COBEM 99
XV Congresso Brasileiro de Engenharia Mecânica
15th Brazilian Congress of Mechanical Engineering
22 - 26 de Novembro de 1999 / November 22 - 26, 1999 Águas de Lindóia, São Paulo,

ALGORITHM TO DETERMINE THE INITIAL POINTS OF THE INTERSECTION CURVES BETWEEN BEZIER SURFACES THROUGH THE SOLUTION OF MULTIVARIABLE POLYNOMIAL SYSTEM

Mário Carneiro Faustini
Prof. Dr. Marcos Sales Guerra Tsuzuki
Universidade de São Paulo, Escola Politécnica, Departamento de Engenharia Mecânica
Av. Prof. Mello Moraes, 2231, CEP 05508-900, São Paulo - SP, Brasil

Abstract. *The determination of the intersection curves between Bézier Surfaces may be seen as the composition of two separate problems: determining initial points and tracing the intersection curve from these points. The Bézier Surface is represented by a parametric function (polynomial with two variables) that maps a point in the tridimensional space from the bidimensional parametric space. In this article, it is proposed an algorithm to determine the initial points of the intersection curves of Bézier Surfaces, based on the solution of polynomial systems with the Projected Polyhedral Method. In order to allow the use of this method, the equations of the system must be represented in terms of the Bernstein basis, and towards this goal it is proposed a robust and reliable algorithm to exactly transform a multivariable polynomial in terms of power basis to a polynomial written in terms of Bernstein basis .*

Keywords: *parametric surfaces, intersection curves, multivariable polynomial systems*

1. INTRODUCTION

In the development of solid modelers supporting the geometric representation of surfaces, it is crucial that the problem of determining the intersection curves between surfaces must be algorithmically well resolved. When a solid with a curved boundary, is defined in a CAD/CAM system using Boolean Operations, it is necessary that the intersection curves are exactly determined. The toolpath for sculptured surface milling can be defined according to the zig-zag approach, and in this case it is necessary to determine the intersection curves between the solid's surface boundary itself and cutting planes. As can be seen, such an algorithm has several applications in the field of Mechanical Engineering.

This problem was initially addressed by several authors, like (Lasser, 1986) and (Hosaka, 1992), where it is shown that intersection problem may be split in two different and sequential problems: determining initial points of the intersection curves and, then, tracing them from these points. This way, it is possible to define an algorithm to each application domain. However, some quite common situations couldn't be included in the application domain of existing algorithms. For instance, with the use of these previous methods sometimes it isn't possible to determine the intersection curve's initial points if there are singularities* on the curve.

In this work, it is presented a more general approach to determine the initial points of intersection curves for Bézier surfaces based on the solution of multivariable polynomial systems, with a result presented at the last section.

2. BÉZIER SURFACES

The Bézier surfaces are widely used in CAD applications. Such parametric surfaces are defined by the following expression (Hosaka, 1992):

$$\mathbf{S}(u,v)=\sum_{i=0}^{m}\sum_{j=0}^{n}B_i^m(u)\cdot B_j^n(v)\cdot \mathbf{P}_{ij} \tag{1}$$

where $\mathbf{P}_{ij}$ are the control points of the surface and $B_i^m(u)$ e $B_j^n(v)$ are Bernstein polynomials, defined by:

$$B_i^n(u)=\binom{n}{i}\cdot u^i\cdot(1-u)^{n-i}\ ,\ 0\le u\le 1\ (\text{or } u\in[0,1]) \tag{2}$$

where $\binom{n}{i}$ is the binomial coefficient:

$$\binom{n}{i}=\frac{n!}{i!\cdot(n-i)!} \tag{3}$$

The Bernstein polynomials have, among others, the following main properties:

$$B_i^m(u)\ge 0 \quad \forall u\in[0,1] \tag{4}$$

$$\sum_{i=0}^{m}B_i^m(u)=1 \tag{5}$$

The control points are disposed in a rectangular net of $n+1$ by $m+1$. This way, the position of a surface point in the *xyz* space is given by:

$$\mathbf{S}(u,v)=\begin{bmatrix}x(u,v)\\ y(u,v)\\ z(u,v)\end{bmatrix} \tag{6}$$

where each coordinate is defined by a polynomial in u and v. The surface degree (and therefore, the degree of the polynomials that describe each coordinate) is given by m and n. For a matter of simplicity, the surfaces presented in this work are all 3rd degree Bézier surfaces (m = n = 3). Nevertheless, as it will become clear, the presented method isn't limited by the surface's degree.

