UNIVERSIDADE FEDERAL DE MINAS GERAIS

ESCOLA DE ENGENHARIA

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

AJUSTE DA MALHA TRIANGULADA RESULTANTE DAS OPERAÇÕES BOOLEANAS DE UM MODELADOR DE SÓLIDOS

Aluna: Christiane de Lisieux Leal e Mol

Orientador: Professor Renato Cardoso Mesquita

Julho de 2004

UNIVERSIDADE FEDERAL DE MINAS GERAIS

ESCOLA DE ENGENHARIA

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

AJUSTE DA MALHA TRIANGULADA RESULTANTE DAS OPERAÇÕES BOOLEANAS DE UM MODELADOR DE SÓLIDOS

Aluna: Christiane de Lisieux Leal e Mol

Orientador: Professor Renato Cardoso Mesquita

Dissertação de Mestrado submetida à Banca Examinadora designada pelo colegiado do Programa de Pós-Graduação em Engenharia Elétrica de Minas Gerais, como requisito parcial para obtenção de título de Mestre em Engenharia Elétrica.

Área de Concentração: Automática

Linha de Pesquisa: Otimização e Projeto Assistido por Computador

Belo Horizonte – Julho de 2004

"Tenho amigos que não sabem o quanto são meus amigos. Não percebem o amor que lhes devoto e a absoluta necessidade que tenho deles. A amizade é um sentimento mais nobre do que o amor, eis que permite que o objeto dela se divida em outros afetos, enquanto o amor tem intrínseco o ciúme, que não admite a rivalidade".

E eu poderia suportar, embora não sem dor, que tivessem morrido todos os meus amores, mas enlouqueceria se morressem todos os meus amigos! Até mesmo aqueles que não percebem o quanto são meus amigos e o quanto minha vida depende de suas existências... A alguns deles não procuro, basta-me saber que eles existem. Esta mera condição me encoraja a seguir em frente pela vida. Mas, porque não os procuro com assiduidade, não posso lhes dizer o quanto gosto deles. Eles não iriam acreditar.

Muitos deles estão lendo esta crônica e não sabem que estão incluídos na sagrada relação de meus amigos. Mas é delicioso que eu saiba e sinta que os adoro, embora não declare e não os procure. E às vezes, quando os procuro, noto que eles não têm noção de como me são necessários, de como são indispensáveis ao meu equilíbrio vital, porque eles fazem parte do mundo que eu, tremulamente, construí e se tornaram alicerces do meu encanto pela vida.

Se um deles morrer, eu ficarei torto para um lado. Se todos eles morrerem, eu desabo! Por isso é que, sem que eles saibam, eu rezo pela vida deles. E me envergonho, porque essa minha prece é, em síntese, dirigida ao meu bem estar. Ela é, talvez, fruto do meu egoísmo. Por vezes, mergulho em pensamentos sobre alguns deles. Quando viajo e fico diante de lugares maravilhosos, cai-me alguma lágrima por não estarem junto de mim, compartilhando daquele prazer...

Se alguma coisa me consome e me envelhece é que a roda furiosa da vida não me permite ter sempre ao meu lado, morando comigo, andando comigo, falando comigo, vivendo comigo, todos os meus amigos, e, principalmente os que só desconfiam ou talvez nunca vão saber que são meus amigos! A gente não faz amigos, reconhece-os."".

Vinícius de Moraes

Primeiramente, a Deus pelos amigos que conheci e reconheci nessa caminhada, pelas bênçãos e graças conseguidas, pela sabedoria, paz, e fortaleza providenciada para que eu concluísse esse trabalho.

Aos meus pais, Osvaldo e Theresinha, aos meus irmãos Waleska, Vivian e Wellington, aos meus avós e primos pelo amor, carinho, paciência, companheirismo, convívio, e por estarem ao meu lado sendo meus mestres.

Ao Felipe pelos momentos de descontração.

Agradecimentos

Ao Professor Renato Cardoso Mesquita, pela orientação, ensinamentos, competência e disponibilidade durante a realização deste trabalho.

A grande amiga, Ana Liddy C. de C. Magalhães, que sempre foi uma verdadeira mãe, por ter acreditado no meu esforço, talento e competência quando me selecionou para participar do projeto de pesquisa do doutorado o qual propiciou e fomentou o desenvolvendo desse trabalho de mestrado.

Aos amigos, Adriano C. Lisboa, Cássia R. S. Nunes, Frederico B. R. Soares, Glauce F. e Lourenço, Maristela C. de M. Gaia, Newton C. Selani e Pablo S. Oliveira que conviveram comigo durante a graduação e me ajudaram, de uma maneira ou de outra, a ingressar no mestrado e a concluí-lo.

Aos amigos, Ricardo Ramos, Márcio Matias, Ana Rodrigues, Anderson Rabello, Thierry Toledo, Sérgio Ávila, Tiago Falqueto, Felipe Terra, Otávio Machado e demais integrantes do GOPAC, que sempre estiveram ao meu lado me apoiando, incentivando, cooperando e me enriquecendo com seus convívios e companheirismo.

Aos amigos do CPDEE, Rojane, Parma, Mariza, Hissa, Amilton, Marco Aurélio, Antônia, Glauco, Tarcísio, Dora, Carla, Evandro, Aurea, Waldir, Maria Eugênia e Flavinha, e do CPH, Doriana, Finzi, Hersília, Erivelton, Rafael, Hiara, Darlene e Martinez, pelas horas de descontração, convívio, apoio, compreensão e ajuda.

Aos demais colegas, alunos, professores e funcionários do CPDEE/UFMG e FADOM que, ao longo de todo o desenvolvimento desse trabalho, sempre auxiliaram em tudo que foi possível.

Ao CNPq – Conselho Nacional de Desenvolvimento Científico e Tecnológico - pela concessão da bolsa de Estudos durante parte da realização desse trabalho.

Resumo

Este trabalho está inserido em um contexto maior que envolve o desenvolvimento de um modelador de sólidos (GSM – *Gopac Solid Modeler*), que é utilizado na definição da geometria de modelos computacionais 3D para aplicação em problemas eletromagnéticos. O GSM gera malhas superficiais triangulares sobre objetos sólidos, a serem utilizadas como entrada para geradores de malhas volumétricas.

Os modelos gerados pelo modelador possuem malha superficial inicial de boa qualidade, mas, após a aplicação de uma operação Booleana ou operação de montagem, a malha resultante, geralmente, apresenta elementos mal formados. Elementos com áreas ou ângulos muito pequenos podem ser formados durante a determinação dos pontos de interseção, resultando em uma malha de baixa qualidade. Tentando minimizar o aparecimento desses elementos na malha, foram implementados o método de suavização laplaciana e os algoritmos que juntam os elementos da malha (ponto-ponto e aresta-aresta) que distam entre si de uma tolerância de entrada calculada pelo modelador.

Este trabalho apresenta a fundamentação teórica necessária para aplicar os métodos para ajustar os elementos durante o processo de determinação dos pontos de interseção e geração da malha nas operações booleanas. Em seguida, discutem-se alguns métodos adotados para gerar melhorias na malhas e para finalizar, são apresentados e analisados os resultados obtidos da implementação dos métodos de ajuste da malha.

Abstract

This work presents a new method and algorithm to optimize the quality in finite element meshes. Its related to the GSM – Gopac Solid Modeler – used to define 3D computational geometry in electromagnetic problems. The GSM generates triangular superficial meshes over solid objects that will be employed by volumetric mesh generators.

Usually, the Boolean Operations and Assembly Operations yields the badly formed elements, these may present small angle or area during the determination of the intersection points between two regions meshed.

The method of “Suavização Laplaciana” has been proposed to minimize this problem. However, in order to improve the quality of the mesh, an algorithm to join its elements (point-point, edge-edge and edge-face) may be used to control the intersection point determination. The distance between the elements is smaller than the input tolerance defined by the modeler.

The results show the reduction of the badly formed elements into the intersection regions, and the improvement of the mesh quality.

Sumário

Lista de Figuras

Lista de Quadros

Lista de Abreviaturas e Siglas

B-rep: Boundary Representation
CAD: Computer Aided Design
CAE: Computer Aided Engineering
CSG: Constructive Solid Geometry
FEM: Finite Element Method
GOPAC: Grupo de Otimização e Projeto Assistido por Computador
GSM: Gopac Solid Modeler
MEF: *Make Edge Face*
MEV: *Make Edge Vertex*
MVFR: *Make Vertex Face Region*
STL: Standard Template Library

Capítulo I – Introdução

Desde os primórdios da civilização, é possível observar a necessidade que o homem tem de representar as coisas ao seu redor. Seja por meio de desenhos ou de esculturas para serem utilizadas nas mais variadas aplicações. Como prova disso, é possível citar os desenhos de animais encontrados por vários arqueólogos em muitas cavernas. Estudos mostram que o homem pré-histórico se utilizava do artifício de fazer desenhos nas paredes das cavernas onde residia, pois acreditava que, dessa forma, ao sair para a caça, os Deuses providenciariam os animais ali representados e, caso não houvesse os desenhos, os Deuses não providenciariam o alimento.

Com o passar dos anos, o homem evolui, descobre o universo, expande seus territórios, inventa armas e máquinas e desenvolve a navegação, crescendo assim a sua necessidade de representar as coisas mundanas. O homem passa a criar projetos para desenvolver novas armas, ferramentas e embarcações. Começa a estudar e representar, por meio de desenhos e protótipos, o sistema solar. Desenha os primeiros mapas na tentativa de representar o espaço físico a sua volta.

Assim que surge o computador, essa necessidade de representação do mundo sofre uma adaptação. Tudo que antes era moldado pela mão humana passa a ser modelado pelo ou para o computador, como ilustrado na Figura I-1. Surgem assim estudos e projetos para se criarem ferramentas computacionais (sistemas de modelagem ou sistemas CAD) capazes de produzir réplicas de um produto real ou projetar um produto a ser construído. Para que isso seja possível, é preciso que um modelo computacional possua um nível de detalhamento e informações tal que seja o mais próximo do objeto do mundo real, tanto nas suas características físicas quanto nas geométricas, além de atender às necessidades impostas pela aplicação.

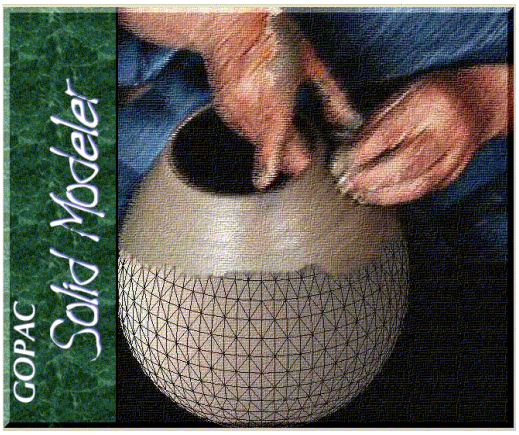


Figura I-1 – Tela de abertura do GSM

O trabalho que será apresentado nessa dissertação está inserido em um sistema CAD – *Computer Aided Design* – denominado GSM – *Gopac Solid Modeler* – que consiste em um Modelador de Sólidos a ser utilizado como pré-processador em aplicações de cálculo de campo eletromagnético empregando o Método de Elementos Finitos (FEM - *Finite Element Method*). O Modelador foi projetado e desenvolvido, inicialmente, pela aluna Ana Liddy Cenni de Castro Magalhães em seu trabalho de Doutorado [MAG00]. Posteriormente, foi complementado pela aluna Cássia Regina Santos Nunes em seu trabalho de Mestrado, que implementou as Operações Booleanas e de Montagem [NUN02].

Nesse trabalho, será feito o estudo e a implementação de algoritmos capazes de fazer uma análise e ajuste da malha superficial resultante das operações booleanas. O objetivo primordial desse trabalho será obter uma malha superficial de boa qualidade, que sirva de ponto de partida para a obtenção de malha volumétrica.

I.1 – Conceitos Envolvendo a Modelagem de Sólidos

Em geral, um **Modelo** é um objeto construído artificialmente, descrevendo uma forma fechada, sólida e tridimensional, com a finalidade de tornar mais fácil a observação de um outro objeto [MÄN88]. Num modelo, tenta-se abstrair apenas as informações essenciais para a aplicação. O conjunto total de dados aplicados a uma classe particular de problemas é denominado de **modelo do objeto**, enquanto os

dados puramente geométricos constituem o **modelo geométrico**. A coleção de métodos usados para definir a forma e outras características geométricas de um objeto é o que pode ser conhecida por **modelagem geométrica** [OLI91], podendo ser dividida em várias ramificações, entre as quais se destacam:

- **Modelagem gráfica**: visa descrever o "desenho do objeto" em termo de vértice e arestas, sem referenciar faces e outras características do objeto real.
- **Modelagem de superfícies**: fornece informação detalhada sobre superfícies curvas complexas, mas nem sempre descreve um volume fechado. Essa modelagem fornece também informação suficiente para determinar todas as propriedades geométricas do objeto definido.
- **Modelagem de sólidos**: é o conjunto de teorias, técnicas e sistemas que focalizam uma representação de sólidos "informacionalmente completa". Permite que qualquer propriedade geométrica bem definida, de qualquer sólido representável, seja automaticamente calculada [REQ83].

Um **sistema de modelagem geométrica** é um sistema computacional que permite a criação, modificação e o acesso à representação de objetos sólidos por meio de modelos geométricos.

Um **modelador de sólidos** é um sistema computacional de modelagem que associa entidades geométricas (sólidos abstratos) a representações simbólicas, mediante esquema de representação [MAG00]. A importância dos sistemas de modelagem de sólidos está, principalmente, em sua capacidade de distinguir entre o interior, o exterior e a superfície de um objeto tridimensional, o que faculta calcular propriedades desta distinção [MOR85].

I.2 – Métodos de Elementos Finitos e Qualidade da Malha

O método de elementos finitos (FEM) é uma ferramenta que simula, normalmente, um fenômeno ou um processo físico modelado por meio de um conjunto de equações envolvendo derivadas parciais e condições de contorno

[SAB93]. É o método numérico universalmente mais usado para cálculo de campo, devido a sua flexibilidade, ligação direta com a física dos fenômenos estudados e a facilidade de programação.

Para aplicação do FEM é necessária a realização da discretização do modelo que consiste na divisão do domínio em subdomínios respeitando as fronteiras e interfaces do domínio inicial. A Figura I-2 traz um bom exemplo.

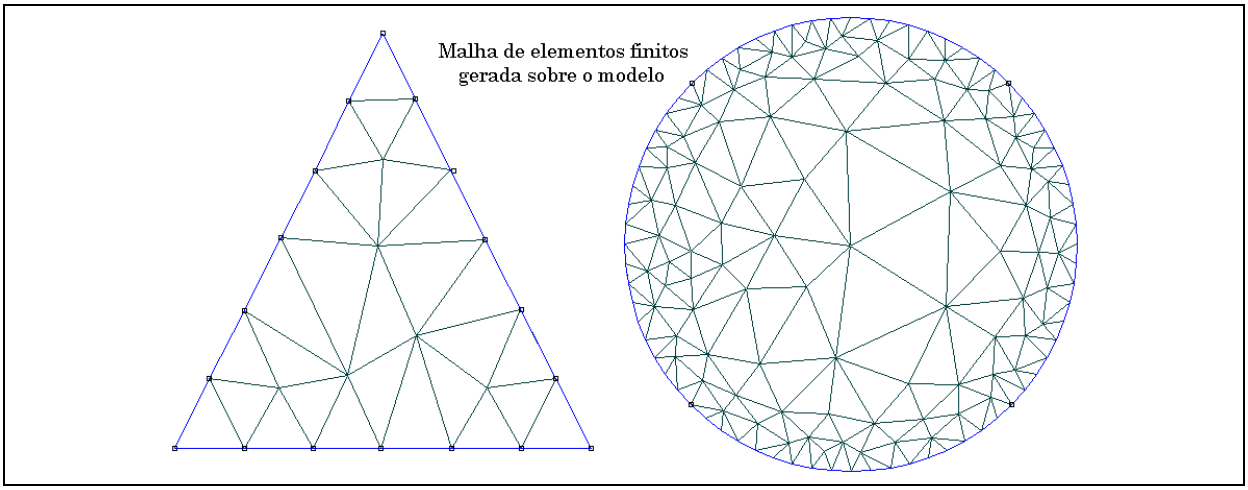


Figura I-2 – Discretização gerando a malha de elementos finitos

A discretização do modelo implica no aparecimento de uma malha composta de um certo número de nós onde são definidos os graus de liberdade utilizados nas equações de elementos finitos.

Durante a discretização do modelo, é de fundamental importância manter a continuidade entre os elementos e garantir a compatibilidade entre as diversas malhas que podem constituir o modelo, como no caso do cone, onde a malha lateral é gerada separadamente da malha de sua tampa.

Na utilização do FEM, a obtenção de resultados satisfatórios está diretamente relacionada com a qualidade da malha final de elementos finitos [RAM90]. O tempo de execução vai depender do número de triângulos presentes na malha. Além disso, a estabilidade e a convergência do método são bastante afetadas pela forma dos triângulos [REI02]. Por isso, a malha de elementos finitos resultante da discretização do modelo deve apresentar uma boa qualidade, ou seja, seus elementos devem ser os mais próximos possíveis de sua forma ideal (triângulos eqüiláteros, quadrados,

tetraedros eqüiláteros, cubos, etc...). Sendo assim, para avaliar se um modelo possui uma malha de boa qualidade, verifica-se a forma de seus elementos.

Visando melhorar a qualidade da malha, muitas vezes torna-se necessário aplicar algum pós-processamento da malha inicial. De forma geral existem dois tipos de pós-processamento:

- Tratamento da forma: garantir que os elementos da malha sejam os mais próximos do ideal. Um método a ser aplicado é a suavização laplaciana – o deslocamento do nó central do polígono para o seu baricentro, como representado na Figura I-3 [SAB93].
- Densidade da malha: obtida pela redução do tamanho dos elementos que constituem a malha.

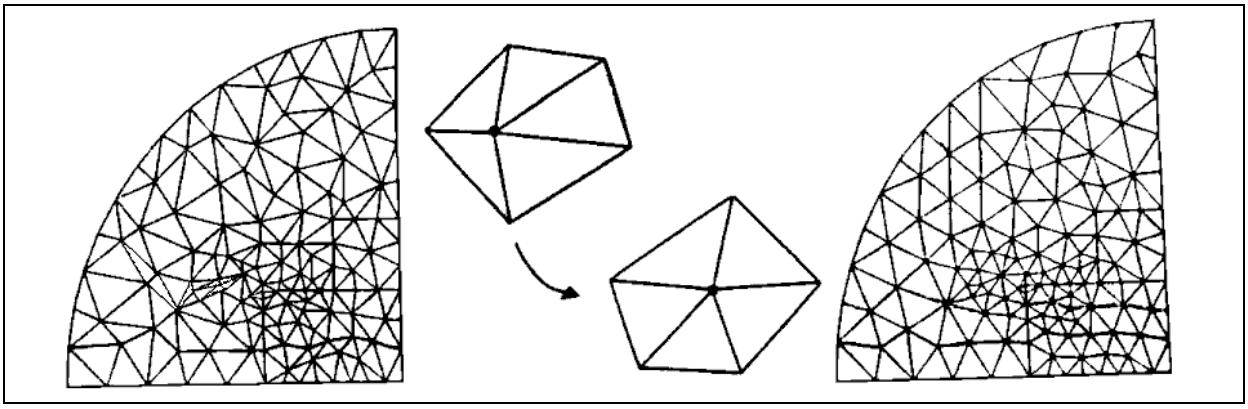


Figura I-3 – Ajuste da forma, usando a Suavização Laplaciana

I.3 – Motivações e Objetivos

O GSM é um sistema que está sendo desenvolvido pelo GOPAC – Grupo de Otimização e Projeto Assistido por Computador – no Centro de Pesquisa e Desenvolvimento em Engenharia Elétrica da Universidade Federal de Minas Gerais – CPDEE/UFMG, sob orientação do Professor Renato Cardoso Mesquita e com a participação de alunos de mestrado, doutorado e iniciação científica. O sistema é um modelador de Sólidos voltado para aplicações em eletromagnetismo, basicamente, cálculo de campos por meio do método de elementos finitos.

O modelador é composto por quatro subsistemas distintos: o de interface, o de modelagem, o de representação e o de geração de malha. Ele suporta a construção de primitivas mais complexas, a partir da combinação de primitivas básicas, por meio das operações booleanas e de montagem.

A necessidade do desenvolvimento desse trabalho surgiu a partir do momento em que se observou que a malha resultante das operações booleanas não apresentava uma boa qualidade. A malha sobre os modelos apresenta, geralmente, triângulos mal formados de áreas desprezíveis e ângulos muito pequenos. Tal malha não serve para a aplicação do método de elementos finitos, que é o objetivo principal do GSM. Simulações utilizando o FEM necessitam de uma malha final de elementos finitos de boa qualidade com um razoável número de triângulos bem formados de maneira a garantir a sua convergência, o seu tempo de execução e obter resultados satisfatórios.

Para o GSM, a presença de um único triângulo mal formado que possua a área ou o ângulo muito pequeno é empecilho na obtenção da malha volumétrica. A geração da malha volumétrica, no GSM, é realizada por um programa gerador de malha 3D, denominado Tetgen, que foi incorporado ao modelador. Esse malhador volumétrico possui algumas restrições quanto à forma dos elementos da malha de entrada, cuja descrição mais detalhada pode ser obtida em [HAN02]. Devido a essas restrições, para muitos modelos resultantes das operações booleanas, não é possível gerar a malha volumétrica.

Este trabalho visa: (i) aplicar métodos e algoritmos capazes de controlar os pontos da malha durante o processo de determinação da interseção sobre os modelos, quando aplicada uma operação booleana, (ii) ajustar a forma dos elementos da malha e (iii) avaliar a qualidade da malha de um modelo, com o intuito de garantir uma melhoria na malha final sobre os modelos.

Entretanto, para que esses métodos pudessem ser aplicados, modificações prévias no modelador foram necessárias. As principais modificações feitas foram: (i) a troca da lista usada na estrutura B-rep implementada, utilizando a lista da biblioteca padrão do C++, por uma nova lista independente que possui uma “cabeça”

cujo manuseio é feito, exclusivamente, pela lista; (ii) substituição dos arquivos utilizados no processo de determinação dos pontos de interseção por uma estrutura na memória.

I.4 – Limites deste Trabalho

A finalidade desse trabalho é apresentar e abordar as metodologias de melhoria da qualidade da malha resultante das operações booleanas adicionadas ao modelador.

Não serão abordados nem estratégia de ajuste de malha que utilize informações sobre as curvas da superfície geradora do modelo e nem métodos que façam o refinamento da malha, aumentando sua densidade. Assim, quando for muito discrepante a densidade da malha de ambos os objetos participantes da operação, a malha resultante pode vir a apresentar uma baixa qualidade, embora possa servir de ponto de partida para se obter a malha volumétrica. Tais estratégias estão sendo trabalhadas em tese de doutorado em desenvolvimento no GOPAC [NUN04].

I.5 – Organização do Trabalho

Este trabalho está dividido em 6 capítulos.

O Capítulo II faz uma apresentação sucinta do Modelador de sólidos GSM descrevendo suas funcionalidades e os quatro subsistemas que o compõem. Descreve as operações booleanas e operação de montagem, mostrando as características de implementação e o processo da interseção.

O Capítulo III descreve importantes alterações realizadas no modelador de sólidos para corrigir erros existentes e melhorar seu desempenho. Serão detalhadas as mudanças realizadas, bem como os motivos de realização de tais mudanças.

O Capítulo IV apresenta uma descrição das metodologias utilizadas para fazer os ajustes na malha, visando a melhoria de sua qualidade. O Capítulo V apresenta os resultados obtidos quando os métodos implementados são aplicados.

O Capítulo VI traz as conclusões do trabalho e, finalizando, é apresentada a bibliografia consultada, tanto para o referencial teórico quanto para o metodológico.

Capítulo II – Descrição Geral do GSM

Nesse capítulo, será apresentada uma breve descrição do modelador de sólidos que está sendo desenvolvido pelo Gopac, de maneira a explicitar o contexto no qual este trabalho está inserido, possibilitando um maior entendimento quando forem descritas as alterações realizadas no modelador para seu melhor funcionamento.

A seção II.1 descreve as principais características presentes no GSM, suas funcionalidades e descreve o arquivo neutro, base de dados definida pelo GOPAC. Nas seções II.2, II.3, II.4 e II.5 serão descritos os quatro subsistemas que compõem o GSM, bem como suas características. Na última seção, II.6, serão descritos o processo e a definição das operações booleanas e de montagem e sua importância no Modelador.

II.1 – Principais Características

O GSM é um sistema CAD de modelagem de sólidos voltado para aplicações em eletromagnetismo, basicamente, cálculo de campos por meio do método de elementos finitos. Esse modelador foi projetado para ser usado em aplicações na área de eletromagnetismo. Portanto, apresenta características como as que são descritas em [MAG99b], [MAG00] e [REQ80], dentre elas:

- Gerar sólidos representáveis (válidos): sólidos com propriedades de rigidez, homogeneidade tridimensional, finitude, fechamento sob operações, finitude de descrição e determinismo de fronteiras;
- Fazer representação de modelos bi ou tridimensionais por meio de esquemas com propriedade de não ambigüidade (completeza), unicidade, potência descritiva, validade, concisão, facilidade descritiva e eficiência do ponto de vista das aplicações;

- Suportar o conceito de faces. Estas devem ser limitadas, de área finita, orientáveis, conexas e homogêneas, além de não se auto-interceptarem;

Além das características citadas acima, o modelador apresenta algumas funcionalidades interessantes, a saber:

- Gera perfis compostos por um ou mais contornos;
- Cria os sólidos pela instanciação de primitivas ou por meio de varredura translacional ou rotacional de perfis abertos ou fechados;
- Gera perfis bidimensionais combinando primitivas 2D que podem sofrer transformações geométricas, o que é intuitivo e de fácil utilização pelo usuário;
- Gera, a partir do modelo CSG, a representação por fronteira. A estrutura B-rep é construída por meio da identificação e inserção dos vértices, arestas e faces durante a avaliação da fronteira.
- Define uma malha superficial triangular de boa qualidade, durante a criação das primitivas, que é usada para a geração da malha volumétrica.
- Permite associar aos nós, às arestas e aos elementos superficiais da malha características do problema que serão utilizadas nas simulações eletromagnéticas, como detalhes do cálculo, condições de contorno e propriedades físicas dos materiais.
- Define sólidos manufaturáveis que podem conter fronteiras internas.
- Permite visualização colorida dos modelos sob vários pontos de vista, além de suportar visões fio-de-arame, com ou sem remoção de linhas escondidas;
- Faz o intercâmbio de dados com outros sistemas por meio da geração de uma base de dados denominada pelo GOPAC de arquivo neutro, cuja descrição se encontra em [ROC95];
- Obtém modelos complexos a partir da união, interseção, diferença ou montagem de dois sólidos, que podem ser primitivas básicas ou modelos complexos obtidos das operações booleanas.

O GSM é composto por quatro subsistemas independentes - Interface, Modelagem, Representação e Geração de Malha – cuja apresentação, características e descrição serão relacionadas nas subseções a seguir. Esses subsistemas interagem entre si conforme ilustrado na Figura II-1.

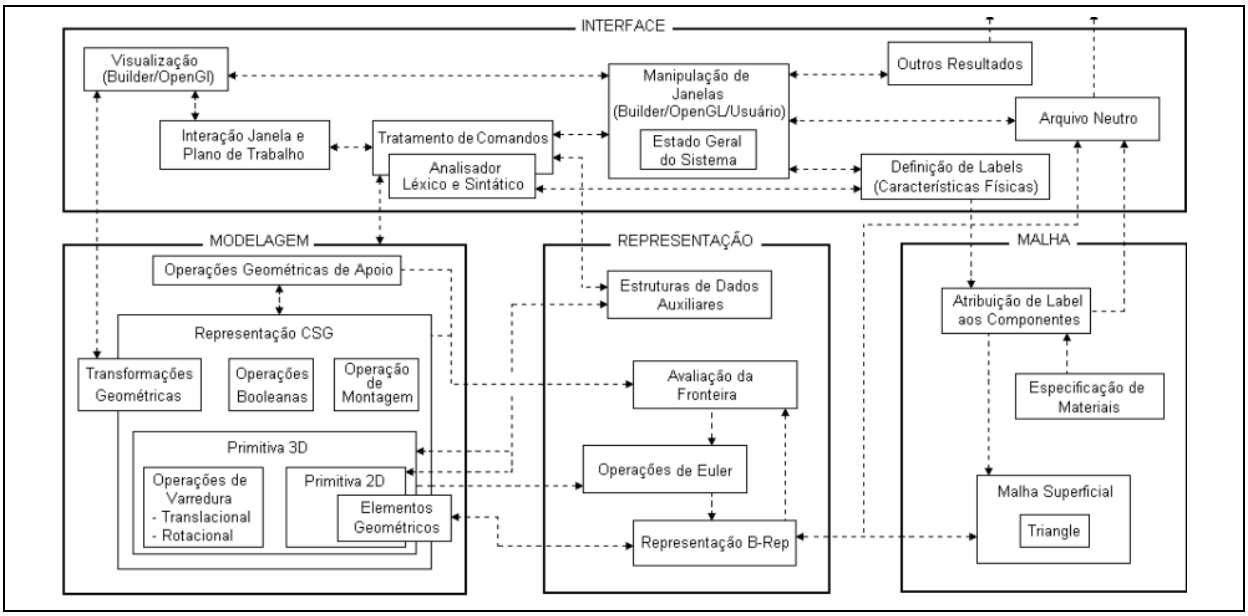


Figura II-1 – Estrutura modular do GSM

II.2 – Subsistema de Interface

A interface de um sistema é o elemento responsável por estabelecer o seu diálogo com o usuário. Sendo assim, uma interface para permitir a comunicação entre o usuário e o ambiente deve ser simples, estruturada e de sintaxe concisa e coerente. Deve possibilitar a visualização do modelo, permitir a comunicação do modelador com outras aplicações e ser de fácil uso, permitindo agilizar o trabalho do usuário.

A tela principal do GSM possui uma área gráfica de trabalho, uma área de menus, uma área de exibição do estado corrente do sistema e uma área para receber a entrada de dados. A Figura II-2 mostra o *layout* da tela principal da interface utilizada no GSM.

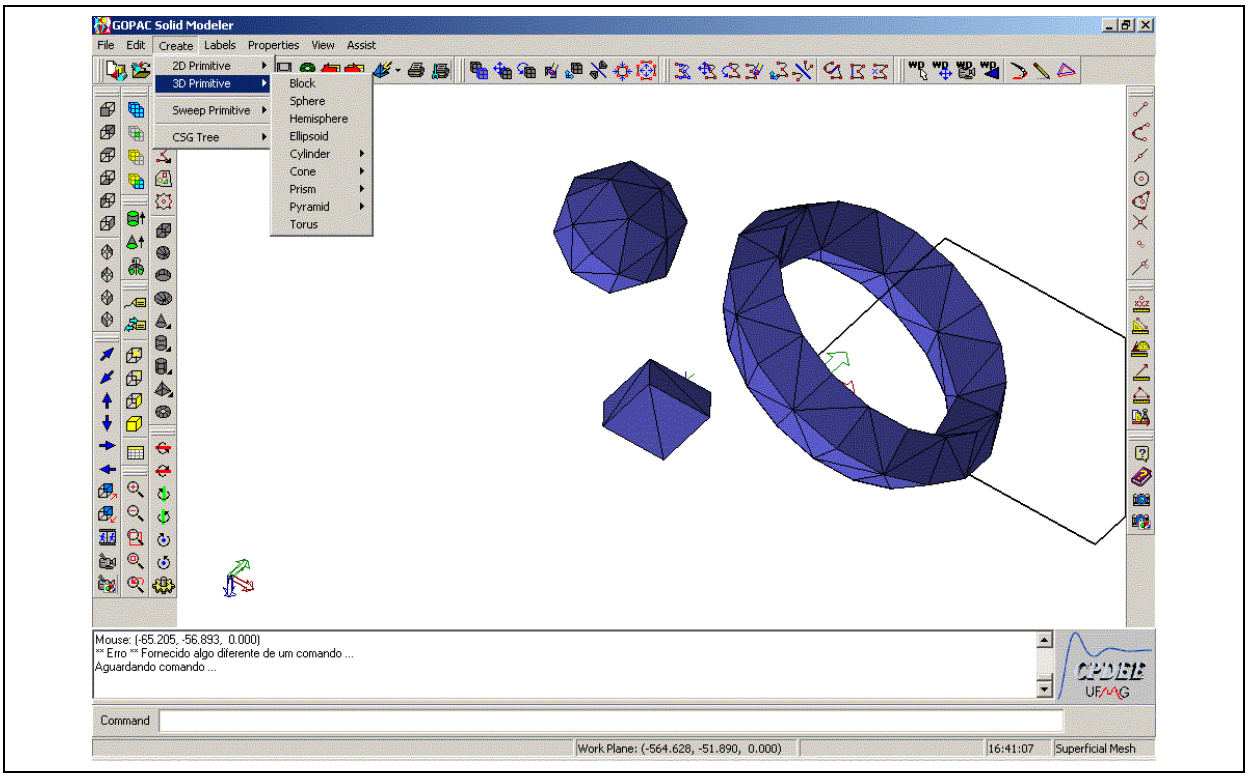
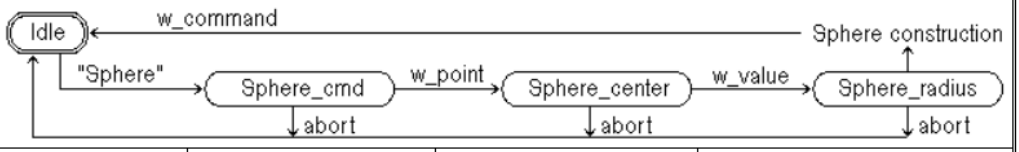


Figura II-2 – *Layout* da tela principal da interface do GSM

Os comandos no modelador podem ser ativados pela seleção de opções dos menus, por ícones disponíveis nas barras de ferramentas ou por linhas de comandos textuais. Neste último caso, é feito o tratamento dos comandos por meio de um analisador léxico e sintático responsável por verificar a validade dos parâmetros e comandos fornecidos pelo usuário, e uma máquina de estados finitos - mostrada no Quadro II-1 - que faz o controle da forma e seqüência em que parâmetros devem ser fornecidos para o sistema.

