Universidade Federal de Minas Gerais
Department of Electrical Engineering

Delaunay refinement for curved complexes

Adriano Chaves Lisboa

Advisor: Prof. Rodney Rezende Saldanha
Co-advisor: Prof. Ricardo Hiroshi Caldeira Takahashi

Belo Horizonte, September 30, 2008

Abstract

This work investigates the Delaunay refinement for curved complexes. A manifold complex is defined as an unambiguous representation for the geometric objects required by a partial differential equation solver. The Chew's and Ruppert's Delaunay refinement algorithms, including an extension for curved complexes, are described under a new and arbitrary dimensional perspective. A theorem for strongly Delaunay simplicial complexes is extended to higher dimensions, as well as a fundamental theorem of the Bowyer-Watson algorithm is extended to intermediate dimensions in the simplicial complex. Some implementation points are also addressed, as the fan search in the incremental Delaunay simplicial complex update, and robust predicates in arbitrary dimensions.

Resumo

Este trabalho investiga o refinamento Delaunay para complexos curvos. Um complexo de manifold é definido como uma representação única para objetos geométricos requeridos na solução de equações diferenciais parciais. Os algoritmos de Chew e Ruppert, incluindo uma extensão para complexos curvos, são descritos uma nova perspectiva em dimensões arbitrárias. Um teorema para complexos simpliciais fortemente Delaunay é estendido para dimensões superiores, assim como um teorema fundamental do algoritmo de Bowyer-Watson é estendido para dimensões intermediárias no complexo simplicial. Alguns pontos de implementação também são abordados, como uma busca em leque para atualizar de maneira incremental um complexo simplicial de Delaunay, e predicados robustos em dimensões arbitrárias.

About this thesis

This work takes my name as author, but many people were important for its accomplishment. I would like to thank:

- my family for the unconditional support throughout the course;
- my advisor, Professor Rodney Rezende Saldanha, for his enthusiastic and encouraging help in my research, and for the dreamful advices I will carry for my life;
- my co-advisor, Professor Ricardo Hiroshi Caldeira Takahashi, for his kind attention and meaningful review of this text;
- Professor Renato Cardoso Mesquita for the friendly talks, which have changed the theme of this thesis;
- Professor Luis Gustavo Nonato for his valuable remarks during the qualifying exam;
- Professor Hani Camille Yehia for the free clues;
- Douglas Alexandre Gomes Vieira, Xisto Lucas Travassos Júnior, Sérgio Luciano Ávila, Alexandre Ramos Fonseca, and Guilherme Parreira for the partnership and friendship;
- CAPES and CNPq for the research encouragement.

Everybody output according to their knowledge, and input according o their knowledge. An eternal loop of creation and recreation in global networks. Not surprisingly, I am not likely to be the first one to publish these ideas. I do not know how many ideas you will miss in this text by thinking they are not here when I wished to tell you, or how many you will create by thinking they are here when I did not noticed, or how many other combinations I missed in this phrase. That is part of the magic.

Contents

Resumo estendido

Introdução

Uma malha pode ser vista como uma partição de um domínio em elementos geralmente simples e de um mesmo tipo. Ela pode ser aplicada em diversos contextos, como visualização gráfica e interpolação. Como foco de aplicação de malhas neste trabalho, está o método de elementos finitos para soluções de equações diferenciais parciais. Este método é bem restritivo quanto aos requisitos de malha, de modo outras aplicações devem ser capazes de usar o mesmo gerador de malhas.

Para que o método de elementos finitos funcione, a malha deve possuir as propriedades topológicas definidas no conceito de complexo. Para que o método de elementos finitos atinja uma solução suficientemente precisa com o menor número de elementos, existem diretivas gerais que um gerador de malhas deve satisfazer. Primeiramente, um bom gerador de malhas deve ser capaz de gerar elementos não achatados, pois elementos achatados tendem a ser maus interpolantes e a gerar erros numéricos (em alguns casos especiais é interessante ter elementos achatados em certas direções). Outro ponto é que um bom gerador de malha deve ser idealmente capaz de gerar a malha com o menor número de elementos possível, e refinar posteriormente onde for necessário. Esse refinamento deve suportar boas gradações, de modo que a densidade de elementos em uma região não interfira significativamente em outra com densidade diferente.

Quando se tem apenas um gerador de malhas que lida com geometrias planas, é necessário fazer uma aproximação linear da entrada antes de gerar a malha. Em um contexto adaptativo, a informação de como deve ser a aproximação linear da entrada não é conhecida a priori. Neste caso torna-se imprescindível que o gerador de malhas seja capaz de lidar com geometrias curvas, onde as aproximações das partes curvas são devidamente refinadas onde for necessário no decorrer do processo.

Dentre as estratégias para geração de malhas, o refinamento Delaunay ocupa lugar de destaque pela sua elegância e garantias teóricas. Este trabalho estende o refinamento Delaunay para complexos curvos de entrada em dimensões arbitrárias. Para tanto, são estabelecidos dois teoremas principais. Um relacionado com o conceito de fortemente Delaunay e que estabelece condições para que um simplexo pertença ao complexo simplicial Delaunay. Os algoritmos de Ruppert e de Chew são descritos sob o ponto de vista deste teorema. O segundo teorema é uma extensão da idéia fundamental do algoritmo de Bowyer-Watson para inserção incremental em um complexo simplicial Delaunay. Existem também contribuições na parte de implementação, como a busca em leque, a avaliação de predicados robustos em dimensões arbitrárias e a determinação de um ponto de Voronoi sobre peças curvas.

Definição de domínios

Um PLC [Miller et al., 1996] (*piecewise linear complex*) define uma geometria formada por partes lineares, como sugere o nome. Neste trabalho, esta definição é estendida para partes curvas como um k-complexo de variedades definido por uma coleção de n-variedades, $n = 0, ..., k$, conectadas não vazias e disjuntas entre si, onde partes de mesma dimensão são desconexas entre si (o prefixo em complexo e em variedade representa a respectiva dimensão). As diferenças fundamentais na definição de um k-complexo de variedades para a de um PLC é que as peças passam a ser variedades (potencialmente curvas), que k-peças também podem ser definidas e que as peças são conjuntos abertos ao invés de fechados.

Qualquer domínio pode ser particionado em um complexo de variedades de maneira única, como ilustrado nas Figuras 1 e 2.

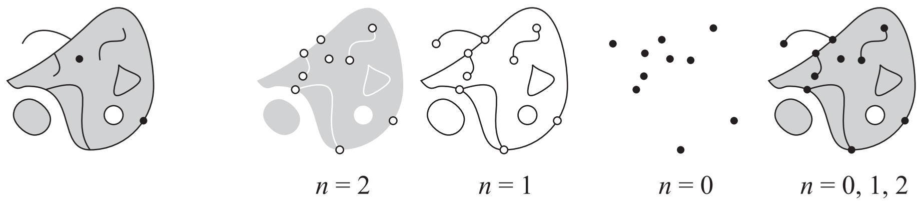

Figure 1: Exemplo de um domínio bidimensional (esquerda) particionado em um 2-complexo de variedades (direita).

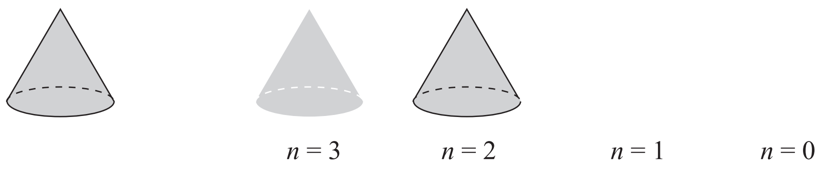

Figure 2: Exemplo de um domínio tridimensional (esquerda) particionado em um 3-complexo de variedades (direita).

Um complexo de variedades C^k é aquele onde todas a variedades membro são C^k. A definição de complexo de variedades garante que ele seja C^0. Para atingir complexos de variedades C^k pode-se definir as partes onde a n-ésima derivada é descontínua como variedades independentes, $n \leq k$, como ilustrado nas Figuras 3 e 4. Dessa maneira propriedades diferenciais são descritas explicitamente no complexo de variedades.

Uma variedade pode apresentar diversas topologias, como ilustrado na Figura 5. Estas características topológicas não podem ser expressas de maneira explicita no complexo de variedades, e assim devem ser identificadas de acordo com a representação das peças. Para evitar tratar os diversos casos especiais, pode-se exigir que

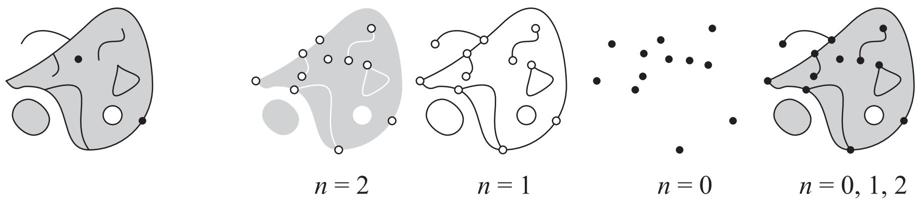

Figure 3: Exemplo de um domínio bidimensional (esquerda) particionado em um 2-complexo de variedades C^1 (direita).

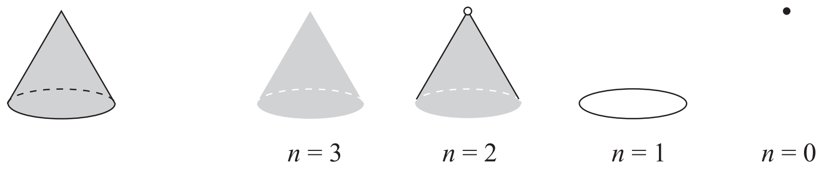

Figure 4: Exemplo de um domínio tridimensional (esquerda) particionado em um 3-complexo de variedades C^1 (direita).

todas a peças tenham a configuração topológica de uma bola, particionando partes que não sejam se necessário.

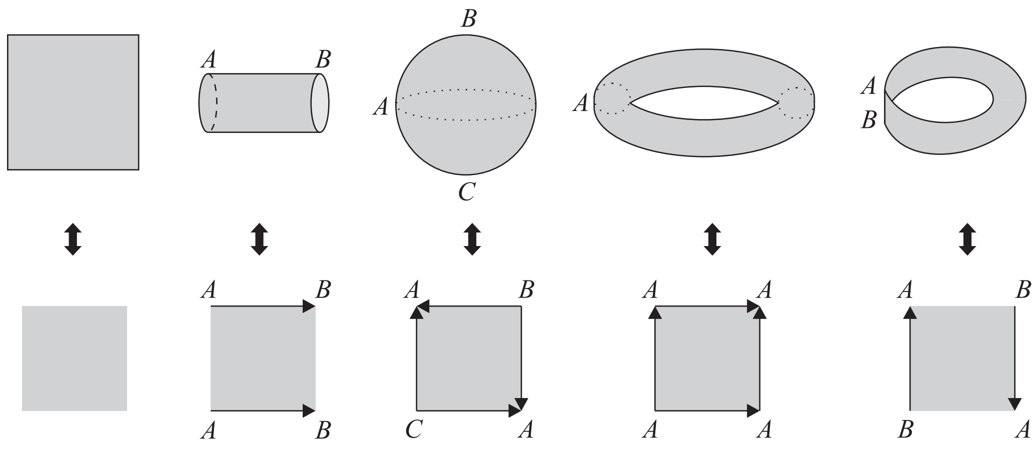

Figure 5: Exemplo de diferentes tipos de configurações topológicas de 2-variedades. Da esquerda para a direita: plano, tudo, esfera, toro and fita de Möbius.

Refinamento Delaunay

Um complexo simplicial pode ser definido como um complexo de variedades onde cada peça é um simplexo e onde as facetas de cada simplexo também fazem parte do complexo, como ilustrado na Figura 6.

Em 1934 Boris Delaunay [Delaunay, 1934] definiu um critério para complexos simpliciais que implicaria em muitas propriedades interessantes. Neste critério, cada simplexo do complexo simplicial possui uma bola circunscrita que não possui nenhum vértice em seu interior. Este complexo simplicial está fortemente relacionado com

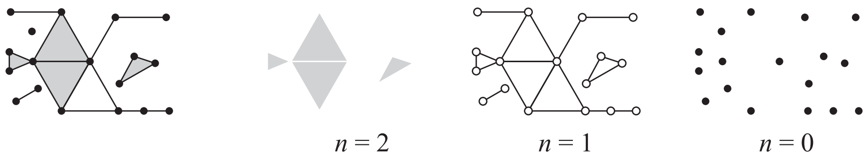

Figure 6: Exemplo de um 2-complexo simplicial.

o diagrama de Voronoi definido por Georgy Voronoi em 1908 [Voronoi, 1908]. O diagrama de Voronoi é um complexo de células compostas por todos os pontos que estão mais perto de um ponto do que qualquer outro ponto de um dado conjunto.

Considerando que cada k-peça do diagrama de Voronoi em $\mathbb{R}^n$ está equidistante a exatamente $n-k+1$ pontos (caso degenerado se forem mais de $n-k+1$ pontos), então cada k-peça do diagrama de Voronoi está unicamente relacionada com um $(n-k)$-simplexo do respectivo complexo simplicial para o mesmo conjunto de pontos, como ilustrado na Figura 7.

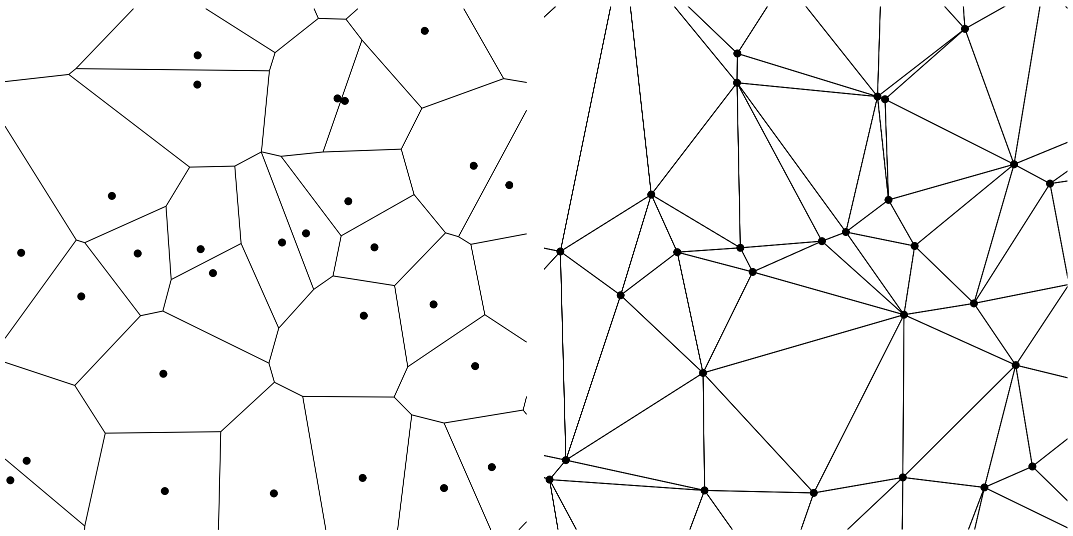

Figure 7: Diagrama de Voronoi e complexo simplicial de Delaunay para o mesmo conjunto de pontos.

Em 1981 David Watson [Watson, 1981] provou que se uma k-peça, $k > 0$, do diagrama de Voronoi é degenerada, então existe pelo menos uma $(< k)$-peça degenerada. Dessa maneira, um Diagrama de Voronoi é degenerado se e somente se existir pelo menos um vértice equidistante a mais de $n + 1$ pontos. Um complexo simplicial de Delaunay dual a um diagrama de Voronoi não degenerado é dito fortemente Delaunay. Este complexo simplicial pode ser visto como aquele onde cada simplexo membro possui uma bola circunscrita que não contém nenhum outro vértice sobre seu contorno ou em seu interior.

Pela unicidade do complexo simplicial fortemente Delaunay é provado neste trabalho que um k-complexo simplicial é fortemente Delaunay se e somente se o subconjunto de n-simplexos é fortemente Delaunay, para um $n \in \{1, ..., k\}$ único e arbitrário. Isto equivale a dizer que se um n-simplexo possui pelo menos uma bola circunscrita vazia então ele faz parte do complexo simplicial fortemente Delaunay, caso contrário ele não pode fazer parte. Esta propriedade é provada por Shewchuk em sua tese [Shewchuk, 1997] apenas para $k = 2$ e utilizando uma outra estratégia. Este teorema será denominado daqui para frente de teorema do fortemente Delaunay.

Em 1987 William Frey [Frey, 1987] introduziu a inserção no circuncentro de um simplexo, originando o refinamento Delaunay (ver Figura 8). Este refinamento possui a propriedade de não criar arestas menores caso forem escolhidos apenas circuncentros de simplexos que possuem circunraio maior que sua menor aresta. Em espaço bidimensionais, isto implica que este refinamento pode gerar garantidamente triângulos com ângulos diedros no intervalo $[30^o, 120^o]$. Infelizmente em dimensões maiores que a segunda não é possível definir limites em medidas de qualidade válidas.

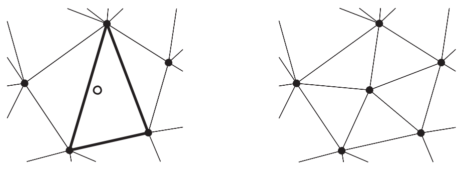

Figure 8: Refinamento Delaunay em uma triangulação.

Utilizando a propriedade de não criar menores arestas, Paul Chew propôs em 1989 [Chew, 1989b] o primeiro algoritmo de refinamento Delaunay para complexos lineares bidimensionais com garantias teóricas de qualidade. Este algoritmo pré-segmenta a entrada de maneira que todos os vértices se encontram a uma distância $[h, h\sqrt{3}]$ entre si, e depois refina todos os triângulos com circunraio maior que h, como ilustrado na Figura 9. Sob o ponto de vista do teorema do fortemente Delaunay, isto garante que os segmentos de entrada terão uma circuncírculo vazio através do processo, e portanto farão parte da triangulação. Este algoritmo pode ser estendido para dimensões arbitrárias de maneira direta, mas o processo de pré-segmentação o torna inviável na maioria dos casos práticos.

Em 1994 Jim Ruppert [Ruppert, 1994] propôs um algoritmo que não necessitava de pré-segmentação da entrada, dando garantias teóricas de qualidade, gradação e tamanho para as triangulações. Sob o ponto de vista do teorema do fortemente Delaunay, este algoritmo primeiro garante que cada aresta representando a entrada terá um círculo diametral vazio (assim ela fará parte da triangulação), e depois refina a triangulação até que critérios sejam atingidos, como ilustrado na Figura 10. A idéia fundamental é que as arestas representando a entrada são refinadas caso seus círculos diametrais sejam invadidos. A primeira etapa é denominada recuperação de facetas, e a segunda de refinamento. Este algoritmo pode ser estendido para dimensões arbitrárias de maneira direta e ainda continua prático.

Em 2002 Charles Boivin e Ollivier-Gooch [Boivin and Ollivier-Gooch, 2002] propuseram uma extensão do algoritmo de Ruppert para lidar com geometrias curvas bidimensionais. Nesta estratégia, as curvas de entrada são segmentadas de maneira que cada subcurva apresente uma variação angular não maior que $\pi/6$, e de maneira que os segmentos não contenham nenhum ponto visível em seu círculo diametral. A

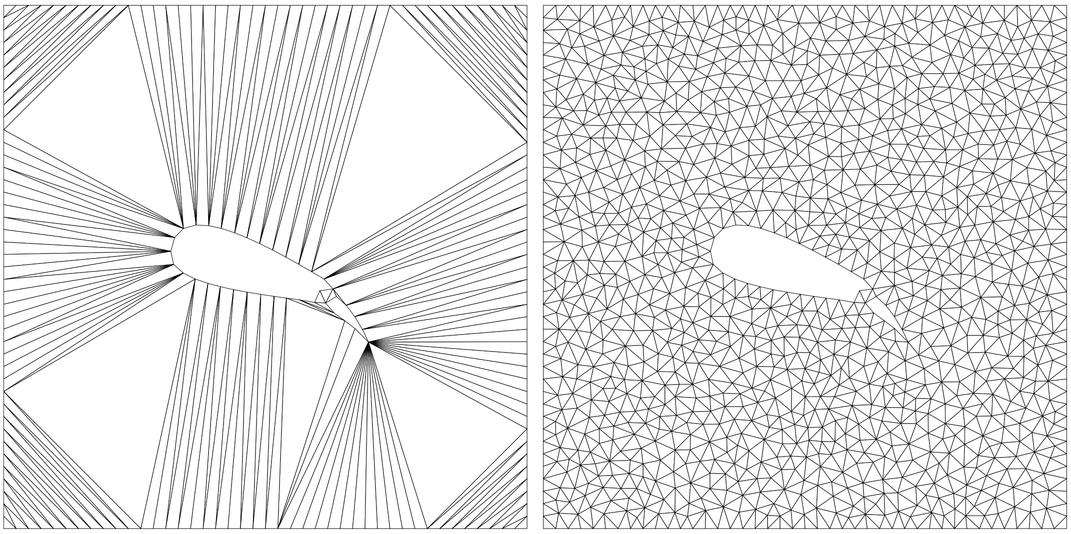

Figure 9: Malhas inicial (esquerda) e final (direita) geradas com o primeiro algoritmo de Chew.

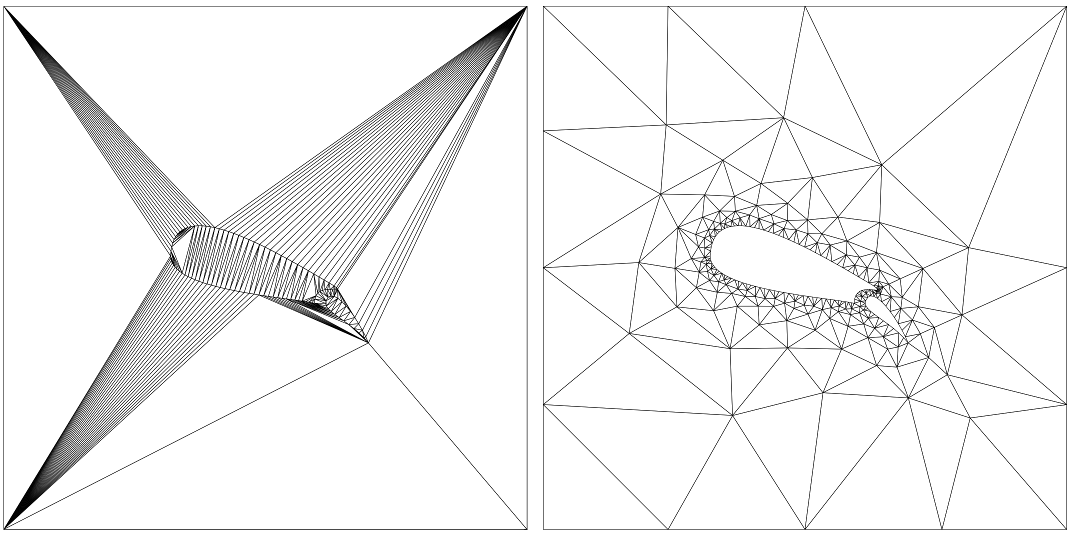

Figure 10: Malhas inicial (esquerda) e final (direita) geradas com o algoritmo de Ruppert.

malha inicial gerada é então refinada até atingir critérios de parada. O comportamento deste algoritmo está ilustrado na Figura 11. O refinamento considera o caso especial onde a inserção de um nó em um segmento causa a interceptação com outro segmento. Neste caso o segmento interceptado é refinado ao invés do segmento que o interceptou.

Para lidar com complexos curvos em dimensões arbitrárias, este trabalho revisita o algoritmo de Ruppert. Essa nova abordagem permite que a malha seja criada dimensão por dimensão, e que o refinamento possa ser feito em qualquer dimensão simultaneamente.

Primeiramente, a etapa de recuperação de facetas pode ser feita de dimensões

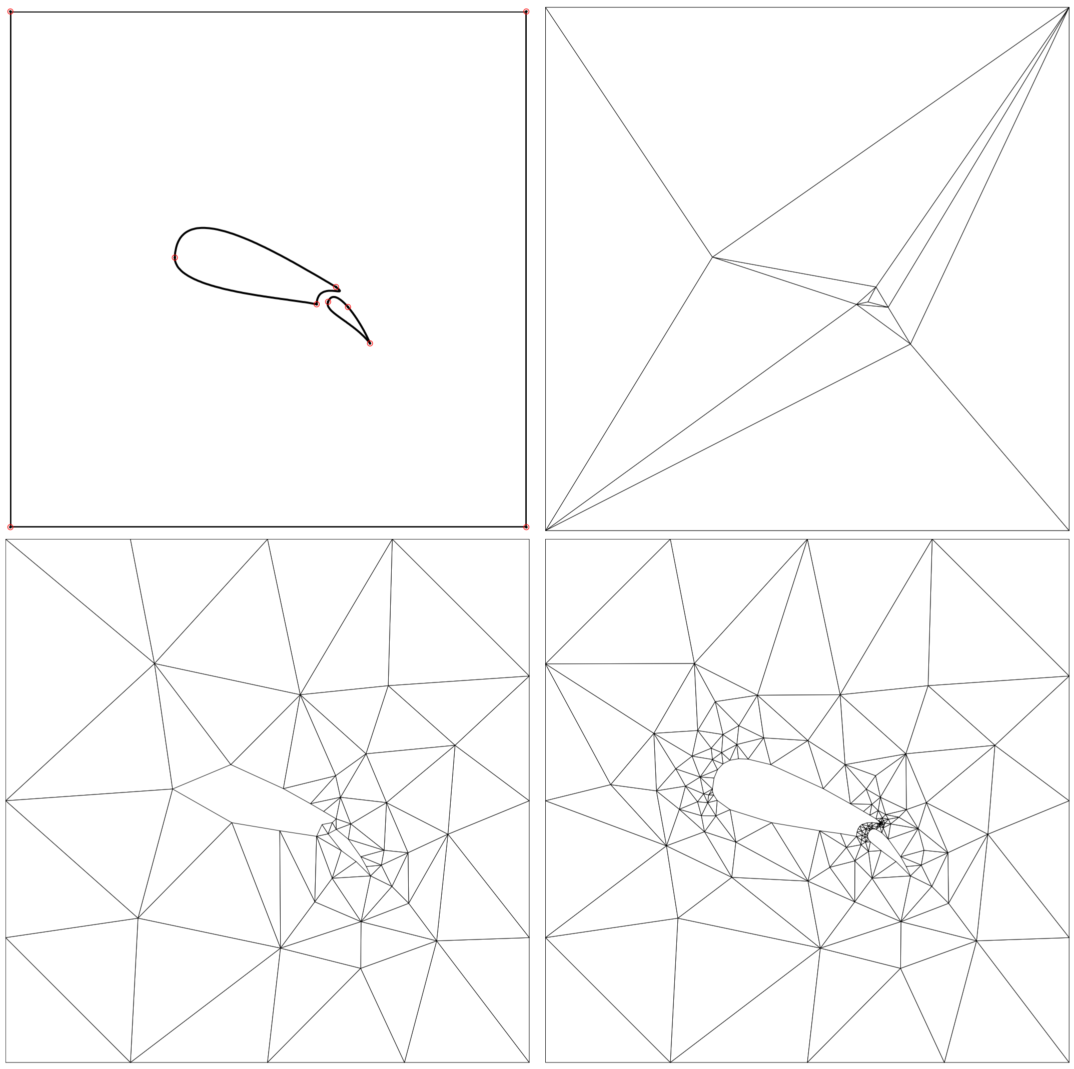

Figure 11: Complexo curvo de entrada (acima-esquerda), malha inicial (acima-direita), malha refinada para qualidade (abaixo-esquerda) e malha refinada para melhor qualidade e precisão de representação (abaixo-direita).

mais baixas para dimensões mais altas. Antes de passar para a dimensão seguinte, os simplexos da dimensão corrente são refinados até que bolas circunscritas pré-estabelecidas sejam vazias, garantindo que eles farão parte dos complexos simpliciais de dimensão superior. Nesta construção, cada peça do complexo de variedades define apenas um espaço paramétrico cujas restrições e fronteiras são dadas por peças de dimensão inferior. Esta etapa está exemplificada na Figura 12.

Para a fase de refinamento, foi definido um teorema que garante que, após a inserção de novo um vértice, os novos simplexos que conformam a geometria irão fazer parte do complexo simplicial Delaunay. Este teorema é uma generalização da idéia fundamental do algoritmo de Bowyer-Watson [Bowyer, 1981, Watson, 1981]. A idéia fundamental deste teorema é que após a inserção de um novo vértice as arestas conectadas a ele terão pelo menos uma bola circunscrita vazia, como exemplificado

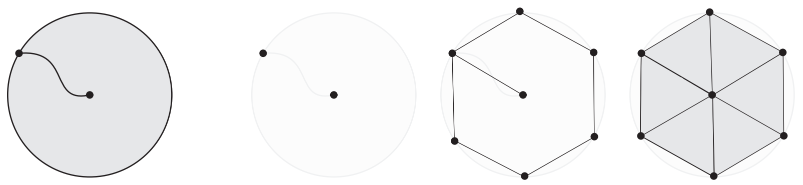

Figure 12: Alguns passos da recuperação de facetas para um complexo de variedades (esquerda).

na Figura 13. Assim, pelo teorema do fortemente Delaunay, elas farão parte do complexo simplicial Delaunay juntamente com os respectivos simplexos de maior dimensão formados. As bolas circunscritas vazias que garantem a presença dos novos simplexos podem não ser as pré-estabelecidas, de modo que os vértices que possivelmente invadirem estas bolas deverão ser removidos.

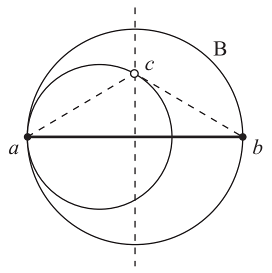

Figure 13: Se a bola diametral $\mathbb{B}$ do segmento ab é vazia, então o subsegmento ac também terá uma bola circunscrita vazia tangente a $\mathbb{B}$ em a desde que $c \in \mathbb{B}$.

Implementação

Cada peça do complexo de variedades é armazenada em um mesmo vetor e sua posição nele a identifica. Cada peça contém uma representação paramétrica com respectiva dimensão, as peças vizinhas como ilustrado na Figura 14 (para a geração de malha é suficiente que sejam armazenados apenas os vizinhos de dimensão inferior), e uma semente que será usada para identificação de simplexos membros.

Para o armazenamento do complexo simplicial, é utilizada uma estrutura de dados específica para simplexos. Cada simplexo contém um ponteiro para cada um de seus vértices, um ponteiro para o vizinho oposto a cada vértice (se o vizinho for de dimensão imediatamente inferior o ponteiro é sinalizado com o bit especial **down**), um ponteiro para cada simplexo de dimensão imediatamente superior conectado a ele (dois ponteiros fixos, sendo que o segundo é sinalizado com o bit especial **joint** caso forem mais de dois), e um ponteiro para a variedade que ele está conformando (ver Figura 15). Cada ponteiro é um índice para o vetor onde está armazenado o objeto apontado, o que facilita a leitura e a paralelização. As principais diferenças para as estruturas de dados apresentadas na literatura é que os vizinhos podem ser

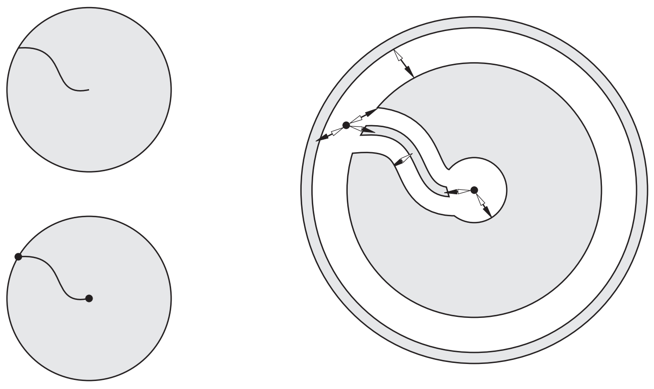

Figure 14: Estrutura de dados (direita) para o complexo de variedades (abaixo-esquerda) para um domínio de entrada (acima-esquerda).

de dimensão imediatamente inferior, e que o ponteiro para a variedade conformada é de obrigatória presença.

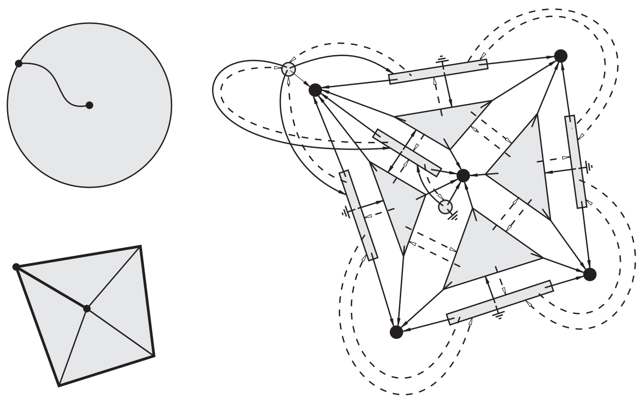

Figure 15: Estrutura de dados (direita) para o complexo simplicial (abaixo-esquerda) homeomórfico a um complexo de variedades (acima-esquerda).

É utilizada uma versão estendida do algoritmo de Bowyer-Watson para fazer a atualização do complexo simplicial Delaunay. Neste processo, a inserção de um novo vértice gera uma cavidade, formada pelos simplexos invadidos, cujo contorno, denominado horizonte, é conectado ao novo vértice para originar os novos simplexos. Para atualizar os vizinhos dos simplexos do horizonte foi desenvolvida neste trabalho uma busca em leque, a qual está graficamente ilustrada na Figura 16. Nesta busca, a faceta do simplexo do horizonte oposta ao vértice cujo vizinho relacionado se deseja determinar, é o "vértice" do leque. A busca segue então pelos vértices da cavidade conectados ao "vértice" do leque.

O predicado que diz de qual lado está um ponto em relação a um hiperplano, assim como aquele que diz se um ponto está dentro ou fora da bola circunscrita de um simplex, é dado pelo sinal do determinante de uma matriz. Pode-se escrever que o sinal do determinante calculado com erros de arredondamento está correto sempre que

$$|A_n| > a_n \epsilon \alpha_n \tag{1}$$

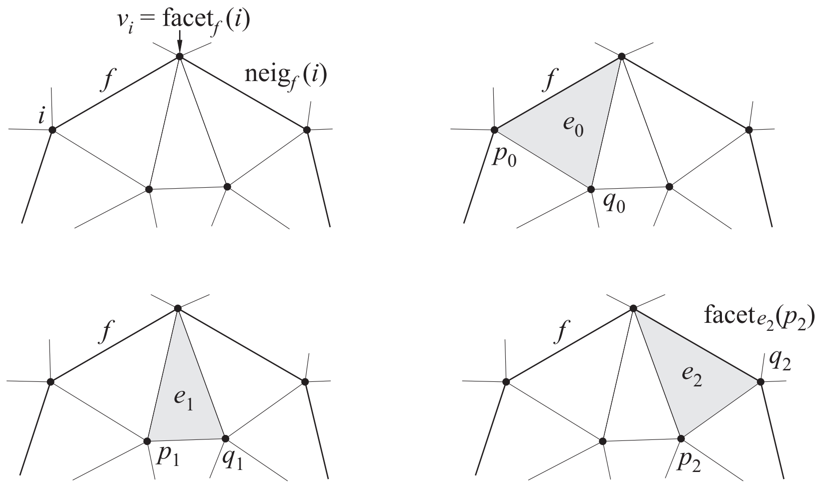

Figure 16: Representação gráfica da busca em leque.

onde A_n é a matriz do predicado na n-ésima dimensão, ϵ é a precisão da máquina, e α_n é o permanente dos valores absolutos de A_n. Os coeficientes a_n para o predicado de orientação em relação ao hiperplano são dados por

$$\frac{n}{2}(n+1) + n - 1 \tag{2}$$

e para o predicado de orientação em relação à bola são dados por

$$\frac{n}{2}(n+1) + 3n + 2 \tag{3}$$

Para calcular o ponto sobre uma n-peça equidistante aos vértices de um n-simplex foi derivado neste trabalho um processo iterativo de Newton-Raphson dado por

$$\begin{aligned} t_{k+1} &= t_k - \frac{f_k}{{f'_k}^T f'_k}\; f'_k \\ &= t_k - \frac{c_k^T c_k}{2(c'_k c_k)^T c'_k c_k}\; c'_k c_k \end{aligned} \tag{4}$$

onde $t \in [0,1]^n$ é o vetor de parâmetros, k é a iteração corrente, $f(t) = \|c(t)\|^2$ é uma função distância, $c(t)$ é a projeção da n-peça no n-plano do n-simplex transladada para o circuncentro, e $f'(t) \in \mathbb{R}^n$ e $c'(t) \in \mathbb{R}^{n\times n}$ são os respectivos gradientes ($c'(t)$ contém como colunas os gradientes das componentes de $c(t)$).

Na Figura 17 estão malhas para um complexo curvo com 44 peças refinadas sob diversos critérios. A malha é refinada a cerca de 400 nós/segundo sobre a superfície, e a cerca de 5.000 nós/segundo sobre o espaço tridimensional. Uma partição do 3-complexo simplicial é mostrada na Figura 18.

Uma análise por histogramas dos ângulos diedros foi feita para uma malha refinada pela maior aresta (Figura 19), para uma malha refinada por precisão de representação de geometria e razão menor aresta e circunraio (Figura 20), e para uma malha refinada por precisão de representação de geometria e menor ângulo diedro (Figura 21). Note a presença de *slivers* na malha refinada pela maior aresta e na malha refinada pela razão menor aresta e circunraio.

