




User-Defined Machining Feature Recognition Based on Semantic Reasoning for B-rep Models

Yingzhong Zhang¹ , Fei Wang² and Tianbao Chen³

¹ School of Mechanical Engineering, Dalian University of Technology, zhangyz@dlut.edu.cn

² School of Mechanical Engineering, Dalian University of Technology, 1755257660@qq.com

³ School of Mechanical Engineering, Dalian University of Technology, a13271213579@163.com

Corresponding author: Yingzhong Zhang, zhangyz@dlut.edu.cn

Abstract. Machining features play a crucial role in the integration of CAD and CAM. Owing to the diversity of product structures, machining methods, equipment and tools, it is necessary to develop an open, shareable and extensible machining feature recognition methods. This paper employs ontology-based semantic technology and proposes an open knowledge-driven machining feature recognition method. Using ontology and knowledge rules users can define the machining features to be recognized according to specific shape structures, machining methods, and tools. The recognition system converts the part B-rep model into a set of machining face knowledge graphs labeled by ontology, and then performs semantic reasoning for the created knowledge graphs according to the defined knowledge rules. Finally a set of machining features are recognized based on reasoning results. The presented recognition approach has been realized using C++ programming in NX system. The testing results of case study demonstrate feasibility and good openness of the presented method.

Keywords: Machining feature recognition, User-defined feature, Ontology, Semantic reasoning.

DOI: <https://doi.org/10.14733/cadaps.2023.763-785>

1 INTRODUCTION

Machining features are a set of abstract shapes with machining semantics on parts, such as holes, boss, and slots, are associated with some manufacturing activities, and play a crucial role in the integration of CAD and CAM as they carry high-level information that can effectively link design and manufacturing [1]. Since 1980s, machining feature recognition has been a hot research direction in industry and academia. At present, many feature recognition approaches [2] have been proposed, which can mainly be classified into two kinds: logic symbol based approaches and data driven approaches. However, currently there are still many issues and challenges in machining feature recognition. Excepting the issues on recognition efficiency, robustness and recognition of interacting features, most recognition systems are closed, namely, they can only recognize the features defined

within in the system. In practice, due to different product structures, machining equipment and tools, it is difficult to define all machining features in a feature recognition system. It is necessary to make the recognition system open so as to recognize user-defined machining features.

From the current recognition approaches, it is necessary to recognize user-defined machining features with an opening way. On the one hand, the essence of features is related to a specific application domain, including machining shape, equipment and tools. On the other hand, the definition of features and recognition results need to be commonly understood, shared, and passed on to various subsequent applications with heterogeneous systems. An open, scalable, and shared feature representation and recognition mechanism are required.

To effectively solve the above issues, it is necessary to apply the semantic technologies to the recognition of machining features because machining features are associated with considerable machining knowledge [3] and shape semantics. Unfortunately, from the public literature, few works have considered the semantic recognition of machining features. There are still some shortage and limitations on recognition of user-defined machining.

Based on above analysis, this paper aims at the openness of feature recognition, focuses on the recognition for user-defined machining features, and presents an open machining feature recognition framework for solid B-rep models using semantic representation and reasoning technology. The main contribution includes: (1) a novel recognition approach and framework based on ontology and knowledge rules is presented, by which all features to be recognized can be customized by users and the recognition system implements feature recognition in accordance with the user-defined feature semantic representation; (2) a satisfiability reasoning strategy and method combining concept matching of instances and geometry computation of instances are proposed.

2 RELATED WORK

Due to the importance of machining features, the recognition of machining features attracts a lot of attention. A large number of recognition approaches have been proposed. Comprehensive reviews of the existing recognition approaches have been provided in [1-2, 4]. Here, we limit our review of previous works to main types of machining feature recognition methods, semantic methods of feature recognition and recognition of user-defined machining features.

2.1 Recognition for Machining Features

According to the published literature, up to now the main approaches can be classified into two categories: the symbolic reasoning based approach and the data-driven approach. The following briefly reviews the characteristics and existing issues on the two approaches.

