machines

MDPI

Article

Novel Integration of CAPP in a G-Code Generation Module Using Macro Programming for CNC Application

Trung Kien Nguyen [1], Lan Xuan Phung [1,*] and Ngoc-Tam Bui [1,2,*]

[1] School of Mechanical Engineering, Hanoi University of Science and Technology, No.1 DaiCoViet Rd, Hanoi 10000, Vietnam; trung.nguyenkien@hust.edu.vn

[2] College of Systems Engineering and Science, Shibaura Institute of Technology, Tokyo 135-8548, Japan

* Correspondence: lan.phungxuan@hust.edu.vn (L.X.P.); tambn@shibaura-it.ac.jp (N.-T.B.); Tel.: +84-935-888-435 (L.X.P.)

Received: 7 September 2020; Accepted: 29 September 2020; Published: 12 October 2020

Abstract: In the modern manufacturing industry, the role of computer-aided process planning (CAPP) is becoming increasingly crucial. Through the application of new technologies, experience, and intelligence, CAPP is contributing to the automation of manufacturing processes. In this article, the integration of a proposed CAPP system that is named as BKCAPP and G-code generation module provides a completed CAD–CAPP–CNC system that does not involve any manual processing in the CAM modules. The BKCAPP system is capable of automatically performing machining feature and operation recognition processes from design features in three-dimensional (3D) solid models, incorporating technical requirements such as the surface roughness, geometric dimensions, and tolerance in order to provide process planning for machining processes, including information on the machine tools, cutting tools, machining conditions, and operation sequences. G-code programs based on macro programming are automatically generated by the G-code generation module on the basis of the basic information for the machining features, such as the contour shape, basic dimensions, and cutting information obtained from BKCAPP. The G-code generation module can be applied to standard machining features, such as faces, pockets, bosses, slots, holes, and contours. This novel integration approach produces a practical CAPP method enabling end users to generate operation consequences and G-code files and to customize specific cutting tools and machine tool data. In this paper, a machining part consisting of basic machining features was used in order to describe the method and verify its implementation.

Keywords: CAPP; G-code generation; macro programming; machining feature; CNC machining

1. Introduction

Nowadays, computer numerical control (CNC) is widely used in production industries, since various curve shape components can be produced with CNC machines with high efficiency and high quality [1,2]. CNC machines are also essential components of flexible manufacturing systems (FMS) and computer-integrated manufacturing (CIM) systems, in which manufacturing automation is utilized to achieve high productivity [3]. In any production system, process planning is an efficient way to increase quality and reduce manufacturing times [4]. The effects of process planning on production capacity and revenue increases have also been analyzed [5]. Additionally, CNC machining performance strongly depends on the cutting parameters that has been determined and assessed through complicated tasks, time-consuming the processes, and a high level of required engineering from calculation, simulation, or experimental works [6–9]. The recent development of

computer software programs such as computer-aided design (CAD), computer-aided engineering (CAE), and computer-aided manufacturing (CAM) software has provided effective tools for parts design and manufacture. However, process planning was previously based mainly on engineers' experience and expertise. Computer-aided process planning (CAPP) acts as a bridge between design and manufacturing in a CIM environment. CAPP involves a computer-converted module that incorporates the parts' technical requirements and other information (e.g., relating to the materials, stock forms, and production types) to form operation sequences that are incorporated with process descriptions, such as descriptions for machine tools, cutting tools, and other equipment [10]. Process planning can be used for many purposes, such as for generating cutting toolpaths in computer-aided manufacturing (CAM) software, for scheduling manufacturing plans, and for production resource management. A CAPP system not only helps to reduce preparation times, but also helps to enhance productivity an automated factory [10,11]. There are two main methods in CAPP: (1) variant and (2) generative methods. In the first type, process planning is achieved by identifying and retrieving an existing plan for a similar part and applying certain modifications to form a new process plan. Although the programming technique used with this method is simple, optimization is not easy to achieve in many processes and the implementation quality still depends on the knowledge and skills of the engineers [12]. With generative method, process planning is implemented through technology databases, technology rules and constraints, and mathematical models combined with selection methods and optimal calculations [13]. Although it is difficult to develop, the generative method can perform process planning in detail and is easy to integrate with CAD–CAM–CNC systems. The generative method was applied in this research, with the main aim of creating a process planning method that is as detailed as possible and able to be integrated with a CAD–CAM–CNC system. In a CAPP system, a foundation method is necessary for machining feature recognition, machine tool and cutting tool selection, and machining operation procedures (process planning). Recent studies have shown that machining feature recognition methods only provide basic geometric information rather than complete information for the technical requirements, geometric intersection relationships and contour shape definition, which are important in process planning [14,15]. Feature recognition methods mainly focus on 2.5D features, such as slots, steps, and holes, but not on hybrid machining features, such as contour pockets with islands [16]. A workpiece's technical requirements are mostly defined through the user interface for each feature or through separated procedures, thus diminishing the integrity of the CAD–CAPP connection [17,18]. The cutting tool selection methods have not been comprehensively evaluated using multiple criteria and they are not suitable for large-database systems with many types and sizes of cutting tools [19]. The methods used to select machine and cutting tools do not usually involve a flexible connection to the manufacturing database for each factory. Moreover, some processing parameters related to heat treatment, stock preparation methods, and product types are not considered in machining operation recognition and cutting tool selection processes [20]. Therefore, the selection results would not be adequate for practical applications. In the most recent studies, CAPP systems were mainly built to create a connection between CAD and CAPP systems through machining feature recognition systems using commercial three-dimensional (3D) CAD software or neutral CAD files [21,22]. The engineers manually applied the results from CAPP systems to generate the toolpaths for each machining feature in commercial CAM software. There have not been many studies on the automatic integration of CAD and CAPP with a CAM module. To create an integrated CAD–CAPP–CNC system, it is necessary to establish a bridge between CAPP and G-code programs, which are directly generated during the process planning in the CAPP system. However, such connections are still limited, since G-code files are mainly based on manual processing or are processed using commercial CAM software. Generally, three basic method types are used to create a G-code file for CNC machines: (1) the traditional manual programming method, (2) the automatic method, and (3) macro or parametric programming methods. The first method uses basic G-code commands and cycles for manual programming. This method is difficult to use for automated programming, since it requires a numerical coordinate for every programming point in each G-code

block. The automatic method uses CAD and CAM software to create G-code files on the basis of toolpath strategies created by users. The programming function that uses application programming interface (API) features inside CAM software is not fully modifiable. Therefore, this method is not easy to integrate with a CAD–CAPP system. Macro or parametric programming methods are CNC programming methods that are based on defined parameters, which are normally the geometric parameters of machining features [23,24]. These parameters can be easily extracted from the CAPP module through machining feature recognition. Therefore, this method can be integrated with CAD-CAPP systems.

This paper proposes a CAPP system, named BKCAPP, containing basic recognition modules based on the development of identification and selection methods, as well as a process planning system combined with an automatic G-code generation module based on macro CNC programming. A sample part was selected in order to evaluate the system's implementation and effectiveness. The geometrical and technological information for each machining feature extracted from the CAPP system was used in the automatic formation of CNC programs. Each machining feature of a part has a CNC subroutine in macro programming. The main program uses subroutines in the machining sequence determined by BKCAPP in order to complete the machining of the part. The G-code generation module can also be used independently in manual programing by providing the parameters of machining features. If there is a change in the geometry of a workpiece then the end users simply provide a different set of feature parameters to change the toolpaths. On the basis of this advantage of CNC programming in terms of the parameter selection method, this study established a new set of commands and functions that use feature parameters extracted from CAPP modules to directly generate the machining program for the entire workpiece. The results are a fundamental step forward for automatic CAD–CAPP–CNC integration, which creates a process planning and G-code files with high efficiency and without time-consuming processes in CAM software.

