,

$$\mathrm{Coef}(E, B_{J-e_j+e_i}) = 1 + J^j (1 + J^i)(b_J^j - J^j/n) =$$

$$(1 - J^j(1 + J^i)/n) + J^j(1 + J^i)b_J^j =$$
$$(1 - J^j(1 + J^i)/n) + (1 + J^i)b_J^j > 0,$$

já que $n > 1$, $J^i \in \{0, 1\}$ e $b_j^j = 0 \Leftrightarrow J^j = 0$. Como $\mathrm{Coef}(E, B_I) = 1$ para os demais coeficientes, segue do teorema de Pólya que H' é positiva em Δ^m, e portanto $H|_{\Delta^m}$ é um difeomorfismo Δ^m-invariante. □

Vamos detalhar um pouco mais o significado dessa proposição. Como mencionamos anteriormente, cada face de Δ^m pode ser identificada por um código binário, conforme o número de coordenadas não nulas de seus pontos. Assim, para $m = 3$ por exemplo, temos

$$(1111) \rightarrow \text{3-simplexo}$$

$$(1110), (1101), (1011), (0111) \rightarrow \text{2-simplexos}$$

$$(1100), (1010), (1001), (0110), (0101), (0011) \rightarrow \text{1-simplexos}$$

$$(1000), (0100), (0010), (0001) \rightarrow \text{0-simplexos}$$

$$(0000) \rightarrow \text{simplexo vazio}$$

Seja G_J uma aplicação polinomial tal que

$$G_J^i = W_i + (b_J^i - J^i/|J|)B_J,$$

onde $J^i \in \{0, 1\}$. Examinando a proposição, vemos que

$$G_J^i = H^i \pmod{\mathcal{M}},$$

portanto G_J e H definem a mesma aplicação em Δ^m e, como foi demonstrado, G_J é um difeomorfismo Δ^m-invariante.

Além disso, pela própria definição, G_J somente afeta os pontos da face δ_J cujo código é J, bem como os pontos das faces incidentes a δ_J que possuam a dimensão maior que a de δ_J. Ou seja, o ponto de controle b_J efetivamente controla a ação de G_J sobre δ_J.

Podemos agora compor os difeomorfismos G_J e assim definir um difeomorfismo G que depende de um certo número de parâmetros associados a cada face de Δ^m. No caso $m = 3$, podemos definir, por exemplo,

$$G = G_{0011}G_{0101}G_{0110}G_{1001}G_{1010}G_{1100}G_{0111}G_{1011}G_{1101}G_{1110}G_{1111}.$$

Fazendo as contas, vemos que G depende de 17 parâmetros independentes, 1 para cada 1-simplexo, 2 para cada 2-simplexo e 3 para o 3-simplexo. Esse esquema é perfeitamente geral e pode ser estendido para qualquer dimensão.

Por outro lado, como as aplicações G_J não comutam em geral, a aplicação resultante G depende da ordem da aplicação das G_J. Isso introduz uma assimetria com relação ao efeito dos pontos de controle que pode ser indesejável em algumas aplicações.

5.2.4 Esquema estratificado

Para superar esse problema, vamos introduzir um *esquema estratificado* no qual G pode ser escrita como uma composição de m aplicações G_j,

$$G = G_1 G_2 \dots G_m,$$

onde,

$$G_j^i = W_i + \sum_{|J|=j+1, J^k \in \{0,1\}} (b_J^i - J^i/|J|) B_J.$$

A idéia é que cada G_j sintetiza o efeito combinado dos pontos de controle associados aos j-simplexos de Δ^m. Mas o que podemos afirmar sobre as aplicações G_j ? São elas difeomorfismos Δ^m-invariantes ?

Note que G_j define a mesma aplicação em Δ^m que a aplicação homogênea de grau $j+1$ H_j, com

$$H_j^i = W_i (W_0 + \dots + W_m)^j + \sum_{|J|=j+1, J^k \in \{0,1\}} (b_J^i - J^i/|J|) B_J.$$

Assim, pelas proposições 5 e 6, resulta que G_j é um difeomorfismo Δ^m-invariante para $j = 1$ e $j = m$. Mas o que pode ser dito para outros valores de j ?

Uma abordagem possível é encarar o determinante jacobiano H_j' como uma função também dos pontos de controle b_J. Como cada b_J está limitado a uma face j-dimensional de Δ^m, resulta que H_j' pode ser vista como uma função definida no produto de simplexos

$$\Delta^m \times \underbrace{\Delta^j \times \dots \times \Delta^j}_{r \text{ vezes}},$$

onde r é o número de simplexos j-dimensionais de Δ^m. Antes de seguir esse caminho, será necessário introduzir algumas notações e discutir alguns fatos relacionados à positividade de funções polinomiais definidas em produtos de simplexos.

Seja $P\colon \Delta^{m_1} \times \Delta^{m_2} \dots \times \Delta^{m_r} \to \mathbb{R}$ uma função polinomial definida num produto de r simplexos de dimensões $m_1, \dots, m_r$, dada por

$$(u_1, \dots, u_r) \mapsto P(u_1^0, \dots, u_1^{m_1}, u_2^0, \dots, u_2^{m_2}, \dots, u_r^0, \dots, u_r^{m_r}),$$

com $P \in \mathbb{R}[U] := \mathbb{R}[U_{10}, \dots, U_{1m_1}, U_{20}, \dots, U_{2m_2}, \dots, U_{r0}, \dots, U_{rm_r}]$.

Pondo $\mathcal{U} := \langle \sum_{l=0}^{m_i} U_{il} - 1; i = 1, \dots, r \rangle$, vemos que se dois polinômios $P, Q \in \mathbb{R}[U]$ são tais que

$$P = Q \pmod{\mathcal{U}},$$

então eles definem a mesma aplicação em $\Delta^{m_1} \times \Delta^{m_2} \dots \times \Delta^{m_r}$, devido ao fato que $\sum_l u_i^l = 1$

Dada uma r-upla $N = (n^1, \dots, n^r) \in \mathbb{N}^r$, dizemos que $P \in \mathbb{R}[U]$ é N-homogênea se

$$P = \sum_{|I_1|=n^1} \dots \sum_{|I_r|=n^r} a_{I_1 \dots I_r} U_1^{I_1} \dots U_r^{I_r}.$$

com $U_i^I := (U_{i0})^{I^0} \ldots (U_{im_i})^{I^{m_i}}$.

Podemos agora enunciar uma generalização do Teorema de Pólya para aplicações polinomiais definidas em um produto de simplexos.

Teorema 23. *Seja $H \in \mathbb{R}[U]$ um polinômio N-homogêneo com $N = (n^1, \ldots, n^r)$. Então $H(u_1, \ldots, u_r) > 0$ para todo $(u_1, \ldots, u_r) \in \Delta^{m_1} \times \Delta^{m_2} \ldots \times \Delta^{m_r}$ se e somente se*

$$H \cdot (U_{10} + \ldots + U_{1m_1})^{R_1} \ldots (U_{r0} + \ldots + U_{rm_r})^{R_r}$$

tem para algum $R = (R_1, \ldots, R_r) \in \mathbb{N}^r$ a forma

$$\sum_{|K_1|=n^1+R_1} \cdots \sum_{|K_r|=n^r+R_r} a_{K_1 \ldots K_r} U_1^{K_1} \ldots U_r^{K_r},$$

com $a_{K_1 \ldots K_r} > 0$.

Demonstração. É o *Satz 3.54* de [29]. □

Seja $C_{m,n} = \{C \in \mathbb{N}^n | 0 \leq C^0 < C^1 < \ldots < C^{n-1} < m\}$, ou seja, $C_{m,n}$ representa as combinações dos m objetos $\{0, 1, \ldots, m-1\}$ n a n. Esse conjunto pode ser linearmente ordenado pela ordem lexicográfica em $\mathbb{N}^n$, portanto vamos denotar por $C_{m,n,i}$ o i-ésimo elemento de $C_{m,n}$, onde i varia entre 1 e $\binom{m}{n}$.

Existe também uma clara bijeção entre o conjunto dos $J \in \mathbb{N}^m$ tais que $J^k \in \{0, 1\}$ e $|J| = n$ e $C_{m,n}$. Vamos denotar por $J(C)$ a imagem de $C \in C_{m,n}$ por tal bijeção, ou seja, $J(C) = (J^0, \ldots, J^{m-1})$, com $J^i = [\exists l, C^l = i]$.

