Geometric algorithm for point projection and inversion onto Bézier surfaces

Jinting XU (✉)1,2, Weijun LIU2, Jianhuang WU3, Hongyou BIAN2, Lun LI2

1 School of Automotive Engineering, Dalian University of Technology, Dalian 116024, China
2 Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110016, China
3 Centre for Human Computer Interaction, Shenzhen Institution of Advanced Integration Technology, Chinese Academy of Sciences, Shenzhen 518067, China

Abstract This paper presents an accurate and efficient method for the computation of both point projection and inversion onto Bézier surfaces. First, these two problems are formulated in terms of solution of a polynomial equation with \( u \) and \( v \) variables expressed in the Bernstein basis. Then, based on subdivision of the Bézier surface and the recursive quadtree decomposition, a novel solution method is proposed. The computation of point projection is shown to be equivalent to the geometrically intuitive intersection of a surface with the \( u-v \) plane. Finally, by comparing the distances between the test point and the candidate points, the closest point is found. Examples illustrate the feasibility of this method.

Keywords point projection, point inversion, Bézier surface

1 Introduction

Projecting a test point onto a parametric surface in order to find the closest point (the point projection problem) and computing the corresponding parameter values of the projection (the point inversion problem) are two basic problems in computational geometry, geometric modeling, computer graphics, and related topics. Both projection and inversion are useful for surface intersection [1], tool path generation, and collision detection in numerical control (NC) machining [2]. They are also a key issue in the inspection of manufactured objects [3]. For these purposes, it is important to have a computational method which is efficient and reliable to find the closest point and the corresponding parameter values on surfaces.

Several algorithms which employ some variations of Newton iteration or numerical optimization have been developed to solve this problem. Mortenson [4] derived equations for different types of surface distance measure and employed the Newton-Raphson method to solve them. Limaiem and Trochu [5] proposed another approach to compute the projection of a point onto parametric curves and surfaces by constructing an auxiliary function and finding its zeros. These algorithms converge quickly, but the sensitivity of the iteration process to initial values is also well known. As an improvement, Hu and Wallner [6] developed a second-order algorithm for point projection onto curves and surfaces. It increases the robustness to the choice of initial values.

This Newton-type methods have been used in CAD/CAM applications and acquired high accuracy because they are efficient (usually exhibiting quadratic convergence rates close to simple roots) and straightforward to program. Unfortunately, this is an error-prone process which often fails for points near the endpoints of the curve or the boundaries of the surface and this procedure typically requires a good initial value to ensure the convergence to the optimal root in the global scene. Such initial approximations are usually obtained by sampling the curve or surface, but this process cannot provide full assurance that all roots have been found. This lack of robustness promotes the development of efficient and reliable techniques. Piegl and Tiller [7] presented an algorithm for point projection on non-uniform rational B-splines (NURBS) surface. It consists of three steps: decomposing the NURBS surface into quadrilaterals, projecting the
test point onto the closest quadrilateral, and recovering the parameter from the closest quadrilateral. Motivated by them, Ma and Hewitt [8] subdivided the NURBS curve or surface into Bézier curves or surfaces, used the relationship between the test point and the control points of Bézier curves or surfaces to eliminate those parts excluding the closest point, and finally applied the Newton-Raphson method on a "flag enough" Bézier curve or surface to improve its accuracy. Selimovic [9] improved the subdivision-based techniques further by employing a stricter exclusion criterion. It increases the robustness and leads to a considerable reduction of computation time.

The main objective of this paper is to develop a novel method, which can solve the point projection and inversion problems. It combines the arithmetic for two multivariate Bernstein-form polynomials with subdivision technology, and employs the convex hull property of Bézier surface to eliminate each of the parts excluding projections. A low subdivision depth can also provide a good initial value for Newton-Raphson or other numerical optimization methods to improve the stability.

