WANG CONGLI

Representação de Curvas e Superfícies em Modeladores de Sólido B-Rep

Disertação apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do título de Mestre em Engenharia.

SÃO PAULO

2004

WANG CONGLI

Representação de Curvas e Superfícies em Modeladores de Sólido B-Rep

Disertação apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do título de Mestre em Engenharia.

Orientador: Prof. Dr. Marcos de Sales de Guerra Tsuzuki

SÃO PAULO

2004

Agradecimentos:

Agradeço à CAPES pelo suporte financeiro e ao Pof. Dr. Marcos de Sales Guerra Tsuzuki pelo a orientação.

CONTEÚDO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SÍMBOLOS

$B_i^n(t)$	São Bases de Bernstein
$Q(t)$	Representam Pontos Sobre Uma Curva Parâmétrica
P_I	Representam Pontos de Controle de Uma Curva
i	Número Inteiro
$S(u,v)$	Representam Pontos Sobre Uma Superfície Parâmétrica
u	Parâmetro Que Está na Faixa $0 \leq U \leq 1$
w	Parâmetro Que Está na Faixa $0 \leq w \leq 1$
j	Número Inteiro
$\mathbf{P}^{\mathrm{I}}$	Um Ponto de Controle de Uma Curva
$\mathbf{P}^{\mathrm{II}}$	Um Ponto de Controle de Uma Curva
$\mathbf{P}^{\mathrm{III}}$	Um Ponto de Controle de Uma Curva
$\mathbf{P}^{\mathrm{IV}}$	Um Ponto de Controle de Uma Curva
$Q_v(u,v)$	Vetor Tangente na Direção v Sobre o Ponto (u,v) de Uma Superfície
$Q_u(u,v)$	Vetor Tangente na Direção u Sobre o Ponto (u,v) de Uma Superfície
C^1	Continuidade de Derivada

RESUMO

Neste trabalho, propusemos uma nova estrutura de dados para representar sincronamente curvas e superfícies em modelagem de sólidos B-Rep. Esta nova estrutura baseia-se na estrutura unificada. Ela separa duas funções principais atribuídas à face: fechar o contorno do sólido topologicamente, e representar a forma geométrica do fechamento. Estas duas funções não são consistentes entre si, pois existem situações em que a face possui três ou cinco lados, e neste caso é necessário dividi-la para que todas as faces possuam apenas quatro lados. Este processo, geralmente, é feito de modo automático pelo sistema CAD. Neste caso, a estrutura de dados proposta separa as informações relativas às intenções do usuário dos elementos criados automaticamente pelo sistema CAD para satisfazer esta necessidade. Também foi definido um nível extra para suportar a definição de faces poligonais planas que aproximam a forma da geometria, e deste modo permitem que propriedades integrais sejam calculadas.

ABSTRACT

In this article a new data structure is defined based on non-manifold representation to support the representation of curves and surfaces in B-Rep solid models. This new data structure will support the synchronization between the geometry of the Geometric Model and an approximated polyhedral. An approximated polyhedral is obtained allowing the calculation of mass properties using several existent algorithms in the literature. And it is possible to reduce the approximated polyhedral for a minimum representation when necessary. The face, in commercial solid modelers, has two functions: represent the boundary of the solid and represent the geometrical shape of the contour. In this new structure, we will separate these two functions, one topological element will represent the boundary of the solid and another topological element will represent the geometrical shape as free form curves and surfaces. This mechanism facilitate the representation of irregular faces (faces with three, five or more sides). Since it is not common for a commercial CAD system to support the direct representation of a three-side surface, usually, those faces are subdivided in a number of four side faces. That fact is an inconsistency between topology and geometry, as the number of sides in a face is defined by topology and geometry usually "forces" a face to have four sides. In this article, we will propose two structures, the principal data structure contains the original solid model, and the auxiliary data structure represents an approximation of the solid model. In this mechanism, an automatic mode is defined that can determine the forms of the irregular faces separated to the mode that surfaces can be modified by users. To synchronize both data structures, we defined some algorithms maintaining the consistence of the data structure.

1 Introdução

Modelagem de sólidos e modelagem geométrica foram desenvolvidas separadamente como apontado por Farin [7]. É muito comum associar sólidos poliedrais para modelagem de sólidos, e curvas e superfícies para modelagem geométrica. Modelos poliedrais têm algumas desvantagens. Primeiro, a característica topológica significante do modelo é perdida – por exemplo, um buraco cilíndrico simples é representado por muitas faces e não por uma expressão matemática. Segundo, a geometria do modelo é aproximada – o nível de precisão geométrica depende do número de faces utilizado para aproximar cada uma das faces curvas.

As necessidades tecnológicas atuais requerem que ambas as tecnologias estejam presentes nos modeladores de sólidos. Entretanto, na literatura temos poucas propostas para integrar ambas as tecnologias. Toriya e Chiyokura [2, 3, 14] descrevem as propostas utilizadas no modelador de sólidos desenvolvido na Ricoh. Eles representam curvas como atributos de arestas e superfícies como atributos de faces. Entretanto, nada foi apresentado para integrar de forma síncrona as duas tecnologias. Mäntylä [11] descreve as propostas utilizadas no modelador de sólidos GWB desenvolvido na Universidade de Tecnologia de Helsinki. As propostas apresentadas são muito semelhantes às propostas apresentadas por Toriya e Chiyokura [14]. Novamente, nada foi apresentado para integrar de forma síncrona as duas tecnologias.

Turner [16], foi mais além, as curvas são representadas como atributos de arestas e as superfícies são representadas como atributos de faces. A sua proposta incluía um mecanismo para subdividir a superfície em um conjunto de faces planas. O mecanismo para subdividir a superfície em faces poligonais se utiliza de um conjunto de operadores básicos conhecidos como Operadores de Euler. Segundo Turner, este mecanismo é muito importante para gerar faces poligonais que aproximam as superfícies geométricas de um sólido. É possível considerar que temos dois sólidos: um sólido com as superfícies geométricas exatas, e outro sólido contendo faces poligonais que aproximam as

superfícies geométricas do sólido anterior. Conforme veremos as duas representações são necessárias.

Para alguns propósitos, a representação poliedral é necessária, por exemplo, para calcular propriedades de massa e visualizar os sólidos representados. Para se calcular as propriedades de massa, os modeladores B-Rep se utilizam de um algoritmo que varre todas as faces poligonais do sólido [15]. Este mesmo algoritmo não poderia ser utilizado diretamente sobre uma representação de curvas e superfícies. A representação de curvas e superfícies é necessária para ter acesso à representação exata da forma. Esta informação é útil para calcular o caminho de ferramentas de usinagem, dentre outras várias aplicações.

Ueda [17] estudou a proposta apresentada por Turner e propôs um mecanismo semelhante para determinar linhas poligonais que aproximam uma curva. Novamente, as curvas são representadas como atributos de arestas e as superfícies são representadas como atributos de faces.

Entretanto, Ueda [17] descobriu que o seu mecanismo para criar linhas poligonais que aproximam uma curva não poderia ser utilizado em conjunto com o mecanismo proposto por Turner [16] para subdividir uma superfície em faces poligonais aproximadoras. O principal efeito colateral é que as faces poligonais aproximadoras também definem uma aproximação para as curvas de contorno que delimitam a superficie que está sendo aproximada. Sendo que outro fator complicador é a manutenção dos atributos, indicando quais arestas aproximam quais curvas e quais faces aproximam quais superfícies.

Neste trabalho, é apresentada uma proposta que permite a convivência dos dois mecanismos aproximadores: aproximação de curvas por linhas poligonais proposto por Ueda [17], e aproximação de superfícies por faces poligonais proposto por Turner [16].

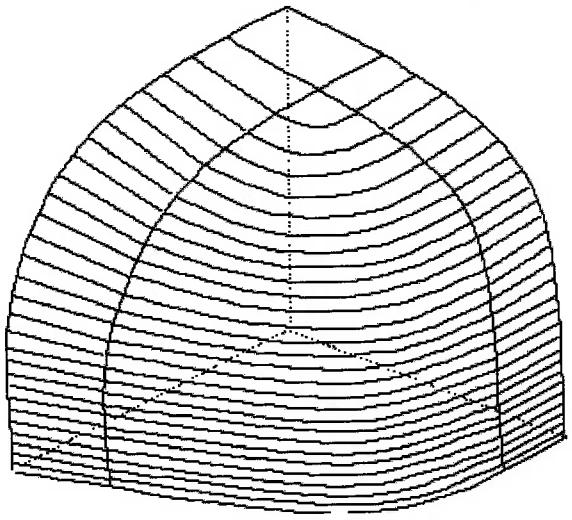


Figura 1. Suavização de sólidos.

1.1 Definição Automática de Superfícies

Quando utilizamos um software de CAD/CAM, imaginamos que temos o controle completo sobre a geometria da forma do sólido que está sendo editado. Entretanto, isto nem sempre é verdade. Para que expressões polinomiais sejam associadas a uma superfície é necessário que a superfície tenha quatro lados. Resta a seguinte pergunta: o que acontece com as superfícies que não possuem quatro lados? Existem propostas na literatura [2, 4, 5, 14] para subdividir uma superfície de 3, 5 e 6 lados em 3, 5 e 6 superfícies de quatro lados, respectivamente. Conforme veremos, estas superfícies que não possuem quatro lados são criadas com muita frequência nos modelos sólidos a partir de variadas operações geométricas. Sendo um caso típico a operação para suavizar arestas (vide Figura 1).

Neste trabalho, é apresentada uma representação que facilita a identificação, inserção e remoção de superfícies geradas automaticamente com a finalidade de associar a geometria à superfície. Na proposta de Turner [16], temos superfícies definidas pelo usuário, superfícies definidas automaticamente e faces poligonais aproximadoras em um único nível. Entretanto, neste trabalho associamos um nível hierárquico a cada um destes três tipos de superfícies. As superfícies definidas pelo usuário estão no nível mais elevado,

pois todas as informações são determinadas a partir dele. No nível intermediário temos as superfícies criadas automaticamente pelo sistema, para que a geometria possa ser associada. E finalmente, no nível mais baixo temos as faces poligonais que aproximam as superfícies que possuem geometria associada.

1.2 Necessidade da Associação Entre Curvas e Arestas

Neste trabalho, também propomos que a curva seja associada às meia-arestas e não às arestas. Para compreendermos melhor esta nova proposta é necessário que aprofundemos nossos conceitos sobre modelagem de sólidos e modelagem geométrica. Apesar de ser um fato simples, considerar a curva como um atributo de meia-aresta facilitará a manipulação das curvas e superfícies, principalmente quando existir arestas que terminam em T (vide Figura 2). Estas casos aparecem tipicamente na aplicação dos algoritmos que definem automaticamente a geometria de superfícies (por exemplo, algoritmo proposto por Chiyokura e Kimura [4]).

Este trabalho está estruturado da seguinte maneira. O Capítulo 2 apresenta uma introdução sobre curvas e superfícies. Bem como sobre o algoritmo para definir superfícies automaticamente proposto por Chiyokura e Kimura [4].

O Capítulo 3 apresenta conceitos básicos sobre modelagem de sólidos. O Capítulo 4

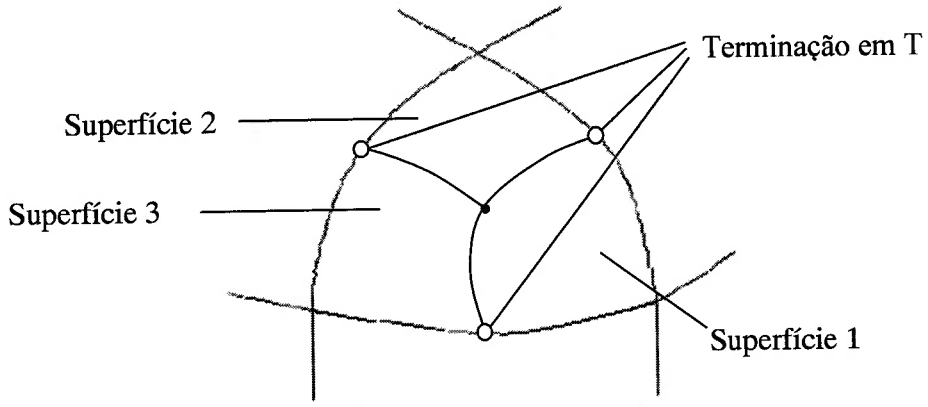


Figura 2 Superfícies em posição T.

apresenta em maior profundidade os métodos propostos por Turner [16] e Ueda [17] para realizar a integração entre modelagem de sólidos e modelagem geométrica.

Com todos estes conceitos presentes, apresentamos no Capítulo 5 a nossa proposta para realizar a integração entre modelagem de sólidos e modelagem geométrica. Os métodos proposto por Turner [16] e Ueda [17] podem coexistir. Também identificamos e manipulamos com maior facilidade, quando comparado aos métodos anteriores, as superfícies definidas pelo usuário, as superfícies que são definidas automaticamente e as faces poligonais de aproximação. No Capítulo 6 apresentamos algumas conclusões.

2 Curvas e Superfícies

Neste capítulo, apresentaremos uma breve introdução sobre modelagem geométrica. Esta introdução versa sobre a representação de curvas e superfícies. Apresentaremos também o algoritmo de subdivisão de superfícies com 3 e 5 lados proposto por Chiyokura e Kimura [4].

2.1 Modelagem de Curvas

Nesta sessão apresentaremos as curvas de Hermite e Bézier. Apresentaremos também o algoritmo para subdivisão de curvas de Bézier. De modo simplicado podemos dizer que a curva de Hermite é definida pelo seu contorno (ponto inicial, final e vetor tangente sobre estes pontos) e a curva de Bézier é definida pelo seu interior. Entretanto, é possível converter uma representação para outra, permitindo o uso de uma ou outra característica que melhor se adeque a solução de um dado problema. Apresentaremos também as condições que garantem continuidade de derivada entre curvas de Bézier.

2.1.1 Curvas de Hermite

A curva de Hermite pode ser escrita segundo a equação matemática abaixo:

$$Q(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \cdot \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} P_i \\ P_{i+1} \\ D_i \\ D_{i+1} \end{bmatrix} \quad (1)$$

Onde P_i e P_{i+1} são os pontos que definem os extremos da curva e D_i e D_{i+1} são os vetores tangentes sobre os extremos da curva. A Figura 3 exibe o exemplo de uma curva de Hermite.

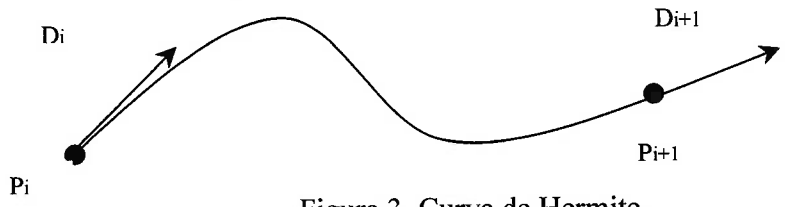


Figura 3. Curva de Hermite.

2.1.2 Curvas de Bézier

As curvas de Bézier [1] são curvas paramétricas. Ela é representada por um polígono de controle definido pelos pontos Pi. Exemplos de curvas de Bézier são exibidos na Figura 4. A curva de Bézier Q(t) de grau n é criada com n+1 pontos de controle e sua representação matemática é dada por:

$$Q(t) = \sum_{i=0}^{n} B_i^n(t) P_i \qquad (0 \le t \le 1) \qquad (2)$$

onde $B_i^n(t)$ são bases de Bernstein, dadas por

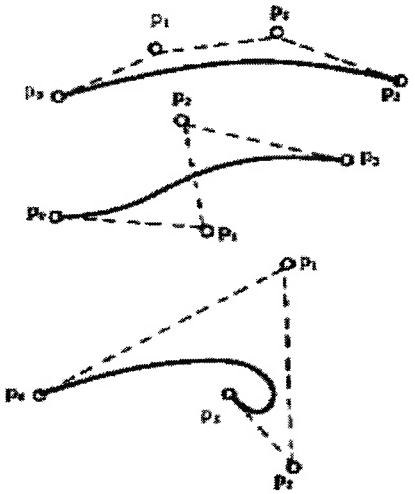


Figura 4. Exemplos de curva de Bézier.

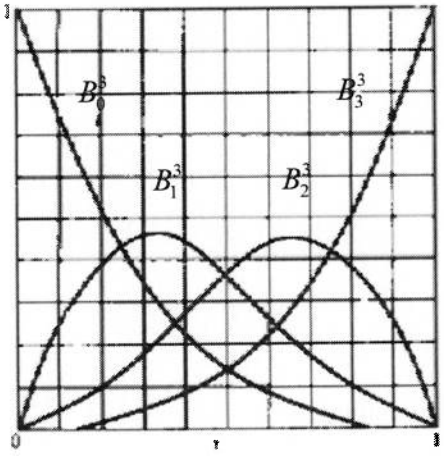


Figura 5. Bases de Bernstein para n=3.

$$B_i^n(t) = \binom{n}{i} t^i (1-t)^{n-1} \qquad (3)$$

e

$$\binom{n}{i} = \frac{n!}{(n-i)!i!} \qquad (4)$$

para n=3, temos

$$B_0^3(t) = (1-t)^3,$$
$$B_1^3(t) = 3t(1-t)^2,$$
$$B_2^3(t) = 3t^2(1-t),$$
$$B_3^3(t) = t^3$$

A Figura 5 mostra o gráfico das bases de Bernstein para n=3. A curva de Bézier passa pelos pontos de início e fim do polígono de controle. E se um ponto de controle arbitrário é movimentado, a forma do polígono de controle muda e assim a forma da curva

também se modifica. A derivada de primeira ordem da curva de Bézier de grau n é obtida derivando-se a equação (**2**):

$$\frac{dQ(t)}{dt} = n\sum_{i=0}^{n-1} B_i^{n-1}(t)a_i \quad (0 \le t \le 1) \qquad (5)$$

Onde

$$a_i = P_{i+1} - P_i \quad (i = 0,...,n-1) \qquad (6)$$

Então, baseando-se na equação (**5**), as derivadas de primeira ordem em cada extremo da curva de Bézier são dadas por:

$$\frac{dQ(0)}{dt} = n(P_1 - P_0), \text{ e } \frac{dQ(1)}{dt} = n(P_n - P_{n-1}) \qquad (7)$$

2.1.3 Conversão entre Curvas de Bézier e Curvas de Hermite

É possivel determinar uma fórmula de conversão entre curvas de Bézier de terceiro grau e curvas de Hermite também de terceiro grau. A curva de Bézier cúbica pode ser escrita em uma forma matricial pela expansão dos coeficientes dos polinômios de Bernstein presentes na definição analítica, e escrevendo estes coeficientes em uma forma matricial utilizando um polinômio em base de potências:

$$P(u) = \sum_{i=0}^{3} V_i B_i^3(u)$$

$$= (1-u)^3 \cdot V_0 + 3 \cdot u \cdot (1-u)^2 \cdot V_1 + 3 \cdot u^2 \cdot (1-u) \cdot V_2 + u^3 \cdot V_3$$

$$= \begin{bmatrix} (1-u)^3 & 3 \cdot u \cdot (1-u)^2 & 3 \cdot u^2 \cdot (1-u) & u^3 \end{bmatrix} \cdot \begin{bmatrix} V_0 \\ V_1 \\ V_2 \\ V_3 \end{bmatrix} \qquad (8)$$

$$= \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \cdot \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} V_0 \\ V_1 \\ V_2 \\ V_3 \end{bmatrix}$$

