



Ray-triangular Bézier patch intersection using hybrid clipping algorithm*

Yan-hong LIU^{1,2}, Juan CAO^{†‡1,2}, Zhong-gui CHEN³, Xiao-ming ZENG^{1,2}

(¹School of Mathematical Sciences, Xiamen University, Xiamen 361005, China)

(²Fujian Provincial Key Laboratory of Mathematical Modeling and High-Performance Scientific Computation, Xiamen University, Xiamen 361005, China)

(³School of Information Science and Engineering, Xiamen University, Xiamen 361005, China)

[†]E-mail: juancao@xmu.edu.cn

Received Nov. 10, 2015; Revision accepted Mar. 28, 2016; Crosschecked Sept. 11, 2016

Abstract: In this paper, we present a novel geometric method for efficiently and robustly computing intersections between a ray and a triangular Bézier patch defined over a triangular domain, called the hybrid clipping (HC) algorithm. If the ray pierces the patch only once, we locate the parametric value of the intersection to a smaller triangular domain, which is determined by pairs of lines and quadratic curves, by using a multi-degree reduction method. The triangular domain is iteratively clipped into a smaller one by combining a subdivision method, until the domain size reaches a prespecified threshold. When the ray intersects the patch more than once, Descartes' rule of signs and a split step are required to isolate the intersection points. The algorithm can be proven to clip the triangular domain with a cubic convergence rate after an appropriate preprocessing procedure. The proposed algorithm has many attractive properties, such as the absence of an initial guess and insensitivity to small changes in coefficients of the original problem. Experiments have been conducted to illustrate the efficacy of our method in solving ray-triangular Bézier patch intersection problems.

Key words: Ray tracing, Triangular Bézier surface, Ray-patch intersection, Root-finding, Hybrid clipping
<http://dx.doi.org/10.1631/FITEE.1500390>

CLC number: TP391.7

1 Introduction

Ray tracing is a valuable tool for rendering scenes with amazing realistic light effects in computer graphics (Haines *et al.*, 1989; Joy *et al.*, 1989), including soft shadows, caustics, reflection, transparency, and motion blur. In a ray tracing based algorithm, the ray-scene intersection test is a fundamental operation that is required frequently. The efficiency of ray tracing relies heavily on the compu-

tation of the intersection test. Objects in the scene vary from simple triangles, polygons, and spheres to complex surfaces such as parametric splines and implicit surfaces. Among these, triangular Bézier patches have been accepted as a primitive choice by the modeling community because of their flexibility in modeling objects with complicated topology. However, the intersection test for a ray with objects is a nontrivial technique even for the simplest case. In this study, we focus on the problem of ray-triangular Bézier patch intersections and present a more efficient algorithm for finding such intersections.

The number of ray-patch intersection tests can be vastly reduced by using the space partition

[‡] Corresponding author

* Project supported by the National Natural Science Foundation of China (Nos. 61100105, 61572020, and 61472332), the Natural Science Foundation of Fujian Province of China (No. 2015J01273), and the Fundamental Research Funds for the Central Universities, China (Nos. 20720150002 and 20720140520)

ORCID: Juan CAO, <http://orcid.org/0000-0002-8154-4397>

© Zhejiang University and Springer-Verlag Berlin Heidelberg 2016

method, i.e., bounding volume hierarchy. A bounding volume can be easily placed around a patch. Then a ray is first tested against the bounding volume. The patch with the bounding volume intersecting with the ray is chosen as the candidate surface for further tests. The simple geometric bounding primitives include axis-aligned bounding boxes (Sweeney and Bartels, 1986), bounding spheres (Haines *et al.*, 1989), oriented slab boxing (Yen *et al.*, 1991), and parallelepipeds (Barth and Stürzlinger, 1993). Methods for finding an intersection between a ray and the candidate patches can be classified into four types: subdivision, approximation, Newton's method, and geometric clipping method. The most direct approach is subdivision (Woodward, 1989); i.e., a surface is split into smaller patches and the ones that cannot intersect with the ray are discarded. A surface is subdivided until a maximum depth is reached. The concept of subdivision is simple, and the algorithm is easy to implement. The subdivision method is generally inefficient and requires high temporary memory. Apart from the direct subdivision method, we can compute the intersection between a ray and a surface approximation, including its tessellation (i.e., a set of triangles or other primitives) or implicit approximations (Hanrahan, 1983). This compromise simplifies the original intersection problem; however, this method still suffers from the drawbacks of high memory requirement and computational cost.

Alternatively, if we write a ray in a parametric form, then the ray-triangular Bézier patch intersection problem can be analytically reduced to the problem of finding roots of a system of nonlinear equations. Many numerical methods in computational mathematics are available for solving this problem. For example, Newton's method is commonly used to find the intersection between a ray and a bicubic Bézier surface (Markus and Oliver, 2005), a nonuniform rational B-spline (Martin *et al.*, 2000), a triangular Bézier surface (Stürzlinger, 1998), and so on. Newton's method is not unconditionally stable. In addition, a good initial guess, which is sufficiently close to the solution, is required to guarantee the convergence. The convergence speed of Newton's method can be improved by combining it with interval analysis (Moore and Jones, 1977), i.e., performing Newton search in intervals in which convergence is guaranteed. However, this approach requires an extremely large number of surface splits and guar-

anteed convergence tests to obtain an appropriate interval (Toth, 1985).

In contrast to the methods mentioned above, geometric clipping methods are numerically more stable and require less memory. As a matter of fact, an initial guess for the parameters of intersections is unnecessary and the intersections are guaranteed to be found within a reasonable time as long as they exist. The original geometric clipping method arises from the problem of finding the intersection between a ray and a trimmed rational surface (Nishita *et al.*, 1990). It iteratively narrows down the parametric domain of the surface by using the convex hull property until a prespecified accuracy threshold is reached. Afterward, this concept is generalized to find roots of polynomials (Bartoň and Jüttler, 2007a; Liu *et al.*, 2009), curve/curve intersections (Sederberg and Nishita, 1990; Schulz, 2009; Lou and Liu, 2012), ray-triangular Bézier patch intersections (Roth *et al.*, 2000), and zeros of bivariate polynomial systems (Bartoň and Jüttler, 2007b; Jüttler and Moore, 2011), where Descartes' rule of signs and the subdivision strategy (Garloff and Smith, 2001; Rouillier and Zimmermann, 2004) are used to isolate roots. As all the methods mentioned in this paragraph gradually narrow the range of roots or the parametric interval of intersections by using the geometric properties of Bézier curves/surfaces, we collectively refer to them as geometric clipping methods.

In this study, we propose a geometric clipping method to solve the ray-triangular Bézier patch intersection problem. Note that among the aforementioned geometric clipping methods, only the so-called Bézier clipping method in Roth *et al.* (2000) is tailored for solving the ray-triangular Bézier patch intersection problem, which cuts out the parametric region without intersections by using the convex hull of a triangular Bézier surface. Rather than relying solely on the convex hull, our method combines convex hulls and low-degree approximations of triangular Bézier surfaces to narrow the parametric region. Hence, our method can be viewed as a generalization of the Bézier clipping method in Roth *et al.* (2000). Our method inherits all the merits of the Bézier clipping method and is numerically more efficient. Theoretical analysis shows that our algorithm has a cubic convergence rate for a single root. For the sake of convenience, 'ray-triangular Bézier patch

intersection' is hereafter referred to as 'ray-patch intersection' (RPI) when ambiguity is not an issue.

2 Computing ray-triangular Bézier patch intersection by the hybrid clipping algorithm

2.1 Triangular Bézier surface and the 'ray-patch intersection' problem

Let $\triangle T_1 T_2 T_3$ be a nondegenerate triangle in \mathbb{R}^2 with vertices T_1 , T_2 , and T_3 , which are ordered counterclockwise. Then any point P in \mathbb{R}^2 can be uniquely expressed as follows:

$$P = uT_1 + vT_2 + wT_3,$$

where

$$(u, v, w) = \left(\frac{\text{Area}(PT_2 T_3)}{\text{Area}(T_1 T_2 T_3)}, \frac{\text{Area}(T_1 P T_3)}{\text{Area}(T_1 T_2 T_3)}, \frac{\text{Area}(T_1 T_2 P)}{\text{Area}(T_1 T_2 T_3)} \right)$$

are the barycentric coordinates of P with respect to $\triangle T_1 T_2 T_3$. $\text{Area}(\cdot)$ represents the oriented area of the triangle in this term and $u + v + w = 1$.