2 Outline of the algorithm

The basic idea of the algorithm is that the point projection problem on Bézier surface is formulated in terms of solution of a polynomial equation $s(u, v)$ with $u$ and $v$ variables expressed in the Bernstein basis. We search the solutions by subdividing a surface, $s(u, v)$, recursively. The surface is the graph of $s(u, v)$ represented by a Bézier surface over the $u$-$v$ parametric plane. In each step, we subdivide the surface into four parts at the midpoint of the parametric domain. After each subdivision, we can eliminate the parts excluding solutions by analyzing the relationship between the control polygon and the $u$-$v$ plane. The subdivision process stops as soon as the computing accuracy is satisfied. We compare the distances between the test point and the candidate points in order to select the closest point. The efficiency of the algorithm substantially relies on the ability to eliminate the parts which do not contain solutions and its accuracy can be also improved by using the Newton-Raphson method.

3 Projection problem formulation

A Bézier surface is given by

$$\Sigma^B : r(u, v) = \sum_{i=0}^{m} \sum_{j=0}^{n} b_{i,j} B_{i,m}(u) B_{j,n}(v),$$  \hspace{1cm} (1)

where $\{b_{i,j}\} \in \mathbb{R}^3$ are the control points of Bézier surface $\Sigma^B$ and in natural ordering they are the vertices of the Bézier control net. For a fixed point $p_0 = [x_0, y_0, z_0]^T$, the squared distance function between the point and an arbitrary point on surface $\Sigma^B$ can be calculated by the following equation

$$D(u, v) = (r(u, v) - p_0), (r(u, v) - p_0).$$ \hspace{1cm} (2)

Let $\nabla = [\partial/\partial u, \partial/\partial v]^T$ be the differential gradient operator. The minimum of the distance function $D(u, v)$ occurs where its partial derivatives satisfy $D_u = 0$ and $D_v = 0$. Namely, the gradient function is equal to zero, $\nabla D = 0$. The partial derivatives of the distance function with respect to the parameters $u$ and $v$ are given by

$$D_u = \frac{\partial D(u, v)}{\partial u} = 2 \left( \frac{\partial r(u, v)}{\partial u}, r(u, v) - p_0 \right), \hspace{1cm} (3)$$

$$D_v = \frac{\partial D(u, v)}{\partial v} = 2 \left( \frac{\partial r(u, v)}{\partial v}, r(u, v) - p_0 \right). \hspace{1cm} (4)$$

To be convenient for the follow-up process, an equation equivalent to $\nabla D = 0$ is rewritten as

$$s(u, v) = D_u^2 + D_v^2 = 0. \hspace{1cm} (5)$$

The problem of searching the closest point from the surface $\Sigma^B$ can be solved by calculating the proper parameter values that satisfy Eq. (5). To avoid the complicated iterative process, we make use of the de Casteljau algorithm and the convex hull property of the Bézier surface to solve this equation. The formula of the derivatives of the Bézier surface $\Sigma^B$ with respect to the parameter $u$ and $v$ is expressed as follows:

$$\frac{\partial r(u, v)}{\partial u} = m \sum_{i=0}^{m-1} \sum_{j=0}^{n} b_{i,j}^1 B_{i,m-1}(u) B_{j,n}(v), \hspace{1cm} (6)$$

$$\frac{\partial r(u, v)}{\partial v} = n \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} b_{i,j}^0 B_{i,m}(u) B_{j,n-1}(v), \hspace{1cm} (7)$$

where $\{b_{i,j}^0\}$ and $\{b_{i,j}^1\}$ are the first forward difference vectors of the control points of the Bézier surface $\Sigma^B$. By using the normality of Bernstein basis function, the point $p_0$ can be expressed as a degenerative bivariate Bézier surface modeled by

$$\Sigma_0^B : r_0(u, v) = m \sum_{i=0}^{m} \sum_{j=0}^{n} d_{i,j} B_{i,m}(u) B_{j,n}(v), \hspace{1cm} (8)$$

where $d_{i,j} = p_0$, for $i = 0, \ldots, m$ and $j = 0, \ldots, n$.

