DETERMINANDO CURVAS DE INTERSECÇÃO ENTRE SUPERFÍCIES DE GREGORY

Marcos de S. G. Tsuzuki
Universidade de São Paulo, Escola Politécnica, Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos, 05508-900, São Paulo, SP, Brasil. E-mail: mtsuzuki@usp.br

Resumo

A determinação das curvas de intersecção entre duas superfícies é um algoritmo básico para várias aplicações, tais como: operação de suavização entre superfícies, operações booleanas entre sólidos e superfícies, geração da trajetória da ferramenta para usinagem de superfícies, entre outras. Existem diversos métodos para solucionar tal problema, com abordagens restritas a alguns tipos de superfícies e possuindo diversas limitações quanto ao tratamento de casos especiais. É possível demonstrar que uma superfície de Gregory com vinte pontos de controle é equivalente a uma superfície racional de Bézier de oito por oito pontos de controle. Esta propriedade nos permite controlar de modo mais intuitivo o interior da superfície e também definir a continuidade entre superfícies adjacentes segundo um plano de tangência. Entretanto, este tipo de superfície está disponível em apenas alguns poucos sistemas de CAD, pois o grau elevado dos seus polinômios torna o seu processamento proibitivo. Além do que, não existe algoritmo para realizar a subdivisão do seu espaço paramétrico. Neste trabalho apresentaremos como este problema foi solucionado.
Palavras-chave: Modelagem Geométrica, CAD/CAM, Computação Gráfica, Superfícies de Gregory, Curva de Intersecção

1. INTRODUÇÃO

Os estudos sobre a modelagem geométrica utilizando-se superfícies de forma livre, começaram logo no início da história dos sistemas CAD. A história das superfícies de forma livre é mais longa que a de Modelagem de Sólidos. Entretanto, a determinação das curvas de intersecção entre duas superfícies de forma livre é um dos problemas mais estudados, principalmente na última década.

Uma das primeiras propostas para a determinação das curvas de intersecção entre superfícies de forma livre foi feita para superfícies de Bézier e se baseava na propriedade de subdivisão dos retalhos de Bézier (Lasser, 1986). Esta proposta apresentava três passos: subdivisão das duas superfícies onde os paralelepípedos envoltórios se interceptam até que o retalho resultante da subdivisão possa ser considerado plano; determinação dos pontos de intersecção entre os retalhos que se interceptam; e coleção dos pontos de intersecção para definir a curva de intersecção.

As propostas mais recentes dividem o problema em dois passos: determinação de pontos iniciais para realizar o traçamento da curva de intersecção; e traçamento da curva de intersecção. O traçamento da curva de intersecção se baseia na determinação do vetor tangente da curva de intersecção no ponto em estudo, o vetor tangente é equivalente ao produto vetorial entre as normais das duas superfícies. Entretanto, pode existir a possibilidade

em que as duas normais são colineares, impossibilitando definir uma direção a ser seguida para o traçamento da curva de intersecção (Wang et. al., 1991; Kriezis et al., 1992; Grandine et al., 1997; Hu et al., 1997).

Podemos classificar a curva de intersecção em dois possíveis casos: curva de intersecção aberta e curva de intersecção fechada. A curva de intersecção aberta é aquela que intersecciona pelo menos uma das curvas de contorno de uma das duas superfícies. A curva de intersecção fechada é aquela que não intersecciona nenhuma das curvas de contorno das duas superfícies. Um caso particular de curva de intersecção fechada é aquele em que as duas superfícies se interseccionam em apenas um ponto (Sedeberg et al., 1989; Grandine et al., 1997).