Voltando a questão do esquema estratificado, vamos parametrizar os pontos de controle b_J em G_j em termos das indeterminadas U. Inicialmente, vamos por $r = \binom{m+1}{j+1}$ e $m_1 = m_2 = \ldots = m_r = j$. No caso r conta o número de faces j-dimensionais de Δ^m. A parametrização é a seguinte:

$$b^k_{J(C_{m+1,j+1,i})} = \begin{cases} U_{il} & \text{, se } C^l_{m+1,j+1,i} = k \\ 0 & \text{, caso contrário.} \end{cases}$$

Substituindo esses b_J nas expressões H_j e aplicando a proposição 3, temos que os coeficientes a_K de H'_j são agora polinômios em $\mathbb{R}[U]$. A idéia é utilizar o teorema de Pólya generalizado para mostrar que $a_K > 0$, donde se conclui que G_j é um difeomorfismo Δ^m-invariante. Para tanto, devemos primeiramente escrever a_K como um polinômio homogêneo.

Vamos definir

$$d(I_0, \ldots, I_m) = \prod_{i=1}^{r} (\sum_{l=0}^{j} U_{il})^{t(i, I_0, \ldots, I_m)} \det(b_{I_0}, \ldots, b_{I_m}),$$

onde $t(i, I_0, \ldots, I_m)$ é um predicado binário que informa apenas se

$$J(C_{m+1,j+1,i}) = I_k$$

para algum k, ou seja,

$$t(i, I_0, \ldots, I_m) = \begin{cases} 0 & \text{, se } \exists k, J(C_{m+1,j+1,i}) = I_k \\ 1 & \text{, caso contrário.} \end{cases}$$

Utilizando as propriedades do determinante, concluimos que $d(I_0, \ldots, I_m)$ é $\mathbb{1}$-homogênea. Além disso,

$$d(I_0, \ldots, I_m) = \det(b_{I_0}, \ldots, b_{I_m}) \pmod{\mathcal{U}}.$$

Portanto se definirmos

$$u_K = \sum_{\substack{I_0 > \ldots > I_m \\ I_0 + \ldots + I_m = K+1}} d(I_0, \ldots, I_m) \det(I_0, \ldots, I_m) \prod_{i=0}^{m} \binom{n}{I_i},$$

temos também que u_K é $\mathbb{1}$-homogênea e que

$$u_K = a_K \pmod{\mathcal{U}}.$$

É importante frisar que u_K é em geral um polinômio com $(j+1)^r$ termos.

Vamos aplicar essas idéias no importante caso tridimensional, ou seja, para $m = 3$. Já sabemos que G_1 e G_3 são difeomorfismos Δ^3-invariantes, resta provar que G_2 também o é. Seguindo a receita para parametrizar os pontos de controle, temos que

$$b_{1110} = (U_{10}, U_{11}, U_{12}, 0), b_{1101} = (U_{20}, U_{21}, 0, U_{22}),$$

$$b_{1011} = (U_{30}, 0, U_{31}, U_{32}) \text{ e } b_{0111} = (0, U_{40}, U_{41}, U_{42}).$$

A seguir, utilizando o pacote de computação simbólica ASIR[27], nós calculamos os polinômios u_K para todo $K \in \mathbb{N}^4$ com $|K| = 8$. Verificamos que cada um destes 165 polinômios possui exatamente $3^{\binom{4}{3}} = 81$ termos, todos possuindo coeficientes positivos. Provamos assim o seguinte fato:

Proposição 7. *G_1, G_2 e G_3 são difeomorfismos Δ^3-invariantes.*

Acreditamos que este resultado possa ser generalizado para $m > 3$, entretanto as contas envolvidas tornam-se mais complexas. Por exemplo, para $m = 4$ os polinômios u_K correspondentes a G^2 possuem $3^{\binom{5}{3}} = 59049$ termos.

5.3 Conclusão

Neste capítulo, nos concentramos em estudar certos aspectos teóricos dos difeomorfismos simpliciais, principalmente dos difeomorfismos simpliciais polinomiais. O próximo capítulo será dedicado a aplicação desses difeomorfismos na modelagem de objetos gráficos, especialmente no caso dos isocomplexos. Antes de prosseguir, vamos levantar algumas questões que surgiram durante a elaboração deste capítulo e que merecem ser consideradas futuramente.

Esquema estratificado O esquema estratificado funciona para $m > 3$? Ou seja, as aplicações G_j são sempre difeomorfismos simpliciais ?

Caso cúbico Se $H = \sum_{|I|=3} b_I B_I$ é uma aplicação de grau 3 ajustada então é verdade que $H|_{\Delta^m}$ é um difeomorfismo Δ^m-invariante ? Já verificamos

isso para $m = 1$. Usando técnicas simbólicas semelhantes as empregadas na sub-seção 5.2.4, verificamos que no caso $m = 2$, se os pontos de controle estão restritos a uma certa faixa de atuação (por exemplo, $b_I^i > 1/20$ para I diferente de e_0, e_1 e e_2) então de fato isso é verdade.

Condições suficientes para o caso geral É possível encontrar condições suficientes razoavelmente fáceis de se verificar para o caso geral ?

Aproximação polinomial Qualquer difeomorfismo simplicial pode ser aproximado por um difeomorfismo simplicial polinomial ? E por composições de polinômios do tipo G_j ?

Difeomorfismos simpliciais racionais Seja Q a aplicação racional

$$Q = \frac{1}{\sum_{|I|=n} q_I B_I} \sum_{|I|=n} q_I b_I B_I,$$

onde os q_I são certos pesos associados aos pontos de controle b_I. Dizemos que Q está ajustada se $\chi(b_I) = \chi(I)$. É fácil ver que se Q está ajustada então Q é uma função Δ^m-invariante. Sob que condições Q é um difeomorfismo Δ^m-invariante ? Note que a aplicação Y_y que apareceu no lema 13 é um difeomorfismo simplicial racional.

Capítulo 6

Aplicações

Vamos, finalmente, neste capítulo de aplicações, juntar as peças do quebra-cabeças e mostrar como os conceitos tratados nos capítulos anteriores podem ser sintetizados para dar origem a uma nova maneira de se representar objetos gráficos. Antes de descrever as aplicações, vamos definir exatamente o problema que estamos atacando.

Classicamente, objetos gráficos possuem dois tipos básicos de representação. Na representação *implícita*, um objeto é representado como o conjunto solução de uma certa equação. Na representação *paramétrica*, um objeto é representado por uma coleção de pedaços que o cobrem e que estão colados uns aos outros. Nosso objetivo é definir uma representação que é, de certa forma, a um só tempo implícita e paramétrica.

A idéia é deformar o isocomplexo $O = (K, f)$, onde K é uma malha conforme n-dimensional no $\mathbb{R}^n$, com a ajuda do difeomorfismo K-invariante X, obtendo a hipersuperfície

$$\mathcal{O} = (\overline{f} \circ X)^{-1}(0) = X^{-1}(\overline{f}^{-1}(0)),$$

que denominamos *isocomplexo curvilinear* dado por (K, f, X). Aqui $\overline{f}$ denota a função definida em $|K|$ através da interpolação linear dos valores de f nos vértices de K. Como essa deformação é bijetiva, contínua e preserva as relações de incidência de K, resulta que $\mathcal{O}$ tem a mesma topologia de O.

As duas igualdades na definição acima possuem interpretações ligeiramente diferentes. No primeiro caso,

$$\mathcal{O} = (\overline{f} \circ X)^{-1}(0),$$

é claramente um objeto implícito, sendo o conjunto dos pontos $p \in \mathbb{R}^n$ que satisfazem a equação

$$\overline{f}(X(p)) = 0.$$

No segundo caso,

$$\mathcal{O} = X^{-1}(\overline{f}^{-1}(0)),$$

diz que o conjunto $\mathcal{O}$ está coberto por uma série de parametrizações da forma $X^{-1}(\theta)$, onde θ é uma isocélula de O, sendo portanto um objeto paramétrico. Por isso dizemos que esta representação é *implícita-paramétrica*.

