

Universidade Federal do Ceará
Centro de Ciências
Departamento de Computação
Mestrado e Doutorado em Ciência da Computação

Geração Adaptativa de Malhas de Superfícies Paramétricas com Controle de Curvatura

Daniel Márcio Batista de Siqueira

DISSERTAÇÃO DE MESTRADO

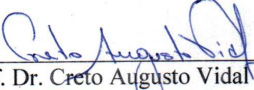
Fortaleza
Abril de 2009

***Geração Adaptativa de Malhas de Superfícies Paramétricas com
Controle de Curvatura***

Daniel Márcio Batista de Siqueira

Dissertação apresentada ao Curso de Mestrado em Ciência da Computação da
Universidade Federal do Ceará, como parte dos Requisitos para a obtenção do Grau
de Mestre em Ciência da Computação.

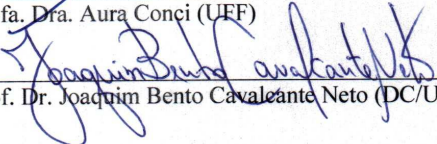
Composição da Banca Examinadora:



Prof. Dr. Creto Augusto Vidal (Presidente) DC/UFC



Profa. Dra. Aura Conci (UFF)



Prof. Dr. Joaquim Bento Cavaleante Neto (DC/UFC)

Aprovada em 24 de abril de 2009

A minha esposa Ana Lúcia e ao meu filho Matheus.

Geração Adaptativa de Malhas de Superfícies Paramétricas com Controle de Curvatura

por

Daniel Márcio Batista de Siqueira

Dissertação apresentada ao
Mestrado em Ciência da Computação
Universidade Federal do Ceará – UFC

Orientador: Creto Augusto Vidal
Co-orientador: Joaquim Bento Cavalcante Neto

Fortaleza, Abril de 2009

Sumário

Agradecimentos.....	i
Resumo.....	ii
Abstract.....	iii
Capítulo 1.....	1
Introdução e Motivação.....	1
1.1 Introdução.....	1
1.2 Motivação e descrição do problema.....	1
1.3 Organização do trabalho.....	4
Capítulo 2.....	5
Trabalhos Relacionados.....	5
2.1 Introdução.....	5
2.2 Geração de malha com controle de curvatura.....	5
2.3 Considerações finais.....	10
Capítulo 3.....	11
Estratégia de Adaptação.....	11
3.1 Introdução.....	11
3.2 Modelo inicial.....	12
3.2.1 Descrição geométrica.....	12
3.2.2 A malha inicial.....	13
3.3 Qualidade da malha.....	14
3.3.1 Curvaturas analíticas.....	15
3.3.2 Curvaturas discretas.....	16
3.3.3 Estimativa de erro baseada nas curvaturas.....	19
3.3.4 Estimativa global de erro.....	21
3.4 Adaptação das curvas.....	21
3.4.1 Inicialização da árvore binária.....	23
3.4.2 Rediscretização da árvore binária.....	23
3.4.3 Atualização da discretização da curva com base na árvore binária.....	25
3.5 Adaptação da malha.....	26
3.5.1. Geração da malha no interior do domínio por uma quadtree.....	27
3.5.2. Geração da malha da zona de transição.....	30
3.6 Considerações finais.....	32
Capítulo 4.....	33
Exemplos e Análise de Resultados.....	33
4.1 Introdução.....	33
4.2 Exemplos.....	33
4.2.1 Modelo com interior plano e bordas curvas.....	33
4.2.2 Modelo com curvatura alta e fronteira plana (bolha alta).....	35
4.2.3 Modelo com curvatura alta e base circular (bolha alta circular).....	37
4.2.4 Modelo com curvatura baixa e base circular (calota).....	40
4.2.5 Modelo com curvatura e borda não planar (sela).....	41
4.2.6 Superfície com um canto rebaixado.....	42
4.2.7 Modelo com dois patches adjacentes (rampa).....	43
4.2.8 Dois patches adjacentes, com grande curvatura (pneu).....	44
4.2.9 Modelo com vários patches com diversas curvaturas (nariz).....	46
4.2.10 Modelo com patches sem curvatura (cubo).....	48
4.3 Considerações finais.....	48
Capítulo 5.....	49

Conclusões e Trabalhos Futuros.....	49
5.1 Conclusões.....	49
5.2 Trabalhos futuros.....	50
Referências bibliográficas.....	51

Índice de ilustrações

Figura 3.1: Resumo do processo	11
Figura 3.2: Definição de uma curva de Hermite: ponto inicial P_0 , tangente no ponto inicial DP_0 , ponto final P_1 e tangente no ponto final DP_1	12
Figura 3.3: Orientação das curvas na definição de um patch.....	13
Figura 3.4: Discretização inicial de um patch.....	14
Figura 3.5: Malha inicial para o patch da Figura 3.4.....	14
Figura 3.6: Curvatura de uma superfície bi-paramétrica.....	15
Figura 3.7: A curvatura Gaussiana de um vértice interno.....	16
Figura 3.8: A curvatura Gaussiana em um vértice de borda.....	17
Figura 3.9: O ângulo γ para a curvatura discreta média.....	18
Figura 3.10: Exemplo onde a curvatura Gaussiana é nula numa região não plana.....	19
Figura 3.11: Cenários para análise de erro entre as curvaturas analítica e discreta.....	20
Figura 3.12: Refinamento de uma curva C_i	24
Figura 3.13: Árvore binária para a curva C_i da Figura 3.12.....	24
Figura 3.14: Exemplo de refinamento da fronteira.....	25
Figura 3.15: Discretização da borda de um patch.....	26
Figura 3.16: Quadrees: inicial, ajustada pelos erros e restrita.....	28
Figura 3.17: padrões para a malha do interior.....	30
Figura 3.18: Definindo as células da zona de transição e geração da malha interior por padrões.....	30
Figura 3.19: Malha final gerada.....	31
Figura 4.1: Modelo para a superfície de Hermite com interior plano.....	34
Figura 4.2: Geração da malha para o patch de Hermite.....	34
Figura 4.3: Geometria do modelo para o exemplo da bolha alta.....	35
Figura 4.4: Modelo para a bolha alta.....	36
Figura 4.5: Geração da malha para o modelo da bolha alta.....	36
Figura 4.6: No trio superior, a malha do passo 2 (105 nós e 192 elementos); no trio inferior, a malha do passo 3 (1024 nós e 2014 elementos). Vistas em 45°, superior e lateral.....	37
Figura 4.7: Modelo para bolha alta com base circular.....	38
Figura 4.8: Geometria, vista de cima, do modelo para o exemplo com base circular.....	38
Figura 4.9: Geração da malha para o modelo da bolha alta com base circular.....	39
Figura 4.10: Superfície da calota.....	40
Figura 4.11: Da esquerda para a direita: a malha inicial, passo 1 (17 nós e 24 elementos), passo 2 (73 nós e 120 elementos) e passo 3 (515 nós e 940 elementos).....	40
Figura 4.12: Modelo para a sela.....	41
Figura 4.13: Geração da malha da sela. (a) Malha inicial muito desrefinada (4 elem. e 5 nós). (b) Passo 1 (104 elem. e 65 nós). (c) Passo 2 (1936 elem. e 1033 nós).....	41
Figura 4.14: Modelo e geometria para superfície de canto rebaixado.....	42
Figura 4.15: Malhas inicial, passo 1, 2 e 3 para a superfície com ponta baixa. Acima, vista a 45°; abaixo, vista superior.....	42
Figura 4.16: Geometria para a rampa fornada pelos patches P_1 e P_2	43
Figura 4.17: Malha inicial e malha do passo 1.....	43
Figura 4.18: Malhas dos passos 2 e 3 da rampa.....	44
Figura 4.19: Geometria para o pneu.....	44
Figura 4.20: Trio superior, a malha inicial extremamente desrefinada para o pneu (8 elementos e 6 nós). Trio do meio, a malha para o passo 1 (48 elementos e 32 nós). No trio inferior, a malha para o Passo 4 (15424 elementos e 8010 nós).....	45
Figura 4.21: Modelo de um nariz para um avatar.....	46

Figura 4.22: Visão esquerda da malha inicial e dos passo 1 e 2 do modelo do nariz.....	46
Figura 4.23: O modelo do nariz. (a) A malha inicial (48 elementos e 60 nós). (b) Passo 1 (932 elementos e 587 nós). (c) Passo 2 (15704 elementos e 8371 nós).....	47
Figura 4.24: Da esquerda para direita: malha inicial refinada com o patch dividido em 16 regiões, seguida das malhas para os passos 1 e 2.....	48

Lista de Tabelas

Tabela 3.1: Tipos de superfície.....	16
Tabela 3.2: Estimativa para os novos tamanhos dos elementos.....	19

Agradecimentos

Aos meus orientadores, Prof. PhD. Creto Augusto Vidal e Prof. Dr. Joaquim Bento Cavalcante Neto, pelo incentivo, pela compreensão em momentos críticos, por seus ensinamentos, pelos conselhos e pressões nas horas certas.

A minha família, pelo incentivo e compreensão.

Ao Prof. Dr. Romildo José da Silva.

A todos os meus amigos do Crab (*Computer Graphics, Virtual Reality and Animation*), em especial ao Markos Freitas, Ricardo Lenz e Marco Diego.

A Universidade Federal do Ceará e ao Programa de Mestrado e Doutorado em Ciência da Computação.

Ao CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico) pelo apoio financeiro.

Resumo

Este trabalho apresenta uma estratégia hierárquica de geração de malhas adaptativas de superfícies paramétricas baseada em uma estrutura de árvore. O erro entre as curvaturas analítica e discreta, calculado para a superfície da malha, guia o processo adaptativo. Enquanto a curvatura analítica é uma representação matemática que modela o domínio, a curvatura discreta é uma aproximação dessa curvatura e depende diretamente da configuração da malha utilizada. A estratégia apresentada possui as seguintes propriedades: é capaz de refinar e desrefinar as diversas regiões da malha; garante a compatibilidade entre regiões vizinhas; considera a contribuição dos erros locais a fim de garantir uma boa qualidade global para a malha e funciona para qualquer tipo de geometria de superfície paramétrica, uma vez que todo o processo se dá no espaço paramétrico.

Abstract

This work presents an hierarchical adaptive mesh generation for parametric surfaces based on a tree structure. The error measured between the analytical and discrete curvatures guides the adaptive process. While the analytical curvature is an mathematical representation that models the domain, the discrete curvature is an approximation of that curvature and depends directly on the used mesh configuration. The presented strategy possess the following proprieties: it is able to refine and coarsen regions of the mesh; it ensures compatibility between neighboring regions; it considers the contribution of the local error measures to ensure good global mesh quality and it works with any type of parametric surface geometry, since the process is performed in the parametric space.

Capítulo 1

Introdução e Motivação

1.1 Introdução

A primeira Seção deste Capítulo discute a importância das estratégias de geração de malhas adaptativas de superfície, descreve o problema abordado, os objetivos e contribuições deste trabalho. A Seção 1.3 apresenta sua organização geral.

1.2 Motivação e descrição do problema

Malhas triangulares constituem uma representação espacial de grande importância. Elas são amplamente utilizadas em diversas áreas, e possibilitam a manipulação e visualização de superfícies complexas com relativa facilidade. A geração de malhas de superfícies é um assunto que vem sendo estudado desde a década de 70, a partir de algoritmos propostos com base nas técnicas de geometria computacional e ainda hoje é um tópico de grande interesse na engenharia e na computação gráfica.

Dentre os principais aspectos levados em consideração na avaliação de uma técnica de geração de malhas de superfície, destacam-se o tempo de processamento para geração e a qualidade da malha gerada. O tempo de processamento depende muito da discretização desejada, ou seja, do número de elementos presentes nas diversas regiões do modelo. Já a qualidade da malha, leva em conta, a quantidade, a distribuição e a forma de seus elementos. Essa qualidade é fundamental para as mais diversas aplicações: em computação gráfica, por exemplo, a qualidade influencia diretamente o aspecto visual do modelo.

Por muitas vezes, dependendo da aplicação, é necessário economizar a quantidade de elementos utilizados na malha sem que ela perca sua qualidade. Para alcançar tal objetivo utiliza-se malhas adaptativas, que são aquelas que possuem maior concentração de elementos em regiões de maior curvatura e uma concentração mais baixa de elementos em regiões de menor curvatura. A adaptação de malhas visa melhorar o tamanho, a organização e disposição

dos elementos, facilitando o armazenamento ou transmissão da malha e processos como renderização, simulações etc. Na busca desses objetivos, o estudo da geração de malhas adaptativas encontra espaço para crescer e amadurecer. Assim, várias técnicas de geração adaptativa vêm sendo propostas nos últimos anos.

As técnicas de geração de malhas adaptativas procuram gerar a malha de acordo com algum critério adotado, modificando-a sempre que necessário. Dentre as modificações mais comuns destacam-se: o redimensionamento, inserção e remoção de elementos. Essas mudanças são aplicadas até que um critério de parada seja alcançado. Os critérios adotados para a modificação não devem ser caros computacionalmente. Algumas dessas técnicas estão relacionadas com análise de elementos finitos, guiadas pela estimativa de erro [0-4].

Contudo, o uso da análise dos elementos finitos como critério de adaptação da malha não é o foco deste trabalho. O custo computacional dessas análises seria desnecessariamente alto, quando o propósito deste trabalho é apenas gerar uma malha de boa qualidade para aplicações como renderização, por exemplo. No presente trabalho, adota-se uma abordagem de baixo custo e com boa convergência, que é a utilização de critérios geométricos [7-11]. Neste aspecto, considera-se a curvatura como um parâmetro importante ao se determinar a densidade da distribuição dos elementos. Quanto mais próxima da curvatura analítica estiver a discretização da malha, mais fiel esta será às características geométricas da superfície.

Seguindo esse conceito, este trabalho apresenta uma estratégia de geração hierárquica de malhas adaptativa de superfícies paramétricas baseadas na estrutura de árvore, acrescentando as vantagens de duas abordagens clássicas: o avanço de fronteira e a triangulação de Delaunay.

As estratégias de adaptação de malhas podem ser divididas em duas classes principais:

- A priori: procura-se gerar uma malha inicial ideal. Os custos concentram-se, portanto, na geração.
- A posteriori: pode-se partir de uma malha inicial grosseira e a partir dela, realiza-se as modificações necessárias para se chegar a malha otimizada.

Neste trabalho, adota-se uma estratégia a posteriori, onde o usuário fornece uma definição geométrica para o modelo, e o sistema se encarrega de, a partir dessa geometria e de uma primeira aproximação bem desrefinada, construir malhas cada vez mais próximas da definição matemática da superfície. A estratégia apresentada procura garantir as seguintes

propriedades: i) ser capaz de refinar e desrefinar as regiões da malha; ii) garantir uma boa transição entre regiões da malha com maior e menor refinamento; iii) garantir a compatibilidade entre as diferentes regiões; iv) considerar a contribuição do erro local para medir o erro global, a fim de garantir uma boa qualidade para a malha.

Tais propriedades são particularmente importantes para aplicações de realidade virtual e computação gráfica, uma vez que proporcionam uma boa qualidade na renderização por causa da distribuição apropriada dos elementos. Logo, a estratégia proposta redimensiona os elementos e os concentra em regiões de maior curvatura além de reduzir a densidade de elementos em regiões de menor curvatura, se necessário.

Uma boa distribuição dos elementos é uma questão muito importante em geração de malhas, especialmente para aplicações de computação gráfica e realidade virtual e, embora também possa ser obtida utilizando-se modeladores geométricos, tais como 3DStudio, Blender entre outros, pode tornar-se um processo muito demorado e demandar muita intervenção humana. Outra questão importante a ser considerada é que a simplificação correta da malha é crucial para aplicações em tempo real, onde a taxa de quadros (*frame rate*) desempenha um papel central. E, finalmente, a distribuição dos elementos em uma malha é muito importante em aplicações onde os níveis de detalhes (*level of details*) são frequentemente aplicados, como na simulação de multidões (*crowd simulation*) e muitos outros problemas, onde uma malha não-refinada pode ser usada quando o modelo está longe da câmara e uma malha muito mais refinada é necessária quando o modelo está próximo.

Para o processo de adaptação aqui apresentado, consideramos um modelo inicial composto por *patches* de Hermite e suas malhas. A malha inicial do modelo junto com sua geometria, servirão de entrada para o processo adaptativo, que gerará a malha final com melhor qualidade. É importante mencionar, entretanto, que a estratégia proposta funciona para qualquer tipo de superfície (Bezier, etc) desde que parametrizável, pois ela é genérica.

1.3 Organização do trabalho

Este trabalho está organizado em 5 Capítulos. O Capítulo 2 mostra alguns trabalhos relacionados, noções sobre alguns operadores para cálculo de curvatura existentes e algumas técnicas de geração de malhas de superfície utilizando controle de curvatura.

O Capítulo 3 apresenta uma visão geral da estratégia adotada, explicando cada fase de seu processo adaptativo. Também são mostrados os conceitos matemáticos empregados para o cálculo das curvaturas. Tais curvaturas são utilizadas no cálculo de estimativa de erro que guia o processo adaptativo.

O Capítulo 4 traz algumas resultados conseguidos ao gerar malhas utilizando-se a estratégia. O resultados mostram sua eficácia e robustez ao lidar com os mais diversos níveis de curvatura. No Capítulo 5, as conclusões a cerca do trabalho e da estratégia apresentada são apresentadas, bem como algumas sugestões de trabalhos futuros.

Capítulo 2

Trabalhos Relacionados

2.1 Introdução

A geração de malhas de superfícies é um tópico de grande interesse em aplicações de engenharia, computação gráfica e realidade virtual. Um dos principais aspectos levados em consideração por uma técnica de geração de malhas é a qualidade da malha obtida. A fim de alcançar malhas de boa qualidade, muitas técnicas de adaptação de malhas têm sido propostas. Algumas dessas técnicas estão relacionadas à adaptação de malhas por elementos finitos baseada em estimativa de erro. Outras técnicas adaptativas foram propostas para problemas de modelagem geométrica e são muito úteis em aplicações de computação gráfica e realidade virtual.

A Seção seguinte apresenta algumas noções sobre o cálculo da curvatura para malhas triangulares e algumas estratégias de geração de malhas que utilizam a curvatura como controle para a geração.

2.2 Geração de malha com controle de curvatura

Do ponto de vista puramente teórico, malhas triangulares não possuem curvatura, uma vez que todas as faces são planas e a curvatura não está propriamente definida ao longo das arestas e vértices já que a superfície não possui continuidade C^2 definida para arestas e vértices. Mas, ao levar em consideração que uma malha triangular é uma aproximação da superfície contínua, pode-se estimar as curvaturas mediante as informações presentes na malha. Sob essa perspectiva, muitos pesquisadores vêm trabalhando em diversas formas de estimar a curvatura em uma superfície discretizada.

Dentre as formas mais conhecidas para se estimar a curvatura de uma superfície tem-se a curvatura Gaussiana. Essa medida vem sendo extensivamente utilizada nos mais diversos campos tais como processamento de imagens, *computer aided geometric design*, robótica,

computação gráfica e muitos outros. Sendo uma invariante tão amplamente utilizada, seria natural a demanda pelos mais diversos operadores que estimassem, da forma mais aproximada, a curvatura Gaussiana a partir da discretização de uma superfície.

Surazhsky [12] e Magid [13] apresentam alguns dos algoritmos mais utilizados para o cálculo das curvaturas Gaussiana e média. Nesses trabalhos, tais algoritmos são apresentados, analisados e testados na busca de determinar qual a melhor alternativa para o cálculo das curvaturas. Os algoritmos são divididos nas seguintes abordagens: aproximação local (*local fitting*) destacando *paraboloid fitting* e o *circular fitting*, o esquema Gauss-Bonnet, também conhecido como esquema de déficit angular (*angular deficit scheme*), Watanabe-Belyaev e Taubin. Nas abordagens de aproximação local, uma função de encaixe (*fitting function*) que aproxima a superfície localmente em um determinado ponto é construída, e a partir dessa função a curvatura de Gauss é analiticamente calculada. Através de seus testes comparativos, os autores concluem que os melhores algoritmos para o cálculo da curvatura Gaussiana são aqueles que empregam esquema de Gauss-Bonnet.