The symbolic reasoning based approach is the first proposed approach [2, 4], which includes graph-based approaches [5], hint-based approaches, hybrid approaches, and volumetric decomposition approaches. Joshi and Chang [5] were the first to propose the attributed adjacency graph (AAG) technique for feature recognition. Using graph-based approaches the part's faces can be represented as an AAG and the graph is analyzed to find interesting sub-graphs as machining features. The graph-based approaches are able to effectively recognize non-interacting features but have problems in recognizing interacting features. Numerous efforts have been made to enhance the capability of addressing the feature interaction problem. For example, hint-based recognition approaches were proposed to deal with interacting features. Furthermore, hybrid approaches combining graph-based approaches with hint-based approaches or heuristic rules are employed. However, there are still some issues and challenges in the logical rule-based approach. For example, as feature interactions may cause topology variations among geometric entities, recognizing interacting machining features from complex B-rep models is still a difficult task. In addition, the recognition performance in scalability, reusability, and sharing is poor. An effective mechanism for combining feature recognition with knowledge representation is still lacking.

In recent years, deep learning has achieved great success in the field of computer vision. Among them, convolutional neural network (CNN) has been widely used because of its strong feature extraction ability [6]. The data-driven recognition approach using 3D CNN has made preliminary development in feature recognition, which provides a new direction for solving the problem of machining feature recognition. The input of typical convolution network is two-dimensional, three-dimensional and even multi-dimensional matrix data, such as image format and 3D voxels. Based on 3D voxel mesh data, Zhang et al. [7] proposed a machining feature recognition method using a 3D convolution neural network model: FeatureNet. Shi et al. [8] constructed a feature recognizer: MsvNet by using the multi view methods. 3D point cloud is a special CAD coding model. Based on the research of point clouds, Qi et al. [9] proposed a convolutional neural network model: PointNet for three-dimensional point cloud data. The network model overcomes the disadvantages of disorder and geometric transformation invariance of the input point cloud, and directly takes the point cloud as the data input to extract the data features. Ma et al. [10] proposed a convolution neural network structure for machining feature recognition based on PointNet. However, it is mainly oriented to the cube model with a single feature. Although the data-driven recognition approach is a hot research topic, there is still a long way to go in practical applications.

2.2 Feature Recognition based on Ontology and Semantics

As machining features are associated with considerable machining knowledge and shape semantics, it is necessary to use the semantic technologies to the recognition of machining features. Recent years, semantic technologies based on ontology are used to feature recognition and feature design. Stefano et al. [11] proposed a method for the recognition of the semantics of the features of a component from a geometric representation. In their work, the concept of the semanteme is introduced. Their approach gives a wide flexibility for generic feature recognition, but does not combine concepts, rules, and knowledge reasoning. Wang and Yu [12] proposed an ontology-based feature recognition framework. In their presented recognition framework, features are captured transparently and hierarchically represented with a formal Web Ontology Language (OWL) [13], and the feature recognition is achieved by applying an efficient backward-chained ontology inference. Their work demonstrates a high level of flexibility and explainability for both representing and recognizing features. However, their work still lacks analysis and representation for the semantics of the geometric shape and machining processes. Zhang et al. [14] developed a semantic approach to the automatic recognition of machining features. In their approach an ontology-based concept model for machining features and a recognition approach using semantic reasoning are proposed.

Sanfilippo and Borgo [15] reviewed the state of art of feature-based modeling approaches by concentrating on how features are conceptualized. Gupta et al. [16] proposed a feature-based ontological framework for semantic interoperability in product development. Mandorli et al. [17] proposed an ontological approach that achieves a newer level of semantic representation of features, and enhance the cognitive understanding of the final CAD model. Sanfilippo [18] proposed an ontological approach to carry out feature-based product modelling.

