

# Feature-based Recognition of CNC Processes and Automatic Program Generation

Wen-Ya Shi<sup>1,2\*</sup>, Sheng-Wu Kong<sup>1</sup>, Feng Yang<sup>1</sup>, Hu-Li Hua<sup>1</sup>  
Xiang-Yun Yi<sup>1</sup>, and Yong Zhang<sup>1</sup>

<sup>1</sup> Department of Mechanical Engineering, Hebei Institute of Mechanical and Electrical Technology,  
Xingtai City 054000, Hebei Province, China

18903199996@163.com, hbjdksw@163.com, yangfeng5978@163.com,  
hulihua890210@163.com, xiangyun7987@126.com, ysujsxr1982@163.com

<sup>2</sup> Xingtai Advanced Rolling Intelligent Equipment Technology Innovation Center,  
Xingtai City 054000, Hebei Province, China

*Received 2 November 2025; Revised 4 December 2025; Accepted 12 December 2025*

**Abstract.** This paper presents a novel approach to Computer numerical control (CNC) process identification and program generation through automatic identification of features from computer-aided design (CAD) models. The system extracts geometric and topological features such as holes, pockets, and bosses from boundary representation data and accurately identifies them as machining operations such as milling, drilling, and turning. A rule-based classifier allocates these characteristics to their corresponding processes and an automated code generator produces machine-specific CNC programs through parametric templates and post-processing. A no manual programming framework integrates, significantly reduces human error, and enhances efficiency in CNC manufacturing processes. Experimental verification on six industrially relevant mechanical components confirms robustness of the system and attains high accuracy in feature recognition and process classification. Apart from this, the approach reduces programming time by over 60% compared to conventional manual coding, which proves its feasibility for use in real-world digital manufacturing environments. The results show that the proposed feature-based approach offers an intelligent and scalable solution for CAD-to-CNC automation.

**Keywords:** feature recognition, CNC process automation, CAD-to-CNC integration, program generation

## 1 Introduction

Computer numerical control (CNC) machining is now a foundation of contemporary manufacturing because it can make parts with high speed, high repeatability, and high precision [1]. Nevertheless, one major hurdle in digital manufacturing is the process of converting computer-aided design (CAD) models into executable CNC programs. This conversion process is typically a labor-intensive process that relies heavily on the expertise and knowledge of CNC programmers to interpret part geometry, select appropriate machining operations, and generate G-code suitable for a particular machine tool and control system. This is a labor-intensive, error-prone, and difficult-to-scale-by-hand process, especially in an application where manufacturing needs to be flexible and have quick turnaround. Due to these issues, feature-based recognition has emerged as a promising solution to automatic CNC process planning. Feature-based recognition involves using the geometry and topology of the CAD model to discover manufacturing-critical features such as holes, slots, cavities, and bosses. The extracted features are then employed for finding the corresponding machining operations, on which automatic tool-path creation and CNC programming are dependent. Despite the potential in this direction, there are still some key challenges to real-world implementation, including robust feature extraction in complexity geometries, robust process classification under varying conditions, and the generation of accurate, machine-specific CNC code. This work aims to present an integrated approach addressing these limitations directly through a three-stage automated system: feature recognition, CNC process classification, and program generation. The proposed feature extraction algorithm filters boundary representation (B-rep) data to identify primary manufacturing features and describe them as parametric descriptors [2]. These features are then categorized by a decision tree-based system to identify

---

\* Corresponding Author

appropriate machining processes based on geometric parameters, spatial orientation, and tool considerations. Automatic CNC code generation through the utilization of pre-defined templates and post-processors is the last step towards maintaining compatibility with specific machine controllers. By integrating all such stages into a single pipeline, end-to-end automation of the CAD-to-CNC conversion is accomplished by the current approach. Experimental results show that the system can reduce the programming time, improve process reliability, and obtain high recognition accuracy and classification accuracy of features and processes, respectively. This work adds to the discipline of smart manufacturing a scalable and realistic solution for automatic program generation through feature recognition, significantly improving the autonomy and efficiency of CNC process planning.

Sections 1 through 6 collectively present a complete CAD-to-CNC automation solution. Section 1 outlines the goal of reducing manual programming workload through a feature-based framework; Section 2 reviews previous work on feature recognition, intelligent planning, and NC code development, highlighting the lack of a fully integrated automation solution; Section 3 outlines the proposed method, which integrates B-rep feature extraction, rule-guided CNC machining process identification, and template-driven NC code generation into a consistent workflow; Section 4 describes the experimental setup, including reference CAD models, expert annotation, hardware and software configuration, and rigorous evaluation protocols; Section 5 describes simulation and machining results, demonstrating higher feature and machining accuracy, significantly reduced programming time, and a lower error rate compared to manual expert programming; Section 6 concludes that the framework successfully integrates design and manufacturing while adapting to future expansions, such as more complex features, multi-axis machining, adaptive control, and scalable cloud deployment.

## 2 Related Work

In feature-based process planning, the process of manufacturing generally begins with a CAD model of the final part, the blank geometry, and the set of machine tools. One of the most difficult parts of this process is identifying the geometric features and relating them to respective machining operations correctly with reliability, which is crucial for achieving automation in CNC programming. Chandrasegaran et al. [3] gave a milestone overview of feature-based manufacturing technologies, summarizing various recognition methods such as B-Rep, volume recognition using convex decomposition, and volume features defined for process planning. Their work emphasized the necessity of precise feature identification in making automation possible in modeling and NC programming. Subsequent research by Mo et al. [4] focused on automating the translation of identified features into executable CNC code, in the general direction of reducing human programming effort and enhancing process efficiency. Intelligent algorithms have subsequently been incorporated to improve feature recognition and process planning accuracy. As an example, Yeo et al. [5] used artificial neural networks (ANNs) to identify prismatic machining features from B-Rep CAD models directly and generate feature vectors to be used for classification and firmer feature-process linkage. Optimization techniques have also made significant contribution to process planning. Oysu and Bingulutilized [6] genetic algorithms (GAs) to optimize machining operation sequences, for example, drilling paths, to minimize tool travel distance and machining time. Vision-based techniques have also been incorporated into trajectory planning. Yao et al. [7] developed a snakes-and-ladders-inspired algorithm for generating safe and efficient tool paths in dual-axis CNC machines, demonstrating how intelligent planning could be extended to multi-axis systems. In micro-scale manufacturing, automatic NC code generation and feature extraction techniques specifically for micro-features, showcasing the scalability of feature-based recognition methods. Swarm intelligence-based algorithms have also been employed to optimize advanced CNC programs. Al-Shaery et al. [8] employed particle swarm optimization (PSO) and artificial bee colony (ABC) algorithms to improve CNC turning program generation, addressing the limitations of traditional heuristic-based CAM systems. Finally, advances in machine learning recently extended beyond feature recognition to process monitoring. Jia et al. [9] proposed a Siamese Region Proposal Network (RPN)-based approach for detecting the working status of CNC machines through autonomous robotic systems. This is just one piece of the growing trend towards intelligent process monitoring in real-time as part of the broader paradigm of automated CNC programming. Together, these papers bring the discipline nearer to an entirely automated chain incorporating feature recognition, process identification, and CNC code generation with reduced involvement of humans and more efficient and reliable existing manufacturing systems. However, despite the broad scope of existing research, several key gaps remain unresolved. First, previous studies have largely treated feature recognition, process classification, and CNC code generation as independent modules, lacking a unified end-to-end automation framework. This fails to ensure data consistency throughout the process, hindering the realization

of design intent. Second, existing recognition and classification methods are often tested with complex multi-surface or intersecting features, and few methods offer robust performance across a wide range of industrial geometries under real-world production conditions. Third, optimization methods such as genetic algorithms (GA), particle swarm optimization (PSO), or artificial neural networks (ANN) typically focus on isolated tasks toolpath optimization, feature classification, or condition monitoring rather than integrating these components into a unified system specific to the machine and controller. Finally, practical validation remains limited, with most studies relying on synthetic or simplified datasets rather than full-scale machining tests on industrial CNC equipment. To bridge these gaps, this study develops a fully integrated, feature-driven CAD-CNC automation framework that combines feature extraction, process recognition, and validation procedure generation, providing a practical and scalable solution that surpasses the fragmented approaches found in previous research.