Xu [14] analisa a convergência do esquema de Gauss-Bonnet, que foi apresentado e explorado em [12] e [13]. A convergência desse esquema de discretização foi considerada por Meek e Walton [15], onde estes concluíram que o esquema não convergiria. Surpreendentemente, um dos métodos mais utilizados na literatura não apresentaria resultados muito precisos. Contudo, Xu demonstra em seu trabalho que se uma triangulação de superfície é obtida pela amostragem e quantização de uma superfície paramétrica, então sob certas condições, a aproximação do esquema de Gauss-Bonnet converge sob uma taxa quadrática. Logo, os resultados obtidos por Xu reforçam as conclusões alcançadas pelos experimentos de Surazhsky [12] e Magid [13].

Kim [10] apresenta uma métrica de erro visando gerar diversas malhas com refinamentos diferentes com a finalidade de empregá-las em aplicações que envolvem níveis de detalhes (*levels of details*). Essa métrica utiliza o cálculo das curvaturas discretas como critério de simplificação, os operadores utilizados seguem o esquema de Gauss-Bonnet e podem ser empregados para o cálculo das curvaturas tanto para os nós localizados no interior da malha quanto para aqueles localizados em uma borda da mesma. Devido a essa característica, e por seguirem o esquema de Gauss-Bonnet, tais operadores foram escolhidos para o cálculo das curvaturas discretas no presente trabalho.

Meyer [16] apresenta um método para estimar atributos geométricos de uma superfície discreta, tais como curvaturas e vetores normais. Seu trabalho define e deriva atributos diferenciais de primeira e segunda ordem, tais como vetores normais, curvaturas principais, curvatura Gaussiana, curvatura média, e direções principais para superfícies lineares e malhas triangulares arbitrárias. Os operadores apresentados são generalizados a fim de serem utilizados em *2-manifolds* e *3-manifolds*, numa dimensão arbitrária, oferecendo ferramentas para suavização de campos vetoriais e volumes de dados. Meyer segue o esquema de Gauss-Bonnet, e sugere um operador de cálculo de curvatura Gaussiana baseado na área de Voronoi da vizinhança de um determinado nó da malha.

Lau e Lo [17,18] apresentam uma técnica de geração de malhas de elementos finitos, utilizando a curvatura como medida para determinar o tamanho dos elementos e guiar a discretização. No primeiro deles [17], é proposto um esquema para geração automática de malhas triangulares, com distribuição arbitrária dos elementos sobre superfícies curvas e sem o uso de um espaço paramétrico. Os elementos são gerados na própria curva baseados na técnica de avanço de fronteira. A frente inicial é uma união disjunta de *loops* simples e fechados dos segmentos que aproximam as curvas de borda do domínio. Os segmentos do *loop* exterior são definidos no sentido anti-horário, enquanto os *loops* do interior no sentido horário. Durante o processo de geração da malha, um segmento base, AB, é retirado da frente Γ . O “melhor” triângulo é formado pelo segmento AB com um vértice C_r dentro da frente Γ . Em seguida a possibilidade de formar um triângulo melhor C_1 é examinada. Se a qualidade do triângulo ΔABC_1 , julgando-se, entre outros fatores, a curvatura, é melhor que a do triângulo ΔABC_r , então o triângulo ΔABC_1 é formado. Caso contrário, forma-se ΔABC_r . Buscando controlar o erro induzido pela curvatura da superfície, o ângulo entre o triângulo já existente em AB, ΔAOB , e o novo triângulo ΔABC é calculado e deve ser menor que um ângulo máximo tolerado Φ_{max} antes do novo triângulo ser aceito.

O segundo trabalho [18] traz uma abordagem semelhante, mas propondo um controle maior sobre a curvatura. Nesse trabalho, o mínimo raio principal da curvatura r_{min} em P é determinado com o objetivo de calcular o tamanho ideal dos elementos em um ponto $P = X(u, v)$. O tamanho do elemento desejado é calculado assumindo que a porção da superfície em P pode ser representada por uma esfera de raio r_{min} passando por P com o plano tangente coincidindo com o plano tangente da superfície.

Dyn, Hormann, Kim e Levin [19] propõem um algoritmo para obter uma malha triangulada ótima, utilizando a operação de *swapping* de arestas de forma sequencial com o objetivo de diminuir uma função de custo que mede a qualidade da malha. A função de custo usada pelos autores é desenvolvida de forma a levar em consideração a curvatura da superfície. Começando com uma triangulação inicial arbitrária, o algoritmo executa as trocas de aresta de maneira gulosa, reduzindo de forma maximal, a cada passo, a função de custo, terminando quando a função de custo alcançar o valor mínimo. Infelizmente, a estratégia gulosa pode levar a mínimos locais de forma que a malha ótima não é alcançada. A proposta do algoritmo é de servir como um pré-processo para outros algoritmos que irão operar sobre a malha tridimensional, oferecendo um bom ponto de partida que melhora a performance desses algoritmos.

Com relação a geração de malhas para superfícies, existem vários trabalhos na literatura que abordam esse problema. Miranda e Martha [8] descrevem um algoritmo para geração de malhas trianguladas de elementos finitos em superfícies arbitrárias com alta curvatura. A malha é gerada no espaço paramétrico e mapeada para o espaço 3D. O algoritmo possui três características: produz elementos de boa qualidade, evitando elementos degenerados ou com uma razão de aspecto ruim; garante a compatibilidade entre malhas de *patches* vizinhos, já que a malha de uma região é gerada de acordo com a discretização de suas curvas de bordo; apresentam uma transição suave entre regiões com alta variação no tamanho dos elementos. O processo de geração da malha é composto por quatro fases: discretização da fronteira; geração da *quadtrees*; aplicação do avanço de fronteira; suavização da malha.

Teixeira e Creus [11] apresentam um trabalho para geração automática de malhas não estruturadas que trabalha com recortes de superfícies paramétricas com domínios arbitrários e grande curvatura. As malhas em superfícies curvas são construídas com um método de avanço de fronteira (*advancing front*) que usa uma malha de fundo feita através de uma subdivisão adaptativa por *quadtrees*, para determinar o tamanho dos elementos em função da curvatura local. O algoritmo trabalha no espaço paramétrico, e é controlado pela métrica do espaço 3D em todos os seus passos: discretização do contorno, geração da malha de fundo, avanço de fronteira e suavização da malha. O algoritmo utiliza tolerâncias angulares aplicadas aos vetores normais das curvas e da superfície, de modo a levar em conta as curvaturas locais, tanto na geração do contorno, como na malha de fundo. O processo está organizado em quatro

etapas: determinação de sub-domínios; discretização das linhas de contorno; geração da malha de fundo, para estabelecer o tamanho dos elementos, em função da curvatura e do contorno, através da subdivisão do domínio; geração da malha não estruturada com o método de avanço de fronteira; suavização da malha.

Lee e Joun [21] apresentam uma abordagem para geração e suavização de malhas adaptativas para superfícies. Seu método começa com uma malha triangular grosseira, que vai sendo modificada de modo a obter a densidade desejada para a distribuição dos elementos. Várias transformações incluindo *splitting*, *collapsing* e *swapping* de arestas e vértices são empregadas para geração de novos triângulos e para garantir a qualidade da malha. Essas operações são propostas de forma a respeitar o critério de Delaunay e manter a consistência topológica. Para reduzir os inevitáveis erros que surgem durante a rediscritização da malha, esta é interpolada por *patches* de Bezier. A curvatura é considerada um parâmetro geométrico importante na determinação da densidade da distribuição dos elementos. O usuário especifica uma função de densidade que vai guiar a distribuição dos elementos da malha. Esta função é identificada como uma combinação de funções primitivas de densidade definidas por geometrias como esferas, cilindros e canos. A curvatura, multiplicada por um peso é adicionada à função especificada pelo usuário. Wang [22] , Glut e Jurczyk [23] seguem uma abordagem similar.

Souza [20] propõe uma estratégia adaptativa de geração de malhas que serve de base para o presente trabalho. Contudo, os operadores de curvatura discreta escolhidos não eram eficientes nos cálculos efetuados nas curvas de borda dos *patches*. Isso obrigava sua estratégia a utilizar *patches* auxiliares para calcular a curvatura nas bordas, o que aumentava o custo computacional da estratégia, além de gerar elementos de má qualidade nessas regiões. Esse trabalho também analisava os cenários possíveis para as curvaturas discretas e analíticas. O resultado dessa análise ditava o rumo que a estratégia devia tomar: refinar ou desrefinar uma determinada região. O presente trabalho sugere operadores mais eficientes e que trabalham nas bordas, eliminando a necessidade desses *patches* auxiliares. Além disso, os cenários possíveis para as curvaturas discretas e analíticas foram revisitadas, reavaliando-se quais as melhores decisões a se tomar em cada caso.

2.3 Considerações finais

Este Capítulo mostrou a importância das malhas adaptativas para aplicações de realidade virtual e computação gráfica e como a qualidade da malha pode estar diretamente ligada ao resultado desejado. Também apresentou alguns trabalhos da literatura que apresentam métodos para o cálculo de curvatura, a importância do cálculo dessa curvatura e algumas técnicas de geração de malhas que fazem uso desse cálculo para conseguir malhas de boa qualidade.

Capítulo 3

Estratégia de Adaptação

3.1 Introdução

Para dar início ao processo adaptativo é necessário um modelo, o qual é composto pela descrição geométrica dos vários *patches* que compõem a superfície a ter sua malha gerada juntamente com as malhas iniciais de cada um desses *patches*. A estratégia assume que essa superfície é composta por *patches* paramétricos, tais como *patches* de Coons, os quais são delimitados por curvas paramétricas. A descrição geométrica desses *patches* é fornecida pelo usuário; o sistema encarrega-se de, com base nessa descrição, construir suas malhas iniciais. É importante mencionar que as malhas iniciais dos *patches* também podem ser fornecidas ao invés de serem geradas. Definidas as geometrias e as malhas iniciais dos *patches*, o sistema constrói iterativamente novas malhas mais refinadas, tornando a discretização da superfície cada vez mais próxima de sua definição matemática.

Como mostra a Figura 3.1, a estratégia consiste de duas fases por iteração: primeiro, dados os modelos geométricos dos *patches* e das curvas, bem como a malha inicial da iteração corrente, a qualidade da malha é mensurada com base nos cálculos dos erros entre as curvaturas analíticas e discretas; segundo, com base na malha inicial da iteração corrente, uma nova malha é gerada, pela adição de mais elementos em certas regiões, e remoção de elementos em outras regiões da malha inicial. O processo iterativo para quando um critério de

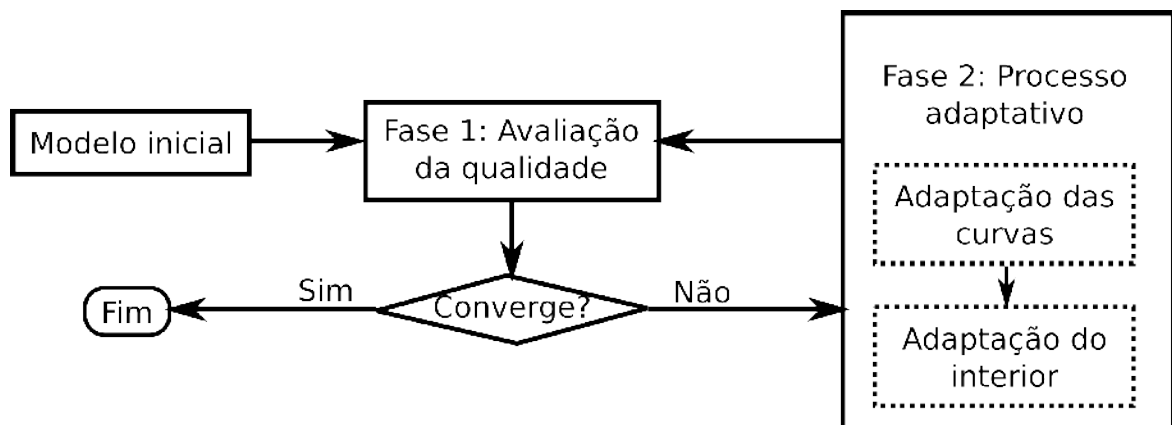


Figura 3.1: Resumo do processo adaptativo.

qualidade global predefinido é atingido. Na fase 2, as curvas são rediscritizadas primeiro e, somente após

as curvas, os *patches* são rediscritizados. Esse capítulo descreve em detalhes cada parte da estratégia adaptativa.

3.2 Modelo inicial

Como já mencionado, a descrição matemática dos *patches*, juntamente com a descrição das curvas delimitantes, definem a geometria do modelo. Essa geometria será utilizada em todos os passos da estratégia adaptativa. A descrição da geometria e uma malha inicial compõem o modelo inicial que dá início ao processo adaptativo. É importante ressaltar que embora o sistema possa construir sua própria malha inicial, a estratégia funciona para qualquer outra malha inicial fornecida.

3.2.1 Descrição geométrica

O modelo inicial é composto pela malha inicial e por sua geometria. Esta geometria é composta por diversos *patches* e estes por suas curvas delimitantes. Cada uma dessas curvas pode fazer parte de dois *patches* vizinhos. Sendo assim, a descrição da geometria deve ser feita de forma hierárquica: primeiro, vem a descrição das curvas; em seguida, a descrição dos *patches*.

As descrições das curvas e *patches* seguem suas definições matemáticas. Por exemplo, ao usar a estratégia com superfícies de Hermite, deve-se definir as quatro curvas delimitantes como curvas de Hermite, os pontos dos quatro cantos e os vetores *twists* desses quatro cantos. Cada curva de Hermite é definida por seus pontos extremos e as tangentes nesses pontos, seguindo a orientação da curva. É importante ressaltar, entretanto, que a estratégia funciona

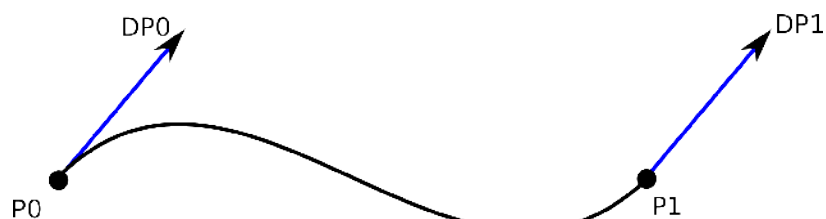


Figura 3.2: Definição de uma curva de Hermite: ponto inicial P0, tangente no ponto inicial DP0, ponto final P1 e tangente no ponto final DP1.

para qualquer tipo de curvas e *patches*, desde que eles possam ser parametrizados de [0 a 1], já que ela trata curvas e *patches* de uma forma genérica.

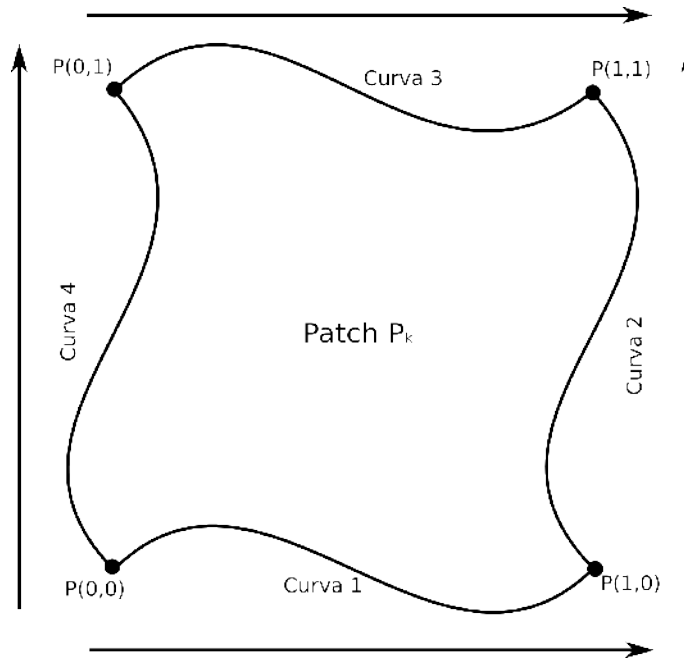


Figura 3.3: Orientação das curvas na definição de um *patch*.

O modelo inicial mostrado na Figura 3.3 descreve um *patch* P_k e suas curvas C_i . Esse modelo é utilizado ao longo deste trabalho para ilustrar a estratégia adaptativa.

3.2.2 A malha inicial

As malhas iniciais de todos os *patches* compõem a malha inicial do modelo inteiro. Essa é a primeira aproximação da malha desejada. Se P_k é um *patch* e C_i representa suas curvas delimitantes, a geração da malha inicial M_k começa com a discretização de C_i em segmentos. Logo, a malha inicial é gerada com base nesses segmentos, garantindo assim a compatibilidade da malha nas bordas de *patches* vizinhos.

O número de elementos que formam a malha M_k depende da discretização das curvas delimitantes de P_k . Na Figura 3.4, as curvas foram discretizadas em três regiões cada, o que determina a discretização do *patch* em nove regiões. A subdivisão das curvas e *patches* é feita no espaço paramétrico e mapeada para o espaço tridimensional.

Na geração da malha inicial, para cada região do *patch* determina-se seu vértice médio. Este vértice, juntamente com os quatros vértices dos cantos da região, forma um “guarda-chuva” (*triangle fan*) de quatro triângulos correspondendo àquela região. O ponto

médio é calculado primeiramente no espaço paramétrico, e em seguida mapeado para o espaço tridimensional. A Figura 3.5 mostra a malha inicial criada para o *patch* da Figura 3.4 seguindo esse princípio.

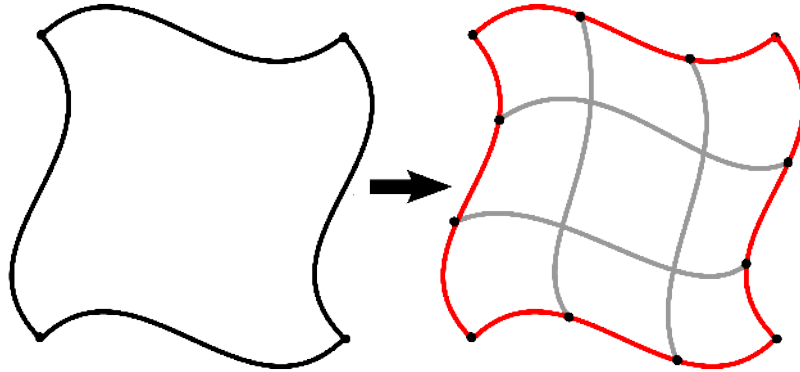


Figura 3.4: Discretização inicial de um *patch*.

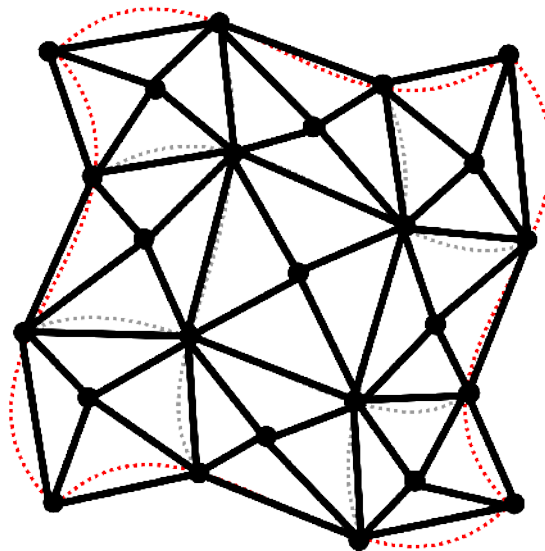


Figura 3.5: Malha inicial para o *patch* da Figura 3.4.

No entanto, novamente vale ressaltar que não importa como a malha inicial seja definida, a estratégia se encarregará de adaptá-la seguindo a geometria do modelo.