2. Procedure and Implementation Approach

2.1. Principle of Proposed Approach

Figure 1 presents a diagram describing the automatic process used to generate G-code files from a 3D solid model in *.sldprt format. First, from a 3D solid model of the part designed in commercial CAD and CAM software, the geometrical and technical requirement data are extracted. On the basis of extracted geometrical data, the rule-based method is then applied for machining feature recognition. In the next step, the technical requirement is extracted and used as input data for machining operation recognition and machining step separation. The whole process of extraction and recognition proceeds automatically. The intermediate and final extraction and recognition results are saved to the database in a Microsoft SQL server for use in subsequent processing steps in process planning, such as in cutting tools, machine tool selection, and operation sequence generation. At this stage, the automatic connection between CAD and CAPP is established. The G-code, which is used for CNC machining for each operation, can be automatically created with the G-code generation module on the basis of process planning generation results and geometrical data extraction. The G-code file for the whole part is generated by adding all G-code segments for each operation. Therefore, the integrated CAD–CAPP–CNC system is automatically constructed G-code files for CNC machining from a 3D solid model. This work focuses on developing a CAPP system, called BKCAPP, which has functional modules that include feature extraction and machining feature recognition, machining operation recognition, cutting tool selection, machine tool selection, and operation sequence generation modules. A database is used for the modules to store extraction data, recognition rules, machining condition types, cutting tool and machine tool libraries, and intermediate and final results in the calculation and processing stages.

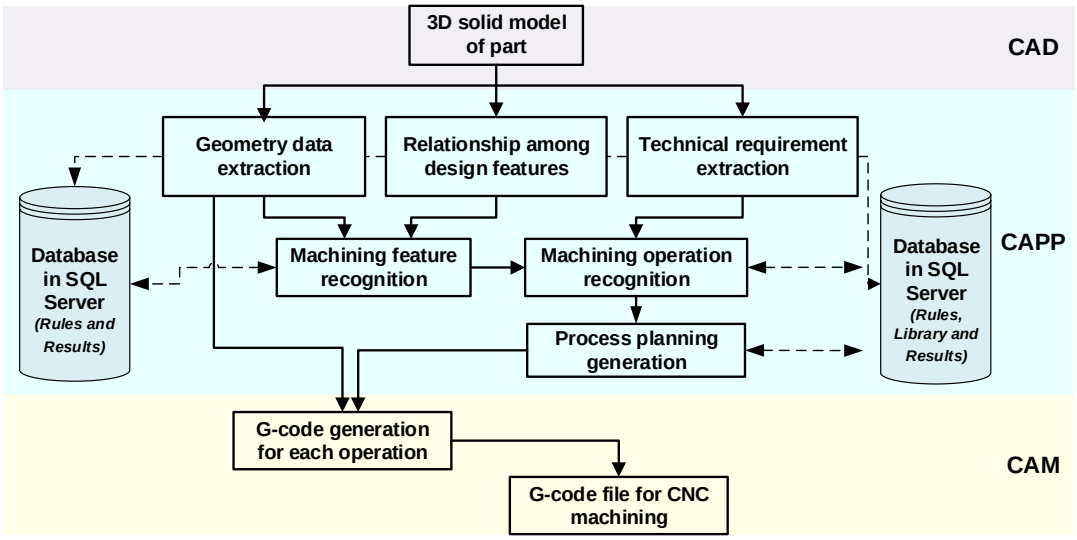


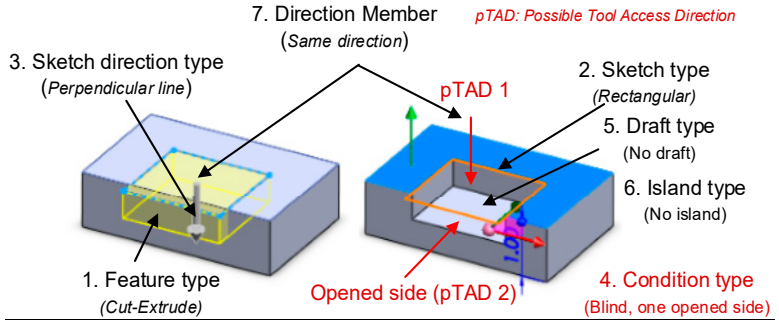
Figure 1. Principles of the integrated computer-aided design (CAD)–computer-aided process planning (CAPP)–computer numerical control (CNC) G-code flow.

2.2. Extraction Process

Extraction process for geometry data: In the data extraction process, it is necessary to extract all geometrical data, including the modeling method and all dimensions for each unit feature of the 3D solid model. The following seven data types were determined in order to fully understand the modeling method of the design features, except for the hole feature, as presented in Table 1. All seven data types are extracted and used as inputs for the machining feature recognition step of the design feature.

Table 1. Definition of seven geometrical data extraction types.

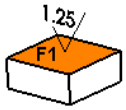
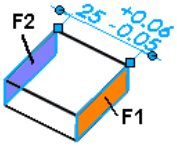
1. Feature type	Real feature type for each unit feature, such as extrude, sweep, or revolve, in a boss or cut type.
2. Sketch type	Sketch shape types, such as circular, rectangular, polylineal, and splined.
3. Sketch direction type	Line types for most design feature cases, such as line, circular, and spline, for sweep design features.
4. Condition type	Most important characteristics of geometrical extraction data. For each design feature, it is necessary to determine which characteristics are closed or open and blind or through with cut-extrude-type design features, and whether they are free or interact with any boss-extrude type design features.
5. Draft type	Existence of draft shape of design feature to determine no draft, draft+, or draft type.
6. Island type	Existence of the island inside the design feature.
7. Direct member	By checking the reverse direction of a normal vector of a sketch with one of the possible tool access directions to determine the same or a different direction.



The hole feature built from the Hole Wizard design feature in SolidWorks can be easily extracted with up to 25 different parameters by using the GetDefinition function of the WizardHoleFeatureData2 feature. The specific dimensions and contour shapes of each unit feature are also important for the determination of machining processes, such as machine tools, cutting tools, cutting data selection, and G-code generation, in order to determine the feature creation types. The most important extracted dimensions include the depth, width, and length of a feature. For the complex pocket compositions of islands, which are protrusion or boss-extrude design features created inside cut-extrude, other specific dimensions are extracted, such as the minimal dimensions between walls, the minimal dimensions between a wall and island, and the dimensions between two protrusion islands in a pocket. The base face and contours of machining features are also obtained. The open characteristics of each feature are also determined in this step. These open directions are also called the possible tool access directions (pTADs) of a feature. For example, the feature presented in Table 1 has two pTADs, which are combined with the base face's characteristics to select the main machining direction or main tool access direction (TAD) for the operation sequence and G-code generation process.

