Visualização Automática de Mapas Tridimensionais

Luis Arturo Pérez Lozada

Tese de Doutorado

Visualização Automática de Mapas Tridimensionais

Este exemplar corresponde à redação final da Tese devidamente corrigida e defendida por Luis Arturo Pérez Lozada e aprovada pela Banca Examinadora.

Campinas, 23 de Março de 2001.

Dr. Jorge Stolfi (Co-orientador)

Tese apresentada ao Instituto de Computação, UNICAMP, como requisito parcial para a obtenção do título de Doutor em Ciência da Computação.

Instituto de Computação
Universidade Estadual de Campinas

Visualização Automática de Mapas Tridimensionais

Luis Arturo Pérez Lozada

23 de Março de 2001

Banca Examinadora:

- Dr. Candido F. Xavier de Mendonça Neto (Orientador)
- Dr. Sóstenes Luiz Soares Lins
  DMA - Universidade Federal de Pernambuco
- Dr. Luiz Henrique de Figueiredo
  Instituto de Matemática Pura e Aplicada
- Dr. Pedro Jussieu de Rezende
  IC - Universidade Estadual de Campinas
- Dr. Cid Carvalho de Souza
  IC - Universidade Estadual de Campinas
- Dra. Wu Shin-Ting (Suplente)
  FEE - Universidade Estadual de Campinas
- Dr. Flávio Keidi Miyazawa (Suplente)
  IC - Universidade Estadual de Campinas

Dedicatória

À minha querida família: *Grimaldo, Hortência, Yanett, Gisella e Carmen*

Agradecimentos

Ao prof. Jorge Stolfi pela sua brilhante orientação.
Ao prof. Candido Xavier pelo seu grande incentivo.
Ao contribuinte, que com suas aportações viabiliza a execução de projetos de pesquisa.
A Liliane, minha futura esposa.
A Richard Hollasch por disponibilizar seu programa de visualização *Wire4*.
À FAPESP, CAPES e FAEP pelo suporte financeiro.
À profa. Anamaria Gomide por ter cedido gentilmente sua sala.
Ao Roland, Euclides, Ademilson, Roseli, Jeronimo, Islene, Oliva, Delano, Ulises, Vagner.
A todos os professores, funcionários e colegas do Instituto de Computação da Unicamp.

Resumo

Um mapa tridimensional é uma coleção de elementos topológicos primitivos — células, faces, arestas e vértices — conectados por relações de adjacência e incidência. Abordamos nesta tese o problema de produzir automaticamente uma representação geométrica "bonita" de um mapa 3D, dada apenas sua representação topológica na forma de um esquema de colagem de poliedros.

O modelo geométrico que construímos consiste de um grande número de tetraedros com faces planas, resultantes da subdivisão topológica das células do mapa dado. A geometria do modelo é determinada por funções numéricas, denominadas *energias*, que quantificam vários tipos de defeitos estéticos e utilitários do modelo. A geometria é escolhida minimizando-se uma combinação apropriada das funções de energia, por métodos padrões de otimização não-linear. O modelo geométrico é então exibido com diversas técnicas de visualização.

Abstract

A three-dimensional map is a collection of primitive topological elements — cells, faces, edges and vertices — connected by adjacency and incidence relations. We consider in this dissertation the problem of automatically producing a "nice" geometric representation of a 3D map, given only its topological description, presented as a polyhedral gluing model.

The geometric model that we build consists of a large number of tetrahedra with planar faces, resulting from topological subdivision of the cells of the given map. The model's geometry is determined by numerical functions called *energies*, which quantify several kinds of aesthetic and utilitarian defects of the model. The geometry is chosen by minimizing a suitable combination of energy functions, through standard non-linear optimization methods. The geometric model is then rendered using a variety of visualization techniques.

Conteúdo

Lista de Figuras

Capítulo 1

Introdução

A visualização de objetos de dimensões maiores do que três é um problema desafiante, uma vez que a nossa experiência visual limita-se ao espaço euclidiano tridimensional. Dentre tais objetos, os *mapas tridimensionais* (que incluem a fronteira de politopos 4D) são de grande interesse, não somente em matemática [22], mas também em ciências aplicadas como a robótica [27] e a mecânica [38].

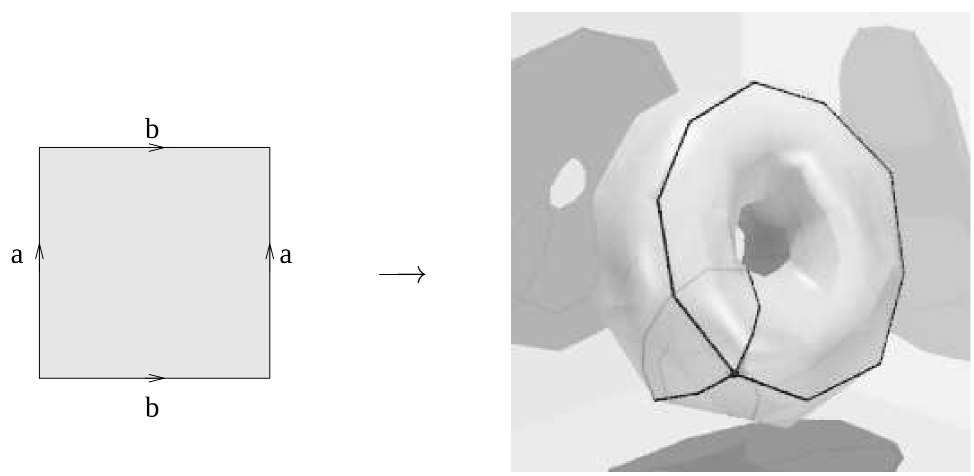
Em tais aplicações, frequentemente encontramos a necessidade de visualizar a estrutura topológica de um mapa tridimensional para o qual contamos apenas com uma descrição combinatória. Para solucionar este problema, precisamos de uma ferramenta que permita gerar, de maneira automática, representações geométricas do mapa 3D que possam ser facilmente entendidas pelo usuário. Os resultados promissores obtidos por R. M. Rosi [30] para o problema análogo em duas dimensões nos motivaram a explorar o uso de técnicas similares para a visualização de mapas 3D.

Para construir um modelo geométrico de um mapa tridimensional $\mathcal{C}$, nós subdividimos cada um de seus elementos em partes com topologia simples (simplexos). O resultado é uma *triangulação* $\mathcal{T}$ do mapa $\mathcal{C}$. Uma representação geométrica da triangulação $\mathcal{T}$ pode então ser especificada mapeando-se cada vértice para um ponto no $\mathbb{R}^m$, onde $m \geq 3$. Determinamos as posições destes vértices minimizando uma *função de energia* $\mathcal{E}$ que penaliza características visuais indesejáveis do modelo. Finalmente, a configuração otimizada $\phi(\mathcal{T})$ é projetada para o espaço $\mathbb{R}^3$ e renderizada através de técnicas de visualização específicas para sólidos com estrutura interna.

1.1 Trabalhos Relacionados

Desenho de grafos. Existe ampla literatura tratando o desenho de grafos no plano [37], uma tarefa que pode ser vista como a versão unidimensional de nosso problema. A dizer a verdade, muitas das ferramentas que nós usamos — tais como o uso de molas entre vértices para evitar auto-interseções e uniformizar o tamanho dos elementos — foram diretamente inspiradas em métodos inicialmente desenvolvidos para o desenho de grafos no plano [18].

Mapas bidimensionais. A visualização automática de mapas bidimensionais (subdivisões de superfícies) foi abordada por Rosi em sua tese de mestrado [30]. A figura 1.1 ilustra uma descrição combinatória de uma mapa sobre o toro $\mathbb{T}^2$ e um modelo geométrico do mesmo, obtido automaticamente pelas ferramentas desenvolvidas por Rosi. Nossas técnicas de modelagem e otimização constituem extensões naturais das técnicas descritas naquele trabalho.



(a) (b)

Figura 1.1: Visualização automática de um mapa bidimensional: (a) representação combinatória, (b) modelo geométrico.

Politopos quadri-dimensionais. Mesmo antes da era digital, muitos autores procuraram tornar visível a estrutura de objetos quadri-dimensionais através da construção de modelos físicos de suas projeções no espaço $\mathbb{R}^3$; vários destes modelos podem ser vistos no livro de Coxeter [4]. Mais tarde, ainda nos primordios da computação gráfica (1967), o pioneiro M. A. Noll produziu filmes de politopos 4D rodando no $\mathbb{R}^4$. Sua técnica era desenhar cada quadro com um *plotter* e fotografar o resultado [26].

Nas últimas décadas, muitos outros autores exploraram a visualização de politopos 4D baseados em projeções. Podemos citar, por exemplo, Carey et al. [3], Hanson e Pheng [13], S. Hollasch [16], Steiner e Burton [33] e muitos outros.

Outros trabalhos. No filme *Not Knot* [12], Gunn e Maxwell fornecem um excelente exemplo da visualização de mapas tridimensionais na forma de uma visão "interior" de um ladrilhamento de um espaço hiperbólico 3D por dodecaedros regulares de ângulo reto.

1.2 Estrutura desta tese

Descrevemos nesta tese os conceitos e técnicas que desenvolvemos e avaliamos para o problema acima, aproximadamente na ordem em que eles são empregados na visualização de um mapa dado. Nos capítulos 2 e 3 recordamos os conceitos de *variedade topológica* e de *mapas n-dimensionais* (partições de tais variedades em células). Nos capítulos 4 e 5 tratamos da decomposição dos elementos de um mapa tridimensional em simplexos topológicos e a representação geométrica dos mesmos como coleções de simplexos num espaço $\mathbb{R}^m$ ($m \geq 3$). No capítulo 6 definimos algumas *funções energia* que quantificam os critérios estéticos e utilitários que definem um "bom" modelo geométrico. No capítulo 7, descrevemos dois algoritmos de otimização não linear que utilizamos em nossos testes para minimizar essas funções energia. No capítulo 8, descrevemos várias técnicas de computação gráfica que empregamos para facilitar a visualização do modelo geométrico otimizado. Nos capítulos 9 e 10, apresentamos alguns exemplos adicionais de realizações geométricas obtidas pelos nossos métodos. Finalmente, no capítulo 11, resumimos as contribuições e descrevemos vários problemas que restam a resolver.

Capítulo 2

Variedades topológicas

Este capítulo tem como finalidade apresentar algumas noções essenciais de variedades topológicas necessárias para esta tese.

2.1 Conceitos básicos

Consideramos conhecidos os conceitos básicos de topologia de conjuntos, tais como, espaço topológico, homeomorfismo, fecho, interior, etc. No apêndice B apresentamos um resumo destes prerequisitos; mais detalhes podem ser encontrados em textos introdutórios de topologia de conjuntos [19, 24, 34, 35].

Se X é um sub-espaço de um espaço topológico Y, denotamos por $\partial_Y X$ a fronteira de X em Y, por $\kappa_Y X$ o fecho de X e por $\iota_Y X$ o seu interior. (Omitiremos o subscrito Y quando for óbvio do contexto). Também denotamos por $\mathbb{B}^d$ a bola d-dimensional unitária aberta de $\mathbb{R}^d$, isto é, $\mathbb{B}^d = \{(x_1, \ldots, x_d) : x_1^2 + \cdots + x_d^2 < 1\}$ e por $\mathbb{S}^{d-1}$ sua fronteira $\{(x_1, \ldots, x_d) : x_1^2 + \cdots + x_d^2 = 1\}$. (Note-se que $\mathbb{B}^d$ é homeomorfo a $\mathbb{R}^d$.) Finalmente denotamos por $\mathbb{H}^d$ o semi-espaço fechado $\{(x_1, \ldots, x_d) \in \mathbb{R}^d : x_1 \geq 0\}$.

2.1.1 Simplexo geométrico

A *envoltória convexa positiva* de um conjunto de pontos $v_0, \ldots, v_d$ é o conjunto de todos os pontos $\alpha_0 v_0 + \cdots + \alpha_d v_d$, onde os pesos α_i são estritamente positivos e $\sum_i \alpha_i = 1$.

Um *k-simplexo geométrico aberto* é a envoltória convexa positiva de $k+1$ pontos independentes de $\mathbb{R}^m$, $m \geq k$, chamados de *cantos* do simplexo. Observe-se que um k-simplexo Δ é homeomorfo à k-bola $\mathbb{B}^k$. Considerando as envoltórias convexas positivas de todos os subconjuntos dos cantos de Δ, obtemos 2^{k+1} simplexos, disjuntos aos pares. Os $\binom{k+1}{j+1}$ subconjuntos com $j+1$ cantos, $j \leq k$ são chamados as *j-faces de* Δ. As $(k-1)$-faces são chamadas as *facetas* de Δ.

2.1.2 Politopos convexos

O conceito de poliedro regular pode ser generalizado para dimensões arbitrárias. Em geral, um *politopo convexo de dimensão* d é o fecho convexo de pelo menos $d+1$ pontos de $\mathbb{R}^m$ que não estão contidos em nenhum sub-espaço euclidiano de dimensão d.

Dizemos que um hiperplano orientado h de $\mathbb{R}^m$ *suporta* um politopo P se ele intercepta P e P está contido no semi-espaço não negativo de h. As interseções de P com seus hiperplanos de suporte são as *faces* de P. Os *vértices* são as faces de dimensão 0 e as *facetas* são as faces de dimensão $d-1$. Verifica-se que toda face de P é um politopo convexo cujas faces são um subconjunto das faces de P.

2.2 Variedade topológica

Um espaço topológico X é dito uma *variedade d-dimensional* se todo ponto de X possui uma vizinhança homeomorfa a $\mathbb{H}^d$ ou $\mathbb{R}^d$. Os pontos de X que possuem vizinhanças homeomorfas a $\mathbb{R}^d$ constituem o *interior absoluto* αX de X; os demais pontos constituem a *borda* da variedade, denotada por βX. As figuras 2.1 a 2.4 ilustram variedades topológicas 1-dimensionais, 2-dimensionais e 3-dimensionais, respectivamente. Exemplos adicionais serão vistos mais adiante neste capítulo.

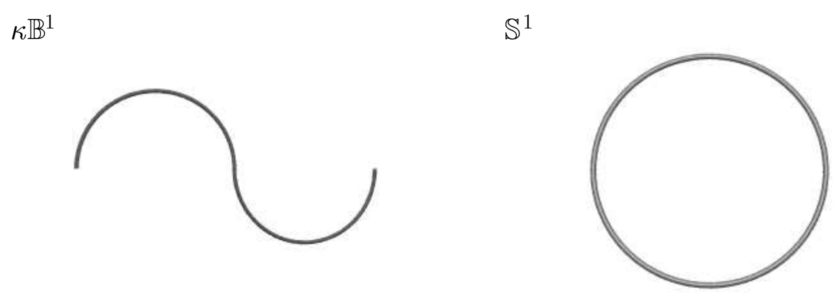


Figura 2.1: Exemplos de 1-variedades topológicas.

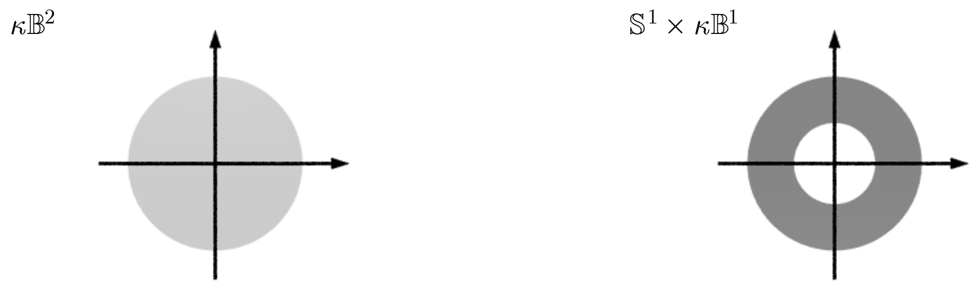


Figura 2.2: Exemplos de 2-variedades topológicas contidas no $\mathbb{R}^2$.

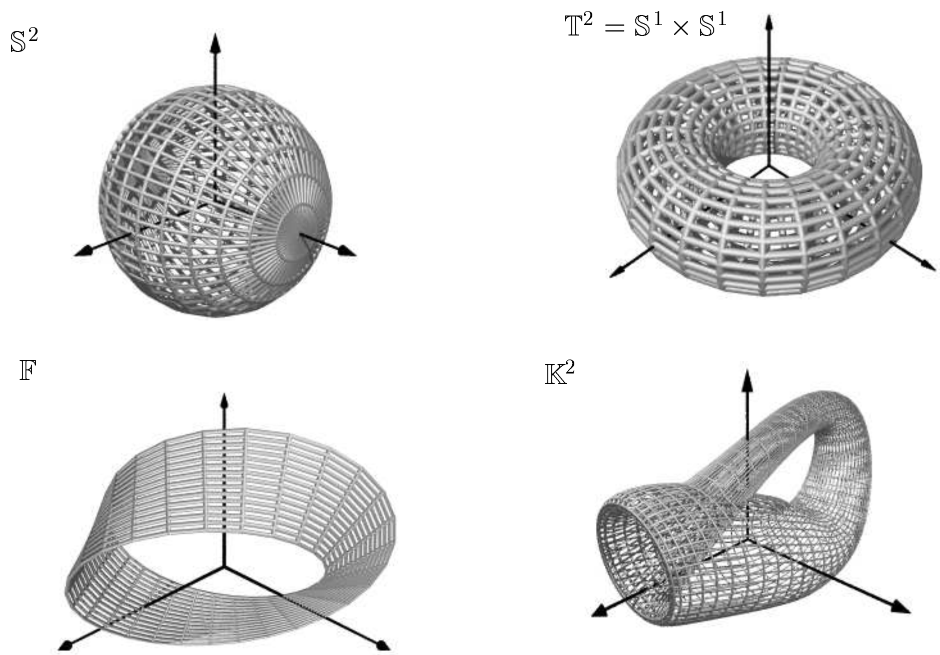


Figura 2.3: Exemplos de 2-variedades topológicas contidas no $\mathbb{R}^3$. ($\mathbb{F}$ é a fita de Möbius. No caso da garrafa de Klein $\mathbb{K}^2$, mostramos uma projeção tridimensional de um modelo contido no $\mathbb{R}^4$.)

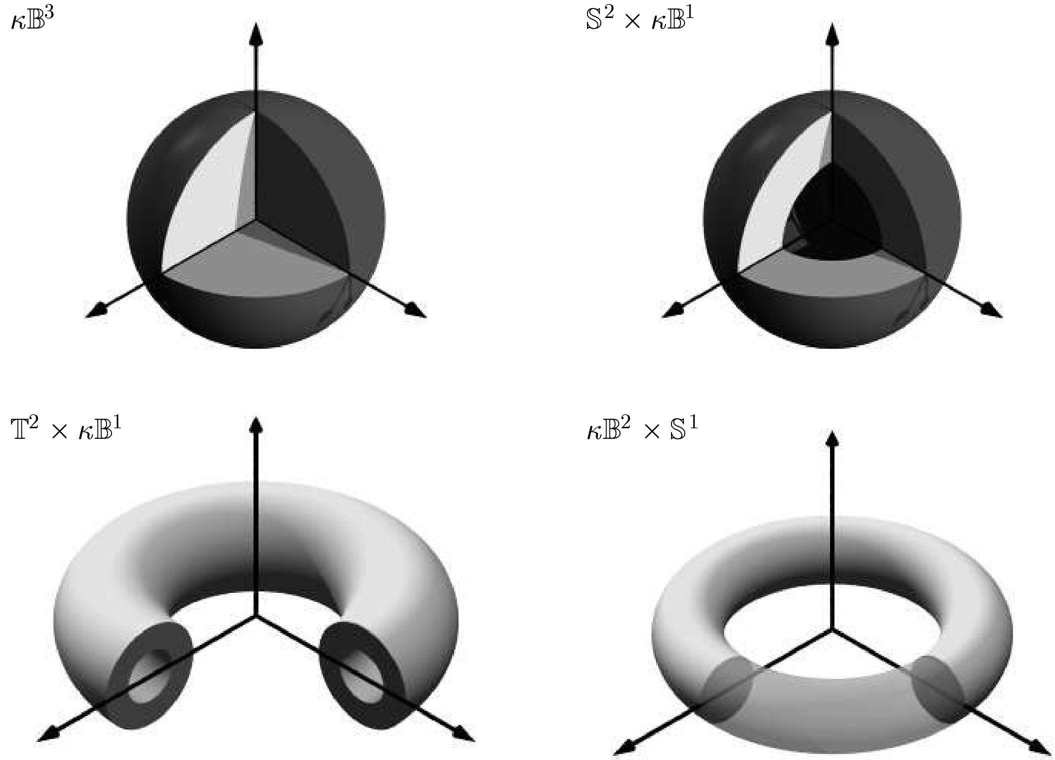


Figura 2.4: Exemplos de 3-variedades topológicas contidas no $\mathbb{R}^3$.

2.3 Variedades orientáveis

Definimos um *caminho* num espaço topológico X como sendo uma função contínua ϕ do intervalo $[-1, +1]$ para X. Dizemos que o caminho é um *ciclo* se $\phi(-1) = \phi(+1)$. Definimos também um *caminho orientado* em X como sendo uma função contínua $\phi\colon \mathbb{B}^d \times [-1, +1] \to X$, tal que para todo t a função $\phi_t(x) = \phi(x, t)$ é um homeomorfismo de $\mathbb{B}^d$ para $\phi(\mathbb{B}^d, t)$. Dizemos que o caminho é um *ciclo orientado* se $\phi(x, -1) = \phi(x, +1)$ para todo $x \in \mathbb{B}^d$.

A *trajetória* de um caminho orientado ϕ é o caminho $\phi_0\colon [-1, +1] \to X$ definido por $\phi_{\mathbf{0}}(t) = \phi(\mathbf{0}, t)$ para todo t, onde $\mathbf{0} = (0, \ldots, 0)$ é o centro de $\mathbb{B}^d$. Uma variedade é *orientável* se todo ciclo é a trajetória de algum ciclo orientado.

2.4 Técnicas para definição de espaços topológicos

Espaços topológicos podem ser definidos de muitas maneiras. Apresentamos a seguir as principais técnicas que se encontram na literatura. Exemplos adicionais destas construções, especializadas para duas e três dimensões, serão apresentadas nas seções 2.5 e 2.6.

2.4.1 Modelo algébrico implícito

Um espaço topológico pode ser definido como o subconjunto de pontos no $\mathbb{R}^n$ que satisfazem uma ou mais equações algébricas $f_i(x_1, x_2, \ldots, x_n) = 0$. Este subconjunto recebe implicitamente a estrutura topológica induzida pela topologia do $\mathbb{R}^n$. Dois exemplos são a esfera $\mathbb{S}^{n-1}$ e a bola $\mathbb{B}^n$ definidas anteriormente. Note que nem sempre o espaço topológico definido desta forma é uma variedade e que sua dimensão d é em geral menor que n.

2.4.2 Modelo paramétrico

Outra variante da construção algébrica é a *construção paramétrica* onde o espaço desejado X é a imagem $\phi(Y)$ de um conjunto qualquer Y de $\mathbb{R}^m$ sob uma função ϕ de Y para um espaço topológico Z. Neste caso, a topologia de X é induzida pela topologia de Z. Observe que X não é necessariamente homeomorfo a Y, pois ϕ pode não ser um homeomorfismo. Por exemplo, a fita de Möbius $\mathbb{F}$ no $\mathbb{R}^3$ é a imagem da faixa infinita $Y = \mathbb{R} \times [-1, +1]$ pela função $\phi(\theta, s) = ((2 + s\cos(\theta/2))\cos\theta, (2 + s\cos(\theta/2))\sin\theta, s\sin(\theta/2))$.

2.4.3 Espaço produto

Outra construção importante é o produto cartesiano de dois ou mais espaços topológicos. A estrutura topológica do produto é implicitamente derivada da topologia dos espaços fatores (vide apêndice B). Exemplos desta construção são o espaço $\mathbb{R}^n = \underbrace{\mathbb{R}^1 \times \ldots \times \mathbb{R}^1}_{n}$ e o *toro n-dimensional* $\mathbb{T}^n = \underbrace{\mathbb{S}^1 \times \ldots \times \mathbb{S}^1}_{n}$.

2.4.4 Espaço quociente

Um espaço topológico também pode ser obtido pelo quociente de outro espaço X por uma relação de equivalência "$\equiv$" sobre os pontos de X. O espaço quociente $X/\!\equiv$ tem por pontos as classes de equivalência da relação "$\equiv$". Um conjunto A de pontos de $X/\!\equiv$ é aberto se e somente se $\bigcup A$ é aberto em X.

Por exemplo, seja $X = \mathbb{R}^n \setminus \{\mathbf{0}\}$ e seja "$\equiv$" a relação de equivalência sobre X definida por $x \equiv y \Leftrightarrow x/|x| = y/|y|$. Verifica-se que o quociente $X/\!\equiv$ é um espaço topológico homeomorfo a $\mathbb{S}^{n-1}$.

2.4.5 Ladrilhamento

Um caso particular importante da construção por espaço quociente é quando a relação de equivalência "$\equiv$" é definida por um conjunto $\Phi = \{\phi_1, \phi_2, \ldots, \phi_n\}$ de bijeções do espaço X para X. Essas bijeções geram um grupo $\hat{\Phi} = \langle \phi_1, \phi_2, \ldots, \phi_n \rangle$ que consiste de todas as combinações finitas das funções ϕ_i e de suas inversas. Dois pontos x, y são por definição *equivalentes módulo* Φ se e somente se existe $\phi \in \hat{\Phi}$ tal que $\phi(x) = y$. O espaço quociente de X por esta relação, que denotamos por X/Φ, é o *espaço de ladrilhamento* definido por Φ. Cada elemento desse espaço consiste de um ponto x e suas imagens sob todas as combinações das bijeções ϕ_i. Por exemplo, se X é o espaço $\mathbb{R}^n$ e ϕ_i é a translação $x \to x + e_i$, onde e_i é o i-ésimo vetor da base canônica, verifica-se que X/Φ é o toro n-dimensional $\mathbb{T}^n$.

2.4.6 Colagem

Uma instância importante da construção por espaço quociente é a *colagem* de dois subespaços topológicos Z' e Z'' de um espaço X, por uma bijeção $\phi\colon Z' \leftrightarrow Z''$ dada. A bijeção ϕ define implicitamente uma relação de equivalência "$\equiv$" sobre X:

$$x_1 \equiv x_2 \Leftrightarrow (x_1 = x_2) \vee (\,\phi(x_1) = x_2) \vee (\,\phi(x_2) = x_1)$$

O resultado da colagem é, por definição, o espaço quociente $X/\equiv$. Por exemplo, seja X a união de duas cópias disjuntas da bola $\mathbb{B}^d$, isto é, $X = \mathbb{B}^d \times \{-1, +1\}$, $Z' = \mathbb{S}^d \times \{-1\}$, $Z'' = \mathbb{S}^d \times \{+1\}$ e $\phi(x, -1) = \phi(x, +1)$. A colagem de Z' com Z'' transforma X num espaço topológico homeomorfo à esfera $\mathbb{S}^d$.
O conceito de colagem se estende de maneira óbvia a um conjunto de colagens simultaneas por duas ou mais bijeções $\phi_i\colon Z'_i \leftrightarrow Z''_j$. Note que Z'_i e Z''_j não precisam ser disjuntos.

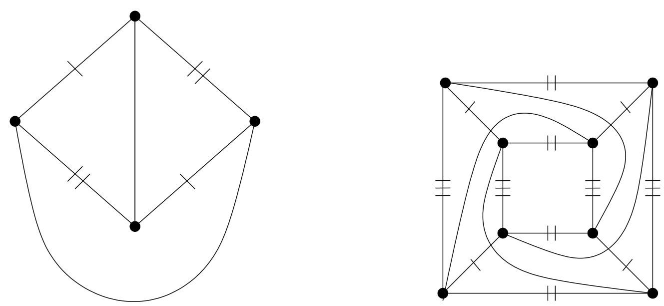
2.4.7 Cirurgia

Uma variante da técnica de colagem é a *cirurgia* de duas variedades topológicas X', X''. Nestas operações, remove-se de X' e X'' dois subconjuntos abertos Z' e Z'' e identifica-se cada ponto da fronteira de Z' com um ponto na fronteira de Z'', por um homeomorfismo dado ϕ. O resultado da cirurgia é a colagem de $X' \cup X''$ definida por ϕ. Note-se que X' e X'' podem ser a mesma variedade.

2.4.8 d-gemas

Uma maneira simples e eficaz de especificar variedades por meio de colagem é através de uma estrutura combinatória denominada *gema* (da abreviação do nome inglês, *graph-encoded manifold*), definida por Sóstenes e Mandel [22].

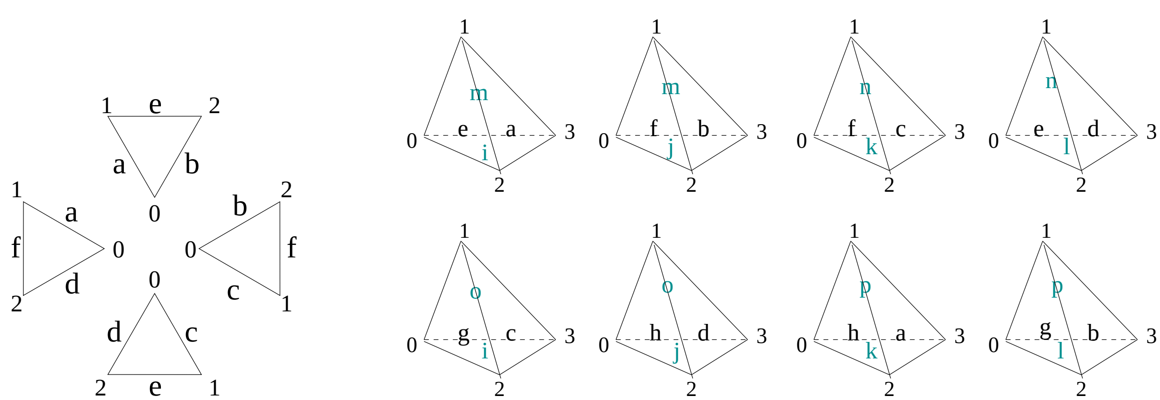
Uma *gema de dimensão d* (ou *d-gema*) consiste de um conjunto finito T de *nós* e $d+1$ emparelhamentos perfeitos $\sigma_0, \sigma_1, \ldots, \sigma_d$ de T (isto é, bijeções de T em T tais que $\sigma_i(x) \neq x$ e $\sigma_i^2(x) = x$, para todo $x \in T$). Uma d-gema pode ser representada de maneira óbvia por um grafo não dirigido sem laços, cujas arestas são rotuladas com "cores" $0, 1, \ldots, d$, de tal forma que em cada vértice incide exatamente uma aresta de cada cor. Veja a figura 2.5.



(a) (b)

Figura 2.5: Exemplos de gemas: (a) uma 2-gema para o plano projetivo $\mathbb{RP}^2$, (b) uma 3-gema para o espaço projetivo $\mathbb{RP}^3$. As marcas nas arestas indicam as cores.

O espaço topológico especificado por uma d-gema, por definição, é o resultado da colagem de um ou mais d-simplexos geométricos, segundo especificado pelos emparelhamentos σ_i. Mais precisamente, para cada elemento $x \in T$ tomamos um simplexo s_x, com vértices rotulados com as "cores" $0, 1, \ldots, d$; e, para cada i, colamos as faces de s_x e $s_{\sigma_i(x)}$ opostas aos vértices de cor i. A correspondência $\phi_{x,i}$ entre pontos das duas faces é feita por interpolação linear, casando-se vértices da mesma cor. A figura 2.6 mostra os esquemas de colagem especificados pelas gemas da figura 2.5.



(a) (b)

Figura 2.6: Interpretação das gemas da figura 2.5 como modelos de colagem: (a) o plano projetivo $\mathbb{RP}^2$, (b) o espaço projetivo $\mathbb{RP}^3$.

Observe-se que um mesmo espaço topológico pode ser definido de muitas maneiras distintas. Note-se também que os espaços resultantes das construções acima não são necessariamente variedades.

2.5 Construção de variedades bidimensionais

Para exemplificar os métodos acima, vamos a seguir ilustrar seu uso na construção de algumas variedades topológicas bidimensionais clássicas.

2.5.1 Modelos algébricos

Conforme já observamos, a esfera $\mathbb{S}^2$ pode ser definida implicitamente como o subconjunto de $\mathbb{R}^3$ que satisfaz a equação $x^2 + y^2 + z^2 = 1$. Outro exemplo é o toro $\mathbb{T}^2$ que pode ser definido implicitamente no $\mathbb{R}^4$ por $\mathbb{T}^2 = \{\, (x, y, u, v) \;:\; x^2 + y^2 = 1 \,\wedge\, u^2 + v^2 = 1 \,\}$.

A esfera $\mathbb{S}^2$ e o toro $\mathbb{T}^2$ podem ser definidos também como sub-espaços do $\mathbb{R}^3$ pela equações paramétricas:

$$\mathbb{S}^2 = \{\, (\sin\alpha\cos\beta,\, \sin\alpha\sin\beta,\, \cos\alpha) \;:\; \alpha \in [0, \pi], \beta \in [0, 2\pi] \,\}$$

$$\mathbb{T}^2 = \{\, ((2 + \cos\beta)\cos\alpha,\, (2 + \cos\beta)\sin\alpha,\, \sin\beta) \;:\; \alpha \in [0, 2\pi], \beta \in [0, 2\pi] \,\}$$

A garrafa de Klein $\mathbb{K}^2$ e o plano projetivo real $\mathbb{RP}^2$ podem ser modelados parametricamente no $\mathbb{R}^4$ pelas fórmulas:

$$\mathbb{K}^2 = \left\{ (2\cos\alpha, 2\sin\alpha\cos\beta\cos\frac{\alpha}{2},\, \sin\beta,\, \cos\beta\sin\frac{\alpha}{2}) \;:\; \alpha, \beta \in [0, 2\pi] \right\}$$

$$\begin{aligned}\mathbb{RP}^2 = \{\, ((1 + \cos\alpha)\cos\beta, (1 + \cos\alpha)\sin\beta,\, \sin\alpha\sin\frac{\beta}{2},\, \sin\alpha\cos\frac{\beta}{2}) \;:\; \\ \alpha \in [0, 2\pi], \beta \in [0, 4\pi] \,\}\end{aligned}$$

Note-se que estas variedades não são orientáveis.

2.5.2 Espaço quociente

Um exemplo clássico de variedade bidimensional definida por quociente é o plano projetivo (real) $\mathbb{RP}^2$, que é o quociente da esfera unitária $\mathbb{S}^2$ de $\mathbb{R}^3$ pela relação de equivalência "$\equiv$" que identifica cada ponto de $\mathbb{S}^2$ com seu antípoda, isto é $p \equiv -p$, para todo $p \in \mathbb{S}^2$.

2.5.3 Produto cartesiano

Além dos exemplos já vistos, $\mathbb{R}^2 = \mathbb{R} \times \mathbb{R}$ e $\mathbb{T}^2 = \mathbb{S}^1 \times \mathbb{S}^1$, outros exemplos de variedades bidimensionais que podem ser definidas por produto cartesiano são o *cilindro oco* (ou *anel*) $\mathbb{S}^1 \times \kappa\mathbb{B}^1$ e o *toro bidimensional* $\mathbb{T}^2 = \mathbb{S}^1 \times \mathbb{S}^1$ — ou seja,

$$\mathbb{T}^2 = \left\{ (x_1, x_2, x_3, x_4) \ : \ x_1^2 + x_2^2 = 1 \wedge x_3^2 + x_4^2 = 1 \right\}$$

2.5.4 Colagem de polígonos

O toro bidimensional $\mathbb{T}^2$ também pode ser obtido pela colagem de pares de lados opostos de um quadrado, como indicado na figura 2.7(a). As setas indicam a ordem de emparelhamento dos pontos em cada par de lados. A garrafa de Klein $\mathbb{K}^2$ também é obtida por uma colagem semelhante, em que dois dos lados são identificados em sentidos opostos, como indicado na figura 2.7(c).

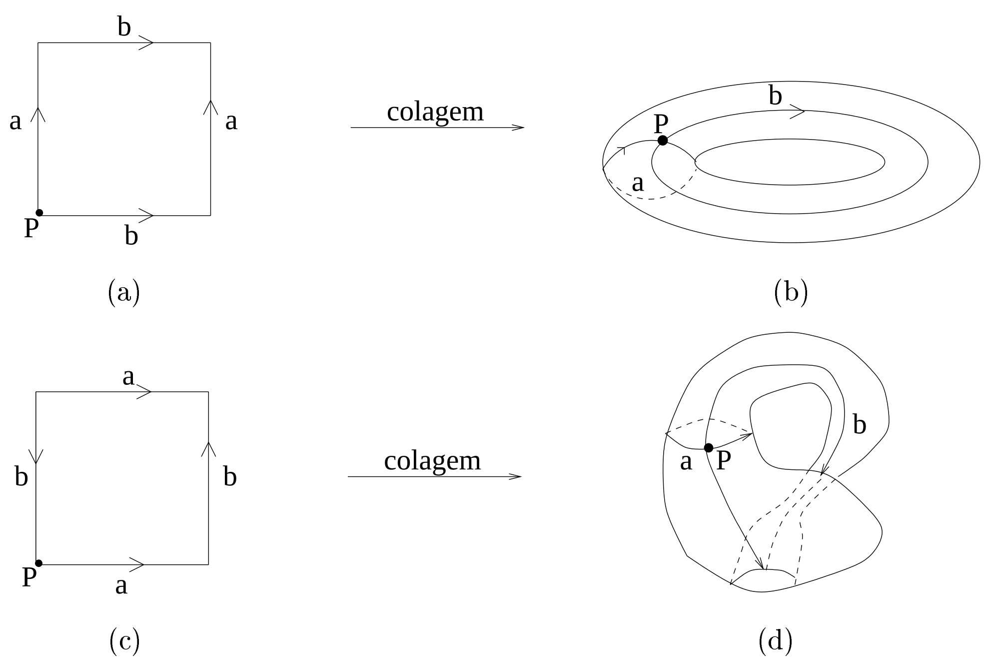


Figura 2.7: Esquema de colagem para a obtenção do toro $\mathbb{T}^2$ (a) e a garrafa de Klein $\mathbb{K}^2$ (c).

Outros esquemas de colagem do quadrado resultam na esfera $\mathbb{S}^2$, no plano projetivo $\mathbb{RP}^2$, na fita de Möbius $\mathbb{F}$ e no cilindro $\kappa\mathbb{B}^1 \times \mathbb{S}^1$. Na verdade, um resultado fundamental da topologia de conjuntos é que toda variedade bidimensional conexa e compacta pode ser obtida pela colagem de pares de lados de um polígono regular com número suficiente de lados [19].

2.5.5 Cirurgia

No caso de variedades bidimensionais, toda cirurgia pode ser reduzida a uma sequência de cirurgias simples denominadas *somas conexas*. A soma conexa de duas variedades bidimensionais disjuntas X_1 e X_2, denotada por $X_1 \# X_2$ consiste em remover o interior de dois discos fechados $D_1 \subseteq X_1$ e $D_2 \subseteq X_2$ e colar as bordas dos mesmos. Verifica-se que a topologia do resultado depende apenas de X_1 e X_2, quaisquer que sejam os discos D_1, D_2 e o homeomorfismo usado para colar suas bordas.

A figura 2.8 ilustra a soma conexa de dois toros, dando origem ao *bi-toro* $\mathbb{T}^2 \# \mathbb{T}^2$.

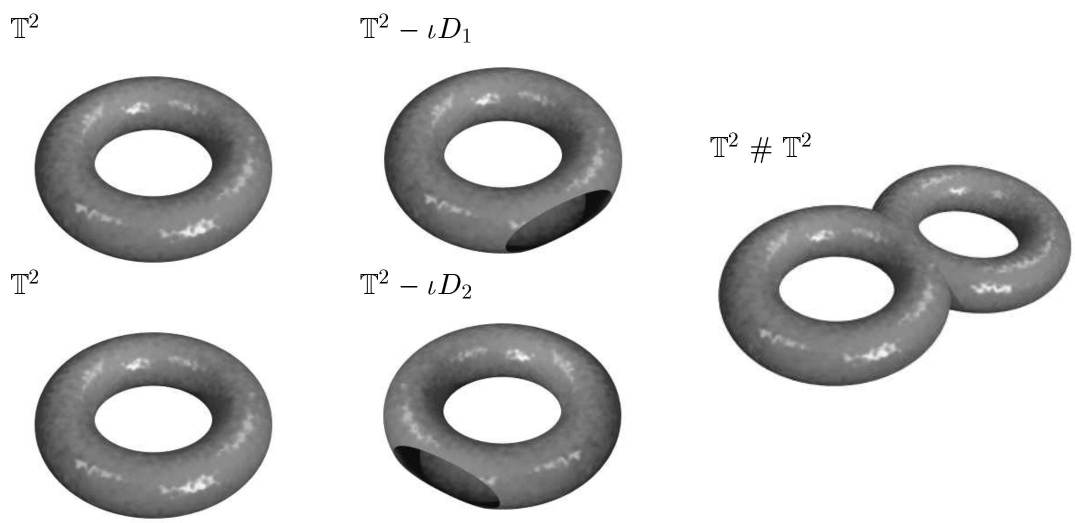


Figura 2.8: Soma conexa de duas 2-variedades.

Verifica-se que a garrafa de Klein $\mathbb{K}^2$ pode ser obtida como a soma conexa de dois planos projetivos, $\mathbb{RP}^2 \# \mathbb{RP}^2$.

Na teoria das variedades bidimensionais, as expressões *costurar uma alça* e *costurar uma fita de Möbius* são usados como sinônimos de efetuar a soma conexa com o toro $\mathbb{T}^2$ e com o plano projetivo $\mathbb{RP}^2$, respectivamente. Outro resultado fundamental da topologia é que toda variedade bidimensional conexa, compacta e sem borda pode ser obtida a partir de uma esfera, de um plano projetivo, ou de uma garrafa de Klein, por costura de um número finito de alças [19].

2.6 Construção de variedades tridimensionais

Nesta seção, vamos ilustrar o uso das técnicas da seção 2.4 para a construção de variedades tridimensionais. A maioria destes exemplos são extensões óbvias de construções de variedades bidimensionais; entretanto, em alguns casos, a extensão pode ser definida de várias maneiras, com resultados diferentes.

2.6.1 Modelo algébrico implícito

Dois exemplos de definições algébricas para variedades tridimensionais são a *hiperesfera* $\mathbb{S}^3 = \{\, (x, y, z, w) \ : \ x^2 + y^2 + z^2 + w^2 = 1 \,\}$ (um subconjunto do $\mathbb{R}^4$) e o *toro tridimensional*

$$\mathbb{T}^3 = \left\{\, (x, y, r, s, u, v) \ : \ x^2 + y^2 = 1 \ \wedge \ r^2 + s^2 = 1 \ \wedge \ u^2 + v^2 = 1 \,\right\}$$

(um subconjunto do $\mathbb{R}^6$).

2.6.2 Modelo paramétrico

A bola fechada $\kappa\mathbb{B}^3$ no $\mathbb{R}^3$ pode ser definida também pela equação paramétrica:

$$\kappa\mathbb{B}^3 = \{\, (r \cos\beta \cos\alpha,\ r \cos\beta \sin\alpha,\ r \sin\beta) \ : \ \alpha \in [0, 2\pi], \beta \in [-\pi/2, \pi/2], r \in [0, 1] \,\}$$

A esfera $\mathbb{S}^3$ pode também ser realizada pela equação paramétrica:

$$\mathbb{S}^3 = \{\, (\sin\alpha \sin\beta \cos\gamma, \sin\alpha \sin\beta \sin\gamma, \sin\alpha \cos\beta, \cos\alpha) \ : \ \alpha, \beta, \gamma \in [0, 2\pi] \,\}$$

2.6.3 Espaços quociente

O *espaço projetivo* (*real*) tridimensional $\mathbb{RP}^3$ pode ser definido, de maneira análoga ao plano projetivo $\mathbb{RP}^2$, como o quociente da hiperesfera $\mathbb{S}^3 = \{(x_1, \ldots, x_4) : x_1^2 + \cdots + x_4^2 = 1\}$ pela relação de equivalência que identifica cada ponto x de $\mathbb{S}^3$ com seu antípoda $-x$. Ao contrário de $\mathbb{RP}^2$, verifica-se que $\mathbb{RP}^3$ é uma variedade orientável.

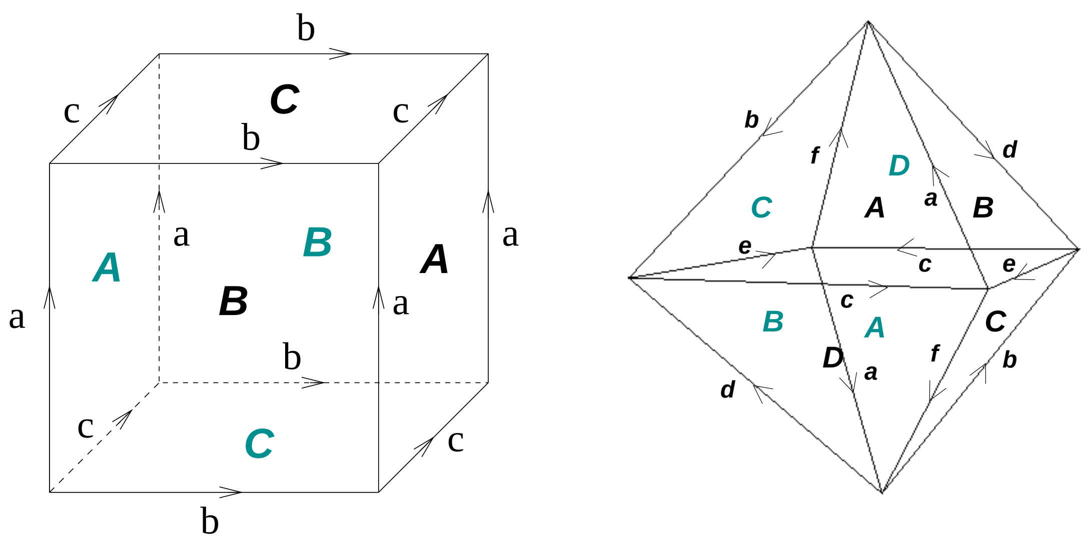
2.6.4 Produto cartesiano

Todo produto cartesiano de três variedades unidimensionais, ou de uma variedade bidimensional com uma variedade unidimensional, é uma variedade tridimensional. Por exemplo, $\mathbb{S}^1 \times \mathbb{S}^1 \times \mathbb{S}^1$ é outra maneira de definir o toro tridimensional $\mathbb{T}^3$. Note-se que $\mathbb{S}^2 \times \mathbb{S}^1$ é uma outra variedade tridimensional, distinta de $\mathbb{S}^3$ e de $\mathbb{T}^3$. Um outro exemplo é a *argola*, ou *toro maçico*, que é o produto cartesiano do círculo $\mathbb{S}^1$ e do disco $\kappa\mathbb{B}^2$ (veja a figura 2.4).

2.6.5 Colagem de poliedros

Poincaré propôs o esquema de colagem de poliedros em 1895, num de seus trabalhos clássicos [28]. Esta técnica é análoga ao esquema de colagem de polígonos para variedades 2D. A variedade 3D é descrita como o resultado da colagem de faces (A_i, B_i) de um ou mais poliedros, por meio de bijeções dadas ϕ_i. O emparelhamento das faces pode ser especificado, por exemplo, por rótulos colocados no interior das mesmas; e os homeomorfismos ϕ_i podem geralmente ser especificados por rótulos sobre as arestas ou sobre os vértices. Veja a figura 2.9. (Em alguns casos, entretanto, esta segunda convenção pode resultar em ambiguidades.)

A figura 2.9(a) mostra um esquema de colagem que resulta no toro tridimensional $\mathbb{T}^3$. Cada colagem identifica duas faces do cubo, de tal forma que arestas com mesmo rótulo são casadas respeitando a orientação. A figura 2.9(b) mostra um esquema de colagem para a construção do espaço projetivo real $\mathbb{RP}^3$.



(a) (b)

Figura 2.9: Esquema de colagem para a construção de algumas variedades tridimensionais.

A família dos *espaços-lente* proporciona infinitos exemplos de construção por colagem. O *espaço-lente de ordem* (m, n) é a variedade tridimensional resultante da colagem das faces superiores e inferiores de uma bipirâmide de n lados, após uma rotação de $2\pi(m/n)$ radianos. A figura 2.10 mostra o esquema de colagem do espaço-lente de ordem $(1, 3)$. Em particular, o espaço-lente de ordem $(1, 1)$ é a hiperesfera $\mathbb{S}^3$ e o espaço-lente de ordem $(2, 4)$ é o espaço projetivo $\mathbb{RP}^3$ (figura 2.9(b)).

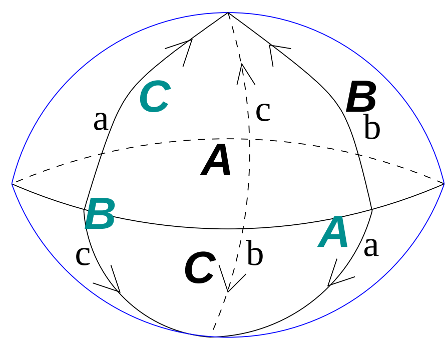


Figura 2.10: O esquema de colagem do espaço-lente de ordem $(1, 3)$.

Outros exemplos de construção por colagem são as construções por *coberturas ramificadas* de $\mathbb{S}^3$ [34]. Em particular, o espaço lente de ordem (m, n) é uma cobertura de n camadas da hiperesfera $\mathbb{S}^3$ com duas curvas de ramificação.

2.6.6 Ladrilhamento

O toro $\mathbb{T}^3$ também pode ser definido como o ladrilhamento do espaço $\mathbb{R}^3$ gerado pelas translações unitárias ϕ_x, ϕ_y, ϕ_z paralelas aos três eixos de coordenadas cartesianas.

2.6.7 Cirurgia

A cirurgia no caso tridimensional é mais complicada que seu análogo bidimensional. Recordamos que, no caso bidimensional, basta considerar cirurgias de duas variedades disjuntas, onde as partes removidas são os interiores de dois discos fechados. Como as bordas desses discos são círculos, elas podem ser casadas de apenas duas maneiras distintas, que produzem o mesmo resultado. Para o caso tridimensional, infelizmente, é necessário considerar a remoção de subconjuntos mais complicados, cujas fronteiras podem ser casadas de infinidade de maneiras, resultando numa infinidade de variedades não equivalentes.

Cirurgia de Dehn

Dehn propôs a construção de variedades tridimensionais orientáveis por cirurgia em 1910 [6]. Ele demonstrou que toda variedade tridimensional finita e orientável pode ser construída partindo-se da hiperesfera $\mathbb{S}^3$ através da remoção de um número finito de argolas $\mathbb{B}^2 \times \mathbb{S}^1$, abertas e disjuntas duas a duas e costurando-se essas argolas de volta com colagens apropriadas de suas fronteiras.

Decomposições de Heegaard

Outra técnica de cirurgia para a construção de variedades tridimensionais foi proposta por Heegaard em 1916 [15]. Heegaard mostrou que toda variedade tridimensional orientável pode ser obtida pela colagem de dois *sólidos com alças* ao longo de suas bordas. Um sólido com alça é uma variedade tridimensional compacta com borda contida no $\mathbb{R}^3$. A borda de tal sólido é uma variedade bidimensional orientável compacta, que, como mencionado na seção 2.5.5, é a soma conexa de um número finito de toros $\mathbb{T}^2$. Note-se que estes sólidos podem ser colados de inúmeras e complicadas maneiras pelas suas superfícies. Um diagrama mostrando como esta colagem é feita recebe o nome de *diagrama de Heegaard*.

2.7 Classificação topológica de variedades

A classificação topológica das variedades bidimensionais é uma teoria fechada. Sabe-se que a topologia de qualquer variedade 2D compacta, conexa e sem borda é inteiramente determinada por dois parâmetros — sua *característica de Euler* (um valor inteiro) e sua *orientabilidade* (um valor booleano). Estes dois parâmetros são *invariantes topológicos*, significando que eles são preservados por homeomorfismos e são *completos* no sentido que duas variedades são homeomorfas apenas se elas têm os mesmos invariantes.

Infelizmente, o êxito alcançado na classificação das variedades bidimensionais não pôde ainda ser emulado no caso das variedades tridimensionais. Em particular, não se conhece nemhum conjunto completo de invariantes topológicos para 3-variedades e ainda não se conhece um algoritmo para decidir se duas 3-variedades são homeomorfas, dadas suas representações (em qualquer dos modelos enumerados na seção 2.6) [34]. Nem sequer se conhece algum algoritmo para decidir se uma variedade tridimensional é homeomorfa à hiperesfera $\mathbb{S}^3$, que é a mais simples das variedades 3D.

A classificação de variedades com dimensão ≥ 4 promete ser ainda mais difícil. Na verdade, a falta de um algoritmo para reconhecer $\mathbb{S}^3$ implica que não sabemos sequer decidir se um espaço topológico 4D (descrito, por exemplo, por colagem de facetas de um politopo quadridimensional) é uma variedade.

Capítulo 3

Mapas d-dimensionais

3.1 Definição e conceito básicos

3.1.1 Mapas e elementos

Um *mapa d-dimensional* é uma partição finita $\mathcal{C}$ de uma variedade d-dimensional X tal que: (1) a fronteira $\partial_X c$ de todo elemento $c \in \mathcal{C}$ é a união de elementos de $\mathcal{C}$; e (2) para cada j-elemento c de $\mathcal{C}$, existe uma função contínua ψ_c de $\kappa\mathbb{B}^j$ sobre κc tal que a imagem inversa ψ_c^{-1} de quaisquer k-elemento e de $\mathcal{C}$ contido na fronteira de c é um número finito de subconjuntos disjuntos de $\partial\mathbb{B}^j$, cada um destes homeomorfo a e. Veja a figura 3.1.

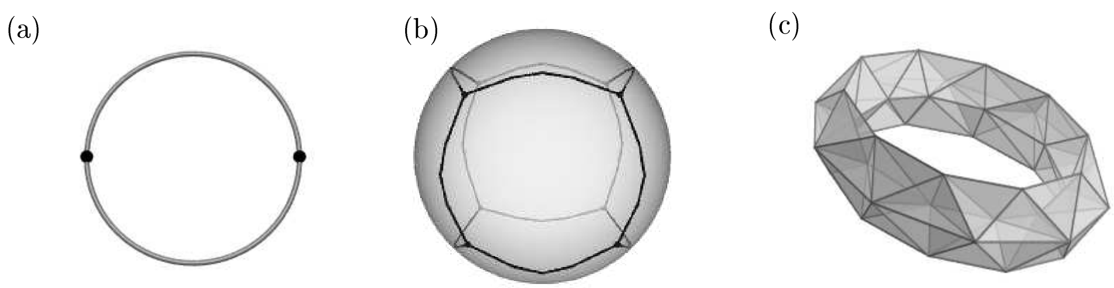


Figura 3.1: Mapa 1D (partição da borda do círculo) (a), mapa 2D (partição da esfera) (b) e mapa 3D (partição de $\kappa\mathbb{B}^2 \times \mathbb{S}^1$).

Nesta tese estamos interessados principalmente em mapas tridimensionais, cujos elementos têm dimensão menor ou igual a 3. Chamamos um 0-elemento de *vértice*, um 1-elemento de *aresta*, um 2-elemento de *face* e um 3-elemento de *célula*. Denotamos por $\mathbf{V}\mathcal{C}$, $\mathbf{E}\mathcal{C}$, $\mathbf{F}\mathcal{C}$ e $\mathbf{C}\mathcal{C}$ os vértices, arestas, faces e células de um mapa $\mathcal{C}$.

3.1.2 Isomorfismo

Dois mapas $\mathcal{C}'$, $\mathcal{C}''$ definidos sobre variedades X', X'' são ditos *isomorfos*, ou *topologicamente equivalentes*, se existe um homeomorfismo ϕ de X' para X'' tal que a imagem de cada elemento de $\mathcal{C}'$ por ϕ é um elemento de $\mathcal{C}''$. Nesse caso dizemos que ϕ é um *isomorfismo* de $\mathcal{C}'$ para $\mathcal{C}''$.

A *topologia* de um mapa $\mathcal{C}$ é a classe de equivalência de todos os mapas que são isomorfos a $\mathcal{C}$. A topologia pode ser descrita combinatorialmente de muitas maneiras. O método mais conhecido é o *esquema de colagem de poliedros*, que consiste de um conjunto finito de poliedros disjuntos, com colagens (bijeções) dadas entre pares de faces. As bijeções podem ser especificadas por meio de rótulos associados às arestas e faces, como discutido na seção 2.6.5. O interior de cada poliedro representa uma célula do mapa; e suas faces, arestas e vértices são os demais elementos. Veja a figura 3.2.

Verifica-se que a variedade subjacente ao mapa da figura 3.2 é o toro tridimensional $\mathbb{T}^3 = \mathbb{S}^1 \times \mathbb{S}^1 \times \mathbb{S}^1$. Vale notar que os modelos de colagem ilustrados nas figuras 3.2 e 2.9(a) (capítulo 2), descrevem a mesma variedade, mas mapas distintos (não isomorfos).

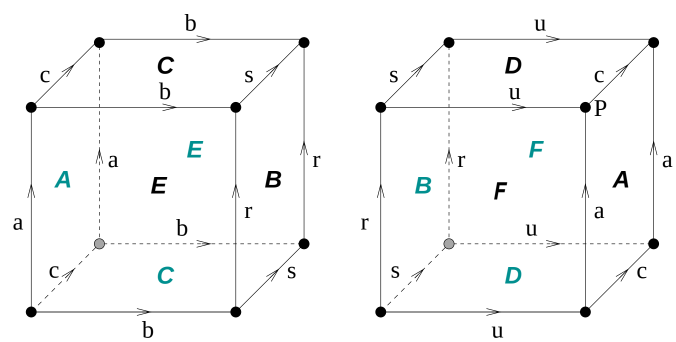


Figura 3.2: Esquema de colagem de poliedros para um mapa 3D sob o toro tridimensional $\mathbb{T}^3$.

3.1.3 Característica de Euler

A *característica de Euler* para um mapa topológico tridimensional $\mathcal{C}$ é definida pela fórmula

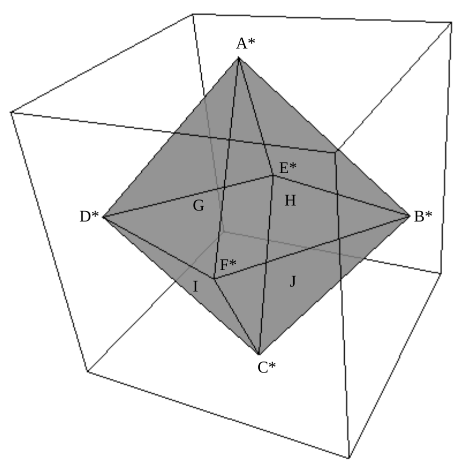
$$\chi(\mathcal{C}) = |\mathbf{V}\mathcal{C}| - |\mathbf{E}\mathcal{C}| + |\mathbf{F}\mathcal{C}| - |\mathbf{C}\mathcal{C}| \,. \tag{3.1}$$

Verifica-se que a característica de Euler só depende da topologia do espaço subjacente a $\mathcal{C}$ [4, 34]. Em particular, se $\mathcal{C}$ é um mapa tridimensional sem borda, verifica-se que $\chi(\mathcal{C}) = 0$. Segue-se que quaisquer dois mapas isomorfos têm a mesma característica de Euler, mas a recíproca não vale.

3.1.4 Dualidade

Dizemos que dois mapas $\mathcal{C}$ e $\mathcal{C}^*$, sobre uma mesma variedade n-dimensional, são *duais*, se existe uma bijeção $*$ entre os k-elementos de $\mathcal{C}$ e os $(n-k)$ elementos de $\mathcal{C}^*$, para todo k, tal que dois elementos c e d são incidentes em $\mathcal{C}$, se e somente se os elementos c^* e d^* são incidentes em $\mathcal{C}^*$, com as relações de ordem preservadas. Veja a figura 3.3.

(a)



(b)

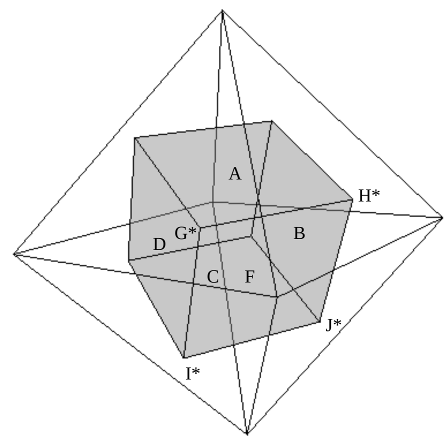


Figura 3.3: Um mapa bidimensional (a) e seu dual (b).

3.2 Exemplos de mapas tridimensionais

A seguir definimos alguns mapas tridimensionais que usaremos como exemplos nos capítulos seguintes. Por conveniência, a cada mapa atribuiremos um identificador convencional como `4-face`, `8-cell`, etc.

3.2.1 Mapas mínimos

Segue da definição que os mapas tridimensionais mais simples possuem pelo menos um vértice, uma aresta, uma face e uma célula. A figura 3.4 mostra dois mapas com estas propriedades.

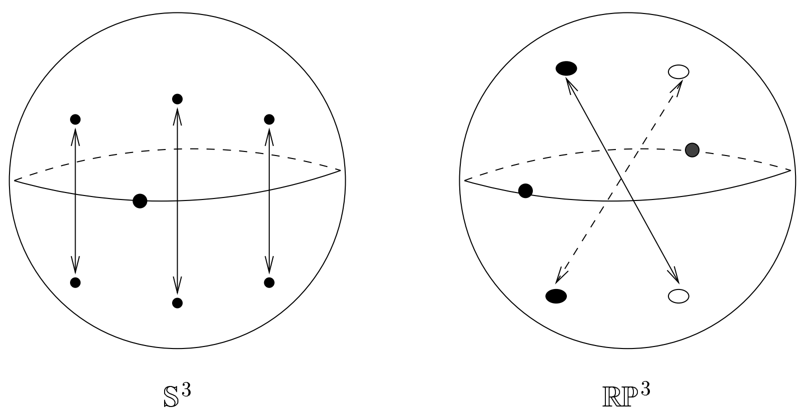


Figura 3.4: Esquemas de colagens para os mapas mínimos.

