The International Journal of Advanced Manufacturing Technology
https://doi.org/10.1007/s00170-026-17394-1

ORIGINAL ARTICLE

An efficient cutter-facet model projection method and applications to five-axis tool path computation

Xiyan Li[1] · Pengcheng Hu[1] · Lulu Huang[1] · Haokun Chen[1] · Molong Duan[2] · Xinfang Zhang[1]

Received: 14 November 2025 / Accepted: 31 December 2025

Abstract
In five-axis machining, generating gouge-free tool paths efficiently is essential. However, the projection method – widely regarded as an effective approach for gouge-free machining – remains largely applied to fixed tool-axis tool-path computation due to practical difficulties. This paper presents a comprehensive framework for projecting a cutter along arbitrary direction onto triangulated models, focusing on resolving the critical torus-edge tangency problem. We demonstrate that for two specific cases, tool-axis-aligned torus-edge projection and arbitrary-direction ball-end cutter projection, the problem can be reduced to a quartic equation and solved analytically, ensuring both robustness and computational efficiency. For the general case of arbitrary-direction torus-edge projection, we develop three tailor-made numerical methods: Bézier clipping, edge-search, and torus-search. The proposed method is validated through two applications: gouge avoidance in blade machining and gouge-free iso-planar tool path generation. Simulations and physical machining experiments confirm the method's practical effectiveness and computational performance, demonstrating its readiness for industrial use.

Keywords Five-axis machining · Tool path generation · Projection method · Torus-edge tangency · Gouge-free machining

1 Introduction

Five-axis machining offers distinct advantages over its three-axis counterpart, including fewer setups, enhanced tool accessibility, reduced machining time, and improved surface finish [1, 2]. These capabilities make it indispensable for manufacturing complex freeform components such as aero-engine blades and automobile molds [1]. However, the generation of high-quality, gouge-free tool paths remains a critical challenge. This is particularly true when leveraging high-performance non-spherical cutters (e.g., toroidal cutters) and the full flexibility of arbitrary tool orientations – both essential for optimizing the trade-off between machining efficiency, surface finish, and geometric accuracy.

This paper addresses the core computational bottleneck in achieving this goal: the arbitrary-direction torus-edge projection problem. We develop a comprehensive framework for arbitrary-direction cutter-facet projection to bridge the gap between the inherent gouge-avoidance of projection methods and the demands of versatile five-axis machining. To contextualize this work, we review the state of the art, drawing on key literature to map the evolution of methods, highlight persistent limitations, and clarify the motivation for our integrated approach.

1.1 State-of-the-art in five-axis tool path optimization

The evolution of five-axis tool path generation can be traced along three interrelated technical directions: cutter selection and modeling, path generation strategies, and auxiliary optimization.

1.1.1 Cutter selection and modeling

The choice of cutter geometry is pivotal for machining precision and efficiency. Toroidal cutters (also referred to as

✉ Pengcheng Hu
pengchenghu@hkust-gz.edu.cn

[1] Hong Kong University of Science and Technology (Guangzhou), Guangzhou, China

[2] Hong Kong University of Science and Technology, Hong Kong, China

Published online: 13 February 2026

fillet [3] or radius cutters [4]) are widely adopted for free-form surfaces due to their superior performance in reducing scallop height and improving surface roughness compared to ball-end or flat-bottom cutters [4, 5]. Bedi et al. [4] provided direct experimental evidence of these advantages, while Jensen et al. [5] advanced the field with "curvature-matched machining," tailoring tool parameters to the local part geometry.

Subsequent research has strengthened the mathematical foundation for cutter modeling. Lee [6] established robust models for various endmills and addressed tool placement in multi-axis contexts. The optimal tool shape selection method proposed by Patel et al. [7] for three-axis machining was later extended to five-axis scenarios by Li and Tang [8] via a partition-based strategy. Despite this progress, a critical capability remains underdeveloped: modeling the projection of non-spherical cutters along arbitrary direction, which is fundamental to exploiting the full flexibility of five-axis systems. Most current methods remain confined to fixed tool-axis orientations [8, 9].

1.1.2 Path generation strategies

Tool path generation strategies are typically categorized into three dominant paradigms (Fig. 1), each with distinct strengths and limitations.

The CL-surface strategy generates paths from a "cutter location (CL) surface" defined by tangential contact points. While it incorporates interference detection, its applicability is largely restricted to three-axis machining [10, 11]. For five-axis machining with non-spherical cutters, the CL surface lacks a unique definition due to the additional rotational degrees of freedom, fundamentally limiting its utility. Attempts to extend it, such as the curve-based approach by Jun et al. [10] for polyhedral models, have confirmed its inadequacy for non-spherical tools in five-axis contexts.

The contact-driven strategy, which dominates industrial practice, follows a "CC-to-CL" paradigm: it first generates a sequence of cutter contact (CC) points on the part surface and then computes the corresponding CL points for a predefined tool orientation. Its numerous variants highlight both its versatility and its inherent challenges, which can be examined in three key areas.

The first area concerns path pattern generation, which aims to plan the distribution of CC points on the surface. In this category, iso-parametric methods [12, 13] often result in uneven path spacing in regions of high curvature. Iso-planar methods [14–16] effectively control scallop height but typically rely on fixed tool orientations. Ma et al. [17] improved path smoothness by integrating feed-direction vector fields, though their adaptability was limited by pre-defined patterns. Meanwhile, iso-scallop methods [11, 18–20] provide excellent surface quality control, but the post-hoc corrections for scallop violations required by methods like [21] introduce significant computational overhead.

Besides the widely adopted iso-parametric, iso-planar, and iso-scallop strategies, "sweep path generation" has also emerged as an efficient approach for machining

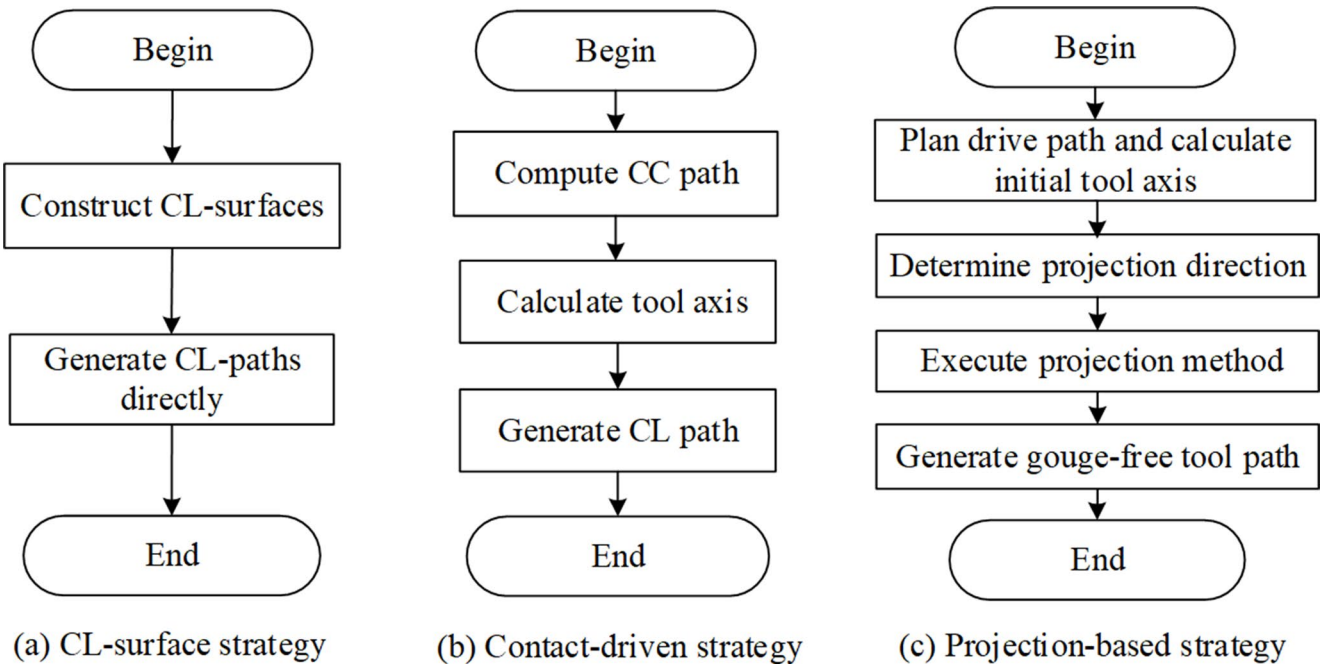


Fig. 1 Three tool-path computation strategies

elongated or structured surfaces. For instance, Hu et al. [22] proposed a spiral curve-based method for efficient five-axis sweep scanning of barrel-shaped surfaces, which optimizes tool motion continuity and machining efficiency. Zhang et al. [23] further extended this concept by introducing a skeleton curve-guided sweep scanning strategy for surfaces with multiple holes, enhancing path adaptability in complex geometries. Moreover, the integration of in-process machining data, as modeled by Zhou et al. [24], enables real-time adaptation and optimization of sweep paths, contributing to improved machining accuracy and tool life. These studies illustrate the ongoing diversification and refinement of path generation strategies, particularly in handling structured and repetitive surface features.

The second area focuses on tool orientation planning. Field-based methods, such as the machining potential field introduced by Chiou et al. [25] and extended by Hu et al. [26] to machine constraints, offer a powerful guiding principle. Yet, as noted in Makhanov's survey [27], they have not fully resolved the challenge of arbitrary tool orientations. Boundary-conformed and offset methods [28–30] also struggle with the tool orientation variability inherent in five-axis machining.

The third area involves cutter-contact computation for discrete models, addressing the fundamental geometric problem of calculating the contact point between a general cutter and a triangulated model. This includes methods for positioning generalized cutters on STL models [31], solutions to the critical torus-edge tangency problem [32] – a central computational challenge for non-spherical cutters – and the multi-point machining approaches by Duvedi et al. [33–35] which refine tool positioning by satisfying multiple contact constraints simultaneously.

The projection-based strategy inverts the logic of contact-driven methods. It follows a "Drive-path-projection" paradigm: a cutter is projected from a predefined drive path onto the part surface along a specified direction (Fig. 2). This process inherently preserves the drive path's regularity and ensures gouge-free results by halting at the first point of tangential contact.

The concept of arbitrary-direction projection was pioneered by Hansen et al. [36], though no implementation details were provided. Subsequent research largely focused on computationally simpler tool-axis-aligned projection, suitable for three-axis machining. Even recent methods for collision avoidance often rely on fixed tool orientations, as demonstrated by Cao et al. [37] in their

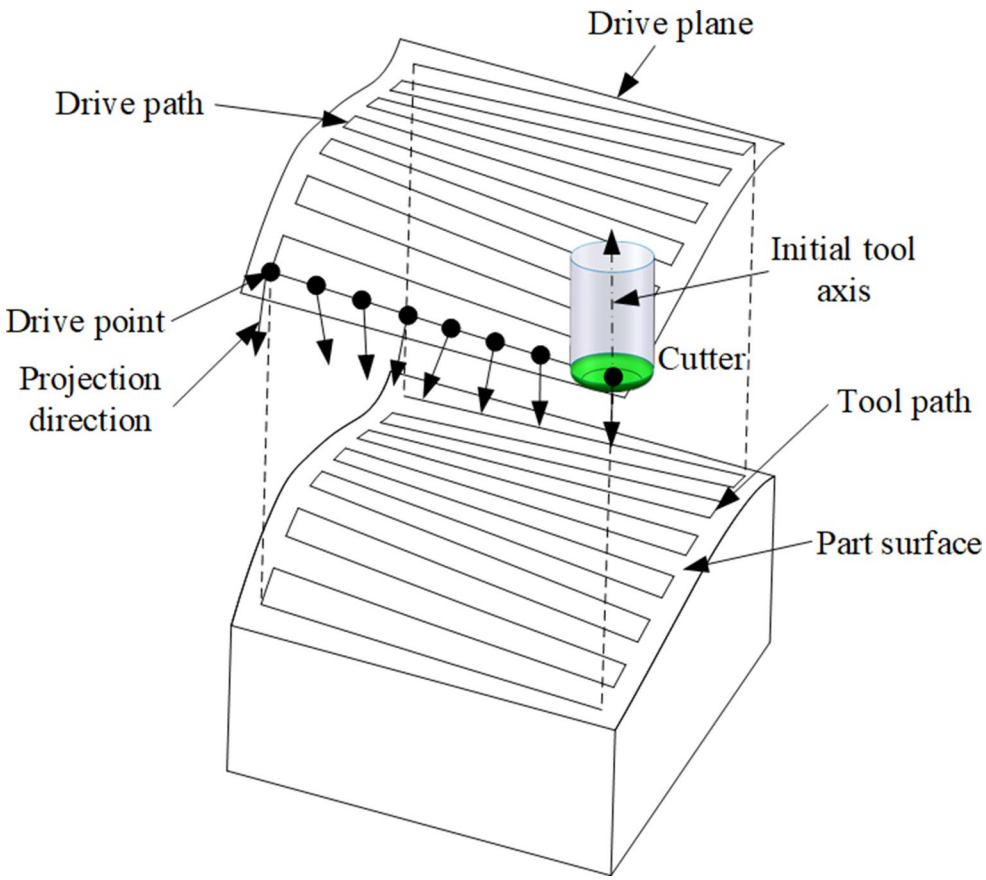


Fig. 2 Schematic of the projection-based tool path generation strategy

geometric decoupling strategy for three rotary axes milling heads. While robust algebra-based solvers like Bézier clipping [38–40] can solve the high-order equations arising from these projections, their computational expense often precludes real-time application in five-axis contexts. Extensions to five-axis machining have been constrained; for instance, the linear morphing cone approach by Zhang et al. [41] effectively employs a fixed projection direction, failing to leverage the full flexibility of five-axis systems.

Consequently, a generalized, efficient, and truly arbitrary-direction projection framework for five-axis machining with non-spherical cutters remains an open challenge. The core of this challenge, the arbitrary-direction torus-edge tangency problem, is the central focus of this paper.

1.1.3 Auxiliary optimization

To enhance overall path quality, significant research efforts have been directed toward three key auxiliary optimization areas alongside primary path generation methods.

Research in tool attitude smoothing has developed effective strategies for managing tool orientation. Wu et al. [42] established a "Safe Attitude Corridor" through convex optimization to prevent abrupt tool movements, while Lee [43] created admissible tool orientation controls specifically for gouge avoidance. Despite these advances, these smoothing techniques typically operate as post-processing steps rather than being integrated directly with path generation, which can result in somewhat disjointed tool motions.

The field of accessibility analysis has seen notable progress through different computational approaches. Xie et al. [44] successfully accelerated tool accessible space computation using programmable graphics pipelines, though their method remained limited to standard ball-end cutters. Meanwhile, Duvedi et al. [33–35] employed multi-point machining strategies to address accessibility concerns for triangulated surfaces, yet their approach encountered significant challenges when applied to non-spherical cutting tools.

Intelligent algorithms represent a growing area of interest for path optimization challenges. Wan et al. [45] and Zhang et al. [46] independently demonstrated the adaptability of reinforcement learning for path planning and orientation optimization respectively, though both approaches share limitations including dependence on extensive training data and insufficient real-time performance. Notably, recent advances in neural computation have shown promise for manufacturing optimization; for instance, Liu et al. [47] developed a neural co-optimization method for fiber-reinforced composites that simultaneously considers structural topology, manufacturable layers, and path orientations. In a different application, Demir et al. [48] implemented NSGA-II to optimize non-cutting operations such as tool changes, representing a valuable contribution that nevertheless focuses on auxiliary processes rather than the fundamental task of cutting path generation.

1.2 Critical limitations of existing methods

Despite decades of progress, the development of versatile five-axis tool path generation continues to face several interconnected limitations that restrict its full potential.

Existing projection-based and contact-driven methods for non-spherical cutters remain constrained by their reliance on fixed projection directions, typically aligned with the tool axis [8, 9, 31–36]. This fundamental design characteristic inherently restricts their adaptability to the complete spectrum of arbitrary orientations, thereby limiting their effectiveness in complex curved regions where five-axis systems should offer maximum flexibility.

A persistent challenge emerges from the unavoidable trade-off between computational performance and methodological generality. Geometry-based approaches such as projection and offset methods deliver high precision but tend to be specialized for specific cutter types [44]. Partition-based strategies like the 3 + 2-axis method by Hao et al. [49] effectively handle non-spherical tools but still operate within fixed orientation constraints. Meanwhile, data-driven intelligent algorithms [45, 46] demonstrate impressive adaptability while suffering from substantial training data requirements and limited real-time responsiveness. Although algebra-based numerical methods [38, 40] provide solutions to high-order equations, their computational efficiency is compromised by the need for extensive iterative calculations.

Perhaps the most significant barrier to advancement lies in the current fragmented approach to multi-objective optimization. Critical objectives including tool attitude smoothing [42], scallop height control [21], and gouge avoidance [9] are typically addressed through sequential or isolated optimization processes. This lack of integrated treatment frequently results in disjointed tool motions, necessitates corrective post-processing operations, and ultimately compromises both path quality and machining efficiency.

These limitations collectively highlight the need for a more unified and adaptable framework that can simultaneously address directional flexibility, computational efficiency, and multi-objective integration in five-axis tool path generation.

1.3 Scope and contribution of this work

This paper develops a comprehensive arbitrary-direction cutter-facet projection framework specifically designed to overcome the limitations outlined above. Our core contributions are threefold:

First, we propose a unified framework for projection along arbitrary direction. Our framework supports genuine projection along arbitrary direction for toroidal and non-spherical cutters. A key innovation is the strategic calculation of contact points within the Cutter Coordinate System (CCS) prior to transformation to the Workpiece Coordinate System (WCS). This approach reduces computational complexity while fully enabling the adaptability required for five-axis tool orientation variability.