2. SUPERFÍCIE DE GREGORY

Gregory estendeu a proposta da superfície de Coons de modo que cada um dos vetores tangentes cruzados associados às curvas de contorno possam ser especificados de forma independente. Chiyokura e Kimura (1983) aplicaram a mesma estenção à superfície cúbica de Bézier. A seguir discutiremos algumas das propriedades da superfície de Gregory na forma de superfície cúbica de Bézier. Como ilustrado na Figura 1, 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:

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

onde:

$$Q_{ij}(u,v) = P_{ij0} = P_{ij1} \qquad (2)$$

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

$$\begin{aligned} Q_{11}(u,v) &= \frac{u\cdot P_{110} + v\cdot P_{111}}{u+v} \\ Q_{21}(u,v) &= \frac{(1-u)\cdot P_{210} + v\cdot P_{211}}{(1-u)+v} \\ Q_{12}(u,v) &= \frac{u\cdot P_{120} + (1-v)\cdot P_{121}}{u+(1-v)} \\ Q_{22}(u,v) &= \frac{(1-u)\cdot P_{220} + (1-v)\cdot P_{221}}{(1-u)+(1-v)} \end{aligned} \qquad (3)$$

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 especificadas de forma independente:

$$\begin{aligned} Q_v(u,0) &= 3\cdot\sum_{i=0}^{3} B_i(u)\left(P_{i10} - P_{i00}\right) \quad Q_u(0,v) = 3\cdot\sum_{i=0}^{3} B_i(v)\left(P_{1i0} - P_{0i1}\right) \\ Q_v(u,1) &= 3\cdot\sum_{i=0}^{3} B_i(u)\left(P_{i30} - P_{i20}\right) \quad Q_u(1,v) = 3\cdot\sum_{i=0}^{3} B_i(v)\left(P_{3i0} - P_{2i1}\right) \end{aligned} \qquad (4)$$

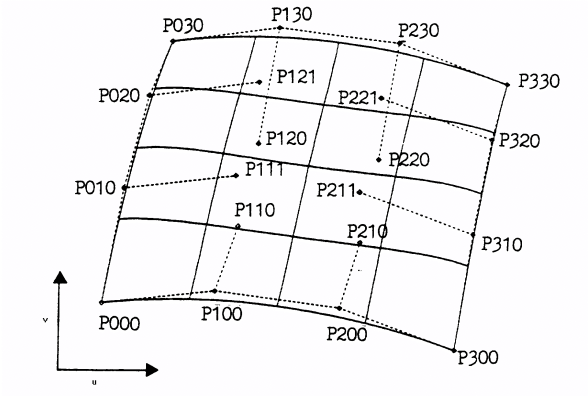

Figura 1. Superfície de Gregory.

3. ESTUDO DO PROBLEMA

Vamos definir o problema da determinação das curvas de intersecção entre superfícies da seguinte forma: dadas $\mathbf{F}(u,v)$ e $\mathbf{G}(s,t)$, superfícies quaisquer parametrizadas no domínio $[0,1]^2$, deseja-se obter os pares (u,v) e (s,t) tais que:

$$\mathbf{F}(u,v) - \mathbf{G}(s,t) = 0 \qquad (5)$$

A expressão acima compreende, na verdade, um sistema de 3 equações (uma para cada coordenada do espaço) e 4 incógnitas, o que torna extremamente complexa uma solução direta e controlada para o mesmo. O conjunto solução pode consistir em nenhuma, uma ou mais curvas isoladas. Neste trabalho, para traçar as curvas de intersecção entre as superfícies, será utilizado o método da caminhada (Hu et al, 1997), que a partir do ponto inicial da curva de intersecção determina seus pontos seguintes, os quais satisfazem a equação (5), através de procedimentos numérico-geométricos.

Pelo método apresentado por Hu et al. (1997), podem-se encontrar pontos iniciais para as curvas abertas (aquelas cujos pontos inicial e final no espaço paramétrico são distintos – é interessante notar que estes pontos podem corresponder ao mesmo ponto no espaço tridimensional) pertencentes ao conjunto solução do problema. Para tal, iguala-se, em (5), cada parâmetro (u, v, s e t) a zero e, depois, a um. Assim, após cada uma destas substituições, resta em (5) um sistema $3x3$ que pode ser resolvido diretamente, conforme será detalhado em seção à frente. Feito isto, tem-se os pontos das extremidades de todas as curvas abertas do conjunto solução, uma vez que as mesmas sempre começam e terminam nas bordas dos domínios paramétricos de uma das superfícies do par em questão.