REPRESENTAÇÃO GRÁFICA				
STATUS ATUAL	IDLE	SPHERE_CMD	SPHERE_CENTER	SPHERE_RADIUS
STATUS ESPERA	W_COMMAND	W_POINT	W_VALUE	W_COMMAND
MENSAGEM DO SISTEMA	Aguardando comando...	Sphere: Forneça o centro da esfera:	Sphere: Forneça o raio da esfera	Sphere: Gerando a esfera...
MENSAGEM DE ERRO	* Erro * Comando inválido !	* Erro * Sphere: centro ou ponto inválido	* Erro * Sphere: raio ou ponto inválido	* Erro * Sphere: geração não foi bem sucedida

Quadro II-1 – Máquina de estados finitos para o comando sphere. [MAG99c]

A necessidade do uso da máquina de estados finitos se deve ao fato do sistema de janelas do Windows trabalhar com eventos que são executados um de cada vez. Não é possível iniciar um evento estando dentro de outro evento. Por exemplo, se o usuário digita "*sphere*" na linha de comando, o GSM verifica se este comando há na lista de comandos. Se existir, ativa o evento "*CMD_SPHERE*" para construção da esfera. Dentro do evento deveria ser pedido que o usuário fornecesse o centro e o raio da esfera. Entretanto, como a obtenção das coordenadas do centro e o valor do raio são outros dois eventos, eles não podem ser executados durante a construção da esfera. Para que o sistema saiba que, ao finalizar o evento "*CMD_SPHERE*", ele deve esperar pela entrada das coordenadas do centro e, depois, pelo valor do raio, deve haver uma estrutura que permita fazer esse controle. A máquina de estados finitos considera todas as condições que podem ocorrer quando se deseja passar de um estado para outro. A transição entre dois estados está associada à ocorrência de eventos.

II.3 – Subsistema de Modelagem

Este é o subsistema responsável pela manipulação da forma dos componentes do modelo, tendo como principal função fazer a tradução das descrições dos objetos e operações vindas da interface em forma de comandos para a geração e edição das representações internas. Dessa forma, garante a criação válida e edição correta das primitivas, permitindo realizar sobre elas as operações booleanas, a operação de montagem e as transformações geométricas, bem como dar suporte à visualização dos modelos.

O subsistema de modelagem é responsável pela construção de modelos CSG, disponibilizando os procedimentos necessários para a construção correta da representação B-rep correspondente, além de permite criar, editar ou alterar a estrutura de representação do objeto.

II.3.1 – GEOMETRIA SÓLIDA CONSTRUTIVA - CSG

O GSM utiliza o modelo CSG na representação dos objetos devido ao fato desse esquema ser capaz de representar modelos complexos consumindo pouco espaço de memória, ser conciso, não ambíguo e possibilitar a realização de cálculos de FEM de maneira precisa.

A modelagem de sólidos tem como princípio a Geometria Sólida Construtiva que se baseia na construção de sólidos complexos a partir de sólidos regulares simples combinados entre si por meio de um conjunto regularizado de operações booleanas (união, interseção e diferença) ou de transformação (translação, rotação ou escalonamento). A representação CSG convencional utiliza uma árvore binária para representar seu objeto, na qual cada folha é uma primitiva e cada nó interno pode ser uma operação booleana ou uma transformação geométrica. Para o modelador, foi utilizada uma variação dessa estrutura CSG de maneira a incluir a operação de montagem, que é muito importante para a área do eletromagnetismo, pois permite modelar partes de um objeto que possuem material diferente e que estão em contato direto e permite até mesmo modelar o ar que envolve esse objeto.

A Figura II-3 mostra um exemplo de árvore CSG que é utilizada no modelador para a obtenção de um modelo complexo. É possível observar que essa estrutura não possui representação única, pois o mesmo objeto complexo pode ser obtido a partir de árvores diferentes.

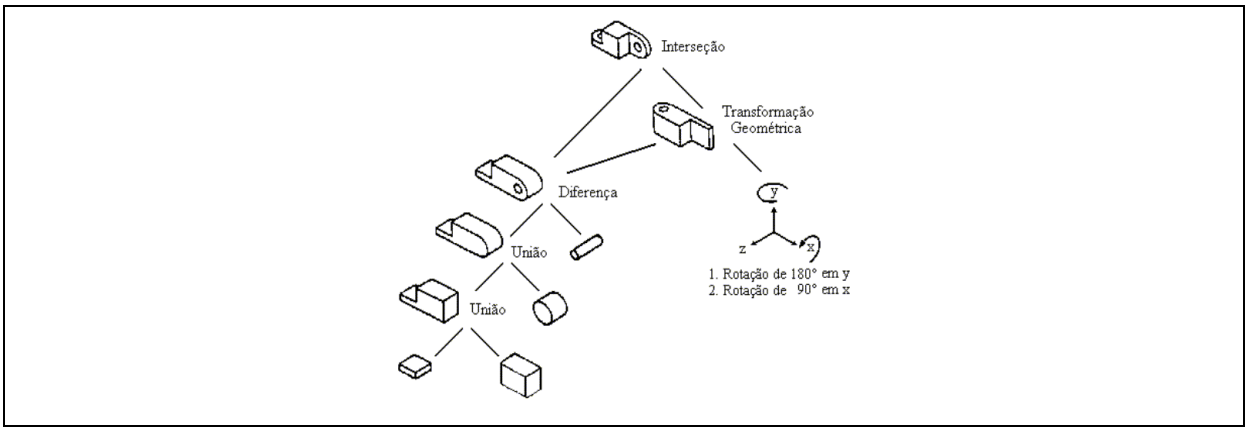


Figura II-3 – Exemplo da árvore CSG utilizada no GSM

II.4 – Subsistema de Representação

Também conhecido como o Núcleo do sistema, o subsistema de representação é o encarregado de gerar e de manipular a estrutura de dados B-rep – *Boundary Representation* – que descreve o modelo por meio de sua fronteira. É responsável pelo gerenciamento e acesso direto às principais estruturas de dados que o compõem e pelo cuidado de todos os acessos solicitados pelos demais subsistemas, de forma a garantir a integridade das estruturas de dados e fornecer informações para comunicação com outros modeladores e aplicações externas.

Representar um objeto por meio de sua fronteira significa criar uma hierarquia de componentes onde geometrias de maiores dimensões são delimitadas por meio de geometrias de menores dimensões, ou seja, um sólido é delimitado por uma superfície composta por um conjunto de faces, que são delimitadas por arestas, que por sua vez são delimitadas por vértices. Sendo assim, é possível gerar superfícies que definem uma partição do espaço completa, fechada e orientada, capaz de informar com exatidão a posição espacial do sólido.

No modelador de sólidos GSM, a estrutura de dados B-rep que faz representação por fronteira de um sólido seja obtida a partir da avaliação da fronteira do modelo CSG associado ao sólido em questão.

II.4.1 – A ESTRUTURA DE DADOS B-REP

A estrutura de dados B-rep presente no GSM foi projetada com o propósito de armazenar informações topológicas, geométricas e físicas do modelo e do problema a ser resolvido. Ela utiliza o conceito de s-set [ARB90] acoplado aos conceitos de semi-aresta e de “uso” da face, aresta e vértice como definidos por Ana Liddy em sua tese de doutorado [MAG00]. Os conceitos presentes nesta estrutura são ilustrados na Figura II-4. Como se observa, as semi-arestas definem a orientação das faces e os usos da face, da aresta e do vértice - vão definir a representação correta para a orientação da face em relação a uma região, o sentido da semi-aresta de uma aresta e o vértice associado a cada semi-aresta, respectivamente.

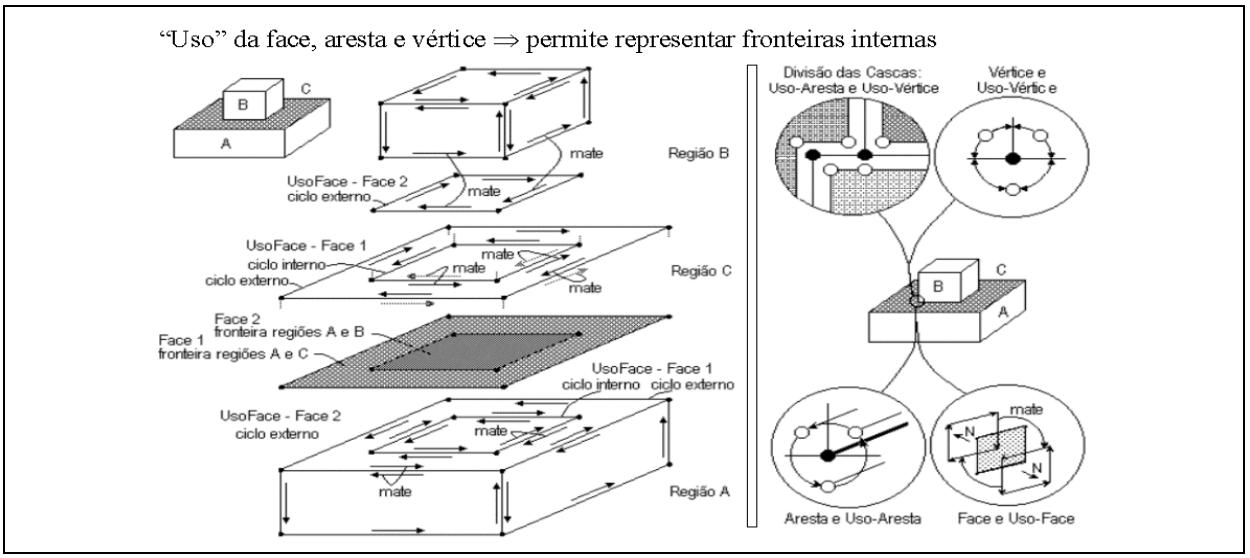


Figura II-4 – Conceito de "uso".

A estrutura B-rep definida para o modelador, ilustrada na Figura II-5, é responsável por fazer a representação da geometria e da topologia de um modelo por fronteira. Nela são armazenados elementos primitivos como vértice, arestas, faces, ciclos, cascas e regiões que descrevem o modelo. A B-rep é implementada a partir de listas circulares duplamente encadeadas, que possuem um nó cabeça e nós comuns onde são armazenados os elementos primitivos.

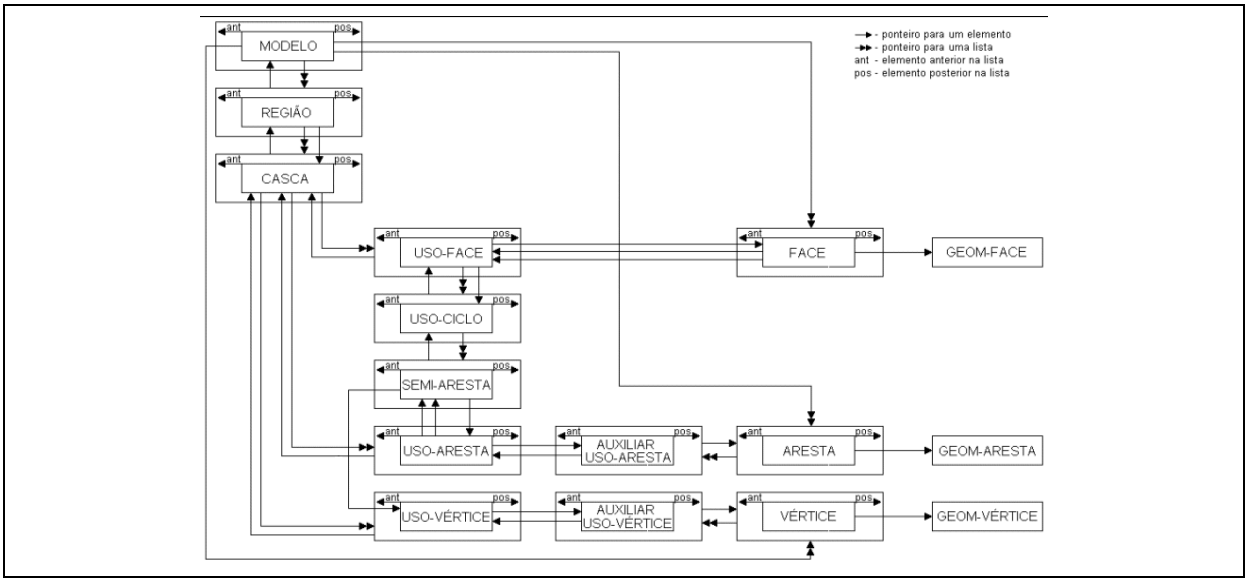


Figura II-5 – Estrutura do B-rep

A construção, manipulação e destruição da estrutura de dados B-rep é feita por meio dos operadores de Euler, que garantem a integridade topológica dos objetos gerados [MAG94]. Os operadores de Euler definidos e implementados no modelador estão apresentados no Quadro II-2 e representados na Figura II-6.

Nível	Operador	Funcionalidade
Baixo	MVFR	Constrói um vértice, uma face e uma região. Primeiro operador utilizado para gerar um modelo ou uma nova região.
	KVFR	Operador inverso ao MVFR, pois destrói um vértice, uma face e uma região.
	MEV	Constrói uma aresta e um vértice.
	KEV	Operador inverso ao MEV, pois destrói uma aresta e um vértice.
	MEF	Constrói uma aresta e uma face.
	KEF	Operador inverso ao MEF, pois destrói um vértice e uma face.
	MEKL	Constrói uma aresta e destrói um ciclo.
	KEML	Destrói uma aresta e constrói um ciclo.
	MFKLH	Constrói uma face e destrói um ciclo que define um buraco.
	KFMLH	Destrói uma face e constrói um ciclo que define um buraco.
	MSKR	Constrói uma casca que define um buraco e destrói uma região.
	KSMR	Destrói uma casca que define um buraco e constrói uma região.
	MFFR	Constrói uma região e duas faces. Esse operador é utilizado para dividir uma região em duas.
	KFFR	Operador inverso ao MFFR, pois destrói duas faces e uma região. Faz duas regiões, que possuem uma face colada, ser uma única região.
Alto	AR	Junta duas regiões por duas faces iguais
	DR	Separa duas regiões contendo face em comum
	AF	Após AR, junta duas outras faces iguais
	DF	Separa uma face em duas, antes de DR
	GR	Cola duas regiões por duas faces iguais
	UR	Descola região por uma face
	GF	Após UR, cola duas outras faces iguais
	UF	Descola uma face -vira duas– aplica antes de UR

Quadro II-2 – Operadores de Euler presentes no GSM

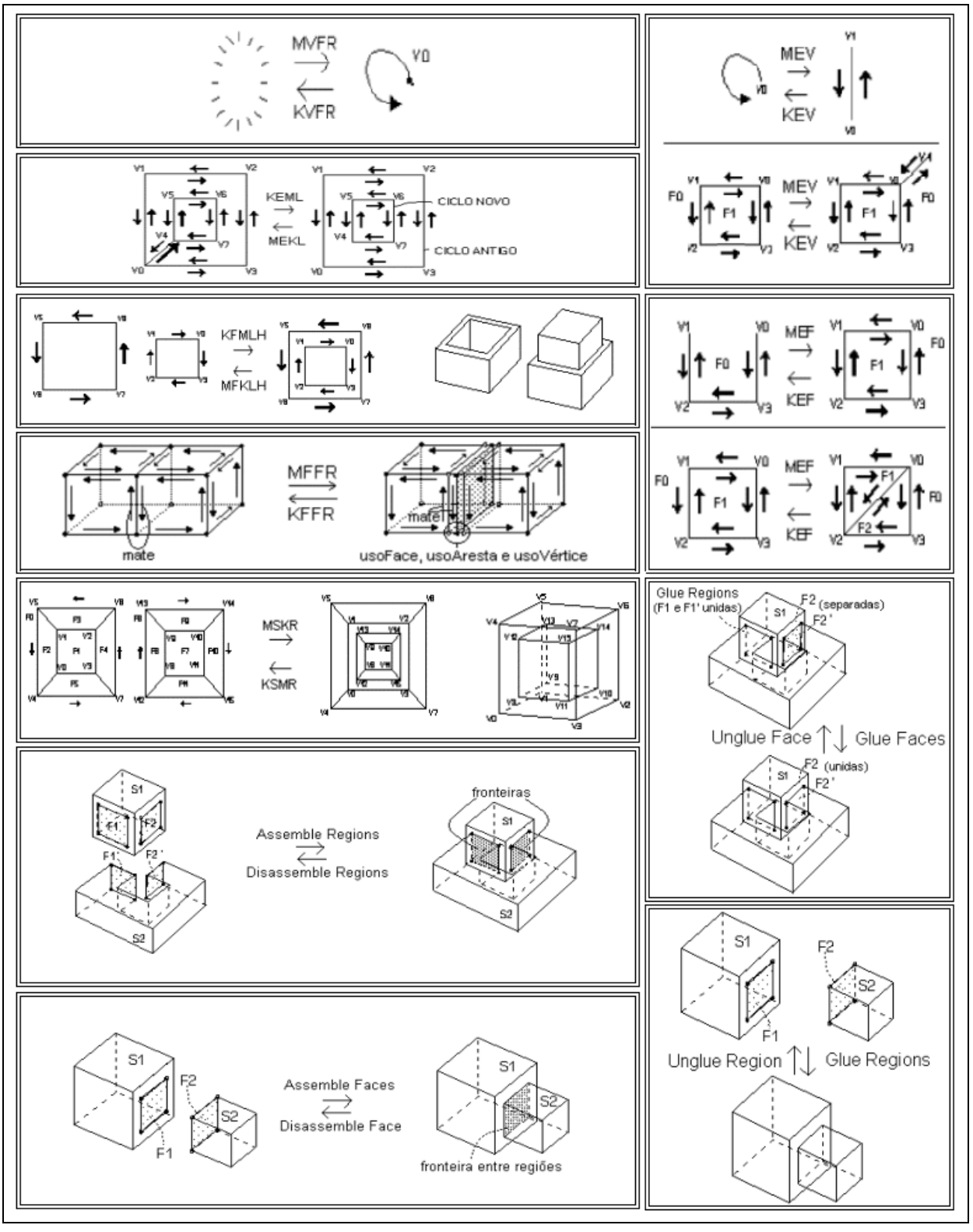


Figura II-6 – Os Operadores de Euler do GSM.

O esqueleto da estrutura de dados B-rep foi implementado utilizando-se uma lista, cuja classe foi denominada *List*, derivada da lista da biblioteca padrão do C++ STL (*Standard Template Library*), cujo nó básico é definido por uma classe puramente virtual denominada *BasicListNode* que une duas outras classes concretas e independentes, *ListHead* (elemento cabeça, primeiro elemento da lista) e *ListNode* (nó que possui um dos elementos primitivos), de maneira a serem manipuladas como sendo do mesmo tipo. A Figura II-7 mostra a representação UML da estrutura hierárquica da *BasicListNode*, *ListHead e ListNode.*

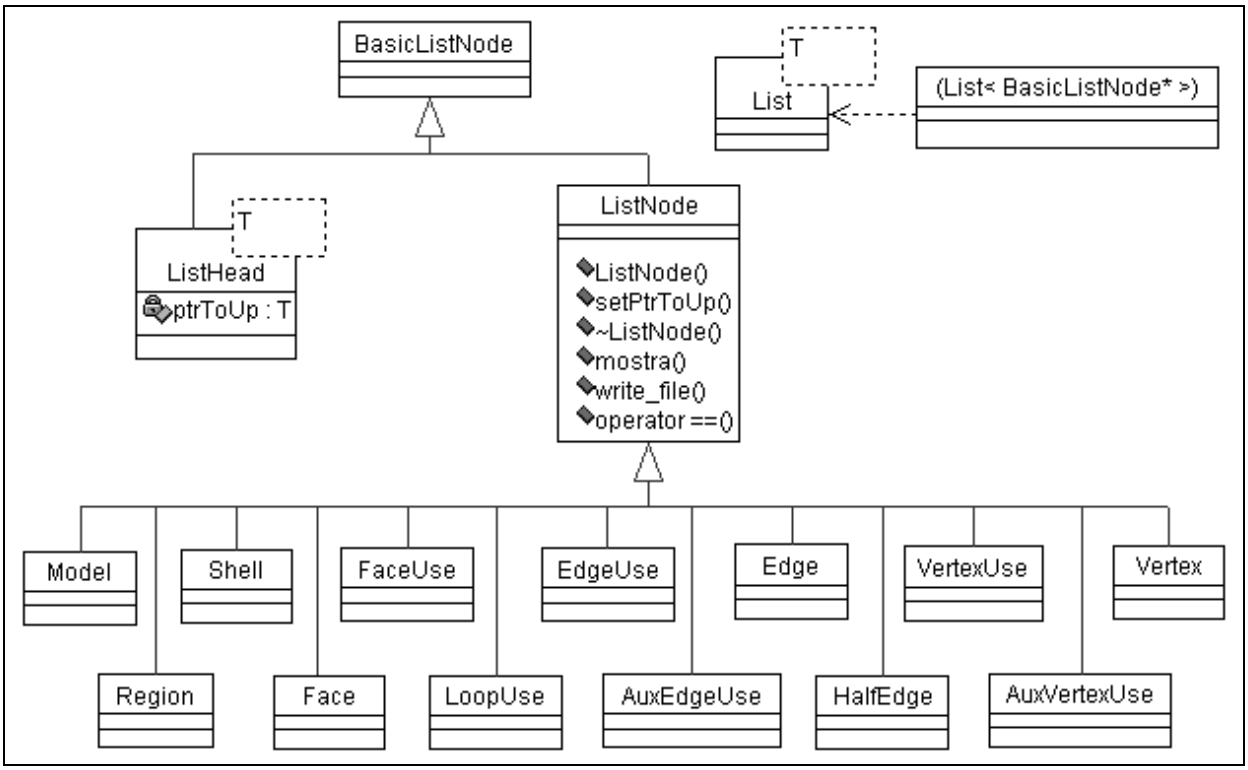


Figura II-7 – Representação em UML do nó básico da Lista.

O elemento cabeça difere dos demais nós pelo fato de possuir um “ponteiro de volta”, que aponta para um elemento superior, ou seja, aponta para o elemento possuidor da lista que o contém. Dessa forma, é possível varrer toda a estrutura de dados. É possível sair de um vértice do modelo armazenado na estrutura e chegar a uma semi-aresta (*HalfEdge*) ou até mesmo a uma casca (*Shell*) do modelo também armazenada nessa mesma estrutura que descreve o modelo.

Essa implementação adotada para a lista do B-rep permite varrer toda a estrutura de dados B-rep, além de eliminar o armazenamento de informações redundantes presente em outras estruturas, como mostrado na Figura II-8.

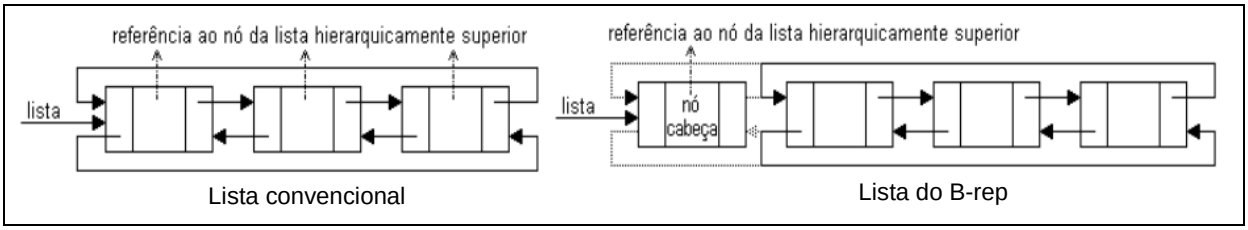


Figura II-8 – Lista usada pelo B-rep [MAG00].

II.5 – Subsistema de Malha

O Subsistema de Geração de Malha é o responsável por: cuidar da geração da malha superficial de elementos finitos, de forma a permitir que esta possa ser utilizada na geração da malha volumétrica; atribuir características físicas e condições de contorno aos elementos geométricos e de malha que compõem o modelo; controlar a inclusão e o posicionamento de nós auxiliares, de forma a garantir a compatibilidade e a qualidade da malha resultante, isto é, os nós devem estar nas arestas do modelo e a malha deve ter o mínimo possível de elementos mal formados. No GSM, a geração da malha superficial se faz juntamente com a criação das primitivas, de maneira que a malha seja aproveitada nas operações booleanas, agilizando o processo de obtenção da malha final, como ilustra a Figura II-9.

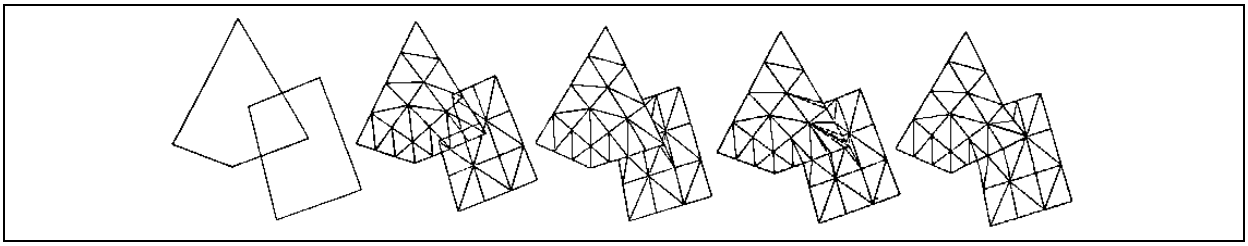
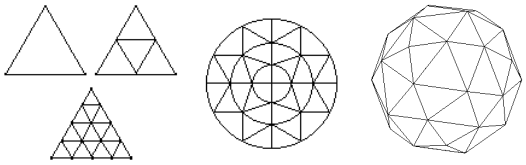


Figura II-9 – Técnica CSG com geração de malha sobre primitivas [KAM91]

O processo de geração de malha superficial utilizado no modelador segue uma estratégia semelhante à proposta por Kamel e Chen [KAM91], ou seja, sobre cada primitiva é gerada uma malha, utilizando qualquer técnica de geração de malha

disponível, como por exemplo, a propagação frontal da malha ou a triangulação de *Delaunay* [BER00], juntamente com a baseada em uma técnica para a discretização de esferas que parte da projeção sobre a superfície dos pontos obtidos da subdivisão de um octaedro nela inscrito, como está esquematizado na Figura II-10.



a) recursividade na discretização de uma esfera

Figura II-10 – Exemplos de Discretização para geração da malha [MAG00].

No GSM, a malha superficial é resultante da composição da malha gerada sobre a lateral do objeto com a malha produzida sobre o perfil gerador – as tampas do objeto. A malha gerada sobre a lateral do objeto é obtida a partir dos processos de varredura translacional ou rotacional, enquanto que a malha gerada sobre as faces planares – as tampas – definidas sobre o perfil gerador após a execução da varredura rotacional parcial ou translacional, é obtida utilizando o programa Triangle [SHE96]. Esse programa foi desenvolvido pela *Carnegie Mellon University* com o objetivo de gerar ou refinar malhas bidimensionais, por meio do método de *Delaunay,* em superfícies poligonais planares que podem ou não conter buracos.

Para a geração da malha volumétrica, foi acoplado ao modelador um gerador de malha tridimensional, criado e desenvolvido por Si Hang [HAN02]. O programa, denominado *Tetgen* recebe como dado de entrada o arquivo neutro gerado pelo GSM e, a partir dele, retorna a malha volumétrica.

II.6 – As Operações Booleanas e Operação de Montagem

Como parte do subsistema de modelagem, as operações booleanas e a operação de montagem constituem uma importante classe de manipulação de sólidos seguindo a abordagem CSG. Elas interagem com o subsistema de interface, modelagem e representação, pois são nós internos da árvore CSG. A partir de tais operações, modelos sólidos de complexidade mais elevada podem ser obtidos

aplicando-se uma simples interseção, adição, montagem ou subtração de modelos simples.

A seguir, uma abordagem detalhada será realizada a respeito das operações booleanas de união, interseção e subtração, além da operação de montagem e das etapas do processo para aplicação dessas sobre as primitivas, pois o tema desse trabalho se insere nesse contexto.

II.6.1 – OPERAÇÃO DE MONTAGEM

A operação de montagem foi criada para permitir modelar fronteira comum a duas regiões que possuem características diferentes. No caso de problemas de eletromagnetismo, essa característica pode ser o tipo de material ou outras características eletromagnéticas.

A montagem pode ser vista como uma colagem de sólidos por meio de suas fronteiras, não podendo, então, ser aplicada em objetos que estejam parcialmente dentro e/ou parcialmente fora um do outro. No GSM, a montagem também se aplica em objetos que estejam um contido totalmente dentro do outro. Nesse caso, a montagem consiste de uma seqüência de operações – (i) indica as regiões, uma interna a outra; (ii) realiza uma cópia da região interna; (iii) subtrai a região copiada da região externa, gerando um buraco; (iv) realiza sua colagem com a região interna que foi copiada. A Figura II-11 ilustra os passos desse processo da operação de montagem.

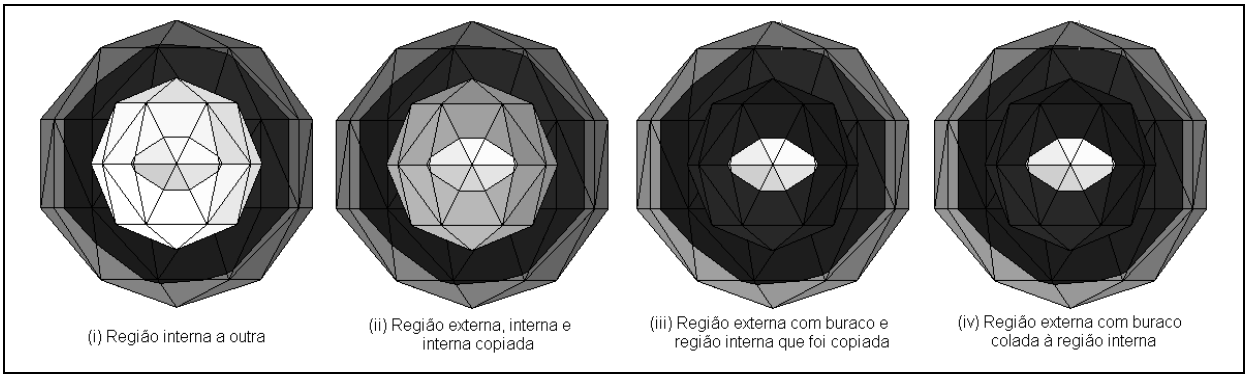


Figura II-11 – Passos da operação de montagem de duas regiões

A operação de montagem permite o encaixe e a manipulação conjunta dos objetos preservando suas características físicas. Além disso, na operação de montagem, cada fronteira existente entre os componentes do modelo é interpretada de maneira única e a compatibilidade da malha de elementos finitos é garantida.

II.6.2 – OPERAÇÕES BOOLEANAS

Para a modelagem de sólidos, um **conjunto** corresponde a uma coleção de pontos, sendo o **conjunto universo** E, o espaço Euclidiano formado por pontos com a dimensão desejada. Novos conjuntos podem ser gerados a partir da combinação de outros conjuntos.

Um conjunto no contexto da modelagem de sólidos obedece às mesmas propriedades e regras da teoria de conjunto convencional definida pela matemática. Entretanto, vale ressaltar que, nesse caso, nem todos os subconjuntos de E serão aceitáveis como forma de objetos sólidos, pois um objeto sólido deve ser homogêneo, "volumoso", ocupar uma porção finita do espaço e não possuir faces, arestas e vértices soltos e pendentes. A Figura II-12 faz a representação das operações entre conjuntos enquanto que, no Quadro II-3, são apresentadas as propriedades de conjunto existentes.