Second, we introduce a suite of hybrid analytical-numerical solutions. For two critical special cases – tool-axis-aligned torus-edge projection and arbitrary-direction ball-edge projection – we derive analytical solutions by reducing the inherently eighth-order equations to solvable quartic forms, thereby eliminating the risks and iterations of numerical root-finding. For the general case of arbitrary-direction torus-edge projection, we propose three custom, efficient numerical methods: an algebra-based Bézier clipping method, adapting to the specific geometry of the torus-edge problem; a geometry-based edge-search method that leverages a gradient-based secant search to circumvent the high-order equations entirely; and a geometry-based torus-search method that minimizes iterations by focusing search efforts directly on the torus curve parameters.

Third, our framework achieves integrated multi-objective control. It provides inherent gouge avoidance through the first-contact nature of projection. Its seamless integration with iso-planar path planning directly ensures constant scallop height, eliminating the need for the post-hoc corrections. Furthermore, by aligning the tool attitude with the projection direction, we directly address the disjoint motion issue present in auxiliary smoothing techniques, achieving integrated optimization.

The remainder of this paper is organized as follows: Section 2 formulates the general cutter-facet projection framework and details the derivation of analytical solutions. Section 3 is dedicated to the three numerical methods for general arbitrary-direction projection. Section 4 presents industrial applications, including a gouge avoidance algorithm for blade machining and an iso-planar tool path strategy for molds. Section 5 provides comprehensive validation through simulation and physical machining experiments. Finally, Section 6 offers conclusions and outlines future research directions.

2 Preliminaries

This section formulates the problem of projecting a cutter onto a facet model, with particular emphasis on torus-edge projection. It begins with a description of the general APT cutter and its cutting regions. The projection problem is classified into three categories: cutter-facet, cutter-edge, and cutter-vertex, with the principles of the first two being examined in detail. The discussion then focuses on the central challenge of torus-edge tangency. A mathematical model is derived, resulting in an eighth-order equation for an arbitrary projection direction. Finally, it is shown that for projections aligned with the tool axis or involving a ball-end cutter, this equation reduces to a quartic, which is solved analytically.

2.1 Projection method of a cutter onto facet models

This section details the core framework for generating gouge-free tool paths via cutter projection. It covers the cutter geometry, the fundamental projection algorithm, the hierarchical search strategy for contact points, and culminates in identifying the central computational challenge: the torus-edge tangency problem.

- **Cutter Geometry and Projection Principle**

The geometric configuration of a general milling cutter and its associated cutting regions are delineated in Fig. 3. The APT cutter's cross-sectional profile, shown in Fig. 3(a), is characterized by a sequence of linear and circular segments. The revolution of a circular segment generates a toroidal surface, whereas a linear segment generates a conical surface. Furthermore, the center coordinates (d, h) for the circular segments are calculated from the six standard parameters: total cutter height H, the flute length L, the tip angle A_1, the tapper angle A_2, the diameter D, and the corner radius R. The corresponding cutting regions, presented in Fig. 3(b), encompass upper and lower conical regions as well as toroidal regions. It should be noted that the lower conical region ceases to exist when $A_1 = 0$. A toroidal region functions as a cutting region only when $R > 0$, and the upper conical region is non-existent when $A_2 \leq 0$.

For a given cutter location on the drive path, the cutter is projected onto the facet model along a specified direction. This process begins by constructing an axis-aligned bounding box (AABB) along the projection vector, as illustrated in Fig. 4. Facets intersected by the AABB are selected as candidates. Each candidate facet is then tested, and the projection distance is computed. The point with

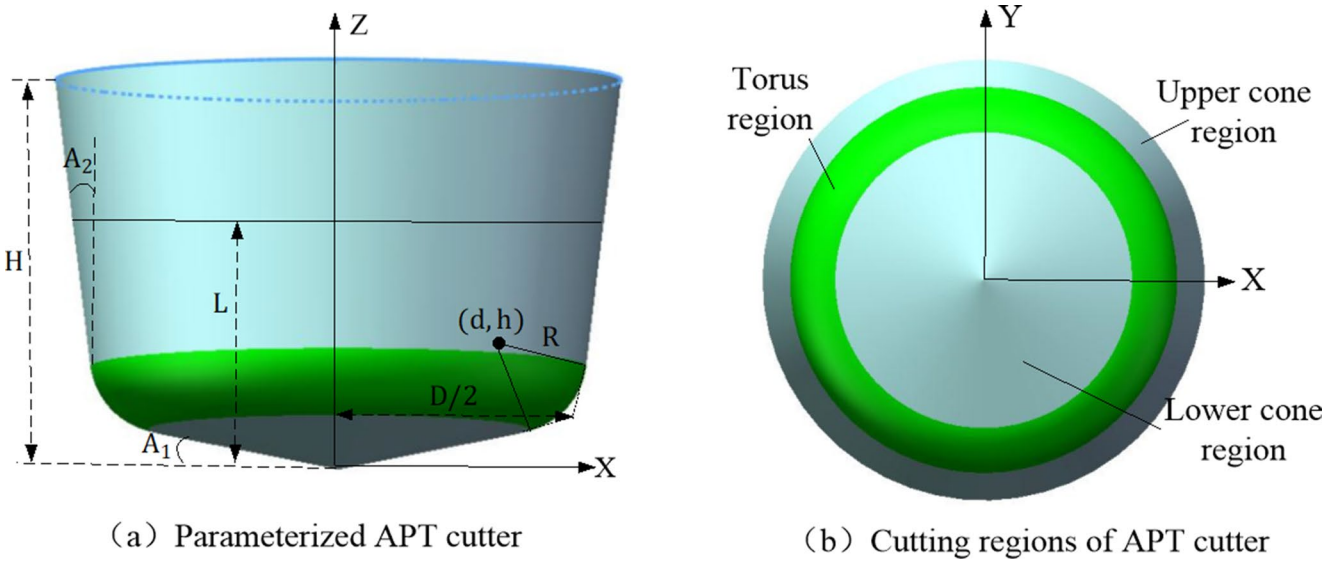


Fig. 3 APT cutter profile and its cutting region definition

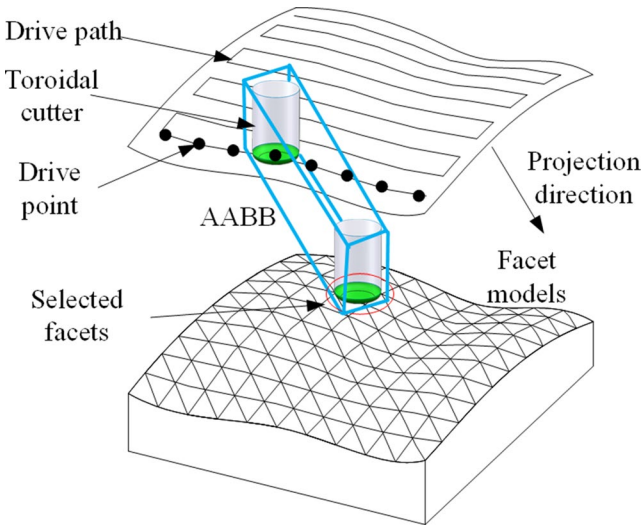
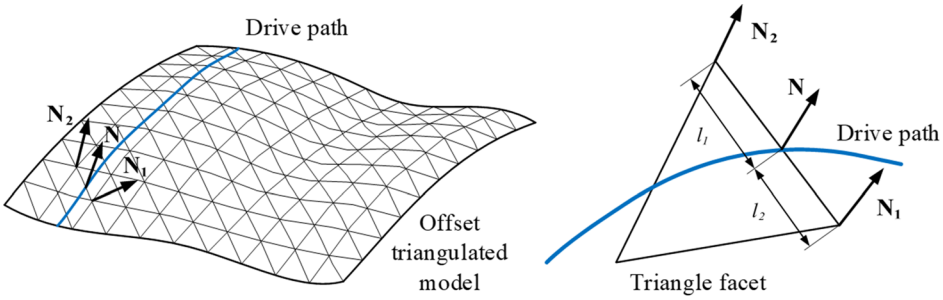


Fig. 4 Selection of facets in the AABB shadow

the shortest projection distance among all candidates is identified as the final cutter contact point. By repeating this procedure for each location on the drive path, a complete gouge-free tool path, which closely follows the drive path, is generated.

The definition of the projection direction is a prerequisite for projecting a drive point onto the part surface. In this work, a robust and geometry-conforming strategy is employed. The drive path is planned on a facet model that is offset from the original part surface. As shown in Fig. 5, the projection vector $\boldsymbol{P}_V$ for a drive point is then set to be opposite to the surface normal vector $\boldsymbol{N}$ at that point on the offset model, i.e., $\boldsymbol{P}_V = -\boldsymbol{N}$. This choice effectively reverses the offsetting process, guiding the cutter from the drive path directly back towards the part surface, which maximizes the likelihood of a successful projection. The surface normal $\boldsymbol{N}$ at a point within a facet is calculated via linear interpolation of the normals at the

Fig. 5 Schematic of the projection direction calculation



(a) Computation of the normal vectors along the drive CC path on the offset triangulated model

(b) Enlarged illustration of the computation of normal vectors on a triangle facet

facet's vertices ($\boldsymbol{N} = \boldsymbol{N}_1 l_1 + \boldsymbol{N}_2 l_2$). This defines a consistent, pre-computed vector field for projection, which is a key input to the algorithms described in the following sections.

- **Hierarchical Search for Contact Points**

The cutter may contact a triangular facet in three fundamental ways (Fig. 6): (1) on the facet interior, (2) on an edge, or (3) at a vertex.

To efficiently locate a valid contact point, a hierarchical search strategy is employed (Fig. 7). The algorithm first attempts a cutter-facet projection (a). If this fails, it proceeds to cutter-edge projection (b), which poses the core computational challenge: solving the torus-edge tangency problem. For this problem, the framework adopts a hybrid analytical-numerical solver. Analytical solutions are used for two special cases; for the general case, three numerical methods are provided. Should both previous steps fail, the algorithm finally resorts to cutter-vertex projection (c). This case is treated as a cone-line or torus-line intersection problem, which is computationally straightforward.

(a) **Cutter-facet projection**

The coordinate systems for the workpiece (WCS) and the cutter (CCS) are postulated in Fig. 8. The CCS is configured with the Z-axis parallel to the tool axis $\boldsymbol{T}_A(0,0,1)$, while the X-Z plane is contingent on the orientation of both $\boldsymbol{T}_A$ and the facet normal $\boldsymbol{N}_F$. The projection vector $\boldsymbol{P}_V$ is ascertained from the normals of sampled surface points. Consequently, the projection distance $\mathrm{d_P}$ is characterized as the displacement required for the cutter to reach the contact point along $\boldsymbol{P}_V$.

The detailed algorithm is as follows:

Synopsis:

($\boldsymbol{CC}$, $\boldsymbol{CL}$, d_P**)=CutterFacetProjection(** $\mathbf{T}_A$, $\boldsymbol{CL}_0$, $\boldsymbol{P}_V$, T_F, T_s**)**.

Input:

(1) Tool-axis $\mathbf{T}_A(0,0,1)$ and initial cutter location point $\boldsymbol{CL}_0$.

(2) Projection vector $\boldsymbol{P}_V$.

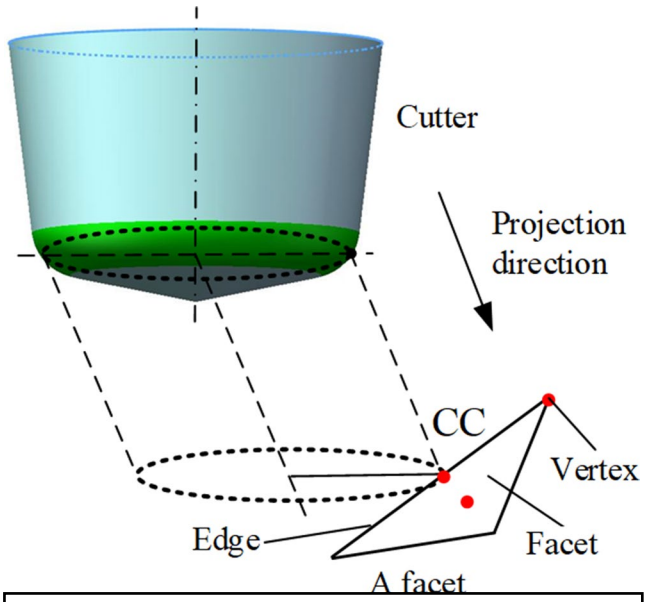


Fig. 6 Three cases of the contact point (in red)

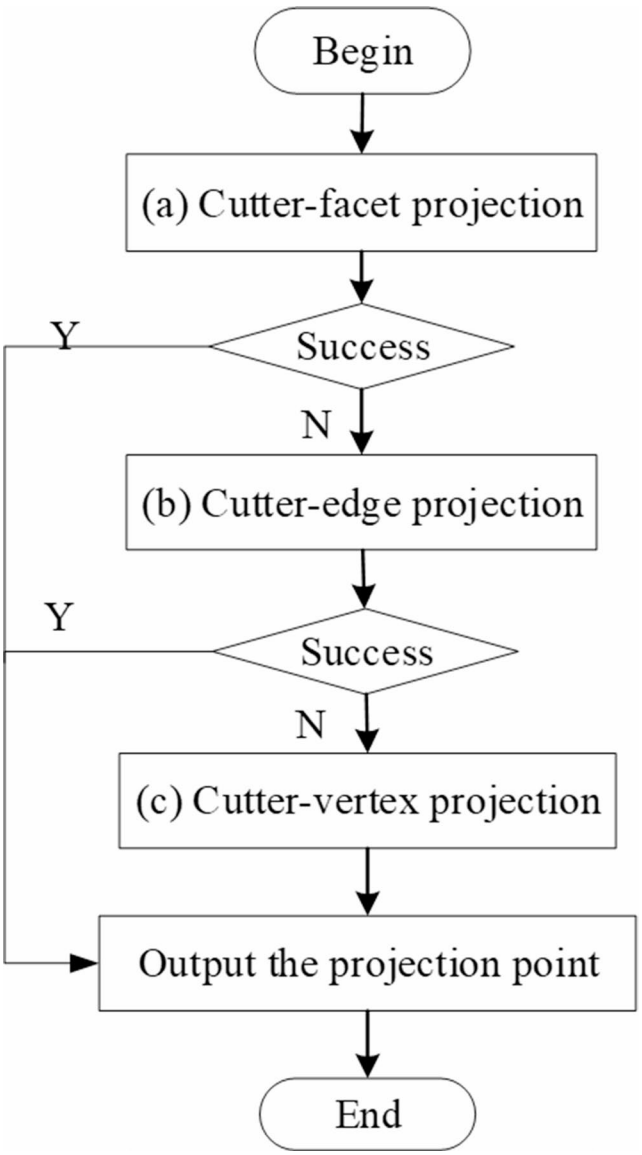


Fig. 7 Search strategy of the projection method

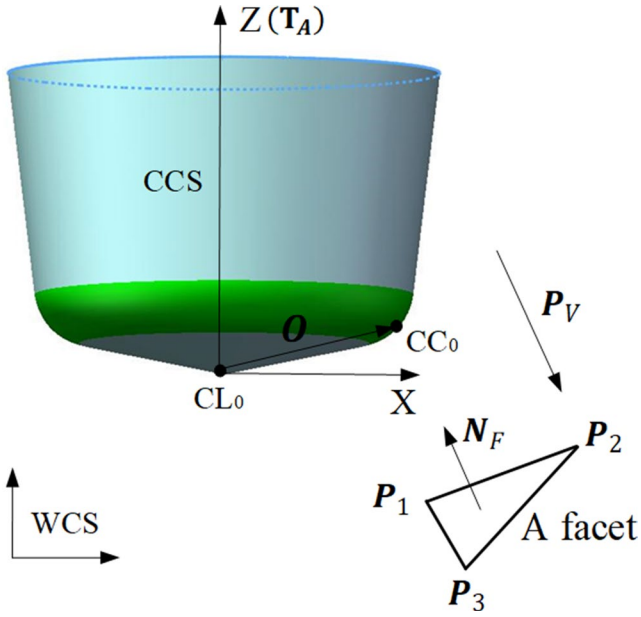


Fig. 8 Cutter-facet projection

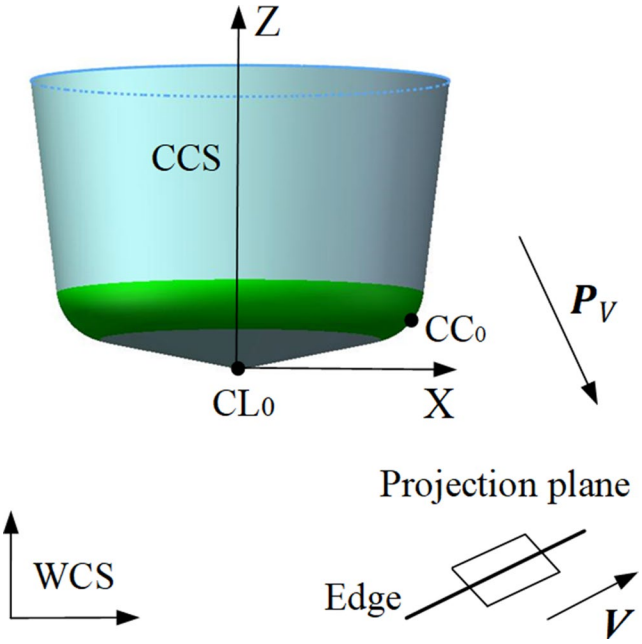


Fig. 9 Cutter-edge projection

(3) The facet model $\mathrm{T}_F\{P_1, P_2, P_3\}$.
(4) Tool shape T_s.