A degree n triangular Bézier surface on a triangle T is defined as follows:

$$\mathbf{R}(u, v, w) = \sum_{|\mathbf{i}|=n} \mathbf{R}_{\mathbf{i}} B_{\mathbf{i}}^n(u, v, w), \quad 0 \leq u, v, w \leq 1, \quad (1)$$

where $\mathbf{i} = (i, j, k)$, $|\mathbf{i}| = i + j + k = n$, $B_{\mathbf{i}}^n(u, v, w) = \frac{n!}{i!j!k!} u^i v^j w^k$ with (u, v, w) the barycentric coordinates with respect to T , and $\mathbf{R}_{\mathbf{i}} \in \mathbb{R}^3$ are the control points.

Given that $w = 1 - u - v$, surface (1) can also be written as follows:

$$\mathbf{R}(u, v) = \sum_{j=0}^n \sum_{i=0}^{n-j} \mathbf{R}_{ij} B_{ij}^n(u, v), \quad 0 \leq u, v \leq 1, \quad (2)$$

where $\mathbf{R}_{ij} = (x_{ij}, y_{ij}, z_{ij})$ ($j = 0, 1, \dots, n$, $i = 0, 1, \dots, n-j$) are the control points and $0 \leq u + v \leq 1$.

A ray can be represented by

$$\hat{\mathbf{R}}(t) = \mathbf{r}_0 + t\mathbf{r}_d, \quad t > 0, \quad \|\mathbf{r}_d\| = 1,$$

where \mathbf{r}_0 is the origin of the ray and \mathbf{r}_d is the normalized direction.

For a surface $\mathbf{R}(u, v)$ and a ray $\hat{\mathbf{R}}(t)$, seeking values u, v , and t such that

$$\mathbf{R}(u, v) = \hat{\mathbf{R}}(t)$$

is called an RPI problem.

2.2 Reduction of the 'ray-patch intersection' problem

In the Bézier clipping method (Roth *et al.*, 2000), the RPI problem is first converted into a simpler form. Analogously, we first reduce the RPI problem to the problem of solving a system of equations with two unknowns.

The ray $\hat{\mathbf{R}}(t)$ can be represented as the intersection of two orthogonal planes as follows (Roth *et al.*, 2000):

$$\begin{cases} a_{\alpha}x + b_{\alpha}y + c_{\alpha}z + d_{\alpha} = 0, \\ a_{\beta}x + b_{\beta}y + c_{\beta}z + d_{\beta} = 0, \end{cases} \quad (3)$$

where $a_k^2 + b_k^2 + c_k^2 = 1$ ($k \in \{\alpha, \beta\}$) and $(a_{\alpha}, b_{\alpha}, c_{\alpha}) \cdot (a_{\beta}, b_{\beta}, c_{\beta}) = 0$.

Given that RPIs lie on both the ray and the patch, intersections should also satisfy the following equations, which are obtained by substituting Eq. (2) into Eq. (3):

$$f^k(u, v) = \sum_{j=0}^n \sum_{i=0}^{n-j} f_{ij}^k B_{ij}^n(u, v) = 0, \quad (4)$$

where $k \in \{\alpha, \beta\}$, $0 \leq u + v \leq 1$, $0 \leq u, v \leq 1$, and

$$f_{ij}^k = a_k x_{ij} + b_k y_{ij} + c_k z_{ij} + d_k \quad (5)$$

with x_{ij} , y_{ij} , and z_{ij} the coordinates of the control points of surface (2). Hence, the RPI problem is converted into solving the system of equations with respect to variables u and v as follows:

$$\begin{cases} f^{\alpha}(u, v) = 0, \\ f^{\beta}(u, v) = 0, \end{cases} \quad (6)$$

where $f^{\alpha}(u, v)$ and $f^{\beta}(u, v)$ are defined by Eq. (4). This conversion reduces the number of computing operations.

2.3 Hybrid clipping algorithm

In this subsection, we introduce a geometric clipping method to solve the reduced RPI problem. That is, to find the roots of a system of two bivariate

polynomial (Eq. (6)) over a triangular parametric domain (hereinafter, referred to as tri-box) \mathcal{D} . Bézier clipping is a typical geometric algorithm for solving the reduced RPI problem. However, the Bézier clipping method potentially generates wrong intersections, due to the essential step of the method, i.e., projection. In contrast, our algorithm avoids the projection step and ensures the correct results.

The basic idea of our algorithm is as follows. Note that the roots of system (6) are precisely the intersections of the two curves defined by equations in system (6) within a tri-box. Then both the curves in system (6) are individually bounded in a strip formed by a pair of curves obtained by a degree reduction method (Zhang and Wang, 2005; Lu and Wang, 2006). We refer to the strip as a fat curve. Hence, the intersections of the two original curves are bounded in fat curves. By calculating the intersections between fat curves and using the subdivision method, we obtain sub-tri-boxes with the intersections of the original curves within them. The aforementioned procedure is repeatedly applied until the diameter of the sub-tri-box becomes smaller than a prespecified threshold ϵ . The pseudocode of this procedure is shown in Algorithm 1. As fat curves with different degrees are used in Algorithm 1, the algorithm is called the hybrid clipping (HC) algorithm. For the remainder of this subsection, we provide a detailed description of each step in the HC algorithm.

Algorithm 1 HC($f^\alpha, f^\beta, \mathcal{D}, \epsilon$)

```

1: if diam( $\mathcal{D}$ )  $\geq \epsilon$  then
2:    $\hat{f} \leftarrow$  preprocessing( $f^\alpha, f^\beta, \mathcal{D}$ )
3:    $\mathcal{L} \leftarrow$  compute the fat line of  $\hat{f} = 0$ 
4:    $\mathcal{Q} \leftarrow$  compute the fat curve of  $f^\beta = 0$ 
5:    $\mathcal{D}' \leftarrow$  clipping( $\mathcal{L}, \mathcal{Q}, \mathcal{D}$ )
6:   if diam( $\mathcal{D}'$ )  $\neq 0$  and check-sign( $f^\alpha, f^\beta, \mathcal{D}'$ )=0
       then
7:     if  $u_{\min} + v_{\min} + w_{\min} \leq 0.5\text{diam}(\mathcal{D})$  then
8:       split( $\mathcal{D}'$ ) ( $\mathcal{D}'$  is split into four tri-boxes) and
           apply the HC algorithm to each tri-box
9:     else
10:      return HC( $f^\alpha, f^\beta, \mathcal{D}', \epsilon$ )
11:    end if
12:  end if
13: else
14:  return  $\mathcal{D}$ 
15: end if

```

2.3.1 Preprocessing

This step aims to improve the rate of convergence of our algorithm and we will give the proof in Lemma 1. The underlying principle involves converting the original system (6) into an equivalent system with the Hessian matrix of one equation vanishing at the center of tri-box \mathcal{D} . The equivalent system is obtained by replacing the first equation of system (6) by $\hat{f}(u, v) = \ell(u, v)f^\alpha(u, v) + (1 - \ell(u, v))f^\beta(u, v)$, where $\ell(u, v) = \ell_0 + \ell_1 u + \ell_2 v$. The coefficients $\ell_0, \ell_1, \ell_2 \in \mathbb{R}$ are determined by allowing the Hessian $\nabla^2 \hat{f}$ to vanish at the incenter (u_c, v_c) of the corresponding tri-box. In particular, the coefficients are obtained by solving the linear system as follows:

$$\mathbf{A}(u_c, v_c) \begin{pmatrix} \ell_0 \\ \ell_1 \\ \ell_2 \end{pmatrix} + \begin{pmatrix} \frac{\partial^2 f^\beta}{\partial u^2} \\ \frac{\partial^2 f^\beta}{\partial u \partial v} \\ \frac{\partial^2 f^\beta}{\partial v^2} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \quad (7)$$

where

$$\mathbf{A}(u_c, v_c) = \begin{pmatrix} \Delta_{uu} & u\Delta_{uu} + 2\Delta_u & v\Delta_{uu} \\ \Delta_{uv} & u\Delta_{uv} + \Delta_v & v\Delta_{uv} + \Delta_u \\ \Delta_{vv} & u\Delta_{vv} & v\Delta_{vv} + 2\Delta_v \end{pmatrix} \quad (8)$$

and

$$\Delta_x = \frac{\partial(f^\alpha - f^\beta)}{\partial x}, \quad \Delta_{xy} = \frac{\partial^2(f^\alpha - f^\beta)}{\partial x \partial y}, \quad x, y \in \{u, v\}.$$

If matrix $\mathbf{A}(u_c, v_c)$ is of full rank, then coefficients ℓ_0, ℓ_1, ℓ_2 can be determined uniquely, and the degree of \hat{f} is $n + 1$. Otherwise, we simply set $\hat{f} = f^\alpha$. Then the equivalent system, i.e.,

$$\begin{cases} \hat{f}(u, v) = 0, \\ f^\beta(u, v) = 0, \end{cases} \quad (9)$$

rather than the original one (Eq. (6)), is used.