Todavia, podem existir curvas de intersecção que sejam fechadas dentro do domínio paramétrico, de modo a não cruzarem as bordas dos mesmos. São os chamados laços de intersecção. Para traçar os laços, os mesmos devem ser quebrados em curvas abertas, identificando-se, então, os pontos das extremidades das mesmas. Mas para que eles possam ser quebrados, os laços devem ser, primeiro, detectados. Sedeberg et al. (1989) demonstram que se duas superfícies de continuidade ao menos C^1 se interceptam ao longo de um laço, então existe uma reta perpendicular a ambas superfícies e que atravessa o interior desta curva. O par formado pelos pontos, um de cada superfície, nos quais esta reta se intercepta é chamado par de pontos com normais colineares. Ou seja, o vetor normal de uma superfície no primeiro ponto é colinear ao da outra superfície no segundo ponto. Neste caso, Sedeberg et al. (1989) demonstram que é necessário solucionar um sistema $4x4$ para determinarmos os

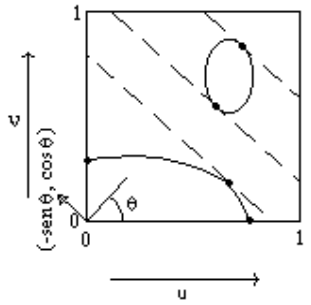

Figura 2. Exemplo de determinação dos pontos de virada das curvas de intersecção através das linhas de varredura.

pontos com normais colineares sobre cada domínio paramétrico. A partir de cada ponto encontrado, é possível subdividir sucessivamente o domínio paramétrico de modo a quebrar as curvas fechadas de intersecção em curvas abertas cujos pontos iniciais serão os pontos de cruzamento das mesmas com as bordas de cada novo subdomínio. Para o caso de superfícies de Gregory, não poderemos utilizar esta proposta, pois não existe algoritmo que permita subdividir o seu domínio paramétrico.

Grandine et al. (1997) apresentaram outra proposta para determinar os pontos iniciais das curvas fechadas de intersecção. Neste caso, faz-se uma varredura (no espaço paramétrico) com linhas paralelas inclinadas de θ em relação à direção do parâmetro v, conforme ilustra a Figura 2. Desta forma, o vetor $(-\text{sen}\theta, \cos\theta)$ será paralelo às linhas de varredura, sendo $0 \le \theta \le \pi/2$. Isto permite determinar os pontos em que as curvas de intersecção fazem o retorno em relação às bordas, eles se caracterizam pelo fato de que a tangente à curva neste ponto é paralela à direção das linhas. Estes pontos são chamados por pontos de virada. No exemplo da Figura 2, mostra-se como as linhas de varredura irão encontrar os pontos de virada tanto das curvas abertas quanto dos laços de intersecção, detectando assim a presença destes últimos e fornecendo pontos iniciais para os mesmos. Conforme detalhado em Gradine et al. (1997), os pontos de virada são dados pelas soluções do seguinte sistema $4x4$:

$$\begin{aligned} &\mathbf{F}(u,v) - \mathbf{G}(s,t) = 0 \\ &(\mathbf{F}_u \,\text{sen}\theta - \mathbf{F}_v \cos\theta) \circ \mathbf{G}_s \times \mathbf{G}_t = 0 \end{aligned} \qquad (6)$$

Nota-se que o conjunto solução deste sistema deve conter também todos os pontos em que as curvas de intersecção se encontram, uma vez que as mesmas só se cruzam em pontos nos quais o gradiente desaparece - pontos críticos. Em última análise, a idéia principal deste método é encontrar pontos iniciais para cada curva do conjunto solução do problema de intersecção. Com isto, simplifica-se a solução em si, posto que se adiciona uma equação à expressão original do problema, com 3 equações e 4 incógnitas, originando um sistema $4x4$. Assim, nota-se que devem ser escolhidos métodos numéricos adequados para resolver os sistemas de polinômios cujas soluções irão representar os pontos iniciais desejados. Para realizar tal tarefa, utiliza-se o método do poliedro projetado que será apresentado a seguir.