2.3 Recognition of User-defined Machining Features

In practice, due to different product structures, machining equipment, tools and machining experience of engineers, it is necessary to employ the user-defined machining features. A few user-defined feature recognition approaches have been developed as follows:

Gaines and Hayes [19] considered that a feature recognizer should be able to produce different feature decompositions when different tools are available, and presented a customizable feature recognizer, which accepts user-defined cutting tools as input and automatically identifies features that can be cut using the custom tools. Li and Shah [20] proposed a method to automatic recognition of user-defined turning features on mill/turn parts. The presented method supports user-defined turning features that are represented using N-REP, a neutral feature representation language. Liu et al. [21] proposed an approach that the machining features are defined by users according to the factors of enterprise manufacturing resources, structure of parts, and programming preference of

process engineers. Ma et al. [22] presented a customizable process planning approach for rotational parts based on multi-level machining features and ontology. In their presented methods, the high-level custom features are defined based on the directed acyclic graph.

From the current recognition approaches, it is difficult to recognize user-defined machining features with an opening way. On the one hand, the essence of features is related to a specific application domain, including machining shape, equipment and tools. At present, there is a lack of a formal method to describe the semantics of machining features. On the other hand, the recognition method relies on internal algorithms and lacks the reasoning for external semantic rules.

3 OVERVIEW FOR THE PRESENTED APPROACH

3.1 Requirements for User-defined Machining Feature Recognition

Machining features can be defined as a group of shapes with specific machining semantics on parts. In different machining contexts, different machining features are defined. In this paper, the discussed machining features are mainly formed by cutting surfaces of parts, such as holes, slots, and pockets. Hence, a machining feature can be defined as a set of surfaces that are formed by removing the material volume in a single operation with a single tool on a single machine [23]. The single operation denotes that the specific shape can only be obtained by a single machining operation. For example, a hole may require drilling and boring operations, but a drilling operation can achieve the basic shape of holes.

Currently, most feature recognition systems recognize the predefined feature. However, different users might have their own interpretation of a part in terms of machining features, depending on their equipment and tools to be used, their experiences and application. Figure 1 shows four types of machining features with different interpretations.

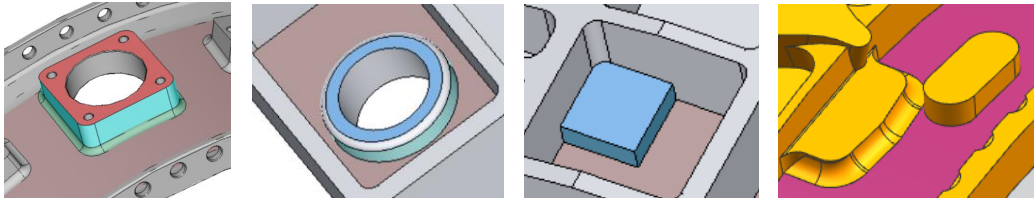


Figure 1: Various forms of boss features: (a) Island form boss, (b) Circular island form boss, (c) Step form boss, and (d) Oblong boss.

Hence, a user-defined feature recognition system should allow users to define new features according to their needs. There are the following requirements for user-defined machining feature recognition:

- Open definition mode

The open definition mode is that the user can define the machining features to be recognized through text methods such as natural language or rules, and the recognition system can correctly recognize them without modifying any recognition programs.

- Machining feature library

For a user-defined feature recognizer, there must be a mechanism for users to define their own features and place them in a library. In subsequent feature recognition, the feature type to be recognized can be selected according to the needs of process planning. For example, in the turning process planning, only turning features are needed to be recognized. At the same time, the feature library can also manage user-defined features, such as adding new user-defined features into the library, and modifying or deleting features from the library.

- Feature recognition based on semantic reasoning for user's definitions

As mentioned above, when users define new machining features, the recognition system should recognize the newly defined features without modifying any recognition programs. Hence, it is required to use new recognition mechanism and methods. Semantic representation and reasoning provide a new solution to addressing above issues.

