Contents lists available at ScienceDirect

Results in Engineering

journal homepage: www.sciencedirect.com/journal/results-in-engineering

Research paper

Unified 2D polygon-based CAM framework integrating tool path generation, machinability evaluation, and cutting-force simulation

Kazuki Kaneko [a,*], Hiroto Takayasu [b], Atsuya Suzuki [c], Hiroyuki Kodama [a]

[a] Faculty of Environmental, Life, Natural Science and Technology, Okayama University, 3-1-1 Tsushimanaka, Kita-ku, Okayama, Okayama 700-8530, Japan
[b] Graduate School of Science and Engineering, Ibaraki University, 4-12-1 Nakanarusawa-cho, Hitachi, Ibaraki 316-8511, Japan
[c] Mechanical Systems Engineering Program, Okayama University, 3-1-1 Tsushimanaka, Kita-ku, Okayama, Okayama 700-8530, Japan

ARTICLE INFO

Keywords:
Computer-aided manufacturing (CAM)
Polygon
Tool path generation
Machinability
Cutting force prediction

ABSTRACT

This study proposes a unified two-dimensional (2D) polygon-based computer-aided manufacturing (CAM) framework that enables tool path generation, machinability evaluation, material removal simulation, and cutting-force prediction within a single computational environment. The proposed method represents three-dimensional geometries as aggregates of orthogonal 2D polygon sets, obtained by slicing the model in the *xy*-, *yz*-, and *zx*-parallel planes and superposing the three polygonal datasets. A novel convolutional offsetting algorithm is developed to perform three-dimensional inflation and shrinkage by incorporating adjacent cross-sectional relationships, thereby achieving accurate 3D offsets independent of the slicing orientation. The inflated 2D polygons are directly utilized to generate contour and scanning tool paths, and sequential inflation–-shrinkage analysis enables visualization of unmachinable regions for tool accessibility evaluation. Furthermore, the framework integrates an instantaneous cutting force model that accurately predicts the cutting force waveform by detecting intersections between the cutting edge points and 2D polygon aggregations. The system is experimentally validated via ball-end milling. The results demonstrate that tool paths can be generated in under one minute using only a CPU. Furthermore, the simulated cutting forces closely align with experimental measurements. These findings demonstrate that the proposed 2D polygon-based framework provides an efficient and extensible foundation for integrating mechanical simulation and tool-path generation.

1. Introduction

Recent advances in machining accuracy and the widespread adoption of multi-axis machine tools have made end milling of complex-shaped components increasingly important in the manufacturing industry. Therefore, the role of computer-aided manufacturing (CAM) systems in generating efficient and reliable NC programs has expanded significantly. By providing the tool geometry, machining regions, machining strategies, and cutting conditions as input data, CAM systems can semi-automatically compute tool paths. However, the generation of highly efficient and precise tool paths requires CAM operators to appropriately determine the input parameters. As this process depends strongly on the operator experience and expertise, complete automation of tool-path generation has not been achieved in conventional CAM systems. Therefore, several studies have been conducted to develop technologies for the automatic generation of optimal tool paths. Matsukawa et al. proposed a geometric computation method for automatically generating tool paths in five-axis indexing machining employing complex STL models [1]. Mandeep et al. developed a method for deriving spiral tool-paths from 3D point-cloud models obtained via scanning to facilitate automated tool-path generation for reverse engineering [2]. Recently, methods employing machine learning techniques have been proposed for deriving optimal tool paths [3,4]. In addition to tool-path generation algorithms, considerable research has been conducted on the automatic recognition of machining features. Zhang et al. proposed a feature recognition method employing NC programs for three-axis machining [5], and several machine learning-based approaches using graph neural networks have also been reported [6]. Collision avoidance must be incorporated in tool path generation [7]. Chichell et al. proposed a high-speed collision detection method for free-form five-axis machining and developed a collision-free tool-path generation technique. Inui et al. introduced a method for calculating tool orientations that ensures collision-free conditions between the tool holder and workpiece [8]. George et al. developed a process planning

[*] Corresponding author.
E-mail address: k-kaneko@okayama-u.ac.jp (K. Kaneko).

https://doi.org/10.1016/j.rineng.2025.108948
Received 10 November 2025; Received in revised form 25 December 2025; Accepted 30 December 2025
Available online 30 December 2025

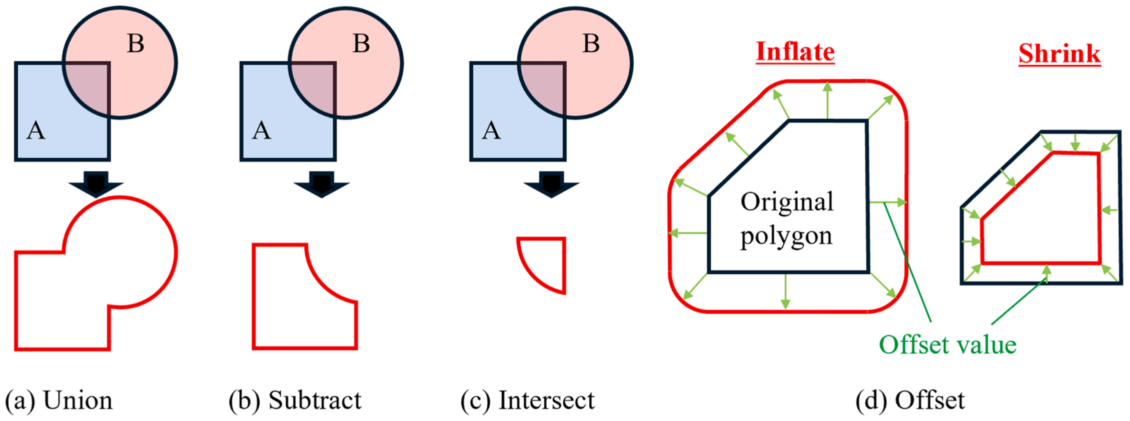


Fig. 1. Functions used in the development of the proposed system.

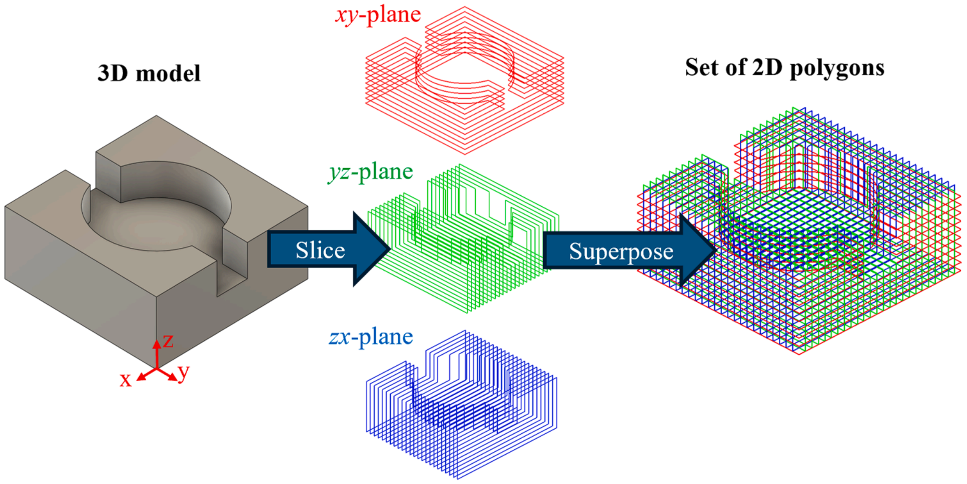


Fig. 2. 3D workpiece representation based on 2D polygons.

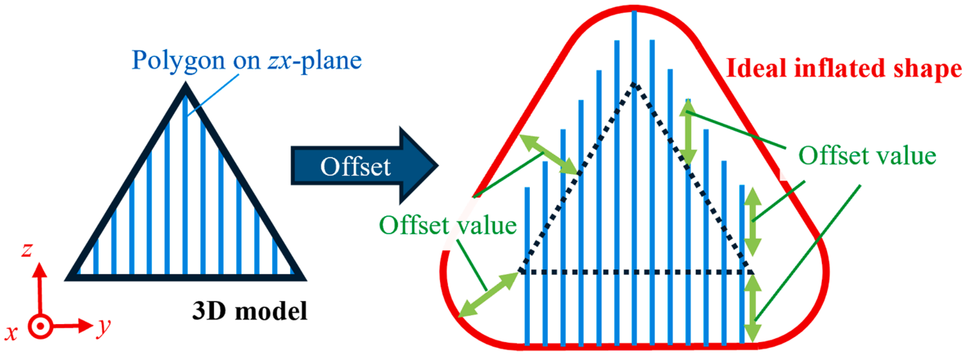
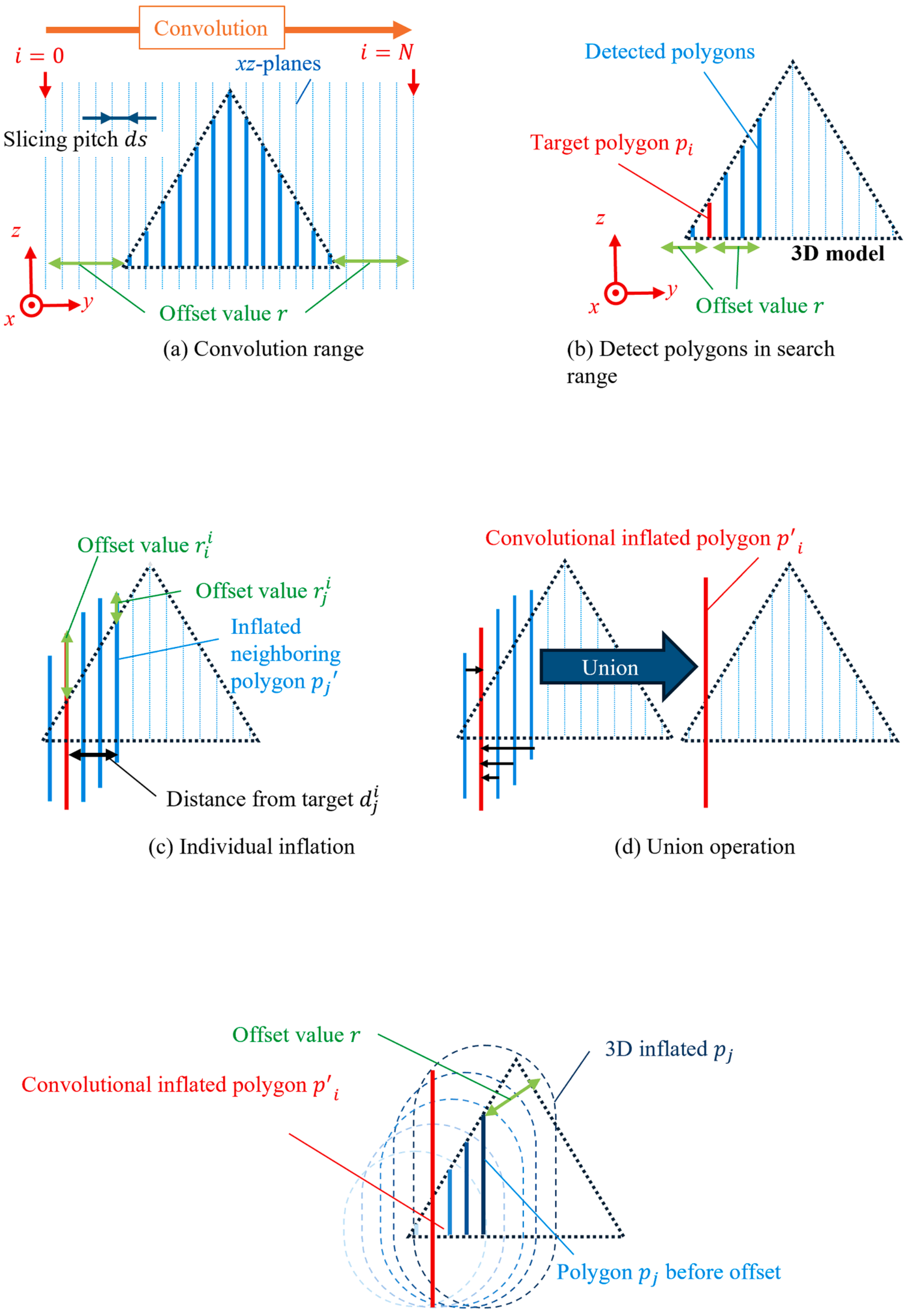