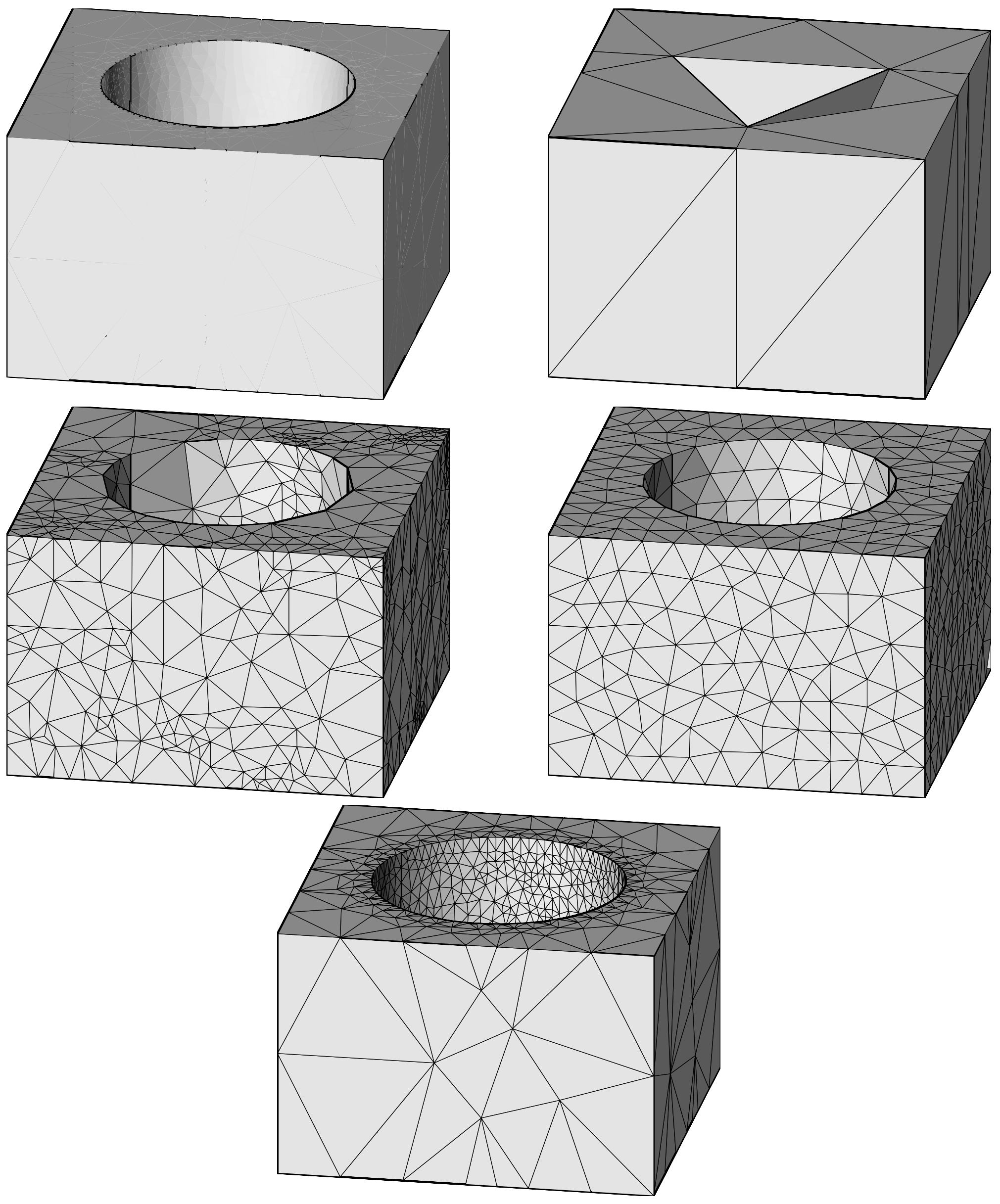

Figure 17: Malhas para o complexo curvo (acima-esquerda): sem refinamento com 26 nós (acima-direita), refinada aleatoriamente mantendo $\ell/R \geq 1/\sqrt{2}$ com 801 nós (meio-esquerda), refinada pela maior aresta com 729 nós (meio-direita), e refinada para representação precisa da geometria de entrada mantendo $\ell/R \geq 1/\sqrt{2}$ com 2006 nós (abaixo).

Conclusão

A geração de malhas para complexos curvos é rápida, especialmente se considerado que raramente um ponto é inserido nas partes curvas durante o refinamento (*e.g.* refinamento uniforme). A qualidade e a gradação do complexo simplicial refinado para um complexo curvo apresentam um comportamento bem semelhante às observadas para complexos lineares.

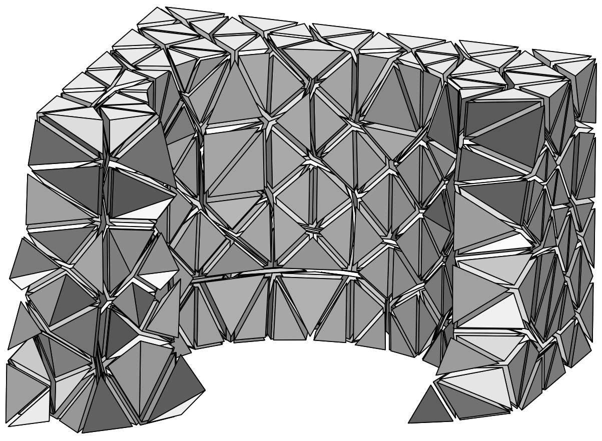
Figure 18: Partição de um 3-complexo simplicial homeomórfico a um complexo curvo.

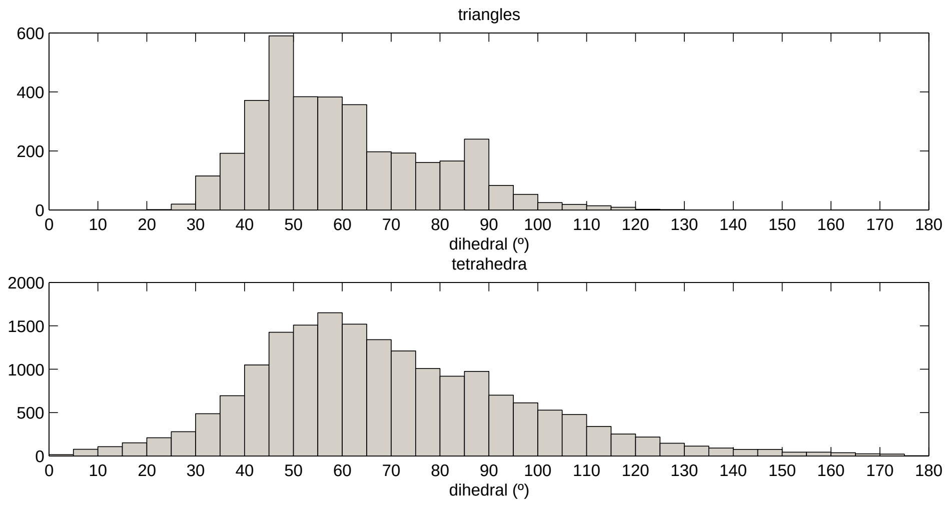

Figure 19: Histograma de ângulos diedros de triângulos e tetraedros de um 3-complexo simplicial refinado pela maior aresta.

O maior gargalo no desempenho do refinamento Delaunay em dimensões tipicamente maiores que 4 é a avaliação de predicados robustos. A manutenção das filas de refinamento e a própria avaliação dos critérios de refinamento consomem também um tempo razoável de processamento.

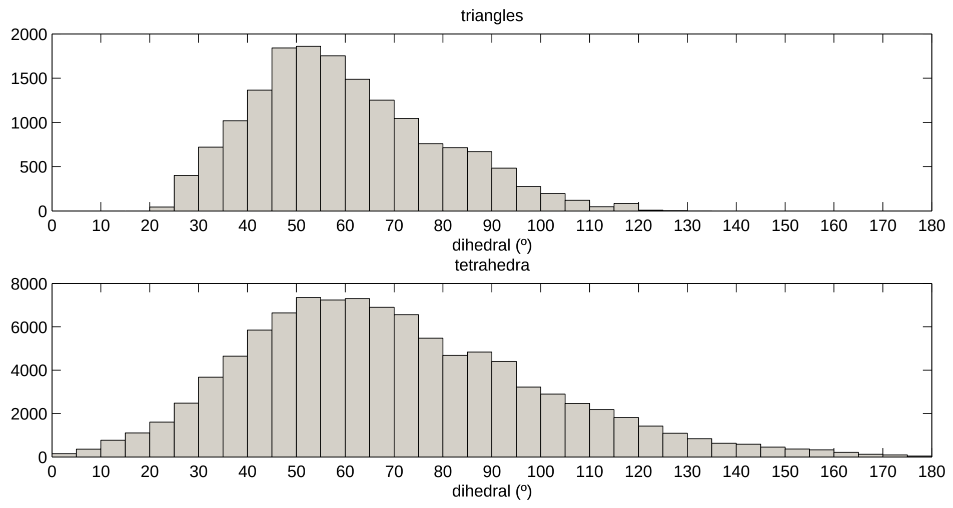

Figure 20: Histograma de ângulos diedros de triângulos e tetraedros de um 3-complexo simplicial refinado para representação precisa mantendo $\ell/R \geq 1/\sqrt{2}$.

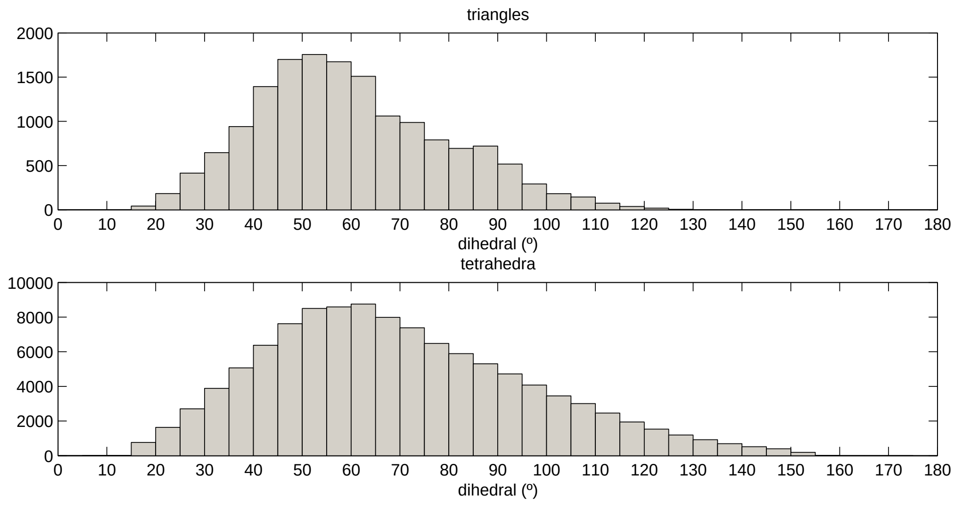

Figure 21: Histograma de ângulos diedros de triângulos e tetraedros de um 3-complexo simplicial refinado para representação precisa mantendo ângulos diedros acima de 15º.

Chapter 1

Introduction

Defining roughly in a single sentence, a mesh is a partition of a geometric object into small pieces of simple shapes. Concrete illustrative examples are a honey comb, a mosaic, a spider's web, or a woven - that suggests the name (see Figure 1.1).

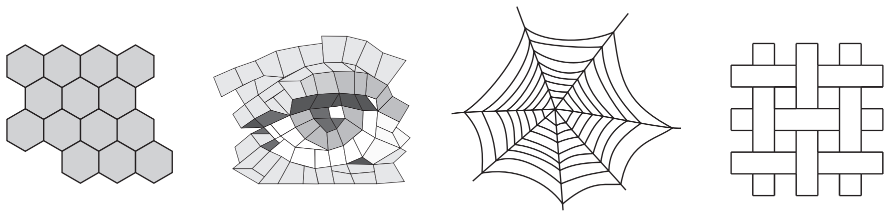

Figure 1.1: Illustrative examples of meshes: honey comb, mosaic, spider's web and woven.

A mesh embues simplicity to the geometric object it represents. The locality of its elements enables approximating assumptions inside their domain. For example, each piece in a mosaic may be considered to own a single color. Furthermore, the definition of shapes and connectivity as simple entities is the key for the development of a simple general mathematical framework, that is fundamental in computational applications. For example, a woven arrangement may represent any shape of cloth and provide uniform local features to it.

Numerous applications may be cited for meshes. A graphical engine uses meshes to represent complex geometric objects, so that shading models, projective transformations or ray tracing may be fast cast on them. Meshes may interpolate nonuniform samples of a function or a shape using their nodal connectivity. This work emphasizes on the investigation of a mesh generator algorithm for use in numerical methods to solve partial differential equations, like the finite element method. The requirements are very strict in this class of application, so that many of its mesh concepts are useful in other classes.

In focus in this thesis as a mesh generator, is the Delaunay refinement algorithm, that has theoretical guarantees on size, grading and quality supporting its good performance in practice. This algorithm generates simplicial meshes and most theoretical guarantees consider piecewise linear complex inputs. The main theme and results are in the extension of this algorithm for piecewise curved complexes.

1.1 Good meshes and mesh generators

Elements are the basis of a mesh. A good mesh have elements with properties and placement inside the domain that satisfy the problem requirements. A good mesh generator must be able to generate good meshes in a sufficient flexible process.

Lying in the context of numerical methods to solve partial differential equations, a mesh of the problem domain is used to create an interpolating function over it. Typically, physical properties are assigned to disjoint regions of the domain, boundary conditions are set on some lower dimensional parts of the domain, and then the ruling partial differential equations are forced to be satisfied within each element with boundary conditions defined by neighbor elements. The nodal values of the interpolating function are then given by a global system of linear equations.

An output mesh must conform the input features of the domain it represents or approximates (see Figure 1.2). Hence, each subset required in the problem domain must have a representative version in the mesh. If an input piece is linear, then an exact representation must be in the mesh. However, if an input piece is curved, then a homeomorphic approximate representation must be in the mesh, considering that the mesh elements are linear like simplices or boxes.

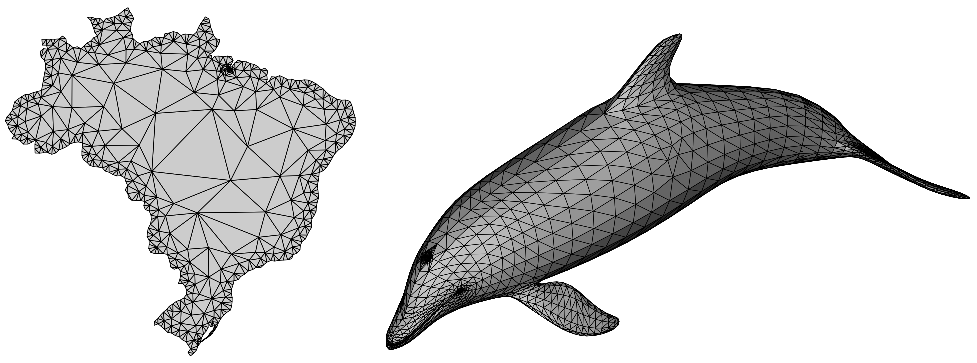
Figure 1.2: Triangular mesh of the Brazilian territory and surface triangular mesh of a dolphin.

Connectivity is also a key in numerical methods for partial differential equation solutions. It must hold general enough to represent any input domain connectivity, but it must be simple enough to allow a systematic and computationally efficient interactivity within the neighbors of each element. The most used connectivity constraints for meshes are the basis of the well known definition of simplicial complexes.

Output meshes can be required to be finer in some regions and coarser in others, so that the interpolating function accurately approximates the problem solution with fewer elements (see Figure 1.2, where an accurate boundary representation is achieved). Hence, refinement and grading must be supported by the mesh generator. A quasi-uniform mesh is in Figure 1.3c and a grading mesh is in Figure 1.3d, for the same input piecewise linear complex. This feature is known as grading optimality.

Refining a mesh is usually cheaper than coarsening. In a general context, only the problem solution can say correctly where the mesh must be finer. Thus a good mesh generator must be able to generate meshes not too finer than the coarsest possible mesh for an input domain. Where and when the mesh will be refined must be under user control. For instance, a mesh generator whose coarsest mesh is the one in Figure 1.3c is worse than the one that is able to generate the one in Figure 1.3b. This feature is known as size optimality.

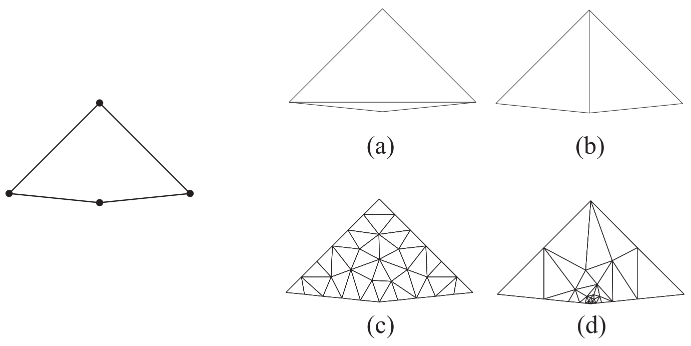

Figure 1.3: Some examples of meshes for the same input geometry (left).

Due to numerical and interpolation errors, elements are required to look as round as possible. Two too close vertices, in relation to other vertices in the element, increase machine roundoff errors. A vertex too close to a facet, in relation to the facet size, is very likely to cause high interpolation errors and gradients in a linear interpolation. For instance, the mesh shown in Figure 1.3a is worse than the one in Figure 1.3b. This feature is known as mesh quality. Grading optimality and mesh quality are conflicting objectives. Nonetheless very good tradeoffs are possible in practice. There are specific problems where skinny elements aligned to the flow direction lead to much improved results. These anisotropic meshes deserve and have received attention, but they will not be discussed in this text.

Every aforementioned desirable property for meshes is unlikely to be better attained by a structured mesh than by an unstructured one. Even though, because of its simplicity, faster generation and point location, some applications still justify the usage of structured meshes. Simplicity and speed are always desirable in algorithms, as long as all requirements are satisfied.

1.2 Why curved complexes?

Inputting only piecewise linear complexes requires any curved object in the domain to be firstly approximated by flat objects. This breaks the mesh generation into two well defined steps. However, because it is a requirement of the mesh generator, the conversion is sometimes considered to be a modeling task. Treating curved objects like real curved objects during the mesh generation makes clear what is modeling and what is meshing. The geometric model will only have to answer questions made by the mesh generator.

A worse drawback of pre-meshing an input is that the output mesh size becomes predefined by the input. If the input is overrefined, the output will have more elements than necessary. If the input is underrefined, any point insertion on it will consider the linear version of it, not the curved one. Furthermore, if the mesh generation is refined according to an adaptive strategy using the problem solution, any size guess will not be reasonable (otherwise the term adaptive is not applicable) and input remeshing will eventually be inevitable.

Another important point is that the input is a constraint. If skinny elements are in the input, they will have to be also in the output mesh. That could happen even if the curved objects did not feature any small angle between their facets. Unfortunately, only 2-dimensional meshes are not prone to this.

The best feature achieved by a mesh generator for curved complex inputs is coarser output meshes. Where and when the mesh will be refined is user defined. However, the input will never be exactly represented in the mesh, so that the curved model must always be queried, and even detecting the moment to query the curved model spends time.

1.3 State of the art

Since its beginning in 1987 with the circumcenter point insertion [Frey, 1987], many extensions have been proposed to conform the Delaunay mesh refinement to geometric objects. The most remarkable ones are the Chew's 1st algorithm [Chew, 1989b], Ruppert's algorithm [Ruppert, 1994] and Chew's 2nd algorithm [Chew, 1993]. All of them have been developed for the triangulation of planar straight line graphs, and the latter two were extended for the tetrahedralization of piecewise linear complexes in Shewchuk's thesis [Shewchuk, 1997]. Recent studies have been published on triangulations [Boivin and Ollivier-Gooch, 2002] and tetrahedralizations [Borovikov et al., 2005, Cheng et al., 2007b] for curved geometric objects. Nevertheless, mesh generation for this kind of input is still a wide open problem. A brief history up to the current stage is given next.

The nowadays known as Delaunay condition for simplicial complexes was proposed in 1934 by Boris Delaunay [Delaunay, 1934]. The first Delaunay triangulation algorithm was curiously published, unaware of their realization, by Frederick et al. [Frederick et al., 1970]. In 1977 Lawson [Lawson, 1977] proposed the flip Delaunay triangulation incremental algorithm and proved that it maximizes the minimum angle. In 1981 Bowyer and Watson [Bowyer, 1981, Watson, 1981] published an n-dimensional Delaunay incremental algorithm based on cavities. A generalized concept of Delaunay triangulation was introduced by Lee in 1978 [Lee, 1978], which was named constrained Delaunay triangulation by Paul Chew in 1989 [Chew, 1989a]. In 1998 Shewchuk [Shewchuk, 1998] proposed a definition for constrained Delaunay simplicial complexes and stated a condition for their existence in higher dimensions. Chew also introduced in 1993 a Delaunay criterion over curved surfaces [Chew, 1993], which was generalized in 1994 by Edelsbrunner and Shah [Edelsbrunner and Shah, 1994]. Chen and Bishop [Chen and Bishop, 1997] proposed a Delaunay criteria for curved surfaces based on maps from circles to ellipses in the parametric space of a surface.

The first provably good conforming Delaunay refinement algorithm was proposed by Chew in 1989 [Chew, 1989b], where all triangles are guaranteed to have angles in the range $[30^o, 120^o]$, where the lower bound is only violated near small input angles. In this algorithm, the input piecewise linear complex must be segmented to the edge length range $[h, h\sqrt{3}]$ and the output mesh is quasi-uniform, as shown in Figure 1.4.

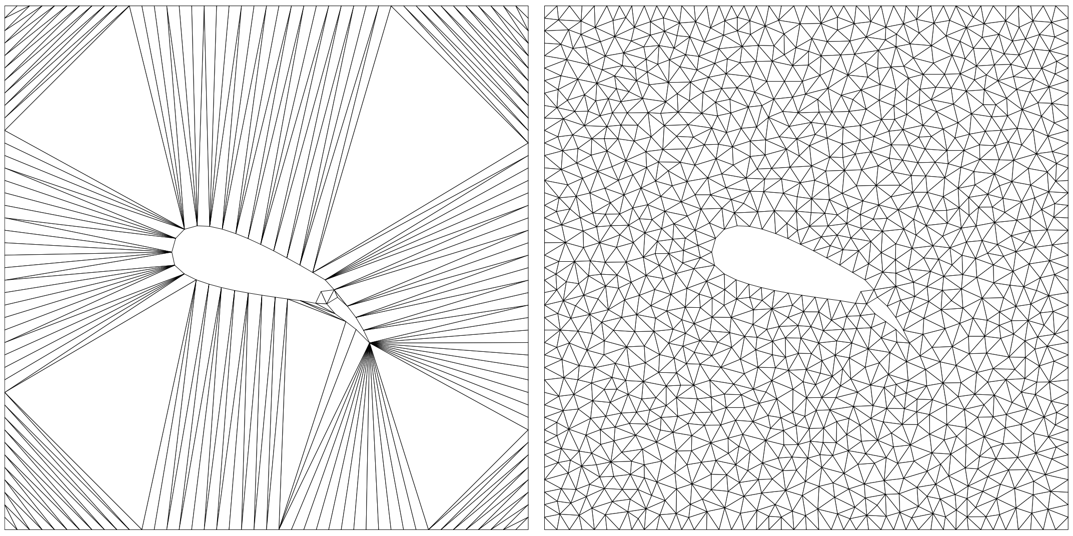

Figure 1.4: Initial (left) and final mesh with 2293 triangles (right) generated by Chew's first algorithm.

In 1994 Ruppert [Ruppert, 1994] proposed the first conforming Delaunay refinement algorithm with theoretical guarantees on quality, grading and size. In this new algorithm, the input piecewise linear complex do not need to be segmented, because the missing edges in the initial Delaunay triangulation are recovered in a former stage. After edge recovery, new vertices are added to the conforming edges as needed, as shown in Figure 1.5. Ruppert proved that all triangles in the final mesh can have angles in the range $[20.7^o, 138.6^o]$ ($\arcsin \sqrt{2}/4 \approx 20.7^o$), latter on proved to be $[26.45^o, 127.1^o]$ ($\arcsin 2^{-7/6} \approx 26.45^o$) by Miller et al. [Miller et al., 2003], and still hold grading and size optimality. However, all input angles must be larger than 90^o, which was latter proven to be 60^o by Shewchuk [Shewchuk, 1997], and latter on to be about 36.53^o ($\arctan[(\sin 36.53^o)/(2 - \cos 36.53^o) \approx 26.45^o$) by Miller et al. [Miller et al., 2003]. To handle small input angles $\phi \leq 60^o$, Shewchuk [Shewchuk, 2000] proposed a strategy that terminates with angles in the range $[\arcsin[\sin(\phi/2)/\sqrt{2}], \pi - 2\arcsin[\sin(\phi/2)/\sqrt{2}]]$, but without grading nor size optimality guarantees. Miller et al. [Miller et al., 2003] also introduced a strategy for treating small input angles $\phi < 36.53^o$, so that the output mesh angles range is $[\arctan[(\sin\phi)/(2 - \cos\phi)], \min\{137.1^o, \pi - 2\arctan[(\sin\phi)/(2 - \cos\phi)\}]$ ($\pi - 2\arcsin[(\sqrt{3}-1)/2] \approx 137.1^o$), keeping the grading optimality (see Figure 1.6) with analysis considering the input piece linear complex containing its own convex hull boundary (slightly more restrictive). Moreover, both strategies for small input angles generates new small angles only nearby them. Shewchuk [Shewchuk, 1997] also extends Ruppert's algorithm to tetrahedralizations which terminates for shortest edge length to circumradius ratios ℓ/R no smaller than $1/2$ with grading theoretical guarantees (see Figure 1.7), but without quality nor size guarantees. The algorithm requires input dihedral angles greater than 60° between input edges, greater than 69.3° ($\arccos \sqrt{2}/4 \approx 69.3°$) between input edges and planes, and greater than 90° between input planes. Shewchuk also conjectured that, in higher dimensions, Ruppert's algorithm terminates for $\ell/R \leq 2^{(1-d)/2}$ with good grading.

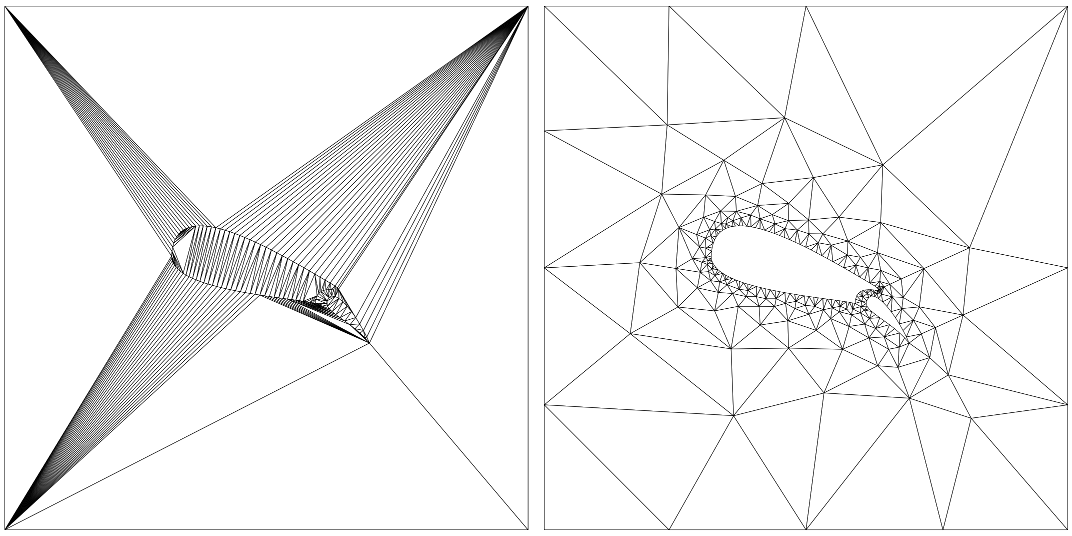
Figure 1.5: Initial (left) and final mesh with 426 triangles (right) generated by Ruppert's algorithm.

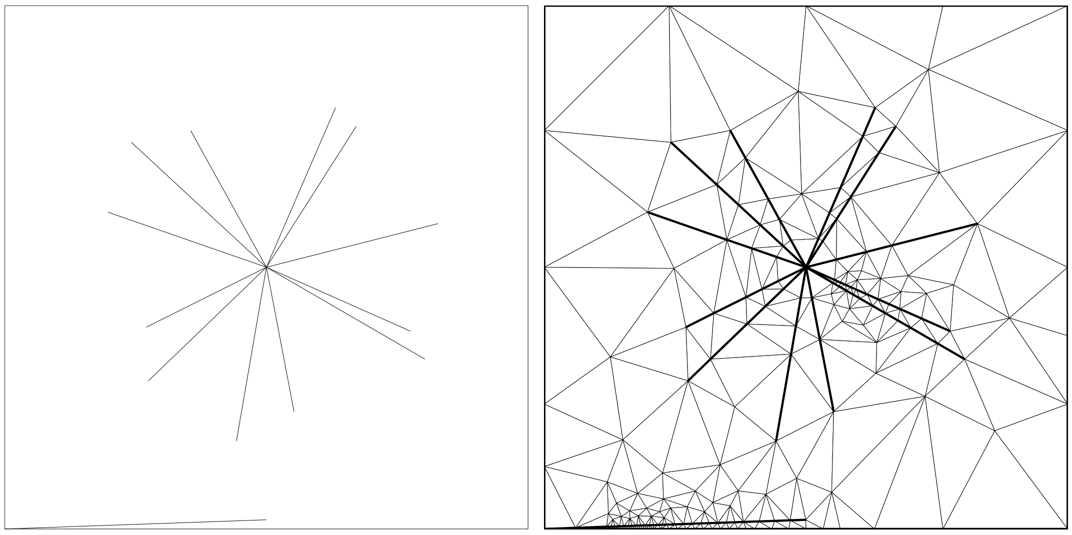
Figure 1.6: Input piecewise linear complex with small angles (left) and respective mesh (right) created with Ruppert's algorithm.

In parallel to Ruppert's work, Chew [Chew, 1993] proposes in 1993 a second algorithm that is very similar but slightly better and generates a constrained Delaunay triangulation. Shewchuk [Shewchuk, 1997] saw that this new algorithm

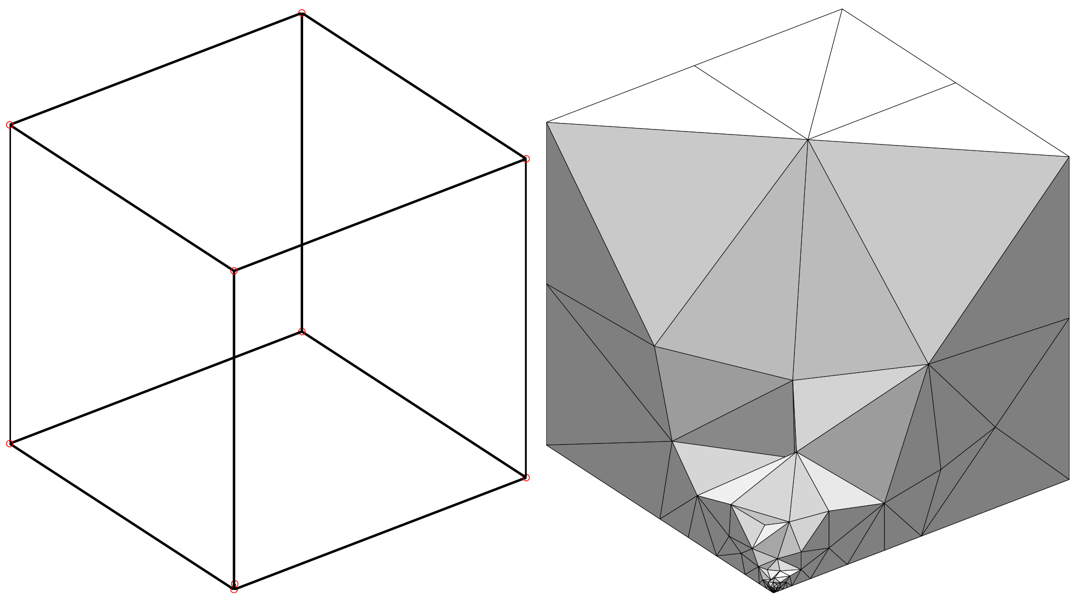

Figure 1.7: Input piecewise linear complex (left) and cut on the final tetrahedral mesh (right) generated by Ruppert's algorithm.

has the same basis of Ruppert's algorithm by replacing the diametral circles by lenses, and considering the constrained triangulation. This algorithm guarantees output angles in the range $[26.57^o, 126.9^o]$ ($\arcsin 1/\sqrt{5} \approx 26.57^o$), $[28.6^o, 122.8^o]$ with Miller et al. analysis [Miller et al., 2003], and requires the same input angle constraints as Ruppert's algorithm. The second Chew's algorithm for tetrahedralizations [Shewchuk, 1997] terminates for shortest edge length to circumradius ratios no smaller than $\sqrt{6}/4 \approx 0.6124$ with grading guarantees. Shewchuk also conjectured that, in higher dimensions, Chew's second algorithm terminates for $\ell/R \leq \sqrt{3/2^d}$ with good grading.

In 2002 Boivin and Ollivier-Gooch [Boivin and Ollivier-Gooch, 2002] extended Ruppert's Delaunay triangulation algorithm for curved boundaries, which was latter fixed and simplified by Gosselin and Ollivier-Gooch [Gosselin and Ollivier-Gooch, 2007]. The fundamental idea of this strategy is to create an initial Delaunay triangulation just fine enough to allow a fast refinement, as shown in Figure 1.8. According to Gosselin and Ollivier-Gooch, the subsegments in the initial triangulation cannot contain any visible vertex in their diametral lenses, and the angular variation of their subcurves should be less than $\pi/6$. Quality, grading and size optimality follows straightforwardly from Chew's second algorithm because a piecewise linear approximation of the input is in the triangulation. Borovikov et al. [Borovikov et al., 2005] proposed an intricate conforming constrained Delaunay tetrahedralization algorithm based on a Delaunay property in the parametric space of curved manifolds (input surfaces). Most ideas of this algorithm are empirical and its theoretical guarantees were not deeply investigated. Cheng et al. [Cheng et al., 2007b] proposed a different approach with theoretical guarantees based on conditions established by Edelsbrunner and Shah [Edelsbrunner and Shah, 1994] for homeomorphism between

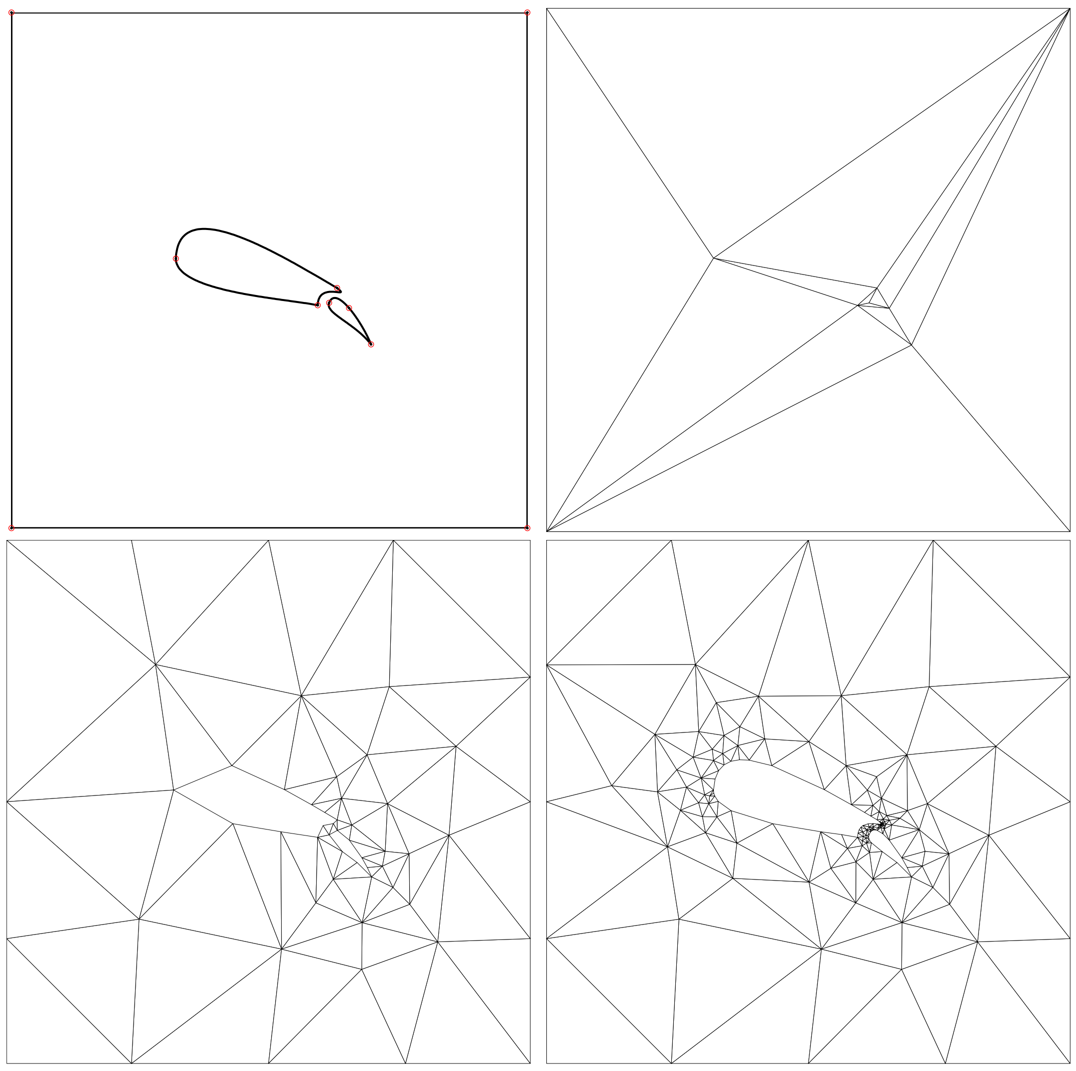
Figure 1.8: Input curved complex (top-left), initial mesh with 14 triangles (top-right), mesh refined for quality with 80 triangles (bottom-left) and mesh refined for boundary accuracy with 284 triangles (bottom-right).

a manifold and the restricted Delaunay simplicial complex for samples over it. The algorithm was latter simplified [Cheng et al., 2007a] to one that guarantees an output homeomorphic restricted simplicial complex as samples on input get denser.

A guaranteed quality Voronoi refinement algorithm for anisotropic triangular mesh generation was proposed by Labelle and Shewchuk [Labelle and Shewchuk, 2003], and extended for domains with curves by Yokosuka and Imai [Yokosuka and Imai, 2006].

The first provably good mesh generation algorithm was introduced in 1988 by Baker et al. [Baker et al., 1988]. Bern et al. [Bern et al., 1990] proposed the first algorithm for conforming triangulations based on quadtrees with quality, grading and size optimality guarantees (see Figure 1.9 for an example without boundary conformity, and with an unbalanced quadtree). All output angles are guaranteed to lie in the range $[18.43^o, 90^o]$ ($\arctan 1/3 \approx 18.43^o$) and the output mesh is notoriously axis aligned. This algorithm was generalized in 2000 to higher dimensions by Mitchell and Vavasis [Mitchell and Vavasis, 2000].