Technical requirements for the extraction process: The set of technical requirements of a machining part is another factor that determines the quality of the part machining process. This proposed process also considers each feature and its technical requirements, meaning recognition of the technical requirements is then needed. In this study, the most important technical requirements for the machining process are the surface roughness, geometrical dimensioning, and tolerancing. Table 2 presents some examples of technical requirement extraction.

Table 2. Examples of technical requirement extraction data.

Example	Requirement	Type	Value (μm)	Datum	Base 1	Base 2	Level
	Surface roughness	Ra	1.25	-	F1	-	7
	Dimension accuracy	Dimension Dimension tolerance	25 +0.06 –0.05	-	F1	F2	11

The extraction process for the relationships between machining features: In machining feature recognition, the interactions between features are complex problems. A feature can have adjacent or volume interactions with other features. The existence of these interactions sometimes changes the machining process of the feature. Therefore, it is necessary to identify the relationships between machining features. These data are important for recognizing the right machining features and for generating suitable operation sequences with machining constraints in the process planning stage.

2.3. Recognition Process and Process Planning Generation

Machining feature recognition: The machining feature recognition process consists of two different procedures. Figure 2 gives examples of recognition rules for both the Hole Wizard tool and a feature without Hole Wizard. The machining direction is also defined by the TAD and the base face. The first procedure is used for the Hole Wizard feature and the second is applied for the other features except for Hole Wizard. The rules were constructed to recognize hole machining features on the basis of the values of specific parameters in the list of design parameters extracted from the Hole Wizard design feature. According to these feature recognition rules, a design feature can be recognized based on several machining features, such as drill holes, drill holes and countersinks, and drill holes and tapping. For other design feature types, recognition rules are established and saved in the database on the basis of seven extracted information items, as shown in Table 1. This can determine whether a machining

feature is a step, slot, pocket, or boss. Moreover, the feature characteristics are determined as closed or open and as blind or through, while the shape of the feature is determined as being rectangular, circular, or a complex contour. Figure 3 shows examples of results automatically obtained using the machining feature extraction and recognition process with a BKCAPP system.

If a design feature has
Diameter *parameter is not equal to 0 and*
Depth *parameter is not equal to 0 and*
Drill angle *parameter is not equal to 0 and*
Then
Machining feature is ***Drilled hole***

(a) Machining feature for Hole Wizard feature recognition.

If a design feature has
1. *Feature type is* ***Cut-Extrude*** *or* ***Sweep-Cut*** *and*
2. *Sketch type is* ***Rectangular*** *and*
3. *Sketch direction type* ***is Perpendicular line***
4. *Condition type is* ***Two opened sides and blind*** *and*
5. *Draft type is* ***No draft*** *and*
6. *Island type is* ***No island*** *and*
7. *Direct member is* ***Same direction***

Then
Machining feature is ***2.5 Blind rectangular step***

(b) Machining-feature recognition (except Hole Wizard feature)

Figure 2. Examples of machining feature recognition rules.

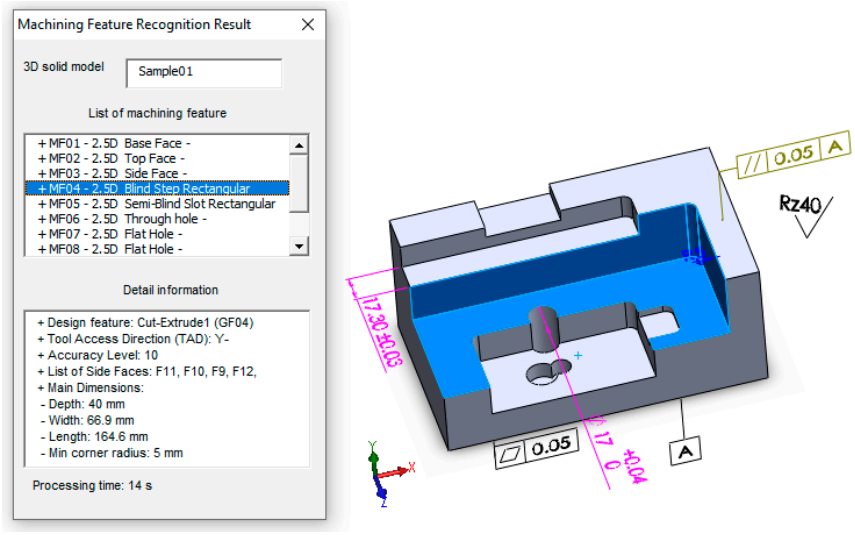


Figure 3. Example of machining feature recognition.

Machining operation recognition: The application of machining operation recognition to each machining feature is very important for further processes in the selection of machines and cutting tools. The recognition rules determine machining operations for each machining feature on the basis of the dimensions and types of machining features and the machining quality. There are two stages in the recognition process. First, recognition rules are constructed in order to determine basic machining operation types, such as shoulder milling, pocket milling, face milling, slot milling, profile milling, volume milling, and hole making, for each machining feature on the basis of a machining feature type. In the second stage, the list of cutting tools is determined on the basis of the predetermined machining operation type, machining feature dimensions, and quality level. For example, the pocket milling operation can use many types of cutting tools, such as open pocket, roughing, semi-finishing,

finishing, and rest milling tools. Cutting tools for hole machining features can include start drills, drills, bore tools, and reamers, and sometimes end milling can be used for hole making, depending on the availability of cutting tools in the library. The detailed machining operation steps can be determined on the basis of the cutting tool selection and the main machining operation type.

Operation sequence generation: For each machining feature, some data are determined for the operation sequence generation process, including the machine, cutting tool, setup, and precedence constraints. The operation sequence is generated using a developed clustering algorithm in order to minimize machining costs based on the traveling costs for the machine, setup, and cutting tool changes, while still ensuring that the precedence constraints are not violated [25]. The concept for the algorithm is based on the calculation of a similarity coefficient between any two operations from the operation list for a part. The two operations with the highest similarity coefficient that indicated by the lowest machining costs are pushed in the operation sequence list. The grouping process is done when all operations in the operation list are grouped.

2.4. G-Code Generation Module

G-code generation module: From the extraction results, the basic dimensions, the Cartesian coordinates of vertex points, and the contour types of each machining feature are determined. In default mode, workpiece coordinates are automatically defined by the origin of the 3D model. However, the user can manually choose the origin using the stock definition section in BKCAPP, whereby an origin (or zero points) conversion module is established. The G-code generation module was developed in two forms, as shown in Table 3. The first is an integrated function in the BKCAPP system, while the second works as a G-code cycle (work-alone function) that allows for users to easily export machining programs to a specific CNC machine support parameter program (using macro programming). In both forms, the G-code generation module includes a series of subgeneration functions to convert geometric and machining data into G-code form. Each subgeneration function creates a G-code subprogram for a basic type of machining feature, such as a face, pocket, boss, slot, or hole. The subgeneration function calculates coordinates and generates the G-code while using input parameters. The number of input parameters differs depending on the complexity of the machining feature. The input parameters are the extracted data from the geometric and technical information, as described in the previous section. For simple machining features, such as a square or circle pocket, the geometric parameters are quite simple, including for the shape dimensions (e.g., width, length, and diameter). However, if the machining feature is a contour feature that is a combination of straight lines and arcs, there should be a G-code list describing the entire profile, which is automatically obtained from the feature extraction process.