* Singularities are defined as intersection points for which the two respective normal vectors, referring to each surface, are parallel.

3. DETERMINING INITIAL POINTS

The problem of determining the intersection curve between two surfaces is defined in the following way: given **F**(u,v) and **G**(s,t) surfaces parametrized within the domain $[0,1]^2$, it is desired to obtain all the pairs (u,v) and (s,t) so that

$$\mathbf{F}(u,v) - \mathbf{G}(s,t) = 0 \qquad (7)$$

The vectorial expression above encloses a system of 3 equations and 4 unknowns, so that a direct solution is quite complex. This way, in order to trace the intersection curves some marching method (Hu et al., 1997) may be used to generate them from some discrete points on the curves (initial points). Obtaining these points is the main task of this work.

The solution of Eq. (7) may consist of none, one or more than one isolated curves, as shown by Fig. 1. To use the marching method, it is necessary to have at least one initial point for each isolated intersection curve.

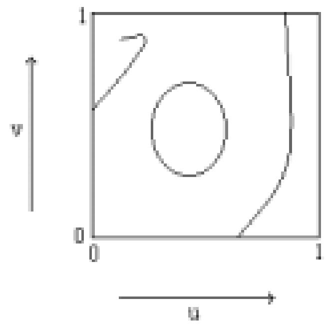

Figure 1- Example of a set of intersection curves from two surfaces, seen by the parametric domain of one of the surfaces.

The following approach, presented by (Grandine & Klein, 1997), intends to find the initial points for the intersection curves. This way, given the surfaces **F**(u,v) and **G**(s,t), it is needed to obtain the set u, v, s and t so that Eq. (7) is true.

If we reparametrize Eq. (7) by the variable τ representing the arc length of the intersection curve, it becomes:

$$\mathbf{F}(u(\tau), v(\tau)) - \mathbf{G}(s(\tau), t(\tau)) = 0 \qquad (8)$$

In order to determine the points of the intersection curves in which they cross the borders of the parametric domains of **F** and **G**, we may substitute each parameter for zero or one in Eq. (7) and solve the remaining 3x3 system. For u = 0 or 1 and v = 0 or 1, we obtain initial points for the intersection curves that cross the borders of the parametric domain of **F** and that may correspond to any point inside the parametric domain of **G**. In the other hand, for s = 0 or 1 and t = 0 or 1, we obtain the initial points for curves that may begin and end anywhere inside the domain of **F**, but that for sure are limited by the borders of the parametric domain of **G**. This way, the cases of the border points of **F** and **G** may be processed separately.

Supposing that all the zeros on the borders of the parametric domain of **F** have been determined (u = 0 and 1, v = 0 and 1 in Eq. (7)). The resulting points could be those seen in Fig. 2.

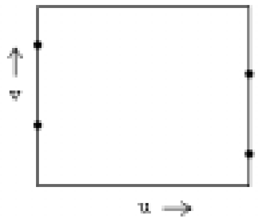

Figure 2- Initial points on the borders of a parametric domain.

So, the possible topologies of the intersection curves limited by the points in Fig. 4 are the ones shown in Fig. 3. Other than those, we may have intersection curves that cross no border of any parametric domain. These curves are called intersection loops, and they may be in any number and at any position inside the parametric domain.

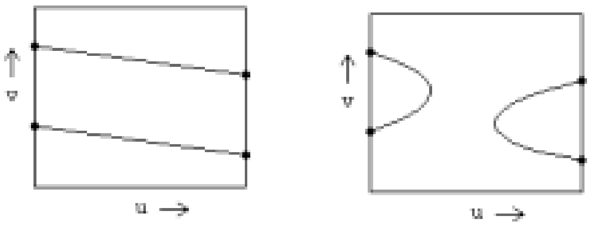

Figure 3- Possible topologies of the intersection curves from Fig. 4.

In order to determine these loops, we "scan" the parametric domain of the surface **F** with parallel lines inclined of θ in relation to the direction of parameter v, as shown in Fig. 4.

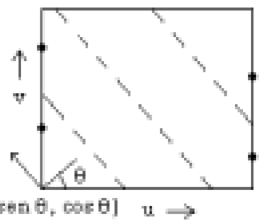

Figure 4- Parallel lines to "scan" the parametric domain.