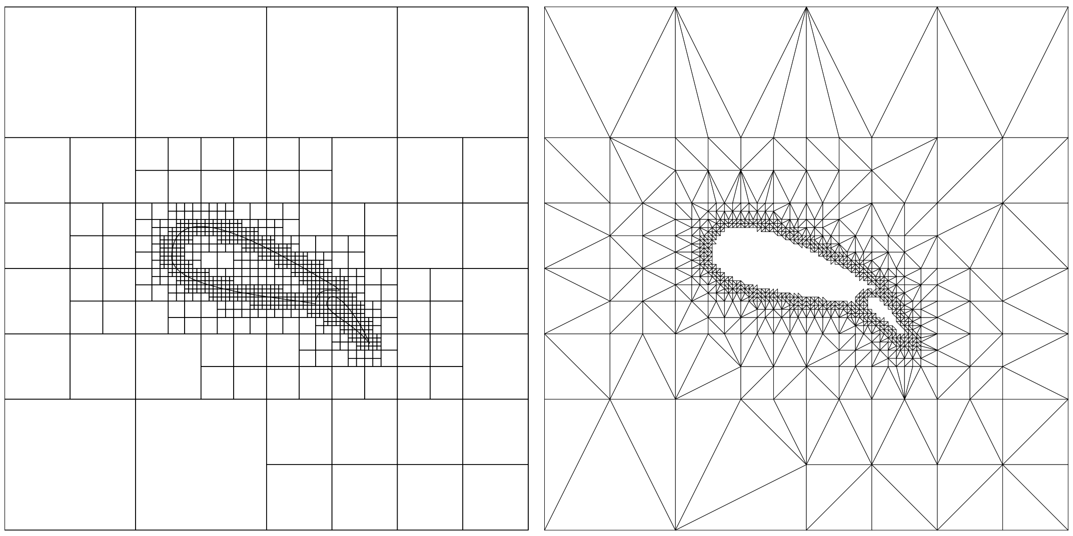

Figure 1.9: Triangular mesh (right) generated from the quadtree decomposition (left).

Some approaches to mesh generation are physically based on particle simulation, usually with analogy to bubble packing [Yamakawa and Shimada, 2000] or spring network [Persson, 2005]. They start with a crude mesh and improve the quality forcing equilibrium or finding the minimum entropy point. The high quality output meshes, as the example shown in Figure 1.10 from the mesh generator for implicit geometries developed by Persson [Persson, 2005], are extremely well graded and sized. Such a great result comes at a price and is limited. The points must be well distributed from the beginning, otherwise a local minimum may trap the algorithm. The update is expensive because all vertices must be moved in every iteration, and the number of iterations is theoretically unbounded. It is not natural to conform boundaries separating sub-domains or any lower dimensional object other than the domain contour. They usually use the Delaunay criterion on their connectivity scheme. All these features essentially define this approach as a mesh optimization process for interior points, where there are no guarantees to reach good results for arbitrary input geometries, but start meshes will not get worse.

1.4 Main results and overview of the thesis

The geometrical input for a Delaunay refinement algorithm is usually given by a piecewise linear complex (PLC) or, recently, by piecewise smooth complex (PSC). The former one deals with non-curved complexes, and the latter one deals with a subset of general curved complexes. A general domain definition is introduced in this work as a natural extension of the piecewise linear complex: the manifold complex. For any given set, it is proven that there is a unique manifold complex

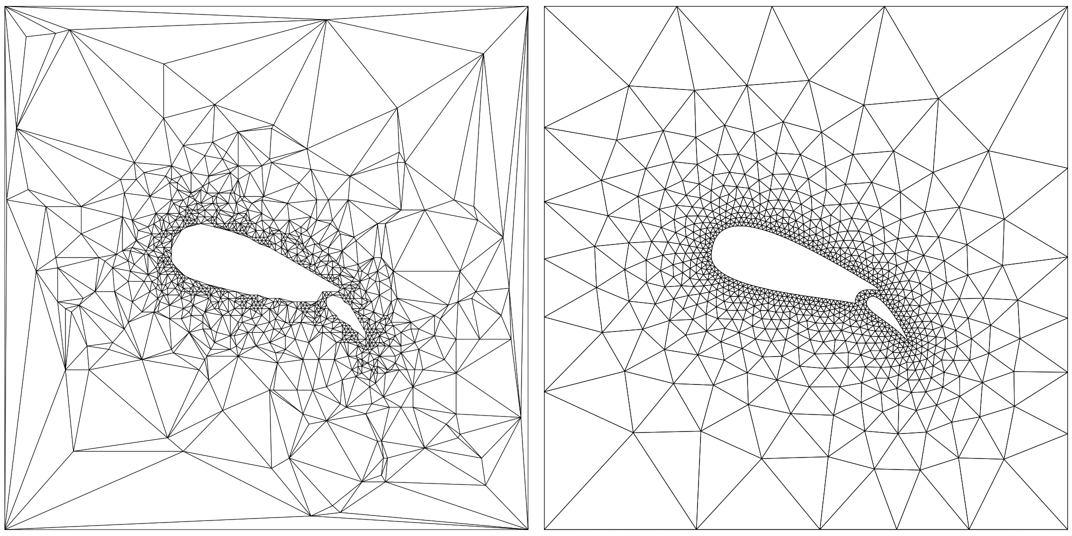

Figure 1.10: Initial (left) and final (right) mesh generated by physical simulation.

representation. Some differential and topological properties for a manifold complex are sketched, as well as its computational representation.

Shewchuk has proven in his thesis that a triangulation is strongly Delaunay if there exists a circumcircle, for every edge, that contains no other vertex of the triangulation other than the edge endpoints. In this work this theorem for strongly Delaunay simplicial complexes is extended to higher dimensions by stating that a simplicial complex is strongly Delaunay iff codimensional simplices are strongly Delaunay, for any single dimension.

With the theorem just stated at hands, the Chew's and Ruppert's algorithms are revisited and presented in the point of view of protecting balls. Under this perspective a Delaunay refinement algorithm for curved complexes arises more naturally.

A generalization of the fundamental idea of the Bowyer-Watson algorithm is the basis of the generalization of the Ruppert's and Chew's second algorithm for dealing with manifold complexes. This extension is presented as a theorem and guarantees that simplices representing input pieces will be part of the simplicial complex throughout the meshing process. It is proposed a feasible algorithm with expected theoretical guarantees on termination and quality.

During the implementation small contributions take place. A data structure for manifold complexes is given under the mesher requirements, and a triangle like data structure is derived to store the simplicial complex. The fan search in the simplicial complex update in a incremental algorithm is also introduced. A Newton-Raphson iterative algorithm is derived to find a Voronoi point over a curved surface. Robust predicates are generalized to arbitrary dimensions. Many results are given on Delaunay refinement algorithms. The Delaunay high performance is verified, achieving speeds of about 50,000 nodes/second in 2-dimensional meshes, and 10,000 nodes/second in 3-dimensional meshes in a single processor computer.

Chapter 2

Domain representation

In the finite element method, the partial differential equations are discretely solved over a bounded domain $\mathbb{S} \subset \mathbb{R}^n$, as shown in Figure 2.1, which may be disconnected and contain lower dimensional parts. Disconnected parts may represent independent problems, or may be virtually connected through boundary conditions. Lower dimensional parts represent thin objects, where a suitable formulation is applied.

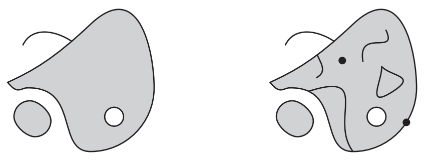

Figure 2.1: A bounded domain in $\mathbb{R}^2$ where the finite element method is applied (left) and some subsets inside it (right).

The set $\mathbb{S}$ must be represented by a collection of subsets (see Figure 2.1). These subsets are assigned to physical properties, or boundary conditions... or whatever the problem requires. This chapter defines an unambiguous partition of $\mathbb{S}$, in order to guarantee a nice representation of its features.

2.1 Manifold complex

Before defining the fundamental idea of the domain partition - the manifold complex -, some basic definitions and concepts are first introduced to let the text self-contained and to ease comprehension.

Definition 1 (disjoint) *Two sets are disjoint iff they have no element in common.*

Definition 2 (cover) *The cover of set a $\mathbb{S}$ is a collection of nonempty non-duplicated subsets whose union is $\mathbb{S}$.*

Definition 3 (partition) *A partition of a set $\mathbb{S}$ is any cover of $\mathbb{S}$ in which subsets are pairwise disjoint.*

Definition 4 (neighborhood) *A neighborhood of a point p is any set containing a ball centered at p and with radius $\epsilon > 0$.*

The neighborhood of a point introduces into a set the very simple, but also very important, concept of open and closed.

Definition 5 (open set) *A set is open iff any point in it has a neighborhood lying in the set.*

Definition 6 (closed set) *A set is closed iff any point outside it has a neighborhood disjoint from the set.*

Definition 7 (closure) *The closure of a set is the smallest closed set containing it.*

Definition 8 (boundary) *The boundary of a set $\mathbb{S}$ is the intersection between the closure of $\mathbb{S}$ and the closure of the complement of $\mathbb{S}$.*

The idea of open sets brings to small scales the concept of infinite, like in the definition of limits. The infinitely far is where anything beyond makes no difference but there is always something beyond. The infinitely close is where anything in between makes no difference but there is always something in between. When the boundaries of a set are taken away, what remains includes something infinitely close to it, which characterizes an open set.

Definition 9 (topological space) *A topological space is a set $\mathbb{X}$ together with a collection of open subsets $\mathcal{T}$ that satisfies the conditions*

1. $\emptyset \in \mathcal{T}$
2. $\mathbb{X} \in \mathcal{T}$
3. *The intersection of a finite number of sets in $\mathcal{T}$ is also in $\mathcal{T}$.*
4. *The union of an arbitrary number of sets in $\mathcal{T}$ is also in $\mathcal{T}$.*

The union of an arbitrary number of closed sets can get infinitely close to a point, which defines an open set. However, the union of an arbitrary number of open sets will never lead to a closed set. A similar analysis is valid for intersection, where the intersection between two sets is the complement of the union of their complementary sets. Hence, the union of the complement of an arbitrary number of open sets (i.e. union of closed sets complementary to the intersection of open sets) can get infinitely close to a point, whose complement defines a closed set. However, the intersection of an arbitrary number of closed sets will never lead to an open set.

When refereing to a neighborhood of a point in a set or a topological space, the neighborhood is subtended to be its intersection with them, which is the same as neighborhood relative to them.

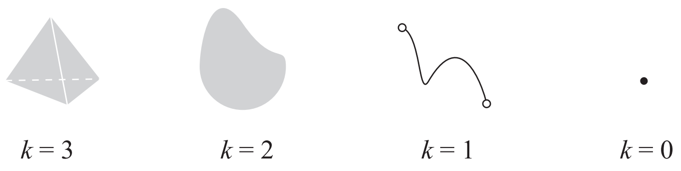

Figure 2.2: Examples of k-manifolds projected on the plane.

Definition 10 (k-manifold) *A k-manifold is a topological space where there is a neighborhood around every point that is topologically the same as an open k-ball.*

The definition of boundary of a manifold follows straightforwardly from the definition of boundary of a set. The boundary of a k-manifold is not part of it, by definition. This induces a natural partition of a set into n-manifold pieces, $n = k, ..., 1, 0$. It is interesting to note that there are manifolds without boundaries, like a circle or a torus, called closed manifolds.

The partition of a set using k-manifolds in $\mathbb{R}^n$ is not unique when the neighborhood of a point is topologically the same as the union of more than two distinct half open k-balls (see Figure 2.3). This only happens when $k < n$ and it is closely related to the connectivity between subsets.

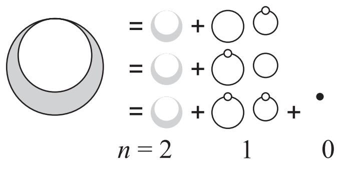

Figure 2.3: Example of a set in $\mathbb{R}^2$ with a junction point and its different partitions into n-manifolds, $n = 0, 1, 2$.

Definition 11 (connected) *Two sets are connected iff there is at least one element in a set that lies in the closure of the other set.*

A set is connected iff it cannot be partitioned into two disconnected nonempty subsets. Notice that the definition also applies to the connectivity between disconnected sets. Joint and connected have different meanings, as well as disjoint and disconnected. With this last concept, it is possible to define a manifold complex.

Definition 12 (manifold k-complex) *A manifold k-complex is a collection of nonempty connected pairwise disjoint n-manifolds, $n = 0, ..., k$, where pieces of the same dimension are pairwise disconnected.*

A collection of disjoint sets - the domain with its features - can be element-wise partitioned into a manifold complex, as shown in Figures 2.4 and 2.5. The name complex is used for a collection of subsets. Each element in this collection is a piece, so that a manifold is a special instance of a piece. The specification of the n-dimensionality of a piece is denoted by n-piece, as used in n-manifold.

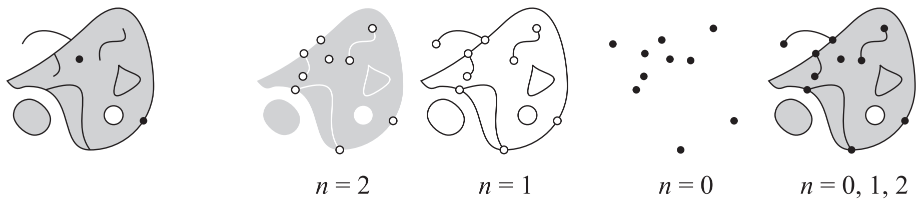

Figure 2.4: Example of a 2-dimensional domain (left) partitioned into a manifold 2-complex (right).

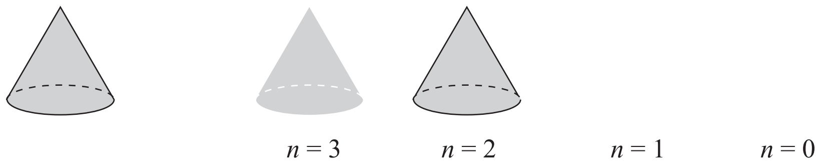

Figure 2.5: Example of a 3-dimensional domain (left) partitioned into a manifold 3-complex (right).

Definition 13 (junction degree) *The junction degree of an n-manifold $\mathbb{X}$ in a manifold k-complex $\mathcal{M}$ is the maximum number of disjoint $(n+1)$-submanifolds in $\mathcal{M}$ that contain $\mathbb{X}$ in their closure.*

Notice that a single manifold can have up to 2 disjoint submanifolds containing the same lower dimensional manifold in their closure. Each $(n+1)$-submanifold counted up is called a wing of the junction n-manifold. Two manifolds are incident iff their closure are joint.

Theorem 1 *The partition of a set in $\mathbb{R}^k$ into a manifold k-complex is unique.*

Proof. For k-manifolds, there are no partition ambiguities. For n-manifolds where $n < k$, all locations where the neighborhood is not topologically an n-ball are glued together into lower dimensional subsets, which will be also subject to partition. This partition scheme goes on until 0-dimensional pieces, where there is no connectivity and, thus, no partition ambiguities. □

In a manifold complex, each lower dimensional k-manifold connected to a higher dimensional n-manifold, $n > k$, is a facet of $\mathbb{S}$. If $\mathbb{S}$ has boundaries, then it is delimited by lower dimensional facets.

The concept of open sets played an important role in the definition of manifold complexes. However, it makes no difference in computational representations of open or closed sets, since the missing piece must also be described in open sets. Manifold complexes may represent open, closed or half-open domains, simply by marking the pieces that are not in the domain.

2.1.1 Differential properties

In order to define differential properties of manifold complexes, derivatives on manifolds must be defined first.

Definition 14 (continuous function) *A function f is continuous at a point x_0 iff for any $\epsilon > 0$ there exists a $\delta > 0$ such that $|f(x_0) - f(x)| < \epsilon$ for any x in the neighborhood of x_0 within a radius δ.*

Definition 15 (C^k function) *A function f is k-continuous, denoted by C^k, iff its k derivative is continuous.*

Definition 16 (homeomorphism) *A function $\phi : \mathbb{X} \mapsto \mathbb{Y}$, between two topological spaces $\mathbb{X}$ and $\mathbb{Y}$, is a homeomorphism iff it is a bijection and continuous in both directions.*

Definition 17 (C^k n-manifold) *An n-manifold is k-continuous, denoted by C^k, iff there is a C^k homeomorphism $\phi : \mathbb{U} \mapsto \mathbb{V}$, between the neighborhood $\mathbb{U}$ of any point in it and an open subset of the n-dimensional Euclidean space $\mathbb{V} \subset \mathbb{R}^n$.*

A manifold complex is C^k if all its manifold pieces are C^k. The definition of a manifold complex guarantees that it is C^0. To make it C^k for $k \geq 1$, all points where the l-th derivative is not continuous, $l \leq k$, must become pieces of the complex (see Figures 2.6 and 2.7).

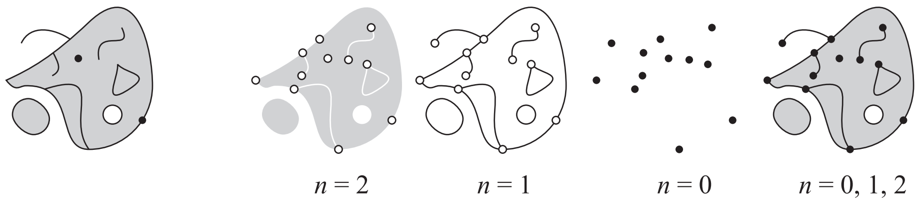

Figure 2.6: Example of a 2-dimensional domain (left) partitioned into a C^1 manifold 2-complex (right).

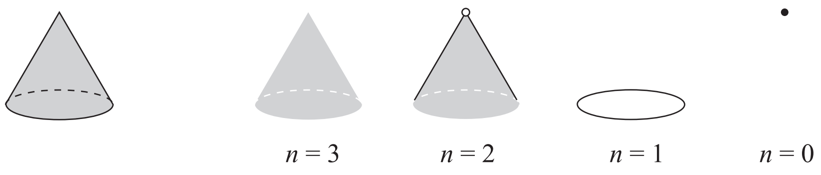

Figure 2.7: Example of a 3-dimensional domain (left) partitioned into a C^1 manifold 3-complex (right).

Differential discontinuities occur only in m-pieces in $\mathbb{R}^n$ where $m < n$. Thus, their locations are p-pieces where $p < m \Rightarrow p \leq n - 2$.

The C^k continuity of a manifold complex may serve to store locations of discontinuities. Otherwise, important features in C^1, or even C^2, discontinuities are very likely to be smoothed during the mesh generation.

2.1.2 Topological properties

There are several possible topological configurations for a single piece, like the surface examples in Figure 2.8. In this figure, parallel edges join one another with the orientation indicated with arrows, so that each letter corresponds to a distinct point.

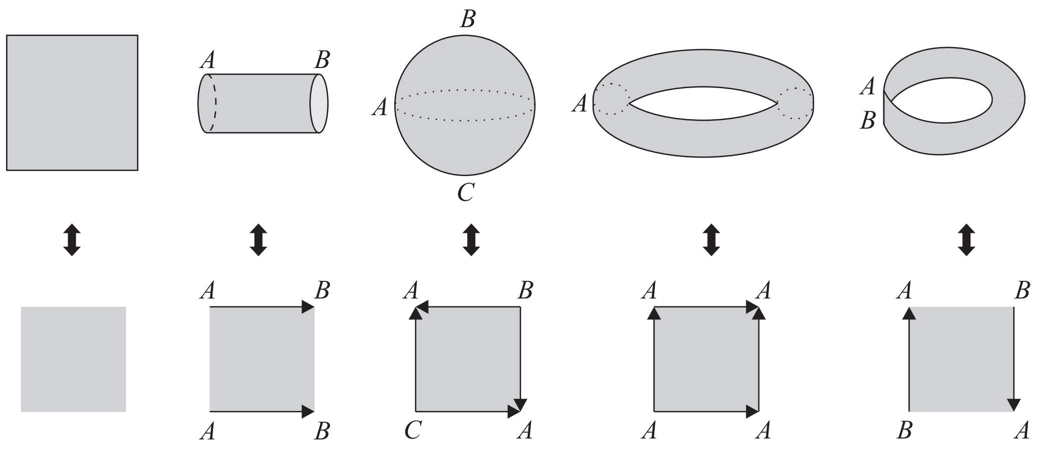

Figure 2.8: Example of different kinds of topological configurations of 2-manifold pieces. From left to right: plane, pipe, sphere, torus and Möbius strip.

Homeomorphism, homotopy, contractibility, genus and orientability are important invariants observed in topological configurations.

Definition 18 (homeomorphic) *Two sets are homeomorphic iff there exists a homeomorphism between them.*

Definition 19 (homotopic) *Two maps $f, g : \mathbb{X} \mapsto \mathbb{Y}$ are homotopic iff there is a continuous map $F : \mathbb{X} \times [0, 1] \mapsto \mathbb{Y}$ such that $F(x, 0) = f(x)$ and $F(x, 1) = g(x)$.*

Definition 20 (homotopy equivalent) *Two spaces $\mathbb{X}$ and $\mathbb{Y}$ are homotopy equivalent iff there are continuous maps $f : \mathbb{X} \mapsto \mathbb{Y}$ and $g : \mathbb{Y} \mapsto \mathbb{X}$ such that the composition $f \circ g$ is homotopic to the identity map of $\mathbb{Y}$ and $g \circ f$ is homotopic to the identity map of $\mathbb{X}$.*

Less formally speaking, two sets are homotopy equivalent if one can be continuously deformed into the other. Homeomorphic is stricter than homotopy equivalent. Every set $\mathbb{X}$ homeomorphic to $\mathbb{Y}$ is also homotopy equivalent to $\mathbb{Y}$, but a set $\mathbb{X}$ homotopy equivalent to $\mathbb{Y}$ is not necessarily homeomorphic to $\mathbb{Y}$. For example, a circle is homotopy equivalent to a torus, but not homeomorphic to. A disk is homotopy equivalent to its center, but a circle is not.

Definition 21 (contractible) *A set in $\mathbb{R}^n$ is contractible iff it is homotopy equivalent to one of its points.*

Definition 22 (genus) *The genus of an n-manifold is the largest number of nonintersecting simple closed $(n-1)$-manifolds that can be drawn on it without separating it.*

An n-manifold is simple closed iff it is homeomorphic to an n-hypersphere. The genus of a manifold is a special kind of hole. For instance, a sphere has genus 0 and a torus has genus 1. Any submanifold of $\mathbb{R}^2$ has genus 0, since an attempt to draw a closed curve on it will separate the enclosing points (even when a hole is enclosed). Analogously, any n-manifold in $\mathbb{R}^n$ has genus 0.

Definition 23 (orientable) *An n-manifold is orientable iff it has a partition of open n-balls where each one has the same orientation of its neighbors.*

The sense of orientation is the same right or left handedness used in differential geometry or vector algebra. A famous example of a non-orientable manifold is the Möbius strip (right most example in Figure 2.8). The orientability of a piece is also inherited by the mesh elements that represent it.

The topology of a piece is not explicitly represented in the manifold complex. Thus, it must be identified according to the representation of a piece in the complex. Topological properties are essential to create an initial mesh homeomorphic to an input manifold complex.

2.1.3 Flat complex

The Delaunay refinement algorithms were originally developed for PLC (short for piecewise linear complex) inputs [Miller et al., 1996]. It is also used the name PSLG (short for planar straight line graph), which is a PLC in the 2-dimensional space. The flat complex is defined next as a natural special case of a manifold complex, but representing the same pieces as PSLG and PLC.

Definition 24 (linear combination) *A linear combination of a set of vectors, columns of a matrix $V = [\nu_1, ..., \nu_k] \in \mathbb{R}^{n\times k}$, is a weighted sum of its elements $V\lambda$, $\lambda \in \mathbb{R}^k$.*

Definition 25 (affine combination) *An affine combination of a set of vectors, columns of a matrix $V = [\nu_1, ..., \nu_k] \in \mathbb{R}^{n\times k}$, is a weighted sum of its elements $V\lambda$, $\lambda \in \mathbb{R}^k$, where $\sum_{i=1}^{k} \lambda_i = 1$.*

Definition 26 (linearly independent) *A set of vectors is linearly independent iff no element in it may be expressed as a linear combination of the others.*

Definition 27 (affinely independent) *A set of vectors is affinely independent iff no element in it may be expressed as an affine combination of the others.*

Definition 28 (k-flat) *A k-flat is the set of all affine combinations of a set of $k+1$ affinely independent vectors.*

Definition 29 (flat k-complex) *A flat k-complex is a manifold k-complex where each n-manifold piece is contained in an n-flat, for all $n \in \{0, ..., k\}$.*

A flat k-complex is an open piecewise linear complex where member k-pieces are also defined. They represent exactly the same type of domains, but the new named definition was stated to avoid confusion with the "open" qualifier or the presence of k-pieces, and also to emphasize it as a natural special case of a manifold complex.

2.2 Representation of pieces

The representation of pieces may be divided into implicit and parametric, which are complementar in several aspects.

Definition 30 (implicit representation) *Let $\mathbb{P}$ be a k-piece in $\mathbb{R}^n$. An implicit representation for $\mathbb{P}$ is defined by $\mathbb{P} = \{p \in \mathbb{R}^n \mid f(p) = 0, \quad f : \mathbb{R}^n \mapsto \mathbb{R}\}$.*

Definition 31 (parametric representation) *Let $\mathbb{P}$ be a k-piece in $\mathbb{R}^n$. A parametric representation for $\mathbb{P}$ is defined by $\mathbb{P} = \{p \in \mathbb{R}^n \mid p = f(u), \quad f : [0,1]^k \mapsto \mathbb{R}^n\}$.*

Let $\mathbb{P}$ be the set of points that compose a piece of a complex. Implicit representations define distances from $\mathbb{P}$ (in the sense that they are null only on the piece, so that they are not necessarily distance functions), and parametric representations define member points of $\mathbb{P}$. The conversion between these two representations is not always simple, which create boundaries between them.

An implicit representation is suitable to answer questions on membership points of a piece: *e.g.* $f = 0$ means "on the piece", $f < 0$ means "in one side of the piece" and $f > 0$ means "in the other side of the piece". It is also easy to project a point onto the piece if the function is an Euclidean distance function: the gradient of the function tells the direction of the closest point on the piece, and the function value tells the step length to get there. Furthermore, Boolean operations can be trivially applied by taking minimum or maximum between two distance functions operands. In mesh generation context, the distance function value itself is very worthy to size control.

A parametric representation is very suitable to fast generate points on the piece it represents. Furthermore, it is very intuitive to model objects with it, specially free shape ones. The parametric space, as shown in Figure 2.9, is very handful to deal with mapping to a known Cartesian coordinate system. In mesh generation context, it is useful to localize features of the pieces, like boundaries or topological features .

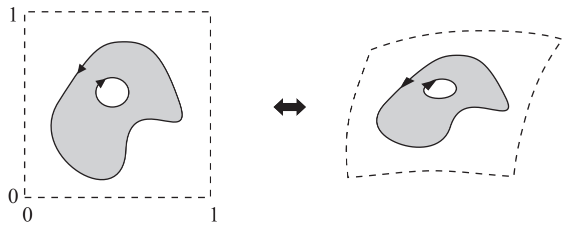

Figure 2.9: Parametric space of a 2-piece in $\mathbb{R}^3$.

2.3 Measures on domains

Again, in order to let this text self-contained and also to ease comprehension, some basic concepts will be introduced before defining important measures on domains used in a mesh generation context. These metrics are useful to quantify the quality, grading and size optimality of meshes.

2.3.1 Distance

The most basic measure on a domain is the distance to a piece of it. The distance between two sets is the shortest distance between one member point of each set. The usual distance function between two points, denoted by $\text{dist}(p, q)$, is the Euclidean norm, denoted by $\|p - q\|_2$, which is adopted as default. Since a norm is also used to describe length or size, the notation $\text{dist}(p, q) = \|p - q\| = |p - q| = |pq|$ is abstracted. A distance function is formally defined as a path integration in a metric space. When the path is omitted, it subtends to be the one with the smallest distance.

Definition 32 (metric) *A real valued function $f(p, q)$ between two points $p, q \in \mathbb{S}$ is a metric in $\mathbb{S}$ iff*

1. $f(p, q) \geq 0$ *(nonnegative)*
2. $f(p, q) \leq f(p, s) + f(s, q)$, $\forall s \in \mathbb{S}$ *(triangle inequality)*
3. $f(p, q) = f(q, p)$ *(symmetric)*
4. $f(p, q) = 0$ *iff* $p = q$

If the last condition in definition 32 is $f(p, q) = 0$ if $p = q$ (notice that now the function may vanish when $p \neq q$), then the function is called a pseudometric. A set that has a metric is called a metric space.

Figure 2.10 shows the level curves of the distance function for a set. Distance functions are basic, many other metrics are derived from them. Persson [Persson, 2005] uses distance functions to represent pieces in its mesh generator for implicit geometries, so that any point can be easily projected on the geometry contour.

Definition 33 (geodesic distance) *The geodesic distance $g(p, q)$ between two points $p, q \in \mathbb{S}$ in a set $\mathbb{S}$ is the length of the shortest path in $\mathbb{S}$ linking p and q.*

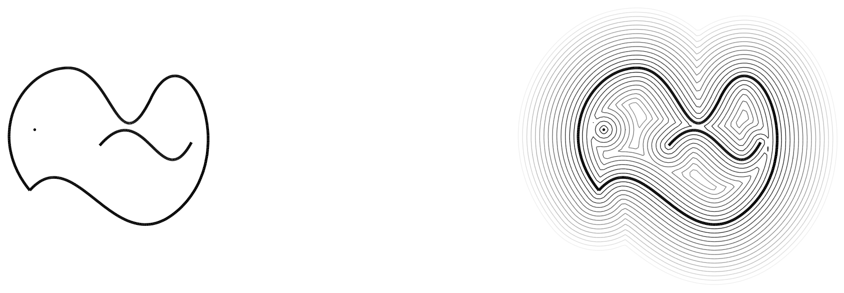

Figure 2.10: Contour curves of the distance function (right) to a set (left).

2.3.2 Medial axis

Amenta and Bern [Amenta and Bern, 1998] have used the medial axis to reconstruct shapes with theoretical guarantees.

Definition 34 (medial axis) *The medial axis of a set $\mathbb{S}$ is the the set of all points that are equidistant to at least two points in $\mathbb{S}$.*

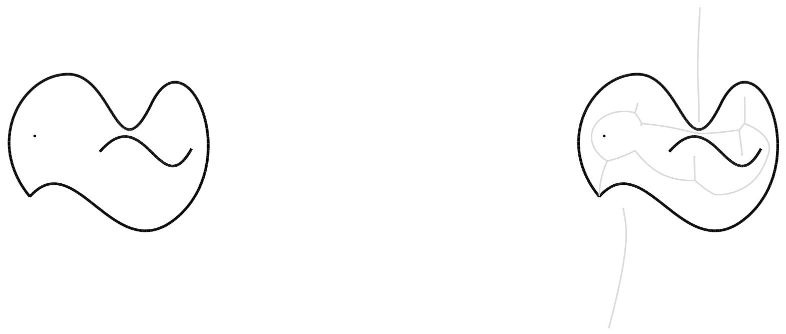

Figure 2.11: Medial axis (right) of a set (left).

The medial axis (see Figure 2.11) is closely related to the Voronoi diagram of a point set. If a point p is closest to two points on a curve, it lies on the edge of their respective Voronoi cells. If p is closest to more than two points, it lies on a vertex of their respective Voronoi cells. Contiguous parts of the curve leads to contiguous parts of the medial axis, by infinitesimal analysis. Hence, the medial axis is composed by the small edges of the Voronoi diagram if sample points on the curve are enough close. This algorithm was used to create the medial axis shown in Figure 2.11. A similar analysis is valid in higher dimensions.

When the curve is not smooth, the medial axis has spurious edges meeting the non-smooth points, since the bisectrix of two incident segments is equidistant to both. Similarly, the medial axis will get closer to the curve where the curvature is greater. This analysis also generalizes to higher dimensions.

2.3.3 Local feature size

A local feature size is any function that quantifies the size of a geometry in any point. It is commonly used in mesh generation and shape reconstruction to state theoretical guarantees.

Ruppert [Ruppert, 1994] proposed a local feature size function (see Figure 2.12) for a flat complex and proved that it yields planar meshes within a constant factor of the optimal size. Shewchuk [Shewchuk, 1997] used the same measure to prove grading bounds for tetrahedralizations.

Definition 35 (local feature size) *The local feature size of a point p relative to a flat k-complex $\mathcal{F}$ in $\mathbb{R}^n$, denoted by $\mathrm{lfs}_{\mathcal{F}}(p)$, is the radius of the smallest n-ball centered at p that intersects two non-incident pieces of $\mathcal{F}$.*

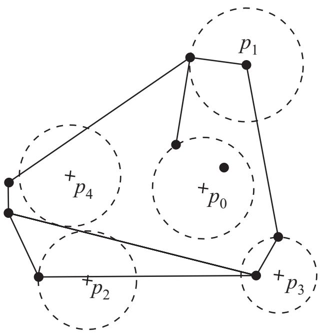

Figure 2.12: Local feature size at the disks centers p_i, $i = 0, ..., 4$, graphically represented by the respective radius.

Theorem 2 (1-Lipschitz [Ruppert, 1994]) *The local feature size is 1-Lipschitz.*

Proof. The n-ball $\mathbb{B}$ of radius $r = \mathrm{lfs}_{\mathcal{F}}(p)$ centered at p intersects two non-incident pieces of $\mathcal{F}$, by definition of local feature size. The ball of radius $r' = r + \|q - p\|$ centered at q contains $\mathbb{B}$ and thus intersects the same two disjoint pieces of $\mathcal{F}$ as $\mathbb{B}$ does. Hence

$$\mathrm{lfs}_{\mathcal{F}}(q) \leq r' = r + \|q - p\| = \mathrm{lfs}_{\mathcal{F}}(p) + \|q - p\| \tag{2.1}$$

and, since the same analysis is valid for p in relation to q,

$$\frac{|\mathrm{lfs}_{\mathcal{F}}(q) - \mathrm{lfs}_{\mathcal{F}}(p)|}{\|q - p\|} \leq 1 \tag{2.2}$$

□

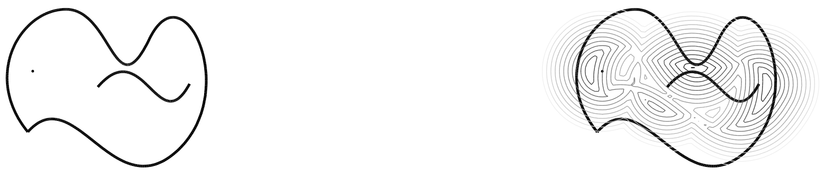
Figure 2.13: Level curves of the local feature size (right) for a manifold complex with 3 pieces (left).

The local feature size can be extended to manifold complexes by stating that it is the radius of the ball that intersects two non-incident pieces or that contains a whole (≥ 1)-piece. This new definition (see Figure 2.13 for an illustrative example) also leads to a 1-Lipschitz function.

A common definition of local feature size for curved geometries used in shape reconstruction is the distance to the medial axis. With this definition, Amenta and Bern [Amenta and Bern, 1998] proved that sample points $\mathbb{S} \subset \mathbb{F}$, from a smooth curve or surface $\mathbb{F}$, spaced a constant factor of the local feature size, is guaranteed to be reconstructed to a homeomorphic simplicial complex. Since the reconstructed shape is a homeomorphic mesh for the unknown $\mathbb{F}$, the sampling condition also guarantees the existence of a homeomorphic mesh to $\mathbb{F}$. However, this condition is much stricter than necessary when $\mathbb{F}$ is known. Similarly to the previous definition, the distance to the medial axis is also 1-Lipschitz.

2.4 Conclusions

The main result of this chapter is the definition of manifold complex, together with its properties. Its definition as a collection of open sets makes all connected pieces to a piece to lie in its boundary. One could make an equivalent definition in terms of closed sets. Some intersection properties are lost in the definition as open sets since all pieces are disjoint.

In mesh generation the local feature size is ideally used to provide a measure of the coarsest mesh possible for an input geometry. For curved manifold complexes this measure is not well defined since there is no bound on how far it is from the coarsest possible simplicial complex. In this case the local feature size is usually degenerated to the coarsest mesh size according to the capabilities of the algorithm.

Chapter 3

Delaunay refinement

3.1 Basic concepts

3.1.1 Simplicial complex

Definition 36 (simplicial k-complex) *A simplicial k-complex is a manifold k-complex or, more specifically, a flat k-complex where all n-pieces are open n-simplices, $n = 0, 1, ..., k$, and all open j-simplicial facets of a n-simplex are in the complex, $j = 0, 1, ..., n$.*

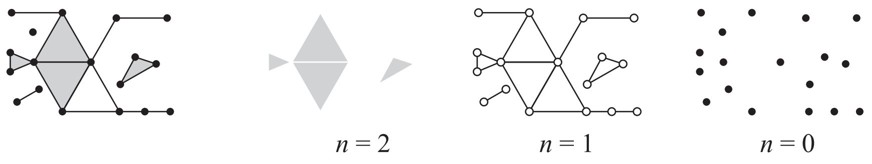

Figure 3.1: Example of a simplicial 2-complex (left) decomposed into its n-dimensional pieces.

Usually a simplicial k-complex is defined as a collection $\mathcal{K}$ of closed n-simplices, $n = 0, 1, ..., k$, where

1. every facet of a simplex in $\mathcal{K}$ is also in $\mathcal{K}$;
2. a nonempty intersection of any two simplices in $\mathcal{K}$ is a facet of each of them.

In this text the definition as a collection of open n-simplices was employed to treat a simplicial k-complex as a natural special case of a manifold k-complex. Hence, a simplicial k-complex from a mesh generator must be an approximate partition of a manifold k-complex, or a finer partition of a flat k-complex.

3.1.2 Delaunay

In 1934, Boris Delaunay [Delaunay, 1934] introduced a condition for simplicial complexes that leaded to many interesting properties. It asserts that no vertices lie inside a circumscribed ball of any element in the simplicial complex, as shown in Figure 3.2, known as the empty ball property. This is the basis of the Delaunay refinement and will be defined next under several aspects.