Ainda é possível fazer outra distinção, baseada no fato que normalmente é mais difícil calcular o valor de $X^{-1}(p)$ do que o valor de $X(p)$. Poderíamos ter escrito

$$\mathcal{O} = (\overline{f} \circ X^{-1})^{-1}(0) = X(\overline{f}^{-1}(0)) = \bigcup_{\theta \in O} X(\theta),$$

onde os papéis de X e de sua inversa aparecem trocados. Dizemos que esta representação é *paramétrica-implícita*.

Para ver a diferença entre essas formulações, vamos considerar dois problemas típicos: (1) determinar se um ponto $p \in \mathbb{R}^n$ está dentro ou fora de $\mathcal{O}$ e (2) calcular um ponto $q \in \mathcal{O}$. No caso implícito-paramétrico, para resolver (1), inicialmente descobrimos qual célula $\sigma \in K$ contém o ponto p, calculamos $q = X|_\sigma(p)$ e avaliamos o sinal de $\overline{f}_\sigma(q)$. Para o problema (2), escolhemos um ponto $p \in |O|$, isso pode ser feito escolhendo-se uma isocélula $\theta \in O$ e aplicando-se a parametrização 3.1, e calculamos $q = X^{-1}(p)$. Nesse caso o teste de pertinência foi feito com o difeomorfismo direto e a amostragem com o difeomorfismo inverso, o que lembra mais a representação implícita, pois tivemos que resolver uma equação para encontrar um ponto sobre $\mathcal{O}$, no caso a equação $X(q) = p$. Intercambiando o difeomorfismo direto e o inverso, obtemos a representação paramétrica-implícita, na qual a amostragem usa o difeomorfismo direto e o teste de pertinência usa o inverso, o que lembra mais a representação paramétrica.

Em geral, o difeomorfismo simplicial X depende de um conjunto de *parâmetros intrínsecos* β, de modo que denotaremos o difeomorfismo por X_β quando for necessário explicitar essa dependência. A principal tarefa das aplicações é ajustar esses parâmetros aos dados de entrada. A forma exata dos parâmetros intrínsecos depende do tipo de difeomorfismo.

No caso das aplicações deste capítulo, empregaremos o esquema estratificado para difeomorfismos simpliciais polinomiais, descrito no capítulo anterior. Temos, portanto, que X é obtido pela composição de difeomorfismos simpliciais X_i, $i = 1, \ldots, n$, cada X_i estando associado às faces i-dimensionais de K. Um simplexo i-dimensional de K contribui com um ponto de controle que possui i graus de liberdade. Assim, β pode ser interpretado como um grande vetor com i entradas para cada elemento i-dimensional de K. As estradas estão limitadas a certas faixas de valores, já que os pontos de controle devem estar restritos às suas faces correspondentes.

Podemos ver pela definição dos difeomorfismos simpliciais do esquema estratificado que eles são essencialmente perturbações da identidade. Para aumentar o efeito de cada difeomorfismo, recorremos a iteração de sua aplicação. O difeomorfismo resultante tem a seguinte forma:

$$X = X_1^{j_1} X_2^{j_2} \ldots X_n^{j_n},$$

onde j_i é um inteiro que denota quantas vezes o difeomorfismo X_i é iterado. Tipicamente, empregamos os valores $i_1 = 2$, $i_2 = 4$ e $i_3 = 6$.

A seguir vamos descrever três aplicações que têm em comum o fato de gerarem isocomplexos curvilineares como saída. A primeira aplicação é um programa interativo para modelagem *free-form* de objetos implícitos. A segunda

aplicação produz um isocomplexo curvilinear que aproxima uma hipersuperfície dada analiticamente. Por fim, apresentaremos uma aplicação de reconstrução de hipersuperfícies dadas por pontos. A abordagem será genérica com respeito a dimensão, mas para fixar as as idéias pode-se considerar simplesmente o caso $n = 2$, no qual o objeto implícito $\mathcal{O}$ é uma curva, o hipercubo $[0,1]^n$ é um quadrado e assim por diante.

6.1 Modelagem free-form de objetos implícitos

Vamos descrever brevemente como o usuário interage com a aplicação e em seguida detalhar seu funcionamento.

A idéia é que o usuário inicia a interação visualizando uma malha inicial K_0 bastante grosseira. Ele pode refinar essa triangulação, usando o esquema de subdivisão de Maubach, obtendo assim uma triangulação K, ao mesmo tempo que altera certos parâmetros associados aos vértices e arestas de K, parâmetros estes que vão determinar um isocomplexo curvilinear $\mathcal{O}$ com suporte K.

Em cada vértice v, o usuário pode efetuar três operações: arrastar o vértice, deslocando-o para outra posição; alterar o sinal $s(v)$, fazendo com que o v pertença ao interior do objeto, caso $s(v) = -1$, ou ao exterior, caso $s(v) = +1$; e alterar o valor escalar $r(v)$. Isso permite construir um isocomplexo $O = (K, f)$, onde $f(v) = s(v)r(v)$, que aproxima linearmente o objeto que se deseja modelar.

Em cada aresta ϵ que é inteceptada por O, o usuário tem acesso a dois controles, que são responsáveis por controlar o difeomorfismo K-invariante X: há um controle que pode deslizar pela aresta, dando uma dica do ponto $p(\epsilon)$ onde $\mathcal{O}$ deve interceptar a aresta; o outro é um vetor unitário $n(\epsilon)$, representado por uma flecha, que indica a direção que a normal a $\mathcal{O}$ deve possuir no ponto de $p(\epsilon)$.

Esses parâmetros de modelagem são intuitivos para o usuário, mas devemos calcular os parâmetros intrínsecos β. Para tanto, observamos que, interpretados matematicamente, os parâmetros de modelagem devem satisfazer a

$$\overline{f} \circ X_\beta(p(\epsilon)) = 0 \text{ e } \frac{\nabla(\overline{f} \circ X_\beta)(p(\epsilon))}{\|\nabla(\overline{f} \circ X_\beta)(p(\epsilon))\|} = n(\epsilon),$$

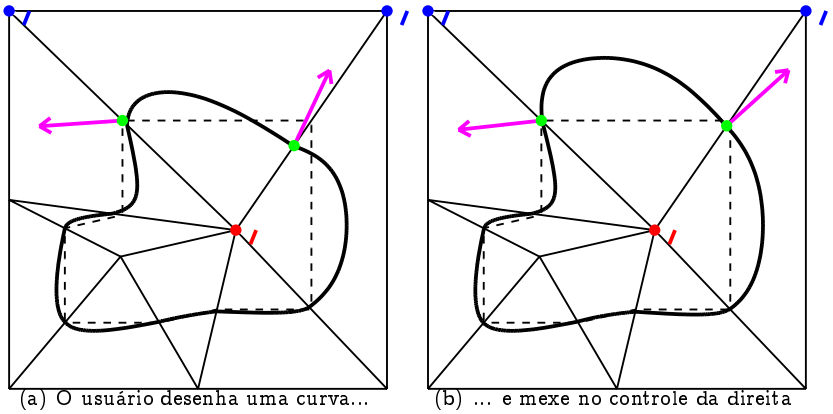
para todas as arestas ϵ que são interceptadas por $\mathcal{O}$, que são aquelas cujos vértices possuem sinais opostos. Vamos definir uma função de erro

$$\mathrm{err}(\sigma, \beta) = \sum_{\substack{\epsilon=\{v_0,v_1\}\in\sigma \\ s(v_0)s(v_1)<0}} \left(\frac{F^2_{\sigma,\beta}(p(\epsilon))}{\|\nabla F_{\sigma,\beta}(p(\epsilon))\|^2} + \alpha \left\| \frac{\nabla F_{\sigma,\beta}(p(\epsilon))}{\|\nabla F_{\sigma,\beta}(p(\epsilon))\|} - n(\epsilon) \right\|^2\right),$$

onde $F_{\sigma,\beta} = \overline{f}_\sigma \circ X_\beta$, que associa a cada célula σ, um erro $\mathrm{err}(\sigma,\beta)$ que mede quanto os parâmetros β satisfazem os requisitos geométricos em cada célula σ. O parâmetro α é um peso que serve para controlar se mais ênfase será dada ao ponto de interseção ou a direção da normal. Para determinar os parâmetros β, simplesmente minimizamos o erro sobre todos as células de K:

$$\min_\beta \sum_{\sigma\in K} \mathrm{err}(\sigma,\beta).$$

Algumas iterações desse programa de otimização são efetuadas a cada vez que o usuário atualiza os parâmetros de modelagem. É possível também rodar o programa de otimização um maior número de iterações para obter um melhor resultado. Na figura 6.1 vemos um exemplo de interação do usuário.