3.3 Qualidade da malha

A qualidade de toda a malha gerada durante o processo adaptativo é avaliada de acordo com um critério de erro. Neste trabalho, busca-se representar o domínio em questão da melhor forma possível, utilizando um critério de erro que não seja caro computacionalmente e que seja adequado. Portanto, adota-se um critério geométrico que calcula em cada nó da malha o

erro entre as curvaturas analítica e discreta da superfície. Se a distribuição geral do erro ficar abaixo de uma precisão desejada, a qualidade da malha é considerada boa e o processo adaptativo termina. O erro local indica se a malha deve ser refinada ou desrefinada em uma determinada região.

3.3.1 Curvaturas analíticas

Segundo Rogers & Adams [26], ao pegar-se um vértice v qualquer sobre um *patch*, e por ele definir um plano normal em relação a superfície, a interseção do plano com a superfície define uma curva que permite calcular a curvatura analítica K_a do *patch* em v . Ao girar-se esse plano, tendo a normal de v como eixo de rotação, a medida da curvatura muda. Mas somente encontrar-se-á uma única direção onde a interseção do plano com a superfície define uma curvatura máxima K_{max} , e outra única direção que define uma curvatura mínima K_{min} . Estas duas curvaturas são chamadas curvaturas principais.

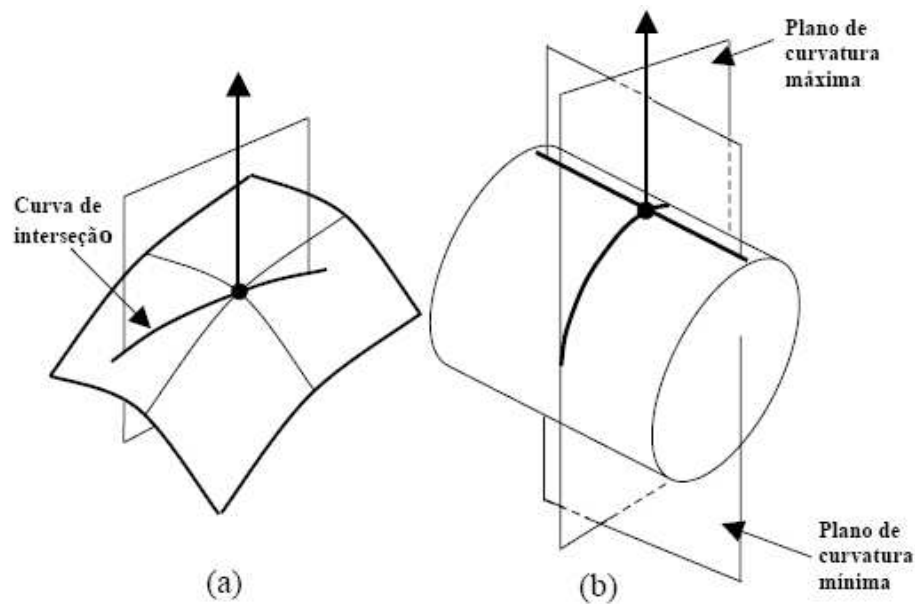


Figura 3.6: Curvatura de uma superfície bi-paramétrica.

As duas curvaturas principais são de particular interesse, pois sua combinação dá as curvaturas média (H) e Gaussiana (K):

$$H = \frac{K_{min} + K_{max}}{2} \quad (3.1)$$

$$K = K_{min} \cdot K_{max} \quad (3.2)$$

Dill [27] mostra que para superfícies bi-paramétricas, as curvaturas média e Gaussiana são dadas pelas seguintes equações:

$$H = \frac{A \cdot |Q_w|^2 - 2 \cdot B \cdot Q_u \cdot Q_w + C \cdot |Q_u|^2}{2 \cdot |Q_u \times Q_w|^2} \quad (3.3)$$

$$K = \frac{A \cdot C - B^2}{|Q_u \times Q_w|^4} \quad (3.4)$$

onde

$$\begin{aligned} A &= [Q_u \times Q_w] \cdot Q_{uu} \\ B &= [Q_u \times Q_w] \cdot Q_{uw} \\ C &= [Q_u \times Q_w] \cdot Q_{ww} \end{aligned} \quad (3.5)$$

e o subscrito indica a notação para derivadas parciais, i.e. $Q_u = \delta Q / \delta u$, etc.

Como mostra a Tabela 3.1, o sinal da curvatura Gaussiana serve para caracterizar a forma local da superfície: elíptica, hiperbólica, cilíndrica ou cônica. Por esse motivo, neste trabalho, é dada a preferência ao uso da curvatura Gaussiana em primeiro lugar pois ao conhecer o valor da curvatura Gaussiana em um determinado vértice de uma região de um *patch*, pode-se inferir a forma da superfície na região visualizada. Isso irá ajudar na compreensão dos resultados.

Tabela 3.1: Tipos de superfície.

$k_{min} \cdot k_{max}$	K	Forma
Mesmo sinal	> 0	Elíptica
Sinais contrários	< 0	Hiperbólica (ponto de sela)
Uma ou ambas iguais a zero	0	Cilíndrica, Cônica ou Plano

3.3.2 Curvaturas discretas

Calcular as curvaturas analíticas requer a formulação matemática para o *patch*, por outro lado, avaliar a curvatura discreta requer informações locais sobre a malha. Ou seja, as curvaturas discretas são calculadas com base num dado nó da malha e baseia-se na configuração de seus triângulos adjacentes. As curvaturas discretas, Gaussiana e Média, utilizadas no presente trabalho são computadas pelos operadores de curvatura apresentados por Kim [10]. Esses

operadores seguem o esquema de Gauss-Bonnet, qual já visto no Capítulo 2 tratar-se da melhor abordagem para o cálculo da curvatura Gaussiana.

A curvatura discreta Gaussiana K para um nó interior da malha é expressa por

$$K = \frac{2\pi - \sum_{i=1}^n \Phi_i}{\frac{1}{3} \cdot A} \quad (3.6)$$

onde A é a soma das áreas de cada face e Φ_i denota o ângulo do nó considerado. É fácil ver que à medida que a região do nó se aproxima de um plano, a curvatura Gaussiana se aproxima de zero (Figura 3.7).

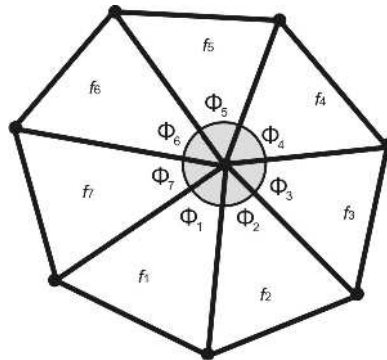


Figura 3.7: A curvatura Gaussiana de um vértice interno.

Para um nó de borda, a curvatura Gaussiana é definida pela seguinte equação

$$K = \frac{\pi - \sum_{i=1}^n \Phi_i}{\frac{1}{3} \cdot A} \quad (3.7)$$

onde A novamente representa a soma das áreas dos triângulos e Φ_i , o ângulo naquele nó (Figura 3.8).

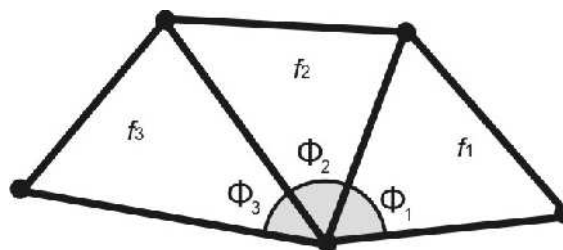


Figura 3.8: A curvatura Gaussiana em um vértice de borda.

A curvatura discreta Média é dada por

$$H = \frac{\sum m(e_i)}{\frac{1}{3} \cdot A} \quad (3.8)$$

onde e_i representa uma aresta incidente em um nó, e $m(e_i)$ é uma função da aresta e_i que retorna o ângulo entre os dois vetores normais das faces adjacentes em e_i (Figura e equação 3.9).

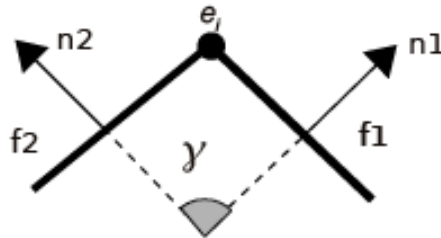


Figura 3.9: O ângulo γ para a curvatura discreta média.

$$m(e_i) = \begin{cases} \gamma & , \text{ se } e_i \text{ for convexa} \\ 0 & , \text{ se } e_i \text{ for plana} \\ -\gamma & , \text{ se } e_i \text{ for côncava} \end{cases} \quad (3.9)$$

3.3.3 Estimativa de erro baseada nas curvaturas

O processo adaptativo consiste na discretização das curvas e dos *patches*. Essa discretização é feita com base na estimativa de erro da curvatura discreta em relação a curvatura analítica avaliada em cada nó da malha. Se o erro calculado indica uma grande discrepância entre as duas curvaturas, sugere-se que a malha seja localmente refinada ou desrefinada, com base num parâmetro de tamanho, h .

A Tabela 3.2 ilustra os possíveis cenários considerados neste trabalho e como determinar o novo parâmetro de tamanho para cada caso. A curvatura analítica é representada por K_a e K_d representa a curvatura discreta. A estimativa de erro é calculada com base na

curvatura Gaussiana, uma vez que ela descreve melhor a geometria local da superfície. A curvatura média só será utilizada nos casos em que a curvatura Gaussiana seja zero.

No primeiro cenário, Ka é aproximadamente igual a Kd . Se esses valores não forem próximos a zero ($Ka \not\rightarrow 0$), a malha captura bem a curvatura local da superfície e nenhum refinamento se faz necessário ($h_{novo} = h_{velho}$). No entanto, se a curvatura analítica Ka for próxima a zero ($Ka \rightarrow 0$), a superfície é localmente plana e um desrefinamento da malha se faz necessário ($h_{novo} = h_{velho} * f, f > 1$).

Tabela 3.2: Estimativa para os novos tamanhos dos elementos.

$Ka \approx Kd$ ($Ka / kd \rightarrow 1$)	$Ka \rightarrow 0$	$h_{novo} = h_{velho} * f$	<i>desrefinamento</i>
	$Ka \not\rightarrow 0$	$h_{novo} = h_{velho}$	<i>parar</i>
$Ka \gg Kd$	$Ka \rightarrow 0$	$h_{novo} = h_{velho} / f$	<i>refinamento</i>
	$Ka \not\rightarrow 0$	$h_{novo} = h_{velho} / f$	<i>refinamento</i>
$Ka \ll Kd$	$Ka \rightarrow 0$	$h_{novo} = h_{velho} / f$	<i>refinamento</i>
	$Ka \not\rightarrow 0$	$h_{novo} = h_{velho} / f$	<i>refinamento</i>

Nos outros cenários, a discrepância entre Ka e Kd é grande. Então, quando a curvatura analítica Ka for próxima de zero ($Ka \rightarrow 0$), a malha não é suficientemente refinada para capturar a planaridade local da superfície e deve ser refinada. Quando Ka não for próxima de

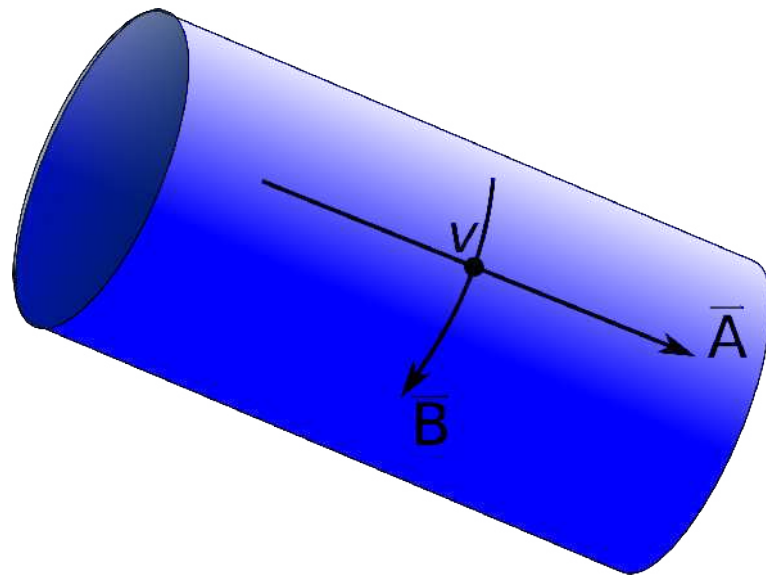


Figura 3.10: Exemplo onde a curvatura Gaussiana é nula numa região não plana.

zero ($Ka \rightarrow 0$), a malha também não é boa o suficiente para capturar a curvatura correta e deve ser refinada. O fator f usado na discretização é determinante para a taxa de convergência do processo. Quanto maior for o valor de f , mais rápido será a taxa de convergência.

No caso de o cálculo da curvatura Gaussiana apresentar resultado nulo, passa-se a considerar a curvatura média pelo simples fato de que uma curvatura Gaussiana nula não implica em uma região localmente plana. Observando a Figura 3.10, é possível ver que a curvatura Gaussiana no vértice v é nula na direção \vec{A} , já na direção \vec{B} ou em qualquer outra direção, ela é diferente de zero.

A Figura 3.11 ilustra uma visão de um corte longitudinal onde pode-se observar a superfície e a malha que a representa para os casos em que a discrepância entre as curvaturas é alta.

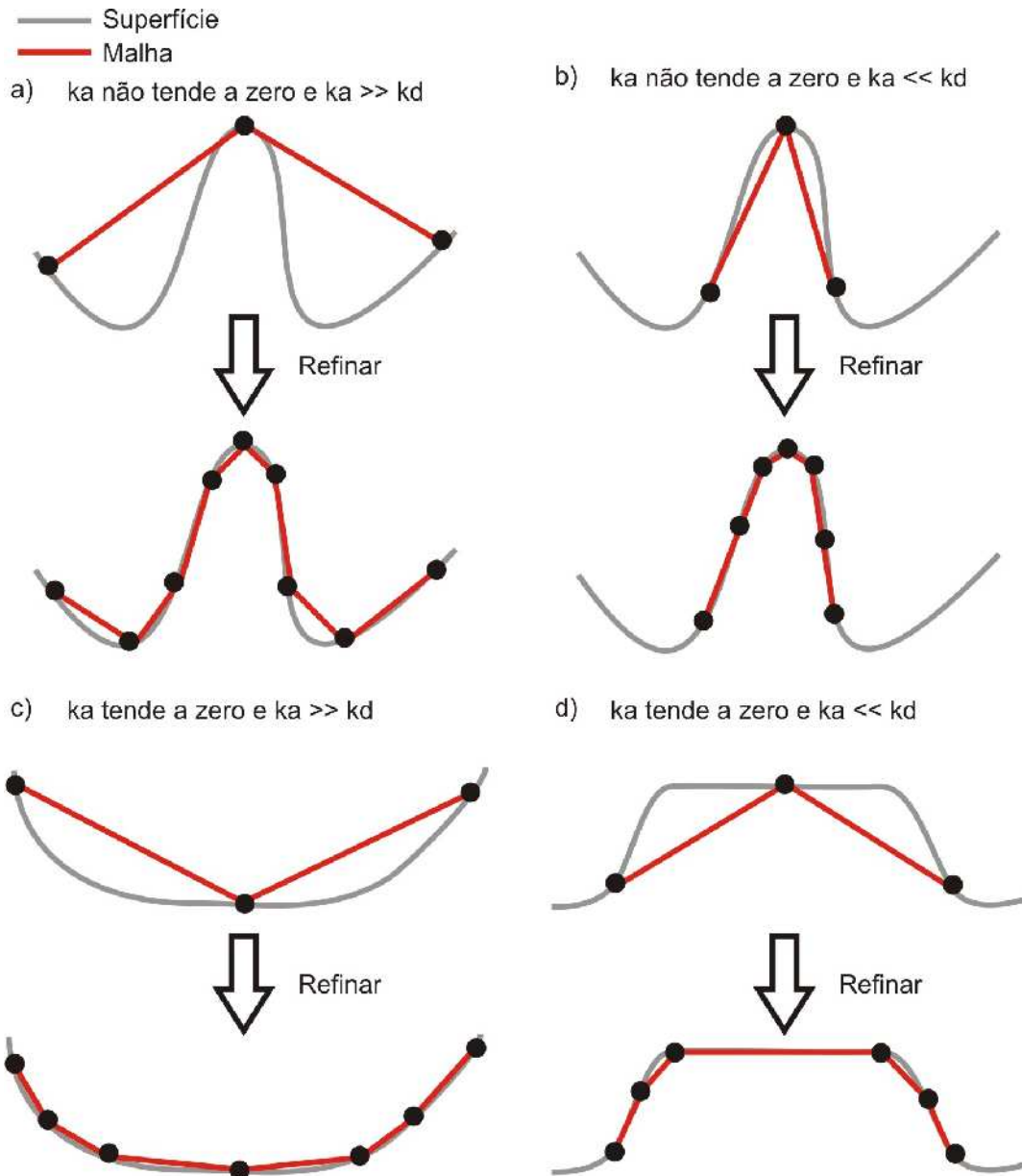


Figura 3.11: Cenários para análise de erro entre as curvaturas analítica e discreta.

3.3.4 Estimativa global de erro

Embora a discretização seja baseada nos erros locais das curvaturas, é necessária uma medida global que guie todo o processo iterativo. Quando a qualidade global é alcançada o processo para. Essa medida implica o uso de uma estimativa global de erro, η_{global} , o qual é calculado, neste trabalho, por

$$\eta_{global} = \frac{\sum_{j=1}^{N_v} \eta_j}{N_v}, \quad (3.10)$$

onde N_v é o número de nós de toda a malha; e η_j é o valor absoluto da diferença relativa entre as curvaturas analítica e discreta em um nó, cujo cálculo é dado por

$$\eta_j = \frac{|(K_a - K_d)|}{|K_a|}. \quad (3.11)$$

A malha possui uma boa qualidade quando $\eta_{global} < \varepsilon$. A escolha de um ε que seja considerado razoável é uma questão de julgamento. Portanto, neste trabalho, nenhuma sugestão sobre qual valor de ε utilizar é feita. Apenas é mostrado que o valor do erro decresce à medida que a malha é melhorada, indicando a convergência do processo iterativo.

3.4 Adaptação das curvas

O primeiro passo para a estratégia proposta requer que as curvas delimitantes de cada *patch* sejam discretizadas de forma independente e antes da discretização do domínio. Isso é uma das grandes vantagens da estratégia proposta, uma vez que leva a uma discretização mais regular das fronteiras dos *patches*, consistente com as características geométricas da superfície próxima a essas curvas. Isso também é importante para garantir a compatibilidade da malha entre dois *patches* adjacentes. Essa estratégia foi primeiramente proposta para domínios bidimensionais em [3]; no entanto, no trabalho original, a discretização não era feita com base em critérios geométricos. A nova discretização do conjunto completo de curvas delimitantes define a base para a discretização do domínio, gerando uma nova malha para o modelo.

O procedimento de discretização das curvas delimitantes é uma versão unidimensional do procedimento baseado na *quadtree* usada para refinar o domínio [25]. Esse procedimento emprega uma técnica de enumeração espacial recursiva, associada a uma estrutura de dados de árvore binária, e adota um critério baseado no erro da curvatura para refinar ou desrefinar a discretização da curva, levando em conta as curvaturas características das faces adjacentes. Logo, em um dado nó da aproximação poligonal da curva, as curvaturas analítica e discreta são calculadas. A curvatura discreta considera todos os triângulos adjacentes ao nó, inclusive os que estão nas superfícies vizinhas. A curvatura analítica, por

outro lado, considera a máxima curvatura calculada para as superfícies vizinhas que compartilham a curva. O erro entre as medidas das curvaturas guia a forma como se determina o novo tamanho dos triângulos adjacentes àquele nó.

A discretização de cada curva é feita em três fases, como descrito a seguir, para uma curva genérica, e mostrada no Algoritmo 3.1.