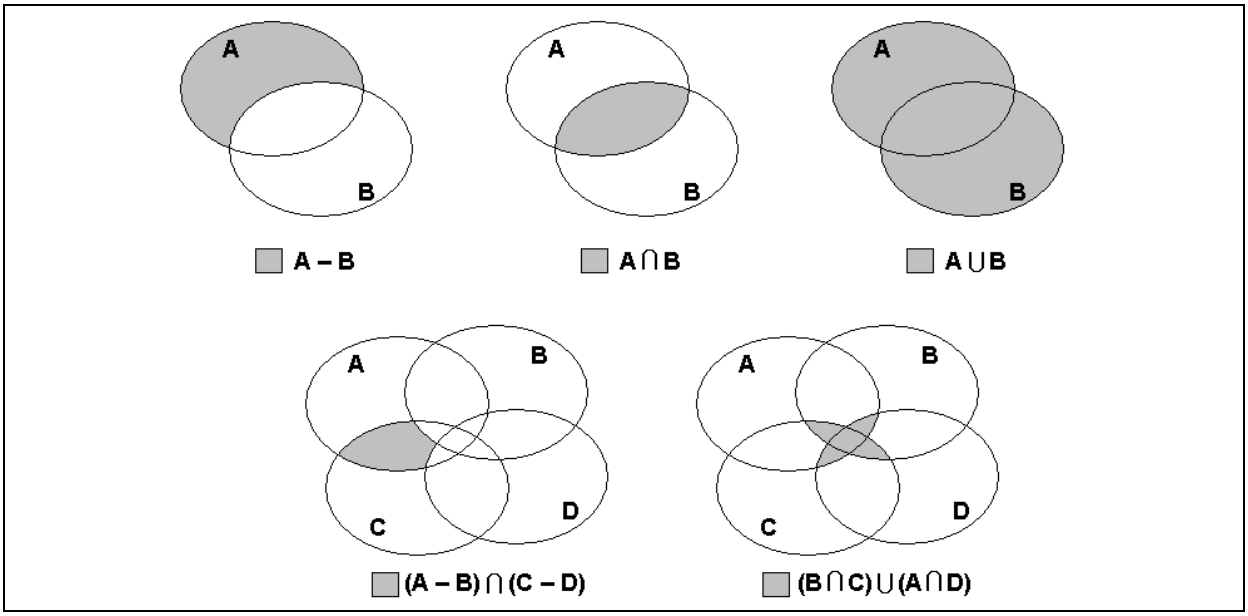


Figura II-12 – Diagramas de Venn

Propriedades da União	
1. $A \cup B$ é um conjunto	propriedade do fechamento
2. $A \cup B = B \cup A$	propriedade comutativa
3. $(A \cup B) \cup C = A \cup (B \cup C)$	propriedade associativa
4. $A \cup \varnothing = A$	propriedade da identidade
5. $A \cup A = A$	propriedade da idempotência
6. $A \cup cA = E$	propriedade do complemento

Propriedades da Interseção	
1. A ∩ B é um conjunto	propriedade do fechamento
2. A ∩ B = B ∩ A	propriedade comutativa
3. (A ∩ B) ∩ C = A ∩ (B ∩ C)	propriedade associativa
4. A ∩ E = A	propriedade da identidade
5. A ∩ A = A	propriedade da idempotência
6. A ∩ cA = ∅	propriedade do complemento

Propriedades do Complemento	
1. cE = ∅	O complemento do conjunto universo é o conjunto vazio
2. c∅ = E	O complemento do conjunto vazio é o conjunto universo
3. c (cA) = A	O complemento do complemento de um conjunto A é A
4. c (A ∪ B) = cA ∩ cB	Lei de De Morgan
5. c (A ∩ B) = cA ∪ cB	Lei de De Morgan

Propriedades Distributivas	
1. A ∪ (B ∩ C) = (A ∪ B) ∩ (A∪ C)	A união é distributiva em relação à interseção
2. A ∩ (B ∪ C) = (A ∩ B) ∪ (A ∩ C)	A interseção é distributiva em relação à união

Quadro II-3 – Propriedades da teoria de conjuntos [MOR85]

No GSM, o resultado da combinação de outros dois sólidos regulares (homogêneos em dimensão) é também um sólido válido. Isso é possível devido ao fato das operações booleanas serem regularizadas. Tais operações garantem que a dimensionalidade dos objetos iniciais seja preservada, o que significa manter a dimensão de todos os objetos participantes da operação booleana. Como exemplo, na Figura II-13, são apresentados os passos de uma operação booleana de interseção regularizada.

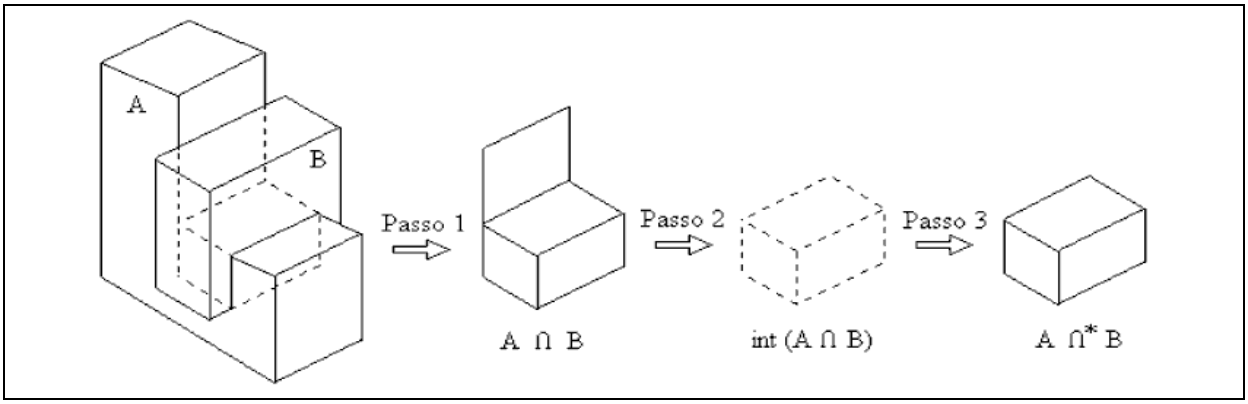


Figura II-13 – Passos da execução da interseção regularizada [HOF89]

Várias operações booleanas podem ser aplicadas a um conjunto de objetos, sendo que a seqüência de sua aplicação interfere no resultado obtido, pois esses são diferentes. A Figura II-14 ilustra os resultados que podem ser obtidos de operações booleanas, sendo mostrado o resultado da união regularizada na Figura II-14 a), o resultado da diferença regularizada na Figura II-14 b), o resultado da interseção regularizada na Figura II-14 c) e as interseções não regularizadas nas demais figuras.

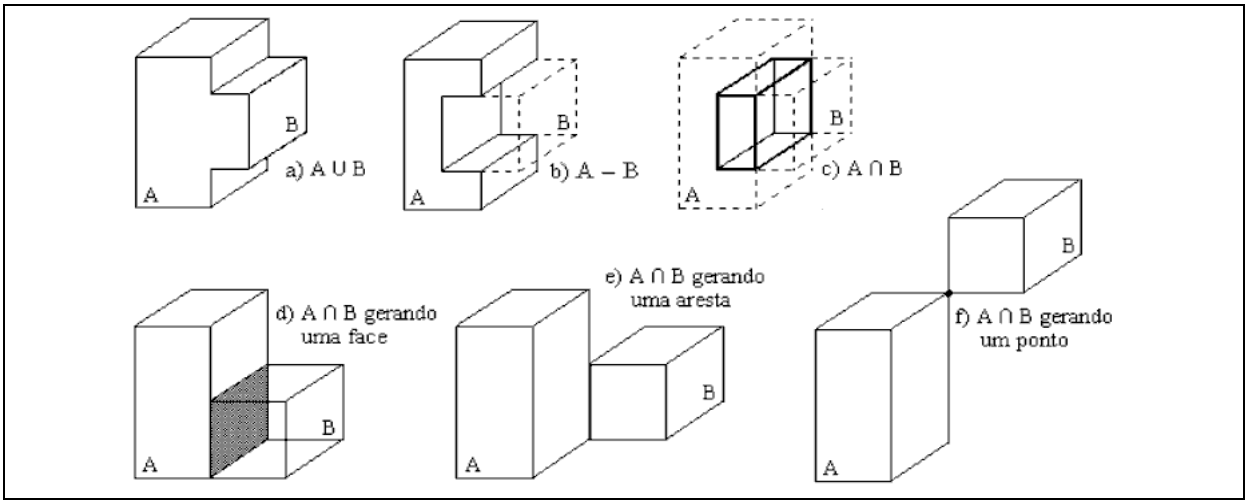


Figura II-14 – Resultado de operações booleanas em sólidos [MOR85]

Tanto para a união, interseção e/ou diferença o resultado de uma operação regularizada deve ser um objeto sólido tridimensional fechado, homogêneo e com volume. Uma aresta, um vértice ou uma face isolada são exemplos de resultados

inválidos de operações booleanas em sólidos, e são tratados pelo GSM por meio da avaliação da fronteira dos objetos.

II.6.3 – AVALIAÇÃO DA FRONTEIRA

A avaliação da fronteira é executada toda vez que uma operação booleana ou uma operação de montagem é aplicada. Seu processo é realizado em várias etapas, como representado no fluxograma da Figura II-15, Ela é baseada no princípio de "dividir para conquistar", discutido em [REQ85] e [TIL80], que leva a procedimentos de classificação recursivos. O resultado de cada etapa do processo consiste em uma malha poligonal consistente e não redundante, que é utilizada na etapa posterior do processo, até obter a fronteira completa do modelo [NUN02].

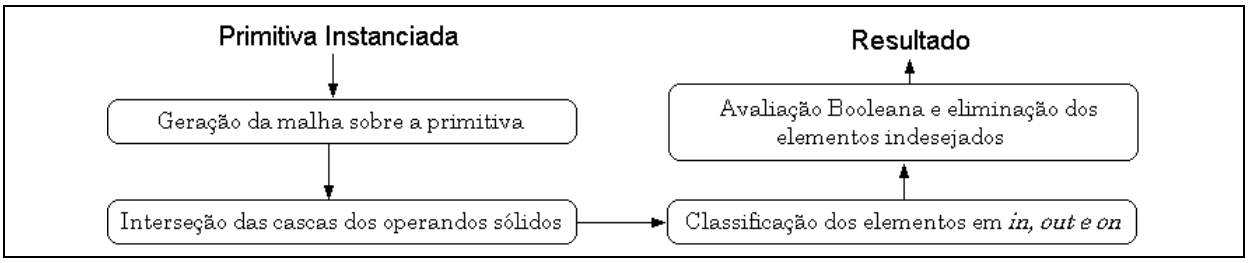


Figura II-15 – Fluxograma das etapas da avaliação da fronteira

A primeira etapa da avaliação da fronteira é a geração da malha superficial sobre a primitiva instanciada extraída da árvore do CSG. Essa malha tem como característica ser poligonal, homogênea e sem interseções entre seus polígonos. Esse tipo de representação poligonal é uma aproximação. Para o GSM, essa aproximação é feita por uma malha triangulada, ou seja, as superfícies curvas são aproximadas por facetas triangulares.

O GSM utiliza o conceito de tolerância absoluta para controlar a natureza e magnitude dos erros que são introduzidos nessa aproximação, ou seja, ele limita a diferença entre qualquer ponto da aproximação e o ponto correspondente no sólido original, sendo expressa como distância absoluta na unidade utilizada. Essa tolerância pode ser fornecida pelo usuário, caso contrário um valor mínimo padrão é atribuído.

Na etapa de interseção entre as cascas, a interferência entre as malhas dos operandos é computada. O processo é dividido em quatro partes distintas onde são utilizadas algumas estratégias para identificação das interseções e corte das duas cascas A e B. As partes que compõem essa etapa estão apresentadas no fluxograma da Figura II-16.

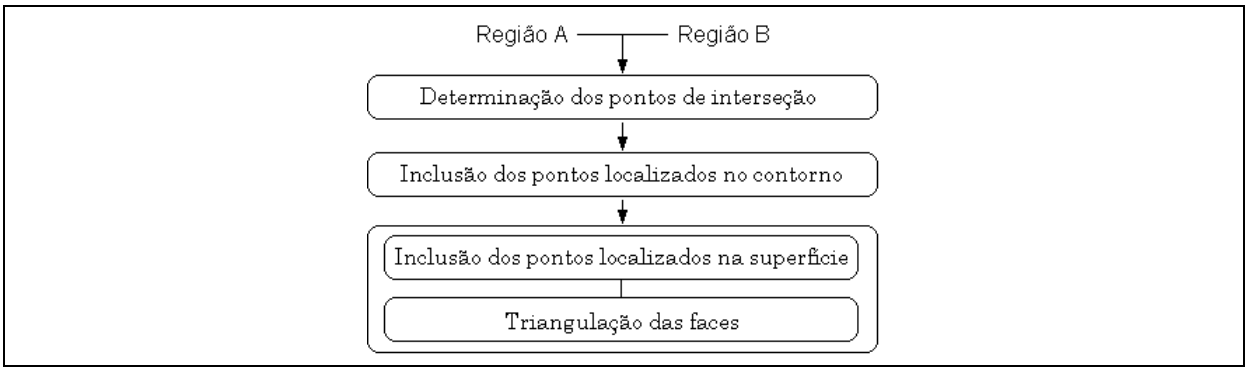


Figura II-16 – Fluxograma do processo de determinação da interseção

A estratégia adotada consiste em varrer todos os elementos de A e compará-los com os elementos de B, armazenando, em um arquivo, todos os pontos comuns a ambos. O Quadro II-4 esquematiza o processo de determinação dos pontos de interseção das cascas dos operandos.

```
para <cada casca da região A> faça
 | para <cada face na casca de A> faça
 |   para <cada casca da região B> faça
 |     para <cada face na casca de B> faça
 |       se <existe interseção da face em A com a face em B> então
 |        | <calcular os pontos de interseção>
 |        | para <cada ponto obtido> faça
 |        |   se <o ponto obtido não é um vértice> então
 |        |    | se <o ponto obtido está em uma aresta da face A> então
 |        |    |   <armazena no arquivo ptEdge1.tmp>
 |        |    | senão <armazena no arquivo ptFace1.tmp>
 |        |    | fim se
 |        |    | se <o ponto obtido está em uma aresta da face B> então
 |        |    |   <armazena no arquivo ptEdge2.tmp>
 |        |    | senão <armazena no arquivo ptFace2.tmp>
 |        |    | fim se
 |        |   fim se
 |       fim se
fim para
```

Quadro II-4 – Processo para determinação dos pontos de interseção.

Os pontos de interseção obtidos são classificados de acordo com o seu posicionamento em relação às faces envolvidas, seja no contorno ou na superfície, como mostrado na Figura II-17.

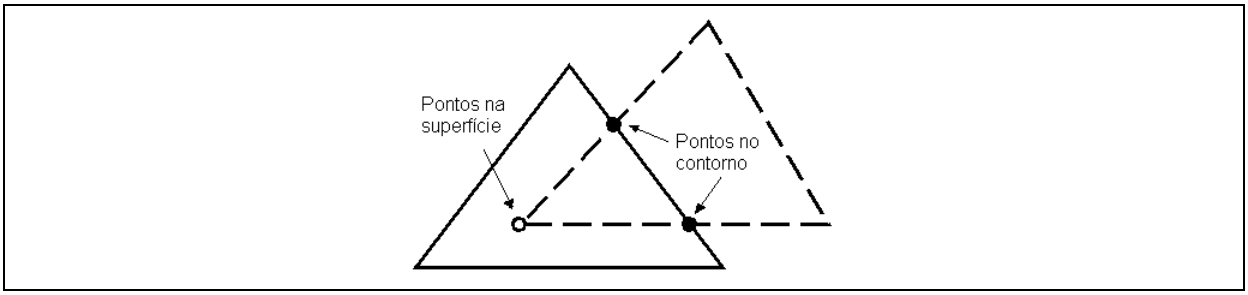


Figura II-17 – Classificação dos pontos como sendo de contorno ou superfície

Em seguida, inicia-se a fase de inserção dos pontos obtidos na estrutura de dados B-rep por meio dos operadores de Euler. Primeiramente, são adicionados os pontos localizados no contorno das faces para, posteriormente, serem inseridos os pontos localizados na superfície interna das faces que compõe as regiões participantes. Finalizando o processo, é feita a triangulação das faces poligonais.

Os pontos localizados no contorno são lidos do arquivo e inseridos na estrutura por meio do operador de Euler MEV. Nessa fase é feito um controle dos pontos para não haver duplicação de informação, ou seja, o mesmo ponto ser inserido duas vezes no contorno.

Tanto a inserção dos pontos localizados na superfície interna da face quanto a triangulação das mesmas são realizadas conjuntamente. Os pontos a serem inseridos em uma face são lidos do arquivo e passados para o programa Triangle que determina qual a melhor maneira de gerar a malha da superfície e, posteriormente, são aplicados os operadores MEV e MEF para a construção da malha. Nesse processo, é feito o controle para garantir a compatibilidade entre as malhas das cascas. Entretanto, não é garantido que a malha resultante apresente uma boa qualidade.

O passo seguinte à interseção entre as cascas é a Classificação dos elementos de uma casca com relação à outra. Nessa etapa, vértices, arestas, ciclos e faces serão classificados, obedecendo a essa ordem, como estando dentro (*in*), fora (*out*)

ou sobre a superfície (*on*) da outra casca. O processo está esquematizado no Quadro II-5.

```
para <cada casca da região A> faça
| para <cada face da casca de A> faça
| | se <a face não está classificada> então
| | | para <cada aresta da face de 1> faça
| | | | se <a aresta não está classificada> então
| | | | | para <cada vértice da aresta de 1> faça
| | | | | | se <o vértice não está classificado> então
| | | | | | | para <cada face da região B> faça
| | | | | | | | se <vértice sobre alguma face de 1> então
| | | | | | | | | <classifica o vértice como ON>
| | | | | | | | senão
| | | | | | | | | <classifica o vértice utilizando a técnica Raytracing>
| | | | | | | | fim se
| | | | | | fim se
| | | | | <classifica a aresta a partir da classificação dos vértices>
| | | | fim se
| | | <classifica a face a partir da classificação das arestas>
| | fim se
| fim para
Fim para
```

Quadro II-5 – Processo de classificação dos elementos da casca

A classificação do vértice de uma casca A consiste em verificar se ele está sobre alguma face da casca B. Caso não esteja, aplica-se a técnica de *raytracing* para determinar a posição do vértice, como mostrado na Figura II-18. Tal técnica consiste em determinar a posição do vértice a partir do número de interseções existente entre as faces da outra casca e um raio traçado a partir do vértice até o infinito. Se o número de interseções é ímpar o vértice está dentro, caso seja par o vértice está fora da casca.

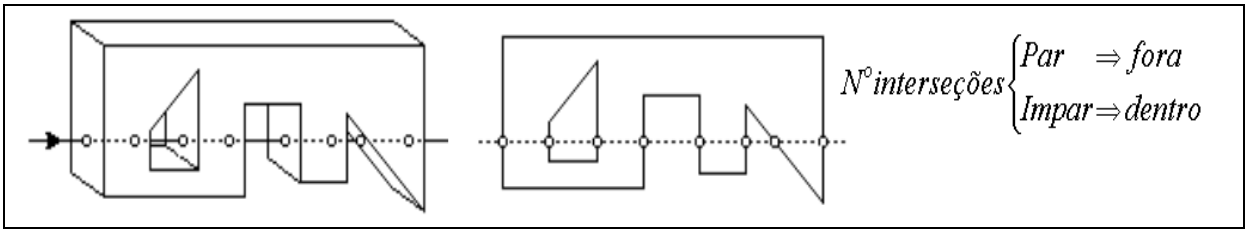


Figura II-18 – Técnica de *raytracing* para classificação do vértice [MAG00]

A classificação da aresta é feita a partir da combinação da classificação dos dois vértices que a limitam. Se os vértices não estiverem sobre a casca B, a aresta é classificada de acordo com os vértices. Se ambos estiverem sobre a casca B, a classificação da aresta será obtida pela análise do posicionamento de seu ponto médio por meio da técnica de *raytracing*. As combinações de classificação de vértices que geram a classificação da aresta, estão resumidas no Quadro II-6.

vértices	OUT/OUT	OUT/ON	ON/ON	ON/IN	IN/IN	IN/OUT
Aresta	OUT	OUT	RAYTRACING	IN	IN	ERRO

Quadro II-6 – Classificação da aresta a partir de seus vértices.

Similarmente, a classificação de uma face é feita a partir da combinação das classificações de suas arestas e orientação. Caso todas as arestas da face sejam classificadas como estando "*ON*" a casca de B, a face é classificada como sendo "*ON*". Entretanto, deve existir uma outra face que possua o mesmo conjunto de arestas na topologia da casca B. Essas faces serão *shared* se possuírem mesma orientação e *anti-shared* se possuírem orientação diferentes. As combinações possíveis para se classificar uma face triangular são apresentadas no Quadro II-7.

Arestas	OUT/OUT/OUT	OUT/OUT/ON	OUT/ON/ON	ON/ON/ON	ON/ON/IN
Resultado	OUT	OUT	OUT	ON	IN
Arestas	ON/IN/IN	IN/IN/IN	IN/IN/OUT	IN/ OUT/OUT	OUT/IN/ON
Resultado	IN	IN	ERRO	ERRO	ERRO

Quadro II-7 – Classificação das faces a partir das arestas

Depois de realizada a classificação dos elementos topológicos dos objetos, dá-se início ao processo de avaliação booleana para decidir quais elementos devem ser eliminados e quais devem permanecer. O objeto ao qual o elemento faz parte recebe a classificação *on*. Sendo assim, os elementos de A são classificados como *onAinB*, *onAonBshared*, *onAonBanti-shared* ou *onAoutB*, enquanto que os elementos de B são classificados como *inAonB*, *onAonBshared*, *onAonBanti-shared* ou *outAonB*. A partir dessas oito classificações, é possível definir a ação apropriada para a eliminação dos elementos de acordo com a operação aplicada. O Quadro II-8 relaciona as ações com suas operações.

A	B	$A+B$	$A \cap B$	$A \cup B$	$A-B$	$B-A$
ON	IN	MANTER	MANTER	DESCARTAR	DESCARTAR	MANTER/INVERTER
ON	ON SHARED	MANTER	MANTER	MANTER	DESCARTAR	DESCARTAR
ON	ON ANTI-SHARED	MANTER	DESCARTAR	DESCARTAR	MANTER	MANTER
ON	OUT	MANTER	DESCARTAR	MANTER	MANTER	DESCARTAR
IN	ON	MANTER	MANTER	DESCARTAR	MANTER/INVERTER	DESCARTAR
ON SHARED	ON	DESCARTAR	DESCARTAR	DESCARTAR	DESCARTAR	DESCARTAR
ON ANTI-SHARED	ON	DESCARTAR	DESCARTAR	DESCARTAR	DESCARTAR	DESCARTAR
OUT	ON	MANTER	DESCARTAR	MANTER	DESCARTAR	MANTER

Quadro II-8 – Tabela de decisão booleana [MAG00]

A Figura II-19 a) representa todas as possibilidades de classificação dos elementos apresentadas no Quadro II-8, enquanto que a Figura II-19 b) até f) ilustra as tomadas de decisão representando os resultados obtidos para cada operação.

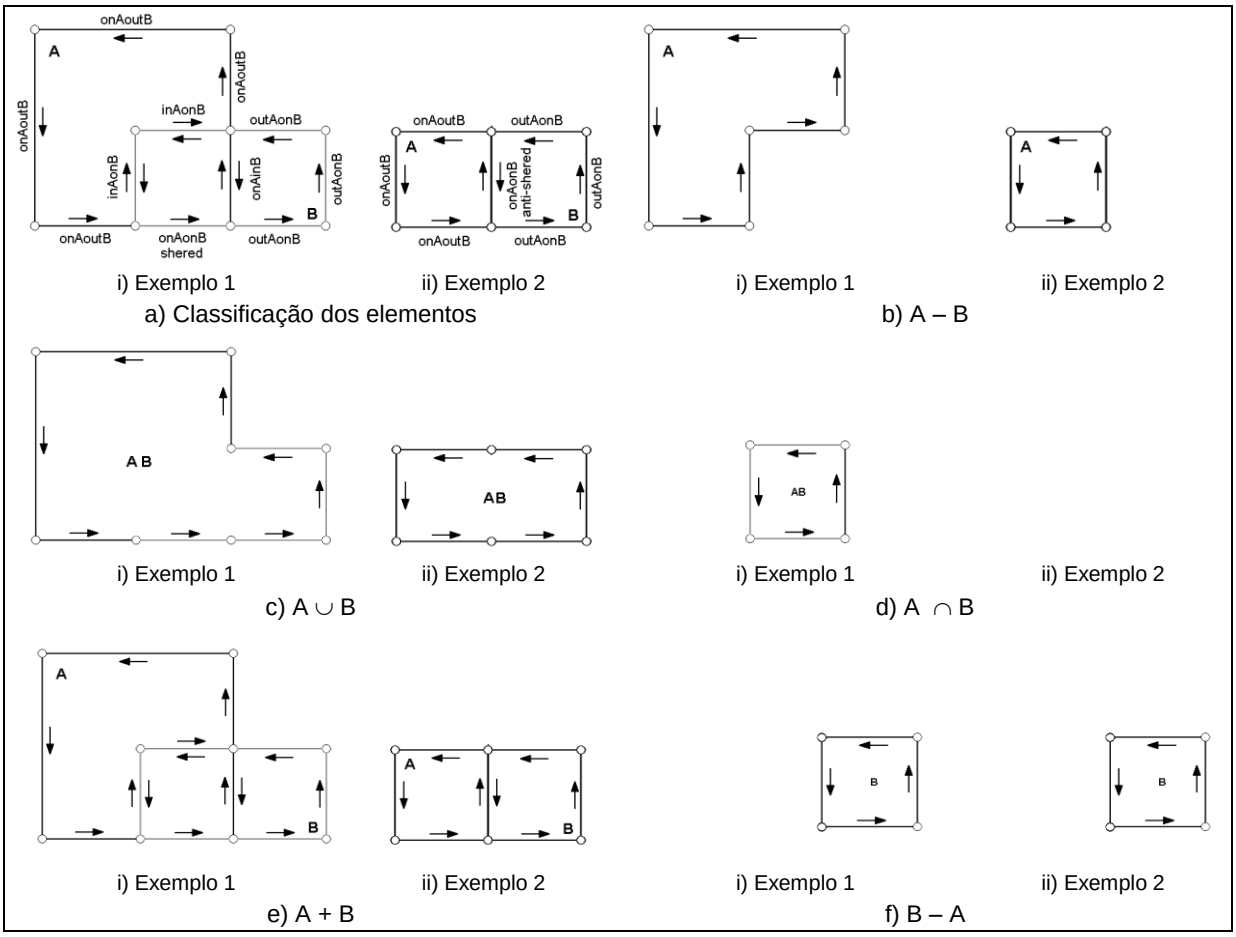


Figura II-19 – Classificação e avaliação booleanas [MAG00]

Capítulo III - Modificações Realizadas no GSM

Antes de dar início à implementação dos métodos para melhorar a qualidade da malha resultante das operações booleanas, foi necessário realizar algumas alterações no Modelador não só para corrigir restrições existentes como também melhorar seu desempenho, principalmente na estrutura de dados B-rep presente no subsistema de representação.

Como dito anteriormente, a estrutura de dados B-rep foi implementada utilizando listas circulares duplamente encadeadas derivadas da lista da STL onde o primeiro elemento é o "elemento cabeça". Optou-se por utilizar a lista da STL como base para facilitar a implementação da estrutura de dados e para garantir uma portabilidade do sistema, uma vez que a STL é a biblioteca padrão do C++. Entretanto, devido a algumas limitações observadas e problemas encontrados ao se utilizar a lista da STL, foi necessário trocar a implementação da lista do GSM descrita no item II.4.1.

Ao descrever as operações booleanas e a operação de montagem foi mostrado que o processo é feito em várias etapas. Uma delas é a determinação dos pontos de interseção comuns às regiões envolvidas na operação. Os pontos de interseção são calculados e armazenados em arquivos para depois serem inseridos na estrutura por meio dos operadores de Euler. Entretanto, em alguns casos, foram detectados erros de inserção dos pontos na estrutura. Pontos de interseção, que deveriam ser inseridos em uma determinada aresta B, são inseridos na aresta A quando o algoritmo que verifica se o ponto deve ser inserido nessa aresta, retorna "SIM", como indicado na Figura III-1.

Essa verificação errônea ocorre devido a problemas de precisão numérica. Se o ponto, que deve ser inserido em B, é muito próximo ao vértice comum às arestas A e B, devido ao truncamento nas coordenadas do ponto (quando esse é armazenado no arquivo) e a arredondamentos computacionais durante os cálculos, a verificação da posição do ponto em relação às arestas, sempre resulta em "pertence às duas

arestas". Assim, foi preciso fazer a substituição dos arquivos por uma estrutura interna para armazenamento na memória.

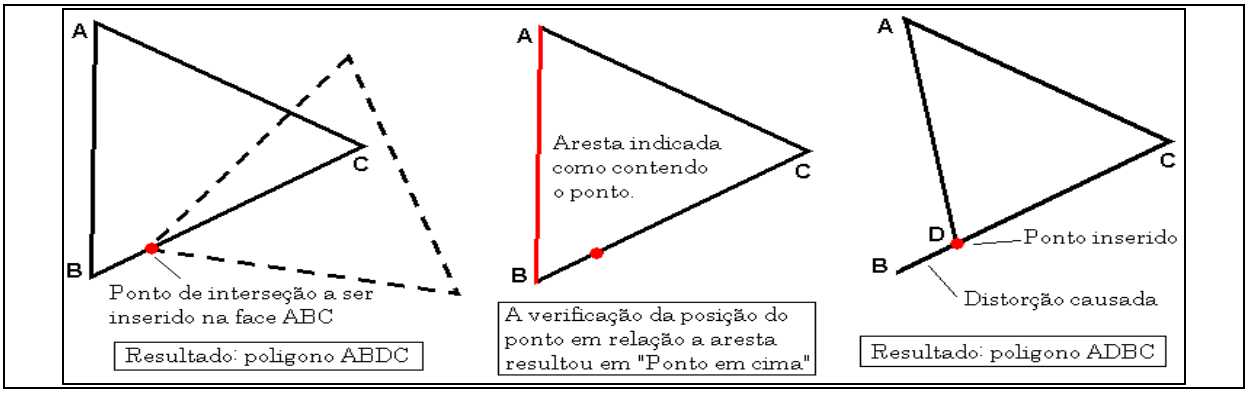


Figura III-1 - Exemplo de Inserção errada dos pontos.

Nas seções seguintes, é feito o detalhamento dos problemas apresentados, bem como as modificações propostas para melhorar o desempenho do modelador.

III.1 – Troca da Lista usada no B-rep

Nessa seção, serão descritas a lista da STL que foi base para o projeto da nova lista, a nova lista implementada para a estrutura do B-rep, além das alterações realizadas no modelador de forma a adequá-lo à nova lista.

III.1.1 – A LISTA DA BIBLIOTECA PADRÃO DO C++

A biblioteca STL é composta de várias classes *template*, funções e algoritmos confiáveis e bastante genéricos. Nela, é definido um conjunto de variados tipos de *containers* – elementos que contêm outros elementos – como vetores, pilhas, listas, árvores, dicionários e outros.

As funções e algoritmos existentes na STL, por serem genéricos, podem ser aplicados a qualquer tipo de *container* da biblioteca ou qualquer outra estrutura de dados que acesse seus elementos através dos iteradores. O uso dos iteradores permite a intercomunicação, de forma única, entre as funções e as diversas

containers. O iterador ou *iterator* foi uma maneira encontrada pelos projetistas da STL para permitir que um algoritmo ou uma função atue sobre os elementos de qualquer uma das diversas estruturas *container* de forma única e independente das características da estrutura de dados.

O iterador substitui o ponteiro convencional para o elemento existente na maioria das estruturas de dados desse tipo, podendo ser considerado como uma generalização dos ponteiros, pois são objetos que apontam para outro objeto. Na STL, são definidos por meio da classe *iterator* que possui como atributo de classe apenas um ponteiro para um nó da lista. Os ponteiros para o elemento anterior e o próximo da lista estão presentes nos nós. No *Iterator,* encontram-se apenas os métodos e operadores que permitirão varrer a lista e acessar seus demais nós. O nó de lista consiste em uma *struct* que a STL denomina de *__Node e* que é composta de um ponteiro para o próximo nó, um ponteiro para o nó anterior e um campo de dado onde o elemento é armazenado. A Figura III-2 mostra uma simplificação do *iterator* e o nó da lista da STL.

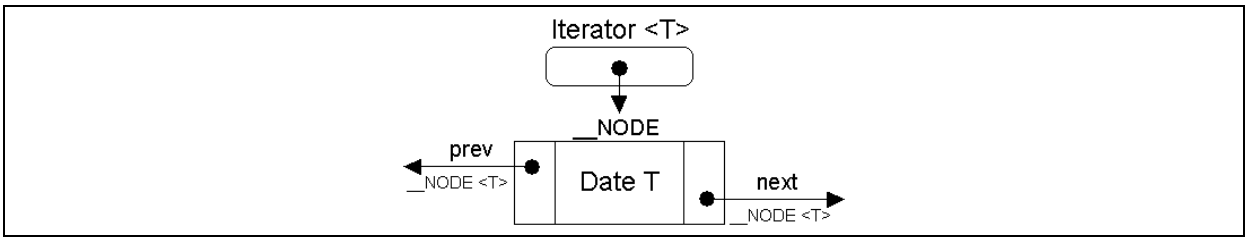


Figura III-2 – Esboço do Iterator presente na STL.