2.3.2 Fat curve computation

To simplify the description, we first introduce three types of norm for bivariate polynomials $f(u, v)$ that are associated with tri-box \mathcal{D} in the form of Eq. (4):

1. Normalized L_2 norm:

$$\|f(u, v)\|_2^{\mathcal{D}} = \frac{1}{\text{Area}(\mathcal{D})} \left(\int_{\mathcal{D}} \|f(u, v)\|^2 du dv \right)^{1/2}. \quad (10)$$

2. L_∞ norm:

$$\|f(u, v)\|_\infty^{\mathcal{D}} = \max_{(u,v) \in \mathcal{D}} \|f(u, v)\|. \quad (11)$$

3. Bernstein-Bézier (BB) norm:

$$\|f(u, v)\|_{\text{BB}, \infty}^{\mathcal{D}} = \max_{0 \leq i+j \leq n} |f_{ij}|, \quad (12)$$

where f_{ij} ($0 \leq i + j \leq n$) are the Bernstein-Bézier coefficients of $f(u, v)$.

Let $f_{k, \mathcal{D}}$ be the best approximation of degree k ($k < n$) to f in the L_2 -norm over tri-box \mathcal{D} , which can be obtained using the degree reduction method (Lu and Wang, 2006). On the other hand, the degree of $f_{k, \mathcal{D}}$ can be elevated to the same degree as f using the degree elevation algorithm. Denote the Bernstein-Bézier coefficients of $f_{k, \mathcal{D}}$ with the degree elevated to n by \bar{f}_{ij} and define $\delta_{f, \mathcal{D}} = \|f - f_{k, \mathcal{D}}\|_{\text{BB}, \infty}^{\mathcal{D}} = \max_{i,j} |f_{ij} - \bar{f}_{ij}|$. By following the convex hull property of the triangular Bézier surface, curve $f = 0$ can be bounded by curves $f_1 = f_{k, \mathcal{D}} + \delta_{f, \mathcal{D}} = 0$ and $f_2 = f_{k, \mathcal{D}} - \delta_{f, \mathcal{D}} = 0$ over tri-box \mathcal{D} . Hence, curves $f_1 = 0$ and $f_2 = 0$ form the degree k fat curve of f , denoted by $\mathcal{L} = (f_1 = 0, f_2 = 0)$. An example of the curve defined by $f(u, v) = \sum_{j=0}^n \sum_{i=0}^{n-j} f_{ij} B_{ij}^n(u, v) = 0$ and its corresponding quadratic fat curve are shown in Fig. 1a. Note that any polynomial $f(u, v)$ in the form of Eq. (4), i.e., $f(u, v) = \sum_{j=0}^n \sum_{i=0}^{n-j} f_{ij} B_{ij}^n(u, v)$ with $\mathcal{D} = \{(u, v) | 0 \leq u + v \leq 1\}$, $f_{ij} \in \mathbb{R}$, can be considered as a triangular Bézier surface $\mathbf{f}(u, v) = \sum_{j=0}^n \sum_{i=0}^{n-j} \mathbf{f}_{ij} B_{ij}^n(u, v)$ on a triangular domain with control points $\mathbf{f}_{ij} = (i/n, j/n, f_{ij}) \in \mathbb{R}^3$ and $0 \leq i + j \leq n$. Hence, $f = 0$ can be considered as the intersection curve of surface \mathbf{f} and plane $z = 0$. Fig. 1b shows the 3D illustrations of \mathbf{f} , \mathbf{f}_1 , and \mathbf{f}_2 . Algorithm 2 summarizes the steps for computing the degree k fat curve of a given polynomial f in the form of Eq. (4).

In particular, we compute the linear and quadratic fat curves of \hat{f} and f^β in the HC algorithm.

Algorithm 2 Fat curves(f, \mathcal{D}, k) (computing degree k fat curves)

-
- 1: $f_{k, \mathcal{D}} \leftarrow$ generate a degree k surface of f by degree reduction
 $\delta_{f, \mathcal{D}} \leftarrow$ compute $\|f - f_{k, \mathcal{D}}\|_{\text{BB}, \infty}^{\mathcal{D}}$
 $f_1 \leftarrow f_{k, \mathcal{D}} + \delta_{f, \mathcal{D}}$ (upper bound)
 $f_2 \leftarrow f_{k, \mathcal{D}} - \delta_{f, \mathcal{D}}$ (lower bound)
 - 2: **return** $\mathcal{L} = (f_1 = 0, f_2 = 0)$
-

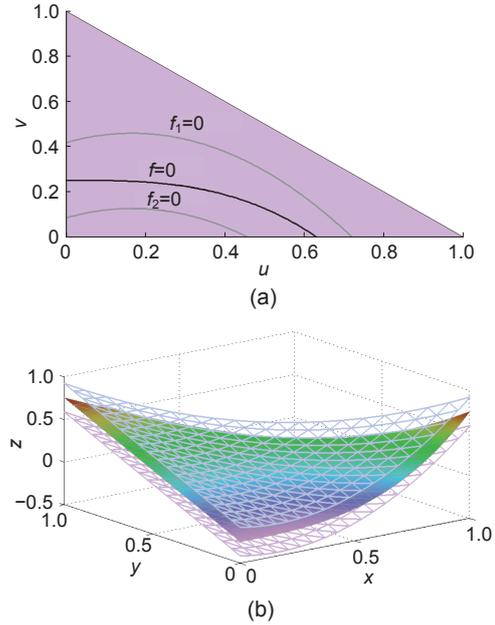


Fig. 1 Curve $f(u, v) = \sum_{j=0}^n \sum_{i=0}^{n-j} f_{ij} B_{ij}^n(u, v) = 0$ bounded by two conic curves $f_1 = 0$ and $f_2 = 0$ over the purple domain \mathcal{D} (a) and the functions in 3D view (b) (surface $f(u, v)$ (the middle one) is bounded by two degree 2 surfaces (the top and bottom meshes defined by f_1 and f_2 , respectively)). References to color refer to the online version of this figure

More precisely, we denote

$$\delta_{l, \mathcal{D}} = \|\hat{f} - l_{\mathcal{D}}\|_{\text{BB}, \infty}^{\mathcal{D}}, \quad (13)$$

$$\delta_{q, \mathcal{D}} = \|f^\beta - q_{\mathcal{D}}\|_{\text{BB}, \infty}^{\mathcal{D}}, \quad (14)$$

where $l_{\mathcal{D}}$ and $q_{\mathcal{D}}$ are the best linear approximation to \hat{f} and the best quadratic approximation to f^β in the L_2 -norm over tri-box \mathcal{D} , respectively. Then we have the linear fat curve $\mathcal{L} = (l_1 = 0, l_2 = 0)$ and the quadratic fat curve $\mathcal{Q} = (q_1 = 0, q_2 = 0)$, where $l_1 = l_{\mathcal{D}} + \delta_{l, \mathcal{D}}$, $l_2 = l_{\mathcal{D}} - \delta_{l, \mathcal{D}}$, $q_1 = q_{\mathcal{D}} + \delta_{q, \mathcal{D}}$, and $q_2 = q_{\mathcal{D}} - \delta_{q, \mathcal{D}}$. Hence, $\hat{f} = 0$ and $f^\beta = 0$ are bounded in \mathcal{L} and \mathcal{Q} , respectively; i.e., the roots of Eq. (6) are bounded in \mathcal{L} and \mathcal{Q} .

2.3.3 Clipping

This step involves cutting away the region of the parametric domain with no intersection based on the computation of fat curves and the subdivision method for a Bézier surface. For a Bézier surface defined over tri-box \mathcal{D} , assume that the parameter u (resp. v and w) can be narrowed down to a smaller range $[u_{\min}, u_{\max}]$ (resp. $[v_{\min}, v_{\max}]$ and $[w_{\min}, w_{\max}]$), where u_{\min}/u_{\max}

(resp. v_{\min}/v_{\max} and w_{\min}/w_{\max}) are the possible minimum/maximum values of u (resp. v and w) obtained by a certain method. Then the parameter of an intersection point (u_0, v_0) can be restricted to a smaller polygonal region by cutting away the region outside $[u_{\min}, u_{\max}]$, $[v_{\min}, v_{\max}]$, and $[w_{\min}, w_{\max}]$, as shown in Fig. 2a. Note that the obtained region is generally not triangular because the three parameters u, v, w depend on each other, and thus the corresponding clipped surface does not maintain a Bézier form. Hence, we settle for cutting out regions $u < u_{\min}$, $v < v_{\min}$, and $w < w_{\min}$ to maintain the triangular shape of the remainder region as was done in Roth *et al.* (2000) (Fig. 2b). Thus, the corresponding trimmed surface maintains a Bézier form by subdividing the original surface at $u = u_{\min}$, $v = v_{\min}$, and $w = w_{\min}$. We denote the candidate triangular-shaped domain (i.e., a tri-box) by \mathcal{D}' . To obtain \mathcal{D}' , we need to compute the following four types of critical points:

1. The intersection points between fat curves \mathcal{L} and \mathcal{Q} ;
2. The vertices of tri-box \mathcal{D} , which lie in the quadrilateral region formed by fat curves \mathcal{L} and \mathcal{Q} ;
3. The intersection points between fat curve \mathcal{L} (resp. \mathcal{Q}) and the boundary of tri-box \mathcal{D} , which lie in the strip area formed by \mathcal{Q} (resp. \mathcal{L});
4. The values u^* , v^* , and w^* , which define lines $u = u^*$, $v = v^*$, and $w = w^*$ tangent to fat curve \mathcal{Q} .