Para converter, devemos igualar as expressões **(1)** e **(8)**:

$$\begin{bmatrix} u^3 \\ u^2 \\ u \\ 1 \end{bmatrix}^T \cdot \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} V_0 \\ V_1 \\ V_2 \\ V_3 \end{bmatrix} = \begin{bmatrix} u^3 \\ u^2 \\ u \\ 1 \end{bmatrix}^T \cdot \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} P_i \\ P_{i+1} \\ D_i \\ D_{i+1} \end{bmatrix} \qquad (9)$$

Eliminando os vetores com os parâmetros u é possível em seguida isolar os pontos $V_0, V_1, V_2 \, e \, V_3$ em função de $P_i, P_{i+1}, D_i \, \text{e} \, D_{i+1}$:

$$\begin{bmatrix} V_0 \\ V_1 \\ V_2 \\ V_3 \end{bmatrix} = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}^{-1} \cdot \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} P_i \\ P_{i+1} \\ D_i \\ D_{i+1} \end{bmatrix} \qquad (10)$$

Assim, obtemos:

$$
\begin{aligned}
V_0 &= P_i \\
V_1 &= P_i + 1/3 \cdot D_i \\
V_2 &= P_{i+1} - 1/3 \cdot D_{i+1} \\
V_3 &= P_{i+1}
\end{aligned}
\qquad (11)
$$

2.1.4 Divisão de Curvas de Bézier

Como apresentado acima, algumas curvas de contorno precisam ser divididas. A expresão abaixo define uma forma recursiva para a curva de Bézier:

$$P_i^r(t) = (1-t)P_i^{r-1}(t) + tP_{i+1}^{r-1}(t) \qquad (12)$$

onde r e i variam como

r=1,...,n

i=0,...,n-r

e $P_i^0(t) = P_i$ representam os pontos de controle originais da curva. O vetor posição $P_0^n(t)$ definido pela equação (**12**) é um ponto especificado pelo parâmetro t da curva de Bézier. Considerando o caso da curva de Bézier cúbica, a equação **(12)** pode ser expandida para:

$$
\begin{aligned}
P_0^1(t) &= (1-t)P_0^0(t) + tP_1^0(t) \\
P_1^1(t) &= (1-t)P_1^0(t) + tP_2^0(t) \\
P_2^1(t) &= (1-t)P_2^0(t) + tP_3^0(t) \\
P_0^2(t) &= (1-t)P_0^1(t) + tP_1^1(t) \\
P_1^2(t) &= (1-t)P_1^1(t) + tP_2^1(t) \\
P_0^3(t) &= (1-t)P_0^2(t) + tP_1^2(t)
\end{aligned}
\qquad (13)
$$

Na última equação, $P_0^3(t)$ representa o ponto com parâmetro t sobre a curva de Bézier. Estas equações estão ilustradas na Figura 6. Quando a curva é dividida segundo o parâmetro t, os pontos de controle das duas curvas de Bézer cúbicas criadas são:

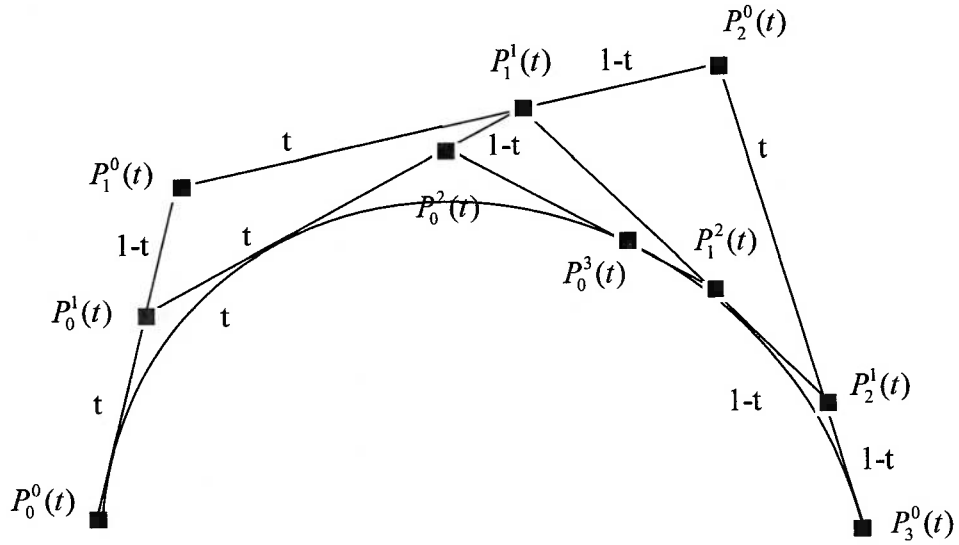


Figura 6. Divisão de curvas de Bézier.

$$P_0^0(t), P_0^1(t), P_0^2(t), P_0^3(t)$$
$$e$$
$$P_3^0(t), P_2^1(t), P_1^2(t), P_0^3(t)$$

Os pontos de controle das duas curvas criadas pela divisão podem ser obtidos facilmente pela repetição de interpolações lineares. O mesmo método pode ser aplicado a qualquer curva de Bézier de grau n.

2.1.5 Conexão Entre Curvas de Bézier

A partir da expressão (1) é possível determinar o valor da primeira derivada $d/du\,\mathbf{P}(u)$ ao longo da curva de Bézier:

$$\frac{d}{du}\mathbf{P}(u) = \begin{bmatrix} 3\cdot u^2 & 2\cdot u & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{bmatrix} \qquad (14)$$

Algo que ocorre com certa freqüência em sistemas de CAD, é a necessidade de conectar duas curvas de Bézier de modo que exista continuidade C^1 no ponto de conexão. Portanto, considere a curva de Bézier abaixo:

$$\mathbf{P}^{\mathbf{I}}(u) = \sum_{i=0}^{3} B_i^3(u) \cdot \mathbf{P}_i^{\mathbf{I}} \qquad (15)$$

E, deseja-se conectar esta curva de Bézier com a curva de Bézier fornecida abaixo:

$$\mathbf{P}^{\mathbf{II}}(u) = \sum_{i=0}^{3} B_i^3(u) \cdot \mathbf{P}_i^{\mathbf{II}} \qquad (16)$$

Assim, no ponto de conexão, pela continuidade de posição, tem-se que $\mathbf{P}_0^{\mathbf{II}} = \mathbf{P}_3^{\mathbf{I}}$. A Figura 7 ilustra a conexão entre duas curvas de Bézier de grau 3. A primeira derivada de uma curva de Bézier é dada pela expressão (14), igualando-se as derivadas nos pontos de conexão entre as duas curvas, obtém-se [14]:

$$\alpha \cdot 3 \cdot (\mathbf{P}_1^{\mathbf{II}} - \mathbf{P}_0^{\mathbf{II}}) = 3 \cdot (\mathbf{P}_3^{\mathbf{I}} - \mathbf{P}_2^{\mathbf{I}}) \qquad (17)$$

Assim, percebe-se que os pontos de controle $\mathbf{P}_2^{\mathbf{I}}$, $\mathbf{P}_3^{\mathbf{I}} = \mathbf{P}_0^{\mathbf{II}}$ e $\mathbf{P}_1^{\mathbf{II}}$ devem ser colineares. Em algumas aplicações, entretanto, a continuidade C^1 não é exigida. Sendo necessário apenas, que os três pontos presentes na expressão (17) sejam colineares.

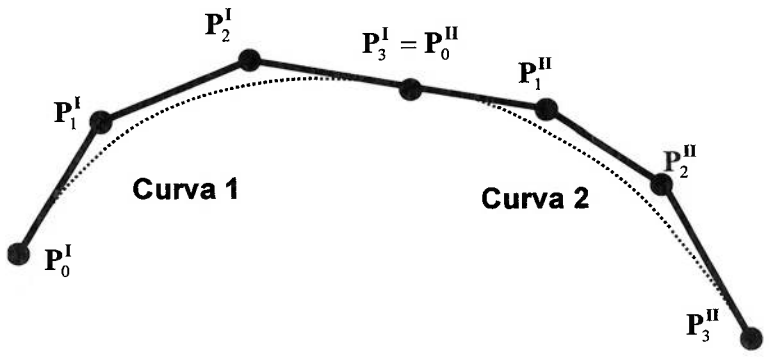


Figura 7. Conexão entre duas curvas de Bézier de grau 3.

2.2 Modelagem de Superfícies

As superfícies de Coons [6] são superfícies genéricas definidas pelas suas curvas de contorno. Sendo que as curvas de contorno são curvas, genéricas, podendo ser curvas de Bézier, Hermite, Interpolação Spline, B-Spline e outras. Neste capítulo apresentaremos um caso particular de superfície de Coons, onde as quatro curvas de contorno são curvas de Hermite. Apresentaremos também a superfície de Bézier. Mostraremos que é possível converter esta superfície de Coons particular para uma superfície de Bézier. Apresentaremos também como conectar superfícies de Bézier de modo que exista continuidade de derivada na transição entre uma superfície e outra. Será mostrado também que existe um grande problema em manter a continuidade, pois a modificação feita em uma superfície acarreta modificações nas superfícies vizinhas. Este problema será minimizado quando utilizarmos a superfície de Gregory.

2.2.1 Superfície de Coons

Como mostrado pela equação (1), as curvas de Hermite podem ser expressas como

$$Q(u) = UM_H G_H \qquad (18)$$

Onde $U = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix}$ e $M_H = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$

Considere as quatro curvas definidas abaixo:

$$Q(u,0) = UM_H \begin{bmatrix} P_{00} & P_{10} & P_{00}^u & P_{10}^u \end{bmatrix}^T$$

$$Q(u,1) = UM_H \begin{bmatrix} P_{01} & P_{11} & P_{01}^u & P_{11}^u \end{bmatrix}^T$$

$$Q(0,w) = WM_H \begin{bmatrix} P_{00} & P_{01} & P_{00}^w & P_{01}^w \end{bmatrix}^T$$

$$Q(1,w) = WM_H \begin{bmatrix} P_{10} & P_{11} & P_{10}^w & P_{11}^w \end{bmatrix}^T .$$

Também será necessário considerar que sobre o contorno estão definidos os seguintes vetores tangentes cruzados:

$$Q_u(0,w) = WM_H \begin{bmatrix} P_{00}^u & P_{01}^u & P_{00}^{uw} & P_{01}^{uw} \end{bmatrix}$$

$$Q_u(1,w) = WM_H \begin{bmatrix} P_{10}^u & P_{11}^u & P_{10}^{uw} & P_{11}^{uw} \end{bmatrix}$$

$$Q_w(u,0) = WM_H \begin{bmatrix} P_{00}^w & P_{10}^w & P_{00}^{uw} & P_{10}^{uw} \end{bmatrix}$$

$$Q_w(u,1) = WM_H \begin{bmatrix} P_{01}^w & P_{11}^w & P_{01}^{uw} & P_{11}^{uw} \end{bmatrix}$$

A superfície de Coons onde as quatro curvas de contorno são curvas de Hermite pode ser representada como [12]:

$$Q(u,w) = UM_H B M_H^T W^T \qquad (19)$$

Onde

$$B = \begin{bmatrix} P_{00} & P_{01} & P_{00}^{w} & P_{01}^{w} \\ P_{10} & P_{11} & P_{10}^{w} & P_{11}^{w} \\ P_{00}^{u} & P_{01}^{u} & P_{00}^{uw} & P_{01}^{uw} \\ P_{10}^{u} & P_{11}^{u} & P_{10}^{uw} & P_{11}^{uw} \end{bmatrix}$$

Esta forma da superfície de Coons também é conhecida como Ferguson Patch.

2.2.2 Superfíce de Bézier

A superfíce de Bézier é definida por uma matriz de pontos de controle. Uma superfíce de Bézier de grau $n \times m$ é representada por:

$$Q(u,w) = \sum_{i=0}^{n} \sum_{j=0}^{m} B_{i,n}(u) B_{j,m}(w) P_{i,j}, \text{ onde } 0 \le u, w \le 1 \qquad \textbf{(20)}$$

Aqui, $P_{i,j}$ indica os pontos de controle. Sendo que n+1 pontos de controle ficam na direção u e m+1 pontos de controle ficam na direção w, totalizando $(n+1)\times(m+1)$ pontos de controle. $B_{i,n}(u)$ e $B_{j,m}(u)$ são polinômios de Bernstein. Tal como as superfícies de Coons, as superfícies de Bézier podem ser expressas como:

$$Q(u,w) = UM_B P M_B^T W^T \qquad \textbf{(21)}$$

onde:

$$P = \begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} \\ P_{21} & P_{22} & P_{23} & P_{24} \\ P_{31} & P_{32} & P_{33} & P_{34} \\ P_{41} & P_{42} & P_{43} & P_{44} \end{bmatrix},$$

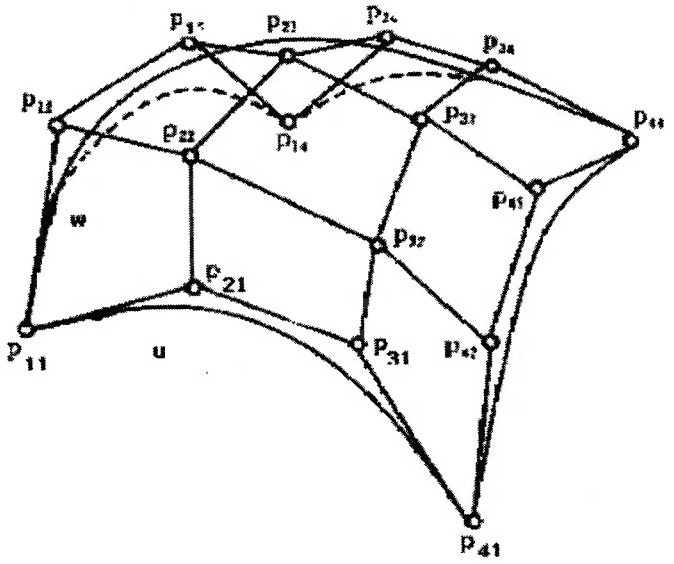


Figura 8 Exemplo de superfície de Bézier.

$$M_B = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

E para o caso (m=n=3) é possivel observar que 12 pontos de controle definem o contorno da superfície, e 4 pontos de controle ($P_{22}, P_{23}, P_{32}, P_{33}$) não estão sobre o contorno da superfície, mas definem a sua forma. A Figura 8 ilustra um exemplo de superfície de Bézier com m=n=3. Se temos a mesma superfície representada por Coons e Bézier, então pelas equações **(18)** e **(21)** obtemos:

$$UM_H BM_H^T W^T = UM_B PM_B^T W^T \qquad \textbf{(22)}$$

então

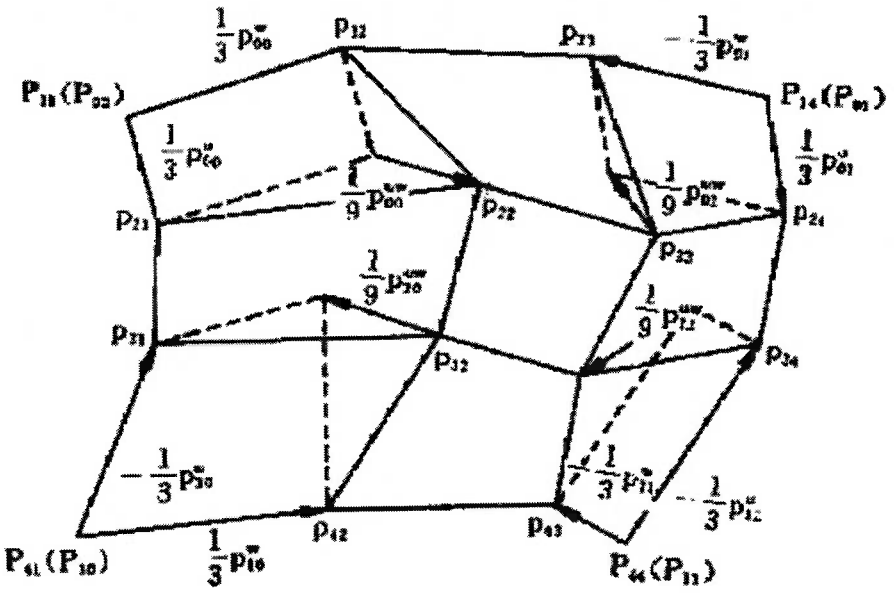


Figura 9 Conversão entre superfície de Bézier e superfície de Coons.

$$P = \mathrm{M_B^{-1}M_H B M_H^T}\left[\mathrm{M_B^T}\right]^{-1}$$

$$= \begin{bmatrix} P_{00} & P_{00}+\frac{1}{3}P_{00}^{w} & P_{01}-\frac{1}{3}P_{01}^{w} & P_{01} \\ P_{00}+\frac{1}{3}P_{00}^{u} & P_{00}+\frac{1}{3}(P_{00}^{u}+P_{00}^{w})+\frac{1}{9}P_{00}^{uw} & P_{01}+\frac{1}{3}(P_{01}^{u}-P_{01}^{w})-\frac{1}{9}P_{01}^{uw} & P_{01}+\frac{1}{3}P_{01}^{u} \\ P_{10}-\frac{1}{3}P_{10}^{u} & P_{10}-\frac{1}{3}(P_{10}^{u}-P_{10}^{w})-\frac{1}{9}P_{10}^{uw} & P_{11}-\frac{1}{3}(P_{11}^{u}+P_{11}^{w})+\frac{1}{9}P_{11}^{uw} & P_{11}-\frac{1}{3}P_{11}^{u} \\ P_{10} & P_{10}+\frac{1}{3}P_{10}^{w} & P_{11}-\frac{1}{3}P_{11}^{w} & P_{11} \end{bmatrix} \quad (23)$$

Onde $P_{00}^{uw}, P_{01}^{uw}, P_{10}^{uw}$ e P_{11}^{uw} são vetores de torção que são nulos em nosso caso. A equação (23) determina os pontos de controle da superfície de Bézier a partir dos parâmetros da superfície de Coons. A Figura 9 ilustra uma interpretação para a equação (23).