The way to declare a profile is similar to the declaration of a turning cycle in the Fanuc control system. Figure 4 shows an example of the declaration of a machining profile feature. The absolute coordinates of a reference point that determine the position of the feature on the workpiece must also be declared. The reference points are separately specified for each different feature. For example, the center of the pocket is taken as a reference parameter for circular pocket features, while the lower-left corner point is declared for the square pocket feature. For contour features, the coordinates of the points in the contour express the position of these features. If the entire feature is rotated at a certain angle, besides the coordinates, the feature's rotation angle to the X+ axis is also declared. The technological parameters are basically declared by the cutting parameters (spindle speed S and feed-rate F) and distance between the two toolpaths (stepover). There are also parameters that are related to the type of cutting, such as the total depth of cut (DOC), depth of cut per toolpath, and the rough machining or finish machining process.

Table 3. Two G-code forms for contour features.

First form (integrated function in BKCAPP system): G65 P1040 Z … K … H … E … T … B … C … R … M … U … W … F … S …	
Second form (work-alone function): G104 Z … K … H … D … E … T … G105 P … Q … B … C … R … M … U … W … F … S...	
- Z: absolute coordinates of profile bottom in Z (+/–); - K: height/depth of the island (+)/pocket (–); - H: tool radius compensation; - D: diameter of cutting tool; - E: pocket machining (E = 1)/island machining (E = 0); - T: tool number; - P: block number start defines a profile; - Q: block number end defines a profile;	- B: step over toolpaths (%tool radius); - C: depth of cut in Z (+); - R: retract plane; - M: maximal depth of cut in XY; - U: depth of cut in XY in finishing step; - W: depth of cut in Z in finishing step; - F: feed rate; - S: spindle speed.

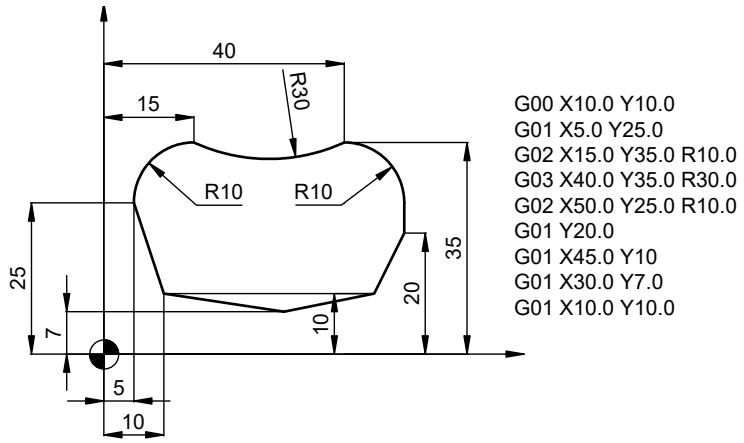
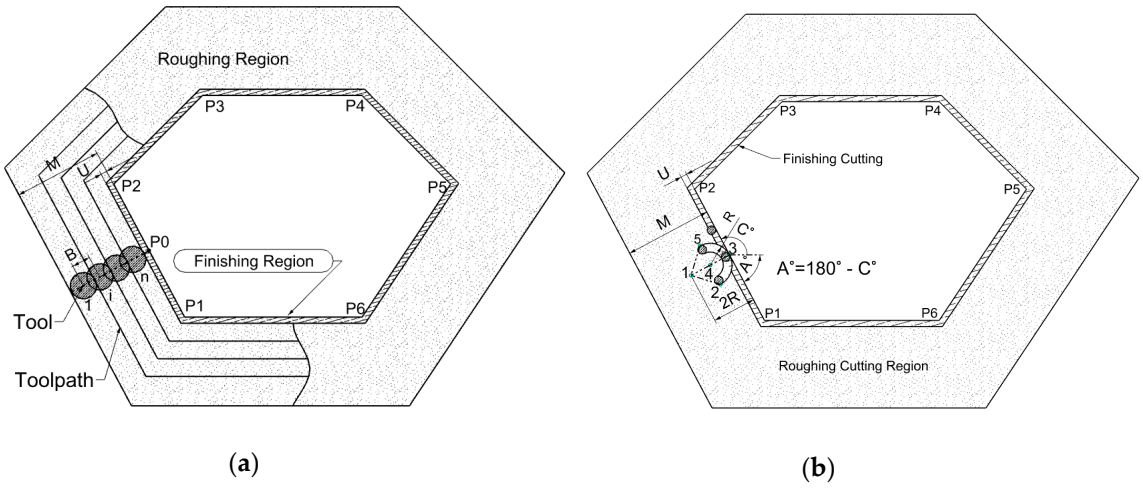


Figure 4. Example of contour feature declaration.

Toolpath generation algorithm: The toolpath generation module in CAM software mostly computes the actual coordinates of the tool reference points. In the G-code generation module, the toolpath generation algorithm only computes the start point, points of the profiles, endpoint, and tool radius compensation. On the basis of process planning results, a set of machining processes, machine and cutting tools, cutting conditions, and G-code files is automatically generated. The calculation of the actual toolpath is executed through G41/G42 cutter radius compensation by the CNC controller. When multiple toolpaths are parallel to the profile, the cutter radius compensation value is changed accordingly. With this algorithm generation method, the toolpaths for rough machining are simple and straightforward with fixed forms, similar to toolpaths in manual programing. In the finishing process, the toolpaths are similar to those in CAM software. The straightforward forms help the user in controlling the order and reliability of toolpaths, however the cutting quality is still comparable to that achieved with CAM software. Toolpaths for contour machining feature samples with rough machining and finishing processes are shown in Figure 5. In rough machining, toolpaths are parallel paths to the machining profile ($i = 1 \div$ n; n is the total number of contours per cutting layer with the same depth of cut in Z). The coordinates of the programing points on these toolpaths are automatically calculated by changing the cutter radius compensation value Hi corresponding to each ith toolpath on the basis of the total DOC (M), stepover (B), finishing DOC (U), tool diameter (D), and logic value K ($K = 1, 0$), as shown in Equation (1). This toolpath generation algorithm produces simple programs for rough machining that can be easily checked, verified, and modified.

$$H_{i(rough)} = \left[M - \frac{D}{2} + (i-1)B\right]K + \left[\frac{D}{2} + U\right](1-K) \quad (1)$$

In finish machining, the finishing toolpath (i = n + 1) cuts off the finishing DOC to form a machined profile with the level of required quality that is determined in the process planning stage. The toolpaths in this step is slightly different than those in rough machining, as entry and exit toolpaths are added according to the curvature radius (R), which is specified by the programmer or by default calculation from the G-code generation module. Therefore, programming points for the entry (1→2→3) and exit (3→5→1) toolpaths, as shown in Figure 5b, are calculated on the basis of the geometry of the start profile segment (P1–P2), the curvature radius, and total DOC. Equations (2)–(6) show the calculation of these points' coordinates based on angle C of the profile segment relative to the X+ axis.