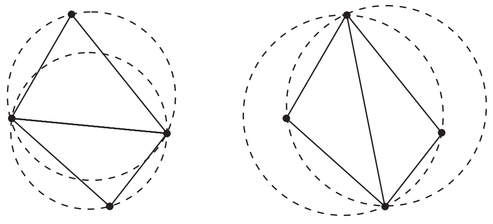
Figure 3.2: Example of a Delaunay triangulation (left) and a non Delaunay triangulation (right) for the same set of vertices.

Definition 37 (Delaunay simplicial complex [Delaunay, 1934]) *A simplicial complex $\mathcal{S}$ is Delaunay iff there exists an open circumscribed ball for every simplex $\mathbb{S} \in \mathcal{S}$ that contains no vertices.*

Definition 38 (strongly Delaunay simplicial complex) *A simplicial complex $\mathcal{S}$ is strongly Delaunay iff there exists a closed circumscribed ball for every simplex $\mathbb{S} \in \mathcal{S}$ that contains no vertices other than the vertices of $\mathbb{S}$.*

A Delaunay simplicial complex always exists for a set of points in $\mathbb{R}^n$, but it is not unique when more than $n+1$ points lie in a common closed n-ball. In turns, a strongly Delaunay simplicial complex does not exist when more than $n+1$ points in $\mathbb{R}^n$ lie in a common closed n-ball, but it is always unique.

3.1.3 Voronoi

In 1908 Georgy Voronoi [Voronoi, 1908] introduced the concept of partitioning a set into closest points to each point of a set.

Definition 39 (Voronoi cell) *A Voronoi cell $\mathbb{V}$ of a point $p \in \mathbb{S} \subset \mathbb{R}^n$ is the set of all points that are closer to p than to any other point in $\mathbb{S}$, i.e. $\mathbb{V} = \mathbb{V}_p = \{x \in \mathbb{R}^n \mid \|x-p\| < \|x-q\|, \forall q \in \mathbb{S}, q \neq p\}$.*

Definition 40 (Voronoi flat complex) *The Voronoi flat complex of a point set $\mathbb{S} \subset \mathbb{R}^n$ is a flat n-complex that contains the Voronoi cells of all elements in $\mathbb{S}$ as n-flat manifolds and their boundary partitioned into lower dimensional flat manifolds.*

The Voronoi flat complex $\mathcal{V}_{\mathbb{P}}$ of a point set $\mathbb{P} \in \mathbb{R}^n$ is commonly known as Voronoi diagram. It is closely related to a Delaunay simplicial complex of $\mathbb{P}$, as shown in Figure 3.3. For any two distinct points in $\mathbb{P}$, the set of equidistant points to both is

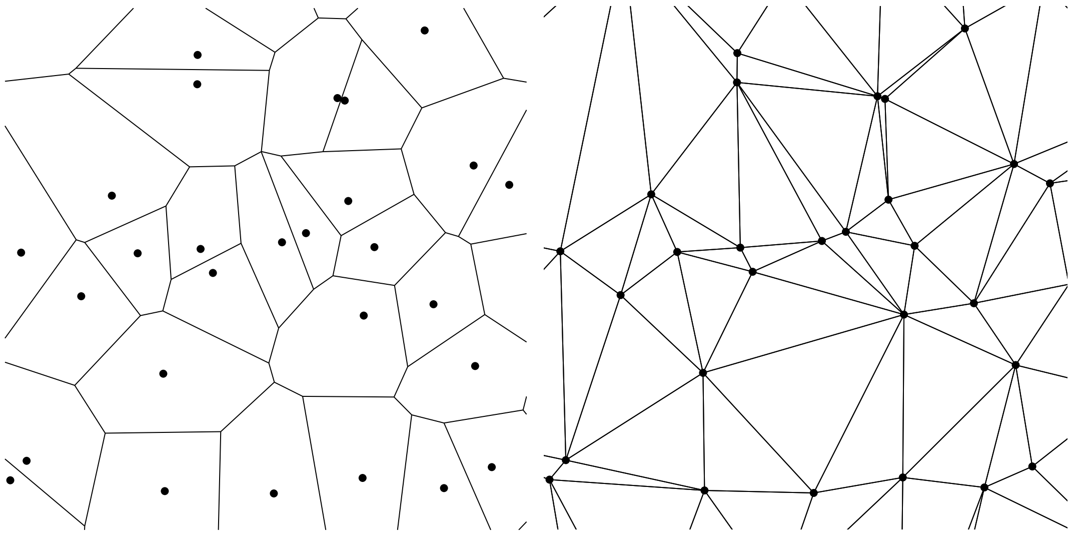
Figure 3.3: Voronoi diagram and Delaunay triangulation for the same set of points.

a $(n-1)$-flat. Conversely, every $(n-1)$-flat in $\mathcal{V}$ is equidistant to two distinct points in $\mathbb{P}$. Following the sequence, every k-flat in $\mathcal{V}$ is equidistant to at least $n-k+1$ points in $\mathbb{P}$, $k = 0, 1, ..., n$. (More than two distinct points can lie in the boundary of the same (> 1)-ball. Hence the limit of $n-k+1$ points is obviously tight iff every point in a k-flat is equidistant to exactly $n-k+1$.)

Definition 41 (fully Delaunay simplicial complex) *Let $\mathcal{D}$ be a simplicial complex and $\mathcal{V}$ be the Voronoi flat complex of the vertices in $\mathcal{D}$. Then $\mathcal{D}$ is fully Delaunay iff it is Delaunay and every k-piece in $\mathcal{V}$ is equidistant to exactly $n-k+1$ vertices in $\mathcal{D}$.*

Every n-simplex of a fully Delaunay simplicial complex $\mathcal{D}$ is uniquely related to a vertex of the Voronoi flat complex $\mathcal{V}_{\mathbb{P}}$ of $\mathbb{P}$ because the set of points closest to $n+1$ affinely independent points in $\mathbb{R}^n$ is a single point. Analogously, every $(n-k)$-simplex in $\mathcal{D}$ is uniquely related to a k-piece in $\mathcal{V}$. So there is a duality relationship between Voronoi flat complexes and Delaunay simplicial complexes.

Theorem 3 ([Watson, 1981]) *Let $\mathbb{P} \in \mathbb{R}^d$ be a set of points. If a subset $\mathbb{A}$ of $n+2$, points in $\mathbb{P}$ lie in a common $(n-1)$-hypersphere, where $n < d$, then there exists a subset $\mathbb{B}$ of $n+3$ points in $\mathbb{P}$ that lie in a common n-hypersphere.*

Proof. Since a $(n-1)$-hypersphere is a cross-section of an n-hypersphere, any point added to $\mathbb{A}$ will lie in a common n-hypersphere with $\mathbb{A}$. □

Corollary 1 *A simplicial complex is fully Delaunay iff it is strongly Delaunay.*

The definition of fully Delaunay was given only to clarify the intrinsic property of a strongly Delaunay simplicial complex. Since they are completely equivalent, only the latter will be used henceforth.

Theorem 4 (strongly Delaunay) *A simplicial complex $\mathcal{D}$ in $\mathbb{R}^d$ is strongly Delaunay iff the sub-collection of n-simplices in $\mathcal{D}$ is strongly Delaunay, for a single arbitrary n.*

Proof. By definition, all n-simplices, for a given n, in a Delaunay simplicial complex have the empty ball property. To prove the converse implication, consider the Voronoi flat complex $\mathcal{V}$ for the vertex set of $\mathcal{D}$. If the simplicial complex is strongly Delaunay, every vertex of $\mathcal{V}$ is uniquely related to a d-simplex of $\mathcal{D}$. Suppose, by contradiction, a $(d-1)$-simplex not in $\mathcal{D}$ has the empty ball property. Then grow this ball until another vertex is met. The resulting d-simplex must be uniquely related to a vertex in $\mathcal{V}$, which cannot be true since it is not in $\mathcal{D}$. By induction, every k-piece of $\mathcal{V}$ is uniquely related to a $(d-k)$-simplex of $\mathcal{D}$. Then grow this ball until another vertex is met. The resulting $(d-k)$-simplex must be uniquely related to a k-piece in $\mathcal{V}$, which cannot be true since it is not in $\mathcal{D}$. □

Shewchuk [Shewchuk, 1997] has proven in his thesis the strongly Delaunay theorem for $n = 2$ using a different approach.

3.1.4 Conforming Delaunay

Definition 42 (conforming Delaunay simplicial complex) *A simplicial complex $\mathcal{S}$ is conforming Delaunay to a flat complex $\mathcal{F}$ iff $\mathcal{S}$ is Delaunay and each piece of $\mathcal{F}$ is a union of pieces in $\mathcal{S}$.*

In a conforming Delaunay simplicial complex of $\mathcal{F}$, the vertices of $\mathcal{F}$ are augmented by additional vertices, called Steiner points, so that its simplices conforms $\mathcal{F}$ and are Delaunay. Edelsbrunner and Tan [Edelsbrunner and Tan, 1993] show that any flat 2-complex $\mathcal{F}$ can be triangulated with no more than $O(m^2 n)$ augmenting vertices, where m is the number of edges in $\mathcal{F}$, and n is the number of vertices in $\mathcal{F}$.

3.1.5 Constrained Delaunay

Definition 43 (visible) *A point $p \in \mathbb{R}^d$ is visible from another point $q \in \mathbb{R}^d$ in respect to a $(d-1)$-piece $\mathbb{S}$ contained in a $(d-1)$-flat $\mathbb{F}$ iff either the line segment pq does not intersect the closure of $\mathbb{S}$, or pq lies in $\mathbb{F}$.*

Definition 44 (constrained Delaunay simplicial complex) *A simplicial complex $\mathcal{S}$ is constrained Delaunay upon a flat complex $\mathcal{F}$ iff it has no vertices not in $\mathcal{F}$, each piece of $\mathcal{F}$ is a union of pieces in $\mathcal{S}$, and there exists an open circumscribed ball for every simplex $\mathbb{S} \in \mathcal{S}$ that contains no vertices visible from $\mathbb{S}$ in respect to $\mathcal{F}$.*

Unlike conforming Delaunay simplicial complexes, constrained Delaunay simplicial complexes do not allow any augmenting vertex. Because of this, some flat k-complexes, for $k \geq 3$, do not have any constrained simplicial complex, like the

Schönhardt's polyhedron [Schönhardt, 1928], as shown in Figure 3.4. Furthermore, they are not necessarily Delaunay (see Figure 3.5). Their definition is an extension of the Delaunay condition when the input contains not only vertices but also higher dimensional flat pieces. A simplicial complex can be conforming constrained.

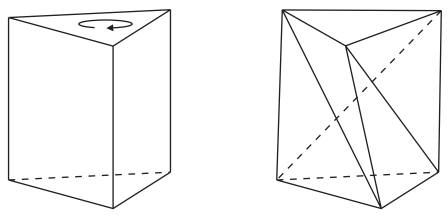

Figure 3.4: Schönhardt's polyhedron (right) is created by rotating one end of a triangular prisma (left).

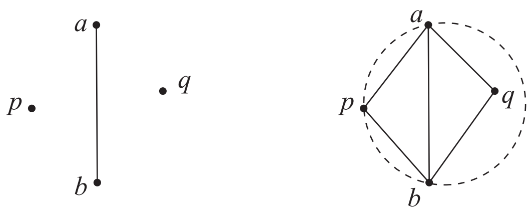

Figure 3.5: Constrained Delaunay triangulation (right) for the input vertices p and q, and segment ab (left). The point q is allowed to lie in the circumcircle of the triangle $\triangle pab$ because ab occludes q from any point in the interior of $\triangle pab$.

The concept of constrained Delaunay was first introduced by Lee [Lee, 1978], but it was named by Chew [Chew, 1989a]. Shewchuk [Shewchuk, 1998] proposed the definition 44 for higher dimensions and proved that it exists if the n-simplices, for all $n \leq d - 2$, are strongly Delaunay for a flat d-complex in $\mathbb{R}^d$.

3.1.6 Weighted Delaunay

Definition 45 (weighted point) *A weighted point is the ordinate pair* $\hat{p} = (p, P^2) \in \mathbb{R}^n \times \mathbb{R}$.

The notation $\hat{p} = (p, P^2)$, where a hat identifies a weighted point and a small and a capital letter represent its respective ordinates, will be adopted henceforth. The second ordinate is defined squared so that P is imaginary when $P^2 < 0$.

Definition 46 (weighted distance) *The weighted distance between two weighted points* $\hat{p}$ *and* $\hat{q}$ *is* $\|\hat{p} - \hat{q}\| = \sqrt{\|p - q\|^2 - P^2 - Q^2}$.

From the definition of weighted distance, a weighted point $\hat{p}$ is interpreted as a ball centered at p with radius P. To clarify this, consider the weighted points $\hat{p} = (p, 0)$ and $\hat{q} = (q, Q^2)$ (see Figure 3.6). Let also t be a tangent point on $\hat{q}$ of

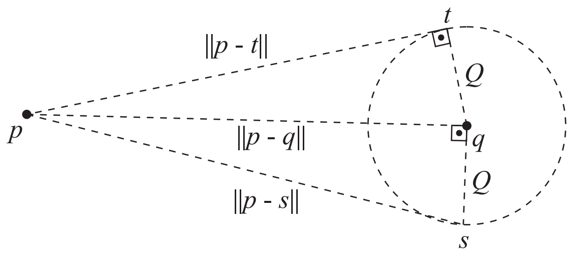

Figure 3.6: Geometric representation of a weighted point $\hat{q}$. The weighted distance to $\hat{p} = (p, 0)$ is $\|p - t\|$ when $Q^2 > 0$, and $\|p - s\|$ when $Q^2 < 0$.

a line that goes through p, and s be the point on $\hat{q}$ so that the line segment qs is orthogonal to pq. The weighted distance between $\hat{p}$ and $\hat{q}$ is $\|p - t\|$ when $Q^2 > 0$, and $\|p - s\|$ when $Q^2 < 0$.

If the respective balls of two weighted points are joint, then the weighted distance between them is imaginary. A geometric interpretation of the weighted distance between two weighted points with non null weights is given in Figure 3.7. Geometric interpretation for other weighted points follows similarly.

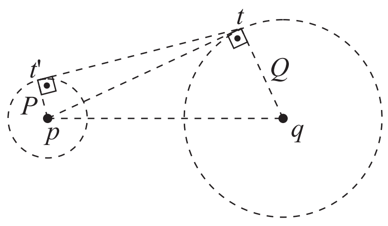

Figure 3.7: Geometric representation of two weighted points. The weighted distance between them is $\|t' - t\|$ when $P^2, Q^2 > 0$.

Two weighted points $\hat{p}$ and $\hat{q}$ are orthogonal when the weighted distance between them vanishes. This can be visualized in Figure 3.6, by replacing the null weight P by $\|p - t\|$ when $Q^2 > 0$ (or by $\|p - s\|$ when $Q^2 < 0$), so that $\|\hat{p} - \hat{q}\| = 0$. A weighted point is said to be orthogonal to a set of weighted points if it is orthogonal to every element in it.

Definition 47 (orthoball) *An orthoball for a set of weighted points* $\mathbb{V} = \{\hat{\nu}_1, ..., \hat{\nu}_k\}$ *is a ball whose interpretation as a weighted point is orthogonal to* $\mathbb{V}$.

Definition 48 (weighted Delaunay simplicial complex) *A simplicial complex is weighted Delaunay iff there exists an open orthoball for every simplex in it that contains no vertices.*

3.1.7 Restricted Delaunay

Definition 49 (restricted Delaunay simplicial complex) *A simplicial complex is restricted Delaunay to a topological space* $\mathbb{X}$ *iff every piece of the dual Voronoi flat complex intersects* $\mathbb{X}$.

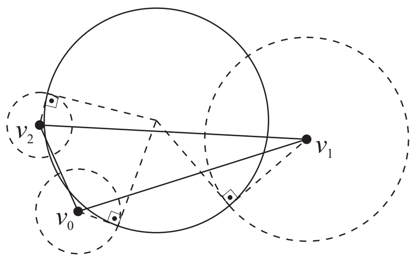

Figure 3.8: Geometric representation of an orthocircle for weighted points with positive weights.

The definition 49 was given by Edelsbrunner and Shah [Edelsbrunner and Shah, 1994] (see Figure 3.9), as a generalization of the Delaunay condition over curved surfaces proposed by Chew [Chew, 1993]. The Chew's criterion is the emptiness of the open circumscribed ball with center on the surface, for every triangle in the mesh.

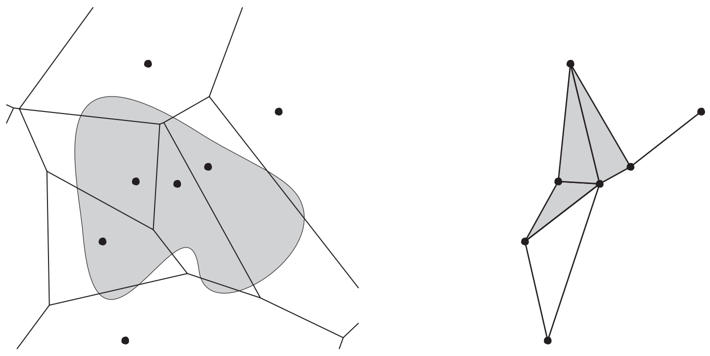

Figure 3.9: Delaunay simplicial complex (right) restricted to a set.

Theorem 5 (nerve [Leray, 1945]) *If the intersection between a set $\mathbb{S}$ and the Voronoi cell of each point $p \in \mathbb{P}$ is either empty or contractible, then the restricted Delaunay simplicial complex of $\mathbb{P}$ restricted to $\mathbb{S}$ is homotopy equivalent to $\mathbb{S}$.*

A graphical representation of the nerve theorem is given by Figures 3.9 (where the condition is not verified) and 3.10 (where the condition is verified).

The restricted Delaunay simplicial complex concept in $\mathbb{R}^2$ is closely related to the medial axis, as shown in Figure 3.11.

3.1.8 Homeomorphic

Definition 50 (homeomorphic simplicial complex) *A simplicial complex $\mathcal{S}$ is homeomorphic to a manifold complex $\mathcal{F}$ iff every manifold $\mathbb{M}$ in $\mathcal{F}$ is homeomorphic to a union of simplices in $\mathcal{S}$ with all vertices lying in $\mathbb{M}$.*

Note that a homeomorphic simplicial complex also requires the vertices of simplices to lie in the pieces they represent. The coarsest homeomorphic simplicial

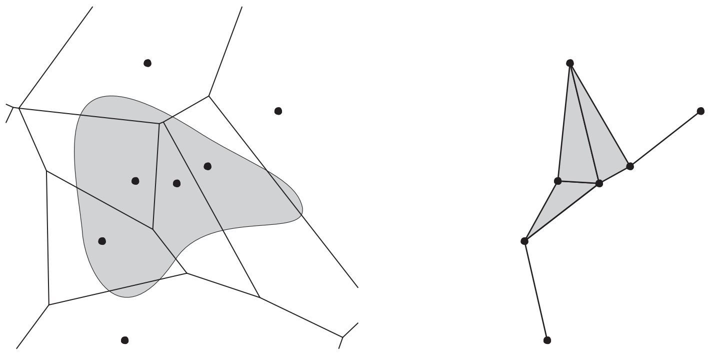
Figure 3.10: A restricted Delaunay simplicial complex (right) that is homotopy equivalent to its restricting set.

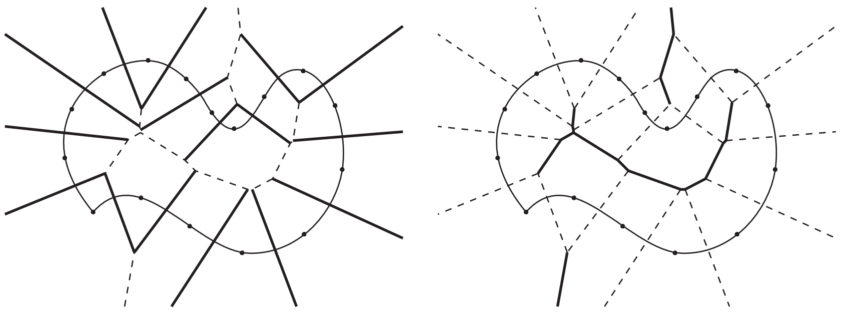
Figure 3.11: Voronoi diagram of point samples on a curve. The set of edges from the Voronoi diagram that intersects the curve (left) is complementar to the one that approximates the medial axis of the curve (right).

complex of an input manifold complex must be considered as reference for size optimality (see Figure 3.12). This is consistent with the size optimality defined for a flat complex input, which is a special case of a manifold complex.

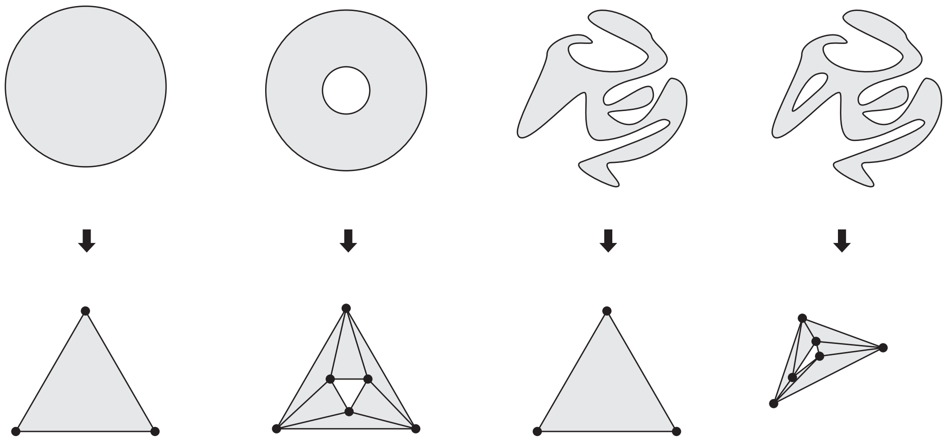
Figure 3.12: Example of coarsest homeomorphic simplicial complexes for several manifold complexes.

Theorem 6 (topological ball property [Edelsbrunner and Shah, 1994]) *Let $\mathbb{S} \subset \mathbb{R}^n$ be a set of points over a compact d-manifold $\mathbb{X}$, $\mathcal{D}_{\mathbb{S}}$ be the Delaunay simplicial n-complex of $\mathbb{S}$, and $\mathcal{V}_{\mathbb{S}}$ be the Voronoi flat n-complex of $\mathbb{S}$. If the intersection with $\mathbb{X}$ of the dual Voronoi $(n-d+k)$-piece of any $(d-k)$-simplex in $\mathcal{D}_{\mathbb{S}}$, $k \leq d$, is either empty or homeomorphic to a k-ball, then the Delaunay simplicial complex $\mathcal{D}_{\mathbb{S}}$ restricted to $\mathbb{X}$ is homeomorphic to $\mathbb{X}$.*

Size optimal does not mean best representation of the input pieces, but that it is represented in the coarsest manner. Optimal sampling points for best representation of Lipschitz surfaces have been proposed by Boissonnat and Oudot [Boissonnat and Oudot, 2007] with theoretical guarantees. It may serve as refinement criteria if one desires this kind of output simplicial complex. This emphasizes why the coarsest simplicial complex is always desirable.

3.2 Delaunay refinement

The Delaunay refinement is based on the point insertion in the circumcenter of a simplex [Frey, 1987], preserving the Delaunay property of the simplicial complex (see Figure 3.13).

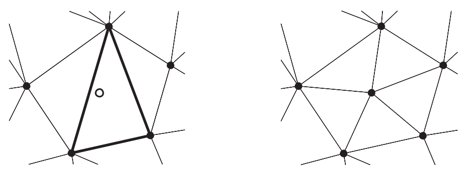
Figure 3.13: Fundamental idea of the Delaunay refinement.

Theorem 7 (termination) *Let ℓ be the minimum edge length of a simplex and R its circumradius. If the point insertion is only in the circumcenter of simplices with $\ell < R$, then the Delaunay refinement will lead to a simplicial complex with $\ell \geq R$ for every element.*

Proof. The point insertion in the circumcenter of a simplex $\mathbb{S}$ cannot create edges with length smaller than R, since the open circumscribed ball of $\mathbb{S}$ is empty by the Delaunay property. Thus $\ell' \geq R > \ell$ for every new edge created with length ℓ' and a simplex with $\ell < R$. No creation of smaller edges cannot hold forever while new nodes are inserted, implying that $\ell \geq R$ will hold true for every element after some iterations. □

The termination theorem of the Delaunay refinement implies that every triangle will have internal angles in the interval $[30^o, 120^o]$ in a simplicial 2-complex. This comes from the relation $\ell = 2R \sin\theta$ between an angle θ and its opposite edge length ℓ in a triangle (see Figure 3.14), which for $\ell = R$ leads to $\sin\theta = 1/2 \Rightarrow \theta = 30^o$. Note in Figure 3.14 that, since the triangles Δrcq and Δrcp are isosceles, $\angle rcq = \pi - 2(\theta + \beta)$ and $\angle rcp = \pi - 2\beta$. Thus $\alpha = \angle rcp - \angle rcq = 2\theta$.

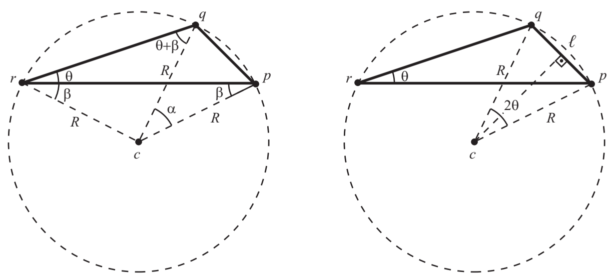

Figure 3.14: Properties between a triangle and its circumcircle: $\alpha = 2\theta$ (left) and $\ell/R = 2\sin\theta$ (right).

The ratio ℓ/R as a quality measure, *i.e.* distance to a regular simplex, works well for triangles, but it fails for n-simplices when $n \geq 3$. An example of this failure is the 3-simplex sliver, shown in Figure 3.15. In 1999, Cheng et al. [Cheng et al., 1999] proved the existence of a sliver free weighted Delaunay tetrahedralization for any Delaunay tetrahedralization, without moving or adding any vertex.

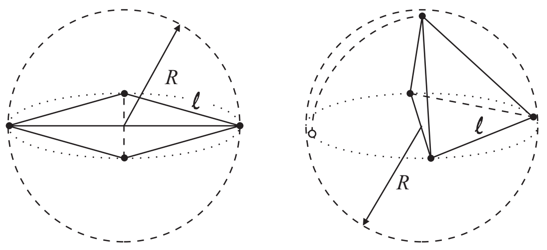

Figure 3.15: The flat tetrahedron (sliver) on the left has the same ℓ/R ratio of the one in the right.

3.3 Delaunay refinement for flat complexes

The Chew's first algorithm for quality quasi-uniform conforming Delaunay simplicial complex output; the Ruppert's algorithm for quality, optimum size and grading conforming Delaunay simplicial complex output; and the Chew's second algorithm for quality, optimum size and grading conforming constrained Delaunay simplicial complex output; are the most remarkable Delaunay refinement algorithms for flat complex inputs. They are depicted next.

3.3.1 Chew's first algorithm

Chew's first algorithm [Chew, 1989b] follows straightforwardly from the Delaunay refinement. It was proposed for flat 2-complexes as the first provably good quality Delaunay refinement algorithm. The fundamental idea is to segment all input edges to a length h and then apply the Delaunay refinement to the resulting vertices until no more simplex with circumradius greater than h lasts. Assuming the vertices are at least h away from each other in the beginning, no edge lengths smaller than h

will be in the final triangulation, so that all input edges will be conformed, as shown in Figure 3.16. Any procedure that guarantees that a simplex is part of a simplicial complex is called protection.

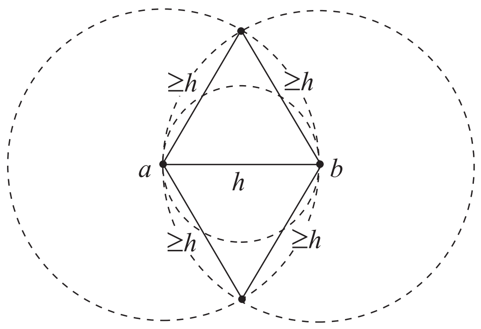

Figure 3.16: If the distance between any two points is at least h, then any two points a and b that are h away from each other will have an empty diametral circle.

After the segmentation process all edge lengths must be in the range $[h, L]$. To establish the upper bound L, consider a worst case regular k-simplex in $\mathbb{R}^n$ with edge length L (see Figure 3.17). Because no simplex whose circumradius is less than h will be selected for refinement, consider the closest point p lying h away from the vertices v_i, $i = 0, 1, ..., k$, of the k-simplex. The circumcenter q of the n-simplex containing p and the k-simplex must be inside the n-balls of radius h centered at the k-simplex vertices, so that it will not be selected for refinement and the $(k+1)$-simplex formed by the vertices v_i and p has an empty ball (by the strongly Delaunay theorem this ensures that the simplex is in the Delaunay simplicial complex). The limit is when q is also h away from the vertices, where $h^2 = (h/2)^2 + R_k^2$ and R_k is the smallest circumradius of the k-simplex (see relations (A.13)). Thus

$$L = h\sqrt{\frac{3(k+1)}{2k}} \tag{3.1}$$

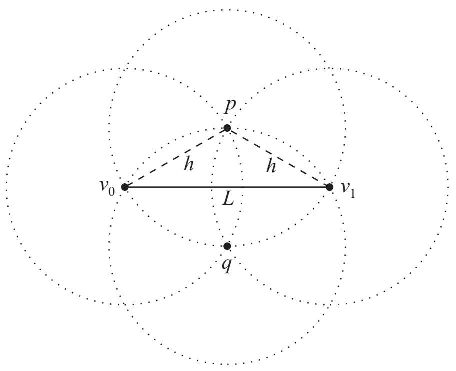

Figure 3.17: Worst case in the edge protection of the Chew's first algorithm.

Within input angles smaller than 60° at a vertex p, it is impossible to place vertices h away from each other, as shown in Figure 3.18. Under this situation, segmenting vertices that do not satisfy the h distance in a small input angle at p can be placed exactly on hyperspheres centered at p with radius ih, for integer i, so that the open diametral balls of the conforming segments contains no vertex.

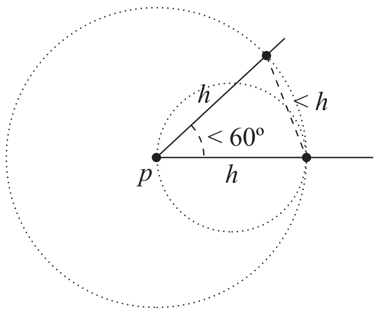

Figure 3.18: Input angles smaller than 60^o at a vertex p makes it impossible to place vertices h away from each other during the segmentation process.

Any simplex with an edge of length greater than $2h$ has a circumradius greater than h and will be selected for refinement. Thus, in the final simplicial complex, all edge lengths will be in the range $[h, 2h]$, where the lower bound is only violated within small input angles. Hence, the output conforming simplicial complex is quasi-uniform. Notice also that not even the Chew's algorithm itself would be able to generate the input pre-discretization which is required to be in the range $[h, L]$.

3.3.2 Ruppert's algorithm

Ruppert [Ruppert, 1994] proposed a Delaunay refinement algorithm for triangulations, latter extended for tetrahedralizations by Shewchuk [Shewchuk, 1997], where the input is refined throughout the refinement instead of being pre-segmented. The fundamental idea is to guarantee that the minimum circumscribed ball of lower dimensional simplices are empty (see Figure 3.19). Hence, lower dimensional simplices conforming the input will have at least one empty circumscribed ball and, according to the strongly Delaunay theorem, they will be part of the conforming Delaunay simplicial complex.

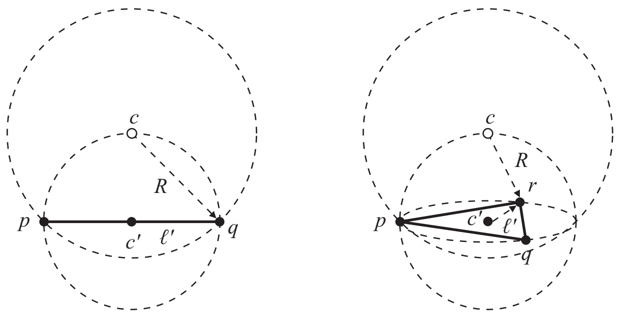

Figure 3.19: Protection of lower dimensional simplices by minimum circumscribed disk (left) and ball (right).

If a new vertex v lies inside the protecting ball of a simplex, then v is said to encroach upon that simplex. Whenever a new vertex encroaches upon a $(d-1)$-simplex, it is rejected for refinement, and the circumcenter of that $(d-1)$-simplex is elected for insertion. This recursive insertion attempt can go on until 1-simplices. Under this scheme, the Ruppert's algorithm can be split up into two major steps: recovery and refinement.

During the recovery, all input pieces are recursively meshed from lower to higher dimensional ones until every simplex has an empty minimum circumscribed ball. Every flat containing a piece defines a parametric space for its own mesh. The circumcenter of a simplex is elected for insertion if its protecting ball is not empty. Once all simplices of a piece are protected, they are guaranteed to be in the simplicial complex. Hence, after the recovery step, the simplicial complex is guaranteed to conform the input flat complex, and the input flat complex may be discharged since it is exactly represented in the simplicial complex.

During the refinement, new vertices are inserted in the conforming simplicial complex until specified output features are attained. Usually bounds on quality or edge length are the features supported by theory. The circumcenter of a d-simplex is elected for insertion if it does not satisfy the requirements.

Small angles

If two input edges are incident in small angle, then the algorithm may stuck in an infinite encroachment, as shown in Figure 3.20. The largest angle θ where this problem is verified is when the point q lies in the pole of the diametral circle of vp, so that p' lies in the pole of the diametral circle of vq, and q' lies in the pole of the diametral circle of vp'... and therefore $\theta = 45^{\circ}$.

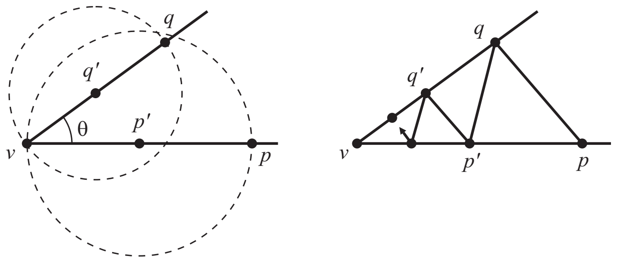

Figure 3.20: The point q encroaches upon vp, which is split at p', which encroaches upon vq, which is split at q'...

To solve this problem Ruppert proposed a modified segment splitting using concentric shells, as shown in Figure 3.21. The first split of an input segment is at its midpoint. The subsequent splits are at its intersection with concentric circles (shells) nearest to its midpoint. The concentric shells have radius $r2^i$, where i is integer and r is a small constant. This modified split synchronizes adjacent segments splits and prevents the infinite encroachment.

Skinny triangles constrained by small input angles must not be elected for refinement because there is no way to get rid of them. Miller et al. [Miller et al., 2003]

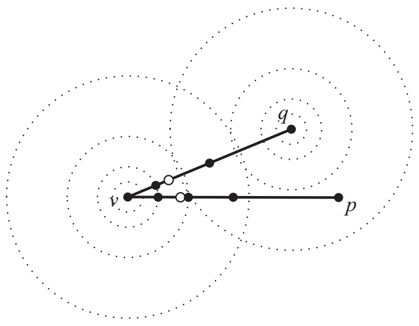

Figure 3.21: Concentric shells used to split segments. The first split of an input segment is at its midpoint, and the subsequent splits are at its intersection with the concentric shell nearest to its midpoint.

proposed a modified refinement selection which detects this situation. Let a and b be vertices of a skinny triangle $\triangle acb$ lying on distinct input incident edges. The triangle is only selected for refinement if c is an input vertex and the angle $\angle acb > 60^o$ (see Figure 3.22). Notice that this modified refinement allows new small angles (*i.e.* skinny triangles not constrained by the input) to appear inside small input angles. Fortunately, its implementation is simple and it always terminates.

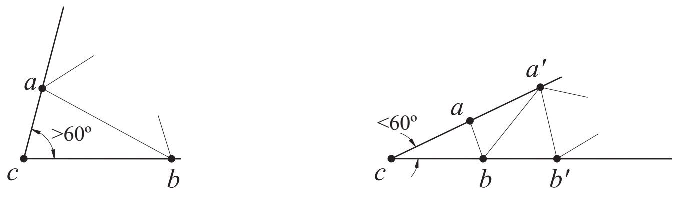

Figure 3.22: The triangle $\triangle abc$ on the left can be selected for refinement, while the triangles $\triangle abc$, $\triangle aba'$ and $\triangle a'b'b$ on the right cannot.

Shewchuk [Shewchuk, 2002] has shown a counterexample where skinny triangles - consider anyone with a dihedral angle smaller than θ - appear outside an small input angle ϕ next to a large input angle (see Figure 3.23), whenever any skinny triangle not constrained by the input can be selected for refinement. In this case, other skinny triangles pop up outside the small input angle, which unleashes an endless refinement towards the small input angle. Therefore, there always exists an input planar graph where no meshing algorithm can triangulate it without creating new angles smaller than θ.

Theorem 8 (grading [Mitchell, 1994]) *If triangles incident are incident in a common node p have no angle smaller than θ, then the ratio b/a, $b > a$, between two opposite edges, with respective lengths a and b, along a line containing p is*

$$\frac{b}{a} \leq (2\cos\theta)^{\pi/\theta}, \quad \theta \in [0, \pi/3] \tag{3.2}$$

and the limit is tight when π/θ is integer.

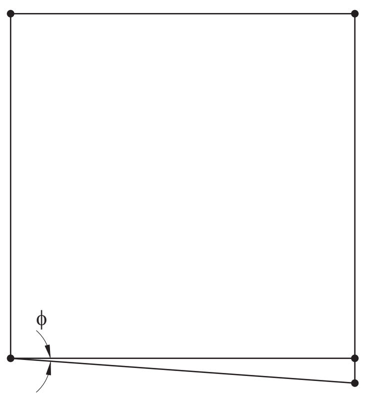

Figure 3.23: Shewchuk's counterexample that shows skinny triangles near, but outside, small input angles. These triangles can be selected for refinement, but other skinny triangles will take place.