Os esquemas na figura 3.4 indicam a colagem da célula esférica consigo mesma através da identificação da sua borda livre. Estas colagens podem ser indicadas através das funções $(x, y, z) \leftrightarrow (x, y, -z)$, para o primeiro esquema e da função $(x, y, z) \leftrightarrow (-x, -y, -z)$ para o segundo esquema. Verifica-se que as variedades subjacentes a estes mapas são homeomorfas à hiperesfera $\mathbb{S}^3$ e ao espaço projetivo $\mathbb{RP}^3$, respectivamente.

3.2.2 Mapas derivados de poliedros regulares

Todo poliedro pode ser considerado um mapa tridimensional com borda, que possui uma única célula (o interior do poliedro) e cujas faces, arestas e vértices são os elementos correspondentes da fronteira do poliedro. Portanto, podemos falar no *mapa cubo*, *mapa octaedro*, etc. A figura 3.5 ilustra os mapas derivados de poliedros regulares e os identificadores que usaremos para os mesmos.

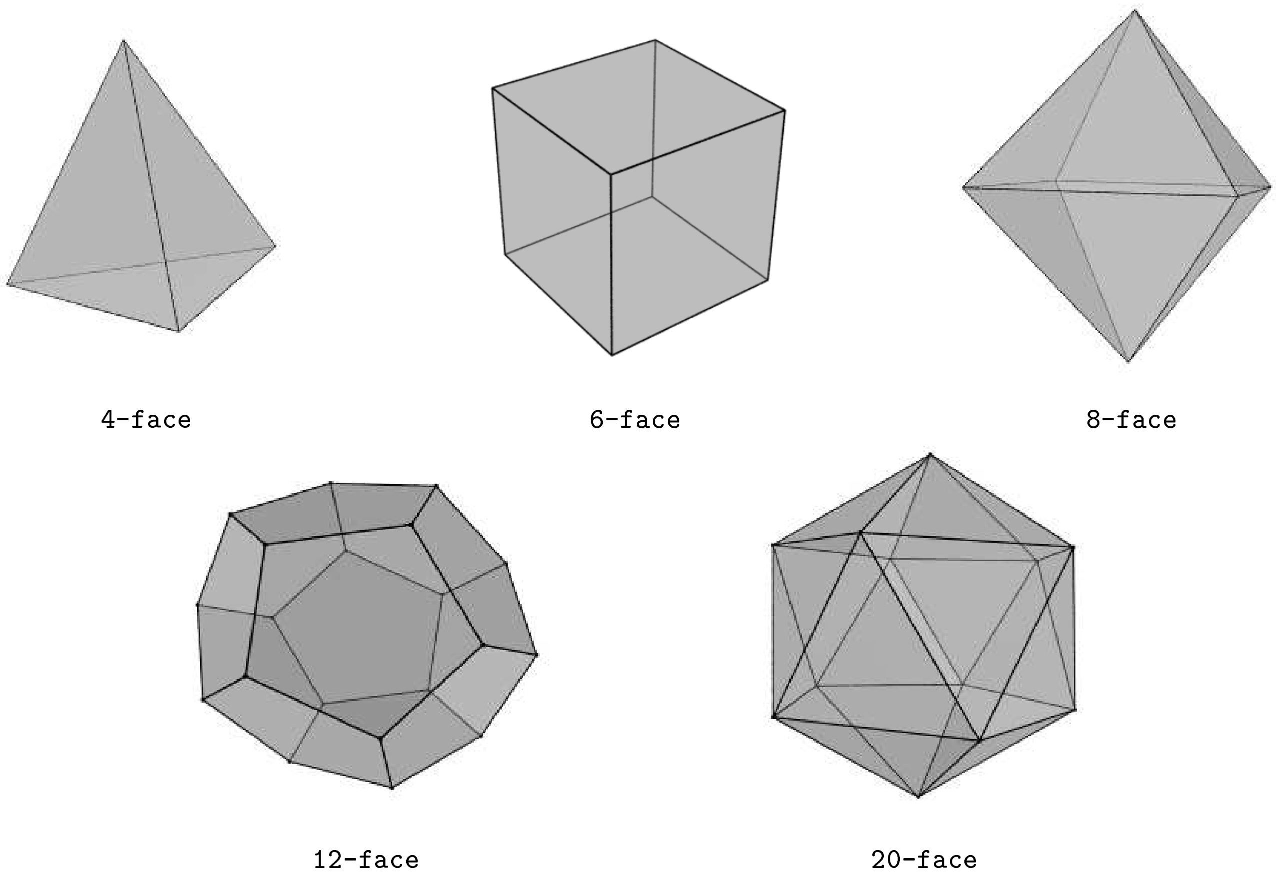


Figura 3.5: Mapas derivados de poliedros regulares.

3.2.3 Estrelas de arestas

Definimos o mapa *estrela de arestas de ordem* n, denotado por `edgestar[`n`]`, como sendo o resultado da colagem de n tetraedros em torno de uma aresta comum. Vide figuras 3.6 e 3.7.

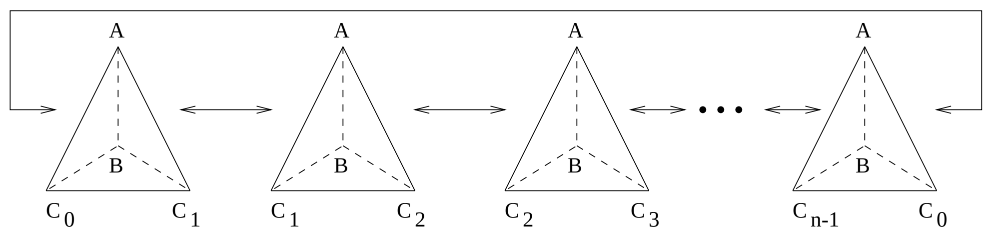
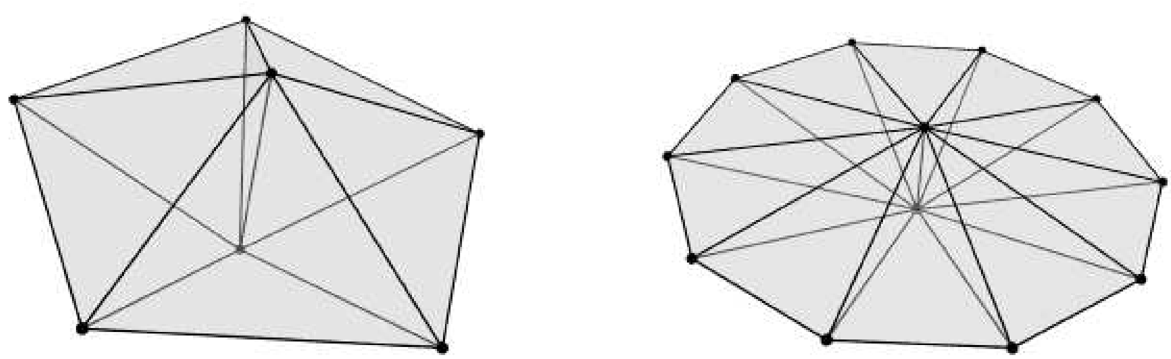


Figura 3.6: Esquema de colagem para o mapa estrela de arestas de ordem n (`edgestar[`n`]`).



`edgestar[5]` `edgestar[10]`

Figura 3.7: Exemplos de mapas estrelas de arestas.

3.2.4 Estrelas de vértices

Uma *estrela de vértice* é um mapa formado por n tetraedros colados de forma a cercar completamente um vértice comum. A topologia do mapa é determinada pela topologia de sua borda, o mapa 2D formado pelas faces dos tetraedros opostas a este vértice comum. Definimos em especial os mapas `4-vstar`, `8-vstar`, `20-vstar` e `60-vstar` cujas bordas são isomorfas às fronteira dos poliedros regulares tetraedro, octaedro, icosaedro e pentakisdodecahedro [2] respectivamente. A figura 3.8 mostra estes mapas.

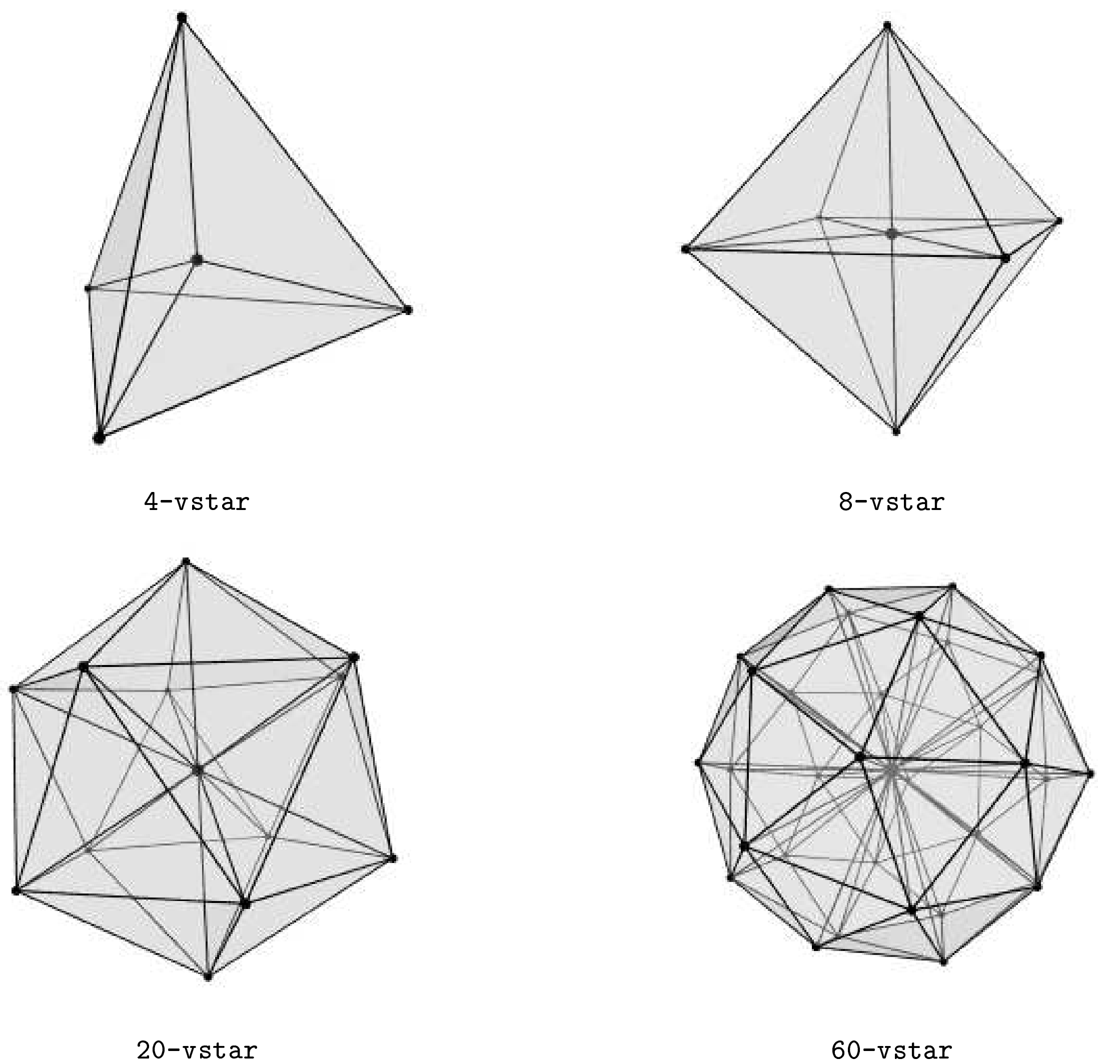


Figura 3.8: Exemplos de estrelas de vértices regulares.

3.2.5 Bipirâmides alongadas

A *bipirâmide alongada de ordem* n é um mapa estrela de vértice que consiste de $4n$ tetraedros colados em 4 camadas, como indicado na figura 3.9.

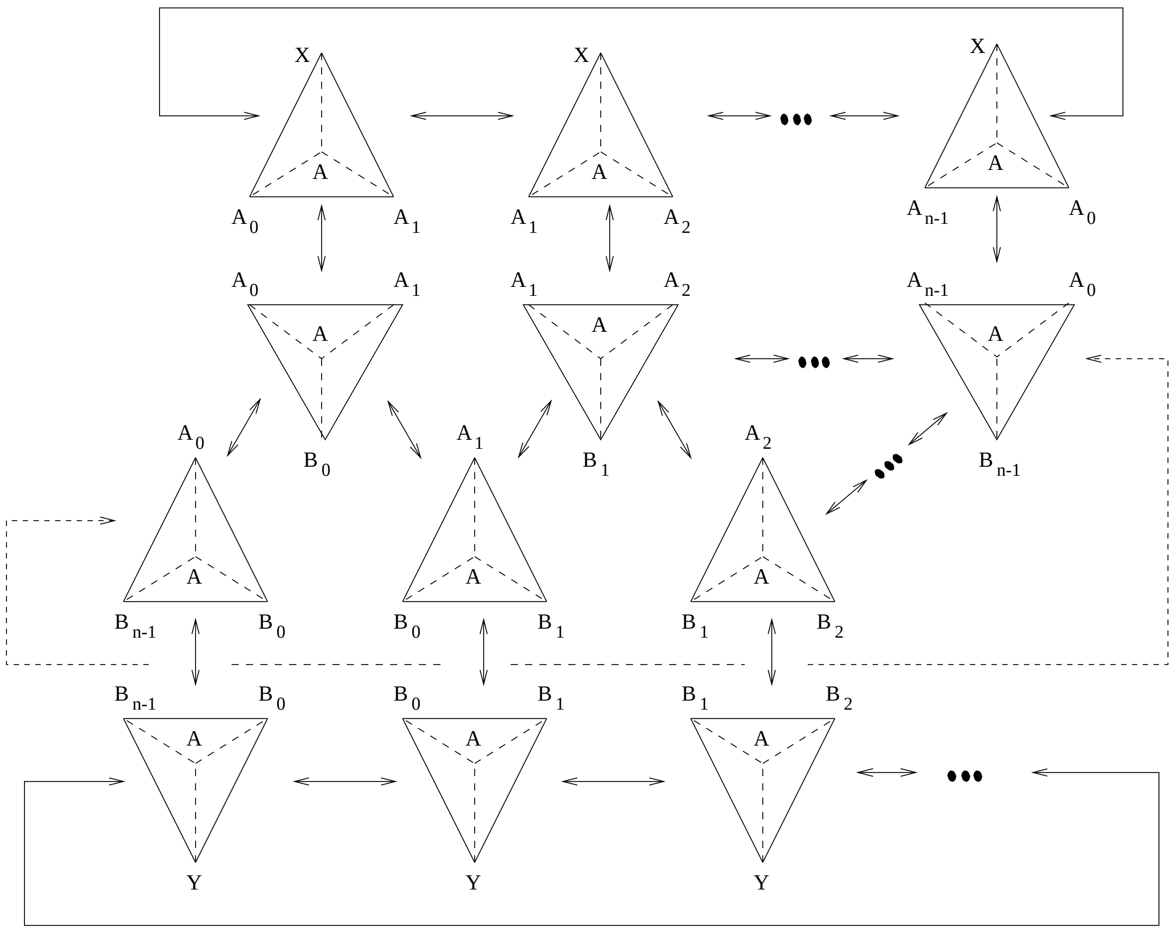


Figura 3.9: Esquema de colagem para mapas bipirâmide alongada.

As camadas extremas formam duas estrelas de arestas de ordem n, com um vértice comum A (o centro da estrela). Cada uma das camadas intermediárias é uma coroa de tetraedros unidos apenas por arestas. Denotamos estes mapas por `elongbip[`n`]`. Note-se que `elongbip[5]` é outro nome para a estrela de vértices `20-vstar`.

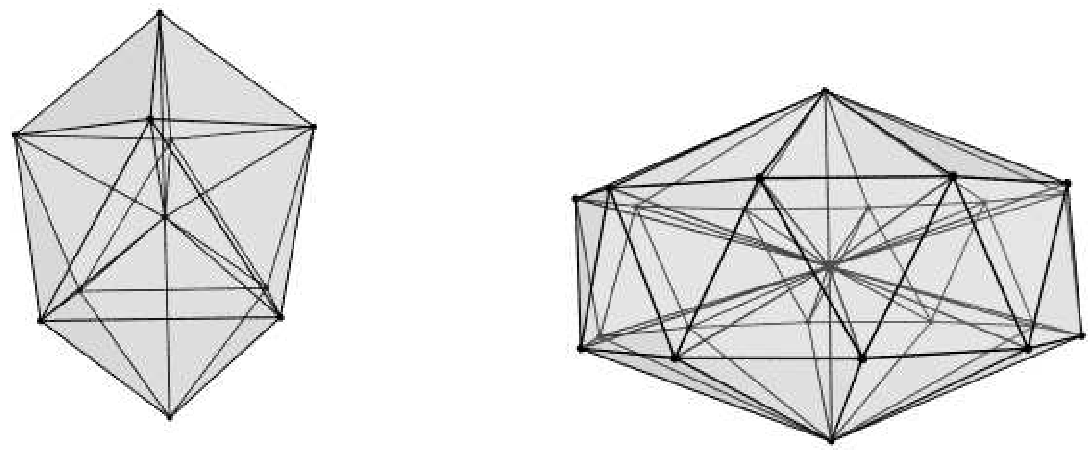
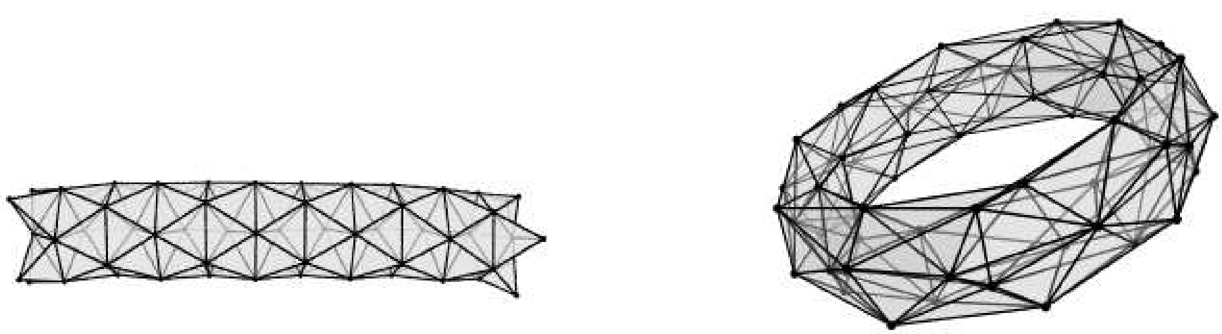


Figura 3.10: Os mapas `elongbip[4]` e `elongbip[10]`.

3.2.6 Barra e toro maciço

Mais geralmente, podemos empilhar n estrelas de arestas alternadas com $2n$ coroas de tetraedros, ambas de ordem m, de modo a formar uma barra aproximadamente cilíndrica, que denotamos por `column`$[m, n]$. Colando os dois extremos de uma barra, obtemos uma triangulação com a topologia de um toro maciço ($\kappa\mathbb{B}^2 \times \mathbb{S}^1$) que demotamos por `donut`$[m, n]$. A figura 3.11 mostra exemplos destes mapas.



`column`$[5, 10]$ `donut`$[5, 10]$

Figura 3.11: Exemplos de mapas barra e toro maciço.

3.2.7 Mapa prisma

Definimos o *mapa prisma de ordem* n, denotado por `prism[`n`]`, como sendo o resultado da colagem de $3n$ tetraedros, como ilustrado na figura 3.12.

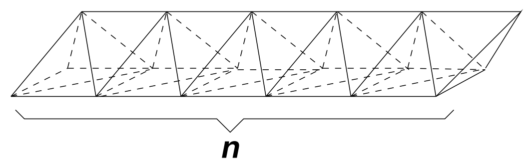


Figura 3.12: O mapa prisma `prism[`n`]`.

3.2.8 Mapa lâmina

Definimos o mapa *lâmina de ordem* (m, n), denotada por `slab[`m, n`]`, como o mapa que consiste de $2n$ cópias do mapa `prism[`m`]`, coladas em orientações alternadas como indicado na figura 3.13. Note-se que o número de tetraedros no mapa `slab[`m, n`]` é $6mn$.

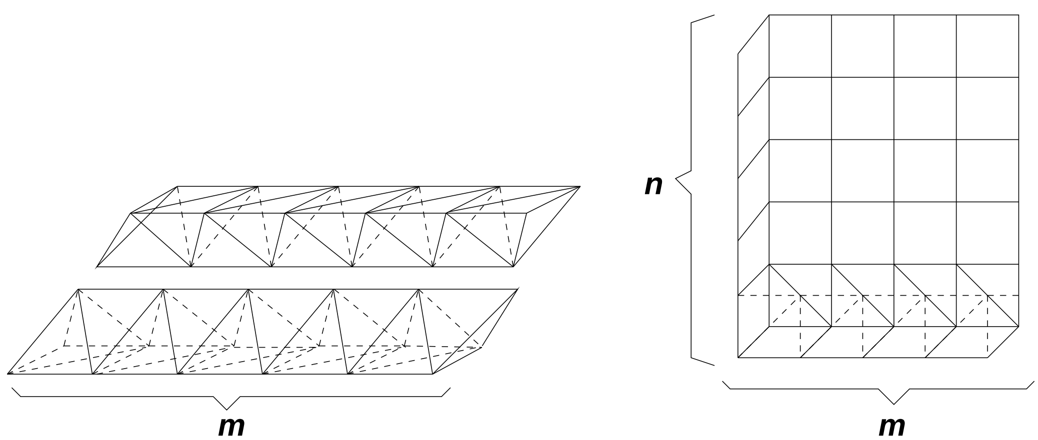


Figura 3.13: O mapa lâmina `slab[`m, n`]`.

3.2.9 O mapa tubo fechado

Colando as quatro bordas de um mapa lâmina `slab`[m, n], aos pares opostos, obtemos um mapa que denominamos *tubo fechado*, e denotamos por `tube`[m, n]. A variedade subjacente a este mapa é o *toro espesso* $\mathbb{T}^2 \times \kappa\mathbb{B}^1$; sua borda consiste de duas componentes conexas, cada uma com $2mn$ triângulos e topologia do toro $\mathbb{T}^2$.

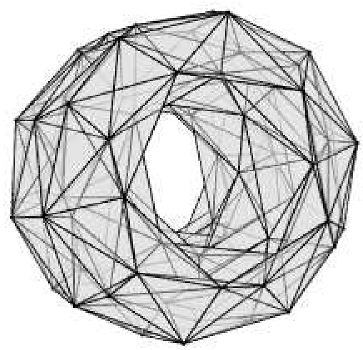


Figura 3.14: O mapa `tube`[5, 10].

3.2.10 O mapa monoplexo

O mapa *monoplexo*, denotado por `monoplex`, é uma triangulação degenerada que consiste de uma única célula tetráedrica, com as quatro faces coladas aos pares como indicado na figura 3.15.

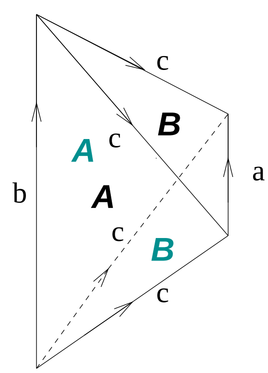


Figura 3.15: Esquema de colagem para a construção do mapa `monoplex`.

O mapa monoplexo possui apenas dois vértices, três arestas, duas faces e uma célula. A variedade subjacente é a hiperesfera $\mathbb{S}^3$.

3.2.11 Mapas associados a politopos regulares

O conjunto das faces de dimensões $j \leq 3$ de um politopo convexo de dimensão 4 constitui um mapa 3D, cuja variedade é homeomorfa à hiperesfera $\mathbb{S}^3$ (assim como a fronteira de um poliedro convexo é um mapa bidimensional, cuja variedade é homeomorfa à esfera $\mathbb{S}^2$).

De particular interesse são os politopos convexos *regulares*. Dizemos que um politopo convexo n-dimensional P é *regular* se para todo par de faces u, v da mesma dimensão existe uma isometria do $\mathbb{R}^n$ que leva P para P e u para v. Uma condição necessária (mas não suficiente) para que um mapa tridimensional $\mathcal{C}$ seja a fronteira de um politopo convexo e regular é que a estrela de cada célula c de $\mathcal{C}$ seja isomorfa a um mesmo mapa poliedro convexo k_c e que a soma dos ângulos diedrais dos poliedros em torno de cada aresta seja menor que 2π. Outra condição é que a soma dos ângulos dos poliedros em torno de cada vértice de $\mathcal{C}$ seja menor que 4π. A partir destas condições, prova-se que existem exatamente seis politopos convexos e regulares de dimensão 4, que denotaremos por `5-cell`, `8-cell`, `16-cell`, `24-cell`, `120-cell` e `600-cell` [4], conforme seu número de células.

Os politopos `5-cell`, `8-cell` e `16-cell` pertencem a famílias gerais, com representantes em qualquer dimensão m. Em particular, essas famílias incluem o tetraedro, o cubo e o octaedro no $\mathbb{R}^3$, respectivamente. Os politopos `120-cell` e a `600-cell` podem ser considerados análogos ao icosaedro e dodecaedro respectivamente, no sentido que estes são dois poliedros regulares que tem simetrias rotacionais de ordem 5 e não pertencem a famílias gerais. O politopo `24-cell` é específico do $\mathbb{R}^4$ e não possui nenhum análogo natural no $\mathbb{R}^3$.

A seguinte tabela relaciona o número de k-elementos ($0 \leq k \leq 3$) destes mapas e os respectivos mapas duais.

Mapa	$\|\mathbf{V}\mathcal{C}\|$	$\|\mathbf{E}\mathcal{C}\|$	$\|\mathbf{F}\mathcal{C}\|$	$\|\mathbf{C}\mathcal{C}\|$	Mapa Dual
`5-cell`	5	10	10	5	`5-cell`
`8-cell`	16	32	24	8	`16-cell`
`16-cell`	8	24	32	16	`8-cell`
`24-cell`	24	96	96	24	`24-cell`
`120-cell`	600	1200	720	120	`600-cell`
`600-cell`	120	720	1200	600	`120-cell`

O mapa `5-cell`, conhecido também pelos nomes de *pentacoro*, *pentatopo*, *pentacélula* e *hipertetraedro*, é a fronteira de um simplexo regular de dimensão 4. Veja a figura 3.16. Este mapa consiste de cinco tetraedros, cada um colado por uma face a um dos outros quatro. Observe-se que em toda aresta do mapa `5-cell` incidem exatamente 3 faces e 3 tetraedros.

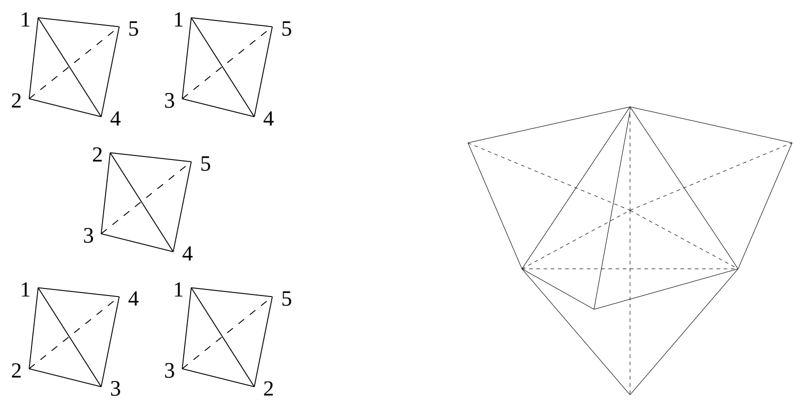


Figura 3.16: Esquema de colagem e colagem parcial, do mapa `5-cell`.

O mapa `8-cell` é a fronteira do *hipercubo* ou *tesseract*, que é a envoltória convexa dos 16 pontos $\{(\pm 1, \pm 1, \pm 1, \pm 1)\}$. Veja a figura 3.17. O mapa possui 8 células cúbicas, 24 faces quadradas, 32 arestas e 16 vértices. Note que em cada arestas incidem 3 faces e 3 células.

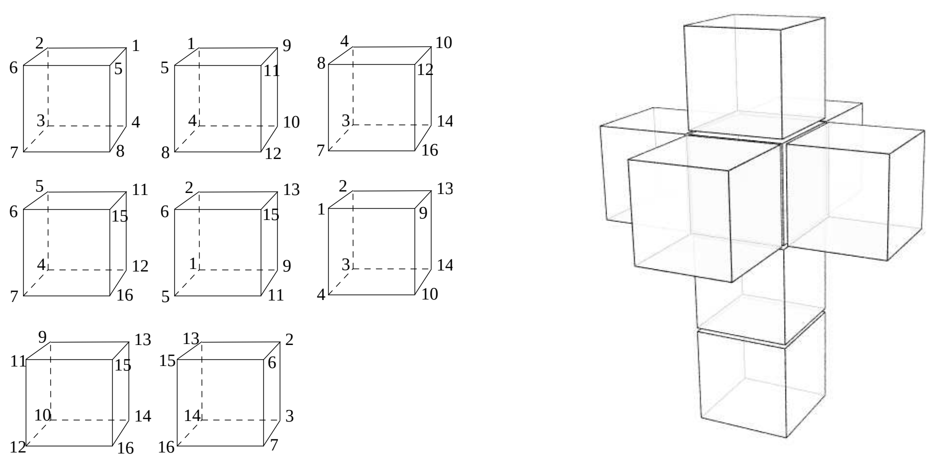


Figura 3.17: Esquema de colagem e colagem parcial, do mapa `8-cell`.

O mapa `16-cell` é o dual do mapa `8-cell`, e representa a topologia da fronteira do *hiperoctaedro*, ou *politopo em cruz*, que é a envoltória convexa dos 8 pontos abaixo:

$$\{(\pm 1, 0, 0, 0), (0, \pm 1, 0, 0), (0, 0, \pm 1, 0), (0, 0, 0, \pm 1)\}\,.$$

Veja a figura 3.18. O mapa consiste de 16 tetraedros, 32 faces triangulares, 24 arestas e 8 vértices. Cada aresta é incidente a 4 faces e 4 tetraedros.

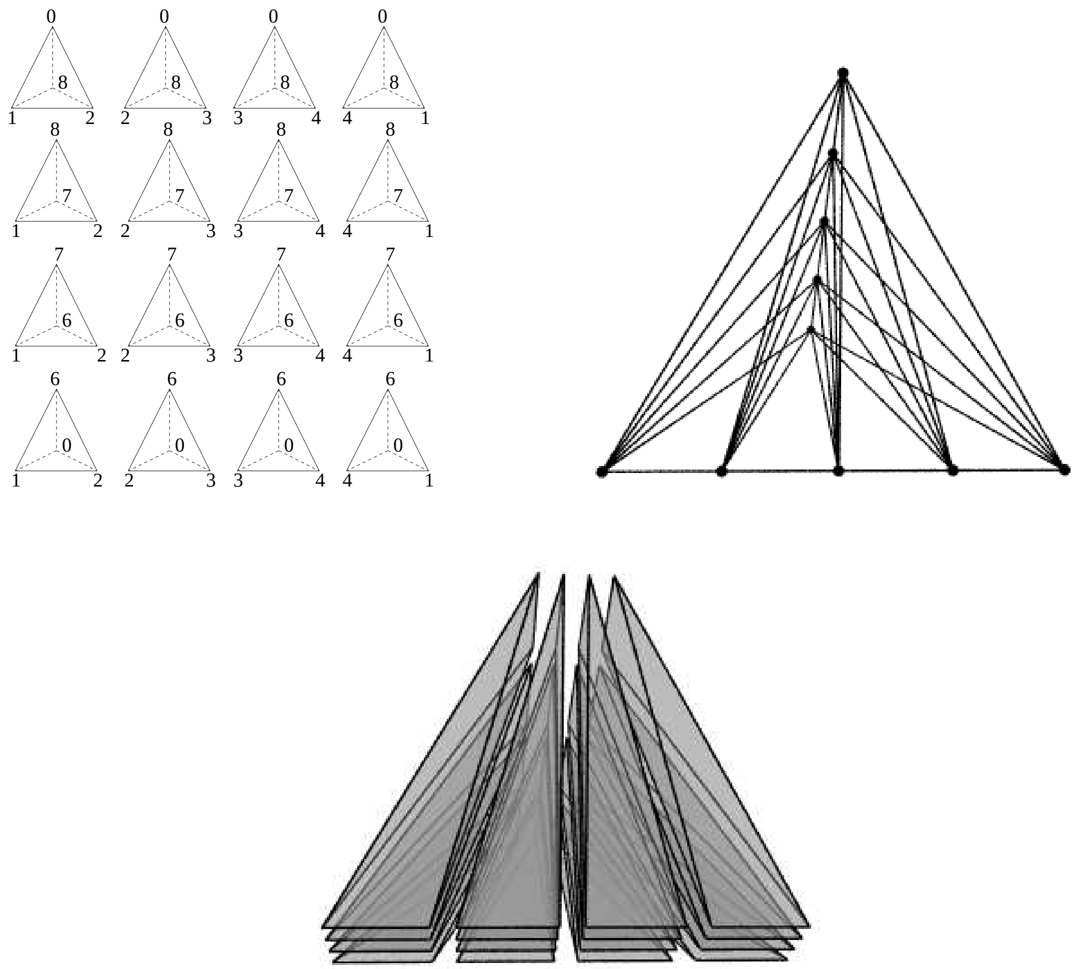


Figura 3.18: Esquema de colagem e colagens parciais do mapa `16-cell`.

O mapa `24-cell` corresponde à fronteira da envoltória convexa dos 24 pontos abaixo:

$$\{ \quad (\pm1,\pm1,0,0),(\pm1,0,\pm1,0),(0,0,\pm1,\pm1), \\ (0,\pm1,\pm1,0),(0,\pm1,0,\pm1),(\pm1,0,0,\pm1) \quad \} \ .$$

Este mapa consiste de 24 células octaedrais, coladas como indicado na figura 3.19. Ele possui 96 faces, 96 arestas e 24 vértices. Em cada aresta incidem três octaedros e em cada vértice incidem seis octaedros.

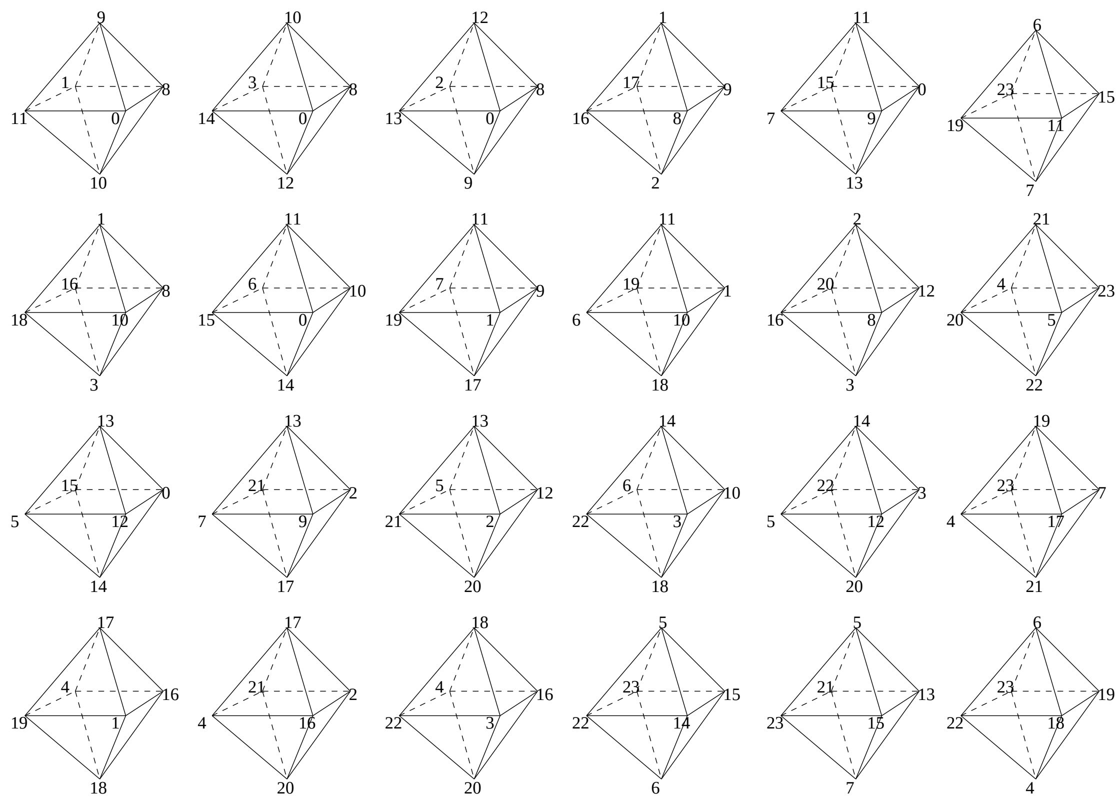


Figura 3.19: Esquema de colagem do mapa `24-cell`.

O mapa `600-cell` possui 120 vértices, 720 arestas, 1200 faces triangulares e 600 células tetraedrais. Este mapa é a fronteira do politopo *hipericosaedro*, que é a envoltória convexa dos 120 pontos abaixo:

- $(\pm 1, 0, 0, 0)$, em todas as permutações e sinais (8 pontos)
- $\frac{1}{2}(\pm 1, \pm 1, \pm 1, \pm 1)$, em todos os sinais (16 pontos)
- $\frac{1}{2}(\pm 1, \pm\rho, \pm\rho^{-1}, 0)$, todas as permutações pares (96 pontos)

onde $\rho = (1 + \sqrt{5})/2 = 1.61803\ldots$ é a "razão de ouro". Veja a figura 3.20.

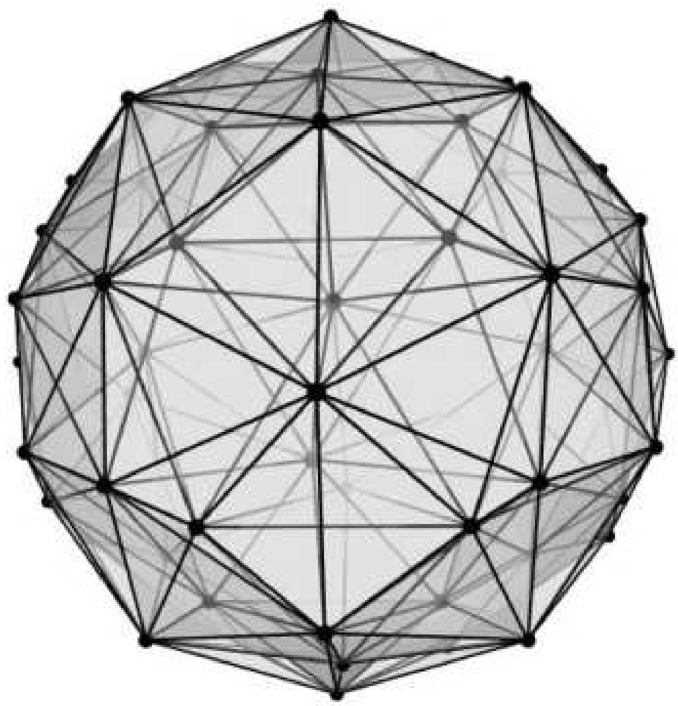


Figura 3.20: Uma projeção no $\mathbb{R}^3$ do mapa `600-cell`.

A montagem do mapa `600-cell`, que consiste de 600 tetraedros, é demasiado complexa para ser efetuada manualmente. Consequentemente, implementamos um algoritmo especifico para este propósito, que usa a técnica de cirurgia de Dehn (veja seção 2.6), conforme descrita por Davis [5]. O algoritmo constrói inicialmente duas argolas maciças `donut`$[5, 10]$ cada uma com 150 tetraedros e uma argola oca `tube`$[5, 10]$, com 300 tetraedros que atua como uma camada de cola entre elas. A borda de cada argola maciça e cada uma das bordas da argola oca, têm 200 faces triangulares.

O mapa `120-cell` é o dual do mapa `600-cell`. Ele é a fronteira de um politopo convexo regular, conhecido como *hiperdodecaedro*, que possui 600 vértices, 1200 arestas, 720 faces pentagonais e 120 células dodecaedrais. Veja a figura 3.21.

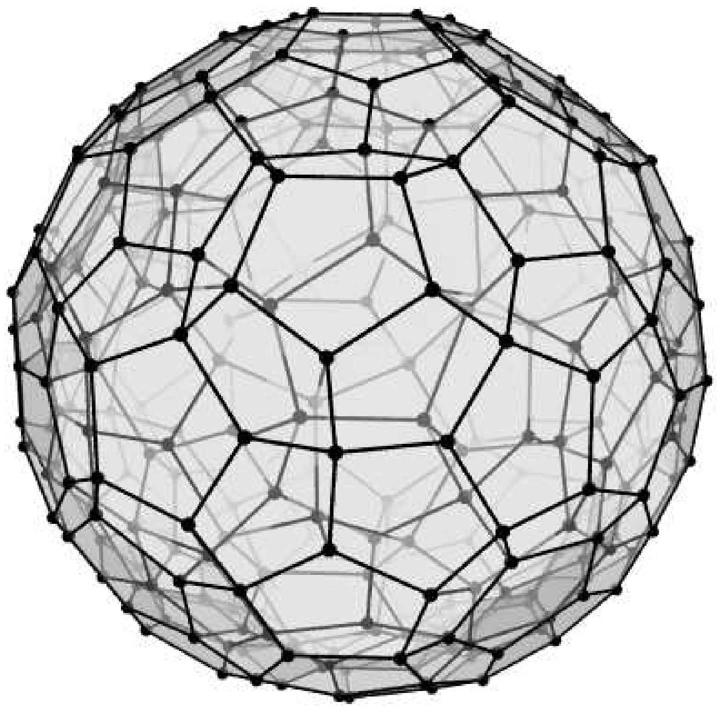


Figura 3.21: Uma projeção no $\mathbb{R}^3$ do mapa `120-cell`.

O mapa `120-cell` foi construido a partir do mapa `600-cell`, por dualidade topológica (veja-se o programa `DualPolytope` no apêndice A).

3.2.12 O mapa simplexo truncado

O mapa *simplexo truncado*, que denotamos por `10-cell`, é a fronteira de um politopo 4D convexo que possui 10 vértices, 30 arestas, 30 faces e 10 células, sendo 5 destas tetraedros e 5 octaedros. Este politopo não é regular, uma vez que possui dois tipos diferentes de células (tetraedros e octaedros). O mapa `10-cell` é seu próprio dual. Este mapa pode ser construido mediante o esquema de colagem ilustrado na figura 3.22.

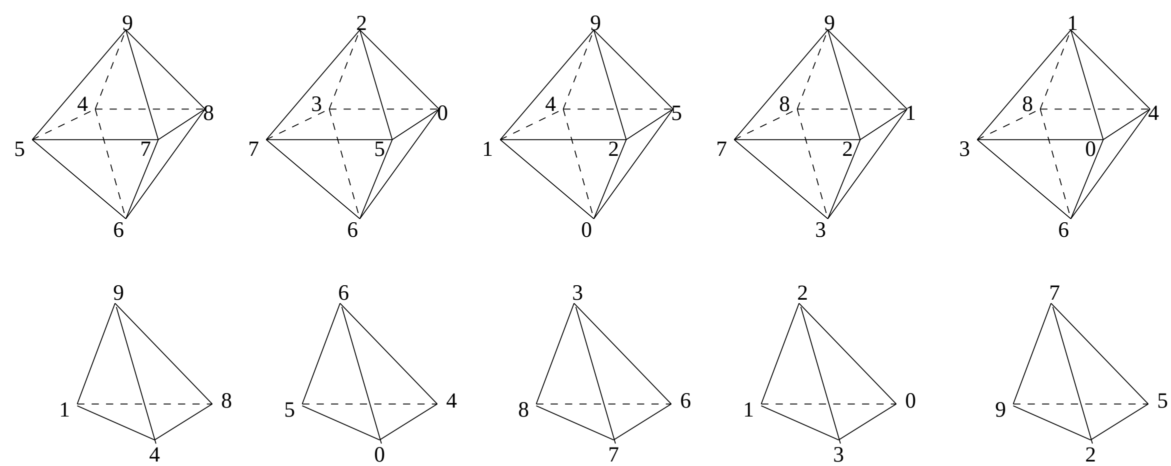


Figura 3.22: Esquema de colagem do mapa simplexo truncado.

3.2.13 Grades toroidais

Nas seções 2.6.5 e 3.1.2 vimos exemplos de mapas com células cúbicas na variedade $\mathbb{T}^3$ (veja também as figuras 2.9(a) e 3.2) respectivamente com uma e duas células. Generalizando esses exemplos, definimos uma *grade toroidal tridimensional* de ordem $m \times n \times p$, denotada por `3-torus`$[m, n, p]$, como sendo o resultado da colagem de mnp células cúbicas $c_{i,j,k}$ para $i \in [0, \ldots, m-1]$, $j \in [0, \ldots, n-1]$ e $k \in [0, \ldots, p-1]$. Especificamente, a face do cubo $c_{i,j,k}$ que está voltada para a direção $+X$ é colada com a face $-X$ do cubo $c_{i'\,j,k}$ onde $i' = (i+1) \bmod m$; e analogamente para as outras direções. Veja as figuras 3.23 e 3.24.

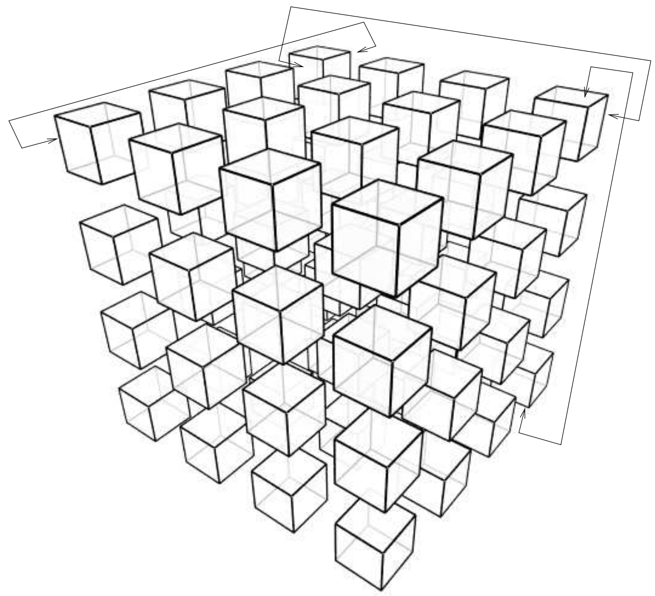


Figura 3.23: Esquema de colagem do mapa `3-torus[4,4,4]`.

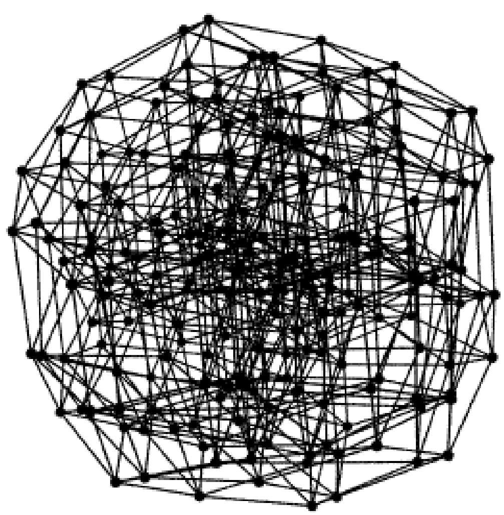


Figura 3.24: Uma projeção no $\mathbb{R}^3$ do mapa `3-torus[6,6,6]`.

Capítulo 4

Triangulação do mapa

Uma célula ou face de um mapa tridimensional pode ter um número arbitrário de vértices e a sua fronteira pode apresentar topologias arbitrariamente complexas. A fim de construir modelos geométricos de tais elementos, é quase que obrigatório subdividi-los em partes menores, de complexidade topológica limitada.

4.1 Simplexos e triangulações

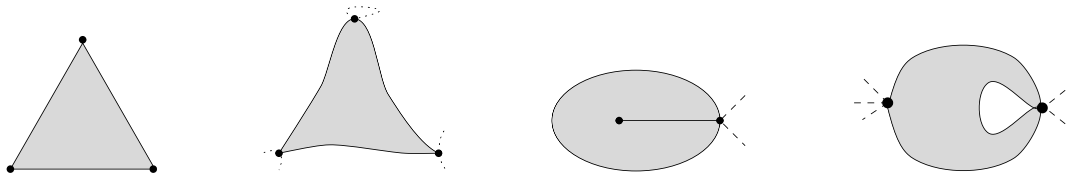
4.1.1 Refinamentos

Em outras palavras, em vez de modelar o mapa original $\mathcal{C}$, nós modelamos um mapa $\mathcal{C}'$ que é um *refinamento* de $\mathcal{C}$, no sentido de partição de conjuntos; ou seja, todo elemento de $\mathcal{C}$ é uma união finita de elementos de $\mathcal{C}'$ e todo elemento de $\mathcal{C}'$ está contido em algum elemento de $\mathcal{C}$.

Em geral, as faces, arestas e vértices adicionais introduzidas neste processo de refinamento serão omitidas nas imagens finais. Mais precisamente, cada elemento de $\mathcal{C}'$ deve herdar os atributos do elemento original do mapa $\mathcal{C}$ que o contém.

4.1.2 Simplexo topológico

Um k-elemento c de um mapa $\mathcal{C}$ é dito um *simplexo topológico* se existe um mapa contínuo ψ, do fecho $\kappa\Delta$ de algum simplexo geométrico Δ para o fecho de c, tal que a restrição de ψ a cada face f de Δ é um homeomorfismo de f para um elemento de $\mathcal{C}$. Dizemos que um k-simplexo topológico c de $\mathcal{C}$ é *próprio* se ψ pode ser escolhido de forma a ser um homeomorfismo (isto é, se as imagens das j-faces de Δ são disjuntas aos pares), como ilustrado na figura 4.1(b). Caso contrário dizemos que c é um simplexo *impróprio*, como ilustrado nas figuras 4.1(c) e 4.1(d).



(a) (b) (c) (d)

Figura 4.1: Um 2-simplexo geométrico (a) e 2-simplexos topológicos: próprio (b), impróprios (c) e (d).

4.1.3 Triangulação

Uma *triangulação d-dimensional* é um mapa d-dimensional cujos elementos são simplexos topológicos. Uma triangulação é *própria* se todos os seus simplexos são próprios.

4.1.4 Orientação topológica de um simplexo

Uma ordenação dos vértices de um d-simplexo topológico c define uma *orientação* para o mesmo. Duas ordenações definem a mesma orientação se estas diferem num número par de transposições de pares de vértices. Denotamos por $\langle v_0, \ldots, v_d \rangle$ a orientação definida pela ordenação $v_0, \ldots, v_d$, isto é, a classe de permutações de vértices com a mesma paridade que $v_0, \ldots, v_d$. Assim, por exemplo $\langle abcd \rangle$ e $\langle adbc \rangle$ denotam a mesma orientação, que é diferente de $\langle bcda \rangle$.

Dizemos que uma orientação $\langle v_0, \ldots, v_d \rangle$ de um d-simplexo c *induz* a orientação $\langle v_0, \ldots, v_{d-1} \rangle$, sobre a faceta f de c com vértices $\{v_0, \ldots, v_{d-1}\}$. Veja a figura 4.2. Por esta definição, considerando todas as permutações equivalentes dos vértices de c, nós obtemos orientações induzidas para todas suas facetas. Observe-se que duas ordens equivalentes dos vértices de c induzem orientações equivalentes em cada faceta de c.

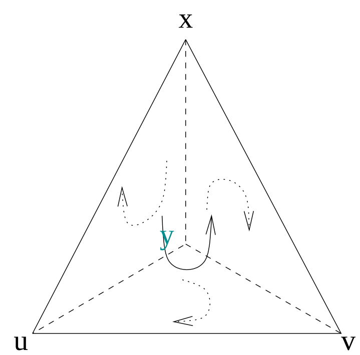


Figura 4.2: Orientações induzidas nas faces de um 3-simplexo pela orientação $\langle u, v, x, y\rangle$ do mesmo.

Dizemos que duas orientações σ_1 e σ_2 de dois simplexos topológicos adjacentes são *coerentes* se estes induzem orientações *opostas* sobre qualquer faceta compartilhada. Por exemplo, $\langle w, u, v, x\rangle$ e $\langle w, v, u, y\rangle$ são orientações coerentes para os dois tetraedros da figura 4.3.

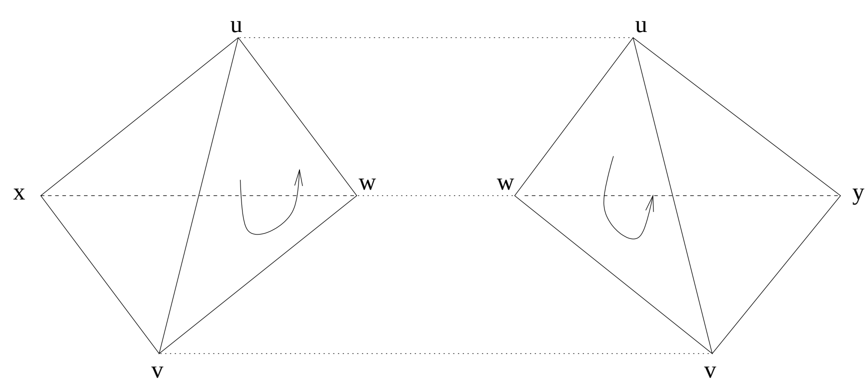


Figura 4.3: Orientações coerentes para tetraedros adjacentes.

Dizemos também que uma triangulação é *orientável* quando podemos atribuir uma orientação a cada um de seus 3-simplexos de tal forma que todos os pares adjacentes possuam orientações coerentes. Finalmente, dizemos que um mapa tridimensional $\mathcal{C}$ é *orientável*, se ele possui um refinamento $\mathcal{T}$ que é uma triangulação orientável. Verifica-se que para $d \leq 3$ uma variedade é orientável (no sentido da seção 2.3) se e somente se todo (ou algum) mapa sobre a mesma é orientável.

4.2 Subdivisão baricêntrica

A *subdivisão baricêntrica topológica* é um refinamento padrão $\mathcal{C}^\Delta$ de qualquer mapa d-dimensional $\mathcal{C}$ em simplexos topológicos próprios.

A subdivisão baricêntrica do j-esqueleto de $\mathcal{C}$ é definida recursivamente como segue. Para $j = 0$, a subdivisão é trivial, isto é, todo vértice de $\mathcal{C}$ é um vértice de $\mathcal{C}^\Delta$. Para $j > 1$, subdividimos recursivamente o i-esqueleto de $\mathcal{C}$, para todo $0 \leq i < j$. Para cada j-elemento c de $\mathcal{C}$, seja ψ uma função de $\kappa\mathbb{B}^j$ para κc que satisfaz as condições da definição de mapa (veja seção 3.1). Relembramos que a imagem inversa ψ^{-1} de qualquer j-elemento e de $\mathcal{C}$, contido na fronteira de c, é um número finito de subconjuntos disjuntos de $\partial\mathbb{B}^j$, cada um destes homeomorfo a e. Para cada j-elemento t de $\mathcal{C}^\Delta$ contido em ∂c e para cada componente t' de $\psi^{-1}(t)$, seja t^+ a união de todos os raios abertos que conectam o centro o de $\mathbb{B}^j$ a pontos em t'. A subdivisão de c em $\mathcal{C}^\Delta$ consiste do vértice $\{\psi(o)\}$ e as imagens de tais conjuntos t^+ pela função ψ. Veja figura 4.4.

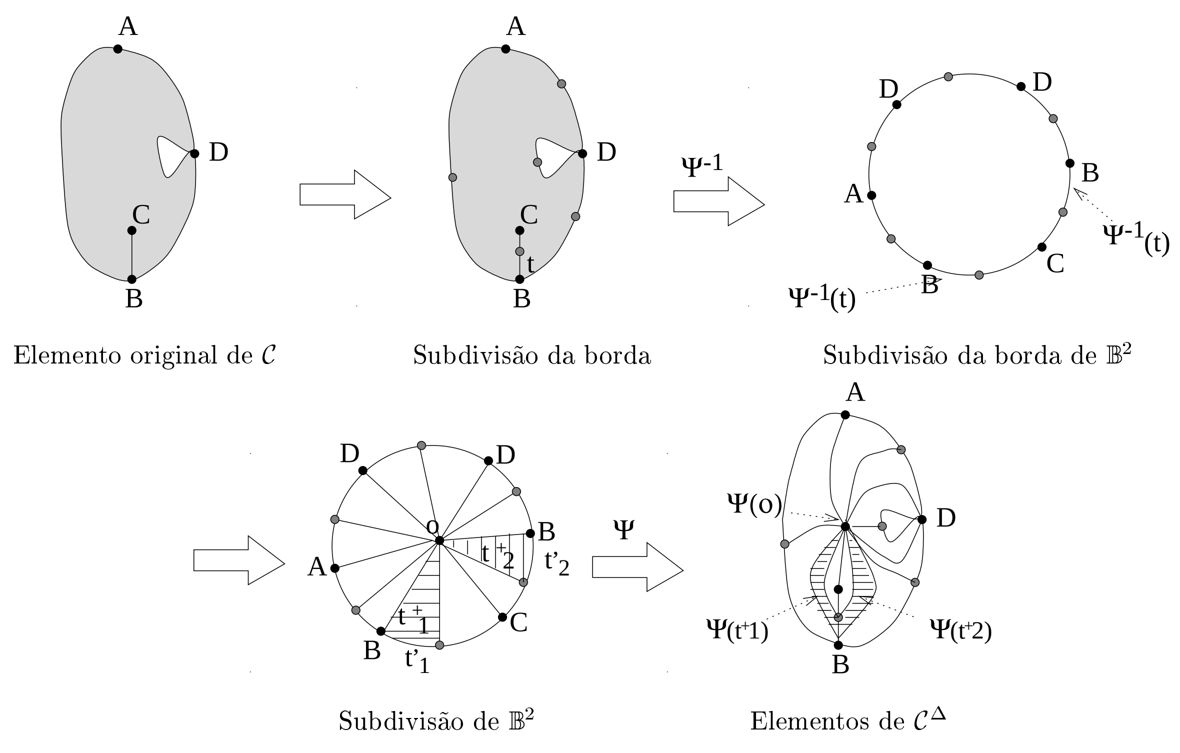


Figura 4.4: Um 2-elemento e sua subdivisão baricêntrica topológica.

A figura 4.5 ilustra esta seqüência de passos na subdivisão baricêntrica do mapa `6-face`.

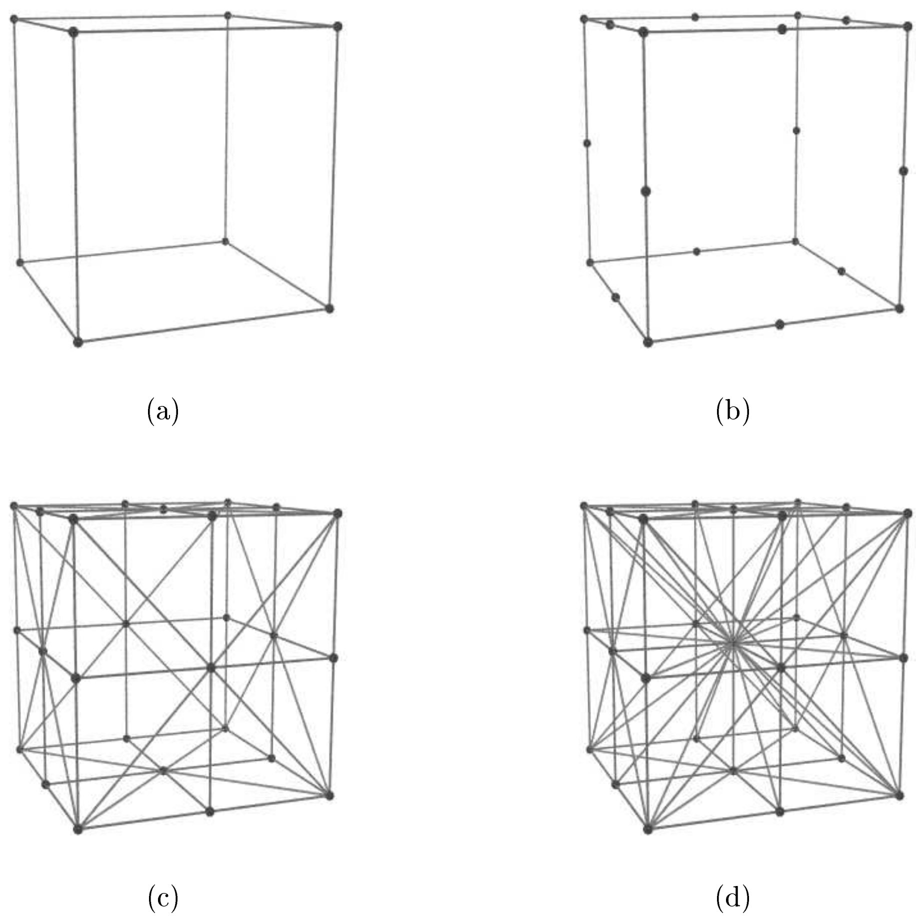


Figura 4.5: O mapa `6-face` (a), a subdivisão do 1-esqueleto (b), a subdivisão do 2-esqueleto (c) e a subdivisão do 3-esqueleto (d).

A figura 4.6 exibe a subdivisão baricêntrica, de maneira explodida, de outros mapas derivados de poliedros regulares.