### 3 Methodology

The envisioned system ingests native CAD files or neutral B-Rep files and drives a unified, feature-based CNC automation process consisting of three tightly coordinated yet loosely coupled phases: feature extraction, process identification, and program generation in Fig. 1. This process is based on geometric reasoning, rule-based classification, and template-driven code generation. The feature extraction and identification phase begins by normalizing the geometry using a B-Rep description. Next, a geometric/topological evaluation is performed to detect primitive and composite features, including planes, cylinders, cones, holes, bosses, grooves, notches, chamfers, fillets, and threads. This is done using analytical fitting, RANSAC for free-form approximation, and a sub-graph isomorphism algorithm based on face adjacency, with cross-validation in the presence of PMI. Each detected feature is encoded in a controller-independent record capture type, along with the part's pose, critical dimensions, tolerances/surface finish, preferred machining surface, priority/interaction labels, and uncertainty/quality flags to facilitate downstream control decisions or request review. Next, CNC process identification analyzes feature attributes to infer process primitives as well as work piece clamping and setup orientation. A rules-plus-cost classifier maps each feature to a tool family and strategy, while checking spindle limits, reach, collision envelopes, and setup grouping to minimize tool changes and re-clamping. Meanwhile, the manufacturability constraint module enforces DFM limits, raises actionable exceptions when conflicts arise, and prioritizes operations. Finally, automated CNC program generation is proposed through: (i) a parametric template application that links validated operation templates to feature parameter, material, and tool libraries, automatically calculating feeds/speeds from a database or model and correcting for wear, overhang, and coolant; (ii) motion command construction that uses WCS/datum components to implement safe trajectories, tool length/radius compensation, optimized links, lead-in/out arcs, remaining machining boundaries, and in-process probing, and uses conditional logic to accommodate tolerance results; (iii) a post-processor that translates to a specific controller and generates compatible G/M code, macros, and canned cycles, as well as setup tables, tool lists, and cycle time estimates; and (iv) automated background drawing and lightweight collision check gates for stock/fixture and template acceptance criteria, ensuring that only validated programs become production-ready CNC code. Throughout the process, modules communicate through stable data contracts, preserving design intent and are designed to operate independently and provide consistent outputs, enabling full automation from digital design to shop floor execution while still supporting human interaction at critical points of failure. The modular design also allows for straightforward upgrades without disrupting upstream logic, ensuring end-to-end accuracy, traceability, and maintainability.

#### 3.1 Feature Extraction and Recognition

Feature extraction is initiated by depicting the geometric and topological form of the CAD model through B-rep that characterizes a solid through its bounding surfaces [10]. This representation is implemented by the majority of commercial CAD systems and is highly suitable for close scrutiny of faces, edges, and vertices. The algorithm targets a list of machining-pertinent features like holes, slots, pockets, bosses, and chamfers, each of which equates to a group of manufacturing operations. Each face in the B-rep is evaluated for curvature, normal vectors, and edge adjacency. To detect cylindrical holes, the system computes the principal curvature of a face, where:

$$k_{1,2} = H \pm \sqrt{H^2 - K} \quad (1)$$

$$R_{1,2} = \frac{1}{|k_{1,2}|} \quad (2)$$

Here,  $R$  is the radius of curvature derived from the Gaussian curvature  $K$  and mean curvature  $H$ . A face is classified as cylindrical if:

$$|K| \leq \varepsilon_K \quad (3)$$

$$Var_{(u,v) \in f} (\max \{|k_1|, |k_2|\}) \leq \varepsilon_{var} \quad (4)$$

Additionally, topological relationships are assessed using an adjacency matrix, where:

$$A_{ij} = 1 \Leftrightarrow (e_i \cap e_j \neq \emptyset) \wedge (n_i \cdot n_j \leq \varepsilon) \quad (5)$$

This matrix supports loop detection, boundary identification, and feature closure verification. Once detected, each feature is encoded as a parametric descriptor.

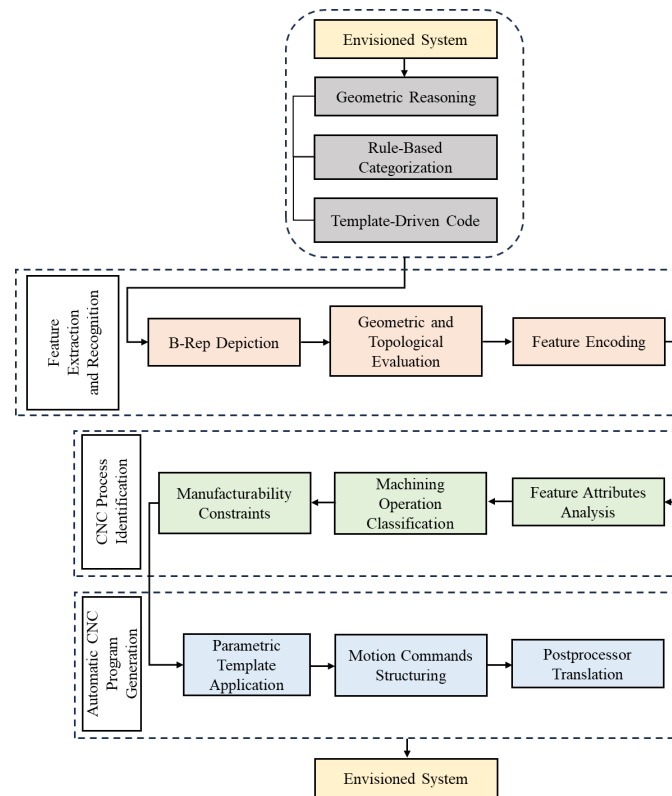


Fig. 1. A flowchart in the digital diagram

### 3.2 CNC Process Identification

The CNC Process Identification module is tasked with bridging the gap between manufacturing intent and geometric comprehension, as noted by [11]. Once features are extracted and identified, each feature described by at-

tributes such as type, size, shape, and spatial location, is evaluated to determine the most appropriate machining operation. This is achieved using a decision tree classifier  $T$ , which maps each feature based on a set of attributes:

$$F_i = \{type, size, depth, orientation, tool - accessibility, symmetry, machine - capability\} \quad (6)$$