Output:

(1) A tangent cutter contact point $\boldsymbol{CC}$.
(2) A cutter location point $\boldsymbol{CL}$.
(3) The projection distance d_P.

Procedure:

Step1: Transform the normal vector $\mathbf{N}_F$ and $\boldsymbol{P}_V$ from WCS to CCS;

Step2: Compute the cutter offset vector $\boldsymbol{O}$ in CCS from the vector $\mathbf{N}_F$ and the $\boldsymbol{CC}_0$ point on cutter;

Step3: Intersect the line determined by the $\boldsymbol{CC}_0$ point and projection $\boldsymbol{P}_V$ with facet in CCS, the $\boldsymbol{CC}$ point in CCS and the projection distance d_P can be obtained;

Step4: Calculate the $\boldsymbol{CL}$ point in CCS by using the formula $\boldsymbol{CL} = \boldsymbol{CL}_0 + d_P\boldsymbol{P}_V$;

Step5: If $\boldsymbol{CC}$ is outside the facet, turn to the cutter-edge projection, else turn to step5;

Step6: Transform $\boldsymbol{CC}$ point and $\boldsymbol{CL}$ point from CCS to WCS, output $\boldsymbol{CC}$, $\boldsymbol{CL}$, and d_P.

(b) **Cutter-edge projection**

The input and output of cutter-edge projection is the same as that of cutter-facet projection. Some parameters are defined in Fig. 9. The procedures is described below.

Step1: Build the projection plane determined by the projection vector $\boldsymbol{P}_V$ and the edge vector $\boldsymbol{V}$;

Step2: Intersect the projection plane with the cutter to compute the $\boldsymbol{CC}$ point in CCS and the projection distance d_P;

Step3: Calculate the $\boldsymbol{CL}$ point in CCS using the formula $\boldsymbol{CL} = \boldsymbol{CL}_0 + d_P\boldsymbol{P}_V$;

Step4: If $\boldsymbol{CC}$ is not on the edge, turn to the cutter-vertex projection, else turn to step5;

Step5: Transform $\boldsymbol{CC}$ point and $\boldsymbol{CL}$ point from CCS to WCS, output $\boldsymbol{CC}$, $\boldsymbol{CL}$, and d_P.

- **Problem Abstraction: Torus-Edge Tangency**

Based on the preceding analysis of cutter-facet and cutter-edge projection, the problem of projecting a cutter onto a facet and its boundaries can be fundamentally abstracted as a torus-edge tangency problem. The mathematical representation of a torus is quartic, whereas that of a cone is quadratic, which makes projection onto the toroidal region of a cutter computationally more complex. While the contact point for torus-facet projection can be determined directly by aligning the torus normal vector with the facet normal, the torus-edge projection case presents a significant computational bottleneck. Consequently, this work focuses on resolving the challenge of torus-edge projection along an arbitrary direction.

- **Management of STL Discretization Error**

The accuracy of the computed tool path is inherently linked to the resolution of the facet (STL) model, due to the surface

approximation error between the triangulated model and the original CAD geometry. In this work, this error is actively controlled. The facet model is prepared such that the maximum deviation of any triangle from the CAD surface is strictly less than the user-defined machining tolerance. This is achieved through recursive subdivision, a standard industrial practice, which ensures the surface approximation error remains a secondary, bounded component within the total machining error budget.

2.2 Formulation of arbitrary direction torus-edge projection

This section formulates the mathematical model for the core problem of projecting a toroidal cutter onto an edge along an arbitrary direction, which results in a torus-edge tangency condition.

As illustrated in Fig. 10(a), a point on the torus surface can be described using the circular segment center (d, h) and the rotating angles (α, θ). The projection geometry, shown in Fig. 10(b), is defined within a plane determined by the edge vector $\boldsymbol{V}(V_x, V_y, V_z)$ and the projection vector $\boldsymbol{P}_V(U_x, U_y, U_z)$, passing through a vertex $\boldsymbol{P}_1(P_x, P_y, P_z)$ of the edge. The edge vector $\boldsymbol{V}$ is defined by the two vertices P1 and P2.

A point $\boldsymbol{P}_T$ on the torus surface is parametrically expressed as:

$$\boldsymbol{P}_T = \begin{bmatrix} P_{TX}(\alpha, \theta) \\ P_{TY}(\alpha, \theta) \\ P_{TZ}(\alpha, \theta) \end{bmatrix} = \begin{bmatrix} (d + R\cos(\alpha))\cos(\theta) \\ (d + R\cos(\alpha))\sin(\theta) \\ h - R\sin(\alpha) \end{bmatrix} \quad (1)$$

Where, $0 \leq \alpha \leq \frac{1}{2}\pi, 0 \leq \theta < 2\pi$.

The corresponding normal vector $\boldsymbol{N}_T$ at $\boldsymbol{P}_T$ is given by:

$$\boldsymbol{N}_T = \frac{\frac{\partial \boldsymbol{P}_T}{\partial \alpha} \times \frac{\partial \boldsymbol{P}_T}{\partial \theta}}{\left|\frac{\partial \boldsymbol{P}_T}{\partial \alpha} \times \frac{\partial \boldsymbol{P}_T}{\partial \theta}\right|} = \begin{bmatrix} \cos(\alpha)\cos(\theta) \\ \cos(\alpha)\sin(\theta) \\ -\sin(\alpha) \end{bmatrix} \quad (2)$$

The normal vector $\boldsymbol{N}_P$ of the projection plane is the cross product of the projection vector and the edge vector:

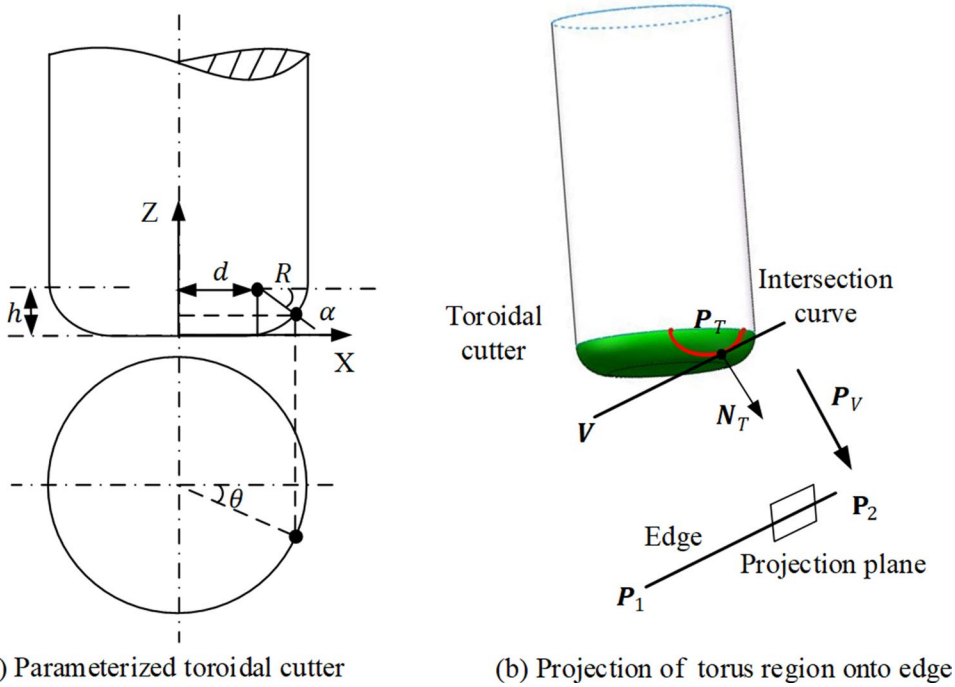
$$\boldsymbol{N}_P = \boldsymbol{P}_V \times \boldsymbol{V} = \begin{bmatrix} -U_yV_z + U_zV_y \\ U_xV_z - U_zV_x \\ -U_xV_y + U_yV_x \end{bmatrix} \quad (3)$$

As shown in Fig. 10(a), tangency between the edge and the torus is achieved when two constraints are simultaneously satisfied: (1) the torus normal at the tangent point is perpendicular to the edge vector, and (2) the tangent point lies on the projection plane. These constraints yield Eq. (4) and Eq. (5), respectively:

$$\boldsymbol{N}_T \bullet \boldsymbol{V} = 0 \quad (4)$$

$$(\boldsymbol{P}_T - \boldsymbol{P}_1) \bullet \boldsymbol{N}_P = 0 \quad (5)$$

This system of two nonlinear simultaneous equations in α and θ defines the tangency condition. By combining and rearranging these equations, the problem can be reduced to a single-variable equation in θ:



(a) Parameterized toroidal cutter (b) Projection of torus region onto edge

Fig. 10 Projection of a toroidal cutter onto edge

$$\begin{aligned}&(S_1^2-1)(d\,(V_zS_2-U_zS_1)+Q)^2\\&+R^2\left((U_xV_x+U_yV_y+U_zV_z)\,S_1-S_2\right)^2=0\end{aligned}\tag{6}$$

Where, $S_1=\cos(\backslash\theta)\,V_y-sin\,(\backslash\theta)\,V_x, S_2=cos\,(\backslash\theta)\,U_y-sin\,(\backslash\theta)\,U_x,$ and $Q=hU_xV_y-P_xU_yV_x+P_yU_xV_y-P_yU_xV_y+P_zU_yV_1.$

Actually, by substituting $\cos(\theta)=\frac{1-tan\left(\frac{\theta}{2}\right)^2}{1+tan\left(\frac{\theta}{2}\right)^2}$, $\sin(\theta)=\frac{2tan\left(\frac{\theta}{2}\right)}{1+tan\left(\frac{\theta}{2}\right)^2}$ and $x=tan\left(\frac{\theta}{2}\right)$ in the Eq. (6), the Eq. (7) is obtained. Coefficients $a_i(i=0,1,2\cdots 8)$ are related to the known parameters d, h, α, θ, $\boldsymbol{V}$, $\boldsymbol{P}_V$, $\boldsymbol{P}_1$. An eighth-order equation over the single unknown variable $x=tan\left(\frac{\theta}{2}\right)$ can be written as:

$$\begin{aligned}&f(x)=a_8x^8+a_7x^7+a_6x^6+a_5x^5\\&+a_4x^4+a_3x^3+a_2x^2+a_1x+a_0=0\end{aligned}\tag{7}$$

Solving this general eighth-order equation for arbitrary projection direction requires numerical methods. Section 3 presents three custom numerical solvers for this purpose. However, under specific conditions, the problem simplifies significantly, allowing for analytical solutions, as detailed in the following subsection.

2.3 Two special cases of arbitrary direction torus-edge projection

To mitigate the computational complexity of the general case, we identify two special cases where the eighth-order equation (Eq. (7)) reduces to a quartic, enabling robust and fast analytical solutions.

2.3.1 Case 1: Tool-axis-aligned projection

When the projection vector $\boldsymbol{P}_V(U_x,U_y,U_z)$ aligns with the tool axis $\boldsymbol{T}_A(0,0,1)$, it follows that $S_2=0$. Substituting this into Eq. (6) simplifies it to the following quartic equation in S_1:

$$(S_1{}^2-1)(\mathrm{d}\,(-U_zS_1)+\mathrm{Q})^2+r^2((U_xV_x+U_yV_y+U_zV_z)\,S_1)^2=0\tag{8}$$

While similar forms of this quartic equation have been noted in prior work [29, 31, 32], they were typically solved numerically. Numerical root-finding methods (e.g., Newton-Raphson) can typically find only one root per interval, often necessitating costly domain subdivision to locate all real roots. In contrast, we implement a full analytical solution based on the Abel-Ruffini theorem and Ferrari's method, which directly computes all roots without iteration or subdivision. As demonstrated in Sect. 5.1, this analytical approach reduces the average computation time by 24.28% compared to the numerical method.

2.3.2 Case 2: Ball-end cutter projection

When a toroidal cutter specializes to a ball-end cutter ($d=0$ and $h=\mathrm{R}$), Eq. (6) simplifies to the ball-edge projection equation. A similar half-angle substitution then yields a quartic equation(Coefficients $c_i(i=0,1,2\cdots 4)$ are related to the known parameters d, h, α, θ, $\boldsymbol{V}$, $\boldsymbol{P}_V$, $\boldsymbol{P}_1$):

$$\begin{aligned}&(-\cos(\theta)^2v_x^2+\cos(\theta)^2v_y^2-2\cos(\theta)\sin(\theta)\,v_xv_y-v_y^2-v_z^2)Q^2+\\&R^2(\cos(\theta)\,u_xv_xv_y+\sin(\theta)\,u_xv_y^2+\sin(\theta)\,u_xv_z^2-\cos(\theta)\,u_yv_x^2\\&-sin(\theta)\,u_yv_xv_y-cos(\theta)\,u_yv_z^2-sin(\theta)\,v_zu_zv_x+cos(\theta)\,v_zu_zv_y)^2=0\end{aligned}\tag{9}$$

$$c_4x^4+c_3x^3+c_2x^2+c_1x+c_0=0\tag{10}$$

The analytical solution for the tool-axis-aligned case (Case 1) is particularly impactful and is employed to construct the gouge avoidance algorithm detailed in Sect. 4.1. Its practical application and validation in five-axis blade machining are presented in Sect. 5.1.

3 Numerical methods of arbitrary torus-edge direction projection

To enable robust and efficient five-axis tool path generation, this section presents three distinct numerical methods developed for solving the general case of arbitrary-direction torus-edge projection, which does not yield an analytical solution. These methods – algebra-based Bézier clipping, geometry-based edge-search, and geometry-based torus-search – represent parallel approaches to the same core problem. A comprehensive performance evaluation (Sect. 5) demonstrates that the geometry-based torus-search method offers superior computational efficiency. Consequently, it is adopted as the standard numerical solver within our framework for this general projection case. The principles of all three methods are detailed below for completeness.

Algebra-based Bézier Clipping Method: This method finds all real roots of the derived eighth-order torus-edge equation (Eq. (7)) using a refined Bézier clipping technique. The root corresponding to the shortest projection distance is selected to compute the contact point.

Geometry-based Edge-Search Method: This approach searches for the contact point directly along the edge by leveraging geometric constraints and the gradient of the projection distance, thereby avoiding the need to solve the high-order equation.

Geometry-based Torus-Search Method: This method searches for the contact point along a specific curve on the torus surface, defined by geometric constraints, using the gradient of the distance between the edge and the projection line. Its design balances robustness and speed, making

it the preferred choice for integration into the computational pipeline.

3.1 Algebra-based Bézier clipping method

This method computes all real roots of the eighth-order torus-edge tangency equation (Eq. (7)) within a defined initial interval using a recursive Bézier clipping algorithm, which involves constructing a Bézier curve and its convex hull from the polynomial (Fig. 11).

(a) **Determine the initial root interval**

The initial root interval is determined using an implicit geometrical constraint. As per the hierarchical projection strategy (Fig. 7), the torus-edge projection is only attempted after cutter-facet projection fails. The constraint dictates that the reverse normal vector $\boldsymbol{N}_T$ of the contact point must lie between the normal vectors $\boldsymbol{N}_{F1}$ and $\boldsymbol{N}_{F2}$ of the two adjacent facets (Fig. 12). Given $\boldsymbol{N}_{F1}$ and $\boldsymbol{N}_{F2}$, the corresponding angles (α_1, θ_1) and (α_2, θ_2) can be determined from Eq. (2). Substituting θ_1 and θ_2 into the variable $x = \tan\left(\frac{\theta}{2}\right)$ of Eq. (7) yields the initial root interval $x \in \left[\tan\left(\frac{\theta_1}{2}\right), \tan\left(\frac{\theta_2}{2}\right)\right]$.

$$\boldsymbol{N}_{F1} = -\begin{bmatrix} \cos(\alpha_1)\cos(\theta_1) \\ \cos(\alpha_1)\sin(\theta_1) \\ -\sin(\alpha_1) \end{bmatrix}, \boldsymbol{N}_{F2} = -\begin{bmatrix} \cos(\alpha_2)\cos(\theta_2) \\ \cos(\alpha_2)\sin(\theta_2) \\ -\sin(\alpha_2) \end{bmatrix} \quad (11)$$

(b) **Construct Bézier curve and compute minimum convex hull**

The polynomial from Eq. (7) is converted to Bernstein basis, and its Bézier coefficients are computed. The Bézier curve is then constructed, and its minimum convex hull [39, 47, 48] is determined.