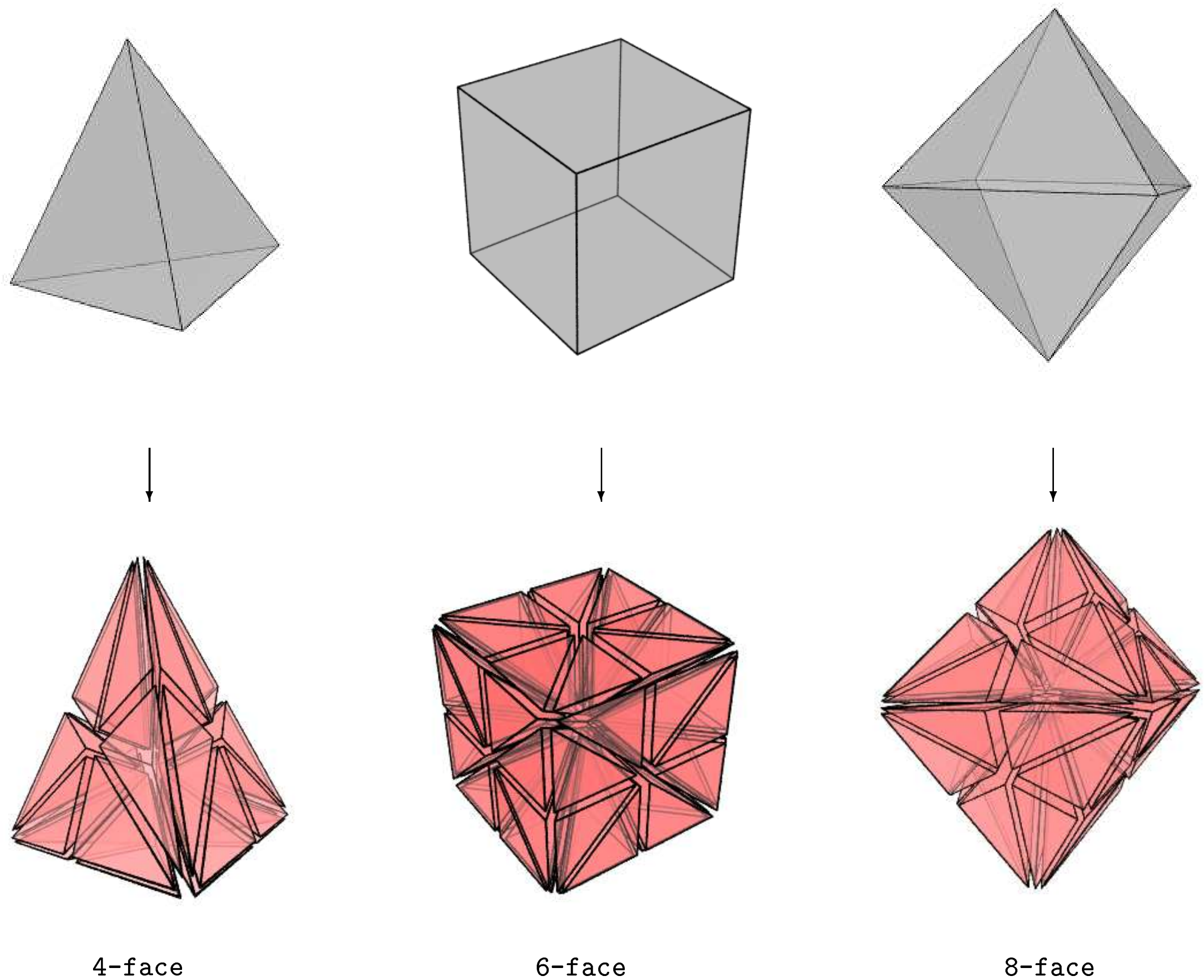


Figura 4.6: Subdivisão baricêntrica de mapas derivados de poliedros regulares.

4.3 Propriedades da subdivisão baricêntrica

A subdivisão baricêntrica de um mapa tridimensional $\mathcal{C}$ é essencialmente única; isto é, todas as subdivisões $\mathcal{C}^{\Delta}$ obtidas a partir de um mapa $\mathcal{C}$ por diferentes mapeamentos ψ são topologicamente equivalentes.

Se $\mathcal{C}$ é um mapa d-dimensional, é fácil deduzir que todo d-elemento de $\mathcal{C}^{\Delta}$ tem pelo menos um vértice que pertence a um j-elemento de $\mathcal{C}$, para cada j em $\{0, 1, \ldots, d\}$. Consequentemente, todo k-elemento de $\mathcal{C}^{\Delta}$ possui $k+1$ vértices distintos entre si. Em outras palavras a subdivisão baricêntrica de um mapa é uma triangulação própria.

Por outro lado, a subdivisão baricêntrica pode não produzir uma triangulação suficientemente densa para a construção de um modelo geométrico efetivo. Subdivisões adicionais podem ser necessárias para esse fim, como veremos nas seções 4.4 e 4.5.

Além disso, a aplicação iterada da subdivisão baricêntrica resulta em modelos com vértices de graus excessivamente desiguais. A figura 4.7 ilustra este problema, na subdivisão baricêntrica do mapa `3-torus[`n, n, n`]`.

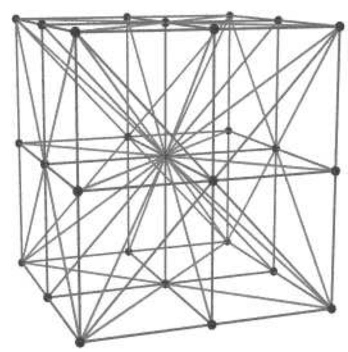


Figura 4.7: A subdivisão baricêntrica de uma célula do mapa `3-torus[`n, n, n`]`.

Observe-se que o novo vértice acrescentado no interior de cada célula, pela primeira subdivisão baricêntrica, possui grau 26, enquanto que os vértices acrescentados no interior das faces e arestas possuem grau 10 e os vértices originais do mapa passam a ter grau 26. Uma segunda etapa de subdivisão baricêntrica produzirá vértices com grau variando entre 8 e 146. Esta desigualdade de graus gera problemas na obtenção de modelos geométricos adequados, em particular modelos em que as arestas têm comprimentos aproximadamente iguais (vide capítulo 6).

4.4 Refinamento seletivo

A subdivisão baricêntrica torna-se necessária quando tratamos mapas tridimensionais que não são triangulações. Como vimos, uma única aplicação da subdivisão baricêntrica é suficiente para obter uma triangulação topológica própria.

A solução para os inconvenientes da subdivisão baricêntrica é utilizar no máximo um nível da mesma seguido por outros esquemas de subdivisão — de preferência, aplicando-a apenas aos simplexos em que ela é necessária.

4.4.1 Refinamento minimal

As figuras 4.8, 4.9 e 4.10 ilustram esquemas de refinamento seletivo minimal (com número mínimo de partes) para células, faces e arestas, respectivamente, de uma triangulação tridimensional. É importante notar que, para $k = 1$ ou $k = 2$, a subdivisão de todo k-elemento e implica na subdivisão dos j-elementos incidentes a e.

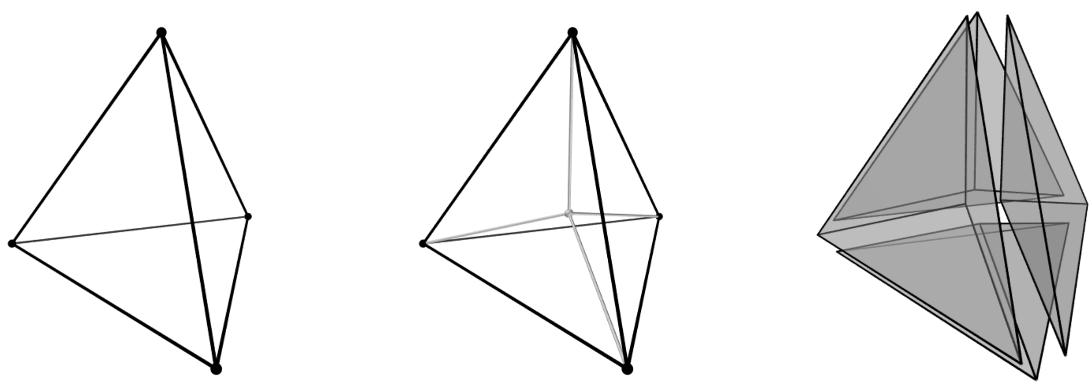


Figura 4.8: Subdivisão seletiva de um tetraedro.

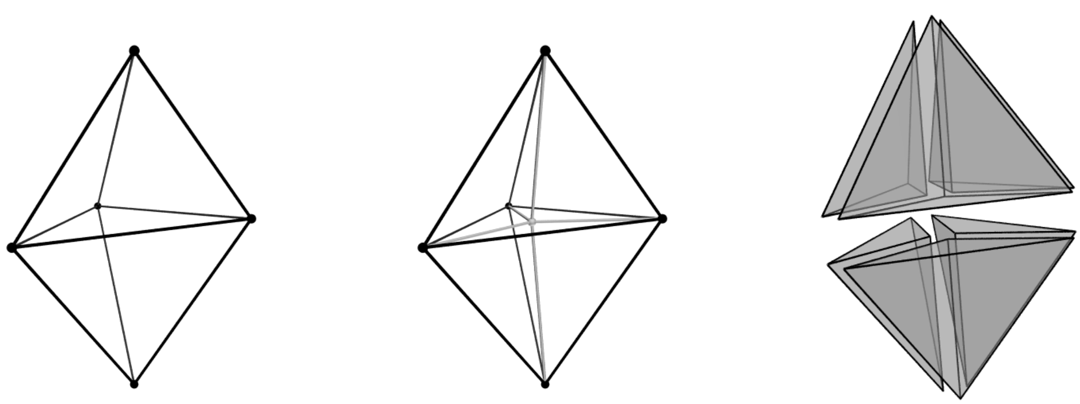


Figura 4.9: Subdivisão seletiva de uma face e dos tetraedros incidentes a esta.

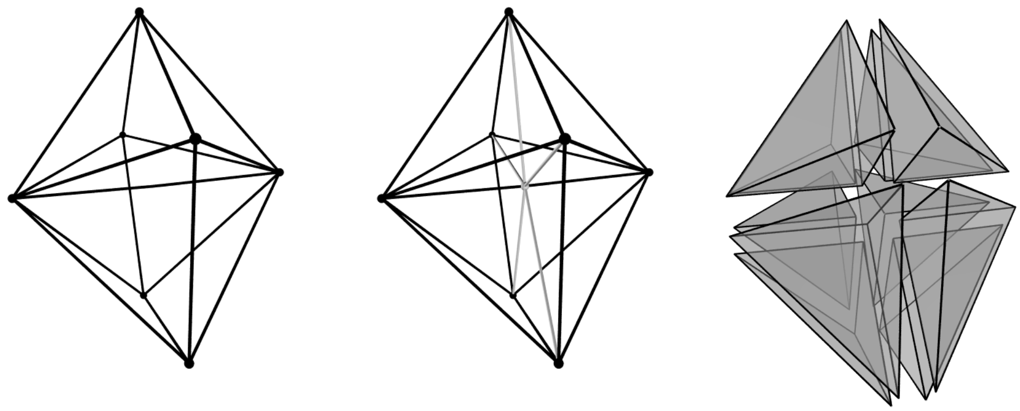


Figura 4.10: Subdivisão seletiva de uma aresta e do anel de tetraedros incidentes na mesma.

4.5 Refinamento uniforme

O esquema de *refinamento uniforme de ordem k* é baseado na subdivisão de um tetraedro geométrico em $\frac{1}{3}(k^3 + 2k)$ tetraedros e $\frac{1}{6}(k^3 - k)$ octaedros, para $k \geq 1$, por $k - 1$ planos equidistantes paralelos a cada face; sendo que cada octaedro é então subdividido em oito tetraedros, formando um `8-vstar` (vide seção 3.2.4). O número total de tetraedros na malha refinada é portanto $T(k) = \frac{5}{3}k^3 - \frac{2}{3}k$. Cada aresta e do tetraedro é dividida em k segmentos, e cada face f é dividida em k^2 faces triangulares.

Esta solução é análoga ao esquema de refinamento de triângulos utilizado por Rosi e Stolfi no caso bidimensional [23, 31]. A figura 4.11 mostra o refinamento de um tetraedro numa malha de tetraedros de ordem $k = 2$, dando origem a 12 tetraedros menores. A figura 4.12 corresponde à ordem $k = 4$, dando origem a 104 tetraedros menores.

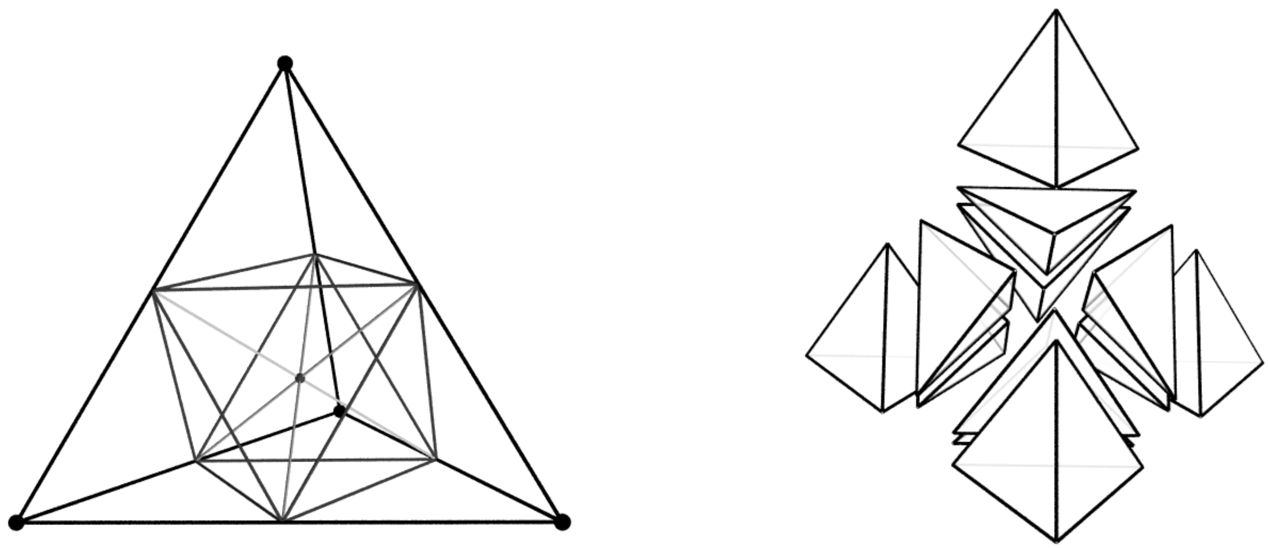


Figura 4.11: Refinamento uniforme de tetraedros de ordem 2.

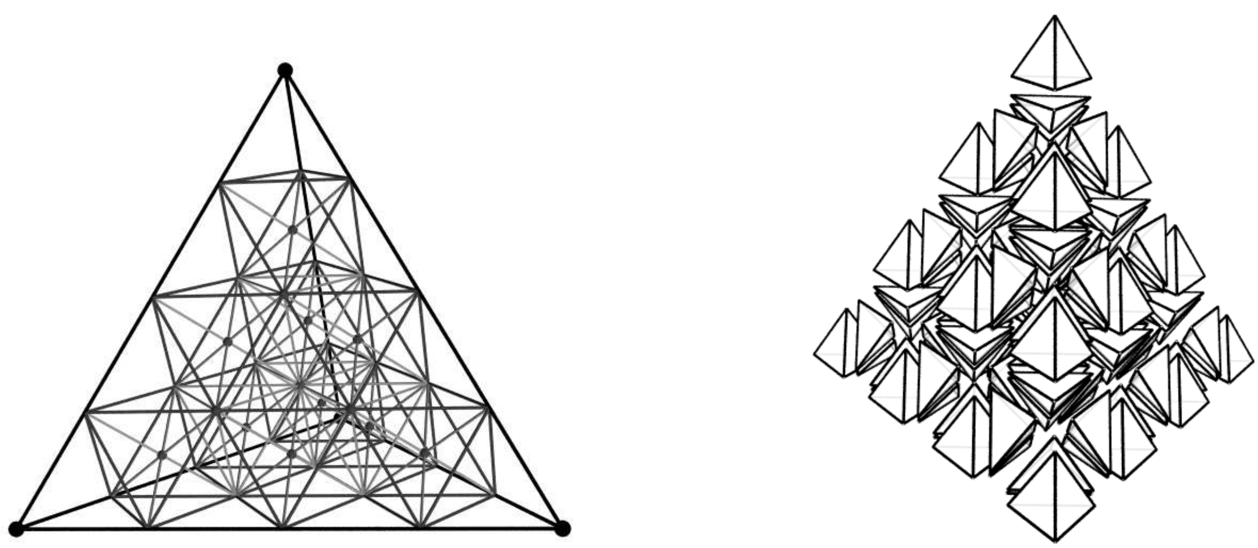
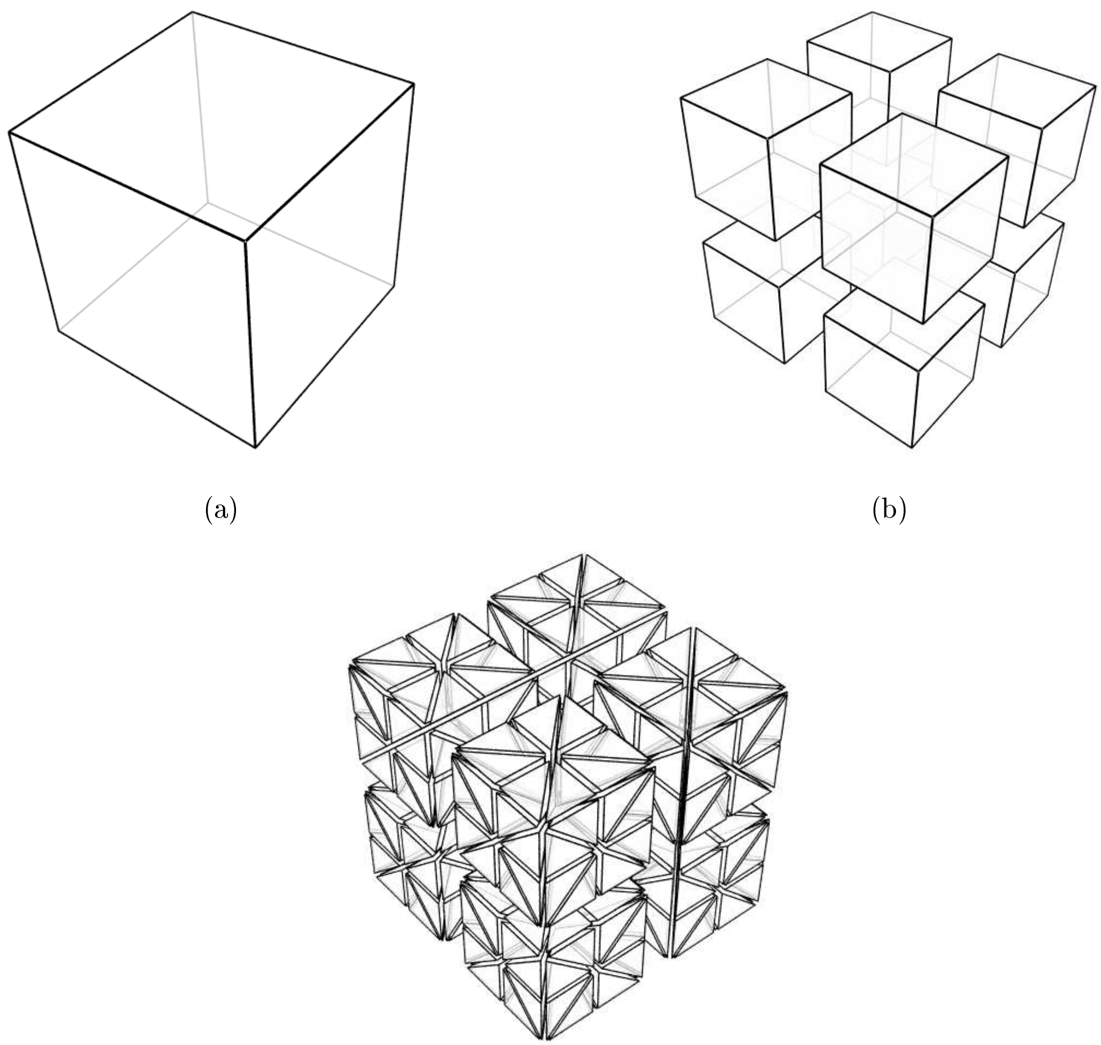


Figura 4.12: Refinamento uniforme de tetraedros de ordem 4.

4.6 Outros esquemas de refinamento

Para o caso de mapas cujas células são cubos topológicos, outra abordagem é subdividir cada cubo numa grade de k^3 cubos menores e aplicar subdivisão baricêntrica a estes últimos. A figura 4.13 ilustra este refinamento.



(a) (b)

(c)

Figura 4.13: Outro esquema de refinamento: célula cúbica original (a), refinamento uniforme de cubos de ordem 2 (b) e subdivisão baricêntrica de cada célula cúbica (c).

Capítulo 5

Representação geométrica dos mapas

Uma vez construída uma triangulação adequada do mapa d-dimensional original $\mathcal{C}$, o próximo passo é construir uma representação geométrica da mesma. Neste trabalho, utilizamos uma representação simples em que cada tetraedro é representado por um tetraedro geométrico retilíneo (contido num sub-espaço afim tridimensional do $\mathbb{R}^m$).

Outra possibilidade seria modelar cada tetraedro por um elemento finito descrito parametricamente por polinômios cúbicos — o equivalente tridimensional de um retalho triangular de Bézier [8]. Esta alternativa permitiria obter modelos mais suaves, com continuidade de primeira ordem — às custas de um aumento no número de variáveis por tetraedro e da introdução de restrições entre essas variáveis, que por sua vez exigiriam algoritmos de otimização mais elaborados.

5.1 Configurações

A partir de uma triangulação topológica $\mathcal{T}$ do mapa d-dimensional $\mathcal{C}$, podemos construir uma *representação simplicial* de $\mathcal{C}$ em algum espaço $\mathbb{R}^m$, $m \geq d$, atribuindo a todo simplexo topológico de $\mathcal{T}$ um simplexo geométrico em $\mathbb{R}^m$, preservando as relações de incidência e adjacência entre elementos.

Uma representação simplicial é completamente determinada pelas coordenadas cartesianas dos vértices de $\mathcal{T}$. Portanto, definimos uma *configuração geométrica* de uma triangulação $\mathcal{T}$ como uma função $\phi\colon \mathbf{V}\mathcal{T} \to \mathbb{R}^m$. Adicionalmente, para cada elemento c de $\mathcal{T}$ com vértices $\{v_0, \ldots, v_k\}$, denotamos por $\phi(c)$ a envoltória convexa positiva do conjunto $\{\phi(v_0), \ldots, \phi(v_k)\}$. Também denotamos por $\phi(\mathcal{T})$ a coleção $\{\phi(c) : c \in \mathcal{T}\}$.

Supondo $m \geq d$, se $\mathcal{T}$ é uma triangulação d-dimensional própria e a configuração ϕ está em posição geral, então $\phi(c)$ será um simplexo geométrico homomeorfo a c, para todo $c \in \mathcal{T}$. Mais ainda, se os elementos c_1, c_2 de $\mathcal{T}$ são incidentes entre si, o mesmo será válido para os correspondentes simplexos $\phi(c_1)$ e $\phi(c_2)$. Portanto, exceto por eventuais intersecções entre elementos, a coleção $\phi(\mathcal{T})$ será uma representação simplicial de $\mathcal{C}$, como definido acima.

5.2 Degenerações geométricas

Obviamente, uma representação simplicial $\phi(\mathcal{T})$ será útil para fins de visualização somente se elementos distintos de $\mathcal{T}$ forem mapeados para simplexos geométricos distintos. Porém, se $\mathcal{T}$ contém dois j-elementos distintos c_1, c_2 que compartilham os mesmos $j+1$ vértices, os simplexos $\phi(c_1)$ e $\phi(c_2)$ serão sempre coincidentes, para qualquer configuração ϕ. Um par de elementos nessas condições constitui uma *degeneração geométrica* da triangulação $\mathcal{T}$.

Verifica-se que a subdivisão baricêntrica $\mathcal{C}^{\Delta}$ de um mapa $\mathcal{C}$ pode ainda conter degenerações geométricas. Por exemplo, o mapa `monoplex` possui uma única célula que é um simplexo topológico degenerado. A subdivisão baricêntrica do mesmo possui 24 tetraedros; todos esses tetraedros são próprios. Porém, eles formam 12 pares geometricamente degenerados. Veja a figura 5.1.

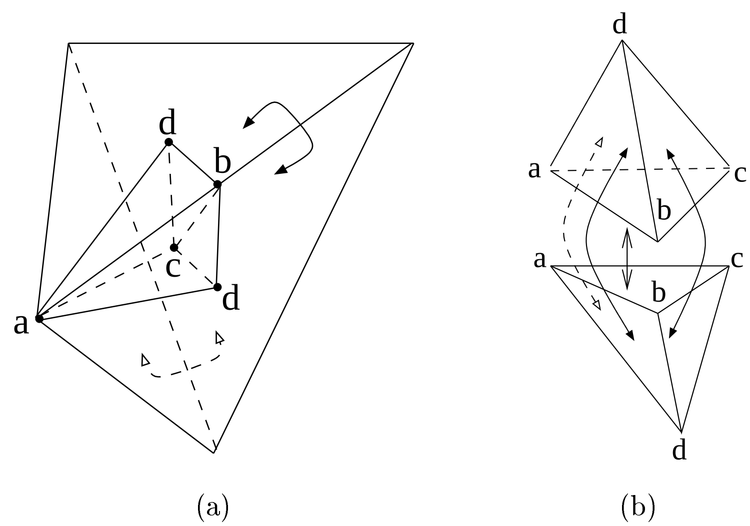


Figura 5.1: Subdivisão baricêntrica do mapa `monoplex` (parcialmente colado) (a) e pares de tetraedros geometricamente degenerados (b).

A fim de eliminar as degenerações geométricas, pode ser necessário subdividir ainda mais os simplexos de $\mathcal{C}^{\Delta}$. O seguinte teorema mostra uma maneira de obter esse resultado:

Teorema 5.1 *Seja $\mathcal{C}$ um mapa 3D arbitrário, e $\mathcal{C}^{\Delta\Delta}$ o resultado de aplicar a subdivisão baricêntrica duas vezes a $\mathcal{C}$. Quaisquer dois j-elementos de $\mathcal{C}^{\Delta\Delta}$ tem pelo menos um vértice não em comum.*

Prova. Sejam c_1 e c_2 dois j-elementos de $\mathcal{C}^{\Delta\Delta}$. Existem dois casos que precisamos analisar. Em primeiro lugar, suponha que c_1 e c_2 são j-elementos pertencentes a dois k-elementos diferentes e_1, e_2 de $\mathcal{C}^{\Delta}$ (para qualquer $k \geq j$). Pela definição de subdivisão baricêntrica (seção 4.2), c_1 tem pelo menos um vértice v_1 no interior de e_1 e similarmente c_2 tem um vértice v_2 no interior de e_2. Portanto $v_1 \neq v_2$ e o teorema está provado.

Agora suponha que c_1 e c_2 pertencem ao mesmo k-elemento e de $\mathcal{C}^{\Delta}$. A demonstração é feita por indução em j. Para $j = 0$, o teorema é trivial; portanto consideramos $j \geq 1$. Uma vez que e é um simplexo topológico próprio, ao construir a segunda subdivisão baricêntrica nós podemos escolher o mapa $\psi\colon \kappa\mathbb{B}^k \to \kappa e$ da definição (seção 4.2), de maneira a ser um homeomorfismo. Pela definição, c_i é a imagem sob ψ de um conjunto t_i que é a união dos raios conectando o centro O de $\mathbb{B}^k$ a pontos de um conjunto t'_i na fronteira de $\mathbb{B}^k$. Por sua vez, t'_i é a imagem inversa de um $(j-1)$-elemento de $\mathcal{C}^{\Delta\Delta}$ pertencente à fronteira subdividida de e. Por hipótese de indução, quaisquer dois $(j-1)$-elementos de $\mathcal{C}^{\Delta\Delta}$ (em particular t'_1 e t'_2) possuem pelo menos um vértice não em comum. Uma vez que ψ é um homeomorfismo, segue-se que c_1 e c_2 possuem também pelo menos um vértice não em comum. □

Como consequência do teorema 5.1, quando $m > d$, qualquer função injetora ϕ de $\mathbf{V}\mathcal{T}$ para $\mathbb{R}^m$ possui a propriedade que $\phi(c) \neq \phi(e)$ sempre que $c \neq e$.

O teorema 5.1 demonstra que é possível eliminar as degenerações geométricas por uma subdivisão adequada dos tetraedros de $\mathcal{T}$. Entretanto, como já observamos na seção 4.3, a subdivisão baricêntrica recursiva produz triangulações muito irregulares e com um número elevado de tetraedros. Na prática, é preferível aplicar subdivisão seletiva, como descrito na seção 4.4, apenas onde necessário. Mais especificamente, subdividimos os elementos envolvidos em denegerações geométricas, começando com os elementos de maior ordem: primeiro pares de tetraedros (figura 4.8), a seguir faces (figura 4.9) e por último arestas (figura 4.10). Esta ordem garante que não serão criadas novas degenerações geométricas durante a subdivisão seletiva. Em cada caso, aplicamos o esquema de subdivisão minimal apropriado (seção 4.4) aos dois elementos envolvidos.

5.3 Mergulho de uma triangulação

A ausência de degenerações geométricas é uma condição necessária mas não suficiente. Uma boa representação geométrica de $\mathcal{C}$ deve também estar livre de auto-interseções. Em termos de nosso modelo, é desejável que os simplexos $\phi(c)$, para todo $c \in \mathcal{T}$, sejam disjuntos dois a dois. Uma realização geométrica com esta propriedade é chamada de *mergulho* *simplicial* de $\mathcal{T}$. O teorema a seguir garante a existência de um mergulho simplicial para qualquer triangulação d-dimensional, num espaço euclidiano de dimensão suficientemente alta.

Teorema 5.2 (Whitney) *Se $\mathcal{T}$ é uma triangulação d-dimensional, podemos atribuir a cada vértice v de $\mathcal{T}$ um ponto $\phi(v)$ no espaço $\mathbb{R}^{2d+1}$, de tal maneira que todos os elementos de $\phi(\mathcal{T})$ são disjuntos dois a dois.*

Demonstrações deste teorema 5.2 podem ser encontradas no livro de Stillwell [34] e no livro de Giblin [9]. Em particular, o teorema 5.2 garante que a triangulação de todo mapa tridimensional $\mathcal{C}$ é mergulhável em $\mathbb{R}^7$. Este resultado é somente um limite superior e levanta a seguinte questão: existem mergulhos simpliciais em espaços de dimensão menor que $2d+1$? Na verdade, o próprio Whitney demonstrou posteriormente que toda d-variedade diferenciável (espaço mais restrito que a variedade topológica) tem um mergulho diferenciável em $\mathbb{R}^{2d}$. Isto explicaria porque a garrafa de Klein pode ser mergulhada no $\mathbb{R}^4$, enquanto que o teorema 5.2 garante apenas sua realização em $\mathbb{R}^5$. Gitler [10] conjetura que toda variedade diferenciável d-dimensional tem um mergulho diferenciável no $\mathbb{R}^{2d-\alpha(d)+1}$, onde $\alpha(d)$ é a soma dos dígitos binários do número d.

Além disso, se $d=2$, podemos sempre obter mergulhos simpliciais em $\mathbb{R}^3$ se a triangulação é orientável e em $\mathbb{R}^4$, caso contrário. Por analogia, podemos esperar que toda triangulação tridimensional seja mergulhável no $\mathbb{R}^4$ se orientável e no $\mathbb{R}^5$ caso contrário. Nós ainda não conseguimos determinar se estas conjeturas são válidas.

Em todo caso, devemos ter em mente que uma realização no $\mathbb{R}^4$ com auto-interseções pode ser mais fácil de visualizar do que uma realização no espaço $\mathbb{R}^7$ sem auto-interseções. Por essa razão, em todos os nossos testes, nós utilizamos $\mathbb{R}^4$ como espaço de modelagem.

Capítulo 6

Comparando realizações

Neste capítulo abordamos o problema de escolher, dentre todas as possíveis configurações $\phi(\mathcal{T})$ de um mapa $\mathcal{C}$, uma configuração que seja suficientemente "bonita" ou "eficaz" — isto é, que facilite ao máximo a compreensão da topologia do mapa $\mathcal{C}$ através de inspeção visual do modelo.

6.1 Funções de energia

A fim de quantificar a noção de "beleza", emprestamos da literatura sobre desenho de grafos o conceito de *função de energia* — uma medida numérica $\mathcal{E}(\phi)$ de "feiura" de uma configuração particular ϕ. Uma função de energia é portanto um mapeamento de $(\mathbb{R}^d)^n$ para $\mathbb{R}$, que tem idealmente seu valor mínimo em configurações ϕ que minimizam determinados defeitos visuais. Em especial, as funções de energia que utilizamos tentam capturar os seguintes objetivos estéticos:

(A) uniformizar o tamanho e forma dos elementos do mapa;

(B) evitar intersecções entre elementos do mapa;

(C) minimizar a curvatura do modelo.

As experiências de Rosi [30] na visualização de mapas 2D sugerem que a exigência (A) deve ser aplicada à triangulação $\mathcal{T}$, em vez do mapa original $\mathcal{C}$. A justificativa é que as partes do mapa $\mathcal{C}$ que precisam ser subdivididas mais finamente são em geral as partes onde a topologia é mais complicada. Portanto, ao representar os elementos de $\mathcal{T}$ com tamanho uniforme, estaremos representando essas partes de $\mathcal{C}$ em escala maior, permitindo assim visualizar sua topologia com clareza.

6.1.1 Funções de energia mistas

Em geral, as funções energia mais simples capturam apenas um critério estético específico, como suavidade da variedade ou uniformidade do tamanho dos elementos. Portanto, uma configuração que é ótima para uma função energia usualmente deixa de atingir outros critérios estéticos visuais importantes. Assim, as funções energia que utilizamos são em geral combinações convexas de energias simples:

$$\mathcal{E}(\phi) = \alpha_1 \, \mathcal{E}_1(\phi) + \alpha_2 \, \mathcal{E}_2(\phi) + \cdots$$

Os pesos α_i determinam a importância relativa dos critérios representados pelas funções $\mathcal{E}_i$.

6.2 Energia elástica de arestas

O critério (A) de uniformidade no tamanho dos elementos é capturado pela *energia elástica de arestas* $\mathcal{E}_{edge}(\phi)$, em que cada aresta é modelada por uma mola retilínea:

$$\mathcal{E}_{edge}(\phi) = K_{edge} \sum_{e \in \mathbf{E}\mathcal{T}} S_e \left[\left(\frac{l_e}{L_e} \right)^2 + \left(\frac{L_e}{l_e} \right)^2 - 2 \right] . \tag{6.1}$$

Nesta fórmula, l_e é o comprimento corrente da aresta $e = (u, v)$ na configuração ϕ ou seja $|\phi(u) - \phi(v)|$; L_e é o comprimento de repouso da mola (isto é, o comprimento ideal da aresta e); S_e é a rigidez da mola; e K_{edge} é um fator de normalização (vide seção 6.6). Esta energia foi utilizada por Hoppe et al. [17] na otimização de malhas triangulares. Note-se que o custo computacional de avaliar $\mathcal{E}_{edge}$ é linear no tamanho de $\mathcal{T}$.

Uma primeira justificativa para a fórmula (6.1) é que cada termo tem valor mínimo (zero) quando $l_e = L_e$. Mais geralmente, cada termo é aproximadamente $S_e(l_e/L_e)^2$ para $l_e \gg L_e$, $S_e(L_e/l_e)^2$ para $l_e \ll L_e$ e $2S_e((l_e - L_e)/L_e)^2$ quando $l_e \approx L_e$. Veja a figura 6.1. A menos de um fator constante, esta última fórmula é a energia elástica de uma mola física de comprimento de repouso L_e e comprimento atual l_e.

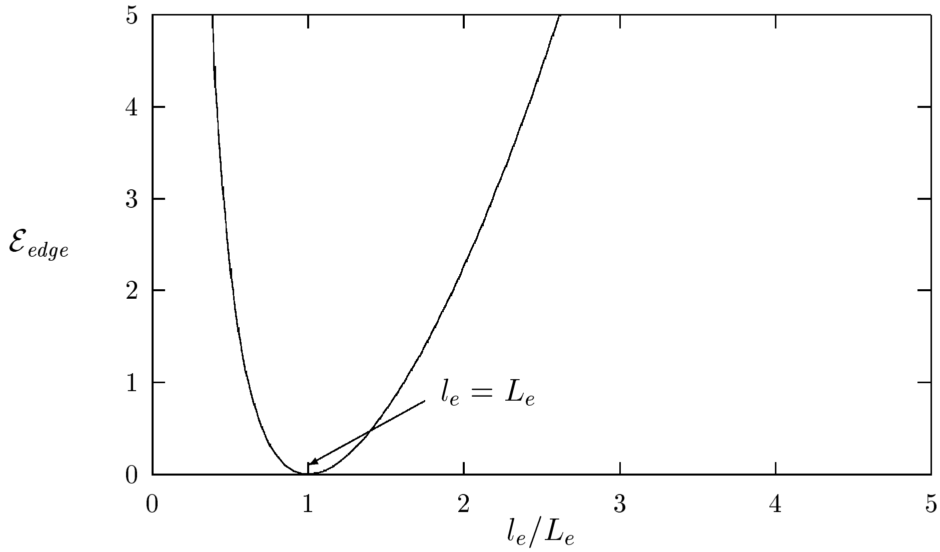
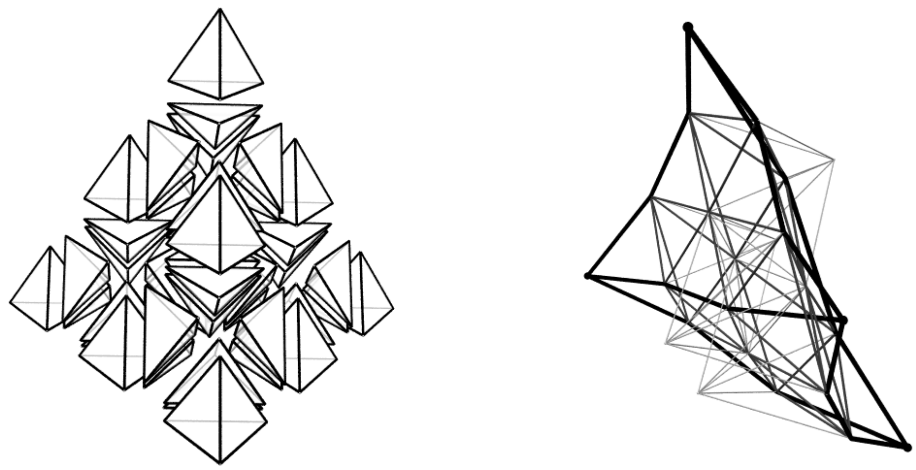


Figura 6.1: Comportamento da energia de elástica de arestas.

Portanto, minimizar $\mathcal{E}_{edge}$ tende a fazer com que os comprimentos das arestas sejam próximos do seu comprimento ideal L_e. A uniformidade nos comprimentos das arestas implica também outras características visuais desejáveis, tais como áreas uniformes para os triângulos, volumes uniformes para os tetraedros e ângulos diedrais uniformes entre os triângulos incidentes sobre uma mesma aresta.

6.2.1 Triangulações não rígidas

A energia elástica de arestas $\mathcal{E}_{edge}$ mantêm vértices adjacentes afastados uns dos outros, mas não impede a configuração de dobrar-se sobre si mesma, criando auto-intersecções não essenciais. Por exemplo, a figura 6.2(b) mostra uma configuração do mapa `4-face` após subdivisão uniforme de ordem 3 (vide seção 9.1.2) que é mínima para a energia $\mathcal{E}_{edge}$.

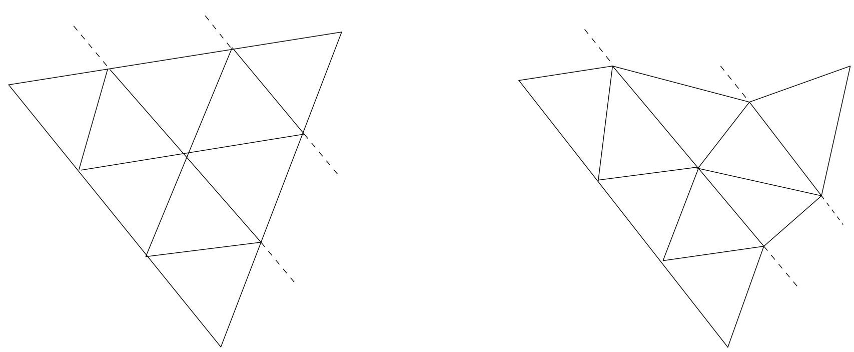


(a) (b)

Figura 6.2: Uma triangulação uniforme de ordem três do mapa `4-face` (a), e uma configuração da mesma que é um mínimo local para a energia elástica de arestas $\mathcal{E}_{edge}$ (b).

Observe-se que a triangulação em questão poderia ser realizada no $\mathbb{R}^3$; e essa normalmente seria a realização desejada. Entretanto, na configuração da figura 6.2(b) os vértices não estão todos num mesmo hiperplano de $\mathbb{R}^4$.

Neste caso, o problema ocorre em parte porque a triangulação, mesmo quando todas as arestas tem comprimento fixo, não é rígida — isto é, pode ser dobrada no $\mathbb{R}^4$ ao longo de certos planos, sem alterar os comprimentos das arestas. A figura 6.3 ilustra a versão bidimensional deste problema — a triangulação 6.3(a) pode ser dobrada no $\mathbb{R}^3$ ao longo das linhas pontilhadas, mantendo todas as arestas com comprimentos iguais.



(a) (b)

Figura 6.3: Uma triangulação bidimensional que não é rígida.

Mesmo quando a triangulação é rigída, em geral há muitas configurações indesejáveis que são mínimas para $\mathcal{E}_{edge}$. Por exemplo, se numa configuração ótima ϕ os vizinhos de um vértice v estão todos aproximadamente no mesmo hiperplano $\mathcal{H}$ de $\mathbb{R}^m$, então a configuração ϕ' que é obtida a partir de ϕ por reflexão do vértice v através de $\mathcal{H}$ também será ótima para $\mathcal{E}_{edge}$.

Por estes motivos, a otimização de $\mathcal{E}_{edge}$, apenas sem outras energias, geralmente produz modelos altamente dobrados. Na verdade, a configuração da figura 6.2(b) é atípica; a grande maioria das configurações mínimas são altamente contorcidas, como mostrado na figura 6.4.

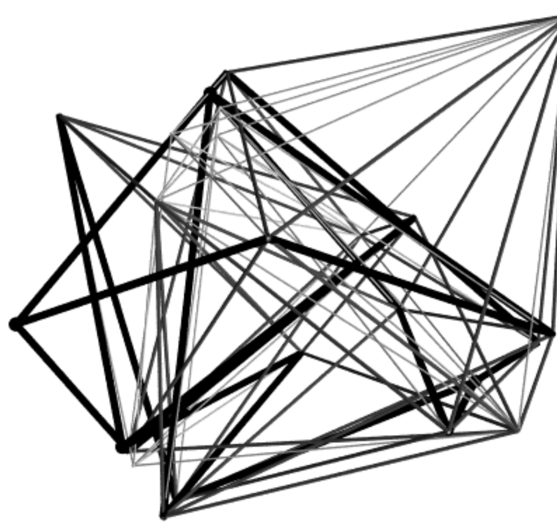


Figura 6.4: Projeção 3D de uma configuração mínima típica para a energia $\mathcal{E}_{edge}$. A triangulação é a mesma da figura 6.2.

6.2.2 Distorção inevitável da variedade

Na verdade, em certas triangulações, a minimização da energia $\mathcal{E}_{edge}$ (com parâmetros S_e e L_e uniformes) *necessariamente* resulta num modelo geométrico bastante retorcido. Este problema ocorre especialmente em triangulações cujos vértices tem graus muito grandes ou muitos pequenos.

Para melhor apreciar o problema em questão, considere o caso de uma triangulação bidimensional $\mathcal{T}$ com borda, consistindo de n triângulos cercando um vértice comum como as ilustradas na figura 6.5. Se quisermos que todas as arestas de $\mathcal{T}$ tenham o mesmo comprimento, então, na realização do mapa da figura 6.5(a), teremos que colocar o vértice v para fora do plano dos seus quatro vizinhos como ilustrado em 6.5(a$'$). Por outro lado, no caso do mapa da figura 6.5(c), os vizinhos do vértice v não poderão ficar no mesmo plano. Apenas vértices de grau seis, como o caso do mapa da figura 6.5(b), têm chance de permanecer no mesmo plano que todos os seus vizinhos.

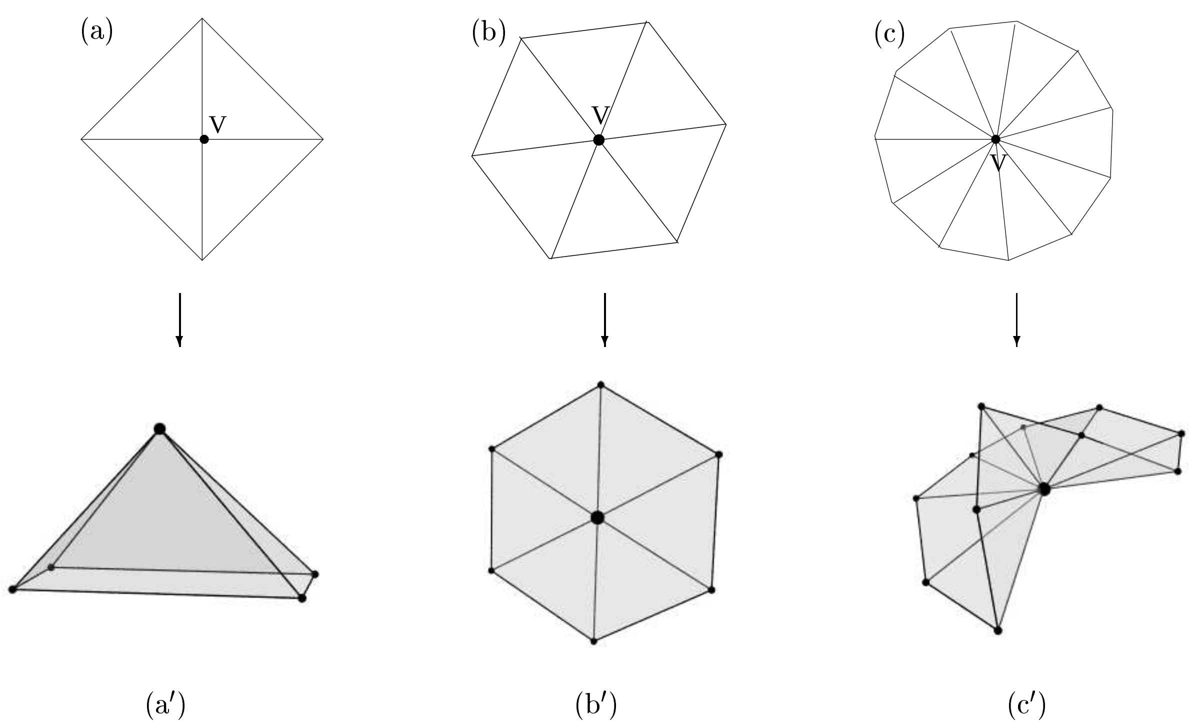


Figura 6.5: Alguns mapas bidimensionais simples (no alto) e configurações mínimas dos mesmos para a energia $\mathcal{E}_{edge}$ (em baixo).

A figura 6.6 mostra exemplos tridimensionais análogos aos casos 6.5(a) e 6.5(c). É uma infelicidade da geometria tridimensional euclidiana que não existe um análogo 3D da configuração 6.5(b) — ou seja, não é possível colar n tetraedros *regulares* de forma a envolver completamente um vértice comum, ou uma aresta comum.

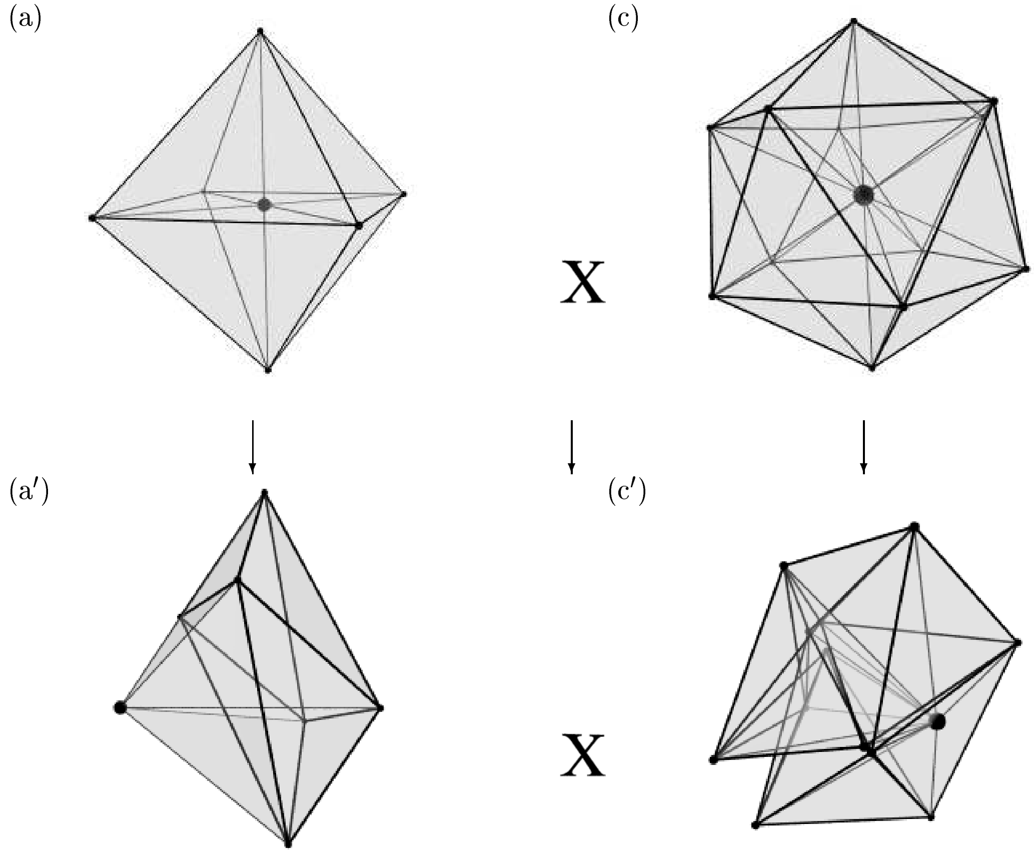


Figura 6.6: Os mapas tridimensionais `8-vstar` e `20-vstar` (no alto) e configurações mínimas dos mesmos para a energia $\mathcal{E}_{edge}$ (em baixo).

Note-se que o problema da figura 6.6(a) ocorre, em particular, na figura 6.2(b) — os vértices de grau seis (com estrela octaedral) estão fora do hiperplano de seus vizinhos. Na verdade, exceto por esses vértices, a triangulação da figura 6.2 pode ser realizada no $\mathbb{R}^3$ com todas as arestas iguais.

O problema da distorção inevitável da variedade poderia ser solucionado com uma escolha mais inteligente dos parâmetros L_e e S_e; porém não conhecemos nenhum algoritmo para esse fim. Soluções parciais para esse problema, baseados na combinação de $\mathcal{E}_{edge}$ com outras funções energia, serão discutidas nas próximas seções.

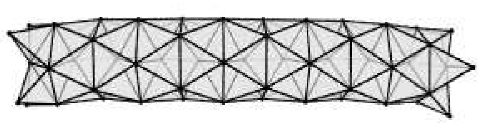
6.3 Energia de molas

Uma maneira de tentar evitar superposição de tetraedros na projeção, e auto-intersecções em geral — critério estético (B) — é usar uma versão mais abrangente da energia de elasticidade de arestas, a *energia de molas* $\mathcal{E}_{spring}$, originalmente proposta por Kamada e Kawai para o desenho de grafos no plano [18]. A idéia é inserir uma mola de comprimento adequado entre cada par de vértices u, v, adjacentes ou não:

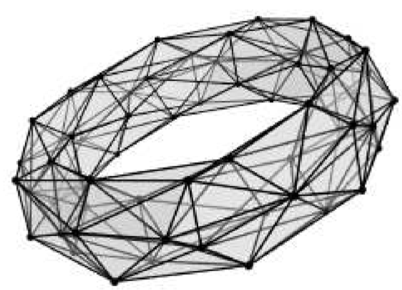
$$\mathcal{E}_{spring}(\phi) = K_{spring} \sum_{\substack{u,v \in \mathbf{V}\mathcal{T} \\ u \neq v}} S_{uv} \left[\left(\frac{d_{uv}}{D_{uv}} \right)^2 + \left(\frac{D_{uv}}{d_{uv}} \right)^2 - 2 \right], \tag{6.2}$$

Nesta fórmula, $d_{uv} = |\phi(u) - \phi(v)|$ é distância corrente entre os vértices u and v; D_{uv} é a distância ideal entre u e v; S_{uv} é a rigidez da mola; e K_{spring} é um fator de normalização (vide seção 6.6). Kamada e Kawai sugerem utilizar D_{uv} igual à distância topológica entre u e v na triangulação $\mathcal{T}$ (no sentido da teoria dos grafos), e $S_{uv} = 1/D_{uv}^2$. Estes são os valores que utilizamos em nossos testes, exceto que nós dividimos D_{uv} pela distância topológica máxima entre os vértices de $\mathcal{T}$, a fim de manter o tamanho da configuração ótima aproximadamente invariante sob refinamentos topológicos.

A figura 6.7 mostra realizações dos mapas `column[5,10]` e `donut[5,10]`, com 150 tetraedros, que são mínimas para $\mathcal{E}_{spring}$. A figura ilustra como a minimização da energia $\mathcal{E}_{spring}$ tende a distribuir as deformações impostas pela topologia do mapa de maneira mais uniforme possível. Observe-se como os tetraedros da configuração 6.7(b), apesar de deformados em relação à figura 6.7(a) (por causa da topologia da triangulação), ainda têm tamanhos e formas bastante uniformes.



(a)

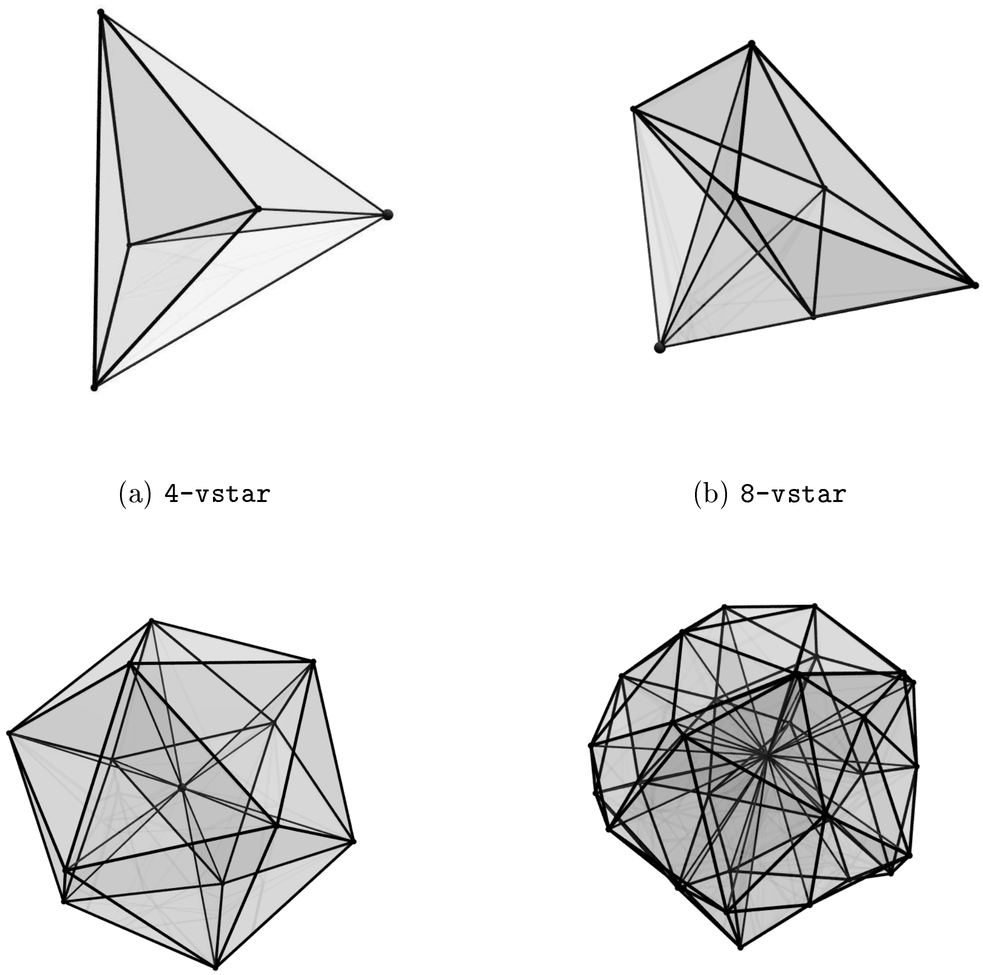


(b)

Figura 6.7: Configurações mínimas para a energia $\mathcal{E}_{spring}$ nos mapas `column[5,10]` (a) e `donut[5,10]` (b).

A substituição de $\mathcal{E}_{edge}$ por $\mathcal{E}_{spring}$ reduz até certo ponto o problema da distorção da variedade descrita na seção 6.2.2. Em particular, a introdução de molas entre os vértices não-adjacentes da figura 6.5(c) ajuda a manter os vizinhos de v no mesmo plano, desde que o grau não seja muito elevado. Entretanto, a energia $\mathcal{E}_{spring}$ não tem efeito sobre o problema de vértices de grau baixo, ilustrado na figura 6.5(a).

A figura 6.8 ilustra realizações geométricas ótimas para a energia $\mathcal{E}_{spring}$ dos mapas estrela de vértices. Observe-se que o modelo mínimo para `20-vstar` é regular e está contido num sub-espaço tridimensional euclidiano do $\mathbb{R}^4$ e portanto pode se projetado no $\mathbb{R}^3$ sem deformações. Em particular, a configuração da figura 6.6(c′) que é ótima para $\mathcal{E}_{edge}$, não é ótima para $\mathcal{E}_{spring}$. Por outro lado, a configuração do mapa `8-vstar` que é mínima para $\mathcal{E}_{spring}$ (veja figura 6.8) ainda tem o vértice central fora do hiperplano dos seus vizinhos, como na figura 6.6(a′); e a configuração ótima da triangulação `60-vstar` continua bastante retorcida.



(a) `4-vstar` (b) `8-vstar`

(c) `20-vstar` (d) `60-vstar`

Figura 6.8: Realizações ótimas de alguns mapas estrela de vértice para a energia $\mathcal{E}_{spring}$. A maior esfera identifica o centro topológico do mapa.

Uma maneira de diminuir a severidade deste problema consiste em multiplicar o comprimento ideal D_{uv} de cada aresta (u, v) pelo fator $\sqrt{N_u/N_v + N_v/N_u}$ onde N_u e N_v são os graus de seus vértices extremos. Denotamos esta versão modificada da energia de molas por $\mathcal{E}_{springvar}$. A figura 6.9 ilustra o efeito da inclusão da energia $\mathcal{E}_{springvar}$ no mapa `60-vstar`. Comparando-se esta figura com a da figura 3.8, pode-se verificar que a configuração ótima é praticamente a representação ideal do mapa.

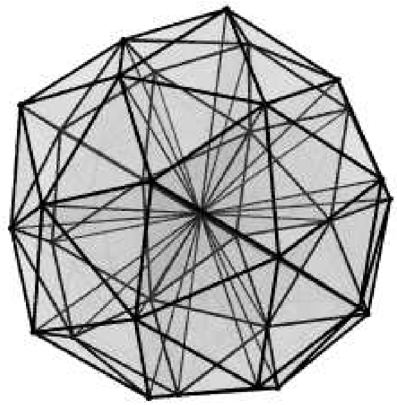


Figura 6.9: Uma configuração do mapa `60-vstar` que é um mínimo local para a energia de molas $\mathcal{E}_{springvar}$.

A figura 6.10 mostra uma configuração mínima da energia $\mathcal{E}_{spring}$ para a mesma triangulação da figura 6.2. Note-se que persiste o problema da distorção da variedade nos vértices octaedrais, como mostrado nas figuras 6.2(b) e 6.8(b); porém a configuração (altamente dobrada) da figura 6.4 já não é mais ótima para a energia $\mathcal{E}_{spring}$.

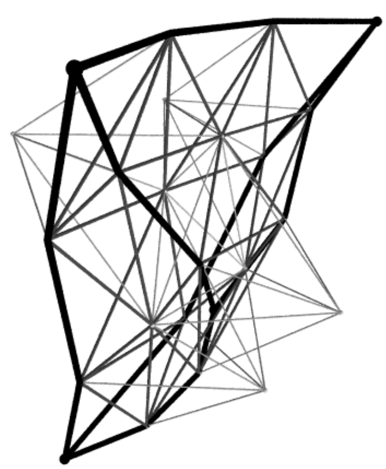


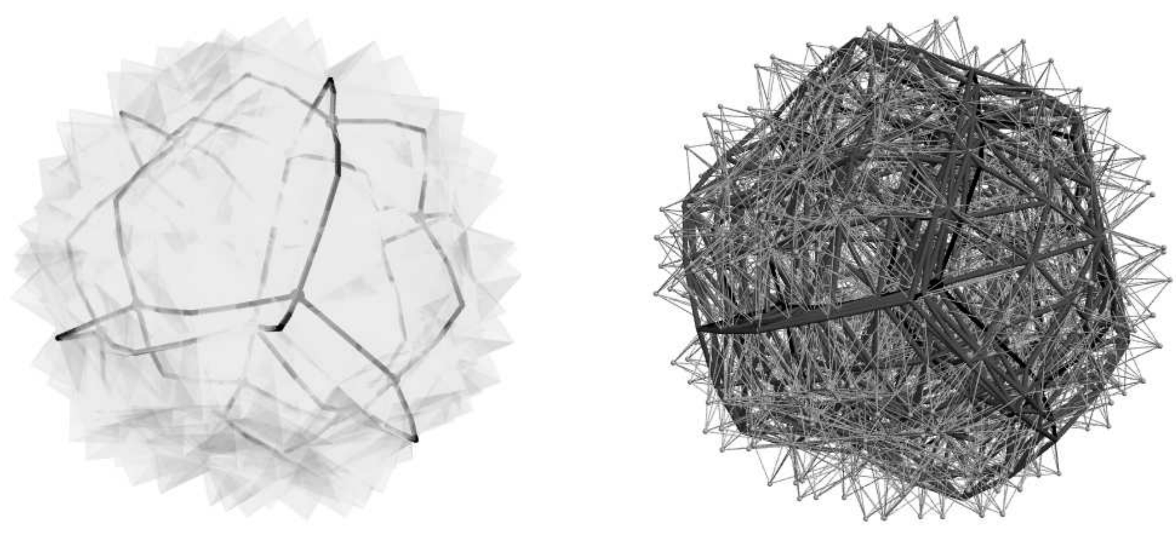
Figura 6.10: Uma configuração que é um mínimo local para a energia de molas $\mathcal{E}_{spring}$. A triangulação é a mesma da figura 6.2.