A lista da STL é uma estrutura linear duplamente encadeada, que armazena seus elementos em nós e é definida por meio da classe *list* implementada na STL. Ela pode ser vista contendo como atributos de classe: um contador denominado *__length* que indica quantos elementos a lista armazena no momento e dois ponteiros, um para o primeiro *iterator* (*begin*) e outro para o último *iterator* da lista (*end*). Essa descrição é uma simplificação da implementação real existente na STL. A Figura III-3 apresenta o diagrama de classe da lista da STL, com seus relacionamentos. O diagrama foi obtido aplicando-se, sobre os arquivos da STL, a engenharia reversa presente no Rational Rose.

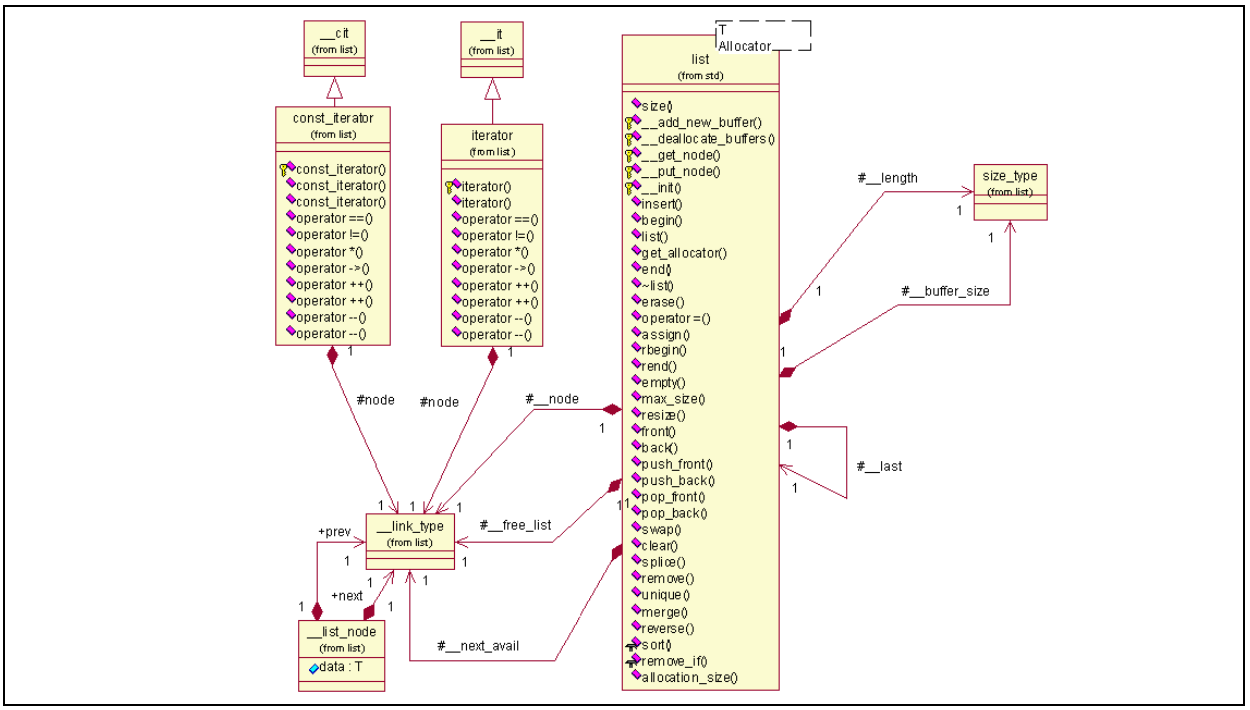


Figura III-3 – Diagrama de classe da STL para a lista

Baseado no modelo da lista apresentado anteriormente, a Figura III-4 esboça uma simplificação da estrutura da lista presente na biblioteca padrão do C++. Observe que a lista possui um *iterator* que aponta para NULL. Isso é feito para indicar o final da lista.

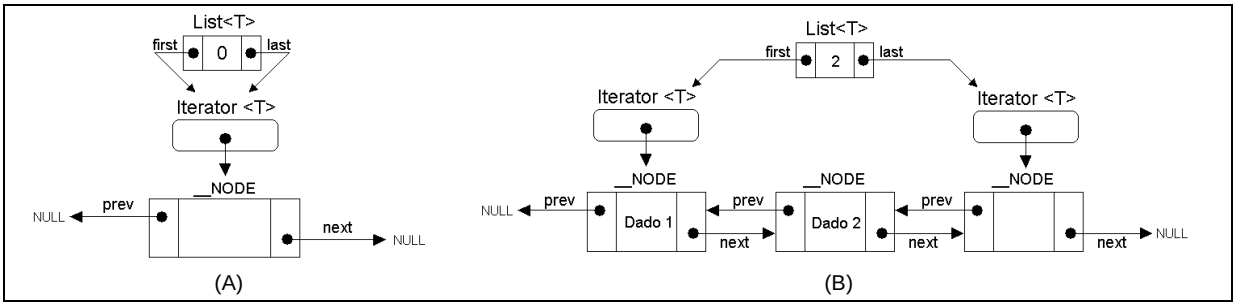


Figura III-4 – Esboço da lista da STL (A) vazia e (B) com 2 elementos.

Como pode ser observado, o *Iterator* possui um ponteiro para a *struct __Node* para armazenar o elemento inserido na lista em uma variável simples. A lista da STL não faz nenhum tipo de verificação quanto à validade da informação armazenada nela e nem quanto à validade das informações armazenadas na *struct __Node*. A lista permite que um *iterator* tenha um nó com informações inválidas – nó central da

lista sem elemento armazenado, ou os ponteiros (*next* e *prev*) apontam para endereços inválidos. Esse foi o principal motivo da troca de lista, pois não é possível verificar se o nó de um *iterator* possui, ou não, informação correta, acarretando alguns outros problemas.

A lista da STL possui uma restrição de funcionamento que não tem como ser detectado devido ao problema descrito acima. Explicando melhor: quando se manipulam elementos várias vezes na lista, acontece que, ao tentar remover um de seus elementos (fazendo a chamada da função e passando corretamente os parâmetros) a lista retorna da função que o elemento foi removido, mas na realidade ela não consegue removê-lo, permanecendo os iteradores e o nó. Imagine que o elemento que deve ser removido da lista seja uma face do modelo que está sendo destruída. Se, ao tentar remover a face da estrutura de dados B-rep, acontecer esse problema, ou seja, a lista não remover o nó que a armazenava, mas indicar que ela foi removida, então, esta será imediatamente destruída. Sendo assim, posteriormente, ao tentar varrer a lista de faces para buscar as informações armazenadas, é feito o acesso ao nó "supostamente removido", que está vazio, causando estouro de pilha da memória. Esse estouro ocorre porque, como o nó não foi apagado, ele existe sem elemento a ser acessado. Não tendo como verificar se o conteúdo do nó da lista é ou não válido, não tem jeito de evitar esse acesso inválido ao elemento.

Outro motivo que incentivou a troca da lista foi o acesso aos seus elementos. Quem define esses acessos são os iteradores e não tem como restringir o acesso a apenas parte dos elementos da lista. Como a estrutura B-rep utiliza uma lista cujo primeiro elemento é o "elemento cabeça" que não é "protegido", qualquer *iterator* consegue acessá-lo, movê-lo ou até mesmo removê-lo da lista, ficando a cargo do programador fazer o controle do que deve ou não ser movido. Se o programador não fizer tal controle, isso pode levar a duas situações problemáticas:

1. O elemento cabeça ser removido da lista e apagado: não é mais possível acessar o elemento que possui a lista. Com isso, a estrutura B-rep perde seu poder de acesso rápido a qualquer elemento do modelo.

2. O elemento cabeça ser movido de lista: haverá lista sem cabeça – o elemento foi removido da lista – assim como pode haver lista com duas cabeças – elemento foi movido para outra lista. Sendo assim, não se tem como saber qual cabeça é a correta.

III.1.2 – PROJETO DA LISTB E SEU ITERATORL

A nova lista desenvolvida para ser utilizada na estrutura B-rep foi denominada de *ListB e* foi projetada com o objetivo primordial de "proteger" o elemento "cabeça" que compõe a lista. No projeto, é possível notar que apenas os métodos da lista acessem, criem, movam ou destruam sua "cabeça", garantindo assim seu manuseio correto. Além disso, foram englobados aspectos tanto da lista proposta inicialmente para a estrutura B-rep quanto da lista da STL, de maneira a preservar as características propostas no trabalho de doutorado da aluna Ana Liddy [MAG00], para a estrutura de dados B-rep. Somado a isso, algumas adaptações na estrutura B-rep foram realizadas, de modo que as falhas presentes na antiga estrutura fossem eliminadas. Como uma nova lista foi projetada, mudanças na sua estruturação foram feitas de forma a simplificar a antiga lista e garantir o correto funcionamento da estrutura de dados. As principais alterações realizadas estão descritas a seguir.

Tentando simplificar a estrutura da lista do B-rep já implementada e aproveitar ao máximo o código já existente, foi dada preferência por seguir a metodologia utilizada pela lista da STL no projeto da *ListB*, ou seja, ter um elemento com as mesmas características do *Iterator* e que, além disso, permita armazenar os dados. Para a nova lista, tal elemento foi denominado de *IteratorL* para não haver confusão com o *Iterator* da STL, pois a classe *IteratorL* é abstrata e é o topo de uma hierarquia de "iteradores", ao passo que a classe *Iterator* é uma classe concreta.

No projeto, foi feita uma junção da classe *iterator,* da *struct __Node* com a da estrutura hierárquica *BasicListNode* utilizada no B-rep mostrada na Figura II-7 da seção II.4.1. Sendo assim, para o iterador da nova lista, optou-se por definir uma hierarquia de classe onde o *IteratorL* é a classe base da qual se derivam duas outras classes: *Head,* classe que define a "cabeça" da lista, por possuir apenas o ponteiro

de volta, e *Body,* classe que define o "corpo" da lista – possui os dados – e que possui os ponteiros de "ida e volta" que permitem percorrê-la. Dessa maneira, a cabeça deixa de ser um elemento adicionado à lista e passa a ser parte da lista que fica responsável pelo controle total de sua "cabeça". A Figura III-5 mostra a lista e a estrutura projetada para o *IteratorL* de *ListB* e o resultado da junção realizada.

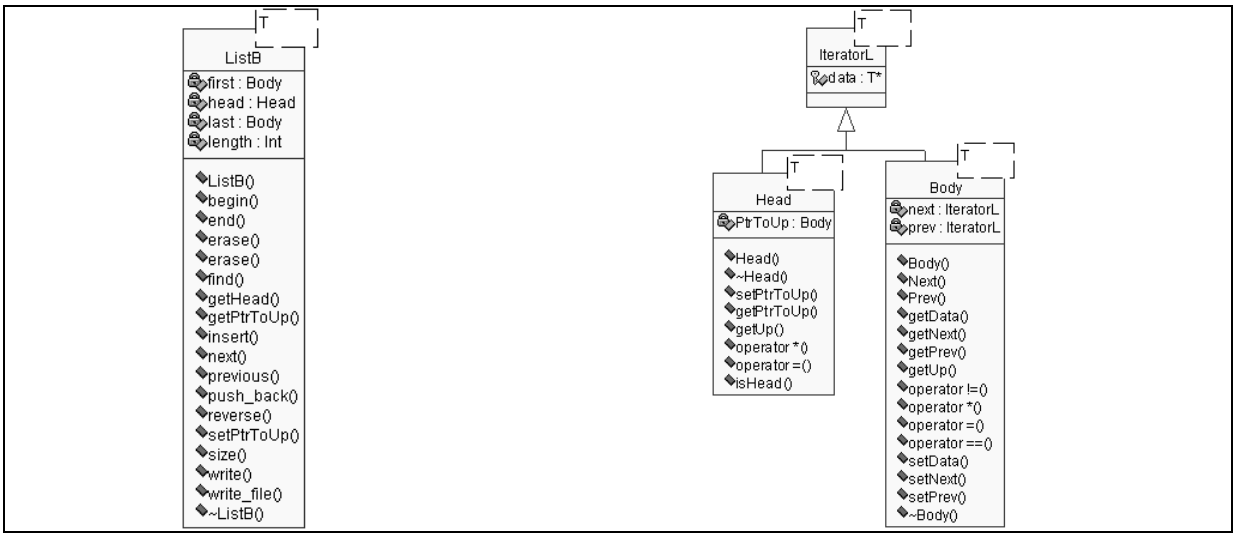


Figura III-5 – Estrutura da nova lista *ListB* e seu *IteratorL*.

Essa hierarquia determinada para o *IteratorL* foi a maneira encontrada para garantir a integridade da "cabeça" da lista, além de simplificar a estrutura da lista STL e da hierarquia *BasicListNode* que foi substituída por uma estrutura hierárquica simples, como mostrado na Figura III-6.

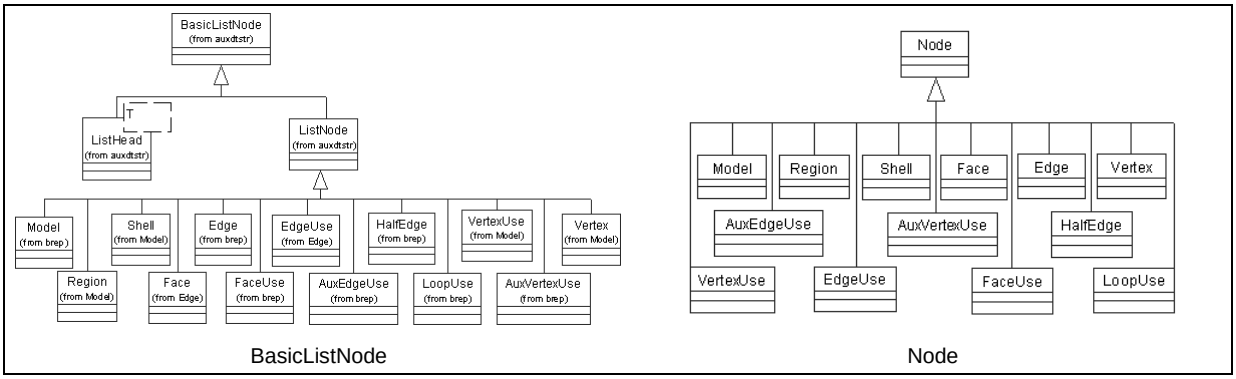


Figura III-6 – Comparação entre os componentes da lista

A classe *ListB* possui como atributo de classe: (i) um ponteiro para a "cabeça da lista", (ii) dois ponteiros para o "corpo da lista", sendo um para o "primeiro"

elemento inserido e outro para o último elemento e, (iii) um contador do número de elementos inseridos na lista. Tal implementação permite o acesso direto à “cabeça” da lista, ficando a classe *Body* encarregada de controlar a varredura na lista e o acesso aos elementos. Dessa maneira, é restringido o acesso à “cabeça” mas apenas à lista, pois somente elementos do tipo *Body* são liberados para serem varridos. A *ListB* é a única classe que possui o controle total sobre a “cabeça” da lista.

O resultado do projeto obtido para a *ListB* pode ser entendido melhor por meio do esboço de ambas as listas. Na Figura III-7 e na Figura III-8 é representada a iteração entre as listas que compõe a estrutura de dados B-rep. É utilizada a lista antiga na primeira figura e a nova lista na segunda. Comparando as figuras, é possível observar as alterações e simplificações feitas na estrutura atual. A nova lista ficou bem mais simples, mesmo mantendo as mesmas características funcionais. Com a simplificação na lista, o acesso aos dados na estrutura B-rep e sua construção ficaram mais rápidos. Isso se deve ao fato da STL ter uma estrutura de controle mais complexa do que a mostrada para a *ListB*.

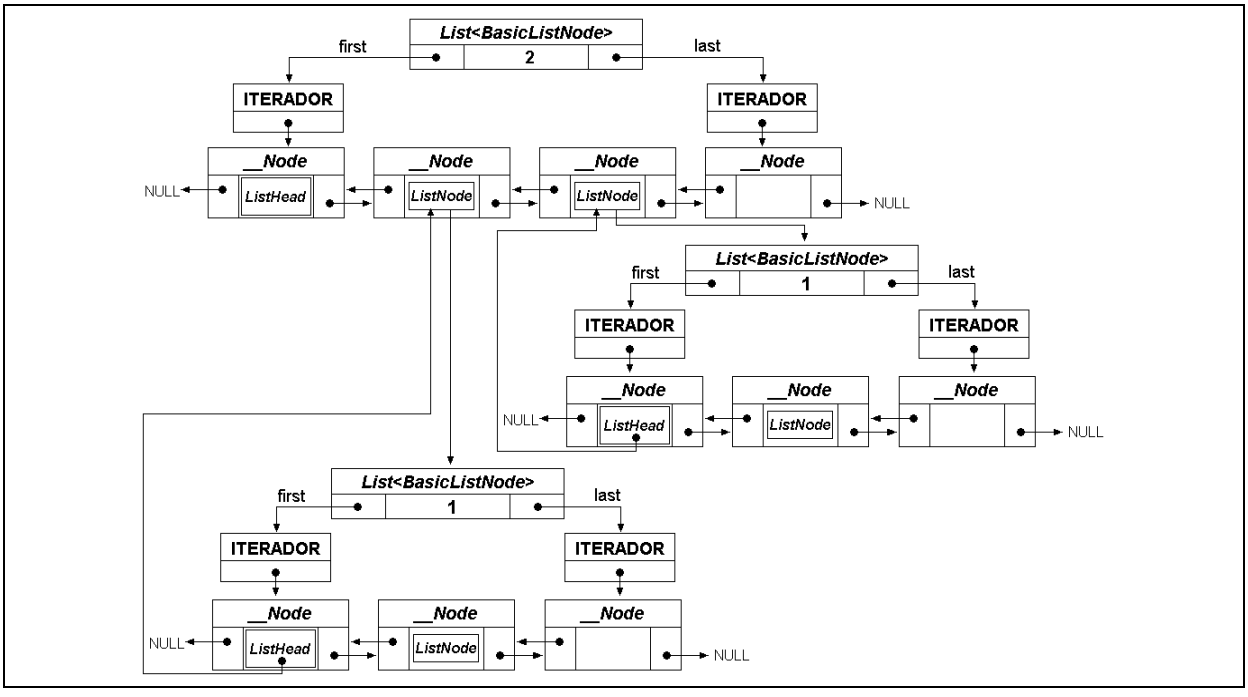


Figura III-7 – Estrutura de *List<BasicListNode>*

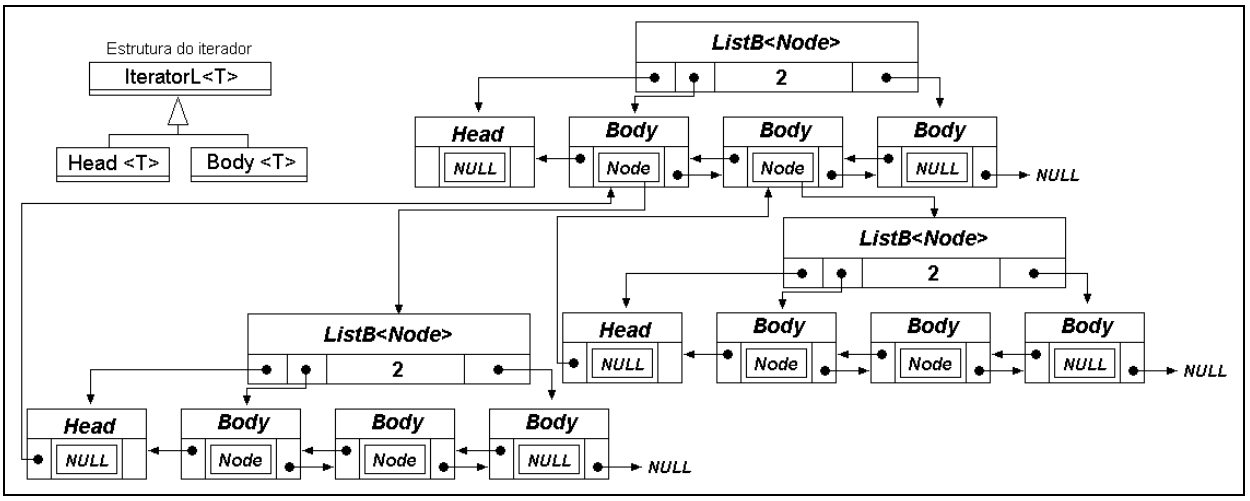


Figura III-8 – Estrutura de *ListB*

Como na nova lista a cabeça faz parte da hierarquia do iterador e o acesso aos seus elementos é feito pelo body e não mais pelo Iterator, foi necessário fazer algumas alterações na estrutura B-rep e no Subsistema de Representação para adaptá-los a essas mudanças. Nos demais subsistemas, adaptações semelhantes também foram necessárias.

III.2 – Arquivos de interseção x Estrutura na memória

Problemas relacionados à precisão numérica estão muito presentes nos cálculos computacionais e, principalmente, na modelagem de sólidos. Pequenos arredondamentos no cálculo das coordenadas de um ponto podem causar os mais diversos erros. O que acontece muito no GSM é a inserção errada dos pontos de interseção na estrutura de dados B-rep devido à determinação errada das arestas que os possui, causando, assim, distorções na topologia do modelo.

A Figura III-9 ilustra o caso de inserção errada do ponto na face, causando distorção na topologia da face. É possível observar que o correto seria a inserção do ponto na aresta BC e este foi inserido na aresta AB. Essa inserção errada é causada por problemas de precisão numérica. Uma explicação mais detalhada do problema é dada mais adiante nesta mesma seção.

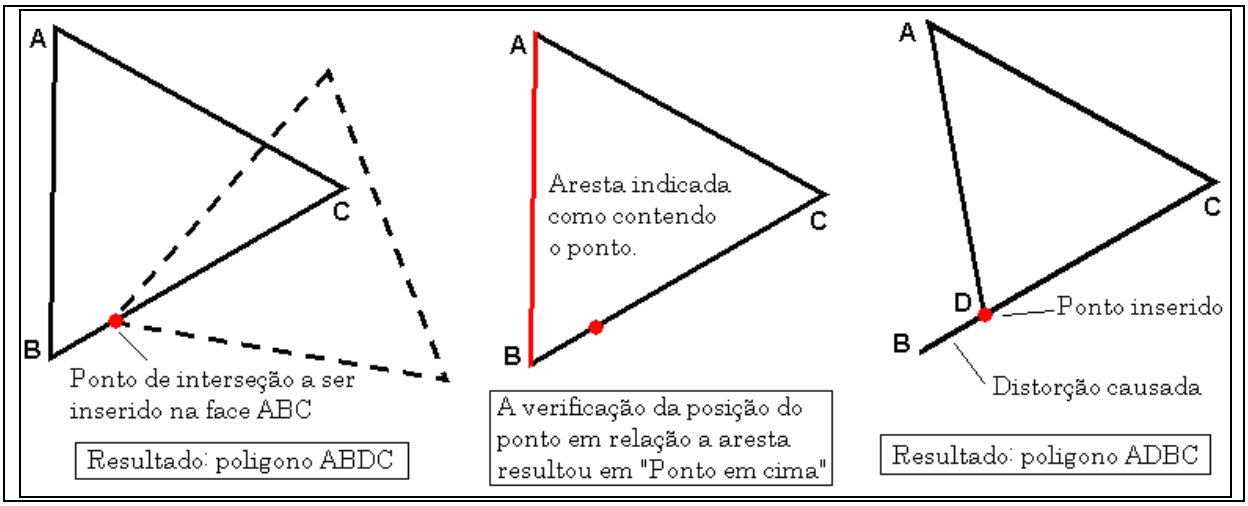


Figura III-9 – Exemplo de Inserção errada dos pontos.

Tentando minimizar o problema relacionado à precisão numérica, foi feita a troca dos quatro arquivos utilizados durante o processo da interseção por uma estrutura na memória. Essa foi outra modificação realizada no GSM. O uso da estrutura na memória evita possíveis truncamentos numéricos que acontecem quando um número é escrito em um arquivo ou lido dele, como é o caso das coordenadas dos pontos de interseção.

Como exposto na seção II.6.3 que descreve a avaliação da fronteira, para cada região A e B são utilizados dois arquivos. Um arquivo – *ptEdge1* – armazena os pontos de interseção posicionados sobre as arestas de uma das faces de A envolvidas na operação e um outro - *ptFace1* – armazena os pontos de interseção posicionados sobre o plano interno à face. O arquivo *ptEdge1, que* contém os pontos posicionados na aresta para a primeira região, possui os seguintes dados:

- Número identificador da face de A onde o ponto será inserido.
- Número identificador da face de B que também possui o ponto.
- Número identificador do posicionamento do ponto no contorno da face.
- A coordenada do ponto de interseção.

Já o arquivo *ptFace1* contém não só informação dos pontos posicionados no interior da face, como as arestas de interseção comuns às duas regiões A e B. Os dados são armazenados da seguinte forma:

- Número identificador da face de A onde o ponto será inserido.
- Número identificador da face de B que também possui o ponto.
- Número identificador de informação: ponto na face ou conjunto de arestas comuns às duas regiões.
- Se a informação é sobre o ponto, é escrita sua coordenada. Se a informação é sobre as arestas, primeiro, é escrita a quantidade de aresta e, em seguida, os pontos limites dessas arestas.

Depois de gerados os arquivos da interseção, o processo de inserção dos pontos na estrutura é iniciado. Isso é feito em duas etapas: primeiro são inseridos os pontos posicionados nas arestas, para depois serem inseridos os pontos posicionados no interior da face. Para a inserção de um ponto no contorno segue-se o algoritmo descrito no Quadro III-1.

```
Enquanto <não acabar o arquivo> faça
| para <cada linha lida do arquivo> faça
| | buscar a face na estrutura B-rep que possui a identificação.
| | para <cada aresta da face> faça
| | | se <ponto pertence a aresta> então
| | | | adicionar o ponto utilizando operado de Euler MEV
| | | fim se
| | fim para
| fim para
fim enquanto
```

Quadro III-1 – Algoritmo para inserir os pontos localizados no contorno da face.

As coordenadas dos pontos calculados na interseção, ao serem escritas no arquivo, são truncadas em 5 casas decimais o que diminui a precisão. Isso pode levar a duas situações distintas quando for feita a verificação se o ponto pertence a uma certa aresta X ou não:

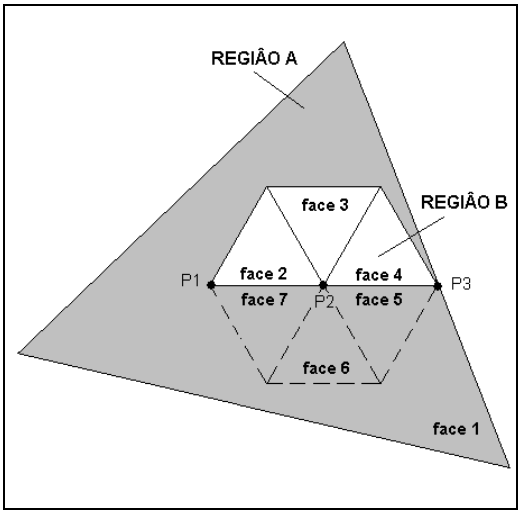
1. O ponto pertence à próxima aresta da face (aresta Y) e a verificação se o ponto pertence à aresta X resulta em verdadeiro, pois o ponto está muito próximo do vértice. Conseqüência: o ponto é inserido na aresta errada prejudicando a orientação da face, como mostrado na Figura III-9.
2. O ponto pertence à aresta X da face e a verificação se o ponto pertence à aresta X resulta em falso. Conseqüência: o ponto pode não ser inserido em nenhuma aresta. Isso afetará a etapa de classificação dos vértices.

Para solucionar esse problema, foi feita a substituição dos quatro arquivos de interseção por uma estrutura interna na memória. Optou-se por fazer uma troca de forma rápida e simples. Sendo assim, criou-se uma estrutura capaz de armazenar as informações obedecendo ao mesmo formato dos dados contidos nos quatro arquivos. A estrutura projetada e implementada reúne, para uma região A, todas as informações contidas nos arquivos *ptEdge1* e *ptFace1*, ou seja, traz informações sobre os pontos de interseção, que podem estar localizados na aresta ou no interior da face e sobre as arestas de interseção obrigatórias que devem aparecer na região de interseção.

Na estrutura, as informações contidas em cada linha dos arquivos foram armazenadas em uma classe denominada *InterPoint*. Tal classe possui como atributos:

- dois inteiros (nf1, nf2) que armazenam o identificador das faces onde o ponto está localizado sendo, uma face da região A e a outra da região B;
- um inteiro (id) que indica se as informações contidas no *InterPoint* são referentes aos pontos de interseção ou se são referentes às arestas obrigatórias que a região deverá apresentar na interseção. Se o *InterPoint* armazena informação sobre pontos de interseção (0,1 ou 2), o ponto pode se posicionar no vértice (0), na aresta (1) ou no plano interno da face (2). Se o *InterPoint* armazena informação sobre as arestas (4), este apresentará a aresta formada;
- um vetor de pontos contendo ou as coordenadas do ponto de interseção, ou as coordenadas dos pontos limites das arestas de interseção.

Assim, a estrutura de interseção armazena separadamente os *InterPoint* gerados tanto para a região A quanto para a região B. A Figura III-10 mostra o conjunto de *InterPoint* gerado para duas regiões A e B, durante o processamento da interseção entre uma face da região A e as faces da região B que se interceptam.



***InterPoint* - Região A**				***InterPoint* - Região B**			
nf1	nf2	id	Ponto/Aresta	nf1	nf2	id	Ponto/aresta
1	2	2	coord. P1	2	1	0	coord. P1
1	2	2	coord. P2	2	1	0	coord. P2
1	2	4	coord. P1-P2	2	1	4	coord. P1-P2
1	3	2	coord. P2	3	1	0	coord. P2
1	4	2	coord. P2	4	1	0	coord. P2
1	4	1	coord. P3	4	1	0	coord. P3
1	4	4	coord. P2-P3	4	1	4	coord. P2-P3
1	5	1	coord. P3	5	1	0	coord. P3
1	5	2	coord. P2	5	1	0	coord. P2
1	5	4	coord. P3-P2	5	1	4	coord. P3-P2
1	6	2	coord. P2	6	1	0	coord. P2
1	7	2	coord. P2	7	1	0	coord. P2
1	7	2	coord. P1	7	1	0	coord. P1
1	7	4	coord. P2-P1	7	1	4	coord. P2-P1

Figura III-10 – Interseção de triângulos não coplanares e os InterPoints

Os *InterPoint* relacionados à região A são adicionados em um vetor A e os relacionados à região B são adicionados em um vetor B. A estrutura da interseção consiste basicamente em um vetor de duas posições cuja primeira posição é o vetor A e a segunda posição é o vetor B. A separação da informação em dois vetores, um para cada região, agiliza o processo de recuperação da informação. Se não fosse feito dessa maneira, em cada *InterPoint* seria necessário armazenar informação sobre qual região ele está associado, além de ser necessário buscar a região para cada um deles. Isso significa dispor de mais memória e processamento.

Terminada a troca da lista que armazena os dados da estrutura B-rep e feitas as devidas adaptações nos subsistemas do modelador, iniciou-se a implementação de métodos que irão atuar na malha para melhorar o resultado das operações booleanas e operação de montagem do modelador. Tais métodos serão apresentados e discutidos na próxima seção, após a apresentação de uma revisão bibliográfica sobre o assunto.

Capítulo IV – Métodos para Melhorar as Malhas

Na utilização do FEM, a obtenção de resultados satisfatórios está diretamente relacionada à qualidade da malha final de “elementos finitos” [RAM90]. O tempo de execução depende do número de triângulos presentes na malha. A estabilidade e a convergência do método são bastante afetadas pela forma dos triângulos [REI02]. Os elementos que compõem a malha devem ser distribuídos de forma contínua, além de apresentar um formato adequado. No GSM, por se utilizar uma malha triangular, é desejável que tais elementos tenham sua forma a mais próxima possível de triângulos eqüiláteros.

Devido à importância da qualidade da malha de elementos finitos, vários pesquisadores se empenham em estudar, descrever e comparar métodos utilizados em aplicações que necessitam de uma malha superficial de boa qualidade. Tais métodos devem ser capazes de gerar malhas superficiais sobre modelos diversos, avaliar a sua qualidade, fazer a junção de duas ou mais malhas ou modificar e melhorar a qualidade da malha. Owen [STE98] faz uma comparação de várias técnicas empregadas na geração e refinamento de malha, tanto superficial quanto volumétrica.

Nesse capítulo, as principais abordagens utilizadas na melhoria da qualidade da malha superficial de modelos sólidos serão apresentadas e discutidas. Será analisada a viabilidade de implementação, no modelador de sólidos, de algoritmos seguindo essas abordagens. Por último, serão apresentadas as metodologias empregadas e as estratégias adotadas, no GSM, para melhorar a qualidade da malha sobre os modelos resultantes das operações booleanas.

IV.1 – As Principais Abordagens para Melhoria da Malha

Na literatura, existem diversos métodos e algoritmos desenvolvidos para serem aplicados na área de modelagem de sólidos. Estes são encarregados de fazer:

- A discretização de modelos, gerando malhas superficiais de elementos finitos de boa qualidade. A triangulação de Delaunay, descrita por Renka [REN97], é um exemplo de algoritmo muito utilizado na geração da malha superficial de modelos;
- O cálculo do fator de qualidade de uma malha. O valor é dado pelo elemento da malha de pior forma. Essa forma é analisada de duas maneiras: pela razão do tamanho de sua maior aresta pelo tamanho da menor de suas alturas [REI02], ou pela razão entre o diâmetro do círculo inscrito e o raio do círculo circunscrito [RAM90].
- O pós-processamento da malha. Esse pós-processamento pode trabalhar tanto a forma quanto a densidade de malha;
- A determinação e o controle dos pontos de interseção nos casos onde há a junção de duas ou mais malhas. O controle é feito de forma a garantir que a malha final tenha uma boa qualidade.