(c) **Determine the set of sub-intervals containing a unique root**

Existing methods that are used to divide the root intervals may produce false positive answers and excessive subdivisions. According to the convex-hull property, the graph of polynomial lies within the convex hull. The classical Bézier clipping methods [38, 39] get the divided intervals by intersecting convex-hull with the t-axis. However, these methods may produce false positive answers if the graph of the polynomial gets very close to the t-axis. B. Mourrain et al. [40] presents an improved method by taking into account control polygon and convex-hull simultaneously. After getting a series of intervals, the subdivision strategy which utilizes **Lemma** based on variation diminishing of Bézier is used to check the existence of a solution in every

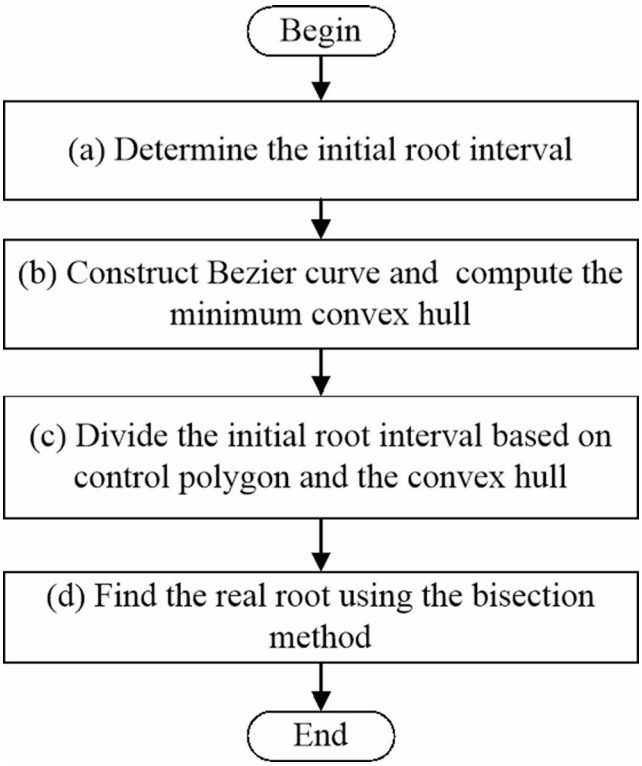


Fig. 11 The flow chart of the algebra-based Bézier clipping method

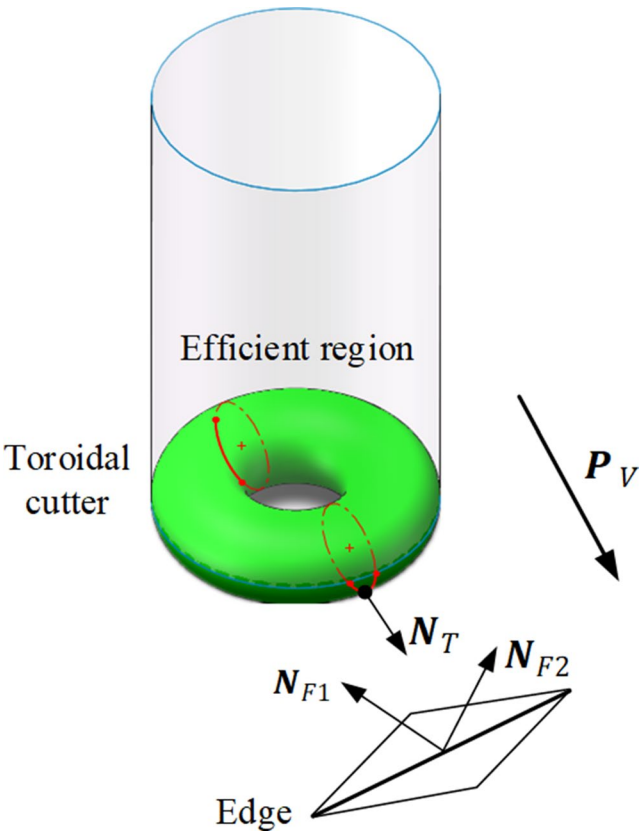


Fig. 12 Determination of the initial root interval

search interval. But this recursive subdivision approach cannot avoid extensive subdivisions.

To avoid extensive subdivisions, we improve the subdivision approach for the zero-or-two-root sub-interval with the strategy by choosing the prior sub-interval more likely containing roots to divide first. According to **Lemma**, the number of real roots of the polynomial in $[\alpha, \beta]$ is bounded by the number V of sign changes of Bernstein coefficients. If $V = 0$, there is no root in $[\alpha, \beta]$ and if $V = 1$, there is one root in $[\alpha, \beta]$. Otherwise, the interval may contain no root or two roots and needs further analysis. Since control points $D_i(\gamma_i, b_i)$ can pull the spline curve close to the control polygon, we spilt the divided interval I into two sub-intervals I_1 and I_2 based on the minimum or maximum control points. Then we respectively compute the Bernstein coefficients of the polynomials in the sub-interval I_1 (I_2) and use the **Lemma** to judge the existence of roots. We recursively subdivide zero-or-two-root intervals until the number of roots in each sub-interval can be determined for sure. The pseudocode of the process to determine the set of sub-interval containing a unique root is as follows.

FUNCTION: **DivideZeroOrTwoRootIntervals** (P, I, F)
INPUT: P // the polynomial
I // the zero-or-two-root interval
OUTPUT: F // the set of sub-interval containing a unique root
BEGIN
1. IF the length of $I > tolerance$ THEN
2. $\{b_i\}$ ←Call **ComputeBernsteinCoeff**(P, I)
3. Find the minimum or maximum of $\{b_i\}$ and corresponding parameter γ
4. Split I by γ into two subsection I_1 and I_2
5. $\{b_i\}^1$ ←Call **ComputeBernsteinCoeff**(P, I_1)
6. $\{b_i\}^2$ ←Call **ComputeBernsteincoeff**(P, I_2)
7. IF the sign of $\{b_i\}^1$ does not change, THEN
8. Call **DivideZeroOrTwoRootIntervals** (P, I_2, F)
9. IF the sign of $\{b_i\}^2$ does not change, THEN
10. Call **DivideZeroOrTwoRootIntervals** (P, I_1, F)
11. IF the sign of $\{b_i\}^1$changes once, THEN
12. Output F that contains I_1 and I_2
13. IF the sign of $\{b_i\}^1$ and $\{b_i\}^2$ changes twice simultaneously
14. Call **ChoosePriorRootSub-intervalToDivideFirst** (P, I_1, I_2, F)
15. ELSE
16. Output F that equals $\emptyset$
END

When the two sub-intervals simultaneously need to be subdivided, we choose the prior sub-intervals that containing roots with high possibility to divide first based on minimum or maximum control points. The pseudocode of the detailed procedures is as follows.

FUNCTION: **ChoosePriorRootIntervalsToDivideFirst**(P, I, I_1, I_2, F)
INPUT: P // the polynomial
I // the Interval containing I_1 and I_2
I_1, I_2// the Interval maybe containing root
OUTPUT: F // the set of sub-interval containing a unique root
BEGIN
1. $\{b_i\}^1$ ←Call **ComputeBernsteinCoeff**(P, I_1)
2. $\{b_i\}^2$ ←Call **ComputeBernsteinCoeff**(P, I_2)
3. IF $([b_0]^1 > 0$ && $[b_0]^2 > 0)$
4 Compute the minimum $[b_l] \in \{b_i\}^j \ (j = 1,2)$
5. Call **DivideZeroOrTwoRootIntervals** (P, I_j, F)
6. IF F is empty
7. Call **DivideZeroOrTwoRootIntervals** $(P, I - I_j, F)$
8. ELSE
9. Compute the maximum $[b_k] \in \{b_i\}^j \ (j = 1,2)$
10. Call **DivideZeroOrTwoRootIntervals** (P, I_j, F)
11. IF F is empty
12. Call **DivideZeroOrTwoRootIntervals** $(P, I - I_j, F)$
END

(d) **Find the real root using the bisection method for each sub-interval**

Once all sub-intervals containing a unique root are identified, the bisection method is applied within each to find the root to a desired precision. The root yielding the shortest projection distance is selected to calculate the final contact point.

While this method accurately finds all roots, the recursive iterations can be computationally expensive. More efficient geometric methods are presented in Sect. 3.2 and 3.3.

3.2 Geometry-based edge-search method

This method searches for the contact point directly along the edge using geometric constraints and gradient information, circumventing the eighth-order equation. As shown in Fig. 13, $\boldsymbol{P}_i$ and $\boldsymbol{Q}_i$ respectively represent the ith point on edge and torus. The distance $l_i = |\boldsymbol{P}_i\boldsymbol{Q}_i|$ is the projection distance. Define the variable λ_i as the edge parameter. The variable $l_i' = \frac{dl_i}{d\lambda_i}$ is the derivative of the projection distance l_i over the edge parameter λ_i. The contact point can be computed when the objective function $l_i' = 0$ is

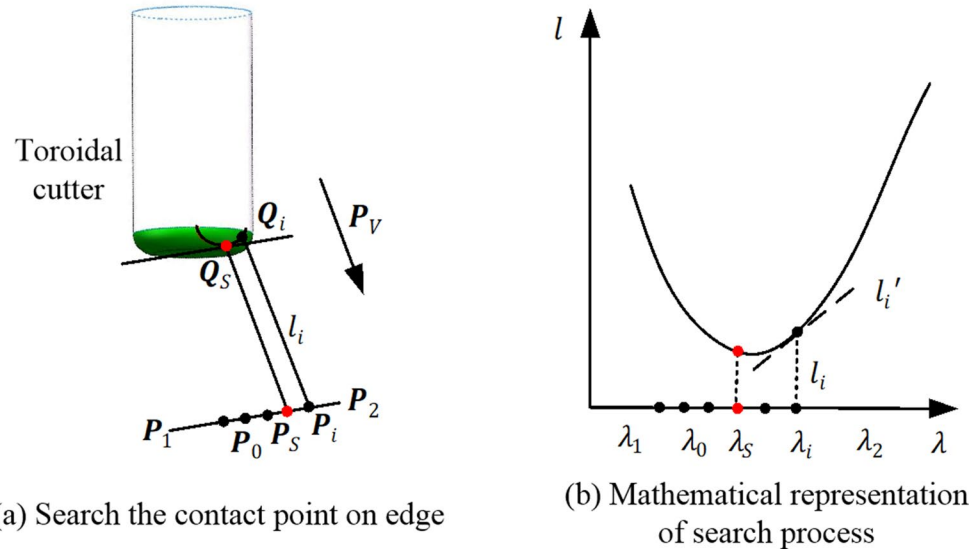


Fig. 13 Edge-search method for finding the contact point

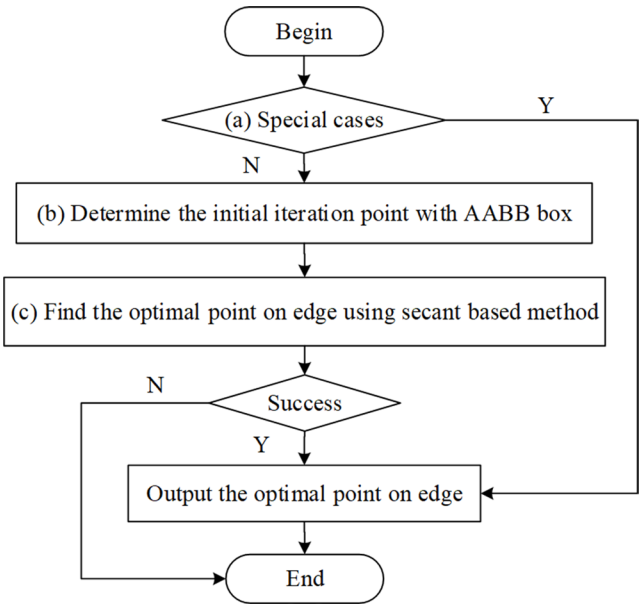


Fig. 14 The Flow chart of edge-search method

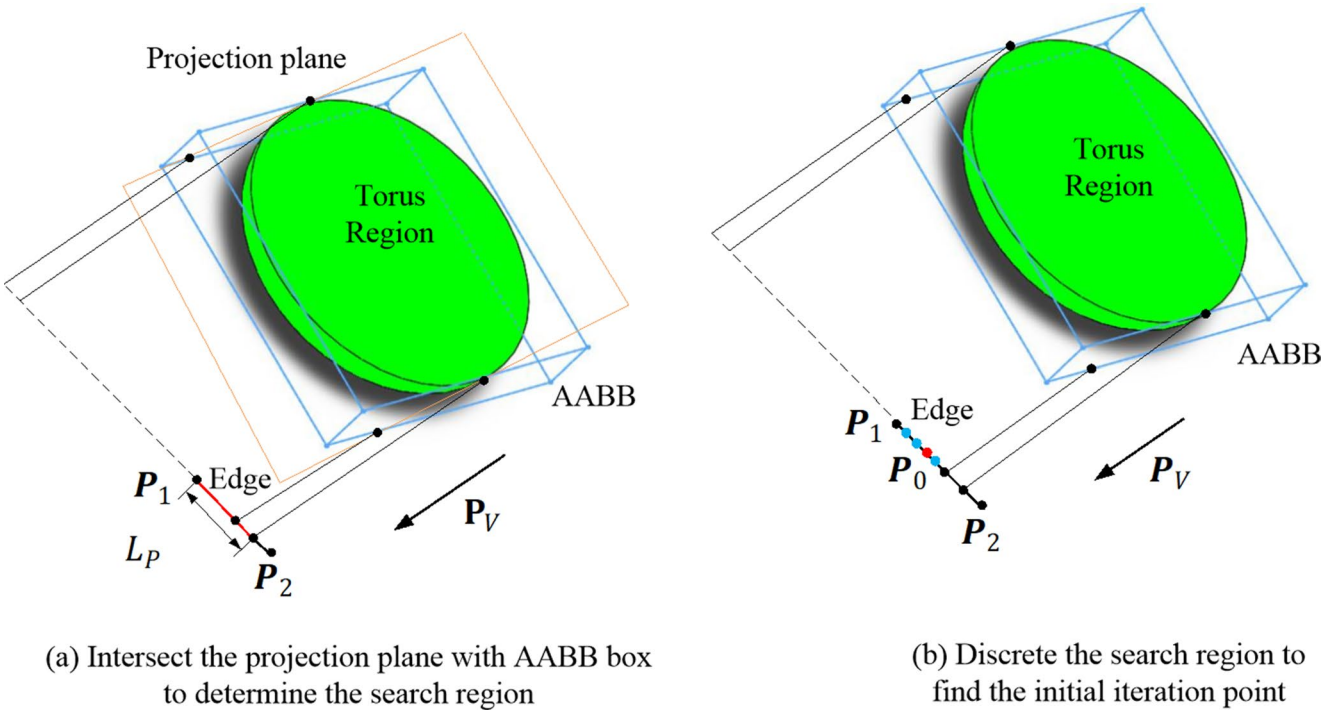
satisfied. To find the optimal point $\boldsymbol{P_s}$ on edge, the search region need to be determined first. To simplify the computation, AABB containing the torus region of the cutter is built. The initial iteration point $\boldsymbol{P}_0$ near to the optimal point $\boldsymbol{P_s}$ is obtained by subdividing the search region and projecting the divided points onto the torus region. The simplified secant search algorithm is used to search the contact point. Figure 14 shows three main steps of this geometry-based edge-search method.

(a) **Special cases**

There are two special cases where the optimal point can be directly determined without numerical iterations. When the edge vector is parallel to the projection vector, we select the vertex near to the torus as the optimal point. When the vertex $\boldsymbol{P}_1(\boldsymbol{P}_2)$ has projection and the derivative $l_1' > 0\,(l_2' < 0)$ is satisfied, the point $\boldsymbol{P}_1(\boldsymbol{P}_2)$ is the optimal point.

(b) **Determine the initial iteration point with AABB**

An Axis-Aligned Bounding Box (AABB) enclosing the cutter's torus region is constructed (Fig. 15). This AABB is intersected with the projection plane, yielding six intersection points. These points are projected onto the edge to define a search region. This region is discretized, and each point is projected onto the torus. The point with the shortest valid projection distance is chosen as the initial iteration point $\boldsymbol{P}_0$.



(a) Intersect the projection plane with AABB box to determine the search region

(b) Discrete the search region to find the initial iteration point

Fig. 15 Determination of the initial point with AABB

(c) **Find the optimal point on edge using the secant search algorithm**

The secant search algorithm is simplified and its specific implementation procedures are presented. Generally, the secant search algorithm needs two initial iteration points to search for the optimal point. Since the efficient torus region of a cutter is the two discontinuous outer boundaries, finding one initial iteration point near the optimal point can avoid the fluctuation in the searching process. As shown in Fig. 16, the objective function is $l_i' = 0$ without requiring the calculation of the second derivative l_i''. The first derivative l_i' can be accurately calculated if the point $\boldsymbol{P}_i$ on edge is known. The iteration point $\boldsymbol{P}_k$ can be derived with the derivative l_i' and the search step α_i. It is worth mentioning that the first search step is $\alpha_0 = |l_0'|$ and the ith ($i \neq 0$) search step is $\alpha_i = \frac{(P_i - P_{i-1})}{l_i' - l_{i-1}'}$. The process of calculating the point $\boldsymbol{P}_i$ on edge and the corresponding derivative l_i' is repeated until the termination criterion $l_i' = 0$ is satisfied.

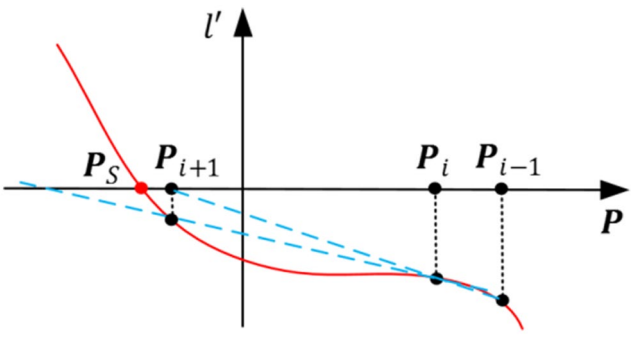


Fig. 16 The secant search algorithm

The secant search algorithm can be expressed as Eq. (12):

$$\boldsymbol{P}_{i+1} = \boldsymbol{P}_i - \alpha_i g_i \tag{12}$$

Where, $\alpha_i = \frac{(P_i - P_{i-1})}{l_i' - l_{i-1}'}$ is the search step, and $g_i = l_i'$ means the search gradient.