Note-se que a fórmula $\mathcal{E}_{spring}$ inclui todos os termos da fórmula de $\mathcal{E}_{edge}$. Portanto, minimizar a energia $\mathcal{E}_{spring}$ tende a manter todas as arestas de $\phi(\mathcal{T})$ com comprimento próximo ao seu comprimento ideal D_{uv} (como quando minimizamos $\mathcal{E}_{edge}$); e ao mesmo tempo a evitar intersecções entre partes de $\phi(\mathcal{T})$ que não são adjacentes em $\mathcal{T}$. Em particular, minimizar a energia $\mathcal{E}_{spring}$ também diminui a ocorrência de pares de tetraedros adjacentes sobrepostos. A razão é que ela introduz uma mola repulsiva entre os vértices não compartilhados dos dois tetraedros — o que tende a mantê-los em lados opostos da face comum, numa projeção geral.

Uma desvantagem da energia $\mathcal{E}_{spring}$ é seu alto custo computacional, uma vez que o número de termos cresce quadraticamente com o tamanho de $\mathcal{T}$.

6.3.1 O "efeito abacaxi"

Pelas razões expostas acima, a minimização da energia $\mathcal{E}_{spring}$, por si, geralmente produz uma configuração extremamente distorcida, no sentido que todo vértice de grau baixo geralmente está bem longe do sub-espaço tridimensional médio de seus vizinhos. Veja a figura 6.11.



(a) (b)

Figura 6.11: Uma configuração que minimiza a energia $\mathcal{E}_{spring}$, mostrando o "efeito abacaxi". A triangulação é um refinamento uniforme de ordem 2 da subdivisão baricêntrica do mapa `8-cell`. O modelo (a) mostra unicamente os vértices e arestas originais do mapa $\mathcal{C}$, enquanto o modelo (b) mostra todas as arestas da triangulação $\mathcal{T}$ de $\mathcal{C}$.

Denominamos este problema, que causa sérios problemas na visualização do modelo, de "efeito abacaxi". Ele será tratado na próxima seção.

6.4 Energias de curvatura

A fim de combater o efeito abacaxi, precisamos introduzir outras energias que obriguem a variedade subjacente a $\phi(\mathcal{T})$, a ser, na vizinhança de cada ponto p de $\mathcal{T}$, o mais próxima possível de um sub-espaço tridimensional euclidiano V_p.

Em outras palavras, precisamos minimizar a curvatura tridimensional aparente da variedade $\phi(\mathcal{T})$. Mais do que isso, se a topologia de $\mathcal{C}$ só pode ser realizada com uma variedade curva, então precisamos distribuir a curvatura inevitável de $\phi(\mathcal{T})$ de maneira mais uniforme possível.

6.4.1 Energia de curvatura tridimensional

Uma vez que cada tetraedro de $\phi(\mathcal{T})$ está contido num sub-espaço euclidiano tridimensional de $\mathbb{R}^m$, que tem curvatura 0, a curvatura da variedade $\phi(\mathcal{T})$ está toda concentrada nas faces. Em geral, se t_1, t_2 são dois tetraedros adjacentes, suas imagens $\phi(t_1)$ e $\phi(t_2)$ estão contidas em dois sub-espaços 3D diferentes U_1, U_2 de $\mathbb{R}^m$, cuja intersecção é o plano contendo a face compartilhada f (veja figura 6.12). Sejam r_1, r_2 dois vetores do $\mathbb{R}^4$ perpendiculares à face f, contidos respectivamente em U_1 e U_2, que apontam para o interior de cada tetraedro. A curvatura da variedade 3D na face é nula quando $U_1 = U_2$, isto é, quando $r_1 = -r_2$. Veja figura 6.12.

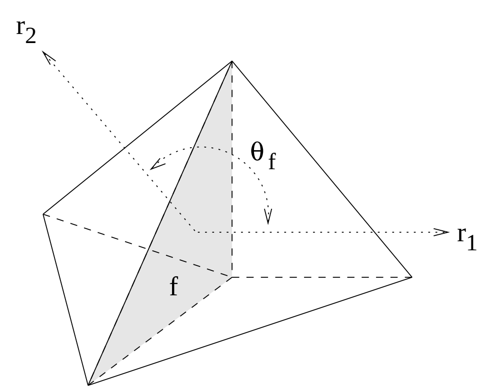


Figura 6.12: Ângulo diedral θ_f entre sub-espaços tridimensionais.

Mais geralmente, a curvatura na face f depende do ângulo θ_f entre r_1 e r_2, que é dado pela fórmula $\cos\theta_f = (r_1 \cdot r_2)/(|r_1|\,|r_2|)$. O objetivo de equalizar a curvatura pode então ser atingido minimizando-se a *energia de curvatura 3D*, que definimos pela fórmula

$$\mathcal{E}_{curv3}(\phi) = K_{curv3} \sum_{f \in \mathbf{F}\mathcal{T}} \left(\frac{1}{1-\cos\theta_f} - \frac{1}{2} \right) . \tag{6.3}$$

A constante K_{curv3} é um fator de normalização (vide seção 6.6). Pode-se verificar que quando θ_f é próximo de π (isto é, quando a curvatura na face f é próxima de zero), temos que $\cos\theta_f \approx -1 + (\theta_f - \pi)^2$ e portanto o termo da equação (6.3) é aproximadamente $\frac{1}{4}(\theta_f - \pi)^2$. A figura 6.13 ilustra o comportamento desse termo, em função do ângulo θ_f. Portanto, ao minimizar $\mathcal{E}_{curv3}$ estamos tentando aproximar todos os ângulos θ_f de π.

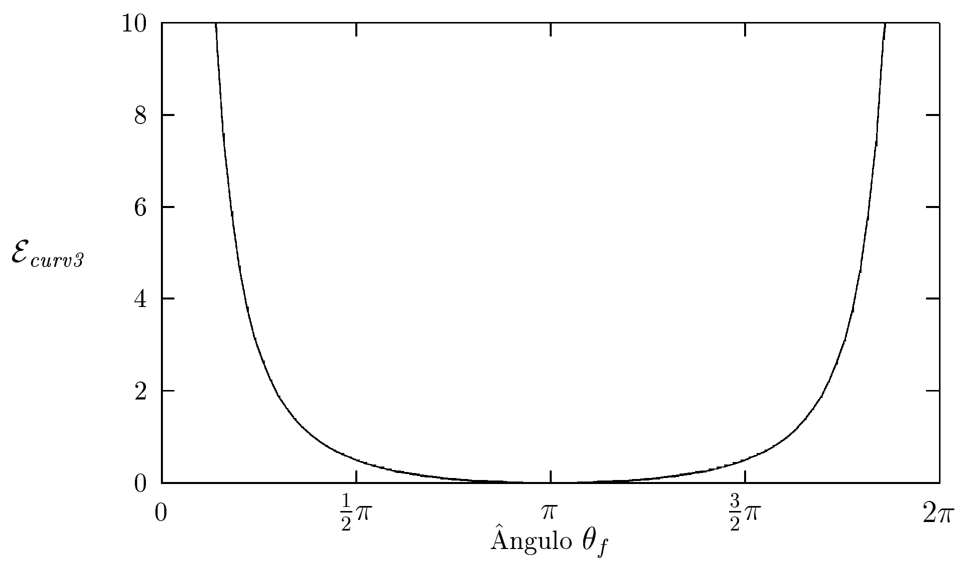
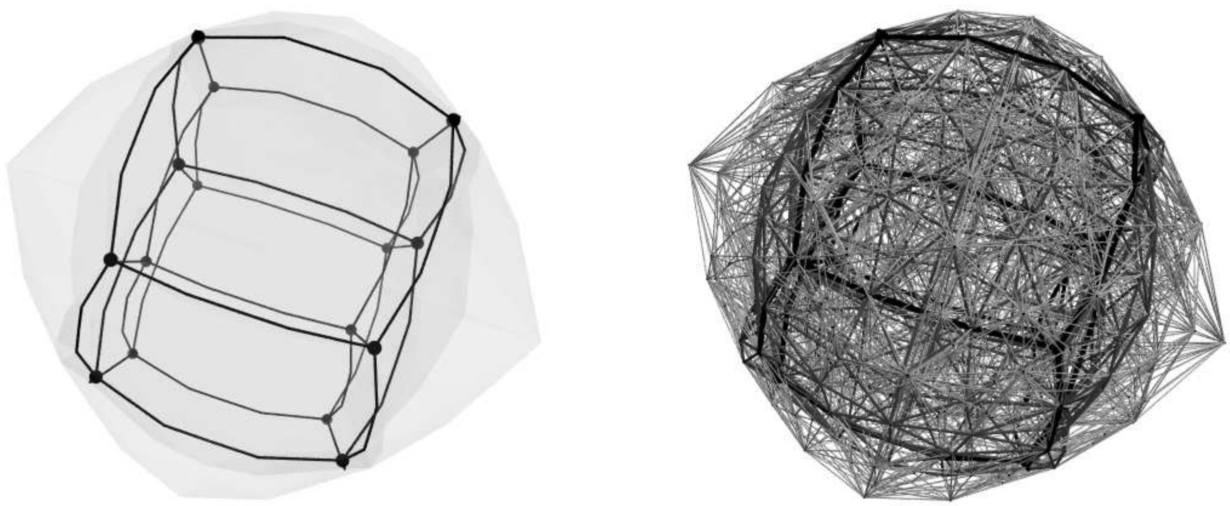


Figura 6.13: Comportamento da energia de curvatura tridimensional para dois tetraedros adjacentes, em termos do ângulo θ_f entre os respectivos subespaços tridimensionais.

A figura 6.14 mostra uma configuração da mesma triangulação da figura 6.11, otimizada para a energia mista $\mathcal{E} = 0.99\ \mathcal{E}_{curv3} + 0.01\ \mathcal{E}_{spring}$. Note-se que a inclusão de $\mathcal{E}_{curv3}$ praticamente elimina o "efeito abacaxi", satisfazendo o critério (C) sem prejuízo de (A) e (B).



(a) (b)

Figura 6.14: Configuração mínima da energia $\mathcal{E} = 0.99\ \mathcal{E}_{curv3} + 0.01\ \mathcal{E}_{spring}$. A triangulação é a mesma da figura 6.11 (um refinamento uniforme de ordem 2 da subdivisão baricêntrica do mapa `8-cell`). O modelo (a) mostra unicamente os vértices e arestas originais do mapa $\mathcal{C}$ e as faces de silhueta (vide seção 8.3.1) enquanto o modelo (b) mostra todas as arestas da triangulação $\mathcal{T}$ de $\mathcal{C}$.

Em particular, se o mapa $\mathcal{C}$ é uma partição de uma variedade que pode ser mergulhada no $\mathbb{R}^3$, (necessariamente orientável e com borda), toda configuração que minimiza a energia $\mathcal{E}_{curv3}$ estará contida num único sub-espaço euclidiano tridimensional de $\mathbb{R}^m$. Por exemplo, a figura 6.15 mostra uma configuração do mesmo mapa da figura 6.2, que é um mínimo para $\mathcal{E}_{curv3}$. Observe que os vértices octaedrais agora estão no mesmo espaço tridimensional que os demais.

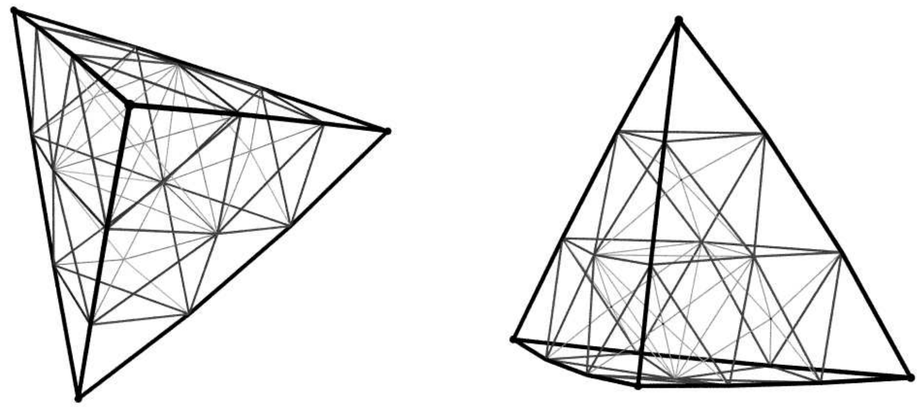


Figura 6.15: Duas projeções de $\mathbb{R}^4$ para $\mathbb{R}^3$ de uma mesma configuração do mapa `4-face` (após subdivisão uniforme de ordem 3), que é um mínimo local para a energia de curvatura $\mathcal{E}_{curv3}$.

Na verdade, a configuração da figura 6.15 é extremamente atípica — pois a energia $\mathcal{E}_{curv3}$, por si só, não impede auto-intersecções nem garante uniformidade de tamanho dos elementos. A grande maioria das configurações da triangulação 6.15 que são ótimas para $\mathcal{E}_{curv3}$ são extremamente irregulares, como mostra a figura 6.16. Portanto, na prática, $\mathcal{E}_{curv3}$ é normalmente usada em combinação com $\mathcal{E}_{spring}$.

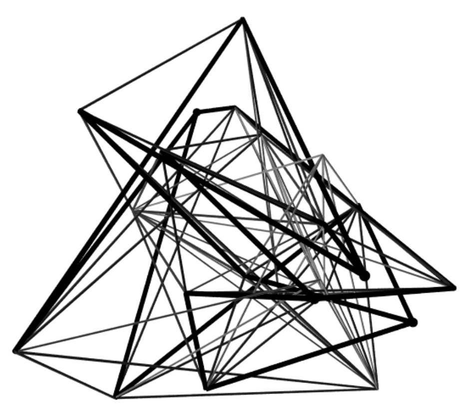


Figura 6.16: Projeção 3D de uma configuração mínima típica para a energia $\mathcal{E}_{curv3}$. A triangulação é a mesma da figura 6.2.

6.4.2 Energia de curvatura bidimensional

A energia de curvatura $\mathcal{E}_{curv3}$ depende apenas da variedade tridimensional subjacente a $\phi(\mathcal{T})$ e não da posição das faces e arestas de $\mathcal{C}$ dentro dessa configuração. Portanto, uma configuração que minimiza $\mathcal{E}_{curv3}$ (possivelmente combinada com $\mathcal{E}_{spring}$) ainda pode ter faces bastantes contorcidas. Isto pode acontecer mesmo em modelos que estão contidos inteiramente em $\mathbb{R}^3$ e portanto tem $\mathcal{E}_{curv3}$ mínima. Veja as figuras 6.16 e 6.17.

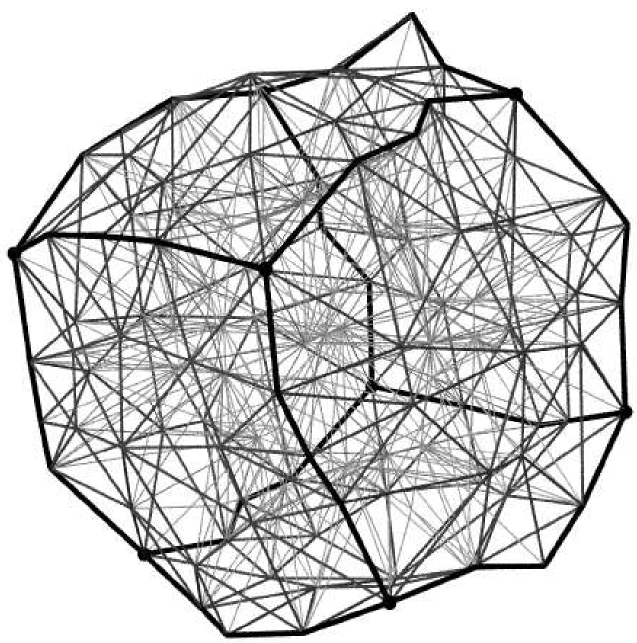


Figura 6.17: Projeção 3D de uma configuração que minimiza a energia $\mathcal{E} = 0.90\ \mathcal{E}_{curv3} + 0.10\ \mathcal{E}_{spring}$. A triangulação é um refinamento de ordem 2 da subdivisão baricêntrica do mapa `6-face`.

Para evitar este problema, é necessário definir uma energia sensível à curvatura das faces, como descrito a seguir. Sejam t_1, t_2 dois triângulos adjacentes de $\mathcal{T}$ contidos numa mesma face f de $\mathcal{C}$ e sejam r_1, r_2 dois vetores do $\mathbb{R}^4$ perpendiculares à aresta e, comum aos dois triângulos, tais que o vetor r_i está contido no plano U_i do triângulo t_i e aponta para o interior de t_i. A curvatura da face f na aresta e é mínima quando $U_1 = U_2$, isto é, quando $r_1 = -r_2$. Veja figura 6.18.

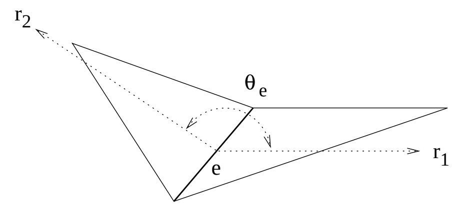


Figura 6.18: Ângulo diedral θ_e entre duas faces adjacentes de $\mathcal{T}$.

Definimos então a *energia de curvatura bidimensional* pela fórmula

$$\mathcal{E}_{curv2}(\phi) = K_{curv2} \sum_{e \in E'} \left(\frac{1}{1 - \cos\theta_e} - \frac{1}{2} \right) , \tag{6.4}$$

onde o conjunto E' inclui toda aresta $e \in \mathbf{E}\mathcal{T}$ que está contida em alguma face f do mapa original $\mathcal{C}$; e θ_e é o ângulo entre os vetores r_1 e r_2 associados aos triângulos da face f incidentes à aresta e, como definimos acima.

Como observado na seção 6.4.1, o termo geral da somatória na fórmula (6.4) é aproximadamente $(\theta_e - \pi)^2$ quando θ_e é próximo de π. Portanto, minimizar $\mathcal{E}_{curv2}$ tende a aplainar as faces de $\mathcal{C}$, distribuindo a curvatura inevitável das mesmas entre as arestas de $\mathcal{T}$ da maneira mais uniforme possível. Veja a figura 6.19.

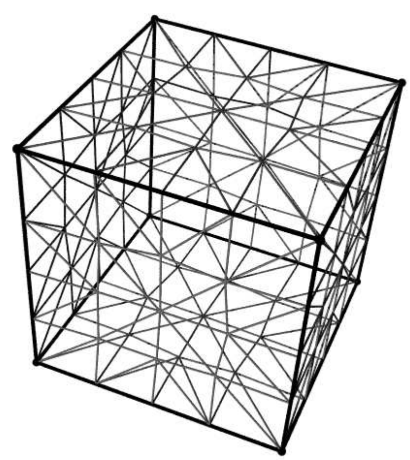


Figura 6.19: Projeção 3D de uma configuração que minimiza a energia $\mathcal{E} = 0.9\ \mathcal{E}_{curv2} + 0.1\ \mathcal{E}_{curv3}$. A triangulação é a mesma da figura 6.17 (um refinamento de ordem 2 da subdivisão baricêntrica do mapa `6-face`).

6.4.3 Energia de curvatura unidimensional

Na maioria dos mapas de interesse prático, toda aresta pertence a pelo menos duas faces que, numa configuração ótima, se encontram formando um ângulo menor que π. Nesse caso, a suavidade das faces do mapa implica até certo ponto na suavidade das arestas. Entretanto, em casos especiais, pode ser necessário impor este último critério explicitamente.

Para esse fim, vamos considerar duas arestas adjacentes e_1, e_2 de $\mathcal{T}$ pertencentes a uma mesma aresta h de $\mathcal{C}$. Sejam r_1, r_2 dois vetores do $\mathbb{R}^4$ definidos sobre e_1, e_2, apontando para longe do vértice compartilhado v. A curvatura da aresta h no vértice v é mínima quando $r_1 = -r_2$. Veja a figura 6.20.

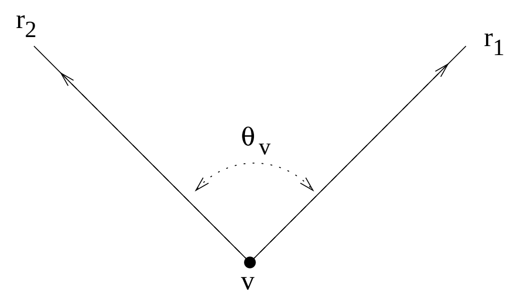


Figura 6.20: Ângulo θ_v entre sub-espaços unidimensionais.

Portanto, definimos a *energia de curvatura unidimensional* pela equação

$$\mathcal{E}_{curv1}(\phi) = K_{curv1} \sum_{v \in V'} \left(\frac{1}{1 - \cos\theta_v} - \frac{1}{2} \right) , \tag{6.5}$$

onde V' é o conjunto de todo vértice v de $\mathcal{T}$ que está contido em alguma aresta $e \in \mathbf{E}\mathcal{C}$, θ_v é o ângulo entre os vetores r_1 e r_2 associados às duas arestas de $\mathcal{T}$ incidentes em v e K_{curv1} é um fator de normalização (vide seção 6.6). Minimizar a energia $\mathcal{E}_{curv1}$ implica em deixar todas as arestas de $\mathcal{C}$ retilíneas, na medida do possível.

6.5 Outras funções de energia

Além das energias acima descritas, nós experimentamos várias outras energias, que eventualmente resultaram supérfluas e/ou não deram resultados satisfatórios. Algumas dessas energias são descritas a seguir.

6.5.1 Energia de excentricidade

A energia de excentricidade $\mathcal{E}_{excen}$ mede o quanto cada vértice v esta longe do baricentro do conjunto N_v de seus vizinhos. Especificamente,

$$\mathcal{E}_{excen}(\phi) = K_{excen} \sum_{v \in \mathbf{V}\mathcal{T}} |c_v - b_v|^2 \,, \tag{6.6}$$

onde c_v são as coordenadas cartesianas do vértice v; b_v é o baricentro de todos os vizinhos do vértice v; e K_{excen} é um fator de normalização (vide seção 6.6). Minimizar $\mathcal{E}_{excen}$ tende a equalizar o comprimento de todas as arestas incidentes ao vértice v e também a manter o vértice v no mesmo sub-espaço tridimensional que seus vizinhos.

Esta energia revelou-se supérflua, pois minimizar $\mathcal{E}_{excen}$ é aproximadamente equivalente a minimizar alguma combinação da energia elástica de arestas $\mathcal{E}_{edge}$ e da energia de curvatura $\mathcal{E}_{curv3}$; e, por sua vez, $\mathcal{E}_{edge}$ é menos útil que a energia de molas $\mathcal{E}_{spring}$.

6.5.2 Energia de espalhamento

A *energia de espalhamento* é definida pela fórmula

$$\mathcal{E}_{spread}(\phi) = K_{spread} \sum_{v \in \mathbf{V}\mathcal{T}} |c_v|^2 \,, \tag{6.7}$$

onde c_v são as coordenadas cartesianas do vértice v, e K_{spread} é um fator de normalização (vide seção 6.6). Minimizar esta energia tende a manter todos os vértices próximos à origem do $\mathbb{R}^4$. Esta energia é supérflua pois $\mathcal{E}_{spring}$ já impede que os vértices se afastem além de um certo limite.

6.5.3 Energia de compressão de volume

A *energia de compressão de volume* é definida por

$$\mathcal{E}_{compr}(\phi) = K_{compr} \sum_{t \in \mathbf{T}\mathcal{T}} \left[\left(\frac{v_t}{V_t}\right)^2 + \left(\frac{V_t}{v_t}\right)^2 - 2 \right] \,, \tag{6.8}$$

onde v_t é o volume corrente do tetraedro t, V_t seu volume ideal e K_{compr} é um fator de normalização (vide seção 6.6). Minimizar $\mathcal{E}_{compr}$ tende a manter os volumes dos tetraedros próximos aos volumes ideais V_t. Esta energia é até certo ponto supérflua pois a minimização de $\mathcal{E}_{edge}$ ou $\mathcal{E}_{spring}$ já produz este efeito (além de outros).

6.5.4 Energia elástica de tetraedros

Esta energia é uma generalização das energias $\mathcal{E}_{compr}$ e $\mathcal{E}_{edge}$. Ela considera que cada tetraedro t de $\mathcal{T}$ é feito de algum material elástico, com uma forma de repouso $\phi_0(t)$ especificada. A *energia elástica de tetraedros* mede o grau de deformação necessário para transformar a forma canónica $\phi_0(t)$ na forma corrente $\phi(t)$. A energia é dada pela fórmula

$$\mathcal{E}_{elast}(\phi) = K_{elast} \sum_{t \in \mathbf{T}\mathcal{T}} V_t^0 \left[\frac{\alpha}{32} \left(\Delta_t^2 + \frac{1}{\Delta_t^2} - 2 \right) + \frac{\beta}{6} \left(\Gamma_t^2 - 3\Sigma_t \right) \right] , \tag{6.9}$$

onde V_t^0 é o volume ideal do tetraedro $\phi_0(t)$. Os parâmetros α e β são os módulos de elasticidade, volumétrico e de rigidez do material, respectivamente; Δ_t, Γ_t e Σ_t são os invariantes de rotação da deformação corrente do tetraedro t, calculados a partir da matriz M que leva $\phi_0(t)$ para $\phi(t)$. A constante K_{elast} é um fator de normalização (vide seção 6.6). Uma descrição detalhada desta energia, em particular dos invariantes de rotação, pode ser encontrada na tese de R. L. W. Liensenfeld [21].

A energia $\mathcal{E}_{elast}$ resultou supérflua porque duplica até certo ponto um dos objetivos da energia $\mathcal{E}_{edge}$.

6.5.5 Energia de ângulos diedrais

A *energia de ângulos diedrais* é definida por

$$\mathcal{E}_{equa}(\phi) = K_{equa} \sum_{e \in \mathbf{E}\mathcal{T}} \sum_{i=1}^{d_e} \left[\left(\frac{\theta_i}{\hat{\theta}_e} \right)^2 + \left(\frac{\hat{\theta}_e}{\theta_i} \right)^2 - 2 \right] , \tag{6.10}$$

onde d_e é o número de faces de $\mathcal{T}$ incidentes na aresta e, θ_i é o ângulo diedral entre faces consecutivas i e $(i+1) \bmod d_e$, na ordem topológica; $\hat{\theta}_e = 2\pi/d_e$ é o ângulo diedral medio ideal; e K_{equa} é um fator de normalização (vide seção 6.6). Esta energia é computada somente para arestas que não estão na borda de $\mathcal{T}$. Minimizar $\mathcal{E}_{equa}$ tende a deixar os ângulos diedrais próximos a seus valores ideais $\hat{\theta}_e$, o que ajuda a reduzir auto-intersecções da variedade subjacente a $\phi(\mathcal{T})$.

Esta energia revelou-se supérflua, uma vez que produz, aproximadamente, os mesmos efeitos da energia $\mathcal{E}_{spring}$ — pelo menos, para arestas de grau moderado.

6.5.6 Energia de orientação de tetraedros

A *energia de orientação de tetraedros* aplica-se quando a triangulação $\mathcal{T}$ é orientável (seção 4.1.4) e pressupõe que todos seus tetraedros estão orientados de maneira coerente. Ela é definida pela fórmula

$$\mathcal{E}_{orien}(\phi) = K_{orien} \sum_{t \in \mathbf{T}\mathcal{T}} \left[M \; (\sqrt{1+\Delta_t^2} - \Delta_t) + \Delta_t^2 - 2 \right] , \tag{6.11}$$

onde

$$\Delta_t = \begin{vmatrix} x_0 & x_1 & x_2 & 1 \\ y_0 & y_1 & y_2 & 1 \\ z_0 & z_1 & z_2 & 1 \\ w_0 & w_1 & w_2 & 1 \end{vmatrix},$$

e x, y, z, w são as coordenadas tridimensionais (após a projeção) dos vértices do tetraedro $t \in \mathbf{T}\mathcal{T}$ na ordem determinada pela orientação de $\mathcal{T}$. O parâmetro M é uma constante suficientemente grande, e K_{orien} é um fator de normalização (vide seção 6.6). A energia $\mathcal{E}_{orien}$ penaliza configurações $\phi(\mathcal{T})$ que possuem tetraedros cuja orientação geométrica discorda da orientação topológica. A minimização desta energia tende a reduzir dobramentos não essenciais. Quando a variedade subjacente é homoeomorfa a um subconjunto do $\mathbb{R}^3$, a minimização desta energia geralmente produz um modelo que é localmente um mergulho no $\mathbb{R}^3$.

6.6 Normalização das funções de energia

Para usar as funções de energia eficazmente, é desejável que seus valores relativos não sejam significativamente afetados por refinamentos do modelo geométrico (vide seção 4.4). Essa propriedade nos permitiria otimizar vários refinamentos de um mesmo mapa $\mathcal{C}$, sem ter que ajustar os pesos α_i das energias mistas para cada refinamento.

Mais especificamente, queremos garantir, para qualquer energia mista $\mathcal{E}$, que as configurações mínimas ϕ da triangulação $\mathcal{T}$ são próximas das configurações mínimas ϕ' de qualquer subdivisão topológica $\mathcal{T}'$ de $\mathcal{T}$.

Foi pensando neste objetivo que definimos a distância ideal D_{uv}, na fórmula da energia $\mathcal{E}_{spring}$, como inversamente proporcional ao diâmetro topológico da triangulação $\mathcal{T}$ (vide seção 6.3). Foi também por esse motivo que incorporamos em cada energia $\mathcal{E}_i$ um fator de normalização K_i. Nos testes apresentados nesta tese, usamos $K_i = 1$ para todas as energias. Em princípio, sería possível alcançar o objetivo acima calculando-se valores apropriados para K_i (e para os demais parâmetros da fórmula de $\mathcal{E}_i$), a partir da topologia de $\mathcal{T}$. Infelizmente, este parece ser um problema difícil para o qual ainda não temos nenhuma solução adequada.

Capítulo 7

Otimização

Uma vez estabelecida a função de energia $\mathcal{E}$ a minimizar, o próximo passo consiste em encontrar uma configuração com energia mínima. Este é um problema não trivial, uma vez que a função $\mathcal{E}$ depende de $m \cdot |\mathbf{V}\mathcal{T}|$ variáveis ($m \geq 3$), e geralmente possui um grande número de mínimos locais.

Nossa abordagem é utilizar métodos gerais de otimização não-linear para transformar uma dada configuração inicial ϕ_0 numa configuração ϕ que é um mínimo local de $\mathcal{E}$. Repetimos esta busca para um certo número de configurações iniciais ϕ_0, escolhidas aleatoriamente e selecionamos a configuração final com menor energia.

Infelizmente, não existe um critério automático para estabelecer que valor da energia $\mathcal{E}$ corresponde a uma boa visualização. Na prática, fixamos um limite de tempo e retornamos a melhor configuração encontrada dentro deste limite.

Nós estudamos a possibilidade de usar métodos gerais de otimização combinatória, tais como recozimento simulado (*simulated annealing*), para estender a busca além do mínimo local encontrado pelos métodos de otimização genéricos. Porém, a experiência obtida na visualização de complexos 2D [30] nos levou a crer que tais métodos não seriam mais eficazes do que a simples repetição da minimização a partir de várias configurações iniciais aleatórias.

No nosso trabalho fizemos uso de dois métodos de otimização genéricos: minimização coordenada a coordenada (que denotaremos por `Coord`), com iterações diagonais periódicas [29]; e descida pelo gradiente (que denotaremos por `Grad`), com passo adaptativo [29]. Estes algoritmos estão descritos nas seções 7.1 e 7.2 respectivamente.

A figura 7.1 mostra várias etapas do progresso na otimização do mapa `600-cell` para a energia $\mathcal{E}_{spring}$.

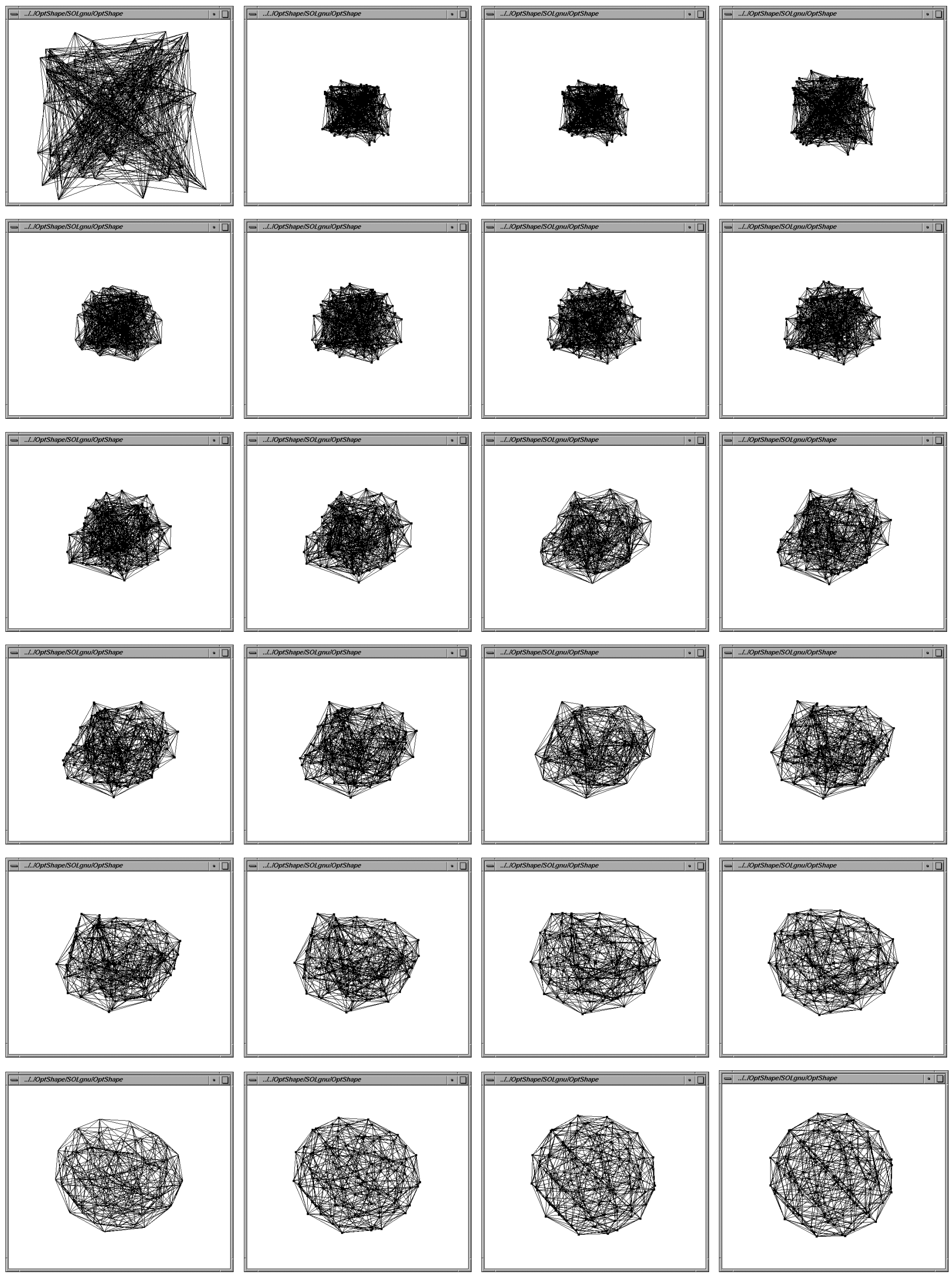


Figura 7.1: Otimização do mapa `600-cell`. Na sequência: a configuração inicial e configurações intermediárias após, 40, 50, 60, 110, 120, 130, 140, 160, 180, 190, 200, 210, 220, 230, 240, 250, 260 280, 300, 320, 330, 350 e 430 passos do método `Grad`, na minimização da energia $\mathcal{E}_{spring}$.

7.1 Minimização coordenada a coordenada

O método `Coord` de otimização coordenada a coordenada consiste em otimizar uma coordenada de cada vértice por vez, usando um método de minimização unidimensional — como por exemplo o algoritmo de Brent [29] — enquanto todos os outros vértices são mantidos fixos. Após cada $n+1$ iterações do método acima, fazemos uma iteração "diagonal" extra ao longo da linha conectando as configurações resultantes da primeira e da última iteração. Demonstra-se que, se a função objetivo fosse um polinômio quadrático nas variáveis, (com matriz hessiana positiva definida), esta linha reta passaria pelo mínimo global.

O método `Coord` está detalhado no algoritmo 7.1, que usa os procedimentos auxiliares `UniMin`, `EvalAxial` e `EvalDiagonal`, descritos mais adiante. Este método pode ou não usar o gradiente da função energia, dependendo do minimizador `UniMin` empregado.

7.1.1 Procedimento principal

Algoritmo 7.1 (`Coord`)**: Minimização Coordenada a Coordenada**

Entrada:

- *evalF: procedimento que calcula a função $F(x)$ e seu gradiente $\nabla F(x)$.*
- *n: número de variáveis.*
- *δ: distância estimada do ponto inicial ao ótimo.*
- *ε: precisão desejada na posição do ponto ótimo.*
- *umax: número máximo de avaliações de $F(x)$ permitidas por coordenada.*
- *emax: número máximo total de avaliações de $F(x)$ permitidas.*

Entrada e Saída:

- *x: ponto inicial no $\mathbb{R}^m$ (entrada) e ponto de mínimo (saída).*
- *f: valor da função F em x (entrada e saída).*
- *gf: gradiente da função F em x (entrada e saída).*

Variáveis locais:

- s, r, y, k, nev, m, f_0, t_0.

Passos:

1. $nev \leftarrow 0$ [*total de avaliações da função*]
2. $k \leftarrow 0$ [*coordenada sendo minimizada*]
3. $m \leftarrow 0$ [*minimizações axiais desde a última minimização diagonal*]

4. $s_i \leftarrow \delta/\sqrt{n}$, $i = 1 \dots n$ [*último passo efetuado em cada coordenada*]
5. enquanto $nev < emax$, faça:
   5.1. se $m \leq n$ então [*realize uma minimização axial*]:
      5.1.1. $t_0 \leftarrow x_k$ [*valor de* x_k *antes da minimização axial*]
      5.1.2. $y \leftarrow x$ [*origem da linha axial*]
      5.1.3. $t \leftarrow x_k$; $h \leftarrow f$; $dh \leftarrow gf_k$;
      5.1.4. `UniMin`(`EvalAxial`, $|s_k| + \varepsilon/10, \varepsilon, umax, t, h, dh$)
      5.1.5. $m \leftarrow m + 1$
      5.1.6. $s_k \leftarrow x_k - t_0$
      5.1.7. se $m = 1$ então $f_0 \leftarrow f$ [*valor de* F *após a primeira minimização axial*]
   5.2. senão [*realize uma minimização diagonal*]:
      5.2.1. se $|s| \leq \varepsilon/20$ então retorne;
      5.2.2. $y \leftarrow x$ [*origem da linha diagonal*]
      5.2.3. $r \leftarrow s/|s|$ [*direção da diagonal*]
      5.2.4. $t \leftarrow 0$; $h \leftarrow f$; $dh \leftarrow r \cdot gf$; [*distância ao longo da linha diagonal*]
      5.2.5. `UniMin`(`EvalDiagonal`, $|s| + \varepsilon/10, \varepsilon, umax, t, h, dh$)
      5.2.6. se $|(f_0 - f)|/(|s| + t) \leq \varepsilon$ então retorne;
      5.2.7. $m \leftarrow 0$;
   5.3. $k \leftarrow (k + 1) \bmod n$;
6. retorne.

7.1.2 Procedimento UniMin

Algoritmo 7.2 (`UniMin`)**: Minimização unidimensional**
Entrada:

- *evalH: procedimento que calcula a função* $H(t)$ *e sua derivada* $(dH/dt)(t)$.
- δ*: distância estimada do* t *inicial ao ótimo.*
- ε*: precisão exigida.*
- *umax: número máximo de cálculos de* $H(t)$ *permitidos.*

Entrada e Saída:

- t*: argumento inicial (entrada) e ótimo (saída).*
- h*: valor de* $H(t)$ *(entrada e saída).*
- dh*: derivada de* H *em* t *(entrada e saída).*

O procedimento `UniMin`($evalH, \delta, \varepsilon, a, b, umax, t, h, dh$) busca um mínimo local de uma função H de um único argumento real. Inicialmente, a variável t deve conter uma estimativa inicial para a posição do mínimo, enquanto que h e dh devem conter os correspondentes valores da função H e sua derivada. O procedimento dado $evalH(t)$ deve calcular o valor de $H(t)$ e de sua derivada. O procedimento `UniMin` termina a busca quando determina algum intervalo de tamanho menor ou igual a ε que contém um mínimo local, ou após efetuar $umax$ cálculos de H. Em qualquer caso, o procedimento devolve em t o mínimo encontrado, e em h e dh os correspondentes valores da função H e sua derivada.

O algoritmo que utilizamos em nossos testes é o método de minimização unidimensional de Brent [29], que não usa a informação de derivada.

7.1.3 Procedimento EvalAxial

O procedimento `EvalAxial` (t) é invocado pelo minimizador univariado `UniMin` para computar o valor da função F num ponto ao longo de uma linha axial (paralela a um dos eixos de coordenadas). A reta é definida pelas variáveis externas y (um ponto sobre a reta) e k (índice da coordenada em questão), definidas no procedimento `Coord`. Mais precisamente, `EvalAxial` calcula $H(t) = F(y \stackrel{k}{\leftarrow} t)$, onde $y \stackrel{k}{\leftarrow} t$ denota o ponto y com a coordenada y_k substituída por t. O procedimento atualiza também as variáveis (x, f, gf) do algoritmo `Coord`, quando o valor calculado resulta menor que o ótimo corrente f.

Algoritmo 7.3 (`EvalAxial`)**: Cálculo da função ao longo de uma reta axial**

Variáveis globais:

- k: *coordenada que está sendo minimizada.*
- y: *ponto na reta axial.*
- x, f, gf, nev: *vide algoritmo* `Coord`.

Entrada:

- t: *novo valor para a coordenada* x_k.

Saída:

- h: *valor da função* $H(t) = F(y \stackrel{k}{\leftarrow} t)$.
- dh: *derivada de* $H(t) = F(y \stackrel{k}{\leftarrow} t)$ *em relação a* t.

Variáveis locais:

- x', f', gf'.

Passos:

1. $x' \leftarrow y$; $x'_k \leftarrow t$;
2. $(f', gf') \leftarrow evalF(x')$; $nev \leftarrow nev + 1$;
3. $h \leftarrow f'$; $dh \leftarrow gf'_k$;
4. se $f' < f$ então $x \leftarrow x'$; $f \leftarrow f'$; $gf \leftarrow gf'$;

7.1.4 Procedimento EvalDiagonal

O procedimento `EvalDiagonal`(t) é invocado pelo minimizador univariado `UniMin` para computar o valor da função F num ponto ao longo de uma reta diagonal que liga os extremos da primeira e última otimização axiais. Mais precisamente, `EvalDiagonal` calcula a função $H(t) = F(y + t\ r)$, onde y é o fim do último passo axial e r é a direção da diagonal, ambos definidos no procedimento `Coord`.

Algoritmo 7.4 (`EvalDiagonal`): **Cálculo da função ao longo de uma diagonal**

Variáveis globais:

- y*: origem da diagonal.*
- r*: direção da diagonal.*
- x, f, gf, nev*: vide algoritmo* `Coord`.

Entrada:

- t*: distância a partir de* t_0 *ao longo da diagonal.*

Saída:

- h*: valor da função* F *no ponto* $y + t\, r$*.*
- dh*: derivada de* $F(y + t\, r)$ *em relação a* t*.*

Variáveis locais:

- x', f', gf'.

Passos:

1. $x' \leftarrow y + t\, r$;
2. $(f', gf') \leftarrow evalF(x')$; $nev \leftarrow nev + 1$;
3. $dh \leftarrow gf' \cdot r$;
4. se $f' < f$ então $x \leftarrow x'$; $f \leftarrow f'$; $gf \leftarrow gf'$;

7.2 Método descida pelo gradiente

O método `Grad` de minimização por descida pelo gradiente tenta seguir a trajetória $x(t)$ definida pela equação diferencial

$$\frac{dx}{dt} = -\nabla f(x(t)) \tag{7.1}$$

partindo do ponto inicial $x(0)$, até atingir um ponto onde $\nabla f(x(t)) = 0$. Note que esta equação obriga o ponto x a evoluir sempre na direção de maior decrescimento da função.

Resolvemos a equação (7.1) numericamente usando um simples integrador de Euler com passo adaptativo [29]. Na verdade, nosso otimizador integra a equação diferencial

$$\frac{dx}{dt} = -\frac{\nabla f(x(t))}{|\nabla f(x(t))|} \,. \tag{7.2}$$

enquanto $\nabla f(x(t)) \neq 0$. Vale observar que as equações (7.1) e (7.2) definem a mesma curva, parametrizada de maneira diferente. Com a equação (7.1), a velocidade de percurso é proporcional à magnitude do gradiente, e portanto diminui rapidamente à medida que o otimizador se aproxima do mínimo exigindo ajustes frequentes no passo de integração. É por essa razão que utilizamos a equação (7.2), em que a trajetória é percorrida com velocidade constante ($|dx/dt| = 1$).

O algoritmo 7.5 descreve o método `Grad` em detalhe. O algoritmo usa os procedimentos auxiliares `Diff` e `CheckStep`, definidos mais adiante.

7.2.1 Procedimento principal

Algoritmo 7.5 (`Grad`)**: Minimização por descida pelo gradiente**

Entrada:

- *evalF: procedimento que calcula a função $F(x)$ e seu gradiente $\nabla F(x)$.*
- *n: número de variáveis.*
- *δ: distância estimada do ponto inicial ao ótimo.*
- *ε: precisão desejada na posição do ótimo.*
- *emax: número total máximo de avaliações permitidas.*

Entrada e Saída:

- *x: ponto inicial no $\mathbb{R}^m$ (entrada) e ponto de mínimo (saída).*
- *f: valor da função F em x (entrada e saída).*
- *gf: gradiente da função F em x (entrada e saída).*

Variáveis locais:

- *err: estimativa de erro do estado.*
- *s, s': estados do integrador.*
- *v, v': derivadas dos estados do integrador.*
- *nev: número de avaliações de F.*
- *dt, t, t'', t': variáveis de tempo.*

Passos:

1. se $|gf| = 0$, retorna.
2. $s \leftarrow x$; $t \leftarrow 0$; [*inicializa o estado do sistema*]
3. $t'' \leftarrow t + \delta$
4. enquanto $t < t''$ faça:
   4.1. $v \leftarrow$ `EvalDiff`(t, s);
   4.2. se $|v| = 0$ ou $nev \geq emax$ então retorna;
   4.3. repita [*tenta avançar até algum tempo t' no intervalo $(t, \cdots, t'']$*]
      4.3.1. $t' \leftarrow t''$
      4.3.2. $(s', v', err) \leftarrow$ `Step`(t, s, v, t') [*extrapola até t'*]
      4.3.3. $t'' \leftarrow$ `CheckStep`(t, s, t', s', err) [*verifica se o erro é aceitável*]
      4.3.4. se $t'' \leq t$ então retorna;
   4.4. até que $t'' \geq t'$;
   4.5. $t \leftarrow t'$; $s \leftarrow s'$ [*avança o sistema para o tempo t'*]
5. retorna.

O passo $t' - t$ é automaticamente determinado em cada passo, pelo procedimento `CheckStep` descrito mais adiante.

7.2.2 Procedimento Step

O procedimento `Step`(t, s, v, t') executa um passo genérico do integrador adaptativo de Euler para a equação $ds/dt = D(t, s)$, onde D é uma função dada. O procedimento extrapola linearmente a trajetória do sistema desde o tempo t, dados o estado s e a velocidade $v = ds/dt(t)$, até um tempo dado $t' > t$. O resultado é o novo estado $s' = s(t')$, a nova velocidade $v' = \dot{s}(t')$ e um vetor *err* que dá o erro estimado de integração em cada componemte do estado s'.

Algoritmo 7.6 (`Step`)**: Passo do integrador de Euler**

Entrada:

- t: *tempo inicial.*
- s: *estado do sistema no tempo* t.
- v: *derivada do sistema no tempo* t.
- t': *tempo final.*

Saída:

- s': *estado do sistema no tempo* t'.
- *err: estimativa de erro para o estado.*
- v': *derivada do estado no tempo* t'.

Passos:

1. $s' \leftarrow s + (t' - t)v$;
2. $v' \leftarrow \texttt{EvalDiff}(t', s')$;
3. $err \leftarrow (t' - t)(v - v')/2$;

7.2.3 Procedimento EvalDiff

Este procedimento é invocado (várias vezes por passo) pelo integrador de Euler para computar a derivada $v(t) = ds/dt$ do estado s no tempo t.

Algoritmo 7.7 (`EvalDiff`)**: Calcula a derivada do estado**

Variáveis globais:

- $evalF$, x, f, gf, *nev: vide algoritmo* `Grad`.

Entrada:

- t: *um instante de tempo.*
- s: *o estado correspondente* $s(t)$.

Saída:

- v: *derivada* ds/dt *no instante* t.

Variáveis locais:

- x', f', gf'.

Passos:

1. $x' \leftarrow s$;
2. $(f', gf') \leftarrow evalF(x')$; $nev \leftarrow nev + 1$;
3. Se $f' < f$ então $x \leftarrow x'$, $f \leftarrow f'$, $gf \leftarrow gf'$;
4. Se $|gf'| = 0$ então $v \leftarrow \mathbf{0}$, senão $v \leftarrow -gf'/|gf'|$

7.2.4 Procedimento CheckStep

O procedimento `CheckStep`(t, s, t', s', err) é invocado pelo integrador de Euler para decidir se o último passo — do estado s no tempo t até o estado s' no tempo t', com erro estimado err — é aceitável. Em caso afirmativo, o procedimento devolve um tempo $t'' \geq t$, que é o tempo final sugerido para o próximo passo. Caso contrário, `CheckStep` devolve um tempo $t'' < t'$, para o qual ele estima que o erro seria aceitável.

Algoritmo 7.8 (`CheckStep`): Verifica a validade de um passo.

Entrada:

- t: *tempo inicial.*
- s: *estado inicial.*
- t': *tempo final.*
- s': *estado final.*
- err: *estimativa de erro cometido no passo* $t \to t'$.

Saída:

- t'': *tempo final apropriado, para este passo ou para o próximo.*

Variáveis globais:

- nev, $emax$, ε, δ: *vide algoritmo* `Grad`.

Variáveis locais:

- *speed, dt, dtmin, dtmax, dtideal, dtnew.*

Passos:

1. se $nev > emax$ então retorna t'; [*termina integração*]
2. $speed \leftarrow |s - s'|/(t' - t)$; [*velocidade corrente*]
3. $dtmin \leftarrow max(1\text{-}100, 0.001\ \varepsilon/speed)$; [*menor passo aceitável*]

4. $dtmax \leftarrow min(1{+}100, 100\ \delta/speed)$; [*maior passo aceitável*]
5. $dt \leftarrow t' - t$; [*passo corrente*]
6. se $|err| > 0$ então:
   6.1. $\alpha \leftarrow \sqrt{\varepsilon/|err|}$ [*fator de ampliação do passo*]
   6.2. se $\alpha \leq 0.1$ então $dtideal \leftarrow dt/10$
      6.2.1. senão, se $\alpha \geq 10$ então $dtideal \leftarrow 10\ dt$;
      6.2.1. senão $dtideal \leftarrow \alpha\ dt$;
   6.3. senão $dtideal \leftarrow 10\ dt$;
7. $dtnew \leftarrow \min(dtmax, \max(dtmin, dtideal))$
8. se $|err| < \varepsilon$
   8.1. então retorna $t'' \leftarrow t' + dtnew$ [*passo adequado, prosiga*]
   8.1. senão retorna $t'' \leftarrow t + dtnew$ [*tente novamente com passo menor*]

7.3 Comparação entre métodos de otimização

Nesta seção comparamos o desempenho do método de descida pelo gradiente (`Grad`) e minimização coordenada a coordenada (`Coord`).

Para facilitar a comparação dos diferentes otimizadores, definimos um *passo* do otimizador como sendo uma avaliação da função energia $\mathcal{E}(\phi)$ e seu gradiente se for o caso. A justificativa é que estes cálculos representam de longe a parte mais dispendiosa da otimização.

7.3.1 Custo do gradiente

Uma desvantagem do método de descida pelo gradiente (`Grad`), em relação ao métodos que não usam o gradiente, é a necessidade de computar o gradiente da função de energia $\nabla\mathcal{E}$. À primeira vista, este cálculo pode parecer extremanente dispensioso, pois exige calcular n derivadas cuja complexidade é geralmente bem maior que a da função. Entretanto, como técnicas apropriadas [1], pode-se reduzir o custo de computar o gradiente de qualquer função matemática a um custo bem modesto — apenas três ou quatro vezes o custo de calcular a própria função — aplicando-se sistematicamente a regra da cadeia a cada operação. Com essas técnicas, o custo de se calcular o gradiente é quase sempre compensado pela aceleração da convergência que pode ser obtida com o mesmo, como mostrado a seguir.

A figura 7.2 mostra a evolução da energia $\mathcal{E}(\phi)$ da melhor configuração obtida por cada método, em função do número t de passos (avaliações da função de energia) efetuados.

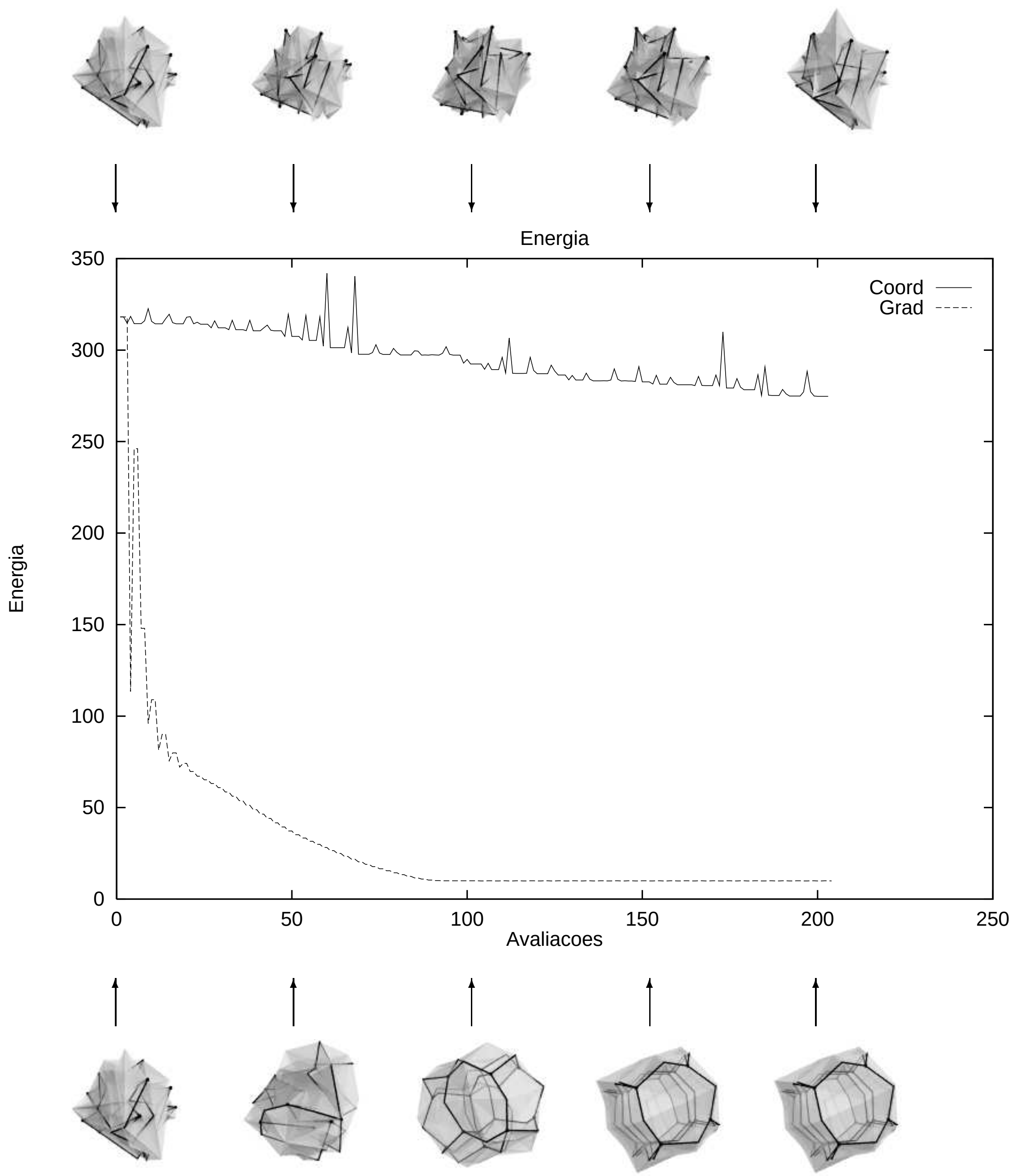


Figura 7.2: Evolução da energia $\mathcal{E}_{spring}(\phi)$ otimizada com os métodos `Coord` e `Grad` partindo de uma configuração aleatória, para uma triangulação do mapa `8-cell` (80 vértices). As configurações exibidas na parte superior (respectivamente inferior) na figura, foram obtidas com o método `Coord` (respectivamente `Grad`).

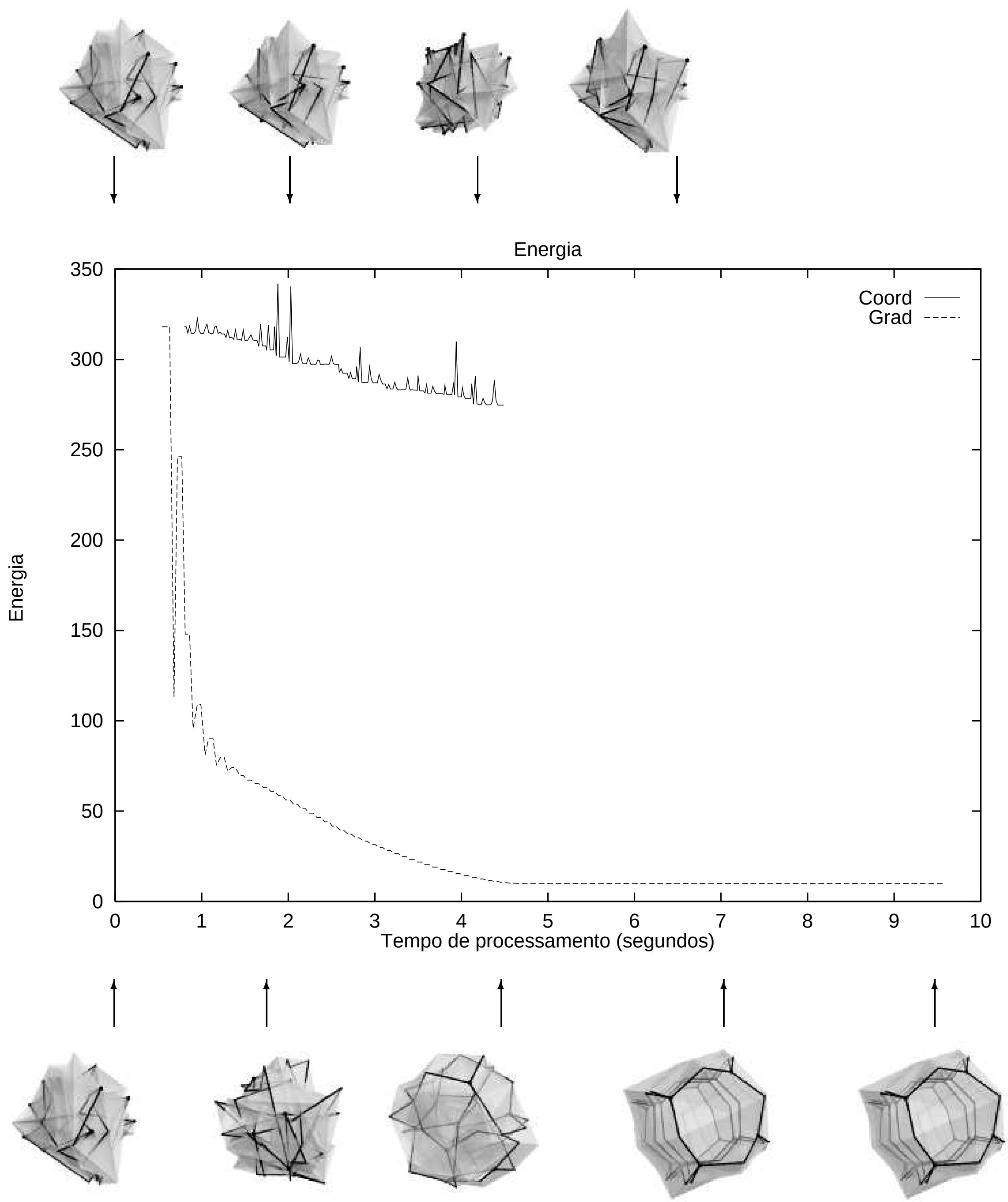


Figura 7.3: Evolução da energia $\mathcal{E}_{spring}(\phi)$ otimizada com os métodos `Coord` e `Grad` partindo de uma configuração aleatória. A triangulação é a mesma da figura 7.2. As configurações na parte superior (respectivamente inferior) da figura, foram obtidas com o método `Coord` (respectivamente `Grad`).