3.2 Recognition Framework

In order to realize the recognition for user-defined machining features, we propose a semantic recognition framework as shown in Figure 2. The recognition framework consists of the following two layers:

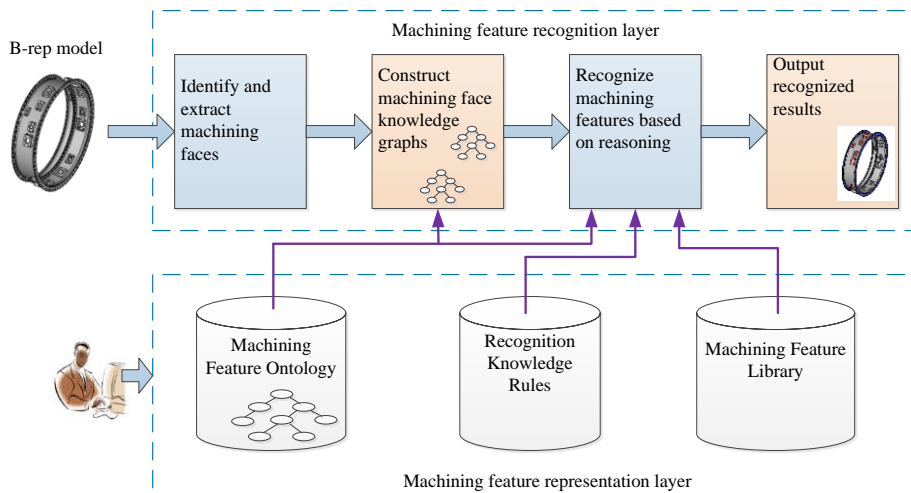


Figure 2: The framework of the presented approach.

- Machining feature representation layer

This layer includes three parts: (1) machining feature ontology, (2) recognition knowledge rules, and (3) a machining feature library. The machining feature ontology defines all the machining feature concepts and other concepts involved in the recognition, which are defined as ontology classes by protégé tool [24]. Recognition knowledge rules are defined with SWRL (Semantic Web Rule Language) [25], which provide recognition knowledge and help recognition reasoning. The machining feature library consists of a set of defined machining features, including user-defined features. The machining feature concept is defined in the feature ontology and the shape characteristic of each machining feature is defined using SWRL rules. As a result, the machining feature representation layer can be independent of the recognition layer, and users can define machining features by themselves according to requirements through protégé tool.

- Machining feature recognition layer

In the machining feature recognition layer, the B-rep model of the part to be recognized is input and all machining faces are identified and extracted from the B-rep model. Machining Face Knowledge Graphs (MFKGs) are constructed, in which a machining face is defined as a node of the graph, and the adjacency relation and spatial relation between two faces are defined as edges of the graph. The adjacency relation includes convex adjacency, concave adjacency, and blend adjacency. Then the constructed MFKGs are implemented semantic reasoning according to user-defined knowledge rules for machining features. A set of user-defined machining features can be recognized.

4 SEMANTIC REPRESENTATION FOR MACHINING FEATURES

4.1 Ontological Definition for Machining Faces

4.1.1 Machining face concepts

As mentioned above, machining features are a set of abstract shape with machining semantics on parts. The abstract shape with machining semantics can be referred as a set of surfaces that are formed by removing the material volume [23]. Hence, we give the following definition:

Definition 1: The geometric surface that is formed by removing the material volume is defined as the machining face.

A machining face corresponds to one or more geometric surfaces. Due to different cutting tools and different machining motion modes, machining faces with different characteristics will be formed. For example, the bottom face and side faces are generated by milling cutters, and the inner hole faces are formed by drilling operations. Figure 3 provides an illustration for the process of milling an open slot feature. In fact, the open slot feature includes two machining faces: a bottom face and a side face. The side face consists of two geometric surfaces but it is formed by the side edge of the milling cutter at one time.

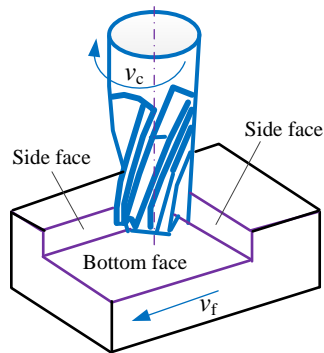


Figure 3: Illustration for the process of milling an open slot feature.