This way, the vector (- sin θ , cos θ) is parallel to the scanning lines, with $0 \leq \theta \leq \pi/2$. This allows us to determine the points where the intersection curves change their direction from these lines and return towards them. These are called *turning points*, and they are characterized by the fact that the tangent vector of the curve on this point is parallel to the direction of the scanning lines. In the example of the intersection curves of Fig. 5, it is shown how the scanning lines "find" the *turning points* both for opened curves and loops, detecting the presence of the last ones and providing their initial points.

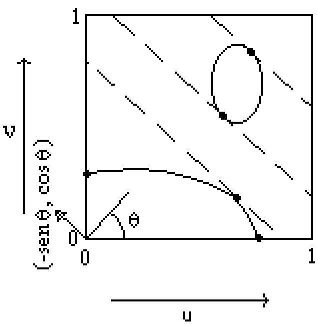

Figure 5- Example of the determination of *turning points* of intersection curves by scanning lines.

A *turning point* is, then, given by:

$$u'(\tau)\cdot\cos\theta + v'(\tau)\cdot\sin\theta = \mathbf{0} \qquad (9)$$

what means that ($u'(\tau)$, $v'(\tau)$) is perpedicular to (cos θ , sin θ).

Differentiating Eq. (8):

$$\mathbf{F}_u u'(\tau) + \mathbf{F}_v v'(\tau) - \mathbf{G}_s s'(\tau) - \mathbf{G}_t t'(\tau) = 0 \qquad (10)$$

So, we have that the vector ($u'(\tau)$, $v'(\tau)$, $s'(\tau)$, $t'(\tau)$) is inside the null space of the 3x4 matrix [$\mathbf{F}_u$ $\mathbf{F}_v$ $-\mathbf{G}_s$ $-\mathbf{G}_t$]. This null space is given by multiples of the vector:

$$\begin{pmatrix} \det[\mathbf{F}_v \quad \mathbf{G}_s \quad \mathbf{G}_t] \\ -\det[\mathbf{F}_u \quad \mathbf{G}_s \quad \mathbf{G}_t] \\ -\det[\mathbf{F}_u \quad \mathbf{F}_v \quad \mathbf{G}_t] \\ \det[\mathbf{F}_u \quad \mathbf{F}_v \quad \mathbf{G}_s] \end{pmatrix} \qquad (11)$$

From Eqs. (9), (10) and (11), it can be seen that the *turning points* are the solutions to the following 4x4 system:

$$\begin{aligned} &\mathbf{F}(u,v) - \mathbf{G}(s,t) = 0 \\ &(\mathbf{F}_u sen\theta - \mathbf{F}_v \cos\theta) \circ \mathbf{G}_s \times \mathbf{G}_t = 0 \end{aligned} \qquad (12)$$

The solution set for the system above will also contain the points where one intersection curve crosses another (called *critical points*) in addition to the *turning points*.

Ultimately, the main idea of this method is to find initial points for each of the intersection curves. With this, the solution of the original 3 equations and 4 unknowns system in expression Eq. (7) is simplified with the addition of a 4^{th} equation, originating the 4x4 system of Eq. (12). It is interesting to notice that this system fundamentally deals with intersection loops, since initial points for opened intersection curves are given by the determination of the points where the intersection curves cross the borders of the parametric domains of the surfaces, as explained in the beginning of this section.

Therefore, it must be chosen a robust and reliable numerical method to solve the multivariable polynomial systems that arise, since their solutions will represent the desired initial points which will be classified in lists that will define the relations among them, allowing a complete topological analysis of the intersection curves as seen through the parametric domains. In order to fulfill this task, it is used the Projected Polyhedral Method, explained in the next section.

4. PROJECTED POLYHEDRAL METHOD

Solving multivariable polynomial systems means, in algebraic terms, determining all the n-uples $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ so that

$$f_1(\mathbf{x}) = f_2(\mathbf{x}) = \ldots = f_n(\mathbf{x}) = 0 \qquad (13)$$

with $x_i \in [0,1]^\dagger$ and f_k a polynomial with n variables.

The projected polyhedral method was fully presented in (Sherbrooke & Patrikalakis, 1993). Its first step is to transform each equation f_k to the Bernstein basis, what means:

$$f_k(\mathbf{x}) = \sum_{i_1=0}^{d_1^{(k)}} \sum_{i_2=0}^{d_2^{(k)}} \cdots \sum_{i_n=0}^{d_n^{(k)}} \left[w_{i_1 i_2 \cdots i_n}^{(k)} \right] \cdot B_{i_1}^{d_1^{(k)}}(x_1) \cdot B_{i_2}^{d_2^{(k)}}(x_2) \cdots B_{i_n}^{d_n^{(k)}}(x_n) \qquad (14)$$

where $d_i^{(k)}$ is the degree of the variable x_i in f_k.