Para este trabalho, cujo objetivo é melhorar a qualidade da malha resultante das operações booleanas ou da operação de montagem, foram analisados os métodos que permitem avaliar, garantir ou melhorar a qualidade da malha. Garantir a qualidade da malha sobre um modelo é uma tarefa muito árdua, pois é necessário ter o total controle sobre os seus elementos. A qualidade da malha depende da continuidade, homogeneidade e da forma dos elementos que a constituem. Além disso, a malha deve garantir a descrição correta do modelo real. Isso significa que, para fazer a modificação nos nós de malha, um conjunto de requisitos deve ser verificado para que o resultado seja correto.

Nas seções a seguir, serão apresentados, analisados e discutidos alguns métodos estudados que permitem fazer o refinamento da malha, controle de pontos da interseção e determinação da posição dos pontos de interseção.

IV.1.1 – REFINAMENTOS NA MALHA

O refinamento de malha pode ser aplicado de maneira a tratar a forma dos elementos que a compõem, tornando-os mais próximos da forma ideal ou de forma a aumentar a quantidade de elementos com a inserção de novos nós.

SABONNADIÈRE descreve o método de Suavização Laplaciana que é capaz de melhorar a qualidade da malha, ajustando a forma de seus elementos. Nesse método, a forma dos elementos é modificada pelo deslocamento dos nós centrais da malha [SAB93]. O deslocamento dos nós é baseado em informações apenas dos vértices vizinhos, não computando nenhum outro tipo de informação – por exemplo, informações sobre a geometria do modelo. O método consiste basicamente em definir na malha um polígono que possua apenas um nó central. Calcula-se o baricentro desse polígono. Move-se o nó central para o baricentro calculado. Esse processo é repetido até que nenhum nó da malha seja mais movido.

Existem algumas variações desse método que fazem o deslocamento do nó computando informações, não só dos nós vizinhos, mas também da geometria do modelo. Basicamente, o que esses métodos fazem é analisar o baricentro calculado em relação à curvatura da superfície original do modelo, fazendo a sua projeção para ela. Entretanto isso nem sempre é possível, pois não é todo sistema de modelagem que armazena informação sobre as superfícies curvas originais. Há aplicações que recebem a malha superficial do modelo pronta e fazem apenas a verificação da qualidade da malha e, se esta apresentar baixa qualidade, é refinada com o intuito de se obter melhores resultados.

A Figura IV-1 mostra em A), uma superfície curva que foi malhada por um malhador genérico; em B) o resultado obtido aplicando-se o método de refinamento descrito por SABONNADIÈRE, que ajusta a forma computando apenas informações da malha e em C) o resultado obtido aplicando-se uma variação do método de refinamento descrito por SABONNADIÈRE que ajusta a forma computando-se informações da malha e da superfície original.

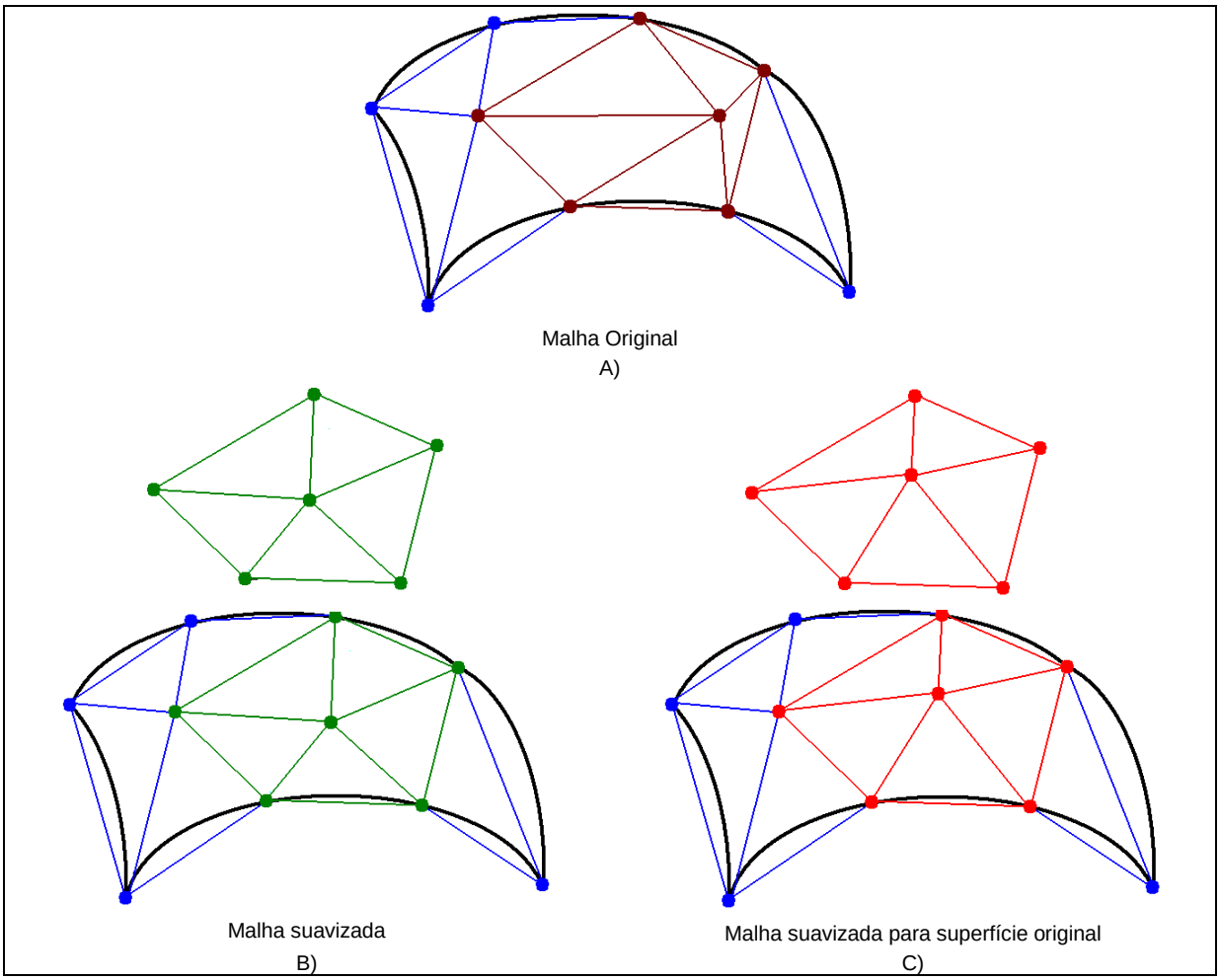


Figura IV-1 – Refinamento da forma da malha.

Aplicar este método de pós-processamento à malha, ajustando-se apenas a forma de seus elementos, pode levar a duas situações (i) melhorar a qualidade da malha, mas não garantir que a malha seja boa o suficiente para usá-la na geração da malha volumétrica, ou (ii) pode não alterar a qualidade da malha, pois elementos mal formados podem não sofrer alteração. Esses casos acontecem com freqüência em duas situações distintas: (ii-a) quando a densidade da malha é muito desigual; (ii-b) quando o modelo resulta da junção de malhas sem que haja qualquer tipo de controle do posicionamento dos pontos de interseção e do tamanho das arestas, gerando, assim, contornos com arestas de dimensões muito pequenas. A Figura IV-2 ilustra estas situações, mostrando o modelo antes e depois do ajuste. Em (A), é mostrado um ajuste aplicado em uma malha cuja densidade é bem desigual e, em (B), é mostrado um modelo resultante de uma operação booleana ao qual aplicou-se

o método e não houve o ajuste na malha. É possível observar nos dois casos apresentados que a malha final ainda apresenta elementos mal formados, como estão destacados na figura.

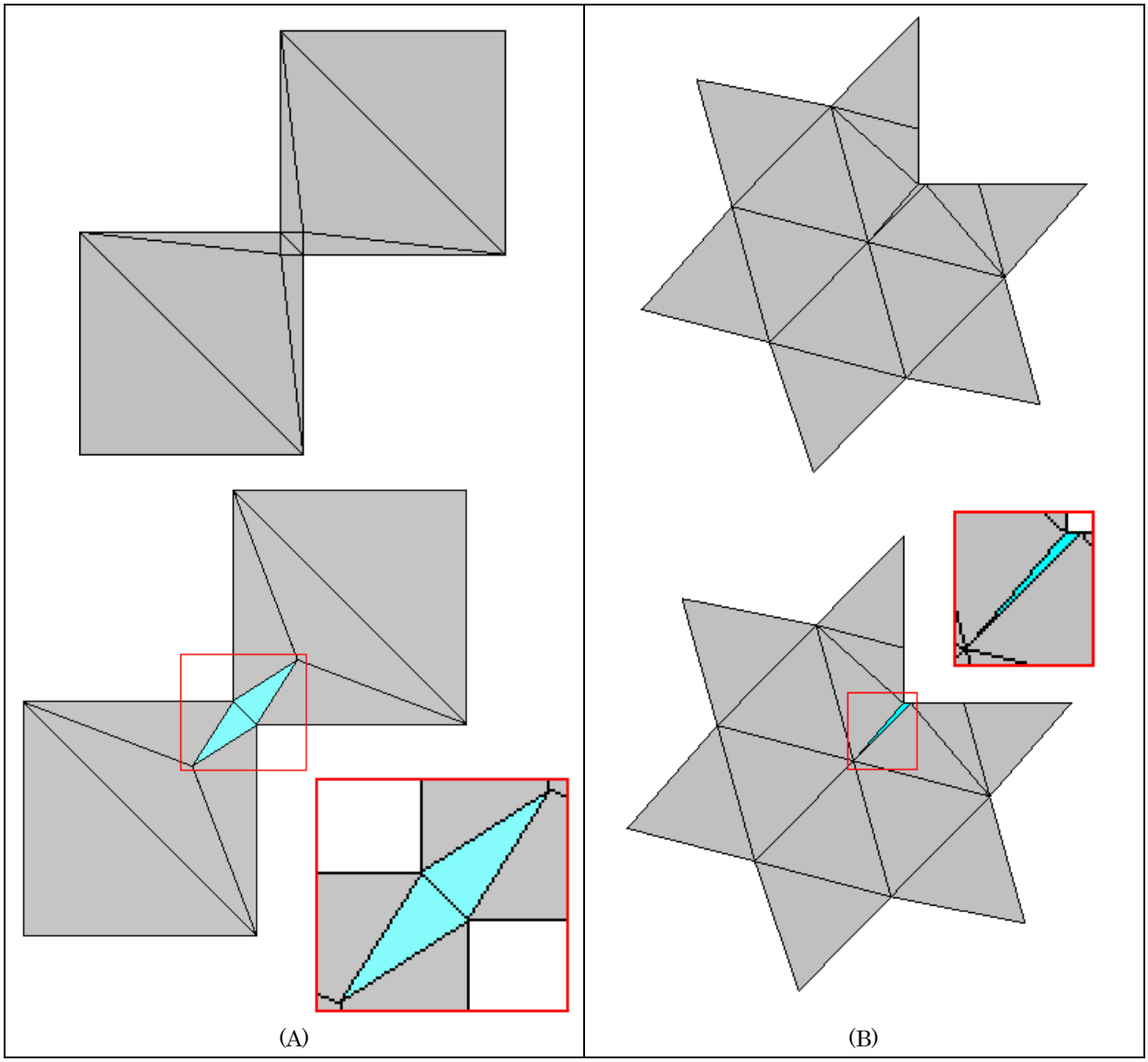


Figura IV-2 – Ajuste da malha

Mesmo que os elementos da malha possuam boa forma, não se garante que o método de elementos finitos convirja e que os resultados obtidos sejam bons. Isso acontece por que é necessário que, em alguns pontos do modelo, a malha seja mais densa devido à distribuição de campo (cálculos eletromagnéticos) ou de forças sobre o modelo (cálculos mecânicos). Ramirez [RAM90] apresenta dois métodos capazes

de fazer a subdivisão dos elementos da malha aumentando sua densidade. Aumentar a densidade de malha consiste em adicionar à malha novos nós, diminuindo a área de seus elementos. Devido a essa característica, tal abordagem não é interessante para o GSM, pois o Tetgen, o gerador de malha volumétrica que foi acoplado no modelador, não suporta modelos cuja malha superficial tenha elementos com área muito pequena e muito variada.

IV.1.2 – CONTROLE DA MALHA DURANTE A INTERSEÇÃO

Na seção anterior, os métodos descritos são aplicados em regiões planares, com malha de baixa qualidade, com o intuito de se obter uma malha final de melhor qualidade que a inicial. Entretanto, esses métodos não se aplicam adequadamente em modelos resultantes da junção de duas ou mais malhas ou resultante de uma operação booleana. Ao se aplicar uma união, diferença, interseção ou montagem sobre dois objetos já discretizados com malha superficial, estes possuirão regiões de sobreposição da malha que podem gerar, no contorno, arestas muito pequenas que resultarão em elementos mal formados. Na seção de resultados, esse tipo de problema é bem ilustrado.

Para os casos onde há a união de duas malhas de modelos distintos para obtenção de um outro modelo, o que se utilizam para melhorar a malha final são estratégias que determinam e controlam o posicionamento dos vértices da região de interseção das malhas. Dessa maneira, é possível evitar que elementos mal formados sejam gerados. Tais elementos podem possuir ângulos ou áreas muito pequenas e a presença deles na malha final pode tornar inviável a aplicação de qualquer método de refinamento da malha pois pode gerar resultados indesejados – por exemplo, regiões com malha muito densa e seus elementos muito pequenos – ou causar grande deformação na descrição do modelo real.

Como tentativa de determinar os pontos de interseção e controlar o melhor posicionamento destes para a geração da malha final sobre o modelo, duas metodologias diferentes foram implementadas no GSM e que serão apresentadas a seguir. Existem outras metodologias que poderiam ser aplicadas para melhorar a

interseção entre malhas, mas que não serão apresentadas nesse trabalho por não serem aplicáveis ao modelador. Tais métodos seriam aplicáveis em sistemas que armazenam as informações geométricas separadamente da malha e que, ao fazer a interseção de objetos, eliminam a malha existente sobre as regiões de interseção, unem os objetos usando as informações geométricas e geram uma nova malha sobre o modelo obtido. No modelador, como o estruturado no momento, isso não pode ser feito, pois a malha sobre os objetos é gerada da discretização do modelo e ela armazena as informações geométricas e topológicas do objeto.

Uma outra possibilidade está sendo explorada no GOPAC [NUN04]. Neste trabalho é feita a aproximação da superfície implícita por superfícies parametrizadas, e se utilizam métodos para solução do problema a partir de modificações locais da malha superficial.

Coelho, Gattass e Figueiredo [COE99] descrevem um algoritmo eficiente que permite juntar duas regiões malhadas, garantindo a geração de uma malha de boa qualidade sobre a superfície da região de interseção. O algoritmo pode ser dividido em três etapas distintas, conforme apresentadas no fluxograma da Figura IV-3.

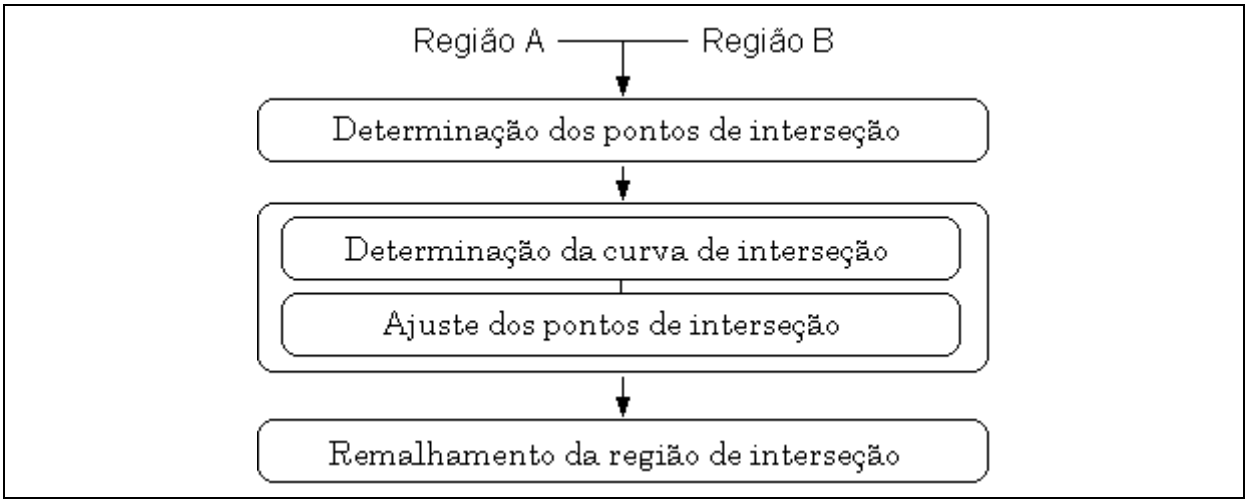


Figura IV-3 – Fluxograma do processo de junção de duas malhas.

Para cada uma das três etapas do processo mostrado no fluxograma, há um conjunto de passos a serem seguidos e executados de forma a garantir que a malha resultante da interseção seja de boa qualidade. Esses passos estão apresentados no Quadro IV-1.

1. Determinação dos pontos de interseção:
   a) Calcular e armazenar os pontos de interseção das arestas da região A com as faces de B;
   b) Calcular e armazenar os pontos de interseção das arestas da região B com as faces de A;
2. Determinação da curva de interseção
   a) Agrupar os pontos de interseção obtidos no passo 1 em um conjunto de linhas poligonais que irá representar a curva de interseção;
   b) Fazer a Interpolação paramétrica da curva com os pontos;
   c) Calcular a distribuição uniforme dos pontos sobre a curva paramétrica;
   d) Mover os pontos para cada superfície.
3. Remalhar
   a) Determinar as sub-regiões para remover seus vértices e arestas próximas à curva de interseção
   b) Inserir novas arestas usando os novos pontos contidos na curva de interseção
   c) Triangularizar as sub-regiões da superfície
   d) Ajustar a malha.

Quadro IV-1 – Algoritmo para junção das malhas.

Na tentativa de ilustrar o processo de forma a ajudar a visualizar e facilitar o entendimento de cada uma dessas etapas, a Figura IV-4 mostra, de (A) a (F), os principais passos de cada etapa do processo.

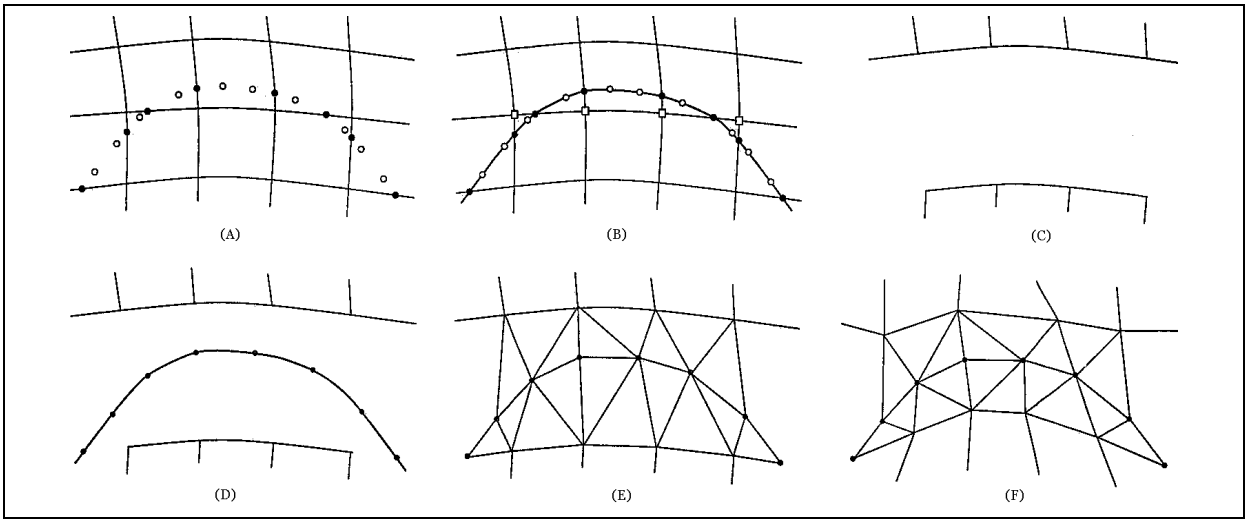


Figura IV-4 – Processo para junção das malhas.

A Figura IV-4 mostra em (A) os pontos de interseção computados na etapa um, sendo os círculos preenchidos equivalente aos pontos obtidos no passo (1, a) e os vazados, os obtidos no passo (1, b) dessa primeira etapa do processo. Em (B), é

mostrada a curva de interseção obtida da segunda etapa do processo. A curva é gerada a partir dos pontos de interseção. Em (C), é apresentado o resultado da eliminação dos elementos da malha, como arestas vértices e faces, interceptados pela curva de interseção, enquanto que em (D), são apresentadas as novas arestas que irão compor a curva de interseção. Observe que nessa etapa do processo o número de pontos na interseção é menor que o número inicial. Isso acontece devido ao ajuste dos pontos de interseção de maneira a homogeneizar as distâncias entre os pontos. Pontos próximos são agrupados em um único ponto. Finalizando, em (E) é apresentada a nova malha da região e, em (F), o resultado obtido após o ajuste da malha.

O algoritmo apresentado é possível ser incorporado ao modelador devido às semelhanças existentes no GSM e no sistema desenvolvido em [COE99]. O algoritmo apresenta os mesmos passos seguidos pelo GSM para determinar os pontos de interseção, exceto o controle e ajustes dos pontos de interseção. Além disso, é utilizada uma estrutura de dados do tipo DCEL – *Doubly-Connected Edge List* [BER00] – para armazenar as informações sobre a malha, que tem funcionalidades semelhantes à do modelador. Entretanto, o modelador, da forma como está estruturado, não permite a geração de objetos não manufaturáveis o que faz com que haja a necessidade de se fazer uma adaptação no método. O método gera uma descontinuidade temporária no modelo, quando há a eliminação dos elementos da malha (arestas, vértices e faces). Outro problema presente no modelador é de não haver separação entre a representação geométrica e da representação da malha. Destruindo parte da malha, é perdida parte da informação sobre a geometria do modelo.

Outro método é descrito por White e Saigal [WHI02], que mostram problemas que ocorrem ao juntar duas regiões previamente malhadas e sugerem um novo algoritmo que permite computar e gerar o contorno de interseção, como mostrado na Figura IV-5, e agrupar duas regiões adjacentes utilizando, para isso, uma tolerância de interseção como entrada.

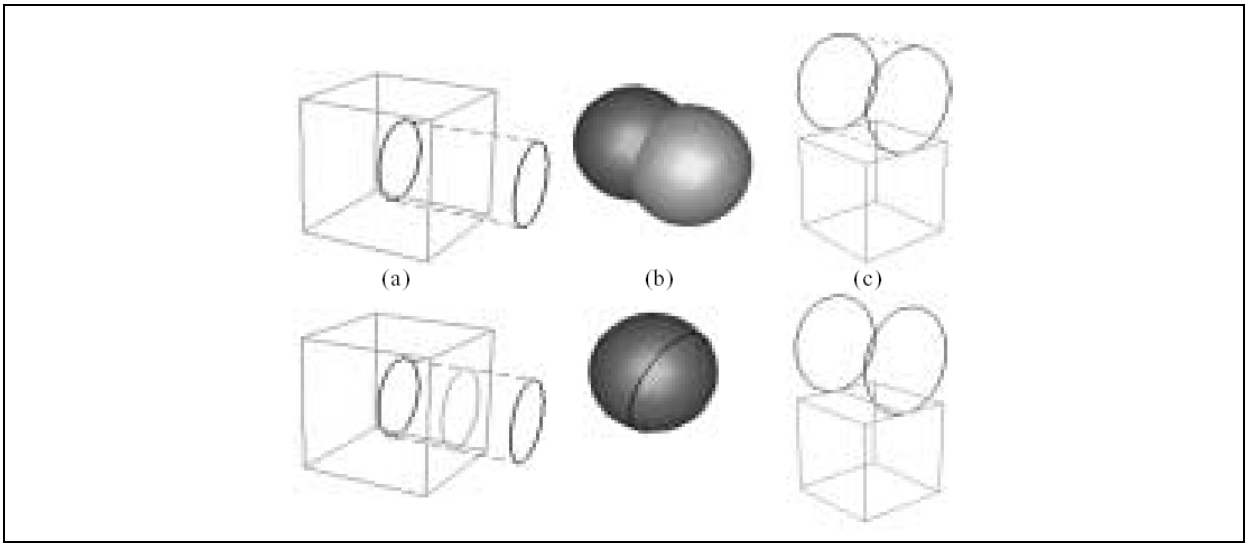


Figura IV-5 – Contorno da interseção de duas regiões

O algoritmo responsável por gerar o contorno de interseção consiste nas cinco etapas seguintes:

1. Discretizar a fronteira
2. Interceptar os segmentos de reta
3. Construir o gráfico parcial de interseção
4. Construir as novas arestas
5. Separar arestas e faces antigas e juntar novas faces

O primeiro passo do algoritmo faz a aproximação das faces por arestas, ou seja, nesse passo é feita a discretização do modelo, como no GSM. No segundo passo, é computada a interseção entre os segmentos de reta. Nessa etapa, para cada segmento de reta, é feita a interseção com os elementos próximos a eles contidos na outra face. Isso pode ser feito eficientemente pelo uso de uma estrutura de dados R-Tree. Ao computar as interseções, uma **zona de amortecimento** entre as arestas é criada, de forma que qualquer espaço entre dois segmentos, que estejam nela contidos, constitui uma interseção. Isso é ilustrado na Figura IV-6. A zona de amortecimento é definida a partir da tolerância de entrada.

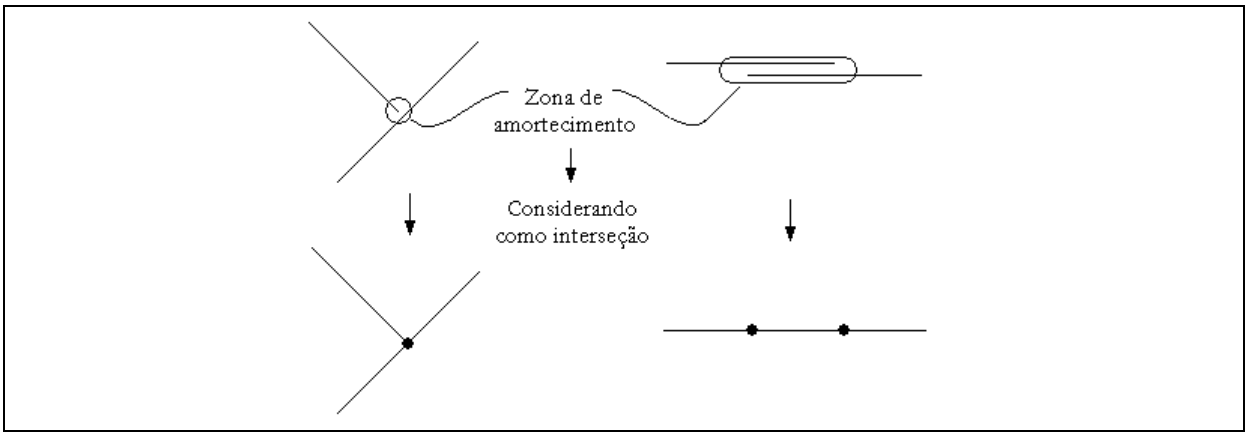


Figura IV-6 – Zona de amortecimento da interseção

Os pontos de interseção obtidos nessa etapa são classificados e adicionados em uma estrutura denominada de grafo parcial de interseção, para depois serem incluídos no modelo. Esses pontos são obtidos da interseção entre as arestas. Na Figura IV-7 são apresentadas as maneiras com que os segmentos podem se interceptar.

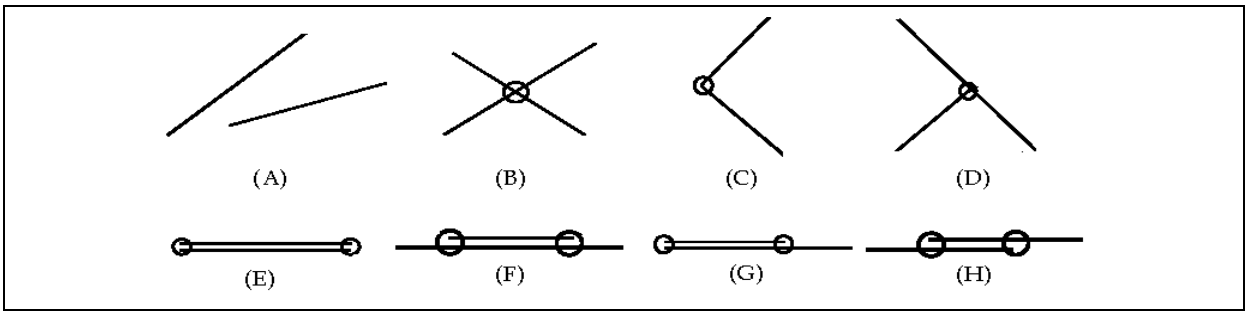


Figura IV-7 – Possíveis interseções entre segmentos

IV.2 – Métodos Implementados no Modelador

Nessa seção, serão apresentadas as principais funções implementadas no GSM durante a realização desse trabalho. Essas funções são utilizadas no modelador com o intuito de avaliar a qualidade da malha sobre um objeto, suavizar a malha do modelo, determinar o melhor ponto de interseção ou eliminar pontos próximos que geram elementos mal formados.

IV.2.1 – AVALIAÇÃO DA QUALIDADE DA MALHA

A primeira função implementada no GSM foi a *MeshAval().* Essa função avalia a qualidade da malha gerada sobre a superfície do objeto, seja um objeto recém modelado ou um objeto resultante de uma operação booleana e/ou operação de montagem. Essa função está disponível no modelador como uma opção do menu. A Figura IV-8 mostra a barra de menu com a opção *Avaliate* para chamada da função de avaliação da qualidade da malha. A barra de menu traz também, a opção *Shape* que faz chamada à função que aplica a suavização laplaciana, ajustando os nós da malha. Essas funções permitem ao usuário selecionar a quais regiões elas deverão ser aplicadas.

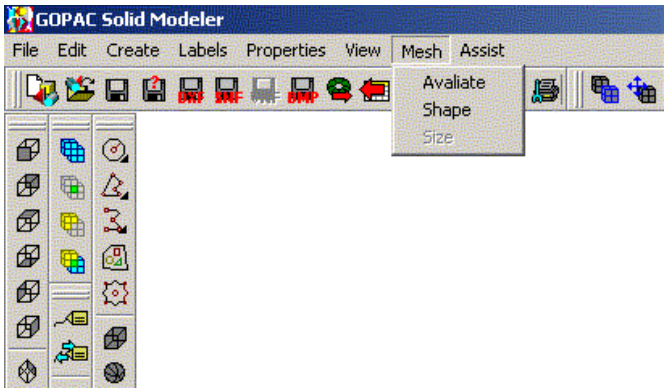


Figura IV-8 – Comandos para a malha no menu de opções.

Avaliar a qualidade da malha consiste em analisar a forma de cada um dos elementos que a constituem, sendo o elemento de pior forma o determinador do fator de qualidade da malha. Para o GSM, os elementos da malha devem ser o mais próximo de triângulos eqüiláteros, ou seja, o fator de qualidade da malha é dado pela forma dos triângulos da malha. O fator é o resultado do valor mais baixo obtido da análise do fator de qualidade de cada triângulo que compõe a malha que está sendo analisada.

Seja um triângulo ABC cujos lados medem a, b e c respectivamente. O fator de qualidade q do mesmo é dado pela equação:

$$q = \frac{8 \times (s-a) \times (s-b) \times (s-c)}{a \times b \times c} \qquad (1)$$

onde s é o semi-perímetro do triângulo ABC.

No GSM, a avaliação da qualidade da malha é feita por região. Escolhida a região, o modelador varre todas as suas faces, calculando seu fator de qualidade e retorna o menor valor obtido.

O fator de qualidade de um triângulo pode variar de zero a um, sendo este, respectivamente, um triângulo degenerado em um conjunto de pontos colineares e um triângulo eqüilátero [LIN83]. Essa variação da forma e do fator de qualidade dos triângulos é mostrada na Figura IV-9. O primeiro triângulo apresentado é um triangulo eqüilátero (fator de qualidade próximo de 1) e o último próximo de um segmento de reta (fator de qualidade próximo de 0).