4. RESOLVENDO SISTEMAS POLINÔMIAIS PELO MÉTODO DO POLIEDRO PROJETADO

Resolver sistemas de polinômios significa, em termos algébricos, determinar todas as n-uplas $\mathbf{x} = (x_1, x_2, \cdots, x_n)$ de modo que

$$f_1(\mathbf{x}) = f_2(\mathbf{x}) = \ldots = f_n(\mathbf{x}) = 0 \qquad (7)$$

com $x \in [0,1]$[1] e f_k sendo um polinômio com n variáveis. O método do poliedro projetado foi apresentado por Sherbrooke & Patrikalakis (1993). Este algoritmo baseia-se na idéia de converter cada polinômio $f_i(x)$ em um polinômio equivalente em base de Bernstein. Neste sentido, é possível realizar uma analogia entre os sistemas polinomiais e a determinação da intersecção de curvas de Bézier pela intersecção dos *convex hulls* de seus pontos de controle. A seguir, apresentaremos uma descrição suscinta do algoritmo para facilitar a sua compreensão. Sua primeira etapa é transformar cada equação f_k para a base de Bernstein, o que significa:

$$f_k(\mathbf{x}) = \sum_{i_1=0}^{d_1^{(k)}} \sum_{i_2=0}^{d_2^{(k)}} \cdots \sum_{i_n=0}^{d_n^{(k)}} \left[w_{i_1 i_2 \cdots i_n}^{(k)} \right] \cdot B_{i_1}^{d_1^{(k)}}(x_1) \cdot B_{i_2}^{d_2^{(k)}}(x_2) \cdots B_{i_n}^{d_n^{(k)}}(x_n) \qquad (8)$$

onde $d_i^{(k)}$ é o grau da variável x_i em f_k. Agora, vamos redefinir o problema da equação (7) como a tarefa de determinar a intersecção dos grafos[2] de cada f_k (cada grafo pode ser visto como uma hipersurperfície em $\Re^{n+1}$) com o hiperplano $x_{n+1} = 0$. Cada grafo de f_k é dado por

$$\mathbf{F}_k(\mathbf{x}) = (x_1, x_2, \cdots, x_n, f_k(\mathbf{x})) = (\mathbf{x}, f_k(\mathbf{x})) \qquad (9)$$

Já que cada grafo $\mathbf{F}_k$ também é uma hipersuperfície paramétrica, então é possível determinar os seus pontos de controle. Conseqüentemente, uma nova maneira de lidar com o problema de resolver o sistema da equação (7) seria determinar a intersecção entre as hipersuperfícies definidas por cada $\mathbf{F}_k$. Uma aproximação inicial para este problema seria determinar a intersecção entre os *convex hulls* que são formados pelos pontos de controle destas hipersuperfícies no espaço $\Re^{n+1}$. Além disto, é possível converter este problema de dimensão $(n+1)$ em n problemas bidimensionais projetando-se os pontos de controle de cada $\mathbf{F}_k$ em cada plano do espaço $\Re^{n+1}$ formado pela coordenada relativa à variável x_i e pela última coordenada da equação (9), relacionada com a função original f_k (ou seja, um plano para cada variável do sistema inicial da equação (7)). Desta maneira, os pontos de controle de cada $\mathbf{F}_k$ são projetados em n planos. É formado então, em cada plano, o *convex hull* bidimensional dos pontos de controle de cada $\mathbf{F}_k$. E dentro de cada plano, é determinada a intersecção entre todos os *convex hulls* com o segmento $[0,1]$ (que está relacionado com o domínio da variável x_i) da abscissa daquele plano. O resultado pode ser um segmento, um ponto ou nulo (pois todos os *convex hulls* projetados são polígonos convexos). Se for nulo, então não há nenhum valor para a variável deste plano que seja solução do sistema. Se o resultado é um ponto, seu valor (um real dentro de $[0,1]$) é uma solução para a variável no sistema da equação (7). E se resultado é um segmento, então ainda é possível que haja um ou mais pontos de solução para aquela variável do sistema. Isto significa que, para cada variável

---

[1] ou qualquer outro domínio finito $[a_i, b_i]$ o qual deve ser, então, mapeado no domínio $[0,1]$ através da transformação de variáveis $x_i' = a_i + x_i \cdot (b_i - a_i)$ em cada equação f_k.