Then u_{\min} , v_{\min} , and w_{\min} are the minimum values of the corresponding parameters of the aforementioned four types of points, and the clipped tri-box \mathcal{D}' can be obtained. Combining with the subdivision method, potential multiple intersections can be isolated, and the parameter of each intersection point (u_0, v_0) can be recursively narrowed down until the size of the triangular region reaches a prespecified threshold.

2.3.4 Check-sign step

This step, given in line 6 of Algorithm 1, aims to discard the tri-box in which the parameters of the intersection points are impossible to be included. According to Descartes' rule of signs, for a bivariate function represented in a Bézier form, if all the Bézier coefficients have the same sign, then no intersection will be formed with plane $z = 0$. In particular, $f^\alpha = 0$ and $f^\beta = 0$ will have no intersection if all the coefficients of f^α (or f^β) have the same sign. Hence,

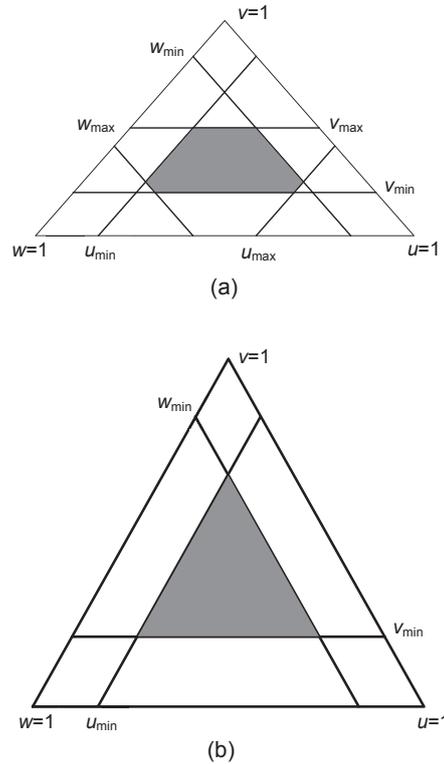


Fig. 2 Candidate domain (filled in gray): (a) candidate polygonal parameter domain by clipping away the region outside $[u_{\min}, u_{\max}]$, $[v_{\min}, v_{\max}]$, and $[w_{\min}, w_{\max}]$; (b) candidate triangular parameter domain obtained by cutting out regions $u < u_{\min}$, $v < v_{\min}$, and $w < w_{\min}$

$\text{check-sign}(f^\alpha, f^\beta, \mathcal{D}')$ returns zero if and only if both the coefficients of f^α and f^β have different signs.

2.3.5 One-to-four split

The one-to-four split operation in line 8 of Algorithm 1 is used to guarantee that the convergence rate of our algorithm will not be lower than that of the subdivision method. In particular, if the diameter of tri-box \mathcal{D}' is larger than half of the size of \mathcal{D} , i.e., $u_{\min} + v_{\min} + w_{\min} \leq 0.5 \text{diam}(\mathcal{D})$, then we split tri-box \mathcal{D}' into four sub-tri-boxes by connecting the midpoints of edges. The surfaces are simultaneously subdivided into four patches. Then the HC method is applied to each sub-tri-box.

Note that preprocessing is an optional operation. Hereinafter, we refer to the algorithm with preprocessing as the HC algorithm by default. The algorithm without preprocessing will be explicitly indicated as ‘HC without preprocessing’. In particular, if we set both fat curves in the HC algorithm without preprocessing as linear curves, then we obtain a

special case of the HC algorithm, called the linear clipping algorithm.

3 Convergence rate

In this section, we theoretically prove that Algorithm 1 provides cubic convergence for a single root. In particular, we make the following assumptions:

(A1) Eq. (6) has exactly only one root $\mathbf{r} = (u_0, v_0)$.

(A2) The determinant of matrix \mathbf{A} defined in Eq. (8) has a nonzero value at root \mathbf{r} , i.e., $\det \mathbf{A}(u_0, v_0) \neq 0$. That is, \hat{f} can be uniquely determined.

(A3) $\hat{f} = 0$ and $f^\beta = 0$ transversely intersect each other at \mathbf{r} . More precisely, gradients $\nabla \hat{f}$ and ∇f^β are linearly independent at root \mathbf{r} .

Lemma 1 For any polynomial \hat{f} obtained by the preprocessing step of Algorithm 1, a constant C_l , which depends solely on \hat{f} , exists, such that for any tri-box $\mathcal{D} \subseteq \mathcal{D}_0$, the bound $\delta_{l,\mathcal{D}}$ defined in Eq. (13) satisfies

$$\delta_{l,\mathcal{D}} \leq C_l(\text{diam}(\mathcal{D}))^3. \quad (15)$$

Proof For any triangular domain \mathcal{D} , we consider the following three types of norms: $\|\cdot\|_2^{\mathcal{D}}$, $\|\cdot\|_\infty^{\mathcal{D}}$, and $\|\cdot\|_{\text{BB},\infty}^{\mathcal{D}}$, which are defined in Eqs. (10)–(12), respectively. Given that all norms are equivalent in a finite-dimensional vector space, there exist two constants C_1 and C_2 such that

$$\|f\|_{\text{BB},\infty}^{\mathcal{D}} \leq C_1 \|f\|_2^{\mathcal{D}} \quad \text{and} \quad \|f\|_2^{\mathcal{D}} \leq C_2 \|f\|_\infty^{\mathcal{D}}, \quad (16)$$

where constants C_1 and C_2 are independent of each other in domain \mathcal{D} . Denote the second-order Taylor polynomial for \hat{f} at point \mathbf{p} by $T_{\mathbf{p}}\hat{f}$, where \mathbf{p} is the incenter of tri-box \mathcal{D} . Given that all entries of the Hessian matrix of \hat{f} are zeros, $T_{\mathbf{p}}\hat{f}$ is virtually a linear function. Meanwhile, considering that $l_{\mathcal{D}}$ is the best linear approximation to \hat{f} under the norm $\|\cdot\|_2^{\mathcal{D}}$ (as defined in Eq. (13)), we obtain

$$\begin{aligned} \delta_{l,\mathcal{D}} &= \|\hat{f} - l_{\mathcal{D}}\|_{\text{BB},\infty}^{\mathcal{D}} \\ &\leq C_1 \|\hat{f} - l_{\mathcal{D}}\|_2^{\mathcal{D}} \\ &\leq C_1 \|\hat{f} - T_{\mathbf{p}}\hat{f}\|_2^{\mathcal{D}} \\ &\leq C_1 C_2 \|\hat{f} - T_{\mathbf{p}}\hat{f}\|_\infty^{\mathcal{D}} \\ &\leq \frac{1}{6} C_1 C_2 C_3 (\text{diam}(\mathcal{D}))^3, \end{aligned}$$

where

$$\begin{aligned} C_3 &= \max_{(u,v) \in \mathcal{D}_0} \left(\left| \frac{\partial^3 \hat{f}}{\partial u^3}(u,v) \right| + \left| \frac{\partial^3 \hat{f}}{\partial v^3}(u,v) \right| \right. \\ &\quad \left. + 3 \left| \frac{\partial^3 \hat{f}}{\partial u^2 \partial v}(u,v) \right| + 3 \left| \frac{\partial^3 \hat{f}}{\partial u \partial v^2}(u,v) \right| \right). \end{aligned}$$

Hence, C_l can be set to $\frac{1}{6} C_1 C_2 C_3$.