The classifier makes its prediction by computing the maximum posterior probability:

$$T(F_i) = \arg \max_{o_j \in O} P(o_j | F_i) \quad (7)$$

Where  $o$  represents the set of machining operations such as drilling, milling, or turning. For example, vertical through holes are mapped to drilling, orthogonal side wall planar cavities to milling, and radially symmetric bosses to turning operations [12]. The classifier is further refined by integrating essential manufacturability constraints to ensure both technological and economic viability. These include the minimum tool diameter constraint

$$d_{tool}^{\min} \leq w_f \quad (8)$$

Which ensures the tool fits within the feature; the maximum cutting depth constraint

$$d_f \leq d_{cut}^{\max} \quad (9)$$

To prevent tool breakage and overheating; and the approach angle constraint

$$\theta_{approach} \in \Theta_{permitted} \quad (10)$$

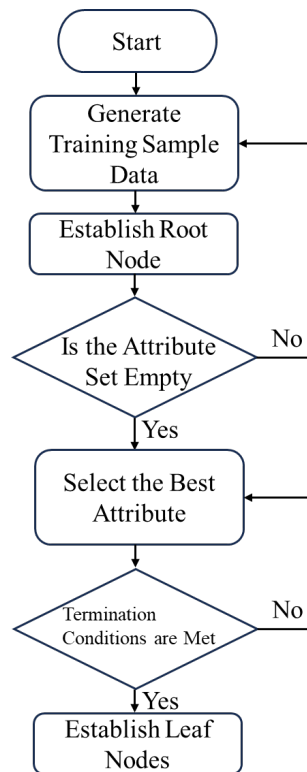


Fig. 2. Decision tree model structure

To confirm accessibility. These rule-based conditions also provide logical flexibility in handling imprecise instances, such as distinguishing between shallow pockets and blind holes. By systematically pre-checking each feature against machining limits [13], the classifier maximizes process predictability and significantly reduces the necessity for manual judgment. As a result, this pre-mapping approach not only standardizes operation selection using industrial best practices but also ensures repeatability and reliability in CNC program generation. The decision tree model used in this article is shown in Fig. 2.

### 3.3 Automatic CNC Program Generation

Following classification, each feature is translated into CNC code using a parametric template, where motion commands are structured as:

$$\text{Toolpath} : r(t) = [x(t), y(t), z(t)]^T t \in [0, T], \quad (11)$$

$$\text{Spindle speed} n = \frac{1000v_c}{\pi D_{\text{tool}}} \quad (12)$$

$$\text{Feed} : F = v_f = n_z f_z \quad (13)$$

$$\text{Blocks} : B = \{G\text{-codes}, M\text{-codes}, S(n), F, X, Y, Z, I, J, K, \dots\} \quad (14)$$

With  $T_k$  the selected tool,  $S_k$  the spindle speed,  $F_k$  the feed rate, and  $P_k(t)$  the 3D tool-path over time, parameterized by cutting direction and function describing plunge or retraction. To make it compatible with a specific CNC machine, the code passes through a post-processor module [14]. This post-processor module translates the generic intermediate G-code into machine-specific dialects such as FANUC, Siemens, or HAAS controller dialects. The dialects vary in syntax, command order, and treatment of auxiliary functions. The post-processor ensures the code adheres to the limitations and conventions of the machine, minimizing the likelihood of execution errors or manual correction.

$$v_c T_n = C \quad (15)$$

$$h_s = R_{\text{eff}} - \sqrt{R_{\text{eff}}^2 - \left(\frac{S}{2}\right)^2} \quad (16)$$

$$\text{MRR} = a_p w v_f \quad (17)$$

$$\tau = \frac{60P}{2\pi n} \quad (18)$$

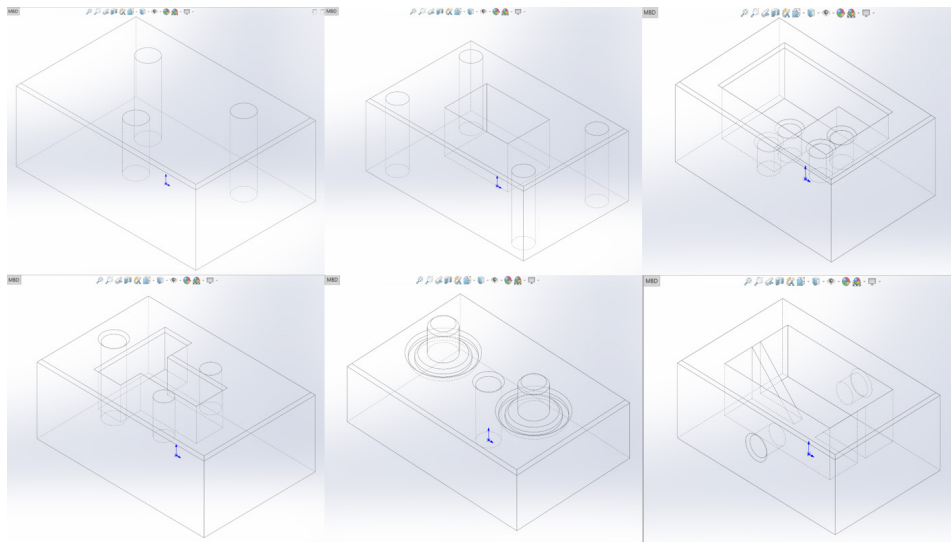
$$v_{f,\text{arc}} \leq \omega_{\text{max}} R \quad (19)$$

Overall, this code generation phase completes the cycle of automation, producing validated, machine-ready code straight from CAD input. It supports a range of feature types, adheres to a variety of controller standards, and provides a scalable solution for machining operations of a wide range. The automated CNC program generation module uses parametric templates to convert each categorical element into executable code for the machine. These parametric templates define tool selection, spindle speed, feed rate, and the 3D toolpath. These generic instructions are then processed by a specialized post-processor that adapts the intermediate code (G-code) to the machine programming language, taking into account syntactic differences, instruction order, and auxiliary function conventions. Simultaneously, full compliance with the control specifications is ensured. In addition to generating correct output, the module improves reliability through adaptive toolpath templates. These include, among other things, helical feeds, step feed planning, safe retraction, and collision-aware motion logic tailored to the geometry of each element. Safety-critical states are automatically inserted to reduce operator workload

and prevent machine downtime. The architecture is scalable and allows the integration of new machining strategies, templates, and control profiles without altering the core logic. By fully automating the conversion of CAD-derived features into validated CNC programs, the system significantly reduces process planning cycles and enables rapid reprogramming across different CNC platforms. This functionality provides a scalable foundation for creating manufacturing environments for small-batch, multi-component production runs and robust, controller-centric CAM automation.

#### 4 Experimental Setup

Six SolidWorks 2023 parts were designed, encompassing industrial geometries such as through/blind holes, rectangular/irregular cavities, T-slots, bosses, and chamfers, ranging from simple axial features to multi-surface/complex cavities. Each model was independently annotated by three CNC experts with over five years of experience, and disagreements were resolved through consensus, establishing a high-quality benchmark for feature recognition and process mapping. The system, implemented in MATLAB, comprises a B-Rep feature extraction engine, a decision tree operation classifier, and a template-based/post-processing G-code generator. Experiments were run on Windows 10 workstations, and the generated programs were post-processed and executed on a FANUC Robodrill  $\alpha$ -D21MiB equipped with a standard tool-set for physical verification. Evaluation used three metrics: feature recognition accuracy, process classification accuracy, and programming time reduction, supplemented by qualitative robustness feedback and post-machining inspections. The validation pipeline loads each STEP model, automatically detects features, maps them to operations, generates parametric tool-paths, and produces FANUC-compliant G-code; the program is dry-run on the machine to check motion/syntax, and then the part is cut under standard conditions while two experienced engineers perform parallel manual programming on the same model, enabling a direct end-to-end comparison of accuracy, speed, and reliability between the automated and manual workflows, both in simulation and in production. The characteristic structures of the six parts designed in this article are shown in Fig. 3.