[2] Seja $f : \Re^n \to \Re$ uma função de $\mathbf{x}$. Então, seu grafo é a função $F : \Re^n \to \Re^{n+1}$ definida por: $F(\mathbf{x}) = (\mathbf{x}, f(\mathbf{x}))$.

x_i, nós teremos diminuído a faixa de busca por soluções do domínio completo da variável inicial ($[0,1]$) para um segmento $[a_i, b_i]$. Assim, pode-se reduzir o box de soluções do domínio inicial total $[0,1]^n$ para o box formado pelos segmentos de intersecção encontrados:

$$S = [a_1, b_1] \times [a_2, b_2] \times \cdots \times [a_n, b_n] \quad (10)$$

Neste ponto, para cada k, definimos uma nova função f_k' de modo que:

$$f_k'(\mathbf{x}) = f_k(a_1 + (b_1 - a_1) \cdot x_1, a_2 + (b_2 - a_2) \cdot x_2, \ldots, a_n + (b_n - a_n) \cdot x_n) \quad (11)$$

Esta função será mapeada dentro do domínio $[0,1]^n$ da mesma maneira que f_k era mapeada no domínio da equação (7). Este processo é então repetido iterativamente até que sejam encontrados boxes de solução (como os da equação (10)) com todos os lados pequenos bastante para que o box seja visto como raiz isolada para as variáveis do sistema. Em outras palavras: em S (equação (10)), $\forall i$ tal que $1 \le i \le n$, deve ser verdadeiro que $(b_i - a_i) \le tol$. Esta *tolerância* define a precisão das soluções. Entretanto, após ter processado a intersecção dos *convex hulls* dentro dos planos, é possível que o valor $(b_i - a_i)$ de algumas variáveis em S (equação (10)) possa não ter diminuído significativamente de 1. Isto acontece quando há mais de uma solução no sistema da equação (7). Em tal caso, nós devemos dividir em dois o segmento da variável que não foi reduzida consideravelmente no passo anterior. A partir daqui, os dois boxes resultantes serão tratados como problemas independentes e separados.

5. METODOLOGIA ADOTADA

Este trabalho foi baseado no trabalho realizado por Faustini (1999) que determinava a curva de intersecção entre superfícies NURBS. Entretanto, esta primeira implementação solucionava apenas dois tipos de sistemas polinomiais: com três equações e três variáveis, e de quatro equações e quatro variáveis (que eram os tipos de sistemas presentes no problema em estudo). Ele utiliza uma implementação que fazia uso de números em ponto flutuante com dupla precisão e estava perdendo soluções, mesmo com sistemas de ordem relativamente baixa (potência 7). Portanto, desenvolveu-se uma nova implementação para o método do poliedro projetado. A seguir detalham-se os pontos principais da pesquisa:

- Adaptação do algoritmo do poliedro projetado para fazer uso da técnica de álgebra intervalar. É conhecido que a simples substituição do intervalo em algoritmos de ponto flutuante pode gerar como solução um intervalo exageradamente grande que não possui utilidade prática;
- Estudo do o algoritmo de conversão do polinômio em base de potência para base de Bernstein, que é uma possível fonte de imprecisão para o algoritmo do poliedro projetado;
- Estudo do algoritmo de conversão de variável, que em conjunto com o item anterior são as possíveis fontes de imprecisão para o algoritmo do poliedro projetado;
- O algoritmo da caminhada é o mesmo que o apresentado por Faustini (1999).