Lemma 2 For any polynomial f^β , a constant C_q , which depends solely on f^β , exists, such that for any tri-box $\mathcal{D} \subseteq \mathcal{D}_0$, the bound $\delta_{q,\mathcal{D}}$ defined in Eq. (14) satisfies

$$\delta_{q,\mathcal{D}} \leq C_q (\text{diam}(\mathcal{D}))^3. \quad (17)$$

Proof By following the same process in the proof of Lemma 1, we obtain

$$\begin{aligned} \delta_{q,\mathcal{D}} &= \|f^\beta - q_{\mathcal{D}}\|_{\text{BB},\infty}^{\mathcal{D}} \\ &\leq C_1 \|f^\beta - q_{\mathcal{D}}\|_2^{\mathcal{D}} \\ &\leq C_1 \|f^\beta - T_{\mathbf{p}}f^\beta\|_2^{\mathcal{D}} \\ &\leq C_1 C_2 \|f^\beta - T_{\mathbf{p}}f^\beta\|_\infty^{\mathcal{D}} \\ &\leq \frac{1}{6} C_1 C_2 C_4 (\text{diam}(\mathcal{D}))^3, \end{aligned}$$

where $q_{\mathcal{D}}$ is the best quadratic approximation to f^β (as defined in Eq. (14)), $T_{\mathbf{p}}f^\beta$ is the second-order Taylor polynomial for f^β at center \mathbf{p} of tri-box \mathcal{D} , and

$$\begin{aligned} C_4 &= \max_{(u,v) \in \mathcal{D}_0} \left(\left| \frac{\partial^3 f^\beta}{\partial u^3}(u,v) \right| + \left| \frac{\partial^3 f^\beta}{\partial v^3}(u,v) \right| \right. \\ &\quad \left. + 3 \left| \frac{\partial^3 f^\beta}{\partial u^2 \partial v}(u,v) \right| + 3 \left| \frac{\partial^3 f^\beta}{\partial u \partial v^2}(u,v) \right| \right). \end{aligned}$$

Therefore, C_q can be chosen as $\frac{1}{6} C_1 C_2 C_4$.

Lemma 3 For any sequence of tri-boxes $\{\mathcal{D}_i\}_{i=1}^\infty$ generated by recursively applying Algorithm 1, we obtain

$$\begin{cases} \lim_{i \rightarrow \infty} \nabla l_{\mathcal{D}_i}(\mathbf{r}) = \nabla \hat{f}(\mathbf{r}), \\ \lim_{i \rightarrow \infty} \nabla q_{\mathcal{D}_i}(\mathbf{r}) = \nabla f^\beta(\mathbf{r}), \\ \lim_{i \rightarrow \infty} \nabla^2 q_{\mathcal{D}_i}(\mathbf{r}) = \nabla^2 f^\beta(\mathbf{r}). \end{cases} \quad (18)$$

Proof Let us introduce a new type of norm over domain \mathcal{D}_i as presented in Lou and Liu (2012):

$$\|f\|_*^{\mathcal{D}_i} = \|f\|_\infty^{\mathcal{D}_i} + h_i \|\nabla f\|_\infty^{\mathcal{D}_i} + h_i^2 \|\nabla^2 f\|_\infty^{\mathcal{D}_i}, \quad (19)$$

where h_i is the diameter of \mathcal{D}_i . Given that all norms are equivalent in a finite-dimensional vector space,

and a constant M that is independent of the domain \mathcal{D}_i and satisfies $\|r\|_*^{\mathcal{D}_i} \leq M\|r\|_2^{\mathcal{D}_i}$ exists, we obtain

$$\begin{aligned} \|f^\beta - q_{\mathcal{D}_i}\|_*^{\mathcal{D}_i} &= \|f^\beta - q_{\mathcal{D}_i}\|_\infty^{\mathcal{D}_i} + h_i \|\nabla f^\beta - \nabla q_{\mathcal{D}_i}\|_\infty^{\mathcal{D}_i} \\ &\quad + h_i^2 \|\nabla^2 f^\beta - \nabla^2 q_{\mathcal{D}_i}\|_\infty^{\mathcal{D}_i} \\ &\leq M \|f^\beta - q_{\mathcal{D}_i}\|_2^{\mathcal{D}_i} \\ &\leq M \|f^\beta - T_{\mathbf{p}} f^\beta\|_2^{\mathcal{D}_i} \\ &\leq MC_2 \|f^\beta - T_{\mathbf{p}} f^\beta\|_\infty^{\mathcal{D}_i} \\ &\leq \frac{1}{6} MC_2 C_4 h_i^3, \end{aligned} \tag{20}$$

where $T_{\mathbf{p}} f^\beta$, C_2 , and C_4 are defined in the proof of Lemma 2. The expressions following the equality sign and the last inequality sign of Eq. (20) infer that constants M_1 and M_2 , which depend solely on f^β , exist and satisfy

$$\begin{cases} \|\nabla f^\beta - \nabla q_{\mathcal{D}_i}\|_\infty^{\mathcal{D}_i} \leq M_1 h_i^2, \\ \|\nabla^2 f^\beta - \nabla^2 q_{\mathcal{D}_i}\|_\infty^{\mathcal{D}_i} \leq M_2 h_i. \end{cases} \tag{21}$$

For any sequence of tri-boxes $\{\mathcal{D}_i\}_{i=1}^\infty$ generated by recursively applying Algorithm 1, the sequence of diameters tends to approach zero, i.e., $\text{diam}(\mathcal{D}_i) \rightarrow 0$ when $i \rightarrow \infty$. Hence, we obtain

$$\begin{cases} \lim_{i \rightarrow \infty} \nabla q_{\mathcal{D}_i}(\mathbf{r}) = \nabla f^\beta(\mathbf{r}), \\ \lim_{i \rightarrow \infty} \nabla^2 q_{\mathcal{D}_i}(\mathbf{r}) = \nabla^2 f^\beta(\mathbf{r}). \end{cases}$$

The first equality in Eq. (18) can be obtained in the same way.

Theorem 1 If Eq. (6) has a single root (u_0, v_0) within a given triangular domain \mathcal{D} , then the clipping step in Algorithm 1 will generate a sequence of tri-boxes $\{\mathcal{D}_i\}_{i=1}^\infty$ that converges to the root (u_0, v_0) , and the diameters of the successive tri-boxes will satisfy

$$\text{diam}(\mathcal{D}_{i+1}) \leq C(\text{diam}(\mathcal{D}_i))^3 \tag{22}$$

with constant C .

Proof Denote the curved quadrilateral region formed by the fat line \mathcal{L} and the fat curve \mathcal{Q} generated at the i th level of recursion by \mathcal{S}_i . Given that each tri-box \mathcal{D}_{i+1} is obtained from \mathcal{D}_i in the previous recursive level through a clipping operation, we naturally acquire $\mathcal{S}_{i+1} \subseteq \mathcal{D}_{i+1}$ and $\mathcal{D}_{i+1} \subseteq \mathcal{D}_i$. By contrast, a rectangular region \mathcal{B}_{i+1} and an isosceles right triangular region \mathcal{T}_{i+1} that tightly bound the curved quadrilateral region \mathcal{S}_{i+1} always exist, as shown in Fig. 3. Given that $\mathcal{D}_{i+1} \subseteq \mathcal{T}_{i+1}$ and $\text{diam}(\mathcal{T}_{i+1}) \leq 2\text{diam}(\mathcal{B}_{i+1})$, we obtain $\text{diam}(\mathcal{D}_{i+1}) \leq 2\text{diam}(\mathcal{B}_{i+1})$.

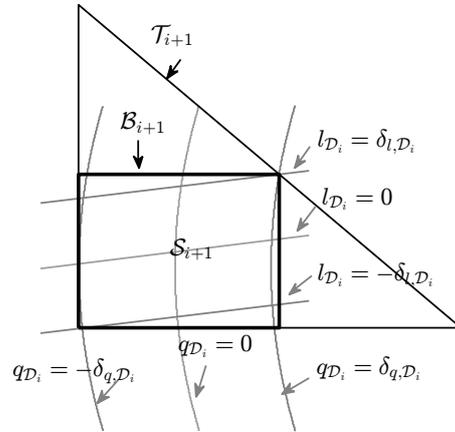


Fig. 3 Quadrilateral region \mathcal{S}_{i+1} formed by fat lines and quadratic fat curves, rectangular bounding box \mathcal{B}_{i+1} , and triangular bounding box \mathcal{T}_{i+1}

Then we prove that $\text{diam}(\mathcal{B}_{i+1}) \leq L(\text{diam}(\mathcal{D}_i))^3$, where L is a constant.