(a) O usuário desenha uma curva... (b) ... e mexe no controle da direita

Figura 6.1: Exemplo de interação na aplicação de modelagem *free-form*.

6.2 Poligonização de objetos implícitos

A entrada da aplicação de poligonização de objetos implícitos é uma função $g\colon [0,1]^n \to \mathbb{R}$, que supomos contínua e diferenciável, e queremos determinar um isocomplexo curvilinear $\mathcal{O}$ tal que

$$\mathcal{O} \subset g^{-1}((-a,a)),$$

para a pequeno, ou seja, queremos que $\mathcal{O}$ esteja numa vizinhança de $g^{-1}(0)$.

A idéia é construir uma seqüência de complexos $\mathcal{O}_i = (K_i, f_i, X_{\beta_i})$ que aproxima $g^{-1}(0)$. Colocamos inicialmente K_0 igual a triangulação canônica de $[0,1]^n$, $f_0(v) = g(v)$, para todo $v \in K_0$, e X_{β_0} igual a identidade. Para descrever o passo indutivo que calcula $\mathcal{O}_{i+1}$ a partir de $\mathcal{O}_i$, precisamos definir a função $\mathrm{err}(\sigma_i, \beta_i)$ que associa a cada célula $\sigma_i \in K_i$ um erro que mede quão próximo $\mathcal{O}$, restrita a σ_i, está de $g^{-1}(0)$.

Seja θ_i a isocélula que possui σ_i como suporte. Calculamos m pontos amostrais $p_1, \ldots, p_m$ sobre θ_i e definimos

$$\mathrm{err}(\sigma_i, \beta_i) = \sum_{p_j \in \theta_i} |g(X_{\beta_i}(p_j))|^2,$$

ou seja, $\mathrm{err}(\sigma_i, \beta_i)$ mede quão bem θ_i é levado a $g^{-1}(0)$ por X_{β_i}.

O passo indutivo se dá da seguinte maneira: escolhemos a célula $\sigma_i \in K_i$ que possui o maior erro. Em seguida fazemos

$$K_{i+1} = \textsc{MaubachSubdivide}(K_i, \sigma_i).$$

Após tal subdivisão, k novos vértices $v_1, \ldots, v_k$ são inseridos, nessa ordem. Para cada vértice v_k, fazemos $f_{i+1}(v_k) = g(v_k)$, f_{i+1} sendo igual a f_i nos demais vértices, e rodamos o programa de otimização

$$\min_{\beta_{i+1}} \sum_{\sigma_{i+1} \in \mathrm{st}(v_k, K_{i+1})} \mathrm{err}(\sigma_{i+1}, \beta_{i+1}),$$

de modo a calcular $X_{\beta_{i+1}}$. Note que dessa maneira estamos melhorando $X_{\beta_{i+1}}$ apenas nas regiões da malha que foram modificadas pela subdivisão. Iteramos esse passo até o erro total estar abaixo de um limiar pré-estabelecido.

Notamos que neste caso a representação é do tipo *paramétrica-implícita* pois, analisando a construção, temos que

$$\mathcal{O} = (\overline{f} \circ X^{-1})^{-1}(0) = \bigcup_{\theta \in (K,f)} X(\theta).$$

Se o gradiente de g também é conhecido, podemos modificar a função de erro, obtendo esta definição alternativa,

$$\mathrm{err}(\sigma_i, \beta_i) = \sum_{p_j \in \theta_i} \frac{(g \circ X_{\beta_i})^2(p_j)}{\|\nabla(g \circ X_{\beta_i})(p_j)\|^2},$$

a fim de obter uma aproximação mais uniforme.

Nas figuras 6.2, 6.3 e 6.4 mostramos o algoritmo de poligonização aplicado a curva de Taubin,

$$0.004 + 0.110x - 0.177y - 0.174x^2 + 0.224xy - 0.303y^2 -$$

$$0.168x^3 + 0.327x^2y - 0.087xy^2 - 0.013y^3 +$$

$$0.235x^4 - 0.667x^3y + 0.745x^2y^2 - 0.029xy^3 + 0.072y^4 = 0,$$

em três etapas distintas. Note como a geometria está bem capturada mesmo com poucas subdivisões.

6.3 Reconstrução de hipersuperfícies dadas por pontos

Podemos formular o problema da reconstrução de hipersuperfícies dadas por pontos da seguinte forma: dado um conjunto de pontos $\{p_j\} \subset [0,1]^n$, suficientemente amostrado de uma hipersuperfície $\mathcal{H}$, encontrar um isocomplexo curvilinear $\mathcal{O} = (K, f, X)$, tal que

$$\{p_j\} \subset (\overline{f} \circ X)^{-1}((-a, a)),$$

para a pequeno. Além disso, espera-se que $\mathcal{O}$ tenha a mesma topologia de $\mathcal{H}$.

Esse problema é bem mais difícil que o anterior, sendo uma espécie de problema inverso. No problema de poligonização, o valor de $f(v)$ vinha de graça, bastava por $f(v) = g(v)$. Já na reconstrução, o principal problema é justamente estimar o valor de $f(v)$, ou, mais precisamente, o sinal $s(v)$, que indica se v está no interior ou no exterior de $\mathcal{H}$. Por causa dessa dificuldade, vamos dividir

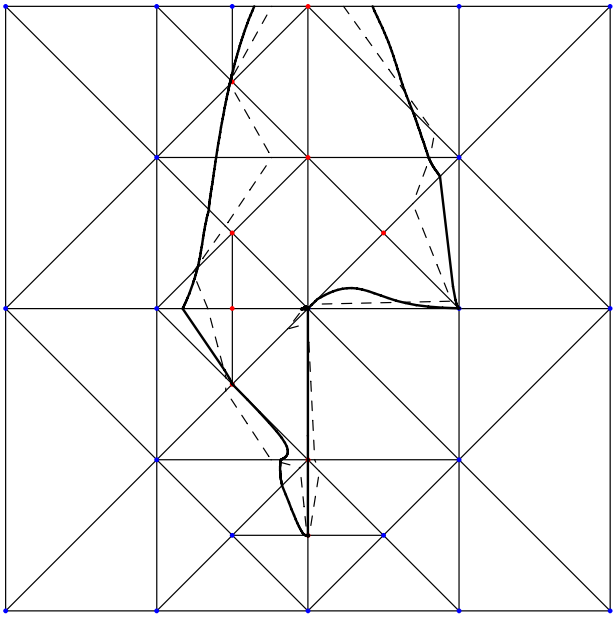


Figura 6.2: Poligonização da curva de Taubin (1).

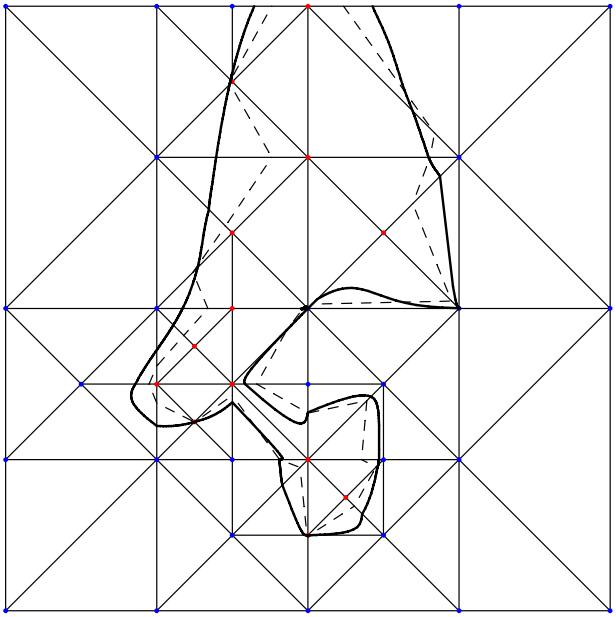


Figura 6.3: Poligonização da curva de Taubin (2).