Now, let's redefine the problem of Eq. (13) as the task of determining the intersection of the graphs‡ of each f_k (each graph can be seen as an hypersurface in $\Re^{n+1}$) and the hyperplane $x_{n+1} = 0$. Each graph of f_k is given by

$$\mathbf{F}_k(\mathbf{x}) = (x_1, x_2, \cdots, x_n, f_k(\mathbf{x})) = (\mathbf{x}, f_k(\mathbf{x})) \qquad (15)$$

So that the system of Eq. (13) becomes:

$$\mathbf{F}_1(\mathbf{x}) = \mathbf{F}_2(\mathbf{x}) = \ldots = \mathbf{F}_n(\mathbf{x}) = (\mathbf{x}, 0) \qquad (16)$$

Since each graph $\mathbf{F}_k$ is also a parametric hypersurface, then its control points would be:

---

† or any other finite domain $[a_i, b_i]$ which must be, then, mapped in the domain [0,1] through the variable transformation $x_i' = a_i + x_i \cdot (b_i - a_i)$ in every equation f_k.

‡ Be $f: \Re^n \to \Re$ a function of $\mathbf{x}$. Then, its graph is the function $F: \Re^n \to \Re^{n+1}$ defined by: $F(\mathbf{x}) = (\mathbf{x}, f(\mathbf{x}))$.

$$\mathbf{v}_I^{(k)} = \left(\frac{i_1}{d_1^{(k)}}, \frac{i_2}{d_2^{(k)}}, \cdots, \frac{i_n}{d_n^{(k)}}, w_I^{(k)} \right) \tag{17}$$

Therefore a new way to look at the problem of solving the system of Eq. (13) would be to work out the equivalent system of Eq. (16). And this means to determine the intersection of the hypersurfaces defined by each $\mathbf{F}_k$. An initial approach to this problem would be determining the intersection among the convex-hulls that are formed by the control points of theses hypersurfaces in the space $\Re^{n+1}$. Moreover, it is possible to convert this problem of dimension (n + 1) into n bidimensional problems by projecting the control points of each $\mathbf{F}_k$ on every plane of the space $\Re^{n+1}$ formed by one coordinate related to the variable x_j and the last coordinate of Eq. (15), related to the original function f_k (in other words, one plane for each variable of the initial system of Eq.(13)). This way, the control points of each and every $\mathbf{F}_k$ are projected on n planes, so that for the plane j, of the variable x_j, their coordinates would be

$$\mathbf{v}_{j,I}^{(k)} = \left(\frac{i_j}{d_j^{(k)}}, w_I^{(k)} \right) \tag{18}$$

Then it is formed, in each plane, the bidimensional convex-hull of the projected control points of each and every $\mathbf{F}_k$. And within each plane, it is determined the intersection of all convex-hulls with the segment [0,1] (that is related to the domain of the variable x_j) of the abscissa of the plane. The result may be one segment, one point or null (because all the projected convex-hulls are convex polygons). If null, then there is no value for the variable of this plane to be considered as a solution of the system. If the result is a point, then its value (a real within [0,1]) is a solution itself for the related variable in the system of Eq. (13). And if the result is a segment, then it is still possible that inside of it there is one or more points of solution to the related variable in the system.

This means that, for each variable x_i, we will possibly have decreased the range of search from the initial variable's domain [0,1] down to a segment [a_i, b_i]. So, we can reduce the box of solutions from the initial full domain $[0,1]^n$ down to the newly found box of solution:

$$S = [a_1, b_1] \times [a_2, b_2] \times \cdots \times [a_n, b_n] \tag{19}$$

At this point, for each k, we define a new function f_k' so that:

$$f_k'(\mathbf{x}) = f_k\left(a_1 + (b_1 - a_1)\cdot x_1, a_2 + (b_2 - a_2)\cdot x_2, \ldots, a_n + (b_n - a_n)\cdot x_n\right) \tag{20}$$

This function will be mapped within the domain $[0,1]^n$ in the same way that f_k was mapped by the domain of Eq.(13).

The whole process is then iteratively repeated until we find boxes of solution (like that of Eq. (19)) with all sides small enough to be seen as isolated roots for the variables of the system. In other words: in S (Eq.(19)), $\forall\ i$ so that $1\leq i \leq n$, it must be true that $(b_i - a_i) \leq$ *tolerance*. This *tolerance* defines the precision of the solutions.