Assume that F is a bivariate function restricted to an arc defined by an implicitly planar curve $G = \text{constant}$. By following the mean value theorem described in Jüttler and Moore (2011), we have that the derivative of F to the arc-length parameter is

$$\nabla F \cdot \frac{(\nabla G)^T}{\|\nabla G\|}, \tag{23}$$

where $(\nabla G)^T$ denotes the transposition of ∇G . Then the distance between any two points \mathbf{P} and \mathbf{Q} on curve $G = \text{constant}$ is less than $|F(\mathbf{P}) - F(\mathbf{Q})|/C$ if the absolute value of Eq. (23) is greater than a positive constant C . Given that $\nabla \hat{f}(\mathbf{r})$ and $\nabla f^\beta(\mathbf{r})$ are linearly independent, as previously assumed, neither $\nabla \hat{f}(\mathbf{r})$ nor $\nabla f^\beta(\mathbf{r})$ is zero in a sufficiently small neighborhood of root \mathbf{r} . By following Lemma 3, we determine that neither $l_{\mathcal{D}_i}$ nor $q_{\mathcal{D}_i}$ is zero in a sufficiently small neighborhood of root \mathbf{r} . In particular, for any $(u, v) \in \mathcal{N}$,

$$\exists L_1 > 0 \text{ s.t. } \left| \nabla l_{\mathcal{D}_i}(u, v) \cdot \frac{(\nabla q_{\mathcal{D}_i}(u, v))^T}{\|(\nabla q_{\mathcal{D}_i}(u, v))^T\|} \right| > L_1$$

and

$$\exists L_2 > 0 \text{ s.t. } \left| \nabla q_{\mathcal{D}_i}(u, v) \cdot \frac{(\nabla l_{\mathcal{D}_i}(u, v))^T}{\|(\nabla l_{\mathcal{D}_i}(u, v))^T\|} \right| > L_2,$$

where both L_1 and L_2 are constants, \mathcal{N} is an appropriate neighborhood of \mathcal{B}_i , $l_{\mathcal{D}_i}$ is the best linear approximation (as defined in Eq. (13)) to \hat{f} , and $q_{\mathcal{D}_i}$

is the best quadratic approximation (as defined in Eq. (14)) to f^β over tri-box \mathcal{D}_i .

Note that \mathcal{B}_{i+1} is the bounding box of the quadrilateral region formed by a pair of lines $l_{\mathcal{D}_i} = \pm\delta_{l,\mathcal{D}_i}$ and a pair of conics $q_{\mathcal{D}_i} = \pm\delta_{q,\mathcal{D}_i}$ (Fig. 3). By applying the mean value theorem to the bivariate functions $l_{\mathcal{D}_i}$ and $q_{\mathcal{D}_i}$, as well as to their restriction curves $q_{\mathcal{D}_i} = 0$ and $l_{\mathcal{D}_i} = 0$ respectively, we determine that the distance between the point in \mathcal{B}_{i+1} and the intersection of $q_{\mathcal{D}_i} = 0$ and $l_{\mathcal{D}_i} = 0$ is no more than $\frac{\delta_{l,\mathcal{D}_i}}{L_1} + \frac{\delta_{q,\mathcal{D}_i}}{L_2}$. In other words, the diameter of \mathcal{B}_{i+1} is smaller than $2\sqrt{2}\left(\frac{\delta_{l,\mathcal{D}_i}}{L_1} + \frac{\delta_{q,\mathcal{D}_i}}{L_2}\right)$, i.e.,

$$\text{diam}(\mathcal{D}_{i+1}) \leq 2\text{diam}(\mathcal{B}_{i+1}) \leq 4\sqrt{2}\left(\frac{\delta_{l,\mathcal{D}_i}}{L_1} + \frac{\delta_{q,\mathcal{D}_i}}{L_2}\right).$$

By following Lemmas 1 and 2, we conclude that $\text{diam}(\mathcal{D}_{i+1}) \leq C(\text{diam}(\mathcal{D}_i))^3$, where $C = 4\sqrt{2}(C_l/L_1 + C_q/L_2)$.

4 Numerical experiments

As mentioned in the introduction, there are mainly four types of methods to solve the RPI problem, i.e., subdivision, approximation, Newton’s method, and geometric clipping method. Subdivision and approximation methods suffer from high memory requirement and computational cost. Newton’s method is not unconditionally stable, where an initial guess sufficiently close to the solution is needed to guarantee the convergence. By contrast, geometric clipping methods are stable and efficient, in the sense that they run in a reasonable amount of time and have a relatively low requirement on memory usage. Among the geometric clipping methods, only the so-called Bézier clipping method in Roth *et al.* (2000) was proposed to address the RPI problem. Hence, in this section, we compare our HC method with the Bézier clipping method. Besides, we report the results generated by the linear clipping method and the HC without preprocessing method, both of which are variants of the HC method. According to the analysis in Section 2.2, the original RPI problem can be converted into a problem of solving a bivariate system of two equations. Hence, we test all the algorithms directly on the converted problem. The computation time and the number of recursions for each method to reach the same accuracy are reported. All the algorithms have been implemented

in MATLAB and run on a PC with a 3.1 GHz Intel i5 processor and 4 GB memory.

In the first example, the converted problem involves finding the root of a system with a single root.

Example 1

$$\begin{cases} f(u, v) = u^3 - v = 0, \\ g(u, v) = u^3 + v - 1/4 = 0, \end{cases} \quad (24)$$

where $\mathcal{D}_0 = \{(u, v) | 0 \leq u, v \leq 1, 0 \leq u + v \leq 1\}$.

Fig. 4a illustrates surfaces $f = f(u, v)$ and $g = g(u, v)$, wherein the black line denotes the intersection curve in three dimensions. Fig. 4b shows the curves $f = 0$ and $g = 0$ and their intersections. The statistics of the number of recursions for different methods to reach various prespecified thresholds ϵ are summarized in Table 1. HC and HC without

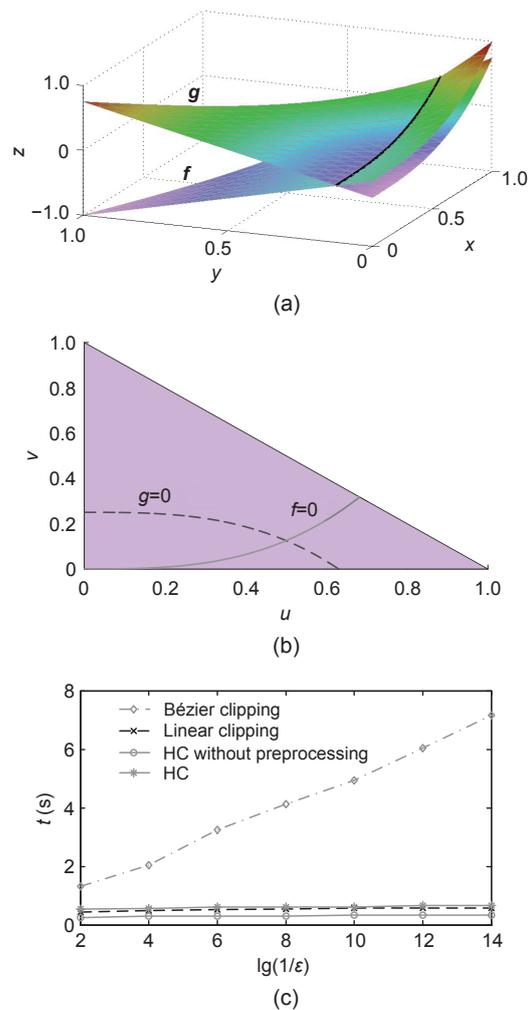


Fig. 4 Example 1: (a) surfaces f and g ; (b) curves $f = 0$ and $g = 0$; (c) computation time vs. accuracy

preprocessing methods generally require fewer recursion levels than the linear clipping method and the Bézier clipping method to reach the same threshold. Fig. 4c presents the computation time reached at each level of accuracy. We can observe that the HC algorithm requires fewer recursions but longer running time than the HC without preprocessing algorithm and linear clipping method to achieve the same accuracy. Table 2 shows the diameters of the tri-boxes obtained at each level of recursion of the HC and HC without preprocessing algorithms. We can see that the former converges faster than the latter. This phenomenon can be explained by the fact that preprocessing is time-consuming but helpful in improving the convergence rate.

Table 1 Statistics of the number of recursions for different methods to reach the same prespecified accuracy in Example 1

Method	Number of recursions						
	lg $\epsilon = -2$	-4	-6	-8	-10	-12	-14
Bézier clipping	9	15	22	29	35	42	49
Linear clipping	4	5	6	6	7	7	7
HC without pre.	4	5	5	5	6	6	6
HC	3	4	5	5	5	6	6

HC: hybrid clipping; HC without pre.: hybrid clipping without preprocessing

Table 2 Diameters of the tri-boxes at each level of recursion of the HC and HC without preprocessing algorithms in Example 1

Level	diam(\mathcal{D}_i)	
	HC without preprocessing	HC
1	0.51	0.45
2	0.17	0.13
3	1.52×10^{-2}	2.65×10^{-3}
4	1.10×10^{-4}	3.40×10^{-6}
5	5.75×10^{-9}	1.48×10^{-12}
6	1.24×10^{-16}	8.32×10^{-17}

HC: hybrid clipping

In the second example, we test the behavior of our algorithm by a system with three single roots.