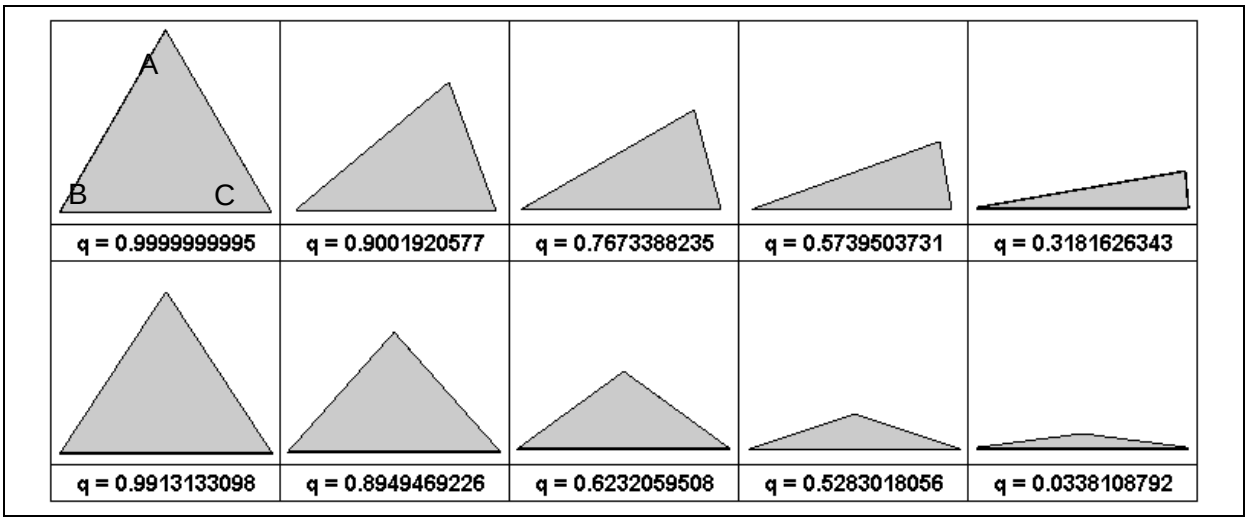


Figura IV-9 – Fator de qualidade para os triângulos.

IV.2.2 – MÉTODO DE SUAVIZAÇÃO LAPLACIANA

Como primeira tentativa de ajuste para melhoria da qualidade da malha superficial, foi implementado o método de suavização laplaciana que consiste em deslocar o nó central do polígono para o seu baricentro. Esse método é muito utilizado para ajustar a forma dos elementos de malha de modelos planares, além de ser de fácil implementação, como pode ser constatado pelo algoritmo apresentado no Quadro IV-2.

```
Enquanto <algum vértice do modelo for deslocado de um valor X> faça
  para <cada vértice pertencente a região> faça
    Pegue todas as faces da região que possui o vértice atual
  fim para
  se <encontrado mais de uma face> faça
    se <todas as faces forem planares> faça
      para <cada face da região que possui o vértice atual> faça
        armazene os dois outros vértices em uma lista
      fim para
      calcule o baricentro desse polígono obtido com os vértices na lista.
      Desloque o vértice atual para o baricentro calculado
    senão verificar próximo vértice da lista.
    Fim se
  fim se
fim enquanto
```

Quadro IV-2 – Algoritmo para ajuste da forma de malha

Como o GSM trabalha com modelos não planares, esse método não poderia ser aplicado. Entretanto, para modelos que possuem partes planares, este tipo de método pode ser aplicado, mas apenas no conjunto de faces que estão no mesmo plano.

Observando rapidamente a Figura IV-10 é possível constatar a existência de modelos compostos por um conjunto de faces planares e não planares. Além disso, como no modelador a própria malha descreve a topologia e a geometria do modelo, cada nó possui um grau de liberdade para definir se ele pode ou não ser deslocado e como ele pode ser movimentado (sobre a aresta, face ou espaço). Sendo assim, tornou-se necessário fazer uma complementação no método de maneira que ele agora identifique as regiões planas e os nós que podem ser movimentados.

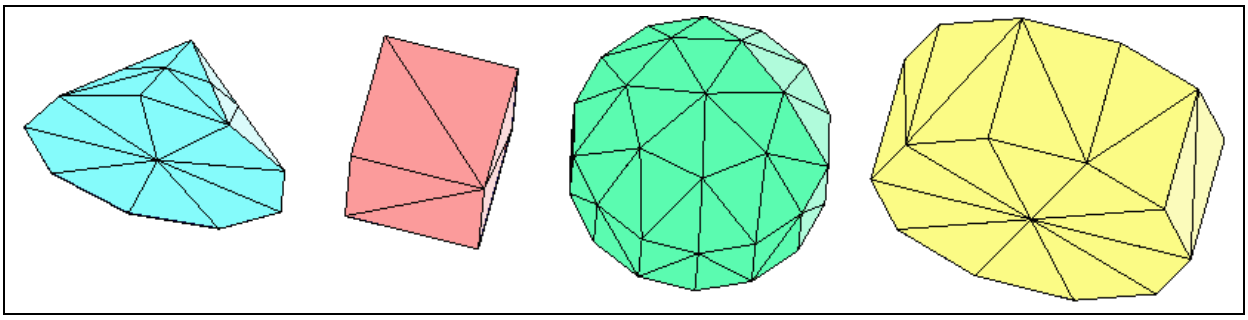


Figura IV-10 – Primitivas contendo conjunto de faces planares

A metodologia utilizada para identificar se um nó é candidato a ser ajustado pelo método é simples. O que é feito, basicamente, é varrer o modelo e, para cada face, buscar um nó e verificar se ele é um vértice que pode ser alterado. Se ele puder ser movido, as faces a ele incidentes são analisadas. Durante a análise, se uma das faces não pertencer ao mesmo plano das outras, o processo é interrompido e um outro nó é buscado. Se todas as faces estão no mesmo plano, o baricentro é calculado e a posição do nó é ajustada. Esse processo é repetido até que todos os nós do modelo não sofram mais nenhuma alteração.

Para a maioria dos modelos com regiões planares, o resultado final é uma malha com elementos melhor formados. Entretanto, existem casos onde o resultado do método não é satisfatório. Ao aplicar uma operação booleana em um modelo, pode acontecer que dois vértices do contorno que não podem ser movidos fiquem muito próximos, formando arestas muito pequenas que resultarão em elementos com ângulos muito pequenos, como mostrado na Figura IV-11. Para esse caso, a solução proposta na literatura é aumentar a densidade da malha nesse ponto, adicionando novos nós ao modelo. Entretanto, essa não é uma boa solução no uso do GSM, pois nesse caso, a malha apresentaria elementos com áreas muito pequenas que o Tetgen – gerador de malha volumétrico acoplado ao GSM – não a suportaria como entrada, devido à restrição já comentada anteriormente nesse trabalho.

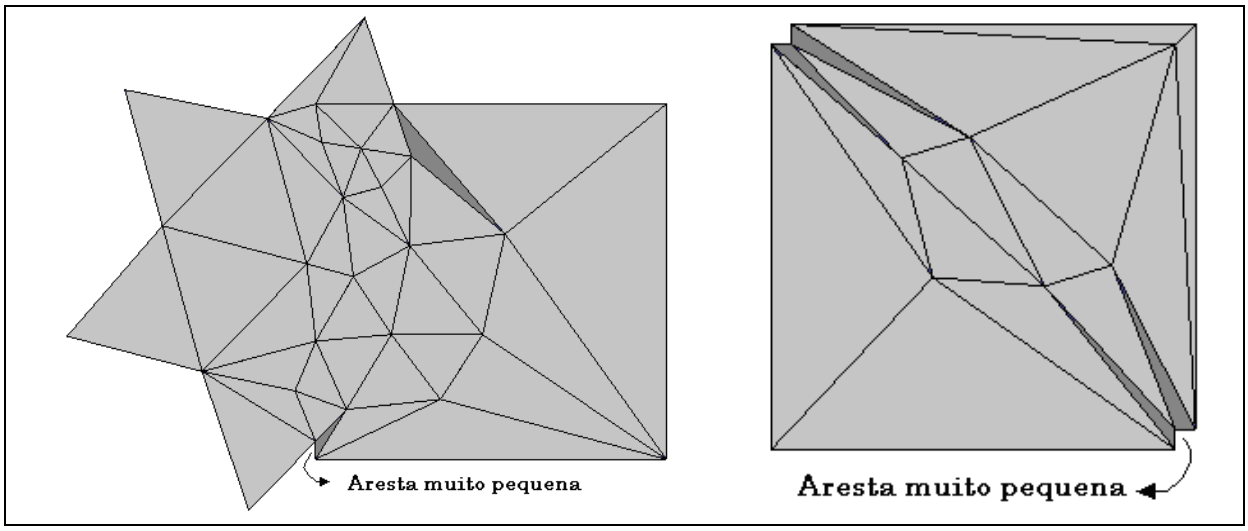


Figura IV-11 – Malha superficial com aresta muito pequena

Para as demais regiões e, até mesmo, para modelos “curvos”, a melhor opção foi aplicar métodos que determinam e controlam a inserção dos pontos de interseção nos modelos, na tentativa de evitar que arestas ou ângulos muito pequenos sejam formados. A descrição dos métodos implementados está na subseção a seguir.

IV.2.3 – CONTROLE DOS PONTOS DE INTERSEÇÃO

Com o objetivo de desenvolver e aplicar métodos capazes de melhorar a qualidade da malha resultante da operação booleana e com o intuito de tentar evitar que elementos mal formados apareçam na malha final sobre o modelo, foram realizadas algumas modificações no processo de determinação da interseção utilizada pelo GSM. Essas mudanças visam controlar o posicionamento dos pontos de interseção de maneira a eliminar a formação de triângulos de áreas extremamente pequenas que podem ser considerados como um único nó ou uma única aresta da malha e a distribuir melhor os pontos no contorno de interseção.

As modificações realizadas no GSM foram feitas principalmente no segundo passo da avaliação de fronteiras, que é responsável por fazer a interseção das cascas de dois operandos sólidos em uma operação booleana ou em uma operação de Montagem. Além disso, foram implementadas algumas rotinas para determinar e controlar o posicionamento dos pontos de interseção durante a avaliação das fronteiras de duas regiões distintas.

Como estratégia para controlar os pontos da interseção, de modo que a malha não possua pontos muito próximos ou elementos mal formados, foram desenvolvidas rotinas que agrupam elementos cujas distâncias são menores que um certo valor de tolerância, transformando pontos que estão próximos em um único nó, aproximando vértices de arestas, inserindo-os nas arestas ou aproximando duas arestas que se cruzam, inserindo o ponto de interseção nelas. Isso é melhor representado na Figura IV-12 que exemplifica, primeiramente, a aproximação dos pontos, em seguida a aproximação de um ponto a uma aresta e por último a aproximação de duas arestas.

Como o GSM trabalha com uma malha cujos elementos são triângulos, a etapa que determina os pontos de interseção da região comum aos dois operandos é feita através da análise dos triângulos de uma região em relação aos triângulos da outra. Como as rotinas de determinação e controle dos pontos de interseção podem inserir, na face triangular, um ou mais pontos a face deixa de ser triangular e passa a ser poligonal. Sendo assim, a principal modificação realizada no processo de obtenção dos pontos de interseção foi modificar suas rotinas de maneira que elas trabalhem com faces poligonais ao invés de faces triangulares. Para isso, algumas outras funções foram criadas.

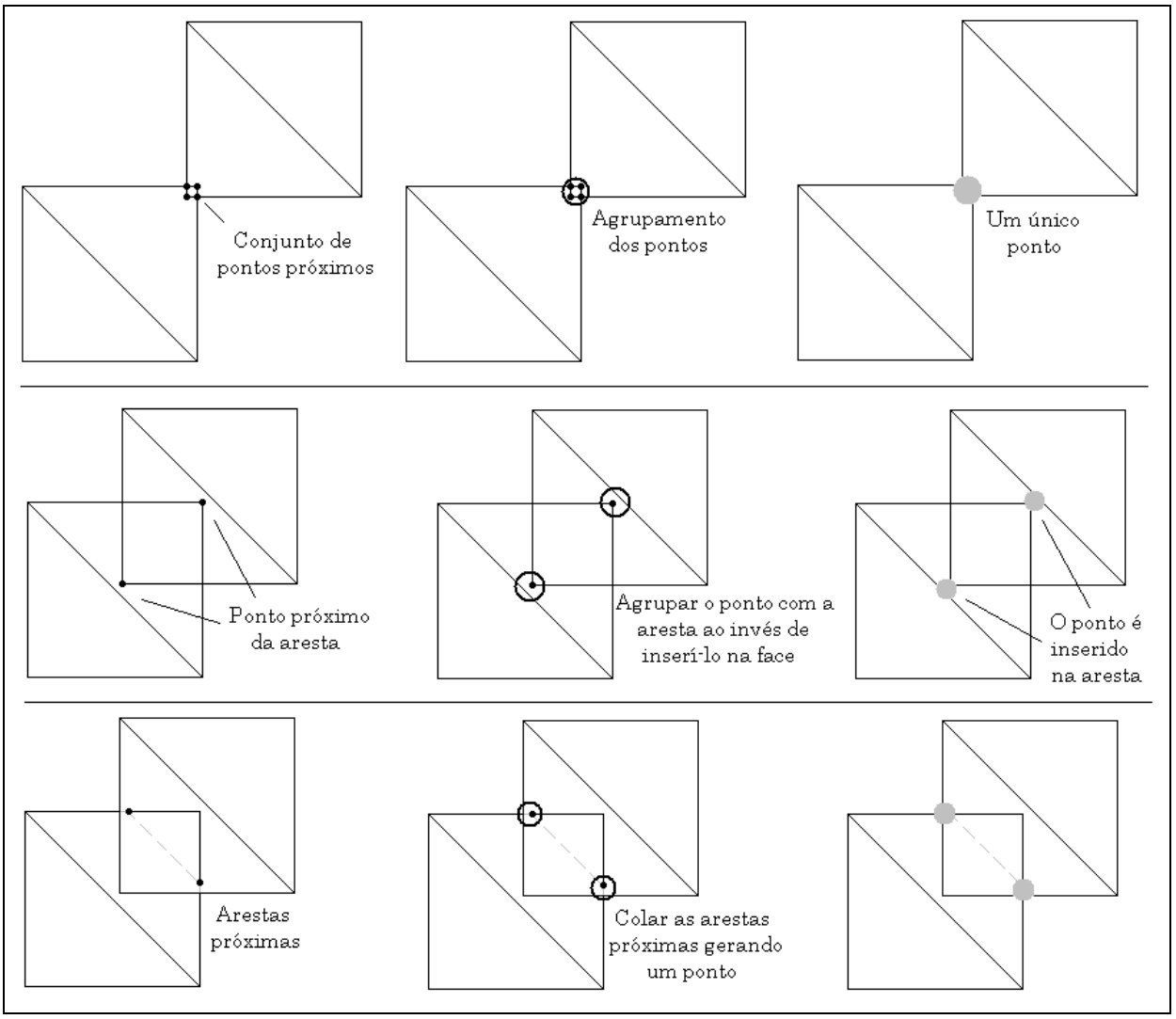


Figura IV-12 – Agrupamento de pontos próximos.

Nas próximas seções, descrições detalhadas das funções implementadas e das modificações realizadas no modelador para tornar a geração dos pontos de interseção poligonais são apresentadas. Primeiramente, serão apresentadas as funções responsáveis por aproximar os elementos e, posteriormente, as alterações realizadas no modelador quanto à análise de faces poligonais na obtenção da interseção.

IV.2.3.1 – AS FUNÇÕES DE APROXIMAÇÃO - JOINNEAR

Para ajustar os pontos de interseção que são muito próximos, foram implementadas três funções que precedem o processo de determinação e geração da interseção. As funções fazem a aproximação dos elementos cuja distância é inferior a um valor de tolerância que é calculado, a priori, pelo modelador e passado como parâmetro. Os elementos analisados podem ser dois vértices, duas arestas ou um ponto e uma aresta que estão tão próximos que podem ser considerados como elementos que se "interceptam". Estes elementos estão em faces distintas de duas regiões quaisquer. As funções responsáveis por analisar os elementos e realizar a aproximação entre eles, foram denominadas de:

- *JoinNearPP* que ajusta dois pontos próximos para um único ponto médio,
- *JoinNearEE* que ajusta duas arestas próximas.

Estas funções foram preparadas para suportar análises em faces poligonais, além de inserir novos pontos no contorno das faces.

Para classificar se dois elementos (vértice-vértice, vértice-aresta ou aresta-aresta) estão próximos ou não um do outro, é utilizado um valor de tolerância que é calculado pelo GSM. O valor da tolerância é dado pela décima parte do menor tamanho das arestas que formam as faces envolvidas no ajuste. Essa classificação é feita por cada uma das funções que utilizam estratégias distintas.

- A FUNÇÃO JOINNEARPP

Esta é a função responsável por ajustar os pontos de duas faces (1 e 2) diferentes pertencentes a duas regiões (A e B) que estão próximas a menos da tolerância. A estratégia que foi utilizada neste caso consiste em varrer cada um dos pontos da região A em relação aos pontos da região B e verificar quais os pares de pontos (região A, região B) que possuem uma distância entre si menor que a tolerância de entrada, para assim, ajustá-los para a posição média entre eles.

O algoritmo utilizado para buscar os pares de pontos próximos de uma região com a outra, se baseia na análise da interseção de duas faces. Se a face 1 da região A intercepta a face 2 da região B, significa que estas são candidatas a terem elementos próximos. Iniciando-se assim o processo de análise da aproximação dos pontos.

A análise é feita seguindo os seguintes passos: (i) tomam-se os vértices da face 1 e se calcula sua distância em relação aos vértices da face 2; (ii) para os dois pontos de menor distância calculada, verifica-se se a distância entre eles é menor que a tolerância de entrada, como mostrado na Figura IV-13. Caso o seja, faz-se o ajuste desses pontos para um valor médio entre eles; (iii) em seguida, toma-se o outro par de pontos cuja distância é a menor obtida e se verifica se a distancia também é menor que a tolerância de entrada. Caso o seja, efetua-se o ajuste do ponto. Esse processo é repetido até que todos os pontos de uma face sejam analisados em relação à outra. O processo termina quando todas as faces forem analisadas.

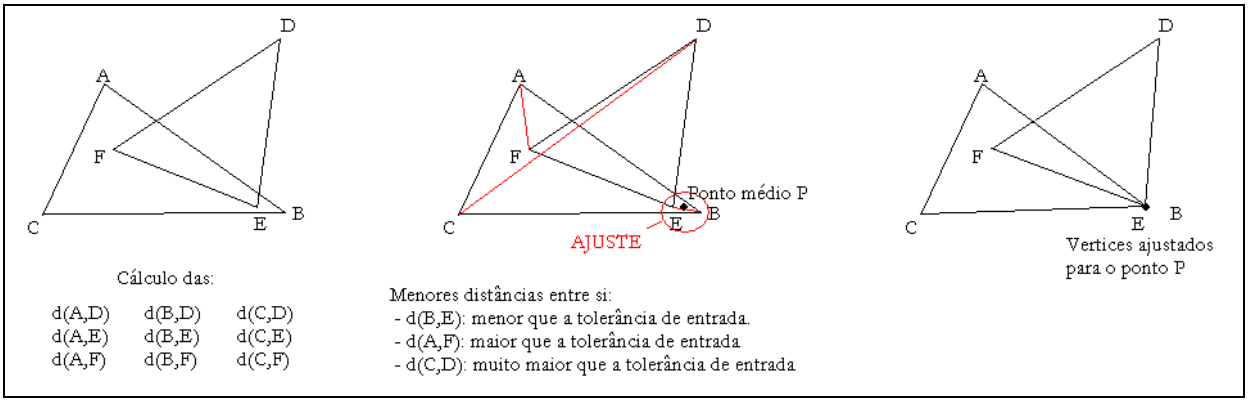


Figura IV-13 – Passos para determinação dos pontos próximos

- A FUNÇÃO JOINNEAREE

Esta tem como objetivo principal aproximar duas arestas de faces distintas, cada uma pertencente a uma região. Neste caso, são varridas as semi-arestas de cada uma das faces e, aquelas que estão próximas a menos da tolerância definida, são processadas e aproximadas entre si. O procedimento consiste em calcular o ponto em cada aresta cuja distância entre elas seja a menor possível, isto é , descobrir o menor segmento de reta, ou ponto de interseção que ligue as duas arestas, como indicado na Figura IV-14.

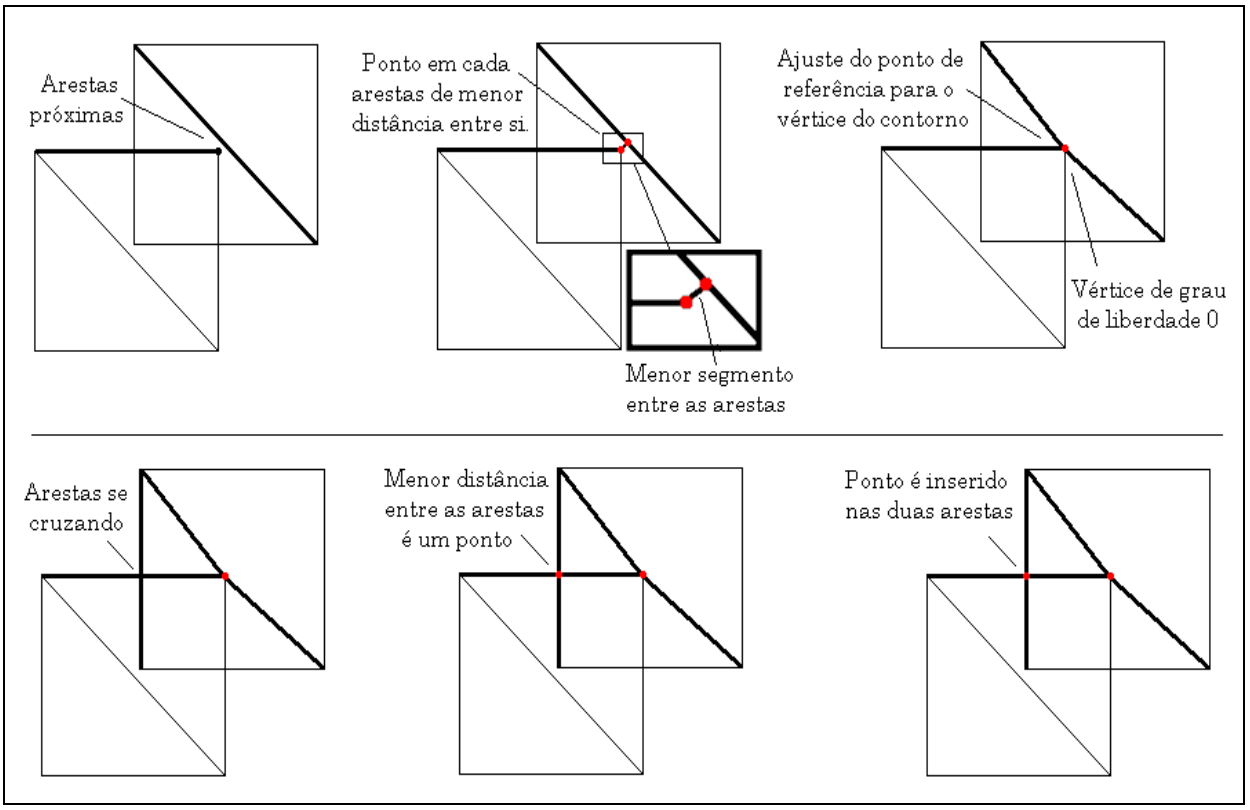


Figura IV-14 – Processo de aproximação das arestas

Se o tamanho encontrado for um segmento de reta cuja distância é menor que a tolerância, é criado um ponto de referência igual ao ponto médio do segmento de ligação. A partir daí, deve-se verificar a posição deste ponto de referência em cada aresta. As possibilidades são:

- O ponto de referência está próximo a um dos pontos limites da aresta. Então, é verificado o grau de liberdade do vértice, para confirmar se será ajustado para o

valor do ponto de referência. O ajuste só acontecerá se o grau de liberdade for maior que 1. Caso contrário, o ponto de referência passa a ter o valor do vértice.

- O ponto não está próximo a nenhum dos pontos limites da aresta. Neste caso, deve-se criar um ponto na aresta contendo as coordenadas do ponto de referência dividindo-a.

Para o caso em que o ponto de referência estiver próximo a pontos limites das duas semi-arestas a aproximação destes pontos limites será feita com base na coordenada do ponto médio entre os três envolvidos: os pontos limites de cada aresta e o ponto de referência. Vale-se frisar que esta análise é feita simultaneamente com as duas semi-arestas em questão sendo que, ao final, as arestas deverão ter um ponto em comum decorrente do ajuste. O objetivo desse método é fazer a aproximação dos elementos que apresentem umas das situações mostradas na Figura IV-15.

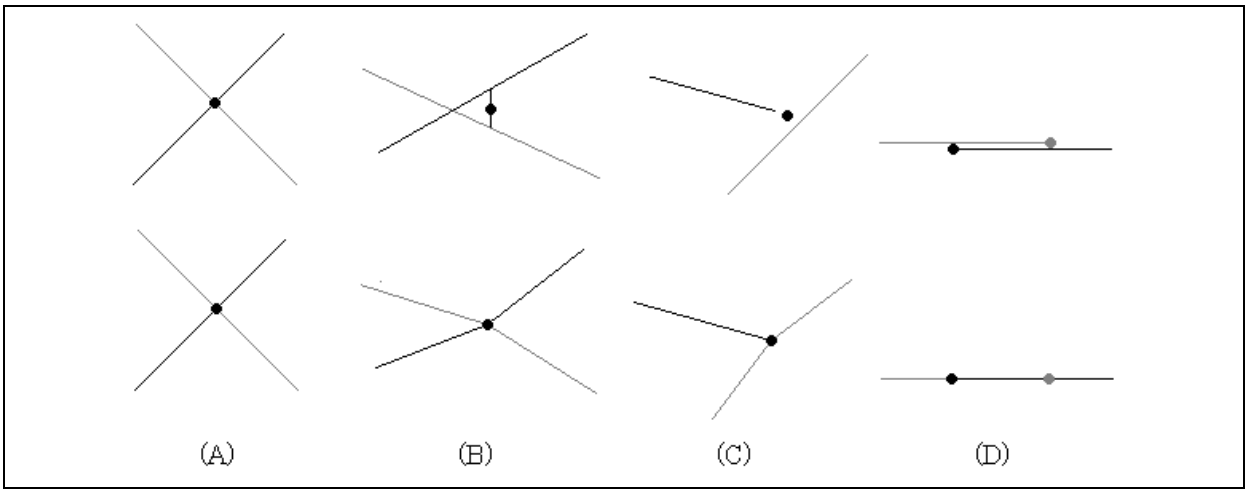


Figura IV-15 – Casos para aplicação da aproximação vértice-aresta.

- A FUNÇÃO VERIFYPOINTADJUST

Para otimizar o processo de inserção de um ponto em uma aresta, foi criada esta função que verifica se o ponto a ser inserido está próximo a qualquer um dos pontos limites da aresta a menos da tolerância passada como parâmetro. Os casos possíveis para análises são: (i) o ponto que deve ser inserido não está próximo a

nenhum dos pontos limites, sendo permitida a inserção; (ii) o ponto que deve ser inserido está próximo de um dos pontos limites que será ajustado caso o grau de liberdade deste ponto limite seja maior que 1. Se puder ser ajustado, o ponto limite recebe os valores de coordenadas do ponto que deveria ser inserido, senão, o ponto de referência será devolvido com o valor das coordenadas do ponto limite próximo. Esta função não insere o ponto na aresta, mas simplesmente devolve um valor *Booleano* sinalizando sua possibilidade de inserção.

IV.2.3.2 – AS FUNÇÕES DE INTERSEÇÃO ENTRE POLÍGONOS

Nesta parte do trabalho serão apresentadas as estratégias utilizadas na implementação das funções capazes de obter os pontos de interseção entre polígonos, processo este utilizado na construção de modelos através das operações booleanas e operação de montagem.

A estratégia utilizada para se realizar a interseção entre dois polígonos consiste em verificar a situação de cada aresta de um polígono em relação ao outro polígono, obtendo os pontos em comum. Para isto, foram implementadas funções que processam a interseção entre semi-reta e polígono, sejam estes coplanares ou não.

- *POLYGON :: INTERSECTIONCOPLANAR*

Esta função é uma sobrecarga da função de interseção coplanar entre reta e polígono, e processa a interseção entre segmento e polígono. A idéia básica desta função é percorrer todas arestas do polígono e verificar se há interseção com o segmento passado como parâmetro, computando-se os pontos de interseção. Após isto é verificado se algum dos pontos limites do segmento parâmetro está situado dentro do polígono, computando-o também caso a resposta seja positiva. Esta função devolve os pontos de interseção entre o segmento parâmetro e o polígono e, também, um conjunto de segmentos resultantes da interseção. A Figura IV-16 mostra o resultado devolvido pela função, os pontos e as arestas de interseção.

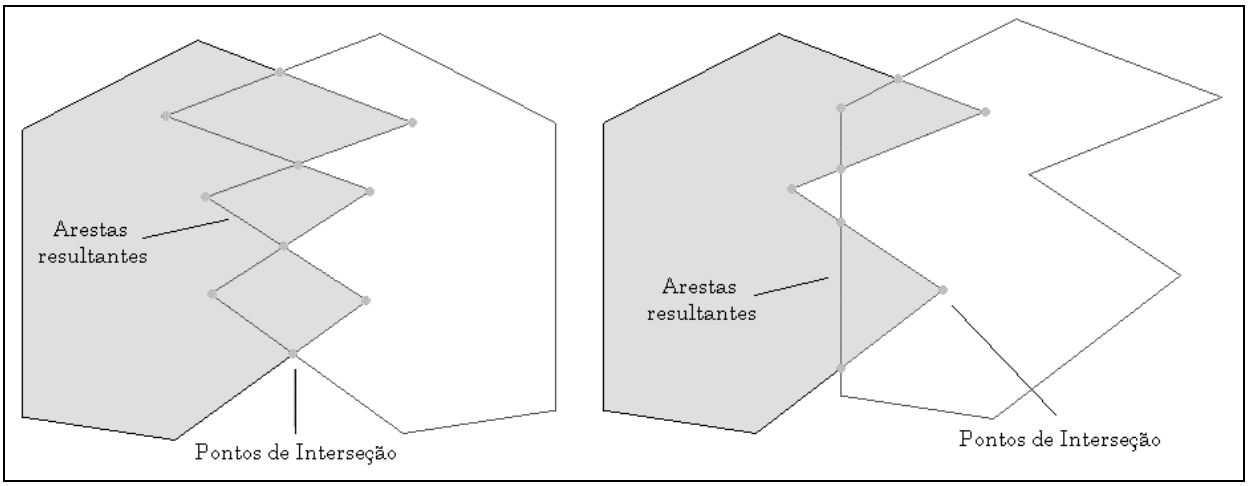


Figura IV-16 – Resultado obtido da interseção coplanar

Ao final, de posse dos pontos que interceptam o polígono, é feita a verificação dos segmentos resultantes da análise da interseção. Esta análise é feita checando se, em cada par de pontos da interseção, o ponto médio está dentro ou fora do polígono. Caso esteja dentro, esta é uma aresta obrigatória advinda da interseção, com seus limites sendo o par de pontos analisado. A Figura IV-17 mostra como é feita a classificação dos segmentos em ser ou não uma aresta de interseção.

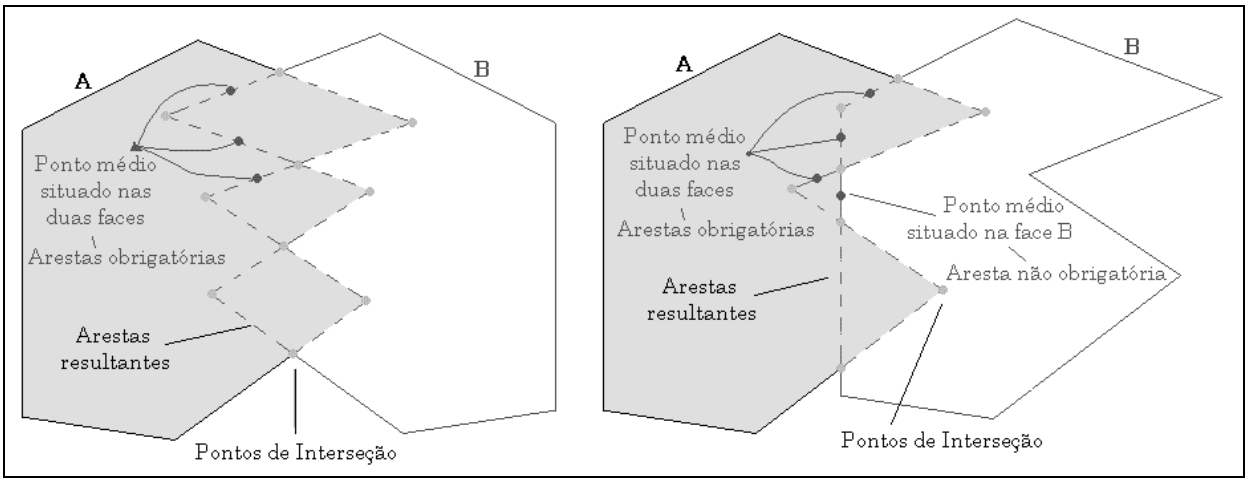


Figura IV-17 – Classificação do segmentos em aresta obrigatória ou não

Esta análise só é possível após fazer o alinhamento dos pontos de interseção feito através da função *Polygon::sortInLinePoints.* Esta função foi criada para que se pudesse, a partir de um vetor com pontos 3D alinhados, ordenar os pontos e colocá-los de volta no vetor. Desta forma, pontos em posições vizinhas do vetor seriam

geometricamente pontos vizinhos. Este processo facilitaria a análise das arestas obrigatórias da interseção, nos casos em que é preciso verificar se o ponto médio entre dois pontos de interseção vizinhos está dentro do polígono.