The procedure for finding the optimal point on edge using the secant search algorithm will be provided below:

FUNCTION: **FindOptimalPointOnEdge**($\boldsymbol{P}_0, \boldsymbol{P}_V$, T, $\boldsymbol{V}, S_E, \boldsymbol{P}_S, \boldsymbol{Q}_S$)
INPUT:$\boldsymbol{P}_0$ // the initial iteration point on edge
$\boldsymbol{P}_V$ // the arbitrary projection vector
T // the toroidal tool
$\boldsymbol{V}$ // the edge vector
S_E // the edge segment composed by vertex point $\boldsymbol{P}_1$ and $\boldsymbol{P}_2$
OUTPUT:$\boldsymbol{P}_S$ // the optimal point on edge
$\boldsymbol{Q}_S$ // the optimal point on torus
BEGIN
1.$i = 0$, g_i ←Call **ComputeAccurateDerivative**($\boldsymbol{P}_0$, $\boldsymbol{P}_V$, T, $\boldsymbol{V}$)
2. **while** $g_i \geq \varepsilon$, not converge **do**
3. **IF** $i = 0$
4. $\alpha_i = |g_i|$, the next point $\boldsymbol{P}_{i+1} = \boldsymbol{P}_i - \alpha_i g_i$
5. g_{i+1} ←Call **ComputeAccurateDerivative**($\boldsymbol{P}_{i+1}, \boldsymbol{P}_V$, T, $\boldsymbol{V}$)
6. **ELSE**
7. **IF** $g_i = g_{i+1} \neq 0$
8. $\alpha_{i+1} = |g_{i+1}|$
9. **ELSE**
10. $\alpha_{i+1} = \frac{(P_{i+1} - P_i)}{g_{i+1} - g_i}$
11. Compute the next search point $P_{i+2} = P_{i+1} - \alpha_{i+1} g_{i+1}$
12. g_{i+1} ←Call **ComputeAccurateDerivative**($\boldsymbol{P}_{i+1}, \boldsymbol{P}_V$, T, $\boldsymbol{V}$)
13. **END DO**
14. **IF** $\boldsymbol{P}_{i+1}$ is not on the edge segment S_E, return false
15. **ELSE**
16. $\boldsymbol{Q}_{i+1}$ ←Call **ProjectPointToTorus**($\boldsymbol{P}_{i+1}, \boldsymbol{P}_V$, T)
17. $\boldsymbol{P}_S = \boldsymbol{P}_{i+1}$, $\boldsymbol{Q}_S = \boldsymbol{Q}_{i+1}$
END

A small computational error can result in a large deviation between the desired and actual contact point. The accurate derivative

l_i' is calculated to guarantee the precision of the obtained contact point. The details are given below.

In Fig. 17, the point $\boldsymbol{P}_i$ can be obtained by moving the vertex $\boldsymbol{P}_1(P_x, P_y, P_z)$ along the line vector $\boldsymbol{V}(V_x, V_y, V_z)$ with the edge parameter λ_i. Similarly, the point $\boldsymbol{Q}_i$ on the torus can be obtained by moving the point $\boldsymbol{P}_i$ along projection vector $\boldsymbol{P_V}(\boldsymbol{U_x}, \boldsymbol{U_y}, \boldsymbol{U_z})$ with the distance l_i. Thus, $\boldsymbol{Q}_i = P_1 + \lambda_i \bullet \boldsymbol{V} + l_i \bullet \boldsymbol{P}_V$ can be expressed as Eq. (13).

$$\boldsymbol{Q}_i = \begin{bmatrix} Q_{ix} \\ Q_{iy} \\ Q_{iz} \end{bmatrix} = \begin{bmatrix} P_x + V_x \lambda_i + U_x l_i \\ P_y + V_y \lambda_i + U_y l_i \\ P_z + V_Z \lambda_i + U_z l_i \end{bmatrix} \tag{13}$$

Obviously, the point $\boldsymbol{Q}_i$ satisfies the torus equation. Substituting and we can get

$$\left(\sqrt{{Q_{ix}}^2 + {Q_{iy}}^2} - d\right)^2 + (h - Q_{iz})^2 - R^2 = 0 \tag{14}$$

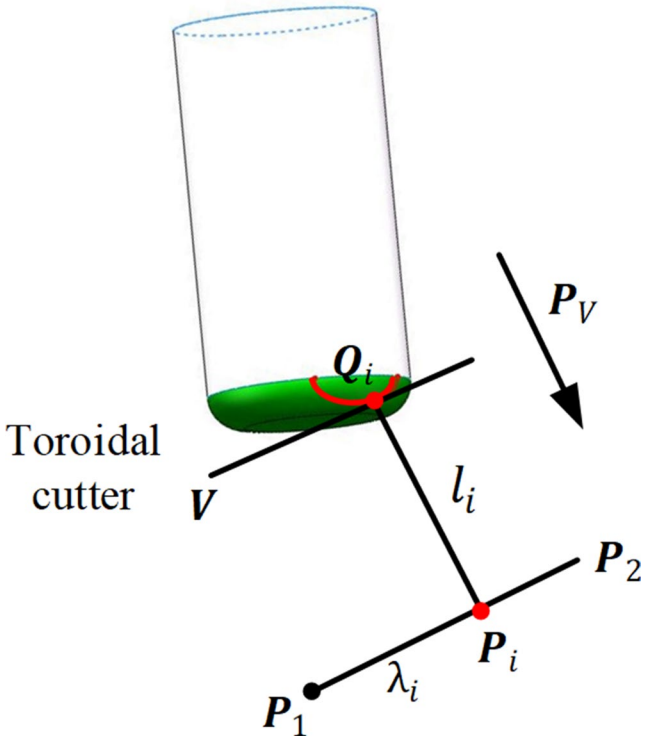


Fig. 17 Projection of the point on edge segment to torus region

Rearranging terms and squaring twice for Eq. (14), gives us Eq. (15), where the coefficient c_i $(i = 0,1,2,3,4)$ is a variable related to the edge parameter λ_i.

$$g(\lambda_i, l_i) = c_4 {l_i}^4 + c_3 {l_i}^3 + c_2 {l_i}^2 + c_1 l_i + c_0 = 0 \quad (15)$$

The accurate derivative $l_i' = \frac{dl_i}{d\lambda_i}$ can be derived as Eq. (16)

$$\begin{aligned}&(4c_4 l_i^3 + 3c_3 l_i^2 + 2c_2 l_i + c_1)\frac{dl_i}{d\lambda_i} = \\ &-(\frac{dc_4}{d\lambda_i} l_i^4 + \frac{dc_3}{d\lambda_i} l_i^3 + \frac{dc_2}{d\lambda_i} l_i^2 + \frac{dc_1}{d\lambda_i} l_i + \frac{dc_0}{d\lambda_i})\end{aligned} \quad (16)$$

In the searching process, the parameters $P_x, P_y, P_z, V_x, V_y, V_Z, U_x, U_y, U_z, c_i$ are determined first and then the accurate derivative l_i' is calculated. When the objective function $l_i' = 0$ is satisfied, an accurate contact point with the pre-defined termination tolerance, the corresponding cutter location point, and the projection distance are obtained.

A quartic equation needs to be repeatedly solved in determining the initial iteration point $\boldsymbol{P}_0$ and calculating the derivative l_i' while searching on edge for the optimal point. Thus, we determine to search the optimal point on torus region to avoid repeatedly solving the quartic equation and saving computation time.

3.3 Geometry-based torus-search method

Similar to the edge-search method, the torus-search method aims at searching the contact point along the curve of torus based on three implicit geometric constraints. The first constraint is that the reverse normal vector $\boldsymbol{N}_T$ of tangent point must be between the two normal vectors $\boldsymbol{N}_{F1}$ and $\boldsymbol{N}_{F2}$ of two adjacent facets. The second constraint is that

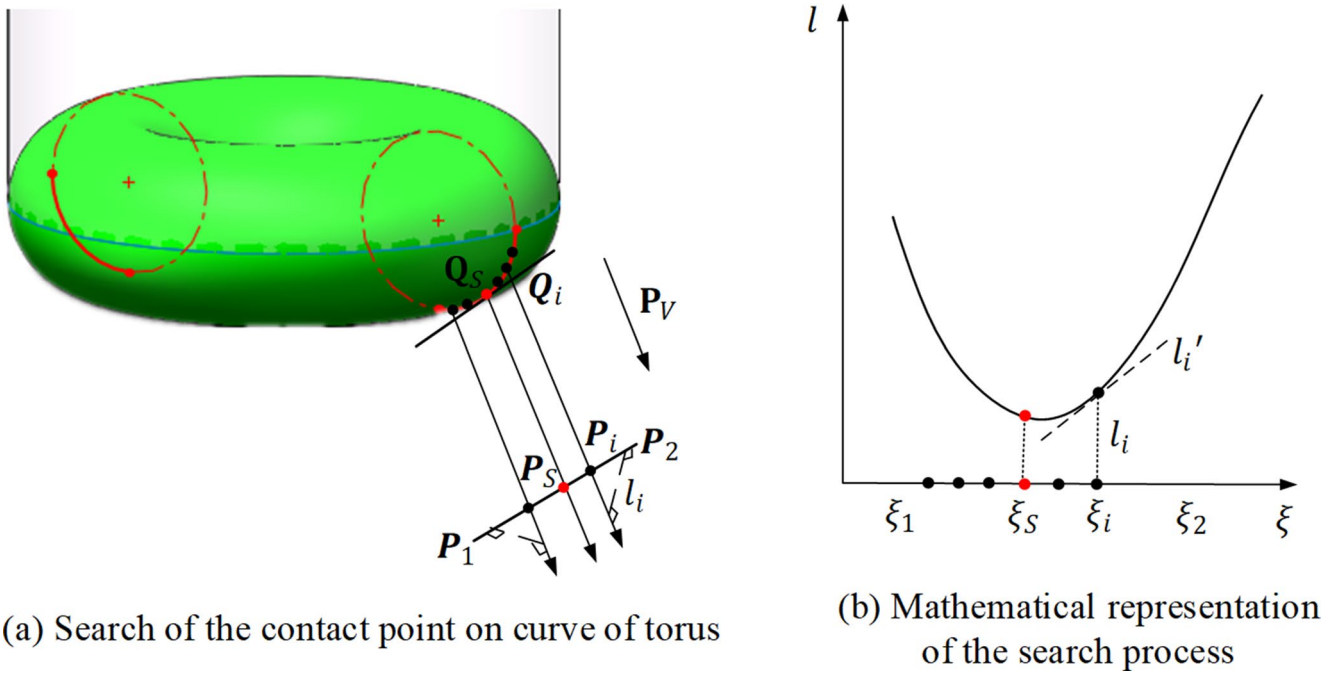


Fig. 18 Torus-search method for finding the contact point

the normal vector $\boldsymbol{N}_T$ of a tangent point on the torus is perpendicular to edge vector, i.e., $\boldsymbol{N}_T \bullet \boldsymbol{V} = 0$. We search on the curve $\boldsymbol{N}_T \bullet \boldsymbol{V} = 0$ to find the optimal angle corresponding to the tangent point. The third constraint is that the distance l between projection line passing through the contact point and the edge line is zero. As shown in Fig. 18, the curve parameter ξ_i is used to represent the point $\boldsymbol{Q}_i$ on torus. The edge line is determined by the vertex $\boldsymbol{P}_1$ and the edge vector $\boldsymbol{V}$, while the projection line is decided by the point $\boldsymbol{Q}_i$ on torus and the projection vector $\boldsymbol{P}_V$. Define the distance between the edge line and the projection line as l_i. The parameter $l_i' = \frac{dl_i}{d\xi_i}$ is the derivative of the distance l_i over the curve parameter ξ_i. When the function $l_i\prime = 0$ is satisfied, the optimal point $\boldsymbol{Q}_S$ on curve of torus and the point $\boldsymbol{P}_S$ on edge can be obtained. The specific procedures of the geometry-based edge-search method for finding the contact point are shown in Fig. 19.

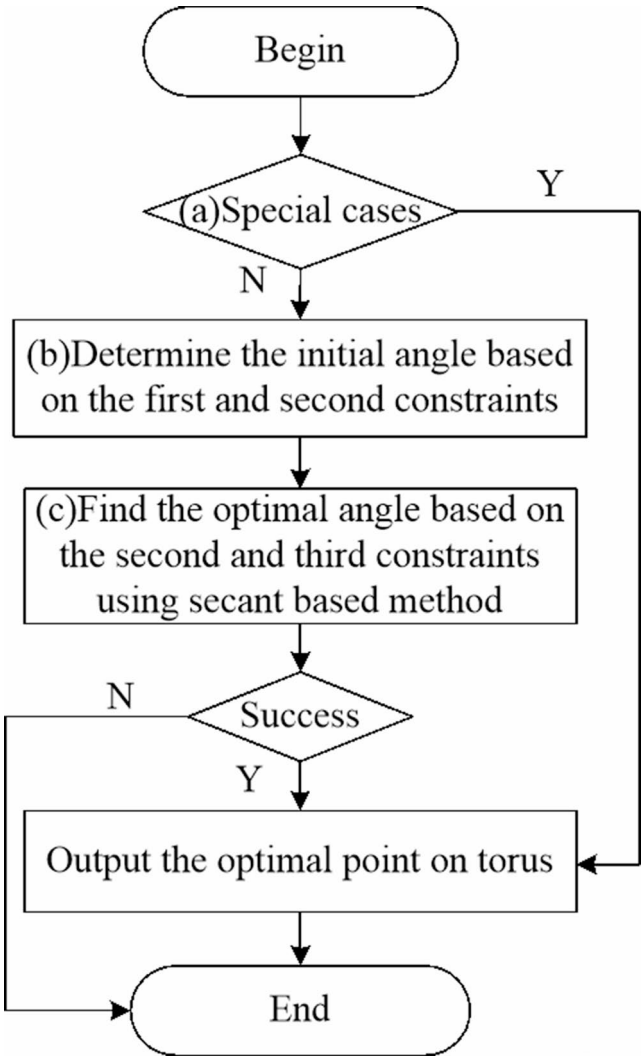


Fig. 19 The flow chart of torus-search method

(a) **Special cases**

The special cases is the same as that in edge-search method in Sect. 3.2.

(b) **Determine the initial search angle**

According to the first constraint, the angle α_1, θ_1, α_2 and θ_2 can be obtained. Based on the second constraint $f(\alpha, \theta) = V_x \cos(\alpha)\cos(\theta) + V_y\cos(\alpha)\sin(\theta) - V_z\sin(\alpha) = 0$, two search cases are given below.

Case1 $\boldsymbol{V_z = 0}$ As shown in Fig. 20, the angle θ unrelated to the angle α only have two different value, when $V_z = 0$. The curve equation can be simplified to $\tan(\theta) = -V_x/V_y$ when the equality $V_z = 0$ and inequality $\cos(\alpha) \neq 0$ satisfy, which indicates that the curve of $f(\alpha, \theta)$ will not be continuous for the angle θ. Taking this into consideration, the angle α is selected to search the contact point. By discretizing the calculated interval $[\alpha_1, \alpha_2]$, the initial iteration angle α_0 corresponding to the shortest distance between the edge line and projection line is determined.

Case2 $\boldsymbol{V_Z \neq 0}$ Depending on the definition of α and θ, it is obvious that there is a one-to-one mapping between a given angle θ and the point on the torus region, while there are two points on torus simultaneously corresponding to a given angle α. Additionally, the curve of $f(\alpha, \theta)$ is continuous for θ. Thus, the angle θ is selected as the search angle. By discretizing the calculated interval $[\theta_1, \theta_2]$, the initial iteration angle θ_0 corresponding to shortest distance between the edge line and projection line is determined.

(c) **Find the optimal angle**

The secant search algorithm in torus-search method is used to find the optimal angle, which is similar to that in edge-search method in Sect. 3.2. Attention is focused on the calculation of the accurate derivative $l_i\prime$.