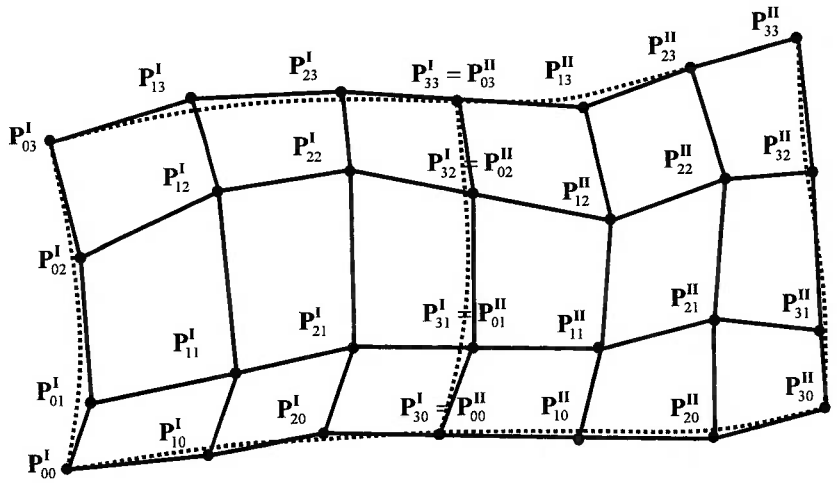


Figura 10. Conexão entre duas superfície de Bézier de grau 3.

2.2.3 Conexão Entre Superfícies de Bézier

Semelhantemente ao problema de conectar duas curvas de Bézier, é necessário conectar duas superfícies de Bézier $\mathbf{P}^{\mathrm{I}}(u,w)$ e $\mathbf{P}^{\mathrm{II}}(u,w)$ ao longo de uma curva de contorno comum, de modo que exista continuidade C^1 ao longo desta curva de contorno. Considere a situação ilustrada na Figura 10, onde a curva de contorno comum é definida pelos pontos de controle $\mathbf{P}^{\mathrm{I}}_{3i} = \mathbf{P}^{\mathrm{II}}_{0i}$ ($i = 0,1,2,3$). É possível demonstrar que um dos requisitos para que esta continuidade exista é que os pontos de controle $\mathbf{P}^{\mathrm{I}}_{2i}$, $\mathbf{P}^{\mathrm{I}}_{3i} = \mathbf{P}^{\mathrm{II}}_{0i}$ e $\mathbf{P}^{\mathrm{II}}_{1i}$ ($i = 0,1,2,3$) devem ser colineares [14].

Caso a análise feita anteriormente seja estendida, e quatro superfícies de Bézier $\mathbf{P}^{\mathrm{I}}(u,w)$, $\mathbf{P}^{\mathrm{II}}(u,w)$, $\mathbf{P}^{\mathrm{III}}(u,w)$ e $\mathbf{P}^{\mathrm{IV}}(u,w)$ adjacentes (vide Figura 11) sejam consideradas. Neste caso têm-se quatro curvas comuns de contorno. A restrição de continuidade C^1 deve ser aplicada ao longo de todas as quatro curvas comuns de contorno. Para a curva comum de contorno entre as superfícies $\mathbf{P}^{\mathrm{I}}(u,w)$ e $\mathbf{P}^{\mathrm{II}}(u,w)$ têm-se que os pontos de controle $\mathbf{P}^{\mathrm{I}}_{2i}$, $\mathbf{P}^{\mathrm{I}}_{3i} = \mathbf{P}^{\mathrm{II}}_{0i}$ e $\mathbf{P}^{\mathrm{II}}_{1i}$ ($i = 0,1,2,3$) devem ser colineares. Como para as outras curvas comuns de

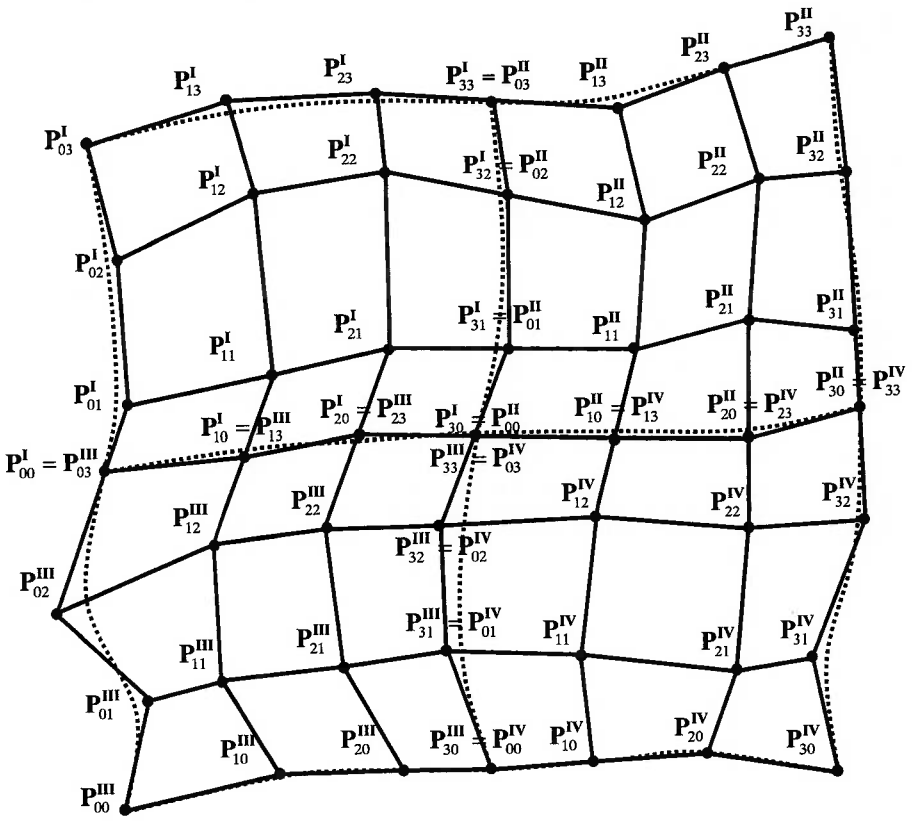


Figura 11. Exemplo de conexão entre quatro superfícies de Bézier.

contorno é possível obter um resultado semelhante, conclui-se que $\mathbf{P}^{\mathbf{I}}_{21}$, $\mathbf{P}^{\mathbf{II}}_{11}$, $\mathbf{P}^{\mathbf{III}}_{22}$ e $\mathbf{P}^{\mathbf{IV}}_{12}$ estão intrinsicamente relacionados. Assim, se um destes pontos for movimentado, os outros três pontos também deverão ser movimentados para manter a condição de continuidade C^1 ao longo das quatro curvas comuns de contorno.

2.2.4 Gregory Patch

A condição de amarramento entre os quatro pontos de controle $\mathbf{P}^{\mathbf{I}}_{21}$, $\mathbf{P}^{\mathbf{II}}_{11}$, $\mathbf{P}^{\mathbf{III}}_{22}$ e $\mathbf{P}^{\mathbf{IV}}_{12}$ (exibidos na Figura 11) é muito forte, restringindo de modo acentuado a possibilidade de modificação da forma da superfície. Tentando diminuir esta condição de amarramento, Chiyokura e Kimura [4] estudaram a superfície de Gregory. A seguir, serão discutidas algumas das propriedades da superfície de Gregory na forma de superfície cúbica de

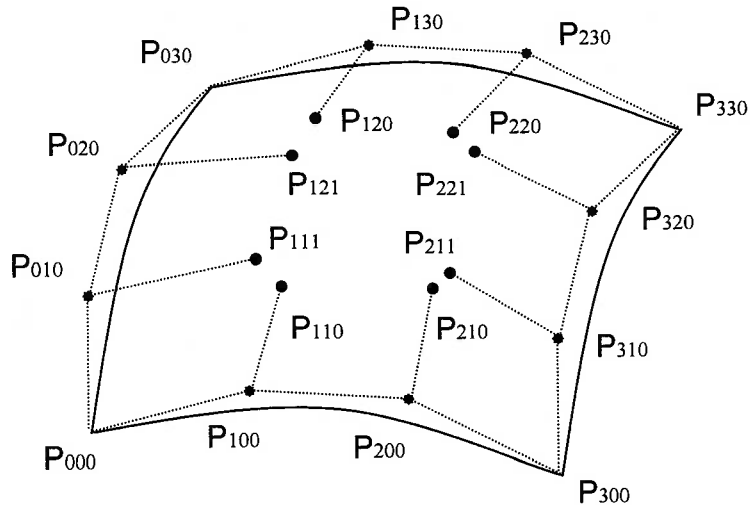


Figura 12 Exemplo de Gregory Patch.

Bézier [8,13]. Como ilustrado na Figura 12, uma superfície de Gregory é definida por um conjunto de 20 pontos de controle P_{ijk} ($i = 0,\cdots,3$, $j = 0,\cdots,3$, $k = 0,1$). As equações da superfície de Gregory são as seguintes:

$$Q(u,v) = \sum_{i=0}^{3}\sum_{j=0}^{3} B_i^3(u)\cdot B_j^3(v)\cdot Q_{ij}(u,v) \quad 0 \le u,v \le 1 \qquad \textbf{(24)}$$

Os pontos $Q_{11}(u,v)$, $Q_{12}(u,v)$, $Q_{21}(u,v)$ e $Q_{22}(u,v)$ são definidos por expressões especiais:

$$\begin{aligned}
Q_{11}(u,v) &= \frac{u\cdot P_{110} + v\cdot P_{111}}{u+v} \\
Q_{21}(u,v) &= \frac{(1-u)\cdot P_{210} + v\cdot P_{211}}{(1-u)+v} \\
Q_{12}(u,v) &= \frac{u\cdot P_{120} + (1-v)\cdot P_{121}}{u+(1-v)} \\
Q_{22}(u,v) &= \frac{(1-u)\cdot P_{220} + (1-v)\cdot P_{221}}{(1-u)+(1-v)}
\end{aligned} \qquad \textbf{(25)}$$

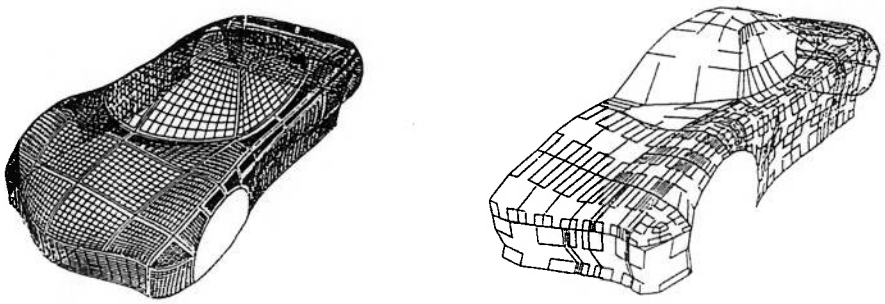


Figura 13. Exemplo de carroceria modelada utilizando 140 superfícies de Gregory.

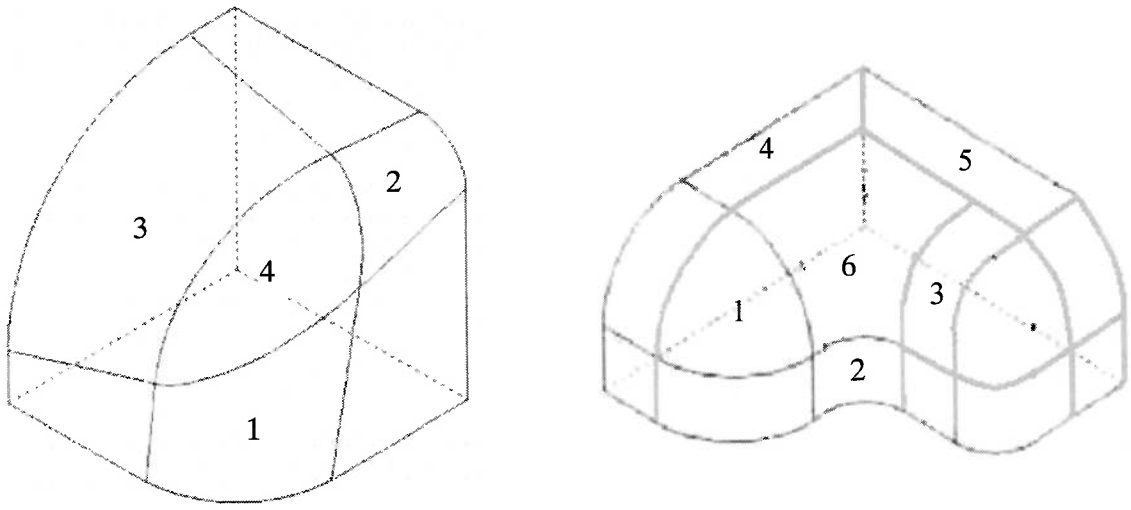
A superfície de Gregory pode ser degenerada para uma superfície de Bézier no caso em que os pontos internos de controle satisfizerem $P_{ij0} = P_{ij1}$. Também é possível determinar os quatro vetores tangentes cruzados às curvas de contorno, que serão especificados de forma independente[5]:

$$\begin{aligned} Q_v(u,0) &= 3\cdot\sum_{i=0}^{3} B_i(u)\left(P_{i10} - P_{i00}\right) \quad Q_u(0,v) = 3\cdot\sum_{i=0}^{3} B_i(v)\left(P_{1i0} - P_{0i1}\right) \\ Q_v(u,1) &= 3\cdot\sum_{i=0}^{3} B_i(u)\left(P_{i30} - P_{i20}\right) \quad Q_u(1,v) = 3\cdot\sum_{i=0}^{3} B_i(v)\left(P_{3i0} - P_{2i1}\right) \end{aligned} \qquad (26)$$

Esta última propriedade permite definir métodos localizados de interpolação. Assim, as superfície de Gregory apresentam uma grande vantagem em relação às superfícies de Bézier, uma vez que os pontos internos de controle estão associados a apenas uma curva de contorno. A Figura 13 ilustra uma carroceria modelada utilizando superfícies de Gregory.

2.3 Superfícies definidas automaticamente

Como já discutido no capítulo anterior, diversos sistemas de CAD/CAM requerem que a superfície tenha quatro lados para que expressões polinomiais sejam associadas. As superfícies que não possuem quatro lados são subdivididas de modo que apenas



(a) (b)

Figura 14 Um sólido que possui uma face triangular e um sólido com uma face de cinco lados.

superfícies com quatro lados sejam criadas. A seguir explicaremos em maior detalhe o algoritmo proposto por Chiyokura e Kimura [4]. A Figura 14 ilustra casos de superfícies que não possuem quatro lados criada por operações de suavização de arestas e vértices.

2.3.1 Face Triangular

Esta proposta foi apresentada por Chiyokura e Kimura [4], sendo que a proposta se concentra em determinar um ponto central sobre cada uma das curvas de contorno, fazendo com que o número de curvas dobre. Em seguida determina-se um ponto P_0 que está sobre a superfície. E por último determinam-se três curvas que conectem o ponto P_0 aos outros três pontos determinados inicialmente. O algoritmo é ilustrado pela Figura 15.

Na Figura 15, $P_{1,0}, P_{2,0}$ e $P_{3,0}$ são pontos sobre as curvas com valores de parâmetro igual a 0.5. Estes pontos são também os extremos das curvas internas. Considerando que os vetores normais são conhecidos então os pontos de controle $P_{i,j}$ $(i = 1,...,3)$ podem ser determinados.

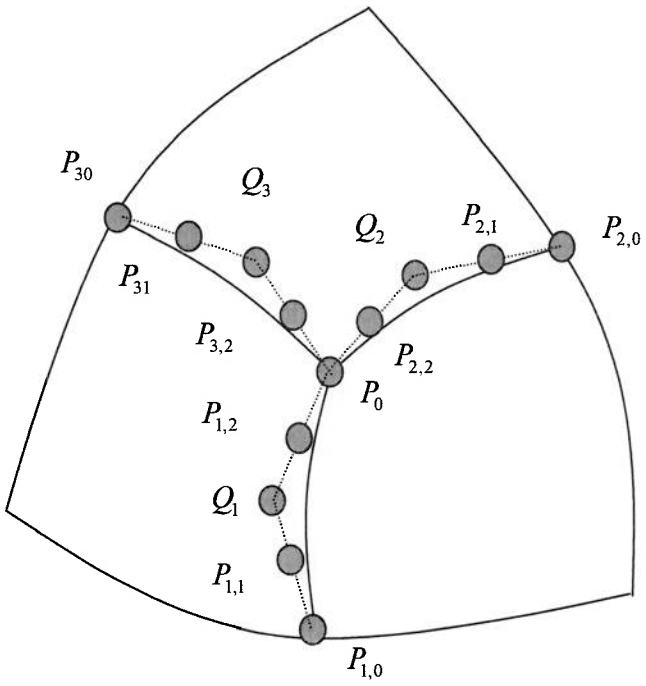


Figura 15. Caso de face com três lados.

Agora, assuma que as curvas internas são quadráticas para simplificar o algoritmo. Nesta condição, os pontos de controle das curvas quadráticas de Bézier, Q_1, Q_2 e Q_3 podem ser determinados: $Q_i = \frac{3P_{i,1} - P_{i,0}}{2}$ $(i = 1...3)$. Em seguida, será determinado o ponto P_0 onde as curvas internas se encontram. O ponto P_0 é determinado pela posição média dos pontos Q_1, Q_2 e Q_3, uma curva quadrática de Bézier é definida pelos pontos $P_i, Q_i, e\ P_0$ $(i = 1...3)$. Finalmente, as curvas internas são representadas como curvas cúbicas pela elevação do grau da curva de Bézier.

2.3.2 Face com Cinco Lados

Como ilustrado na Figura 14, é possível que surja um polígono com cinco lados e Chiyokura e Kimura [4] propuseram um algoritmo semelhante ao da proposta anterior para permitir que a geometria seja associada à face. Primeiro determina-se um ponto

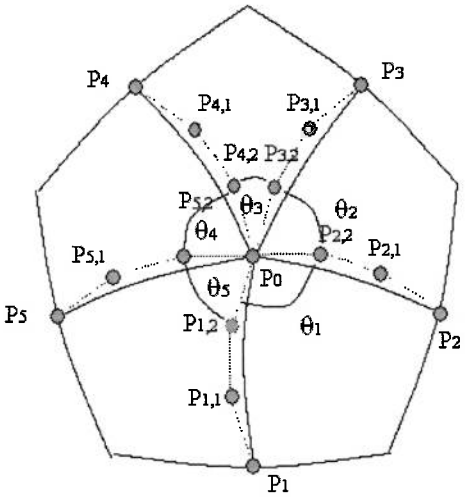


Figura 16. Caso de face com cinco lados.

sobre cada uma das curvas de contorno, fazendo com que o número de curvas dobre. Em seguida um ponto P_0 que está sobre a superfície é determinado. E por último determinam-se cinco curvas que conectam o ponto P_0 aos outros cinco pontos determinados inicialmente (Vide Figura 16).

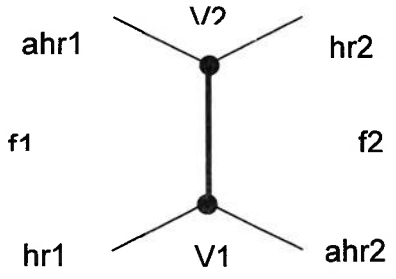
3 Modelagem de Sólidos

Neste capítulo apresentaremos a técnica de modelagem de sólidos conhecida como B-Rep (Boundary Representation) [10, 11]. Inicialmente detalharemos as propostas existentes para a sua estrutura de dados. Em seguida apresentaremos os Operadores de Euler que foram definidos para facilitar a manipulação da estrutura de dados. E por último, detalharemos algumas funções de alto nível, conhecidas como Operadores Locais, que criam alguns sólidos primitivos. Os operadores locais fazem uso exclusivamente de Operadores de Euler para editar a estrutura de dados.