Obviously, from the point of view of the formation of machining faces and manufacturing there are the following characteristics on machining faces:

- The concave geometric surfaces are generally machining faces.
- Geometric surface adjacent to inner loops are machining faces.
- A machining face corresponds to one or more geometric surfaces.
- The shape of machining faces is related to the shape of cutting tools and cutting methods.
- A machining face can extend across several geometric faces.
- A set of adjacent machining faces form a machining face chain.

4.1.2 Ontological definitions for machining faces

From the above analysis on machining faces, we can see that the machining face definition comes from the point of view of engineering. In order to give an open and common understanding definition for machining faces, this paper employs Protégé tool to define the ontology of machining face concepts as shown in Figure 4. The following will elaborate the machining face classes and their hierarchical relationships.

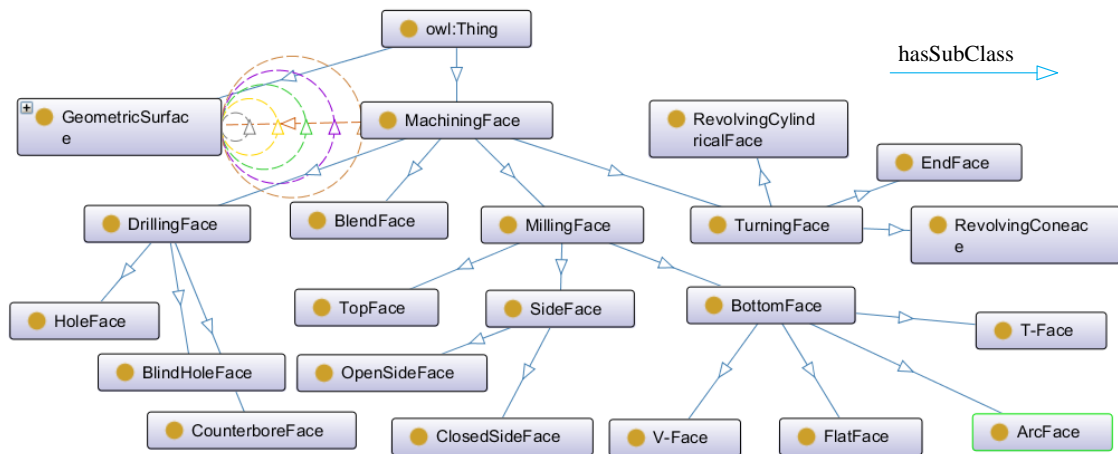


Figure 4: A part of class definition for machining faces.

- The *MachiningFace* class

In general, the forming methods by cutting mainly include drilling, milling and turning. Grinding and boring operation usually do not change the shape greatly. As a result, the *MachiningFace* class is defined as the top class of machining faces. The *DrillingFace* class, the *MillingFace* class and the *TurningFace* class are defined as the subclasses of the *MachiningFace* class, which represent the geometric surfaces formed by drilling, milling and turning respectively.

- The *MillingFace* class

Milling generally processes planes, slots, pockets, bosses, etc. The shape formed by milling operations can mainly be classified as three types: bottom faces, side faces and top faces. The bottom face is generally formed by end blades of a milling cutter, so it is required to be completely accessible from its normal direction (i.e., completely accessible for machining). The bottom face and its adjacent side faces form the shape of some machining features. Hence, the bottom face is an important machining face concept. We employ the *BottomFace* class to represent this type of machining face. At the same time, according to the geometric surface type related by the bottom face, the *BottomFace* class also derives four subclasses: the *FlatFace* class, the *CylindricFace* class, the *V-shapedFace* class, and the *T-shapedFace* class.

The side face is generally formed by side blades of a milling cutter. As the tool moves, side blades of a milling cutter remove materials of workpieces and the shape of side faces are formed. As a result, a side face generally consists of one or more geometric surface, and sometimes two parallel sides are formed. We employ the *SideFace* class to represent this type of machining faces.