(**a**) (**b**)

Figure 5. Toolpaths for (**a**) rough, and (**b**) finish machining.

$$X_3 = \frac{X_{P1} + X_{P2}}{2} \qquad Y_3 = \frac{Y_{P1} + Y_{P2}}{2} \tag{2}$$

$$X_1 = X_3 - 2R * sinA \qquad Y_1 = Y_3 - 2R * cosA \tag{3}$$

$$X_4 = \frac{X_1 + X_3}{2} \qquad Y_4 = \frac{Y_1 + Y_3}{2} \tag{4}$$

$$X_2 = X_4 + R * cosA \qquad Y_2 = Y_4 - R * sinA \tag{5}$$

$$X_5 = 2X_4 - X_2 \qquad Y_5 = Y - Y_2 \tag{6}$$

where $A = 180° - C$; C is the angle of the 1st profile segment to unit vector $\vec{u}[0, 1]$ of the X+ axis:

$$cosC = \frac{0 * (Y_{P2} - Y_{P1}) + 1 * (X_{P2} - X_{P1})}{1 * \sqrt{(X_{P2} - X_{P1})^2 + (Y_{P2} - Y_{P1})^2}} \tag{7}$$

3. Results

3.1. Geometry and Technical Data Exaction

Figure 6 shows the main interface of the BKCAPP system, in which the input is a sample of a 3D model with technical requirements in SolidWorks format (*.sldprt), along with some initial manufacturing parameters, such as the product types or the number of parts per year, the stock preparation method, the heat treatment method, the workpiece material and hardness, and a cutting tool and machine tool library. The BKCAPP system is connected to a database while using a Microsoft SQL server to retrieve and store the processing and results data. The outputs are the lists of machines and cutting tools, the process plan, and the G-code files for the machining of parts. The workpiece sample used for the case study has basic technical requirements related to the surface roughness, geometric dimensions, and tolerance. The manufacturing parameters are: S45C steel for the workpiece

material, small batch production type, forging method for stock preparation, annealing method for heat treatment, a specific machine tool library, and the Sandvik cutting tool library. The processing time for the machining feature recognition process for the sample is 9 s with the BKCAPP. The machining feature recognition module can recognize all features, such as the base face, top face, side faces, pocket, and holes, the basic dimensions and machining accuracy levels for which are described in Table 4. In addition to the basic dimensions of the machining features, the coordinates of points in the profile of the machining contour features (MF05) at the side faces were also extracted for the G-code generation process.

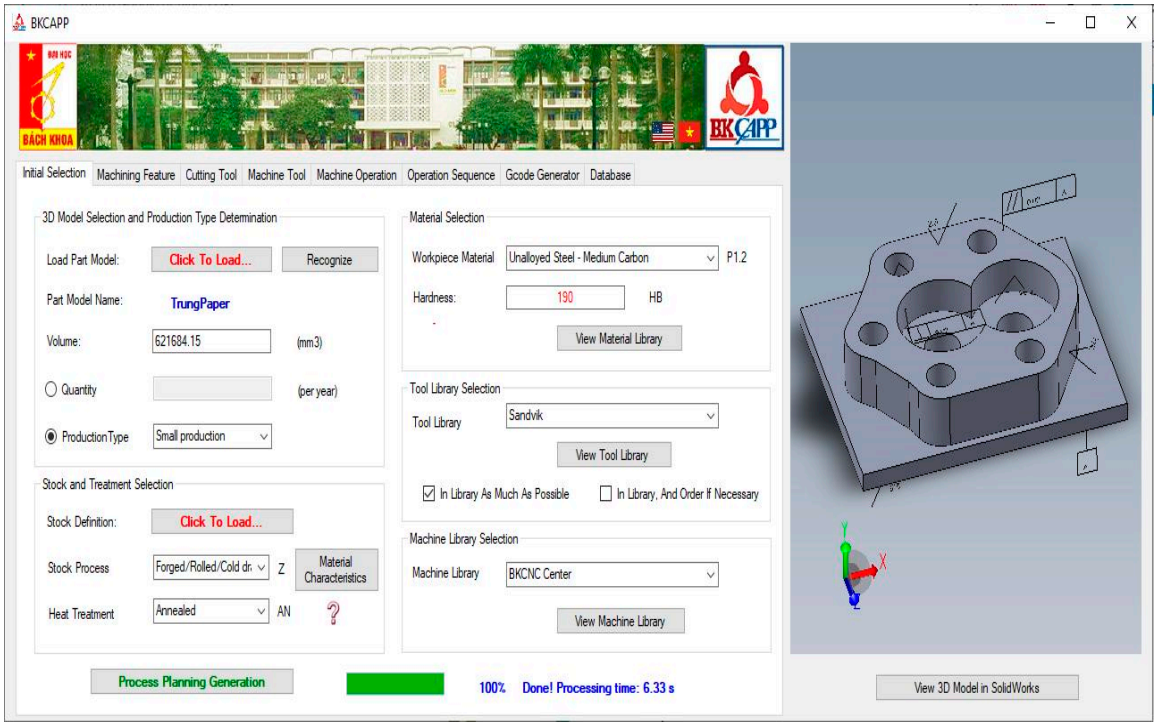


Figure 6. Interface of the BKCAPP system with three-dimensional (3D) part drawing input.

The interaction relationships between machining features were also extracted in order to define constraint inputs to determine the operation sequence. On the basis of machining feature recognition results, the process planning includes a selection of machining processes, machine and cutting tools, and cutting conditions. The optimal operation sequence is established by considering the machining costs and precedence constraints. The operations or steps for each machining feature for the sample presented in Table 5 show that the process planning method involves 11 different machining processes, including rough and finish machining processes, which are determined by the interval of tolerance (IT) level of a machining feature. BKCAPP specifies the cutting tools, conditions, and power for each of these operations, depending on the machine and tool databases. The process planning step for the BKCAPP system takes about 6.3 s for the sample. The intermediate processing steps and results are all stored in an SQL server.

Table 4. Machining feature and technical requirement recognition results.

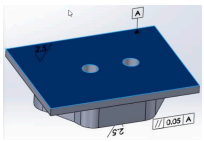
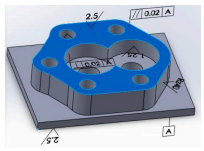
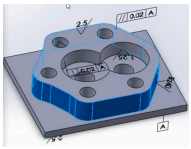
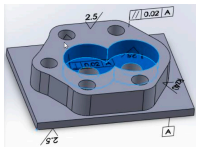
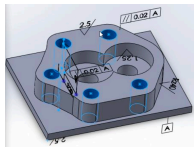
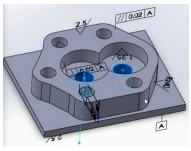
Machining Feature ID	Machining Feature	Depth (mm)	Width (mm)	Length (mm)	IT Level	Geometry ID
MF01 (base face)		0	150	190	7	GF01
MF04 (top face)		0	140	165.2	7	GF04
MF05 (side face boss)		30	140	165.2	9	GF05
MF06 (closed-blind pocket)		20	62	114	7	GF07
MF07 +MF08 (drill hole + tap hole)		40	20	0	13	GF06
MF09 (through-hole)		20	25	0	7	GF08

Table 5. Machining feature and technical requirement recognition results.