However, after processing the intersection of the convex-hulls within the planes, it is possible that the value $(b_i - a_i)$ of some variables in S (Eq. (19)) may have not significantly decreased from 1. This happens when there are more than one solution to the system of Eq. (13). In such case, we must split in two the segment of the variable that wasn't considerably reduced. So on, we deal with the two resulting boxes as independent and separate problems. For instance, if we obtained the box $S=[a_1,b_1]\times[a_2,b_2]\times\cdots\times[a_n,b_n]$ and $(b_2 - a_2) >$ *criteria* ($\approx$1), then two solution sub-boxes will be separately considered at next iteration: $S=[a_1,b_1]\times\left[a_2,\frac{(b_2-a_2)}{2}\right]\times\cdots\times[a_n,b_n]$ and $S=[a_1,b_1]\times\left[\frac{(b_2-a_2)}{2},a_2\right]\times\cdots\times[a_n,b_n]$. It is interesting to notice that such subdivision is done recursively until a sub-box with only one solution is isolated.

5. BASIS CONVERSION OF MULTIVARIABLE POLYNOMIALS FROM POWER TO BERNSTEIN BASIS

Such basis conversion is a fundamental step of the Projected Polyhedral Method, as seen in the previous section, and its evaluation represents the main problem found when implementing such algorithm. A generic approach may be proposed in pursue of this task (Faustini, 1999). This way, the following polynomial, function of the parameters u_1, u_2 … until u_n with the maximum degree of each of them as being, respectively, d_1, d_2 … until d_n:

$$P(u_1,u_2,\ldots,u_n)=\sum_{g_1=0}^{d_1}\sum_{g_2=0}^{d_2}\cdots\sum_{g_n=0}^{d_n}a_{g_1g_2\cdots g_n}u_1^{g_1}u_2^{g_2}\cdots u_n^{g_n} \tag{21}$$

may be rewritten in terms of the Bernstein basis:

$$\begin{aligned}P(u_1,u_2,\ldots,u_n)=&\sum_{g_1=0}^{d_1}\sum_{g_2=0}^{d_2}\cdots\sum_{g_n=0}^{d_n}\sum_{i_1=0}^{g_1}\sum_{i_2=0}^{g_2}\cdots\sum_{i_n=0}^{g_n}w_{i_1i_2\cdots i_n}\cdot\binom{d_1}{g_1}\binom{d_2}{g_2}\cdots\binom{d_n}{g_n}\binom{g_1}{i_1}\binom{g_2}{i_2}\\&\cdots\binom{g_n}{i_n}\cdot(-1)^{g_1+g_2+\cdots+g_n-i_1-i_2-\cdots-i_n}u_1^{g_1}u_2^{g_2}\cdots u_n^{g_n}\end{aligned} \tag{22}$$

Then it is possible to design an algorithm for each case, when needed, from the following generic element of the matrix of weights **W** (as in Eq. (14)):

$$\begin{aligned}w_{g_1g_2\cdots g_n}=&\frac{a_{g_1g_2\cdots g_n}}{\binom{d_1}{g_1}\cdot\binom{d_2}{g_2}\cdots\binom{d_n}{g_n}}-\\&-\sum_{i_1=0}^{g_1-1}\sum_{i_2=0}^{g_2-1}\cdots\sum_{i_n=0}^{g_n-1}w_{i_1i_2\cdots i_n}\cdot\binom{g_1}{i_1}\cdot\binom{g_2}{i_2}\cdots\binom{g_n}{i_n}\cdot(-1)^{g_1+g_2+\cdots+g_n-i_1-i_2-\cdots-i_n}\end{aligned} \tag{23}$$

6. RESULTS

Some examples are now presented. The algorithms to generate them were implemented in C programming language and executed with a Pentium 200MHz computer. All the surfaces presented in both Fig. 5 and Fig. 8 are 3rd degree Bézier surfaces. In Fig. 7 and Fig. 9 we can see the initial points for the intersection curves.