Fig. 3. Discrepancy between 3D shape offset and individual 2D polygon offset.

approach that incorporates the accessibility of cutting tools in hybrid additive–subtractive manufacturing processes [9]. However, the geometric computations required for tool-path generation and collision avoidance are highly time consuming, presenting a significant challenge in practical applications [10]. To address this issue, GPU-based acceleration methods have been proposed [11].

Although various geometric analysis methods have been developed, extensive research has been conducted on mechanical simulations aimed at optimizing cutting conditions. Because cutting-force prediction provides the basis for these simulations, numerous approaches have been proposed, including analytical model-based methods [12–14], finite element method (FEM)-based simulations [15], and machine learning-based prediction models [16,17]. By applying these cutting force models, several studies have analyzed the chatter stability [18] and predicted the machined surface topography generated under chatter-involved cutting conditions [19,20]. In addition to these dynamic analyses, simulation methods have been developed addressing machining errors caused by static deformation of the tool and workpiece owing to cutting forces [21,22]. Furthermore, multiple studies have proposed optimization of cutting conditions via mechanical simulations [23,24].

Numerous studies have been conducted to enable automated and



(a) Convolution range

(b) Detect polygons in search range

(c) Individual inflation

(d) Union operation

(e) 3D inflation of 2D polygons

Fig. 4. Convolutional 2D polygon offset (Inflation).

skill-independent machining, including tool-path generation, machining feature recognition, interference avoidance, tool accessibility analysis, and mechanical simulations involving cutting-force prediction. Integration is desired to enable closed-loop verification and optimization, in which tool-path generation and process simulation are consistently linked. However, because these studies have often been developed independently, the geometric representation employed differs depending on the purpose. For example, boundary-based models are commonly used for tool-path generation, while volumetric discretization, such as voxel- or dexel-based models, is frequently adopted for material-removal simulations and computing the engagement conditions used in force-, deformation-, and vibration-related analyses. This heterogeneity fragments CAM workflows, thereby requiring repeated geometric conversions and hindering seamless closed-loop linkages. Another practical issue is the hardware constraints introduced by GPU-oriented implementations. Although open standards such as OpenCL exist, many high-performance implementations rely on NVIDIA's CUDA ecosystem, which tends to reduce the portability across different hardware environments because CUDA only supports NVIDIA GPUs [25]. Thus, while GPU-based acceleration can effectively reduce computation time, it can also introduce vendor- and hardware-dependent constraints that limit the versatility of CAM systems.

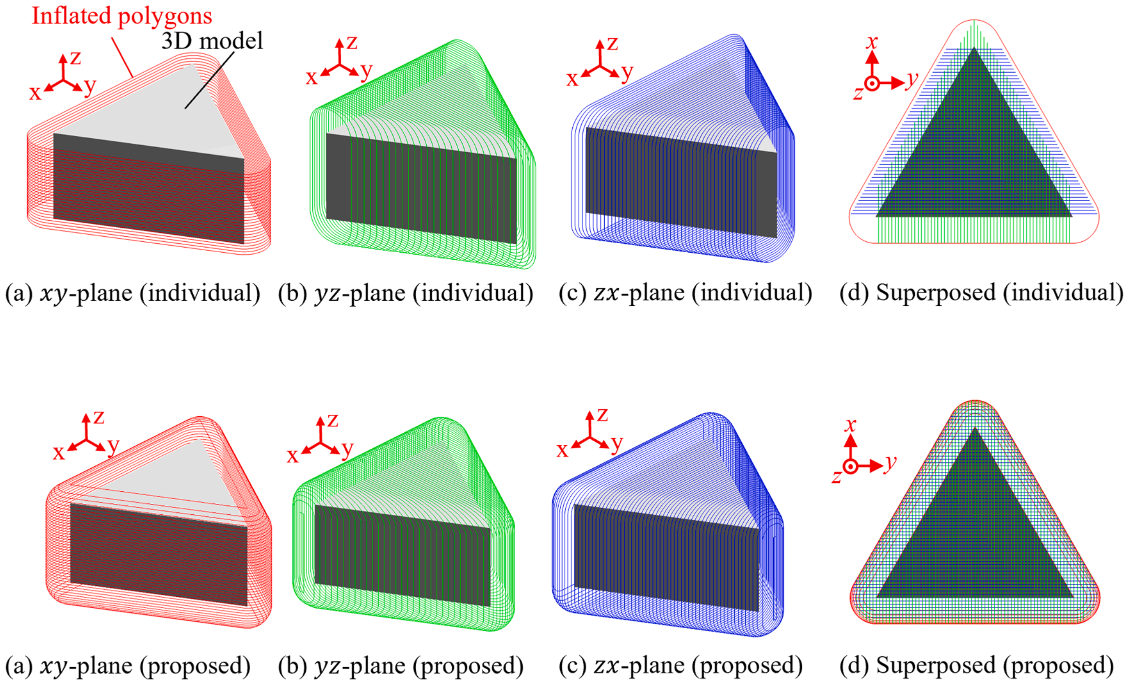

Inflated polygons
3D model
x
y
z
(a) xy-plane (individual)
(b) yz-plane (individual)
(c) zx-plane (individual)
(d) Superposed (individual)
(a) xy-plane (proposed)
(b) yz-plane (proposed)
(c) zx-plane (proposed)
(d) Superposed (proposed)

Fig. 5. Examples of individual and convolutional offset results.

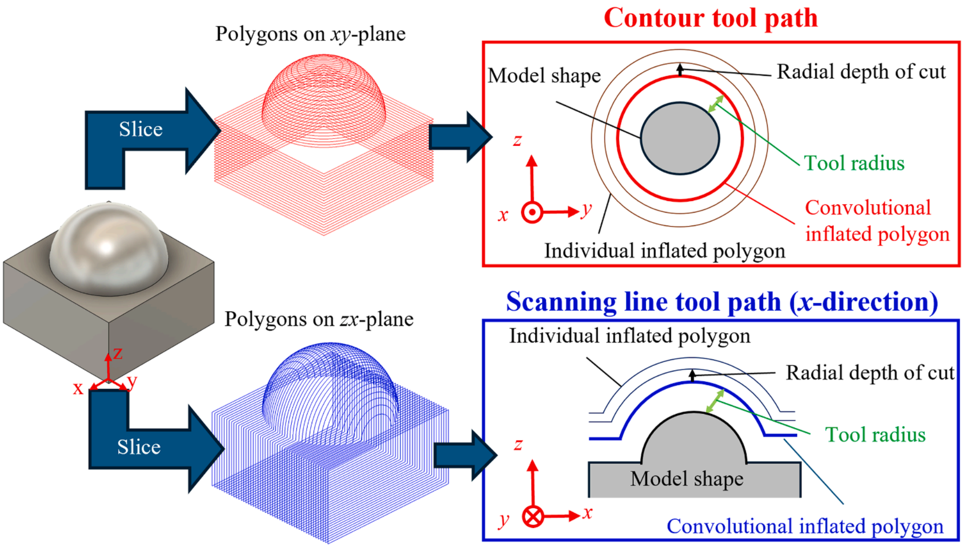


Fig. 6. Contour line and scanning line tool-path generation.

This study proposes a hardware-independent CAM framework that integrates tool-path generation, detection and visualization of unmachinable regions, material-removal simulation, and cutting-force prediction within a unified two-dimensional (2D) polygon-based geometric modeling scheme. The proposed method employs Boolean and convolutional operations of 2D polygons on CPU to achieve a computational efficiency comparable to that of GPU-accelerated systems while maintaining full software portability. This approach reduces operator dependency in practical machining and provides a promising foundation for developing flexible digital-twin-based manufacturing systems.