Algoritmo 3.1: Construção da árvore binária

```

Inicialize a raiz da árvore binária como nó vazio
As coordenadas mínima e máxima do nó raiz recebem (0,1)
Calcule o comprimento da curva,  $L_{curva}$ 

para cada segmento da lista de vértices iniciais,  $S_k$ ,  $k \leftarrow 1$  até  $n_{seg}$  faça
{
    Calcule o tamanho do segmento no espaço tridimensional,  $L_{seg}$ 
     $h_{velho} \leftarrow L_{seg}$ 
    Calcule o centro de  $S_k$  no espaço tridimensional,  $C_{seg_k}$ 
    Calcule  $Ka$  em  $C_{seg_k}$ 
    Calcule  $Kd$  como a média das  $Kds$  dos vértices extremos de  $S_k$ 
        
$$kd = \frac{1}{2}(^{k-1}Kd + ^kKd)$$

    Calcule  $h_{novo}$  de acordo com a tabela 3.2
    Calcule  $h_{par} = \frac{h_{novo}}{L_{curva}}$ ,  $h_{par} \in [0,1]$ 
    Determine o parâmetro correspondente a  $C_{seg_k}$ ,  $u_k$ 
    Determine em que nó da árvore está  $u_k$ 
    enquanto (tamanho do nó >  $h_{par}$ ) faça
    {
        Subdivida o nó em dois filhos
        Incremente a profundidade da árvore
        Determine em que nó da árvore está  $u_k$ 
    }
}

```

3.4.1 Inicialização da árvore binária

A primeira fase consiste em inicializar a árvore binária que guiará a rediscritização da curva. Essa árvore é inicializada com as coordenadas paramétricas máxima e mínima (0, 1) da curva não discretizada. Isso é feito com o objetivo de permitir um processo de refinamento genérico que sirva para qualquer tipo de curva. A profundidade inicial da árvore é zero, uma vez que a mesma só possui um único nível nesse momento.

3.4.2 Rediscretização da árvore binária

Cada curva mantém o conjunto de nós adjacentes anterior, o qual é definido quando a malha inicial é gerada e é atualizado depois que uma nova malha é determinada a cada passo. Uma curva também mantém o conjunto de novos nós que são gerados durante a fase corrente. Portanto, esses dois conjuntos de nós devem coexistir até que uma nova malha seja definida. Esses conjuntos são ordenados de acordo com a parametrização da curva.

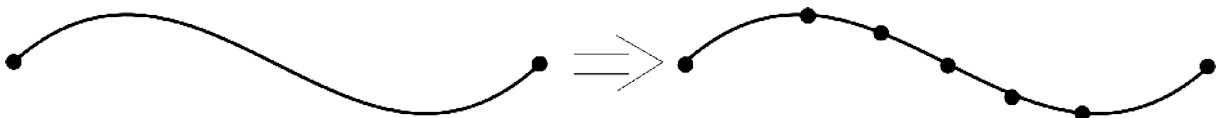
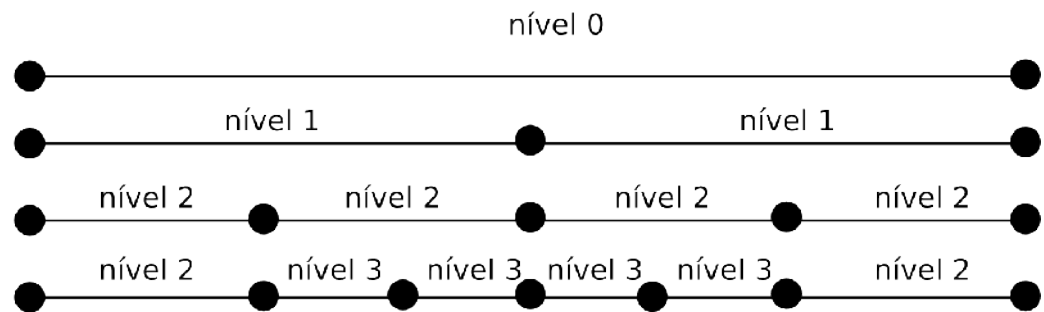


Figura 3.12: Refinamento de uma curva C_i .

A rediscretização da curva pela árvore binária começa percorrendo o conjunto de nós da discretização anterior. No primeiro passo, o conjunto terá as coordenadas paramétricas

a)



b)

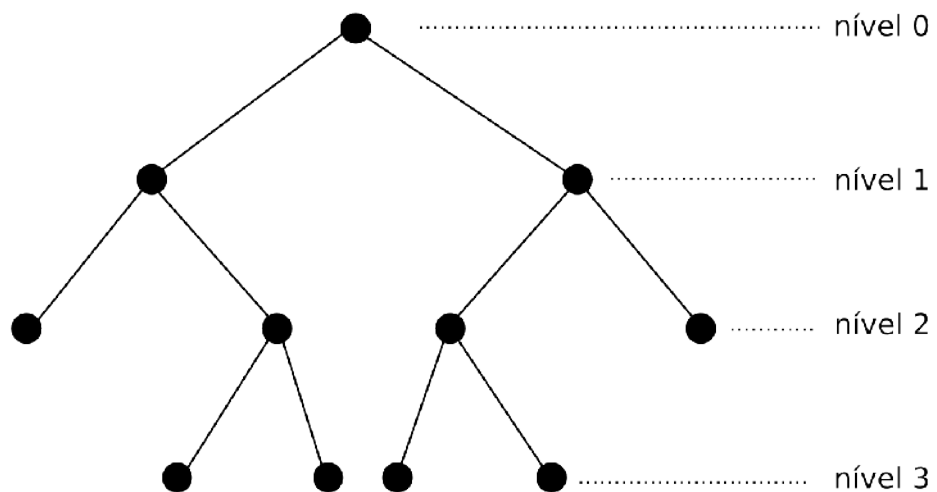


Figura 3.13: Árvore binária para a curva C_i da Figura 3.12.

mínima e máxima $(0, 1)$. Cada dois nós consecutivos representam um segmento da curva.

Cada folha da árvore armazena as coordenadas paramétricas mínima e máxima desse segmento.

Para cada um desses segmentos, calcula-se o seu comprimento no espaço tridimensional, e denomina-se este valor de h_{velho} . Em seguida, calcula-se o ponto médio desse segmento, também no espaço tridimensional. Para esse ponto médio, calcula-se sua curvatura analítica através da formulação matemática da superfície; como não se pode calcular sua curvatura discreta diretamente, uma vez que o vértice na superfície ainda não possui um nó que o represente na malha e, portanto, não há elementos incidentes sobre ele, o valor dessa curvatura é estimado pela média aritmética das curvaturas discretas dos vértices extremos do segmento em questão. De posse do tamanho do segmento h_{velho} e das curvaturas analítica e discreta, calcula-se um novo tamanho, chamado de h_{novo} , de acordo com a Tabela 3.2. Então, divide-se o valor de h_{novo} pelo comprimento total da curva. Essa razão dá um número entre 0 e 1 o qual é denominado de h_{par} .

O próximo passo é mapear o ponto médio para o espaço paramétrico, encontrando sua coordenada paramétrica t ($0 \leq t \leq 1$). Com o valor de t , localiza-se qual folha da árvore binária possui o intervalo que contém t , lembrando que cada folha representa um segmento da curva e guarda as coordenadas paramétricas máxima e mínima desse segmento. Compara-se o tamanho dessa folha com h_{par} , e caso este seja maior, subdividi-se essa folha recursivamente até que seu tamanho seja menor que o valor de h_{par} .

Depois de percorrer todos os segmentos da curva, a árvore binária resultante reflete o novo refinamento de acordo com o erro entre as curvaturas, que é calculado para cada nó da aproximação poligonal da curva C_i . Cada folha da árvore irá gerar um novo segmento, e esses segmentos serão os lados dos triângulos adjacentes à curva quando o domínio for rediscritizado. A Figura 3.12 ilustra um exemplo de refinamento de uma curva C_i e a Figura 3.13 ilustra a árvore binária correspondente.

3.4.3 Atualização da discretização da curva com base na árvore binária

Quando a árvore binária de uma curva C_i é finalmente redefinida, a nova discretização é incluída na curva (Algoritmo 3.2). Ao final desse processo, a nova discretização da curva é obtida de forma ordenada. O novo conjunto de nós é mantido e o conjunto antigo de nós é liberado.

Algoritmo 3.2: Atualizando a discretização da curva C_i .

```
Calcule as coordenadas 3D associadas a  $u = \theta$ 
Inclua essas coordenadas na estrutura da curva
para cada folha da árvore binária faça
{
    Obtenha a coordenada paramétrica máxima
    Calcule as coordenadas 3D correspondentes
    Inclua essas coordenadas na estrutura da curva
}
```

A Figura 3.14 mostra um exemplo de refinamento de fronteira, onde a figura à esquerda representa o modelo original e a figura à direita representa a discretização de borda para o primeiro passo. Como se pode observar nessa figura, locais onde a curvatura é mais alta foram mais discretizados.

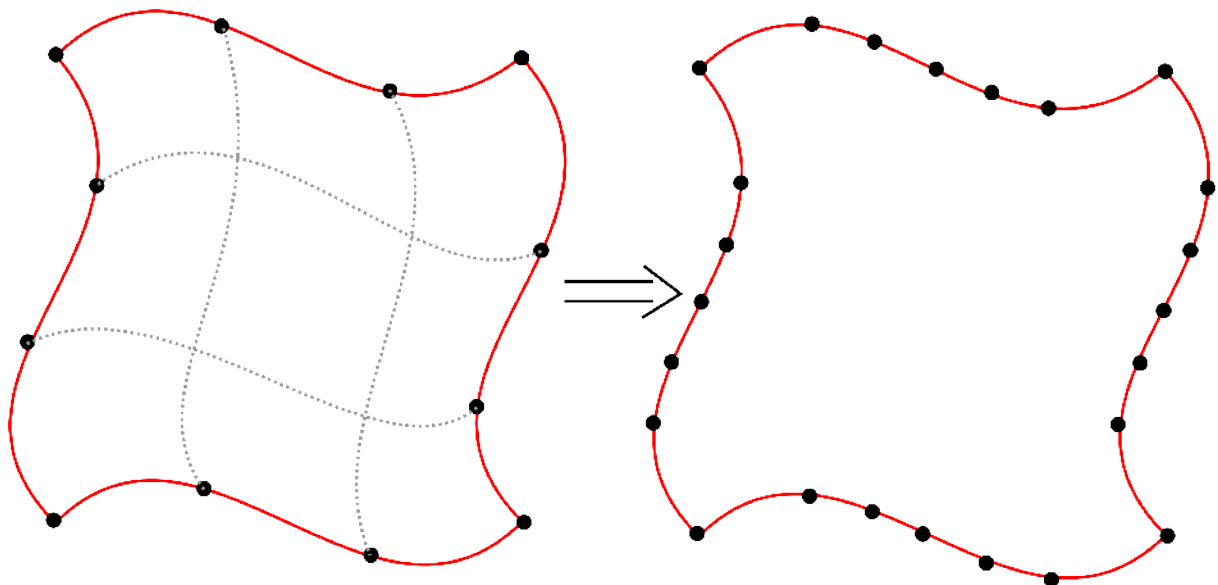


Figura 3.14: Exemplo de refinamento da fronteira.

3.5 Adaptação da malha

Após a discretização das curvas de borda, a próxima fase do processo adaptativo gera uma nova malha no domínio usando um procedimento baseado na *quadtree*. Entretanto, neste trabalho, a *quadtree* não será responsável pela geração da malha em todo o domínio. Uma zona de transição próxima à borda do *patch* será deixada de lado para ser discretizada usando uma técnica baseada na triangulação de Delaunay. Esse procedimento garante que a malha no interior do domínio seja compatível com a discretização das curvas delimitantes, evitando a criação de elementos degenerados, além de permitir a combinação das malhas de *patches* vizinhos.

A triangulação do interior feita pela *quadtree* é baseada no trabalho de Baehmann et al. [25], enquanto que os triângulos gerados na zona de transição próximas às curvas são gerados por uma técnica de avanço de fronteira, combinada com um critério de Delaunay. A técnica de avanço de fronteira foi modificada neste trabalho de forma a considerar as informações geradas pela *quadtree*, a fim de agilizar o processo e evitar que um triângulo seja gerado por um nó longe de sua aresta base. A Figura 3.15 ilustra a discretização das curvas delimitantes para um dado *patch* P_k . Uma vez que o procedimento é baseado na *quadtree* no espaço bidimensional paramétrico, é necessário mapear cada nó do espaço tridimensional para o espaço paramétrico de cada *patch*. No entanto, ao final da geração da malha, é necessário mapear os nós dos triângulos gerados no espaço paramétrico de volta ao espaço tridimensional.

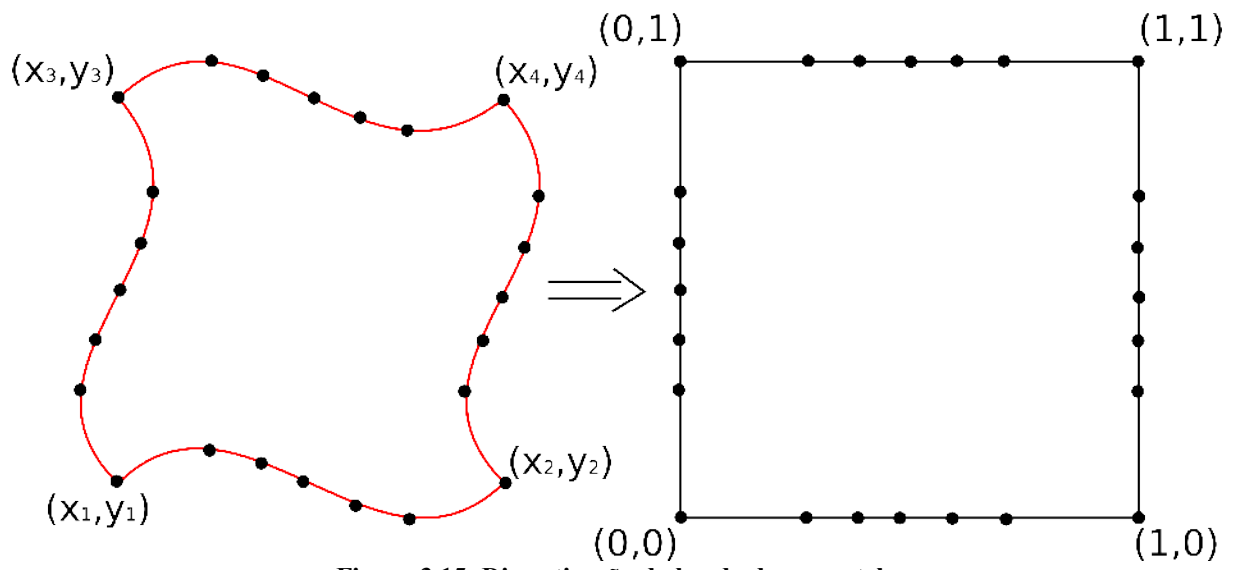


Figura 3.15: Discretização da borda de um patch.

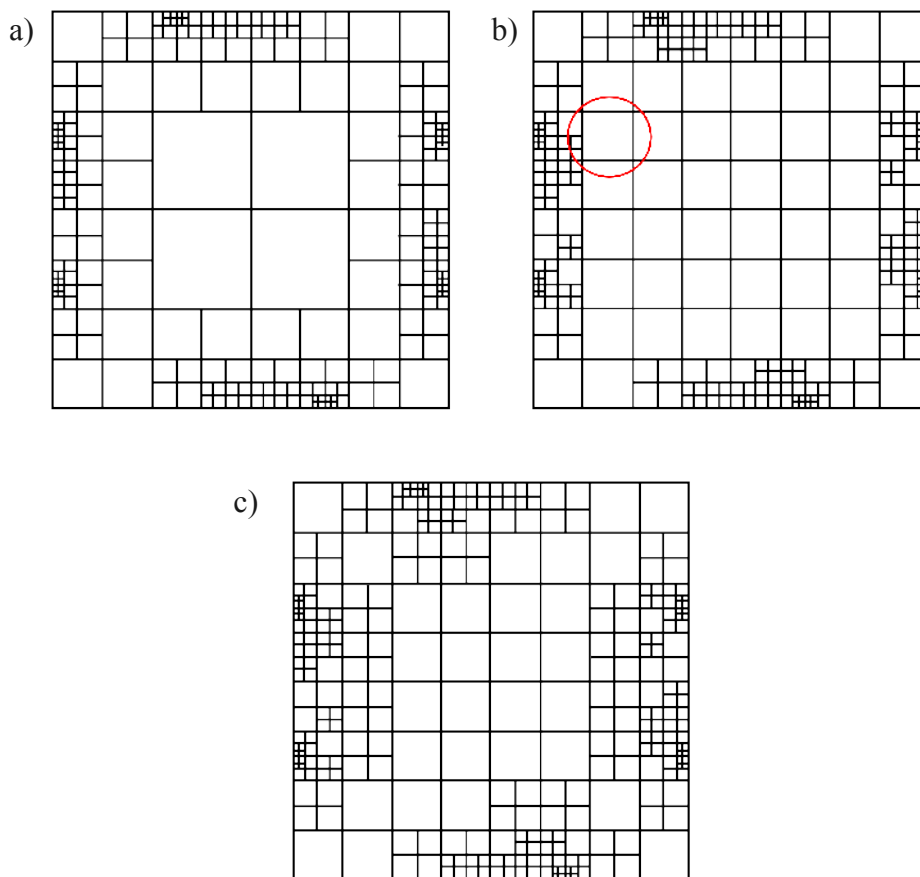


Figura 3.16: Quadtrees: inicial, ajustada pelos erros e restrita.

A geração de uma nova malha para cada *patch* ocorre em três fases: geração da malha no interior por uma *quadtree*; geração da zona de transição por uma abordagem de avanço de fronteira e uma suavização da malha gerada, como descrito a seguir para um *patch* genérico P_k .

3.5.1. Geração da malha no interior do domínio por uma *quadtree*.

Cada *patch* P_k mantém o conjunto de elementos adjacentes do passo anterior, o qual é definido primeiramente na geração da malha inicial e é atualizado após a geração de uma nova malha a cada passo. Um *patch* também guarda o conjunto dos novos elementos que são gerados durante a iteração corrente. Logo, esses dois conjuntos devem coexistir até que uma nova malha seja finalmente gerada.

A *quadtree* não só garante que a densidade das células no interior é sensível à discretização da borda mas também garante uma boa transição entre regiões com diferentes graus de refinamento. O procedimento consiste dos seguintes passos:

1. Construção da *quadtree* inicial;
2. Ajustes devidos ao erros das curvaturas calculadas para os elementos;
3. Transformação da *quadtree* em uma *quadtree* restrita;
4. Eliminação das células próximas à borda;
5. Geração dos elementos nas células do interior por padrões.

A árvore inicial é construída levando-se em conta a discretização das curvas delimitantes do *patch* obtida na fase anterior do processo. A construção é feita no espaço paramétrico de acordo com o algoritmo descrito no Algoritmo 3.3.

A Figura 3.16 mostra a *quadtree* inicial associada a discretização da borda mostrada na Figura 3.15.

A *quadtree* inicial é modificada de acordo as características do domínio da malha e também pelo erro entre as curvaturas em cada um de seus nós. Na avaliação do h_{velho} a área é calculada no espaço tridimensional. Isso ajuda a evitar distorções entre o espaço paramétrico e o Cartesiano, o que pode ocorrer uma vez que um quadrado envolvente $[(0,0),(1,1)]$ é utilizado. Ao final de cada passo, o conjunto de elementos do passo anterior é descartado. A Figura 3.16 mostra o ajuste da *quadtree* após ter sido levado em conta o erro entre as

curvaturas na malha do domínio. O Algoritmo 3.4 mostra como ajustar a *quadtree* devido à curvatura da superfície.