Proof. The largest ratio b/a occurs when all triangles are isosceles with two angles θ, one of them incident to p, as shown in Figure 3.24. The length ratio between the largest edge and one of the edges of an isosceles triangle with minimum angle θ is $2\cos\theta$. Since there may be up to π/θ adjacent triangles incident to p, $b/a \leq (2\cos\theta)^{\pi/\theta}$. □

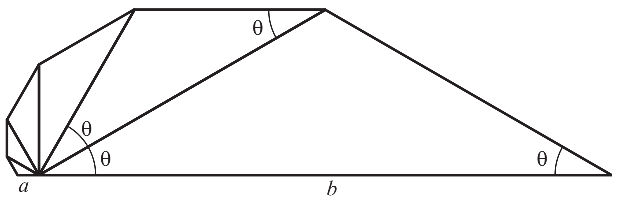

Figure 3.24: Configuration for maximum ratio b/a for triangles with no angle larger than θ.

The grading theorem quantifies the maximum grading allowed in a triangulation with no angle smaller than θ, *i.e.* the trade off between grading and quality. For $\theta = 30^o$ the grading is smaller than 27, but for $\theta = 20^o$ the grading can be as large as 292 (see Figure 3.25).

Let three input segments intersect at an input vertex o, the points p and r share the same concentric shell so that $|po| = |ro| = b$, $\angle por = \phi$ so that $\triangle por$ is the only skinny triangle, p and q lie on the middle segment so that $\angle pqr > \theta$ and $|pq| = a$ (see Figure 3.26). The lower bound for the ratio a/b is given by the grading theorem, considering the fan from q to o with vertex on p. The upper bound for the ratio a/b is given the relation between $\triangle por$ and $\triangle pqr$, where $|pr| = 2b\sin(\phi/2)$ and the height of $\triangle pqr$ is $a\sin\theta = |pr|\sin[(\pi-\phi)/2-\theta]$. Notice that the greater θ is, the less will be a. Hence

$$\frac{a}{b} \geq (2\cos\theta)^{-\pi/\theta} \tag{3.3a}$$

$$\frac{a}{b} \leq \frac{\sin\phi}{\tan\theta} + \cos\phi - 1 = \frac{\sin(\theta+\phi)}{\sin\theta} - 1 \tag{3.3b}$$

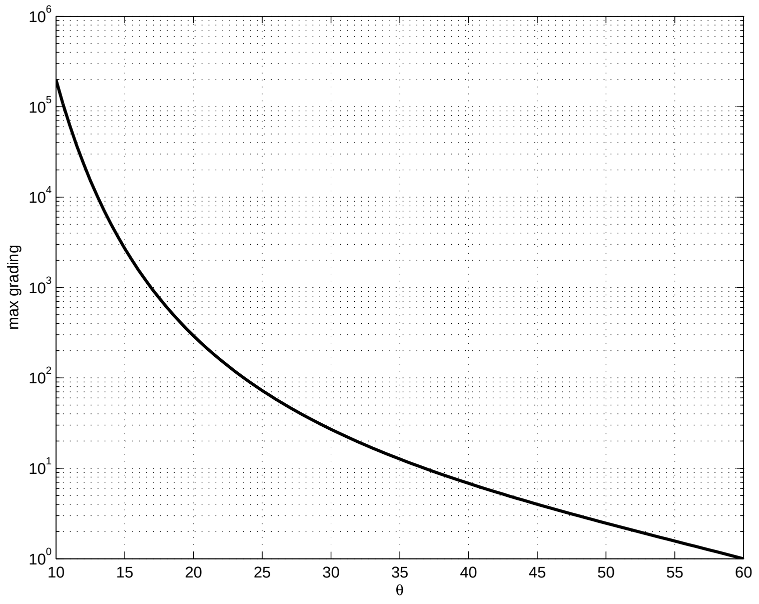

Figure 3.25: Maximum grading allowed in triangulations with no angle larger than θ.

which cannot be simultaneously satisfied for (see Figure 3.27)

$$\phi < \arcsin\{[(2\cos\theta)^{-\pi/\theta} + 1]\sin\theta\} - \theta \tag{3.4}$$

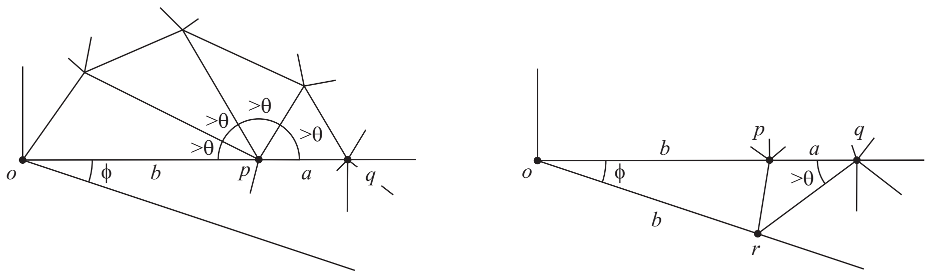

Figure 3.26: Geometric configurations for lower (left) and upper (right) bounds for the ratio a/b.

3.3.3 Chew's second algorithm

Chew's second algorithm [Chew, 1993] is equivalent to Ruppert's algorithm by replacing the minimum circumscribed balls by 30° lenses (see Figure 3.28) and using conforming constrained Delaunay simplicial complexes, as shown by Shewchuk in his thesis [Shewchuk, 1997]. An n-lens is formed by joining two caps of n-balls. The minimum circumscribed n-lens for an $(n-1)$-simplex has the simplex vertices lying in its joint, and for a $(\leq n-2)$-simplex the respective $(\leq n-1)$-lens is rotated around the flat containing the simplex. When an encroachment occurs, all vertices

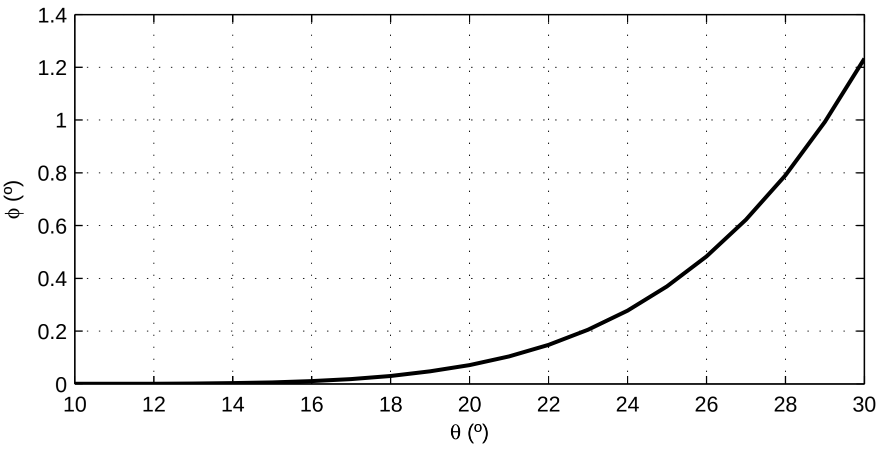

Figure 3.27: By Shewchuk's counterexample, there always exists a planar graph with an angle ϕ that cannot be triangulated without creating new angles smaller than θ.

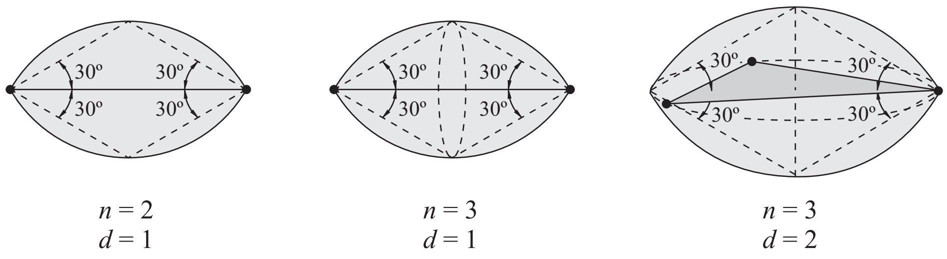

Figure 3.28: Lenses of 30º for d-simplices in $\mathbb{R}^n$.

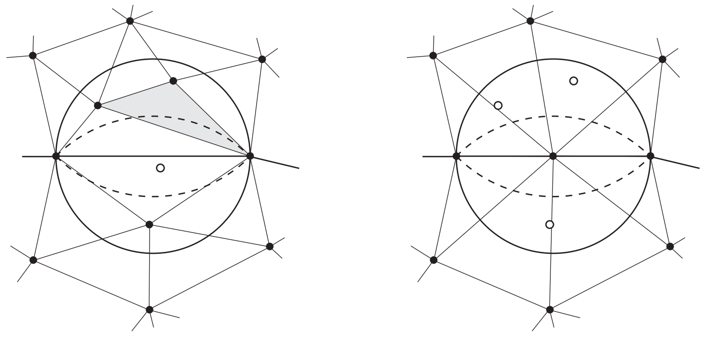

Figure 3.29: When a insertion point encroaches a diametral lens, all points lying inside the respective diametral circle are removed before inserting the segment midpoint.

inside the encroached circumscribed ball are removed before adding a new vertex at its center, as shown in Figure 3.29.

Some properties of lenses are given next.

Theorem 9 (extended Thales) *The angle $\angle apb$, formed by a point p lying in the boundary of the diametral lens of the line segment ab, is $\pi - \theta$, where θ is the angle at which the lens meet ab.*

Proof. Let $a' = a - p$ and $b' = b - p$ for a line segment ab, and c be the circumcenter of the triangle $\triangle abp$. Whenever p lies in the boundary of the diametral lens of ab the triangles $\triangle acp$ and $\triangle bcp$ become isosceles (see Figure 3.30), so that $\angle acp = \pi - 2\alpha$ and $\angle bcp = \pi - 2\beta$. Since $\angle acp + \angle bcp = 2\theta$, $\angle apb = \alpha + \beta = \pi - \theta$. □

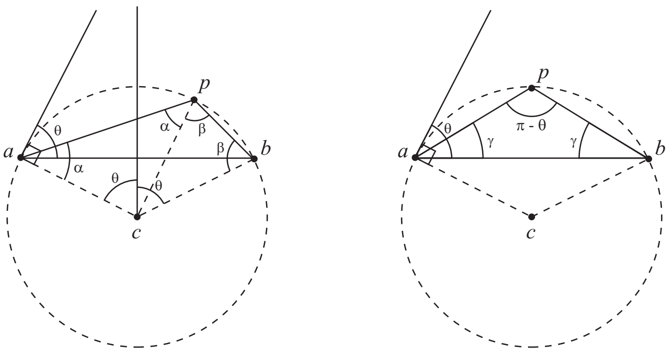

Figure 3.30: Proof of the extended Thales theorem $\alpha + \beta = \pi - \theta$ (left), and its corollary $\gamma = \theta/2$ (right).

Corollary 2 *The angle $\angle pab = \theta/2$ when p lies in the lens pole.*

Proof. When p lies in the lens pole, the triangle $\triangle pab$ becomes isosceles (see Figure 3.30). Hence $\pi - \theta + 2\gamma = \pi$ and hence $\angle pab = \gamma = \theta/2$. □

3.4 Delaunay refinement for manifold complexes

The first issue that comes up on Delaunay refinement for manifold complexes is how to state the Delaunay criterion over curved manifolds. One could think about it as the ball emptiness using the geodesic distance over the manifold as metric. This is equivalent to a Delaunay refinement on a parametric space for the manifold where the Euclidean distance on parametric space equals the geodesic distance on the domain space. Even if such a convenient parametric space was defined, good quality, size or grading bounds would not necessarily imply in good bounds in the domain space. This would only hold true with guarantees when the mesh becomes finer enough to consider approximation bounds between the Euclidean and geodesic distances.

3.4.1 Boivin-Gooch's algorithm

In 2002 Boivin and Ollivier-Gooch [Boivin and Ollivier-Gooch, 2002] derived a Ruppert Delaunay refinement algorithm for manifold 2-complexes, which was revised by Gosselin and Ollivier-Gooch [Gosselin and Ollivier-Gooch, 2007].

In this algorithm the input is firstly sampled so that the total angular variation of each subcurve is less than $\pi/6$. This ensures that the subcurve lies inside the diametral 30° lens of its respective subsegment. Furthermore, any diametral lens of a bisected subsegment lies inside the subsegment diametral circle (see Figure 3.31).

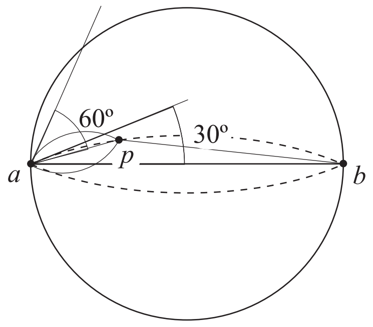

Figure 3.31: As p gets closer to a through the 15° lens, the subsegment lens tends to touch the diametral circle.

From the initial sampling condition, the vertex and segment sets are augmented so that the diametral circle of each segment e does not contain any vertex visible from the interior of e (see Figure 3.32). A constrained triangulation is then created with the augmented vertex and segment sets.

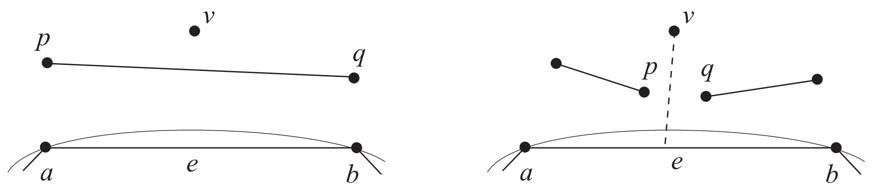

Figure 3.32: The interior of the segment ab cannot see the vertex v (left), and can see the vertex v (right). Testing only the visibility from a and b is sufficient to guarantee a correct sampling condition.

The refinement then proceeds similarly to Chew's second algorithm. Whenever a subsegment lens is encroached, the respective subcurve is split up by placing a new vertex on it that is equidistant to its endpoints. If this insertion causes a crossover (see Figure 3.33), the crossed subsegment is split up instead.

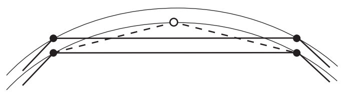

Figure 3.33: The split up of a subsegment could cross over another one.

3.4.2 Cheng-Dey-Ramos's algorithm

Cheng, Dey and Ramos proposed a Delaunay refinement algorithm guided by violations of the conditions established in the topological ball property theorem [Cheng et al., 2007b].

In a first step, all input vertices and samples on input curves are weighted such that, for any two adjacent points p and q in this set, the Voronoi face $\mathbb{V}_{pq}$ dual to the edge pq is the only face that intersects pq, and intersects it only once. This guarantees that all vertices on input curves will be part of the weighted restricted Delaunay simplicial complex. However, no other vertices can be inserted in the curves during refinement. The refinement proceeds firstly in the surfaces, and then in the volumes.

Let $\mathcal{D}$ be a simplicial 3-complex and $\mathcal{M}$ be a manifold 3-complex. The violations of the conditions established in the topological ball property theorem detected by Cheng-Dey-Ramos's algorithm are (see Figure 3.34)

i. The Voronoi edge $\mathbb{V}_t$ dual to a triangle $t \in \mathcal{D}$ intersects the 2-manifold $\mathbb{M} \in \mathcal{M}$ more than once. The farthest intersection point x from the vertices of t is inserted in $\mathcal{D}$.

ii. The normal angular deviation inside $\mathbb{V}_p \cap \mathbb{M}$ from the normal at a point p in a 2-manifold $\mathbb{M} \in \mathcal{M}$, where $\mathbb{V}_p$ is the Voronoi cell of p, is greater than $\theta \in (0, \pi/6)$. A point x where the deviation is θ is inserted in $\mathcal{D}$.

iii. An edge pq restricted to a 2-manifold $\mathbb{M} \in \mathcal{M}$ where $p \in \mathbb{M}$ and $q \notin \mathbb{M}$, or p and q are in the boundary of $\mathbb{M}$ but are non-adjacent. A point in $\mathbb{V}_{pq} \cap \mathbb{M}$ is inserted in $\mathcal{D}$.

iv. The triangles restricted to a 2-manifold $\mathbb{M} \in \mathcal{M}$ incident to a point $p \in \mathbb{M}$ do not form a topological disk. The point $x \in \mathbb{V}_t \cap \mathbb{M}$ farthest from the vertices of a triangle t incident to p is inserted in $\mathcal{D}$.

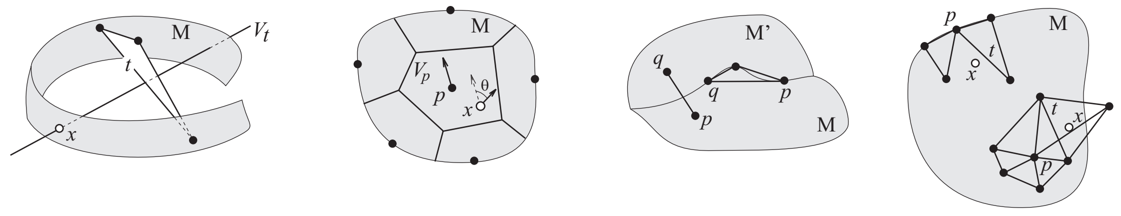

Figure 3.34: Graphical representation of the four kind topological violations, respectively from left to right, detected in a simplicial complex by Cheng-Dey-Ramos's algorithm.

The detection of violations is computationally expensive and hard to implement. It was latter simplified by Cheng et al. [Cheng et al., 2007a] so that a single violation condition is checked (see Figure 3.35)

i. The triangles restricted to a 2-manifold $\mathbb{M} \in \mathcal{M}$ incident to a point $p \in \mathbb{M}$ do not form a topological disk with all vertices lying in $\mathbb{M}$, or p belongs to the boundary of $\mathbb{M}$ and it is connected to a non-adjacent vertex. The point $x \in \mathbb{V}_t \cap \mathbb{M}$ farthest from the vertices of a triangle t incident to p is inserted in $\mathcal{D}$.

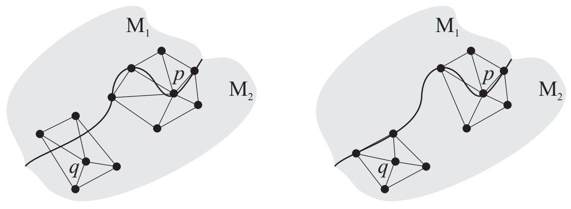

Figure 3.35: Graphical representation of the disk conditions where they are violated (left) and where they are satisfied (right).

3.4.3 Ruppert's algorithm revisited

A Delaunay refinement algorithm can be stated in two steps. The first step, named facet recovery, creates a homeomorphic simplicial complex to the input, so that minimal requirements are satisfied for the second step, named refinement, which incrementally add new vertices to the simplicial complex. Nevertheless, refinement is also used using the facet recovery step.

During the facet recovery step, homeomorphic representations in the simplicial complex are created from lower to higher dimensional pieces of the input manifold complex (see Figure 3.36). Each input piece has a parametric space where vertices of incident lower dimensional pieces can be projected onto. In this point of view, a piece defines only a parametric space, and the incident lower dimensional pieces define its boundaries, trimmings or simply constraints. As soon as a mesh is created for a piece - without any additional vertex because lower dimensional simplices are already protected -, every simplex outside the piece is removed. The mesh over a piece is refined until the strongly Delaunay theorem is verified, and new vertices can be inserted in incident pieces as needed.

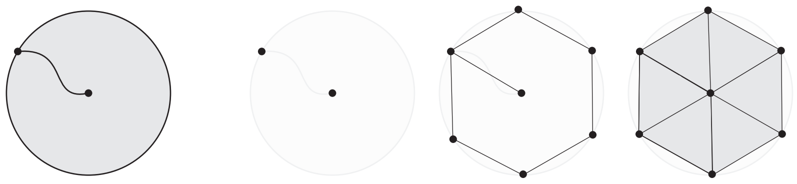

Figure 3.36: Some recovery steps (right) for the manifold complex input (left).

As long as the split point c of a segment ab lies inside the diametral ball of ab, the subsegment ac (and also bc) will also have an empty circumscribed ball as shown in Figure 3.37. This guarantees that the subsegments will also be part of the simplicial complex after ab is split. This property is the fundamental idea of the refinement and will be extended to higher dimensional simplices next.

Theorem 10 (protecting ball) *Let $\mathcal{D}$ be a Delaunay simplicial complex and $\mathcal{C} \subseteq \mathcal{D}$ be the subset of k-simplices that contain a vertex $p \notin \mathcal{D}$ in their circumscribed balls $\mathcal{B}$. Each k-simplex formed by connecting p to a $(k-1)$-simplex in the boundary of $\bigcup \mathcal{C}$ will also have a circumscribed ball whose interior contains no vertex in $\mathcal{D}$.*

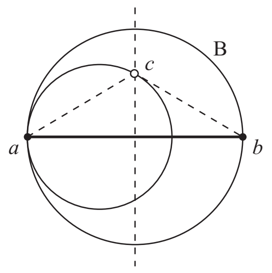

Figure 3.37: If the diametral ball $\mathbb{B}$ of the segment ab is empty, then the subsegment ac will also have an empty circumscribed ball which is tangent to $\mathbb{B}$ at a whenever $c \in \mathbb{B}$.

Proof. By the strongly Delaunay theorem, it is sufficient to prove that every edge incident to p has an empty circumscribed ball (*i.e.* pq is strongly Delaunay). For each edge pq, let $\mathbb{B} \in \mathcal{B}$ be a circumscribed ball that goes through q. Since the interior of $\mathbb{B}$ contains no vertex in $\mathcal{D}$, the interior of the ball tangent to $\mathbb{B}$ at q that goes through p will not contain either, as long as $p \in \mathbb{B}$. Hence pq is strongly Delaunay. □

The protecting ball theorem is the fundamental idea of the Bowyer-Watson incremental algorithm [Bowyer, 1981, Watson, 1981]. Notice, however, that this theorem is valid for every sub-dimension of the simplicial complex.

As long as a new vertex of a k-piece encroaches upon any established empty circumscribed ball of a k-simplex representing it, the refinement will work properly. This condition must be satisfied throughout the refinement.

One may question how the refinement can work considering that there exists infinite many circumscribed balls for k-simplices in $\mathbb{R}^d$, $k < d$. Let $\mathbb{B}$ be an empty circumscribed ball established for a k-simplex $\mathbb{S}$. If $\mathbb{B}$ is encroached by the new vertex, then the sub-simplices will be Delaunay by the protecting ball theorem. If $\mathbb{B}$ is not encroached by the new vertex, then $\mathbb{B}$ will remain empty and $\mathbb{S}$ will still be Delaunay. Notice that this is true even if there exists circumscribed balls encroached and not encroached by the new vertex for the same k-simplex $\mathbb{S}$.

The circumscribed ball chosen to remain empty throughout the Delaunay refinement is the one whose center lies in the simplex flat, or the one whose center lies on the respective curved piece. After inserting a vertex, these balls may not be empty, so that any vertex lying inside them must be removed.

The algorithm just described here behaves like the Ruppert's refinement algorithm when the input is a flat complex. Because an approximate flat complex is kept for a input curved manifold complex, it is expected that all termination and quality guarantees are asymptotically inherit from Ruppert's algorithm (unfortunately quality guarantees fail in dimensions greater than two). However the point insertions are in the curved pieces, so that the introduction of small features must be further analyzed in order to state any kind of grading and size guarantees.

Alternatively, the existence of an initial constrained simplicial complex of each piece is guaranteed by Shewchuk's theorem [Shewchuk, 1998] as long as the strongly

Delaunay theorem is verified for lower dimensional pieces. Combining constrained simplicial complexes with lenses reduces the mesh cardinality.

Degeneracies

When the endpoints of a curve segment are connected by another curve or line segment, as shown in Figure 3.38, the initial simplicial complex would be degenerate to two vertices doubly connected. Notice that the initial line segments are not encroached, and hence they are not split up. To avoid this situation, the curve segment could be split up in two.

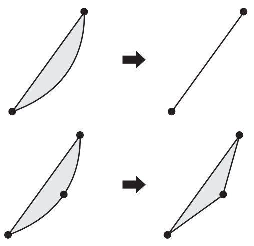

Figure 3.38: Degenerate (top) and non-degenerate (bottom) curves in manifold complexes.

In higher dimensions, this kind of degeneracy may occur in many distinct topological configurations, as shown in the examples in Figure 3.39.

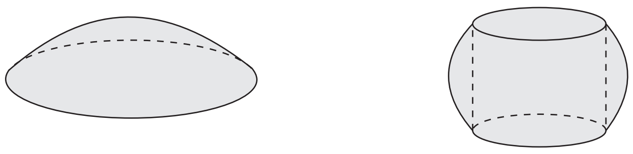

Figure 3.39: Degenerate surfaces in manifold complexes.

3.5 Conclusions

Two new theorems for Delaunay refinement algorithms were introduced in this chapter. The first one, the strongly Delaunay theorem, allows a new point of view of Chew's and Ruppert's algorithms using protecting balls. The second one, the protecting ball theorem, allows an incremental point insertion in the simplicial complex conforming an input manifold complex.

The extension of Ruppert's algorithm to deal with curved manifold complexes inputs lacks theoretical guarantees on grading and size optimality.

The main problem in the Delaunay refinement in dimensions greater than two is handling small input angles. The concentric shells works well for small angles between input segments in any dimension, but for small dihedral angles between higher dimensional pieces there is no good approach. The best treatment of this kind of small input angle up to now is to pre-segment with protecting weighted points. However pre-segmentation constrains the output mesh refinement.

The extended Ruppert's algorithm protects any n-simplex in $\mathbb{R}^d$, $n \leq d-1$, with an empty circumscribed ball. This protection can be relaxed to $n \leq d-2$ by using constrained Delaunay simplicial complexes as in Chew's second algorithm, which reduces the mesh cardinality.

Chapter 4

Implementation

An implementation of the Delaunay refinement algorithm for curved complexes was coded in C++ using the standard template library (STL). A description of its main data structures and algorithms is given next, as well as some meshing and timing results.

4.1 Data structures

There are many data structures available in the literature designed to represent geometric objects, including the winged-edge [Baumgart, 1975], the double connected edge list (DCEL for short) [Muller and Preparata, 1978], the half-edge [Weiler, 1985] (awkwardly, now it is more commonly known as DCEL), and the quad-edge [Guibas and Stolfi, 198 data structures, as well as their extensions to higher dimensions and curved pieces. The former three data structures fully support facet holes and orientable manifolds, and the quad-edge fully supports any manifold but not facet holes (refer to Kettner's paper [Kettner, 1999] for a more complete comparison).

A data structure derived almost straightforwardly from a manifold complex is implemented to represent it. This data structure is simple and allows an efficient access by the Delaunay refinement algorithm, but do not consider any modeling requirement. To represent a simplicial complex, it is used an extension of the triangle-based data structure [Shewchuk, 1997], which considers the features of a unique type of element: the simplex. The main peculiarities of the implemented data structure lies in the signaling, which allows a single neighbor pointer to indicate either a co-dimensional or a lower dimensional simplex, and that allows junction simplices to be given explicitly in the data structure. Both data structures are given in details next.

4.1.1 Data structures for manifold complexes

The data structure for manifold complexes stores the input domain that will be accessed throughout the mesh generation. These data structures must answer the queries from the mesh generator as fast as possible. Memory usage is not a main concern on these data structures.

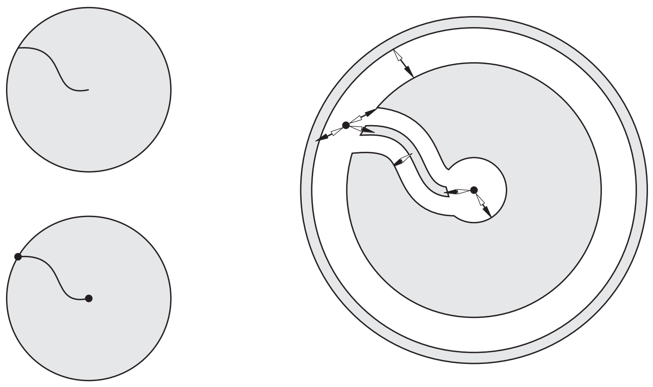

Figure 4.1: Data structure (right) for the manifold complex (bottom-left) for the input domain (top-left).

All pieces are stored in the same container and the position on it identifies each one. Each piece contains the indices of all pieces connected to it, as shown in Figure 4.1. This connectivity information is separated in containers according to the dimensionality of the neighbor. By the definition of manifold complex, co-dimensional pieces cannot access each other directly. They must use lower dimensional pieces or higher dimensional ones for this end. A joint n-piece in $\mathbb{R}^k$, $n \leq k - 2$, has more than two $(n + 1)$-pieces as neighbors.

Each n-piece in $\mathbb{R}^k$ where $n < k$ contains a representation of its member points that will be used to answer queries from the mesh generator. For $n = k$, this representation can be simplified to a single point inside the k-piece, which will be used to label simplices by a flood bounded by lower dimensional simplices. Holes in the domain do not need to be represented, as well as the external set (composed by all the neighborhood of infinite), since any simplex not flood will be removed. Actually, any (≥ 2)-piece also keeps a point on it, so that trimmed places can be correctly removed. Notice that each piece instantiates a topological space, so that its boundaries may let to be represented by lower dimensional neighbors.

The parametric representation is more suitable to represent pieces than the implicit one, due to easier free modeling, and topological or differential features identification. Furthermore, the map to the parametric space is very convenient in many situations, like storing the coordinates of the points on lower dimensional pieces. However additional representations can be held within the same piece, as long as they speed up answering queries.

The aforementioned data structure for manifold complexes is basic and complete for the proposed Delaunay refinement algorithm for curved complexes. More complex data structures may be required by a modeler.

4.1.2 Data structures for simplicial complexes

The massive data of a mesh generator comes from the simplicial complex. The number of records not only increases storage, but also the number of reads and writes to a slower mass storage memory (other than cache). Some compact representations, not in the current implementation, are designed for storing on disk [Szymczak and Rossignac, 2000] with highest compressing rates (*e.g.* 1byte/tetrahedron), or on main memory [Blandford et al., 2003] with lower compressing rates (*e.g.* 7.5bytes/tetrahedron but allowing dynamic queries and updates.

A specialized and generalized version of the triangle-based data structure is implemented, as shown in 4.2, and it will be depicted next.

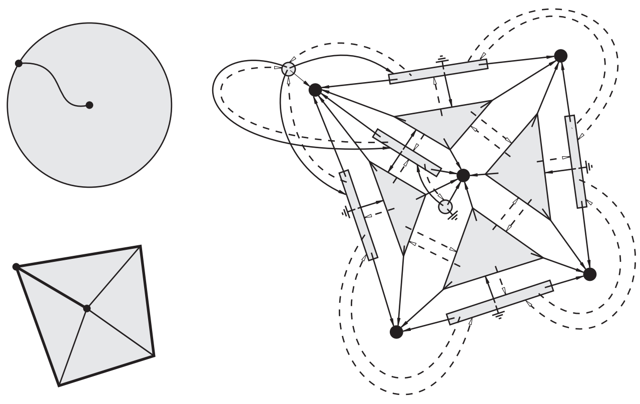

Figure 4.2: Data structure (right) for the simplicial complex (bottom-left) homeomorphic to the manifold complex (top-left).

An n-simplex is only stored if it is part of a homeomorphic representation of an input n-piece (see Figure 4.2). Each n-simplex in the data structure contains the index of the n-piece it represents, called label index.

The connectivity data is stored apart from the vertices coordinates. Each n-simplex in the data structure contains the indices of its $n+1$ vertices. These indices are kept as integer numbers, instead of pointers, to ease comprehension, debug and parallelization of the code and signalization and compression of the data, without any relevant performance loss. The orientation of the simplices are also represented in the connectivity, so that the indices ordering matters. The adopted n-simplex in $\mathbb{R}^n$ numbering convention is shown in Figure 4.3, where the determinant of the vectors from the local vertex 0 to the other $1, ..., n$ vertices is positive.

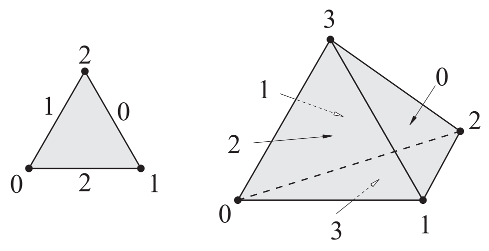

Figure 4.3: Numbering scheme convention for an n-simplex in $\mathbb{R}^n$.

The $n+1$ neighbors indices of each n-simplex are also stored to speed up and ease updates of the data structure during the Delaunay refinement. The i^{th} neighbor of an n-simplex shares the $(n-1)$-facet opposite to the i^{th} node of the simplex, as in the local numbering of facets shown in Figure 4.3. Notice that the local node, facet and neighbor numbering are all the same for any n-simplex. When the neighbor is a lower dimensional simplex, the **down** bit of the neighbor index is set to one. Because the data structure is representing a simplicial complex homeomorphic to a manifold complex, every n-simplex always has $n + 1$ neighbors except 0-simplices that have none.

Each $(n-1)$-simplex in the data structure contains indices for each of the two n-simplices sharing it, called parent indices. When a parent is not part of the complex, its respective index is set to **null**. If an n-simplex in a simplicial k-complex, $n \leq k-2$, is part of a junction n-piece, then it has more than two parents. In this case, the first index contains one of the parents and the second parent index is set to the joint index with the **joint** bit set to one. The other incident parents must be found through the simplicial complex connectivity.

In the data structure just described, an n-simplex can only access $(n + 1)$ or $(n - 1)$-simplices. To not break the data structure when an n-simplex is connected to an $(n+2)$-simplex but not to any $(n+1)$-simplex, a single wing among the complete set of wings (see Figure 4.4) is kept to allow n-simplices and $(n+2)$-simplices access each other through them.

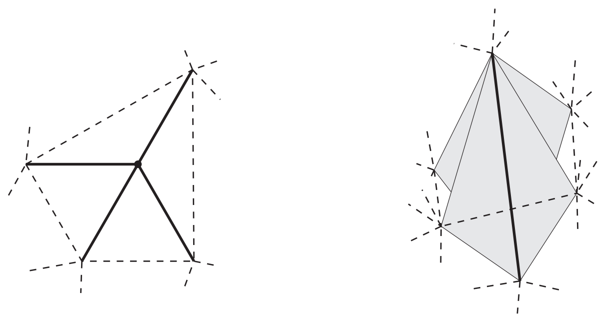

Figure 4.4: Complete set of wings for an orphan 0-simplex (left) and an orphan 1-simplex (right).

The overall number c of indices required for the data structure to store c_n n-simplices representing a simplicial k-complex, $n \leq k$, is

$$c = 4c_0 + [2(k+1)+1]c_k + \sum_{i=1}^{k-1}[2(i+1)+3]c_i, \quad k \geq 1 \tag{4.1}$$

Updating simplicial complexes

The Delaunay refinement algorithm is intrinsically incremental, so that the data structure must efficiently allow insertion of one vertex at a time where it is needed. The implemented incremental update was an extension of the Bowyer-Watson algorithm [Bowyer, 1981, Watson, 1981] to conforming constrained Delaunay simplicial complexes, which also easily generalizes to higher dimensions. The fundamental idea of this algorithm is to remove simplices whose circumballs encloses the new vertex, creating what is called cavity, and then connect the new vertex to the facets of the cavity boundary, which is called horizon. Notice that once the cavity simplices are marked, the remaining task is exclusively a connectivity update (see Figure 4.5).

In a first step, a complete set of facets of the cavity simplices is created, and each facet in it is classified as horizon, twin or encroached, as shown in Figure 4.5. A twin facet has both parents as cavity simplices, whereas a horizon facet has a single one. An encroached facet is a lower dimensional facet that is encroached by the new vertex (notice that if no encroachment occurs, it is a horizon facet). The facet is easily classified by checking its parents, which in turn are set from the neighbor indices of the cavity simplices.

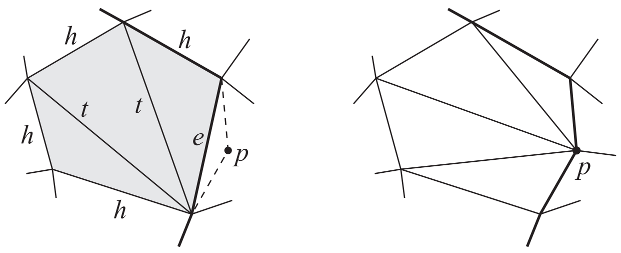

Figure 4.5: Complete set of facets of the cavity and their classification as horizon h, twin t or encroached e (left), and the resulting mesh after connection of the new vertex p to the horizon facets (right).

After facet classification, the connectivity (including ordering) of the resulting mesh follows straightforwardly by connecting the new vertex to the horizon facets. Unfortunately, the neighborhood of the new elements does not. The neighbor of each $(n-1)$-facet v_i opposite to a node i of a horizon n-facet f (see Figure 4.6) is set by searching through elements e_j sharing v_i, until another horizon facet or an encroached facet is met. This kind of search will be henceforth called fan-search.

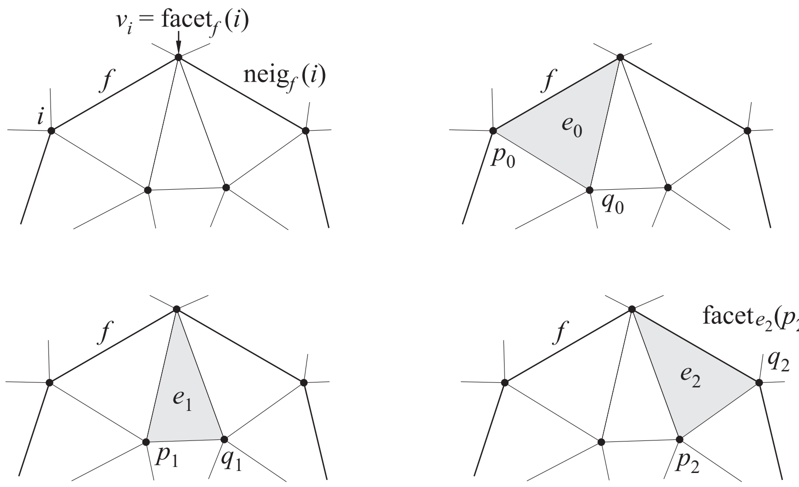

Figure 4.6: Graphical representation of the fan-search used to set the neighborhood of the horizon facets.

In the fan-search, the local index p_0 in e_0 is initialized to i, and q_0 to the vertex opposite to f, as shown in Figure 4.6. Then, while the neighbor opposite to p_j

belongs to the cavity, e_{j+1} is set to the neighbor opposite to p_j, p_{j+1} is set to the position of the node q_j in e_{j+1} connectivity, and q_{j+1} is set to the position of e_j in e_{j+1} neighborhood.