Machining Feature ID	Machining Operation ID	Machining Operation	Rough/ Finish Machining	Tool	Tool Diameter (mm)	DOC (mm)	Cutting Width (mm)	Feed Rate (**)	Cutting Speed m/min	Spindle Speed (rpm)	Power (W)
MF01	MO01	Face milling	Rough	T01	80	1	40	3010	360	1433	3.2
MF01	MO02	Face milling	Finish	T02	80	0.5	40	991	457	1821	0.5
MF04	MO03	Face milling	Rough	T01	80	1	40	3010	360	1433	3.2
MF04	MO04	Face milling	Finish	T02	80	0.5	40	991	457	1821	0.5
MF05	MO05	Side and face milling	Rough	T03	30	0.5	24	896	281	2985	0.3
MF06	MO07	Circular ramping	Rough	T05	40	0.5	40	995	250	1990	0.1
MF06	MO08	Circular milling	Finish	T04	20	0.2	20	993	200	3184	0.3
MF05	MO06	Side milling	Semi-finish	T04	20	16	0.5	1241	250	3980	0.2
MF07	MO09	Drilling	Open hole	T06	10.5	5.25	-	0.25 *	100	3033	1.75
MF09	MO10	Drilling	Open hole	T06	10.5	5.25	-	0.25 *	100	3033	1.75
MF08	MO11	Tapping	Rough	T07	12	1.5	-	1.5 *	33	875	0.56
MF09	MO12	Drilling	Widen hole	T08	24	6.75	-	0.1 *	210	2786	2.6
MF09	MO13	Boring	Rough	T09	24.8	0.4	-	0.6 *	255	3274	1.6
MF09	MO14	Reaming	Finish	T10	25	0.1	-	3.75 *	150	1910	1.5

DOC: depth of cut; (*) feed rate per revolution; (**) feed rate per minute.

3.2. G-Code Generation Module Implementation

On the basis of machining feature recognition results from the geometrical and technical data extraction steps, the G-code generation module automatically creates the G-code list for each machining feature. Table 6 presents the types of segments in the contour profile (MF05 in Table 5) and the coordinates of profile points that were obtained from the BKCAPP system. The line type includes (0) rapid traverse, (1) linear cutting, and (2) circular cutting. This geometrical data file can be used as input data for the G-code generation module. Figure 7 shows the interface of the G-code generation module with an upper command window, which allows for a user to enter parameters of G-code cycles for part machining or to import geometrical data files from the BKCAPP system; the right window shows the contour from the input data. The lower-left window presents the output G-code programs in the Fanuc 21 or Fanuc 31 control system. The G-code generation module also has other functions for checking and simulating toolpaths in a 2D view. The machining simulation of a CNC program for contour features that was created by a G-code generation module for the Fanuc 21 system is shown in Figure 8. Rough machining was performed on a blank rectangular block (210 × 170 × 50 mm) using an end mill (diameter of 30 mm), while the finish cutting was conducted with a tool diameter of 20 mm. Rough machining requires 10 cutting layers in the contour depth direction, with a DOC of 3 mm for each cutting layer. With the stepover at 75% of the tool diameter, each cutting layer in rough machining requires three toolpaths that are parallel to the contour profile. These numbers of cutting layers and toolpaths are automatically calculated by the module on the basis of the geometrical and technological characteristics of the machined feature. Figure 8 shows the machining simulations for the first, third, and fifth cutting layers for rough machining and finishing cutting of the profile wall (set by parameter U0.5 in the G-code rough machining cycle). The simulation results show that the G-code program is appropriate for parts machining.

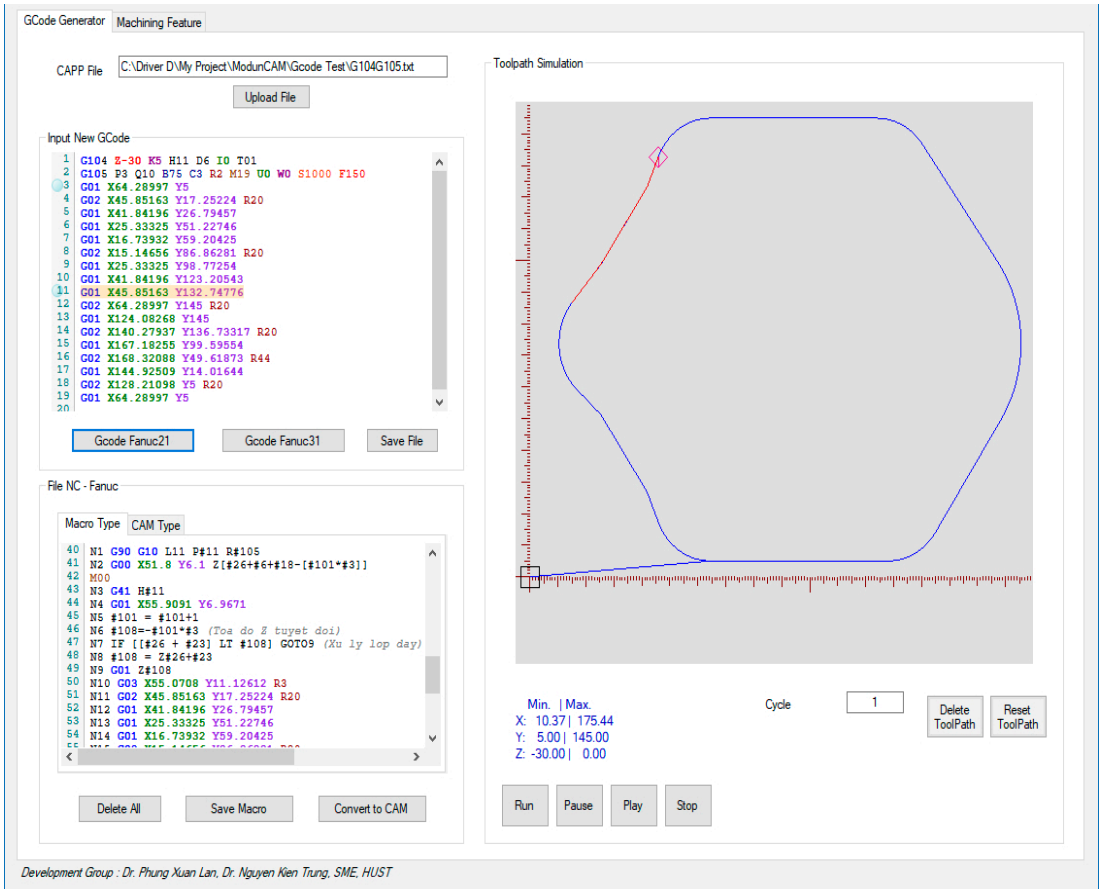


Figure 7. The interface of the G-code generation module.