**Fig. 3.** Schematic diagram of the designed component structure

These six isometric views illustrate the typical geometries used in this study, covering a progressive range of machining feature types and spatial configurations. These parts include simple prisms with isolated axial holes, multi-layered rectangles and irregular cavities, intersecting T-slots, complex hole shapes, combinations of bosses and chamfers, and deep and semi-enclosed cavities formed on multiple surfaces. Each model is designed to systematically cover varying degrees of geometric complexity from easily categorized single features to complex sets of interactive features requiring multifaceted consideration. As shown in Fig. 3, the selected examples

provide a balanced distribution of industrially relevant structures, enabling comprehensive validation of the proposed feature recognition mechanism, operation matching logic, and G-code generation workflow across diverse machining scenarios.

#### 4.1 Test Dataset and Model Preparation

A set of six three-dimensional machinable components was designed as the reference to represent a broad scope of geometries typical for industrial machining in Table 1. The parts were selected to have a mix of significant feature types, such as through holes, blind holes, rectangular and irregular pockets, T-slots, cylindrical bosses, and chamfers. Their geometric elements cover the gamut of feature complexities, from low-level axial holes to several-surface cavities and compound geometries. This is a wide range that ensures a comprehensive testing of the system’s ability to detect, interpret, and assign manufacturing processes by means of various part configurations. All parts were designed using SolidWorks 2023, a standard CAD platform used in industrial design. This ensured that the geometric precision and modeling techniques were the same as those experienced in real-world production scenarios. All the CAD models were then saved as STEP files to maintain neutral, high-fidelity geometric and topological data. The models were then fed into the system’s feature recognition module for automated analysis. To ensure high-quality ground truth for assessment, each CAD model in the dataset was independently annotated by three experts in CNC process planning and feature-based modeling having over five years of experience. Annotators were asked to identify all relevant features, label them with correct feature types, dimensions, and assign relevant machining operations. Discrepancies among annotators were resolved using consensus, which provided high-quality labeled data suitable for bench marking.

**Table 1.** Summary of test data-set and model preparation

Aspect	Description
Number of Models	Six three-dimensional machinable components
Purpose	To represent a wide variety of industrially relevant geometries for CNC feature extraction and process mapping
Feature Types Included	Through holes, blind holes, rectangular and irregular pockets, T-slots, cylindrical bosses, chamfers
Feature Complexity	From simple axial holes to multi-surface and compound cavities
Design Platform	SolidWorks 2023
File Format	STEP - for neutral and high-fidelity geometry and topology retention
Data Input to System	STEP models fed into the system’s feature recognition module for automated analysis
Annotation Procedure	Each CAD model annotated by three independent experts
Annotation Content	Feature type, dimensions, and corresponding machining operation labels
Consensus Process	Discrepancies resolved by group consensus among annotators
Benchmarking Quality	High-quality labeled data-set suitable for bench-marking recognition and process assignment performance

#### 4.2 Hardware and Software Setup

All automated CAD/CAM systems used for CNC machining are implemented in the MATLAB environment. MATLAB was chosen because it has an advanced set of computational geometry tools, powerful mathematical and symbolic processing capabilities, and built-in support for rapid prototyping of complex engineering workflows. MATLAB’s advanced visualization tools can create B-Rep objects composed of entities, geometric primitives, and annotations, making them suitable for feature recognition verification and process mapping. In this environment, the software architecture is divided into three main components: (1) a feature extraction module, which processes STEP-based B-Rep data, performs geometric topology queries, and extracts potential machining features; (2) a decision tree-based process classifier, which aims to match features observed in machining operations by evaluating geometric descriptors, reachability constraints, and process rules; and (3) a CNC program generation module, which uses parameterized templates and machining strategy libraries to generate machine-specific G-code through a post-processing layer. Each module is based on a desktop workstation equipped with an Intel Core i7-12700 processor (12 cores, 2.1 GHz base frequency), 16 GB of DDR4 memory,

and Windows 10 Professional operating system, representing a typical computing environment commonly used in industrial process planning, engineering design, and CAM system development. This hardware configuration provides sufficient computing power for geometric calculations, object graph analysis, and real-time visualization of versatile CAD models, while maintaining compatibility with enterprise-level engineering workflows. For physical verification of the generated CNC programs, the system is connected to a FANUC Robodrill D21MiB machining center a widely used industrial CNC machine tool renowned for its reliability, controller compatibility, and high machining speeds. The Robodrill system used in this study employs a three-axis control architecture, enabling precise execution of operations such as drilling, cavity machining, end-face machining, and contour machining, and can process all features detected by the system. This machine tool is equipped with a standardized toolkit, including turning drills of various diameters, carbide end mills, and face mills, enabling the automation system to map feature types to actual machining operations without human intervention. The FANUC platform was chosen not only for its wide range of industrial applications but also for its extensive documentation and stable controller architecture, ensuring the accuracy of G-code post-processing. The machine tool can perform no-load detection cycles, simulate toolpaths, and provide real-time spindle and feed diagnostics, which are crucial for ensuring the accuracy, safety, and efficiency of custom CNC programs. Therefore, the latest developments combine MATLAB-based algorithm processing with professional CNC execution, ensuring automated testing of the automated program under actual workshop conditions and evaluating the code's accuracy, machinability, and reliability.

### 4.3 Evaluation Metrics

FRA measures the accuracy of the system's recognition of geometric features based on expert ground truth. Each CAD model has a ground truth list containing all real features, including type, pose, and size in Table 2. System output is matched to this list using a one-to-one strategy. A match is considered correct when three conditions are met. First, the predicted type is equal to the ground truth type. Second, the pose error of the feature axis or plane is within the pose tolerance, for example, the axial or normal angle error is below the small angle limit, and the center of mass displacement is less than 0.2 mm. To formalize this condition, pose error is computed as:

$$e_{pose} = \arccos\left(\frac{n_p \cdot n_{gt}}{\|n_p\| \|n_{gt}\|}\right) \quad (20)$$

where  $n_p$  and  $n_{gt}$  are predicted and ground-truth normals. Third, the dimensional error is within the dimensional tolerance, for example, the diameter or width error is less than 0.1 mm or less than 0.5% of the ground truth dimension, whichever is greater. The dimensional error constraint is written as:

$$e_{dim} = |d_p - d_{gt}| \leq \max(0.1mm, 0.005d_{gt}) \quad (21)$$

The report contains the accuracy for each feature type, as well as macroscopic and microscopic averages for the part. By visualizing the differences between different geometries while guiding part machining, confidence intervals can be obtained. Scoring for ambiguous or composite features is based on the primary machining intent recorded in the annotation log. For each correctly recognized feature, the true feature contains the expert-approved operation, such as drilling, boring, reaming, 2D 1/2 pocketing, profile milling, or thread milling, as well as constraints on orientation, tool accessibility, and coolant. A prediction is correct when the selected operation sequence matches the expert plan and the generated plan satisfies all declared constraints. PCA is calculated by dividing the number of correctly classified operations by the number of features that are also present in the ground truth and identified by the system.