The fan-search is also straightforwardly useful to search for parents of a joint simplex, or the complete set of wings of a simplex.

If the neighbor of a horizon facet is also a horizon facet, then the respective elements created with the new vertex will be also neighbors. If the neighbor is an encroached facet instead (see Figure 4.5), then the neighbor is a lower dimensional simplex. In this case, the neighbor of the lower dimensional simplex opposite to q_j is stored in order to restore the correct lower dimensional facet index created in the recursive simplicial complex update.

The update scheme preserves the orientation of co-dimensional simplices representing the same co-dimensional piece. However, if co-dimensional simplices are part of a non-orientable piece, and indeed both orientations are observed on the subset being updated, the update will still work because the numbering scheme is local for each simplex, but the orientation of new simplices would obviously be undefined or, more precisely, each new simplex will have the same orientation of the parent of its respective horizon facet.

When inserting a new vertex v on a curved piece, one of the parents of the simplex $\mathbb{S}$ that represents the curved piece may not contain the new vertex v in its circumscribed ball, as shown in Figure 4.7. In this case, the simplex $\mathbb{S}$ must be considered a horizon facet.

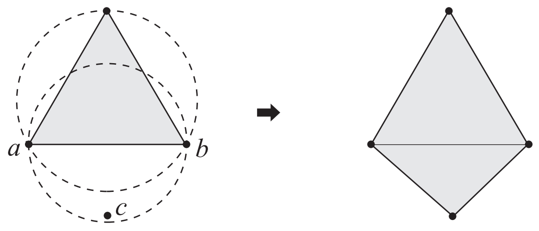

Figure 4.7: During split of the line segment ab at c the circumcircle of the incident triangle to ab does not enclose c, but the diametral circle of ab does.

4.2 Memory management

The label, nodal, neighbor and parent indices are stored in separated containers indexed by the simplex dimension. This pool of data allows block memory allocations and is suitable to separately manage the containers' data (*e.g.* rendering or discharging).

A design pattern used to preserve the data index after updates on data containers is shown in Figure 4.8. Every element deleted from the data container has its respective index pushed into a recycle bin stack. New data is appended to the container end if the recycle bin is empty, otherwise it is placed on the position indicated by the top index popped from the stack. Because the number of delete elements removed from the simplicial complex during updates is bounded in practice, not many stack memory allocations are requested. Only just after facet recovery, a larger amount of elements are removed from the simplicial complex. Hence, every element in the container has its index (respective position in the container) preserved after updates, the elements are kept contiguous in memory, and memory can be allocated in large blocks instead of element-wise small pieces. Nonetheless, the container will eventually be copied to a larger memory free area.

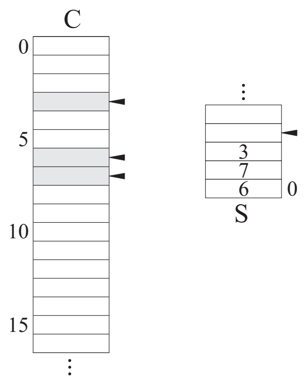

Figure 4.8: Recycle bin design pattern for the memory management of a data container C by a recycle bin stack S.

4.3 Delaunay refinement

4.3.1 Facet recovery

The facet recovery step creates an initial simplicial k-complex homeomorphic to the input manifold k-complex. The facet recovery starts from 0-pieces and goes to k-pieces. An initial mesh is created for each piece. Before proceeding to the next dimension of pieces, the initial meshes are refined in order to guarantee an empty circumscribed ball for each (≥ 1)-simplex representing a (≥ 1)-piece. The 0-pieces are simply converted into vertices in the simplicial complex.

The inicial mesh for a piece is a Delaunay triangulation over its respective manifold space. An initial mesh in the k-piece domain is created (usually the rectangular domain $[0, 1]^k$), and incident vertices to the piece, already part of the simplicial complex, are inserted one by one in the mesh. Since the boundary of the piece becomes well defined after all incident vertices are inserted, the extra outer simplices are removed by flooding from the respective seed of the piece. The update algorithm of the simplicial complex used in the refinement is the same as the one used in the facet recovery.

4.3.2 Refinement

The refinement is done in two steps using an extended Bowyer-Watson algorithm [Bowyer, 1981, Watson, 1981]. First, lists of simplices encroached upon by the insertion point p are created, depicted in Algorithm 1, which define a cavity. Then the cavity is remeshed by a connectivity update, depicted in Algorithm 2.

Algorithm 1 Encroachment lists.

Input: a manifold complex $\mathcal{M}$, a simplicial k-complex $\mathcal{D}$ homeomorphic to $\mathcal{M}$, a point p and a seed n-simplex in list L_n encroached upon by p.
Output: lists L_i, $i = n_{\min}, ..., n_{\max}$, of encroached i-simplices.

```
01. $going.down \leftarrow$ true
02. $n_{\min} \leftarrow n$
03. $n_{\max} \leftarrow n$
04. loop
05.   $any.encroachment \leftarrow$ false
06.   if $n > 1$
07.     $i \leftarrow 0$
08.     while $i <$ size of $L_n$
09.       for every neighbor $v_j$ of $L_{n,i}$
10.         if $v_j$ is not **null**
11.           if $v_j$ is a lower dimensional simplex
12.             if $going.down$ and $p$ encroaches upon $v_j$
13.               $any.encroachment \leftarrow$ true
14.               $p \leftarrow$ split point on the $(n-1)$-piece represented by $v_j$
15.               push $v_j$ into $L_{n-1}$
16.               clear $L_n$
17.               $i \leftarrow -1$
18.               break
19.           else
20.             if $p$ encroaches upon $v_j$ and $v_j$ is not in $L_n$
21.               push $v_j$ into $L_n$
22.       $i \leftarrow i + 1$
23.   if $going.down$ and $any.encroachment$
24.     $n \leftarrow n - 1$
25.     $n_{\min} \leftarrow n_{\min} - 1$
26.   else
27.     $going.down \leftarrow$ false
28.     if $n = k$
29.       break
30.     push all non-null parents of $L_n$ into $L_{n+1}$
31.     if $L_{n+1}$ is empty
32.       break
33.     $n \leftarrow n + 1$
34.     $n_{\max} \leftarrow n$
```

Many refinement criteria may be used to select an insertion point. The implemented ones were:

Algorithm 2 Cavity remesh.

Input: a simplicial k-complex $\mathcal{D}$, a point p and lists L_i, $i = e, ..., k$, of i-simplices encroached upon by p.
Output: a simplicial k-complex $\mathcal{D}$ containing p.

```
01. $\mathcal{H} \leftarrow$ complex set of facets of $n$-simplices in $L_n$
02. classify elements of $\mathcal{H}$ as twin, encroached or horizon facets
03. allocate space for $n$-simplices in $\mathcal{D}$, one for each horizon facet in $\mathcal{H}$
04. search for neighbors of horizon facets in $\mathcal{H}$
05. if $e < n$
06.     recursive call of cavity remesh on $L_{n-1}$
07.     relink encroached neighbors of horizon facets to the new $(n-1)$-simplices
08. set up new $n$-simplices by connecting $p$ to the horizon facets in $\mathcal{H}$
```

- Minimum ratio ℓ/R between the minimum edge length ℓ and the circumradius R of an n-simplex. This criterion is natural for the Delaunay refinement and it is related to the quality of an n-simplex, despite this quality measure fails for $n \geq 3$.

- Maximum edge length of an n-simplex. This criterion controls the mesh size or uniformity of the simplicial complex.

- Maximum distance between the circumcenter on the simplex flat and the circumcenter on the piece. This criterion controls the accuracy of the linear approximation of the simplicial complex relative to the input manifold complex.

The new elements are pushed or not into priority queues according to their refinement criteria measures after every update of the simplicial complex. The priority is higher for elements whose measure is farther from the threshold. The refinement priority usually reduces the final simplicial complex cardinality.

Edges that are incident to the cavity are split up if they are encroached upon by the just inserted vertex. Edges of the simplicial complex are split up according to concentric shells at the input vertices. This was implemented by rounding the subsegment length to the nearest power of two (since in IEEE 754 standard a floating point number is represented in the canonical form $\pm 1.[\text{significand bits}] \times 2^{\pm \text{exponent bits}}$, this may be done by zeroing the significand bits and adding one to the exponent if the first significand bit is one).

4.4 Predicates

4.4.1 Encroached lenses

The encroached lenses predicate checks whether a point lies inside an n-lens. An n-lens is composed by joining the two external parts of an n-ball symmetrically sliced by two parallel hyperplanes equidistant to the origin. The intersection of an n-lens and the k-flat $\mathbb{F}$ that contains its forming k-simplex $\mathbb{S}$, is the circumscribed ball of $\mathbb{S}$ in $\mathbb{F}$. When $k \leq n-2$, the $(k+1)$-lens of $\mathbb{S}$ is rotated around $\mathbb{F}$ to build the n-lens. All this symmetry allows writing the encroached lenses predicate based on the simplex circumcenter c, the circumradius r and the angle θ in which the hypersphere cap meets $\mathbb{F}$. Let p be the query for the lens predicate, $q = p - c$, q_r the component of q on $\mathbb{F}$, q_h the distance from p to $\mathbb{F}$ (see Figure 4.9). The encroached lenses predicate is

$$\begin{gathered}\left(q_h^2 + \frac{r}{\tan\theta}\right)^2 + q_r^2 \leq \frac{r^2}{\sin^2\theta} \\ 2rq_h \leq (r^2 - q^2)\tan\theta\end{gathered} \tag{4.2}$$

To avoid a square root in the predicate (4.2), it can be rewritten as

$$q^2 \leq r^2 \tag{4.3a}$$

$$4r^2 q_h^2 \leq (r^2 - q^2)^2 \tan^2\theta \tag{4.3b}$$

and for 1-simplices the second inequality (4.3b) can be written as

$$4[q^2 r^2 - (\vec{q}\cdot\vec{r})^2] \leq (q^2 - r^2)^2 \tan^2\theta \tag{4.4}$$

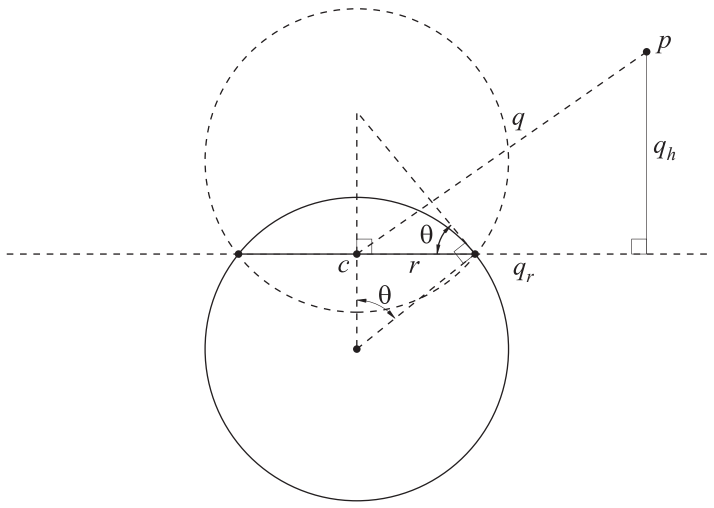

Figure 4.9: Construction of the predicate inside/outside lenses.

Notice that the first inequality (4.3a) is the predicate for minimum circumscribed ball, which is simpler and can be evaluated without q_h and θ.

By the extended Thales theorem, the predicate for a 1-simplex can be simplified to

$$a \cdot b \leq 0 \tag{4.5a}$$

$$\frac{(a\cdot b)^2}{|a|^2\,|b|^2} \geq \cos^2\theta \tag{4.5b}$$

where $a = v_0 - p$, $b = v_1 - p$, and $v_0 v_1$ is the 1-simplex. Notice that the first inequality (4.5a) is the predicate for diametral ball, which is simpler and can be evaluated without θ.

4.4.2 Robust predicates

A common geometric predicate tells which side a point v_n lies relatively to an oriented $(n-1)$-hyperplane in $\mathbb{R}^n$ that goes through points v_0, ..., v_{n-1}. It can be written as the sign of the determinant

$$A_n = \begin{vmatrix} (v_1 - v_0)^T \\ \vdots \\ (v_n - v_0)^T \end{vmatrix} \tag{4.6}$$

The update of the simplicial complex is based on whether a point v_{n+1} lies inside or outside the circumscribed n-ball of an n-simplex in $\mathbb{R}^n$ with vertices $v_0, ..., v_n$. It can be written as the sign of the determinant

$$B_n = \begin{vmatrix} (v_1 - v_0)^T & \|v_1 - v_0\|^2 \\ \vdots & \\ (v_n - v_0)^T & \|v_n - v_0\|^2 \\ (v_{n+1} - v_0)^T & \|v_{n+1} - v_0\|^2 \end{vmatrix} \tag{4.7}$$

If this predicate is not evaluated correctly, then the Delaunay property may not hold and the refinement will fail. Unfortunately, roundoff errors during computations let such errors occur. The exact arithmetic adopted here is the same used by Shewchuk [Shewchuk, 1997].

A floating-point number is represented as $\pm\text{significand} \times 2^{\pm\text{exponent}}$ in processors, where the significand is a p-bit binary number. The p bits of the significand may be not enough to store the result of an exact arithmetic operation. Hence each number is expressed as an arbitrary precision expansion $e = e_1 + e_2 + ...$ [Priest, 1991] where the components e_i are floating-point numbers with a p-bit significand. The components are also nonoverlapping (*e.g.* 1100 and 11 are nonoverlapping, whereas 1101 and 10 overlap) and ordered by increasing magnitude (x_1 is the smallest), so that the largest component is an approximation of the expansion value and has its sign. Exact arithmetic operations over component expansions are given in Shewchuk's thesis [Shewchuk, 1997].

The p-bit addition $\oplus$, subtraction $\ominus$, and multiplication $\otimes$ are assumed to be performed with exact rounding (*e.g.* as in IEEE 754 standard). This means that the nearest p-bit floating-point value $\tilde{x}$ to the exact result x is always produced. Under this assumption, the roundoff error of the result $\tilde{x}$ of any p-bit operation is at most $\epsilon\,|\tilde{x}|$ and $\epsilon\,|x|$, where ϵ is the machine epsilon and x is the exact result.

The determinant A_1 is

$$\begin{aligned} A_1 &= v_{1x} - v_{0x} \\ &= v_{1x} \ominus v_{0x} \pm \epsilon\,|v_{1x} \ominus v_{0x}| \\ &= \tilde{A}_1 \pm \epsilon|\tilde{A}_1| \end{aligned} \tag{4.8}$$

so that the sign of $\tilde{A}_1$ is correct whenever

$$\begin{aligned} |\tilde{A}_1| &\geq \epsilon|v_{1x} \ominus v_{0x}| \\ &= \epsilon(1+\epsilon) \otimes |v_{1x} \ominus v_{0x}| \\ &= (\epsilon + \epsilon^2) \otimes \alpha_1 \end{aligned} \tag{4.9}$$

which is always true.

The effect of the term $\epsilon|\tilde{A}_1|$ can be minimized by concluding that the sign is correct whenever

$$\begin{aligned} |\tilde{A}_1| &\geq \epsilon|\tilde{A}_1| \\ &= \epsilon^2|\tilde{A}_1| \\ &= \epsilon^3|\tilde{A}_1| \\ &= ... \end{aligned} \tag{4.10}$$

The determinant A_2 is

$$\begin{aligned} A_2 &= (v_{1x} - v_{0x})(v_{2y} - v_{0y}) - (v_{1y} - v_{0y})(v_{2x} - v_{0x}) \\ &= t_1 t_2 - t_3 t_4 \\ &= (\tilde{t}_1 \pm \epsilon|\tilde{t}_2|)(\tilde{t}_2 \pm \epsilon|\tilde{t}_2|) - (\tilde{t}_3 \pm \epsilon|\tilde{t}_3|)(\tilde{t}_4 \pm \epsilon|\tilde{t}_4|) \\ &= \tilde{t}_1\tilde{t}_2 \pm (2\epsilon + \epsilon^2)|\tilde{t}_1\tilde{t}_2| - [\tilde{t}_3\tilde{t}_4 \pm (2\epsilon+\epsilon^2)|\tilde{t}_3\tilde{t}_4|] \\ &= t_5 \pm (2\epsilon+\epsilon^2)|t_5| - [t_6 \pm (2\epsilon+\epsilon^2)|t_6|] \\ &= \tilde{t}_5 \pm \epsilon|\tilde{t}_5| \pm (2\epsilon+\epsilon^2)(|\tilde{t}_5| \pm \epsilon|\tilde{t}_5|) - [\tilde{t}_6 \pm \epsilon|\tilde{t}_6| \pm (2\epsilon+\epsilon^2)(|\tilde{t}_6| \pm \epsilon|\tilde{t}_6|)] \\ &= \tilde{t}_5 \pm (3\epsilon + 3\epsilon^2 + \epsilon^3)|\tilde{t}_5| - [\tilde{t}_6 \pm (3\epsilon+3\epsilon^2+\epsilon^3)|\tilde{t}_6|] \\ &= \tilde{t}_5 - \tilde{t}_6 \pm (3\epsilon+3\epsilon^2+\epsilon^3)(|\tilde{t}_5| + |\tilde{t}_6|) \\ &= \tilde{A}_2 \pm \epsilon|\tilde{A}_2| \pm (3\epsilon+3\epsilon^2+\epsilon^3)(|\tilde{t}_5| + |\tilde{t}_6|) \\ &= \tilde{A}_2 \pm (4\epsilon+3\epsilon^2+\epsilon^3)(|\tilde{t}_5| + |\tilde{t}_6|) \\ &= \tilde{A}_2 \pm (4\epsilon+3\epsilon^2+\epsilon^3)(1+\epsilon)^2 \otimes (|\tilde{t}_5| \oplus |\tilde{t}_6|) \\ &= \tilde{A}_2 \pm (4\epsilon+11\epsilon^2+11\epsilon^3+5\epsilon^4+\epsilon^5) \otimes (|\tilde{t}_5| \oplus |\tilde{t}_6|) \end{aligned} \tag{4.11}$$

so that the sign of $\tilde{A}_2$ is correct whenever

$$\begin{aligned} |\tilde{A}_2| &\geq (4\epsilon + 16\epsilon^2) \otimes [|v_{1x} \ominus v_{0x}| \otimes |v_{2y} \ominus v_{0y}| \oplus |v_{1y} \ominus v_{0y}| \otimes |v_{2x} \ominus v_{0x}|] \\ &= (4\epsilon + 16\epsilon^2) \otimes \alpha_2 \end{aligned} \tag{4.12}$$

where the coefficient was rounded up to the next p-bit floating-point value. Since the coefficient must be a p-bit value, it is useless to track terms smaller than ϵ^3 as long as they do not exceed ϵ^3. Notice that the order in which operations are performed matters when computing $\tilde{A}_2$.

The effect of the term $\epsilon|\tilde{A}_2|$ can be minimized by concluding that the sign is correct whenever

$$\begin{aligned}
|\tilde{A}_2| &\geq (3\epsilon + 3\epsilon^2 + \epsilon^3)(|\tilde{t}_5| + |\tilde{t}_6|) + \epsilon|\tilde{A}_2| \\
&= (3\epsilon + 6\epsilon^2 + 4\epsilon^3 + \epsilon^4)(|\tilde{t}_5| + |\tilde{t}_6|) + \epsilon^2|\tilde{A}_2| \\
&= (3\epsilon + 6\epsilon^2 + 7\epsilon^3 + 7\epsilon^4 + 4\epsilon^5 + \epsilon^6)(|\tilde{t}_5| + |\tilde{t}_6|) + \epsilon^3|\tilde{A}_2| \\
&= (3\epsilon + 6\epsilon^2 + 8\epsilon^3)(|\tilde{t}_5| + |\tilde{t}_6|) \\
&= (3\epsilon + 6\epsilon^2 + 8\epsilon^3)(1+\epsilon)^2 \otimes (|\tilde{t}_5| \oplus |\tilde{t}_6|) \\
&= (3\epsilon + 12\epsilon^2 + 23\epsilon^3 + 22\epsilon^4 + 8\epsilon^5) \otimes (|\tilde{t}_5| \oplus |\tilde{t}_6|) \\
&= (3\epsilon + 16\epsilon^2) \otimes \alpha_2
\end{aligned} \tag{4.13}$$

where the coefficient was rounded up to the next p-bit floating-point value.

The error bounds for the determinants $|\tilde{A}_1|$ and $|\tilde{A}_2|$ are given by a coefficient times the permanent α_n of absolute value of the matrix elements. This generalizes straightforwardly for higher dimensions. The coefficient can be deduced from the expansion by minors, so that each additional dimension includes 1 multiplication and n subtractions roundoff errors. Hence

$$\begin{aligned}
a_n &= a_{n-1} + n + 1 \\
&= a_{n-2} + (n-1) + n + 2 \\
&= a_{n-3} + (n-2) + (n-1) + n + 3 \\
&= a_1 + \left(\sum_{i=2}^{n} i\right) + n - 1 \\
&= \frac{n}{2}(n+1) + n - 1
\end{aligned} \tag{4.14}$$

whose first numbers are

$$1, 4, 8, 13, 19, 26, 34, 43, 53, 64... \tag{4.15}$$

and the sign of $\tilde{A}_n$ is correct when

$$|\tilde{A}_n| > a_n \epsilon \alpha_n \tag{4.16}$$

The error bound for B_n follows similarly. The expansion by minors for the column of squared norms shows that B_n has the a_n roundoff operations of the minor determinant, plus $n+2$ roundoff operations for the norm, plus $n+1$ roundoff operations for the weighted sum. Hence

$$\begin{aligned}
b_n &= a_n + 2n + 3 \\
&= \frac{n}{2}(n+1) + 3n + 2
\end{aligned} \tag{4.17}$$

whose first numbers are

$$6, 11, 17, 24, 32, 41, 51, 62, 74, 87... \tag{4.18}$$

and the sign of $\tilde{B}_n$ is correct when

$$|\tilde{B}_n| > b_n \epsilon \beta_n \tag{4.19}$$

where β_n is the permanent of absolute value of the matrix elements.

The error bounds (4.10) and (4.13) (first given by Shewchuk [Shewchuk, 1997]) are tighter than the generalized equation (4.17). The coefficients are valid as long as the higher order terms do not exceed ϵ, which fortunately occurs only in dimensions higher than the ones where meshing is still practical ($n \sim 5$). Gaussian elimination would speed up very much the predicates and improve the error bounds because of its symmetry. However it includes divisions in its computation, which cannot be evaluated exactly. A division-less computation would overflow or underflow very easily.

The exact arithmetic is used only when the error bounds are not verified in the approximate computation, which is a subset of the exact computation when using component expansion. Shewchuk [Shewchuk, 1997] gives error bounds for some other intermediate results.

4.5 Projections onto pieces

Throughout mesh generation, the Delaunay refinement queries a d-piece to find the point on it that is equidistant to $d+1$ points on it, as shown in Figure 4.10.

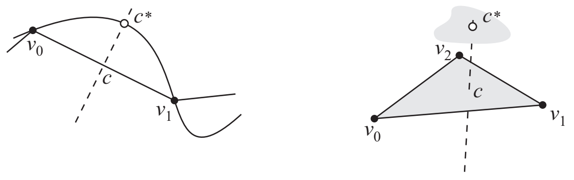

Figure 4.10: Graphical representation of the projection onto a curve (left) and onto a surface (right).

Let $c(t) : [0,1]^d \mapsto \mathbb{R}^d$ be the ortogonal projection of an affinely invariant parametrization of a d-piece (*e.g.* a Bézier, a b-spline or a NURBS depicted in Appendix B) on the d-flat that contains a d-simplex. Let the circumcenter c of the d-simplex be the origin for $c(t)$.

Since the piece parametrization is affinely invariant, the set of parameters t^*, where lies the point c^* equidistant to the vertices of the simplex, is the same where $c(t^*) = 0$. Define the function

$$f(t) = \|c(t)\|^2 \tag{4.20}$$

which contains a unique minimum whenever the projection $c(t)$ is a bijection (Figure 4.11).

Let $c'_i(t) : [0,1]^d \mapsto \mathbb{R}^d$ denote the partial derivative $\partial c(t)/\partial t_i$, $i = 0, 1, ..., d-1$, and $c'(t) : [0,1]^d \mapsto \mathbb{R}^{d\times d}$ be the matrix containing $c'_i(t)$ as rows. Then let $c_k = c(t_k)$ and $c'_k = c'(t_k)$ for short. (Notice that $t_k \in \mathbb{R}^d$ denotes an instance of parameters,

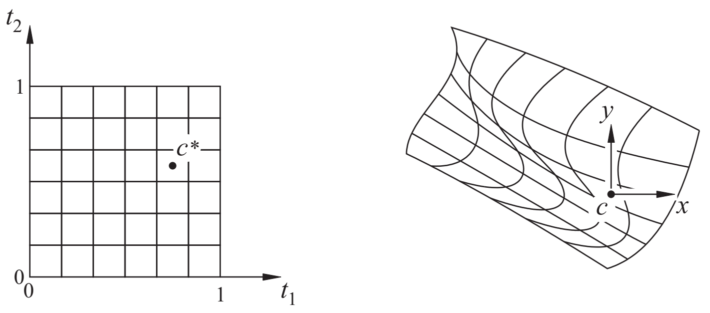

Figure 4.11: Graphical representation of the parametric space (left) and the respective projection onto the 2-flat containing a triangle (right).

and not the k^{th} parameter, and so is c_k and c'_k. This is quite dubious, but eases notation.) The derivative (gradient) $f'(t)$ at t_k of the function $f(t)$ is given by

$$f'(t_k) = f'_k = 2c'_k c_k \tag{4.21}$$

By the Taylor expansion around t_k

$$f(t) = f_k + {f'_k}^T (t - t_k) + O(t^2) \tag{4.22}$$

we define an algorithm that iteratively sets $t_{k+1} = t_k + \lambda f'_k$ (step towards f'_k) so that the linear approximation of $f(t)$ at t_k vanishes $\tilde{f}(t_{k+1}) = 0$. Hence

$$\begin{aligned} \tilde{f}(t_{k+1}) &= f_k + {f'_k}^T (t_{k+1} - t_k) \\ &= f_k + {f'_k}^T (t_k + \lambda f'_k - t_k) \\ &= f_k + \lambda {f'_k}^T f'_k = 0 \end{aligned} \tag{4.23}$$

so that

$$\begin{aligned} t_{k+1} &= t_k - \frac{f_k}{{f'_k}^T f'_k}\, f'_k \\ &= t_k - \frac{c_k^T c_k}{2(c'_k c_k)^T c'_k c_k}\, c'_k c_k \end{aligned} \tag{4.24}$$

which is a kind of Newton-Raphson iterative update. This algorithm is guaranteed to converge only if $f(t)$ is convex.

4.6 Practical performance

4.6.1 Robust n-dimensional predicates

The predicates of orientation and ball containment were tested for random point sets and point sets near the sign transition. The results in Figure 4.12 show that predicates for $n \gtrsim 5$ are impractical. Surprisingly, for odd dimensions greater than 4, the inexact computation of the orientation predicate is almost always sufficient.

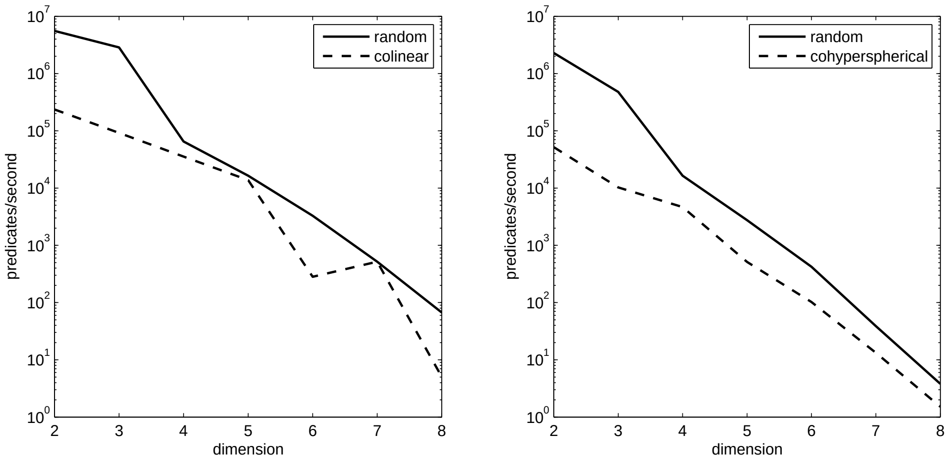

Figure 4.12: Evaluation of robust predicates for simplicial orientation (left) and ball containment (right).

4.6.2 Meshing point clouds

When meshing a point cloud using the Bowyer-Watson algorithm by inserting an arbitrary point at a time, most of the meshing time is spent on locating a simplex that contains the new vertex in its circumscribed ball. This point location is not part of the Delaunay refinement, since the simplex elected to be split up has the new vertex as circumcenter. In order to test the refinement without the point location step, a point insertion in the barycenter of a randomly chosen simplex was adopted since the simplex circumcenter could eventually lie outside the simplicial complex as shown in Figure 4.13. This unconstrained meshing was timed for several dimensions, as shown in Figure 4.14.

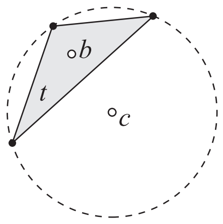

Figure 4.13: The barycenter b of the triangle t always lies inside t, but the circumcenter c may lie outside t.

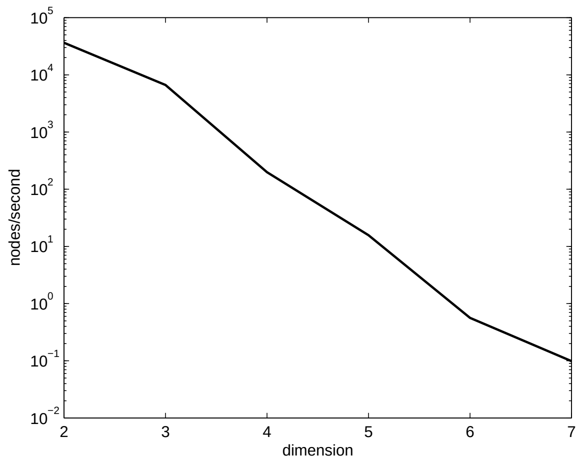

Figure 4.14: Meshing timings for inserting points in the barycenter of randomly chosen simplices.

4.6.3 Meshing small input angles

When an input 2-dimensional geometry contain small angles, the curve segments are split accordingly concentric shells at input vertices in order to guarantee an empty circumcircle for every line segment representing the input. Whenever a segment is split at a small angle, it may force another segment to be split up in order to guarantee an empty circumcircle. Furthermore, two close curved segments are split up such that they do not intersect each other. In Figure 4.15 are meshes for an input geometry where two curved segments meet at a small angle.

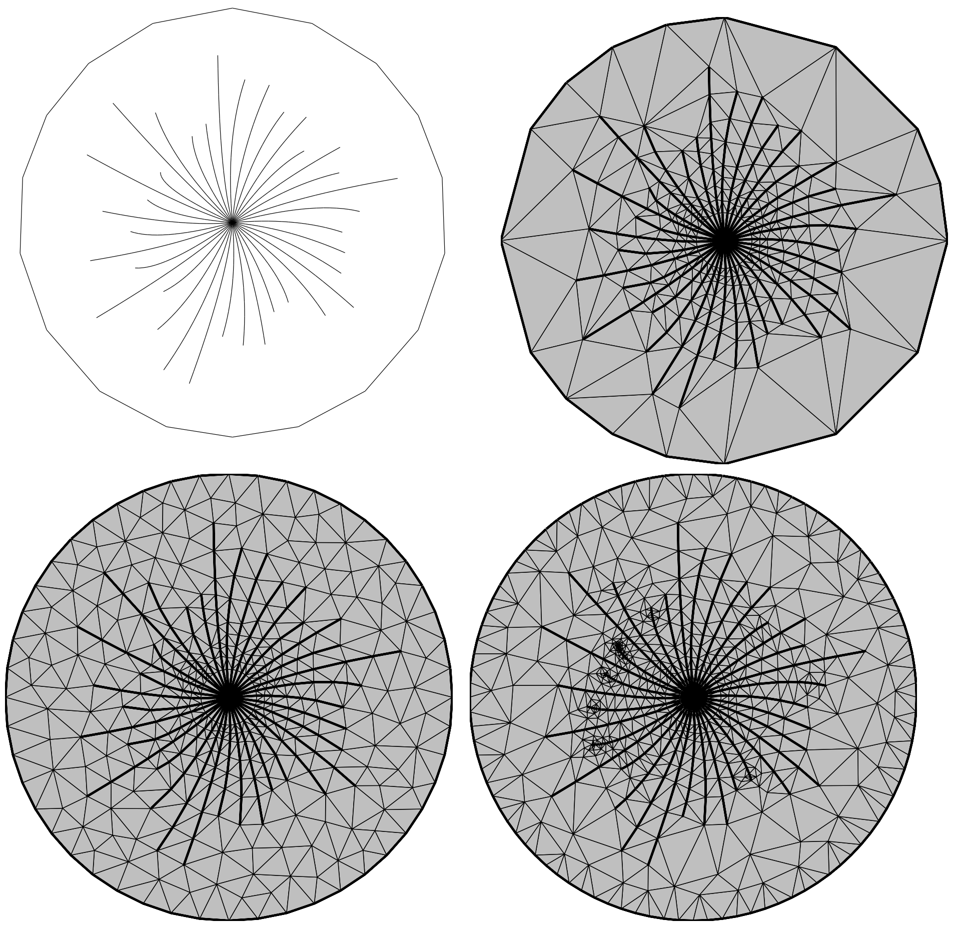

Figure 4.15: Meshing a star (top-left) to a homeomorphic triangulation with 279 nodes (top-right), after uniform refinement with 403 nodes (bottom-left), and after refinement for quality and geometry accuracy with 660 nodes (bottom-right).

4.6.4 Meshing an n-ball

Meshing an n-dimensional ball starting from an n-simplex and electing the circumcenter of a random element for refinement in each iteration (see Figure 4.16) is one of the highest performance contexts for the mesh generator. The respective timings for each dimension is given in Figure 4.17.

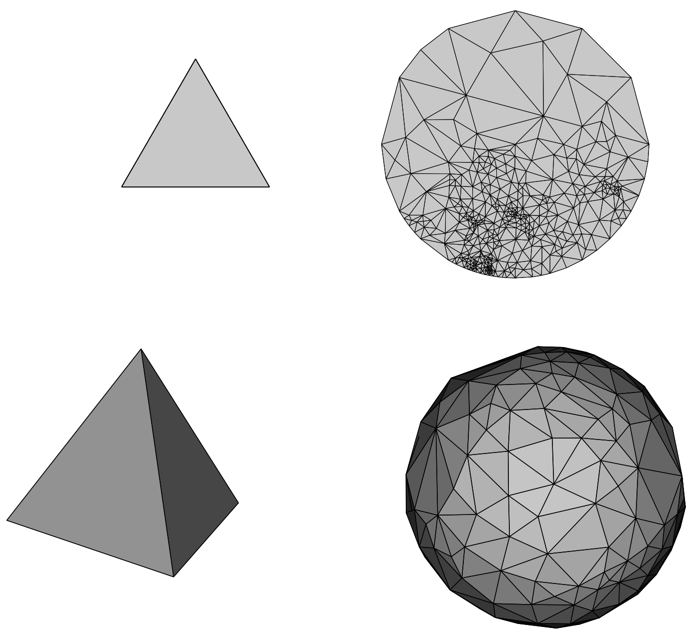
Figure 4.16: Meshing a disk (top) and a ball (bottom) with 400 vertices each, starting from a triangle and a tetrahedron, respectively.

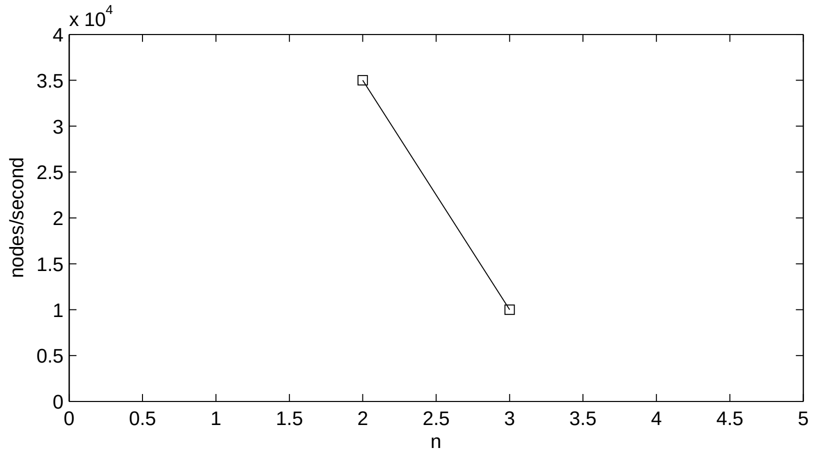

Figure 4.17: Time (in nodes/second) spent on meshing an n-ball.

4.6.5 Meshing a surface

During facet recovery of an input geometry in 3-dimensional space, all piecewise surfaces are meshed separately and then joint into the mesh. In Figure 4.18 are meshes for a single curved 2-piece with boundary refined under many criteria. The surface is a NURBS defined over a rectangular domain $[0, 1]^2$, whose boundary is composed by 4 vertices and 4 curves. The mesh is refined at about 950 nodes/second except when the refinement is for mesh accuracy, which is at about 140 nodes/second.