The computation of the accurate derivative $l_i\prime$ is as below. The function $f(\alpha, \theta) = V_x \cos(\alpha)\cos(\theta) + +V_y\cos(\alpha)\sin(\theta) - V_z\sin(\alpha) = 0$ represents a curve on which the normal vector of each point is perpendicular to edge. We use the parameter ξ to describe $f(\alpha, \theta)$. Given that

$$\frac{df(\alpha,\theta)}{d\xi} = \frac{df(\alpha,\theta)}{d\alpha}\frac{d\alpha}{d\xi} + \frac{df(\alpha,\theta)}{d\theta}\frac{d\theta}{d\xi} = 0 \tag{17}$$

Since the derivative $\frac{d\alpha}{d\xi}$ and $\frac{d\theta}{d\xi}$ are independent, yields:

$$\frac{d\alpha}{d\xi} = \frac{df(\alpha,\theta)}{d\theta} = -V_x cos(\alpha)\, sin(\theta) + V_y cos(\alpha)\, cos(\theta) \tag{18}$$

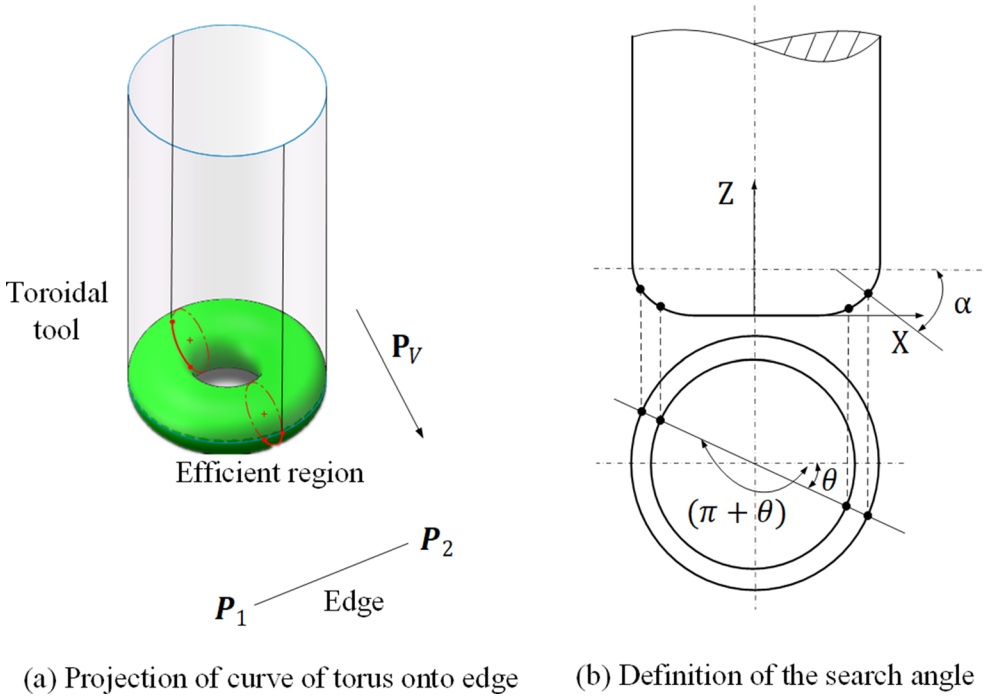


Fig. 20 Determination of the initial search angle is theta

(a) Projection of curve of torus onto edge (b) Definition of the search angle

$$\frac{d\theta}{d\xi} = -\frac{df(\alpha,\theta)}{d\alpha} = V_x sin(\alpha) cos(\theta) + V_y sin(\alpha) sin(\theta) + V_z cos(\alpha) \tag{19}$$

The derivative of any point on the curve $f(\alpha,\theta)$ related to the parameter ξ can be denoted as $\frac{dP_T(\alpha,\theta)}{d\xi}$, given that

$$\frac{dP_T(\alpha,\theta)}{d\xi} = \begin{bmatrix} \frac{dP_{Tx}(\alpha,\theta)}{d\alpha}\frac{d\alpha}{d\xi} + \frac{dP_{Tx}(\alpha,\theta)}{d\theta}\frac{d\theta}{d\xi} \\ \frac{dP_{Ty}(\alpha,\theta)}{d\alpha}\frac{d\alpha}{d\xi} + \frac{dP_{Ty}(\alpha,\theta)}{d\theta}\frac{d\theta}{d\xi} \\ \frac{dP_{Tz}(\alpha,\theta)}{d\alpha}\frac{d\alpha}{d\xi} + \frac{dP_{Tz}(\alpha,\theta)}{d\theta}\frac{d\theta}{d\xi} \end{bmatrix} \tag{20}$$

Where,

$$\frac{dP_T(a,\theta)}{da} = \begin{bmatrix} \frac{dP_{Tx}(a,\theta)}{da} \\ \frac{dP_{Ty}(a,\theta)}{da} \\ \frac{dP_{Tz}(a,\theta)}{da} \end{bmatrix} = \begin{bmatrix} -r\sin(a)\cos(\theta) \\ -r\sin(a)\sin(\theta) \\ -r\cos(a) \end{bmatrix},$$

$$\frac{dP_T(a,\theta)}{d\theta} = \begin{bmatrix} \frac{dP_{Tx}(a,\theta)}{d\theta} \\ \frac{dP_{Ty}(a,\theta)}{d\theta} \\ \frac{dP_{Tz}(a,\theta)}{d\theta} \end{bmatrix} = \begin{bmatrix} -(d\_r\cos(a)\sin(\theta)) \\ (d + r\cos(a)\cos(\theta)) \\ 0 \end{bmatrix}$$

Substituting the derivative $\frac{dP_T(\alpha,\theta)}{d\alpha}$ and $\frac{dP_T(\alpha,\theta)}{d\theta}$ into Eq. (20), given that

$$\frac{dP_T(\alpha,\theta)}{d(\alpha,\theta)} = \frac{dP_T(\alpha,\theta)}{d\xi} \tag{21}$$

According to the third geometric constraint shown in Fig. 18, we define the connect vector between any point on the torus and edge as $\boldsymbol{V}_c$, and the distance as $l = \frac{\mathbf{N}_P \bullet \boldsymbol{V}_c}{|\mathbf{N}_P|^2}$. Substituting the $\mathbf{N}_P$ and $\boldsymbol{V}_c$, the distance l is given in Eq. (22).

$$l = \frac{(P_{Tx}-P_x)(-U_yV_z+U_zV_y)+(P_{Ty}-P_y)(U_xV_z-U_zV_x)+(P_{Tz}-P_z)(-U_xV_y+U_yV_x)}{(-U_yV_z+U_zV_y)^2+(U_xV_z-U_zV_x)^2+(-U_xV_y+U_yV_x)^2} \tag{22}$$

The accurate derivative $l_i' = \frac{dl_i}{d(\alpha,\theta)} = \frac{dl_i}{d\xi_i}$ for each point $\boldsymbol{Q}_i$ on torus can be calculated as Eq. (23), where Eq. (21) is used to compute $\frac{dP_T(\alpha,\theta)}{d(\alpha,\theta)}$ and Eq. (22) to compute $\frac{dl_i}{dP_T(\alpha,\theta)}$, and then l_i' can be obtained.

$$\begin{aligned} \frac{dl_i}{d(\alpha,\theta)} = \frac{dl_i}{d\xi_i} = \frac{dl_i}{dP_{Tx}(\alpha,\theta)} \bullet \frac{dP_{Tx}(\alpha,\theta)}{d(\alpha,\theta)} \\ + \frac{dl_i}{dP_{Ty}(\alpha,\theta)} \bullet \frac{dP_{Ty}(\alpha,\theta)}{d(\alpha,\theta)} + \frac{dl_i}{dP_{Tz}(\alpha,\theta)} \bullet \frac{dP_{Tz}(\alpha,\theta)}{d(\alpha,\theta)} \end{aligned} \tag{23}$$

In the searching process, with the parameters α,θ determined, the derivative l_i' can be calculated precisely. This method is time-saving for searching the contact point without the calculations of nonlinear equations.

The proposed three methods are integrated into the isoplanar tool path generation algorithm in Sect. 4.2. Their computational efficiency is compared in the context of five-axis mold machining in Sect. 5.2.

3.4 Robustness analysis and safeguards

Numerical projection methods can encounter convergence and solution uniqueness issues in geometrically challenging regions such as sharp edges, high-curvature zones, or when the projection direction is nearly parallel to a facet normal. To ensure practical reliability, our framework – particularly the adopted geometry-based torus-search method – integrates multi-level safeguards, which are systematically analyzed below.

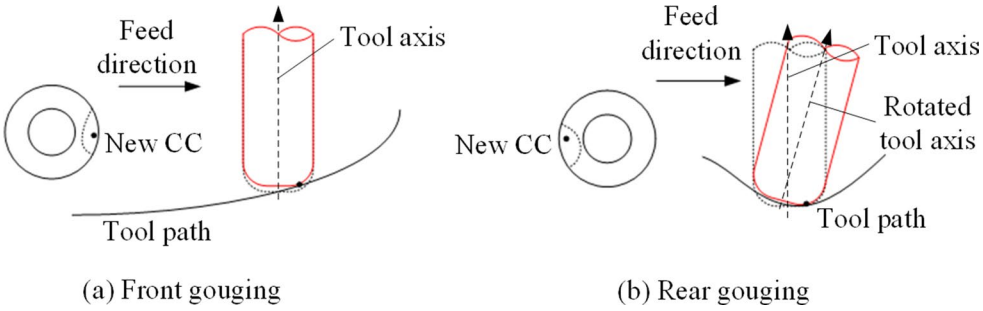


Fig. 21 The gouge categories

A geometrically constrained solution space ensures solution uniqueness. The solver avoids a full parameter search by restricting valid cutter contact points to the physical region bounded by the normal vectors of the two adjacent triangular facets (see Fig. 12). This constraint, derived from the discretized model's geometry, limits the search path on the torus to a narrow, meaningful parameter interval, thereby excluding mathematically possible but physically invalid or ambiguous solutions – especially critical when the projection direction approaches a facet normal.

Controlled iterative solving guarantees convergence stability. High-quality initialization is achieved by deriving the initial search parameters directly from the adjacent facet geometry, placing the starting point close to the true solution and preventing divergence in complex regions. Coupled with a strict termination tolerance (1.0×10^{-6}). and continuous gradient monitoring, this design enables rapid and stable convergence, typically within five iterations, even near high-curvature areas or sharp edges.

A hierarchical fallback strategy provides a system-level safety net. Embedded within a deterministic projection pipeline (see Sect. 2.1 and Fig. 7), the core solver is backed by an automatic fallback mechanism. If the torus-edge projection fails to converge or returns an invalid result due to extreme geometric conditions, the system seamlessly reverts to the robust and unconditionally stable cutter-vertex projection. This ensures that a valid, gouge-free cutter location is always generated, furnishing fundamental algorithmic robustness.

In summary, the integrated approach of proactive geometric constraint, controlled iteration, and passive fault-tolerant fallback constitutes a complete robustness strategy. The effectiveness of this design has been empirically validated in Sect. 5, where high-quality, gouge-free tool paths were successfully generated for complex blade and mold parts containing numerous challenging geometric features.

4 Applications of projection methods

This section details the practical application of the proposed torus-to-edge projection methods, focusing on two key areas: enhancing an existing gouge avoidance algorithm and generating entirely new gouge-free tool paths.

4.1 Gouge avoidance

Gouging in five-axis machining is typically categorized as either front gouge or rear gouge, as illustrated in Fig. 21 [9]. This section demonstrates how the analytical tool-axis-aligned projection method (Sect. 2.3, Case 1) is integrated into a gouge avoidance algorithm to rectify imperfections in tool paths generated via contact-driven strategies.

The algorithm builds upon our previous work [9], which employs a robust cutter-partitioning method to identify and classify gouging. The core enhancement introduced here is the use of the analytical projection to efficiently eliminate detected gouges. The specific procedure, outlined in Fig. 22, is as follows:

(1) The cutter-partitioning algorithm identifies the gouge category.
(2) If front gouging is detected, the cutter is adjusted by projecting it onto the part surface along the tool axis, effectively "lifting and dropping" it to a gouge-free position.
(3) If rear gouging is detected, the tool axis is first rotated to transform the rear gouge into a front gouge. The

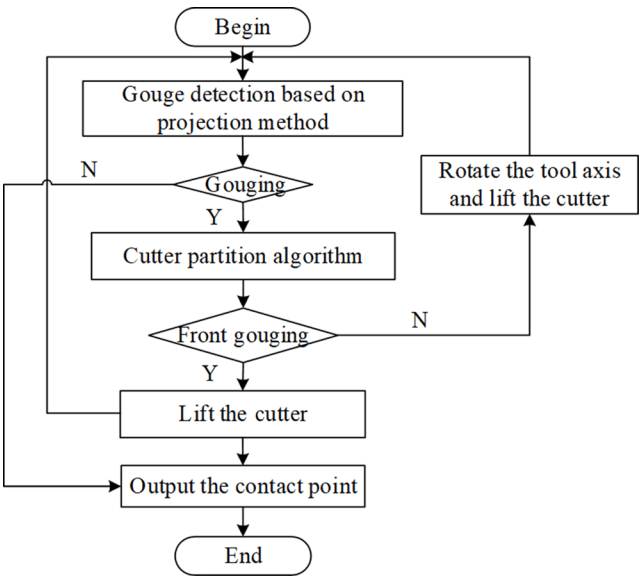


Fig. 22 The flow chart of gouge detection and elimination

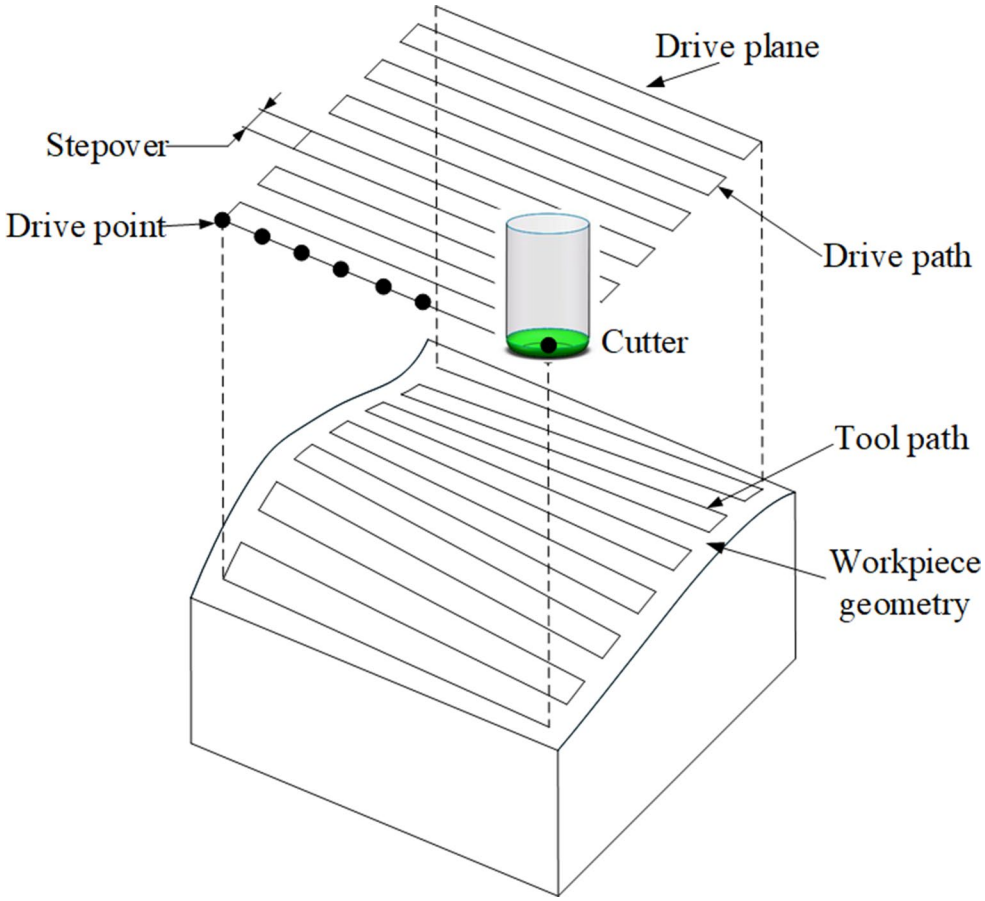


Fig. 23 Iso-planar tool path generation based on projection method

analytical tool-axis projection is then applied to eliminate it.

This integrated approach ensures that cutting is performed with the front edge of the cutter, maintains smooth tool orientation variation, and is applicable to arbitrary free-form surfaces machined with a general APT cutter. The application of this algorithm in five-axis blade machining is validated in Sect. 5.1.

4.2 Gouge-Free Iso-Planar tool path generation

This section presents a projection-based strategy for directly generating gouge-free tool paths, moving beyond mere correction. The proposed iso-planar algorithm seamlessly integrates gouge avoidance with cutter location computation, as illustrated in Fig. 23.

The process begins with the planning of a preliminary drive path. The core of the method lies in using the numerical arbitrary-direction projection (the three methods from Sect. 3) to project the cutter onto the faceted model along a strategically chosen direction. This inherentl y ensures the resulting tool path is gouge-free. Furthermore, the scallop height is calculated and controlled during this process. The specific workflow is detailed in Fig. 24. The algorithm efficiently locates the highest possible contact point on the surface that satisfies both the gouge-free condition and a predefined computational accuracy.

The application of this iso-planar algorithm, powered by arbitrary-direction projection, in five-axis mold machining is presented and evaluated in Sect. 5.2.

5 Implementation and validation

Following the theoretical development of the projection methods and their applications, this section details their software implementation and experimental validation. The proposed framework has been implemented in C++within the Visual Studio 2019 environment and integrated as a functional module of our proprietary CAM software, NC Blade.

Validation is conducted through two distinct industrial case studies:

Section 5.1 demonstrates the application of the analytical tool-axis projection method for gouge avoidance in five-axis blade machining.

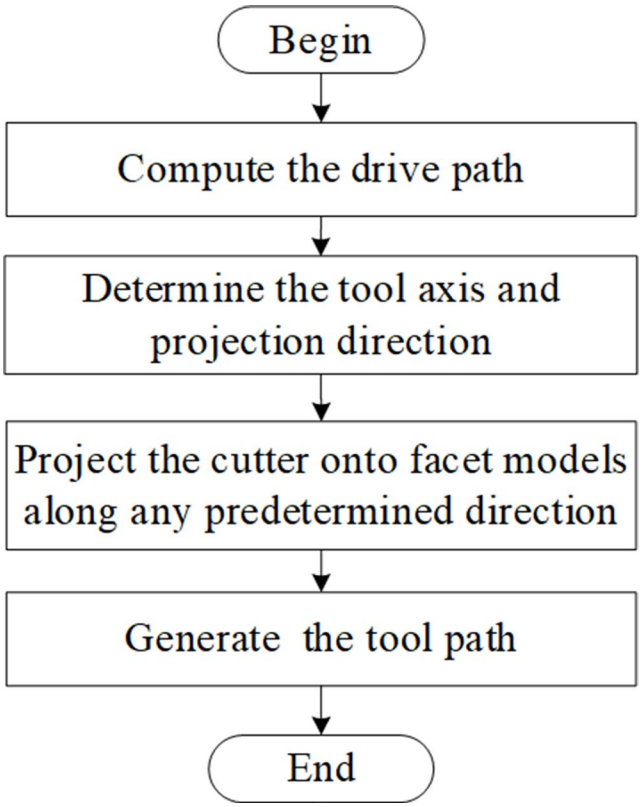


Fig. 24 The flow chart of iso-planar method

Section 5.2 evaluates the performance of the three numerical arbitrary-direction projection methods for generating gouge-free tool paths in five-axis mold machining.

Both studies employ a consistent validation methodology: tool paths generated in NC Blade are first verified via machining simulation in VERICUT® software and then validated through physical machining experiments.

5.1 Five-axis blade machining with gouge avoidance

This case study validates the efficacy of the gouge avoidance algorithm (Sect. 4.1) using the analytical tool-axis projection method for finishing the airfoil surface of a compressor blade.