Example 2

$$\begin{cases} f(u, v) = 10(u - 1/2)(u - 1/4)(u - 1/8) \\ \quad + 1/4 - v = 0, \\ g(u, v) = -10(u - 1/2)(u - 1/4)(u - 1/8) \\ \quad + 1/4 - v = 0, \end{cases} \quad (25)$$

where $\mathcal{D}_0 = \{(u, v) | 0 \leq u, v \leq 1, 0 \leq u + v \leq 1\}$.

Fig. 5a shows curves $f = 0$ and $g = 0$ along with their intersections. Table 3 reports the statistics of the number of recursions for different methods to reach various prespecified thresholds ϵ . The same result as that in Example 1 is observed; i.e., the HC and HC without preprocessing algorithms require fewer recursion levels than the linear clipping method and the Bézier clipping method to reach the same threshold. Table 4 lists the diameters of the tri-boxes at each level of recursion of the HC and HC without preprocessing algorithms at root $(1/8, 1/4)$. We can observe that the former converges faster than the latter. Fig. 5b presents the computation time vs. accuracy of Example 2.

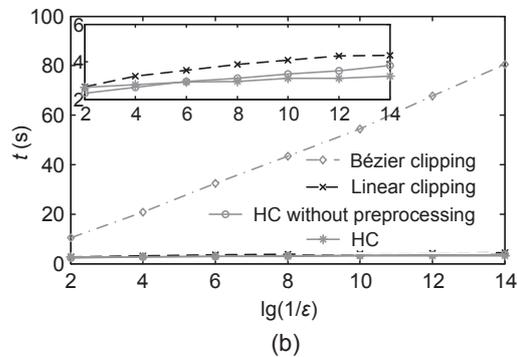
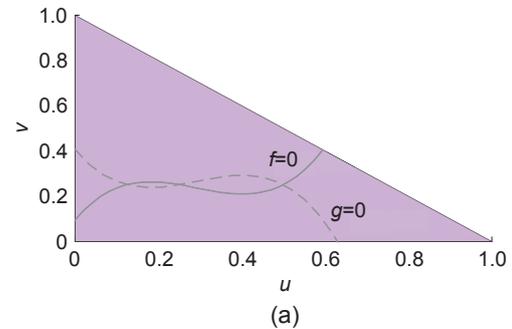


Fig. 5 Example 2: (a) curves $f = 0$ and $g = 0$; (b) computation time vs. accuracy (the box in the upper left corner is a zoom in view of the results of linear clipping, HC without preprocessing, and HC methods)

In the third example, we test the behavior of our algorithm by a system with two roots or a double root.

Example 3

$$\begin{cases} f(u, v, k) = u^2 + v^2 - 1/4(1 + 1/10^k) = 0, \\ g(u, v) = uv - 1/8 = 0, \end{cases} \quad (26)$$

where k is a non-negative integer and $\mathcal{D}_0 =$

$$\{(u, v) | 0 \leq u, v \leq 1, 0 \leq u + v \leq 1\}.$$

This system has two real roots for all constant k and a double root when $k = \infty$; i.e., the root of the system changes from a pair of different roots to a double root when k increases from 0 to infinity. Thus, the larger the k , the closer the roots are to double. Fig. 6 plots the curves defined by systems with $k = 0, 2, 5$.

Table 5 reports the statistics of the number of recursions for different methods to reach various prespecified thresholds ϵ for the systems with $k = 0, 2, 5$. The HC method requires fewer recursion levels to achieve a prespecified accuracy than the other methods. When k increases, the two roots of the system become close to a double root, and

Table 3 Statistics of the number of recursions for different methods to reach the same prespecified accuracy in Example 2

Method	Number of recursions						
	lg $\epsilon = -2$	-4	-6	-8	-10	-12	-14
Bézier clipping	9	15	22	29	35	42	49
Linear clipping	6	7	8	8	9	9	9
HC without pre.	5	6	7	7	8	8	8
HC	5	5	6	6	6	7	7

HC: hybrid clipping; HC without pre.: hybrid clipping without preprocessing

Table 4 Diameters of the tri-boxes at each level of recursion of the HC and HC without preprocessing algorithms in Example 2

Level	diam(\mathcal{D}_i)	
	HC without preprocessing	HC
1	0.70	0.70
2	0.35	0.35
3	0.13	0.11
4	4.3×10^{-2}	1.47×10^{-2}
5	4.21×10^{-3}	7.86×10^{-5}
6	3.35×10^{-5}	2.05×10^{-11}

HC: hybrid clipping

Table 5 Statistics of the number of recursions for different methods to reach the same prespecified accuracy in Example 3

Method	Number of recursions ($k = 0$)							Number of recursions ($k = 2$)							Number of recursions ($k = 5$)						
	lg $\epsilon = -2$	-4	-6	-8	-10	-12	-14	-2	-4	-6	-8	-10	-12	-14	-2	-4	-6	-8	-10	-12	-14
Bézier clipping	9	15	22	29	35	42	48	9	15	22	29	35	42	49	9	14	21	28	34	41	48
Linear clipping	5	6	7	7	8	8	8	6	8	9	9	10	10	10	7	10	12	12	13	13	16
HC without pre.	5	6	6	7	7	7	7	5	7	7	8	8	9	9	6	9	10	11	12	12	13
HC	4	5	6	6	6	6	6	4	5	6	6	6	6	6	5	6	6	6	7	7	7

HC: hybrid clipping; HC without pre.: hybrid clipping without preprocessing

thus the advantages of the HC method become more obvious. Table 6 provides the diameters of the tri-boxes at each level of recursion of the HC and HC without preprocessing methods. Again, the former method provides faster convergence than the latter in almost all the cases, particularly when k is large; i.e., the system approximately has a double root. Fig. 7 plots the computation time versus the number of recursive levels for the cases of $k = 0, 2, 5$. The HC method becomes more efficient than the other three methods when k increases.

5 Conclusions

In this paper, we proposed a new technique called hybrid clipping (HC) to solve the RPI problem. We converted the RPI problem into an equivalent problem that solves a system of two bivariate equations. The system was solved in a geometric manner, i.e., by combining geometric properties such as the convex hull property of a triangular Bézier surface and Descartes' rule of signs as well as using geometric operations such as surface degree reduction

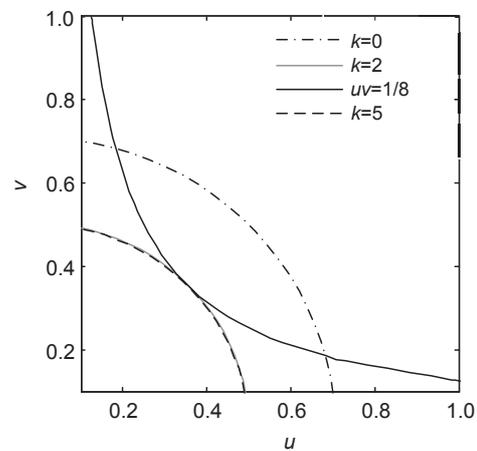


Fig. 6 Curves of the systems with $k = 0, 2,$ and 5 in Example 3

Table 6 Diameters of the tri-boxes at each level of recursion of the HC and HC without preprocessing algorithms in Example 3

Level	diam(D_i) ($k=0$)		diam(D_i) ($k=2$)		diam(D_i) ($k=5$)	
	HC without pre.	HC	HC without pre.	HC	HC without pre.	HC
1	0.70	0.70	0.70	0.70	0.70	0.70
2	0.31	0.35	0.20	0.20	0.20	0.20
3	0.11	0.12	8.95×10^{-2}	7.88×10^{-2}	8.16×10^{-2}	7.04×10^{-2}
4	1.57×10^{-2}	4.21×10^{-3}	2.48×10^{-2}	4.17×10^{-3}	3.07×10^{-2}	9.12×10^{-3}
5	2.76×10^{-4}	1.79×10^{-7}	6.68×10^{-3}	6.54×10^{-7}	1.14×10^{-2}	1.32×10^{-4}
6	8.50×10^{-8}	1.11×10^{-16}	3.43×10^{-4}	5.55×10^{-17}	4.33×10^{-3}	3.92×10^{-10}
7	8.12×10^{-15}		8.95×10^{-7}		1.90×10^{-3}	5.55×10^{-17}
8			6.06×10^{-12}		5.14×10^{-4}	
9					6.08×10^{-5}	