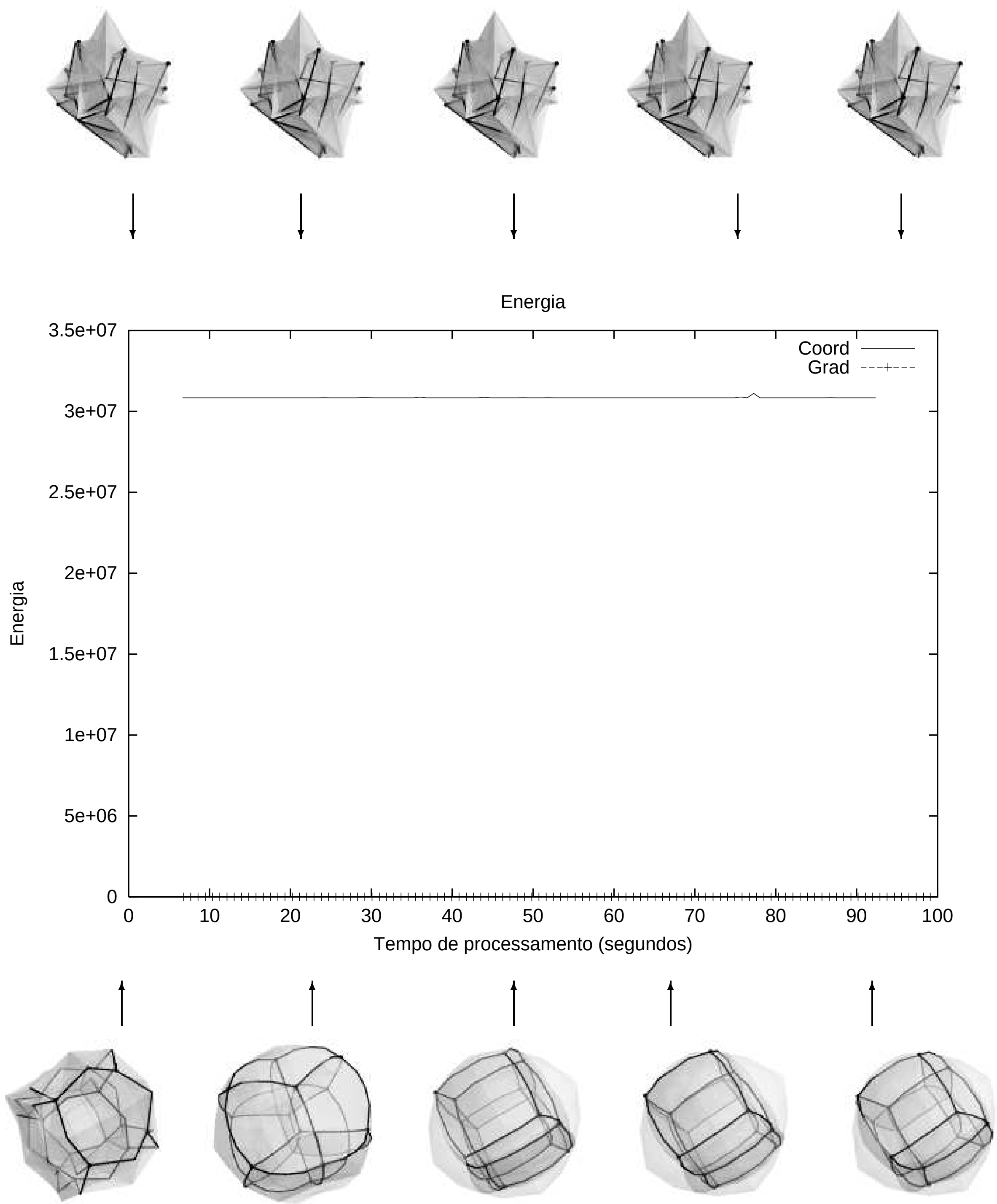


Figura 7.4: Comparação dos métodos `Coord` e `Grad` partindo de uma configuração "amaciada" (energia × tempo processamento) para uma triangulação com 928 vértices e energia $\mathcal{E}_{curv3}$. As configurações nesta figura foram obtidas através da abordagem multiescala, partindo das configurações finais na figura 7.2. As realizações na parte superior (respectivamente inferior) na figura, foram obtidas com o método `Coord` (respectivamente `Grad`).

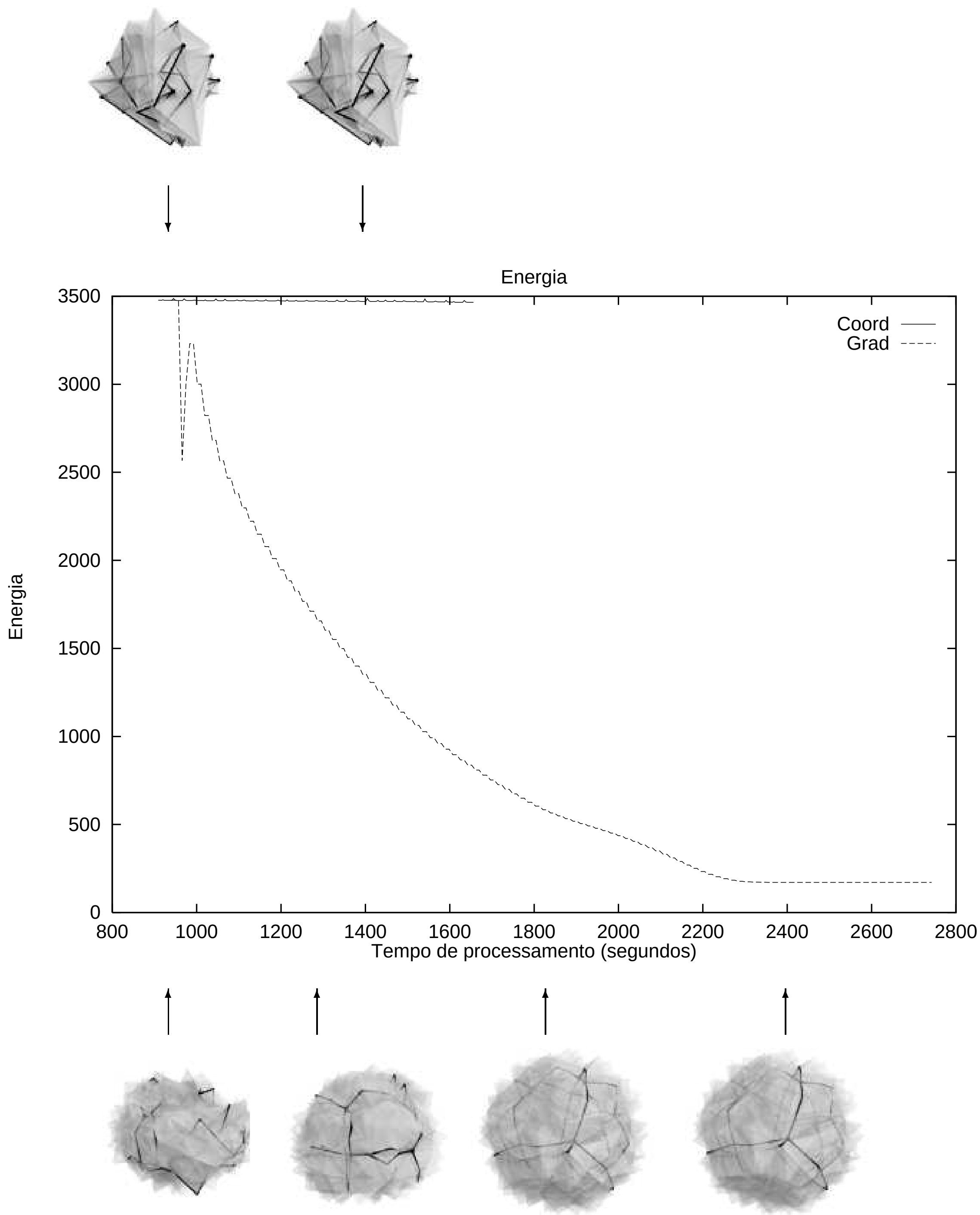


Figura 7.5: Comparação dos métodos `Coord` e `Grad` partindo de uma configuração aleatória e refinada (energia $\times$ tempo processamento) com 928 vértices e energia $\mathcal{E}_{spring}$. As realizações no extremo superior (inferior) na figura, foram obtidas com o método `Coord` (`Grad`).

Resumindo, no limite das nossas experiências, o método de otimização `Grad`, baseado no princípio de descida pelo gradiente, é geralmente mais eficiente do que os algoritmos que não usam o gradiente — em particular, `Coord`.

Por outro lado, no caso particular das energias de curvatura, o método `Grad` mostrou-se efetivo só se configuração inicial já está relativamente "amaciada" (veja figura 7.4).

7.4 Otimização multi-escala

A otimização é muito dispendiosa quando o número de vértices é grande, não somente devido ao alto custo de avaliar as funções de energia, mas também devido ao fato de que um modelo maior possui mais mínimos locais indesejáveis, onde a otimização pode ficar presa. Portanto, para modelos com milhares de vértices, geralmente fazemos uso de uma *aproximação multi-escala*. A saber, partindo de uma configuração aleatória, primeiro otimizamos um modelo grosseiro com o nível mínimo de subdivisão necessário para evitar degenerações geométricas. Uma vez obtida uma configuração ótima aproximada ϕ'_0, subdividimos este modelo, retendo sua geometria (isto é, atribuímos coordenadas iniciais $\phi_1(v)$ aos novos vértices por interpolação linear da configuração ϕ'_0) e usamos esta configuração ϕ_1 como o ponto inicial para um segunda etapa de otimização; e assim repetidamente. Por este processo, temos conseguido otimizar modelos com mais de 15000 tetraedros, em somente algumas horas de cálculo.

Por outro lado o uso da abordagem de otimização multi-escala mostrou-se bastante adequada em comparação ao não uso desta técnica como mostrado na figura 7.5.

Capítulo 8

Visualização do modelo otimizado

Neste capítulo detalhamos as técnicas que usamos para visualizar os modelos geométricos dos mapas tridimensionais.

De modo geral, nossa abordagem consiste em primeiro projetar o modelo $\phi(\mathcal{T})$ do espaço de modelagem $\mathbb{R}^m$ para o $\mathbb{R}^3$, através de uma extensão direta da projeção em perspectiva padrão [16]. O resultado da projeção é um conjunto de tetraedros, colados por suas faces, usualmente com muitos pares de tetraedros sobrepostos. Este objeto tridimensional pode ser renderizado com métodos convencionais de computação gráfica (tais como o traçado de raios), produzindo imagens bidimensionais que podem ser exibidas na tela do computador ou em algum outro dispositivo gráfico.

8.1 Ferramentas de visualização

Para visualizar o modelo, nós utilizamos uma variedade de ferramentas. Algumas dessas ferramentas permitem visualizar diretamente o modelo 4D, enquanto que outras exigem a aplicação prévia de um programa que efetue a projeção de $\mathbb{R}^4$ para o $\mathbb{R}^3$.

8.1.1 Visualização direta interativa

A ferramenta mais simples que utilizamos é o programa *Wire4* de S. R. Hollasch [16], um visualizador *wireframe* baseado na biblioteca *GL* (*Graphics Library*). O *Wire4* mostra a configuração $\phi(\mathcal{T})$ projetando-a do $\mathbb{R}^4$ para o $\mathbb{R}^2$. O programa *Wire4* suporta projeção paralela e/ou perspectiva do espaço $\mathbb{R}^4$ para o $\mathbb{R}^3$, seguida de outra projeção paralela e/ou perspectiva do $\mathbb{R}^3$ para o $\mathbb{R}^2$ e permite ao usuário mudar a posição do observador, tanto no espaço $\mathbb{R}^4$ quanto no $\mathbb{R}^3$.

Para nossos propósitos, modificamos o programa *Wire4* de modo a:

- utilizar cores RGB (24 bits/pixel) em vez de cores mapeadas (8 bits/pixel);
- oferecer opções de *antialiasing* e neblina;
- produzir pares de imagens para visão estereoscópica;
- rodar automaticamente o modelo com velocidade variável; e
- gerar arquivos em formatos reconhecidos por outros programas gráficos, tais como o *Geomview*, *X3D* e *POV-Ray*.

A figura 8.1 mostra uma imagem produzida com a ferramenta *Wire4*. O modelo apresentado é parte do próprio pacote *Wire4*; trata-se do 1-esqueleto do hipercubo (vide seção 3.2.11), constituído de 8 células cúbicas, 24 faces quadrangulares, 32 arestas e 16 vértices.

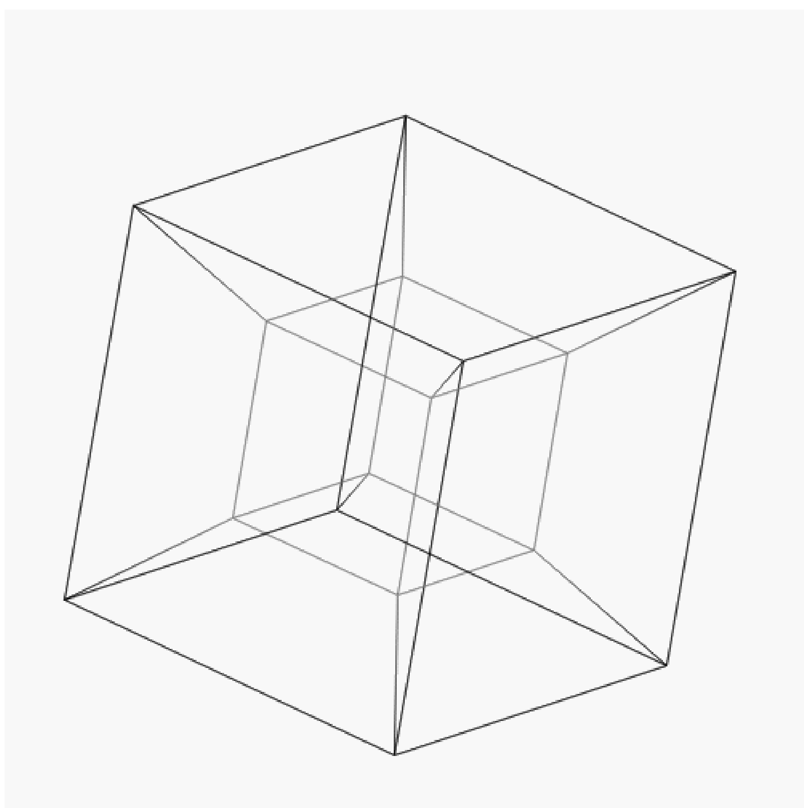


Figura 8.1: Imagem do mapa `8-cell` produzida pelo programa *Wire4*.

Esta figura ilustra o uso de variação de cor para evidenciar a distância no $\mathbb{R}^4$ entre o observador e os vários elementos do modelo. Observe-se que a célula cúbica que aparentemente é "interna" na projeção 3D na realidade encontra-se mais afastada do observador no $\mathbb{R}^4$ em comparação à célula "externa" (e, por esse motivo, é renderizada com cor menos intensa). Além disso, as arestas que conectam essas duas células estão inclinadas em relação ao espaço 3D de projeção e portanto sua projeção é ainda menor que as demais (e são renderizadas com cores graduais).

Na preparação do arquivo de entrada do *Wire4*, o usuário pode omitir arestas e especificar individualmente as cores das demais. Esta liberdade permite obter vários efeitos de visualização. Por exemplo a figura 8.2 é outra imagem produzida com o *Wire4*. O modelo geométrico (que será utilizado na maioria das figuras deste capítulo) é uma triangulação do mapa `8-cell`, obtida por subdivisão baricêntrica e dois estágios do refinamento uniforme de ordem dois. A triangulação resultante foi otimizada segundo a abordagem multi-escala: para a energia $\mathcal{E}_{spring}$, nas etapas iniciais e para a energia $\mathcal{E} = 0.99\,\mathcal{E}_{curv3} + 0.005\,\mathcal{E}_{curv2} + 0.005\,\mathcal{E}_{curv1}$, na etapa final (com 1000 passos do otimizador `Grad`). A configuração otimizada foi por fim projetada sobre a esfera $\mathbb{S}^3$. As linhas mais escuras são arestas originais do mapa, enquanto as linhas mais claras são as arestas das faces de silhueta (vide seção 8.3.1); as demais arestas da triangulação foram omitidas.

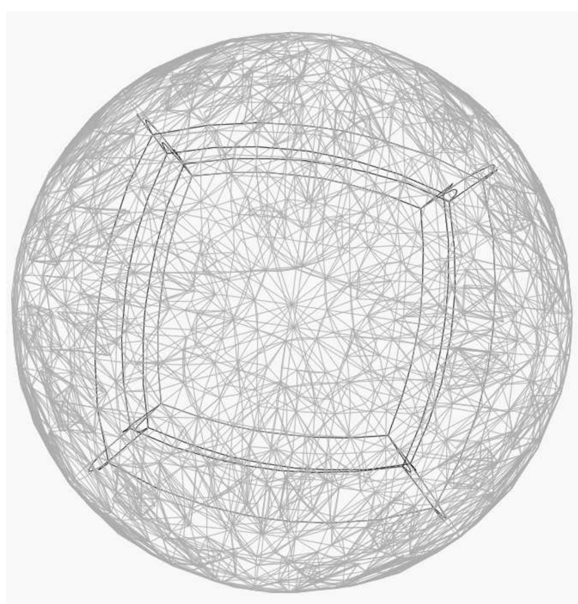


Figura 8.2: Realização do mapa `8-cell` renderizada com o programa *Wire4* (versão modificada).

Como pode ser apreciado pelas figuras, as imagens produzidas com *Wire4* são bastante incompletas, uma vez que não são exibidos os vértices, faces e células do mapa. Porém, o uso de tonalização por profundidade (vide seção 8.3.2) e a rotação interativa da câmara no $\mathbb{R}^4$ contrabalançam as deficiências mencionadas e fornecem uma compreensão bastante útil da estrutura desses mapas.

As "abas" na figura 8.2 que parecem se estender a partir das arestas dos cubos são na verdade faces que são curvas no $\mathbb{R}^4$ e são dobradas quase que sobre si mesmas pela projeção no $\mathbb{R}^3$. A figura 8.3 ilustra o efeito análogo na projeção do $\mathbb{R}^3$ para o $\mathbb{R}^2$. Observe-se que a projeção da face F estende-se por fora da projeção de suas arestas fronteiriças a, b, c, d e que a projeção das arestas c e a estão dobradas em forma de "U".

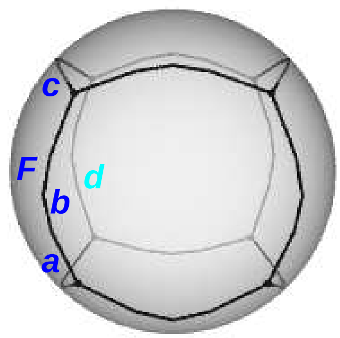


Figura 8.3: Transbordamento dos elementos próximos às arestas da projeção.

8.1.2 Visualização do processo de otimização

Para a visualização da configuração em tempo real durante o processo de otimização, utilizamos um módulo baseado na biblioteca *ScreenPlot* (um modelo *wireframe* simples implementado em *Modula-3* por Stolfi [36]), incorporado ao programa de otimização `OptShape` (veja apêndice A.1). A figura 8.4 mostra várias etapas na otimização do mapa `8-cell` obtidas com esta biblioteca.

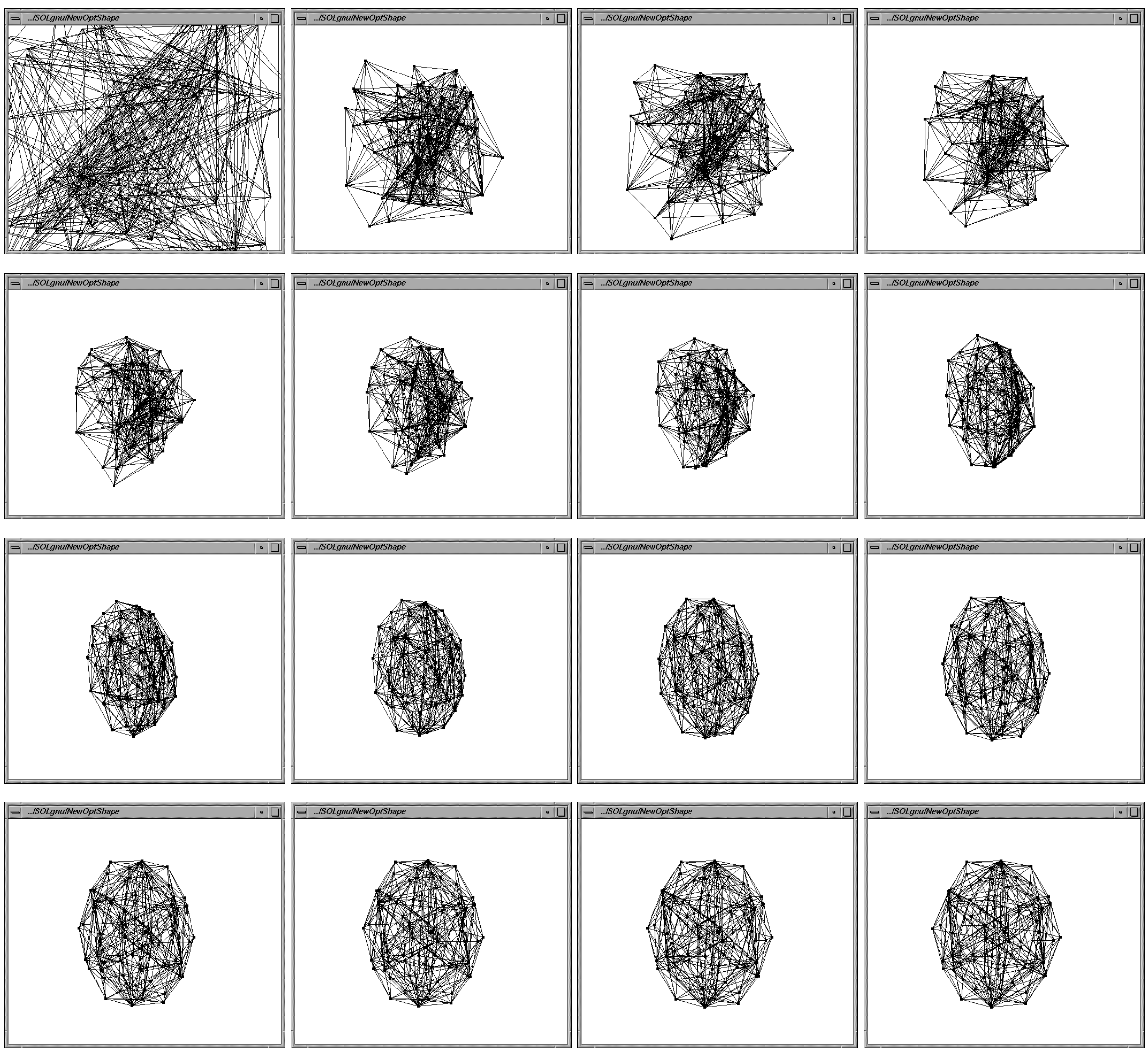


Figura 8.4: Otimização de uma triangulação que é a subdivisão baricêntrica do mapa `8-cell`. Na sequência: a configuração aleatória inicial e configurações intermediárias após, 5, 10, 20, 50, 65, 70, 75, 90, 100, 150, 200, 225, 250, 300 e 600 passos do método `Grad` na avaliação da energia $\mathcal{E}_{spring}$. A sequência deve ser lida por linhas, de esquerda para a direita e de cima para baixo.

8.1.3 Visualização semi-interativa após projeção para o $\mathbb{R}^3$

Para as ferramentas de visualização que só se aplicam a modelos tridimensionais nós implementamos para este último propósito o programa `ProjectTo3D` (veja no apêndice A.1), que suporta projeção paralela ou perspectiva. A teoria desta projeção está descrita na seção 8.2.

Algumas das ferramentas que utilizamos para visualização de objetos 3D são:

- *X3D* [32], um visualizador *wireframe* baseado na biblioteca gráfica *X11*.
- *Geomview 1.6.1 OpenGL* [20], um visualizador interativo de objetos geométricos em diversos formatos (p. ex. linhas poligonais) disponível em diversas plataformas.

A figura 8.5 ilustra uma triangulação do mapa `8-cell` renderizada com o programa *X3D*. Uma limitação severa deste programa é que o elemento básico de desenho é o polígono e portanto ele não permite realçar os vértices e arestas do mapa original $\mathcal{C}$.

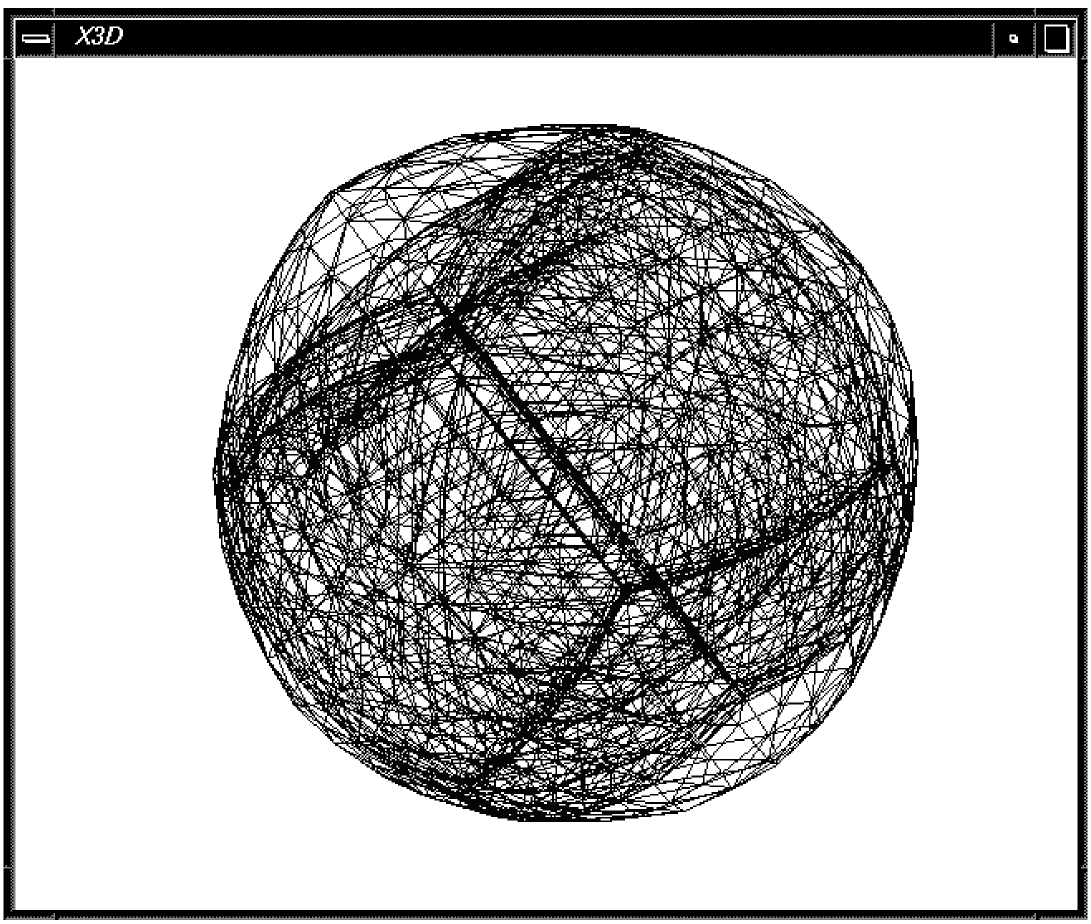


Figura 8.5: Realização do mapa `8-cell` renderizada com o programa *X3D*.

A figura 8.6 ilustra uma triangulação do mapa `8-cell`, visualizada com o programa *Geomview*. Em contraste com o programa *X3D*, este programa permite o uso de técnicas avançadas de visualização, tais como faces semi-transparentes e fontes de luz.

O *Geomview* permite também a produção de imagens estereográficas 3D (bi-coloridas ou polarizadas), como ilustrado na seção 8.3.8. Por outro lado, o *Geomview* também utiliza o polígono como elemento básico de modelagem, e portanto não pode facilmente realçar arestas e vértices do mapa original.

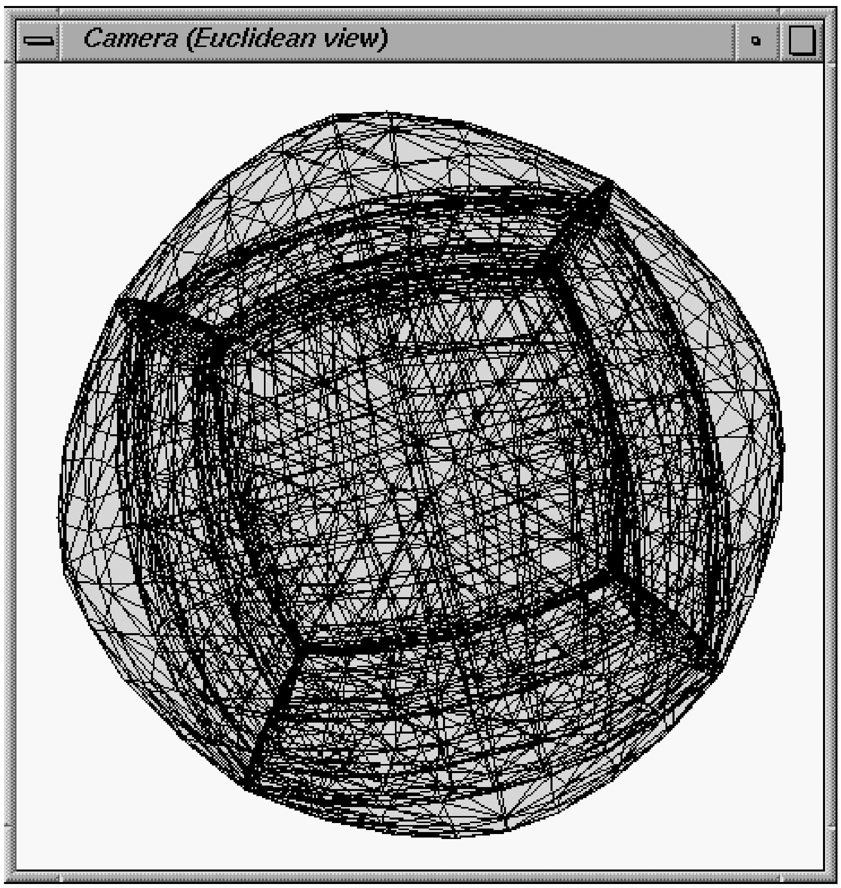


Figura 8.6: Realização do mapa `8-cell` renderizada com o programa *Geomview*, exibindo faces semi-transparentes.

Apesar de suas limitações, os programas *X3D* e *Geomview* são bastante úteis na prática, uma vez que fornecem uma visão geral interativa da projeção no $\mathbb{R}^3$ do modelo geométrico do mapa $\mathcal{C}$. Observe-se, não entanto, que elas não permitem alteração na posição do observador no $\mathbb{R}^4$.

8.1.4 Visualização com traçado de raios

Para produzir imagens de alta qualidade dos modelos tridimensionais, usamos o programa de traçado de raios *POV-Ray* (versão 3.1a), que permite o uso de recursos de foto-realismo como texturas semi-transparentes e sombras.

Uma vez que nosso objetivo é mostrar a estrutura do mapa dado $\mathcal{C}$, em vez da triangulação $\mathcal{T}$ do mesmo, nós renderizamos cada k-simplexo de $\phi(\mathcal{T})$ com estilo diferente, dependendo de suas dimensões e da dimensão do elemento de $\mathcal{T}$ que o contém. Em particular, na versão mais simples desta técnica, os vértices de $\mathcal{T}$ que são também vértices de $\mathcal{C}$ são renderizados como esferas opacas. Os vértices e arestas de $\mathcal{T}$ que constituem arestas de $\mathcal{C}$ são renderizados como cadeias de esferas pequenas e cilindros finos que conectam as esferas-vértice de $\mathcal{C}$. Os triângulos de $\mathcal{T}$ que pertencem a faces de $\mathcal{C}$ são renderizados com aspecto semitransparente. Os demais elementos da triangulação $\mathcal{T}$ são omitidos nas imagens (com exceção das faces de silhueta; vide seção 8.3.1).

A figura 8.7 ilustra esta técnica com uma triangulação do mapa `8-cell`, cuja projeção em *wireframe* pode ser vista na figura 8.2. Note-se que os elementos originais do mapa $\mathcal{C}$ — vértices, arestas, faces — são claramente visíveis nesta imagem, que deve ser comparada com as figuras 8.2, 8.5 e 8.6.

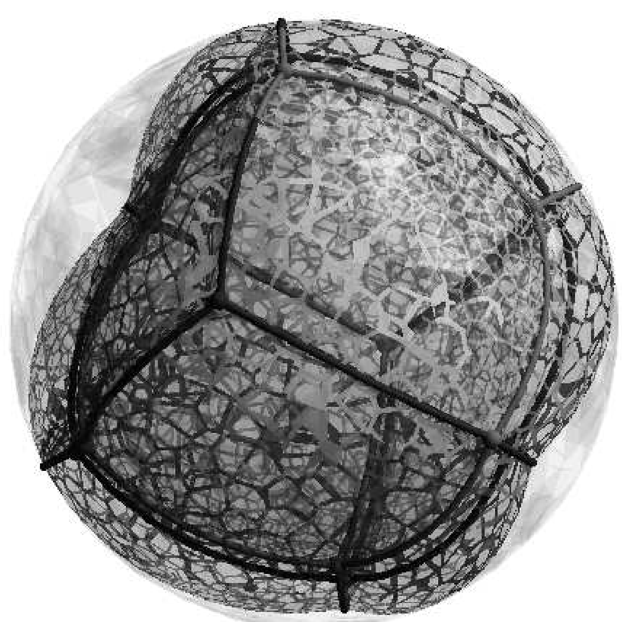


Figura 8.7: Imagem do mapa `8-cell` produzida com o programa *POV-Ray*.

8.2 Projeção perspectiva do $\mathbb{R}^m$ para o $\mathbb{R}^3$

A projeção do modelo $\phi(\mathcal{T})$ do $\mathbb{R}^m$ para o $\mathbb{R}^3$ é uma etapa essencial para todas as técnicas anteriores. Verifica-se que tal projeção pode ser decomposta numa projeção paralela π de $\mathbb{R}^m$ para $\mathbb{R}^4$, seguida de uma projeção paralela ou perspectiva ρ de $\mathbb{R}^4$ para $\mathbb{R}^3$.

Em coordenadas homogêneas, uma projeção perspectiva ou paralela ρ: $\mathbb{R}^4 \to \mathbb{R}^3$ é dada por uma matriz 5×5. As 5 coordenadas homogêneas $[v, w, x, y, z]$ resultantes do produto determinam as três coordenadas cartesianas $x/w, y/w, z/w$ do ponto em questão e a distância v/w do ponto ao hiperplano de projeção no $\mathbb{R}^4$.

A matriz de projeção pode ser especificada por parâmetros análogos aos que definem a projeção perspectiva de $\mathbb{R}^3$ para $\mathbb{R}^2$ [8], a saber: a posição obs_4 do observador no $\mathbb{R}^4$, um ponto de interesse int_4 na cena, e dois vetores adicionais ($\hat{Y}_4$ e $\hat{Z}_4$). Os pontos obs_4 e int_4 definem a *linha de visão* para a cena 4D. Por definição, o *espaço de projeção* I é o hiperplano (sub-espaço tridimensional euclidiano) de $\mathbb{R}^4$ que é perpendicular à linha de visão e contém o ponto int_4. A *projeção perspectiva* de um ponto $p \in \mathbb{R}^4$ é então o ponto p' onde a reta que passa por obs_4 e p intercepta o hiperplano I. Por definição, o *sistema de coordenadas da imagem* tem como origem o ponto int_4 e eixos paralelos aos vetores:

- W= direção do vetor $int_4 - obs_4$, ou seja a direção normal ao espaço I.
- Y= direção da componente de $\hat{Y}_4$ ortogonal a W.
- Z= direção da componente de $\hat{Z}_4$ ortogonal a W e Y.
- X= vetor unitário ortogonal a W, Y, Z, tal que X, Y, Z, W é uma base positiva.

A projeção perspectiva consiste portanto em calcular as coordenadas w, x, y, z do ponto no sistema da imagem e multiplicá-las pelo fator $\alpha = d/(d - w)$, onde $d = |obs_4 - int_4|$ é a distância do observador ao hiperplano de projeção. (Note que $\alpha \to 1$ quando $d \to \infty$). A figura 8.8 tenta ilustrar os parâmetros do modelo de visualização 4D e a geração do sistema de coordenadas da imagem.

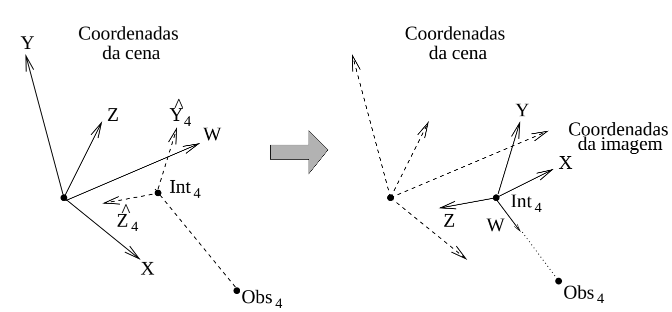


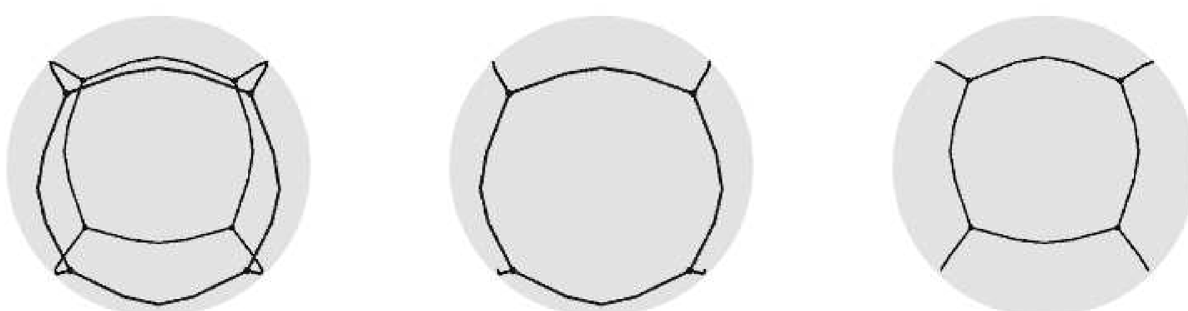
Figura 8.8: Parâmetros de visualização 4D e coordenadas da imagem.

A projeção em perspectiva geralmente produzirá projeções menores para segmentos que estejam mais afastados do ponto de observação 4D. Mais especificamente, segmentos situados no espaço I permanecem inalterados e segmentos paralelos a I são ampliados ou reduzidos conforme estejam na frente ou atrás de I (isto é, mais próximos ou mais afastados de obs_4 do que I).

No caso limite, em que o ponto obs_4 situa-se no infinito, obtém-se a *projeção paralela*, na qual o tamanho da projeção de um objeto não depende de sua distância do espaço de projeção I. Hollasch [16] argumenta que nenhuma das duas projeções — paralela ou perspectiva — é mais intuitiva do que a outra.

8.3 Eliminação de partes ocultas

A projeção de um mapa 3D no $\mathbb{R}^3$, em geral, consiste de dois ou mais sólidos interpenetrados, conectados um ao outro unicamente através de suas superfícies comuns; sendo que cada um destes sólidos está subdividido independentemente em células ou partes de células. Esta situação é análoga à projeção de uma malha fechada de triângulos do $\mathbb{R}^3$ para o $\mathbb{R}^2$: o resultado será geralmente dois ou mais polígonos planos sobrepostos, conectados unicamente através de suas fronteiras comuns, onde cada um destes polígonos está subdividido independentemente em faces ou partes de faces. Veja a figura 8.9.



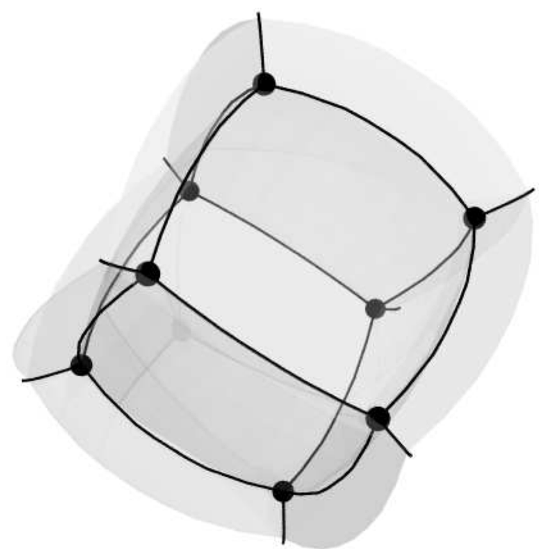
(a) (b) (c)

Figura 8.9: Um mapa 2D modelado como a partição da esfera, em projeção "completa" (a), em projeção da parte "visível" apenas (b) e em projeção da parte "oculta" apenas (c).

Mesmo um mapa tridimensional simples fica difícil de visualizar quando as diversas camadas da sua projeção são renderizadas conjuntamente. Portanto, quando projetamos o modelo do $\mathbb{R}^4$ para o $\mathbb{R}^3$, nós precisamos realizar a *eliminação de partes ocultas*, da mesma maneira como eliminamos partes ocultas quando projetamos malhas triangulares do $\mathbb{R}^3$ para o $\mathbb{R}^2$. A figura 8.10 ilustra esta técnica.

(a)

(b)



(c)

Figura 8.10: Uma realização do mapa `8-cell` em projeção "completa" (a), em projeção da parte "visível" apenas (b) e em projeção da parte "oculta" apenas (c). As faces de silhueta não são exibidas.

A visibilidade das células pode ser determinada, de modo geral, mediante traçado de raios no $\mathbb{R}^4$ [16], ou mediante uma versão tridimensional do método Z-buffer [33]. Como caso particular, se a triangulação no $\mathbb{R}^4$ é o 3-esqueleto de um politopo convexo, a visibilidade de cada tetraedro pode ser determinada através da orientação de sua projeção no $\mathbb{R}^3$. Este método é análogo ao método de remoção de faces ocultas que pode ser usado na projeção de poliedros convexos.

8.3.1 Faces de silhueta

Observe-se na figura 8.10 como a projeção das faces, arestas e vértices do mapa original $\mathcal{C}$ resulta numa imagem relativamente incompleta; em particular, não é possível visualizar a verdadeira extensão das células do mapa $\mathcal{C}$. A fim de superar este inconveniente, devemos exibir também os *triângulos de silhueta* de $\mathcal{T}$ — situados entre os tetraedros da "metade da frente" e os tetraedros "da metade de trás" do modelo projetado — mesmo quando estes triângulos não são parte de faces do mapa original $\mathcal{C}$.

Uma face triangular $f \in \phi(\mathcal{T})$ pertence à silhueta do modelo projetado se as projeções $\rho(t_1)$ e $\rho(t_2)$ dos dois tetraedros t_1, t_2 incidentes em f estão no mesmo lado do plano suporte de $\rho(f)$. Matematicamente, esta condição equivale a dizer que os simplexos ordenados $\langle a, b, c, u, o \rangle$ e $\langle a, b, c, v, o \rangle$ no $\mathbb{R}^4$ (antes da projeção) tem o mesmo sinal; onde a, b, c são os vértices do triângulo f, u e v são os vértices de t_1 e t_2 opostos à face f e o é a posição do observador no $\mathbb{R}^4$ (obs_4). Veja a figura 8.11.

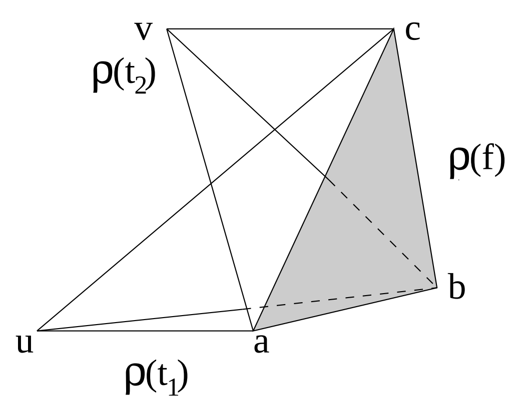


Figura 8.11: Face de silhueta.

A figura 8.12 ilustra a inclusão das faces de silhueta. Compare estas imagens com as da figura 8.10.

(a)

(b)

(c)

Figura 8.12: O mapa `8-cell` em projeção "completa" (a), em projeção da parte "visível" apenas (b), em projeção da parte "oculta" apenas (c), todas incluindo as faces de silhueta.

8.3.2 Tonalização por profundidade

Uma alternativa mais econômica à eliminação de partes ocultas, que não requer o cálculo da visibilidade das células, é variar a tonalidade e/ou brilho dos elementos projetados conforme sua distância do observador no $\mathbb{R}^4$. (Esta técnica é conhecida por *depth cueing* em inglês.) Veja-se por exemplo a figura 8.13, onde os elementos do mapa $\mathcal{C}$ que se encontram mais afastados do observador no $\mathbb{R}^4$ são renderizados com cores menos contrastantes.

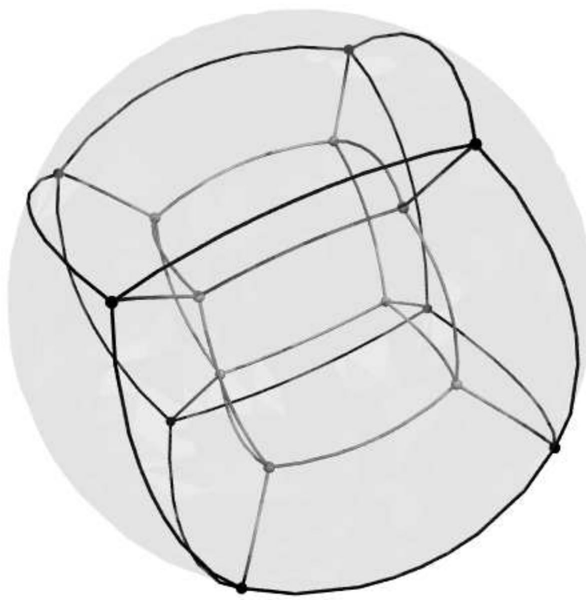
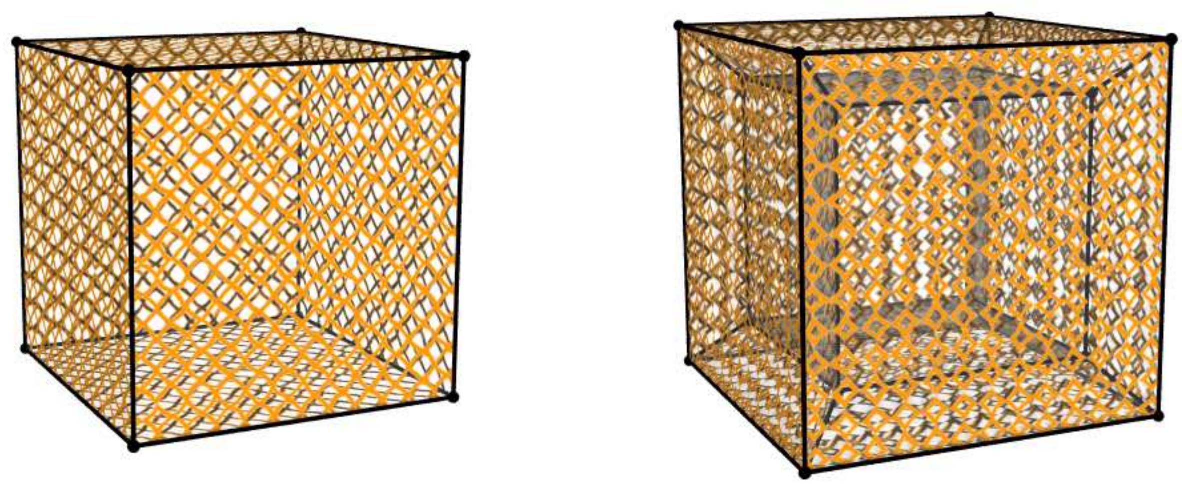


Figura 8.13: Realização do mapa `8-cell` com distância do observador no $\mathbb{R}^4$ evidenciada por meio de variação de contraste.

8.3.3 Faces gradeadas

Mesmo depois da eliminação de partes ocultas o modelo projetado no $\mathbb{R}^3$ ainda consiste de células (ou partes de células) coladas de maneira arbitrariamente complexa. Portanto, quando este modelo é por sua vez projetado no $\mathbb{R}^2$, ainda haverá em geral, muitas faces sobrepostas sobre o mesmo ponto da imagem.

Para tornar visível a existência e ordem dessas faces sobrepostas, podemos pintar cada face do mapa 3D com uma textura semitransparente, simulando uma grade ou malha. Veja a figura 8.14.



(a) (b)

Figura 8.14: Realizações dos mapas não triangulados `6-face` (a) e `8-cell` (b) onde as faces são representadas por grades, modeladas através de texturas.

A malha deve ser suficientemente aberta para que as várias camadas de faces sobrepostas sejam visíveis na imagem e a estrutura celular do modelo projetado possa ser apreciada; mas seus filamentos devem ser grossos o bastante para que, quando dois filamentos de faces diferentes se cruzam, seja possível perceber qual dos dois está mais próximo do observador (obs_3).

A simulação de grades por meio de texturas semi-transparemtes, tal como implementados no traçador *POV-Ray*, tem a desvantagem de que a textura não acompanhar os movimentos das faces no $\mathbb{R}^4$.

Uma outra solução, que é compatível com animação, consiste em modelar a grade como um objeto geométrico tridimensional, formado por cilindros finos ligados por esferas pequenas, como ilustrado na figura 8.15. Em conjunção com esta técnica, o uso de iluminação não uniforme e o cálculo de sombras próprias são importantes para evidenciar a distância e inclinação das várias faces.

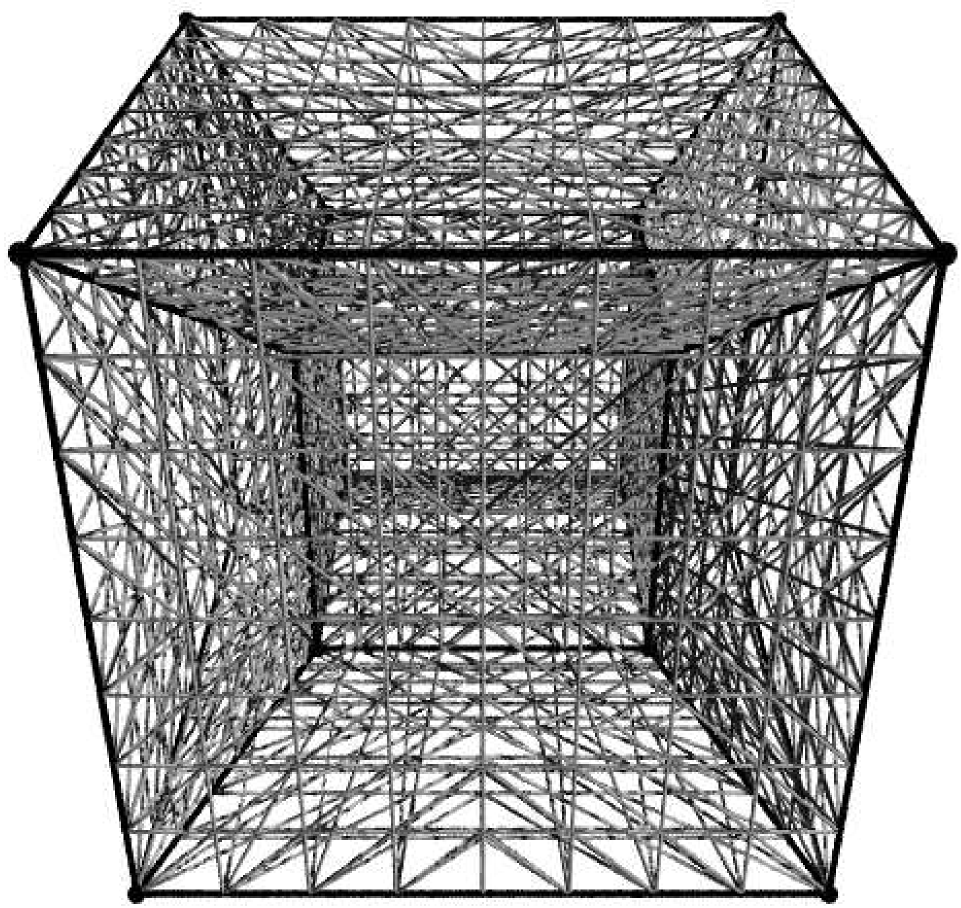
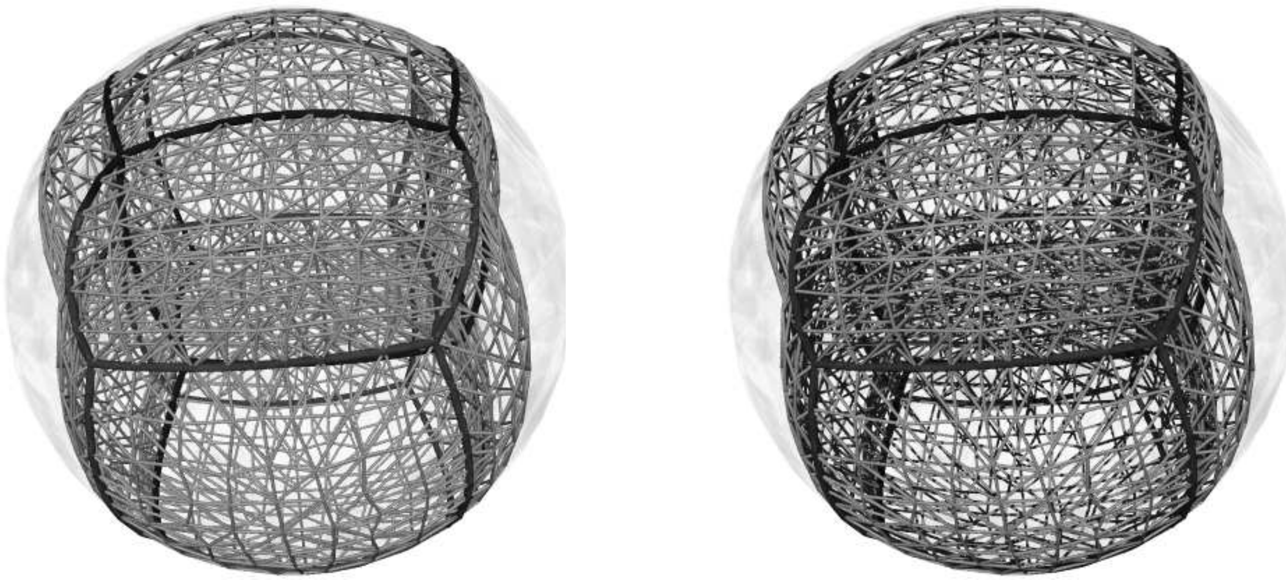


Figura 8.15: Realização do mapa `8-cell` onde as faces são representadas por grades modeladas por cilindros e esferas.

Outro detalhe que contribui para este objetivo é a presença de sombras projetadas por uma face sobre a outra. Compare-se por exemplo, as figuras 8.16(a) e 8.16(b).



(a) (b)

Figura 8.16: Efeito do uso de iluminação não uniforme e modelagem de faces gradeadas na visualização do mapa `8-cell`: sem sombras projetadas (a) e com sombras projetadas (b).

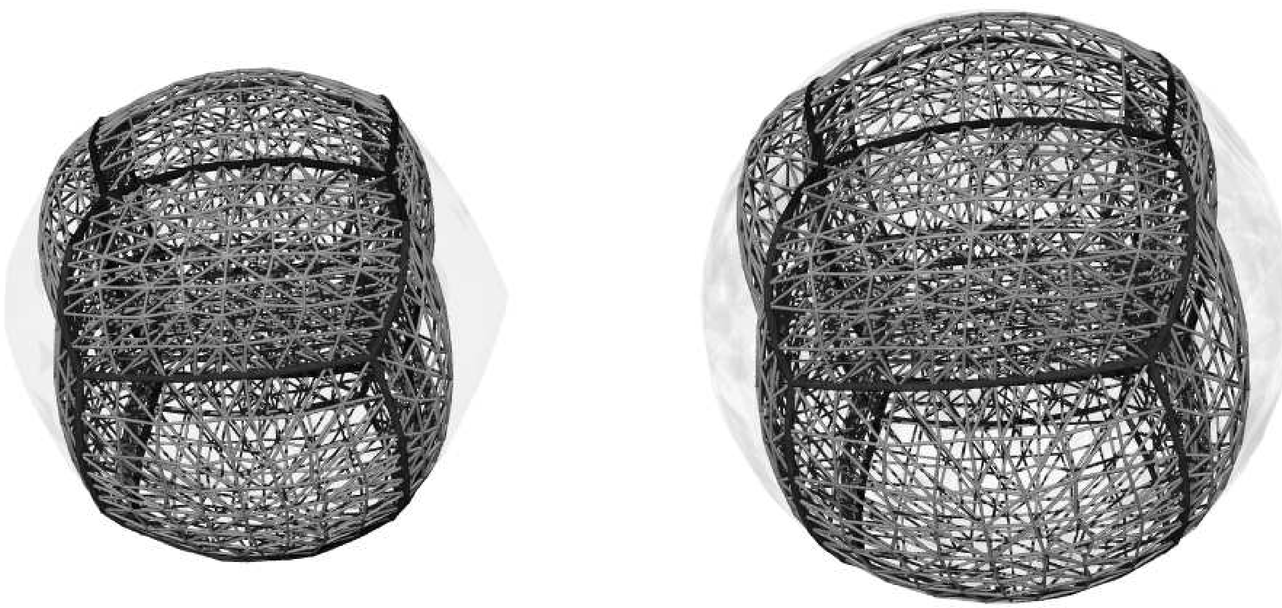
8.3.4 Normalização

Quando a variedade subjacente ao mapa original $\mathcal{C}$ tem a topologia da hiperesfera $\mathbb{S}^3$, é muitas vezes desejável que o modelo geométrico do mapa tenha também (aproximadamente) a geometria tradicional da mesma, o conjunto $\{p \in \mathbb{R}^4;\ |p| = 1\}$.

Verifica-se experimentalmente que se o mapa $\mathcal{C}$ tem uma topologia favorável e a função de energia usada leva em conta os ângulos entre células (por exemplo, pela inclusão de $\mathcal{E}_{curv3}$ com peso adequado) o modelo $\phi(\mathcal{T})$ otimizado é geralmente convexo (mais precisamente; é o esqueleto tridimensional de um politopo convexo do $\mathbb{R}^m$). Nesse caso, a forma do modelo já otimizado pode ser tornada mais próxima do $\mathbb{S}^3$ ideal por meio de uma translação que coloca o baricentro de $\phi(\mathcal{T})$ na origem, seguida de uma normalização que força a restrição $|\phi(v)| = 1$ para todo vértice $v \in \mathbf{V}\mathcal{T}$:

$$\phi(v) \leftarrow \phi(v) - b; \qquad \phi(v) \leftarrow \frac{\phi(v)}{|\phi(v)|}\,, \tag{8.1}$$

onde b é o baricentro de todos os vértices da triangulação $\mathcal{T}$. Veja a figura 8.17.. Idealmente, este objetivo pode ser alcançado também por meio de técnicas de otimização restritas; porém a solução trivial acima, quando possível, é muito mais simples de implementar.



(a) (b)

Figura 8.17: Uma realização do mapa `8-cell` antes (a) e depois (b) do processo de normalização por projeção sobre a hiperesfera $\mathbb{S}^3$.

8.3.5 Descolagem parcial

Quando a variedade subjacente ao modelo geométrico tem a topologia da esfera $\mathbb{S}^3$, é possível construir um modelo visualizável retirando-se uma das células do mapa e modelando o mapa sem borda resultante sobre a variedade $\kappa\mathbb{B}^3$. (Para que esta operação seja possível, é necessário que a célula retirada seja incidente a no máximo um lado de cada face.) O mapa assim obtido pode ser modelado diretamente no $\mathbb{R}^3$; quer por meio de um otimizador especializado, que não usa a quarta coordenada, quer utilizando-se uma energia mista com peso suficientemente grande para a energia de curvatura tridimensional $\mathcal{E}_{curv3}$.

O resultado $\phi(\mathcal{T})$ é uma subdivisão de um $\kappa\mathbb{B}^3$ imerso no $\mathbb{R}^3$. (Para que a imersão seja um mergulho, é necessário utilizar a energia de molas $\mathcal{E}_{spring}$, ou, para topologias mais complexas, a energia de orientação $\mathcal{E}_{orien}$, com refinamento suficientemente fino.) A célula que foi retirada corresponderia então ao complemento $\mathbb{R}^3 \setminus \phi(\mathcal{T})$, essa fronteira será a borda de $\phi(\mathcal{T})$.

A figura 8.18 ilustra esta técnica aplicada ao mapa `8-cell`. O modelo resultante possui 48384 tetraedros, 97152 faces, 58632 arestas e 9865 vértices.

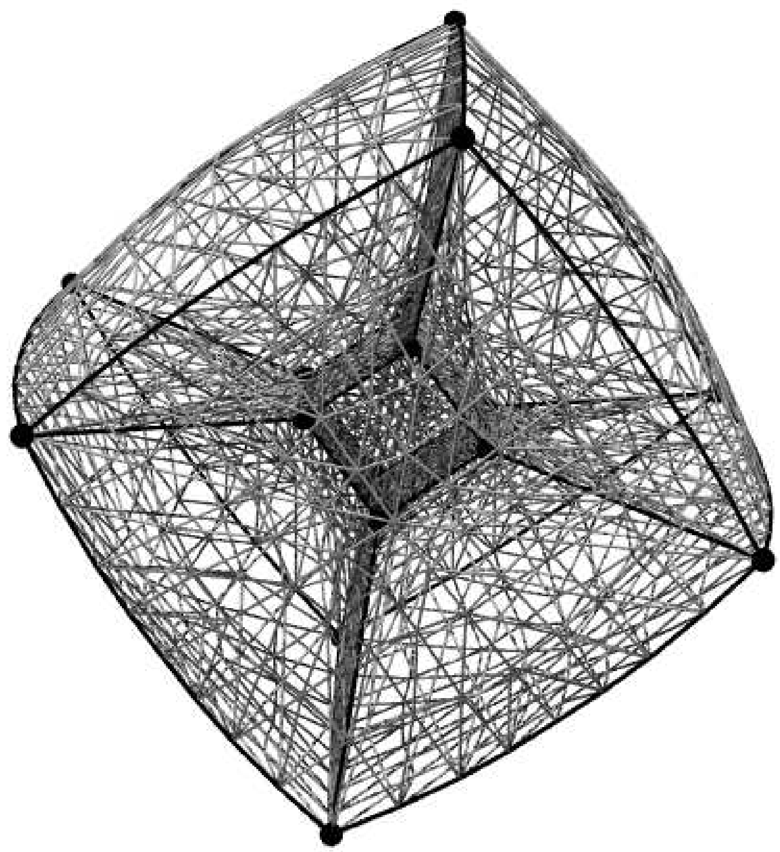


Figura 8.18: Uma projeção 3D do mapa `8-cell` com uma célula cúbica eliminada. A realização foi otimizada segundo a abordagem multiescala — consideramos inicialmente a energia $\mathcal{E}_{spring}$ para a triangulação baricêntrica e, nos refinamentos sucessivos as energias de curvatura $\mathcal{E}_{curv3}$ e $\mathcal{E}_{curv2}$.

8.3.6 Efeitos atmosféricos

Se nós somente renderizamos vértices, arestas e faces do mapa 3D, ignorando as células, nós obteremos uma visão "*paper-frame*" incompleta do mapa — tão incompleta quanto a visão *wireframe* de uma malha triangular. Para obter imagens mais completas dos mapas 3D, pode ser útil fazer que as células sejam também visíveis, preenchendo o interior com algum meio semi-transparente — tal como neblina, líquido colorido, luz incandescente, etc. Não chegamos a efetuar experimentos neste sentido.