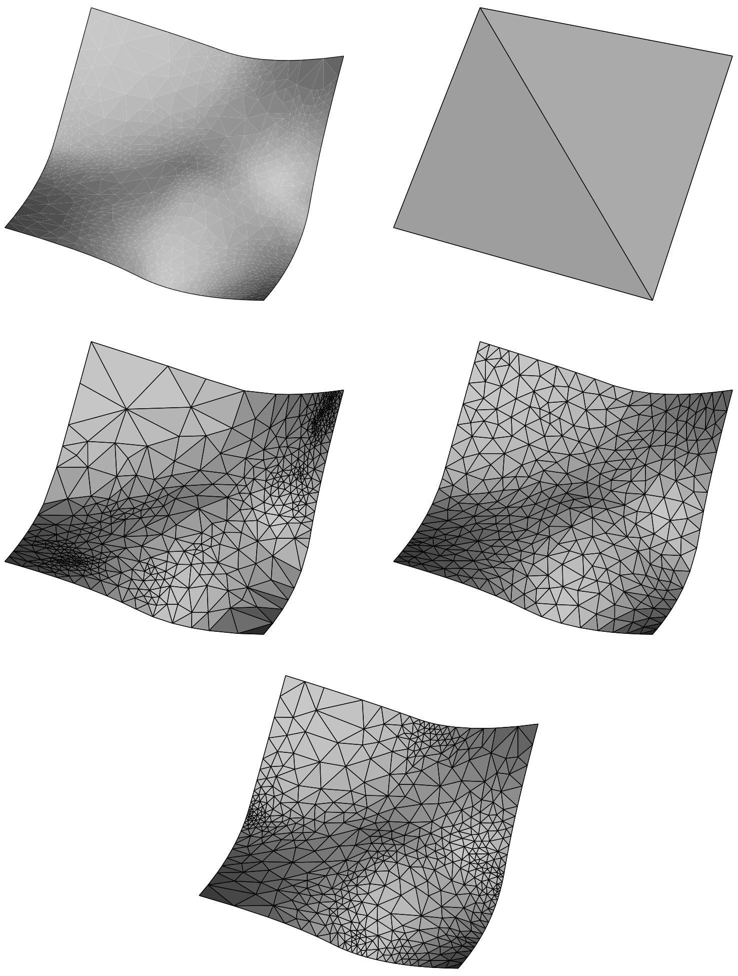

Figure 4.18: Meshes for the input surface (top-left): without any refinement (top-right), randomly refined while keeping $\ell/R \geq 1/\sqrt{2}$ (middle-left), uniformly refined by maximum edge length (middle-right), and refined for accurate input geometry representation while keeping $\ell/R \geq 1/\sqrt{2}$ (bottom).

4.6.6 Meshing a junction

A junction n-piece has more than two parent $(n+1)$-pieces. It is represented as a particular case in the data structure. During the refinement, every parent is triggered as part of the cavity when the respective junction simplex is refined. In Figure 4.19 are meshes for junction line segment containing three curved wings refined under many criteria. The mesh is refined at about $13,000$ nodes/second.

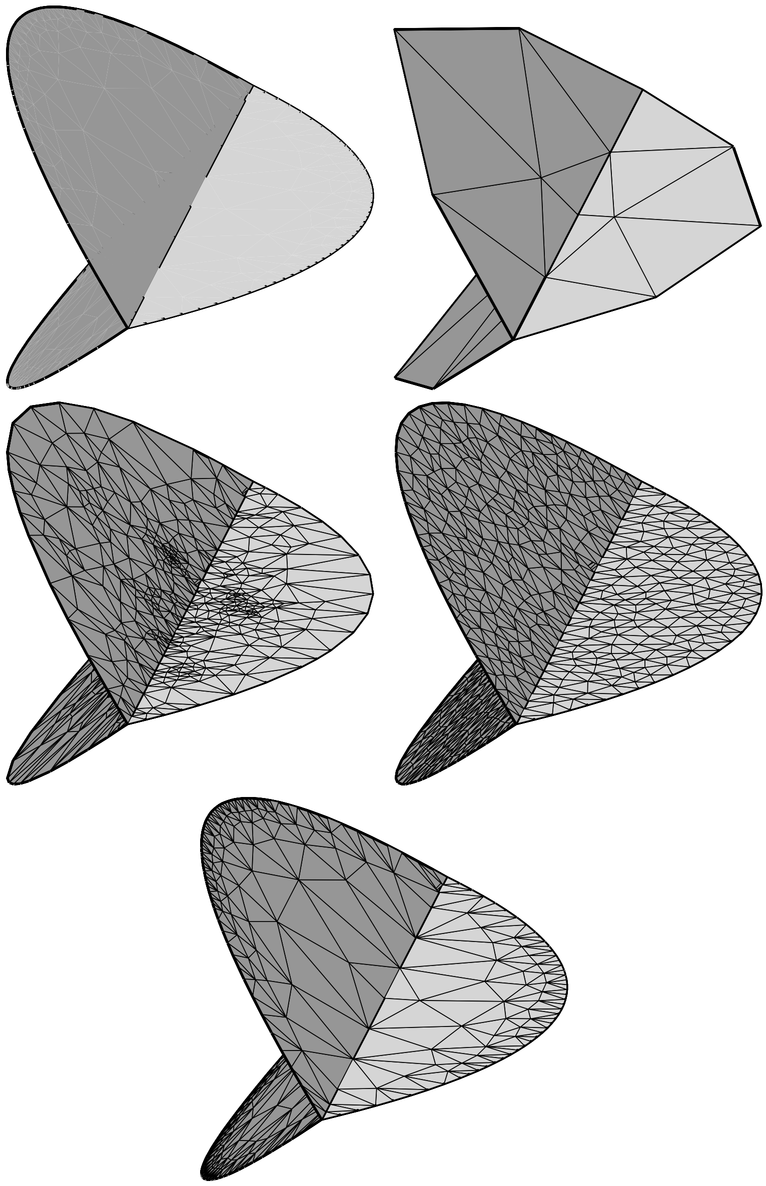

Figure 4.19: Meshes for the input junction (top-left): without any refinement with 17 nodes (top-right), randomly refined while keeping $\ell/R \geq 1/\sqrt{2}$ with 500 nodes (middle-left), uniformly refined by maximum edge length with 646 nodes (middle-right), and refined for accurate input geometry representation while keeping $\ell/R \geq 1/\sqrt{2}$ with 334 nodes (bottom).

4.6.7 Meshing a non-orientable manifold

In a simplicial complex homeomorphic to a non-orientable manifold there is no way to orient the conforming simplices so that all neighbors have the same orientation. The mesh generator must be insensitive to orientation in order to mesh this kind of input. The implemented Delaunay refinement algorithm can mesh non-orientable manifolds and in Figure 4.20 are meshes for a Möbius strip refined under many criteria. The input manifold 2-complex splits up the Möbius strip into eight 2-pieces, so that topology information comes intrinsically.

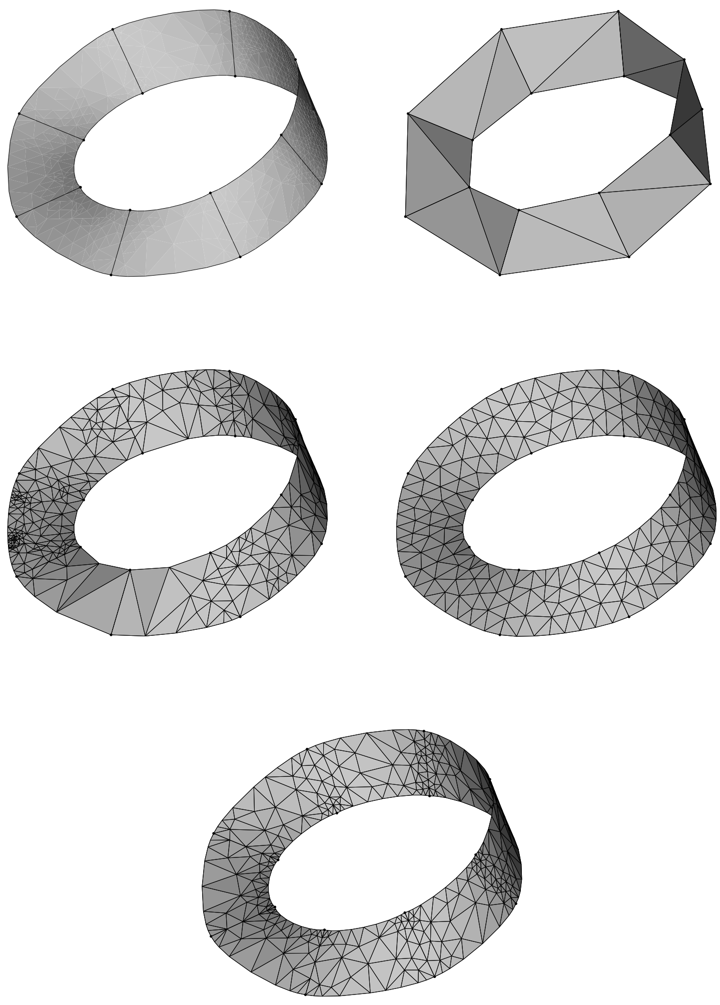

Figure 4.20: Meshes for the input Möbius strip (top-left): without any refinement with 16 nodes (top-right), randomly refined while keeping $\ell/R \geq 1/\sqrt{2}$ with 318 nodes (middle-left), uniformly refined by maximum edge length with 229 nodes (middle-right), and refined for accurate input geometry representation while keeping $\ell/R \geq 1/\sqrt{2}$ with 345 nodes (bottom).

4.6.8 Meshing a piecewise linear complex

In Figure 4.21 are meshes for an input piecewise linear complex with 142 pieces refined under many criteria. The mesh is refined at about 5,000 nodes/second.

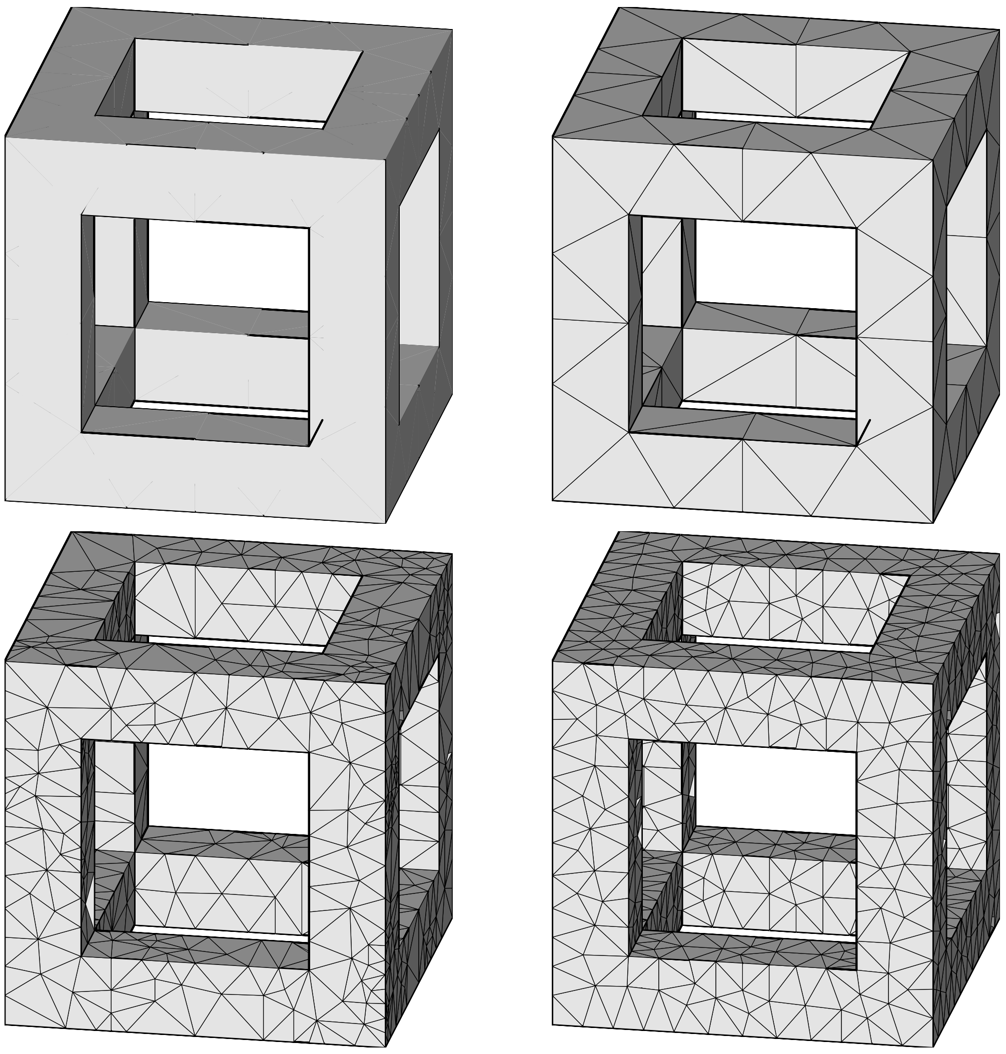

Figure 4.21: Meshes for the input piecewise linear complex (top-left): without any refinement with 112 nodes (top-right), randomly refined while keeping $\ell/R \geq 1/\sqrt{2}$ with 800 nodes (bottom-left), uniformly refined by maximum edge length with 1047 nodes (bottom-right).

4.6.9 Meshing a piecewise curved complex

In Figure 4.22 are meshes for an input piecewise curved complex with 44 pieces refined under many criteria. The mesh is refined at about 400 nodes/second over the surface, and about $5,000$ nodes/second over the 3-dimensional space. A slice of the simplicial 3-complex is shown in Figure 4.23.

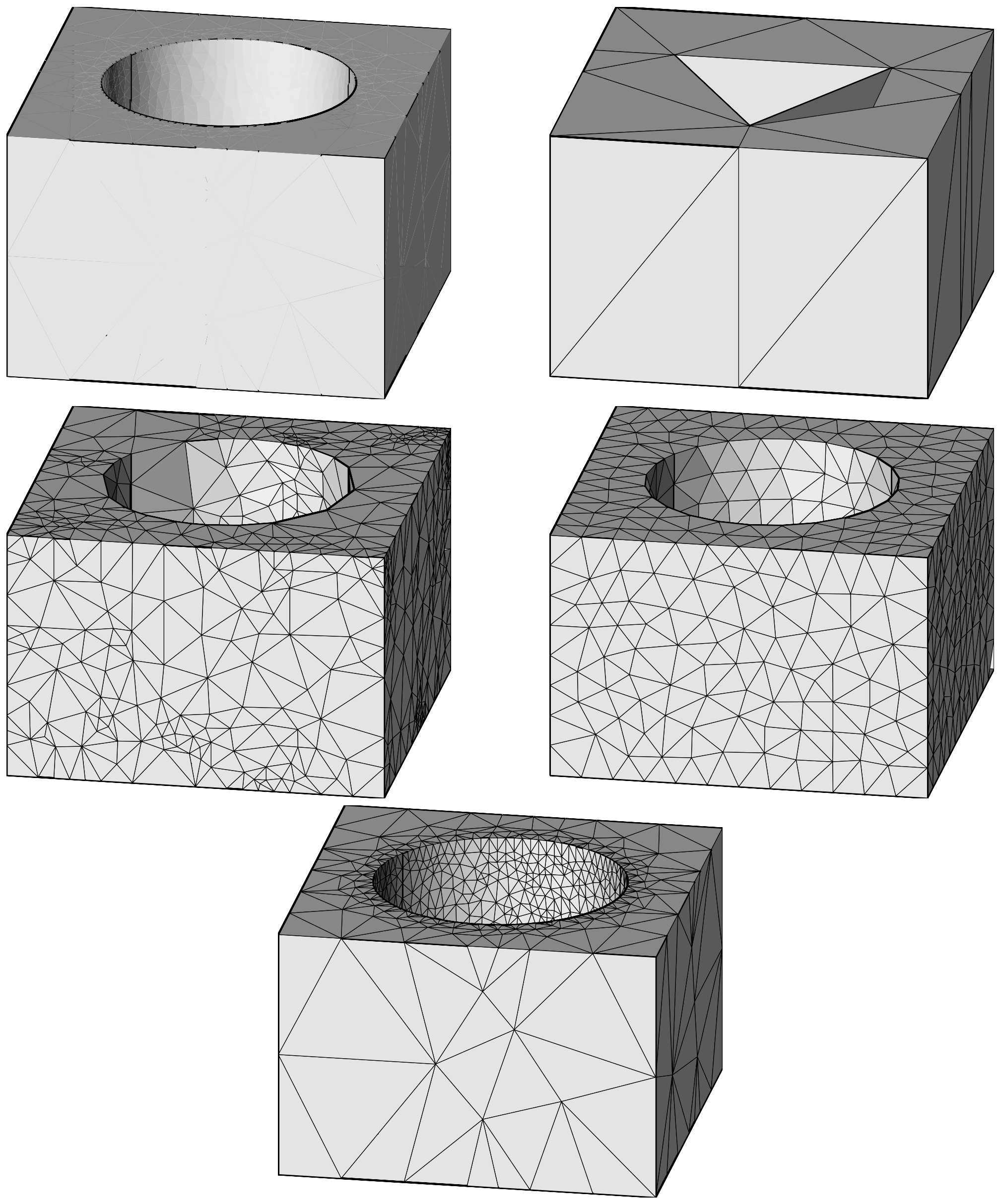

Figure 4.22: Meshes for the input piecewise curved complex (top-left): without any refinement with 26 nodes (top-right), randomly refined while keeping $\ell/R \geq 1/\sqrt{2}$ with 801 nodes (middle-left), uniformly refined by maximum edge length with 729 nodes (middle-right), and refined for accurate input geometry representation while keeping $\ell/R \geq 1/\sqrt{2}$ with 2006 nodes (bottom).

A dihedral angle analysis was performed for a refinement driven by the maximum

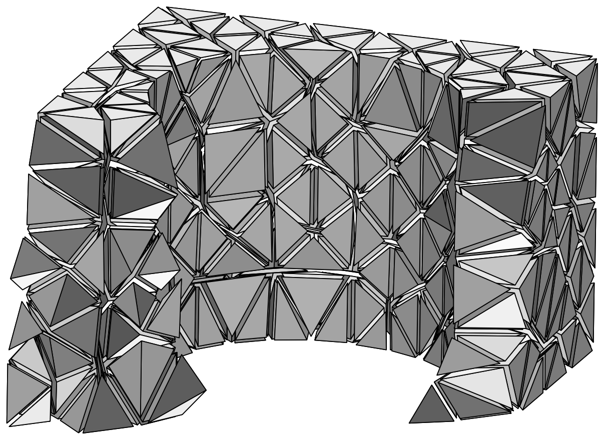

Figure 4.23: Slice of a simplicial 3-complex homeomorphic to an input piecewise curved complex.

edge length (Figure 4.24), by geometry approximation and minimum shortest edge circumradius ratio (Figure 4.25), and by geometry approximation and minimum dihedral angle (Figure 4.26). Notice the presence of slivers for the uniform refinement and the refinement based on the minimum shortest edge circumradius ratio.

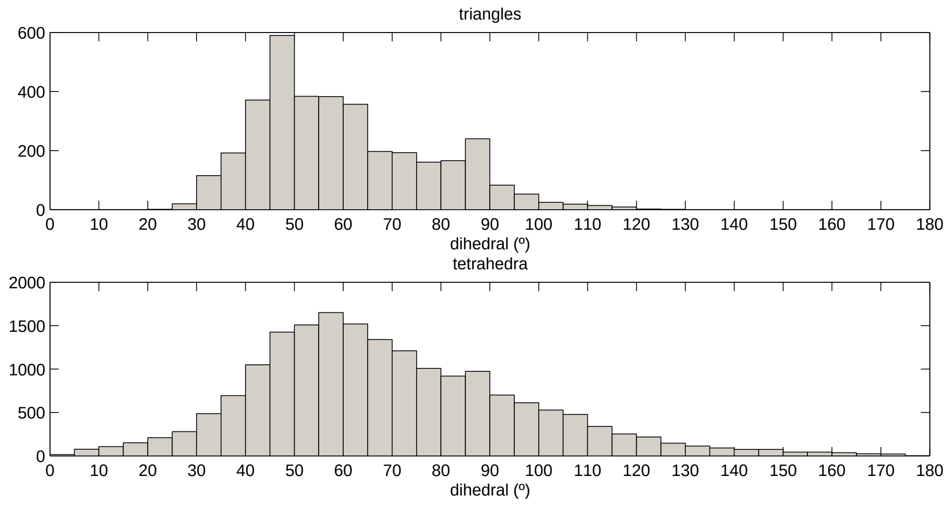

Figure 4.24: Histogram for dihedral angles of triangles and tetrahedra of a simplicial 3-complex refined by maximum edge length.

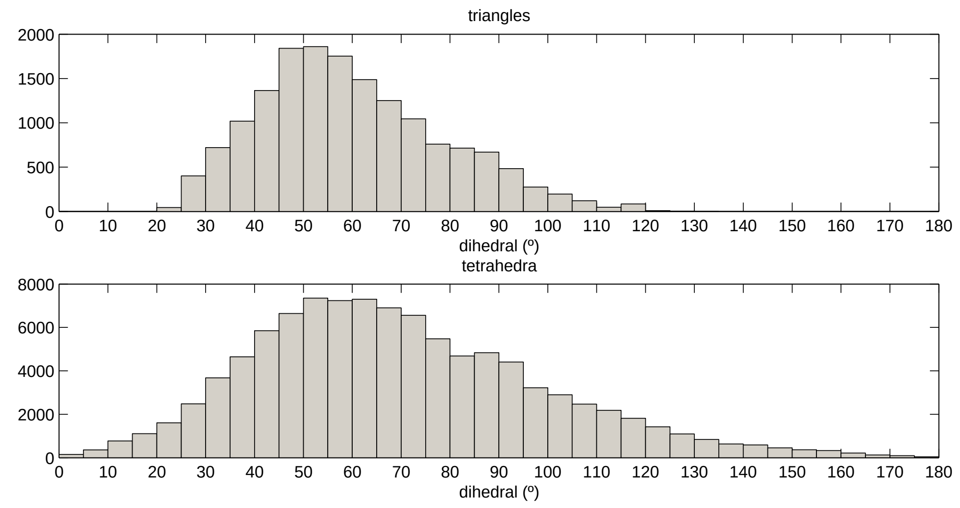

Figure 4.25: Histogram for dihedral angles of triangles and tetrahedra of a simplicial 3-complex refined for accurate input geometry representation while keeping $\ell/R \geq 1/\sqrt{2}$.

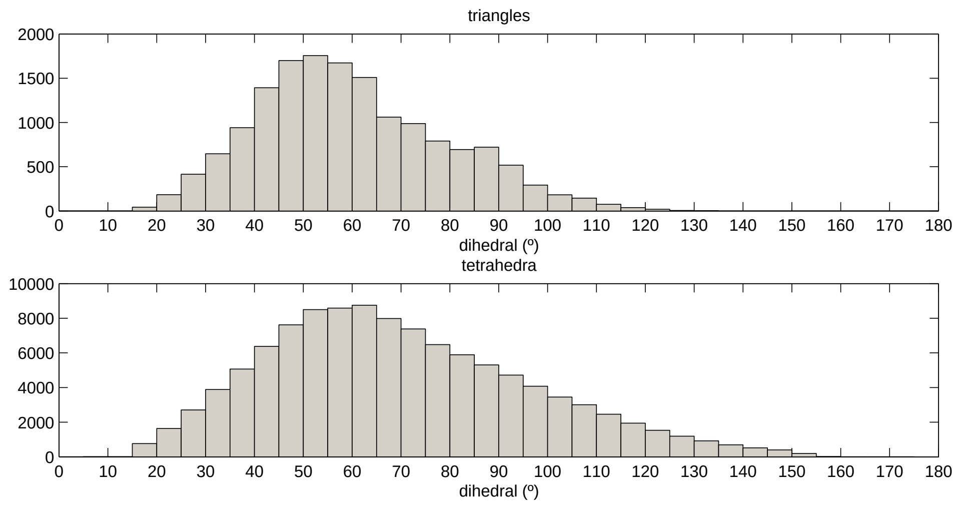

Figure 4.26: Histogram for dihedral angles of triangles and tetrahedra of a simplicial 3-complex refined for accurate input geometry representation while keeping dihedral angles above 15^o.

4.7 Conclusions

The practical performance of a Delaunay refinement outstrips theoretical statements. For instance, the algorithm usually terminates for minimum angles of about 33^o for manifold 2-complexes, where a bound of 26.45^o is expected. Furthermore, the Delaunay refinement algorithm is very fast, being able to insert about $50,000$ nodes/second for manifold 2-complexes in $\mathbb{R}^2$, and about $5,000$ nodes/second for manifold 3-complexes in $\mathbb{R}^3$.

The current implementation of the Delaunay algorithm for manifold complexes lacks some points like fast point location to detected encroachment during facet recovery. Even though it was possible to verify that a Delaunay refinement algorithm for curved manifold complexes is practical, specially in a case where point insertion in curved pieces are rarely performed (e.g. uniform refinement or random refinement).

The main bottle neck of the implemented Delaunay refinement algorithm for higher dimensions (typically greater than 3) lies in the “in ball” predicates evaluation. The predicate determinant should be further investigated to attempt to find feasible faster exact algorithms.

Appendix A

Simplices and balls

Simplices and balls are basic sets for many geometric concepts. The key of their importance is simplicity with handful properties. They are briefly addressed next.

Definition 51 (convex combination) *A convex combination of a set of vectors, columns of a matrix* $V = [\nu_1, ..., \nu_k] \in \mathbb{R}^{n\times k}$*, is a weighted sum of its elements* $V\lambda$*,* $\lambda \in [0,1]^k$*, where* $\sum_i \lambda_i = 1$*.*

Definition 52 (convex hull) *The convex hull of a set* $\mathbb{P}$*, denoted* $\operatorname{conv}(\mathbb{P})$*, is the set of all convex combinations of the elements in* $\mathbb{P}$*.*

Definition 53 (k-simplex) *A* k*-simplex is the convex hull of* $k+1$ *affinely independent points.*

A k-simplex, as shown in Figure A.1, can only be defined in $\mathbb{R}^n$ where $n \geq k$, since for $n < k$ the points are necessarily affinely dependent.

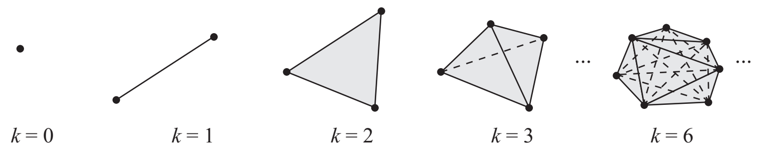

Figure A.1: Graphical representation of the projection of k-simplices on the plane.

All facets of a simplex are lower dimensional simplices (see Figure A.1). This induces a natural partition of a simplex into a collection of simplices.

Theorem 11 (simplex partition) *An* n*-simplex may be partitioned into a flat* n*-complex with* $\begin{pmatrix} n+1 \\ k+1 \end{pmatrix}$ *open* k*-simplices, for all* $k \in \{0, ..., n\}$*.*

Proof. The convex hull of any subset of the simplex vertices $\mathbb{P}$ is a lower dimensional simplex, since any point in it is a convex combination of $\mathbb{P}$ where the weights of the missing vertices are null. Hence, any combination of $k+1$ of the $n+1$ vertices in $\mathbb{P}$ is also a simplex. □

Definition 54 (k-ball) *A k-ball is the set $\mathbb{B}_{k,c,r}$ of all points in $\mathbb{R}^k$ that lies at most r away from a point c, i.e. $\mathbb{B}_{k,c,r} = \{x \in \mathbb{R}^k \mid \|x - c\| \leq r\}$.*

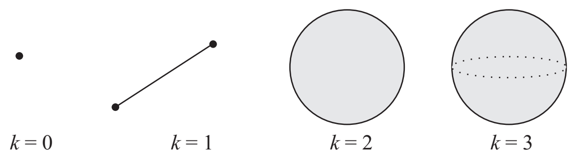

Figure A.2: Graphical representation of the projection of k-balls on the plane.

The abstract definition 54 of a ball depends on a distance function denoted by a norm. It is subtended to be the L-2 norm (see Figure A.2), unless otherwise specified.

Definition 55 (circumscribed ball) *The circumscribed ball of a simplex is the one whose boundary goes through all the vertices of it.*

A circumscribed n-ball for a k-simplex (see Figure A.3) is unique for $n = k$, but there are infinitely many for $n > k$. The minimum circumscribed n-ball for a k-simplex has its center lying the the k-flat that contains the k-simplex. The minimum circumscribed ball for a 1-simplex is called diametral ball, and for a 2-simplex it is called equatorial ball.

A circumscribed n-ball is not always the minimum enclosing n-ball for a k-simplex, even when $n = k$. For instance, in Figure A.3 the circumradius R is larger than the minimum enclosing radius a.

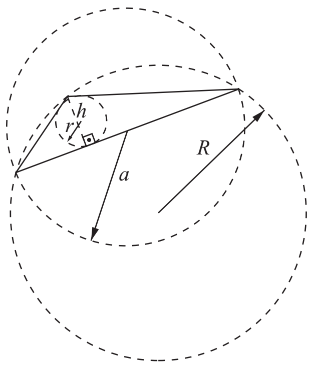

Figure A.3: Some parameters of a triangle: circumradius R, inradius r, height h and minimum enclosing radius a.

Enforcing the boundary of an n-ball to go through the $n+1$ vertices v_k of an n-simplex, $k = 0, ..., n$, leads to a system of linear equations

$$\begin{aligned} \|v_k - c\|^2 &= R^2 \\ \|v_k\|^2 - 2v_k \cdot c + \|c\|^2 &= R^2 \\ 2v_k \cdot c + R^2 - \|c\|^2 &= \|v_k\|^2 \\ 2v_k \cdot c + 2c_0 &= \|v_k\|^2 \end{aligned} \tag{A.1}$$

where $c_0 = (R^2 - \|c\|^2)/2$, c is the circumcenter and R is the circumradius, so that the circumcenter c can be written as

$$c = \frac{1}{2} \begin{bmatrix} (v_0 - v_n)^T \\ \vdots \\ (v_{n-1} - v_n)^T \end{bmatrix}^{-1} \begin{bmatrix} \|v_0\|^2 - \|v_n\|^2 \\ \vdots \\ \|v_{n-1}\|^2 - \|v_n\|^2 \end{bmatrix} \tag{A.2}$$

The minimum circumscribed n-ball of an $(n-1)$-simplex can calculated by adding, to the system of equations, the equation of the hyperplane containing the simplex. The same is valid for n-balls circumscribing lower dimensional simplices, where linearly independent hyperplane equations can be added to complete the system.

Definition 56 (height) *The height relative to a vertex v of a k-simplex is the distance from v to the $(k-1)$-flat that contains the opposite $(k-1)$-simplicial facet.*

Definition 57 (inscribed ball) *The inscribed ball of a simplex is the one whose boundary touches all the faces of it.*

The content V of an n-simplex is the integration of a scaled $(n-1)$-simplicial facet $\mathbb{F}_i$, with content A_i, along the height h_i (see Figure A.3) relative to the opposite vertex v_i, $i \in \{0, ..., n\}$,

$$\begin{aligned} V &= \int_0^{h_i} A_i \left(\frac{h}{h_i}\right)^{n-1} dh \\ &= \frac{1}{n} A_i h_i \end{aligned} \tag{A.3}$$

Now let p be a point in an n-simplex $\mathbb{S}$ and let h_i denote its relative height to each $(n-1)$-facet of $\mathbb{S}$ with content A_i. The content of $\mathbb{S}$ can be written as

$$V = \sum_{i=0}^{n} \frac{1}{n} A_i h_i \tag{A.4}$$

When p lies at the same height to any facet, all heights h_i become the inradius r and p becomes the incenter c

$$r = \frac{nV}{\sum_{i=0}^{n} A_i} \tag{A.5a}$$

$$c = \frac{\sum_{i=0}^{n} A_i v_i}{\sum_{i=0}^{n} A_i} \tag{A.5b}$$

The inscribed n-ball of an n-simplex is always the maximum n-ball contained in it.

When a point is expressed as signed distances from the lines containing the edges of a triangle it is in a trilinear coordinates system. Analogously, when a point is expressed as signed distances from the planes containing the triangular faces of a tetrahedron it is in a quadriplanar coordinates system. Naturally, the center of a inscribed ball is where all its trilinear (quadriplanar, or their generalization to higher dimensions) ordinates are the same.

When a point is expressed as signed contents relative to the n-simplices formed with the $(n-1)$-facets of an n-simplex it is in a barycentric coordinates system. Barycentric coordinates are closely related to trilinear or quadriplanar coordinates, as heights are closely related to contents.

The content V of an n-simplex is given by

$$V = \frac{1}{n} \begin{vmatrix} (v_1 - v_0)^T \\ \vdots \\ (v_n - v_0)^T \end{vmatrix} \tag{A.6}$$

where the sign of the determinant is dependent on the order of the vertices v_i, $i = 0, ..., n$. This sign is related to which side of an n-simplex $(n-1)$-facet a point lies.

A.1 Regular simplex

A regular simplex is the one with constant distance L (edge length) between any pair of its vertices ν_i and ν_j, *i.e.* a simplex where

$$\|\nu_i - \nu_j\| = L,\ \forall i \neq j \tag{A.7}$$

A regular n-simplex can be iteratively built, starting from a 0-simplex, by adding a new vertex equidistant to the $(i-1)$-simplex vertices to create a i-simplex, as shown in Figure A.4 and described in Algorithm 3 in full details. In this procedure, every new vertex is placed h_i orthogonally away from the centroid c_{i-1} of the previous $(i-1)$-simplex $\mathbb{S}_{i-1}$. Since the distance from c_{i-1} to any vertex of $\mathbb{S}_{i-1}$ is the circumscribed radius R_{i-1}, and from the new vertex ν_{i+1} is the edge length L,

$$h_i = \sqrt{L^2 - R_{i-1}^2} \tag{A.8}$$

and, similarly, the distance from c_{i-1} to any facet of $\mathbb{S}_{i-1}$ is the inscribed radius r_{i-1}, and from the new vertex ν_{i+1} is the height h_{i-1},

$$h_i = \sqrt{h_{i-1}^2 - r_{i-1}^2} \tag{A.9}$$

After each vertex insertion, the centroid c_i of the new i-simplex $\mathbb{S}_i$ is placed at the origin of the coordinate system. Taking this into consideration, $\mathbb{S}_i$ only needs to be centered along the new ordinate of the coordinate system, where i vertices are in zero, and the new vertex is in h_i. By definition, the inscribed and circumscribed radii are

$$\begin{aligned} r_i &= \frac{h_i}{i+1} \\ R_i &= \frac{ih_i}{i+1} \end{aligned} \tag{A.10}$$

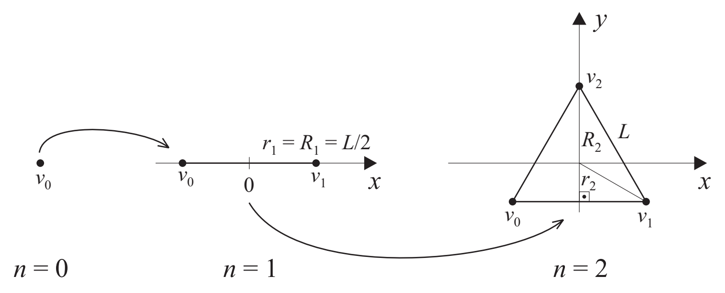

Figure A.4: Iterative construction of a regular simplex.

Algorithm 3 Generation of regular n-simplices.

Input: simplex dimension n and edge length L.
Output: vertices $v_i, i = 1, ..., n+1$ of a regular n-simplex.

```
1. $R_0 \leftarrow 0$
2. for $i \leftarrow 1$ to $n$
3.    $h_i \leftarrow \sqrt{L^2 - R_{i-1}^2}$
4.    $r_i \leftarrow \frac{h_i}{i+1}$
5.    for $j \leftarrow 1$ to $i$
6.       $v_{j,i} \leftarrow -r_i$
7.    $R_i \leftarrow i r_i$
8.    $v_{i+1} \leftarrow \{\underbrace{0, ...,}_{i-1} R_i\}$
```

and thus

$$\frac{R_i}{r_i} = i \tag{A.11}$$

which is a very interesting simple relation between these radii of a regular i-simplex.

The content V_n of an n-simplex is the integration of a scaled $(n-1)$-simplex along its height h_n

$$\begin{aligned} V_n &= \int_0^{h_n} \left(\frac{h}{h_n}\right)^{n-1} V_{n-1} dh \\ &= \frac{h_n}{n} V_{n-1} \\ &= \prod_{i=1}^{n} \frac{h_i}{i} \\ &= \frac{1}{n!} \prod_{i=1}^{n} h_i \end{aligned} \tag{A.12}$$

The inscribed radius r_n, circumscribed radius R_n, height h_n and content V_n of a regular n-simplex may be rewritten as a function of the edge length L and dimension n (see Figure A.5) as

$$r_n = L \sqrt{\frac{1}{2n(n+1)}} \tag{A.13a}$$

$$R_n = L\,\sqrt{\frac{n}{2(n+1)}} \tag{A.13b}$$

$$h_n = L\,\sqrt{\frac{n+1}{2n}} \tag{A.13c}$$

$$V_n = L^n\,\frac{\sqrt{n+1}}{n!\;2^{n/2}} \tag{A.13d}$$

or by recurrence relations

$$r_n = r_{n-1}\,\sqrt{\frac{n-1}{n+1}} \tag{A.14a}$$

$$R_n = R_{n-1}\,\sqrt{\frac{1}{1-1/n^2}} \tag{A.14b}$$

$$h_n = h_{n-1}\,\sqrt{1-\frac{1}{n^2}} \tag{A.14c}$$

$$V_n = V_{n-1}\;L\,\sqrt{\frac{n+1}{2n^3}} \tag{A.14d}$$

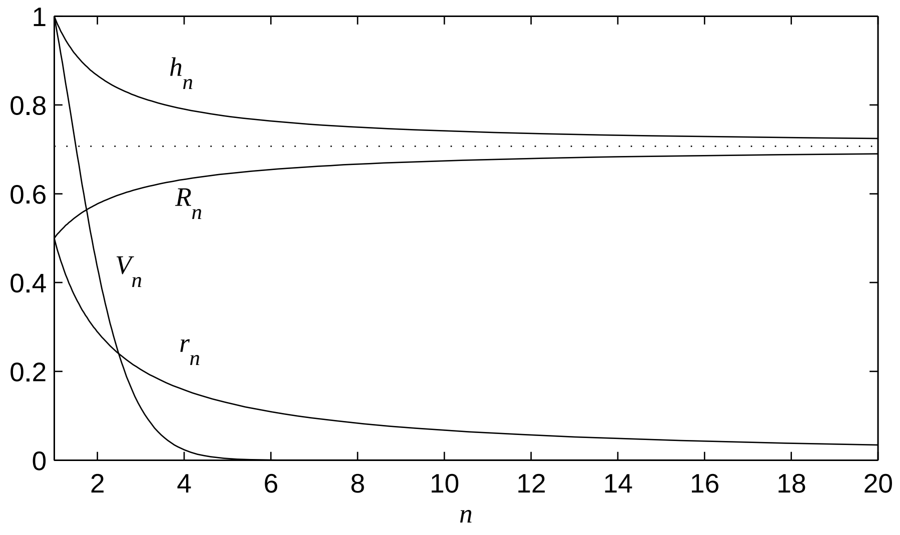

Figure A.5: Variation of inradius r_n, circumradius R_n, height h_n and content V_n of a unit edge length regular n-simplex.