The remainder of this paper is structured as follows: Section 2 introduces the proposed geometric framework based on 2D polygon Boolean operations. Section 3 explains the algorithms for tool-path generation, detection and visualization of unmachinable regions, material removal simulation, and cutting-force prediction integrated within the proposed framework. Section 4 provides experimental validation and a discussion of the computational efficiency. Finally, Section 5 concludes the paper and outlines future research directions.

2. Geometric model based on 2D polygon

2.1. 3D shape representation

This study proposes a geometric modeling framework employing 2D polygons. The developed CAM software was implemented using “Clipper2,” an open-source geometric processing library for 2D polygon operations. As illustrated in Fig. 1, Clipper2 enables Boolean operations, including union, subtraction, and intersection, as well as offset calculations [26,27]. By combining these fundamental functions, a comprehensive 2D-polygon-based system was constructed, capable of

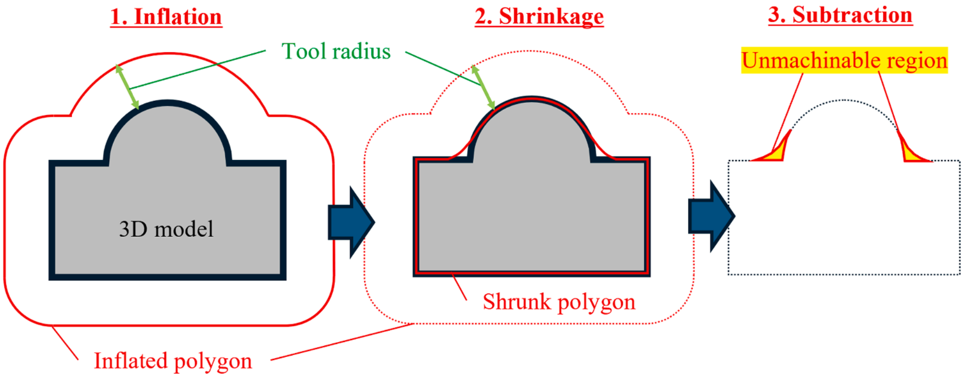


Fig. 7. Unmachinable area detection.

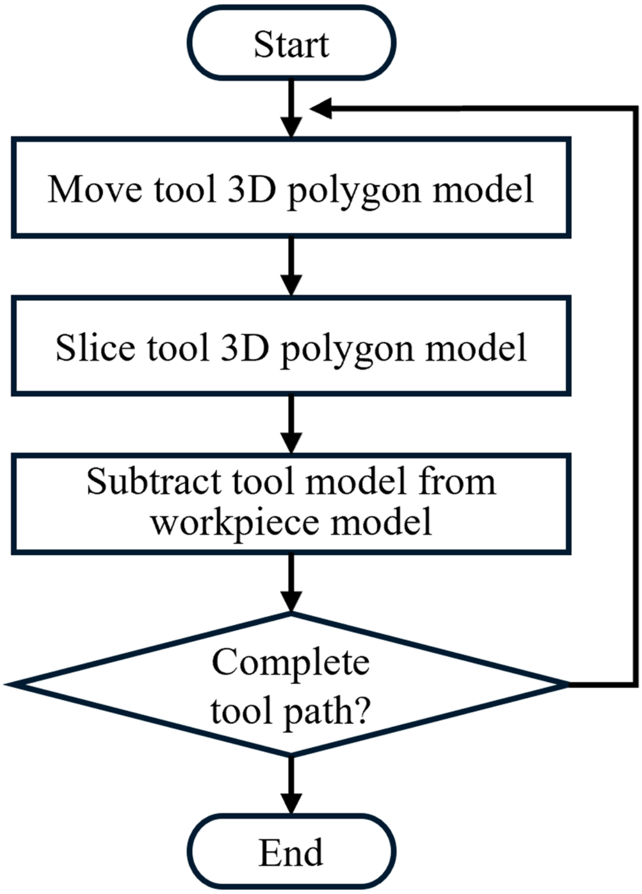

Start
Move tool 3D polygon model
Slice tool 3D polygon model
Subtract tool model from workpiece model
Complete tool path?
End

Fig. 8. Calculation flow of material-removal simulation.

performing tool-path generation, detection of unmachinable regions, and cutting-force estimation within a unified framework. As shown in Fig. 2, the 3D model was discretized by slicing it at constant intervals in three orthogonal directions (*xy*, *yz*, and *zx*). Each slice was converted into a set of 2D polygons, and the superposition of three polygonal datasets enabled the geometric representation of the complete 3D shape. If only *xy* − plane slices are used, the resolution along the *z*-direction is solely determined by the slice pitch; thus, achieving high accuracy requires an excessively fine pitch. By additionally slicing on the *yz*− and *zx*− planes, analogous to the rationale of tri-dexel representations, direction-dependent discretization errors are mitigated, and the need for extremely fine slicing in a single direction is reduced while maintaining geometric accuracy. In this study, polygon vertex coordinates are stored as floating-point values rounded to the fifth decimal place.

2.2. Offset shape calculation

Offsetting 3D geometries is an essential geometric operation in tool-path generation [28]. This study introduces a method for performing offsets on a 3D geometry, represented as an aggregation of 2D polygons. As illustrated in Fig. 3, each 2D cross-section sliced along the *zx*-plane can be inflated independently (positive offset). However, as shown in the figure, such a simple independent inflation of each cross-section does not coincide with the actual 3D inflated shape. This discrepancy occurs because the geometric relationships among adjacent cross-sections are neglected in the independent 2D offset process.

Consequently, a 3D offsetting method employing the convolutional processing of 2D polygon sets was proposed. This procedure is illustrated in Fig. 4. As shown in Fig. 4(a), N consecutive polygons (from 0 to N) are referenced within a region inflated with an offset value r around the model. First, the convolutional inflation process for the target cross section, p_i is incorporated. Within the reference range, all neighboring polygons p_j whose distances from p_i are smaller than r are detected, as shown in Fig. 4(b). Both p_i and each detected p_j were individually inflated using their respective offset values. The offset value r_j^i for section

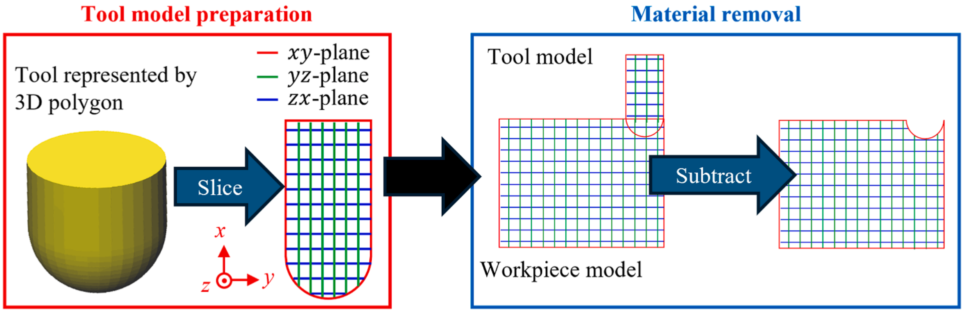
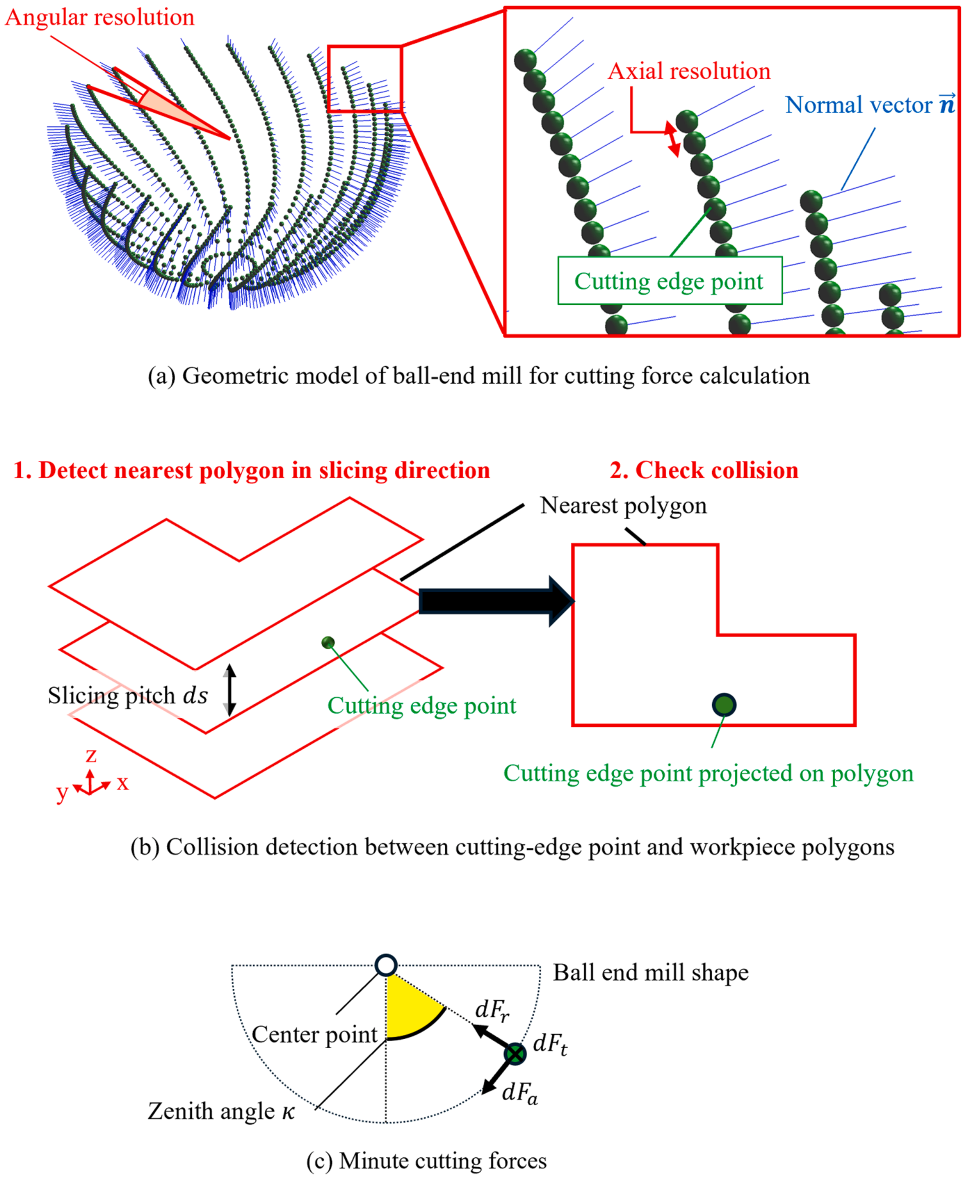


Fig. 9. Material removal process based on 2D polygon representation.