8.3.7 Iluminando o hiperespaço

Hanson e Pheng [13] e Carey et al. [3] investigaram a extensão de modelos familiares de iluminação 3D para o espaço quadri-dimensional. Nestes modelos, uma fonte de luz no $\mathbb{R}^4$ ilumina cada ponto do objeto — incluindo o interior das células — com iluminação cuja intensidade depende da distância da fonte de luz e a inclinação entre a normais (no espaço 4D) das células e a direção da fonte de luz. Como no caso 3D, a luz 4D pode ser bloqueada ou modificada por objetos 3D. O uso pleno desta técnica requer que o interior das células seja modelado com efeitos atmosféricos.

8.3.8 Técnicas de visão estereoscópica

A estrutura 3D do modelo projetado pode ser apreciada também através de *estereogramas*, técnica de apresentação que utiliza nossa visão estereoscópica o que permite uma visualização muito mais clara da profundidade relativa dos elementos no modelo 3D projetado. De modo geral, a técnica consiste em preparar duas imagens do modelo 3D, vistas de cada olho do observador que são apresentadas separadamente para cada olho. Este objetivo pode ser conseguido de várias maneiras, por exemplo através de óculos com lentes coloridas ou polarizadas (vide figura 8.20), ou com olhar cruzado. Esta segunda técnica presupõe um certo treinamento e pode ser desconfortável para algumas pessoas (vide figura 8.21).

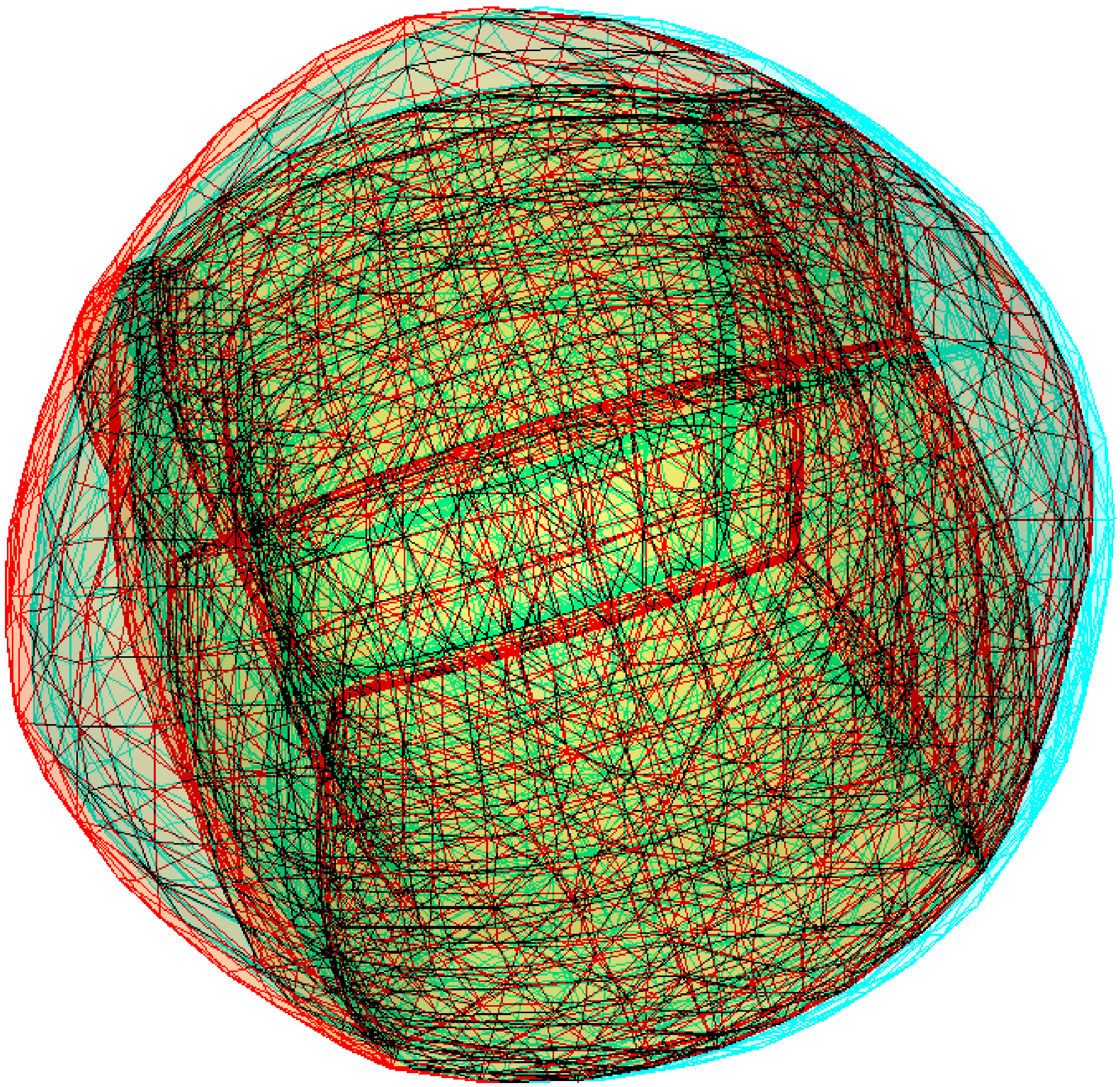


Figura 8.19: Imagem estereoscópica do mapa `8-cell` obtida com o programa *Geomview*.

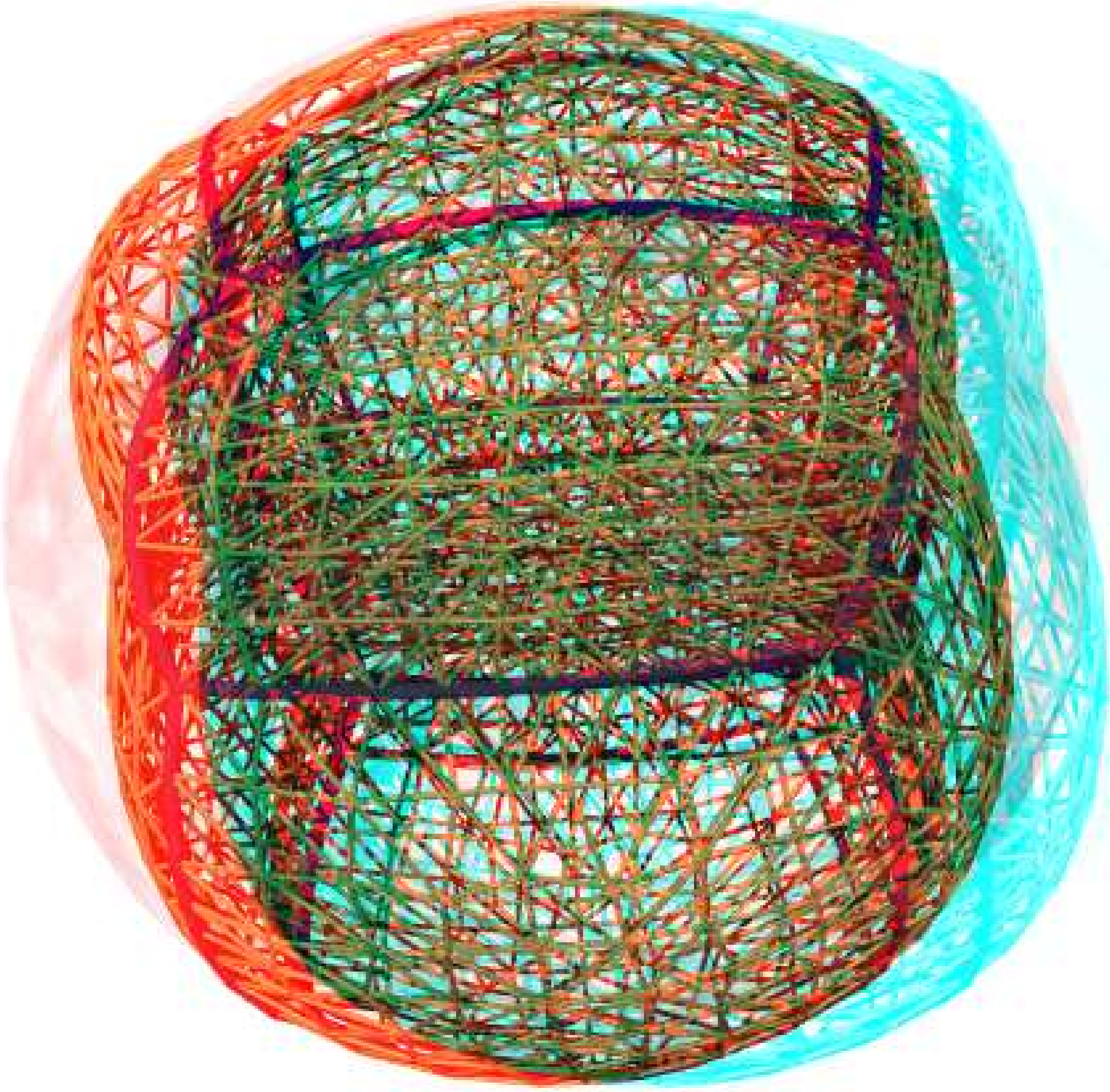


Figura 8.20: Imagem estereoscópica do mapa `8-cell` obtida com o programa *POV-Ray*.

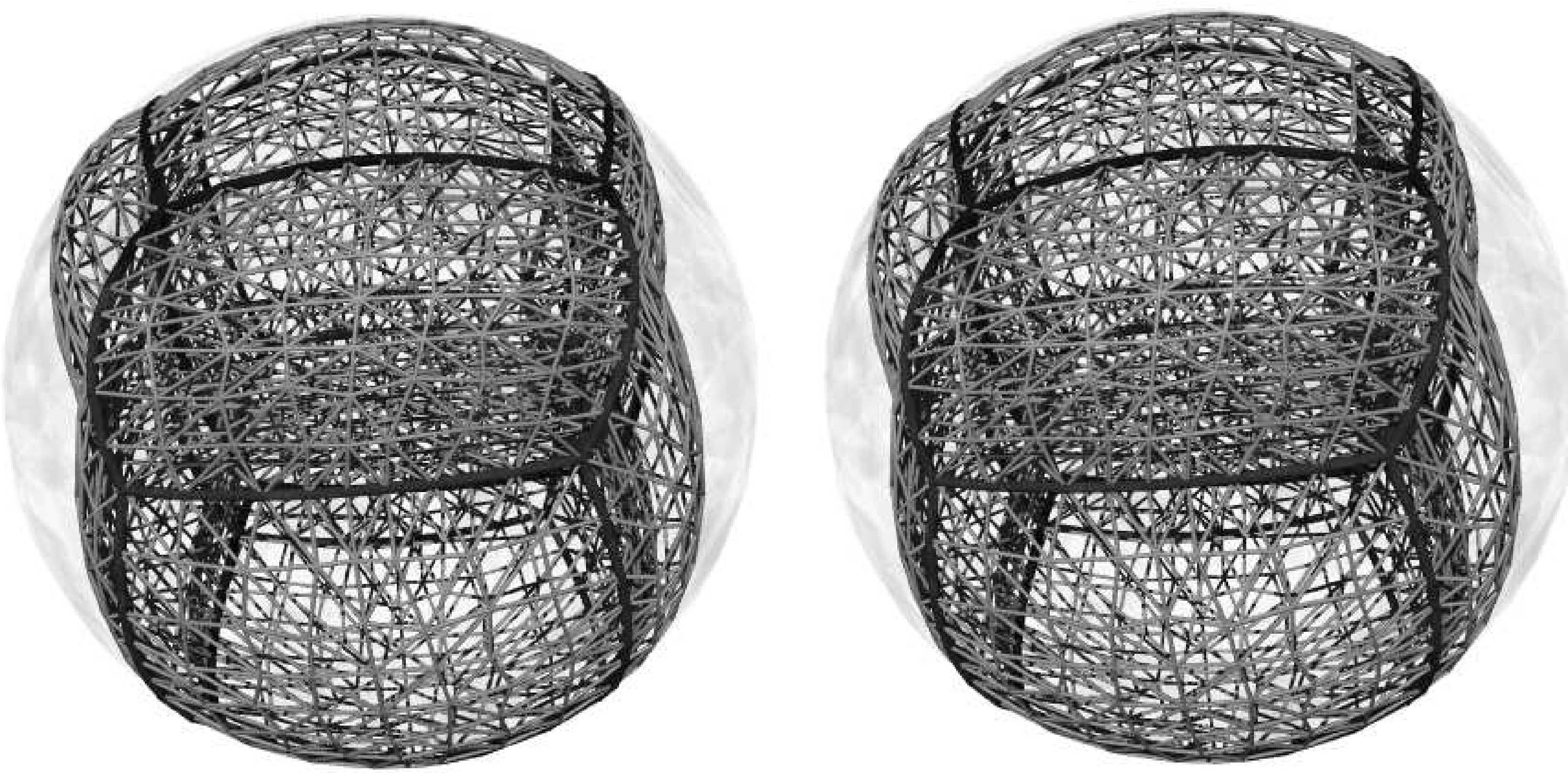


Figura 8.21: O mapa `8-cell` em par estéreo. Estas imagens devem ser apreciadas através da técnica de olhar convergente.

8.3.9 Movimentação da câmara no $\mathbb{R}^m$

Nenhuma técnica de visualização estática é tão efetiva, para exibir a estrutura 4D dos mapas 3D, quanto o uso de animações mostrando o efeito do movimento do observador (ou do objeto). Como mínimo, podemos mover o observador no $\mathbb{R}^3$ (obs_3) para obter uma melhor intuição da estrutura 3D do modelo projetado. Muito mais efetivo, contudo, é movimentar o observador no espaço $\mathbb{R}^4$ (obs_4) o que resulta em modificações da projeção 3D que não correspondem a nenhum movimento rígido desta última (figuras 8.22 e 8.23.)

Em particular, o movimento do observador no $\mathbb{R}^4$ (obs_4) altera a visibilidade 4D dos elementos do modelo $\phi(\mathcal{T})$. O efeito na projeção $\rho(\phi(\mathcal{T}))$ é que os elementos de $\phi(\mathcal{T})$ aparecem e desaparecem gradualmente onde eles encontram a superfície do modelo projetado (as faces de silhueta). As figuras 8.22 a 8.24 ilustram esta técnica.

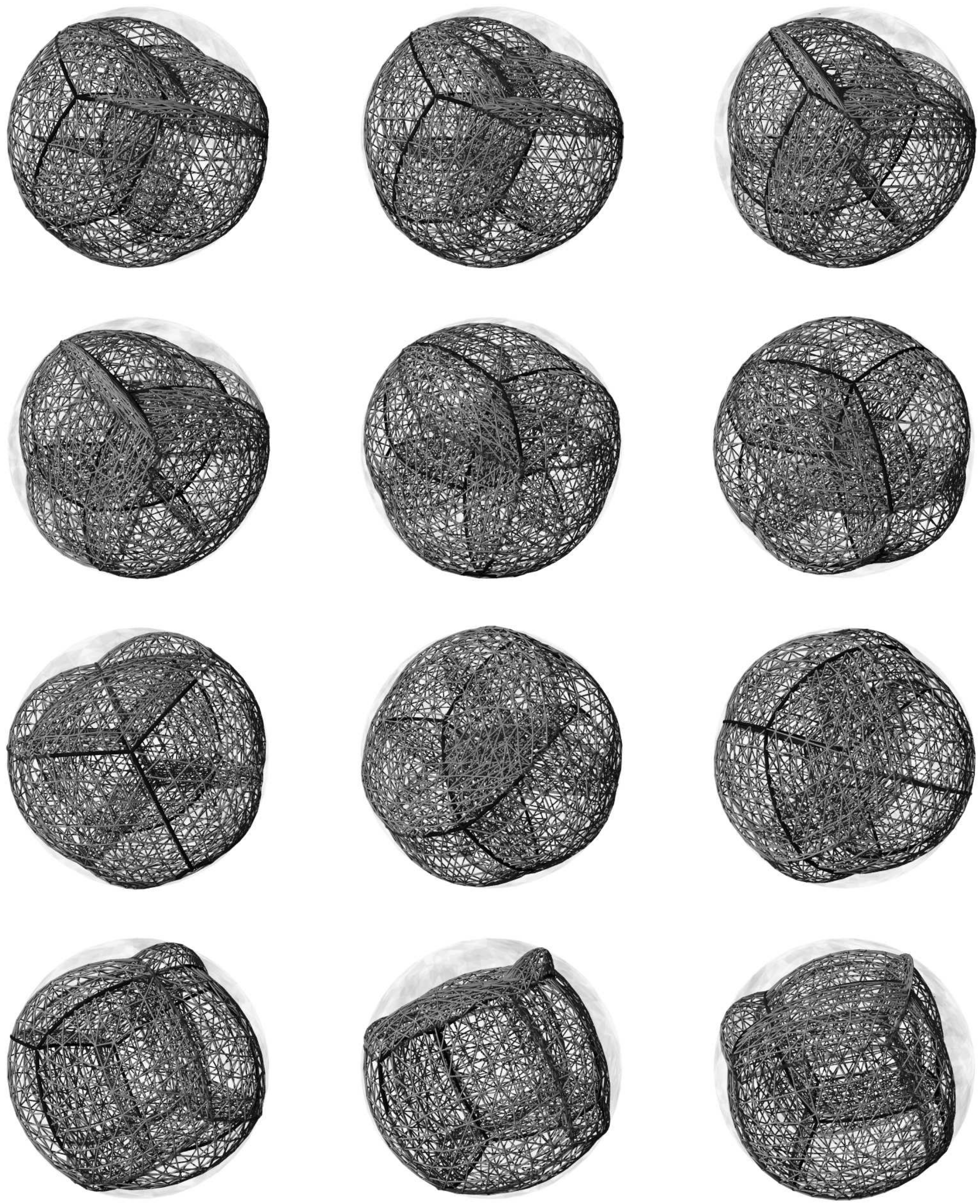


Figura 8.22: Projeções 3D do mapa `8-cell` para várias posições do observador no $\mathbb{R}^4$. A sequência deve ser apreciada por linhas, de esquerda para a direita e de cima para baixo.

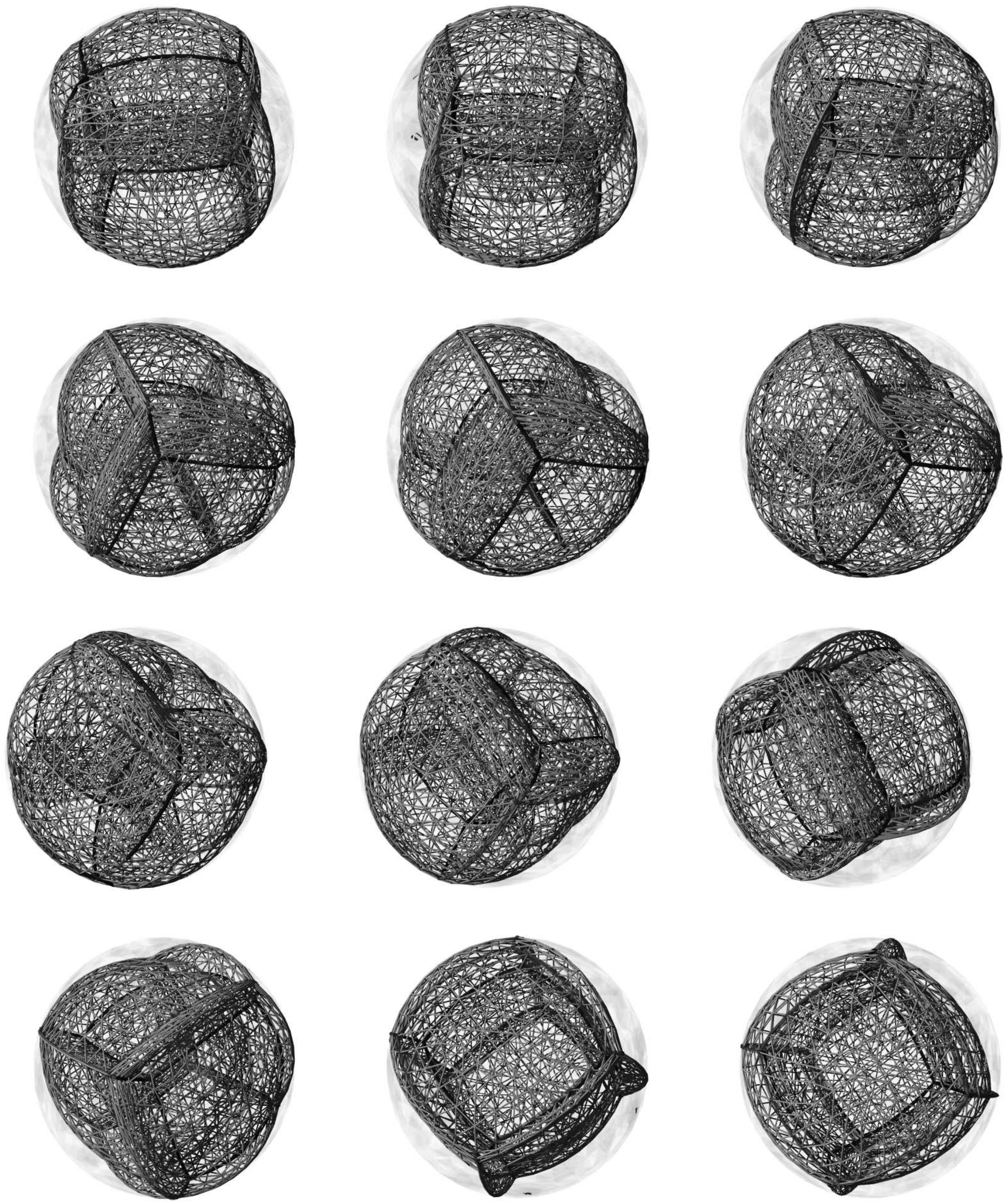


Figura 8.23: (Continuação.)

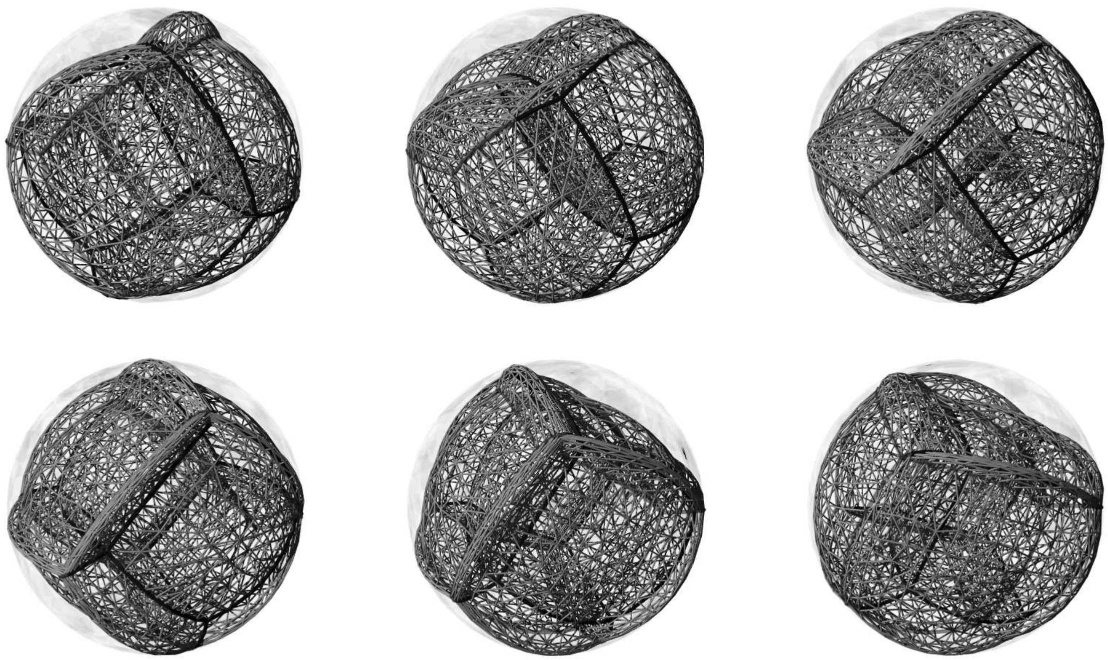


Figura 8.24: (Continuação.)

Capítulo 9

Exemplos: Politopos regulares

Para ilustrar nossa abordagem mostramos neste capítulo algumas realizações de mapas regulares; especificamente, os esqueletos 3D dos politopos convexos e regulares: `5-cell`, `8-cell` (hipercubo), `16-cell` (hiperoctaedro), `24-cell`, `120-cell` e `600-cell`. Incluimos também o mapa semi-regular `10-cell` (esqueleto do 4-simplexo truncado).

Modelos destes mapas poderiam ser facilmente construidos por subdivisão e projeção dos modelos geométricos dos politopos correspondentes encontrados na literatura. O objetivo deste capítulo é mostrar que nossas técnicas permitem obter os mesmos modelos regulares de maneira automática, apenas como resultado da minimização das funções energia vistas no capítulo 6. Além disso, espera-se que estes exemplos familiares ilustrem as técnicas de visualização que temos empregado e preparem o leitor para a compreensão dos mapas mais complexos do capítulo seguinte.

Todos os nossos testes utilizam o $\mathbb{R}^4$ como espaço de modelagem. As otimizações partiram de configurações aleatórias, onde cada coordenada cartesiana foi escolhida de maneira uniforme e independente do intervalo $[-1, +1]$.

9.1 Politopos regulares

9.1.1 O mapa `5-cell`

Lembramos que o mapa `5-cell`, ou pentacélula (vide seção 3.2.11), consiste de 5 tetraedros colados com topologia de $\mathbb{S}^3$. Uma vez que o mapa `5-cell` já é uma triangulação não degenerada, ele pode ser modelado diretamente, sem necessidade de subdivisão baricêntrica. A figura 9.1 mostra uma configuração ótima desse mapa para a energia $\mathcal{E}_{spring}$.

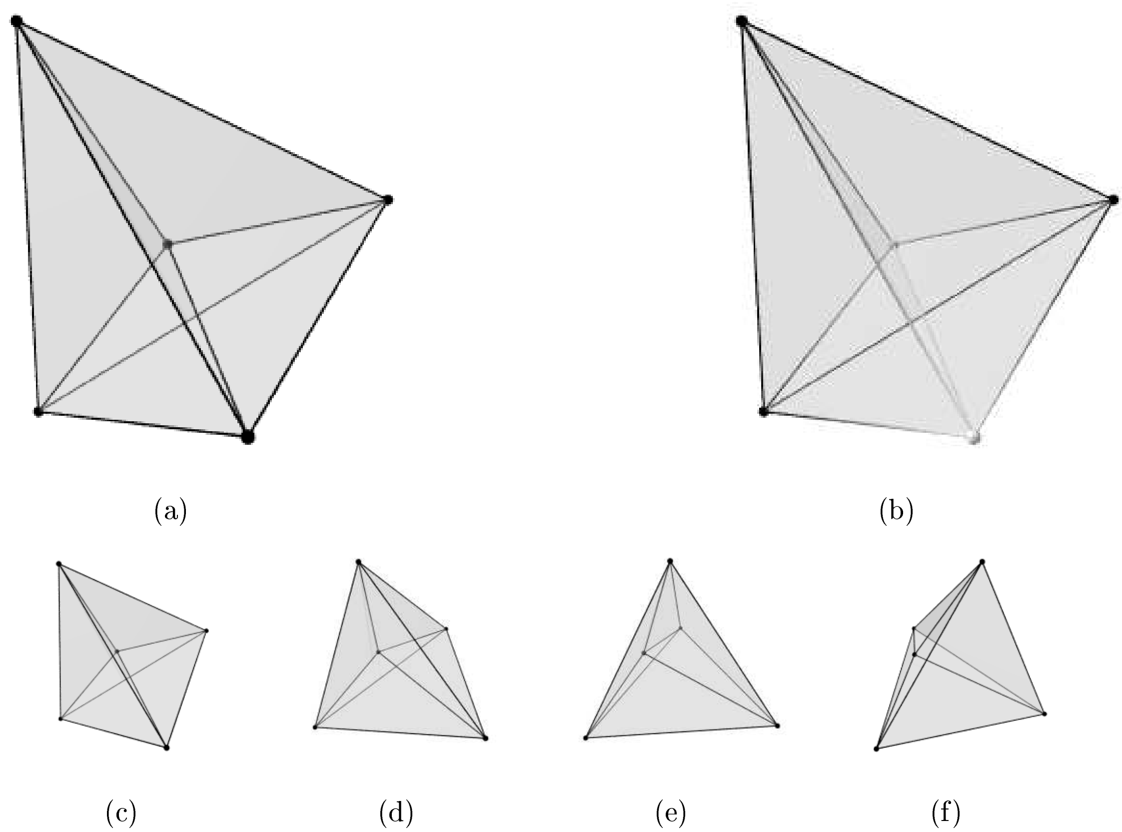


Figura 9.1: Projeção 3D de uma configuração ótima do mapa `5-cell`, exibindo faces parcialmente transparentes (a) e tonalização por profundidade (b). Várias vistas da projeção tridimensional (c), (d), (e) e (f). A realização foi obtida com 200 passos do método `Grad`.

Observe-se na figura 9.1(b) que o vértice central está mais afastado do observador no $\mathbb{R}^4$ e portanto é exibido com cores menos contrastantes. Observe também, que a técnica de tonalização por profundidade não é suficiente para mostrar a distância relativa dos elementos do mapa com respeito à posição do observador no $\mathbb{R}^4$. É necessário também que o tamanho dos mesmos diminua segundo essa distância.

A figura 9.1 é a representação tradicional mínima do mapa `5-cell`; mas, para melhor apreciar a regularidade da topologia, é interessante produzir modelos mais suaves, onde a variedade subjacente tenha geometria mais próxima da hiperesfera $\mathbb{S}^3$.

As imagens nas figuras 9.2 e 9.3 mostram projeções tridimensionais de um modelo geométrico mais refinado do mapa `5-cell`. O modelo é resultante da subdivisão baricêntrica e de dois estágios consecutivos do refinamento de ordem dois de cada tetraedro (veja seção 4.5). A triangulação resultante tem 3480 vértices, 20760 arestas, 34560 faces e 17280 tetraedros. A realização foi otimizada segundo a abordagem multi-escala — consideramos inicialmente a energia $\mathcal{E}_{spring}$ para as triangulações mais grosseiras e na etapa final a energia $\mathcal{E} = 0.99\,\mathcal{E}_{curv3} + 0.005\,\mathcal{E}_{curv2} + 0.005\,\mathcal{E}_{curv1}$.

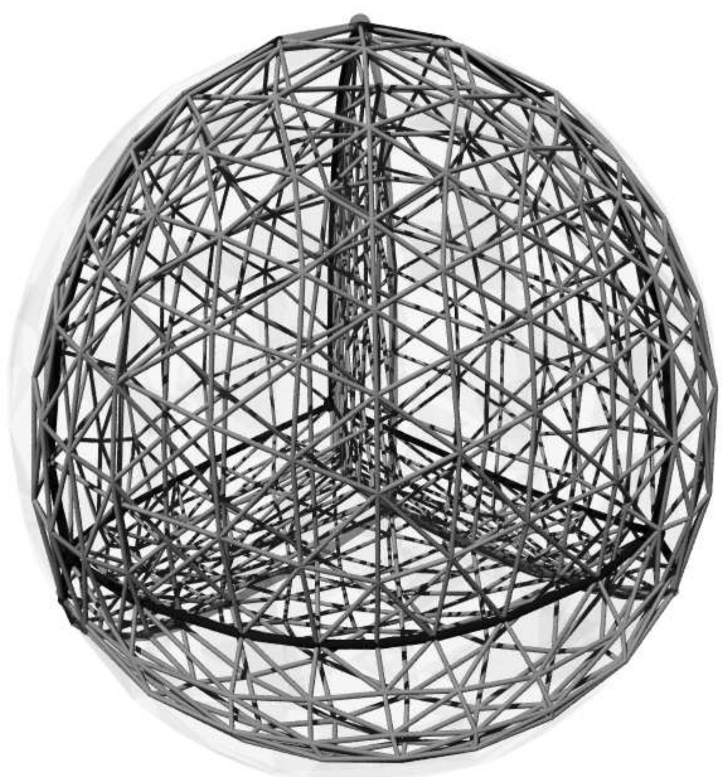


Figura 9.2: Projeção 3D completa de uma triangulação do mapa `5-cell`, visualizado com faces gradeadas e com tonalização dos elementos conforme sua distância do observador no $\mathbb{R}^4$. A configuração final foi otimizada com 500 passos do método `Grad`.

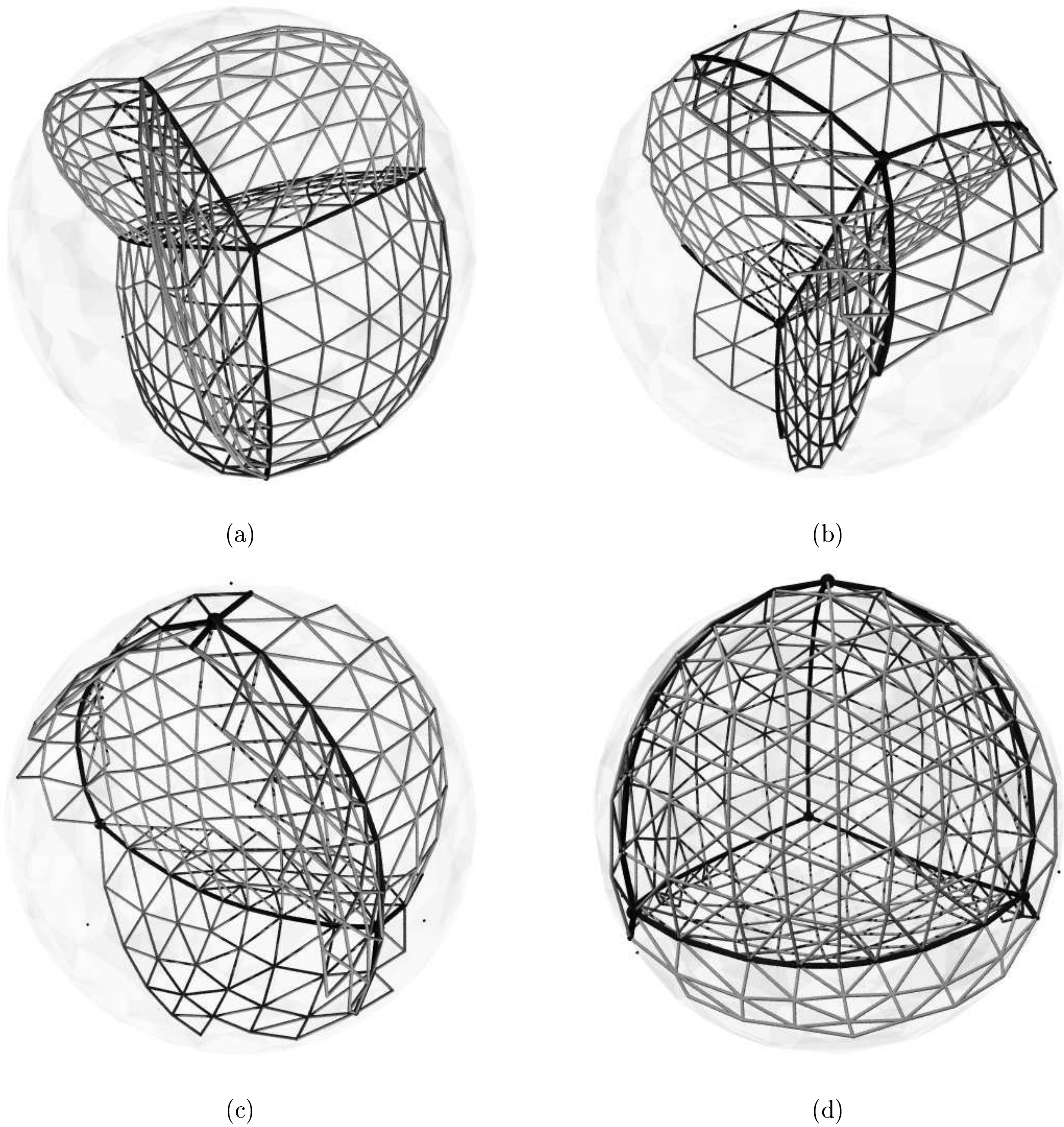


Figura 9.3: Projeções 3D da parte visível da triangulação do mapa `5-cell` ilustrado na figura 9.2 enfatizando as vizinhanças de um vértice (a), uma aresta (b), uma face (c) e uma célula (d) do mapa original.

As animações 9.4 a 9.6 mostram uma sequência de projeções do modelo acima, resultantes do movimento do observador no $\mathbb{R}^4$.

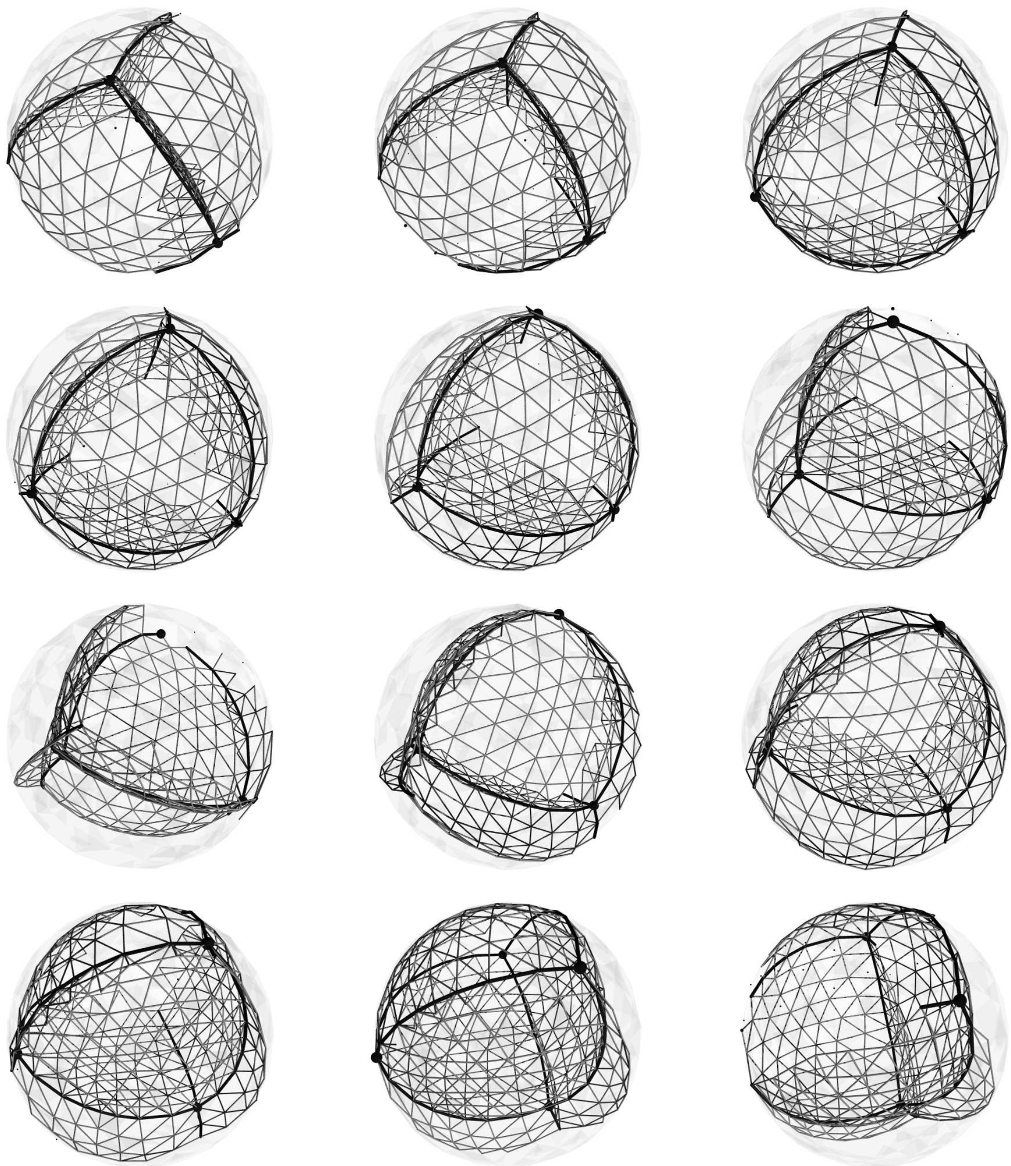


Figura 9.4: Projeções 3D da parte visível do mapa `5-cell` triangulado para várias posições do observador no $\mathbb{R}^4$. A sequência deve ser apreciada por linhas, de esquerda para a direita e de cima para baixo.

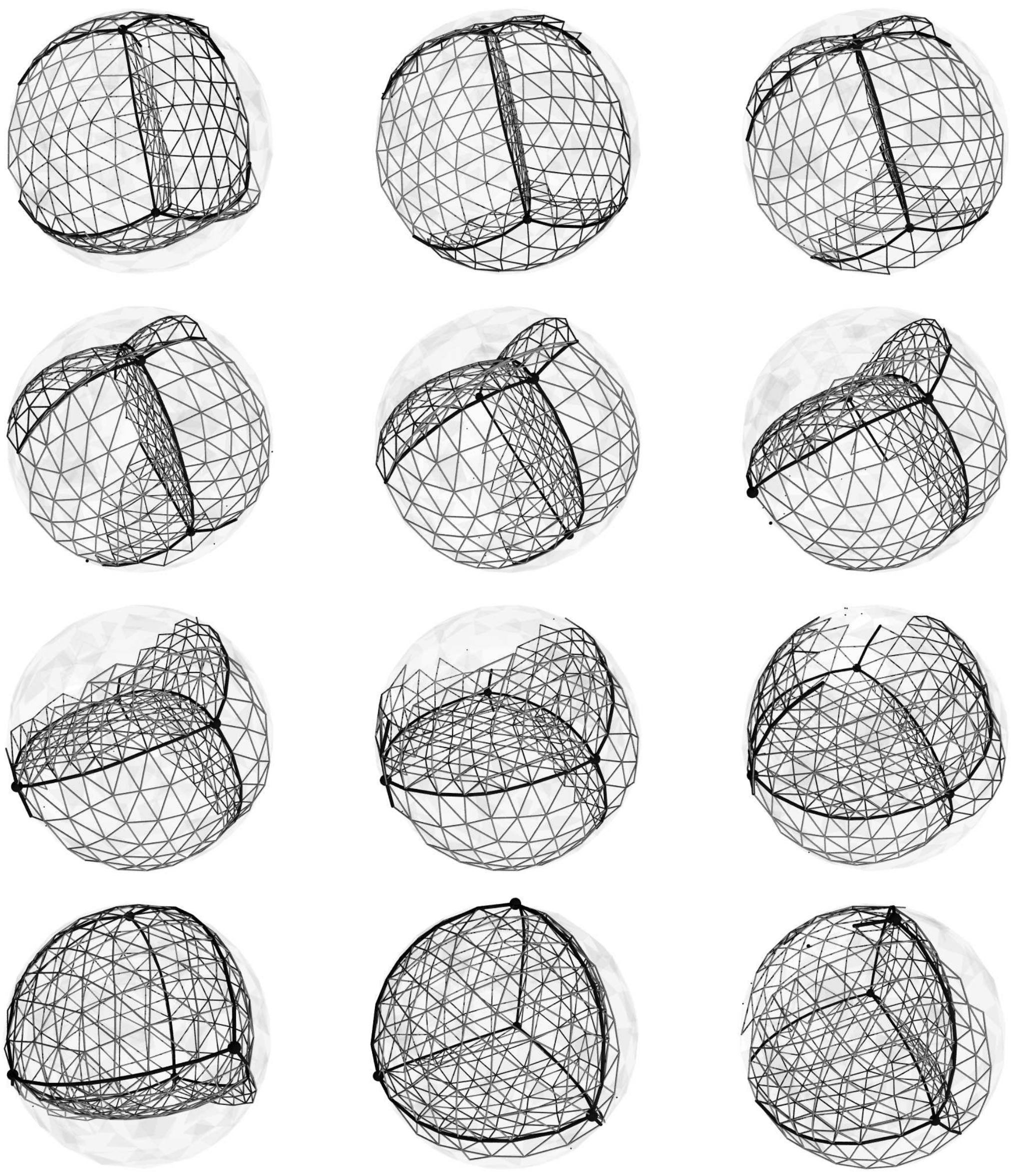


Figura 9.5: (Continuação.)

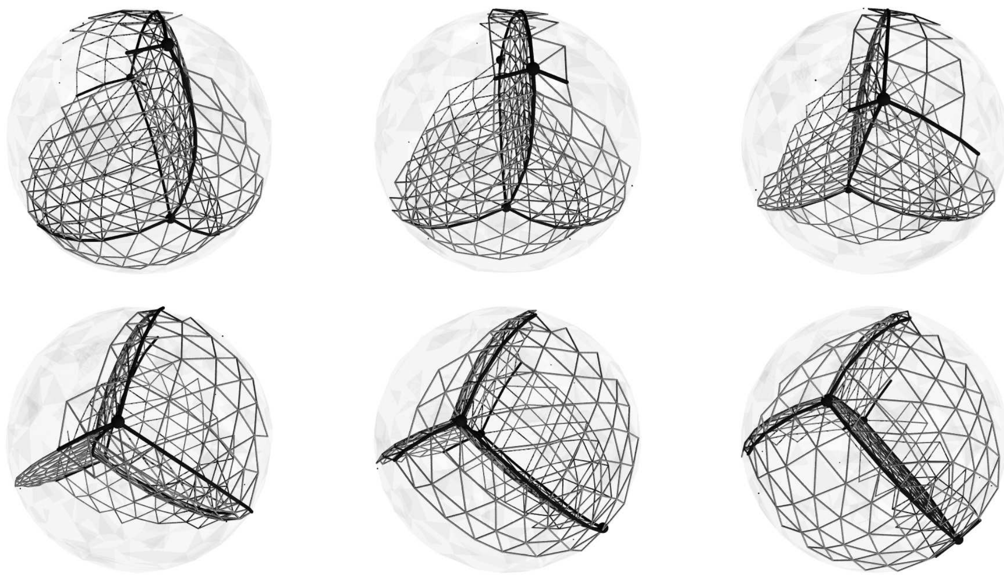


Figura 9.6: (Continuação.)

9.1.2 O mapa `8-cell`

No mapa `8-cell` ou hipercubo (vide seção 3.2.11), as células são cubos e portanto precisamos trabalhar com triangulações do mesmo. A figura 9.7 mostra uma representação geométrica da triangulação baricêntrica do mapa `8-cell` (com 80 vértices, 464 arestas, 768 triângulos e 384 tetraedros), que é otima para a energia $\mathcal{E} = 0.99\,\mathcal{E}_{curv2} + 0.01\,\mathcal{E}_{spring}$. Observe-se como a inclusão da energia $\mathcal{E}_{curv2}$ resulta em faces planas e arestas retílineas.

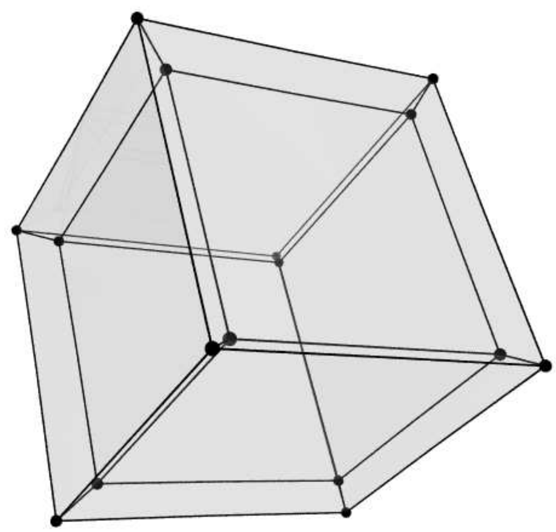


Figura 9.7: Configuração da triangulação baricêntrica do mapa `8-cell`, que foi otimizada para a energia $\mathcal{E} = 0.99\,\mathcal{E}_{curv2} + 0.01\,\mathcal{E}_{spring}$ com 500 passos do método `Grad`.

As figuras 9.8 e 9.9 mostram modelos geométricos mais refinados do mapa `8-cell`, após subdivisão baricêntrica e duas etapas do refinamento uniforme de ordem 2.

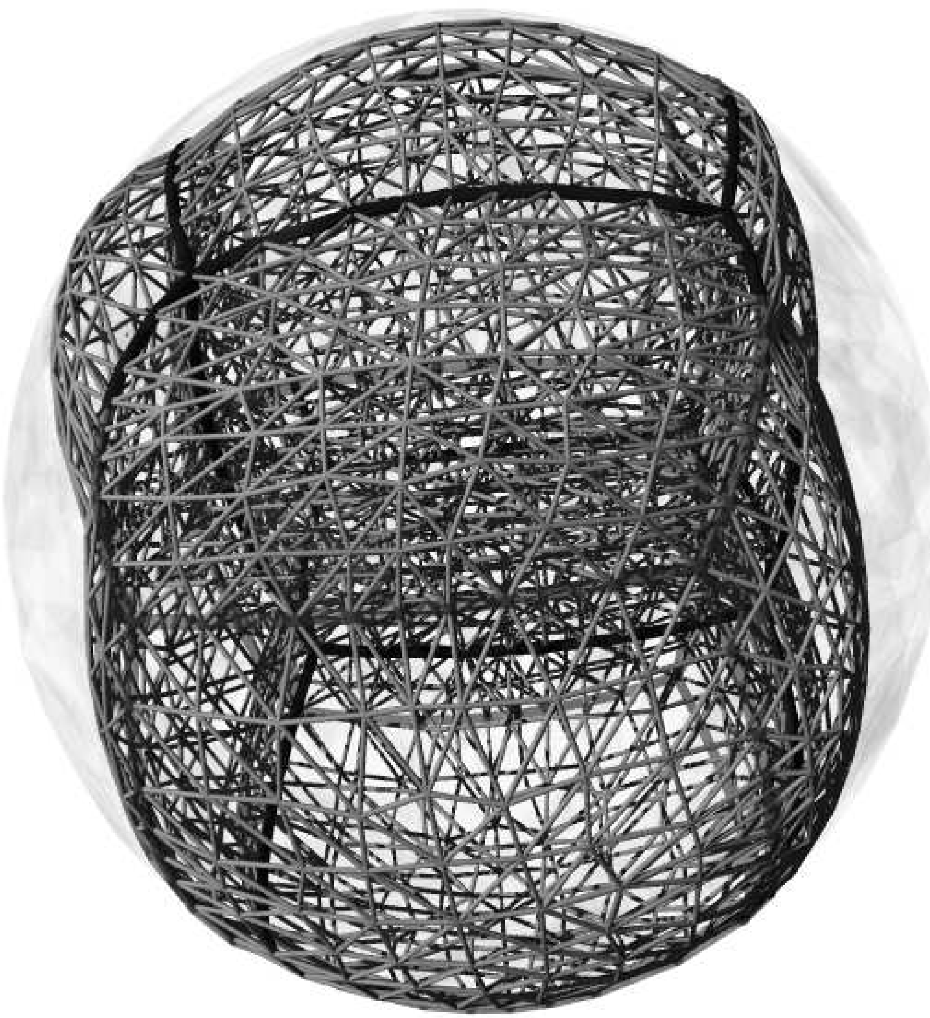
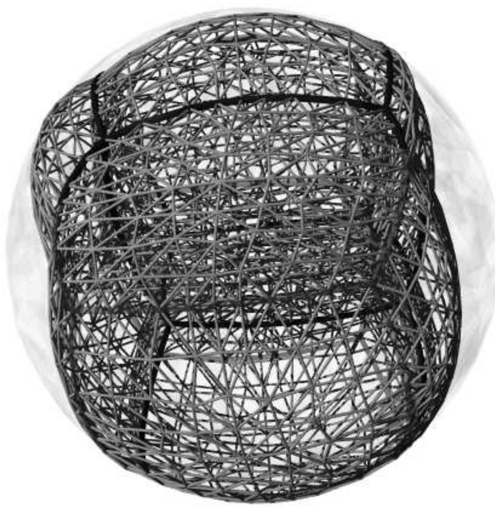
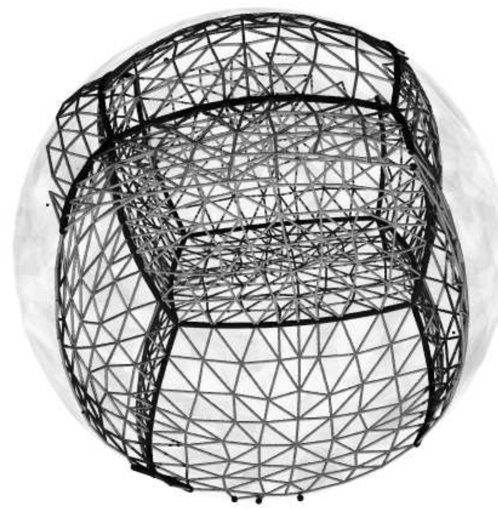


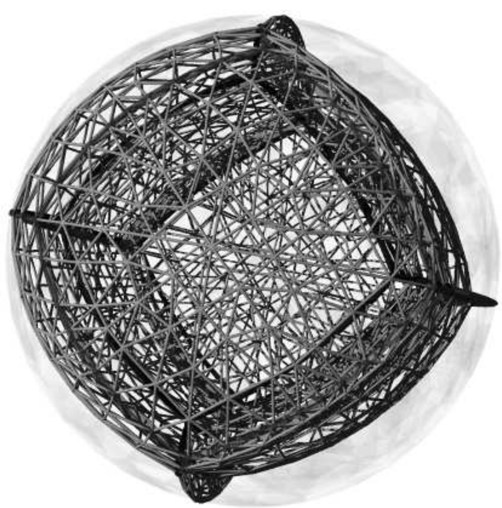
Figura 9.8: Mapa `8-cell` projetado no $\mathbb{R}^3$, visualizado com faces gradeadas e tonalização por profundidade. A realização foi obtida com 1000 passos do método `Grad` para a energia $\mathcal{E} = 0.90\ \mathcal{E}_{curv3} + 0.05\ \mathcal{E}_{curv2} + 0.05\ \mathcal{E}_{curv1}$. A triangulação possui 11072 vértices e 55296 tetraedros.



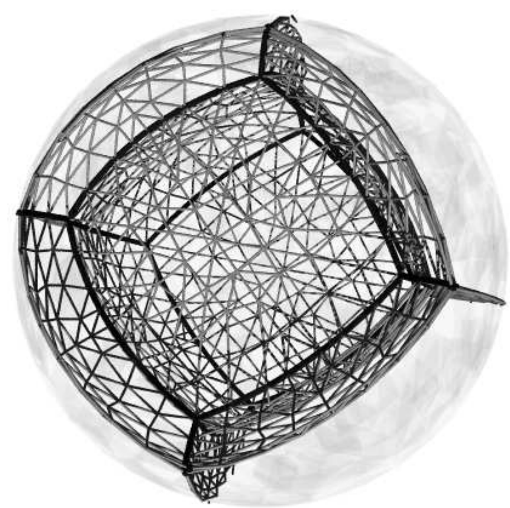
(a)



(b)



(c)



(d)

Figura 9.9: Duas projeções diferentes para o $\mathbb{R}^3$ de uma mesma realização do mapa `8-cell`, visualizado com faces gradeadas e faces de silhueta, em projeção "completa" (a) e (c) e projeção da parte "visível" apenas (b) e (d).

As animações 9.10 a 9.12 mostram o resultado do movimento do observador no $\mathbb{R}^4$.

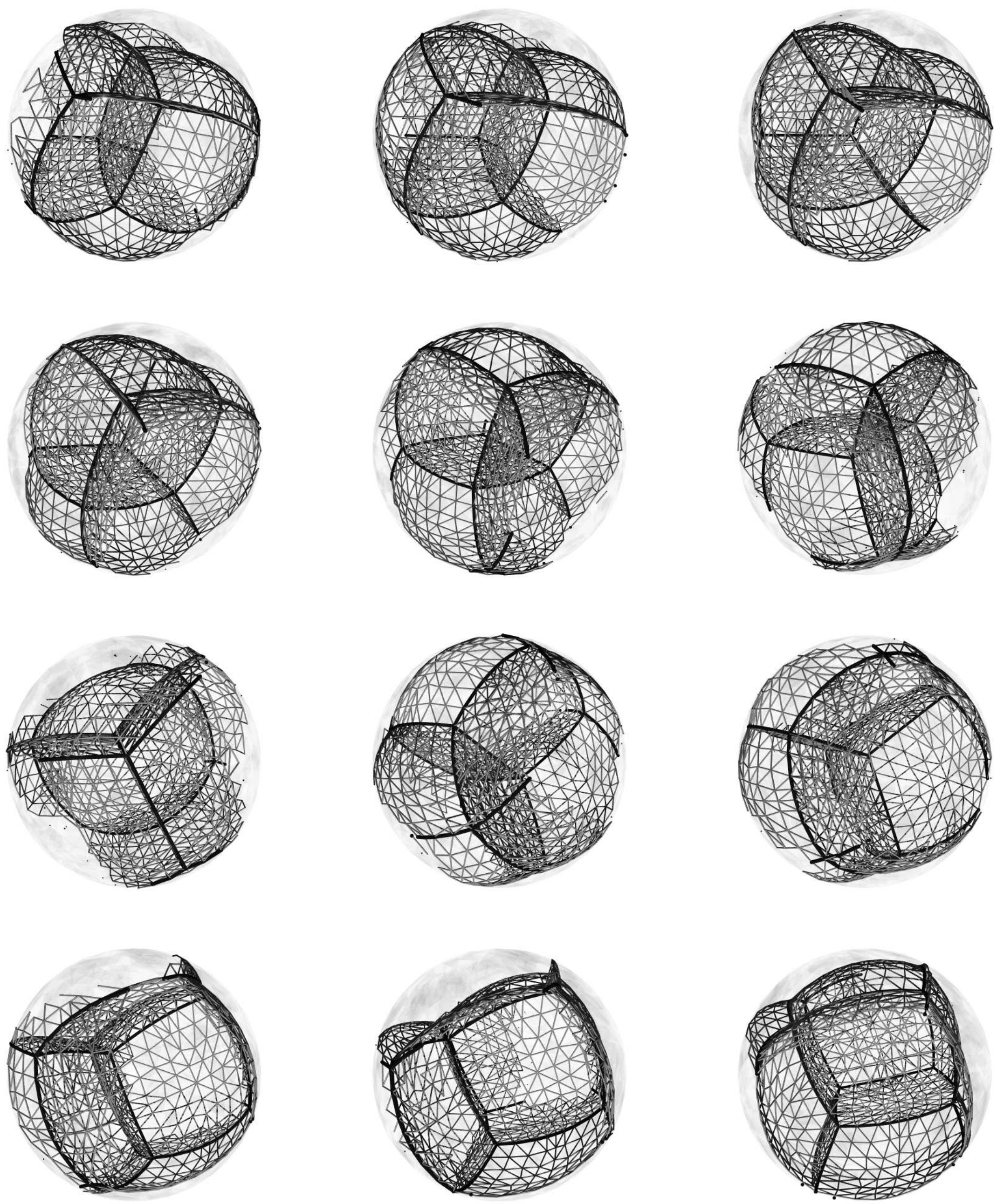


Figura 9.10: Projeções 3D da parte visível do mapa `8-cell` para várias posições do observador no $\mathbb{R}^4$. A sequência deve ser apreciada por linhas, de esquerda para a direita e de cima para baixo.

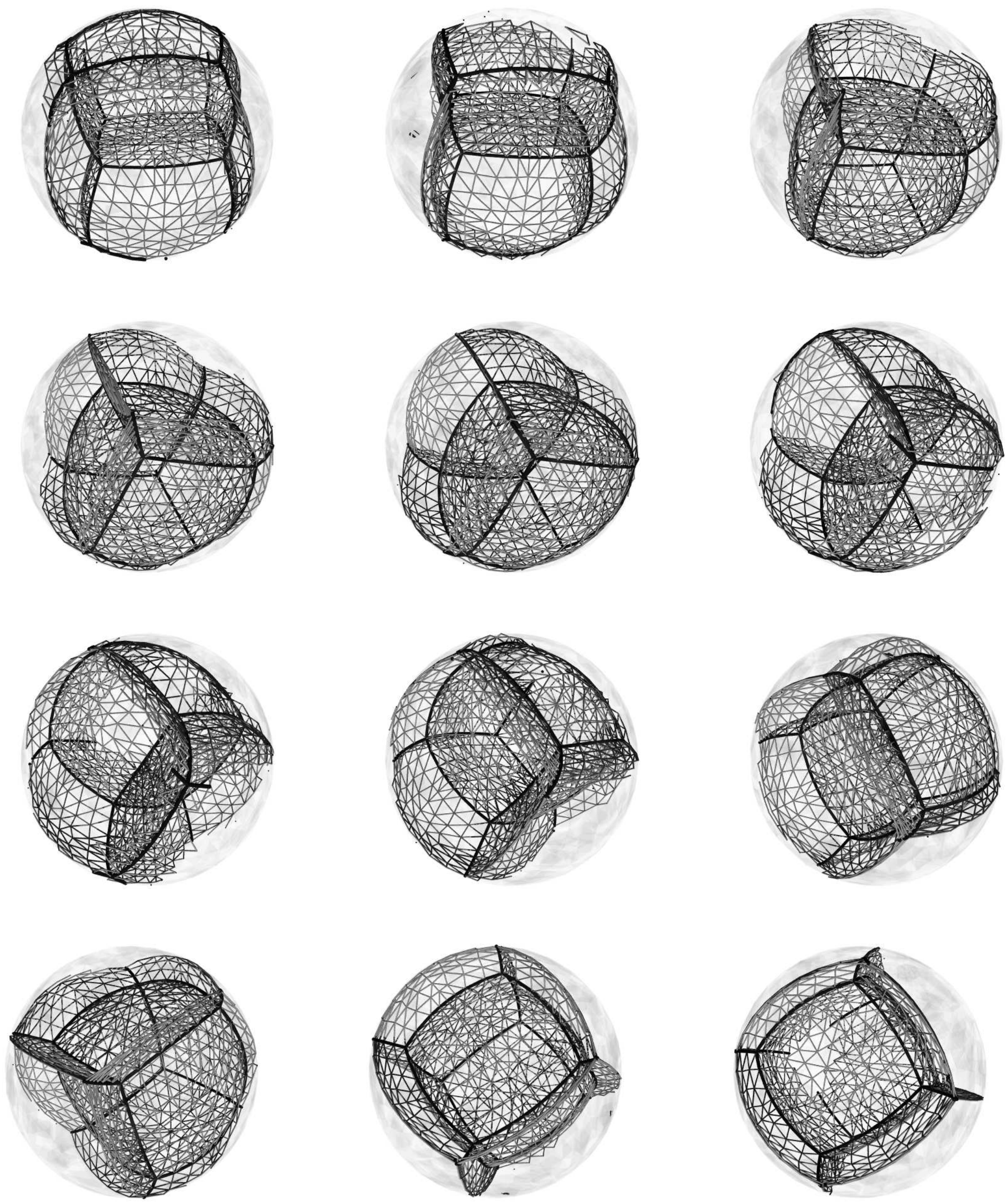


Figura 9.11: (Continuação.)