Algoritmo 3.3: Construção da *quadtree* inicial a partir da discretização da borda.

```

Inicialize a árvore com o nó raiz vazio
para cada curva de borda,  $C_i$ ,  $i \leftarrow 1$  até 4 faça
{
  Calcule o comprimento de arco de  $C_i$  no espaço 3D,  $L_i$ 
  para cada segmento,  $S_j$ ,  $j \leftarrow 1$  até  $n_{seg}$  faça
  {
    Calcule o comprimento de arco de  $S_j$  no espaço 3D,  $L_{segj}$ 
    Calcule o centro de  $S_j$  no espaço 3D,  $C_{segj}$ 
    Encontre as coordenadas paramétricas de  $C_{segj}$ ,  $(u,v)$ 
    Encontre em que célula  $(u,v)$  está localizado
    enquanto (tamanho da célula  $> (L_{segj} / L_i)$ ) faça
    {
      Subdivida a célula em quatro filhos
      Determine em que célula  $(u,v)$  está localizado
    }
  }
}

```

Algoritmo 3.4: Ajustes devido a curvatura da superfície.

```

 $A_{total} \leftarrow 0$ 
para cada elemento da lista de elementos iniciais,  $E_k$ ,  $k \leftarrow 1$  até  $n_{el}$  faça
   $A_{total} \leftarrow A_{total} + A_{E_k}$ 
para cada elemento da lista de elementos iniciais,  $E_k$ ,  $k \leftarrow 1$  até  $n_{el}$  faça
{
  Calcule  $h_{novo}$  de acordo com o Algoritmo 3.5
  Calcule o centro de  $E_k$  no espaço 3D,  $C_{elk}$ 
  Encontre as coordenadas paramétricas de  $C_{elk}$ ,  $(u_k, v_k)$ 
  Determine em que célula  $(u_k, v_k)$  está localizado
  enquanto tamanho da célula  $> h_{novo}$  faça
  {
    Subdivida a célula em quatro filhos
    Determine em que célula  $(u_k, v_k)$  está localizado
  }
}

```

Algoritmo 3.5: Cálculo de h_{novo} para um elemento.

```


$$h_{velho_k} \leftarrow \sqrt{\frac{A_{E_k}}{A_{total}}}$$


$$h_{novo} \leftarrow 0$$

para cada vértice de  $E_k$ ,  $v_j$ ,  $j \leftarrow 1$  até 3 faça
{
  Calcule  $K_a$  e  $K_d$  em  $v_j$ 
  Calcule  $h_j$  de acordo com a Tabela 3.2
}

```

```

    }
    h_novo ← h_novo + h_j
}
h_novo ← h_novo / 3

```

Transformar a *quadtree* obtida em uma *quadtree* restrita se faz necessário pois as células do interior são trianguladas por padrões (Figura 3.17). Além disso, apenas um nível de diferença entre células adjacentes garante uma boa transição na malha. A Figura 3.16 ilustra a *quadtree* restrita, a célula demarcada em (b), por exemplo, deve ser subdividida para que se tenha apenas um nível de diferença em relação às células vizinhas, como se pode ver em (c).

O próximo passo é a triangulação das células do interior. No entanto, próximo à borda do *patch*, devido à discretização das curvas delimitantes, não só os padrões não podem ser aplicados como também triângulos distorcidos podem ser gerados. Para evitar esse problema, uma zona de transição é criada pela união das células que têm contato com a borda (Figura 3.18). As células que não fazem parte da zona de transição são trianguladas com os padrões da Figura 3.17 (aplicados na Figura 3.18). Já as células que estão na zona de transição não são utilizadas nessa triangulação. Ao invés disso, elas são trianguladas por um avanço de fronteira com base no critério de Delaunay.

O processo de eliminação de células resulta numa *quadtree* composta por dois tipos de células: de transição e de interior. Se uma célula toca a borda, é uma célula de transição; caso contrário, é uma célula de interior. As folhas da árvore que são células de interior representam sub-regiões que serão trianguladas por padrões [25].



Figura 3.17: padrões para a malha do interior.

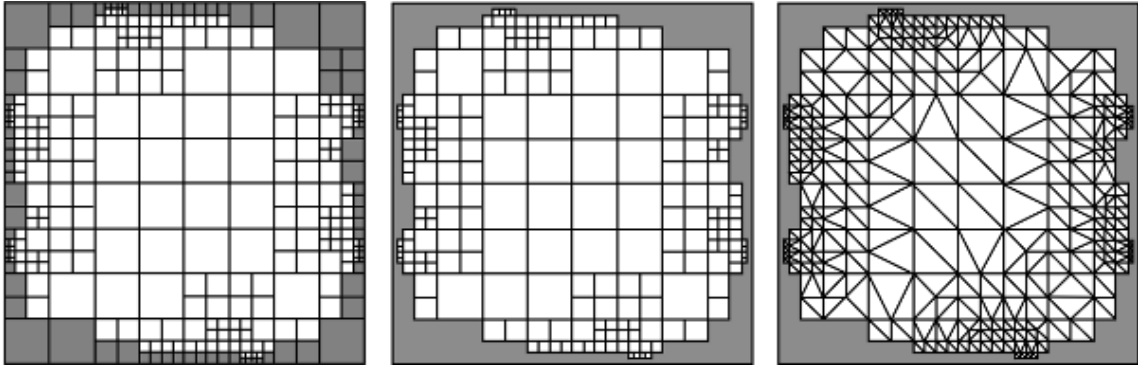


Figura 3.18: Definindo as células da zona de transição e geração da malha interior por padrões.

3.5.2. Geração da malha da zona de transição.

A zona de transição é triangulada por uma técnica de avanço de fronteira baseada na triangulação de Delaunay. Essa é a última fase da geração da malha para um patch P_k , e é realizada conforme o Algoritmo 3.6. A Figura 3.19 ilustra a malha final gerada após o avanço de fronteira ter sido aplicado.

Algoritmo 3.6: Algoritmo par ao avanço de fronteira

A partir de todos os segmentos do contorno fornecidos, construir uma lista de arestas ativas.

enquanto a lista de arestas ativas não esteja vazia **faça**
 {

Escolher uma aresta (base para um novo triângulo).

para cada aresta escolhida **faça**

Construir uma lista de células interiores adjacentes às células que contêm os vértices inicial e final dessa aresta base.

se não houver células **então**

A lista de células será formada pelas oito adjacências das células que contêm os vértices inicial e final da aresta base.

Dentro da lista de células, buscar o vértice que forme o triângulo com o maior ângulo oposto à aresta base.

Após adicionar um triângulo, a lista de arestas ativas deve ser atualizada.

}

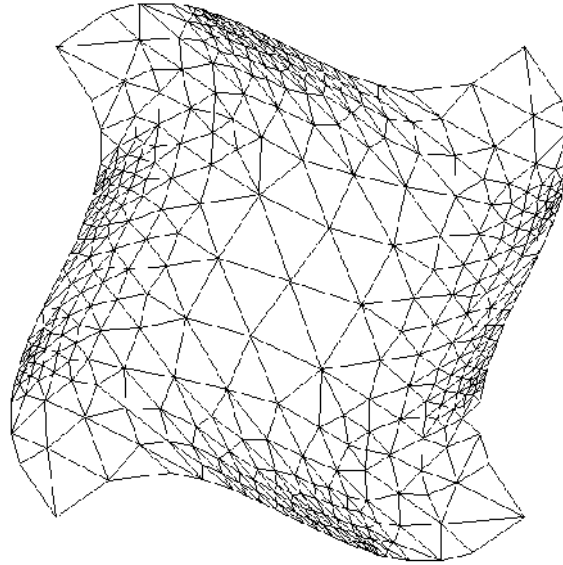


Figura 3.19: Malha final gerada.

Finalmente, após toda a nova malha ser gerada, uma suavização da malha é aplicada. A suavização adotada aqui trata de calcular novas coordenadas, no espaço paramétrico, para todos os nós da malha, exceto àqueles que se encontram na fronteira. A técnica é uma formulação geral do operador de suavização Laplaciano, que utiliza um fator que relaciona as medidas do espaço real e do espaço paramétrico, evitando as distorções

$$X_0^{n+1} = X_0^n + \Phi \frac{\sum_{i=1}^m w_{i0} (X_i^n + X_0^n)}{\sum_{i=1}^m w_{i0}}, \quad (3.12)$$

onde X_0 é o vetor posição do vértice 0, no qual incidem os nós i (X_i), m é o número de vértices incidentes e w_{i0} é um fator que relaciona a distância real e a distância paramétrica entre o vértice i e o vértice 0, e Φ é um fator de relaxação que deve estar entre 0 e 1. Aplica-se este processo em todos os vértices interiores do domínio e pode ser repetido várias vezes. Após a suavização, os nós e elementos são mapeados para o espaço tridimensional.

3.6 Considerações finais

Neste Capítulo, viu-se que para dar início ao processo adaptativo é necessário um modelo, o qual é composto pela descrição geométrica dos vários *patches* que compõem a superfície a ter sua malha gerada juntamente com as malhas iniciais de cada um desses *patches*. A estratégia assume que essa superfície é composta por *patches* paramétricos, tais como *patches* de Coons, os quais são delimitados por curvas paramétricas. Também são descritos os detalhes específicos de cada fase da estratégia proposta neste trabalho; como ela se divide em duas fases: análise de erro e adaptação da malha. Inicialmente, é feita uma análise de erro baseada na curvatura considerando a malha inicial do modelo, ficando para a segunda etapa o processo de adaptação da malha em questão. Uma nova malha é gerada e um ciclo, contendo essas duas etapas, se repete até que um critério de erro global pré-definido seja atendido.

Capítulo 4

Exemplos e Análise de Resultados

4.1 Introdução

Neste Capítulo, alguns exemplos de superfícies são submetidos à estratégia de auto-adaptação proposta por este trabalho e os resultados obtidos são analisados. Todos os exemplos foram gerados por um sistema desenvolvido e implementado com base na estratégia proposta. Os *patches* são modelados como superfícies bi-paramétricas resultantes da mesclagem (*blending*) das curvas de borda. Os modelos de todos os exemplos usam curvas de Hermite. Entretanto, a estratégia é genérica e poder ser usada para qualquer tipo de curva, desde que parametrizável. A formulação completa dos *patches* e curvas de Hermite, bem como todas as informações geométricas podem ser encontradas em [28]. O critério utilizado para medir a convergência foi a estimativa de erro global com base nas curvaturas, tal como descrito no Capítulo 3. Os exemplos mostram que o erro decresce à medida em que a malha é adaptada, indicando a convergência do método.

4.2 Exemplos

4.2.1 Modelo com interior plano e bordas curvas

Este primeiro exemplo mostra uma superfície de Hermite clássica, definida pela geometria da Figura 4.1. Para esse exemplo é necessário a definição de apenas um *patch* de Hermite, com os vértices: $P(0,0) = (-5,0; -5,0; 0,0)$, $P(1,0) = (5,0; -5,0; 0,0)$, $P(0,1) = (-5,0; 5,0; 0,0)$ e $P(1,1) = (5,0; 5,0; 0,0)$ para os quatro cantos e as derivadas: $Q_u(0,0) = Q_u(1,0) = Q_u(0,1) = Q_u(1,1) = (10,0; 10,0; 0,0)$ e $Q_v(0,0) = Q_v(1,0) = Q_v(0,1) = Q_v(1,1) = (-10,0; 10,0; 0,0)$ também definidas nos quatro cantos. O vetor nulo foi utilizado como vetor *twist* para essa superfície.

A Figura 4.2 mostra a malha inicial e os dois primeiros passos para a geração da malha. Nesse exemplo, o erro global decaiu de 62,7% para 5,1%.

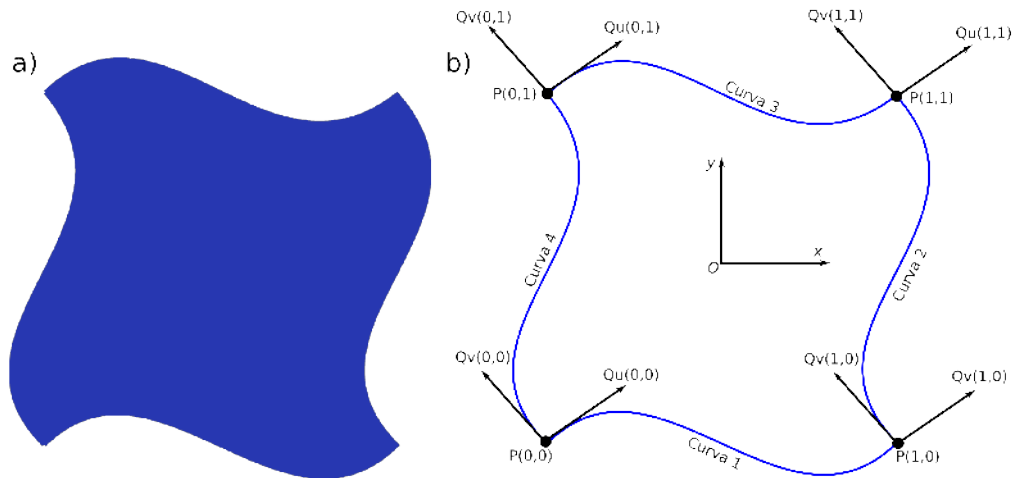


Figura 4.1: Modelo para a superfície de Hermite com interior plano (a); Geometria para superfície (b)As curvas que delimitam o patch: Curva 1, formada pelos pontos $P(0,0)$ e $P(1,0)$ e pelas derivadas $Qu(0,0)$ e $Qu(1,0)$; Curva 2, formada pelos pontos $P(1,0)$ e $P(1,1)$ e pelas derivadas $Qv(1,0)$ e $Qv(1,1)$; Curva 3, formada pelos pontos $P(0,1)$ e $P(1,1)$ e pelas derivadas $Qu(0,1)$ e $Qu(1,1)$ e Curva 4, formada pelos pontos $P(0,0)$ e $P(0,1)$ e pelas derivadas $Qv(0,0)$ e $Qv(0,1)$.

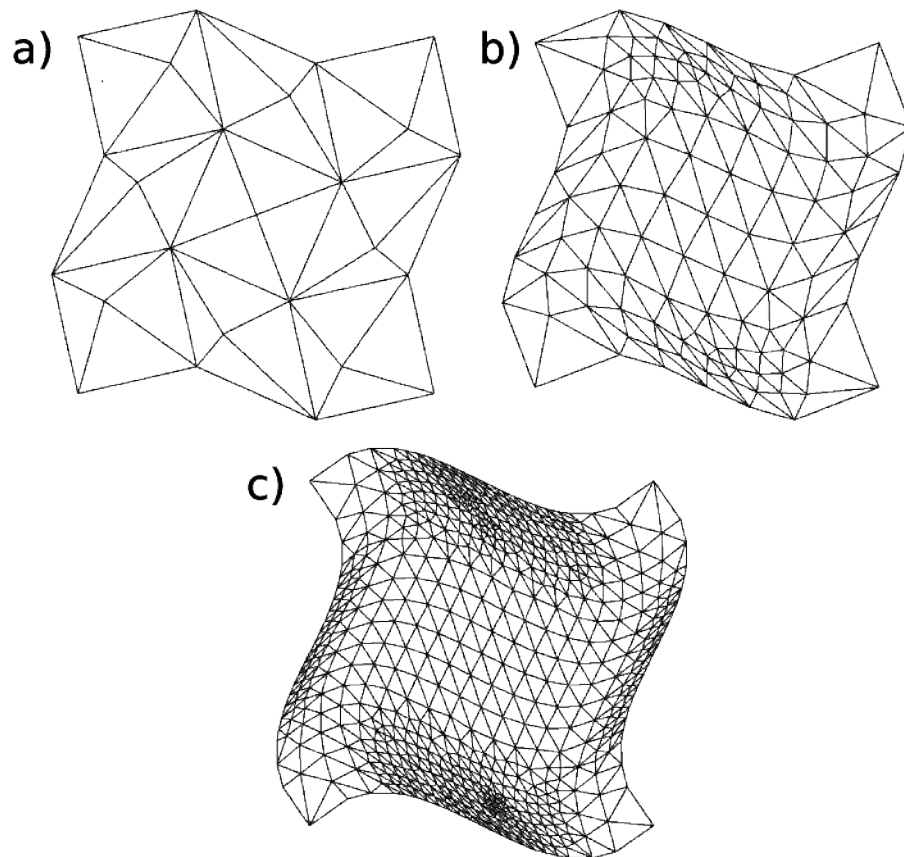


Figura 4.2: Geração da malha para o patch de Hermite. (a) Malha inicial (25 nós e 36 elementos), (b) Passo 1 (109 nós e 196 elementos), (c) Passo 2 (597 nós e 1104 elementos).

4.2.2 Modelo com curvatura alta e fronteira plana (bolha alta)

Este segundo exemplo mostra um modelo com alta curvatura no interior do domínio e fronteira plana, cuja superfície é ilustrada pela Figura 4.4. A representação da geometria é apresentada na Figura 4.3. A geometria do modelo da bolha alta é definida pelos vértices: $P(0,0) = (-0,5; -0,5, 0,0)$, $P(1,0) = (0,5; -0,5; 0,0)$, $P(0,1) = (-0,5; 0,5; 0,0)$ e $P(1,1) = (0,5; 0,5; 0,0)$, nos quatro cantos e pelas derivadas: $Qu(0,0) = Qu(1,0) = Qu(0,1) = Qu(1,1) = (1,0; 0,0; 0,0)$ e $Qv(0,0) = Qv(1,0) = Qv(0,1) = Qv(1,1) = (0,0; 1,0; 0,0)$. O vetor $(0,0; 0,0; 20,0)$ está definido como *twist* para os vértices $P(0,0)$ e $P(1,1)$; para os pontos $P(1,0)$ e $P(0,1)$ o vetor *twist* é dado por $(0,0; 0,0; -20,0)$.

As Figuras 4.5 e 4.6 mostram a malha inicial e os três primeiros passos para a geração da malha da bolha alta, onde o erro decaiu de 93,7% para apenas 4,0%. Nesse exemplo e nos subsequentes, as malhas iniciais são configuradas com apenas 4 elementos. Isso é feito propositalmente para demonstrar a robustez da estratégia, que mesmo partindo de uma malha inicial extremamente desrefinada, consegue convergir para uma malha próxima à definição matemática para a superfície.

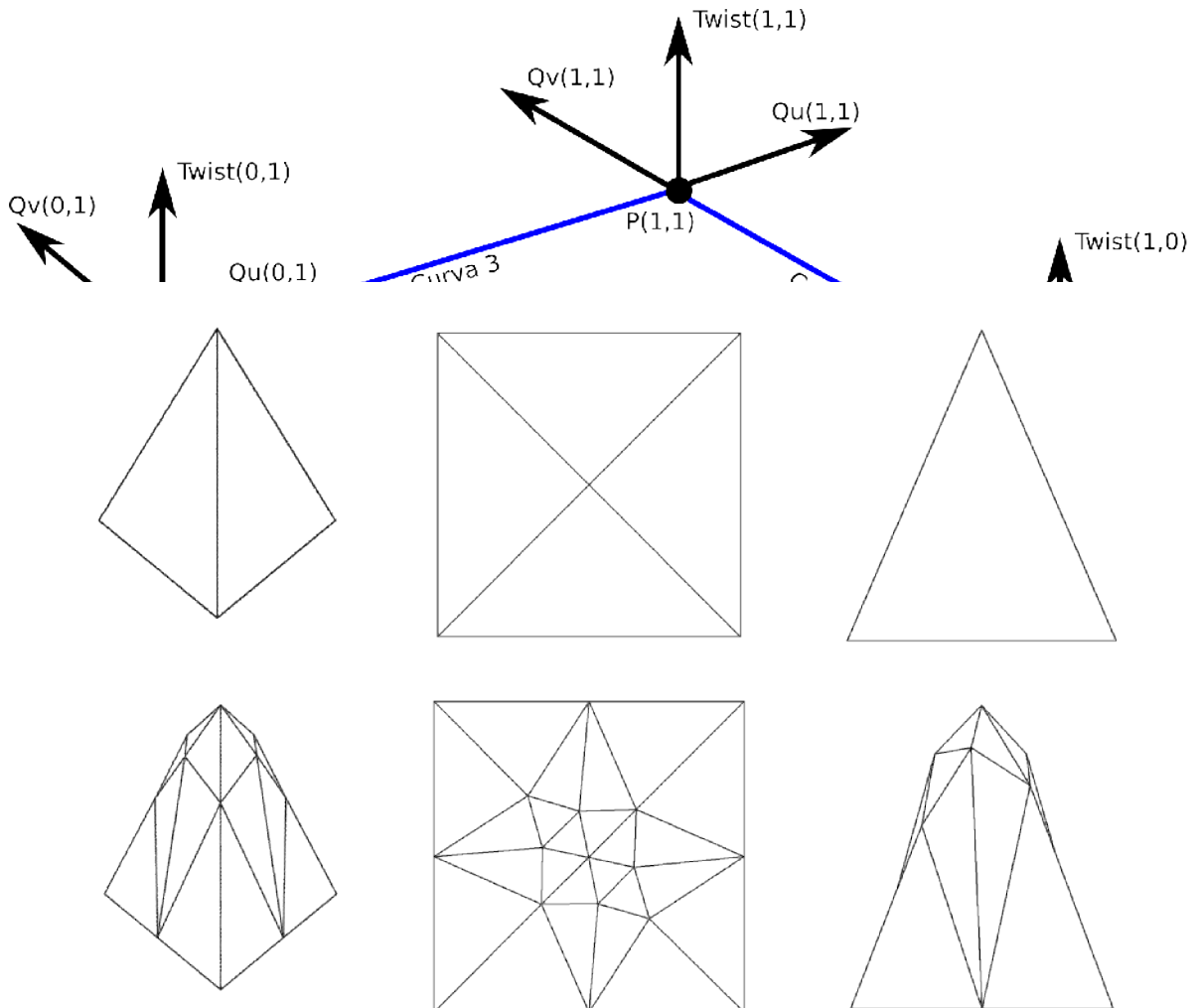


Figura 4.5: Geração da malha para o modelo da bolha alta. No trio superior, a malha inicial com apenas 5 nós e 4 elementos; no trio inferior, a malha do passo 1 (17 nós e 24 elementos). Vistas em 45°, superior e lateral.