a exposição do algoritmo de reconstrução em duas etapas. Primeiramente, na etapa de *estimação da topologia*, vamos mostrar como estimar um isocomplexo $O = (K, f)$ que aproxima C linearmente. Depois, na etapa de *estimação da*

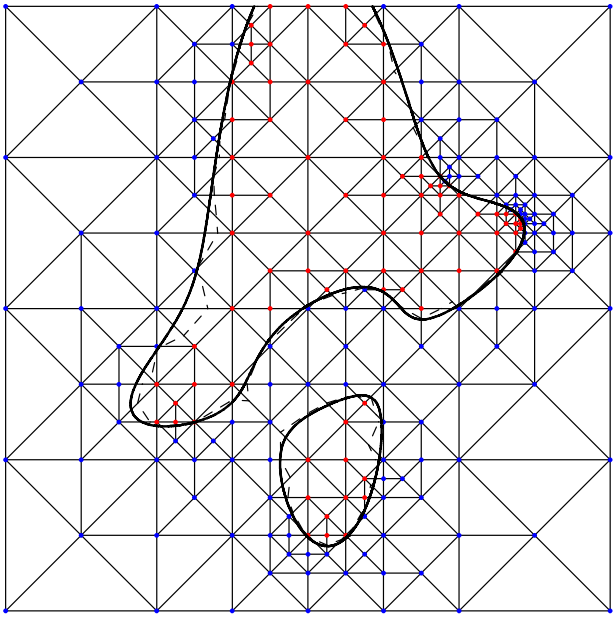


Figura 6.4: Poligonização da curva de Taubin (3).

geometria, vamos mostrar como melhorar a geometria da aproximação com a ajuda de um difeomorfismo K-invariante X.

A idéia da estimação da topologia é semelhante à descrita na seção anterior, ou seja, construir uma seqüência de complexos $O_i = (K_i, f_i)$ que estima progressivamente a topologia de $\mathcal{H}$. Colocamos inicialmente K_0 igual a triangulação canônica de $[0,1]^n$ e $f_0(v) = +1$, para todo $v \in K_0$, indicando que os vértices do hipercubo estão fora de $\mathcal{H}$. Para descrever o passo indutivo que calcula O_{i+1} a partir de O_i, precisamos definir a função $\mathrm{err}(\sigma_i)$ que associa a cada célula $\sigma_i \in K_i$ um erro que mede quão coplanares os pontos p_j contidos em σ_i estão.

Utilizamos a técnica de *análise de componentes principais* para calcular o erro de coplanaridade. Nós calculamos o baricentro p_{σ_i} e a matriz de covariância C_{σ_i} dos pontos p_j contidos em σ_i. A função de erro é dada por

$$\mathrm{err}(\sigma_i) = \frac{\lambda^1_{\sigma_i}}{\lambda^1_{\sigma_i} + \lambda^2_{\sigma_i} + \ldots + \lambda^n_{\sigma_i}},$$

onde $\lambda^1_{\sigma_i} < \lambda^2_{\sigma_i} < \ldots < \lambda^n_{\sigma_i}$ são os autovalores de C_{σ_i}. Também associamos a σ_i o hiperplano Π_{σ_i} que passa por p_{σ_i} e é perpendicular ao autovetor associado a $\lambda^1_{\sigma_i}$. A idéia é que Π_{σ_i} representa um hiperplano secante que aproxima localmente $\mathcal{O}$ dentro de σ_i, com erro de aproximação dado por $\mathrm{err}(\sigma_i)$.

O passo indutivo se dá da seguinte maneira: escolhemos a célula $\sigma_i \in K_i$ que possui o maior erro. Em seguida fazemos

$$K_{i+1} = \textsc{MaubachSubdivide}(K_i, \sigma_i),$$

e estimamos o valor de $f_{i+1}(v) \in \{-1, +1\}$, para todo $v \in K_{i+1}$.

Para tanto, usamos as estimativas locais em cada σ_i dadas pelo hiperplano Π_{σ_i}, definindo

$$s_{\sigma_i}(v) = \begin{cases} -1 & \text{, se } v \in \Pi^-_{\sigma_i} \\ +1 & \text{, se } v \in \Pi^+_{\sigma_i} \end{cases},$$

ou seja, $s_{\sigma_i}(v)$ é o sinal que $f_i(v)$ deve ter do ponto de vista de σ_i.

Entretanto, nada garante que esses sinais são globalmente consistentes. É como se cada célula tivesse sua própria opinião sobre o sinal de seus vértices, ou melhor dizendo, sua própria opinião sobre o sinal de seus vértices a menos de um fator -1. Naturalmente, espera-se que a opinião de uma célula que possui um erro pequeno seja mais confiável do que a de uma célula com erro grande. Assim chegamos à definição de um peso H_ϵ associado a cada aresta ϵ, dado por

$$H_\epsilon = - \sum_{\sigma_i \in \mathrm{st}(\epsilon, K_i)} s_{\sigma_i}(\epsilon[0]) s_{\sigma_i}(\epsilon[1]) \frac{\mathrm{err}_{\max} - \mathrm{err}(\sigma_i)}{\mathrm{err}_{\max} - \mathrm{err}_{\min}},$$

que mede a chance de haver uma mudança de sinal entre os vértices de ϵ. Quanto maior o valor de H_ϵ, maior a chance. Acima, $\mathrm{err}_{\max}$ significa o maior erro $\mathrm{err}(\sigma_i)$ para todo $\sigma_i \in K_i$ e $\mathrm{err}_{\min}$ é definido analogamente. Finalmente, os valores $f_{i+1}(v)$ são estimados minimizando-se o *erro topológico*

$$\mathrm{err}_{\mathrm{top}} = \sum_{\epsilon \in K_{i+1}} H_\epsilon f_{i+1}(\epsilon[0]) f_{i+1}(\epsilon[1]).$$

Empregamos o algoritmo de *simmulated anealing* para minimizar $\mathrm{err}_{\mathrm{top}}$. O passo indutivo é iterado até um conjunto de critérios adaptativos ser satisfeito, tais como o $\mathrm{err}(\sigma_i)$ estar abaixo de um certo limiar, ou $\mathrm{level}(\sigma_i)$ ser maior ou igual a um valor dado. A figura 6.5 mostra que o algoritmo de estimação da topologia consegue capturar imersões não triviais.

Calculado o complexo $O = (K, f)$ que aproxima linearmente $\mathcal{H}$, partimos para a etapa de estimação da geometria. Inicialmente, fazemos

$$f(v) := f(v)\,\mathrm{dist}^2(v, \{p_j\}),$$

ou seja, $f(v)$ agora representa a distância com sinal do vértice v ao conjunto de pontos de entrada. Isso já melhora bastante a aproximação, mas O segue sendo um objeto linear por partes.

Para melhorar ainda mais a aproximação, calculamos um difeomorfismo K-invariante X_β, minimizando o somatório do erro geométrico

$$\mathrm{err}'(\sigma, \beta) = \sum_{p_j \in \sigma} |\overline{f}(X_\beta(p_j))|^2,$$

sobre todos as células $\sigma \in K$. Na prática, esse procedimento de minimização global só funciona em dimensão $n = 2$, pois o número de variáveis do problema cresce bastante já em $n = 3$. A alternativa que empregamos é minimizar $\mathrm{err}'(\sigma, \beta)$ individualmente para cada célula σ, usando poucas iterações, e repetir este processo algumas vezes para obter um melhor resultado.

Na figura 6.6, vemos o resultado da estimação da geometria em um caso simples.