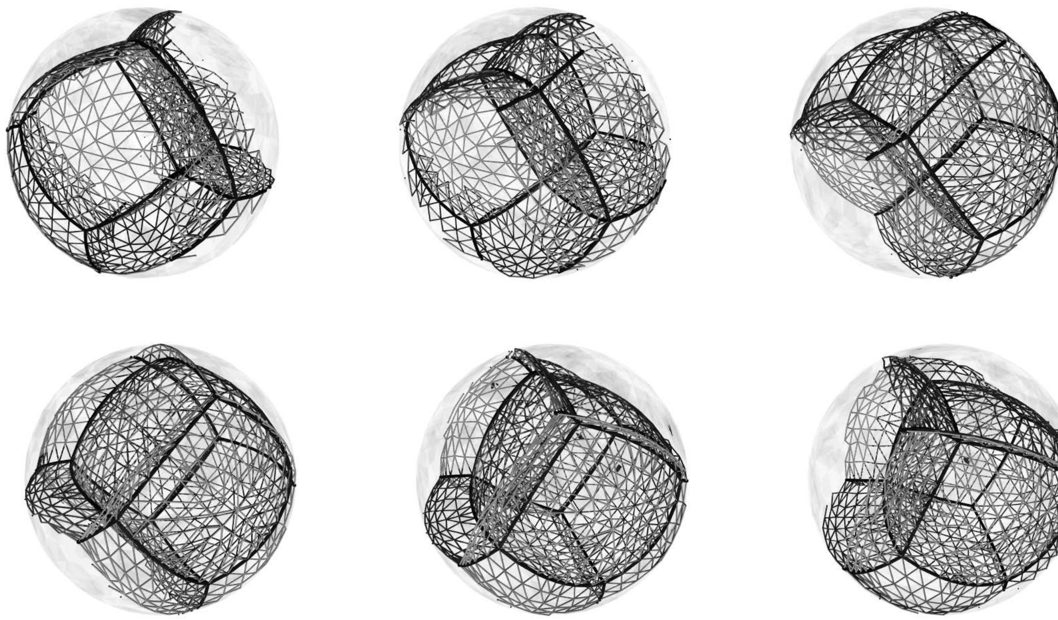


Figura 9.12: (Continuação.)

9.1.3 O mapa `16-cell`

O mapa `16-cell` ou hiperoctaedro é uma triangulação não degenerada e portanto podemos constuir diretamente um modelo otimizado sem necessidade de subdivisão baricêntrica. A figura 9.13 mostra uma configuração ótima deste mapa para a energia $\mathcal{E}_{spring}$.

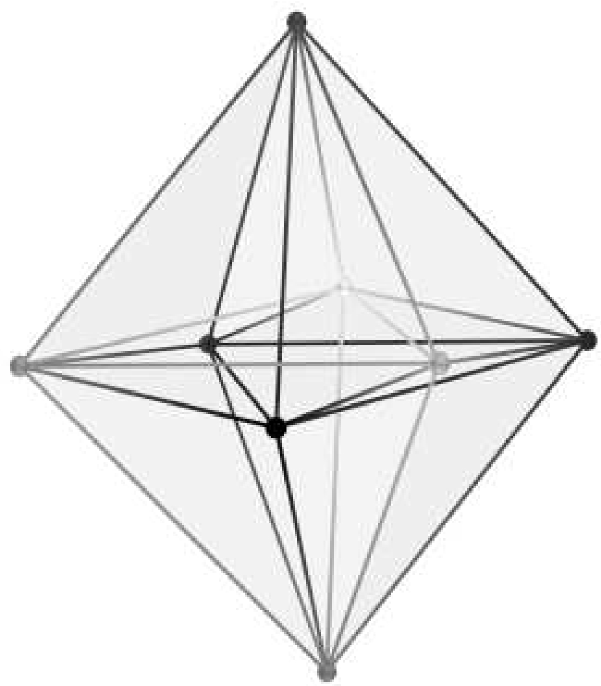


Figura 9.13: Projeção 3D de configuração ótima do mapa `16-cell`. A configuração foi obtida com 500 passos do método `Grad`.

As figuras 9.14 e 9.15 mostram um modelo geométrico mais refinado do mapa `16-cell`, obtido após três etapas de refinamento uniforme de ordem 2, dando origem a uma triangulação com 5568 vértices, 33216 arestas, 55296 faces e 27648 tetraedros. A configuração otimizada foi obtida através do método `Grad` aplicando uma estratégia multi-escala: a triangulação original e cada subdivisão sucessiva foram otimizadas considerando a energia $\mathcal{E}_{spring}$. A triangulação final foi otimizada para a energia $\mathcal{E} = 0.99\ \mathcal{E}_{curv3} + 0.005\ \mathcal{E}_{curv2} + 0.005\ \mathcal{E}_{curv1}$.

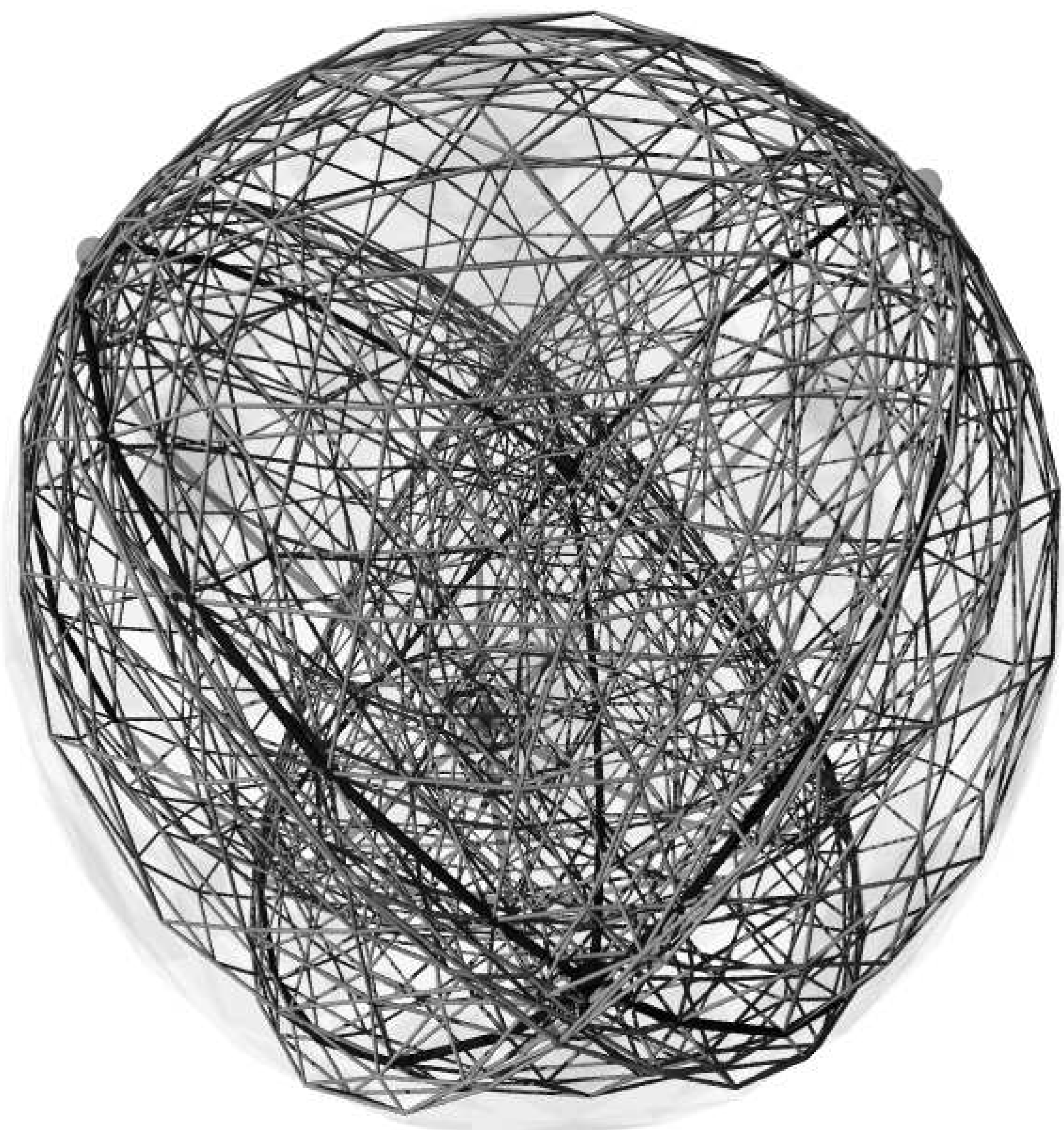
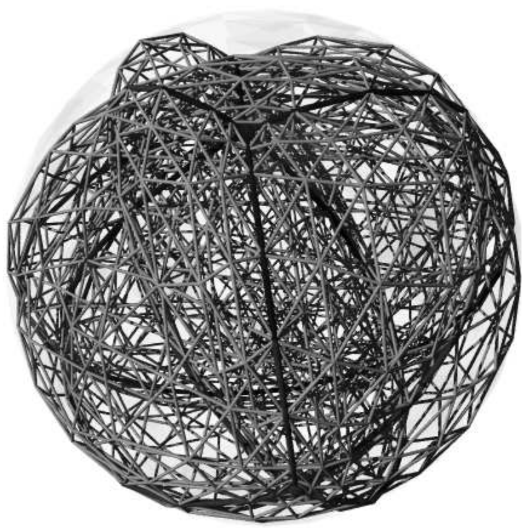
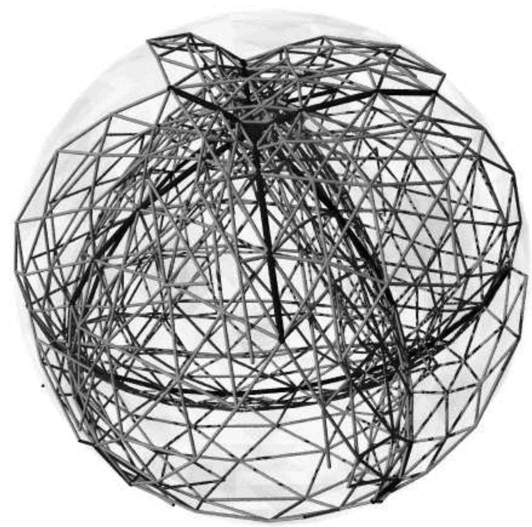


Figura 9.14: Projeção 3D do mapa `16-cell`, visualizado com faces gradeadas e com tonalização dos elementos projetados conforme sua distância do observador no $\mathbb{R}^4$. A configuração final foi otimizada com 1000 passos do método `Grad`.



(a)



(b)

(c)

Figura 9.15: O mapa `16-cell` projetado no $\mathbb{R}^3$, visualizado com faces originais gradeadas e faces de silhueta. Projeções "completa" (a), parte "visível" (b), parte "oculta" (c).

As animações 9.16 a 9.18 mostram o resultado do movimento do observador no $\mathbb{R}^4$.

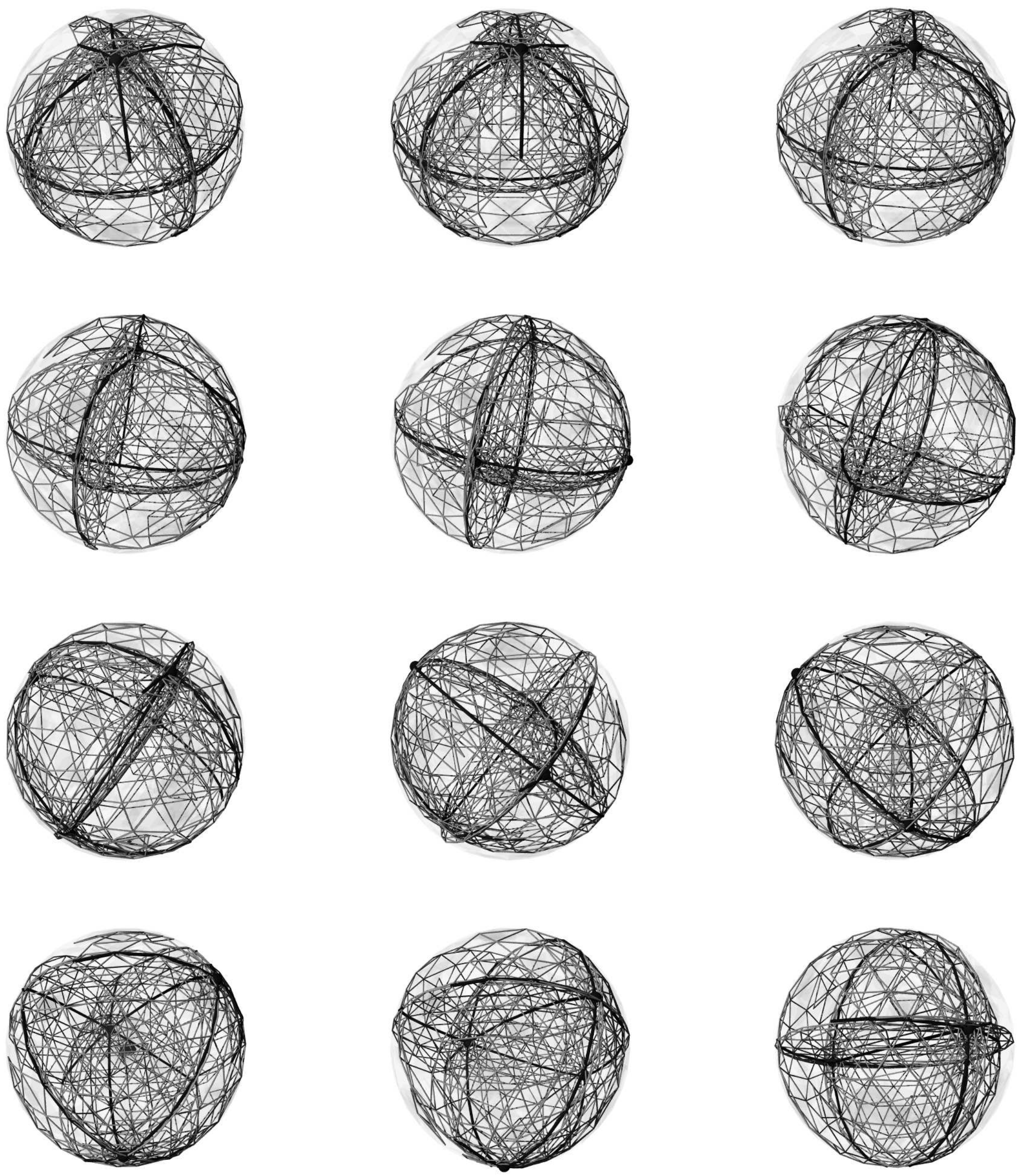


Figura 9.16: Projeções 3D da parte visível de uma triangulação do mapa `16-cell` para várias posições do observador no $\mathbb{R}^4$. A sequência deve ser apreciada por linhas, de esquerda para a direita e de cima para baixo.

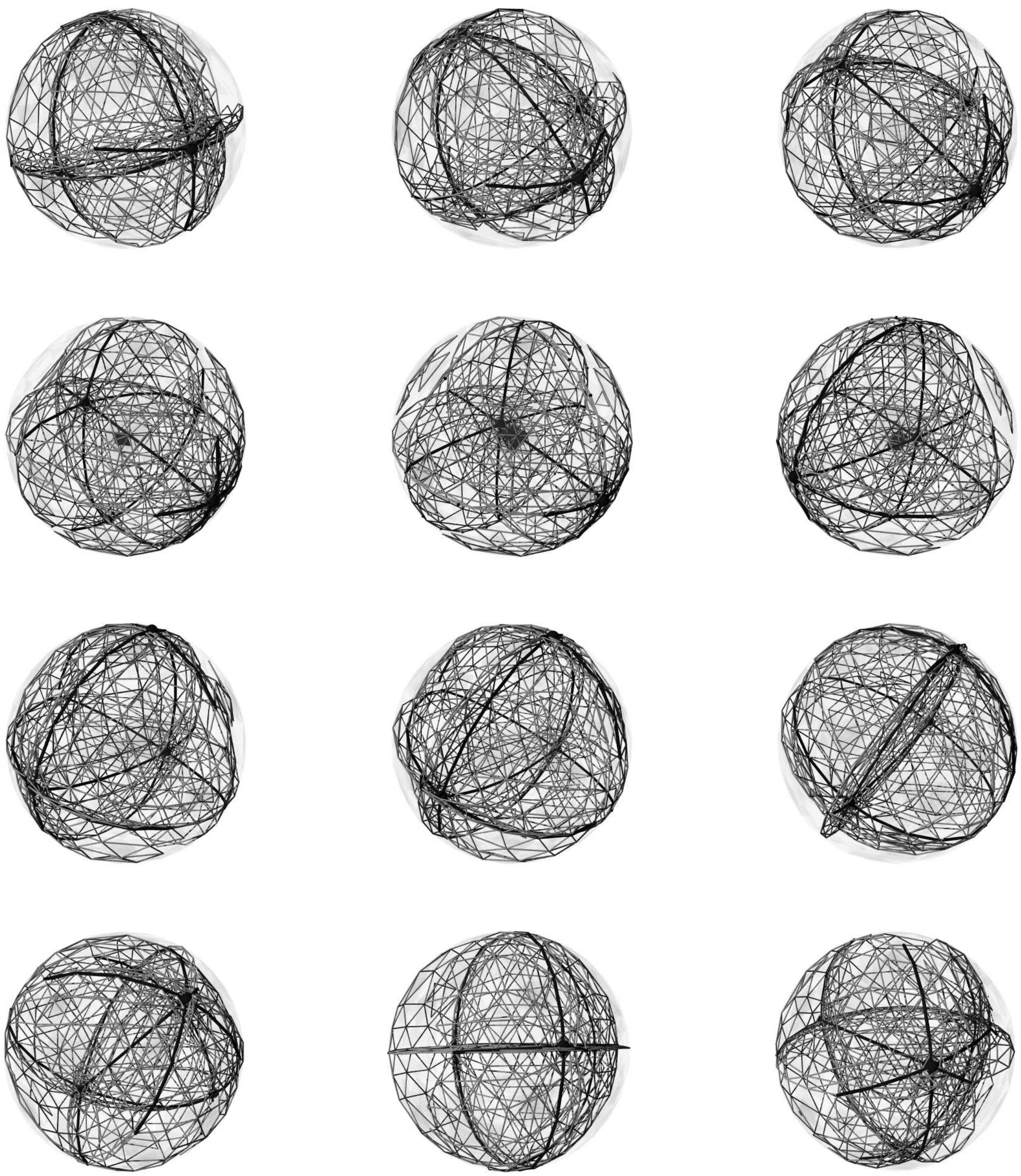


Figura 9.17: (Continuação.)

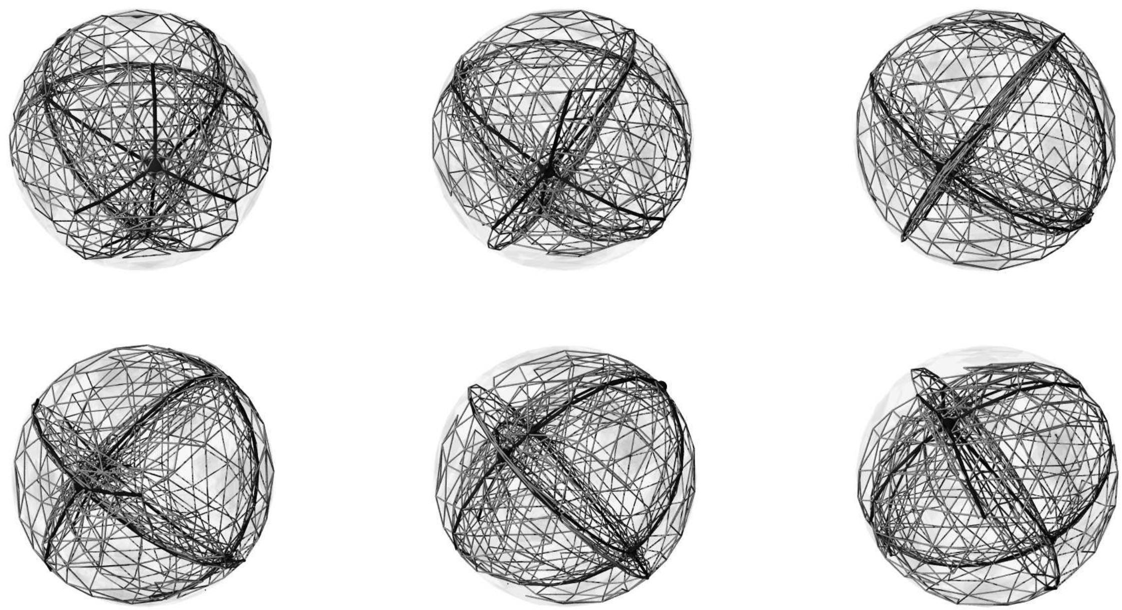


Figura 9.18: (Continuação.)

9.1.4 O mapa `24-cell`

As células do mapa `24-cell` (vide seção 3.2.11) são octaedros, portanto a otimização só pode ser aplicada a triangulações do mesmo. Em nossos testes usamos uma triangulação em que cada octaedro é subdividido radialmente em oito tetraedros, como no mapa `8-vstar` (vide seção 3.2.4). A figura 9.19 mostra uma configuração ótima deste mapa para a energia $\mathcal{E}_{spring}$.

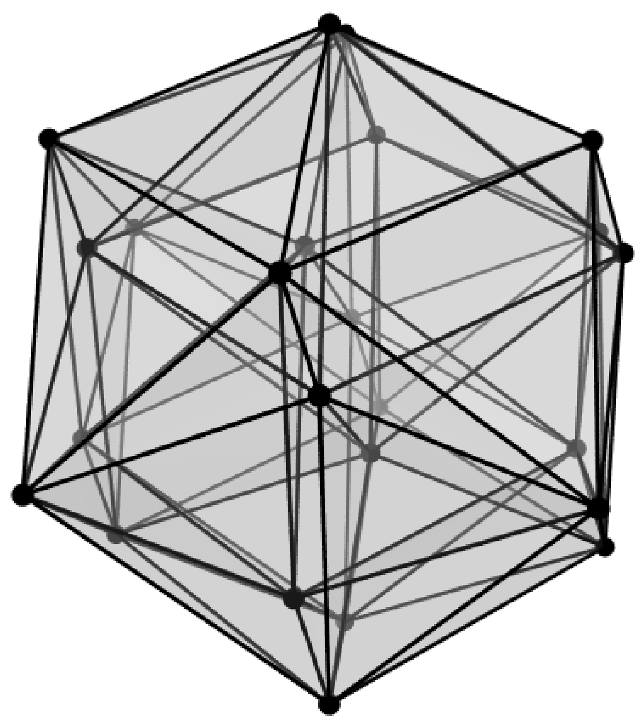


Figura 9.19: Projeção 3D de uma triangulação do mapa `24-cell`, otimizada para a energia $\mathcal{E}_{spring}$. A configuração foi obtida com 500 passos do método `Grad`.

A figura 9.20 mostra várias projeções 3D de um modelo geométrico mais refinado do mapa `24-cell`. Obtivemos estas realizações considerando duas etapas do refinamento uniforme de ordem dois, dando origem a uma triangulação com 5568 vértices, 33216 arestas, 55296 faces triangulares e 27648 tetraedros. A configuração foi obtida por otimização multi-escala, considerando-se a energia $\mathcal{E}_{spring}$ para as triangulações mais grosseiras e a energia mista $\mathcal{E} = 0.90\,\mathcal{E}_{curv3} + 0.05\,\mathcal{E}_{curv2} + 0.05\,\mathcal{E}_{curv1}$ para a triangulação final.

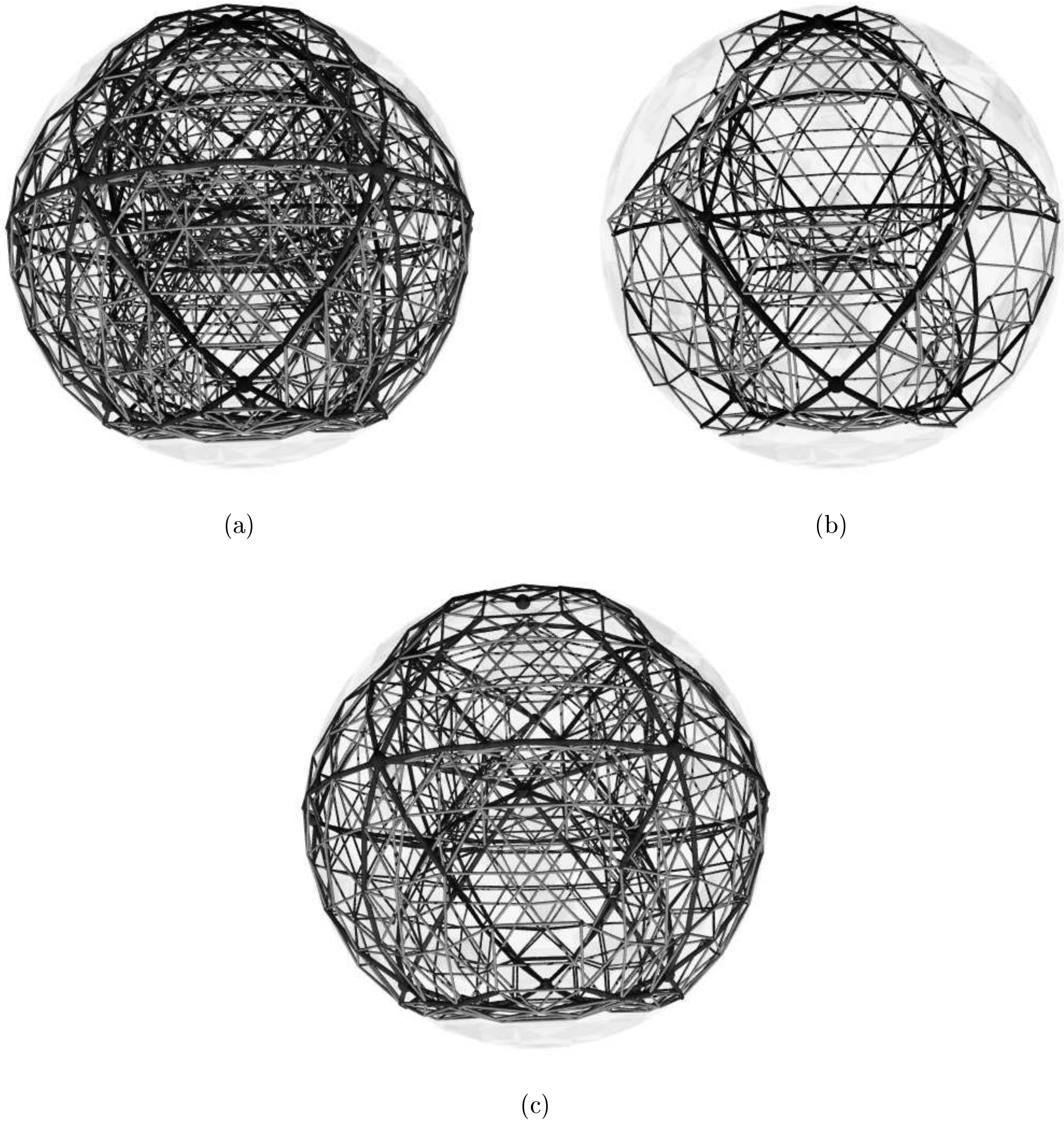


Figura 9.20: O mapa `24-cell` projetado no $\mathbb{R}^3$: em projeção "completa" (a), parte "visível"(b) e parte "oculta" (c). A configuração foi obtida com 500 passos do método `Grad`.

As animações 9.21 a 9.23 mostram o resultado do movimento do observador no $\mathbb{R}^4$.

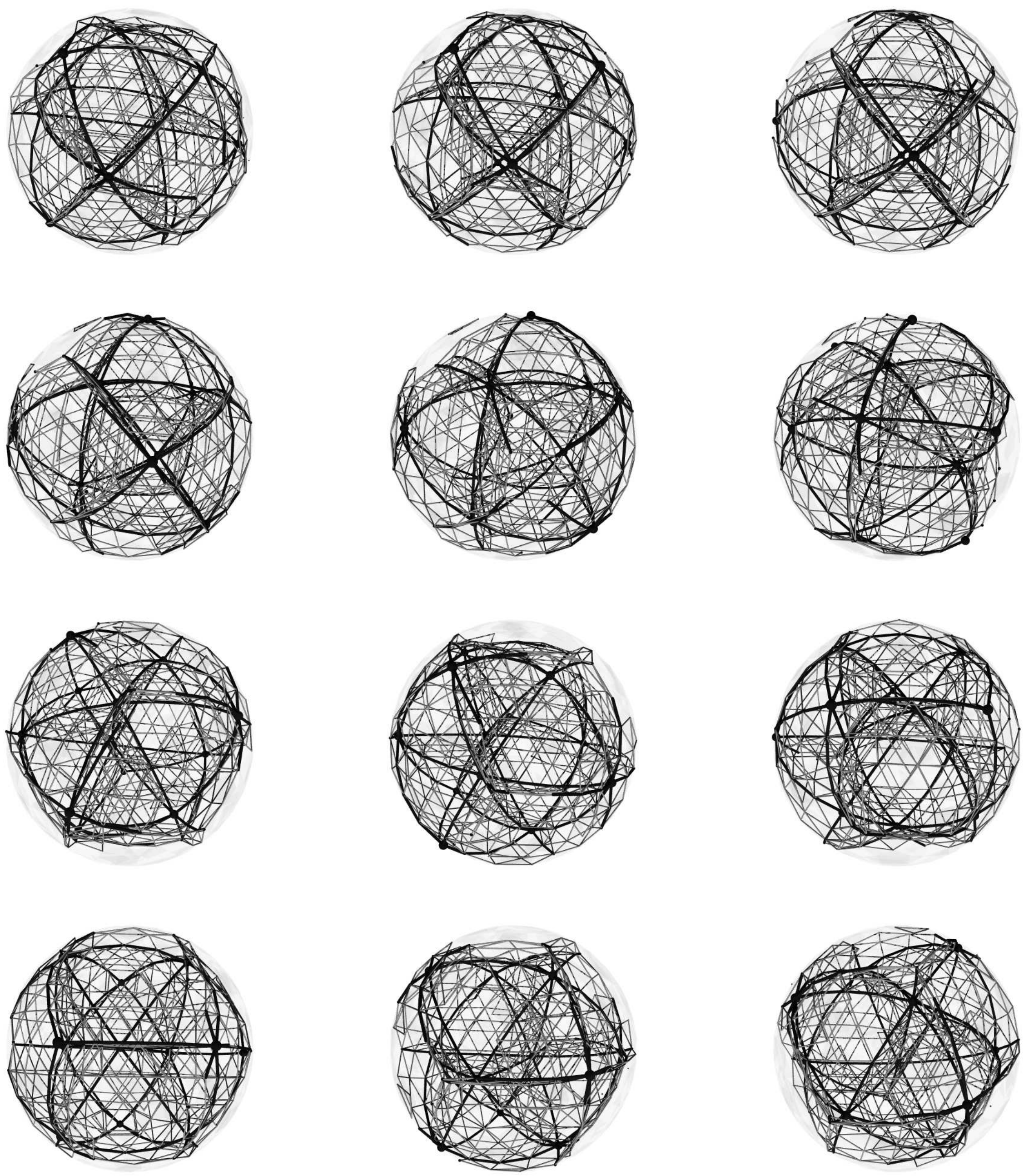


Figura 9.21: Projeções 3D da parte visível de uma triangulação do mapa `24-cell` para várias posições do observador no $\mathbb{R}^4$. A sequência deve ser apreciada por linhas, de esquerda para a direita e de cima para baixo.

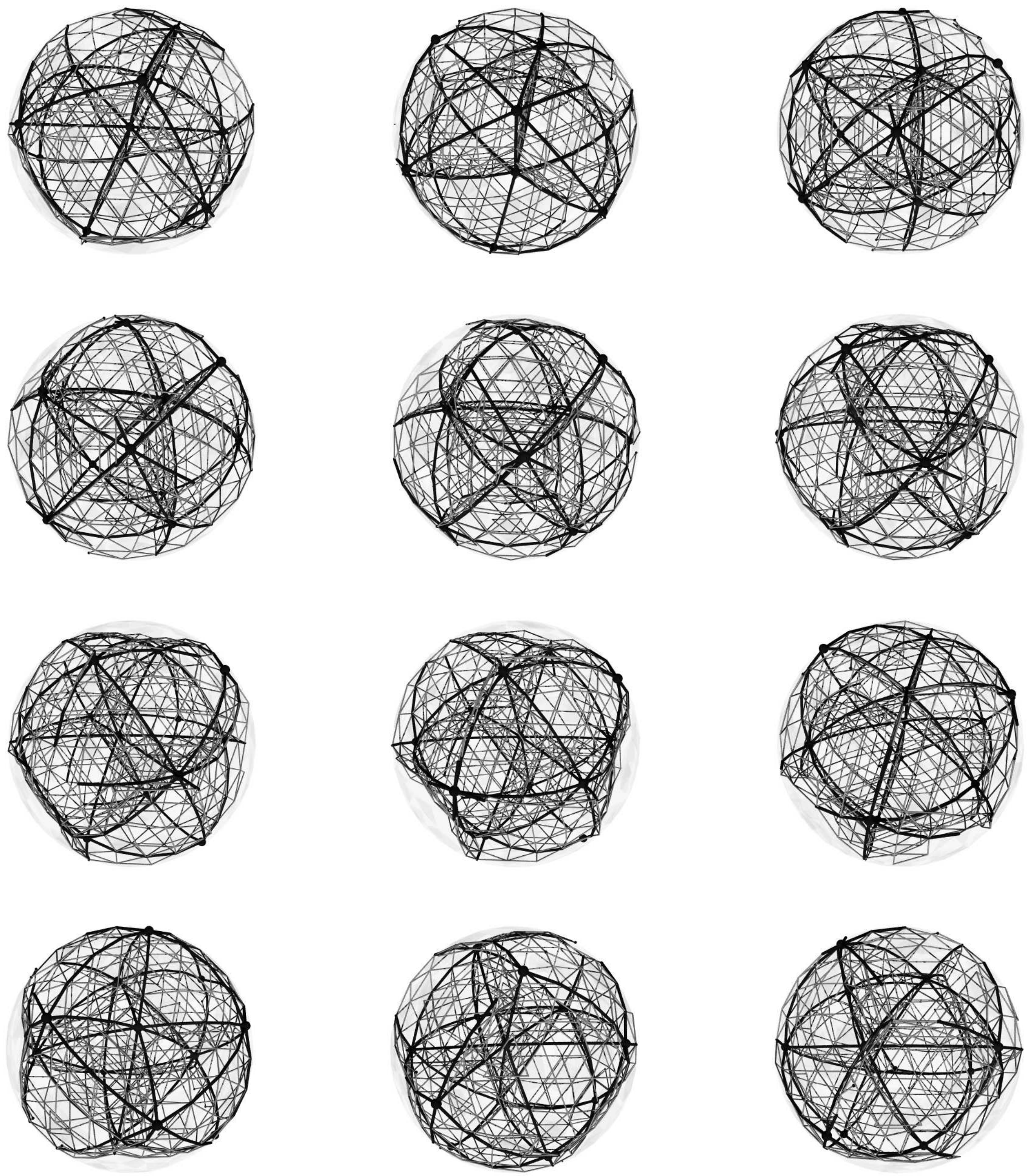


Figura 9.22: (Continuação.)

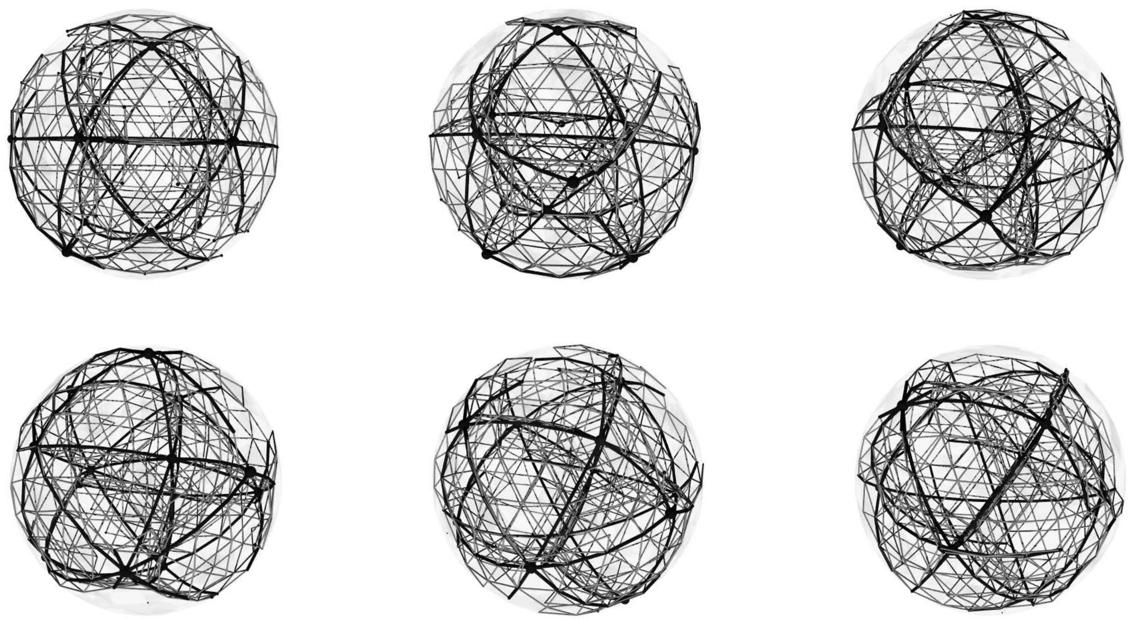


Figura 9.23: (Continuação.)

9.1.5 O mapa `600-cell`

O mapa `600-cell` é também uma triangulação não degenerada, e portanto pode ser modelado diretamente sem necessidade de subdivisão baricêntrica. A figura 9.24 mostra um configuração ótima deste mapa para a energia $\mathcal{E}_{spring}$.

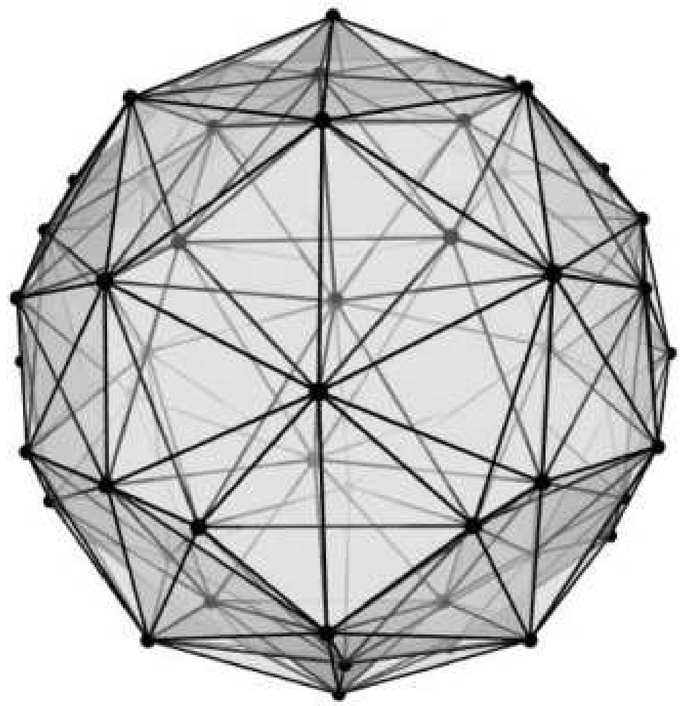


Figura 9.24: Projeção 3D de uma configuração ótima do mapa `600-cell`, otimizada com 500 passos do método `Grad`.

As figuras 9.25 e 9.26 ilustram projeções 3D da parte "visível" de outro modelo deste mapa obtido por refinamento uniforme de ordem dois. A configuração foi obtida através da estrategia multi-escala — refinamos a configuração mostrada na figura 9.24 retendo a sua geometria, e otimizamos a triangulação final com a energia $\mathcal{E}_{curv3}$.

Observe-se nas figuras 9.25 e 9.26 como é difícil visualizar as diferentes camadas de células quando o mapa possui um número grande destas, mesmo quando exibimos unicamente a parte "visível" (frontal) do mapa.

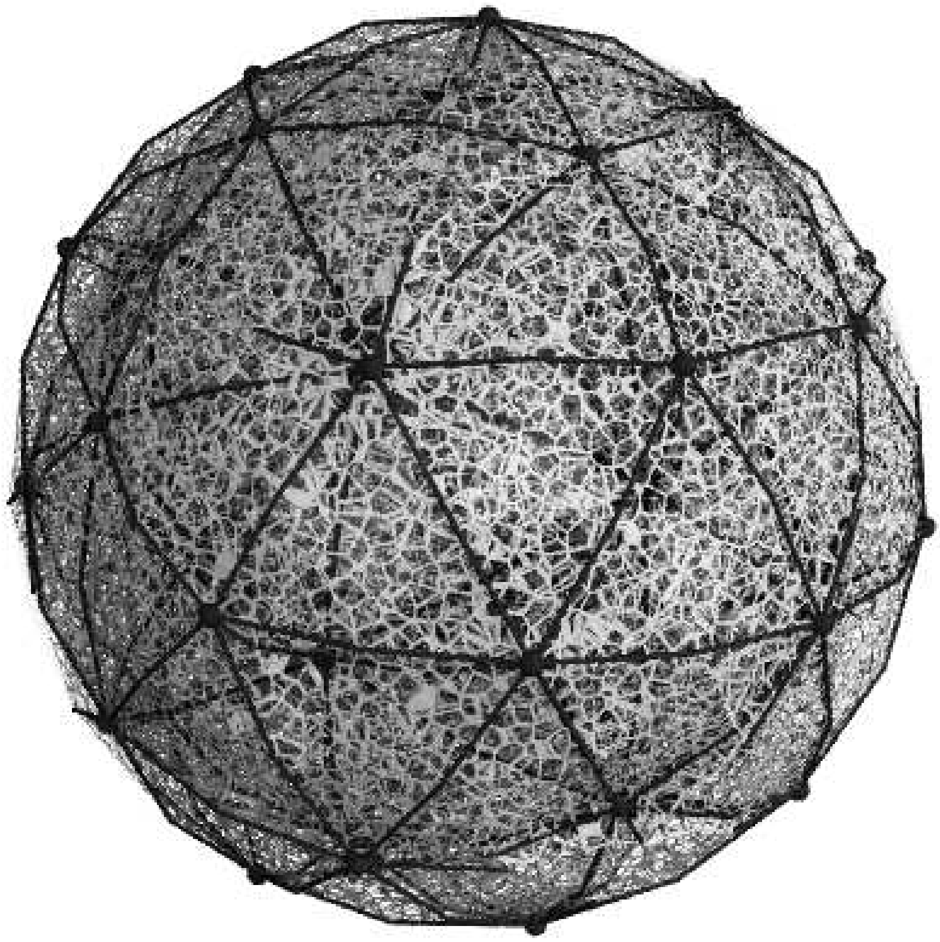


Figura 9.25: Projeção 3D da parte "visível" (frontal) do mapa `600-cell`. A realização exibe faces com textura de transparência modulada. A configuração foi obtida com 500 passos do método `Grad`. A triangulação possui 7200 tetraedros.

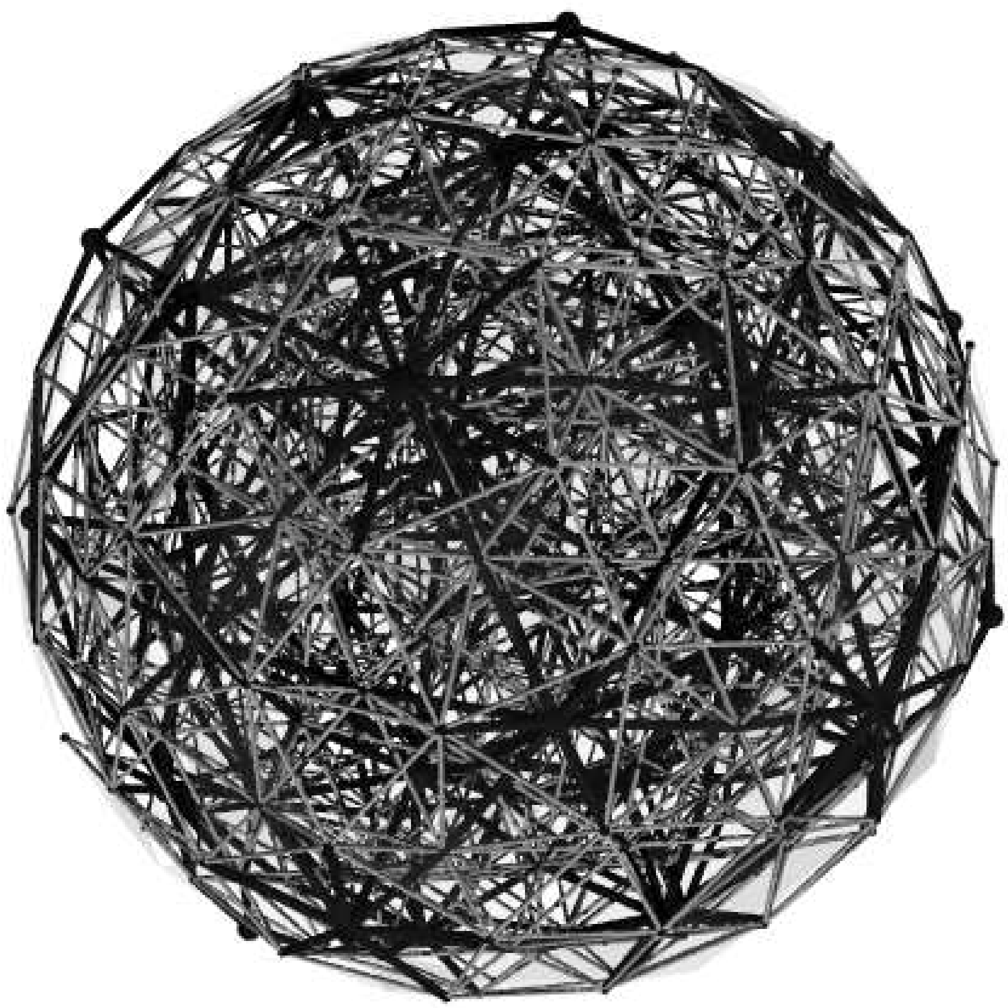


Figura 9.26: Projeção 3D da parte "visível" (frontal) do mapa `600-cell`. A realização exibe faces gradeadas. A configuração é a mesma da figura 9.25.

9.1.6 O mapa `120-cell`

A figura 9.27 mostra uma realização do mapa `120-cell` obtida por dualização topológica e geométrica do modelo do mapa mostrado na figura 9.24.

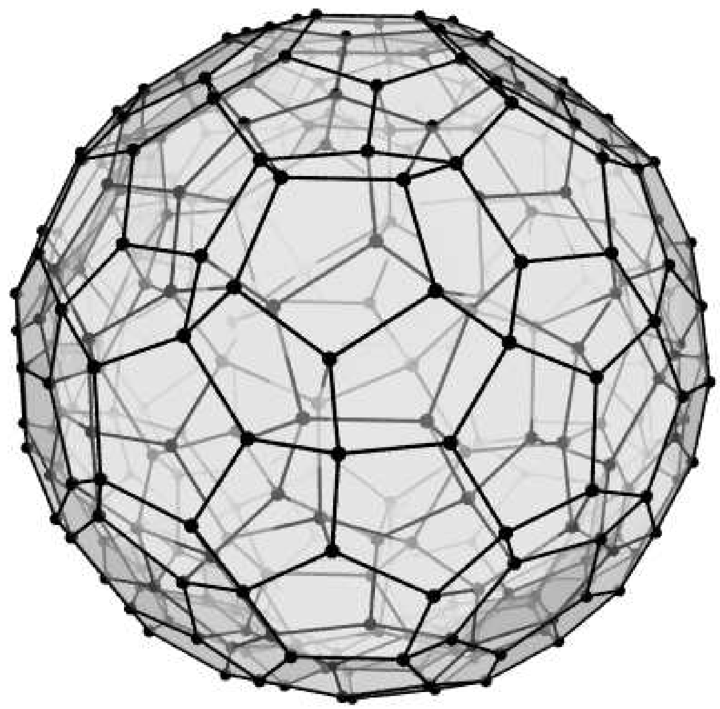


Figura 9.27: Projeção 3D de um modelo do mapa `120-cell`, obtido por dualização de um modelo do mapa `600-cell`.

A figura 9.28 mostra uma realização refinada do mapa `120-cell` obtida por subdivisão baricêntrica da configuração anterior.

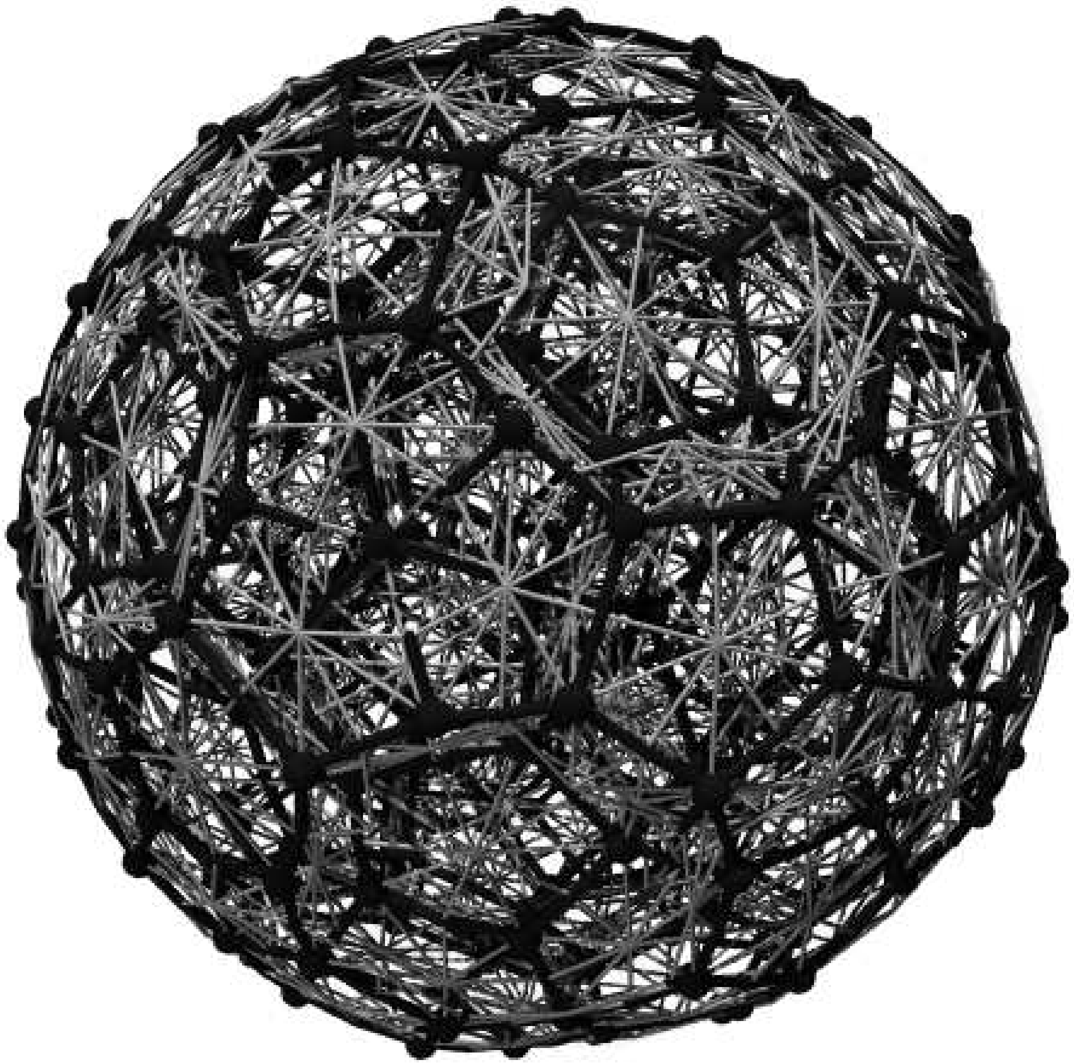


Figura 9.28: Projeção 3D da parte “visível” (frontal) de um modelo triangulado do mapa `120-cell`. A realização exibe faces gradeadas. A configuração foi obtida com 500 passos do método `Grad` para a energia $\mathcal{E}_{spring}$.

Este modelo possui 2640 vértices, 17040 arestas, 28800 faces e 14400 tetraedros.

9.1.7 O mapa `10-cell`

A figura 9.29 mostra uma realização geométrica do mapa `10-cell`, simplexo truncado que possui 5 células octaedrais e 5 tetraédricas. As células octaedrais deste mapa foram modeladas através da colagem de oito tetraedros na topologia `8-vstar`, como no caso o mapa `24-cell` (vide seção 9.1.4).

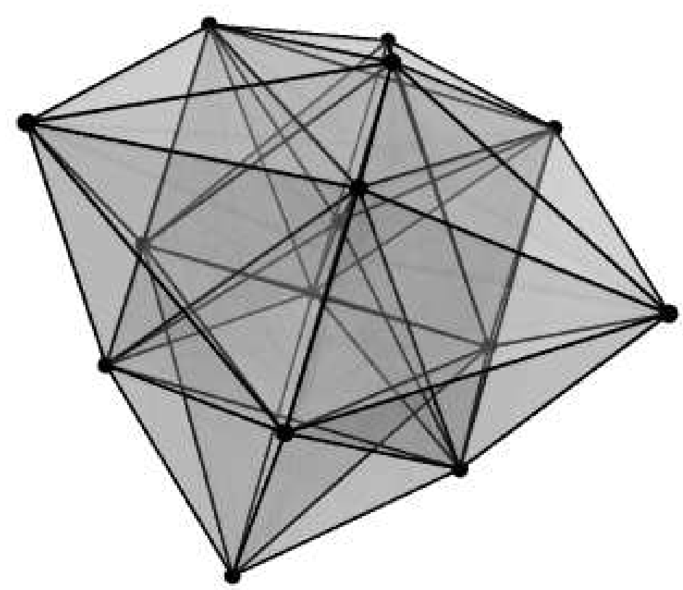


Figura 9.29: Projeção 3D de configuração ótima do mapa `10-cell`. A configuração foi obtida por otimização da energia $\mathcal{E}_{spring}$ após 500 passos do método `Grad`.

A figura 9.30 mostra uma realização geométrica de um refinamento do modelo da figura 9.29, obtido por subdivisão baricêntrica seguida de uma etapa de refinamento uniforme de ordem dois. A triangulação resultante possui 2580 vértices, 15540 arestas, 25920 faces e 12960 tetraedros. A configuração da figura 9.30 foi obtida minimizando-se a energia mista $\mathcal{E} = 0.90\, \mathcal{E}_{curv3} + 0.05\, \mathcal{E}_{curv2} + 0.05\, \mathcal{E}_{curv1}$, a partir da geometria da figura 9.29.

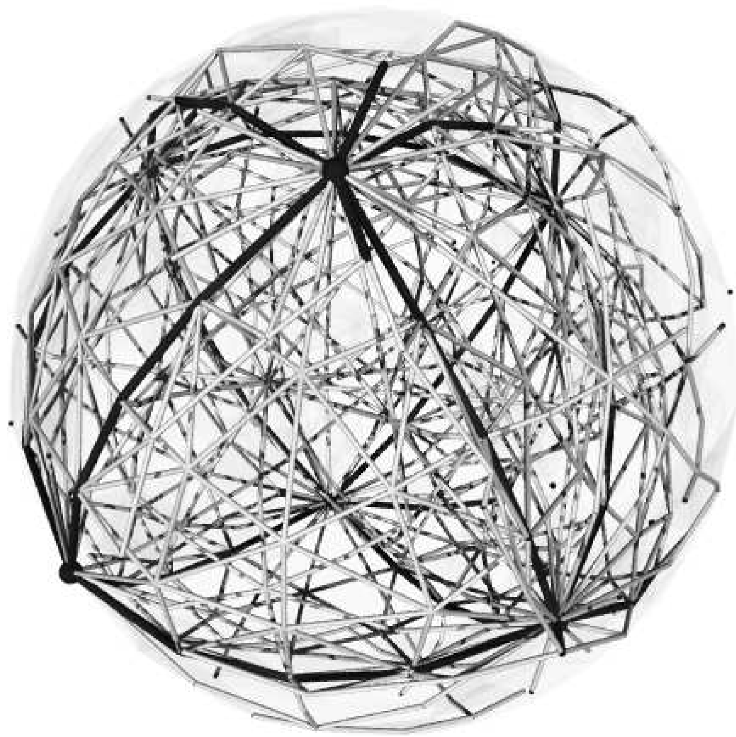


Figura 9.30: Projeção 3D da parte "visível" (frontal) do mapa simplexo truncado. A projeção mostra uma célula octaedral e parte de algumas células tetraedrais. A configuração foi obtida com 500 passos do método `Grad`.

Capítulo 10

Outros mapas

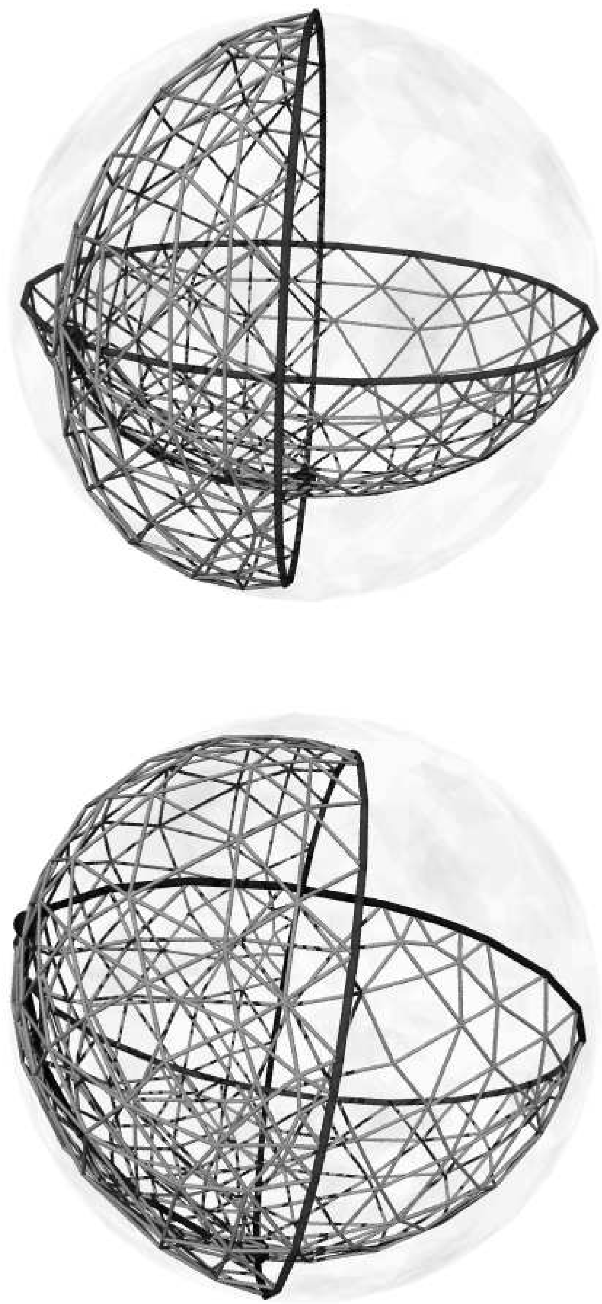
Neste capítulo apresentamos algumas realizações de mapas especiais que não possuem realizações geométricas regulares no $\mathbb{R}^4$: quer por terem topologia irregular, quer por terem realizações regulares apenas em espaços de dimensão maior que 4. Para estes mapas, não existe uma realização "obviamente" ótima e portanto as realizações obtidas pelas nossas técnicas são em certo sentido "criativas".

10.1 O mapa monoplexo

Lembramos que o mapa `monoplex` consiste de uma única célula tetraedral com suas faces coladas aos pares (vide seção 3.2.10). O mapa resultante possui 2 vértices, 3 arestas e 2 faces. Modelamos este mapa através da colagem de um tetraedro subdividido em 16 tetraedros menores, como o ilustrado na figura 3.18(b).

A figura 10.1 mostra duas projeções 3D de um modelo geométrico refinado do mapa monoplexo, resultante de três estágios de refinamento uniforme de ordem $k = 2$. A triangulação final possui 5568 vértices, 33216 arestas, 55296 faces e 27648 tetraedros. A configuração mínima mostrada na figura foi obtida através da técnica multi-escala, usando a energia de molas $\mathcal{E}_{spring}$ nas etapas iniciais e a energia $\mathcal{E} = 0.99\ \mathcal{E}_{curv3} + 0.005\ \mathcal{E}_{curv2} + 0.005\ \mathcal{E}_{curv1}$ na última etapa.

(a)



(b)

Figura 10.1: Projeções 3D completas de uma triangulação do mapa **monoplex**, otimizada para a energia $\mathcal{E} = 0.99\ \mathcal{E}_{curv3} + 0.005\ \mathcal{E}_{curv2} + 0.005\ \mathcal{E}_{curv1}$, com aplicação multi-escala do método **Grad** (500 passos nas etapas iniciais e 1000 passos na etapa final).

10.2 O mapa grade toroidal

Como descrito na seção 3.2.13 o mapa `3-torus[`n, n, n`]` é formado por n^3 células cúbicas, coladas ciclicamente nas três direções de modo a constituir uma subdivisão regular do toro tridimensional $\mathbb{T}^3$. Este mapa admite uma realização regular no $\mathbb{R}^6$ (uma subdivisão do espaço produto $\mathbb{S}^1 \times \mathbb{S}^1 \times \mathbb{S}^1$). Entretanto, uma vez que $\mathbb{T}^3$ é uma variedade orientável, o mapa também admite uma realização (irregular) sem auto-interseções no $\mathbb{R}^4$. (Por análogia, considere que o toro bidimensional $\mathbb{T}^2$ tem uma realização uniforme no $\mathbb{R}^4$ — o "toro chato", $\mathbb{S}^1 \times \mathbb{S}^1$ — mas admite também uma realização não uniforme no $\mathbb{R}^3$ — o "toro clássico".) A figura 10.2 mostra uma realização do mapa `3-torus[`$6, 6, 6$`]`, com $6^3 = 216$ células.