$$PCA = \frac{N_{correctops}}{N_{recognizedfeatures}} \quad (22)$$

To avoid bias, PCA does not consider omitted features. Principal component analysis (PTR) quantifies the efficiency gains of automation compared to skilled manual programming. Time is measured from CAD import to

a validated, machine-usable program. The start time is defined as the moment a file is opened in the tool-chain. The end time is defined as the moment the setup table and tool list, as well as the test run list, are generated for the cleaned and post-processed G-code file. For manual benchmarks, two senior programmers followed the same rules, tool library, fixture, and benchmark. The PTR is equal to 1 minus the automation time divided by the manual time. A positive value indicates a savings. A zero indicates a tie. A negative value indicates a higher automation time. Results are reported on a per-part basis and as paired statistics across parts to mitigate the impact of outliers. To ensure fairness, operators are not allowed to reuse previous programs, and both paths must pass the same verification gates. Quantitative metrics are supplemented by a structured qualitative assessment. Manual programmers record issue types, including post-processing syntax, tool collisions, collisions, tolerance violations, and unnecessary retractions, and rate perceived effort and confidence on a simple five-point scale. After cutting, a sample of critical part features is checked for dimensional consistency using a gauge with a resolution of 0.01 mm, and the surface finish is marked as acceptable or improved according to the drawing requirements. All rework events and operator interventions are recorded. The report includes a failure classification and cause for each part.

**Table 2.** Evaluation metrics and protocols for CAD feature recognition

Metric	Purpose	Counting rules
Feature Recognition Accuracy	Accuracy of geometric feature recognition vs. expert ground truth	One-to-one matching between system outputs and ground-truth features. A prediction counts once; if multiple map to the same GT, keep the highest score, mark others as duplicates.
Process Classification Accuracy	Correctness of assigned machining operations for recognized features	Evaluate only correctly recognized features. A prediction is correct if the operation sequence matches the expert plan.
Programming Time Reduction	Efficiency gain of automation vs. expert manual programming	Two senior programmers as manual baseline; same rules, tool library, fixtures, and verification gates for both paths.
Qualitative Assessment	Context on reliability, safety, and operator effort	Manual programmers log issues: post-processing syntax, collisions, tolerance violations, unnecessary retractions.

#### 4.4 Validation Procedure

All test parts follow a consistent, audit-able validation process and match the actual shop floor version. Each STEP file is first parsed into a canonical B Rep and verified for element consistency, watertightness, and correct face normal. The feature engine then uses analytical fitting and topological hints to shortlist candidate features, such as holes, pockets, slots, bosses, and chamfers. Inspection results are filtered based on type match, center of mass or axis error within 0.2 mm, and dimensional error within 0.1 mm or 0.5% tolerance, and are marked as uncertain. For each feature, a parametric template instantiates a tool-path using cutting parameters. Cutting feed is computed based on surface speed and chip load:

$$F = N_f \cdot f_z \cdot n \quad (23)$$

Where  $N_f$  is number of flutes,  $f$  is chip load per tooth,  $n$  is spindle speed. A layer of rules and decision trees then maps each acceptable feature to a machining operation with clear prerequisites, including setup orientation, work offset, tool family, coolant, entry and exit strategies, and priority, while rejecting infeasible pairs, such as those with insufficient reach or minimum corner radius violations. For operation pair, a parametric template instantiates a tool-path using a live tool library, including diameter, number of flutes, and overhang; cutting data, including surface speed and chip load for different materials; and strategy knobs, including step-over, helix, or pocketing patterns. The program assembler then generates a complete part script, including datums, safe retracts, joint motions, canned cycles, tool changes, probing, and M-code. This program is post-processed into FANUC language, including G-code and M-code, macro and cycle syntax, and automatically checks for modal states, planes, radius compensation, feed or spindle mode, and end of program. Pre-cut verification includes background and travel limit checks, collision clearances with the stock and fixtures, soft limits, feed or speed integrity checks, and dry runs without stock to verify kinematics, I/O interlocks, and probing procedures. Only parts that pass all

checks are released and cut on a FANUC Robodrill alpha D21MiB under standard shop floor conditions, using the same vise or fixture, zeroing procedure, and coolant. Cycle time, spindle load, and any operator intervention are automatically recorded. Simultaneously, two senior CNC engineers independently create a complete program for the same part using mainstream CAM software by manually editing tool-paths constrained to the same tool library, stock, fixtures, datums, and acceptance rules to ensure consistency. For each part, the documented feature-level results, including identified input errors and omissions; process-level consistency with the expert plan within orientation and tool constraints, rated as correct, acceptable, or incorrect; programming time from CAD import to G-code verification; cutting results, including first-pass success, rework events, and surface finish records; dimensional inspection of a sample of critical features using calipers and ID gauges with a resolution of 0.01 mm; and fault classification, including post-processing syntax, tool accessibility, collisions, and tolerance violations. For comparison, the performed paired analyses of programming time and error counts for each part. When normality was in doubt, the applied the Wilcoxon signed-rank test and report the median, inter-quartile range, and effect size as well as the mean. All artifacts were version-controlled to ensure reproducibility, and all randomization steps used a fixed random seed. This end-to-end process enabled us to directly and fairly evaluate the speed, accuracy, and reliability of automated and manual workflows in simulation and real production.

## 5 Simulation Experiment and Result Analysis

Sections 5 demonstrate the broader capabilities of the CNC automation system in terms of feature recognition, process classification, and program generation. Because the system encompasses visual cues with spatial interpretation of CAD information, it is adept at performing feature recognition, correctly identifying and classifying a feature regardless of whether it is a hole or a cavity. This means features of both types may receive the same physical treatment in the downstream processing. In process identification, the classifier has been effective in providing valid machining operations for recognized feature types. In some edge cases of a poorly defined feature geometry, there may be slight inaccuracies between similar geometries, but the overall outcomes are successful. Lastly, in program generation, the impact of an automated approach with even limited live human inputs can provide substantial time, and quality improvements, and mitigate error rates. This level of automation enhances efficiency by replacing a human variables with automated outcomes, which generates reliability and consistency faster and with less human error. Strategic programs that consist of tool selection, parameter adjustments, and G-code generation follow specific programmed logic and templates that enable guidelines to produce consistent and repeatable outcomes. There are clearly benefits to CNC automation, acknowledging that the best approach is one where automated programming results can serve as a guideline with human oversight for edge cases to add robustness amongst an endless set of manufacturing scenarios. Overall, this bundling of automation into a workflow improves speed, accuracy, and consistency of operations, while also contributing to design and material flexibility.