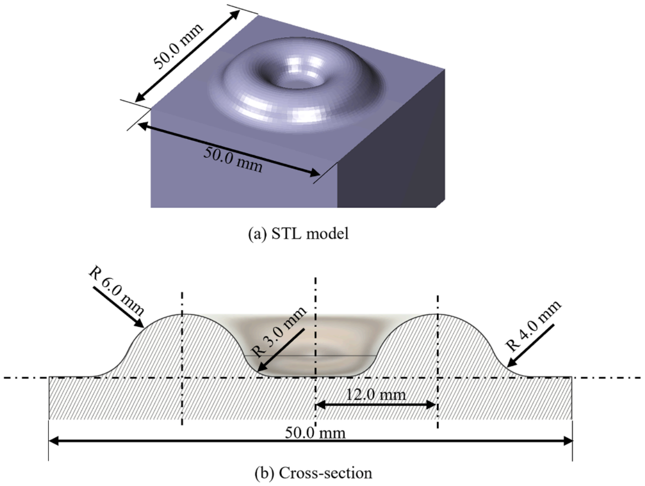


(a) Geometric model of ball-end mill for cutting force calculation

(b) Collision detection between cutting-edge point and workpiece polygons

(c) Minute cutting forces

Fig. 10. Cutting-edge point for cutting force calculation.



(a) STL model

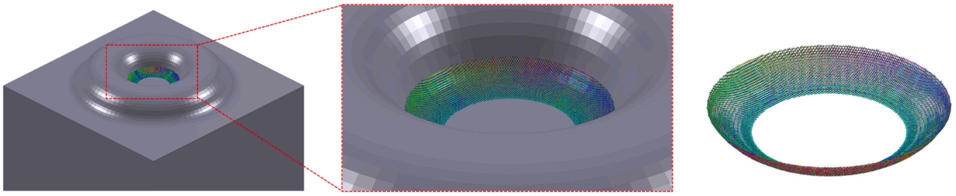
(b) Cross-section

Fig. 11. STL model for case study.

p_j relative to p_i was determined using Eq. (1):

$$r_j^i = \sqrt{r^2 - ((i-j)\cdot ds)^2}\cdot r_{sign} \tag{1}$$

where ds is the slicing pitch and r_{sign} denotes the sign of r (+1 for inflation and −1 for shrinkage). As illustrated in Fig. 4(e), this operation is equivalent to computing the inflated shape at the target polygon p_i when the entire geometry is inflated three-dimensionally by r. The Boolean union of inflated polygons completes the convolutional inflation process for polygon p_i. By performing this operation for all sections p_0–p_n within the convolutional range shown in Fig. 4(a), the 3D inflated geometry sliced along the yz-plane can be obtained. Accordingly, let n_r denote the number of polygons within the convolution range $(-r, +r)$ and n denote the total number of polygons to be offset. The computational complexity of the proposed method is $O(n_r \cdot n)$. Because the convolutional offsetting process for each section is independent, these computations can be parallelized across sections. The same procedure was applied to slices along the xy and zx planes. When the offset distance r is negative, a Boolean intersection is performed instead of a union to compute the contracted geometry.



(c) STL model and unmachinable area represented by 2D polygons

(d) Extracted area represented by 2D polygons

Fig. 12. Result of unmachinable area detection.

Table 1
Conditions for scanning path generation.

	Roughing	Finishing
Tool	φ8 Ball end mill	
Feed direction	One-way (y+)	
Radial depth of cut	2.0 mm	0.25 mm
Axial depth of cut	2.0 mm	0.10 mm
Scanning count	5	1
Margin	0.10 mm	0 mm

This process resulted in a 3D inflated geometry, as shown in Figs. 5 (e)–(h). For comparison, Figs. 6(a)–(d) show the results obtained by independently offsetting each cross section after slicing, as described in Fig. 3. Evidently, in this conventional approach, the inflated geometry varies depending on the slicing direction, indicating that an appropriate 3D inflated shape cannot be obtained. By contrast, the proposed convolutional offset method produced consistent inflated geometries irrespective of the slicing orientation, demonstrating its robustness in representing three-dimensional inflation. Based on this geometric framework, the following sections describe tool-path generation and detection of unmachinable regions.

3. Application

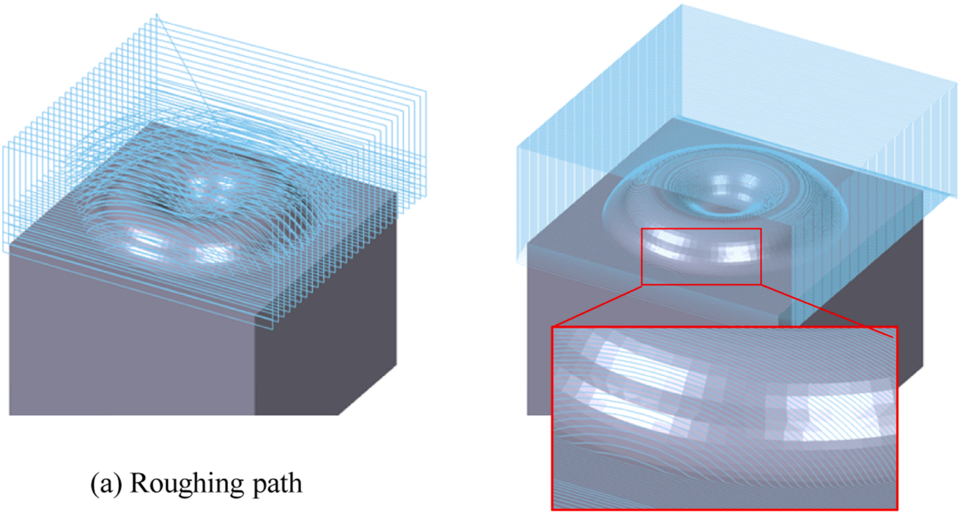
3.1. Tool path generation

When a 3D inflated geometry is obtained, deriving contour and scanning paths becomes straightforward. As shown in Fig. 6, when the 3D model was sliced on the *xy*-plane, the 2D-polygons obtained by inflating the 2D-polygons using the tool radius correspond to the contour tool paths. If multiple contour layers are required, the inflated *xy*-plane polygons can be further expanded by the radial depth of the cut, yielding additional 2D polygons that represent successive tool paths. When a machining allowance is specified, the inflation during the first offset operation can be increased by the allowance value relative to the tool radius.

The same principle applies to the scanning paths. A representative computation of *x*-direction scanning tool paths is illustrated in Fig. 6, where the model is sliced along the *zx*-plane and inflated to generate 2D-polygons corresponding to the scanning paths. Therefore, modeling the workpiece based on 2D-polygons simplifies the generation of both the contour and scanning tool paths for three-axis machining.

3.2. Detection of unmachinable regions

The inflation and shrinkage of 3D geometries can be used to evaluate the machinability of product shapes. Accordingly, for the three-directional 2D polygon aggregation model proposed in this study, the machinability can be similarly evaluated by performing convolutional inflation and shrinkage. An overview of this process is shown in Fig. 7. First, the 3D model of the product is sliced into multiple cross sections and represented as an aggregation of 2D polygons. Next, the inflated geometry is calculated using the tool radius as the offset value ***r***. Subsequently, the inflated geometry is reduced using the same tool radius. If the geometry obtained after shrinkage coincides with the original product shape, the product could be completely machined using the selected tool radius. However, as illustrated in Fig. 7, when differences appear between the original and processed shapes, the corresponding regions are identified as unmachinable. Therefore, by computing the



(a) Roughing path

(b) Finishing path

Fig. 13. Scanning path for machining torus shape.

Table 2
Comparison of computational time for tool path generation with different slicing pitch.

(a) Middle-range CPU (Intel Core i5 14600KF)

Slice pitch	Slice model	Roughing path generation	Finishing path generation	Total
0.250 mm	0.35 s	0.29 s	0.35 s	0.98 s
0.125 mm	0.44 s	0.43 s	0.93 s	1.79 s
0.050 mm	0.90 s	0.57 s	2.10 s	3.57 s
0.010 mm	3.88 s	1.77 s	12.86 s	18.51 s
0.005 mm	8.22 s	3.07 s	24.56 s	35.86 s

(b) High-end CPU (AMD Ryzen 9 9950×3D)

Slice pitch	Slice model	Roughing path generation	Finishing path generation	Total
0.250 mm	0.59 s	0.41 s	0.19 s	1.20 s
0.125 mm	0.50 s	0.43 s	0.59 s	1.52 s
0.050 mm	0.85 s	0.44 s	1.08 s	2.37 s
0.010 mm	4.18 s	1.05 s	6.49 s	11.72 s
0.005 mm	8.29 s	1.66 s	11.36 s	21.31 s

Table 3
Comparison of tool-paths generated by proposed method and commercial CAM.

Slicing pitch	Maximum difference	Within 1.0 µm	Within 2.0 µm	Within 5.0 µm
0.250 mm	52.00 µm	93.99 %	96.46 %	98.68 %
0.125 mm	19.91 µm	93.58 %	97.38 %	99.46 %
0.050 mm	8.75 µm	94.24 %	98.22 %	99.75 %
0.010 mm	9.28 µm	94.45 %	98.38 %	99.88 %
0.005 mm	8.25 µm	98.38 %	99.64 %	99.97 %

difference between the pre-inflation and post-shrinkage 2D polygons, unmachinable regions can be visualized. This functionality is useful in supporting the appropriate selection of tool diameters for CAM operations.