5.1. ARITMÉTICA INTERVALAR ARREDONDADA

A limitada representação inerente à representação interna dos computadores de ponto flutuante compromete os resultados na solução dos sistemas polinomiais representados pelas equações (6) e (5) (esta última quando uma das variáveis é feita igual a 1 ou 0). É preciso estar consciente de que qualquer seqüência de operações em um computador digital é

essencialmente equivalente a uma seqüência finita de manipulações em um conjunto discreto de pontos. Este problema pode ser solucionado substituindo a aritmética de ponto flutuante pela aritmética intervalar arredondada que assegura robustez numérica e fornece resultados com precisão confiável. O intervalo $[a,b]$ é um conjunto de números reais definido por:

$$[a,b]=\{x \mid a \le x \le b\} \qquad (12)$$

Onde as seguintes operações de aritmética intervalar são definidas:

$$\begin{aligned}
&[a,b]+[c,d]=[a+c,b+d]\\
&[a,b]-[c,d]=[a-d,b-c]\\
&[a,b]\cdot[c,d]=[\min(a\cdot c,a\cdot d,b\cdot c,b\cdot d),\max(a\cdot c,a\cdot d,b\cdot c,b\cdot d)]\\
&[a,b]/[c,d]=[\min(a/c,a/d,b/c,b/d),\max(a/c,a/d,b/c,b/d)]
\end{aligned} \qquad (13)$$

Se a aritmética de ponto flutuante é utilizada para implementar estas operações de aritmética intervalar, então não é garantido que o arredondamento dos contornos esteja sendo realizado corretamente. A aritmética intervalar arredondada assegura que os contornos são determinados de modo a conter o intervalo exato, assim definimos as seguintes operações para álgebra intervalar arredondada:

$$\begin{aligned}
&[a,b]+[c,d]=[a+c-\varepsilon_l,b+d+\varepsilon_u]\\
&[a,b]-[c,d]=[a-d-\varepsilon_l,b-c+\varepsilon_u]\\
&[a,b]\cdot[c,d]=[\min(a\cdot c,a\cdot d,b\cdot c,b\cdot d)-\varepsilon_l,\max(a\cdot c,a\cdot d,b\cdot c,b\cdot d)+\varepsilon_u]\\
&[a,b]/[c,d]=[\min(a/c,a/d,b/c,b/d)-\varepsilon_l,\max(a/c,a/d,b/c,b/d)+\varepsilon_u]
\end{aligned} \qquad (14)$$

Onde ε_l representa a diferença entre o número em ponto flutuante determinado e o número em ponto flutuante imediatamente inferior. O mesmo vale para ε_u, mas em relação ao número em ponto flutuante imediatamente superior. Uma maneira de determinar estes parâmetros é demonstrada em Abrams et al. (1998). Sabemos que a simples substituição das variáveis de ponto flutuante por variáveis de aritmética intervalar arredondada pode nos fornecer intervalos exageradamente grandes que não possuem utilidade prática. Entretanto, o algoritmo do poliedro projetado foi adaptado de forma eficiente, por ser um algoritmo de busca por intervalo. Assim, um intervalo $[[a,b],[c,d]]$ da equação (5) que contém a solução pode ser reconfigurado para $[[a,a],[d,d]]$.

Quanto aos outros dois pontos principais de pesquisa, comparamos dois algoritmos para converter um polinômio em base de potência para base de Bernstein. O primeiro foi apresentado por Faustini (1999) que se utilizava de substituições sucessivas. O segundo algoritmo foi baseado na representação matricial para as superfícies de Bézier adotada por Yamaguchi (1988). A forma matricial foi utilizada para definir uma representação matricial n-dimensional. Devido às possibilidades da álgebra matricial, segundo a representação feita por Yamaguchi (1988), é possível converter um polinômio de duas variáveis em base de potência para a base de Bernstein diretamente. Para o caso de um polinômio com n variáveis foi utilizado um algoritmo para converter uma variável por vez para a base de Bernstein. O mesmo procedimento foi feito para implementar o algoritmo de conversão de variáveis. Definimos dois algoritmos: um direto e outro utilizando álgebra matricial. Como é de conhecimento da literatura, duas expressões algébricas distintas, mas equivalentes, produzem resultados distintos em aritmética intervalar arredondada. Entretanto, um dos pontos mais importantes é que os algoritmos definidos utilizando-se de álgebra matricial foram os que produziram um menor alargamento do intervalo e, simultaneamente, foram os mais velozes.