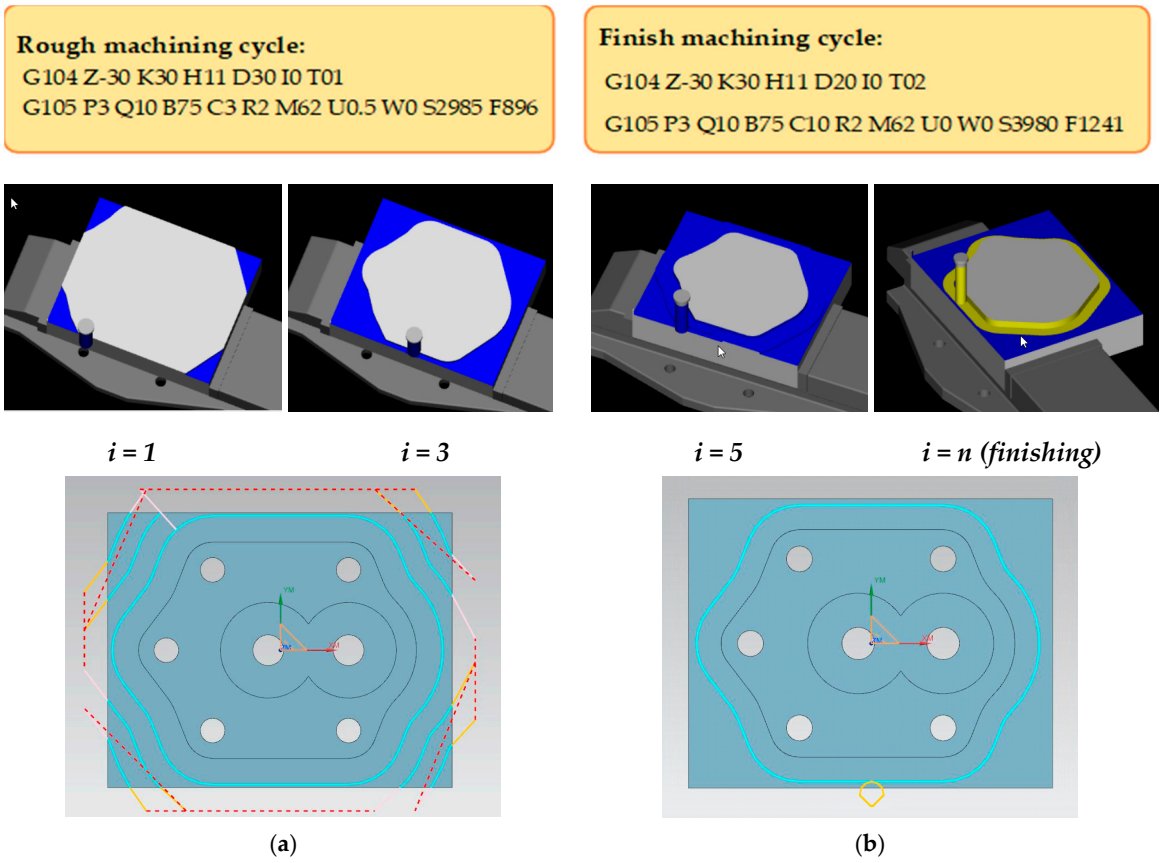


Figure 8. Cutting simulation for contour features in CNC control system using Fanuc 21: (**a**) rough machining toolpaths; (**b**) finish machining toolpaths.

Table 6. The profile extraction results from BKCAPP.

Edge	E49	E50	E51	E52	E53	E54	E55	E56	E57	E58	E59	E60	E45	E46	E47	E48
Point	P0	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
Type	0	2	1	1	1	2	1	1	1	2	1	2	1	2	1	2
X	64.29	45.85	41.84	25.33	16.74	15.15	25.33	41.84	45.85	64.29	124.08	140.28	167.18	168.32	144.93	128.21
Y	5	17.25	26.79	51.23	59.20	86.86	98.77	123.21	132.75	145	145	136.73	99.60	49.62	14.02	5
Z	–30	–30	–30	–30	–30	–30	–30	–30	–30	–30	–30	–30	–30	–30	–30	–30
R		20				20				20		20		44		20

In addition to G-code generation, BKCAPP also provides an algorithm for tool diameter selection in order to minimize the cutting time in rough machining on the basis of dimensional data, including the depth, width, and length of the feature, the minimal dimensions between walls, the minimal dimensions between a wall and island, and the dimensions between two islands. Table 7 indicates the toolsets that can be used for the rough machining of an example contour. The toolsets are different in terms of the tool diameter and number of tools needed from the tool database input for the operation. In the case of an example contour feature, the rough machining can involve toolsets with one, two, or three tools, as shown in Table 6. However, the tool diameter selection algorithm in BKCAPP shows that using one end mill measuring 30 mm for cutting results in a better cutting time, which was proven by simulation in CAM software, giving a total cutting time of 14.52 min.

Table 7. Relevant toolsets for rough contour machining.

Toolset	First Tool				Second Tool		Third Tool		Total Time (min)
	Diameter (mm)	rpm	f_z (mm/tooth)	Time (min)	Diameter (mm)	Time (min)	Diameter (mm)	Time (min)	
1	36	3139	0.1	17.39	-	-	-	-	17.39
2	34	3324	0.1	16.58	-	-	-	-	16.58
3	32	3531	0.1	15.53	-	-	-	-	15.53
4	30	3767	0.1	14.52	-	-	-	-	14.52
5	28	4036	0.1	16.46	-	-	-	-	16.46
6	26	4346	0.1	16.07	-	-	-	-	16.07
7	24	4708	0.1	15.21	-	-	-	-	15.21
8	22	5136	0.1	16.12	-	-	-	-	16.12
9	20	2944	0.11	25.29	-	-	-	-	25.29
10	18	3183	0.1	29.42	-	-	-	-	29.42
11	16	3680	0.09	32.06	-	-	-	-	32.06
12	14	4206	0.08	35.06	-	-	-	-	35.06
13	12	4907	0.071	39.14	-	-	-	-	39.14
14	62	1824	0.1	19.41	22	4.30	-	-	23.71
15	50	2261	0.1	21.37	22	3.25	-	-	24.62
16	40	2826	0.1	17.19	22	2.42	-	-	19.61
17	38	2975	0.1	19.31	22	2.18	-	-	21.49
18	50	21.37	0.1	21.37	38	1.33	22	2.52	25.22
19	62	19.41	0.1	19.41	38	3.11	22	2.44	24.96

The results show that the BKCAPP system created a complete connection for the integrated CAD-CAPP-CAM system, since it automatically generated the G-code files for use in CNC machines from input data as a 3D solid model with technical requirements. The result for the toolpath generation according to feature parameters shows an agreement with the toolpath formed in the commercial CAM software. This is not only applicable to simple CNC macro programming profiles, such as rectangular and circular shapes; the G-code conversion module also allows for G-code files to be generated for complex profiles. It is more convenient for an engineer to make any needed adjustments to G-code files that are based on CNC macro programming than interpolation toolpaths in commercial CAM software. The BKCAPP system also allows for customization with different input data, such as material, machine tool, and cutting tool library data, making it more suitable for practical applications. The BKCAPP system was also integrated with optimization problems in the cutting tool selection and operation sequence generation steps to achieve optimal machining time efficiency.

4. Conclusions

This paper described the novel integration of automatic geometrical and technical data extracted from a CAPP system into a G-code generation module. The following conclusions can be drawn from the system's implementation and from the results:

- BKCAPP is capable of feature recognition for parts with simple features, such as planes, holes, and square or circle pockets, as well as for complex contours. The extracted dimensions were the depth, width, and length of a feature or the point coordinates in the contour, and the geometrical relationships were accurate. Recognition rules determining the machining operations

for each machining feature on the basis of the extracted dimensions were adequate in a proper operation sequence;

- The G-code generation module using the macro programming method was efficient for 2.5D machining features with G-code files directly generated from CAPP data. With parametric programming, the toolpaths were concise and simple to follow and edit. This is not usually the case for toolpaths generated using CAM software with complex manual processes. BKCAPP also provides an algorithm for tool diameter selection to help reduce the machining time, which has not been provided in any other CAM software system;
- The G-code generation module can be used as a new G-code cycle for manual programming, can be combined with CAD–CAPP modules to form a completed CAD–CAPP–CNC integration system, or can be used in a CNC machine. In future research, we will focus on G-code generation for 3D complex curves and machining accuracy in comparison with a CAM output program.