Now, by substituting Eqs. (1), (6)-(8) into Eq. (5) and using the arithmetic operations of the addition, subtraction,
and multiplication of the multivariate Bernstein-form polynomial [10], \( s(u, v) \) can be written as a bivariate Bernstein-form polynomials as shown in Eq. (9).

\[
s(u, v) = \sum_{i=0}^{4n} \sum_{j=0}^{4n} g_{i,j} B_{i,4n}(u) B_{j,4n}(v), \quad (9)
\]

where \( \{g_{i,j}\} \in \mathbb{R} \) are the Bernstein coefficients of the polynomial.

To obtain a more intuitionistic mathematical model of finding the closest point, by using the linear precision property of the Bernstein basis function, the graph of the function \( s(u, v) \) can be represented by a Bézier surface over the \( u \)-\( v \) parametric plane. This Bézier surface patch is modeled by the following parameter equation:

\[
\Sigma : s(u, v) = \begin{pmatrix} u \\ v \\ s(u, v) \end{pmatrix} = \sum_{i=0}^{4n} \sum_{j=0}^{4n} g_{i,j} B_{i,4n}(u) B_{j,4n}(v).
\]

In Eq. (10),

\[
g_{i,j} = \left[ \frac{i}{4m}, \frac{j}{4n}, g_{i,j} \right]^T,
\]

where \( \{g_{i,j}\} \in \mathbb{R}^3 \) are the control points of the Bézier surface \( \Sigma \).

From the above derivation, it is concluded that \( s(u, v) = 0 \) means that the surface \( \Sigma \) is tangential to the parametric plane because \( s(u, v) \geq 0 \) (See Eq. (5)). Now, the problem of searching the closest point on the Bézier surface is transformed into a geometric problem of finding the tangent point of the Bézier surface \( \Sigma \) to the \( u \)-\( v \) plane.

### 4 Searching strategy

Recursive quadtree decomposition on the \( u \)-\( v \) domain is employed to search the regions possibly containing the proper parameter values, as shown in Fig. 1. The surface \( \Sigma \) is subdivided into four sub-surfaces by using the de Casteljau algorithm and the parameter domain is also subdivided simultaneously at its midpoint. For each sub-domain, the subdivision process is continued until the computing accuracy is satisfied or it excludes roots. The searching strategy is to eliminate the sub-surfaces which do not include the solution by using the convex hull property of the Bézier surface. The convex hull property determines if a sub-domain contains roots of the equation \( s(u, v) = 0 \) as shown in Eq. (5). The algorithm checks the signs of the Bernstein coefficients of the polynomial \( s(u, v) \) related to each sub-domain. If all of the coefficients have the uniform sign, the sub-domain is marked as the one that does not include the solution. In this case, the recursive subdivision stops. Otherwise, we continue this operation of subdivision until the size of the sub-domain is less than the computing accuracy \( \delta_{\text{tree}} \), namely, \( 2^{-n} \leq \delta_{\text{tree}} \), where \( n \) is the depth of the quadtree. Then, all regions that are marked as one possibly including the roots are searched and the midpoint of the desirable parametric region is viewed as the proper parameter value.

![Fig. 1](image)

**Fig. 1** An example of quadtree decomposition (the marked dark domains indicate one which possibly contains the closest point)

### 5 Algorithm for the closest point on surface

Now the algorithm based on the recursive quadtree decomposition for computing the closest point is summarized as follows:

1. Calculate the Bernstein polynomial \( s(u, v) \); See Eq. (9).
2. Create the Bézier surface \( \Sigma \); See Eq. (10).
3. Subdivide the parametric domain that possibly contains the solution into four sub-domains at its midpoint.
4. Check the sign of the Bernstein coefficients of the polynomial related to each of the sub-domains to determine whether the sub-domain contains the solution or not. If all the coefficients have the uniform sign, the sub-domain is marked as one excluding the roots, and the algorithm stops. Or else, the sub-domain is marked as one including the roots, and go to step 5).
5. If \( 2^{-n} > \delta_{\text{tree}} \) at the depth \( n \), go to step 3).
6. Stop and return to the closest point.