### 5.1 Feature Recognition Results

Fig. 4 illustrates the result of feature recognition on a CAD model, where two machining features have been recognized: a cavity and a hole. The hole, a vertical red cylinder in the center of the part, is equivalent to a through-hole feature along the Z axis. These types of features are common in drilling operations and are typically identified based on cylindrical face geometry and closed edge loops in the boundary representation [15]. The system successfully segregated the hole from the part geometry surrounding it and highlighted it by using different colors and labels to indicate its classification. The bottom green cuboid is the cavity, a typical prismatic feature that is usually created by milling operations. The fact that it is horizontal in orientation and at the bottom of the part tells us that it is a blind cavity, which can be recognized by its flat surfaces, vertical walls, and specified depth. This graphical output confirms the ability of the feature recognition module to distinguish between planar and axial features, a valuable functionality for downstream CNC process mapping. The external volume transparency of the part provides spatial context to the way that these features are embedded in the work-piece [16]. It enables the better understanding of depth, alignment, and spatial relationship of features, which are significant factors in tool-path planning and tool accessibility. At a practical level, this visualization enables engineers to verify that the system is properly interpreting CAD data before proceeding to process classification and code generation. This visualization has immediate utility in experimental settings where CAD models are put through an auto-

mated pipeline and recognition results are compared against expert annotations. In this case, since the features displayed are correctly identified per ground truth, these features will positively contribute towards the feature recognition accuracy metric. More significantly, their correct recognition has immediate implications for correct machining operation assignment, with implications for classification accuracy. From a usability perspective, the generation of this legible and clear feature visualization facilitates system commissioning and operator trust. It bridges the gap between automatic feature detection and manual confirmation, allowing users to ensure that the system's interpretation is aligned with the design intent. This kind of visual confirmation is particularly important for complex parts with compound or intersecting features.

Recognized Features on CAD Model (Hole and Pocket)

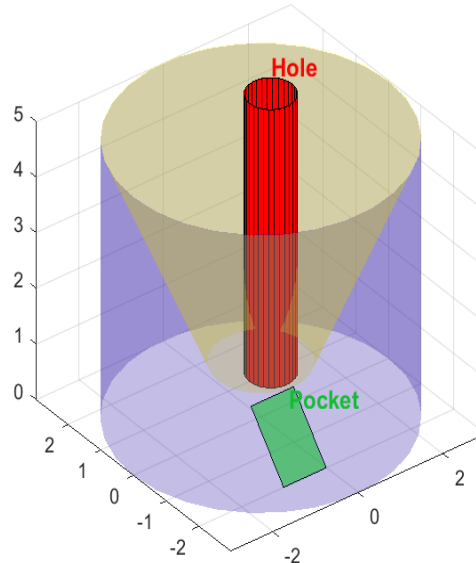


Fig. 4. Feature recognition on a CAD model

## 5.2 Process Identification Performance

Fig. 5's confusion matrix shows how the system performs classification for the task of classifying machining operations using features identified. Absolute prediction counts and percentages normalized are both presented in the matrix, with it providing an explicit idea of how well the model can distinguish between different CNC operations [17]. Accurate classifications are denoted by diagonal cells. Accuracy of drilling and turning classification is very high, with both sharing 100% rates for true positives. In drilling, both test examples were correctly classified and all three examples of turning were correctly classified, illustrating that the system can accurately distinguish between the two processes when appropriate feature input is provided. Conversely, there are a few misclassifications for the milling feature classification. Of the three true milling cases, two were well classified, and one was mislabeled as a drill. This misclassification is presented in the second row and first column of the table. Lastly, the mill classification accuracy is 66.7%, while the false positive rate favoring drilling is 33.3%. This implies that in some situations, the system will confuse closely parameter-similar drilled and milled features, a small circular hole with a through hole. The percentages row-normalized also point to this trend. Two-thirds of the features were correctly labeled in the actual example of milling and only one-third of them were mislabeled. Column-normalized percentages, however, show that there was not a single misclassification among features predicted to be milling or turning, meaning that there were zero misclassifications for both labels. This performance variation can be a sign of imperfections in the feature descriptor detail or the classification model's decision levels. Depth and small-diameter features are unclear whether they are drilling or cavity milling, especially when other traits such as tool accessibility or feature connectivity are insufficiently weighted during the classification.

The confusion matrix supports the prediction that the system performs identification and classification of different machining operations, including drilling and turning, with good accuracy. The lower accuracy in milling suggests lines for possible improvement, for instance, feature descriptor tuning or use of additional geometric cues. The visualization would be applicable as a diagnostic tool to highlight individual regions of misclassification and guide optimization of decision tree logic or sensitivity of feature extraction in the future [18]. The confusion matrix not only validates the reliability of the classification process for different machining operations but also reveals how edge machining or machining processes with similar characteristics challenge existing models. In particular, the misclassification of milling as drilling indicates that overlapping geometric features or toolpaths remain a bottleneck for classifiers. This necessitates improved feature engineering strategies, such as introducing 3D surface curvature descriptors, local toolpath knowledge, or hybrid image-geometric representations, to reduce ambiguity in future iterations. Furthermore, integrating more diverse training samples especially those with partial milling, deep milling, or intermediate milling features can improve the model's generalization ability. In practical applications, the confusion matrix can serve as a real-time diagnostic tool, guiding adaptive adjustments when machining features are detected and enabling model retraining or online threshold optimization when misclassification patterns occur. Therefore, this matrix not only evaluates model performance but also provides actionable insights for iterative system improvements in intelligent machining environments.

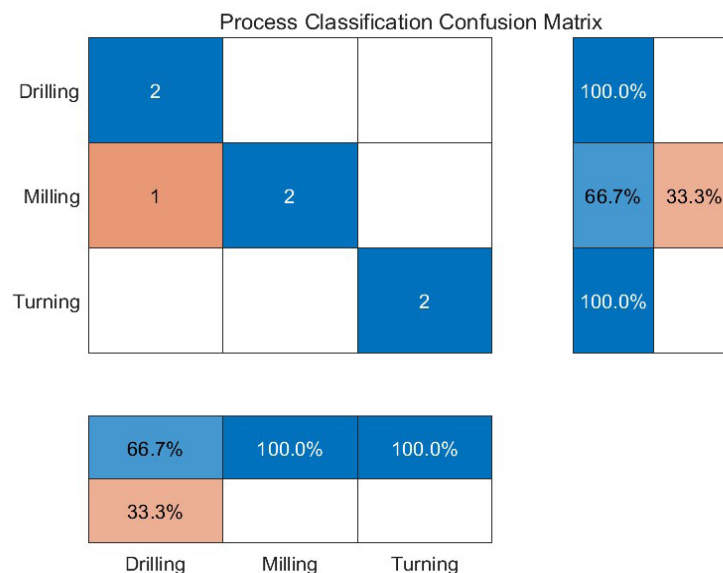


Fig. 5. Process classification confusion matrix

### 5.3 Automatic Program Generation Outcomes

Fig. 6 and Fig. 7 provide comparative description of both manual and automated CNC programming in terms of two major evaluation criteria: error rate and programming time. Together, they offer quantitative evidence for the robustness of the proposed automation system as compared to conventional manual programming performed by experienced CNC operators. In the first comparison of bar graphs programming time, CNC code generation manually is much slower than automated. Manual programming took around 180 seconds, while the automated system took 60 seconds to do the same. This is a reduction in programming time by nearly 67%. The difference indicates the huge boost in efficiency and productivity after the automation process. The most significant time savings are achieved due to the reduction of certain manual operations, interpretation of CAD geometry, selection of machining strategies, adjustment of tool parameters, and generation of G-code. In the automatic process, these tasks are performed algorithmically by pre-defined logic, decision trees, and parameterized templates [19]. This improvement in efficiency is particularly valuable in manufacturing environments where recurring program-