3.3. Material removal simulation and cutting force prediction

A material-removal simulation similar to that implemented in conventional CAM systems can be easily realized within the proposed 2D polygon-based framework. The overall procedure is illustrated in Fig. 8. In this simulation, the tool model is represented as a 3D polygon model, as shown in Fig. 9. First, the 3D polygon model of the tool is translated by a small incremental distance, denoted as Δd. Next, as illustrated in Fig. 9, the tool model is sliced along the *xy*, *yz*, and *zx* planes and converted into an aggregation of 2D polygons. The material removal process was simulated by computing the difference between the 2D polygon sets of the workpiece and tool. The machined geometry corresponding to the generated tool path can be simulated by repeating the operation along the path.

Beyond material removal simulation, the proposed framework was applied to a cutting-force analysis based on the instantaneous rigid force model [29]. Therefore, the tool is represented by the 3D polygon model shown in Fig. 9 and by a set of discrete cutting-edge points, as illustrated in Fig. 10(a). This representation discretizes the swept cutting-edge geometry along both the angular and axial directions of the tool. Each cutting edge point possesses its spatial coordinates as well as an associated normal vector, which defines its local cutting direction. Prior to the cutting-force analysis, intersection detection was performed between the workpiece geometry, which was represented as an aggregation of 2D polygons and cutting-edge points. An example of intersection detection in 2D polygons parallel to the *xy*-plane is shown in Fig. 10(b). First, the polygon is detected among the 2D polygons of the workpiece whose distance from a cutting edge point along the *z*-axis is within $ds/2$ and is closest to the point. If a polygon is obtained, the cutting edge point is projected onto the detected polygon, and an intersection is identified on the xy-plane if the projected point lies inside the polygon, an intersection is identified on the *xy*-plane. Intersection detection was performed in the *xy*, *yz*, and *zx* planes. If the cutting edge point intersects any of the three 2D polygon groups, it is determined to intersect with the

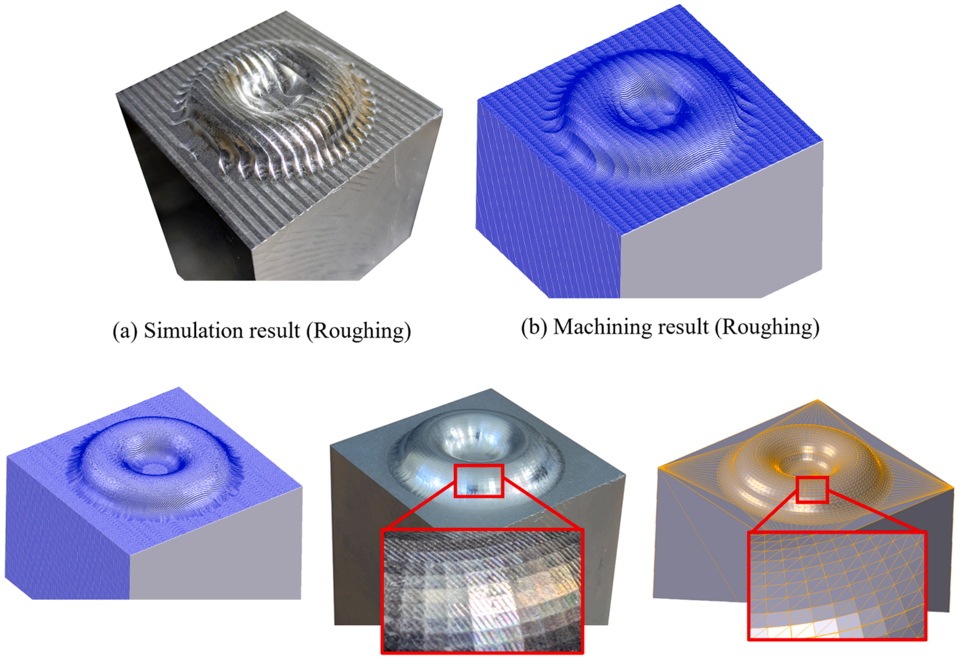
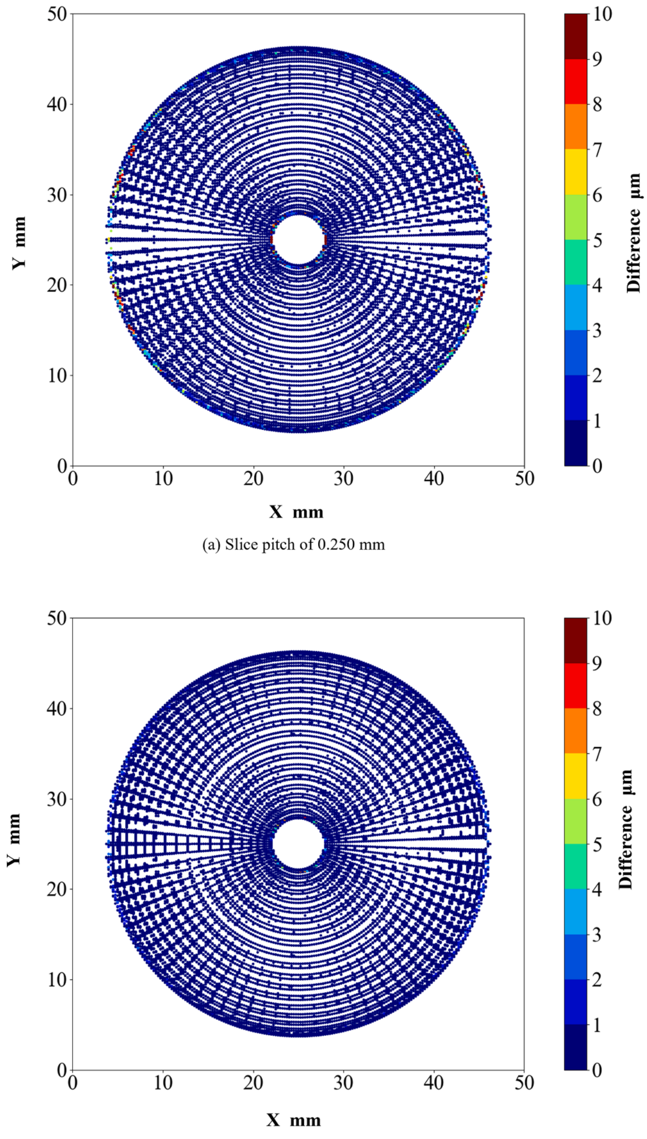

(a) Simulation result (Roughing)
(b) Machining result (Roughing)
(c) Simulation result (Finishing)
(d) Machining result (Finishing)
(e) Triangle mesh of STL model

Fig. 14. Actual and virtual workpiece shape after roughing and finishing path.



(a) Slice pitch of 0.250 mm

(b) Slice pitch of 0.005 mm

Fig. 15. Distribution of difference between tool paths generated by proposed method and commercial CAM.

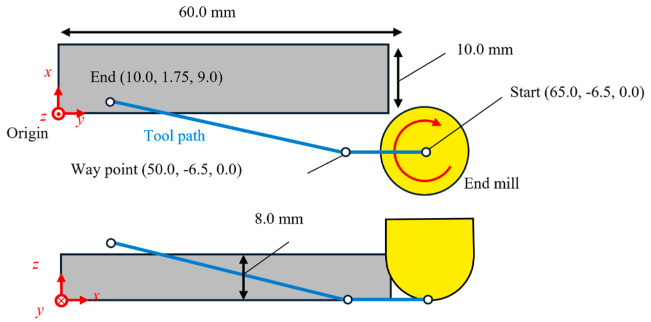


Fig. 16. Overview of experiment for cutting-force simulation.

Table 4
Conditions for cutting force analysis validation.

Machine tool	ROBODRILLα-D14MiB5 Plus	
Tool	Type	Ball end mill
	Diameter	8.0 mm
	Helix angle	30 °
Cutting conditions	Spindle speed	3600 rpm
	Feed rate	200 mm/min
Cutting coefficients	K_{tc}	935.62
	K_{te}	2.0218
	K_{rc}	416.73
	K_{re}	1.8238
	K_{ac}	2.72e-05
	K_{ae}	3.32e-06
Simulation conditions	Axial resolution	0.10 mm
	Radial resolution	1.0 deg.
	Slice pitch	0.10 mm

Table 5
Conditions for cutting coefficient identification.

	Test cut A	Test cut B	Test cut C
Spindle speed	3600 rpm		
Feed rate	100, 200, 300 mm/min		
Axial depth of cut	8.0	5.0	2.0
Radial depth of cut	1.5	3.0	7.0

workpiece geometry.

Subsequently, for each cutting-edge point intersecting the workpiece, the cutting forces were calculated based on an instantaneous cutting-force model. As illustrated in Fig. 10(c), the three infinitesimal force components acting on a single cutting-edge point— the tangential, radial, and axial components–denoted as dF_t, dF_r and dF_a— are computed using Eq. (2):

$$\begin{aligned} dF_t &= [K_{tc}\cdot h + K_{te}]\cdot dz/\sin\kappa \\ dF_r &= [K_{rc}\cdot h + K_{re}]\cdot dz/\sin\kappa \\ dF_a &= [K_{ac}\cdot h + K_{ae}]\cdot dz/\sin\kappa \end{aligned} \tag{2}$$

where K_{tc}, K_{te}, K_{rc}, K_{re}, K_{ac}, and K_{ae} are constant cutting coefficients. The uncut chip thickness, h is calculated geometrically using Eq. (3), based on the feed per tooth S_t, cutting-edge normal vector $\vec{\boldsymbol{n}}$, and feed-direction vector $\vec{\boldsymbol{f}}$:

$$h = S_t\cdot\left(\vec{\boldsymbol{n}}\cdot\vec{\boldsymbol{f}}\right) \tag{3}$$

The infinitesimal forces dF_t, dF_r, and dF_a are transformed into the workpiece coordinate system ($\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{z}$) using the rotation angle $\boldsymbol{\theta}$ and zenith angle $\boldsymbol{\kappa}$, as expressed in Eq. (4):

$$\begin{bmatrix} dF_x \\ dF_y \\ dF_z \end{bmatrix} = \begin{bmatrix} -\sin\theta & -\cos\theta\cdot\sin\kappa & -\cos\theta\cdot\cos\kappa \\ -\cos\theta & -\sin\theta\cdot\sin\kappa & -\sin\theta\cdot\cos\kappa \\ 0 & \cos\kappa & -\sin\kappa \end{bmatrix}\cdot\begin{bmatrix} dF_t \\ dF_r \\ dF_a \end{bmatrix} \tag{4}$$

The total cutting forces $\boldsymbol{F_x}$, $\boldsymbol{F_y}$, and $\boldsymbol{F_z}$ in the $\boldsymbol{x}$, $\boldsymbol{y}$, and $\boldsymbol{z}$ directions, respectively, are obtained by summing all infinitesimal forces, as shown in Eq. (5):

$$F_x = \sum dF_x,\ F_y = \sum dF_y,\ F_z = \sum dF_z \tag{5}$$

When performing a simulation that includes a cutting-force analysis, this calculation procedure is incorporated after each incremental tool movement, as shown in Fig. 8.