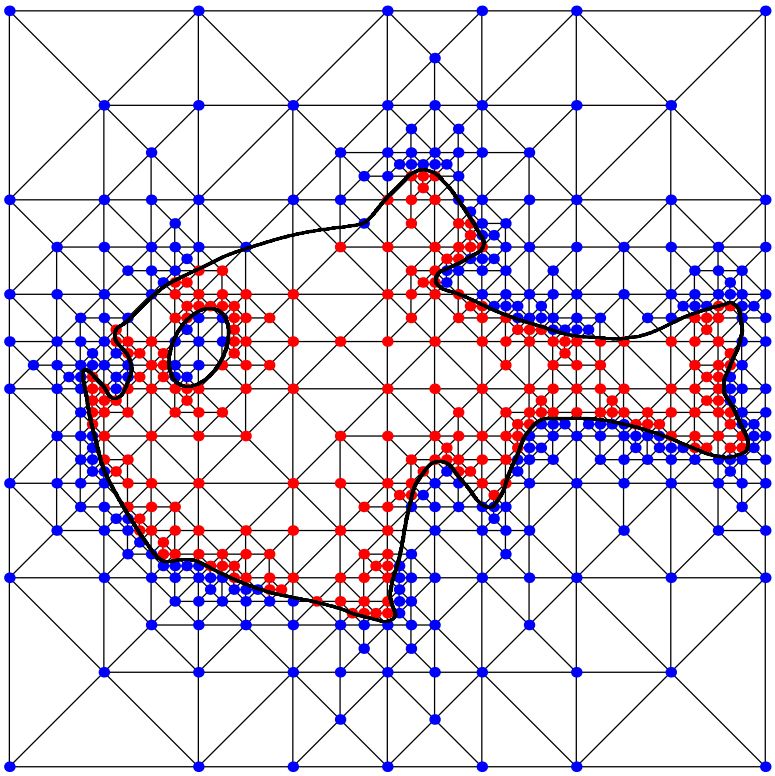
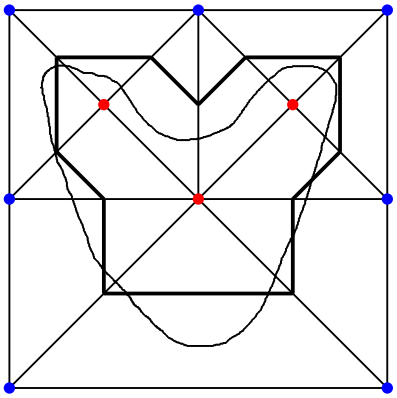


Figura 6.5: Estimação da topologia. O peixe está desenhado com 5411 pontos.

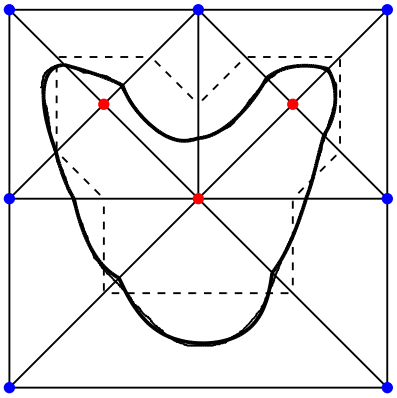
6.4 Vizualização e Suavização

Como dissemos, as aplicações descritas nas seções anteriores geram como saída um isocomplexo curvilinear $\mathcal{O} = (K, f, X)$. Nesta seção, vamos descrever o método que utilizamos para visualizar tais objetos. Em resumo, o método consiste em triangular o complexo $O = (K, f)$, subdividi-lo usando um esquema de subdivisão, e aplicar o difeomorfismo X^{-1} ou X aos vértices da malha resultante, conforme $\mathcal{O}$ seja do tipo implícito-paramétrico ou paramétrico-implícito, respectivamente. Como resultado obtemos uma malha simplicial que pode ser visualizada com aplicações específicas.

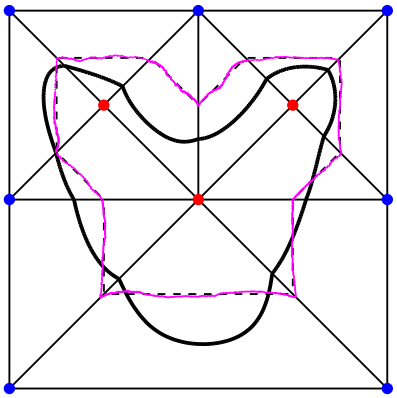
O algoritmo usado para efetuar a primeira etapa do método, ou seja, a triangulação de O, é o descrito na sub-seção 3.2.5. Como resultado obtemos a malha $\mathrm{Tr}(O)$, que será refinada adaptativamente. Entretando, não podemos aplicar os esquemas de subdivisão que estudamos no capítulo 4 diretamente a $\mathrm{Tr}(O)$, pois nada garante que a malha satisfaz os invariantes de Mitchell ou de Maubach. Mas, como vimos, sempre é possível resolver esse problema através da aplicação de um número finito de subdivisões estelares. Podemos, por exemplo, tomar a primeira subdivisão baricêntrica de $\mathrm{Tr}(O)$ como malha inicial M_0 e usar o esquema de Maubach. No caso $n = 3$, entretanto, é mais econômico usar



(a) Entrada da etapa de estimação da geometria, os pontos estão em preto.



(b) Após a execução do algoritmo, os pontos estão próximos a curva $\mathcal{O}$. O complexo O está tracejado.



(c) Em magenta, vemos como os pontos são levados a O pelo difeomorfismo X.

Figura 6.6: Estimação da geometria.

o esquema de Mitchell, pois menos subdivisões são aplicadas para se gerar a malha M_0, com resultados semelhantes.

O refinamento adaptativo ocorre da seguinte maneira: inserimos todas as células $\sigma = \langle v_0, \ldots, v_n \rangle \in M_0$ em uma fila de prioridade Q, onde a prioridade da célula σ é dada pelo volume da célula transformada $\langle X^{-1}(v_0), \ldots, X^{-1}(v_n) \rangle$, no caso implícito-paramétrico, ou da célula $\langle X(v_0), \ldots, X(v_n) \rangle$, no caso paramétrico-implícito. O *loop* de refinamento prossegue extraindo o topo σ_i da fila Q, e fazendo

$$M_{i+1} = \textsc{MitchellSubdivide}(M_i, \sigma_i),$$

ao mesmo tempo que as novas células são inseridas em Q. O processo acaba quando um conjunto de critérios adaptativos é alcançado.

Por fim, aplicamos o difeomorfismo X^{-1} ou X, conforme o caso, a todos os vértices da malha final, gerando assim uma malha simplicial M que aproxima o complexo $\mathcal{O}$. É desnecessário dizer que para $n = 2$ o processo é simplificado de maneira óbvia.

A figura 6.7 ilustra o processo de geração das malhas. Em (a) temos a triangulação inicial M_0, em (b) uma malha intermediária M sem a aplicação do difeomorfismo, em (c) esta mesma malha já deformada e em (d) a malha um pouco mais refinada.

Até aqui não mencionamos nada sobre a suavidade do complexo $\mathcal{O}$ resultante. Certamente $\mathcal{O}$ é suave dentro de cada célula $\sigma \in K$, visto que $X|_\sigma$ é polinomial, mas não podemos garantir nada sobre a suavidade de $\mathcal{O}$ na interseção de duas células adjacentes. Como as aplicações que geraram $\mathcal{O}$ tentavam aproximar um objeto suave, seria razoável supor que tal suavidade fosse herdada. Mas, na prática, isso nem sempre acontece, porque é difícil obter uma boa combinação de adaptação da malha K e ajuste dos parâmetros intrínsecos β. Assim, enquanto não se descubram critérios teóricos suficientes para a suavidade de $\mathcal{O}$, temos que recorrer a alternativas mais pragmáticas.

A forma que encontramos de minimizar a descontinuidade aparente da normal entre células foi justamente forçar a suavidade via otimização. Seja p um ponto de $\mathcal{O}$ que pertence ao bordo de uma célula σ. É possível associar duas normais unitárias a p: a normal $n_\sigma(p)$, que é a normal de $\mathcal{O}$ restrita a σ no ponto p, e a normal $n(p)$ que é a média das normais $n_{\sigma'}(p)$ para todos as células $\sigma' \in K$ tais que $p \in \sigma'$. Para suavizar $\mathcal{O}$, definimos o *erro da normal*

$$\mathrm{err}(\sigma, \beta) = \sum_{p_i} \| n_{\sigma,\beta}(p_i) - n(p_i) \|^2,$$

onde os p_i são pontos amostrais convenientemente escolhidos em $\mathcal{O} \cap \sigma$, e resolvemos o problema de otimização

$$\min_\beta \sum_{\sigma \in K} \mathrm{err}(\sigma, \beta).$$

Esse método da bons resultados em $n = 2$ pois dois pontos amostrais são suficientes.