ming is typical because it allows engineers to focus more time on high-level planning, quality verification, or other critical tasks instead of repetitive code generation. The second bar chart, Error Rate Comparison, complements the time study by assessing the quality and validity of the generated CNC code. The chart shows that the manual system has an error rate of approximately 15%, while the automated system has a very small error rate of approximately 3%. The measure can include programming errors such as incorrect tool selection, syntax errors, forgotten safety lines, or geometric discrepancies between the tool path and actual feature size. This result alludes to one of the greatest disadvantages of hand programming, that of human mistake. Even experienced programmers err, especially when under time pressure or intricate geometries. Errors in CNC code can lead to mangled machining operations, wasted material, or worse, equipment damage. Conversely, computer systems severely reduce the likelihood of the same errors by tapping into machining standards rules and proven models. The use of post-processors additionally includes machine-specific compatibility, reducing syntax errors to a minimum. Consistency of the automated system is the second key consideration. The quality of the manual coding will be a function of the programmer’s experience, fatigue, and familiarity of a specific section. Automatic programming, however, produces consistent results that are invariant to external conditions. The standardization is critical in mass manufacturing environments where consistency between batches or sites is required. Scalability is the second area of controversy. The time and error reductions seen here can have a scaling benefit when applied to more complex, larger parts. The programming time for humans increases nonlinearly with the number of features, whereas automated systems can achieve relatively flat performance by processing the features concurrently or by utilizing optimization algorithms. In addition, when there are more process mappings and feature types included in the system knowledge base, the system’s robustness and usability will be further increased. Nonetheless, although the automated system performed quite well in this comparison, it is important to mention that it still possesses a limited error rate. There are some errors in the automated process that can be caused by edge cases, ambiguous features, or out-of-scope features of the training data or the rule set. As automation increasingly becomes more integrated in CAD/CAM activities, such systems will play central roles [20].

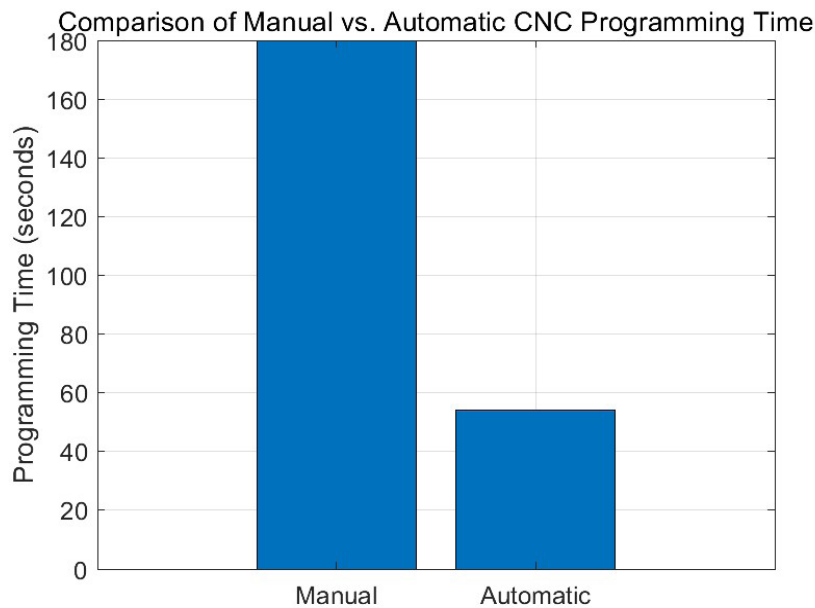
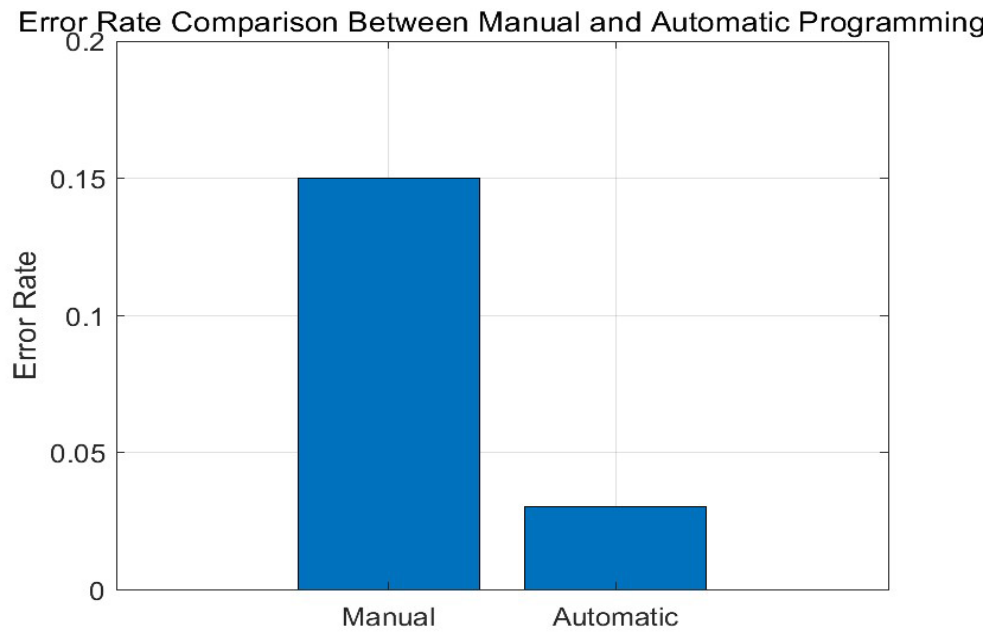


Fig. 6. Comparison of manual vs. automatic CNC programming time



**Fig. 7.** Error rate comparison between manual and automatic programming

#### 5.4 Contributions

This research proposes an end-to-end automated framework that covers the entire process from native or neutral CAD models to machine-executable CNC code [21], eliminating the need for human interpretation or manual coding. It unifies feature understanding, process planning, and code generation into a consistent process, preserving design intent and enabling traceability. At its core, a feature recognition engine processes boundary representation data directly to discover primitive and composite features within industrial part families, including through and blind holes, cavities with regular and irregular boundaries, intersection slots, bosses, chamfers, and other edge treatments. It also records poses, critical dimensions, tolerances, and interaction labels for downstream planning based on priority and manufacturability. Based on this representation, a process classification module assigns a specific machining strategy to each identified feature by combining geometry, accessibility, setup direction, and tool reach into a rule-guided decision tree. This provides clear and auditable options, such as drilling before boring, roughing before finishing, spiral cutting for closed cavities, and thread milling when tapping limits are exceeded. The framework then converts the plan into a controller-specific program via a parametric code generator. The code generator binds feature parameters to validated operation templates, calculates feeds and speeds based on material and tool tables, and constructs safe trajectories using work offsets, datums, compensations, and probing. This code generator ultimately translates into a target post-processor, enabling Fanuc, Haas, Siemens, and Heidenhain controllers to receive valid, readable G and M code containing setup tables and tool lists. Reliability and safety are built in through quality tags attached to features, linting for modal states and compensations, collision and clearance checking for stock and fixtures, dry run verification on the target machine, and versioned artifacts for auditing and rollback, reducing first-time scrap and rework. The framework achieves portability and maintainability through stable data contracts and the decoupling of the recognizer, planner, and post-processors. This enables incremental upgrades, such as adding new detectors, alternative strategies, or additional post-processors, without disrupting the stability of previous steps. Experimental results on mixed batches of industrial parts demonstrate highly accurate feature recognition within tight type and geometric tolerances; consistent process classification within orientation and tool constraints, comparable to expert solutions; significantly reduced programming time and error rates compared to experienced engineers using mainstream CAD tools; and the generated machine-ready code capable of trial runs and shop-floor cutting under standard conditions. These advances will drive the evolution of CAD to CNC automation, enabling intelligent, efficient, and scalable pro-

duction. The integrated system's precise recognition, transparent decision-making, and powerful code generation capabilities will be immediately adopted by process planning teams.