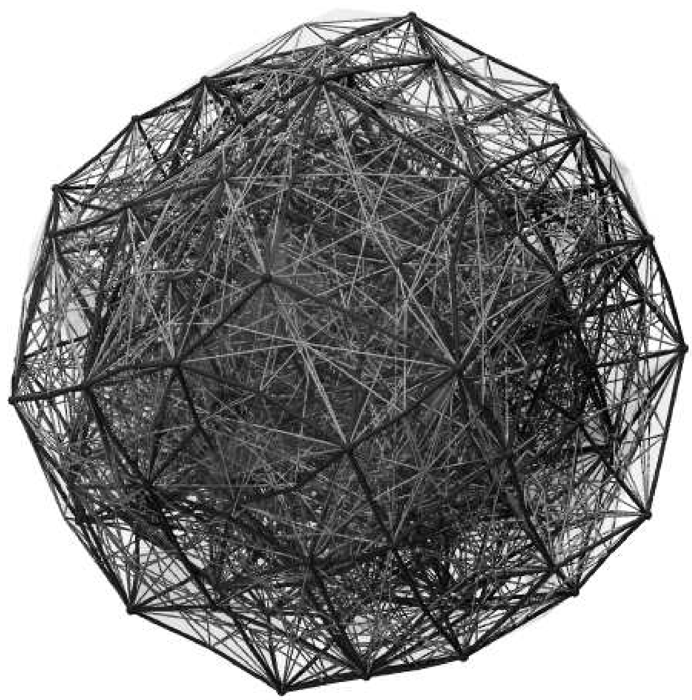


Figura 10.2: Uma projeção completa no $\mathbb{R}^3$ de uma realização do mapa `3-torus[`$6, 6, 6$`]`, otimizada inicialmente para a energia $\mathcal{E}_{spring}$, e, após subdivisão baricêntrica, re-otimizada para a energia $\mathcal{E} = 0.99\ \mathcal{E}_{curv3} + 0.05\ \mathcal{E}_{curv2} + 0.05\ \mathcal{E}_{curv1}$.

10.3 O espaço projetivo

O espaço projetivo $\mathbb{RP}^3$ é uma variedade orientável que pode ser obtida pelo esquema de colagem mostrado na figura 2.9(b), ou pela gema da figura 2.5(b) do capítulo 2. Por dualização desta última obtemos um mapa com 4 células, 12 faces, 16 arestas e 4 vértices.

A figura 10.3 mostra uma realização de um refinamento deste mapa, obtido por subdivisão baricêntrica seguida de dois etapas de refinamento uniforme de ordem dois. A realização foi obtida minimizando-se a energia $\mathcal{E}_{spring}$ para a triangulação baricêntrica (1000 passos do método `Grad`) e da energia mista $\mathcal{E} = 0.85\,\mathcal{E}_{curv3} + 0.15\,\mathcal{E}_{curv2}$ nas etapas seguintes (3000 e 1000 passos do método `Grad` respectivamente).

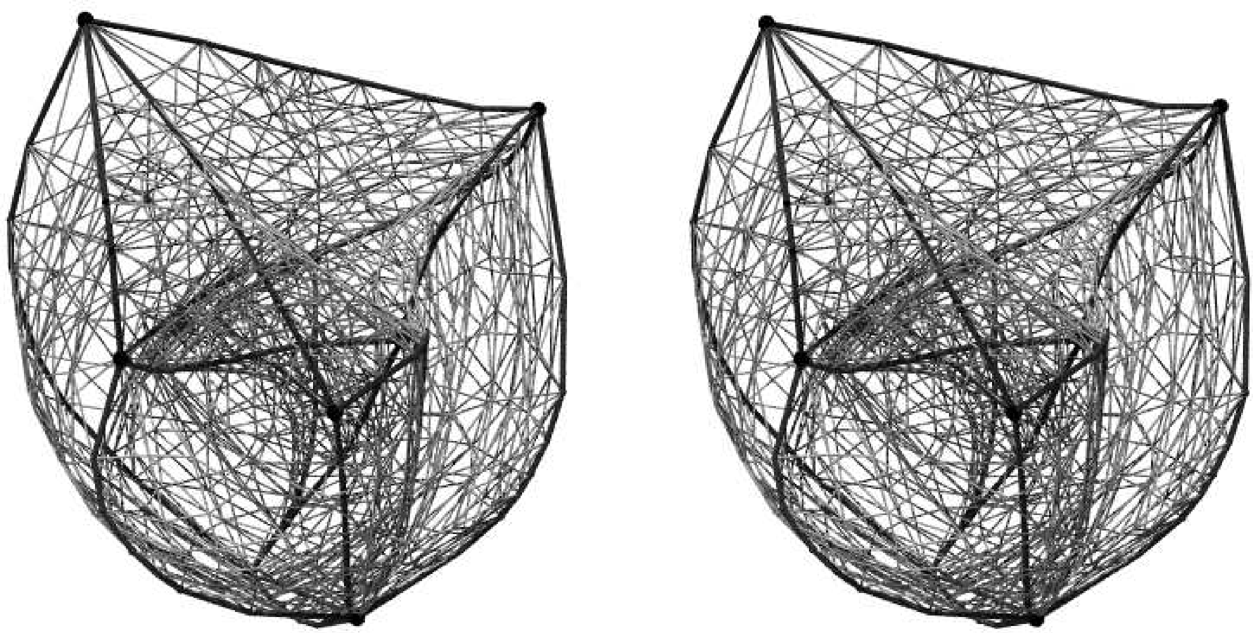


Figura 10.3: O mapa espaço projetivo $\mathbb{RP}^3$ em par estéreo. Estas imagens devem ser apreciadas através da técnica de olhar convergente.

10.4 A gema $\boldsymbol{r_{20}^{24}}$

O exemplo seguinte é um mapa estudado por S. Lins no contexto da caracterização algorítmica de variedades [22]. O mapa é definido por uma gema de 24 nós (r_{20}^{24} na notação de Lins). O modelo de colagem de tetraedros resulta num mapa altamente degenerado com 24 tetraedros, 48 faces, 28 arestas e apenas 4 vértices. A variedade subjacente a este mapa é a esfera $\mathbb{S}^3$. O mapa dual (24 vértices, 48 arestas, 28 faces e 4 células) é altamente simétrico, no sentido que, para quaisquer par de células c_x e c_y, existe um isomorfismo do mapa que leva c_x para c_y.

A figura 10.4 mostra um modelo geométrico de uma única célula deste mapa (24 vértices, 36 arestas e 14 faces) obtido através da técnica de multiescala; inicialmente considerando a energia $\mathcal{E}_{springvar}$ e finalmente a energia $\mathcal{E} = 0.88\ \mathcal{E}_{curv3} + 0.11\ \mathcal{E}_{curv2} + 0.01\mathcal{E}_{springvar}$ após 2000 e 1000 passos do método `Grad` respectivamente.

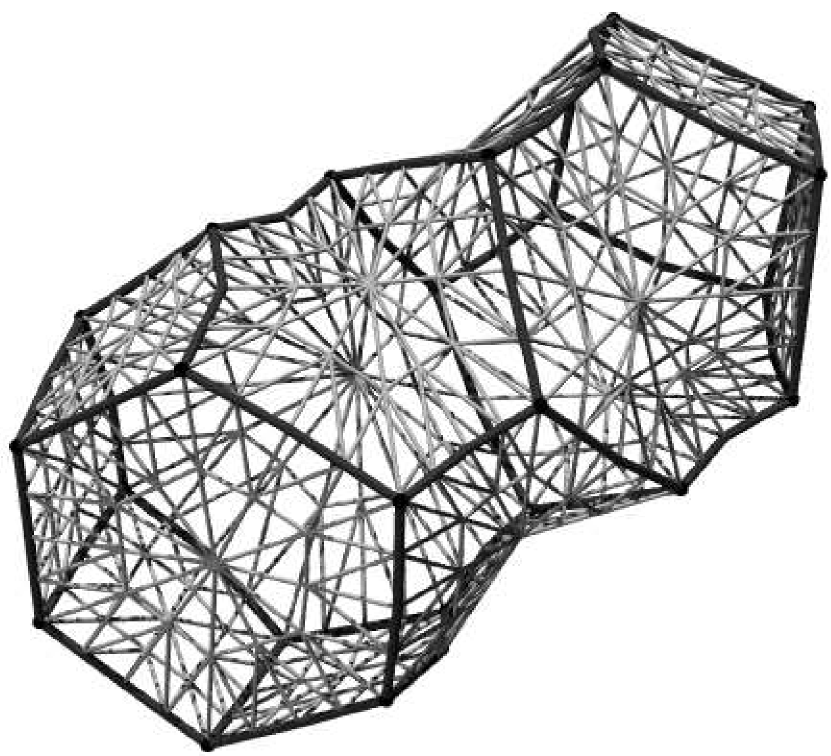


Figura 10.4: Um modelo para uma triangulação de uma única célula do mapa r_{20}^{24}, exibida com faces gradeadas.

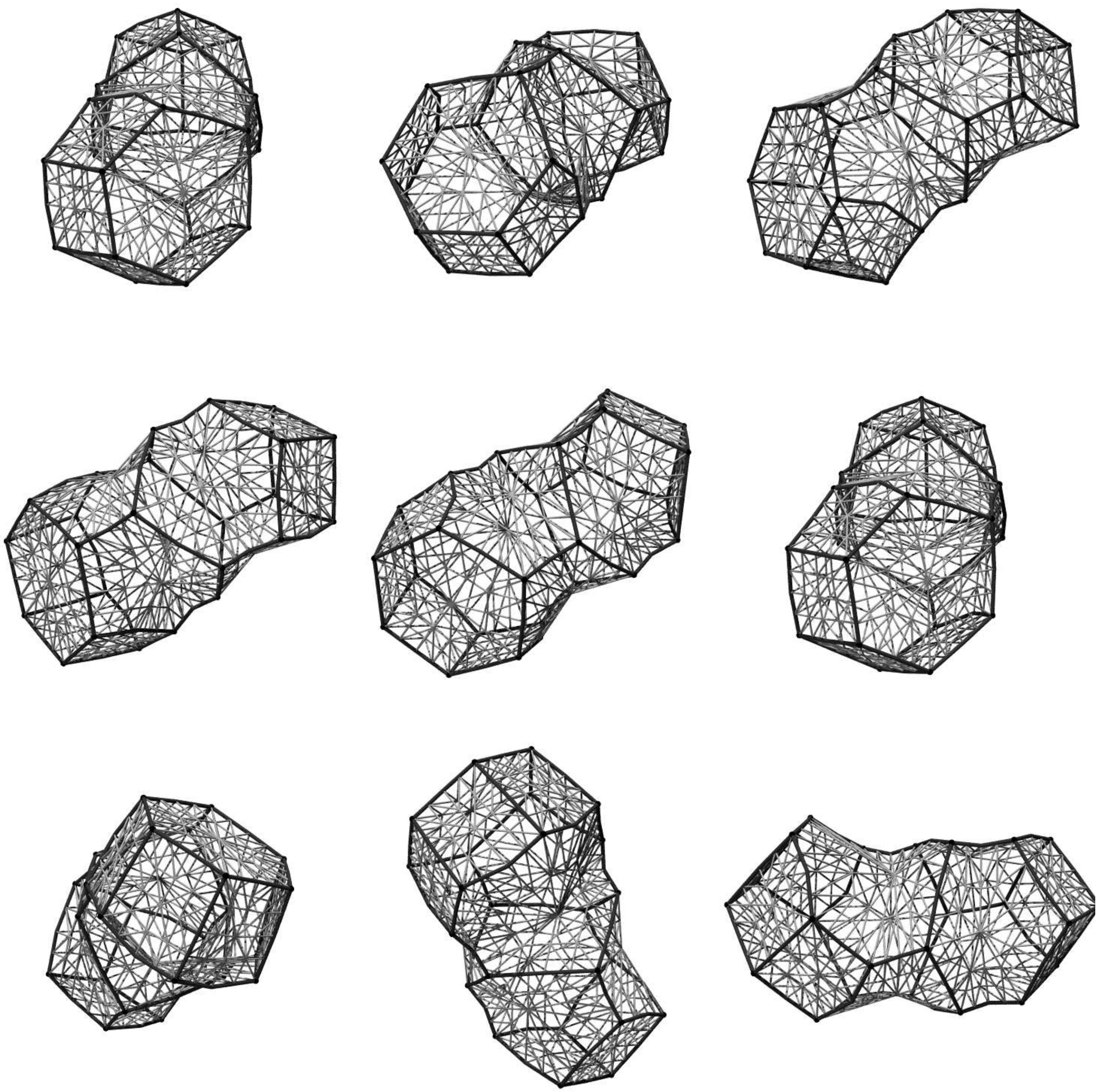


Figura 10.5: Várias vistas do modelo triangulado de uma única célula do mapa r_{20}^{24}.

A figura 10.6 mostra um modelo do mapa dual da gema r_{20}^{24}, menos uma célula (o que resulta num mapa com borda sobre a bola $\kappa\mathbb{B}^3$). O modelo foi obtido aplicando-se subdivisão baricêntrica, que produziu uma triangulação com 432 tetraedros, seguida de subdivisão uniforme de ordem 2, resultando em 5184 tetraedros. O modelo foi otimizada com o método `Grad`, segundo a abordagem multiescala — inicialmente para a energia $\mathcal{E}_{springvar}$, e em seguida para a energia $\mathcal{E}_{curv2}$, com 1000 e 100 passos, respectivamente. (Este modelo ainda está longe de ser um mergulho no $\mathbb{R}^3$, pois aproximadamente metade dos tetraedros têm orientação geométrica contrária à dos demais.)

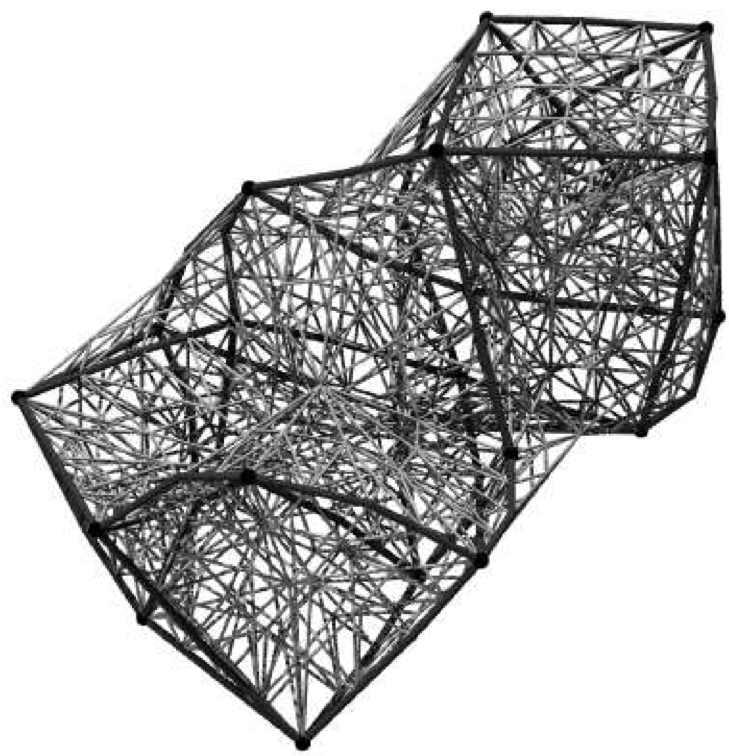


Figura 10.6: Um modelo para uma triangulação do mapa r_{20}^{24}, menos uma célula, exibida com faces gradeadas.

Capítulo 11

Conclusões

Os exemplos apresentados nos capítulos 9 e 10 mostram que as técnicas por nós desenvolvidas — especificamente, a otimização de funções energia — permitem construir bons modelos geométricos no $\mathbb{R}^4$, para mapas com algumas centenas de vértices. Com técnicas de multiescala, conseguimos obter modelos otimizados para triangulações desses mapas com quase 10^5 tetraedros.

Dentre as funções energia, observamos que as mais importantes são a energia de molas $\mathcal{E}_{spring}$ e a energia de curvatura tridimensional $\mathcal{E}_{curv3}$. Usamos a energia de molas para evitar auto-intersecções do modelo bem como uniformizar o tamanho de seus elementos. Usamos a energia de curvatura $\mathcal{E}_{curv3}$ para garantir a suavidade da variedade subjacente ao modelo. Outras funções energia que experimentamos têm aplicação limitada, e são até certo ponto redundantes quando as energias $\mathcal{E}_{spring}$ e $\mathcal{E}_{curv3}$ são utilizadas.

Dentre os métodos de otimização, verificamos que o método de descida pelo gradiente `Grad`, apesar do custo adicional exigido pelo calculo das derivadas da energia, é mais eficiente que os métodos que não utilizam informação de derivadas. Observamos que o método `Grad` geralmente consegue encontrar boas configurações para as energias consideradas apesar de ser teoricamente um método de otimização local apenas.

Quanto à visualização do modelo otimizado no $\mathbb{R}^4$, as técnicas que desenvolvemos tem alcance mais limitado. Quando um mapa tem a topologia de um subconjunto de $\mathbb{R}^3$, o modelo geométrico resultante pode ser efetivamente visualizado através da técnica de faces gradeadas. Técnicas de visão estereoscópica também podem auxiliar na compreensão destes modelos. As mesmas técnicas permitem visualizar mapas com topologias da hiperesfera $\mathbb{S}^3$, quer por remoção de uma célula do mapa, quer por normalização e projeção perspectiva combinadas com eliminação de partes ocultas, tonalização por profundidade ou movimentação do observador no $\mathbb{R}^4$. As técnicas de visualização que temos desenvolvido são ainda inadequadas para mapas com outras topologias.

Um defeito acidental de nossos programas, que poderia ser facilmente consertado, é a limitação a mapas orientáveis, que se deve à escolha da estrutura *facet-edge* para a modelagem da topologia dos mapas e de suas triangulações.

A técnica de otimização multiescala exige, para mapas com topologias complexas ou degeneradas, vários estágios de refinamento sucessivos do mapa original. Observamos que a subdivisão baricêntrica não é apropriada para esse fim, pois tende a produzir vértices com grau muito elevado, que dão origem a simplexos com lados extremamente desiguais e modelos não suaves. Os esquemas de subdivisão uniforme reduzem este problema, mas não o eliminam completamente.

11.1 Sugestões de trabalhos futuros

Naturalmente, há um grande número de problemas em aberto que merecem ser abordados, entre os quais citamos:

- **Escolha dos pesos**. No estágio atual de nossa pesquisa escolhemos os pesos das funções energia mistas por tentativa e erro. Seria altamente desejável encontrar métodos automáticos para essa tarefa; ou pelo menos, compreender melhor os efeitos dos pesos. Em particular, ainda resta determinar os fatores de normalização K_i (vide seção 6.6) que tornariam os pesos invariantes sob refinamento do mapa.

- **Geração de configurações iniciais**. Nossas configurações de partida são geradas atribuindo-se a cada coordenada um valor aleatório e independente no intervalo $[-1, +1]$. Seria desejável gerar configurações de partida que fossem mais próximas da configuração final, e/ou menos sujeita a produzir configurações indesejáveis. Parece possível, por exemplo, estender as heurísticas estudadas por Rosi no caso bidimensional [30] para o caso tridimensional.

- **Eficiência da otimização**. Atualmente, a otimização de uma configuração com alguns milhares de vértices exige várias horas de processamento. Certamente deve ser possível acelerar bastante este processo, por exemplo desenvolvendo métodos de otimização especializados para as funções energia de interesse.

- **Mapas não orientáveis**. A estrutura de dados *facet-edge* não permite a representação de mapas não orientáveis; entretanto, muitas de nossas ferramentas estão preparadas para lidar com tais mapas. Uma vez que os mapas não orientáveis são justamente os mais difíceis de entender é desejável remover esta limitação.

Apêndice A

Descrição dos programas

Nós implementamos os métodos e algoritmos apresentados neste trabalho usando a linguagem Modula-3 [14, 25] e a linguagem C [11]. A programação totalizou cerca de 50.000 linhas de código. Neste capítulo vamos descrever os principais componentes desta implementação. Maiores detalhes podem ser obtidos nas interfaces dos módulos das bibliotecas e nos comentários dos programas.

A.1 Programas principais:

`MakeMap`

- Descrição: Este programa constrói a topologia de alguns mapas tridimensionais sem borda. As coordenadas dos vértices são atribuídas de maneira aleatória, no intervalo $[-1, +1]$.
- Sintaxe:
  `MakeMap -shape` { `sphere` | `torus` | `...` }
- Opções:
  `-shape`, especifica o mapa a ser modelado — `sphere`, `torus`, etc.

`MakeRegularPolytope`

- Descrição: Este programa constrói a topologia e geometria de mapas derivados dos politopos 4D regulares e convexos.
- Sintaxe:
  `MakeRegularPolytope -shape` { `5cell` | `8cell` | `...` } [`-random`]

- Opções:
  `-shape`, especifica o mapa a ser modelado — `5-cell, 8-cell, 16-cell`, etc.

  `-random`, especifica que cada coordenada de vértice é atribuída de maneira aleatória, no intervalo $[-1, +1]$.

DualMap

- Descrição: Este programa constrói a topologia "dual" de um mapa dado.
- Sintaxe:
  `DualMap -inTp` ⟨`name`⟩ `-inSt` ⟨`name`⟩
- Opções:
  `-inTp` ⟨`name`⟩, especifica o arquivo de entrada que contém a topologia do mapa.

  `-inSt` ⟨`name`⟩, especifica o arquivo de entrada que contém a geometria do mapa.

MakeMapWithBorder

- Descrição: Este programa constrói a topologia de alguns mapas tridimensionais com borda. As coordenadas dos vértices são atribuídas de maneira aleatória, no intervalo $[-1, +1]$.
- Sintaxe:
  `MakeMapWithBorder -shape` { `sausage` | `torus2D` | `...` }
- Opções:
  `-shape`, especifica o mapa a ser modelado — `sausage`, `torus2D`, etc.

MakeVStar

- Descrição: Este programa constrói a topologia de mapas estrelas de vértices (vide seção 3.2.4).
- Sintaxe:
  `MakeVStar -shape` { `4-vstar` | `8-vstar` | `...` }
- Opções:
  `-shape`, especifica o mapa a ser modelado — `4-vstar`, `8-vstar`, etc.

`MakeBipyramid`

- Descrição: Este programa constrói a topologia de mapas estrelas de arestas (vide seção 3.2.3) e de mapas bipirâmides alongadas (vide seção 3.2.5).

- Sintaxe:
  `MakeBipyramid -order ⟨num⟩ [ -fixed ] [ -open | -elongated ]`

- Opções:
  `-order`, especifica a ordem do mapa.

  `-fixed`, se especificada, as coordenadas dos vértices são calculadas de modo que a geometria seja aproximadamente igual à indicada nas figuras 3.7 e 3.10. Caso contrário as coordenadas dos vértices são atribuídas de maneira aleatória, no intervalo $[-1, +1]$.

  `-open`, se especificada, esta opção determina que o último tetraedro do mapa estrela de arestas não deve ser colado ao primeiro, deixando portanto o anel de tetraedros aberto.

  `-elongated`, se especificada, constrói o mapa com topologia de bipirâmide alongada.

`BarycentricSubdivision`

- Descrição: Este programa implementa a subdivisão baricêntrica de um mapa dado (vide seção 4.2).

- Sintaxe:
  `BarycentricSubdivision -inTp ⟨name⟩ -inSt ⟨name⟩ [ -net ] [ -fixed ]`

- Opções:
  `-inTp ⟨name⟩`, especifica o arquivo de entrada que contém a topologia do mapa.

  `-inSt ⟨name⟩`, especifica o arquivo de entrada que contém a geometria do mapa.

  `-net`, esta opção implementa a técnica de faces gradeadas (vide seção 8.3.3).

  `-fixed`, se especificada, esta opção subdivide o mapa retendo sua geometria inicial.

`SelectSubdivision`

- Descrição: Este programa implementa o refinamento adaptativo dos elementos — tetraedros, faces e arestas — de um mapa tridimensional envolvidos em degenerações geométricas (vide seção 4.4).

- Sintaxe:
  `SelectSubdivision -inTp ⟨name⟩ -element { edge | face | tetrahedron }`

- Opções:
  `-inTp ⟨name⟩`, especifica o arquivo de entrada que contém a topologia do mapa.

  `-element`, especifica o refinamento minimal de uma aresta, face ou tetraedro.

RandomShape

- Descrição: Este programa gera n variações aleatórias de uma configuração geométrica.

- Sintaxe:

```
RandomShape  -inSt ⟨name⟩ -trials ⟨num⟩ ⟨num⟩
             [ -normalize ] [ -jitter ⟨num⟩ ]
```

- Opções:

  `-inSt <name>`, especifica o arquivo de entrada que contém a geometria do mapa.

  `-trials`, especifica o número de configurações aleatórias a gerar.

  `-normalize`, esta opção produz uma configuração normalizada ,i.e., distância media dos vértices ao origem igual a “1”.

  `-jitter`, valor que especifica o grau ou magnitude da pertubação aleatória.

RefineTriang

- Descrição: Este programa implementa o refinamento uniforme para uma triangulação de um mapa tridimensional, como descrito na seção 4.5.

- Sintaxe:
  `RefineTriang -inTp ⟨name⟩ -inSt ⟨name⟩ -order ⟨num⟩ [ -net ] [ -fixed ]`

- Opções:
  `-inTp ⟨name⟩`, especifica o arquivo de entrada que contém a topologia do mapa.

  `-inSt ⟨name⟩`, especifica o arquivo de entrada que contém a geometria do mapa.

  `-order ⟨num⟩`, especifica a ordem de refinamento.

  `-net`, esta opção implementa a técnica de faces gradeadas.

  `-fixed`, se especificada, esta opção subdivide o mapa retendo sua geometria inicial.

ProjectTo3D

- Descrição: Este programa projeta uma configuração $\phi(\mathcal{T})$ para o $\mathbb{R}^3$, como descrito na seção 8.2.
- Sintaxe:

```
ProjectTo3D -inTp ⟨name⟩ -inSt ⟨name⟩ [ -outFile ⟨name⟩ ]
            -projection [ Perspective | Parallel ]
            [ [ -autoProject ] |
              [ -From4 ⟨num⟩ ⟨num⟩ ⟨num⟩ ⟨num⟩ ]
              [ -To4 ⟨num⟩ ⟨num⟩ ⟨num⟩ ⟨num⟩ ]
              [ -Up4 ⟨num⟩ ⟨num⟩ ⟨num⟩ ⟨num⟩ ]
              [ -Over4 ⟨num⟩ ⟨num⟩ ⟨num⟩ ⟨num⟩ ]
            ]
            [ -Vangle4 ⟨num⟩ ]
            [ -printDepth ]
```

- Opções:

`-inTp ⟨name⟩`, especifica o arquivo de entrada que contém a topologia do mapa.

`-inSt ⟨name⟩`, especifica o arquivo de entrada que contém a geometria do mapa.

`-outFile ⟨name⟩`, especifica o arquivo de saída que contém a projeção do mapa.

`-projection`, especifica o modo de projeção —`Perspective, Parallel`.

`-printDepth`, imprime a profundidade 4D ou componente w dos vértices da configuração.

`-From4`, `-To4`, `-Up4`, `-Over4`, `-Vangle4`, especifica os parâmetros de visualização 4D.

OptShape

- Descrição: Este programa é o otimizador das funções de energia e permite também, escrever arquivos do tipo ".plot" e ".gnu" com a evolução das energias sob minimização para propósitos de comparação.
- Sintaxe:

```
OptShape -inTp ⟨name⟩ -inSt ⟨name⟩ -outFile ⟨name⟩
          -maxEvals ⟨num⟩
          [ -writeEvery ⟨num⟩ | -writeAll ]
```

```
 [ -showAll | -showBest ] [-wait] [-scale]
-minimizer
 [ Grad [ RKF4 | Euler ] | Coord [ Brent | JS ] ]
 [ -energy ⟨wt⟩ Edge ] [ -energy ⟨wt⟩ Spring ]
 [ -energy ⟨wt⟩ Curv3D ] [ -energy ⟨wt⟩ Curv2D ]
 [ -energy ⟨wt⟩ Curv1D ]
```

- Opções:
  `-inTp ⟨name⟩`, especifica o arquivo de entrada que contém a topologia do mapa.

  `-inSt ⟨name⟩`, especifica o arquivo de entrada que contém a geometria do mapa.

  `-outFile ⟨name⟩`, especifica o arquivo de saída contendo a configuração otimizada.

  `-maxEvals`, especifica o número máximo de avaliações da função de energia.

  `-showAll`, opção que indica quando escrever o arquivo com a configuração otimizada.

  `-scale`, especifica a escala de desenho da configuração durante a otimização.

  `-minimizer`, especifica o método de minimização, `Grad` (Gradiente), com integrador `RKF4` ou `Euler` ou `Coord` (Coordenada), com minimizador univariado de `Brent` ou de `Stolfi`).

A.2 Programas auxiliares

Estes programas não são estritamente necessários para a solução do problema, mas são bastante úteis na visualização das configurações geométricas otimizadas. Os programas auxiliares em geral geram arquivos ASCII em formatos reconhecidos por outros programas gráficos como *Geomview* (.skel e .off), *X3D* (.poly), *POV-Ray* (.inc) e *Wire4* (.w4).

TriangToPov

- Descrição: Este programa escreve uma configuração 3D no formato reconhecido pelo programa de traçado de raios *POV-Ray*.

- Sintaxe:
  `TriangToPov -inTp ⟨name⟩ -inSt3 ⟨name⟩ [ -outFile ⟨name⟩ ] [ -all ]`

- Opções:
  `-inTp ⟨name⟩`, especifica o arquivo de entrada que contém a topologia do mapa.

  `-inSt3 ⟨name⟩`, especifica o arquivo que contém a configuração 3D do mapa.

  `-outFile ⟨name⟩`, especifica o arquivo de saída no formato ".inc".

`-all`, esta opção considera para efeitos de desenho todos os elementos da triangulação $\mathcal{T}$, caso contrário somente faces existentes.

`TriangToWire4`

- Descrição: Este programa escreve uma configuração 4D no formato reconhecido pelo visualizador gráfico interativo *Wire4*.

- Sintaxe:

```
TriangToWire4 -inTp ⟨name⟩ -inSt ⟨name⟩ [ -outFile ⟨name⟩ ]
              [ -From4 ⟨num⟩ ⟨num⟩ ⟨num⟩ ⟨num⟩ ]
              [ -To4 ⟨num⟩ ⟨num⟩ ⟨num⟩ ⟨num⟩ ]
              [ -Up4 ⟨num⟩ ⟨num⟩ ⟨num⟩ ⟨num⟩ ]
              [ -Over4 ⟨num⟩ ⟨num⟩ ⟨num⟩ ⟨num⟩ ]
              [ -Vangle4 ⟨num⟩ ]
              [ -From3 ⟨num⟩ ⟨num⟩ ⟨num⟩ ]
              [ -To3 ⟨num⟩ ⟨num⟩ ⟨num⟩ ]
              [ -Up3 ⟨num⟩ ⟨num⟩ ⟨num⟩ ]
              [ -Vangle3 ⟨num⟩ ]
              [ -DepthCueLevels ⟨num⟩ ]
              [ -FogDensity ⟨num⟩ ]
```

- Opções:

  `-inTp` ⟨`name`⟩, especifica o arquivo de entrada que contém a topologia do mapa.

  `-inSt` ⟨`name`⟩, especifica o arquivo de entrada que contém a geometria do mapa.

  `-outFile` ⟨`name`⟩, especifica o arquivo de saída no formato ".w4".

  `-From4`, `-To4`, `-Up4`, `-Over4`, `-Vangle4`; especifica os parâmetros de visualização 4D.

  `-From3`, `-To3`, `-Up3`, `-Vangle3`; especifica os parâmetros de visualização 3D.

  `-DepthCueLevels`, `-FogDensity`; especifica parâmetros de visualização adicionais.

`TriangToGeom`

- Descrição: Este programa escreve uma dada configuração 3D em formatos reconhecidos pelo programa *Geomview*.

- Sintaxe:

  `TriangToGeom -inTp ⟨name⟩ -inSt3 ⟨name⟩ [ -outFile ⟨name⟩ ] [ -all ]`

- Opções:
  `-inTp` ⟨`name`⟩, especifica o arquivo de entrada que contém a topologia do mapa.

  `-inSt3` ⟨`name`⟩, especifica o arquivo que contém a configuração 3D do mapa.

  `-outFile` ⟨`name`⟩, especifica o arquivo de saída nos formatos ".skel" ou ".off".

  `-all`, esta opção considera para efeitos de desenho todas as faces da triangulação $\mathcal{T}$, caso contrário somente as faces existentes.

`TriangToX3D`

- Descrição: Este programa escreve uma configuração 3D no formato reconhecido pelo programa *X3D*.

- Sintaxe:
  `TriangToX3D -inTp` ⟨`name`⟩ `-inSt3` ⟨`name`⟩ `[ -outFile` ⟨`name`⟩ `] [ -all ]`

- Opções:
  `-inTp` ⟨`name`⟩, especifica o arquivo de entrada que contém a topologia do mapa.

  `-inSt3` ⟨`name`⟩, especifica o arquivo que contém a configuração 3D do mapa.

  `-outFile` ⟨`name`⟩, especifica o o arquivo de saída no formato ".poly".

  `-all`, esta opção considera para efeitos de desenho todas as faces da triangulação $\mathcal{T}$, caso contrário somente faces existentes.

`WhatDegenerations`

- Descrição: Este programa permite a detecção de elementos topológicos envolvidos em degenerações geométricas.

- Sintaxe:
  `WhatDegenerations -inFile` ⟨`name`⟩ `[ -detail ]`

- Opções:
  `-inFile` ⟨`name`⟩, especifica o arquivo de entrada que contém a topologia do mapa.

  `-detail`, esta opção, se especificada, imprime informações detalhadas a respeito dos elementos da triangulação envolvidos em degenerações geométricas.

A.3 Programas de Visualização

DepthCueing4D

- Descrição: Este programa implementa a técnica de tonalização por profundidade (vide seção 8.3.2).
- Sintaxe:

```
DepthCueing4D -inTp ⟨name⟩ -inSt ⟨name⟩ [ -outFile ⟨name⟩ ]
              -projection [ Perspective | Parallel ]
              [ [ -autoProject ] |
                [ -From4 ⟨num⟩ ⟨num⟩ ⟨num⟩ ⟨num⟩ ]
                [ -To4 ⟨num⟩ ⟨num⟩ ⟨num⟩ ⟨num⟩ ]
                [ -Up4 ⟨num⟩ ⟨num⟩ ⟨num⟩ ⟨num⟩ ]
                [ -Over4 ⟨num⟩ ⟨num⟩ ⟨num⟩ ⟨num⟩ ]
              ]
              [ -Vangle4 ⟨num⟩ ]
              [ -printDepth ]
              [ -ColorFar ⟨num⟩ ⟨num⟩ ⟨num⟩ ]
              [ -ColorNear ⟨num⟩ ⟨num⟩ ⟨num⟩ ]
              [ -DepthFar ⟨num⟩ ]
              [ -DepthNear ⟨num⟩ ]
              [ -DepthCueLevels ⟨num⟩ ]
              [ -Silhouette [ -color | -opacity ] ]
```

- Opções:

  `-inTp` ⟨`name`⟩, especifica o arquivo de entrada que contém a topologia do mapa.

  `-inSt` ⟨`name`⟩, especifica o arquivo de entrada que contém a geometria do mapa.

  `-outFile` ⟨`name`⟩, especifica o arquivo de saída no formato ".inc".

  `-From4`, `-To4`, `-Up4`, `-Over4`, `-Vangle4`; especifica os parâmetros de visualização 4D.

  `-printDepth`, imprime a profundidade 4D dos vértices da triangulação.

  `-ColorFar`, especifica a cor atribuída aos pontos mais afastados do observador no $\mathbb{R}^4$.

  `-ColorNear`, especifica a cor atribuída aos pontos mais próximos do observador no $\mathbb{R}^4$.

`-DepthFar, DepthNear`, especifica os parâmetros de distância para cálculo da tonalização por profundidade.

`-DepthCueLevels`, especifica os níveis desejados para cálculo da tonalização por profundidade.

`-Silhouette`, esta opção, permite a inclusão no desenho das faces de silhueta.

`-color, -opacity`, especifica a cor e índice de opacidade atribuído às faces de silhueta.

Visibility

- Descrição: Este programa implementa técnica de visualização *eliminação de partes ocultas* (vide seção 8.3 para maiores detalhes).

- Sintaxe:

```
Visibility -inTp ⟨name⟩ -inSt ⟨name⟩ [ -outFile ⟨name⟩ ]
           -side [ both | front | back ]
           [ -silhouette [ -color | -opacity ] ]
           [ -projection [ Perspective | Parallel ]
           [ [ -autoProject ] |
             [ -From4 ⟨num⟩ ⟨num⟩ ⟨num⟩ ⟨num⟩ ]
             [ -To4 ⟨num⟩ ⟨num⟩ ⟨num⟩ ⟨num⟩ ]
             [ -Up4 ⟨num⟩ ⟨num⟩ ⟨num⟩ ⟨num⟩ ]
             [ -Over4 ⟨num⟩ ⟨num⟩ ⟨num⟩ ⟨num⟩ ]
           ]
           [ -Vangle4 ⟨num⟩ ]
```

- Opções:

`-inTp ⟨name⟩`, especifica o arquivo de entrada que contém a topologia do mapa.

`-inSt ⟨name⟩`, especifica o arquivo de entrada que contém a geometria do mapa.

`-outFile ⟨name⟩`, especifica o arquivo de saída no formato ".inc".

`-silhouette`, esta opção, se especificada, inclui no desenho as faces de silhueta.

`-color`, `opacity`; especifica a cor e índice de opacidade das faces de silhueta.

`-From4`, `To4`, `Up4`, `Over4`, `Vangle4`; especifica os parâmetros de visualização 4D.

`-side`, especifica o tipo de projeção desejado — `both, front, back`.

A.4 Formatos dos arquivos

Os programas descritos acima comunicam-se via arquivos ASCII. Os principais formatos de arquivos são: Topologia (".tp"), Estado 4D (".st"), Estado 3D (".st3") e Materiais (".ma"). As bibliotecas `libm3triang` e `libm3nontriang` contém as rotinas necessárias para a escrita e leitura destes arquivos. A seguir detalhamos a estrutura destes arquivos.

A.4.1 Topologia

Este arquivo é usado para descrever a topologia de um mapa 3D. Basicamente contém as relações de conexidade entre pares face-aresta da configuração topológica.

`begin topology (format of` $\langle$*data*$\rangle$`)`
`vertices` $\langle$*número*$\rangle$
`edges` $\langle$*número*$\rangle$
`faces` $\langle$*número*$\rangle$
`polyhedra` $\langle$*número*$\rangle$
$\langle$*face-aresta 0*$\rangle$
$\langle$*face-aresta 1*$\rangle$
$\vdots$
$\langle$*face-aresta* $n-1\rangle$
`end topology`

Cada $\langle$*face-aresta i*$\rangle$ contém a informação : tabela `Fnext`, tabela de origens com os atributos `Org, Dest, Pneg` e `Ppos`, componente face e componente aresta.

A.4.2 Estado 4D

Este arquivo é usado para descrever a geometria 4D (coordenadas dos vértices) de uma dada configuração. O formato é:

`begin state (format of` $\langle$*data*$\rangle$`)`
`vertices` $\langle$*número*$\rangle$
$\langle$*vértice 0*$\rangle$
$\langle$*vértice 1*$\rangle$
$\vdots$
$\langle$*vértice* $(n-1)\rangle$
`end state`

Onde cada $\langle$*vértice* $i\rangle$ contém as coordenadas cartesianas 4D do vértice i.

A.4.3 Estado 3D

Este arquivo é usado para descrever a geometria 3D (coordenadas dos vértices) de uma dada configuração. O formato é:

`begin state3D (format of` $\langle$*data*$\rangle$`)`
`vertices` $\langle$*número*$\rangle$
$\langle$*vértice 0*$\rangle$
$\langle$*vértice 1*$\rangle$
$\vdots$
$\langle$*vértice* $(n-1)\rangle$
`end state3D`

Onde cada $\langle$*vértice i*$\rangle$ contém as coordenadas cartesianas 3D do vértice i.

A.4.4 Materiais

Este arquivo descreve os atributos dos elementos do mapa tais como cores, transparência, raio para o desenho, além dos atributos de existência e degeneração, etc:

`begin materials (format of` $\langle$*data*$\rangle$`)`
`| Vertex data:`
$\langle$*vértice 0*$\rangle$
$\langle$*vértice 1*$\rangle$
$\vdots$
$\langle$*vértice* $(n-1)\rangle$
`| Edge data:`
$\langle$*aresta 0*$\rangle$
$\langle$*aresta 1*$\rangle$
$\vdots$
$\langle$*aresta* $(n-1)\rangle$
$|$ `Face data:`
$\langle$*face 0*$\rangle$
$\langle$*face 1*$\rangle$
$\vdots$
$\langle$*face* $(n-1)\rangle$
$|$ `Polyhedron data:`
$\langle$*poliedro 0*$\rangle$
$\langle$*poliedro 1*$\rangle$
$\vdots$

⟨*poliedro* $(n-1)$⟩

`end materials`

Onde para cada k-elemento: ⟨*vértice i*⟩,⟨*aresta i*⟩, ⟨*face i*⟩ e ⟨*poliedro i*⟩ é listado os atributos das propriedades materiais.

A.5 Utilização dos programas

A figura A.1 resume as principais tarefas envolvidas na visualização automática dos mapas 3D. Basicamente são três etapas principais — Modelagem da topologia, Otimização das configurações geométricas e por último a Visualização das configurações ótimas.

Figura A.1: Diagrama mostrando as principais etapas na visualização automática dos mapas 3D.

A figura A.2 ilustra de maneira mais detalhada o processamento descrito nas linhas acima. Assim, um típico processamento para a visualização automática dos mapas 3D começa com a construção do modelo topológico do mapa, feita através de um dos programas — `MakePolytope`, `MakeMap`, etc.

A seguir, efetuamos, se necessária, a construção da subdivisão baricêntrica do mapa. Esta subdivisão torna-se obrigatória em todo mapa 3D que não é uma triangulação. A subdivisão baricêntrica é um refinamento padrão de qualquer mapa $\mathcal{C}$ em simplexos topológicos próprios (vide seção 4.2). Porém, após este processo o modelo pode apresentar degenerações geométricas. Contamos com o programa `SelectSubdivision` que permite eliminar estas degenerações (vide seção 4.4).

Uma simples etapa da subdivisão baricêntrica ou seletiva pode não produzir uma triangulação suficientemente fina. A fim de obter configurações suaves, aplicamos refinamentos sucessivos através do programa `RefineTriang`.

Se necessário, efetuamos também uma atribuição de coordenadas aleatórias para os vértices do mapa (ou da subdivisão/refinamento do mapa), através do programa `RandomShape`. Em seguida, usamos o programa `OptShape` para determinar uma configuração ótima, segundo uma função energia especificada. Se optamos por uma abordagem multiescala, então a configuração é refinada e novamente otimizada.

Neste ponto, temos várias opções de visualização. Optamos primeiro pela opção da visualização semi-interativa após projeções do modelo $\phi(\mathcal{T})$ para o $\mathbb{R}^3$, obtidas através do programa `ProjectTo3D`. Checamos o resultado da otimização com os programas auxiliares de desenho `TriangToX3D`, `TriangToPov` e `TriangToGeom`. Escolhemos as melhores realizações e rodamos algumas vezes mais o programa `OptShape`, geralmente este processo é iterado manualmente variando os seus diversos parâmetros. Uma vez escolhido uma configuração ótima que realmente revele a topologia do mapa 3D usamos os programas especializados de visualização `Visibility`, `Halo`, `DepthCueing4D`, etc.

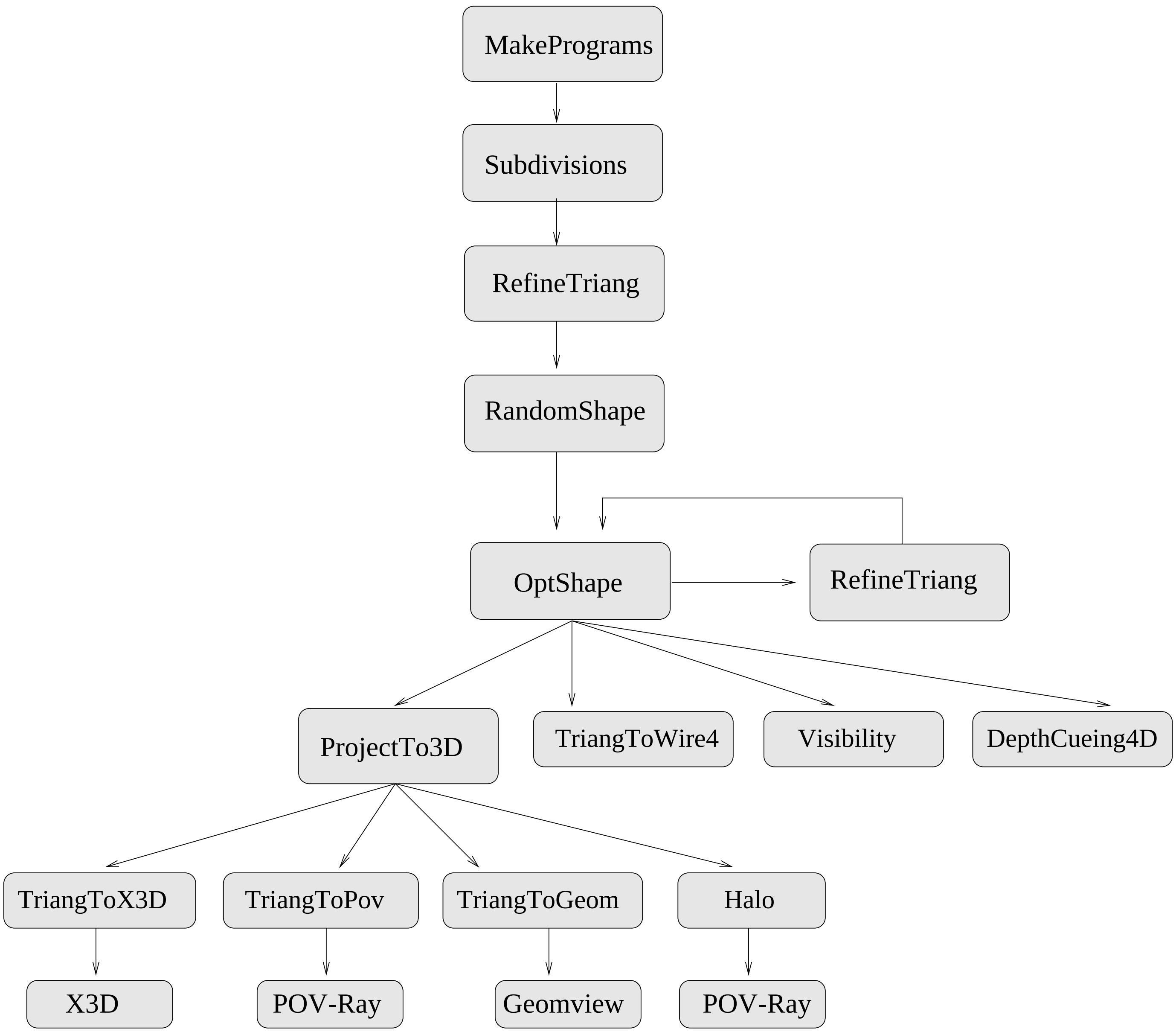


Figura A.2: Diagrama mostrando o uso dos programas.

A.6 Bibliotecas

Os programas de modelagem e otimização utilizam várias bibliotecas de Modula 3, incluindo geometria projetiva (`libm3geo`), otimização (`libm3min`), implementados por Stolfi [36]. Especificamente para este trabalho nós desenvolvemos as bibliotecas: `libm3triang`, para a modelagem topológica de triangulações; `libm3nontriang`, para a modelagem topológica de não triangulações e `libm3energy` para o computo das funções de energia. O programa de visualização *Wire4*, utiliza a biblioteca gráfica *GL* (*Graphics Library*).

A.6.1 `libm3triang`

A biblioteca `libm3triang` contém os seguintes módulos:

`Octf:`

Este programa implementa a estrutura de dados *facet-edge* [7], para a manipulação e representação da topologia dos mapas 3D e de suas triangulações.

`Triangulation:`

Este programa fornece um amplo conjunto de procedimentos para a modelagem e manipulação de triangulações tridimensionais. Assim como procedimentos para a leitura e escrita dos principais formatos de arquivos descritos na seção A.4, este programa fornece também um conjunto de procedimentos que implementa as principais operações geométricas no espaço $\mathbb{R}^4$.

`Tridimensional:`

Este programa provê procedimentos para a escrita e leitura de estados 3D, assim como procedimentos para o cálculo de operações geométricas no $\mathbb{R}^3$.

`Tools:`

Este programa apresenta uma miscelânea de funções e procedimentos auxiliares.

Apêndice B

Conceitos elementares de topologia

Esta seção, extraída da dissertação de Rosi [30], tem como finalidade recordar os conceitos básicos de topologia e estabelecer a notação correspondente. Não é nossa pretensão nos aprofundarmos nestes conceitos. Para maiores detalhes, recomendamos consultar os livros textos padrões de topologia [19, 24, 34, 35].

B.1 Espaço topológico

Um *espaço topológico* é um par $(X, \mathcal{T})$ onde X é um conjunto não vazio e $\mathcal{T}$ é uma família de subconjuntos de X satisfazendo as seguintes condições:

- $\emptyset \in \mathcal{T}$;
- $X \in \mathcal{T}$;
- se $\mathcal{A} \subseteq \mathcal{T}$ então $\cup\mathcal{A} \in \mathcal{T}$;
- se $\mathcal{A} \subseteq \mathcal{T}$ e $\mathcal{A}$ é finito então $\cap\mathcal{A} \in \mathcal{T}$.

Os membros de $\mathcal{T}$ são chamados *conjuntos abertos* do espaço $(X, \mathcal{T})$ e a família $\mathcal{T}$ é a *topologia* do mesmo. Se $U \in \mathcal{T}$, então seu complemento $X \setminus U$ é um *conjunto fechado* do espaço. No restante deste apêndice, vamos supor que $(X, \mathcal{T})$ é um espaço topológico.

B.2 Continuidade e homeomorfismo

Uma função f de um espaço topológico $(X, \mathcal{T})$ para um espaço topológico $(X', \mathcal{T}')$, é dita *contínua* se para todo conjunto aberto U de X' a imagem inversa $f^{-1}(U)$ é um aberto de X.

Um *homeomorfismo* entre dois espaços $(X, \mathcal{T})$ e $(X', \mathcal{T}')$ é uma função bijetora contínua f de $(X, \mathcal{T})$ para $(X', \mathcal{T}')$, cuja inversa f^{-1} também é contínua. Diz-se então que os dois espaços são *homeomorfos* ou *topologicamente equivalentes*.

Uma *propriedade topológica* é uma propriedade de pontos e subconjuntos de um espaço $(X, \mathcal{T})$ que pode ser ser definida apenas em termos dos conjuntos abertos de $\mathcal{T}$ (e das operações da teoria de conjuntos). Dizemos que um espaço X é *conexo* se não existe nenhum subconjunto próprio e não-vazio de X que é ao mesmo tempo fechado e aberto. Dizemos que um conjunto de pontos Z num espaço X *separa* dois pontos $u, v \in X \setminus Z$, se não existe nenhum sub-espaço de $X \setminus Z$ que seja conexo e contenha u e v. Obviamente, um homeomorfismo preserva todas as propriedades topológicas de pontos e subconjuntos de pontos.

B.3 Topologia da reta real

O exemplo mais importante de espaço topológico é $(\mathbb{R}, \mathcal{T}_{\mathbb{R}})$, onde $\mathbb{R}$ é o conjunto dos números reais e um conjunto $U \subseteq \mathbb{R}$ pertence à família $\mathcal{T}_{\mathbb{R}}$ se e somente se todo ponto de U pertence a algum intervalo aberto de $\mathbb{R}$ que está totalmente contido em U. A família $\mathcal{T}_{\mathbb{R}}$ é a *topologia padrão* de $\mathbb{R}$.

B.4 Espaço produto

Se $(X_1, \mathcal{T}_1)$ e $(X_2, \mathcal{T}_2)$ são espaços topológicos, definimos o *espaço produto* $(X_1, \mathcal{T}_1) \times (X_2, \mathcal{T}_2)$ como sendo o par $(X_1 \times X_2, \mathcal{T})$, sendo que um subconjunto $U \subseteq X_1 \times X_2$ pertence a $\mathcal{T}$ se e somente para todo ponto p de U existem $U_1 \in \mathcal{T}_1$ e $U_2 \in \mathcal{T}_2$ tais que $p \in U_1 \times U_2 \subseteq U$.

O produto de espaços topológicos é associativo e portanto esta definição pode ser estendida a produtos múltiplos

$$(X_1, \mathcal{T}_1) \times (X_2, \mathcal{T}_2) \times \cdots \times (X_n, \mathcal{T}_n)$$

e potências cartesianas $(X, \mathcal{T})^n$. Desta forma obtemos, em particular, a *topologia padrão* do espaço cartesiano $\mathbb{R}^n$, para todo $n > 1$.

B.5 Sub-espaços

Seja Y um subconjunto não vazio de X e seja $\mathcal{T}(Y)$ a família $\{\, U \cap Y \;:\; U \in \mathcal{T} \,\}$. Prova-se que $(Y, \mathcal{T}(Y))$ também é um espaço topológico; ele é o *sub-espaço de* $(X, \mathcal{T})$ *induzido por* Y. A família $\mathcal{T}(Y)$ é a *topologia de* Y *relativa a* (ou *induzida por*) $(X, \mathcal{T})$. O conceito de sub-espaços permite-nos construir facilmente muitos espaços topológicos interessantes:

qualquer subconjunto do $\mathbb{R}^n$ — um segmento, uma esfera, uma curva fechada, um disco com buracos — é automaticamente espaço topológico, com a topologia induzida pela topologia padrão do $\mathbb{R}^n$.

B.6 Espaço quociente

Se $\equiv$ é uma relação de equivalência sobre os pontos de um espaço $(X, \mathcal{T})$, definimos o *espaço quociente* $(X, \mathcal{T})/\equiv$ como sendo $(X', \mathcal{T}')$, onde X' é o conjunto $X/\equiv$ das classes de equivalência de X por $\equiv$; e $\mathcal{T}'$ é a família de todos os conjuntos $A/\equiv$, para todo $A \in \mathcal{T}$ que é união de classes de $X/\equiv$. O conceito de espaço quociente foi usado no capítulo 2.

B.7 Separabilidade

Dois pontos de um espaço topológico são *separáveis* se existe um conjunto aberto que contém um ponto mas não o outro; caso contrário, eles são *inseparáveis*. Um espaço é *separável* se todos os pares de pontos são separáveis.

Do ponto de vista topológico, dois pontos inseparáveis são essencialmente indistinguíveis. Não estaremos perdendo muita coisa interessante se restringirmos nossa atenção a espaços separáveis; pois o quociente de qualquer espaço por sua relação de inseparabilidade é um espaço separável, cujos abertos correspondem aos abertos do espaço original de maneira biunívoca e compatível com $\subseteq$.

B.8 Vizinhança, fecho e fronteira

Se p é um ponto de X e V é um subconjunto de X tal que $p \in U \subseteq V$ para algum conjunto aberto U de $\mathcal{T}$, então dizemos que V é uma *vizinhança de* p na topologia $\mathcal{T}$.

Mais geralmente, se A é um subconjunto de X, dizemos que V é uma vizinhança de A em $\mathcal{T}$ se V for uma vizinhança de todo ponto p de A.

Se $A \subseteq X$, dizemos que um ponto $p \in X$ pertence ao *interior a* A na topologia $\mathcal{T}$ se e somente se A é uma vizinhança de p relativa a $\mathcal{T}$. Por outro lado, dizemos que p pertence ao *fecho* de A relativo a $\mathcal{T}$ se toda vizinhança de p encontra A.

Se denotarmos por ιA o interior de A e κA seu fecho (numa determinada topologia $\mathcal{T}$), podemos provar que

$$X \setminus \kappa(X \backslash A) = \iota A \subseteq A \subseteq \kappa A = X \setminus \iota(X \backslash A) \; .$$

O conjunto $\kappa A \backslash \iota A$ é por definição a *fronteira de* A na topologia $\mathcal{T}$; ele contém todo ponto $x \in X \setminus A$ tal que qualquer vizinhança de x encontra A.

B.9 Limite, incidência, compacidade

Seja A um subconjunto de X e p um ponto de X. Dizemos que p é um *ponto de acumulação* (ou *ponto limite*) *de A em $\mathcal{T}$* se e somente se qualquer vizinhança de p contém um número infinito de pontos de A.

Dizemos que um conjunto $A \subseteq X$ *incide* num conjunto $B \subseteq X$ se B possui um ponto de acumulação de A.

Um espaço topológico é *compacto* se todo conjunto infinito de pontos tem um ponto de acumulação.

Bibliografia

[1] W. Baur and V. Strassen. The complexity of partial derivatives. *Theoretical Computer Science*, 22:317–330, 1983.

[2] V. Bulatov. Polyhedra collection. *http://www.physics.orst.edu/~bulatov/polyhedra*, 2000.

[3] S. A. Carey, R. P. Burton, and D. M. Campbell. Shades of a higher dimension. *Computer Graphics World*, X(10):93–94, Outubro 1987.

[4] H. S. M. Coxeter. *Regular Polytopes.* Dover Publications, inc., New York, 1973.

[5] D. Davis. Construction of 120-cell and 600-cell. *http://olympus.math.niu.edu/~rusin/known-math/99/120cell*, 1999.

[6] M. Dehn. Über die topologie des dreidimensional raumes. *Math. Ann*, 69:137–168, 1910.

[7] D. P. Dobkin and M. J. Laszlo. Primitives for the manipulation of three-dimensional subdivisions. *Algorithmica*, 4:3–32, 1989.

[8] J. D. Foley and A. Van Dam. *Fundamentals of Interactive Computer Graphics.* Addison-Wesley Publishing Company, Reading, MA, 1982.

[9] P. J. Giblin. *Graphs, Surfaces and Homology.* Chapman and Hall, London, 1977.

[10] S. Gitler. Immersion and emdedding of manifolds. *Algebraic topology, Proc. Sympos. Pure Math.*, XXII:87–96, 1970.

[11] B. S. Gottfried. *Programando em C.* Makron Books do Brasil Editora Ltda., São Paulo, 1993.

[12] C. Gunn and D. Maxwell. Not knot. *Jones and Bartlett*, 1991.

[13] A. J. Hanson and A. Pheng. Illuminating the fourth dimension. *Comp. Graphics and Appl.*, 12(4):54–62, 1992.

[14] Sam P. Harbison. *Modula-3.* Prentice-Hall, New Jersey, 1992.

[15] P. Heegaard. Sur l'analysis situs. *Soc. Math. France Bull*, 44:161–242, 1916.

[16] S. R. Hollasch. Four-space visualization of 4D objects. Master's thesis, Arizona State University, August 1991. http://research.microsoft.com/~hollasch/.

[17] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Mesh Optimization. *Comp. Graphics Proceedings, Annual Conference Series*, pages 19–26, 1993.

[18] T. Kamada and S. Kawai. An algorithm for drawing general undirected graphs. *Information Processing Letters*, 31(1):7–15, 1989.

[19] C. Kosniowski. *A First Course in Algebraic Topology.* Cambridge University Press, Cambridge, 1980.

[20] S. Levy, T. Munzner, and M. Phillips. Geomview 1.6.1 OpenGL. *http://www.geom.umn.edu*, 1996.

[21] R. L. W. Liesenfeld. Animação dinâmica de corpos elásticos. Master's thesis, Instituto de Computação - UNICAMP, 1997.

[22] S. Lins and A. Mandel. Graph-encoded 3-manifolds. *Discrete Mathematics*, 57:261–284, 1985.

[23] L. P. Lozada, C. F. X. de Mendonça, R. M. Rosi, and J. Stolfi. Automatic visualization of two-dimensional celullar complexes. *Graph Drawing'96 - Lecture Notes in Computer Science*, pages 303–317, 1996.

[24] W. S. Massey. *Algebraic Topology: An Introduction.* (Graduate texts in mathematics; 56), Springer-Verlag, New York, 1977.

[25] Greg Nelson. *Systems Programming with Modula-3.* Prentice-Hall, New Jersey, 1991.

[26] M. A. Noll. A computer technique for displaying n-dimensional hyperobjects. *Communications of the ACM*, 10(8):469–473, 1967.

[27] A. Paoluzzi. Motion planning + solid modeling = motion modeling. *Tech. Rep. 17-89, Dip di Informatica e Sistemistica*, 1989. Univ. di Roma "La Sapienza", Rome, Italy.

[28] H. Poincaré. Analysis situs. *J. de l' École Polyt.*, 2(1):1–123, 1895.

[29] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes: The Art of Scientific Computing.* Cambridge University Press, 1986.

[30] R. M. Rosi. Visualização automática de complexos celulares arbitrários. Master's thesis, Instituto de Computação - UNICAMP, 1995.

[31] R. M. Rosi and J. Stolfi. Automatic visualization of two-dimensional celullar complexes. *Technical Report N<u>o</u> IC-96-02, Instituto de Computação, Universidade Estadual de Campinas, Brazil*, 1996.

[32] M. Spychalla. x3d 2.2, A wireframe 3D object viewer. *http://www.efd.lth.se/~e93rlr/*, 1992.

[33] K. V. Steiner and R. P. Burton. Hidden volumes: The 4th dimension. *Computer Graphics World*, X(2):71–74, Fevereiro 1987.

[34] J. Stillwell. *Mathematics and Its History.* (Undergraduate texts in mathematics), Springer-Verlag, New York, 1989.

[35] J. Stillwell. *Classical Topology and Combinatorial Group Theory.* (Graduate texts in mathematics $N^{\underline{o}}$ 72), Springer-Verlag, New York, 1993.

[36] J. Stolfi. Modula-3 code. *http://www.dcc.unicamp.br/~stolfi*, 2000.

[37] R. Tamassia and P. Eades. Algorithms for drawing graphs: an annotated bibliography. *Technical report CS-89-09, Brown University, Department of Computer Science*, 1989.

[38] W. P. Thurston and J. R. Weeks. The mathematics of three-dimensional manifolds. *Scientific American*, 251(1):94–106, 1984.