Na figura 6.7, o complexo na subfigura (d) foi obtido do complexo em (a) pela aplicação do algoritmo de suavização descrito acima.

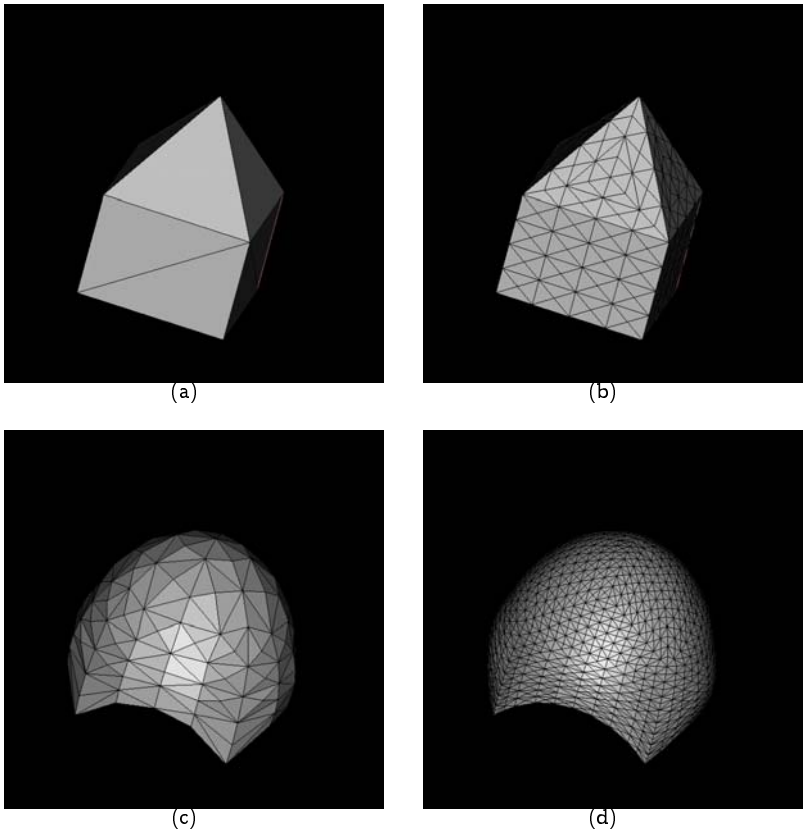


Figura 6.7: Exemplos que ilustram a visualização e a suavização de isocomplexos curvilineares.

Pode-se perguntar por que implementamos a suavização dessa maneira, ao invés de se recorrer aos bem conhecidos critérios de continuidade C^1 para funções ou malhas de Bézier, como parece ser o caso. Para responder a essa pergunta, vamos considerar um isocomplexo curvilinear $\mathcal{O} = (K, f, X)$, onde $K = \sigma - \sigma'$, com $\sigma = \langle p_0, p_1, \ldots, p_n \rangle$ e $\sigma' = \langle p'_0, p_1, \ldots, p_n \rangle$, e X é um difeomorfismo K-invariante polinomial tal que $X|_\sigma = \sum_{|I|=m} c_I B_I$ e $X|_{\sigma'} = \sum_{|I|=m} c'_I B_I$, em coordenadas baricêntricas. Queremos saber sob que condições $\mathcal{O}$ é C^1 na faceta comum a σ e σ', isto é, nos pontos $q \in \delta \cap \mathcal{O}$, onde $\delta = \partial_0 \sigma = \partial_0 \sigma'$.

A primeira vista, deveríamos obrigar a continuidade C^1 do difeomorfismo X, fazendo

$$c_{J+e_0} = \sum_i w_i c'_{J+e_i},$$

onde $|J| = m - 1$, $J^0 = 0$ e $(w_0, \ldots, w_n)$ são as coordenadas baricêntricas de p_0 com relação a σ'. Mas esse não é o caso, pois em geral a função $\overline{f}$ não é C^1 em $X^{-1}(q)$, portanto se X fosse C^1, a composição $(\overline{f} \circ X)$ não seria C^1.

Uma segunda tentativa seria forçar a continuidade C^1 de $(\overline{f} \circ X)$ em δ,

fazendo

$$\overline{f}(c_{J+e_0}) = \sum_i w_i \overline{f}(c'_{J+e_i}).$$

Isso funciona, mas é mais do que precisamos, pois essa condição assegura a continuidade C^1 de $(\overline{f} \circ X)$ em todos os pontos de δ, enquanto é suficiente a continuidade C^1 nos pontos $q \in \delta \cap \mathcal{O}$. Além do mais, é difícil assegurar simultaneamente a continuidade C^1 e o fato de X ser um difeomorfismo.

6.5 Conclusão

A idéia para o conceito de isocomplexo curvilinear veio no período em que estávamos trabalhando na extensão tridimensional do trabalho de Taubin e Ronfard [35], que resultou no artigo [23]. Nesse artigo, expomos essencialmente o algoritmo da seção 6.3, só que restrito a isocomplexos lineares. Analisando os resultados, percebemos que o número de subdivisões poderia ser diminuído se uma deformação fosse aplicada no interior de cada célula, e daí surgiu a definição de isocomplexo curvilinear.

Em todos os problemas de otimização, utilizamos o algoritmo L-BFGS-B [38]. Ele apresenta bons resultados e possui uma interface simples de usar. O gradiente das funções objetivo foi calculado com diferenças finitas, dada a complexidade funções. Futuramente, talvez seja interessante considerar algoritmos de minimização que levem em conta o fato dessas funções serem *parcialmente separáveis* [26].

O algoritmo de poligonização esboçado na seção 6.2 não é robusto, ou seja, componentes conexas de $g^{-1}(0)$ podem não ser descobertas, já que a função de erro que usamos atribui um erro nulo a uma célula que possui todos os sinais iguais, independentemente do que acontece dentro da célula. Uma maneira de resolver esse problema é incorporar métodos intervalares ao algoritmo, como em [18].

O algoritmo de reconstrução da seção 6.3 também pode ser melhorado de muitos modos. O fato é que o algoritmo comporta um grande número de variações e o desafio é encontrar a melhor. Por exemplo, é possível permitir que os vértices se movam, de modo a se afastarem dos pontos. Ou então efetuar a otimização dos parâmetros intrínsecos durante o processo adaptativo, num estilo *multigrid*. Ou ainda executar um algoritmo de simplificação como em [15] após a estimação da topologia e só então efetuar a estimação da geometria. Pretendemos explorar essas variantes futuramente.

Quanto à visualização, não é difícil implementar rasterização no caso $n = 2$ e *ray-tracing* no caso $n = 3$. Essas seriam ótimas maneiras de se mostrar efetivamente a dualidade implícita-paramétrica da representação. Para o *ray-tracing*, entretanto, seria necessário um estudo mais detalhado da questão da suavidade. Aliás, esse é o ponto que mais merece pesquisa num futuro próximo, ou seja, como garantir que o complexo $\mathcal{O}$ seja no mínimo globalmente C^1.

Capítulo 7

Conclusão

Para finalizar, vamos listar as principais contribuições desta tese, ao menos na nossa ótica, e discutir brevemente alguns temas de trabalhos futuros.

No capítulo 3, utilizamos o conceito de complexo semi-simplicial para estabelecer um vínculo entre programação genérica e modelagem topológica. Também isolamos o conceito de isocomplexo, para o qual elaboramos um algoritmo de triangulação genérico.

A estrutura de dados que exibimos permite o acesso às relações de incidência em tempo constante, à custa de um maior espaço de armazenamento. Uma possibilidade interessante de trabalho futuro é parametrizar a estrutura com *políticas* [2], através das quais o usuário possa trocar tempo de acesso por espaço de armazenamento conforme as necessidades da aplicação, mas preservando a interface básica.

No capítulo 4, introduzimos a noção de esquema de subdivisão estelar, aplicando-a a dois esquemas de subdivisão pré-existentes. Graças a um tratamento mais combinatório, podemos dar uma nova demonstração da correção do algoritmo de Maubach. Mostramos também um algoritmo que transforma uma malha qualquer em uma malha que satisfaz os invariantes de Maubach.