3.1 Estrutura de Dados

Nesta sessão explicaremos como os elementos primitivos são representados de modo a definir um sólido B-Rep. Um modelo B-Rep possui sete tipos de elementos primitivos (vértice, meia-aresta, aresta, laço, face, shell e região). Temos três estruturas de dados propostas na literatura que implementam a representação B-Rep. Todas baseadas na aresta como elemento de referência: estrutura *winged-edge,* estrutura *meia-aresta* e estrutura unificada. Neste trabalho, será utilizada a estrutura unificada.

A estrutura w*inged-edge* mantém as informações de adjacência por meio de ponteiros a



v1	v2
hr1	hr2
ahr1	ahr2
f1	f2

hr = aresta no sentido horário
ahr = aresta no sentido anti-horário

Figura 17. Estrutura *winged-edge.*

vários elementos adjacentes à aresta de referência: duas faces, dois vértices e quatro arestas. Cada uma das quatro arestas compartilha com a aresta de referência um vértice e uma face (vide Figura 17).

A estrutura *meia-aresta* representa a metade das informações de adjacência da estrutura *winged-edge*. Cada *meia-aresta* possui apenas uma orientação, e cada face possui um circuito direcional de *meia-arestas.* Para evitar a duplicidade das informações associadas à aresta (por exemplo, atributo de cor) nas duas *meia-arestas*, foi proposta a criação de uma nova estrutura para a aresta que associaria as duas *meia-arestas* que definem a aresta de referência, evitando assim o ponteiro para a outra *meia-aresta*. Esta é a estrutura unificada.

Até este momento, foi assumido que cada face possui apenas um contorno. Entretanto, casos práticos requerem que uma face possua mais de um contorno (como uma face com furos). Faces com mais de um contorno podem ser simuladas pela técnica de aresta-ponte (*bridge-edge*) no qual uma aresta une os contornos de uma face entre si. A aresta-ponte, portanto, possui a mesma face adjacente em ambas as laterais (vide Figura 18 (a)).

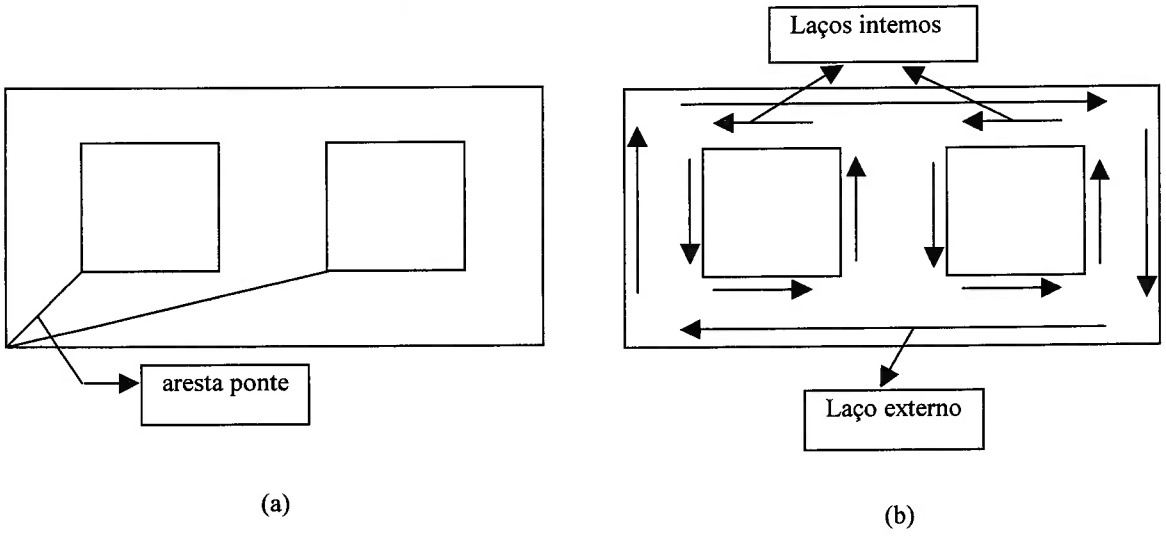


Figura 18. Extensão para faces com múltiplos contornos.

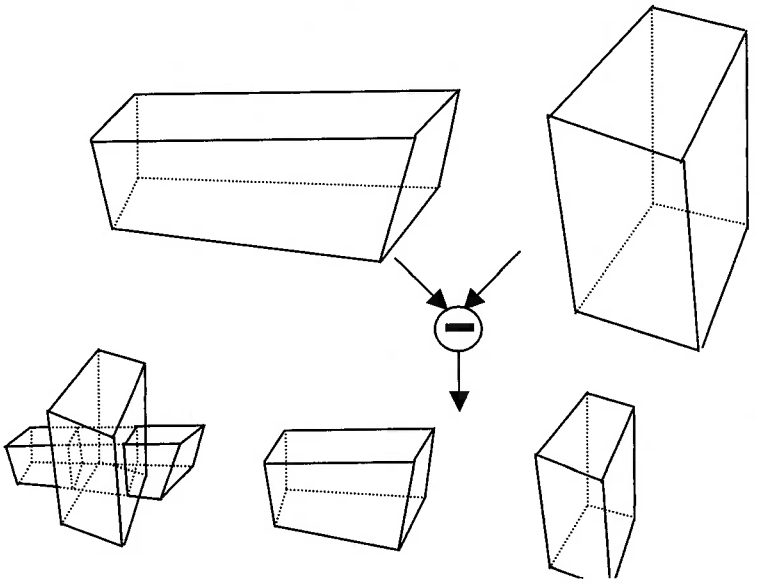


Figura 19. Sólido com duas regiões.

Entretanto, a técnica aresta-ponte não é muito eficiente porque será necessário determinar como os contornos devem ser conectados pelas arestas-ponte, o que criará a necessidade de complexos algoritmos para implementar operações de modelagem. Como exemplo, podemos citar os Operações Booleanas que provavelmente interseccionarão as arestas-ponte. Alterações na estrutura de dados unificada de maneira a suportar faces com mais de um contorno não afetarão a estrutura B-Rep ao nível de aresta, mas sim, ao nível de face. Uma técnica muito comum é adicionar uma estrutura de tamanho fixo chamada laço (loop) que é associada a cada contorno da face. A estrutura laço simplesmente fornece à estrutura face um mecanismo para manter uma lista ligada de ponteiros para os seus múltiplos contornos. Cada face possui um laço externo e zero ou mais laços internos (vide Figura 18 (b)).

Devido ao formalismo das Operações Booleanas, ao combinarmos dois sólidos por uma Operação Booleana, o resultado será sempre apenas um sólido. Entretanto, mesmo em situações especiais, como a situação exemplificada na Figura 19, onde o resultado da Operação Booleana aparenta apresentar dois sólidos, o resultado é considerado como sendo apenas um sólido. Entretanto, nesta situação em especial, considera-se que o sólido resultante possui duas regiões. Em outras palavras, para representar esta situação

especial, foi criado o elemento região que representa conjuntos disconexos de faces no espaço. A Figura 20 ilustra os elementos primitivos de um modelo sólido. A Figura 21 ilustra a hierarquia da estrutura unificada resultante.

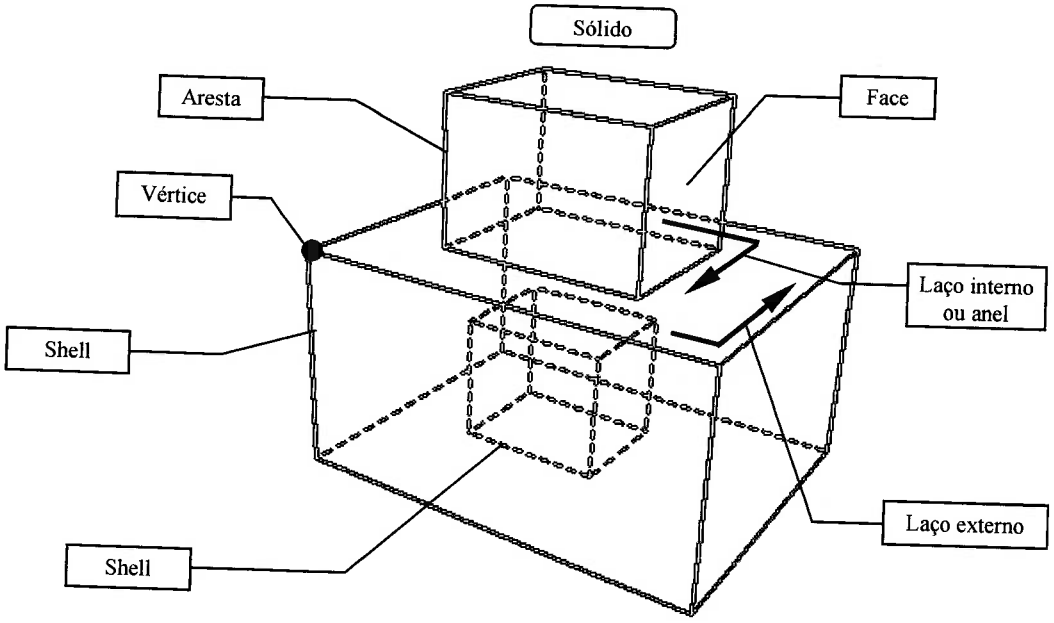


Figura 20. Elementos topológicos de um modelo sólido.

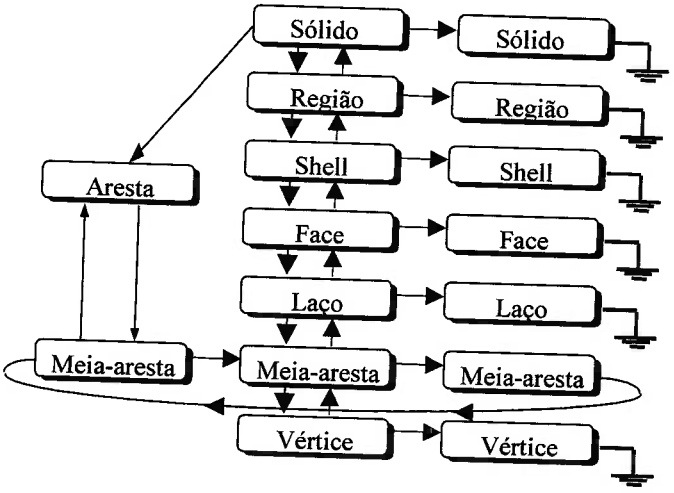


Figura 21. Hierarquia de elemento da estrutura unificada.

3.2 Operadores de Euler

Por conterem informações sobre as adjacências entre os elementos primitivos, as estruturas computacionais anteriormente expostas são bastante complexas e a sua manipulação exige muitos cuidados para que a consistência dos dados seja mantida. Para contornar este problema, um conjunto de operadores foi desenvolvido com o objetivo de tornar a manipulação das estruturas de dados da representação B-Rep mais intuitiva. Eles permitem que a construção do sólido possa ser executada passo a passo, escondendo todos os detalhes de implementação da estrutura de dados. Estes operadores formam o segundo nível de representação do modelador B-Rep.

A equação de Euler-Poincaré diz que um sólido poliédrico é topologicamente válido se a seguinte relação entre as suas quantidades de elementos for verificada:

$$v - e + 2f = 2(s - h) + l; \tag{27}$$

onde *v* é o número de vértices do sólido, *e* o número de arestas, *f* o número de faces, *s* o número de shells, *h* o número de furos e *l* o número de laços. A forma mais conhecida na literatura da equação de Euler-Poincaré supõe ainda a existência de *r* anéis no sólido, onde *r=l-f*. Ficando a equação da seguinte forma:

$$v - e + f = 2(s - h) + r \tag{28}$$

Vários autores demonstram que sete operadores são suficientes para construir todos os sólidos. Enquanto estes sete operadores podem ser escolhidos de várias maneiras, considerações de modularidade e independência criaram apenas pequenas variações na coleção encontrada na literatura [11, 14]. Com a finalidade de facilitar a memorização, os Operadores de Euler estão definidos utilizando a nomenclatura da Tabela 1.

Tabela 1. Nomenclatura dos Operadores de Euler

Símbolo	Significado
M	*make*
K	*kill*
V	*vertex*
E	*edge*

Símbolo	Significado
F	*face*
S	*shell ou solid*
H	*hole*
R	*ring ou region*

Por exemplo, o operador **MEV** deve ser traduzido por *Make Edge, Vertex* (Crie uma aresta e um vértice). A seguir são descritos os sete operadores mais comumente utilizados na literatura e uma representação gráfica para cada operador pode ser observada na Figura 22.

- **MVSF** (*Make Vertex Solid Face)* : este operador cria um sólido inicial com apenas uma face e um vértice;
- **MEV** (*Make Edge Vertex)* : este operador adiciona a um sólido uma aresta e um vértice. A aresta é criada conectando-se um vértice já existente ao novo vértice criado;
- **MEF** (*Make Edge Face)* : este operador adiciona ao sólido uma aresta e uma face. A face é criada pela divisão de uma face já existente acrescentando-se a nova aresta;
- **KEMR** (*Kill Edge Make Ring)* : este operador divide o contorno de uma face em dois laços pela remoção de uma aresta-ponte;
- **KFMRH** (*Kill Face Make Ring Hole)* : nenhum dos operadores discutidos anteriormente é capaz de modificar as propriedades topológicas globais da estrutura de dados, como dividir um sólido em dois componentes ou criar um furo passante. O operador **KFMRH** possui este objetivo;
- **MSFKR** (*Make Shell Face Kill Ring)* : este é outro operador que manipula informações globalmente. Ele transforma o anel de uma face em uma nova face, e todo o conjunto de faces associadas à nova face constituirá uma novo *shell*;

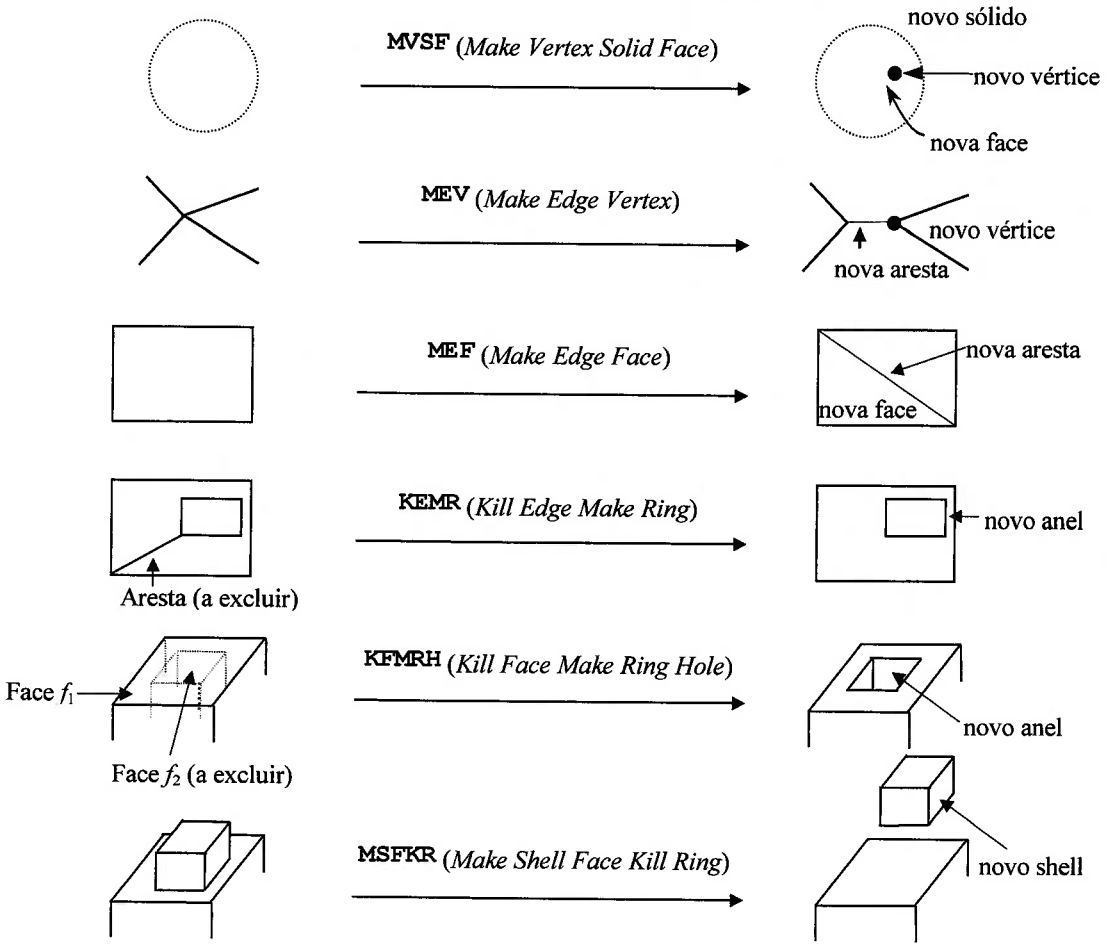


Figura 22. Representação gráfica dos operadores de Euler.

- **MRSFKR** (Make Region Shell Face Kill Ring): este é outro operador que manipula informações globalmente. Ele transforma o anel de uma face em uma nova face, e todo o conjunto de faces associadas à nova face constituirar é uma nova região.