HC: hybrid clipping; HC without pre.: hybrid clipping without preprocessing

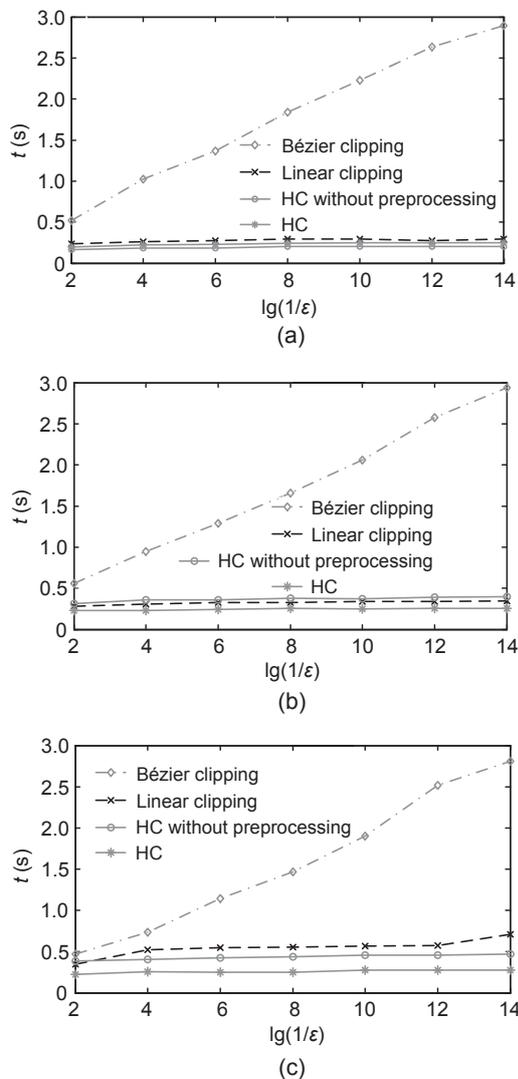


Fig. 7 Computation time vs. accuracy of different methods with $k = 0$ (a), $k = 2$ (b), and $k = 5$ (c) in Example 3

and subdivision. We proved that the HC method has a cubic convergence rate for a single root.

Numerical experiments showed that the HC without preprocessing method is efficient for the single-root case; hence, the time-consuming preprocessing step becomes unnecessary. Meanwhile, for systems with a double root or with two different roots that are close to each other, the HC method has a significant advantage over the other methods because it requires fewer recursion levels and shorter running time than the other methods to achieve a prespecified accuracy. However, we currently have no advanced idea on the number and multiplicity of the roots of systems. Hence, one of our future works is to provide an appropriate indicator to turn on/off the preprocessing step and produce the best trade-off between preprocessing time and total computation time.

References

- Barth, W., Stürzlinger, W., 1993. Efficient ray tracing for Bézier and B-spline surfaces. *Comput. Graph.*, **17**(4):423-430. [http://dx.doi.org/10.1016/0097-8493\(93\)90031-4](http://dx.doi.org/10.1016/0097-8493(93)90031-4)
- Bartoň, M., Jüttler, B., 2007a. Computing roots of polynomials by quadratic clipping. *Comput. Aided Geom. Des.*, **24**(3):125-141. <http://dx.doi.org/10.1016/j.cagd.2007.01.003>
- Bartoň, M., Jüttler, B., 2007b. Computing Roots of Systems of Polynomials by Linear Clipping. SFB F013 Technical Report.
- Garloff, J., Smith, A.P., 2001. Investigation of a subdivision based algorithm for solving systems of polynomial equations. *Nonl. Anal. Theory Methods Appl.*, **47**(1):167-178. [http://dx.doi.org/10.1016/S0362-546X\(01\)00166-3](http://dx.doi.org/10.1016/S0362-546X(01)00166-3)
- Haines, E., Hanrahan, P., Cook, R.L., et al., 1989. An Introduction to Ray Tracing. Academic Press, London, UK.

- Hanrahan, P., 1983. Ray tracing algebraic surfaces. *ACM SIGGRAPH Comput. Graph.*, **17**(3):83-90. <http://dx.doi.org/10.1145/964967.801136>
- Joy, K.I., Grant, C.W., Max, N.L., et al., 1989. Tutorial: Computer Graphics, Image Synthesis. IEEE Computer Society Press, Los Alamitos, CA, USA.
- Jüttler, B., Moore, B., 2011. A quadratic clipping step with superquadratic convergence for bivariate polynomial systems. *Math. Comput. Sci.*, **5**(2):223-235. <http://dx.doi.org/10.1007/s11786-011-0091-4>
- Liu, L., Zhang, L., Lin, B., et al., 2009. Fast approach for computing roots of polynomials using cubic clipping. *Comput. Aided Geom. Des.*, **26**(5):547-559. <http://dx.doi.org/10.1016/j.cagd.2009.02.003>
- Lou, Q., Liu, L., 2012. Curve intersection using hybrid clipping. *Comput. Graph.*, **36**(5):309-320. <http://dx.doi.org/10.1016/j.cag.2012.03.021>
- Lu, L., Wang, G., 2006. Multi-degree reduction of triangular Bézier surfaces with boundary constraints. *Comput.-Aided Des.*, **38**(12):1215-1223. <http://dx.doi.org/10.1016/j.cad.2006.07.004>
- Markus, G., Oliver, A., 2005. Interactive ray tracing of trimmed bicubic Bézier surfaces without triangulation. Proc. 13th Int. Conf. in Central Europe on Computer Graphics, Visualization and Computer Vision, p.71-78.
- Martin, W., Cohen, E., Fish, R., et al., 2000. Practical ray tracing of trimmed NURBS surfaces. *J. Graph. Tools*, **5**(1):27-52. <http://dx.doi.org/10.1080/10867651.2000.10487519>
- Moore, R.E., Jones, S.T., 1977. Safe starting regions for iterative methods. *SIAM J. Numer. Anal.*, **14**(6):1051-1065. <http://dx.doi.org/10.1137/0714072>
- Nishita, T., Sederberg, T.W., Kakimoto, M., 1990. Ray tracing trimmed rational surface patches. *ACM SIGGRAPH Comput. Graph.*, **24**(4):337-345. <http://dx.doi.org/10.1145/97880.97916>
- Roth, S.H.M., Diezi, P., Gross, M.H., 2000. Triangular Bézier clipping. Proc. 8th Pacific Conf. on Computer Graphics and Applications, p.413-414. <http://dx.doi.org/10.1109/PCCGA.2000.883971>
- Rouillier, F., Zimmermann, P., 2004. Efficient isolation of polynomial's real roots. *J. Comput. Appl. Math.*, **162**(1):33-50. <http://dx.doi.org/10.1016/j.cam.2003.08.015>
- Schulz, C., 2009. Bézier clipping is quadratically convergent. *Comput. Aided Geom. Des.*, **26**(1):61-74. <http://dx.doi.org/10.1016/j.cagd.2007.12.006>
- Sederberg, T.W., Nishita, T., 1990. Curve intersection using Bézier clipping. *Comput.-Aided Des.*, **22**(9):538-549. [http://dx.doi.org/10.1016/0010-4485\(90\)90039-F](http://dx.doi.org/10.1016/0010-4485(90)90039-F)
- Stürzlinger, W., 1998. Ray-tracing triangular trimmed free-form surfaces. *IEEE Trans. Vis. Comput. Graph.*, **4**(3):202-214. <http://dx.doi.org/10.1109/2945.722295>
- Sweeney, M.A.J., Bartels, R.H., 1986. Ray tracing free-form B-spline surfaces. *IEEE Comput. Graph. Appl.*, **6**(2):41-49. <http://dx.doi.org/10.1109/MCG.1986.276691>
- Toth, D.L., 1985. On ray tracing parametric surfaces. *ACM SIGGRAPH Comput. Graph.*, **19**(3):171-179. <http://dx.doi.org/10.1145/325334.325233>
- Woodward, C., 1989. Ray tracing parametric surfaces by subdivision in viewing plane. In: Straßer, W., Seidel, H.P. (Eds.), Theory and Practice of Geometric Modeling. Springer Berlin Heidelberg, Germany, p.273-287. http://dx.doi.org/10.1007/978-3-642-61542-9_18
- Yen, J., Spach, S., Smith, M.T., et al., 1991. Parallel boxing in B-spline intersection. *IEEE Comput. Graph. Appl.*, **11**(1):72-79. <http://dx.doi.org/10.1109/38.67703>
- Zhang, R.J., Wang, G., 2005. Constrained Bézier curves' best multi-degree reduction in the L_2 -norm. *Progr. Nat. Sci.*, **15**(9):843-850. <http://dx.doi.org/10.1080/10020070512331343010>