Uma linha de pesquisa posterior é estudar mais detalhadamente as propriedades geométricas e combinatórias do esquema de Maubach. Isso pode levar a maneiras mais eficientes de armazenar e operar com multitriangulações geradas a partir do esquema de Maubach. Uma segunda linha seria o estudo de outros esquemas de subdivisão estelar.

No capítulo 5, definimos um novo tipo de transformação do simplexo, as funções simplicialmente invariantes, e estudamos critérios gerais de injetividade. A seguir aplicamos esses critérios no caso polinomial obtendo uma série de resultados parciais, mas já suficientes para a implementação de aplicações.

Existe uma série de questões que merece ser investigada no tocante aos difeomorfismos simpliciais. O objetivo final é encontrar funções simplicialmente invariantes que sejam, por um lado, dependentes de poucos parâmetros intrínsecos, mas ainda assim capazes de modelar *patches* suficientemente flexíveis, e por outro lado garantidamente difeomorfismos simpliciais. Talvez funções simplicialmente invariantes racionais possam resolver essa questão.

No capítulo 6, aplicamos os resultados dos capítulos anteriores para formular uma nova representação de objetos gráficos que combina características das representações implícita e paramétrica. Mostramos o poder dessa representação ao descrever algoritmos genéricos com respeito a dimensão para três problemas clássicos: modelagem *free-form*, poligonização de objetos implícitos e reconstrução de hipersuperfícies dadas por pontos.

Quanto às aplicações, resta melhorar os resultados, principalmente no que diz respeito à suavidade do isocomplexo curvilinear.

> That I am no skilled mathematician I have had little need to confess. I am 'advanced in these enquiries no farther than the threshold'; but something of the use and beauty of mathematics I think I am able to understand.
>
> D'Arcy Thompson [36]

Bibliografia

[1] J. Alexander. The combinatorial theory of complexes. *Ann. Math.*, 31:294–322, 1930.

[2] A. Alexandrescu. *Modern C++ Design.* Addison-Wesley Professional, 2001.

[3] C. L. Bajaj, F. Bernardini, and G. Xu. Automatic reconstruction of surfaces and scalar fields from 3d scans. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 109–118. ACM Press, 1995.

[4] C. L. Bajaj, J. Chen, and G. Xu. Modeling with cubic *A*-patches. *ACM Transactions on Graphics*, 14(2):103–133, 1995.

[5] E. Brisson. Representing geometric structures in d dimensions: Topology and order. In *Symposium on Computational Geometry*, pages 218–227. ACM, June 1989.

[6] Paulo Roma Cavalcanti, Paulo Cezar Pinto Carvalho, and Luiz Fernando Martha. Non-manifold modelling: an approach based on spatial subdivision. *Computer-Aided Design*, 29(3):209–220, 1997.

[7] Vinícius Mello, Luiz Velho, Paulo Roma Cavalcanti, and Cláudio Silva. A generic programming approach to multiresolution spatial decompositions. *Visualization and Mathematics III*, pages 340–362, 2002.

[8] Sabine Coquillart. Extended free-form deformation: a sculpturing tool for 3d geometric modeling. In *SIGGRAPH*, pages 187–196, 1990.

[9] G. Farin. Triangular Berstein-Bézier patches. *Computer Aided Geometric Design*, 3(2):83–127, 1986.

[10] Leila De Floriani, Paola Magillo, and Enrico Puppo. Efficient implementation of multi-triangulations. In *IEEE Visualization*, pages 43–50, 1998.

[11] J.E. Gain and N.A. Dodgson. Preventing self-intersection under free-form deformation. *IEEE Transactions on Visualization and Computer Graphics*, 7(4):289–298, 2001.

[12] Jonas Gomes and Geovan Tavares. *Métodos Simpliciais em Computação Gráfica.* 17º Colóquio Brasileiro de Matemática, 1989.

[13] Gelfand I.M., Kapranov M.M., and Zelevinsky A.V. *Discriminants, Resultants and Multidimensional Determinants*. Birkhäuser, Boston, 1994.

[14] D.E. Knuth. *The Art of Computer Programming: Generating all combinations*, volume 4, fascicle 3. Addison Wesley Professional, 2006.

[15] Thomas Lewiner, Luiz Velho, Hélio Lopes, and Vinícius Mello. Hierarchical isocontours extraction and compression. In *SIBGRAPI*, pages 234–241, 2004.

[16] W. B. R. Lickorish. Simplicial moves on complexes and manifolds. In *Proceedings of the Kirbyfest*, volume 2, pages 299–320, 1999.

[17] P. Lienhardt. N-dimensional generalized combinatorial maps and cellular quasi-manifolds. *Int. J. Comput. Geometry Appl.*, 4(3):275–324, 1994.

[18] Hélio Lopes, João Batista S. de Oliveira, and Luiz Henrique de Figueiredo. Robust adaptive polygonal approximation of implicit curves. *Computers & Graphics*, 26(6):841–852, 2002.

[19] Ricardo Marroquim, Paulo Roma Cavalcanti, Claudio Esperança, and Luiz Velho. Multi-resolution triangulations with adaptation to the domain based on physical compression. In *SIBGRAPI*, pages 226–233, 2004.

[20] J.M. Maubach. Local bisection refinement for N-simplicial grids generated by reflection. *SIAM J.Sci.Stat.Comp.*, 16:210–227, 1995.

[21] J. Peter May. *Simplicial Objects in Algebraic Topology*. The University of Chicago Press, Chicago and London, 1967.

[22] G. Meisters and C. Olech. Locally one-to-one mappings and a classical theorem on schlicht functions. *Duke Mathematical Journal*, 30:63–80, 1963.

[23] Vinícius Mello, Luiz Velho, and Gabriel Taubin. Estimating the in/out function of a surface represented by points. In *Symposium on Solid Modeling and Applications*, pages 108–114, 2003.

[24] W.F. Mitchell. Optimal multilevel iterative methods for adaptive grids. *SIAM J.Sci.Stat.Comp.*, 13:146–167, 1992.

[25] José Maria Ribeiro Neves. *Visualização Volumétrica de Campos Escalares Definidos em Triangulações Retangulares*. Tese de doutorado, Universidade Federal do Rio de Janeiro, 1999.

[26] J. Nocedal and S. Wright. *Numerical Optmization*. Springer-Verlag New York, 1999.

[27] M. Noro and T. Takeshima. Risa/Asir – a computer algebra system. In *Papers from the international symposium on Symbolic and algebraic computation*, pages 387–396. ACM Press, 1992.

[28] A. Paoluzzi, F. Bernardini, C. Cattani, and V. Ferrucci. Dimension-independent modeling with simplicial complexes. *ACM Trans. Graph.*, 12(1):56–102, 1993.

[29] M. Schweighofer. Algorithmische Beweise für Nichtnegativ- und Positivstellensätze. Diplomarbeit, Universität Passau, 1999.

[30] M. Schweighofer. An algorithmic approach to Schmüdgen's Positivstellensatz. *Journal of Pure and Applied Algebra*, 166:307–319, 2002.

[31] T. Sederberg and S. Parry. Free-form deformation of solid geometric models. In *SIGGRAPH*, pages 151–160, 1986.

[32] E.G. Sewell. *Automatic generation of triangulations for piecewise polynomial approximation.* PhD thesis, Purdue University, West Lafayette, IN, 1972.

[33] J. G. Siek, L.Q. Lee, and A. Lumsdaine. *Boost Graph Library, The: User Guide and Reference Manual.* Addison-Wesley, 2002.

[34] A. A. Stepanov and M. Lee. The Standard Template Library. Technical Report X3J16/94-0095, WG21/N0482, ISO Programming Language C++ Project, 1994.

[35] G. Taubin and R. Ronfard. Implicit simplicial models for adaptive curve reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(3):321–325, 1996.

[36] D'Arcy Thompson. *On Growth and Form.* Cambridge University Press, 1961.

[37] Luiz Velho and Jonas Gomes. Variable resolution 4-k meshes: Concepts and applications. *Comput. Graph. Forum*, 19(4):195–212, 2000.

[38] Ciyou Zhu, Richard H. Byrd, Peihuang Lu, and Jorge Nocedal. Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. *ACM Trans. Math. Softw.*, 23(4):550–560, 1997.

[39] G. Ziegler and M. Aigner. *Proofs from THE BOOK.* Springer-Verlag Heildelberg, 1998.