The dihedral angle between $(n-1)$-facets of a regular simplex is given by

$$\alpha = \arctan\left(\frac{h_n}{r_{n-1}}\right) = \arctan\sqrt{n^2-1} \tag{A.15}$$

which tends to 90^o as n goes to infinite.

Appendix B

Parametrization by control points

The parametrization by control points started with the parallel works of Paul de Casteljau and Pierre Bézier, motivated by the creation of an exact geometric model of a car [Farin, 2002]. The fundamental idea of this approach is to use points as parameters, so that each one controls the shape of the object according to its position.

The first algorithm proposed to blend points into pieces (curves, surfaces, ...) was proposed by Paul de Casteljau [de Casteljau, 1959, de Casteljau, 1963] in technical reports, depicted in Figure B.1. Consider the sequence of points $q_0, ..., q_n$ at iteration $k = 0$. For a curve parameterized by $t \in [0, 1]$, the de Casteljau algorithm iteratively reduces the sequence of points by setting $q_{i,k+1} = q_{i,k} + t(q_{i+1,k} - q_{i,k})$, $i = 0, ..., n - k - 1$, until the sequence becomes a single point after $k = n$ iterations. Paul de Casteljau wrote those pieces in the form

$$c(t) = \sum_{i=0}^{n} f_i(t) q_i \tag{B.1}$$

where $f_i(t) : \mathbb{R}^m \mapsto \mathbb{R}$ are the Bernstein polynomials [Bernstein, 1912] and $q_i \in \mathbb{R}^d$ are the control points for the m-piece. These polynomials were used by Sergi Bernstein to prove that any continuous function can be uniformly approximated by a polynomial inside a closed interval to any degree of accuracy, known as the Weierstrass's approximation theorem.

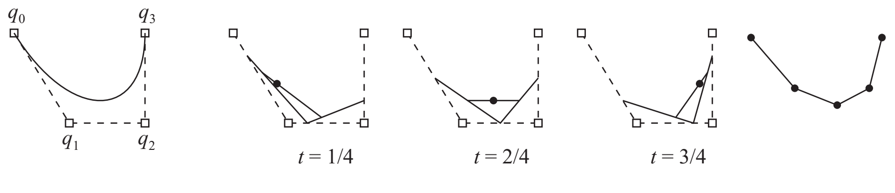

Figure B.1: Geometric representation of the de Casteljau algorithm for the control points q_0, q_1, q_2, q_3 at $t = 1/4, 2/4, 3/4$, which leads to the 4 line segments approximation (right) for the full precision curve (left).

Pierre Bézier reached the same pieces as did de Casteljau, but he published his work [Bézier, 1966, Bézier, 1967]. Forrest saw that the pieces proposed by Bézier could be written using Bernstein polynomials [Forrest, 1972], which popularized them. The de Casteljau work was only recognized in the late seventies.

The function $f_i(t)$ in (B.1) is called blending function, and there are many blending functions other than Bernstein polynomials. Good blending functions induce handful properties into the resulting pieces. These properties include starting and ending in control points, containment in the convex hull of the control points, closure under affine or perspective transformations, control over k-derivative continuity, or range of control of a control point.

The modern theory of splines started in 1946 by Isaac Schoenberg [Schoenberg, 1946] in his work on approximation theory. In 1972 they were written in the form (B.1) by de Boor [de Boor, 1972] and Cox [Cox, 1972], in parallel works. In 1974 the Bézier pieces were proven to be a special case of b-splines by Gordon and Riesenfeld [Gordon and Riesenfeld, 1974].

Other examples of parametrization by control points are the Overhauser curves [Overhauser, 1968] (also known as cubic Catmull-Rom splines) that interpolate all control points, the Ball's rational cubic [Ball, 1974] that represents conic sections exactly, and the parametric cubic [Timmer, 1980] that enforces the curve to go through the internal segment midpoint.

The Bézier and b-spline pieces will be depicted in the next sections.

B.1 Bézier

The Bézier curves (see Figure B.2) are defined as

$$c(t) = \sum_{i=0}^{n} B_{i,n}(t) q_i, \quad t \in [0,1] \tag{B.2}$$

where q_i are the control points, n is the curve degree and $B_{i,n}(t)$ are the Bernstein polynomials (see Figure B.3)

$$B_{i,n}(t) = \frac{n!}{i!(n-i)!}\, t^i (1-t)^{n-i} \tag{B.3}$$

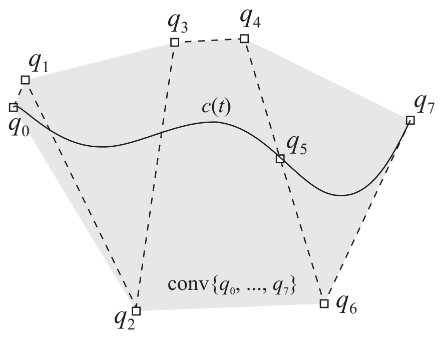

Figure B.2: Example of a Bernstein-Bézier curve of degree $n = 7$.

The Bernstein polynomial $B_{i,n}(t)$ can be seen as the probability of picking exact i white balls in n draws from an urn containing a fraction of t white balls and $1-t$

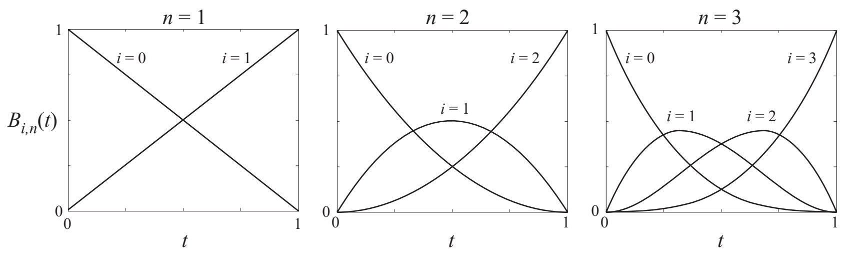

Figure B.3: Bernstein polynomials of degree $n = 1, 2, 3$.

black balls. This makes trivial to prove some useful properties, like the partition of unit

$$\sum_{i=0}^{n} B_{i,n}(t) = 1 \tag{B.4}$$

which states that the Bernstein-Bézier curve $c(t)$, $t \in [0, 1]$, lies in the convex hull of the control points (see Figure B.2), or the fixed point property

$$\begin{aligned} t = 0 \quad &\Rightarrow \quad B_{i,n}(t) = \begin{cases} 1, & \text{if } i = 0 \\ 0, & \text{if } i \neq 0 \end{cases} \\ t = 1 \quad &\Rightarrow \quad B_{i,n}(t) = \begin{cases} 1, & \text{if } i = n \\ 0, & \text{if } i \neq n \end{cases} \end{aligned} \tag{B.5}$$

so that $c(0) = q_0$ and $c(1) = q_n$ (see Figure B.2).

The derivative of $B_{i,n}(t)$ is given by

$$\begin{aligned} B'_{i,n}(t) &= \frac{n!}{i!(n-i)!} \left[i t^{i-1}(1-t)^{n-i} + t^i (n-i)(1-t)^{n-i-1}(-1) \right] \\ &= n \left[\frac{(n-1)!}{(i-1)!(n-i)!} t^{i-1}(1-t)^{n-i} - \frac{(n-1)!}{i!(n-i-1)!} t^i (1-t)^{n-i-1} \right] \\ &= n \left[B_{i-1,n-1}(t) - B_{i,n-1}(t) \right] \end{aligned} \tag{B.6}$$

and thus the Bézier curve derivative is a lower degree Bézier curve

$$\begin{aligned} c'(t) &= \sum_{i=0}^{n} B'_{i,n}(t) q_i \\ &= n \sum_{i=0}^{n} \left[B_{i-1,n-1}(t) - B_{i,n-1}(t) \right] q_i \\ &= n \left[B_{-1,n-1}(t) q_0 - B_{n,n-1}(t) q_n + \sum_{i=0}^{n-1} B_{i,n-1}(t)(q_{i+1} - q_i) \right] \\ &= n \sum_{i=0}^{n-1} B_{i,n-1}(t)(q_{i+1} - q_i) \\ &= \sum_{i=0}^{n-1} B_{i,n-1}(t) q'_i \end{aligned} \tag{B.7}$$

where

$$q'_i = n(q_{i+1} - q_i) \tag{B.8}$$

which, together with the fixed point property, leads to the nice tangent property at fixed points (see Figure B.2)

$$\begin{aligned} c'(0) &= q'_0 = n(q_1 - q_0) \\ c'(1) &= q'_{n-1} = n(q_n - q_{n-1}) \end{aligned} \tag{B.9}$$

B.2 B-spline

The b-splines (short for basis splines, see Figure B.4) are defined as

$$c(t) = \sum_{i=0}^{n} N_{i,p,u}(t) q_i, \quad t \in [u_p, u_{m-p}) \tag{B.10}$$

where $N_{i,p,u}(t)$ are the basis functions (see Figure, B.5) defined by the recurrence relation

$$N_{i,p,u}(t) = \frac{t - u_i}{u_{i+p} - u_i} N_{i,p-1,u}(t) + \frac{u_{i+p+1} - t}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1,u}(t) \tag{B.11}$$

where

$$N_{i,0,u}(t) = \begin{cases} 1, & \text{if } u_i \le t < u_{i+1} \\ 0, & \text{otherwise} \end{cases} \tag{B.12}$$

and the knot vector $u = [u_0, u_1, ..., u_m] \in [0,1]^{m+1}$ is a nondecreasing sequence, and $p = m - n - 1$ is the degree of the b-spline. The knots $u_{p+1}, ..., u_{m-p-1}$ are called internal knots. When the knot values are equally spaced, the b-spline is said to be uniform.

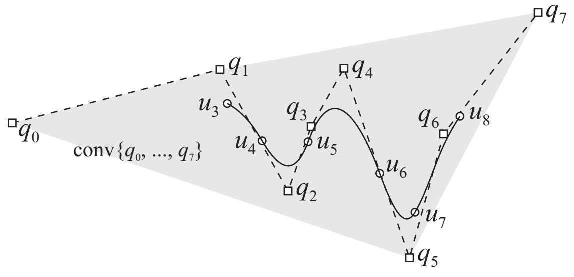

Figure B.4: Example of a b-spline curve of degree $p = 3$.

Each basis polynomial $N_{i,p,u}(t)$ is nonzero through $p+1$ knot subintervals in $[u_p, u_{m-p})$, and there are $p+1$ nonzero polynomials in each knot subinterval in $[u_p, u_{m-p})$ (see Figure B.5). These properties define a range for the effect of a control point, which is a very nice feature, specially in freeform modeling of complex geometries. Furthermore, the partition of unity holds true

$$\sum_{j=i-p}^{i} N_{j,p}(t) = 1, \quad \forall t \in [u_i, u_{i+1}), \quad \forall i = p, ..., m-p-1 \tag{B.13}$$

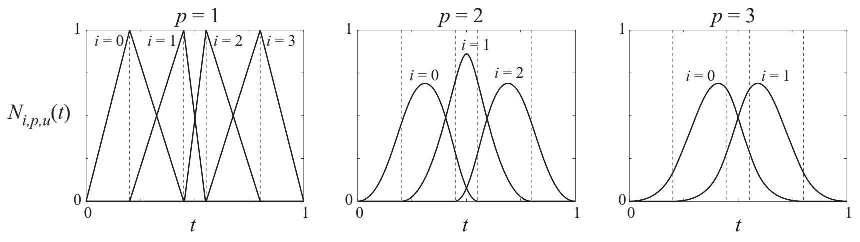

Figure B.5: Basis polynomials of degree $p = 1, 2, 3$ for the same knot vector $u = [0, 0.2, 0.45, 0.55, 0.8, 1]$ (notice that no curve is defined for $p = 3$, since $[u_p, u_{m-p})$ is empty).

for each subinterval in $[u_p, u_{m-p})$, so that the b-spline curve lies in the convex hull of its control points. Actually, this property is stronger. Each subsegment $[u_i, u_{i+1})$ lies in the convex hull of the subset of control points $q_{i-p}, ..., q_i$.

The derivative of $N_{i,p,u}(t)$ is given by

$$\begin{aligned} N'_{i,p,u}(t) &= \frac{(t-u_i)N'_{i,p-1,u}(t) + N_{i,p-1,u}(t)}{u_{i+p}-u_i} + \frac{(u_{i+p+1}-t)N'_{i+1,p-1,u}(t) - N_{i+1,p-1,u}(t)}{u_{i+p+1}-u_{i+1}} \\ &= \frac{p}{u_{i+p}-u_i}\ N_{i,p-1,u}(t) - \frac{p}{u_{i+p+1}-u_{i+1}}\ N_{i+1,p-1,u}(t) \end{aligned} \tag{B.14}$$

and thus the b-spline curve derivative is a lower degree b-spline curve

$$\begin{aligned} c'(t) &= \sum_{i=0}^{n} N'_{i,p,u}(t) q_i \\ &= p\sum_{i=0}^{n}\left[\frac{N_{i,p-1,u}(t)}{u_{i+p}-u_i} - \frac{N_{i+1,p-1,u}(t)}{u_{i+p+1}-u_{i+1}}\right] q_i \\ &= p\left[\frac{N_{0,p-1,u}(t)}{u_p-u_0}q_0 - \frac{N_{n+1,p-1,u}(t)}{u_m-u_{m-p}}q_n + \sum_{i=0}^{n-1} N_{i+1,p-1,u}(t)\ \frac{q_{i+1}-q_i}{u_{i+p+1}-u_{i+1}}\right] \\ &= p\sum_{i=0}^{n-1} N_{i,p-1,u'}(t)\ \frac{q_{i+1}-q_i}{u'_{i+p}-u'_i} \\ &= \sum_{i=0}^{n-1} N_{i,p-1,u'}(t) q'_i \end{aligned} \tag{B.15}$$

where

$$\begin{aligned} q'_i &= p\ \frac{q_{i+1}-q_i}{u'_{i+p}-u'_i} \\ u' &= [u_1, ..., u_{m-1}] \end{aligned} \tag{B.16}$$

A b-spline curve is $p-k$ times differentiable (C^{p-k}) at a k duplicate knot, and smooth (C^∞) elsewhere. Whenever $u_j = u_{j+1} = ... = u_{j+k-1}$, $k \geq p$ and $j \in$

$\{p+1, ..., m-2p\}$ (k duplicate internal knot), the fixed point property

$$\begin{aligned} \lim_{t \to u_j^-} c(t) &= q_j \\ c(u_j) &= q_{j+k-p} \end{aligned} \tag{B.17}$$

and the tangent property

$$\begin{aligned} \lim_{t \to u_j^-} c'(t) &= p(q_j - q_{j-1})/(u_j - u_{j-1}) \\ \lim_{t \to u_j^+} c'(t) &= p(q_{j+k-p+1} - q_{j+k-p})/(u_{j+k} - u_j) \end{aligned} \tag{B.18}$$

holds true. Notice that the curve is not continuous if $k > p$. Thus, the derivative control points q'_i together with the knots u'_i must be removed whenever $u'_{i+p} = u'_i$. The b-spline degree does not change after this removal.

A b-spline curve does not necessarily start at q_0 and end at q_n. To enforce this, the first and last knots must be $p+1$ duplicate, so that $c(u_0) = q_0$ and $\lim_{t \to u_m} c(t) = q_n$. Under this condition, a b-spline is said to be clamped and the tangent property

$$\begin{aligned} c'(u_0) &= p(q_1 - q_0)/(u_{p+1} - u_1) \\ \lim_{t \to u_m} c'(t) &= p(q_n - q_{n-1})/(u_{m-1} - u_{m-p-1}) \end{aligned} \tag{B.19}$$

holds true.

The b-spline curve is a generalization of the Bernstein-Bézier curve, since $N_{i,p,u}(t) = B_{i,n}(t)$ whenever $u = [\underbrace{0, ..., 0}_{p+1}, \underbrace{1, ..., 1}_{p+1}]$ and $n = p$. Indeed, they have many properties in common.

Similarly to the Bézier curve, a b-spline curve can be evaluated by reducing a sequence of points. This is called the de Boor's algorithm. Consider the sequence of points $q_{j-p}, ..., q_j$ at iteration $k = 0$, so that the parameter $t \in [u_j, u_{j+1})$. The de Boor's algorithm iteratively reduces the sequence of points by setting

$$q_{i,k+1} = q_{i,k} + \frac{t - u_{i+k+1}}{u_{i+p+1} - u_{i+k+1}} (q_{i+1,k} - q_{i,k}), \quad i = j-p, ..., j-k-1 \tag{B.20}$$

until the sequence becomes a single point after $k = p$ iterations.

B.2.1 Nonuniform rational b-splines

A NURBS (short for nonuniform rational b-splines) is defined by

$$c(t) = \sum_{i=0}^{n} R_{i,p,u,w}(t) q_i, \quad t \in [u_p, u_{m-p}) \tag{B.21}$$

where w is a weight vector and

$$R_{i,p,u,w}(t) = \frac{N_{i,p,u}(t) w_i}{\sum_{i=0}^{n} N_{i,p,u}(t) w_i} \tag{B.22}$$

The rational basis function of a NURBS curve $R_{i,p,u,w}(t)$ is a partition of unit (denominator is a normalization factor), thus it lies in the convex hull of its control points.

The perspective projection of a NURBS is also a NURBS. To clarify this intrinsic feature of a NURBS, consider the homogeneous coordinate system. By convention, the Cartesian coordinates are given by the perspective projection of the homogeneous coordinates onto $w = 1$, where w is the distance from the perspective vertex (see Figure B.6). Thus, a point h in homogenous coordinates is mapped to a point q in Cartesian coordinates by

$$q = \begin{bmatrix} x \\ y \\ \vdots \\ 1 \end{bmatrix} = \begin{bmatrix} wx \\ wy \\ \vdots \\ w \end{bmatrix} /w = h/w \tag{B.23}$$

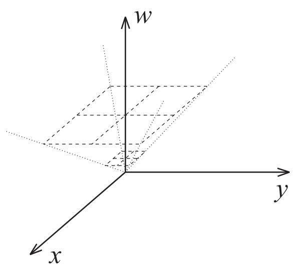

Figure B.6: Geometric representation in a Cartesian coordinate system of the homogeneous coordinate system as projections on a plane.

Since the last ordinate of q is always 1, the Cartesian coordinates are lower dimensional than their respective homogeneous coordinates. A b-spline in homogeneous coordinates is given by

$$c(t) = \sum_{i=0}^{n} N_{i,p,u}(t) h_i \tag{B.24}$$

Each component of $c(t)$ may be evaluated independently since the basis $N_{i,p,u}(t)$ is a scalar real value. Hence, expressing the same $c(t)$ in cartesian coordinates

$$c(t) = \frac{\sum_{i=0}^{n} N_{i,p,u}(t) h_i}{\sum_{i=0}^{n} N_{i,p,u}(t) w_i} = \frac{\sum_{i=0}^{n} N_{i,p,u}(t) w_i q_i}{\sum_{i=0}^{n} N_{i,p,u}(t) w_i} \tag{B.25}$$

leads to (B.21). Note that a NURBS may be evaluated by b-splines in homogeneous coordinates.

The weight was introduced to the NURBS to provide a natural way to apply perspective transformations to a b-spline. Furthermore, the higher weight w_i is, the more effective will be its respective control point q_i. This extra degree of freedom allows NURBS to represent any conical section exactly.

The derivative of a NURBS curve in cartesian coordinates is given by

$$\begin{aligned} c'(t) &= \left[\frac{\sum_{i=0}^{n} N_{i,p,u}(t)h_i}{\sum_{i=0}^{n} N_{i,p,u}(t)w_i}\right]' \\ &= \left[\frac{c_h(t)}{c_w(t)}\right]' \\ &= \frac{c_h'(t)c_w(t) - c_h(t)c_w'(t)}{c_w^2(t)} \end{aligned} \tag{B.26}$$

and thus it is possible to evaluate NURBS derivatives by keeping track of the respective b-spline derivatives in homogeneous coordinates.

The NURBS becomes a common b-spline curve whenever $w = [1, ..., 1]$, where $R_{i,p,u,w}(t) = N_{i,p,u}(t)$.

B.3 Higher dimensional pieces

B.3.1 Rectangular like

A curve parameterized by control points can be generalized to higher dimensions using the tensor product. For instance, a Bézier d-piece is given by

$$c(t) = \sum_i q_i \prod_{k=0}^{d-1} B_{i_k,n_k}(t_k), \quad t \in [0,1]^d \tag{B.27}$$

a b-spline d-piece (see Figure B.7) is given by

$$c(t) = \sum_i q_i \prod_{k=0}^{d-1} N_{i_k,p_k,u_k}(t_k) \tag{B.28}$$

and a NURBS d-piece is given by

$$c(t) = \frac{\sum_i w_i q_i \prod_{k=0}^{d-1} N_{i_k,p_k,u_k}(t_k)}{\sum_i w_i \prod_{k=0}^{d-1} N_{i_k,p_k,u_k}(t_k)} = \sum_i q_i \prod_{k=0}^{d-1} R_{i_k,p_k,u_k}(t_k) \tag{B.29}$$

where

$$i \in \{0, 1, ..., n_0\} \times ... \times \{0, 1, ..., n_{d-1}\} \tag{B.30}$$

defines a rectangular control grid (see Figure B.8).

Let $c_k'(t)$ denote the partial derivative $\partial c(t)/\partial t_k$, and $e_k \in \{0,1\}^d$ denote a vector with 1 in the k^{th} position and 0 elsewhere. The partial derivatives of a rectangular Bézier d-piece are given by

$$c_k'(t) = n_k \sum_i (q_{i+e_k} - q_i) B_{i_k,n_k-1}(t_k) \prod_{j \neq k} B_{i_j,n_j}(t_j) \tag{B.31}$$

The partial derivatives of a b-spline d-piece follow similarly.

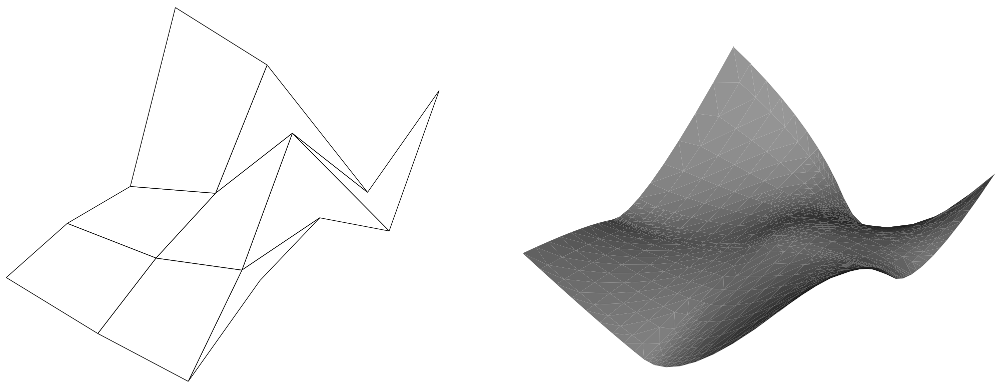

Figure B.7: Control polygon (left) and respective b-spline surface (right).

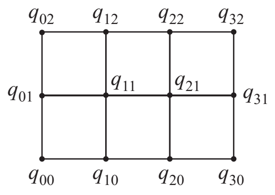

Figure B.8: Numbering scheme for the rectangular control polygon for $n = 3 \times 2$.

B.3.2 Simplicial like

A Bézier curve can be extended to a Bézier d-piece by

$$c(t) = \sum_i B_{i,n}(t) q_i, \quad t \in [0,1]^{d+1}, i \in \{0, 1, ..., n\}^{d+1} \tag{B.32}$$

where the Bernstein polynomial $B_{i,n}(t)$ of degree n is generalized to the multinomial distribution

$$B_{i,n}(t) = n! \, \frac{\prod_{k=0}^{d} t_k^{i_k}}{\prod_{k=0}^{d} i_k} \tag{B.33}$$

and where the constraints

$$\sum_{k=0}^{d} i_k = n \tag{B.34a}$$

$$\sum_{k=0}^{d} t_k = 1 \tag{B.34b}$$

makes the control grid look like a d-simplex (see Figures B.9 and B.10).

By convention, t_d and i_d are considered the dependent variables, given by

$$i_d = n - \sum_{k=0}^{d-1} i_k \tag{B.35a}$$

$$t_d = 1 - \sum_{k=0}^{d-1} t_k \tag{B.35b}$$

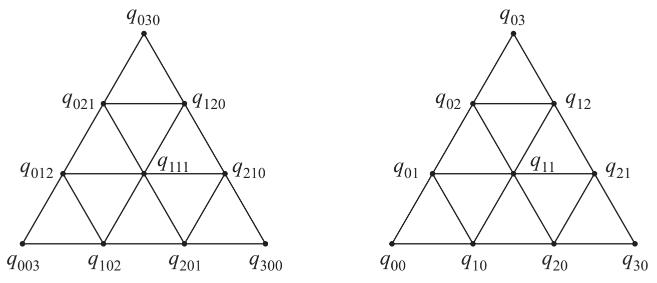

Figure B.9: Numbering scheme for the simplicial control polygon for $n = 3$ (left), and its simplified version (right) where the last index is omitted since it is completely defined by the other ones.

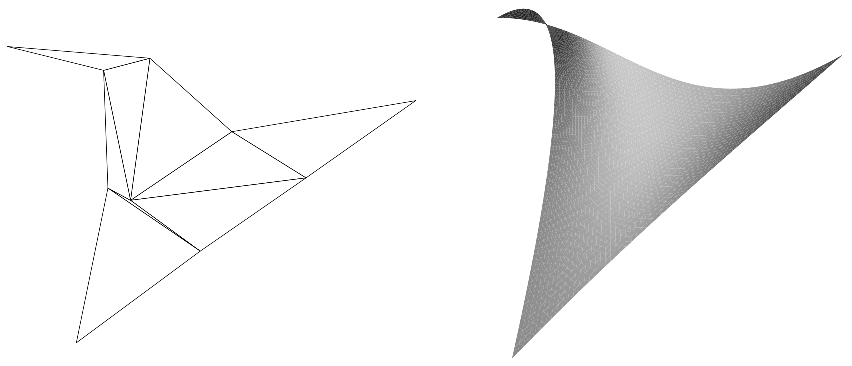

Figure B.10: Control polygon (left) and respective Bézier surface (right).

Let $c'_k(t)$ denote the partial derivative $\partial c(t)/\partial t_k$, and $e_k \in \{0,1\}^{d+1}$ denote a vector with 1 in the k^{th} position and 0 elsewhere. The partial derivatives of a simplicial Bézier d-piece are given by lower degree Bézier d-pieces

$$c'_k(t) = n \sum_i B_{i,n-1}(t)(q_{i+e_k} - q_i) \tag{B.36}$$

Consider an urn containing balls where t_k, $k = 0, ..., d$, is the fraction of the color k in $d+1$ possible colors. Let also i_k, $k = 0, ..., d$, count the number of balls of color k. The extended Bernstein polynomial $B_{i,n}(t)$ can be seen as the probability of picking exact i balls in n draws from the urn. This makes trivial to prove the partition of unit, fixed point, or tangent properties, just like in Bézier curves.

Bibliography

[Amenta and Bern, 1998] Amenta, N. and Bern, M. (1998).
Surface reconstruction by voronoi filtering.
Proceedings of the 14th Symposium on Computational Geometry, pages 39–48.

[Baker et al., 1988] Baker, B. S., Grosse, E., and Rafferty, C. S. (1988).
Nonobtuse triangulation of polygons.
Discrete Computational Geometry, 3(2):147–168.

[Ball, 1974] Ball, A. A. (1974).
Introduction to the conic lofting tile.
Computer-Aided Design, 6:243–249.

[Baumgart, 1975] Baumgart, B. G. (1975).
A polyhedron representation for computer vision.
National Computer Conference Proceedings, pages 589–596.

[Bern et al., 1990] Bern, M., Eppstein, D., and Gilbert, J. (1990).
Provably good mesh generation.
31st Annual Symposium on Foundations of Computer Science Proceedings, 48(3):384–409.

[Bernstein, 1912] Bernstein, S. N. (1912).
Démonstration du théorème de weierstrass fondée sur le calcul des probabilities.
Communications of the Kharkov Mathematical Society, 13:1–2.

[Bézier, 1966] Bézier, P. (1966).
Définition numérique des courbes et surfaces i.
Automatisme, 11:625–632.

[Bézier, 1967] Bézier, P. (1967).
Définition numérique des courbes et surfaces ii.
Automatisme, 12:17–21.

[Blandford et al., 2003] Blandford, D. K., Blelloch, G. E., Cardoze, D. E., and Kadow, C. (2003).
Compact representations of simplicial meshes in two and three dimensions.
12th International Meshing Roundtable Proceedings.

[Boissonnat and Oudot, 2007] Boissonnat, J.-D. and Oudot, S. (2007).
Provably good sampling and meshing of lipschitz surfaces.
Proceedings of the 23rd Symposium on Computational Geometry, pages 158–167.

[Boivin and Ollivier-Gooch, 2002] Boivin, C. and Ollivier-Gooch, C. (2002).
Guaranteed-quality triangular mesh generation for domains with curved boundaries.
International Journal for Numerical Methods in Engineering, 55(10):1185–1213.

[Borovikov et al., 2005] Borovikov, S. N., Kryukov, I. A., and Ivanov, I. E. (2005).
An approach for delaunay tetrahedralization of bodies with curved boundaries.
In *International Mesh Roundtable proceedings*, pages 221–237.

[Bowyer, 1981] Bowyer, A. (1981).
Computing dirichlet tessellations.
Computer Journal, 24(2):162–166.

[Chen and Bishop, 1997] Chen, H. and Bishop, J. (1997).
Delaunay triangulation for curved surfaces.
Proceedings of the 6th International Meshing Roundtable.

[Cheng et al., 1999] Cheng, S.-W., Dey, T. K., Edelsbrunner, H., Facello, M. A., and Teng, S.-H. (1999).
Sliver exudation.
15th Symposium on Computational Geometry ACM, pages 1–13.

[Cheng et al., 2007a] Cheng, S.-W., Dey, T. K., and Levine, J. A. (2007a).
A practical delaunay meshing algorithm for a large class of domainss.
Proceedings of the 16th International Meshing Roundtable.

[Cheng et al., 2007b] Cheng, S.-W., Dey, T. K., and Ramos, E. A. (2007b).
Delaunay refinement for piecewise smooth complexes.
Proceedings of the 18th ACM-SIAM Symposium on Discrete Algorithms, pages 1096–1105.

[Chew, 1989a] Chew, L. P. (1989a).
Constrained delaunay triangulations.
Algorithmica, 4(1):97–108.

[Chew, 1989b] Chew, L. P. (1989b).
Guaranteed quality triangular mesh.
Technical report, Cornell University.

[Chew, 1993] Chew, L. P. (1993).
Guaranteed-quality mesh generation for curved surfaces.
9th Symposium on Computational Geometry ACM, pages 274–280.

[Cox, 1972] Cox, M. G. (1972).
The numerical evaluation of b-splines.
Journal of the Institute of Mathematics and its Applications, 10(2):134–149.

[de Boor, 1972] de Boor, C. (1972).
On calculating with b-splines.
Journal of Approximation Theory, 6:50–62.

[de Casteljau, 1959] de Casteljau, P. (1959).
Outillage methodes calcul.
Technical report, André Citroën Automobiles SA, Paris.

[de Casteljau, 1963] de Casteljau, P. (1963).
Courbes et surfaces à pôles.
Technical report, André Citroën Automobiles SA, Paris.

[Delaunay, 1934] Delaunay, B. N. (1934).
Sur la sphère vide.
Izvestia Akademii Nauk SSSR, 7:793–800.

[Edelsbrunner and Shah, 1994] Edelsbrunner, H. and Shah, N. R. (1994).
Triangulating topological spaces.
10th ACM Symposium on Computational Geometry, pages 285–292.

[Edelsbrunner and Tan, 1993] Edelsbrunner, H. and Tan, T. S. (1993).
An upper bound for conforming delaunay triangulations.
Discrete and computational geometry, 10(2):197–213.

[Farin, 2002] Farin, G. (2002).
A history of curves and surfaces in cagd.
Computer Aided Geometric Design.

[Forrest, 1972] Forrest, A. R. (1972).
Interactive interpolation and approximation by bézier polynomials.
The Computer Journal, 15(1):71–79.

[Frederick et al., 1970] Frederick, C. O., Wong, Y. C., and Edge, F. W. (1970).
Two-dimensional automatic mesh generation for structural analysis.
International Journal for Numerical Methods in Engineering, 2:133–144.

[Frey, 1987] Frey, W. H. (1987).
Selective refinement: a new strategy for automatic node placement in graded triangular meshes.
International Journal for Numerical Methods in Engineering, 24(11):2183–2200.

[Gordon and Riesenfeld, 1974] Gordon, W. J. and Riesenfeld, R. F. (1974).
B-spline curves and surfaces.
Computer Aided Geometric Design, pages 95–126.

[Gosselin and Ollivier-Gooch, 2007] Gosselin, S. and Ollivier-Gooch, C. (2007).
Revisiting delaunay refinement triangular mesh generation on curve bounded domains.
Proceedings of the14th Conference of the Computational Fluid Dynamics Society of Canada.

[Guibas and Stolfi, 1985] Guibas, L. J. and Stolfi, J. (1985).
Primitives for the manipulation of general subdivisions and the computation of voronoi diagrams.
ACM Transactions on Graphics, 4(2):74–123.

[Kettner, 1999] Kettner, L. (1999).
Using generic programming for designing a data structure for polyhedral surfaces.
Theoretical Computer Science, 13:65–90.

[Labelle and Shewchuk, 2003] Labelle, F. and Shewchuk, J. R. (2003).

Anisotropic voronoi diagrams and guaranteed quality anisotropic mesh generation.
Proceedings of the 9th Symposium on Computational Geometry, pages 191–200.

[Lawson, 1977] Lawson, C. L. (1977).
Software for c1 surface interpolation.
Mathematical Software III, pages 161–194.

[Lee, 1978] Lee, D.-T. (1978).
Proximity and reachibility in the plane.
Technical report, Coordinated Science Laboratory, University of Illinois.

[Leray, 1945] Leray, J. (1945).
Sur la forme des espaces topologiques et sur les points fixes des représentations.
Journal de Mathématiques Pures et Appliquées, 54:95–167.

[Miller et al., 2003] Miller, G. L., Pav, S. E., and Walkington, N. J. (2003).
When and why ruppert's algorithm works.
12th International Meshing Roundtable Proceedings, pages 91–102.

[Miller et al., 1996] Miller, G. L., Talmor, D., Teng, S.-H., Walkington, N. J., and Wang, H. (1996).
Control volume meshes using sphere packing: generation, refinement and coarsening.
5th International Meshing Roundtable proceedings.

[Mitchell, 1994] Mitchell, S. A. (1994).
Cardinality bounds for triangulations with bounded minimum angle.
Proceedings of the 16th Canadian Conference on Computational Geometry, pages 326–331.

[Mitchell and Vavasis, 2000] Mitchell, S. A. and Vavasis, S. A. (2000).
Quality mesh generation in higher dimensions.
SIAM Journal on Computing, 29(4):1334–1370.

[Muller and Preparata, 1978] Muller, D. and Preparata, F. (1978).
Finding the intersection of two convex polyhedra.
Theoretical Computer Science, 7:217–236.

[Overhauser, 1968] Overhauser, A. W. (1968).
Analytic definition of curves and surfaces by parabolic blending.
Technical report, Ford Motor Company.

[Persson, 2005] Persson, P.-O. (2005).
Mesh generation for implicit geometries.
PhD thesis, Massachusetts Institute of Technology.

[Priest, 1991] Priest, D. M. (1991).
Algorithms for arbitrary precision floating point arithmetic.
Tenth Symposium on Computer Arithmetic, pages 132–143.

[Ruppert, 1994] Ruppert, J. (1994).
A delaunay refinement algorithm for quality 2-dimensional mesh generation.
Technical report, NASA Ames Research Center.

[Schoenberg, 1946] Schoenberg, I. J. (1946).
Contributions to the problem of approximation of equidistant data by analytic functions.
Quart. Applied Mathematics, 4:45–99; 112–141.

[Schönhardt, 1928] Schönhardt, E. (1928).
Über die zerlegung von dreieckspolyedern in tetraeder.
Mathematische Annalen, 98:309–312.

[Shewchuk, 1997] Shewchuk, J. R. (1997).
Delaunay refinement mesh generation.
PhD thesis, Carnegie Mellon University.

[Shewchuk, 1998] Shewchuk, J. R. (1998).
A condition guaranteeing the existence of higher-dimensional constrained delaunay triangulations.
14th Symposium on Computational Geometry ACM, pages 76–85.

[Shewchuk, 2000] Shewchuk, J. R. (2000).
Mesh generation for domains with small angles.
16th Symposium on Computational geometry ACM, pages 1–10.

[Shewchuk, 2002] Shewchuk, J. R. (2002).
Delaunay refinement algorithms for triangular mesh generation.
Computational Geometry Theory & Applications, 22:21–74.

[Szymczak and Rossignac, 2000] Szymczak, A. and Rossignac, J. (2000).
Grow & fold: compressing the connectivity of tetrahedral meshes.
Elsevier Computer Aided Geometry Design, 32:527–537.

[Timmer, 1980] Timmer, H. G. (1980).
Alternative representation for parametric cubic curves and surfaces.
Computer-Aided Design, 12:25–28.

[Voronoi, 1908] Voronoi, G. F. (1908).
Nouvelles applications des paramètres continus à la théorie de formes quadratiques.
Journal für die reine und angewandte Mathematik, 134:198–287.

[Watson, 1981] Watson, D. F. (1981).
Computing the n-dimensional delaunay tessellation with application to voronoi polytopes.
Computer Journal, 24(2):167–172.

[Weiler, 1985] Weiler, K. (1985).
Edge based data structures for solid modeling in a curved surface enviroment.
IEEE Computer Graphics and Applications, 55(1):21–40.

[Yamakawa and Shimada, 2000] Yamakawa, S. and Shimada, K. (2000).
High quality anisotropic tetrahedral mesh via ellipsoidal bubble packing.
9th International Meshing Roundtable, pages 263–273.

[Yokosuka and Imai, 2006] Yokosuka, Y. and Imai, K. (2006).

Guaranteed quality anisotropic mesh generation for domains with curves. *Proceedings of the European Workshop on Computational Geometry*, pages 27–29.

Index