- *POLYGON :: INTERSECTION* PARA SEGMENTOS E POLÍGONO

Esta função complementa a análise de interseção entre segmento e polígono, incluindo o processamento quando estes elementos são não coplanares. Esta função verifica se o segmento parâmetro é coplanar ao polígono. Caso positivo, chama-se a função PolyGon :: intersectionCoplanar. Caso a resposta seja negativa, faz-se a verificação da existência de interseção entre o segmento e o plano contendo o polígono. Se o ponto de interseção estiver, ao mesmo tempo, dentro do polígono (ou sobre seus limites) e sobre o segmento, então ele é computado como ponto de interseção. Nota-se que para o caso não coplanar, não há segmento resultante da interseção.

- *POLYGON :: INTERSECTION* PARA DOIS POLÍGONOS

A estratégia básica desta função é percorrer cada aresta de um polígono, verificar se há interseção com o outro polígono e processar o resultado da interseção. Cada ponto de interseção deve ser classificado de acordo com sua posição no polígono: sobre um vértice, numa aresta ou sobre a face. Além disto, devem ser feitas análises sobre os segmentos resultantes da interseção, justificando o retorno do conjunto de segmentos em PolyGon :: intersectionCoplanar . Para cada aresta do polígono 1 é verificada a interseção em relação ao polígono 2 e computados os pontos e os segmentos resultantes. Depois, faz-se o processo inverso: arestas do polígono 2 em relação ao polígono 1. Paralelamente a isso, executa-se a análise de cada ponto de interseção, verificando se está sobre um vértice, aresta ou face nos dois polígonos. Ao final, de posse dos segmentos e pontos de interseção, processa-se a análise das arestas obrigatórias. Para isto, é verificado se os polígonos são coplanares, sendo que, para resposta positiva, analisa-se o conjunto de segmentos resultante de PolyGon :: intersectionCoplanar.

Caso os polígonos sejam não coplanares, o conjunto de pontos resultante da interseção é analisado de forma a verificar se algum par de pontos próximos forma uma aresta obrigatória, quer dizer, que pertença aos dois polígonos. Novamente, isto só é possível caso os pontos de interseção estejam alinhados. O formato de saída dos pontos de interseção é um vetor de *InterPoints* que guarda não só as coordenadas do ponto mas também informações sobre a posição nos polígonos verificados.

- A FUNÇÃO DE INTERSEÇÃO ENTRE REGIÕES

A função DoubleOp :: intersectionPoints no arquivo csgop.cpp foi originalmente escrita para suportar faces triangulares na análise da interseção entre regiões. Para que as faces poligonais pudessem ser processadas algumas mudanças simples foram implementadas. A partir de um ponteiro para uma face, advinda da lista de faces de cada casca, pode-se obter um lista das semi-arestas que compõem a face. Deste modo, é possível conseguir as coordenadas dos vértices que compõem a face e assim criar um objeto *PolyGon,* para ser processado pela rotina de interseção junto ao polígono conseguido de uma outra face. Com isso, garante-se que qualquer face que possua um número qualquer de vértices seja analisada para interseção.

Capítulo V – Resultados

No capítulo anterior, foram descritos os algoritmos e as técnicas utilizadas no processo de avaliação e ajuste da malha de um modelo para melhorar sua qualidade, bem como mudanças realizadas no modelador de forma a permitir que os algoritmos propostos pudessem ser implementados.

As modificações realizadas no modelador foram significativas. Uma nova lista tornou possível a realização de várias operações booleanas sucessivas sobre o mesmo modelo. Além disso, a substituição da interseção triângulo-triângulo por polígono-polígono permitiu que os pontos de interseção pudessem ser computados em modelos de faces poligonais planares. Isto permite que as operações booleanas possam ser aplicadas e seus resultados sejam objetos com malha triangular.

A Figura V-1 apresenta um modelo resultante da aplicação de sucessivas operações booleanas de união e diferença aplicadas sobre seis esferas e um cubo. Os pontos de interseção foram determinados a partir da análise de faces poligonais. Observando a área do retângulo marcada na figura, é possível notar que foram aplicados durante as operações, os algoritmos que ajustam a malha.

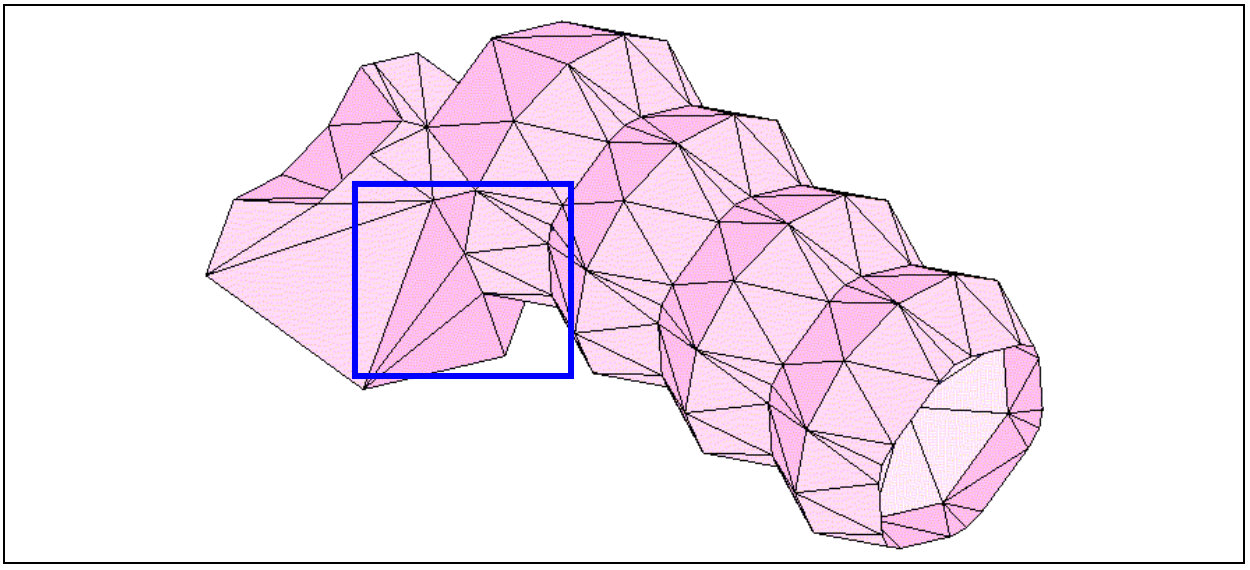


Figura V-1 – Modelo gerado de sucessivas Operações booleanas.

O processo de ajuste da malha utiliza uma metodologia que necessita de um grande esforço computacional, pois cada face de uma região é analisada em relação às faces da outra região quanto à proximidade de seus elementos. A análise de proximidade consiste em, inicialmente, comparar cada vértice de uma face com todos os vértices da outra. Em seguida, comparar os vértices de uma face com as arestas da outra e vice-versa. Finalmente, as arestas de uma face são comparadas com as arestas da outra face. Nesse processo, a tolerância utilizada na análise da aproximação dos elementos é calculada a partir do perímetro das faces envolvidas. Isso se deve ao fato do modelador permitir a criação de modelos com malha de densidades diferentes. Para malhas com densidades maiores (os elementos da malha são menores), a tolerância utilizada é menor e para as malhas de densidade menor (os elementos da malha são maiores), a tolerância utilizada é maior.

Neste capítulo serão apresentados os resultados obtidos pelos algoritmos responsáveis pelo ajuste da malha, pela aproximação dos pontos e pelo controle dos pontos da interseção, bem como os diferentes valores do fator de qualidade obtidos. Esses resultados são apresentados nas seções seguintes.

Na seção V.1, são mostradas as primitivas básicas que serão utilizadas como operandos nas operações booleanas e as malhas resultante da aplicação da interseção, união e diferença entre essas primitivas sem quaisquer dos ajustes da malha descritos nesse trabalho.

Na seção V.2, são apresentados os mesmos modelos da seção anterior, porém com a malha resultante da aplicação dos algoritmos de aproximação de dois pontos, aproximação do ponto à aresta e de aproximação de duas retas. Na seção V.3, é mostrado o resultado obtido aplicando-se a suavização laplaciana sobre os modelos da seção V.2.

Finalizando, na seção V.4, outros exemplos ilustram a melhoria que os algoritmos impõem à malha. Em todas as seções, à medida que os modelos são apresentados, é indicado o valor do fator de qualidade da malha superficial do modelo, com o propósito de comparar e comprovar a melhoria obtida nos ajustes realizados.

V.1 – Modelos com a Malha sem Ajuste

Nessa seção são mostrados alguns exemplos de malha resultante da aplicação das operações booleanas sem utilizar nenhum dos algoritmos de ajuste da malha descrito nesse trabalho. Cada exemplo é composto pelas primitivas a serem utilizadas nas operações booleanas, pelos modelos obtidos após a aplicação de cada uma das operações booleanas e pelo fator de qualidade da malha para cada um dos casos. Esses estão dispostos nas figuras da seguinte maneira: em (A), há os dois objetos envolvidos na operação booleana e o posicionamento entre eles; em (B), (C) e (D) é apresentada a malha dos modelos resultantes das operações de união, interseção e diferença, respectivamente, além de seu fator de qualidade.

Como primeiro exemplo, na Figura V-2 são apresentados dois cubos cujos lados medem 100 cm e cuja distância, tanto em x quanto em y, é de 53 cm. O cubo é um modelo no qual os vértices possuem grau de liberdade 0.

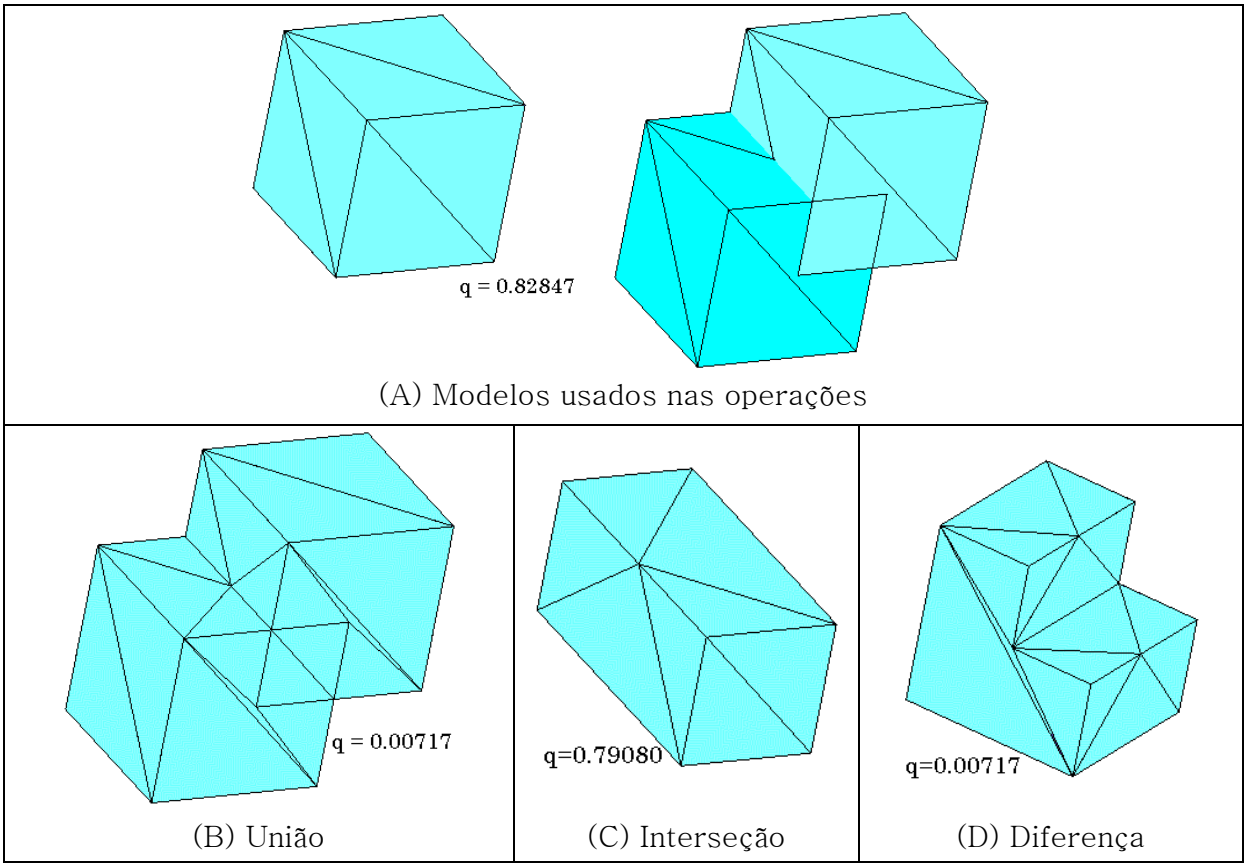


Figura V-2 – As Operações booleanas entre dois cubos

A má qualidade da malha resultante das operações booleanas de união e diferença se deve a presença de triângulos mal formados. Isso fica bem evidente no exemplo da Figura V-2, onde se tem vértice próximo a aresta e nos demais exemplos que serão mostrados a seguir.

A Figura V-3 traz como modelos para as operações, uma esfera de raio de 70 cm e centrada em (100,100,0) e um cubo de lado 100 cm, posicionado em (0,0,0). Nesse exemplo, os modelos apresentam graus de liberdade diferentes: a esfera apresenta grau de liberdade 2 para seus vértices – permite que seja movido na face – enquanto que o cubo possui vértices com grau de liberdade 0 – não permite que o vértice seja movido.

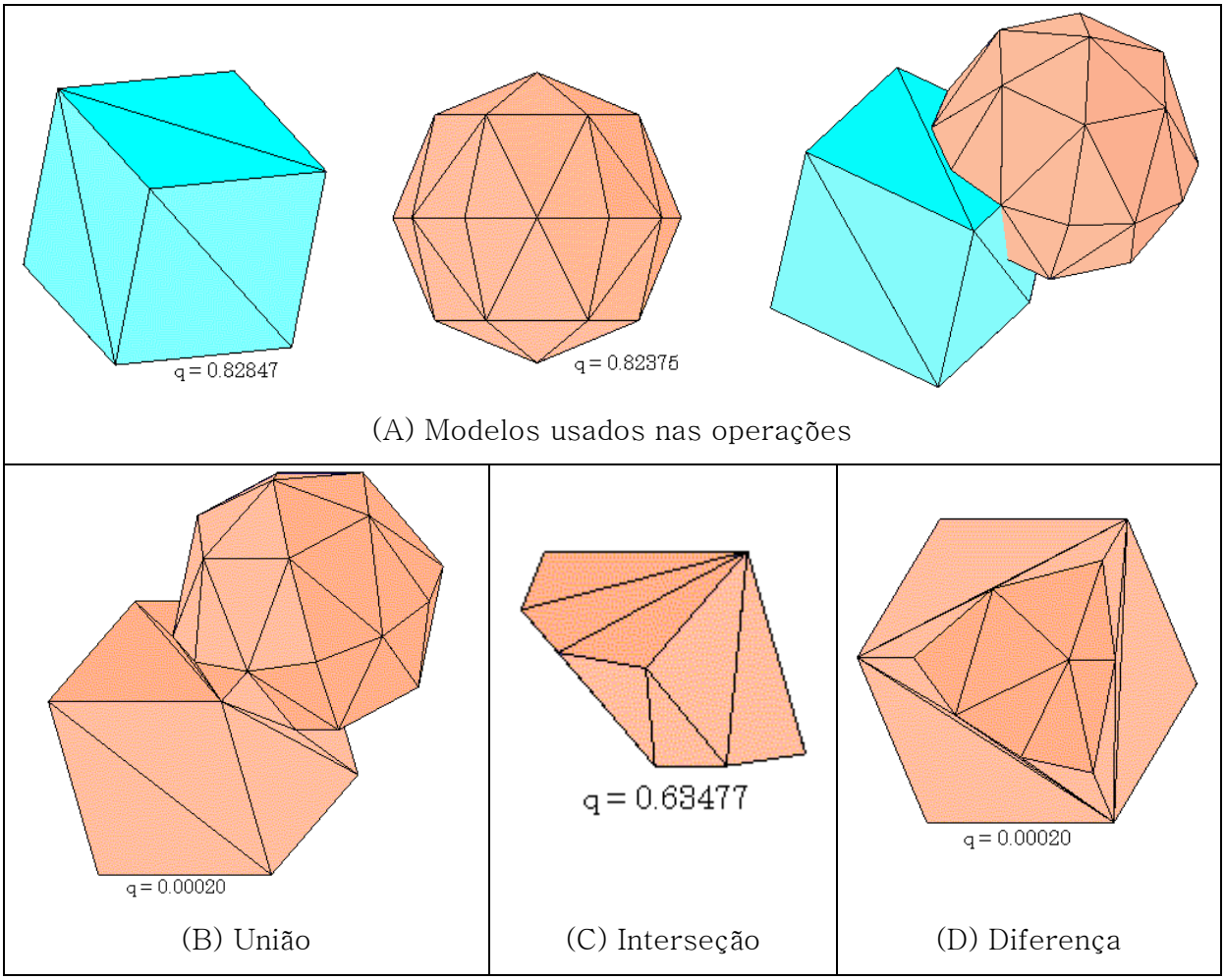


Figura V-3 – Operações booleanas entre cubo e esfera

Finalizando, o exemplo da Figura V-4 mostra a malha resultante das operações booleanas entre um cone de base centrada em (0,0,0), raio 100 cm e de 150 cm de altura e uma esfera centrada em (0,0,0) e raio de 70 cm. O cone é um objeto composto por vértices apresentando grau de liberdade 0, 1 e 2 (não podem ser movidos, ou podem ser movidos na aresta ou na face, respectivamente), enquanto as esferas são modelos em que todos os vértices têm grau de liberdade 2.

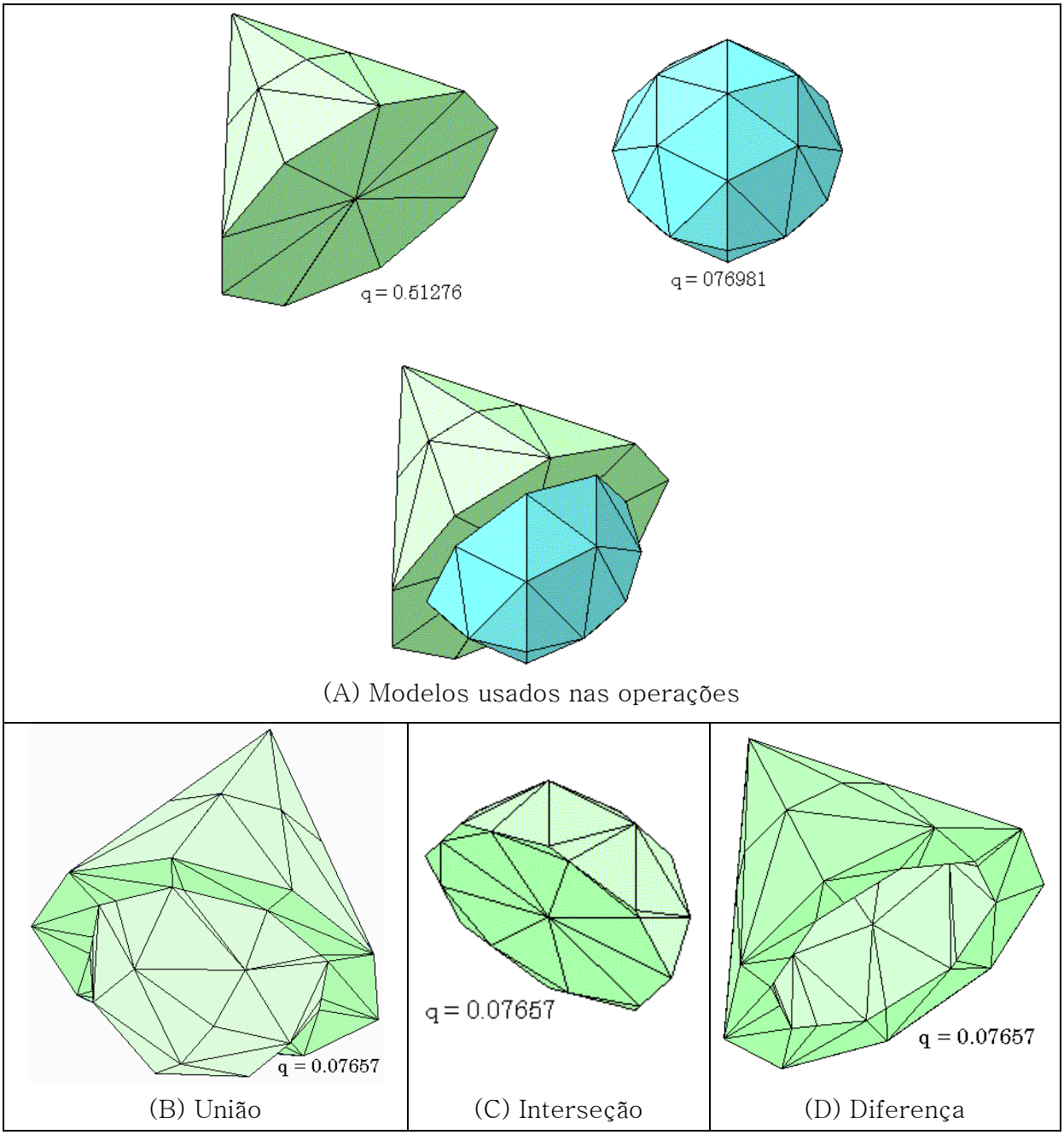


Figura V-4 – Operação Booleana entre duas esferas

V.2 – Modelos com a Malha Ajustada

Nessa seção serão apresentados alguns dos resultados obtidos aplicando os algoritmos de aproximação dos elementos da malha, bem como o que foi conseguido em termos da melhoria da qualidade da malha final sobre os modelos. Para facilitar a comparação, serão repetidos todos os casos apresentados na seção anterior, aplicando-se agora os algoritmos de aproximação.

A Figura V-5 mostra a malha resultante das operações booleanas entre dois cubos. Primeiro, são apresentadas as operações sem os algoritmos de ajuste da malha e, depois, as mesmas operações utilizando os algoritmos de ajuste.

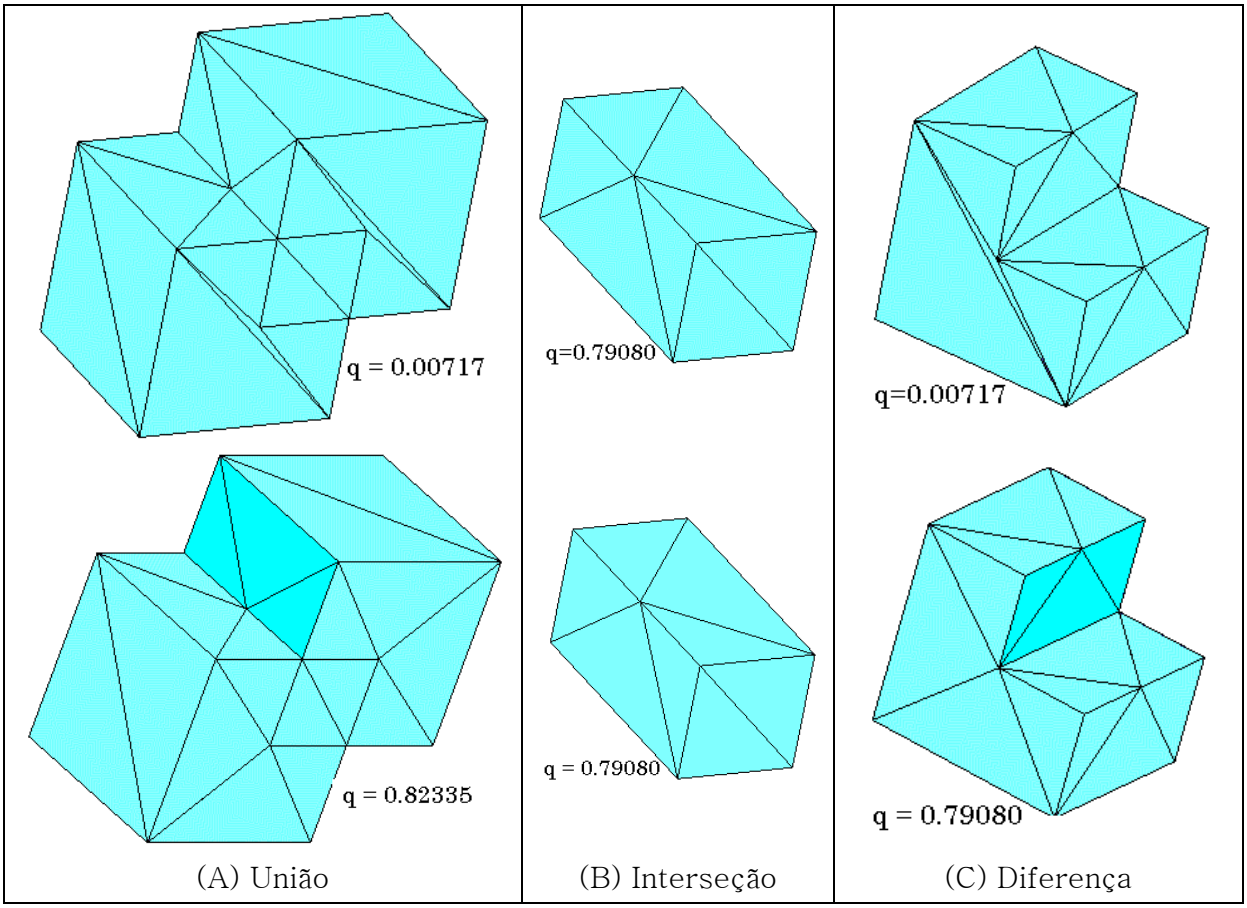


Figura V-5 - As operações booleanas de dois cubos

Nos casos já apresentados, é possível observar que os vértices do cubo não foram alterados devido ao seu grau de liberdade 0. Apenas, cada vértice foi inserido na aresta mais próxima a ele. É por esse motivo que o fator de qualidade da malha

na operação de interseção não sofreu alteração, e nas demais operações houve uma melhoria significativa, passando de q=0.00717 para q=0.82335 na operação de união e de q = 0.00717 para q=0.79080 na operação de diferença.

A Figura V-6 mostra a malha resultante das operações booleanas entre o cubo e a esfera, sendo apresentadas primeiro as operações sem e com os ajustes na malha.

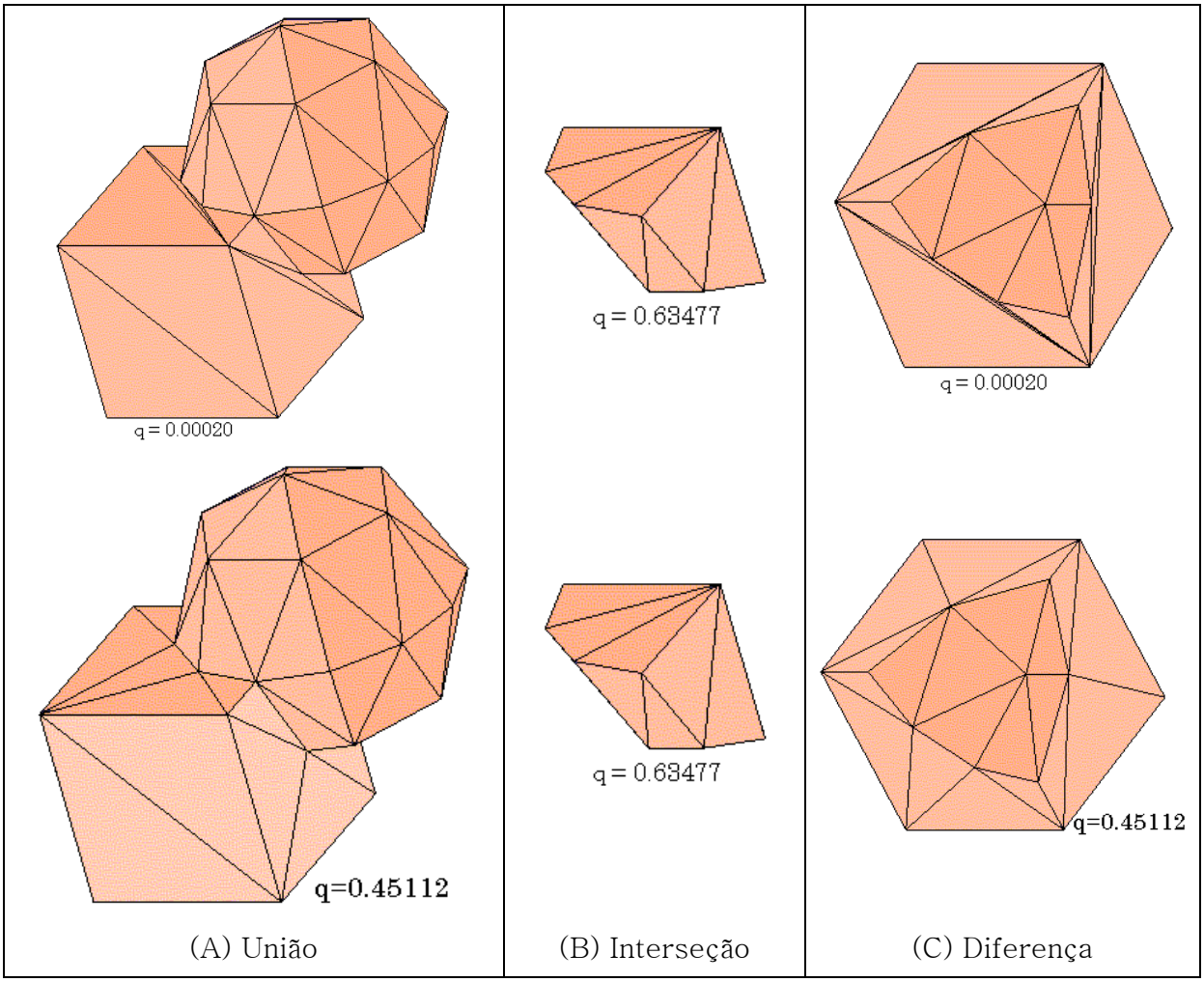


Figura V-6 - As Operações booleanas de um cubo com uma esfera

Finalizando, na Figura V-7, são mostrados os modelos obtidos das operações booleanas com o ajuste da malha aplicadas ao cone e à esfera. Este exemplo mostra a adequação da malha na região de interseção.

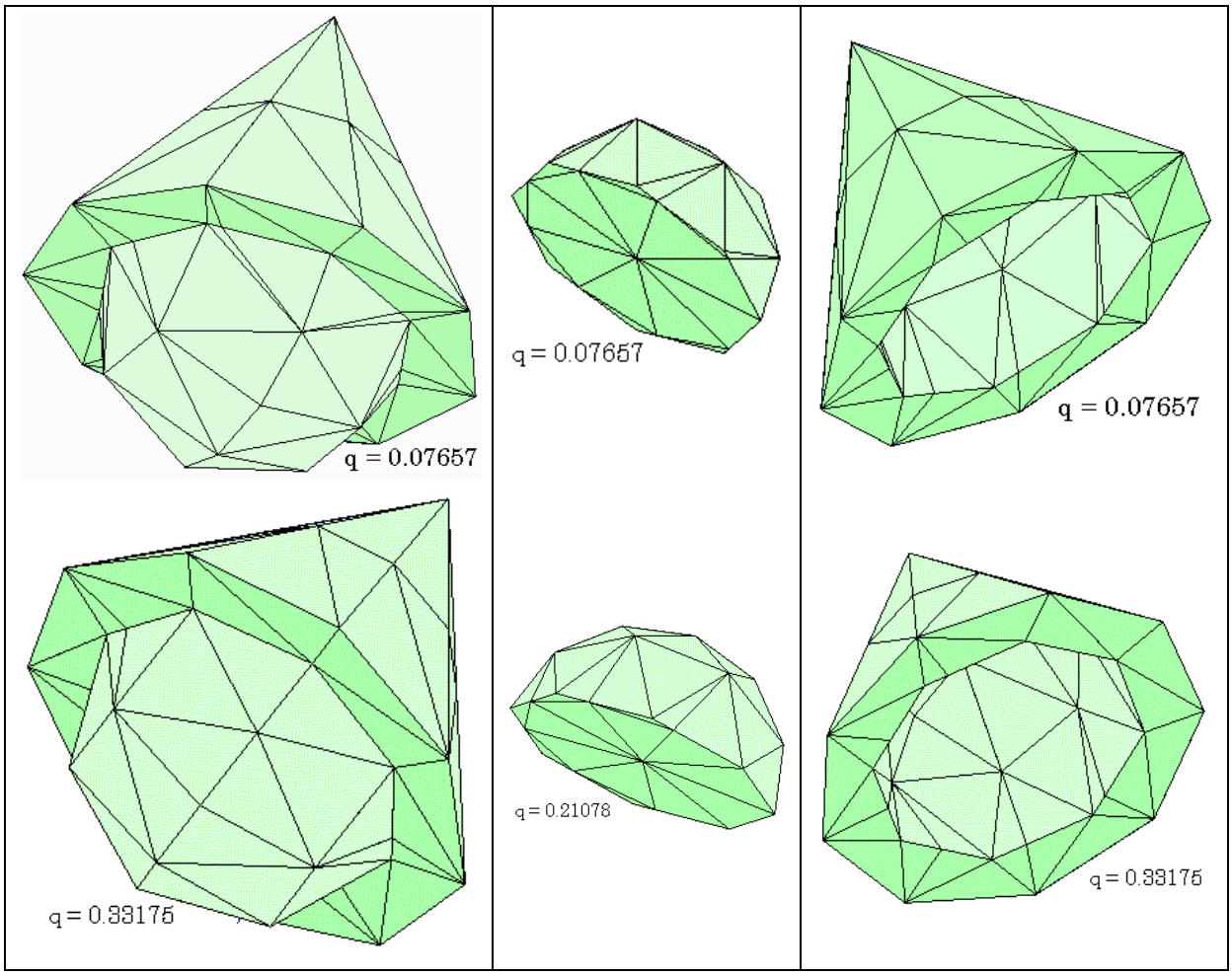


Figura V-7 - As Operações booleanas de cone e esfera

Devido à tolerância de entrada utilizada, em alguns pontos do modelo as formas dos triângulos não são próximas da ideal. Entretanto, os modelos aqui definidos servem de ponto de partida para geração da malha volumétrica, o que era impossível de ser realizado antes da agregação dos algoritmos para melhoria da qualidade da malha.