Os sete operadores detalhados acima são todos construtivos entretanto, para que o sólido possa ser alterado satisfatoriamente ele precisa também de algumas operações destrutivas. Por isto, cada operador construtivo possui um operador correspondente destrutivo. São eles:

Tabela 2. Operadores construtivos e destrutivos

KVSF (*Kill Vertex Solid Face*)	⇔	**MVSF** (*Make Vertex Solid Face*)
KEV (*Kill Edge Vertex*)	⇔	**MEV** (*Make Edge Vertex*)
KEF (*Kill Edge Face*)	⇔	**MEF** (*Make Edge Face*)
MEKR (*Make Edge Kill Ring*)	⇔	**KEMR** (*Kill Edge Make Ring*)
MFKRH (*Make Face Kill Ring Hole*)	⇔	**KFMRH** (*Kill Face Make Ring Hole*)
KSFMR (*Kill Shell Face Make Ring*)	⇔	**MSFKR** (*Make Shell Face Kill Ring*)
KRSFMR (*Kill Region Shell Face Make Ring*)	⇔	**MRSFKR** (*Make Region Shell Face Kill Ring*)

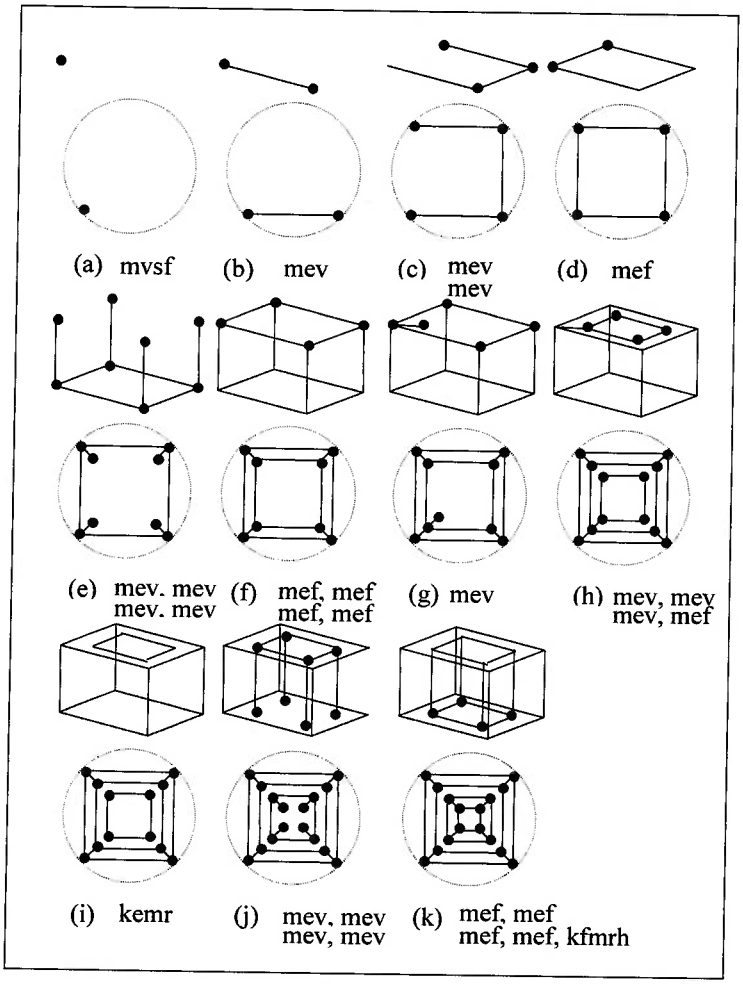


Figura 23. Seqüência de Operadores de Euler para criação de um cubo com furo passante.

Durante o processo de construção de um sólido pela utilização de Operadores de Euler, a validade topológica do mesmo é mantida observando-se a equação de Euler-Poincaré. Entretanto, é comum o agrupamento de Operadores de Euler em uma certa seqüência para mantermos também a geometria válida. É importante observar que não há como manter a geometria válida em todos os estágios da construção. Por isto, os Operadores de Euler devem ser agrupados em seqüências que possuam algum significado. Através de um exemplo simples, um bloco retangular com um furo passante retangular, é possível ilustrar a utilização dos Operadores de Euler (Figura 23).

3.3 OPERADORES LOCAIS

Os operadores locais permitem que algumas características dos sólidos possam ser modificadas diretamente pelo usuário preservando as consistências topológica e geométrica do local modificado. Por não realizarem alterações em áreas externas à localidade em que eles se aplicam, não são realizadas verificações sobre o sólido após a realização de uma operação local. Entretanto, existe a possibilidade de se criar um sólido inválido utilizando-se operadores locais, como por exemplo, um sólido auto-interceptante (Figura 24). Não é uma tarefa fácil verificar se um sólido é auto-interceptante; geralmente, depende da correta utilização destas operações pelo usuário.

Um operador local pode ser considerado uma extensão das Operações de Euler. Um operador local, ao ser acionado pelo usuário, definirá uma seqüência de Operadores de Euler que a implemente. Antes de definirmos os operadores locais, apresentaremos

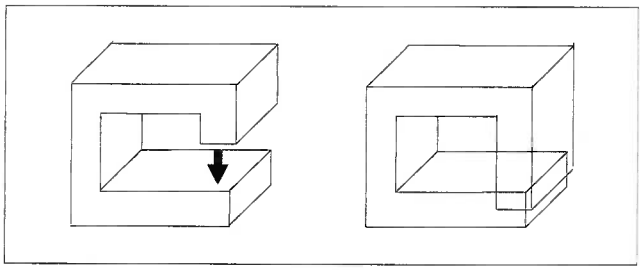


Figura 24. Um sólido auto-interceptante.

algumas rotinas básicas de modelagem.

3.3.1 Cria arco de circunferência

Esta rotina define uma aproximação poligonal de um arco de circunferência, a partir de um ponto fornecido. Assim, o algoritmo que implementa esta rotina pode ser definido utilizando-se apenas o Operador MEV (vide Figura 25). Para criar uma circunferência, será necessário definir um arco de 360° e o último Operador de Euler MEV deve ser substituído por um operador MEF (vide Figura 26).

3.3.2 Extrusão translacional:

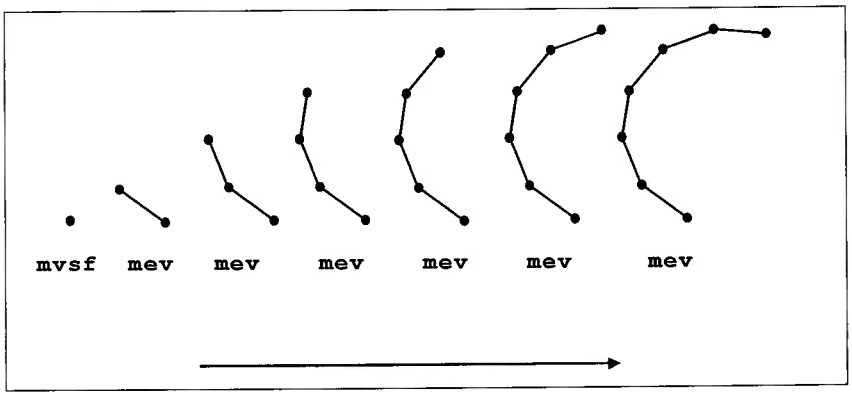


Figura 25. Seqüência de Operadores de Euler que implementam a criação de um arco.

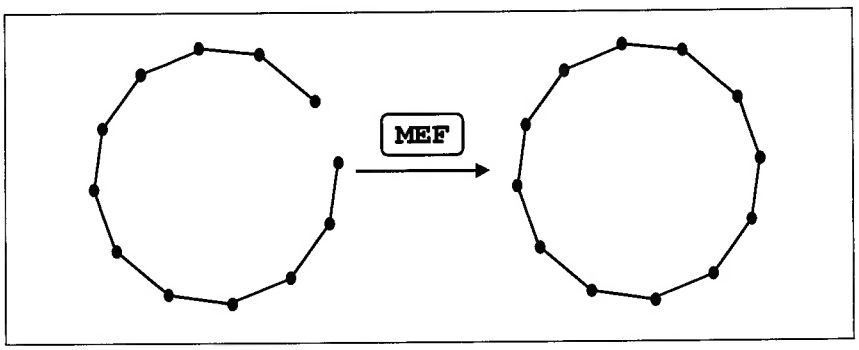


Figura 26. Finalização da criação de uma circunferência.

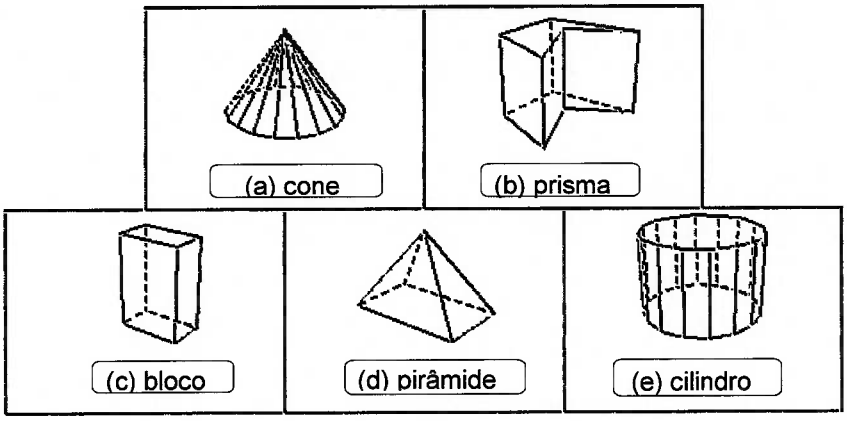


Figura 27. Sólidos obtidos por extrusão translacional.

A extrusão translacional deve ser aplicada a uma face e será realizada segundo um sentido específico. Como exemplo de sólido prímitivo criado por meio deste operador temos o cubo (vide Figura 27 (c)), o cilindro (vide Figura 27 (e)) e o prisma (vide Figura 27 (b)). Se estiver disponível uma função que degenere uma face em um ponto, será possível criarmos sólidos primitivos como o cone (vide Figura 27 (a)) e a pirâmide (vide Figura 27 (d)).

3.3.3 Extrusão rotacional:

A operação de extrusão rotacional consiste em girar uma figura bidimensional, com contorno aberto ou fechado, ao redor de um eixo. Sólidos criados por meio deste operador são: a esfera, o peão de xadrez e o toróide (vide Figura 28).

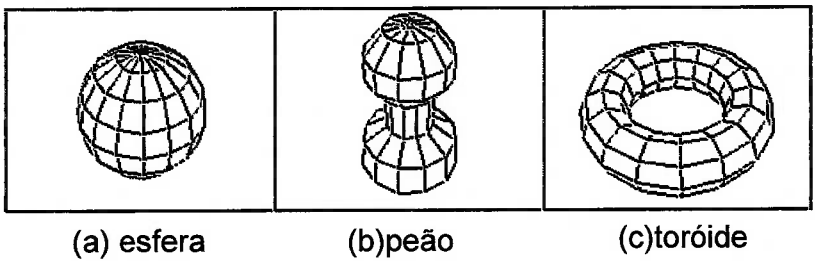


Figura 28. Sólidos gerados por extrusão rotacional.

4 Sincronização da Modelagem de Sólidos e Modelagem Geométrica

Neste capítulo apresentaremos a proposta feita por Turner [16], em que as superfícies são representada como atributos de faces, e que inclui um mecanismo para subdividir a superfície em um conjunto de faces planas. Este mecanismo faz uso de Operadores de Euler. Também apresentaremos a proposta feita por Ueda [17] para determinar linhas poligonais que aproximam uma curva. Nesta proposta as curvas são representadas como atributos de arestas. Finalmente, entenderemos porque os dois mecanismos não são compatíveis entre si.

4.1 Propostas de Turner para Sincronizar Faces e Superfícies

A proposta de Turner [16], está baseada no uso de atributos para faces e arestas. Considere a Figura 29 que apresenta um cilindro aproximado por faces planas e arestas lineares. As faces planas que representam o topo e o fundo do cilindro são não "facet faces". Ou seja, estas faces não estão aproximadas e se mantém na forma original. As

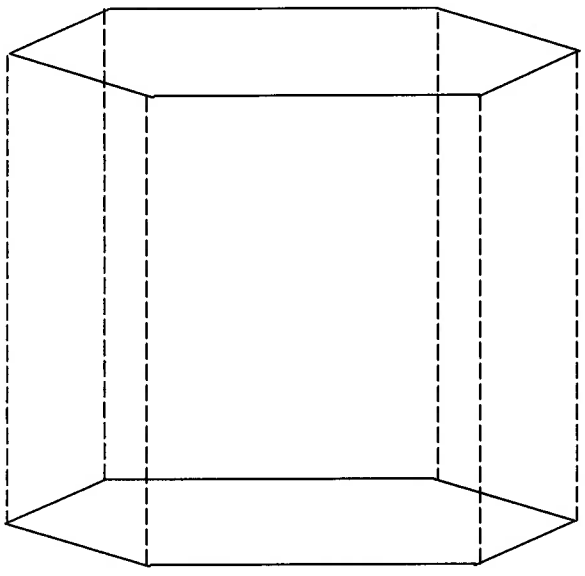


Figura. 29 Um cilindro
Aproximado por faces e arestas.

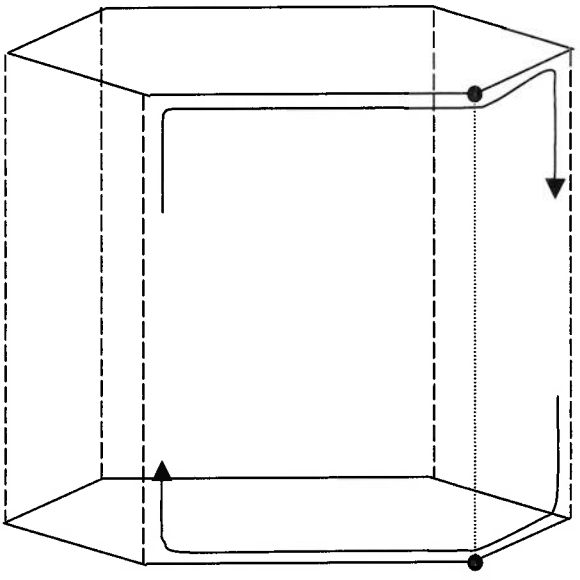


Figura 30. Cilindro
depois da aplicação do operador KEF.

faces planas laterais são classificadas como "facet faces", pois são aproximações da superfície original. Cada "facet face", possui um ponteiro para a representação matemática exata da superfície que ela aproxima. As arestas também podem ser classificadas como "facet edge" e não "facet edge". As arestas classificadas como "facet edge" separam duas faces classificadas como "facet face". As demais arestas são classificadas como não "facet edge". Turner apresentou dois mecanismos: desenvolvimento que cria as faces planas aproximadoras; e enxugamento que remove as faces planas aproximadoras mantendo o número mínimo de elementos primitivos para manter o sólido topologicamente válido e consistente com a geometria associada.

A seguir ilustramos o algoritmo de enxugamento. A Figura 29 apresenta um cilindro totalmente desenvolvido com suas faces aproximadoras. Assim, ele possui seis "facet faces" e seis "facet edge". Na Figura 30 temos que uma face e uma aresta foram removidas pela aplicação do operador de Euler KEF. Na Figura 31 temos que um vértice e uma aresta superiores foram removidas pela aplicação do operador de Euler KEV. Finalmente, após a aplicação de cinco operadores KEF, onze operadores KEV e um operador KEMR obtemos o sólido apresentado na Figura 32.

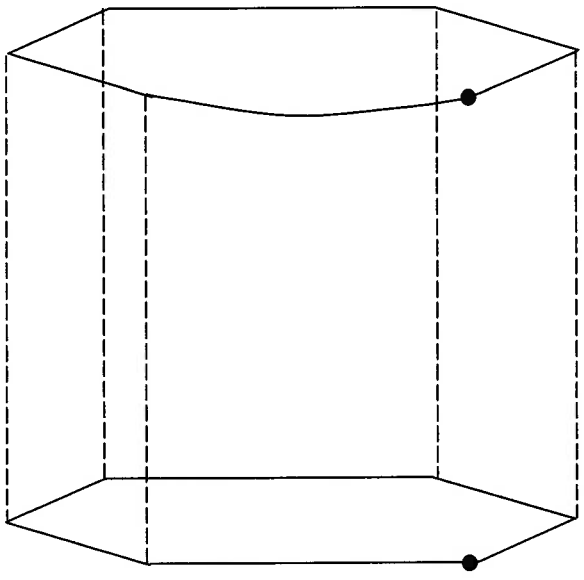


Figura 31. Cilindro
depois da aplicação do operador KEV.

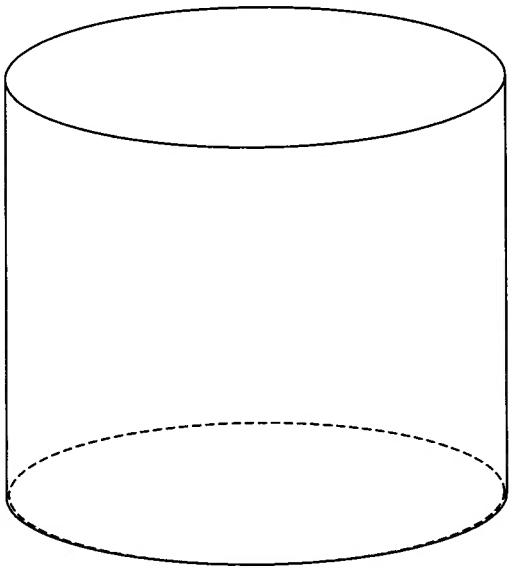


Figura 32. Cilindro
final após o enxugamento.

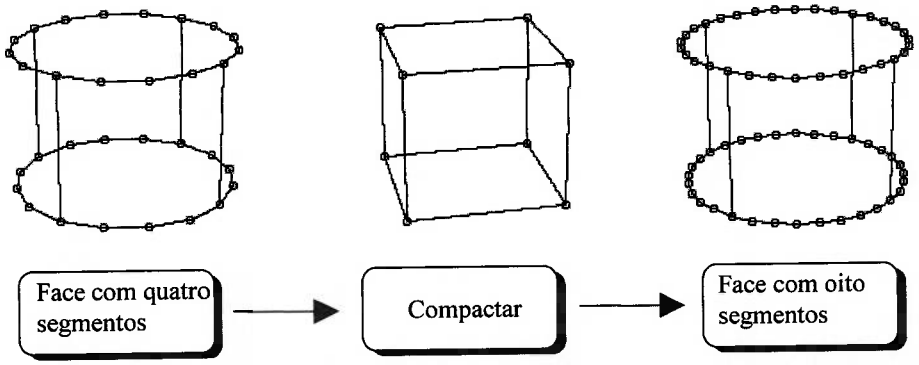


Figura 33. Mecanismo que mostra como aumentar o número de extremidades aproximadas.

O conceito apresentado por Turner é que durante o algoritmo de enxugamento, as arestas classificadas como "facet edge" devem ser removidas pela aplicação do operador de Euler KEF. Entretanto, Turner não comenta sobre como decidir se uma aresta classificada como não "facet edge" deve ser removida ou não. Foi pensando neste problema que Ueda [17] iniciou suas pesquisas.

4.2 Proposta de Ueda para Sincronizar Arestas e Curvas

Ueda [17] propôs dois mecanismos: desenvolvimento que cria arestas aproximadoras; e enxugamento que remove as arestas lineares aproximadoras mantendo o número mínimo de elementos primitivos, para manter o sólido topologicamente válido e consistente com a geometria associada.

É possível imaginar que em algumas situações será necessário aumentar o número de arestas aproximadoras ou reduzi-lo. A Figura 33 exibe um exemplo onde um cilindro é inicialmente criado com quatro arestas. Na figura intermediária, cada curva é associada a apenas uma aresta e na última figura a curva é aproximada por oito arestas. Neste caso precisamos de dois operadores, um para desenvolver a curva segundo um número de arestas e outro para compactar a curva.