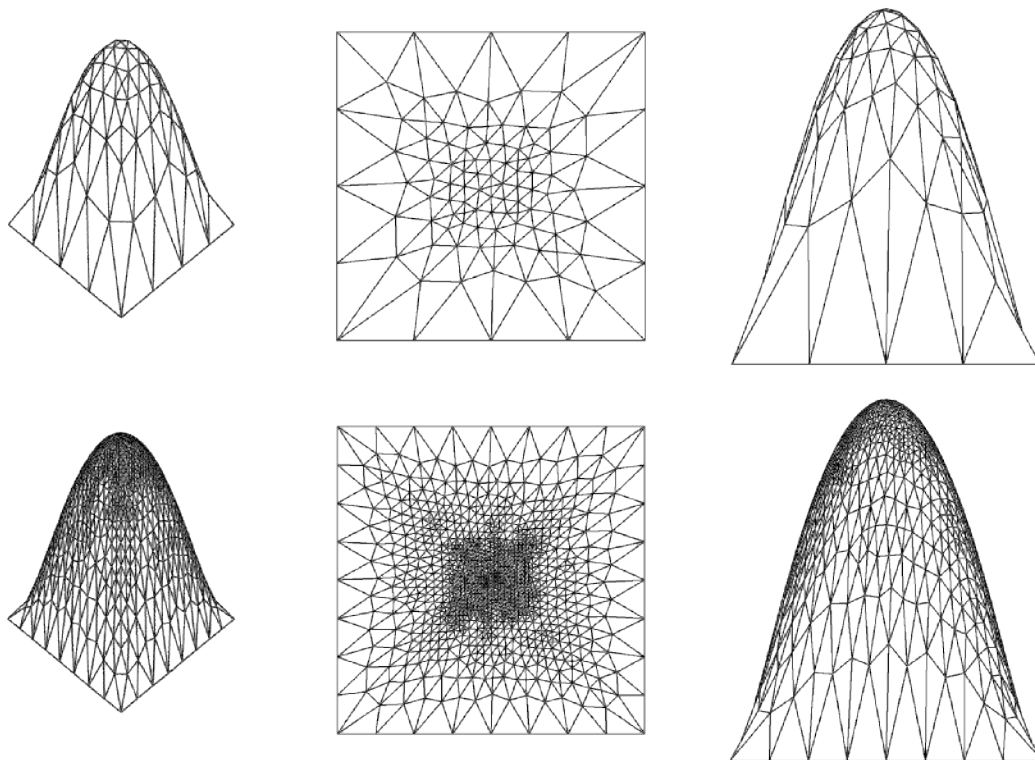


Figura 4.6: No trio superior, a malha do passo 2 (105 nós e 192 elementos); no trio inferior, a malha do passo 3 (1024 nós e 2014 elementos). Vistas em 45°, superior e lateral.

4.2.3 Modelo com curvatura alta e base circular (bolha alta circular)

O terceiro exemplo mostra o modelo da bolha alta, agora com uma base circular. A superfície do *patch* Hermite é ilustrada pela Figura 4.7 e a representação da geometria na Figura 4.8.

A geometria do modelo da bolha alta com base circular é definida pelos vértices: $P(0,0) = (0,0; -5,0; 0,0)$, $P(1,0) = (5,0; 0,0; 0,0)$, $P(0,1) = (-5,0; 0,0; 0,0)$ e $P(1,1) = (0,0; 5,0; 0,0)$, nos quatro cantos do *patch* e pelas derivadas: $Q_u(0,0) = Q_u(1,1) = (8,25; 0,0; 0,0)$, $Q_u(1,0) = Q_u(0,1) = (0,0; 8,25; 0,0)$ e $Q_v(0,0) = Q_v(1,1) = (-8,25; 0,0; 0,0)$, $Q_v(0,1) = Q_v(1,1) = (0,0; 8,25; 0,0)$. O vetor $(0,0; 0,0; 200,0)$ está definido como *twist* para os vértices $P(0,0)$ e $P(1,1)$; para os pontos $P(1,0)$ e $P(0,1)$ o vetor *twist* é dado por $(0,0; 0,0; -200,0)$.

A Figura 4.9 mostra a malha inicial e os três primeiros passos para a geração da malha da bolha alta com base circular, onde o erro decai de 96,9% para apenas 6,4%.

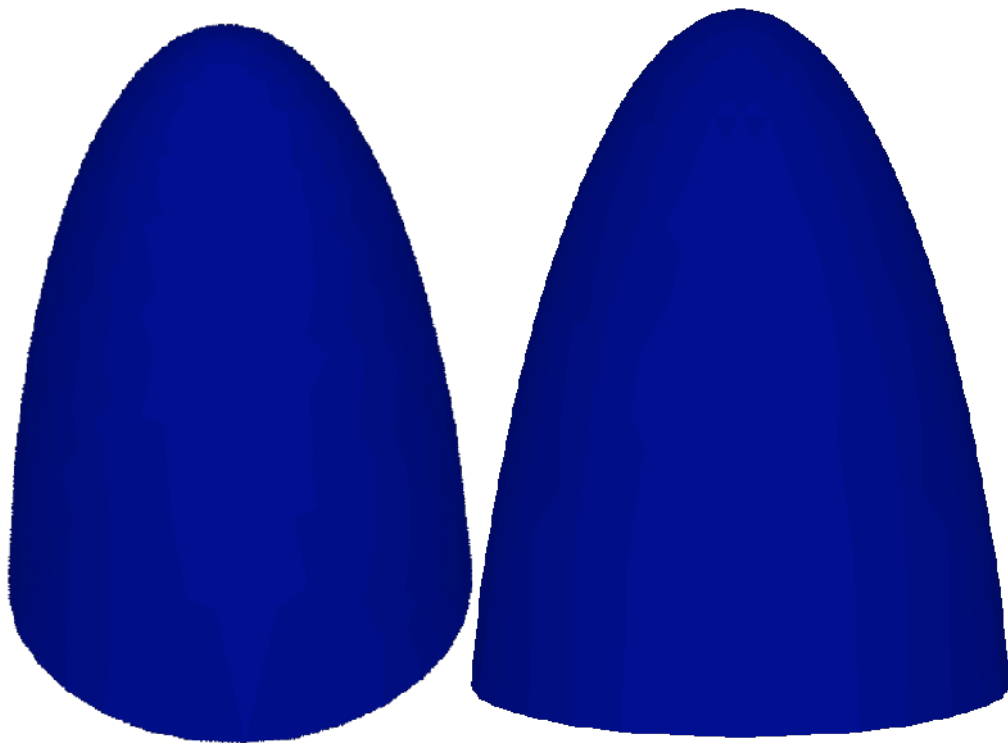


Figura 4.7: Modelo para bolha alta com base circular.

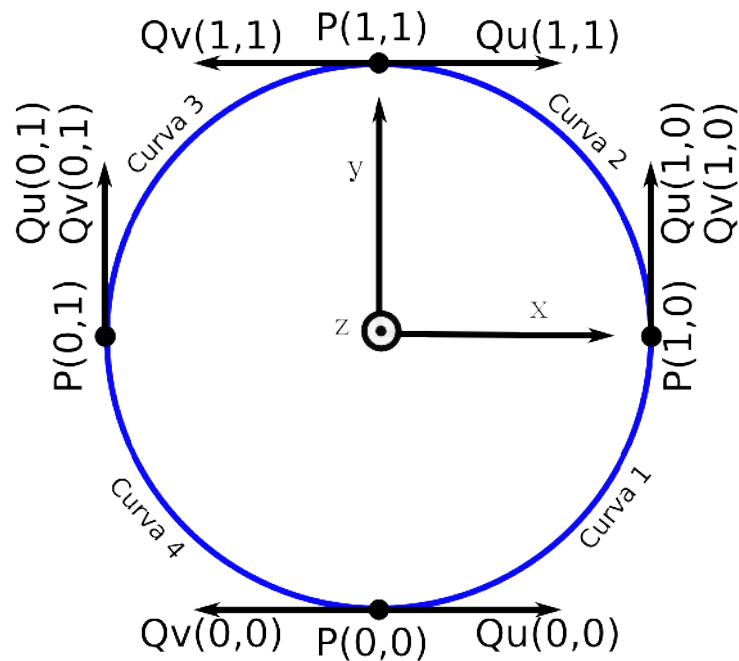


Figura 4.8: Geometria, vista de cima, do modelo para o exemplo com base circular. As curvas que delimitam o patch: Curva 1, formada pelos pontos $P(0,0)$ e $P(1,0)$ e pelas derivadas $Q_u(0,0)$ e $Q_u(1,0)$; Curva 2, formada pelos pontos $P(1,0)$ e $P(1,1)$ e pelas derivadas $Q_v(1,0)$ e $Q_v(1,1)$; Curva 3, formada pelos pontos $P(0,1)$ e $P(1,1)$ e pelas derivadas $Q_u(0,1)$ e $Q_u(1,1)$ e Curva 4, formada pelos pontos $P(0,0)$ e $P(0,1)$ e pelas derivadas $Q_v(0,0)$ e $Q_v(0,1)$.

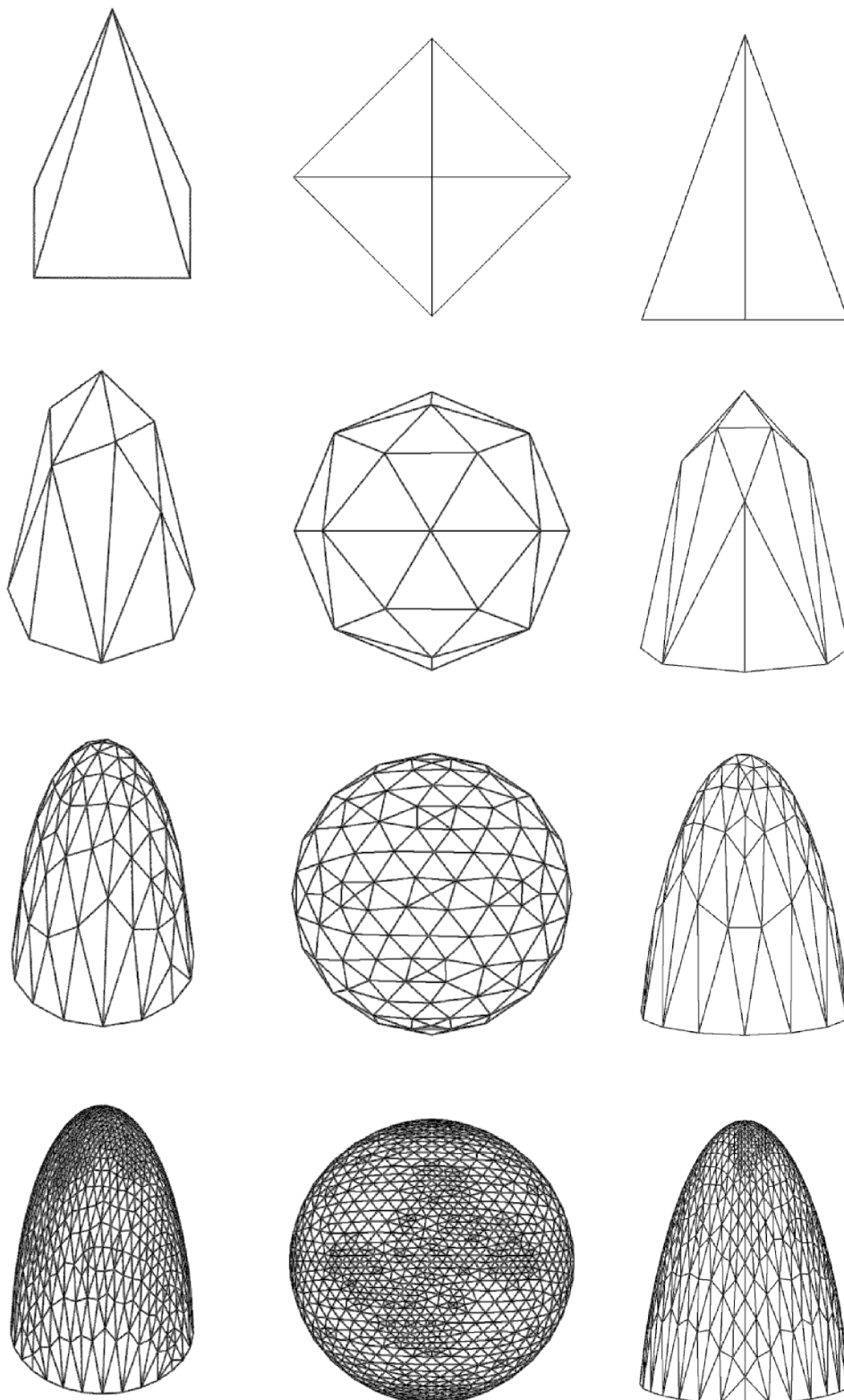


Figura 4.9: Geração da malha para o modelo da bolha alta com base circular. De cima para baixo: o trio correspondente à malha inicial (5 nós e 4 elementos), o segundo trio traz a malha para o passo 1 (17 nós e 24 elementos), o terceiro trio mostra a malha do passo 2 (105 nós e 192 elementos) e por fim a malha do passo 3 (876 nós e 1718 elementos). Da esquerda para direita, vistas em 45°, superior e lateral.

4.2.4 Modelo com curvatura baixa e base circular (calota)

A superfície para o modelo da calota é mostrada na Figura 4.10, vista em 45°, superior e lateral. A geometria é definida com da mesma forma mostrada na Figura 4.8, e os pontos e derivadas são os mesmos do Exemplo 4.2.3, com exceção dos vetores *twist*, que são definidos como $(0,0; 0,0; 10,0)$ para os pontos $P(0,0)$ e $P(1,1)$ e $(0,0; 0,0; -10,0)$ para os pontos $P(1,0)$ e $P(0,1)$.

A Figuras 4.11 mostra a malha inicial e os três primeiros passos para a geração da malha da bolha alta com base circular, onde o erro decai de 54,9% para apenas 4,2%.

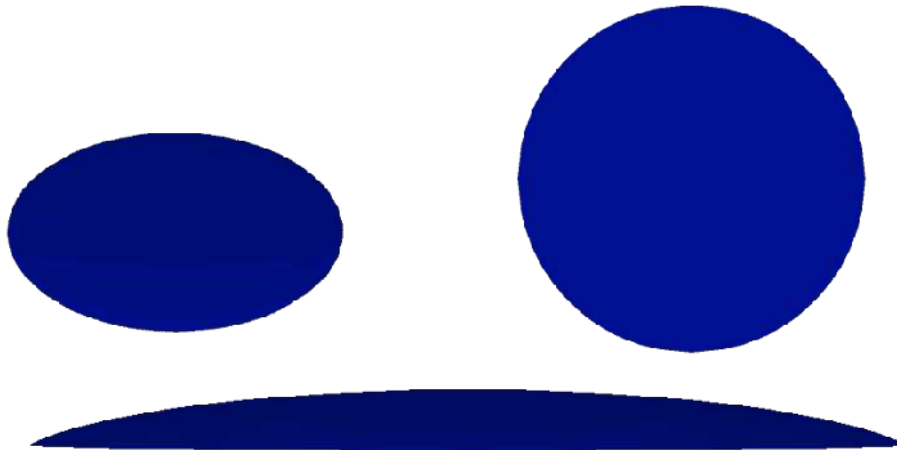


Figura 4.10: Superfície da calota.

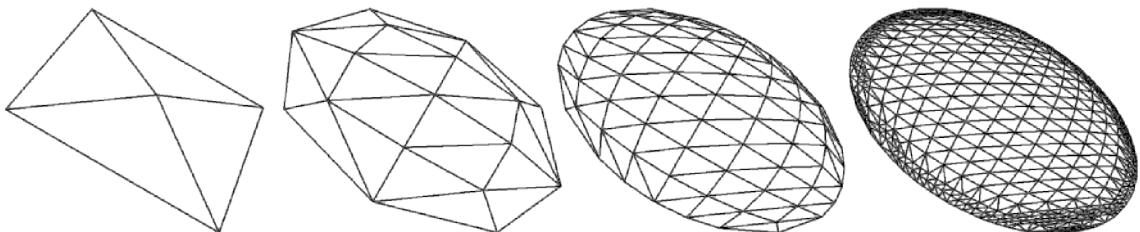


Figura 4.11: Da esquerda para a direita: a malha inicial, passo 1 (17 nós e 24 elementos), passo 2 (73 nós e 120 elementos) e passo 3 (515 nós e 940 elementos).

4.2.5 Modelo com curvatura e borda não planar (sela)

O modelo da sela representa uma superfície com alta curvatura e bordas não planares. A geometria do modelo é ilustrada na Figura 4.12. O *patch* é formado pelas curvas C_1 , C_2 , C_3 e C_4 , possuindo o mesmo vetor *twist* $(0,0,0)$, nos quatros cantos (pt_1 , pt_2 , pt_3 e pt_4). A Figura 4.13 descreve os primeiros três passos para a geração da malha. O erro global cai de 96% para 2,9%, o que indica um alto grau de convergência.

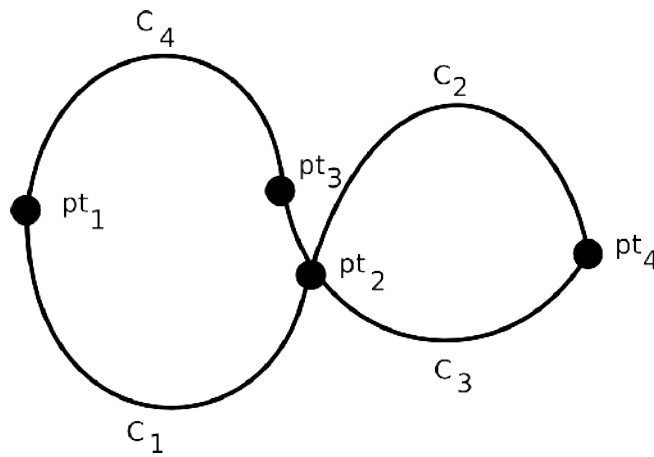


Figura 4.12: Modelo para a sela.