A Figura V-8 mostra a malha volumétrica obtida sobre os modelos com a malha ajustada. Esses modelos são os mesmos mencionados anteriormente. Observe, nos exemplos apresentados, que o Tetgen gera, quando necessário, uma malha volumétrica mais refinada se comparada à malha superficial fornecida.

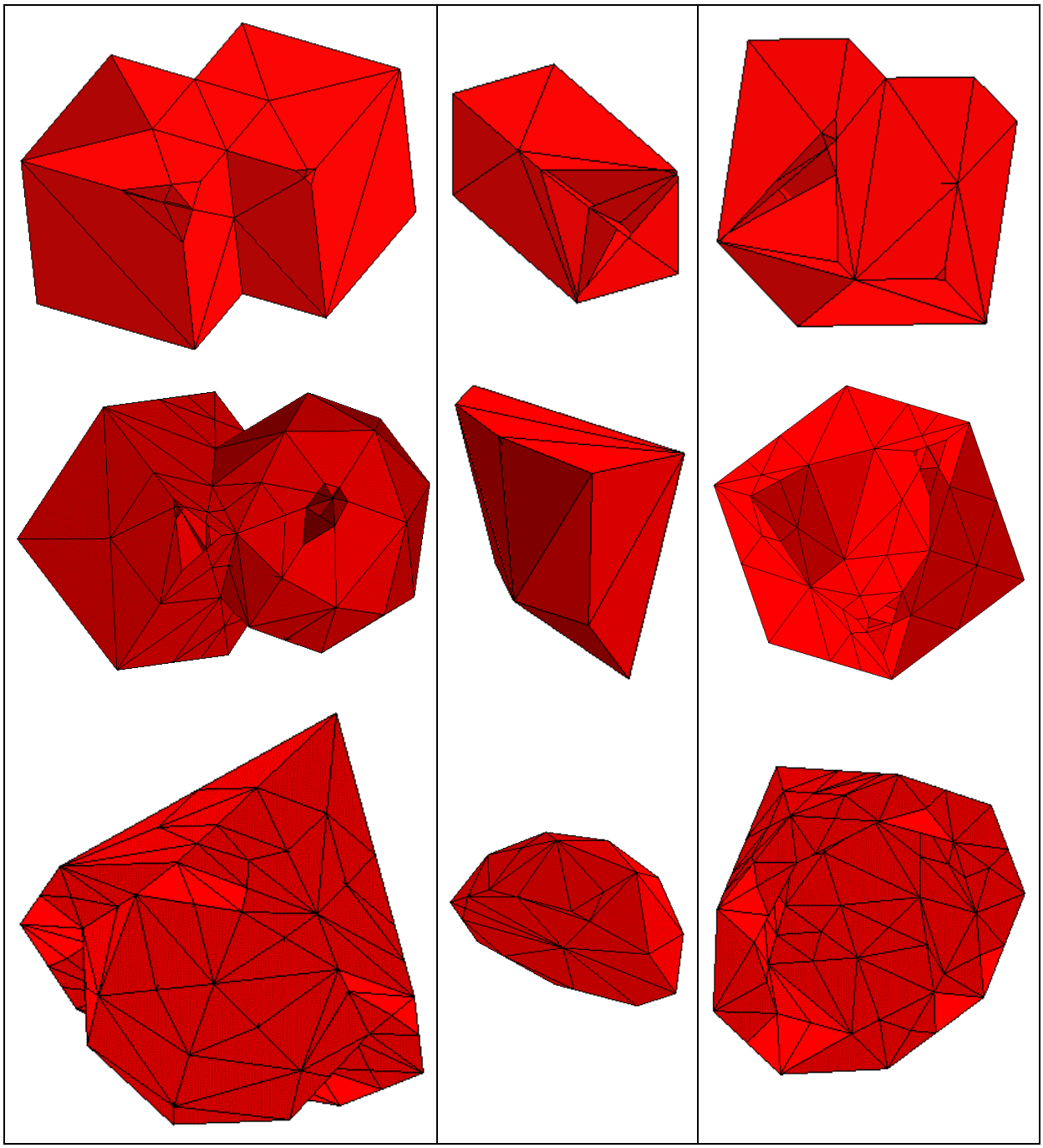


Figura V-8 – A Malha volumétrica sobre os modelos

V.3 – Exemplos com a Suavização da Malha

Um outro método já citado neste trabalho, que é muito empregado no refinamento de malhas superficiais planares, é a Suavização Laplaciana. Nessa

seção serão apresentados os resultados obtidos quando aplicado esse método, com os casos onde o método impõe uma melhoria significativa, pouca melhoria ou nenhuma melhoria na qualidade da malha final.

A Figura V-9 ilustra a aplicação do método sobre dois modelos obtidos da operação de união sobre dois cubos. Observe que o resultado obtido para cada caso é bem diferente. No primeiro caso, a malha apresentou pouca melhoria, enquanto no segundo caso essa melhoria na qualidade foi mais significativa. Entretanto, a malha apresenta um baixo valor no seu fator de qualidade. Para o último caso, a qualidade da malha piorou. Os triângulos em destaques são os responsáveis pelo fator de qualidade da malha.

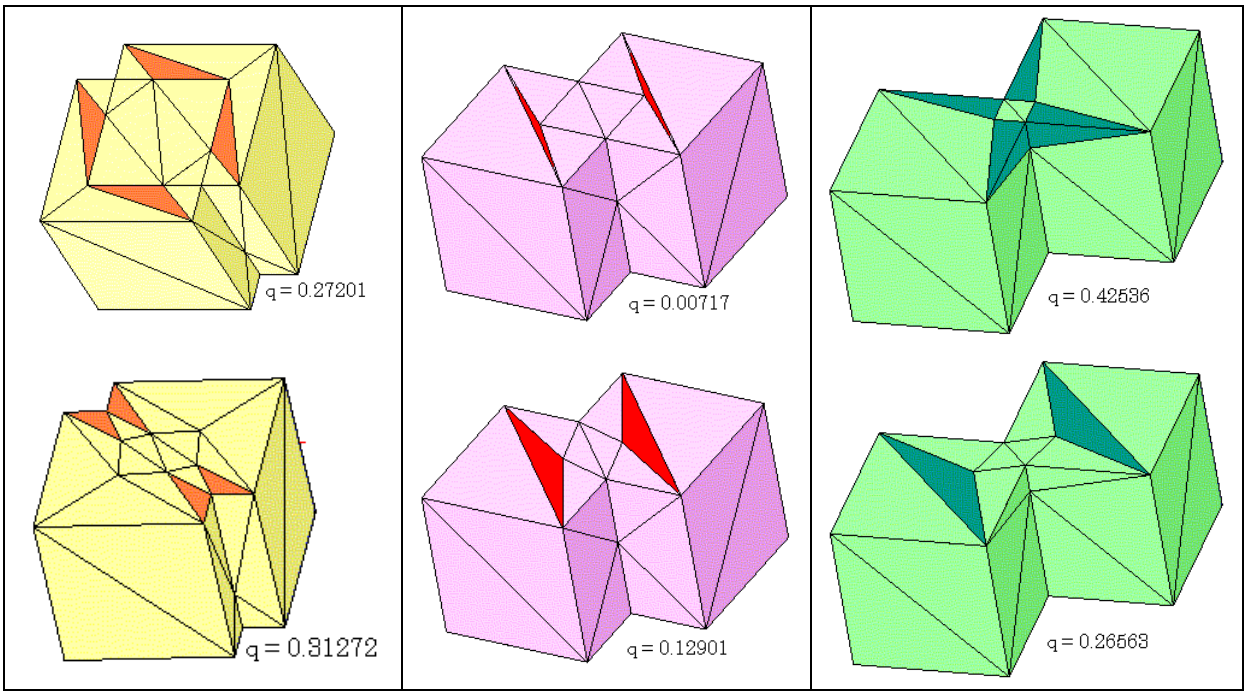


Figura V-9 – Suavização Laplaciana aplicada após a união de dois cubos

Na Figura V-10 são mostrados modelos onde a aplicação do método não causa alteração na qualidade da malha, devido a existência de aresta no contorno de dimensão muito pequena comparada com as demais arestas do próprio modelo. Esses modelos geralmente são resultantes da operação de diferença.

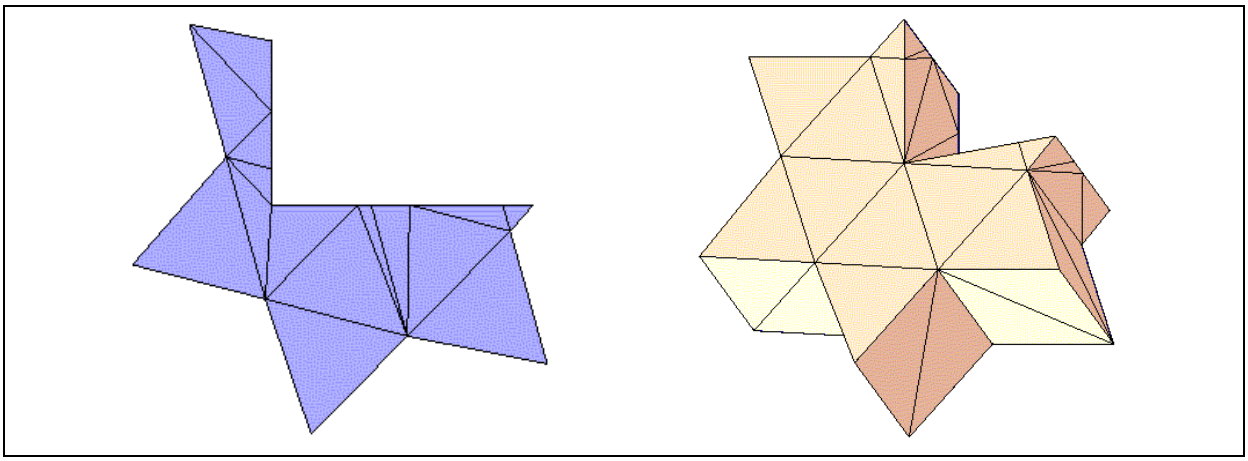


Figura V-10 – Suavização Laplaciana aplicada após a diferença de dois sólidos

V.4 – Outros Exemplos

Nessa seção serão apontados outros exemplos gerados pela aplicação de várias operações booleanas sucessivas. Primeiramente, são apresentados os objetos e a posição entre eles. Em seguida, o resultado da aplicação das operações.

Na Figura V-11 é apresentada a malha resultante de duas esferas que foram unidas. Observe que a malha apresentou pouco ajuste, pois a tolerância adotada foi menor que a distância entre as arestas do modelo que não se interceptam, mas que estão próximas umas das outras.

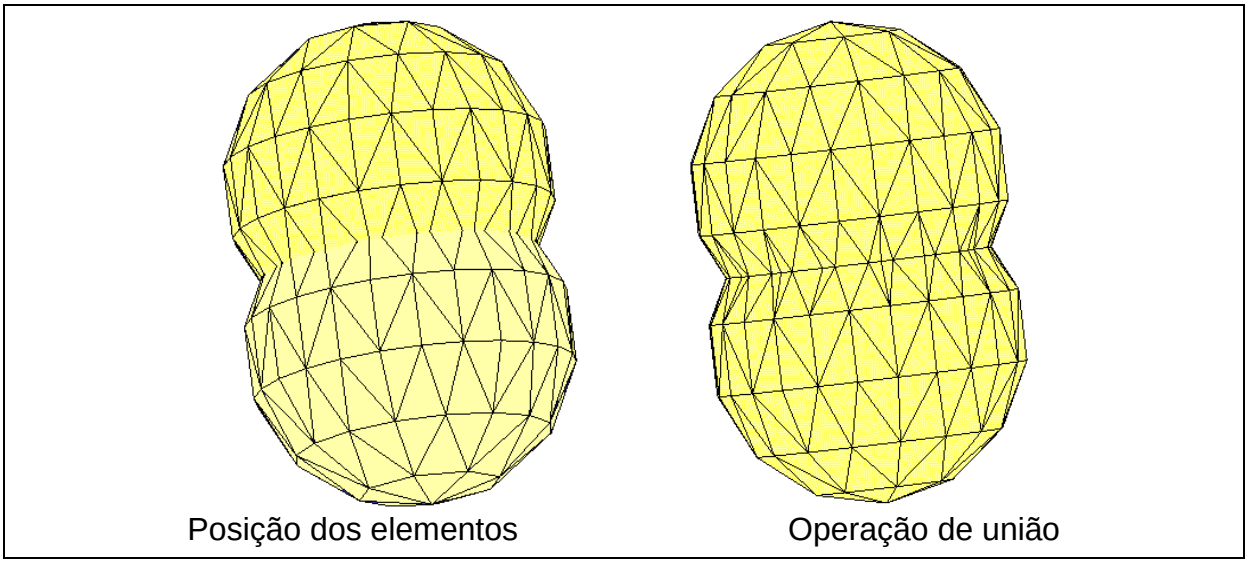


Figura V-11 – Duas Esferas unidas

A Figura V-12 apresenta o resultado obtido pela aplicação da operação de diferença sobre três esferas de 70cm de raio, sendo a primeira centrada em (0,-70,0), a segunda centrada em (0,0,0) e a terceira centrada em (0,70,0).

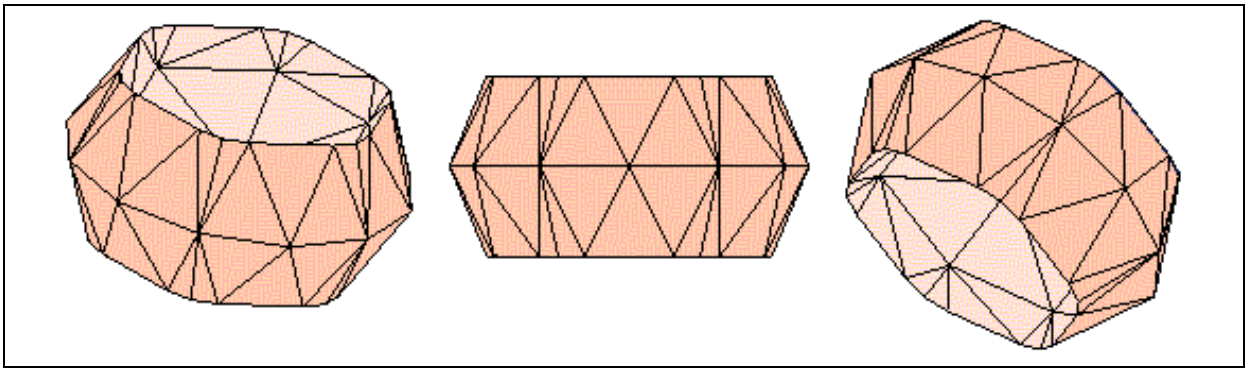


Figura V-12 – Subtração de três esferas

Aplicando-se a operação de união do objeto anterior a um cubo de 100 cm de lado, é obtem-se o modelo da Figura V-13.

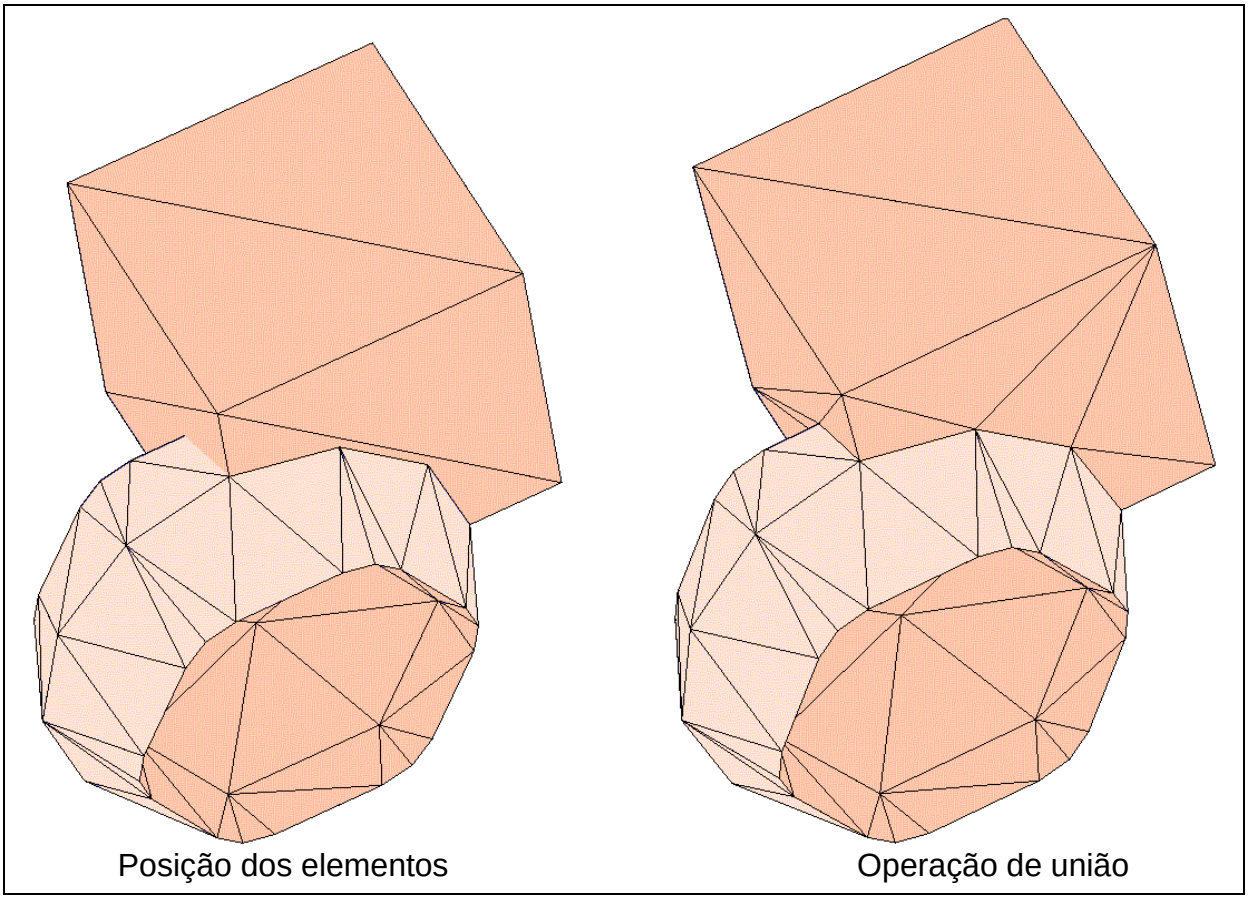


Figura V-13 – Modelo obtido de várias operações booleanas

A Figura V-14 mostra um conjunto de elementos posicionados para serem praticadas as operações booleanas de união entre os elementos. Para esse exemplo, foi alterado o cálculo da tolerância de forma que ela equivale à quinta parte da menor aresta que compõe as faces envolvidas na análise de aproximação. Observe, que elementos mal formados não aparecem no modelo. Entretanto nas áreas dos retângulos, surge uma pequena distorção provocada pela aproximação dos vértices da esfera em relação às arestas do cubo, o que é prejudicial para a representação do objeto inicial.

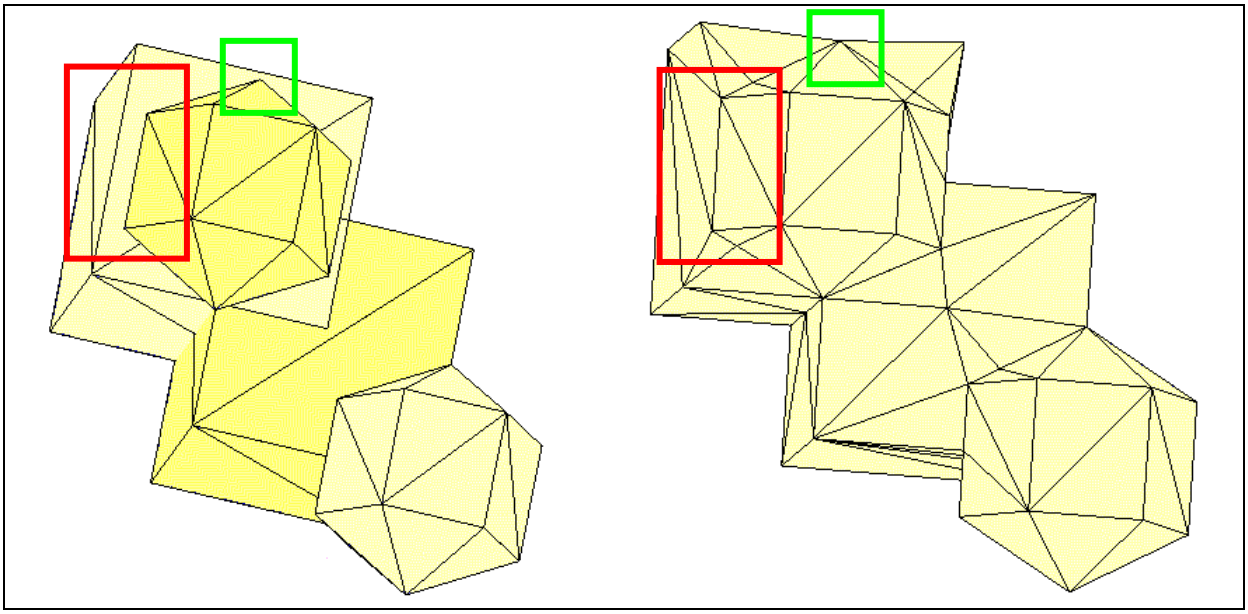


Figura V-14 – Modelo obtido de várias operações booleanas

Capítulo VI – Conclusão

Ao iniciar a realização desse trabalho, observou-se a necessidade de realizar algumas mudanças no modelador de maneira a corrigir erros, melhorar o seu desempenho e adequá-lo ao trabalho com polígonos quaisquer durante o processo de definição da região de intersecção, ao invés de processar apenas triângulos.

A realização das mudanças propostas alcançou o objetivo esperado: diminuiu a ocorrência de erros na lista ou gerados por problemas de precisão, além de ter melhorado significativamente o desempenho do modelador. Antes, apenas modelos com malhas triangulares eram computados pelas operações booleanas. Agora, modelos com faces poligonais podem servir como operandos para as operações booleanas. Entretanto, o resultado da operação ainda é um modelo triangulado, pois é mais adequado para a sua visualização. Entretanto, se o retorno de uma operação booleana fosse um modelo com faces poligonais seria mais adequado, se tratando da qualidade da malha. O Tetgen sabe tratar elementos poligonais, pois ele recebe como entrada um PLC e a partir dessas informações ele gera a malha volumétrica. Quanto menor a densidade da malha superficial passada para o gerador mais liberdade ele terá para geração da malha volumétrica e melhor qualidade a malha volumétrica apresentará.

Em seguida, realizou-se o estudo e a implementação dos algoritmos capazes de analisar e ajustar a malha superficial dos modelos resultantes das operações booleanas. Foi possível comprovar a melhoria que os algoritmos proporcionaram à malha dos modelos, por meio da verificação da forma de seus elementos. Analisar a forma dos elementos de uma malha, verificando o tamanho de seus ângulos, é uma maneira muito boa de determinar a qualidade da malha e aplicar métodos para garantir sua qualidade.

Em alguns casos, devido à tolerância adotada, a melhoria imposta à malha não foi tão significativa. Entretanto, o uso de uma tolerância maior melhoraria a malha gerada sobre esses modelos, mas poderia causar maior deformação do modelo se comparado aos objetos iniciais. No GSM, mesmo para os casos onde o modelo

resultante apresentou baixa qualidade na malha superficial, foi possível gerar a malha volumétrica. Entretanto, para tais casos, o Tetgen necessitou refinar a malha superficial inicial, de maneira a gerar uma malha no volume de melhor qualidade.

VI.1 – Proposta para Trabalhos Futuros

Devido à forma como o modelador está estruturado, alguns métodos empregados na melhoria da malha não puderam ser implementados. Alguns desses algoritmos, ou utilizam informações a respeito da curvatura da superfície original do modelo durante o ajuste, ou empregam métodos onde há a necessidade da representação geométrica e topográfica ser feita separadamente da representação da malha gerada sobre o modelo. Sendo assim, propõe-se para trabalhos futuros, o estudo e projeto de:

a) Uma estrutura de dados para armazenar as informações sobre a malha, associando-a às informações geométricas e topológicas do modelo.

b) Métodos que permitam fazer a aproximação da superfície implícita por superfícies parametrizadas [NUN04].

c) Métodos para ajustar a malha utilizando modificação e parametrização local da malha superficial.

Outra proposta é modificar o restante do processo das operações booleanas de maneira que estas possam ser aplicadas em modelos facetados por polígonos planares durante todo seu processo, e não apenas na determinação dos pontos de interseção, resultando em (i) um modelo que possa apresentar uma malha triangulada ou (ii) um modelo composto por facetas poligonais planares.

Outra sugestão é propor estratégias de ajuste de malha que utilizem informações sobre as curvas da superfície geradora do modelo e métodos que façam o refinamento da malha, aumentando sua densidade e melhorando sua forma. Dessa forma, nos casos em que há uma diferença muito discrepante na densidade da malha de ambos os objetos participantes da operação, o resultado

poderá ser refinado, gerando uma malha de melhor qualidade e que servirá de ponto de partida para se obter a malha volumétrica [NUN04].

Capítulo VII – Referências Bibliográficas

[ARB90] ARBAB. F.. **Set Models and boolean operations for solids and assemblies**. IEEE computer graphics and applications, Los Alamos, p.76-86, nov. 1990.

[BER00] BERG, M., KREVELD, M., OVERMARS, M., SCHWARZKOPF, O.. **Computational Geometry: Algorithms and Applications**, Second Edition, Springer Verlag, p.31-33, 2000.

[COE99] COELHO, L. C. G., GATTASS, M., FIGUEIREDO, L. H., **Intersecting and Trimming Parametric Meshes of Finite-Element Shells**, International journal for Numerical Methods in Engeneering, VOL.0(0), p. 1-100, 1999.

[HAN02] HANG, S. **Tetgen: Quality Tetrahedral Mesh Generator**, <http:// www.weboo.com/sh/tetgen.htm> site visitado em janeiro de 2002, e-mail: <sihang@weboo.com>, Universidade Zhejiang, China.

[HOF89] HOFFMANN, C. M. **Geometric and solid modeling**. San Mateo: Morgan Kaufmann, 1989. 338 p.

[KAM91] KAMEL, H. A., CHEN, L. **Integration of solid modeling and finite element generation**. Computer methods in applied mechanics and engineering, North-Holland, 89, p. 485-496, 1991.

[LIN83] LINDHOLM, A. D., **Automatic Triangular Mesh Generator on Surface of Polyedra**. IEEE transactions on Magnetics. p. 2539-2542, 1983.

[MAG94] MAGALHÃES, A. L. C. C**. Operadores de Euler na modelagem de sólidos por fronteira: conceito, aplicação, estudos de casos**. São Carlos: ICMSC-USP, 1994. 29 p. (Notas Técnicas do ICMSC).

[MAG99b] MAGALHÃES, A. L. C. C. Et al. **Um modelador de sólidos voltado para aplicações em eletromagnetismo**. In: CILAMCE – CONGRESSO IBERO LATINO AMERICANO DE MÉTODOS COMPUTACIONAIS EM ENGENHARIA, 20, 1999, São Paulo. Abstracts... p. 262, CD-ROM – publicação 184, 20 p.

[MAG99c] MAGALHÃES, A. L. C. C. **Tratamento de fronteiras internas em modeladores de sólidos**. CADTEC Mídia, Belo Horizonte, n.7, p. 3-4, dezembro de 1999.

[MAG00] MAGALHÃES, A. L. C. C., **Estudo, projeto e implementação de um modelador de sólidos voltado para aplicações em eletromagnetismo**. Belo Horizonte: PPGEE – UFMG, 2000. Tese de Doutorado do programa de Pós-Graduação em Engenharia Elétrica, Universidade Federal de Minas Gerais.

[MÄN88] MÄNTYLÄ, M. **An introduction to solid modeling**. Maryland: Computer Science, 1988. 401 p.

[MOR85] MORTENSON, M. E. **Geometric Modeling**. New York: John Wiley, 1985. 763 p.

[NUN02] NUNES, C. R. S, **Implementação das operações booleanas e de montagem num modelador de sólidos para aplicação em eletromagnetismo**. Belo Horizonte, PPGEE – UFMG, 2002. Dissertação de Mestrado do programa de Pós-Graduação em Engenharia Elétrica, Universidade Federal de Minas Gerais.

[NUN04] NUNES, C. R. S, **Estratégias para refinamento da Malha superficial em Modeladores de Sólidos**. Belo Horizonte, PPGEE – UFMG, 2004. Exame de qualificação de doutorado do programa de Pós-Graduação em Engenharia Elétrica, Universidade Federal de Minas Gerais.

[OLI91] OLIVEIRA, M. C. F. **Modelagem geométrica: uma introdução**. São Carlos: Instituto de Ciências Matemáticas de São Carlos – Universidade de São Paulo. Relatório Interno, 1991.

[RAM90] RAMIREZ J. A.. **Geração Automática de Malha para o Cálculo de Campos Eletrostáticos e Magnéticos Bidimensionais Através do Método de Elementos Finitos**. Belo Horizonte: PPGEE – UFMG, 1990. Dissertação de Mestrado do programa de Pós-Graduação em Engenharia Elétrica, Universidade Federal de Minas Gerais.

[REI02] REIS, H. L., MAGALHÃES, J. M., RUTHNER, M. P. - **Geração de Malhas Triangulares 2D.** <http://www.inf.puc-rio.br/~heloreis/> site da PUC do Rio de Janeiro acessado em 2002.

[REN97] RENKA, R. J., **Algorithm 772: STRIPACK: Delaunay Triangulation and Voronoi Diagram on the Surface of a Sphere**. ACM Transaction on Mathematical Software, University of North Texas, vol 23, No. 3, September 1997.

[REQ80] REQUICHA, A. G., **Representations for rigid solids: theory, methods and systems**. ACM computing surveys, New York, v. 12, n. 4, p. 437-464, dec. 1980.

[REQ83] REQUICHA, A. G., VOELCKER, H. B., **Solid modeling: current status and research directions.** IEEE computer graphics and applications, Los Alamitos, v. 3, p. 25-37, oct. 1983.

[REQ85] REQUICHA, A. G., VOELCKER, H. B. **Boolean operations in solid modeling: boundary evaluation and merging algorithms**. Proceedings of the IEEE, New York, v. 73, n. 1, p. 30-44, jan. 1985.

[ROC95] ROCHA, L. F. N., MESQUITA, R. C. **Uma base de dados de formato neutro para intercomunicação entre programas de cálculo de campos eletromagnéticos: versão 1.1**, In: CBMAg – CONGRESSO BRASILEIRO DE ELETROMAGNETISMO, 1, 1995, Florianópolis. Anais. , pp: 140-143.

[SAB93] SABONNADIÈRE, J. C., COULOMB, J. L., **Elementos finitos e CAE** - aplicações em engenharia elétrica. São Paulo: Aleph, 1993. 214 p.

[SHE96] SHEWCHUK, J. R. **Triangle: engineering a 2D quality mesh generator and Delaunay triangulator**. In: WORKSHOP ON APPLIED COMPUTATIONAL GEOMETRY, I, 1996, Philadelphia. Proceedings. ACM, 1996. p. 124-133. Available from http://www.cs.cmu.edu/afs/cs/project/quake/public/www/tripaper/triangle0.html; software available from http://www.cs.cmu.edu/~quake/triangle.html.

[STE98] Owen, S. J., **A Survey of Unstructured Mesh Generation Technology.** Pittisburg, Carngie Mellon University Department of Environmental Civil Engineering and Canonsburg, ANSYS Inc. PA. September, 1998. Site <http://www.andrew.cmu.edu>

[TIL80] TILOVE, R. B. **Set membership classification: a unified approach to geometric intersection problems**. IEEE transactions on computers, Los Alamitos, p. 874-883, oct. 1980.

[WHI02] WHITE, D. R., SAIGAL, S., **Improved Imprint and Merge for Conformal Meshing**. Sandia National Labs and Carnegie Mello University, Pittsburgh, PA, U.S.A., 2002.