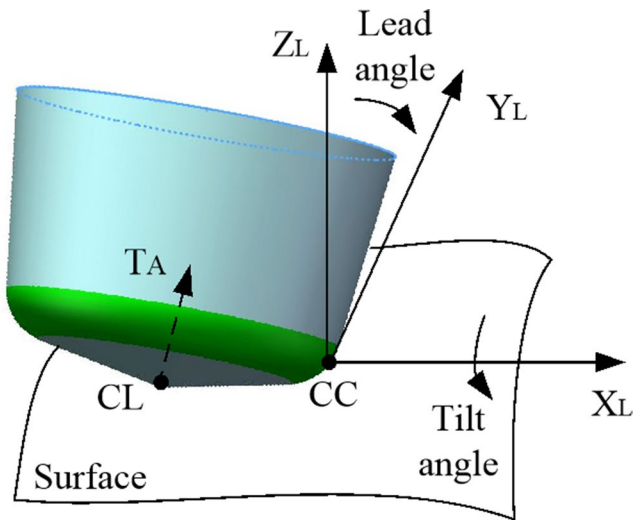


Fig. 26 Illustration of inclination angle and tilt angle in 5-axis machining

5.1.1 Experimental setup

Workpiece: A compressor blade (298 mm × 100 mm × 58 mm) with a complex airfoil surface (157 mm × 62 mm) comprising four compound surfaces (Fig. 25).

Tool Axis Orientation: Defined using lead and tilt angles relative to a Local Coordinate System (LCS) at the cutter contact (CC) point, as illustrated in Fig. 26.

Key Parameters: See Table 1.

Table 1 Experimental parameters

Part/Feature	Compressor Blade/Airfoil surface
Cutter (mm)	D20R1.6
Cutting parameters	Lead angle: 15° Tilt angle: 0°
Machining tolerance (mm)	0.05
Tool path pattern	Spiral

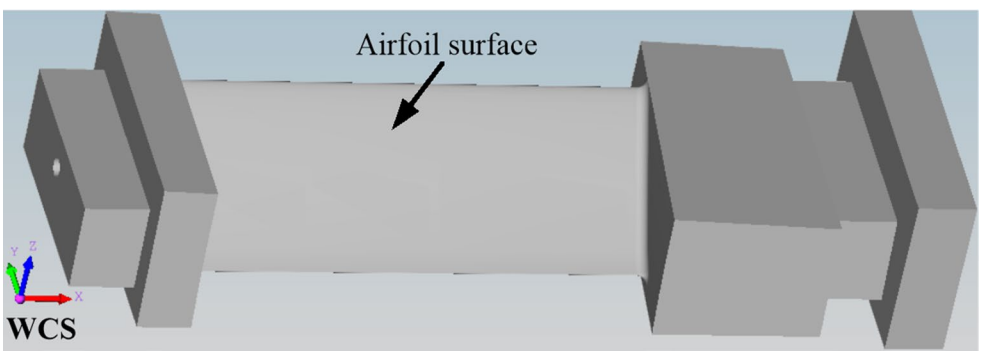


Fig. 25 The airfoil surface of the compressor blade part

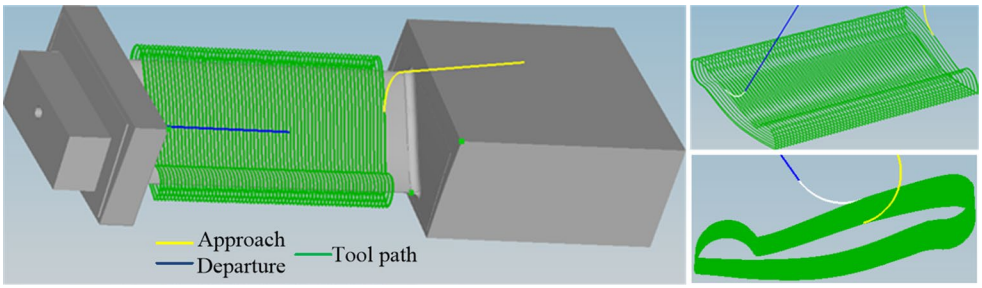


Fig. 27 The generated tool path in Blade NC

Table 2 Comparison of the analytical and numerical tool-axis projection methods

Simulation	Analytical tool-axis projection method	Numerical tool-axis projection method
Average computation time (s)	139.682	184.474

5.1.2 Results and analysis

(1) Gouge Elimination: The algorithm successfully identified and eliminated both front and rear gouging. Front gouge was avoided by projecting the cutter along the tool axis, while rear gouge was addressed by first rotating the tool axis to convert it into a front gouge, which was then resolved via projection. The resulting gouge-free tool path is shown in Fig. 27.

(2) Computational Efficiency: A comparative analysis between the analytical method and the numerical geometry-based torus-search method revealed a 24.28% reduction in average computation time when using the analytical solution (Table 2), underscoring its performance advantage.

(3)Simulation Verification: VERICUT® simulations confirmed the algorithm's effectiveness. As shown in Fig. 28, tool paths generated without gouge avoidance exhibited significant gouging (exceeding 0.05 mm) in concave regions, while paths generated with the proposed algorithm were completely gouge-free.

Fig. 29 Machine tool and cutter in real cutting

5.1.3 Physical machining and metrology validation

(1) Machining Experiment: The experiment was conducted on a C.B.Ferrari A176 five-axis machining center (Fig. 29) using a D20R1.6 toroidal cutter. The results, shown in Fig. 30, verify the effectiveness of the gouge avoidance algorithm, demonstrating a workpiece surface free of gouging.

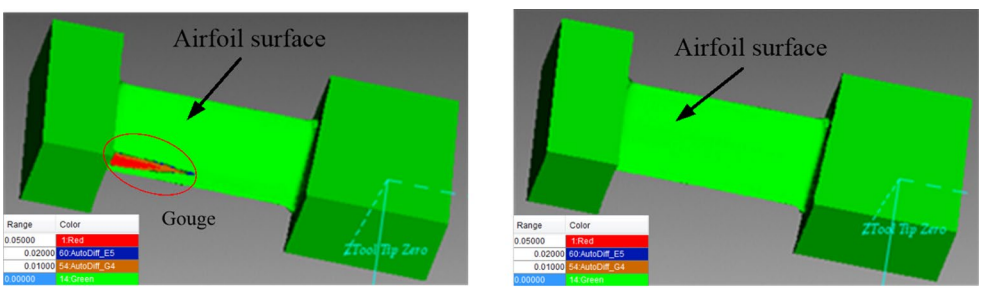


Fig. 28 Simulated machining results for blade part surface in VERICUT software

(a) Simulated machining results with gouge in concave area

(b) Simulated machining results without gouge in concave area

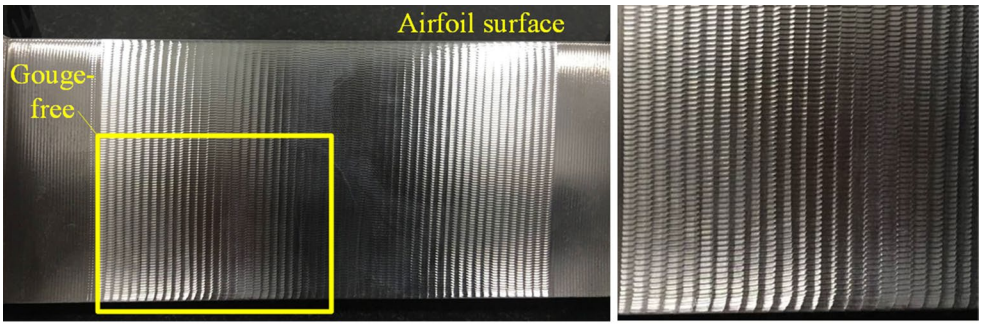


Fig. 30 Machining results for the airfoil surface of the compressor blade

(a) Machining results without gouge in concave area (b) Details of gouge-free area

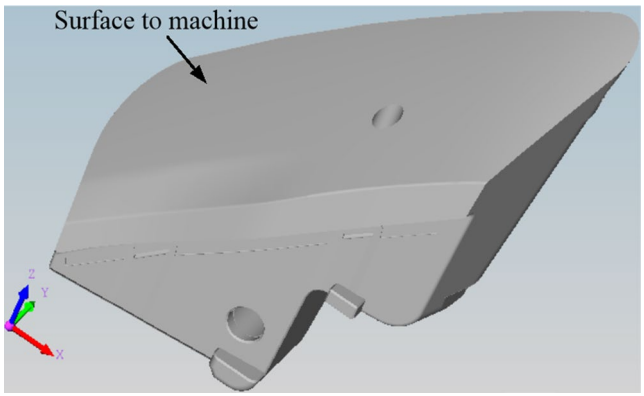


Fig. 31 The injection mold of automobile tail light shell

Table 3 Experimental parameters

Part	The automobile tail light shell
Cutter (mm)	D6R3
Cutting parameters	Lead angle: 15° Tilt angle: 0°
Machining tolerance (mm)	0.05
Scallop height (mm)	0.01
Tool axis mode	Lead and tilt

5.2 Five-axis projection-based iso-planar mold machining

This case study evaluates the three numerical arbitrary-direction projection methods (Sect. 3) for generating gouge-free, iso-planar tool paths on a complex mold surface.

5.2.1 Experimental setup

Workpiece: An automobile tail light shell injection mold (146 mm × 57 mm) with complex compound surfaces (Fig. 31).

Key Parameters: See Table 3. A search tolerance of 10^(−6) mm was used for all numerical methods.

5.2.2 Results and analysis

Tool Path Generation: All three methods successfully generated gouge-free tool paths based on the iso-planar strategy (Sect. 4.2). An example tool path generated by the torus-search method is shown in Fig. 32.

Computational Efficiency Comparison: The geometry-based torus-search method demonstrated superior efficiency, as summarized in Table 4. It ran 3.386 times faster than the algebra-based Bézier clipping method and 1.408 times faster than the geometry-based edge-search method. The Bézier clipping method incurred the highest cost due to its recursive subdivision process.

Table 4 Comparison of different three methods for five-axis mold machining

Simulation	Algebra-based Bézier clipping method	Geometry-based edge-search method	Geometry-based torus-search method
Average computation time (s)	12080.5	5024.68	3567.44

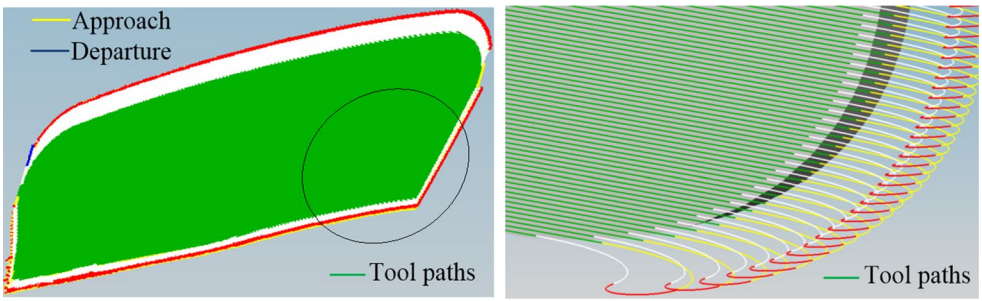
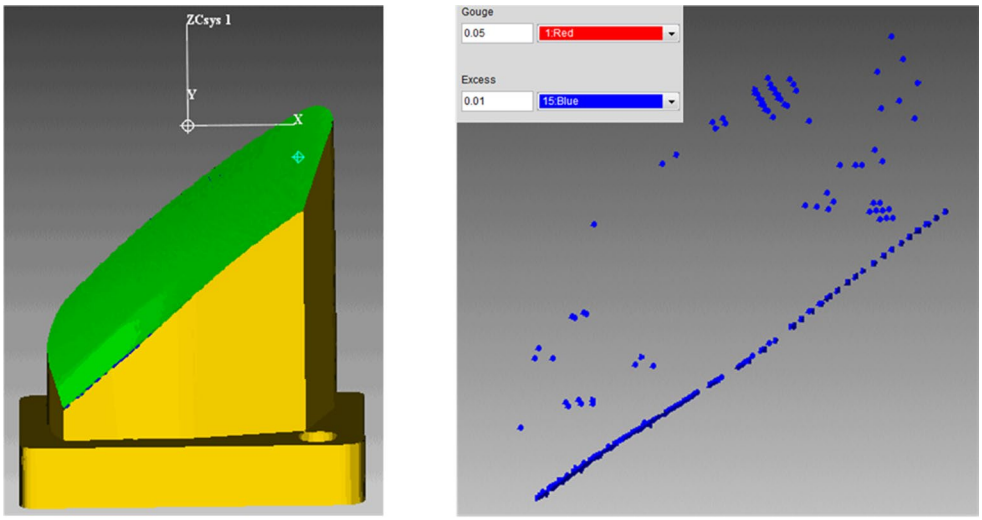


Fig. 32 Tool path of machining the automobile tail light shell

(a) Tool paths generated by torus-search method (b) The details of tool paths



(a) The simulated machining mold surface

(b) The simulated results : gouging and sacllop

Fig. 33 Simulated machining of mold surface in VERICUT software

5.2.3 Validation via simulation and physical machining

Simulation: The tool path from the torus-search method was simulated in VERICUT®. The results (Fig. 33) confirmed a gouge-free machined surface. The maximum scallop height was measured at 0.0159 mm, slightly above the preset 0.01 mm but deemed acceptable considering simulation approximations.

Physical Machining: A physical cutting experiment was performed on a JT-GL8-V five-axis machining center (Fig. 34) using the most efficient torus-search method. The final machined mold (Fig. 35) exhibited a regular, uniform tool path pattern and a high-quality surface finish entirely free of gouging and visible scales, thereby validating the practical effectiveness of the proposed method.

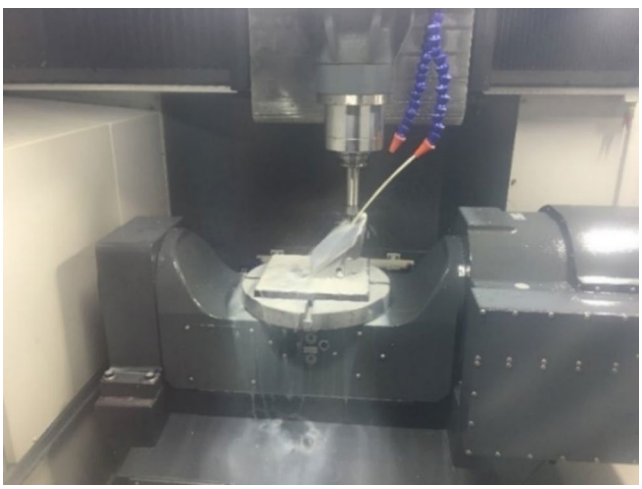


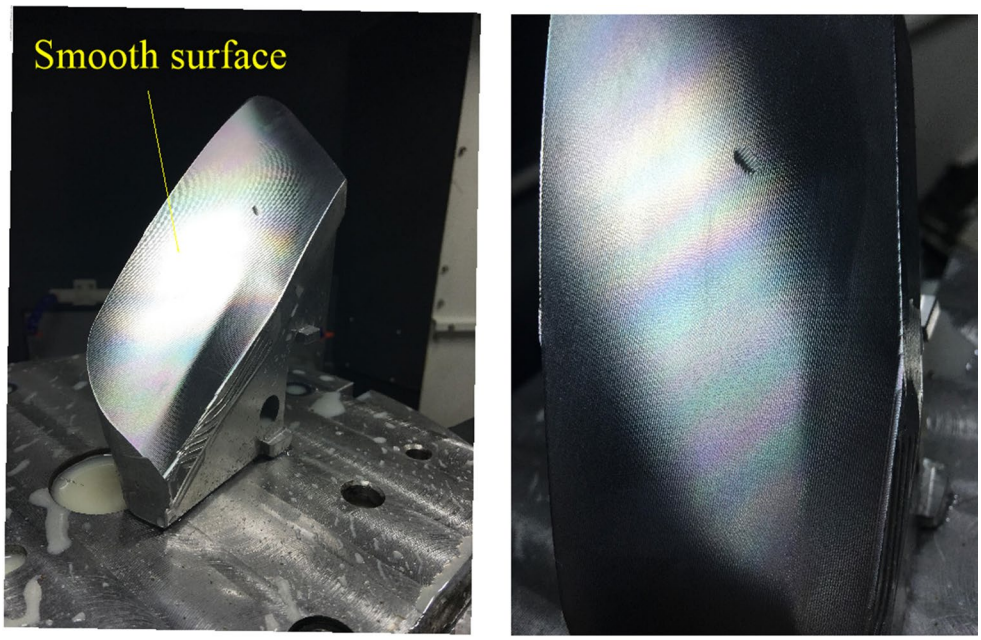
Fig. 34 Machine tool

5.3 Relationship between projection accuracy and machining quality

The final quality of a machined surface is fundamentally governed by the accuracy of the computed tool path, which in projection-based methods is directly determined by the precision of the initial projection calculation. This causal relationship operates through two interconnected mechanisms: the assurance of point-wise geometric accuracy to prevent local gouging, and the enabling of proactive control over the path-wise conformity to manage scallop height and contour fidelity.

High-precision projection first establishes the foundation for gouge-free machining by guaranteeing the geometric accuracy of the Cutter Contact (CC) point. This chain of precision begins with the calculation of the CC point, which defines the theoretical tangency between the tool and the workpiece. Our framework ensures this source accuracy through a hybrid solver strategy: for the reducible special cases (tool-axis-aligned and ball-end cutter projection), zero-error analytical solutions are employed; for the general case of arbitrary-direction projection, custom numerical solvers converge to a strict tolerance (1.0×10^{-6}). The resulting precise CC point, combined with the prescribed tool orientation, yields the accurate Cutter Location (CL) point. This serves as the direct safeguard against local gouging, ensuring the tool physically occupies its intended, interference-free position.