4. Validation

4.1. Geometric calculation

The proposed 2D polygon-based framework was validated to confirm that the extraction of unmachinable regions, tool-path generation, material removal simulation, and cutting-force analysis can be comprehensively performed within a unified system. The program was implemented in C# (. NET 8.0).

First, a case study was conducted to examine the validity of the

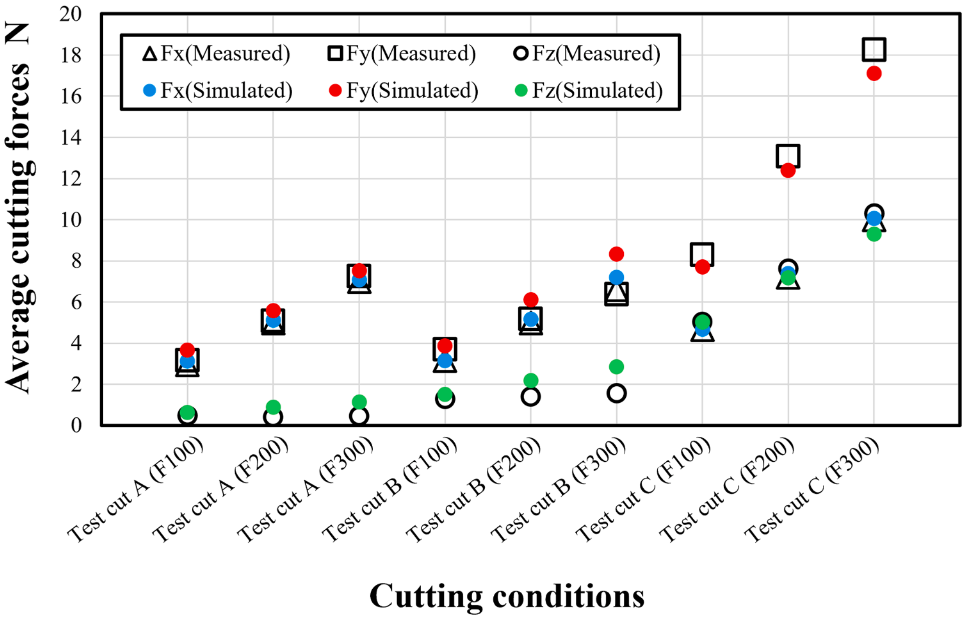


Fig. 17. Matching the average cutting forces for identification of cutting coefficients.

unmachinable-region extraction. The target geometry is shown in Fig. 11. This model comprised a torus with fillets where the outer and inner fillet radii differed. Machining was performed using a ball end mill with a radius of 4.0 mm. Because the outer fillet radius was 4.0 mm, the region was machinable, whereas the inner fillet radius was 3.0 mm, resulting in residual material after machining. Therefore, when inflation and shrinkage are performed using a tool radius of 4.0 mm and the difference from the original shape is computed, a set of 2D polygons corresponding to unmachinable regions is expected to be extracted near the inner fillet. The results of this geometric computation is shown in Fig. 12. As expected, the unmachinable regions were successfully extracted around the inner fillet. These results demonstrate that the proposed convolutional offset method can accurately compute 3D inflation and shrinkage of complex geometries.

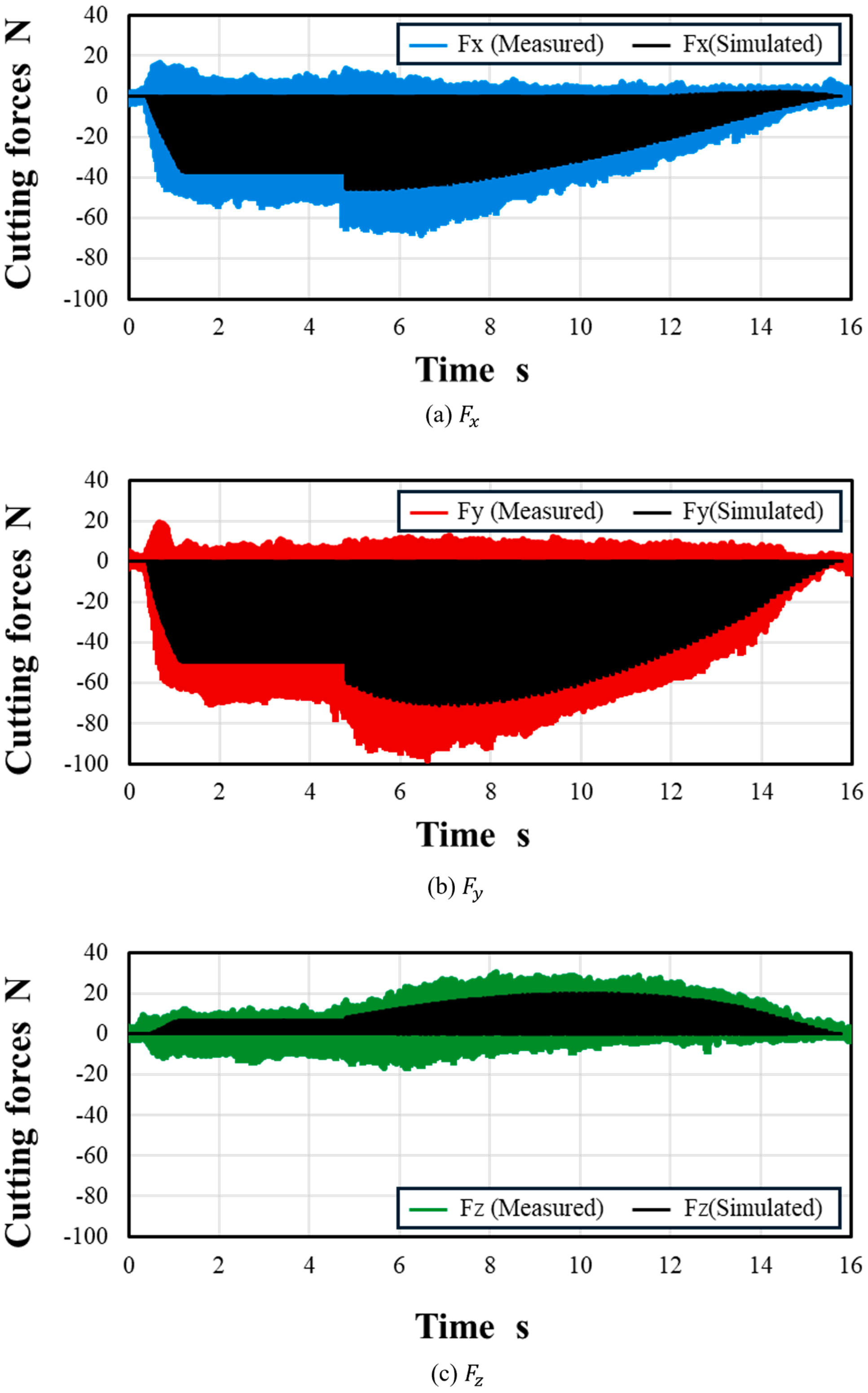
Subsequently, the proposed method was validated for tool-path generation. In this study, a scanning tool path was generated for the torus geometry, as shown in Fig. 11. The tool-path generation conditions are presented in Table 1. For the roughing operation, a machining allowance of 0.10 mm was applied, and five layers of scanning paths were generated by performing five successive inflation processes, as shown in Fig. 13. For the finishing operation, a machining allowance of 0 mm and radial depth of cut of 0.25 mm were applied, yielding a single finishing layer.

To evaluate the computational time and accuracy of the proposed tool-path generation method, the model was sliced into several slice pitches ranging from 0.005 to 0.25 mm, and the total computation time was measured using a middle-range CPU (Intel Core i5 14600KF) and a high-end one (AMD Ryzen 9 9950×3D). Tool-path generation was executed sequentially in three steps: model slicing, roughing-path generation, and finishing-path generation. The computation times for each process are presented in Table 2. The results show that the computation time increases as the slice pitch decreases, for all three processes. Among these, finishing-path generation tended to require the longest computation time. These results demonstrate that even with an extremely fine slice pitch of 0.005 mm, the total computation time is 35.86 s on the middle-range CPU and 21.31 s on the high-end CPU. This confirmed that the proposed CPU-based implementation can generate tool paths with practical efficiency without requiring GPU acceleration.

We also compared the tool paths with those generated by a commercial CAM software (Autodesk Fusion 360) while varying the slice pitch to validate the proposed method. The differences between the proposed method and commercial CAM results are summarized in Table 3. Here, the difference was defined as the shortest distance from each point comprising the tool path generated by the proposed method to the corresponding tool path generated by the commercial CAM, with a tolerance of 0.0005 mm. The distributions of these differences for slice pitches of 0.250 mm and 0.005 mm are shown in Fig. 14. As the slice pitch decreased, the mean difference between the tool paths generated by the proposed method and commercial CAM decreased. In contrast, the maximum difference converged to approximately 9 μm once the slice pitch became 0.050 mm or smaller. Regardless of the slice pitch, more than 90 % of the differences were within 1.0 μm. Fig. 14 also indicates that relatively large differences were localized near the torus edges. In the present validation, a slice pitch of 0.050 mm is sufficient to achieve close agreement with commercial CAM results: the maximum difference is within 10 μm, and more than 99 % of the differences are within 5.0 μm. This result demonstrates that the proposed 3D offset can calculate the offset shape correctly and generate tool paths comparable to those of the commercial CAM software.