### 6 Point inversion

When the minimum distance of a point to surfaces is equal to zero, the point projection problem becomes a point inversion problem. We can apply the above algorithm to extract the closest point on the surface and its corresponding parameter values.

### 7 Experimental results

**Example 1** In this test, we mainly compare the accuracy of our method with that of the Newton-Raphson method in
order to verify its validity for calculating the projection on the Bézier surface.

A Bézier surface and a test point (67.92,37.31,60.46) are shown in Fig. 2(a). Our method subdivides directly the \( u-v \) parametric domain to search the proper parameter values without any additional operation. The subdivision of the surface and parametric domain are shown in Fig. 2(a) and Fig. 2(b) respectively. In Fig. 2(c), the convergence of the Newton-Raphson method with respect to the different initial values is shown. We can see that when the initial value is close to the exact solution, the Newton-Raphson method is convergent (e.g. red line). However, if the initial value is far away from the exact solution, it is emanative (e.g. blue line).

Comparison between the results of the two methods is shown in Table 1. From Table 1, we can see that the minimum distance by ours is nearly equal to that of the Newton-Raphson method under the same computing accuracy. This indicates that our method is accurate and effective. Moreover, it overcomes the disadvantage of the Newton-type method, namely, a good initial value is provided beforehand. Due to the use of the de Casteljau algorithm and the convex hull property of the Bézier surface, our method is particularly suitable for the Bézier surface and B-spline surface. In the case of B-spline surface, it can be converted to a set of Bézier surfaces by matured algorithm. When the closest point is on the boundary, the algorithm degenerates to one of searching the closest point from the boundary curve. It can be solved easily by using the convex hull property of the Bézier curve.

**Table 1** Comparison between Newton-Raphson method and our method

<table>
<thead>
<tr>
<th>Method</th>
<th>Closest point</th>
<th>Minimum distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ours</td>
<td>(72.21, 44.09, 44.88)</td>
<td>17.525</td>
</tr>
<tr>
<td>N-R</td>
<td>(72.24, 44.08, 44.89)</td>
<td>17.519</td>
</tr>
</tbody>
</table>

**Example 2** In this test, we give an example of point projection to the Bézier surface to compare our method with Ma’s method and Selimovic’s method to demonstrate its efficiency. The tested surface is a typical free-form surface as shown in Fig. 3. As referred by Selimovic, the time performance of the algorithms strongly relied on hardware and implementation; we also count the number of necessary subdivisions. In every example, we adopt the same test method as Selimovic’s, namely, to project the points of a \( 10 \times 10 \times 10 \) equidistant grid on the curve and surface and compare the average number of subdivisions.

Comparison of experimental results is shown in Table 2. From Table 2, we can see that our method can clearly decrease the number of necessary subdivisions. This is because, instead of using the relationship between the test and the control points of surfaces, our method employs the first derivatives of the distance function and eliminates the regions excluding the closest point by analyzing the relationship between the convex hull of surfaces and the parametric plane.

**Table 2** The results of point projection for the Bézier surface

<table>
<thead>
<tr>
<th>Surface index</th>
<th>Ma’s method</th>
<th>Selimovic’s method</th>
<th>Our method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fig. 3</td>
<td>26.34</td>
<td>12.73</td>
<td>9.64</td>
</tr>
</tbody>
</table>

**8 Conclusion**

In this paper, we investigate point projection and inversion on Bézier surfaces. As an alternative to the Newton-type methods, the proposed method transforms the abstract algebraic operation of finding the closest point into the geometrically intuitive intersection of a surface with the \( u-v \) plane.
Hereby, it avoids the iterative process and the initial estimate on the closest point, and is very suitable for the Bézier surface and B-spline surface. Experimental results show that the algorithms under consideration are efficient. Furthermore, this method can provide full assurance that all roots have been found, especially in these cases that the closest point is on the boundaries of the surface. A low subdivision depth can also provide a good initial value to ensure reliable convergence for the Newton-Raphson method.

Acknowledgements This work was supported by the National Natural Science Foundation of China (Grant No. 60803108).

References