Os valores são: $pt_1 = (-5,0; -5,0; 0,0)$, $pt_2 = (5,0; -5,0; 0,0)$, $pt_3 = (-5,0; 5,0; 0,0)$, $pt_4 = (5,0; 5,0; 0,0)$, $Qu_{00} = (1,0; 0,0; 18,25)$, $Qu_{10} = (1,0; 0,0; -18,25)$, $Qu_{01} = (1,0; 0,0; 18,25)$, $Qu_{11} = (1,0; 0,0; -18,25)$, $Qv_{00} = (0,0; 1,0; -18,25)$, $Qv_{10} = (0,0; 1,0; -18,25)$, $Qv_{01} = (0,0; 1,0; 18,25)$, $Qv_{11} = (0,0; 1,0; 18,25)$.

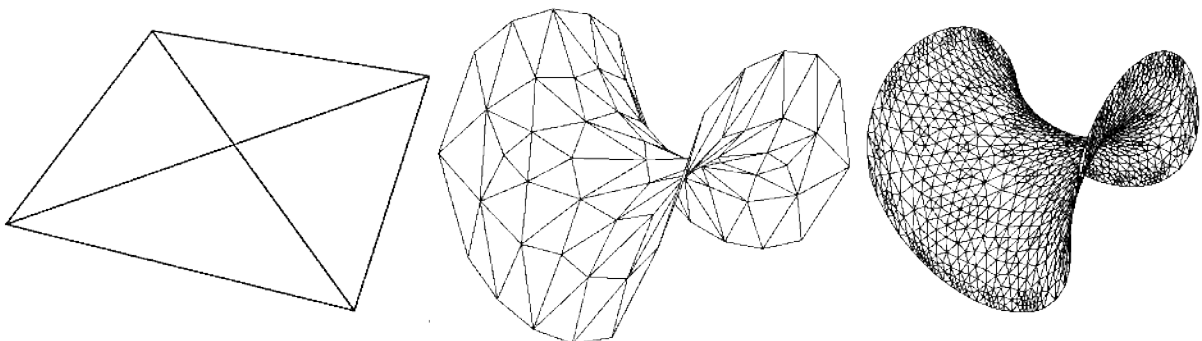


Figura 4.13: Geração da malha da sela. (a) Malha inicial muito desrefinada (4 elem. e 5 nós). (b) Passo 1 (104 elem. e 65 nós). (c) Passo 2 (1936 elem. e 1033 nós).

4.2.6 Superfície com um canto rebaixado

Este exemplo apresenta uma superfície com um dos cantos rebaixados. O modelo e a geometria para a superfície são ilustrados na Figura 4.14. Sua geometria é definida pelos pontos: $P(0,0) = (-0,5; -0,5; 0,0)$, $P(1,0) = (0,5; -0,5; 0,0)$, $P(0,1) = (-0,5; 0,5; 0,0)$, $P(1,1) = (0,5; 0,5; -0,5)$. As tangentes para os quatro cantos são: $Q_u(0,0) = (3,0; 0,0; 0,0)$, $Q_u(1,0) = Q_u(0,1) = Q_u(1,1) = (1,0; 0,0; 0,0)$ e $Q_v(0,0) = (0,0; 3,0; 0,0)$, $Q_v(1,0) = Q_v(0,1) = Q_v(1,1) = (0,0; 1,0; 0,0)$. O vetor nulo foi atribuído aos quatro vetores *twists* dos cantos da superfície.

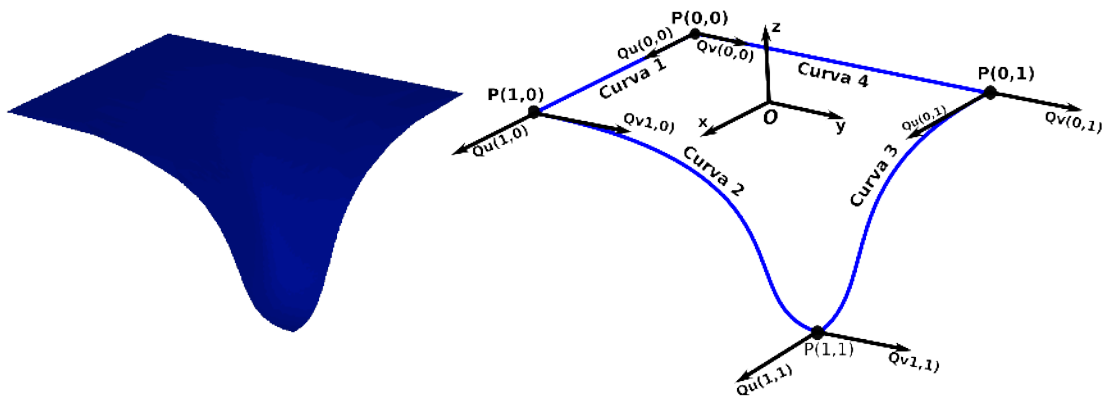


Figura 4.14: Modelo e geometria para superfície de canto rebaixado.

A Figuras 4.15 descreve a malha inicial e as malhas para os três primeiros passos da geração. O erro cai de 92,3% para 5,4%.

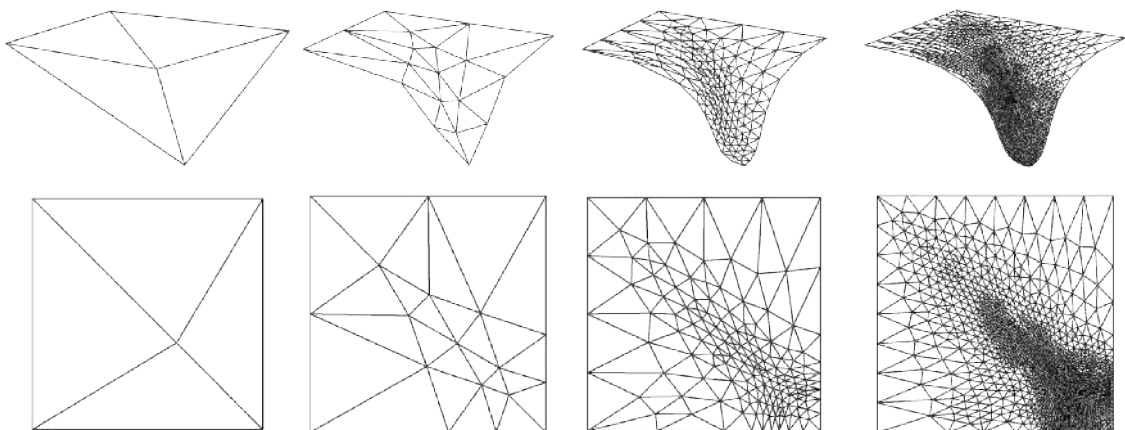


Figura 4.15: Malhas inicial, passo 1, 2 e 3 para a superfície com ponta baixa. Acima, vista a 45°; abaixo, vista superior.

4.2.7 Modelo com dois *patches* adjacentes (rampa)

Este exemplo mostra dois *patches* vizinhos. Enquanto um deles é completamente plano, o outro eleva-se formando uma rampa. A geometria é ilustrada na Figura 4.16, enquanto as Figuras 4.17 e 4.18 mostram a malha inicial e as malhas dos passos 1, 2 e 3. O erro decai dos 54,9% iniciais para 0,3%. Contudo, podemos destacar que uma taxa de erro tão baixa se explica pelo fato de que o erro global (equação 3.14) é computado de forma relativa à curvatura analítica Ka e, como bem podemos observar, o *patch* plano não possui curvatura, embora contribua com o número de nós na equação de erro global. De qualquer forma, a queda no erro global demonstra a eficácia da estratégia proposta.

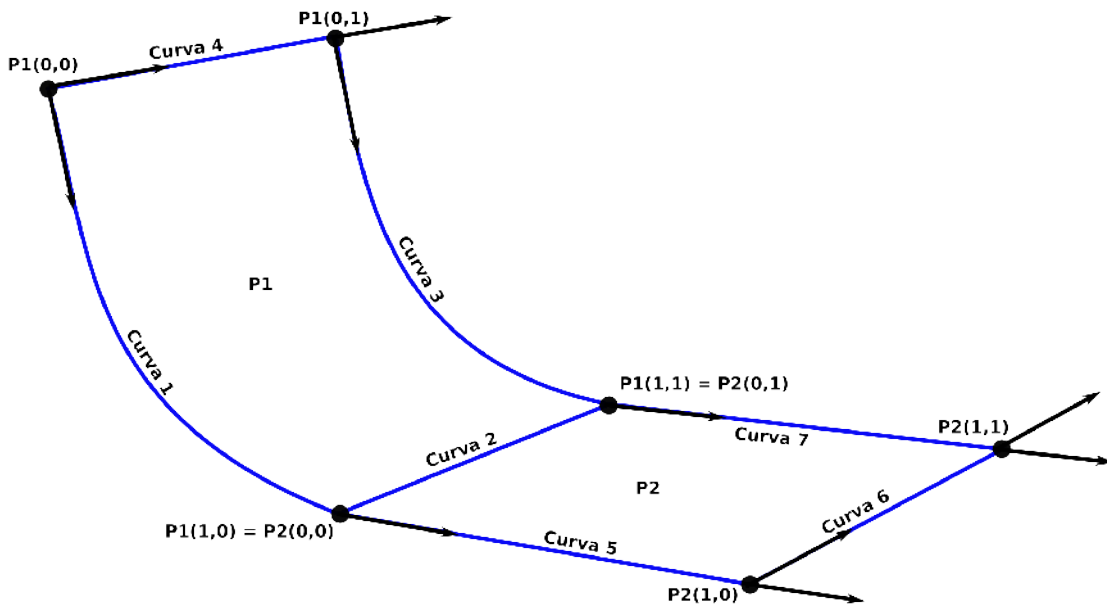


Figura 4.16: Geometria para a rampa fornada pelos *patches* P1 e P2.

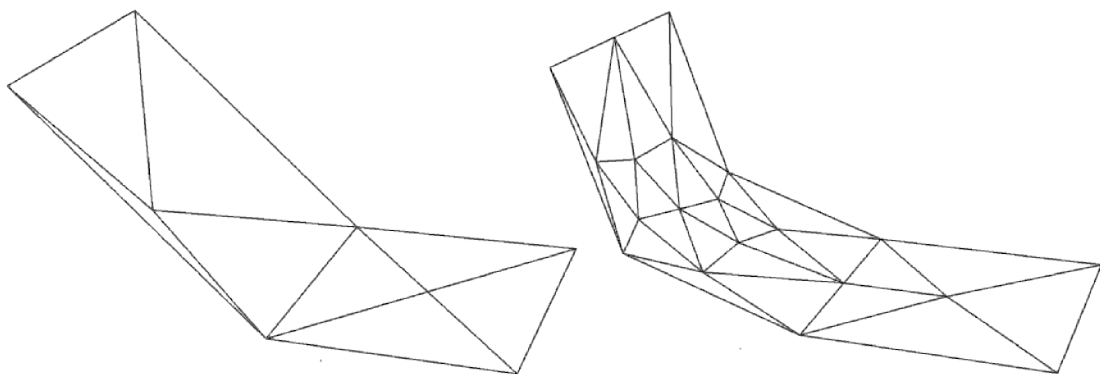


Figura 4.17: Malha inicial e malha do passo 1.

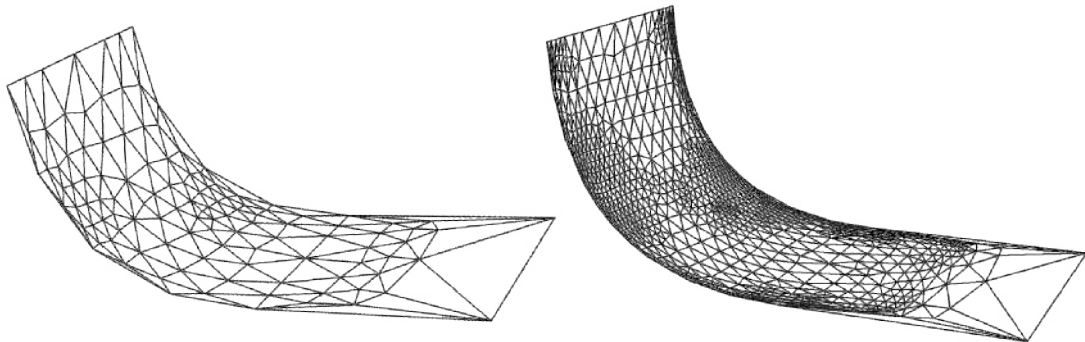


Figura 4.18: Malhas dos passos 2 e 3 da rampa.

4.2.8 Dois *patches* adjacentes, com grande curvatura (pneu)

O modelo do pneu representa uma superfície com dois *patches* cuja geometria é mostrada na Figura 4.19. O *patch* 1 é formado pelas curvas C_1 , C_6 , C_3 , C_5 ; o *patch* 2 pelas curvas C_2 , C_5 , C_4 , C_6 . Os módulos dos vetores *twist* para ambos os *patches* são iguais a 60. A Figura 4.20 descreve a malha inicial e as malhas para os passos 1, 2 e 4 da geração da malha para o pneu. Como a malha inicial do pneu está muito longe da geometria definida, a estratégia necessita de mais iterações para atingir um resultado satisfatório. Isso foi feito propositalmente para demonstrar a eficácia da estratégia proposta, pois mesmo que a malha inicial seja muito ruim, a estratégia é capaz de consertar isso e gerar uma malha final de alta qualidade. Entretanto, nessas situações, podem ser necessários mais passos e isso é atingido no passo 4, para esse exemplo em particular (por isso que o passo 3 não é mostrado nesse exemplo, apesar dele ter sido gerado também). O erro global cai de 57% para 6,7%.

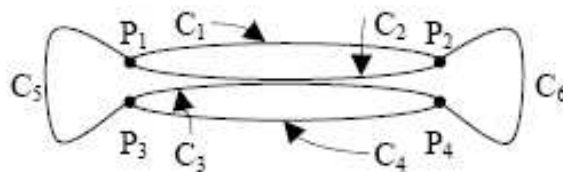


Figura 4.19: Geometria para o pneu.

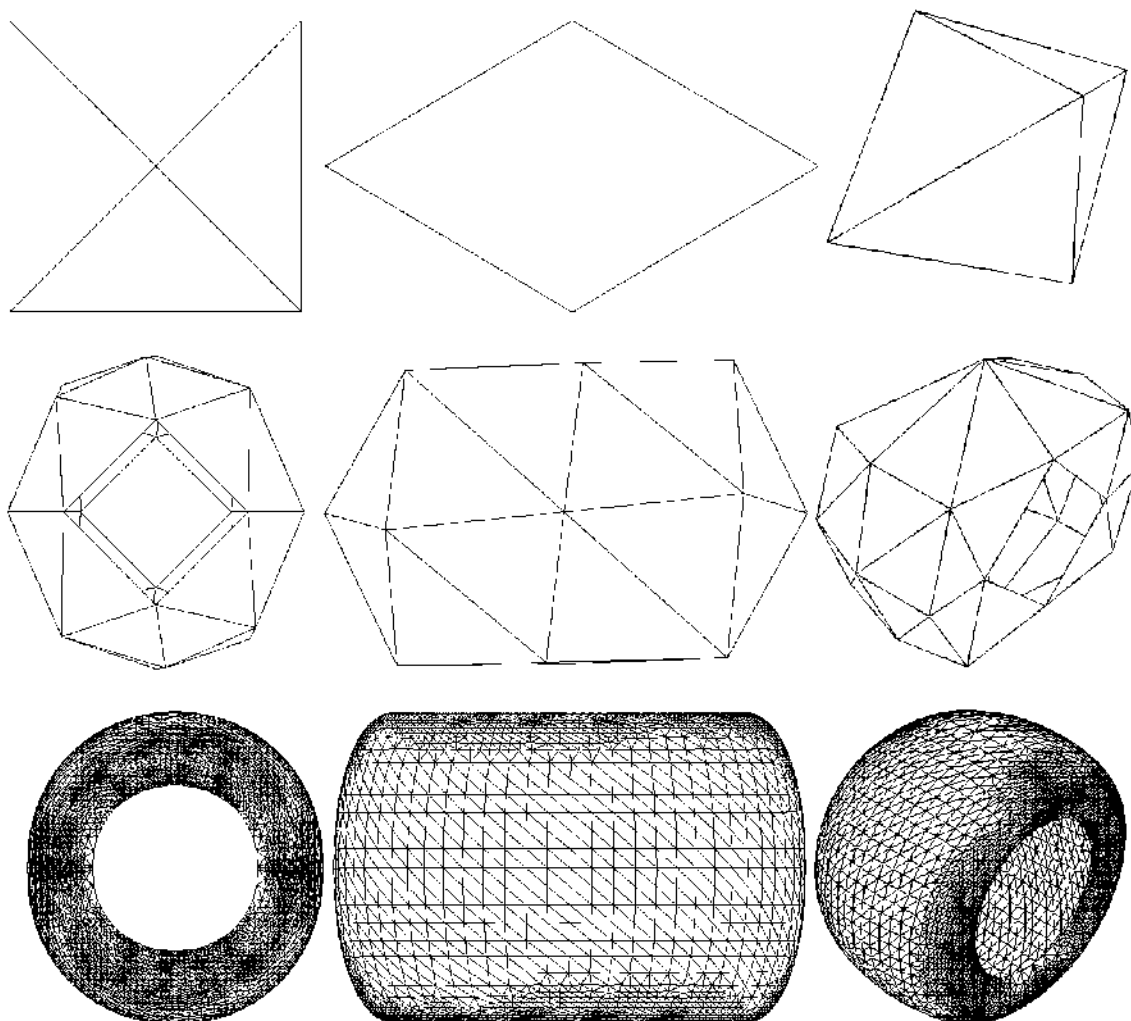


Figura 4.20: Trio superior, a malha inicial extremamente desrefinada para o pneu (8 elementos e 6 nós). Trio do meio, a malha para o passo 1 (48 elementos e 32 nós). No trio inferior, a malha para o Passo 4 (15424 elementos e 8010 nós).

4.2.9 Modelo com vários *patches* com diversas curvaturas (nariz)

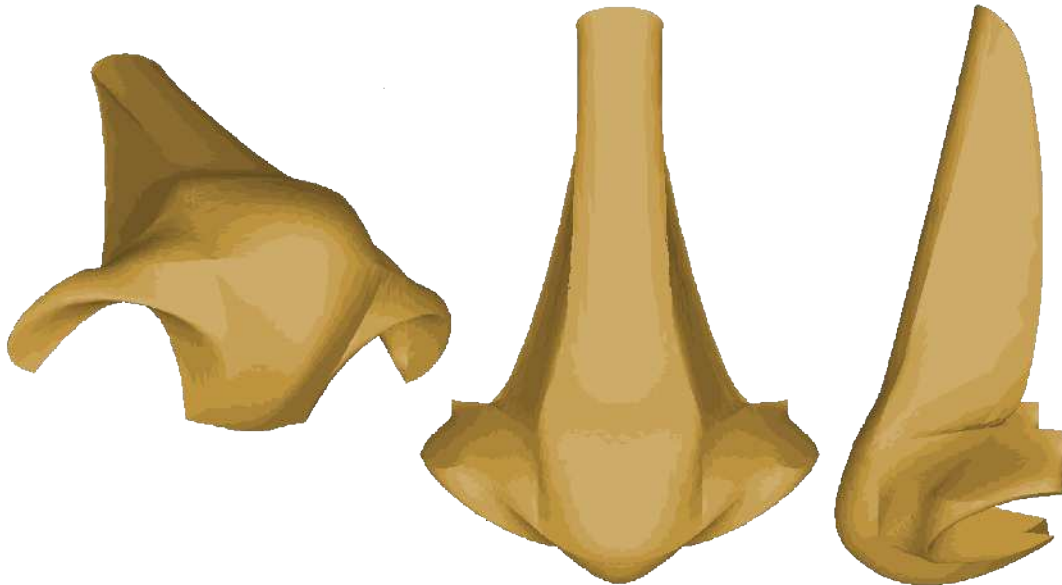


Figura 4.21: Modelo de um nariz para um avatar.