In a previous study, it was reported that tool-path generation for a workpiece size of 60 × 50 mm in the *xy*-plane, discretized with a spatial resolution of 0.02 mm, required 3.3 s on a GPU (NVIDIA RTX 2070) [30]. Although it is not a fair one-to-one comparison because the machining conditions and evaluation settings are different, the proposed method generated tool paths that differ from those produced by a commercial CAM system by less than 10 μm with the slice pitch of 0.050 mm, and it achieved a computation time of 3.57 s even on a middle-range CPU. These results suggest that the proposed CPU-based implementation has the potential to achieve a performance comparable to that of GPU-based approaches.

The machining results obtained using the generated tool paths are shown in Fig. 15. Specifically, Figs. 15(a) and 15(b) show the rough-machined geometry in conjunction with the corresponding material removal simulation results. The simulated and actual machined geometries are consistent, validating that the proposed material-removal simulation accurately represent the machined shape. Figs. 15(c) and 15(d) show a comparison after the completion of the operation. The machined surface faithfully reproduced the fine triangular features that constituted the STL model of the torus. These results demonstrate the validity of both tool-path generation and material removal simulations implemented in the proposed framework.



(a) F_x

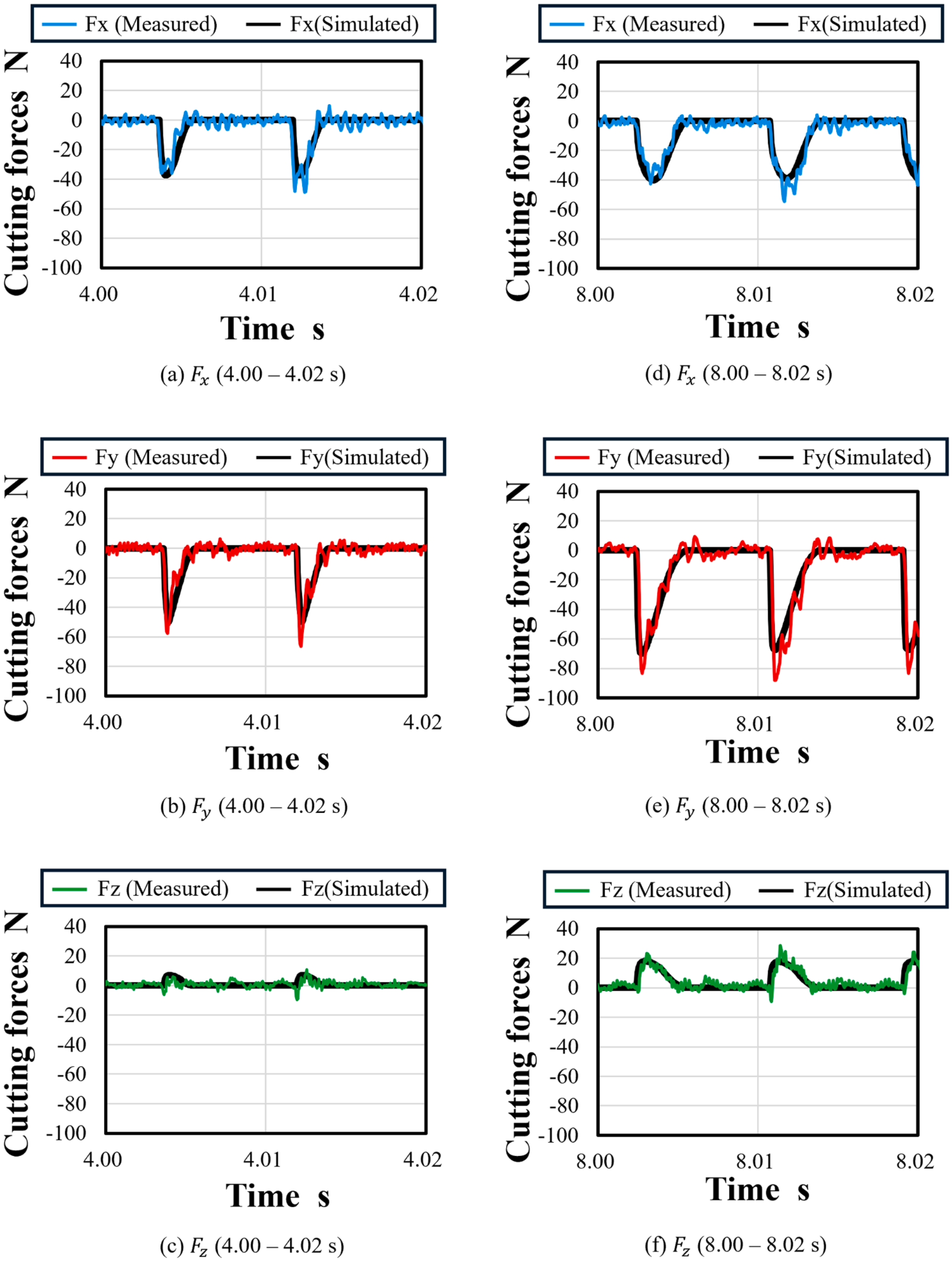
(b) F_y

(c) F_z

Fig. 18. Comparison of entire cutting forces.

4.2. *Cutting force calculation*

The validity of the cutting force simulation was verified. The experimental setup and cutting conditions used for the validation are shown in Fig. 16 and Table 4, respectively. The six cutting coefficients were identified using several simple test cuts. The test cutting conditions are listed in Table 5. The test cuts were performed before the cutting experiment under nine conditions, comprising three feed rates and three combinations of cut depths. As shown in Fig. 17, the cutting coefficients were identified by an iterative search, such that the simulated average cutting forces matched the measured values for each test. In the experiment, machining was performed along a linear tool path using a ball-end mill. Up to the waypoint, the cutting depth remains constant, indicating a steady-state cutting region. Beyond this point, the



(a) F_x (4.00 – 4.02 s)

(d) F_x (8.00 – 8.02 s)

(b) F_y (4.00 – 4.02 s)

(e) F_y (8.00 – 8.02 s)

(c) F_z (4.00 – 4.02 s)

(f) F_z (8.00 – 8.02 s)

Fig. 19. Comparison of cutting-force waveforms.

Table 6
Error analysis of cutting-force prediction.

Duration	Mean errors			Maximum errors		
	F_x	F_y	F_z	F_x	F_y	F_z
4.00–4.02 s	5.81 N	7.10 N	3.58 N	18.63 N	21.53 N	14.90 N
8.00–8.02 s	5.02 N	7.58 N	2.94 N	16.22 N	23.85 N	16.55 N

Cutter–Workpiece Engagement (CWE) varies dynamically and represents a transient cutting region. Cutting forces were measured and simulated along the tool path under steady and transient cutting conditions. A comparison of the measured and simulated cutting forces is presented in Fig. 18.

The variations in the cutting forces in the x-, y-, and z-directions (F_x, F_y, and F_z) were found to be in good agreement between the simulation and experimental measurements, as shown in Fig. 18. However, the measured forces appear to be slightly larger than the simulated values. To investigate this difference in detail, a waveform comparison was conducted, as shown in Fig. 19, in which the cutting-force waveforms at 4.0 s and 8.0 s were compared. The results indicate that the instantaneous waveforms of the simulation were closely aligned with those of the experiment. The measured forces differed slightly from the simulated results. To examine this deviation quantitatively, we evaluated the errors in two short time windows, as shown in Fig. 19. As summarized in Table 6, the mean errors were of the order of a few newtons ($\leq$ 5.81 N for F_x, $\leq$ 7.58 N for F_y, and $\leq$ 3.58 N for F_z), while the maximum errors were below 23.85 N. This residual bias is likely attributed to factors not considered in the instantaneous rigid-force model, such as tool runout and dynamic changes in the uncut chip thickness due to vibration. The simulated waveforms still reproduced temporal variations of the measured forces with good fidelity.

4.3. Limitations

These results demonstrate that the proposed method enables an integrated workflow encompassing tool-path generation, material removal simulation, unmachinable-region extraction, and cutting-force analysis within a unified computational framework. However, this study has some limitations. The geometric accuracy of the tool paths generated by the proposed method depends on the slice pitch. Therefore, sufficiently fine slicing is required to achieve high accuracy. In the current implementation, the scope is limited to tool-path generation for three-axis machining using a ball-end mill, specifically, contour-parallel and scanline paths. Consequently, extending the framework to other tool types (e.g., bullnose- and barrel-end mills) and simultaneous five-axis tool paths remain an important topic for future work. The constant slice pitch is also a limitation. To further improve computational efficiency, the slice pitch should ideally be adjusted adaptively according to the local geometric complexity of the 3D model.

Regarding process simulation, the present study is currently limited to cutting-force estimation. However, in our previous studies, we established a machining-accuracy simulation method that accounted for force-induced elastic deformation based on cutting forces predicted by the instantaneous rigid-force model [31,32]. Thus, in future work, we plan to integrate these methods into the proposed framework to consider the machining accuracy toward a more practical and comprehensive simulation capability.

5. Conclusion

In this study, we proposed a unified 2D polygon-based CAM framework that integrates execution of tool-path generation, material removal simulation, unmachinable-region extraction, and cutting-force prediction.