6. CONCLUSÕES

Neste trabalho definimos um algoritmo para determinar a curva de intersecção entre duas superfícies de Gregory. Um dos grandes problemas para este tipo de superfície é que não existe algoritmo para subdividir o seu domínio paramétrico. Para isto, foi utilizada a proposta feita por Grandine (1997) para determinar pontos iniciais para um algoritmo de caminhada para traçar a curva de intersecção. A proposta de Grandine exige que um conjuntos de sistemas polinômiais sejam solucionados, e para este fim adotamos o algoritmo do Poliedro Projetado. Entretanto, foi necessário realizarmos algumas adaptações ao algoritmo do Poliedro Projetado para que o sistema polinomial de grau 10 (mínimo) fosse solucionado e todas as soluções fossem encontradas. O sistema foi implementado em linguagem C++ utilizando o software Visual C++ 6.0.

7. AGRADECIMENTOS

O autor foi parcialmente suportado pelo CNPq – proc. 300.224/96-6. Este projeto foi suportado pela FAPESP – proc. 99/12447-0.

9. REFERÊNCIAS

- Abrams, S.L.; Cho, W.; Hu, C.Y.; Maekawa, T.; Patrikalakis, N.M.; Sherbrooke, E.C.; Ye, X., 1998, “Efficient and Reliable Methods for Rounded Interval Arithmetic”. Computer Aided Design, Vol. 30.
- Chiyokura, H; Kimura, F., 1983, “Design of Solids with Free Form Surfaces”. Computer Graphics. Vol. 17, N. 3, pp. 289-298.
- Faustini, M.C., 1999, “Proposta de Algoritmo para a Determinação da Intersecção entre Superfícies NURBS”. Dissertação de Mestrado. Dep. Eng. Mecânica da Escola Politécnica da USP.
- Grandine, T.A.; Klein, F.W., 1997, “A New Approach to the Surface Intersection Problem”. Computer Aided Geometric Design, N. 14, pp. 111-134.
- Hu, C.Y.; Maekawa, T.; Patrikalakis, N.M.; Ye, X., 1997, “Robust Interval Algorithms for Surface Intersections”. Computer Aided Design, Vol. 29, N. 9, pp. 617-627.
- Kriezis, A.G.; Patrikalakis, N.M.; Wolter, F.E., 1992, “Topological and Differential-Equation Methods for Surface Intersections”. Computer Aided Design, Vol. 24, N. 1, pp. 41-55.
- Lasser, D., 1986, “Intersection of Parametric Surfaces in the Bernstein-Bézier Representation”. Computer Aided Design, vol. 18, N. 4, pp. 186-192.
- Sedeberg, T.W.; Christiansen, H.N.; Katz, S., 1989, “An Improved Test for Closed Loops in Surface Intersections”. Computer Aided Design, Vol. 21, N. 8, pp. 505-508.
- Sherbrooke, E.C.; Patrikalakis, N.M., 1993, “Computation of the Solutions of Nonlinear Polynomial Systems”. Computer Aided Geometric Design, Vol. 21, N. 10, pp. 379-405.
- Wang, Y,; Gursoz, E.L.; Chen, J.M.; Prinz, F.B.; Patrikalakis, N.M., 1991, “Intersection of Parametric Surfaces for Next Generation of Geometric Modelers”. In Product Modeling for Computer Aided Design and Manufacturing. Elsevier Science Publishers B.V., New York, pp. 75-96.
- Yamaguchi, F., 1988, “Curves and Surfaces in Computer Aided Geometric Design”. Springer Verlag.