O modelo do nariz é composto por 12 *patches* de Coons com diferentes vetores *twist* e 30 curvas. Este exemplo mostra que o método é suficientemente robusto para lidar com muitos níveis diferentes de curvatura, mantendo uma boa transição entre *patches* adjacentes, refinando e desrefinando a malha onde for necessário. As Figuras 4.22 e 4.23 descrevem os primeiros três passos para o nariz.

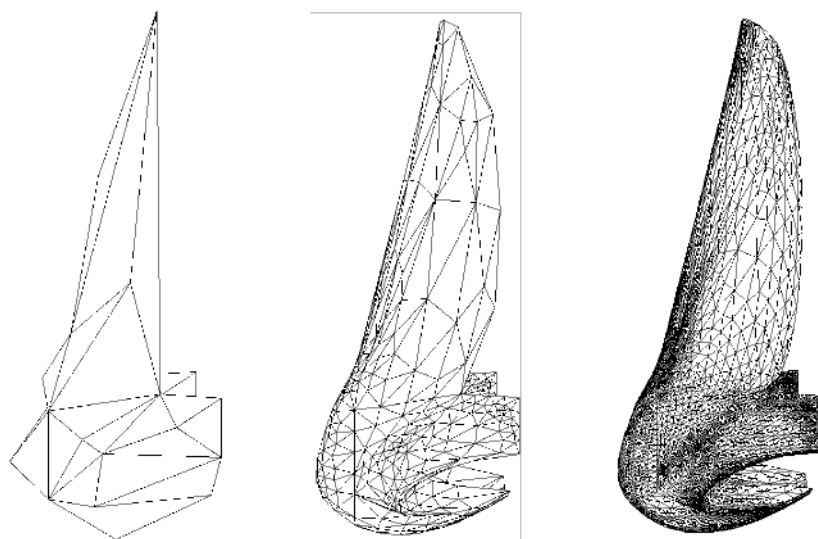


Figura 4.22: Visão esquerda da malha inicial e dos passo 1 e 2 do modelo do nariz.

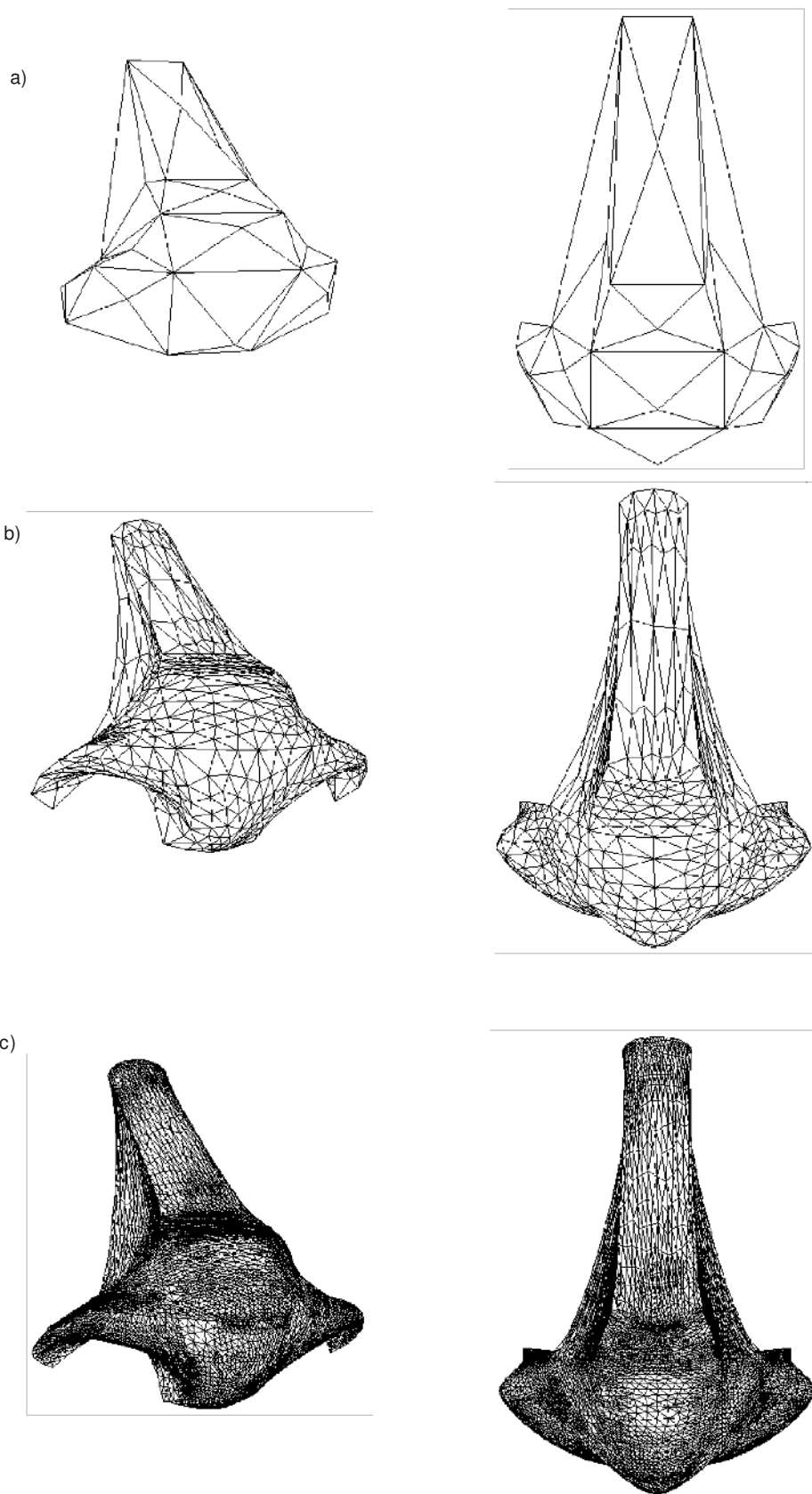


Figura 4.23: O modelo do nariz. (a) A malha inicial (48 elementos e 60 nós). (b) Passo 1 (932 elementos e 587 nós). (c) Passo 2 (15704 elementos e 8371 nós).

4.2.10 Modelo com *patches* sem curvatura (cubo)

Este exemplo preocupa-se em demonstrar a capacidade de desrefinamento da estratégia, ilustrando um cubo contendo seis *patches* (um para cada face). Uma vez que todos os *patches* que compõem o modelo do cubo são planos, não existe curvatura em suas faces. Para demonstrar a eficácia da proposta também para o desrefinamento, a malha inicial é bem refinada propositalmente e as malhas seguintes são progressivamente desrefinadas pela estratégia até a malha ideal. Como não há curvatura, não é possível o cálculo do erro global, que se apresenta sempre como 0%.

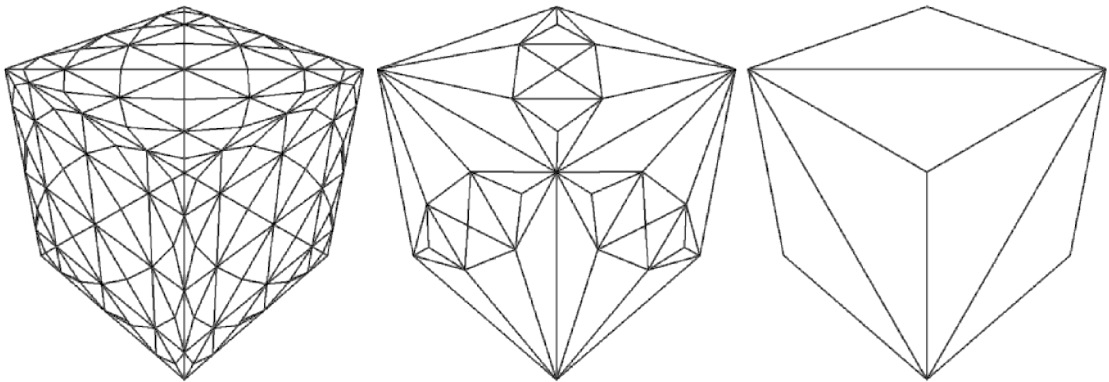


Figura 4.24: Da esquerda para direita: malha inicial refinada com o patch dividido em 16 regiões, seguida das malhas para os passos 1 e 2.

4.3 Considerações finais

Os exemplos apresentados neste Capítulo mostram que a estratégia é capaz de, partindo de uma malha inicial e uma descrição geométrica, gerar uma malha adaptativa para o modelo desejado. A estratégia é capaz de refinar e desrefinar as diversas regiões do modelo conforme seja necessário, concentrando o maior número de elementos em regiões de maior curvatura, deixando as regiões de menor curvatura com uma densidade menor de elementos. Seu grau de convergência é elevado, conseguindo atingir excelentes resultados em poucos passos. A estratégia também é robusta o suficiente para lidar com a combinação de diversos *patches* com diferentes graus de curvatura, como mostram os exemplos do nariz e do pneu, conseguindo lidar até mesmo com os casos onde não há curvatura alguma, como mostrado no exemplo do cubo.

Capítulo 5

Conclusões e Trabalhos Futuros

5.1 Conclusões

Este trabalho apresentou uma estratégia hierárquica de geração de malhas adaptativas para superfícies paramétricas baseada em uma estrutura de árvore para aplicações de computação gráfica e realidade virtual. A estratégia apresentada faz uso de um critério geométrico para guiar o processo de auto-adaptação da malha. Esse critério geométrico é baseado nos erros entre as curvaturas analítica e discreta, calculadas para os diversos vértices da superfície e os correspondentes nós da malha. A curvatura analítica é calculada de acordo com a representação matemática da superfície dada pela descrição geométrica de suas curvas e *patches*. A curvatura discreta é uma aproximação dessa curvatura, calculada de acordo com a configuração da malha utilizada para adaptação. Para minimizar o erro entre as curvaturas, a estratégia é capaz de refinar e desrefinar as diversas regiões da malha, concentrando um maior número de elementos em regiões de maior curvatura e mantendo um menor número de elementos em regiões onde a curvatura é mais baixa. Por possuir uma abordagem hierárquica, onde a discretização das curvas se dá de forma independente e antes da discretização do domínio, a estratégia garante uma boa transição entre regiões refinadas e desrefinadas, mantendo a compatibilidade entre regiões adjacentes. A qualidade global da malha gerada é assegurada pela contribuição das medidas dos erros locais calculados para cada nó da malha. Outro ponto importante a ser considerado é que uma vez que a estratégia trabalha no espaço paramétrico, ela pode ser utilizada com qualquer tipo de superfície paramétrica. Os exemplos mostrados neste trabalho demonstraram que a estratégia adaptativa converge para uma malha de boa qualidade mesmo que a malha inicial seja extremamente desrefinada.

As características do método proposto são particularmente importantes para aplicações de computação gráfica e realidade virtual uma vez que elas garantem uma renderização de boa qualidade pela distribuição apropriada dos elementos da malha, concentrando os elementos em regiões de maior curvatura e reduzindo a densidade de elementos em regiões de menor curvatura. Isso pode ser difícil de conseguir, usando os modeladores geométricos disponíveis, tais como 3DStudio e Blender, e pode ser crucial em aplicações em tempo real,

onde a taxa de quadros exerce um papel central. A estratégia também pode ser utilizada em problemas onde os níveis de detalhes (*level of details*) são frequentemente aplicados.

5.2 Trabalhos futuros

O critério adaptativo, baseado na discrepância entre as curvaturas analíticas e discretas, apresentou resultados bastante satisfatórios. Entretanto, um estudo mais aprofundado sobre a aplicação desse critério em modelos *non-manifolds* seria interessante. A utilização de operadores de curvaturas capazes de lidar com esse tipo de superfície tornaria a estratégia ainda mais abrangente.

O uso de padrões triangulares facilita a geração da malha no interior do domínio, pois sua aplicação é feita de forma direta sobre a configuração obtida pela *quadtree* restrita. Contudo, uma vez que esses padrões são definidos no espaço paramétrico e seus ângulos também são definidos nesse espaço, o mapeamento para o espaço tridimensional pode gerar distorções nos elementos. Para evitar esse problema, pode ser interessante aplicar uma técnica de geração adaptativa de malhas que já leve em conta as curvaturas no interior do domínio.

Os exemplos gerados utilizando-se superfícies de Hermite apresentaram bons resultados. Mas devido à natureza genérica da estratégia, é indicado aplicá-la em outros tipos de superfícies paramétricas.

Finalmente, outro ponto que deve ser explorado é a comparação da estratégia proposta por este trabalho com outras estratégias de geração de malhas adaptativas de superfícies paramétricas presentes na literatura. Fazendo-se uso de uma métrica de comparação justa, pode-se ter parâmetros mais adequados para melhor compreender os pontos fortes e fracos da estratégia apresentada.

Referências bibliográficas

- [0] N. Banichuk, F. Barthold, A. Falk, and E. Stein, “Mesh Refinement for Shape Optimization”, *Structural Optimization*, 1995, v. 9, pp. 46-51.
- [1] P. Katragadda and I.R. Grosse, “A Posteriori Error Estimation and Adaptive Mesh Refinement for Combined Thermal-Stress Finite Element Analysis”, *Computers and Structures*, 1996, v. 59, pp. 1149-1163.
- [2] J.B. Cavalcante-Neto, L.F. Martha, I.F.M Menezes and G.H. Paulino, “A Methodology for Adaptive Finite Element Analysis: Towards an Integrated Computational Environment”, *Computation Mechanics*, 1999, v. 23, pp. 361-388.
- [3] C.K. Lee and R.E. Hobbs, “Automatic Adaptive Finite Element Mesh Generation Over Arbitrary Two-Dimensional Domain Using Advancing Front Technique”, *Computers and Structures*, 1999, v. 71, pp. 9-34.
- [4] G.H. Paulino, I.F.M. Menezes, J.B. Cavalcante-Neto, and L.F. Martha, “A Methodology for Adaptive Finite Element Analysis: Towards an Integrated Computacional Environment” *Computational Mechanics*, 1999, v. 23, pp. 361-388.
- [5] M. Hall and J. Warren, “Adaptive Polymerizationon of Implicit Defined Surfaces”, *IEEE Computer Graphics and Applications*, 1990, v. 10, pp. 33-43.
- [6] L. Velho and L.H. Figueiredo, “Optimal Adaptive Polygonal Approximation of Parametric Surfaces”, *SIBGRAP'96*, 1996, pp. 127-133.
- [7] L. Velho, L.H. Figueiredo and J. Gomes, “A Unified Approach for Hierarchical Adaptive Tessellation of Surfaces”, *ACM Transactions on Graphics*, 1999, v. 18, pp. 329-360.
- [8] C.O. Miranda and L.F. Martha, “Mesh Generation on High-Curvature Surfaces Based on a Background Quadtree Structure”, In: *Proceedings of 11th International Meshing Roundtable*, Sandia National Laboratories, 2002, pp. 333-342.
- [9] M. Meyer, M. Desbrun, P. Schröder and A.H. Barr, “Discrete Differential-Geometry Operators for Triangulated 2-Manifolds”, *Visual and Mathematics III*, 2002, pp. 35-57.
- [10] S.J. Kim, W.k. Jeong and C.H. Kim, “LOD Generation with Discrete Curvature Error Metric”, In *Proceedings of Korea Israel Bi-National Conference*, 1999, pp. 97-104.
- [11] F.G. Teixeira and G.J. Creus, “Geração Automática de Malha Sobre Recortes de Superfícies Paramétricas com Grande Curvatura”, *Mecânica Computacional*, 2003, vol. 22, pp. 2305-2319.
- [12] T. Surazhsky, E. Magid, O. Soldea, G. Elber, and E. Rivlin, “A Comparison of Gaussian and Mean Curvatures Estimation Methods on Triangular Meshes,” *Proceedings of IEEE International Conference on Robotics and Automation*, 2003, pp. 1021-1026.
- [13] E. Magid, O. Soldea, and E. Rivlin, “A Comparison of Gaussian and Mean Curvature Estimation Methods on Triangular Meshes of Range Image Data,” *Computer Vision and Image Understanding*, 2007, vol. 107, no. 3, pp. 139-159.

- [14] G. Xu, “Convergence Analysis of a Discretization Scheme for Gaussian Curvature over Triangular Surfaces”, *Computer Aided Geometric Design*, 2006, vol. 23 no. 2, pp. 193-207.
- [15] D. Meek, D. Walton, “On surface normal and Gaussian curvature approximations given data sampled from a smooth surface”, *Computer Aided Geometric Design*, 2000, 17, pp. 521–543.
- [16] M. Meyer, M. Desbrun, P. Schröder, A. Barr, “Discrete differential geometry operator for triangulated 2-manifolds”, In: *Proceedings of Visual Mathematics*, 2002.
- [17] S.H. Lo and T.S. Lau, “Finit Element Mesh Generation over Analytical Curved Surfaces”, *Computers and Structures*, 1996, vol. 59 no. 2, pp. 301-309.
- [18] S.H. Lo and T.S. Lau, “Mesh Generation over Curved Surfaces with Explicit Control on Discretization Error”, *Engineering Computations*, 1998, vol. 15 No. 3, pp. 357-373.
- [19] N. Dyn, K. Hormann, S.J. Kim and D. Levin, “Optimizing 3D triangulations using discrete curvature analysis”, *Mathematical Methods for Curves and Surfaces*, 2000, pp. 135-146.
- [20] D.L. Souza, “Geração Adaptativa de Malhas Baseada em Erro de Curvatura”, *Dissertação (Mestrado em Ciência da Computação) - Universidade Federal do Ceará, Fundação Cearense de Apoio ao Desenvolvimento Científico e Tecnológico. Orientador: Joaquim Bento Cavalcante Neto*, 2004.
- [21] M.C. Lee and M.S. Joun, “Adaptive Triangular Element Generation and Optimization-Based Smoothing: part 2. On the Surface”, *Advances in Engineering Software*, 2008, vol. 39 no. 1, pp. 39-46.
- [22] D.S. Wang, “A New Method for Efficient Generation of High Quality Triangular Surface Meshes”, *Communications in Computational Physics*, 2006, vol. 1 no. 4, pp. 716-735.
- [23] B. Glut and T. Jurczyk, “Mesh Adaptation Based on Discrete Data”, *Parallel Processing and Applied Mathematics*, 2006, 3911, pp. 559-566.
- [24] J.B. Cavalcante-Neto, L.F. Martha, I.F.M. Menezes, and G.H. Paulino, “A Methodology for Self-Adaptive Finite Element Analysis Using an Object Oriented Approach”, *WCCM*, 1998, v. 2, pp. 1-20.
- [25] P. Baehmann, S. Wittchen, and M. Shephard, “Robust Geometrically Based, Automatic Two-Dimensional Mesh Generation”, *Int. J. Num. Meth. Engng.*, 1987, v. 24, pp. 1043-1078.
- [26] D.F. Rogers, J.A. Adams, “*Mathematical Elements for Computer Graphics*”, McGraw-Hill Science/Engineering/Math, 1989, 2nd edition, pp. 460-461.
- [27] J. Dill, “An Application of Color Graphics to the Display of Surface Curvature”, *Computer Graphics (Proc. SIGGRAPH 81)*, 1981, vol. 15, no. 3, pp. 153-161.
- [28] D. Salomon, “*Curves and Surfaces for Computer Graphics*”, Springer Science+Business Media, Inc., 2005, pp. 111-139.

- [29] P.J. Frey, J.H. Saiaç e Frédéric Alauzet “Simulation Numérique em Geométries Complexes: Apport de Techniques D'Adaption de Maillages”, Applications Aux Calculs de Mécanique des Fluides, 2005.
- [30] S.J. Owen, “A Survey of Unstructured Mesh Generation Technology”, 7th Int. Meshing Roundtable, 1999 , pp. 239-67.
- [31] M. Botsh, M. Pauly, C. Rössl, S. Bischoff, L. Kobbelt, P. Alliez and B. Lévy, “Geometric Modeling Based on Polygonal Meshes”, ACM SIGGRAPH, 2007, Course 23, Article no. 1.