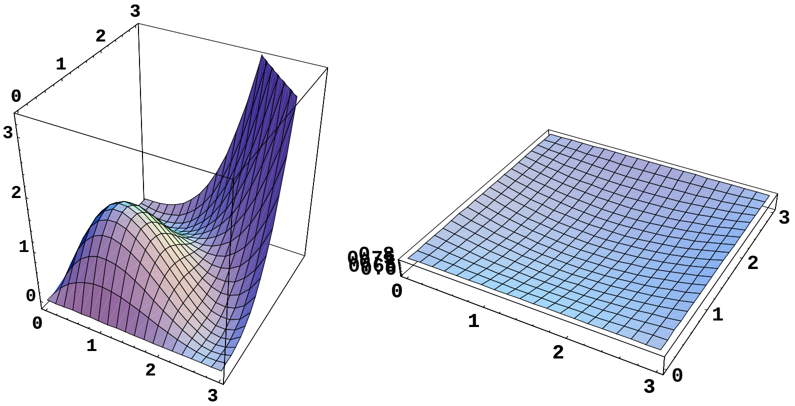

Figure 6- Intercepting Bézier surfaces

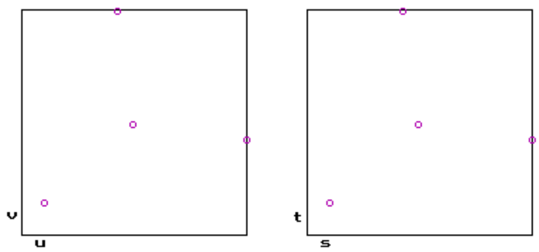

Figure 7- Initial points of the intersection curves from surfaces of Fig. 6, seen from each parametric domain, respectively. Processing time: 51.360s

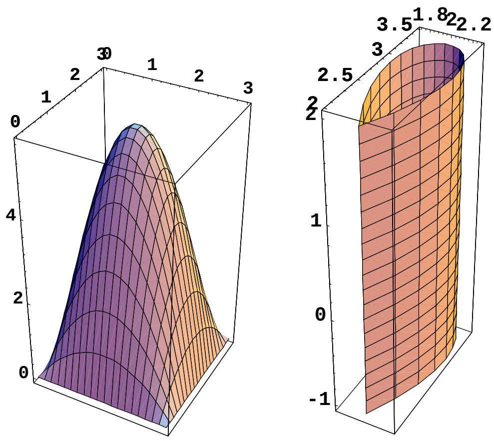

Figure 8- Intercepting Bézier Surfaces

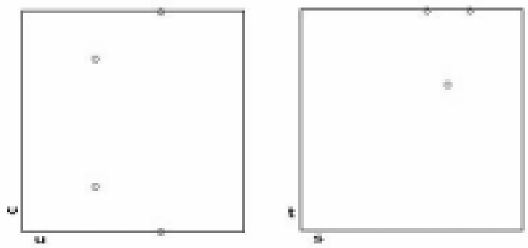

Figure 9- Initial points of the intersection curves from surfaces of Fig. 8, seen from each parametric domain, respectively. Processing time: 5.4s

7. CONCLUSIONS

In this work we presented a complete proposal of determining initial points for the intersection curves between two Bézier Surfaces. And we gave our contribution by presenting a way of converting multivariable polynomials from power basis to equivalent ones in Bernstein basis, which is a fundamental step in the implementation of the Projected Polyhedral Method for solving multivariable nonlinear polynomial systems.

Another aspect to be considered for future improvements is that the Projected Polyhedral algorithm may have its processing time significantly reduced if we take advantage of the possibility of parallel processing that the referred method allows.

REFERENCES

Faustini, M. C., 1999, Proposta de Algoritmo para a Determinação da Intersecção entre Superfícies NURBS, Master Thesis, Dep. de Eng. Mecânica, Escola Politécnica, USP.

Grandine, T. A. & Klein IV, F. W., 1997, A new approach to the surface intersection problem, Computer Aided Geometric Design, No. 14, pp. 111-134.

Hosaka, M., 1992, Modeling of Curves and Surfaces in CAD/CAM, Springer-Verlag, Berlim.

Hu, C. Y., Maekawa, T., Patrikalakis, N. M. and Ye, X., 1997, Robust interval algorithm for surface intersections, Computer-Aided Design, Vol. 29, No. 9, pp.617-627.

Lasser, D., 1986, Interserction of parametric surfaces in the Bernstein-Bézier representation, Computer-Aided Design, Vol. 18, No. 4, pp. 186-192.

Sherbrooke, E. C. & Patrikalakis N. M., 1993, Computation of the solutions of nonlinear polynomial systems, Computer Aided Geometric Design No. 10, pp. 379-405.

Acknowledgements

We would like to thanks FAPESP for supporting the first author (proc. 97/02939-8) and CNPq for supporting the second author (proc. 300.224/96-6).