- **Develop Curve:** este operador de alto nível é utilizado para criar um conjunto de arestas que aproximam a curva associada a uma aresta específica. Este

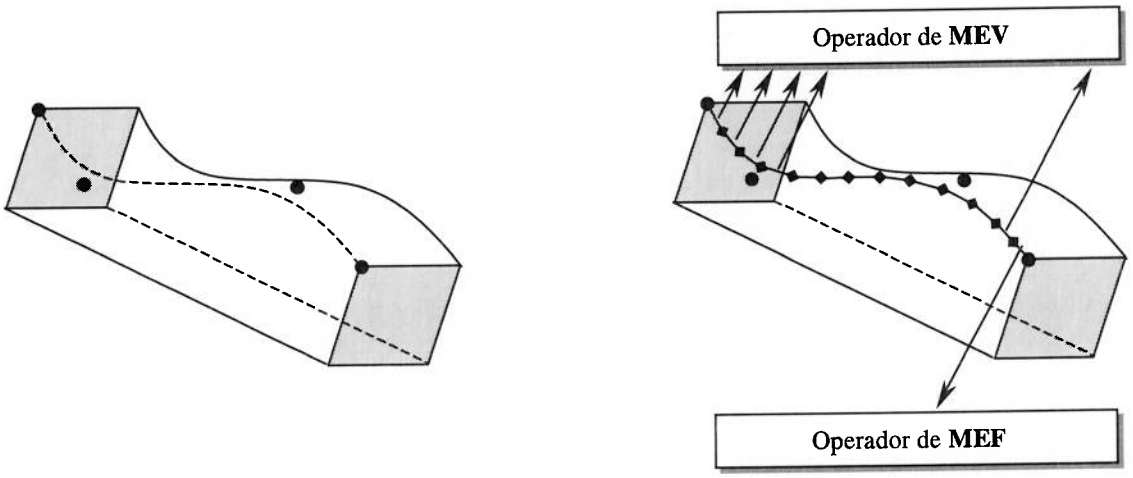


Figura 34. Operador de alto nível Develop Curve.

operador calcula pontos de aproximação para a curva e aplica operadores MEV (Make Edge Vertex). Um ponto importante neste operador é identificar o correto sentido da curva quando ocorrer a aplicação dos operadores MEV para que a curva não seja criada de modo invertido. Como, para uma aresta temos duas meia-arestas, é necessário verificar qual meia-aresta é a mais adequada para efetuar o desenvolvimento da curva. Para toda nova aresta criada, associa-se a aresta à curva que ela aproxima. Este relacionamento permitirá identificar quais são as arestas que aproximam uma curva. A Figura 34 ilusta a aplicação deste operador.

- **ShrCur (Shrink Curve):** O operador de enxugamento remove as arestas de aproximação da curva deixando apenas uma única aresta asociada à curva. A Figura 35 ilusta a aplicação deste operador que é implementado com uma seqüência de operadores KEV. Antes que um operador de desenvolvimento seja utílizado é necessário aplicar o operador de enxugamento, pois o operador de desenvolvimento assume que existe uma única aresta associada à curva a ser aproximada. O operador de enxugamento identifica as arestas vizinhas que possuem a mesma curva associada. Se uma aresta vizinha com a mesma curva associada for encontrada, ela deve ser removida pelo operador KEV. Caso

nenhuma aresta vizinha com a mesma curva associada seja encontrada, significa que o algoritmo deve ser terminado.

O sólido com superfícies aproximadas por faces planas será utilizado para calcular propriedades integrais de massa. O sólido com curvas aproximadas por arestas lineares é utilizado para exibir o sólido de modo wire-frame como na Figura 33. Uma possibilidade inconsistente será se o usuário estiver observando um sólido em modo wire-frame e solicitar o valor de uma propriedade integral.

Uma possibilidade pera solucionar este problema seria enxugar todas as arestas aproximadoras para que as faces planas que aproximam as superfícies possam ser determinadas. Após a determinação das faces planas, a propriedade integral pode ser calculada. Finalmente, ocorre o enxugamento das faces planas aproximadoras para que as arestas aproximadoras sejam criadas a fim de que o usuário possa visualizar o sólido em modo wire-frame.

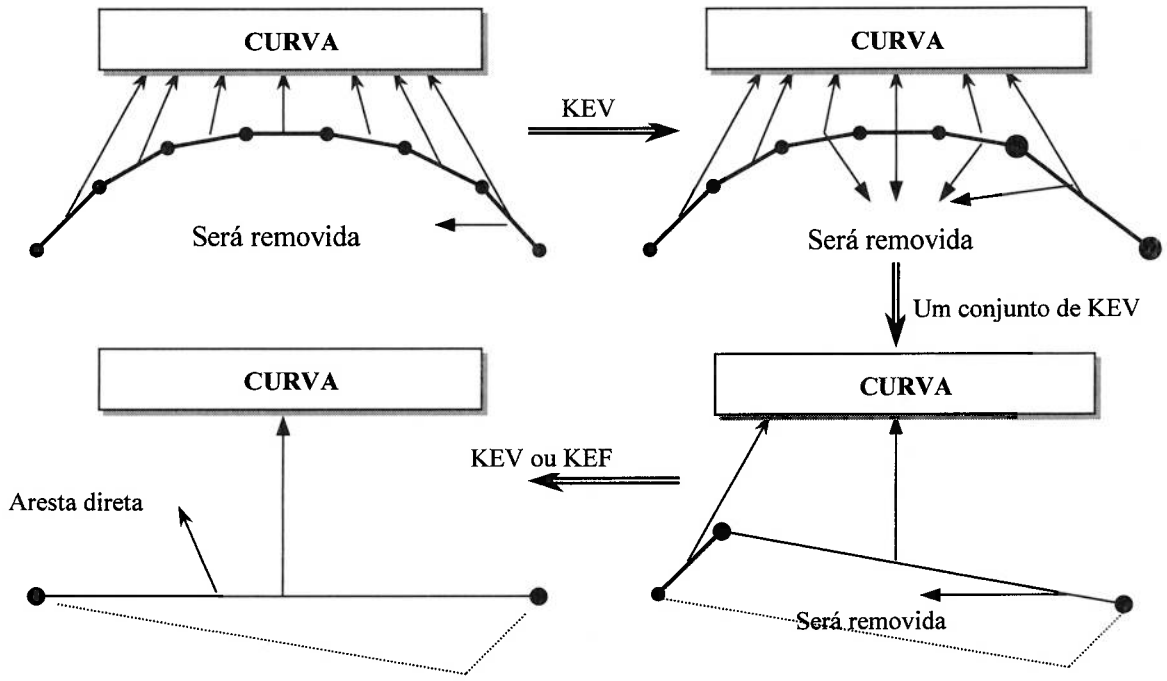


Figura 35 Remove as arestas de aproximação de curva.

Uma segunda solução é observar que as arestas classificadas como não "facet edge" são arestas lineares que aproximam as curvas existentes no modelo. Assim, para exibir em modo wire-frame bastaria varrer todas as arestas existentes no sólido com as superfícies aproximadas por faces planas e exibir apenas as arestas classificadas como não "facet edge". Entretanto, desta maneira, o nível de detalhamento de aproximação das curvas seria o mesmo que o nível de detalhamento da aproximação das superfícies, e isto não é desejável.

O nível de detalhamento para a aproximação das curvas está associado à razão da dimensão da curva sobre à dimensão da câmera [12]. O nível de detalhamento para a aproximação das superfícies está associado à dimensão da superfície sobre a tolerância suportável pelo cálculo da propriedade integral. Deste modo, o ideal é que ambos os níveis de detalhamento sejam independentes. Assim, neste ponto é que iniciamos as nossas pesquisas.

5 Sincronizando Modelagem de Sólidos e Modelagem Geométrica

Neste trabalho, foi utilizada a estrutura meia-aresta proposta por Mäntylä [11]. A estrutura é apresentada na Figura 36. Neste capítulo apresentaremos a nossa proposta para suporte ao sincronismo entre modelagem de sólidos e modelagem geométrica. Inicialmente discutiremos alguns requisitos que tal proposta deve atender. Em seguida apresentaremos em maior profundidade a proposta apresentada por Ueda [17], incluindo: estrutura de dados operadores básicos e operadores locais. Em seguida apresentaremos a nossa proposta para representar superfícies em modeladores de sólidos B-Rep,

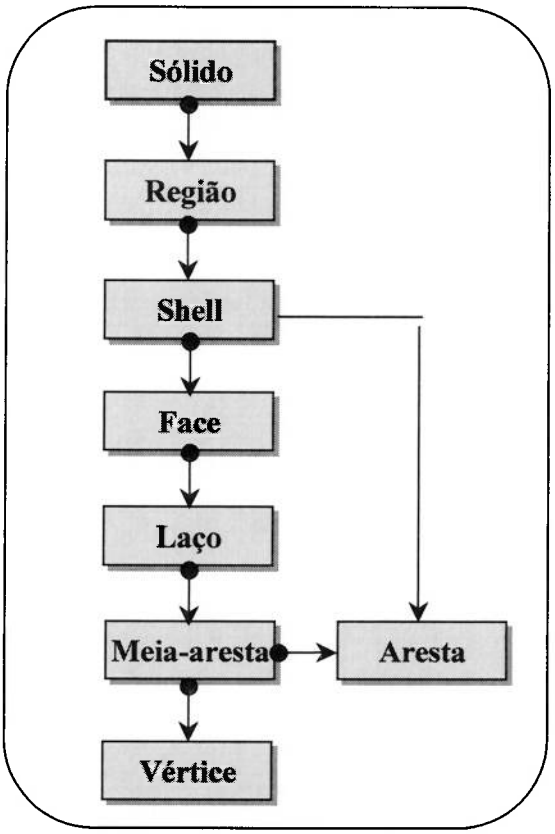
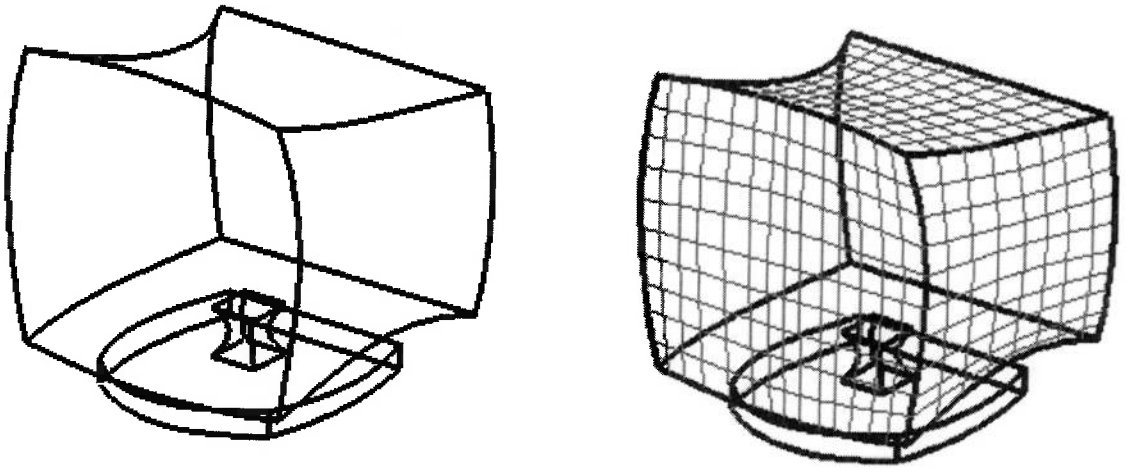


Figura 36. Estrutura meia-aresta.

incluindo: estrutura de dados, operadores básicos e operadores locais.

Apesar de ser possível representar qualquer curva ou superfície por meio de suas aproximações lineares e poligonais, ainda permanecem vários problemas de interação homem-máquina. Considere o seguinte exemplo: o usuário define o conjunto de pontos de controle para uma superfície, em seguida, o modelador de sólidos transforma este conjunto de informações em arestas e polígonos. Após uma certa análise, o usuário decide modificar um ponto de controle, selecionando-o e posicionando-o segundo as coordenadas adequadas. O modelador de sólidos deve saber quais faces poligonais estão associadas à superfície que foi modificada para então atualizá-las. Veja o modelo exibido na Figura 37, cada curva está associada a duas superfícies, e cada superfície possui diversas faces planas aproximadoras. Como devemos identificar quais faces aproximadoras estão associada à curva que foi modificada?

Como veremos neste capítulo, a solução para o problema de associar faces poligonais e superfícies consiste em definir um novo mecanismo para associar a geometria das superfícies às faces poligonais. Isto será feito pela ampliação da estrutura de dados



(a) (b)

Figura 37 Modelo curva e modelo com superfícies.

utilizada no modelo básico, definindo novos conceitos para representar o modelo sólido e modificando os algoritmos para que o modelador de sólidos trabalhe corretamente. Apresentaremos também alguns operadores que foram definidos para encapsular e facilitar a utilização destes novos conceitos.

5.1 Estrutura de Dados para representar Curvas

Mäntylä [11] e Toriya e Chiyodura [14] representam curvas como atributos de arestas. Ueda [17] propôs utilizar o mesmo principio e adaptou o mecanismo de desenvolvimento e enxugamento de superfícies proposto por Turner [16], para ser aplicado às arestas. Assim, o principal objetivo desta proposta é permitir que curvas sejam exibidas e controladas como se fossam sequências de arestas lineares. A estrutura de dados

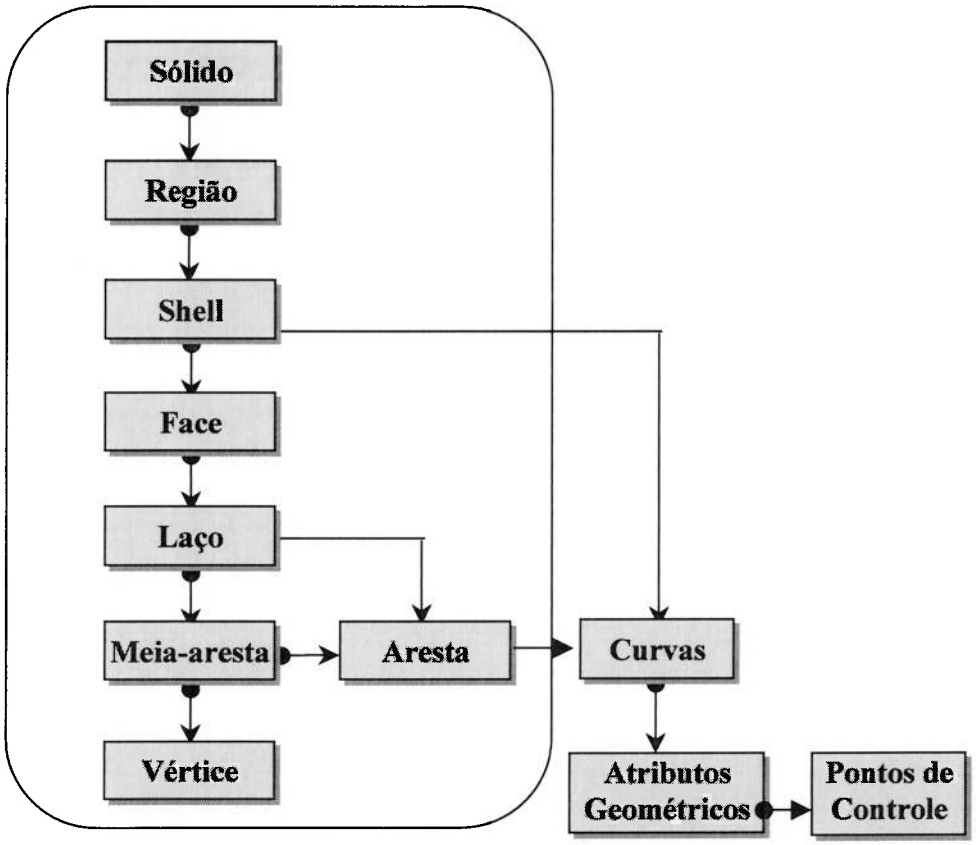


Figura 38 Estrutura Hierárquica para representar curvas.

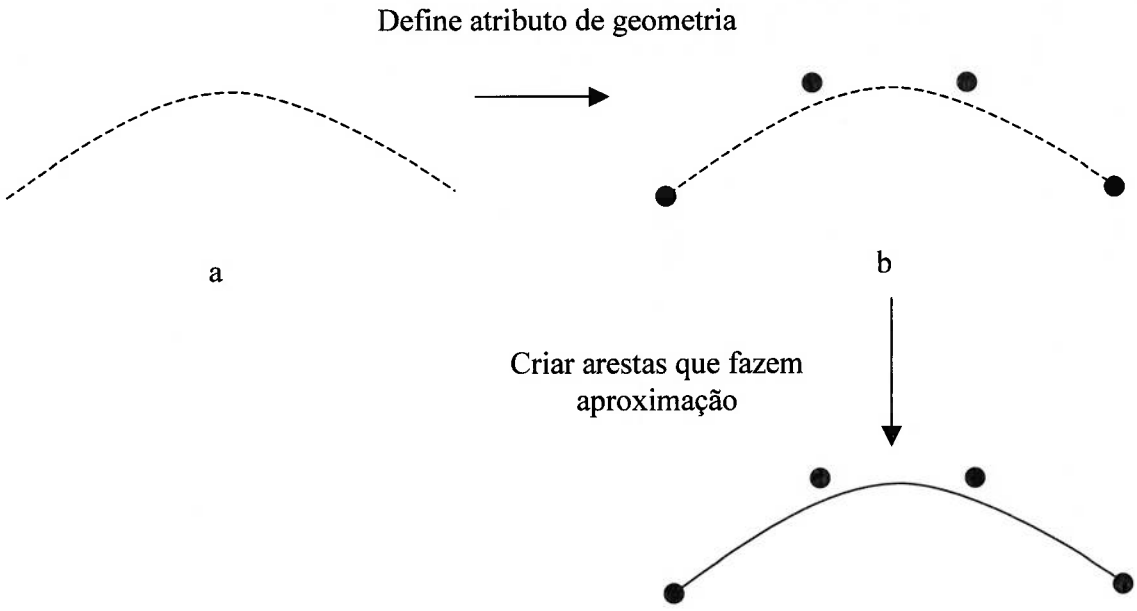


Figura 39. Processo que cria arestas de aproximação de uma curva.

adaptada para representar curvas em modeladores de sólido B-Rep está ilustrada na Figura 38.

Segundo esta representação, para representarmos uma curva de Bézier, é necessário inicialmente definir os seus pontos de controle, em seguida um operador determina o valor das coordenadas dos vértices das arestas que aproximam a curva de Bézier. Ao modificarmos um ponto de controle da curva de Bézier, é relativamente simples definir um algoritmo que atualize os valores das coordenadas dos vértices das arestas que aproximam a curva de Bézier (vide Figura 39).

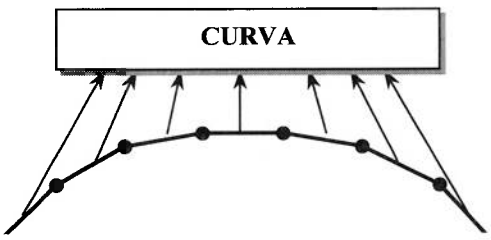


Figura 40. Arestas com ponteiros para sua curva.

Esta estrutura associa a cada meia-aresta um atributo de curva, de modo que elas indicam se são uma aproximação de uma curva, ou não, e se forem uma aproximação é possível saber de qual curva (vide Figura 40). Também foi considerado que toda curva é aproximada por uma seqüência de meia-arestas no circuito de contorno do laço definindo uma lista ligada de modo que o seu acesso e a definição de algoritmos ficam facilitados.