Furthermore, precise projection enables the proactive control of overall path conformity (chord error), which is key to ensuring surface smoothness and contour accuracy. The final surface quality depends more critically on the



(a) Machining results using the Geometry-based torus-search method

(b) Details of machining results

Fig. 35 Results of Machining experiment for mold surface

conformity of the continuous tool trajectory to the design surface, i.e., the chord error. Unlike traditional post-hoc correction methods, our strategy proactively integrates tolerance compliance into the projection stage itself. As illustrated in Fig. 36, if the chord error between projected CC points ($\boldsymbol{Q}_1$ and $\boldsymbol{Q}_2$ from drive points $\boldsymbol{P}_1$ and $\boldsymbol{P}_2$) exceeds the machining tolerance, a binary search algorithm proactively inserts new drive points and performs local re-projection. This closed-loop process ensures that the chord error between successive tool positions is systematically and algorithmically controlled at the time of path generation. The reliability of this entire control loop is contingent on the initial projection accuracy; only trustworthy CC points enable valid chord error assessment and effective, localized refinement.

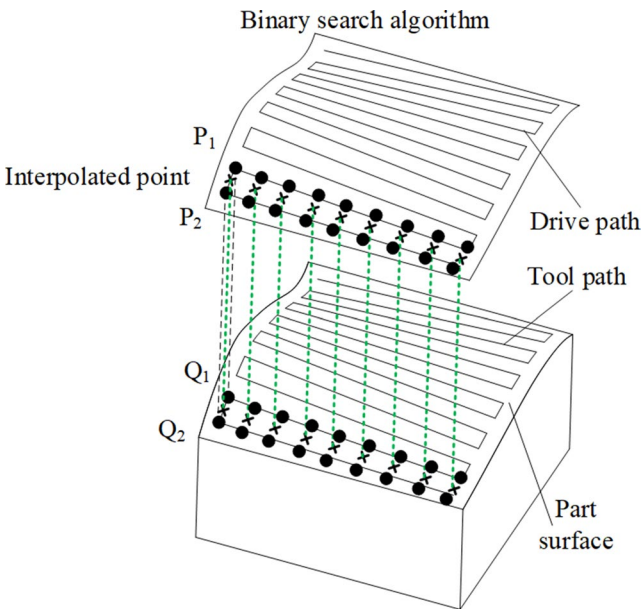


Fig. 36 Schematic of drive-path refinement and point interpolation for tolerance control

In summary, initial projection accuracy is the cornerstone determining final machining quality. It secures the geometric correctness of individual tool positions to prevent gouging and provides the essential precise geometric data required for the proactive, in-process control of path conformity. This integrated "point-precision-to-path-conformity" chain ensures that the theoretical conditions for gouge-free machining and scallop height control are faithfully translated into the physical tool trajectory. The effectiveness of this principle is empirically validated by the high-quality, gouge-free machining results presented for both the blade (Sect. 5.1) and mold (Sect. 5.2) components.

6 Conclusions and future work

6.1 Conclusion

This paper has introduced a comprehensive and unified framework for projection-based, gouge-free tool path generation in five-axis machining of faceted models. The core

of this work is the resolution of the long-standing challenge of arbitrary-direction torus-edge projection, which is pivotal for leveraging the full flexibility of five-axis systems.

We have established a unified projection framework that seamlessly handles cutter-facet, cutter-edge, and cutter-vertex projections along arbitrary directions, forming a complete pipeline for generating intrinsically gouge-free tool paths.

A significant theoretical advancement was made through our hybrid analytical-numerical solver, which demonstrates that two critical special cases, tool-axis-aligned torus-edge projection and arbitrary-direction ball-edge projection, can be reduced to a quartic equation and solved analytically. The implementation of this analytical solution for tool-axis projection resulted in a 24.28% reduction in computation time compared to its numerical counterpart.

For the general arbitrary-direction torus-edge projection problem, we developed three custom numerical methods, with extensive experiments confirming that the geometry-based torus-search method is the most computationally efficient, outperforming the algebra-based Bézier clipping method by a factor of 3.386 and the geometry-based edge-search method by a factor of 1.408.

The practical efficacy of the proposed methods was rigorously validated through two industrial applications, where the analytical method was successfully integrated into a gouge avoidance algorithm for blade machining that effectively eliminated both front and rear gouging, while the numerical methods were employed in a projection-based iso-planar strategy for mold machining, producing tool paths that are regular, uniform, and devoid of gouging as confirmed by both simulation and physical machining experiments.

6.2 Future work

While this work provides a robust foundation, two key areas present promising directions for future research to further enhance performance and precision.

Accelerating candidate facet selection represents one key direction, as the current projection process requires checking all facet elements within the AABB's shadow. Future work will therefore focus on developing an intelligent screening strategy, which could leverage spatial indexing structures or machine learning to rapidly prioritize and select the most relevant candidate facets. This approach is expected to significantly reduce the computational overhead associated with the projection step.

Enhancing precision through direct CAD projection constitutes another critical goal, addressing the inherent limitations that faceted models impose on computed cutter location accuracy. The current framework successfully generates gouge-free tool paths from facet models by actively controlling the surface approximation error against the machining tolerance, as noted in Sect. 2.1. This establishes a direct trade-off between computational efficiency (model size) and geometric fidelity. To transcend this inherent limitation of faceted-model approaches, a primary research direction is to evolve the projection algorithms to operate directly on the original CAD representation (e.g., NURBS surfaces). This advancement would eliminate the surface approximation error entirely, enable the generation of geometrically exact and potentially higher-order continuous tool paths, and resolve the accuracy-efficiency compromise.

Author contributions Xiyan Li: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. Pengcheng Hu: Methodology, Project administration, Supervision, Writing – review & editing. Lulu Huang: Data curation, Investigation, Machining experiment simulation. Haokun Chen: Data curation, Investigation, Physical machining experiments. Molong Duan: Methodology, Supervision. Xinfang Zhang: Resources, Writing – review & editing.

Funding This work is supported by the of Guangdong Provincial Department of Education (2025KTSCX194) and Guangdong S&T Program (2025A0505000030).

Declarations

Competing Interests The authors have no relevant financial or non-financial interests to disclose.

References

1. Lasemi A, Xue D, Gu P (2010) Recent development in CNC machining of freeform surfaces: A state-of-the-art review. Comput -Aided Des 42:641–654. https://doi.org/10.1016/j.cad.2010.04.002
2. Jiang X, Lu W, Zhang Z (2018) An approach for improving the machining efficiency and quality of aerospace curved thin-walled parts during five-axis NC machining. Int J Adv Manuf Technol 97:2477–2488. https://doi.org/10.1007/s00170-018-2129-0
3. Tang K, Cheng CC, Dayan Y (1995) Offsetting surface boundaries and 3-axis gouge-free surface machining. Comput-Aided Des 27:915–927. https://doi.org/10.1016/0010-4485(96)83775-4
4. Bedi S, Ismail F, Mahjoob MJ, Chen Y (1997) Toroidal versus ball nose and flat bottom end mills. Int J Adv Manuf Technol 13:326–332. https://doi.org/10.1007/BF01178252
5. Jensen CG, Red WE, Pi J (2002) Tool selection for five-axis curvature matched machining. Comput-Aided Des 34:251–266. https://doi.org/10.1016/S0010-4485(01)00086-0
6. Lee Y-S (1998) Mathematical modelling using different endmills and tool placement problems for 4- and 5-axis NC complex surface machining. Int J Prod Res 36:785–814. https://doi.org/10.1080/002075498193697
7. Patel K, Salas Bolaños G, Bassi R, Bedi S (2011) Optimal tool shape selection based on surface geometry for three-axis CNC machining. Int J Adv Manuf Technol 57:655–670. https://doi.org/10.1007/s00170-011-3304-8

8. Li Z, Tang K (2021) Partition-based five-axis tool path generation for freeform surface machining using a non-spherical tool. J Manuf Syst 58:248–262. https://doi.org/10.1016/j.jmsy.2020.12.004
9. Li X, Lee C-H, Hu P, Zhang Y, Yang F (2018) Cutter partition-based tool orientation optimization for gouge avoidance in five-axis machining. Int J Adv Manuf Technol 95:2041–2057. https://doi.org/10.1007/s00170-017-1263-4
10. Jun C-S, Kim D-S, Park S (2002) A new curve-based approach to polyhedral machining. Comput-Aided Des 34:379–389. https://doi.org/10.1016/S0010-4485(01)00110-5
11. Lee S-G, Kim H-C, Yang M-Y (2008) Mesh-based tool path generation for constant scallop-height machining. Int J Adv Manuf Technol 37:15–22. https://doi.org/10.1007/s00170-007-0943-x
12. Sarkar S, Dey P (2014) A new iso-parametric machining algorithm for free-form surface. Proc Inst Mech Eng Part E J Process Mech Eng 228:197–209. https://doi.org/10.1177/0954408913495191
13. He W, Lei M, Bin H (2009) Iso-parametric CNC tool path optimization based on adaptive grid generation. Int J Adv Manuf Technol 41:538–548. https://doi.org/10.1007/s00170-008-1500-y
14. Ding S, Mannan MA, Poo AN, Yang DCH, Han Z (2003) Adaptive iso-planar tool path generation for machining of free-form surfaces. Comput-Aided Des 35:141–153. https://doi.org/10.1016/S0010-4485(02)00048-9
15. Feng H-Y, Teng Z (2005) Iso-planar piecewise linear NC tool path generation from discrete measured data points. Comput Aided Des 37:55–64. https://doi.org/10.1016/j.cad.2004.04.001
16. Hu P, Chen L, Tang K (2017) Efficiency-optimal iso-planar tool path generation for five-axis finishing machining of freeform surfaces. Comput-Aided Des 83:33–50. https://doi.org/10.1016/j.cad.2016.10.001
17. Ma J, Lu X, Li G, Qu Z, Qin F (2020) Toolpath topology design based on vector field of tool feeding direction in sub-regional processing for complex curved surface. J Manuf Process 52:44–57. https://doi.org/10.1016/j.jmapro.2020.01.036
18. Su C, Jiang X, Huo G, Sun Y, Zheng Z (2020) Initial tool path selection of the iso-scallop method based on offset similarity analysis for global preferred feed directions matching. Int J Adv Manuf Technol 106:2675–2687. https://doi.org/10.1007/s00170-019-04789-6
19. Hao J, He D, Li Z, Hu P, Chen Y, Tang K (2024) Efficient cutting path planning for a non-spherical tool based on an iso-scallop height distance field. Chin J Aeronaut 37:496–510. https://doi.org/10.1016/j.cja.2023.12.005
20. Hao J, Hu P, He D, Cheng K, Li Z, Li X, Chen H, Du B (2025) Surface partitioning and iso-scallop field-based five-axis machining method with a non-spherical cutting tool. Chin J Aeronaut 103676. https://doi.org/10.1016/j.cja.2025.103676
21. Li Y-F, Li J-R, Xie H-L, Wang Q-H (2024) The generation of supplementary toolpaths for surface machining based on rapid prediction of scallop height distributions. Int J Adv Manuf Technol 135:1205–1220. https://doi.org/10.1007/s00170-024-14415-9
22. Hu P, Zhou H, Tang K, Lee C, Chen J, Yang J, Li L (2018) Spiral curve-based efficient five-axis sweep scanning of barrel-shaped surfaces. J Manuf Sci Eng 140:071001. https://doi.org/10.1115/1.4039383
23. Zhang Y, Zhu L, Zhao P, Hu P, Zhao X (2022) Skeleton curve-guided five-axis sweep scanning for surface with multiple holes. IEEE Trans Autom Sci Eng 19:2471–2486. https://doi.org/10.1109/TASE.2021.3087353
24. Zhou H, Lang M, Hu P, Su Z, Chen J (2019) The modeling, analysis, and application of the in-process machining data for CNC machining. Int J Adv Manuf Technol 102:1051–1066. https://doi.org/10.1007/s00170-018-2963-0
25. Chiou C-J, Lee Y-S (2002) A machining potential field approach to tool path generation for multi-axis sculptured surface machining. Comput Aided Des 34:357–371. https://doi.org/10.1016/S0010-4485(01)00102-6
26. Hu P, Tang K (2016) Five-axis tool path generation based on machine-dependent potential field. Int J Comput Integr Manuf 29:636–651. https://doi.org/10.1080/0951192X.2015.1068451
27. Makhanov SS (2022) Vector fields for five-axis machining. A survey. Int J Adv Manuf Technol 122:533–575. https://doi.org/10.1007/s00170-022-09445-0
28. Xu J, Jin C (2013) Boundary-conformed machining for trimmed free-form surfaces based on mesh mapping. Int J Comput Integr Manuf 26:720–730. https://doi.org/10.1080/0951192X.2013.766934
29. Kim S-J, Yang M-Y (2005) Triangular mesh offset for generalized cutter. Comput-Aided Des 37:999–1014. https://doi.org/10.1016/j.cad.2004.10.002
30. Xu J, Sun Y, Zhang L (2015) A mapping-based approach to eliminating self-intersection of offset paths on mesh surfaces for CNC machining. Comput-Aided Des 62:131–142. https://doi.org/10.1016/j.cad.2014.11.010
31. Yau H-T, Chuang C-M, Lee Y-S (2004) Numerical control machining of triangulated sculptured surfaces in a stereo lithography format with a generalized cutter. Int J Prod Res 42:2573–2598. https://doi.org/10.1080/00207540410001671651
32. Hwang JS, Chang T-C (1998) Three-axis machining of compound surfaces using flat and filleted endmills. Comput-Aided Des 30:641–647. https://doi.org/10.1016/S0010-4485(98)00021-9
33. Duvedi RK, Bedi S, Batish A, Mann S (2014) A multipoint method for 5-axis machining of triangulated surface models. Comput-Aided Des 52:17–26. https://doi.org/10.1016/j.cad.2014.02.008
34. Duvedi RK, Bedi S, Batish A, Mann S (2015) Numeric implementation of drop and tilt method of 5-axis tool positioning for machining of triangulated surfaces. Int J Adv Manuf Technol 78:1677–1690. https://doi.org/10.1007/s00170-014-6636-3
35. Duvedi RK, Bedi S, Batish A, Mann S (2016) The edge–torus tangency problem in multipoint machining of triangulated surface models. Int J Adv Manuf Technol 82:1959–1972. https://doi.org/10.1007/s00170-015-7532-1
36. Hansen A, Arbab F (1988) Fixed-axis tool positioning with built-in global interference checking for nc path generation. IEEE J Robot Autom 4:610–621. https://doi.org/10.1109/56.9299
37. Cao Z, Zhao Y, Wang R, Yang C, Wang Y, Liu Z (2022) A novel collision-free nc code generation method with fixed tool orientation vector using the geometric decoupling strategy based on the three rotary axes milling head. Int J Adv Manuf Technol 122:3255–3279. https://doi.org/10.1007/s00170-022-09890-x
38. Sederberg TW, Nishita T (1990) Curve intersection using Bézier clipping. Comput Aided Des 22:538–549. https://doi.org/10.1016/0010-4485(90)90039-F
39. Bartoň M, Jüttler B (2007) Computing roots of polynomials by quadratic clipping. Comput Aided Geom Des 24:125–141. https://doi.org/10.1016/j.cagd.2007.01.003
40. Mourrain B, Pavone JP (2009) Subdivision methods for solving polynomial equations. J Symb Comput 44:292–306. https://doi.org/10.1016/j.jsc.2008.04.016
41. Zhang R, Hu P, Tang K (2015) Five-axis finishing tool path generation for a mesh blade based on linear morphing cone. J Comput Des Eng 2:268–275. https://doi.org/10.1016/j.jcde.2015.06.013
42. Wu L, Xu J, Lin J, Sun Y (2024) Safe attitude corridor: A convex optimization method for tool orientation smoothing in Five-Axis CNC machining. IEEEASME Trans Mechatron 1–12. https://doi.org/10.1109/TMECH.2024.3510852
43. Lee Y-S (1997) Admissible tool orientation control of gouging avoidance for 5-axis complex surface machining. Comput Aided

Des 29:507–521. https://doi.org/10.1016/S0010-4485(97)00002-X
44. Xie H, Wang Q, Liao Z, Li J, Zhou X (2022) A rapid approach of computing multi-axis tool accessible space for general cutter model by utilizing programmable graphics pipeline. IEEEASME Trans. Mechatron. 27:3373–3384. https://doi.org/10.1109/TMECH.2021.3135417
45. Wan Y, Xu W, Zuo T-Y (2023) Tool path optimization for complex cavity milling based on reinforcement learning approach. IEEE Access 11:66793–66807. https://doi.org/10.1109/ACCESS.2023.3262169
46. Zhang Y, Li Y, Xu K (2022) Reinforcement learning–based tool orientation optimization for five-axis machining. Int J Adv Manuf Technol 119:7311–7326. https://doi.org/10.1007/s00170-022-08668-5
47. Liu T, Zhang T, Chen Y, Wang W, Jiang Y, Huang Y, Wang CCL (2025) Neural co-optimization of structural topology, manufacturable layers, and path orientations for fiber-reinforced composites. ACM Trans Graph 44:128:1-128:17. https://doi.org/10.1145/3730922
48. Demir G, Vural RA (2024) Non-cutting moving toolpath optimization with elitist non-dominated sorting genetic algorithm-II. Appl Sci 14:4471. https://doi.org/10.3390/app14114471
49. Hao J, Li Z, Li X, Xie F, He D, Tang K (2022) Partition-based 3+2-axis tool path generation for freeform surface machining using a non-spherical tool. J Comput Des Eng 9:1585–1601. https://doi.org/10.1093/jcde/qwac077