## 6 Conclusion

This paper outlines a fully automated feature-based NC machining process identification approach and its associated program generation, aiming to address a critical gap in the digital manufacturing process. By integrating powerful geometric feature extraction, intelligent process classification, and template-driven NC code generation, the system significantly reduces the reliance on manual labor in process planning while improving programming efficiency and consistency. Experimental validation on representative machine parts demonstrates that the system has high accuracy in feature identification and machining process mapping, and significantly reduces programming time and error rates compared to manual processes. Its value lies not only in end-to-end automation, but also in the systematic integration of design semantics and manufacturing logic, which has traditionally been underestimated in mainstream CAM practices. This integration provides a smarter and more scalable solution for NC automation, especially for high-mix, low-volume production facilities where rapid reprogramming is critical. Beyond its practical engineering value, this work fills a significant long-standing research gap in design feature-based manufacturing. Existing research typically focuses on isolated workflow components, failing to provide a complete and feasible process that connects CAD geometry, process semantics, and CNC executables. Furthermore, end-to-end frameworks validated on actual machine tools remain extremely rare in the literature. This study proposes a complete and experimentally validated workflow, from B-Rep design feature extraction to FANUC-compatible code execution, laying the foundation for multiple integrated benchmarks that demonstrate both conceptual rigor and industrial feasibility, thus paving the way for future research in intelligent CAM systems and autonomous machining. The framework will be further developed in the future to support more complex and compound features, such as free-form surfaces and multi-axis machining operations. In addition, real-time machine tool sensor feedback and adaptive control can be used to support closed-loop NC programming, enabling dynamic tool path adjustments based on process parameters. Applying deep learning techniques to improve feature generalization and cloud-based deployment to enable collaborative manufacturing are also promising directions to extend the functionality and industrial utility of the system to further limits.

## 7 Acknowledgement

Method for Automatic Recognition of NC Processes and Generation of Programs Based on Part Feature Recognition.

## References

- [1] A. Ullah, T.-C. Chan, S.-L. Chang, Current trends in vibration control and computational optimization for CNC machine tools: a comprehensive review, *The International Journal of Advanced Manufacturing Technology* 139(2025) 5409-5444.
- [2] J.-H. Wang, W. Yan, C. Huang, Surface shape-based clustering for B-rep models, *Multimedia Tools and Applications* 79(2020) 25747-25761.
- [3] S.K. Chandrasegaran, K. Ramani, R.D. Sriram, I. Horváth, A. Bernard, R.F. Harik, W. Gao, The evolution, challenges, and future of knowledge representation in product design systems, *Computer-Aided Design* 45(2)(2013) 204-228.
- [4] F. Mo, M.U. Querejeta, J. Hellewell, H.U. Rehman, M.I. Rezabal, J.C. Chaplin, D. Sanderson, S. Ratchev, PLC orchestration automation to enhance human-machine integration in adaptive manufacturing systems, *Journal of Manufacturing Systems* 71(2023) 172-187.
- [5] C. Yeo, B.C. Kim, S. Cheon, J. Lee, D. Mun, Machining feature recognition based on deep neural networks to support tight integration with 3D CAD systems, *Scientific Reports* 11(2021) 22147.1-22147.11.
- [6] C. Oysu, Z. Bingul, Application of heuristic and hybrid-GASA algorithms to tool-path optimization problem for minimizing airtime during machining, *Engineering Applications of Artificial Intelligence* 22(3)(2009) 389-396.
- [7] Y.-X. Yao, M. Liu, J.-J. Du, L. Zhou, Design of a machine tool control system for function reconfiguration and reuse in network environment, *Robotics and Computer-Integrated Manufacturing* 56(2019) 117-126.
- [8] A.M. Al-Shaery, M.O. Khozium, N.S. Farooqi, S.S. Alshehri, M.A.M.B. Al-Kawa, Problem Solving in Crowd

- Management Using Heuristic Approach, *IEEE Access* 10(2022) 25422-25434.
- [9] F.-Y. Jia, Y.-S. Ma, R. Ahmad, Vision-Based Associative Robotic Recognition of Working Status in Autonomous Manufacturing Environment, *Procedia CIRP* 104(2021) 1535-1540.
- [10] J.-H. Wang, Research on Shape Feature Recognition of Boundary Representation CAD Model, *Journal of Computational and Theoretical Nanoscience* 14(1)(2017) 477-484.
- [11] S. Kumar, A. Nassehi, S.T. Newman, R.D. Allen, M.K. Tiwari, Process control in CNC manufacturing for discrete components: a STEP-NC compliant framework, *Robotics and Computer-Integrated Manufacturing* 23(6)(2007) 667-676.
- [12] G. Loy, A. Zelinsky, Fast radial symmetry for detecting points of interest, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 25(8)(2003) 959-973.
- [13] M. Pavlin, F. Novak, Yield enhancement of piezoresistive pressure sensors for automotive applications, *Sensors and Actuators A: Physical* 141(1)(2008) 34-42.
- [14] X.-W. Xu, S.T. Newman, Making CNC machine tools more open, interoperable and intelligent-a review of the technologies, *Computers in Industry* 57(2)(2006) 141-152.
- [15] N. Ismail, N. Abu Bakar, A.H. Juri, Recognition of cylindrical and conical features using edge boundary classification, *International Journal of Machine Tools and Manufacture* 45(6)(2005) 649-655.
- [16] S. Satorres Martínez, J. Gómez Ortega, J. Gámez García, A. Sánchez García, E. Estévez Estévez, An industrial vision system for surface quality inspection of transparent parts, *International Journal of Advanced Manufacturing Technology* 68(2013) 1123-1136.
- [17] P.G. Mongan, E.P. Hinchy, N.P. O'Dowd, C.T. McCarthy, N. Diaz-Elsayed, An ensemble neural network for optimising a CNC milling process, *Journal of Manufacturing Systems* 71(2023) 377-389.
- [18] A.T. Azar, S.M. El-Metwally, Decision tree classifiers for automated medical diagnosis, *Neural Computing and Applications* 23(2013) 2387-2403.
- [19] M. Žaková, P. Křemen, F. Železný, N. Lavrač, Automating Knowledge Discovery Workflow Composition Through Ontology-Based Planning, *IEEE Transactions on Automation Science and Engineering* 8(2)(2011) 253-264.
- [20] B. Fernandez-Gauna, I. Ansoategui, I. Etxeberria-Agiriano, M. Graña, Reinforcement learning of ball screw feed drive controllers, *Engineering Applications of Artificial Intelligence* 30(2014) 107-117.
- [21] Y.-X. Yang, W.S. Robertson, A. Jafari, M. Arjomandi, Advanced numerical approaches for magnetic force calculations: a comprehensive review, *Progress in Electromagnetics Research B* 115(2025) 78-94.