5.2 Operadores de Euler para Manipular Curvas

É conhecido que os Operadores de Euler podem criar e remover elementos topológicos como faces, arestas e vértices dos sólidos. Os operadores de Euler apresentados no capítulo anterior não permitem modificar características geométricas, assim será necessário definir novos operadores de Euler para este fim (por exemplo, a operação para converter uma linha reta em uma curva). Os operadores definidos são os seguintes :

- **ADDPP (Add Point to Polyline):** acrescenta um ponto de controle à curva.
- **REMPP (Remove Point from Polyline):** remove um ponto de controle da curva.
- **MCI (Make Curve Information):** é uma operação para associar uma curva a uma aresta.
- **KCI (Kill Curve Information):** é a operação inversa do MCI que remove a curva

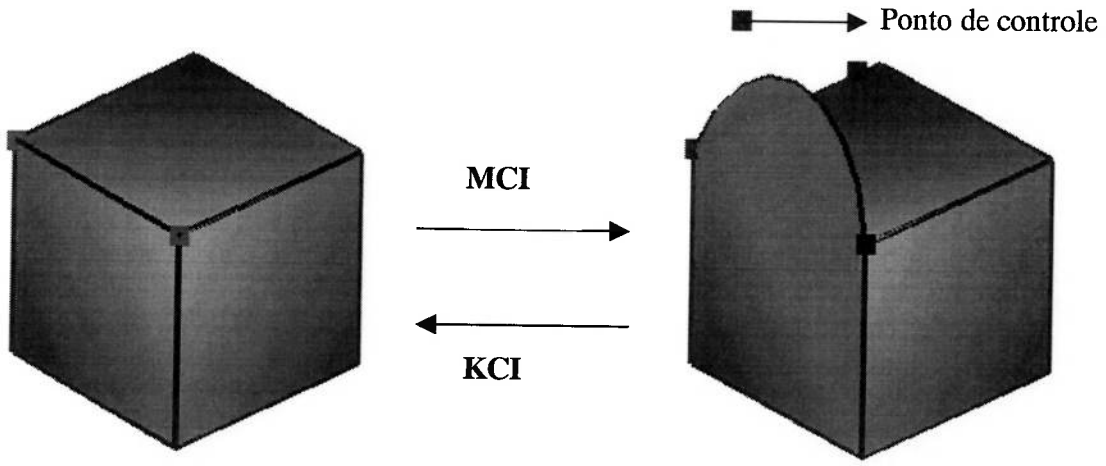


Figura 41. Operações para converter uma aresta reta em uma curva e a sua inversa.

do sólido. A Figura 41 mostra um exemplo de remoção de uma curva do sólido (sem construção de superfície).

- **MCP (Move Control Point):** é uma operação para transladar o n-ésimo ponto de controle de uma curva por um vetor. A operação inversa também é MCP (vide Figura 42). Um fato é que os vértices da aresta correspondem a vértices sobre a curva associada a esta aresta. Assim, existe uma duplicação de informação que necessita ser mantida de modo consistente. Como os operadores de Euler trabalham em um nível básico, a consistência destas informações é garantida pela existência de um operador de nível mais elevado que garante que ao ser feita uma modificação no vértice da aresta, a mesma modificação deve ocorrer para o ponto de controle e vice-versa. A Figura 43 ilustra este tipo de operador em funcionamento, ambos as vértices da aresta e da curva foram movimentados. Para verificar se todo o sistema está funcionando corretamente será utilizada uma função que verifica se as informações entre arestas e curvas estão consistentes.

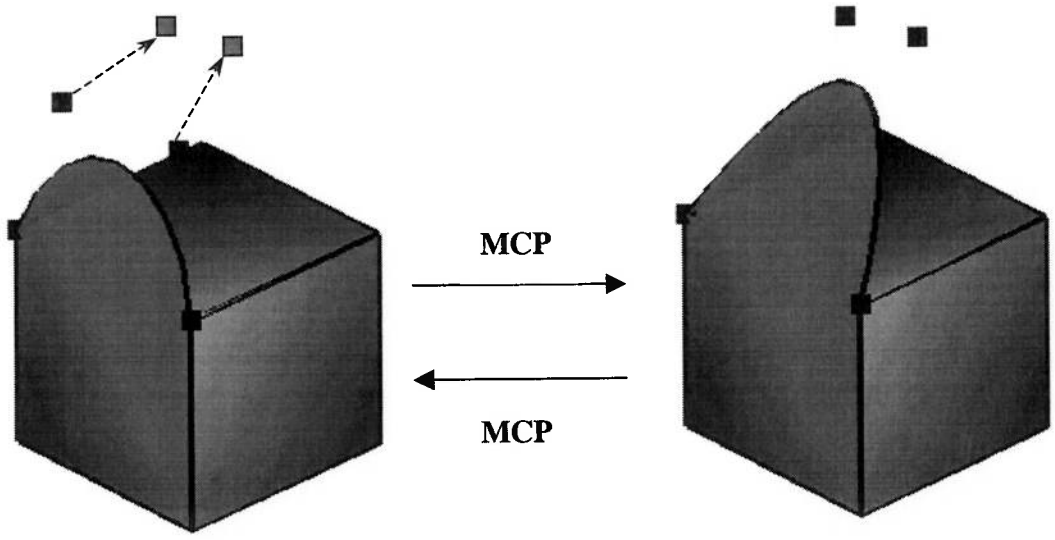


Figura 42 Operação para transladar o n-ésimo ponto de controle de uma aresta curva por um vetor e a sua inversa.

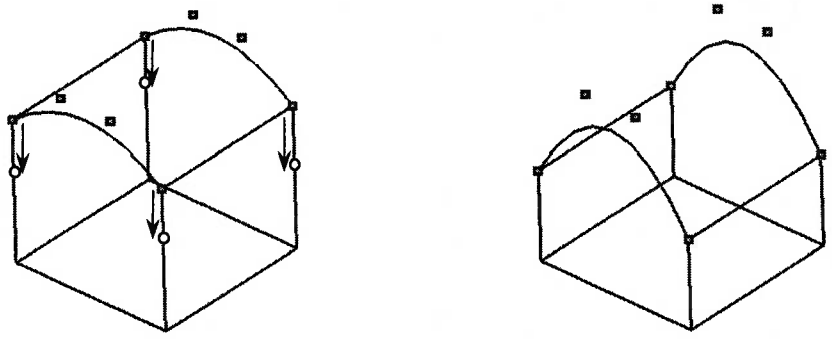


Figura 43 Rejustamento quando pontos de controle sobre os vértices modificados.

5.3 Curvas Como Atributos de Meia-arestas

Nenhum autor propôs associar as curvas às meia-arestas. Em verdade, todos comentam que as curvas devem ser associadas às arestas. Talvez, porque na maioria dos casos, as duas meia-arestas de uma mesma aresta possuem a mesma curva associada. Devemos ver, então, em qual situcação isto não ocorre. Considere a Figura 44 , em que o operador MEF foi aplicado.

A face f1 foi dividida em duas, e a aresta que contém as meia-arestas he1 e he2 foi dividida em duas. As meia-arestas he1s e he1z estão associadas às novas curvas. A união

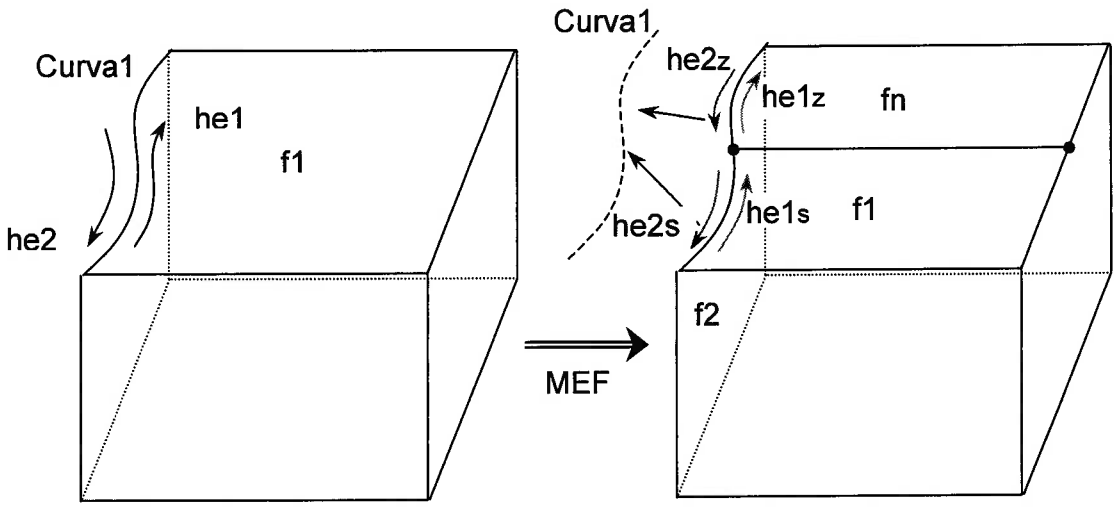


Figura 44 Associação de curvas as meia-arestas, exemplo de aplicação do operador MEF.

destas duas curvas define a curva original. Mas as meia-arestas he2z e he2s estão associadas à curva original, como se nenhuma divisão tivesse ocorrido. Assim, ao contarmos o número de curvas que definem a face f2, teremos quatro curvas. Se a curva fosse associada à aresta, como proposto pelos outros autores, a curva original seria obrigatoriamente dividida e cinco curvas estariam definindo a face f2. Neste caso a face f2 deve ser dividida para que todas as suas subfaces tenham quatro lados, e ao fazermos isto, criaremos novas faces com cinco lados. Assim ao associarmos a curva à meia-aresta, estaremos minimizando esta efeito.

5.4 Estrutura de Dados para Representar Superfícies

Inicialmente discutiremos sobre a função do elemento face apresentado na Figura 36. A nossa principal proposta é desenvolver duas principais funções que o elemento face exerce para facilitar a definição de novos algoritmos. A primeira função é o de representar o fechamento do sólido, topologicamente falando. A segunda função é representar a geometria de fechamento do sólido. Aparentemente estas duas funções são idênticas. Qualquer abertura pode ser fechada topologicamente por uma face. Entretanto, nem toda abertura pode ter sua geometria associada diretamente.

Assim, um fator limitante é a representação geométrica das faces, que requer que a superfície tenha quatro lados. Assim, por limitações matemáticas, é conveniente dividir uma face com mais que quatro lados em várias superfícies de quatro lados. Este é o conflito entre representar a topologia do modelo e representar a geometria do modelo.

A Figura 45 ilustra a estrutura de dados proposta. Estamos propondo que a face possua um atributo de superfície contendo as informações sobre a geometria da face. O atributo da superfície pode conter uma ou mais faces que contém laços, meia-arestas e vértices. Caso a face do sólido tenha quatro arestas como contorno, então a superfície associada a esta face terá apenas uma face. Em outra situação, onde a face do sólido tenha n lados ($n \neq 4$), então a superfície associada a esta face terá n faces, cada uma com quatro lados.

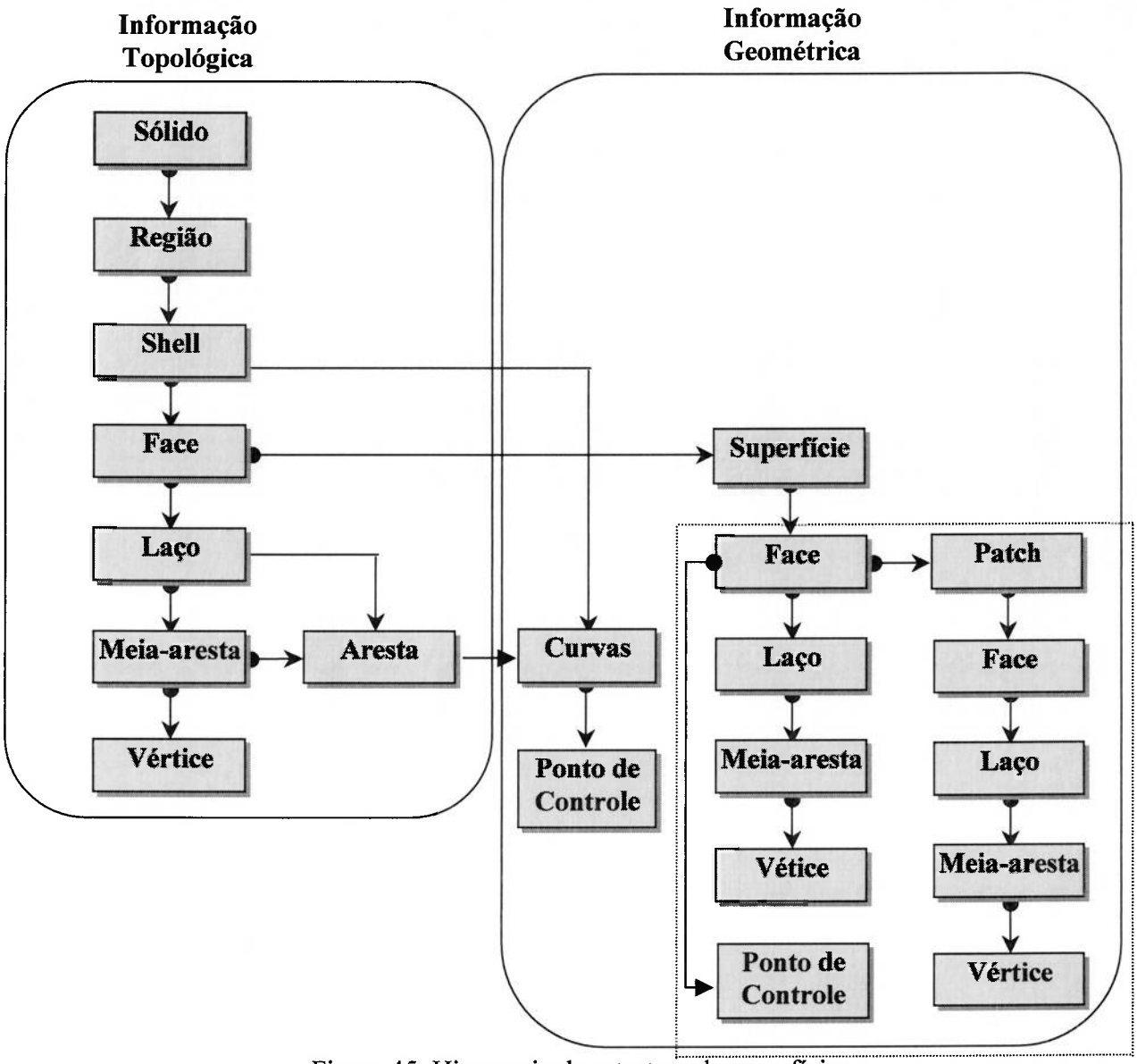


Figura 45. Hierarquia da estrutura de superfícies.

Deste modo fica a pergunta, qual a diferença entre as faces de um sólido e as faces da superfície? As faces do sólido podem ter qualquer número de lados, e representam as modificações feitas pelo usuário. Ou, seja as intenções do usuário estão representadas nesta estrutura. As faces da superfície possuem todas quatro lados, e simultaneamente, todas possuem a sua geometria representada. Assim, caso a face da estrutura original do sólido não possua quatro lados, ela é dividida por um algoritmo de modo que surja um número conveniente de faces para que a sua geometria possa ser definida. Deste modo,

as faces da estrutura superfície nem sempre representam as intenções do usuário, pois em sua maioria o modo de realizar a divisão, e o algoritmo de atribuir geometria são automáticas.

Existe uma terceira estrutura denominada patch que está associada a cada face da estrutura superfície. Esta terceira estrutura existe para criar uma aproximação poliedral da geometria definida na estrutura superfície. Deste modo é possível criar algoritmos que manipulam aproximações poliedrais para calcular propriedades de massa e visualizadores tridimensionais, uma vez que nem todas as superfícies podem ser visualizadas diretamente em aceleradores gráficos tridimensionais, como o Opengl. Em nosso caso particular, as superfícies de Gregory devem ser aproximadas, por um conjunto de superfícies poliedrais para que possam ser visualizadas. A Figura 46 ilusta um exemplo de patches que aproximam uma superfície.

Deste modo, a face do sólido não possui geometria associada, mas possui o seu contorno definido. Ou seja, todas as curvas de seu contorno já estão definidas. As faces da estrutura superfície possuem geometria definida e todas possuem quatro lados. A face da estrutura patch representa a aproximação poliedral da geometria definida na estrutura superfície.

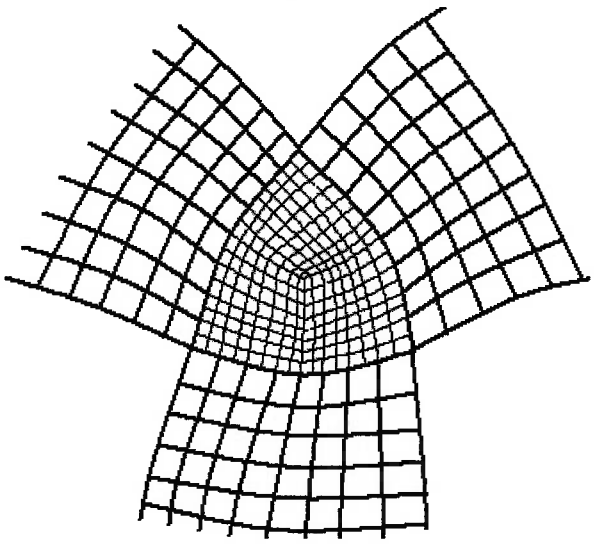


Figura 46 Exemplo de superfícies desenvolvidas.

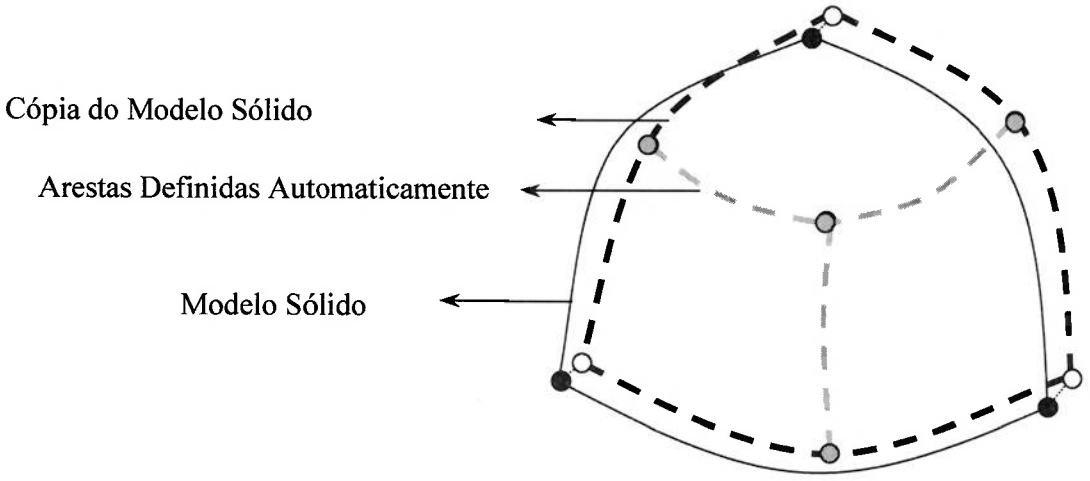


Figura 47 Uma face triangular associadas pelo três "patch".