Author Contributions: Conceptualization, T.K.N., L.X.P.; methodology, T.K.N., L.X.P.; validation, T.K.N., L.X.P., and N.-T.B.; formal analysis, T.K.N., L.X.P., and N.-T.B.; investigation, T.K.N., L.X.P.; writing—Original draft preparation, T.K.N., L.X.P., and N.-T.B. All authors have read and agreed to the published version of the manuscript.

Funding: This study was conducted with financial support from Hanoi University of Science and Technology (HUST) under project number T2018-PC-027.

Acknowledgments: The School of Mechanical Engineering at HUST is gratefully acknowledged for providing funding, guidance, and expertise. This work was also supported by the Centennial Shibaura Institute of Technology Action for the 100th anniversary of Shibaura Institute of Technology to enter the top ten Asian Institutes of Technology.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Bachtiak-Radka, E.; Dudzinska, S.; Grochala, D.; Berczynski, S.; Olszak, W. The influence of CNC milling and ball burnishing on shaping complex 3D surfaces. *Surf. Topogr. Metrol. Prop.* **2017**, *5*, 015001. [CrossRef]
2. Narooei, K.D.; Ramli, R. Application of artificial intelligence methods of tool path optimization in CNC machines: A review. *Res. J. Appl. Sci. Eng. Technol.* **2014**, *8*, 746–754. [CrossRef]
3. Scheer, A.W. *CIM Computer Integrated Manufacturing: Towards the Factory of the Future*; Springer Science and Business Media: Berlin, Germany, 2012.
4. Anderberg, S.; Beno, T.; Pejryd, L. CNC machining process planning productivity: A qualitative survey. In Proceedings of the International 3rd Swedish Production Symposium (SPS 2009), Göteborg, Sweden, 2–3 December 2009; pp. 228–235.
5. Kundrak, J.; Molnar, V.; Deszpoth, I. Comparative Analysis of Machining Procedures. *Machines* **2018**, *6*, 13. [CrossRef]
6. Yu, B.F.; Chen, J.S. Development of an Analyzing and Tuning Methodology for the CNC Parameters Based on Machining Performance. *Appl. Sci.* **2020**, *10*, 2702. [CrossRef]
7. Abas, M.; Salah, B.; Khalid, Q.S.; Hussain, I.; Babar, A.R.; Nawaz, R.; Khan, R.; Saleem, W. Experimental Investigation and Statistical Evaluation of Optimized Cutting Process Parameters and Cutting Conditions to Minimize Cutting Forces and Shape Deviations in Al6026-T9. *Materials* **2020**, *13*, 4327. [CrossRef]
8. Solarte-Pardo, B.; Hidalgo, D.; Yeh, S.S. Cutting Insert and Parameter Optimization for Turning Based on Artificial Neural Networks and a Genetic Algorithm. *Appl. Sci.* **2019**, *9*, 479. [CrossRef]
9. Kuntoğlu, M.; Aslan, A.; Pimenov, D.Y.; Giasin, K.; Mikolajczyk, T.; Sharma, S. Modeling of Cutting Parameters and Tool Geometry for Multi-Criteria Optimization of Surface Roughness and Vibration via Response Surface Methodology in Turning of AISI 5140 Steel. *Materials* **2020**, *13*, 4242. [CrossRef]
10. Xu, X.; Wang, L.; Newman, S.T. Computer-aided process planning: A critical review of recent developments and future trends. *Int. J. Comput. Integr. Manuf.* **2011**, *24*, 1–31. [CrossRef]
11. Al-Shebeeb, O.; Gopalakrishnan, B. Computer-aided Process Planning Approach for Cost Reduction and Increase in Throughput. In Proceedings of the International Conference on Operations Management (IOEM 2016), Detroit, MI, USA, 23–25 September 2016; pp. 632–644.

12. Yusof, Y.; Kamran, L. Survey on computer-aided process planning. *Int. J. Adv. Manuf. Technol.* **2014**, *75*, 77–89. [CrossRef]
13. Besharati-Foumani, H.; Lohtander, M.; Varis, J. Intelligent process planning for smart manufacturing systems: A state-of-the-art review. *Procedia Manuf.* **2019**, *38*, 156–162. [CrossRef]
14. Asiabanpour, B.; Mokhtar, A.; Hayasi, M.; Kamrani, A.; Nasr, E.A. An overview on five approaches for translating cad data into manufacturing information. *J. Adv. Manuf. Syst.* **2009**, *8*, 89–114. [CrossRef]
15. Hayasi, M.T.; Asiabanpour, B. Extraction of manufacturing information from design-by-feature solid model through feature recognition. *Int. J. Adv. Manuf. Technol.* **2009**, *44*, 1191–1203. [CrossRef]
16. Zhou, X.H.; Qiu, Y.J.; Hua, G.R.; Wang, H.F.; Ruan, X.Y. A feasible approach to the integration of CAD and CAPP. *Comput.-Aided Des.* **2007**, *39*, 324–338. [CrossRef]
17. Lau, H.C.W.; Lee, C.K.M.; Jiang, B.; Hui, I.K.; Pun, K.F. Development of a computer-integrated system to support CAD to CAPP. *Int. J. Adv. Manuf. Technol.* **2005**, *26*, 1032–1042. [CrossRef]
18. Tong, Y.F.; Li, D.B.; Li, C.B.; Yu, M.J. A feature-extraction-based process-planning system. *Int. J. Adv. Manuf. Technol.* **2008**, *38*, 1192–1200.
19. Abdelilah, E.; Ahmed, R.; Oussama, J. Optimized-automated choice of cutting tool machining manufacturing features in milling process. In Proceedings of the 11th World Congress on Computational Mechanics (WCCM 2014), Barcelona, Spain, 20–25 July 2014; pp. 747–761.
20. Ouyang, H.B. Intelligent cutting tool selection for milling based on STEP-NC machining features. *Appl. Mech. Mater.* **2014**, *635*, 589–593. [CrossRef]
21. Amaitik, S.M.; Kiliç, S.E. An intelligent process planning system for prismatic parts using STEP features. *Int. J. Adv. Manuf. Technol.* **2007**, *31*, 978–993. [CrossRef]
22. Manafi, D.; Nategh, M.J.; Parvaz, H. Extracting the manufacturing information of machining features for computer-aided process planning systems. *J. Eng. Manuf.* **2017**, *231*, 2072–2083. [CrossRef]
23. Hasan, M.A. A Conceptual Framework of Common Variables in CNC Machines Programming for Fanuc Custom Macros. *J. Mater. Sci. Mech. Eng.* **2016**, *3*, 250–253.
24. Joshi, V.; Desai, K.; Raval, H. Machining of Archimedean spiral by parametric programming. *Int. J. Mod. Manuf. Technol.* **2016**, *8*, 25–30.
25. Phung, L.X.; Van Tran, D.; Hoang, S.V.; Truong, S.H. Effective method of operation sequence optimization in CAPP based on modified clustering algorithm. *J. Adv. Mech. Des. Syst. Manuf.* **2017**, *11*, 1–12. [CrossRef]