The primary findings of this study can be summarized as follows:

1. A novel convolutional offset algorithm was proposed for three-dimensional geometries represented by the aggregations of 2D polygons. This method accurately performs inflation and shrinkage by incorporating the adjacent cross-sectional relationships, achieving consistent 3D offsets independent of the slicing direction.
2. The inflated 2D polygons obtained via the proposed offset method can be utilized directly as tool paths, enabling efficient path generation. When the slice pitch was set to 0.050 mm, the maximum difference from the commercial CAM result was 8.75 μm, and the computation time was only 3.57 s even on a middle-range CPU.
3. By performing sequential inflation and shrinkage operations using a particular tool radius, the unmachinable regions can be visualized, thereby supporting optimal tool selection and process planning.
4. The proposed geometric framework was successfully integrated with an instantaneous cutting force model, enabling cutting force predictions consistent with experimental measurements.

The proposed system was implemented in C# (. NET 8.0) and validated via machining experiments. The results confirmed that tool paths can be generated and simulated in less than one minute even without GPU acceleration, demonstrating both the accuracy and computational efficiency of the method. In future work, the present framework will be extended to incorporate machining error analysis previously developed by the authors, thereby achieving a fully integrated analysis from tool path generation to machining accuracy. Furthermore, the authors aim to establish an AI-driven optimization system that automatically adjusts tool paths and cutting conditions based on dynamic and mechanical simulation results.

Declaration of generative AI in scientific writing

Statement: During the preparation of this work the authors used ChatGPT and Deep L in order to improve the clarity and grammar of the English text. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the published article.

CRediT authorship contribution statement

Kazuki Kaneko: Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Hiroto Takayasu:** Software, Methodology, Investigation. **Atsuya Suzuki:** Validation, Software, Methodology, Investigation. **Hiroyuki Kodama:** Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was supported in part by JSPS KAKENHI (grant number 25K17518).

Data availability

Data will be made available on request.

References

[1] K. Matsukawa, H. Nakatsuji, I. Nishida, Automated tool-path generation for complex shapes applicable to 5-axis indexing machining using STL-format CAD models, Int. J. Autom. Technol. 19 (2025) 698–711, https://doi.org/10.20965/ijat.2025.p0698.

[2] M. Dhanda, A. Kukreja, S. Pande, Adaptive spiral tool path generation for computer numerical control machining using point cloud, Proc. Inst. Mech. Eng. C. J. Mech. Eng. Sci. 235 (2021) 6240–6256, https://doi.org/10.1177/0954406221990077.

[3] F. Lu, G. Zhou, C. Zhang, Y. Liu, F. Chang, Q. Lu, Z. Xiao, Energy-efficient tool path generation and expansion optimisation for five-axis flank milling with meta-reinforcement learning, J. Intell Manuf. 36 (2025) 3817–3841, https://doi.org/10.1007/s10845-024-02412-4.

[4] A. Kukreja, S.S. Pande, Optimal toolpath planning strategy prediction using machine learning technique, Eng. Appl. Artif. Intell. 123 (2023) 106464, https://doi.org/10.1016/j.engappai.2023.106464.

[5] X. Zhang, A. Nassehi, S.T. Newman, Feature recognition from CNC part programs for milling operations, Int. J. Adv. Manuf. Technol. 70 (2014) 397–412, https://doi.org/10.1007/s00170-013-5275-4.

[6] P. Wang, W.-A. Yang, Y. You, A hybrid learning framework for manufacturing feature recognition using graph neural networks, J. Manuf. Process 85 (2023) 387–404, https://doi.org/10.1016/j.jmapro.2022.10.075.

[7] J. Zaragoza Chichell, A. Rečková, M. Bizzarri, M. Bartoň, Collision-free tool motion planning for 5-axis CNC machining with toroidal cutters, Comput.-Aided Des. 173 (2024) 103725, https://doi.org/10.1016/j.cad.2024.103725.

[8] M. Inui, N. Umezu, Fast computation of collision-free cutter postures using discrete representation of workpiece shape, in: 44th Computers and Information in Engineering Conference (CIE) Volume 2A, American Society of Mechanical Engineers, 2024, https://doi.org/10.1115/DETC2024-141766.

[9] G.P. Harabin, M. Behandish, Hybrid manufacturing process planning for arbitrary part and tool shapes, Comput.-Aided Des. 151 (2022) 103299, https://doi.org/10.1016/j.cad.2022.103299.

[10] K. Takasugi, N. Asakawa, Parameter-based spiral tool path generation for free-form surface machining, Precis. Eng. 52 (2018) 370–379, https://doi.org/10.1016/j.precisioneng.2018.01.013.

[11] R. Lynn, D. Contis, M. Hossain, N. Huang, T. Tucker, T. Kurfess, Voxel model surface offsetting for computer-aided manufacturing using virtualized high-performance computing, J. Manuf. Syst. 43 (2017) 296–304, https://doi.org/10.1016/j.jmsy.2016.12.005.

[12] N.T. Anh, T.T. Tung, Cutting force prediction in end milling processes: Analytical models and applications, Appl. Eng. Sci. 23 (2025) 100250, https://doi.org/10.1016/j.apples.2025.100250.

[13] I. Nishida, R. Okumura, R. Sato, K. Shirase, Cutting force simulation in minute time resolution for ball end milling under various tool posture, J. Manuf. Sci. Eng. (2018) 140, https://doi.org/10.1115/1.4038499.

[14] H. Takayasu, K. Kaneko, J. Shimizu, Voxel-based method for predicting workpiece chipping in end milling of unsintered pure iron-powder compact, Results Eng. 27 (2025) 106656, https://doi.org/10.1016/j.rineng.2025.106656.

[15] M. Bäker, Finite element simulation of high-speed cutting forces, J. Mater. Process Technol. 176 (2006) 117–126, https://doi.org/10.1016/j.jmatprotec.2006.02.019.

[16] M. Liu, H. Xie, W. Pan, S. Ding, G. Li, Prediction of cutting force via machine learning: state of the art, challenges and potentials, J. Intell. Manuf. 36 (2025) 703–764, https://doi.org/10.1007/s10845-023-02260-8.

[17] A. Ebrahimi Araghizad, F. Pashmforoush, F. Tehranizadeh, K. Kilic, E. Budak, Improving milling force predictions: a hybrid approach integrating physics-based simulation and machine learning for remarkable accuracy across diverse unseen materials and tool types, J. Manuf. Process 114 (2024) 92–107, https://doi.org/10.1016/j.jmapro.2024.02.001.

[18] Y. Altintas, G. Stepan, E. Budak, T. Schmitz, Z.M. Kilic, Chatter stability of machining operations, J. Manuf. Sci. Eng. 142 (2020), https://doi.org/10.1115/1.4047391.

[19] B. Denkena, O. Pape, T. Grove, A. Mücke, Advanced process design for re-contouring using a time-domain dynamic material removal simulation, Procedia CIRP 79 (2019) 21–26, https://doi.org/10.1016/j.procir.2019.02.005.

[20] B. Denkena, O. Pape, A. Krödel, V. Böß, L. Ellersiek, A. Mücke, Process design for 5-axis ball end milling using a real-time capable dynamic material removal simulation, Prod. Eng. 15 (2021) 89–95, https://doi.org/10.1007/s11740-020-01003-5.

[21] R. Wang, S. Zhang, I. Ullah, M. Wiercigroch, Quasistatic deflection analysis of slender ball-end milling cutter, Int. J. Mech. Sci. 264 (2024) 108807, https://doi.org/10.1016/j.ijmecsci.2023.108807.

[22] K. Kaneko, J. Shimizu, K. Shirase, A voxel-based end milling simulation method to analyze the elastic deformation of a workpiece, J. Manuf. Sci. Eng. 145 (2023), https://doi.org/10.1115/1.4055794.

[23] J.-Y. Oh, W. Lee, Model-based cutting load prediction and feed rate optimization considering cutting conditions and tool wear, Manuf. Lett. 44 (2025) 594–601, https://doi.org/10.1016/j.mfglet.2025.06.070.

[24] K. Kaneko, T. Komatsu, L. Zhou, T. Obuki, H. Ojima, J. Shimizu, Autonomous optimization of cutting conditions in end milling operation based on deep reinforcement learning (Offline training in simulation environment for feed rate optimization), J. Adv. Mech. Des. Syst. Manuf. 17 (2023) 2023, https://doi.org/10.1299/jamdsm.2023jamdsm0064, jamdsm0064.

[25] M. Pavlidakis, C. Kitching, N. Tomlinson, M. Søndergaard, Cross-Vendor GPU Programming: Extending CUDA Beyond NVIDIA, in: Proceedings of the 4th Workshop on Heterogeneous Composable and Disaggregated Systems, ACM, New York, NY, USA, 2025, pp. 45–51, https://doi.org/10.1145/3723851.3723860.

[26] Angus Johnson, Clipper2 – polygon clipping and offsetting library, Https://Www.Angusj.Com/Clipper2/Docs/Overview.Htm (n.d.).

[27] M. Held, S. de Lorenzo, P. Palfrader, An experimental evaluation of offset computation for polygons, Comput. Aided Appl. (2024) 807–818, https://doi.org/10.14733/cadaps.2024.807-818.

[28] M. Inui, N. Umezu, M. Tsukahara, Simple offset algorithm for generating workpiece solid model for milling simulation, J. Adv. Mech. Des. Syst. Manuf. 11 (2017), https://doi.org/10.1299/jamdsm.2017jamdsm0042. JAMDSM0042–JAMDSM0042.

[29] J. Tlusty, P. MacNeil, Dynamics of cutting forces in end milling, Ann. CIRP 17 (1975) 21–25.

[30] A. Kukreja, M. Dhanda, S.S. Pande, Voxel-based adaptive toolpath planning using graphics processing unit for freeform surface machining, J. Manuf. Sci. Eng. 144 (2022), https://doi.org/10.1115/1.4051535.

[31] K. Kaneko, M. Inui, I. Nishida, Fast simulation of machining error induced by elastic deformation of tool system in end milling, J. Adv. Mech. Des. Syst. Manuf. 17 (2023), https://doi.org/10.1299/jamdsm.2023jamdsm0035, 2023jamdsm0035.

[32] K. Kaneko, J. Shimizu, K. Shirase, Voxel-based end milling simulation of machining error induced by elastic deformation of tool and workpiece, J. Adv. Mech. Des. Syst. Manuf. 18 (2024), https://doi.org/10.1299/jamdsm.2024jamdsm0042, 2024jamdsm0042.