Deste modo, separamos o mecanismo de detalhamento de curvas do mecanismo de detalhamento de superfícies. Sendo possível que ambas sejam detalhadas com níveis distintos, conforme a real necessidade (Vide Figura 47).

5.5 Operadores de Euler para Manipulação de Superfícies

Para manipular a informação de geometria no modelador de sólidos, desenvolvemos alguns operadores que podem construir e modificar superfícies. Eles serão descritos neste capítulo.

5.5.1 Operador de Euler

- **ADDPS (Add Point to Surface):** acrescenta um ponto de controle a uma superfície. Por exemplo, uma superfície de Bézier tem 16 pontos de controle, inicialmente o contorno da superfície é definido (12 pontos de controle). Para completar o definição da superfície, é necessário aresentar os 4 pontos de controle internos por este operador.
- **REMPS (Remove Point from Surface):** é o operador inverso ao ADDPS.

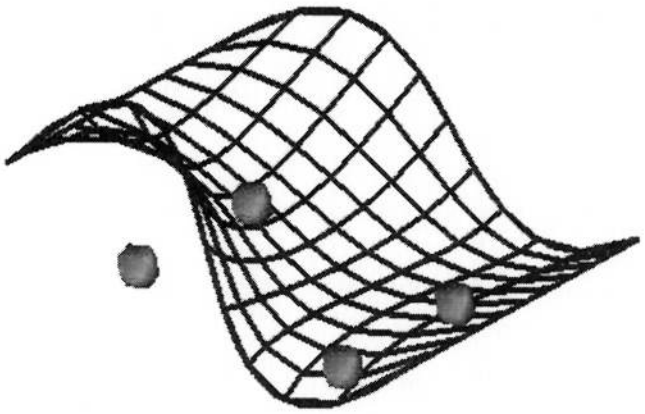


Figura 48 Exemplo de superfície de Bézier.

- **MSI (Make Surface Information):** é o operador para acrescentar uma superfície a uma face A Figura 48 mostra uma superfície que foi definida e acrescentada a uma face.
- **KSI (Kill Surface Information):** é a operador inverso ao MSI que remove a superfície do sólido.
- **TSV (Transform Surface form to a Vector):** é o operador para transformar um ponto de controle de uma superfície. A Figura 49 ilustra um exemplo de aplicação deste operador.

5.5.2 Operadores de Baixo Nível na Estrutura Superfície

- **MSuVF (Make Surface Vertex Face):** é um operador semelhante a MVSF (Make Vertex Solid Face). Ele é utilizado para definir o primeiro vértice e a primeira face na estrutura superfície.
- **MSuEV (Make Surface Edge Vertex):** operador semelhante a MEV (Make Edge Vertex). Ele é utilizado para definir uma aresta e um vértice na estrutura superfície.
- **MSuEF (Make Surface Edge Face):** operador semelhante a MEF (Make Edge Face). Ele é utilizado para definir uma aresta e uma face na estrutura superfície.

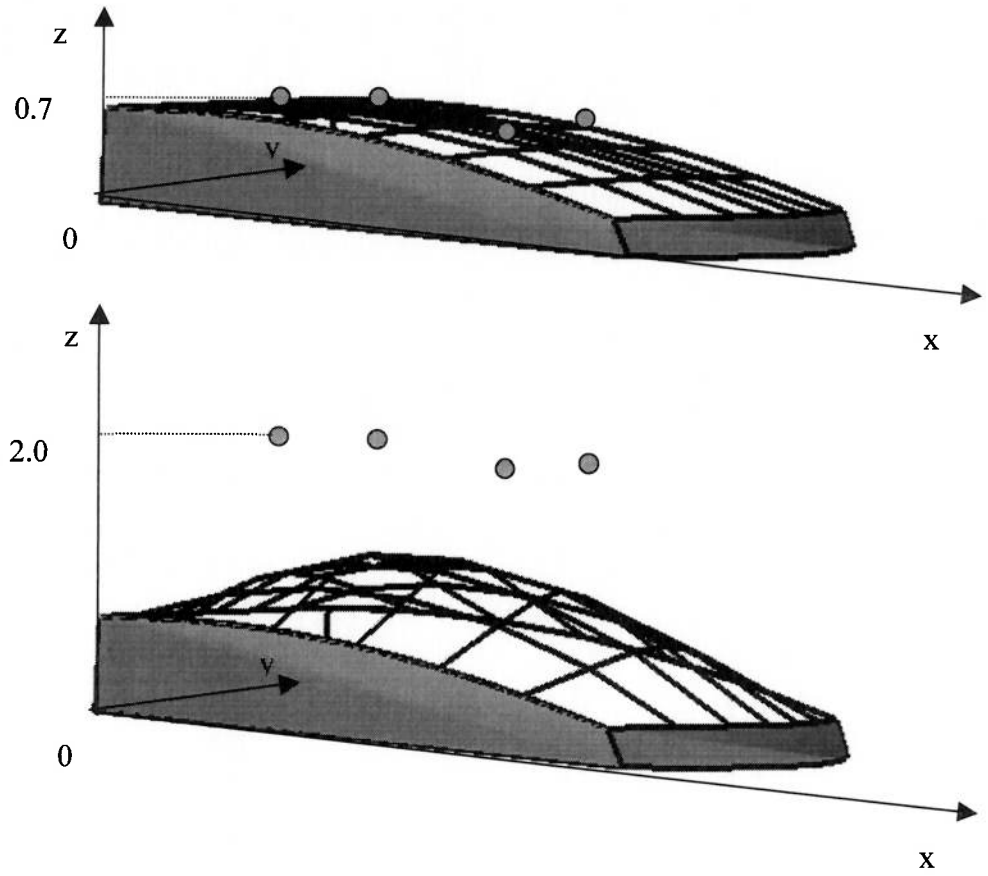


Figura 49 A forma da superficie mudou depois os ponto de controle foram modificados.

- **KSuVF (Kill Surface Vertex Face):** operador semelhante a KVSF (Kill Vertex Solid Face). Ele é utilizado para remover o último vértice e a última face da estrutura superfície.
- **KSuEV (Kill Surface Edge Vertex):** operador semelhante a KEV (Kill Edge Vertex). Ele é utilizado para remover uma aresta e um vértice da estrutura superfície.
- **KSuEF (Kill Surface Edge Face):** operador semelhante a KEF (Kill Edge Face). Ele é utilizado para remover uma aresta e uma face da estrutura superfície.

5.5.3 Operadores de Baixo Nível na Estrutura Patch:

Implementamos um conjunto de operadores de baixo nível para manipular a estrutura patch.

- **MAI (Make Approximation Information):** é o operador para acrescentar a informação de aproximação a uma face na estrutura patch.
- **KAI (Kill Approximation Information):** é o operador para remover a informação de aproximação de uma face na estrutura patch.
- **MApVF (Make Surface Vertex Face):** é um operador semelhante a MVSF (Make Vertex Solid Face). Ele é ultilizado para definir o primeiro vértice e a primeira face na estrutura patch.
- **MApEV (Make Surface Edge Vertex):** é um operador semelhante a MEV (Make Edge Vertex). Ele é utilizado para definir uma aresta e um vértice na estrutura patch.
- **MApEF (Make Surface Edge Face):** é um operador semelhante a MEF (Make Edge Face). Ele é utilizado para definir uma aresta e uma face na estrutura patch.
- **KApVF (Kill Surface Vertex Face):** é um operador semelhante a KVSF (Kill Vertex Solid Face). Ele é utilizado para remover o último vértice e a última face da estrutura patch.
- **KApEV (Kill Surface Edge Vertex):** é um operador semelhante a KEV (Kill Edge Vertex). Ele é utilizado para remover uma aresta e um vértice da estrutura patch.
- **KApEF (Kill Surface Edge Face):** é um operador semelhante a KEF (Kill Edge Face). Ele é utilizado para remover uma aresta e uma face da estrutura patch.

5.5.4 Operadores de Alto Nível

Estes operadores são conjuntos de operadores de Euler. Alguns são apresentados abaixo:

- **CopyFace:** Este operador constrói uma nova face a nível de superfície idêntica à que existe ao nível do modelo sólido. Primeiro, um Operador de Euler MSuVF será executado para construir um novo sólido na superfície, depois utiliza MSuEV e MsuEF para criar uma nova face com as mesmas informações de topologia (Figura 50) .

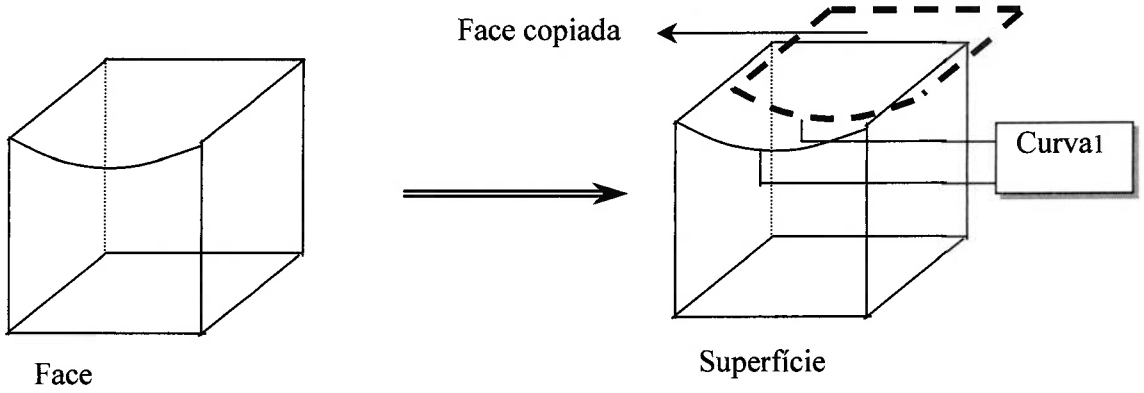


Figura 50. Operador de alto nível copyFace.

- **Render:** Cria um malha de polígonos que aproximam a superfície, facilitando a sua visualização bem com o cálculo de propriedades de massa. Inicialmente as curvas de contorno são definidas. Segundo, os pontos internos são calculados (Figura 51) de acordo com a equação (17) para em seguida serem aplicados os operadores MsuEV e MsuEF (Figura 52) .

- **TransformThreeSidedIntoFourSidedFaces:** Este operador cria três faces com quatro lados segundo o algoritmo proposto por Chiyokura e Kimura [4]).

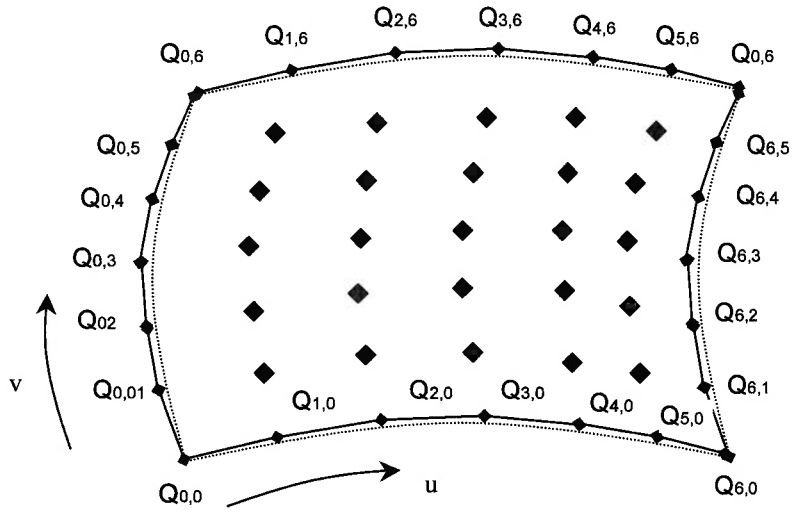


Figura 51. Pontos internos de superfície de Bézier.

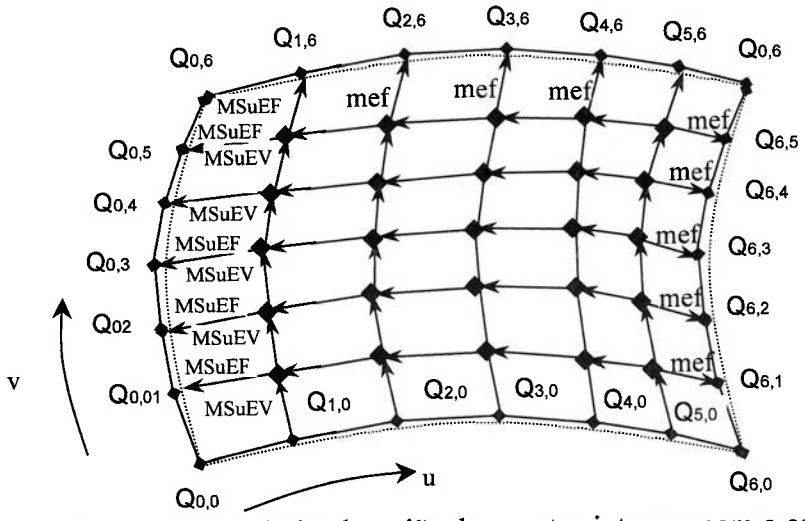


Figura 52 Criação da malha poliedral pelo união dos pontos internos com o contorno da superfície.

5.6 Considerações Finais

Esta estrutura proposta permite que a superfície seja desenvolvida com nível de detalhamento independente do nível de detalhamento escolhido para desenvolver as curvas. Ou seja o nosso objetivo original foi atingido.

6 Conclusão

Neste trabalho apresentamos uma estrutura com três níveis de detalhamento para suportar simultaneamente e de foma independente o mecanismo de criar arestas lineares que aproximam uma curva e o mecanismo de criar faces planas que aproximam uma superfície. Ambas baseadas nas propostas apresentadas por Ueda [17] e Turner [16].

Este conceito se baseia principalmente no fato de que a face representada no modelo sólido possui duas funções: representar o fechamento topológico do sólido e representar a uma forma geométrica. Estas duas funções não são totalmente consistentes, pois em algumas situações é necessário dividir a face de forma conveniente para que a geometria seja associada. Assim, com a estrutura de dados proposta, separamos os elementos que foram criados automaticamente para satisfazer esta necessidade, dos elementos criados satisfazendo as intenções do usuário.

Nesta nova estrutura, propusemos que a curva seja associada à meia-aresta. Deste modo evitaremos a necessidade de provocar divisões em cascata. A figura 53(a)

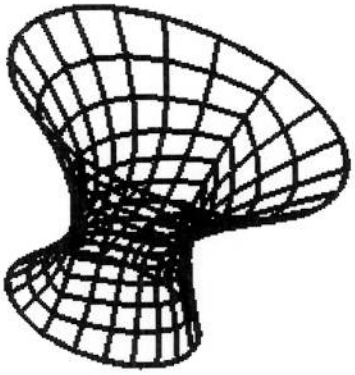


Figura 53 (a) Um sólido com seus superfícies

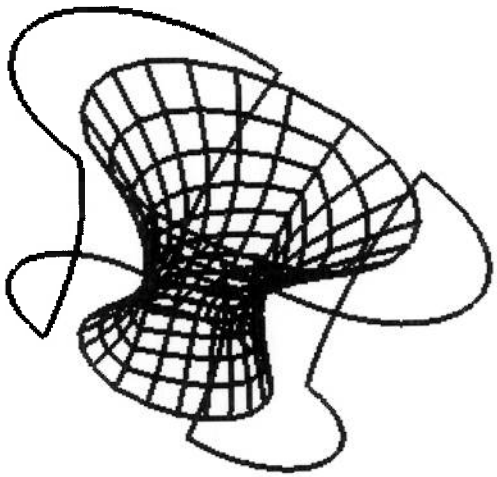


Figura 53 (b) Contorno do sólido e seus superfícies

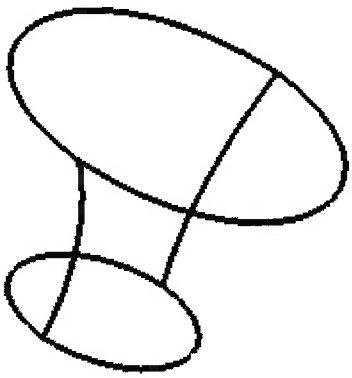


Figura 53 (c) Contorno do sólido

apresenta um exemplo de superfície desenvolvida com nível de detalhamento distinto do aplicado para as curvas (vide Figura 53 (c)).

Resta, entretanto, analizar de forma mais aprofundada as implicações de utilizar esta estrutura com Operadores Booleanos. Neste caso, será necessário também criarmos algoritmos para determinar a curva de intersecão entre superfícies.

7 Referência

[1] P.E. Bezier, **Numerical Control-Mathematics and Applications**, John Wiley and Sons, London, 1972.

[2] H.Chiyokura, **An Extended Rounding Operation for Modeling Solids with Free-Form Surfaces**, Computer Graphics, 249-268, 1987.

[3] H.Chiyokura, **Solid Modelling with DesignBase: Theory and Implementation,** Addison-Wesley, Reading, Massachusetts, 1988.

[4] H Chiyokura e F Kimura, **A New Surface Interpolation Method for Irregular Curve Models,** Computer Graphics Forum 3, 209-218, 1984.

[5] H.Chiyokura e F.Kimura, **Design of Solids with Free Form Surfaces**. Computer Graphics, vol. 17, n. 3, pp. 289-298, 1983.

[6] S.A.Coons, **Surfaces for Computer Aided Design of Space Forms**, MIT Project MAC TR-41, June 1967.

[7] G.Farin, **Curves and Surfaces for Computer Aided Geometric Design** Academic Press, INC San Diego, California, 1991.

[8] M.Hosaka, **Modeling of Curves and Surfaces in CAD/CAM** Computer Graphics T385.H6958 1992.

[9] K.Kado, A.Shimamura e K.Inoda, **A Surface Interpolating Method for a Car-Styling Designer's CAD Work Tool,** CG International, June 1992.

[10] S.Kawabe e M.Taguchi, **A Solid Modeling System with Modularized Structure**, Computer Applications in Production and Engineering, North Holland, 199-214, 1987.

[11] M.MÄNTYLÄ, **An Introduction To Solid Modeling**, Computer Science Press, Rockville, Maryland.

[12] D.F.Rogers e J.A.Adams, **Computer Graphics,** McGraw-hill, 1976.

[13] T.Takamura, **Subdivision Method of the Gregory Patch and its Application**, Ricoh Technical Report, 18:4-9, October 1988.

[14] H.Toriya. e H.Chiyokura, **3D CAD Principles and Applications,** Springer-Verlag, 1991.

[15] M.S.G.Tsuzuki, **Modelagem de Sólidos: Modelos Limitantes (B-Rep),** Departamento de Engenharia Mecânica da EPUSP, 1991.

[16] J.U.Turner, IBM **Accurate Solid Modeling Using Polyhedral Approximations,** IEEE, 1988.

[17] R.Y.Y.Ueda, **Representando curvas e superfícies em modeladores poliedrais B-Rep por meio de atributos associados aos elementos aresta e face,** Departamento de Engenharia Mecânica da EPUSP, 1996.