- The *DrillingFace* class

Drilling generally processes holes whose shape is generally simple. We define the *DrillingFace* class to represent the machining face formed by drilling. The *BottomHoleFace* class, the *HoleFace* class, and the *CounterBoreFace* class are defined as the subclasses of the *DrillingFace* class.

- The *TurningFace* class

Turning generally processes shafts, ring slots, grooves, end faces, etc. The geometric surface of a shaft is generally revolving surfaces. As a result, we define the *TurningFace* class to represent the machining face formed by turning. The *ShaftFace* class, the *GrooveFace* class, and the *RingFace* class are defined as the subclasses of the *TurningFace* class.

4.1.3 The geometric relationships between machining faces

The B-rep model of a part provides its all geometric and topological information, which are the basis to search machining faces. In order to represent the relationship between machining faces and their geometric surfaces, we also employ ontology to abstract and define geometric concepts and their relations.

The *GeometricFace* class is employed to represent the geometric face concept. More specifically, a geometric face concept is described by specific geometric surfaces, such as planes, cylindrical surfaces, spheres, and spline surfaces. Accordingly, the *Plane* class, the *CylindricalSurface* class, the *Sphere* class, and the *SplineSurface* class are defined as the subclasses of the *GeometricFace* class. The object property "hasSurface" of the *MachiningFace* class is used to represent the relationship between machining faces and geometric faces.

The topological and spatial relationships between two geometric faces are important shape characteristics of machining feature. We define an object property set as follows:

$$R = \{adjoinTo\{concaveAdjoin\{concaveTangent\}, convexAdjoin\{convexTangent\}\}, parallelTo, coplanarTo, perpendicularTo, coaxialTo\}.$$

On the above geometric relation set, *adjoinTo*, *parallelTo*, *perpendicularTo*, and *coaxialTo* denotes the adjacent relation, the parallel relation, the perpendicular relation, and the coaxial relation between two machining faces, respectively. At the same time, according to the shape characteristics of the two adjacent faces, the *adjoinTo* property is classified into two sub-properties: the *concaveAdjoin* property and the *convexAdjoin* property, which are used to represent concave adjacency and convex adjacency relations between two faces. Furthermore, the *concaveAdjoin* property derives the *concaveTangent* sub-property and the *convexAdjoin* property derives the *convexTangent* property. Here, the *concaveTangent* property denotes that two faces are tangent.

4.2 Ontological Definition for User-defined Machining Features

In order to carry out the open definition for user-defined features, it is necessary to use ontological definition for machining features. The ontological definition of machining features is conducted as the following steps.

In general, concept is a kind of abstract forms to represent entities and relationships. Concept is represented by terminology. In order to define a machining feature, the first task is to define a concept (or term) for the feature to be defined. For the consistency and comprehensibility of feature terminology, using open ontology definition is a feasible solution.

First, we define the *MachiningFeature* class to represent the machining features, which is a top base class. According to the characteristics of machining processes, in this paper, the machining feature is divided into four categories and they are defined as the subclass of the *MachiningFeature* class. The four subclasses are the *DrillingFeature* class, the *TurningFeature* class, the *MillingFeature* and the *BlendFeature* class, which are used to represent the machining features formed by drilling, turning, milling, and blend operations, respectively. Each subclass can continually derive its subclasses. For example, the *MillingFeature* class derives four subclasses: the *Slot* class, the *pocket* class, the *Boss* class and the *Step* class. As a result, a concept hierarchy of machining features is formed as shown in Figure 5.

In the *MachiningFeature* class, an object property *hasFaces* is defined to relate geometric surfaces. In addition, dependency information and feature attribute information (such as size parameters) of machining features are also defined by object properties and data properties. Due to paper space, this paper will not elaborate them.

Since the definition of machining features is represented in OWL/XML text format, it is easy for users to add their own defined machining features into the hierarchical concept structure. For example, when users have more detailed requirements for slot features, they can define the new machining feature as the child feature of the slot feature, such as adding through slots, blind slots, V-shaped slots.

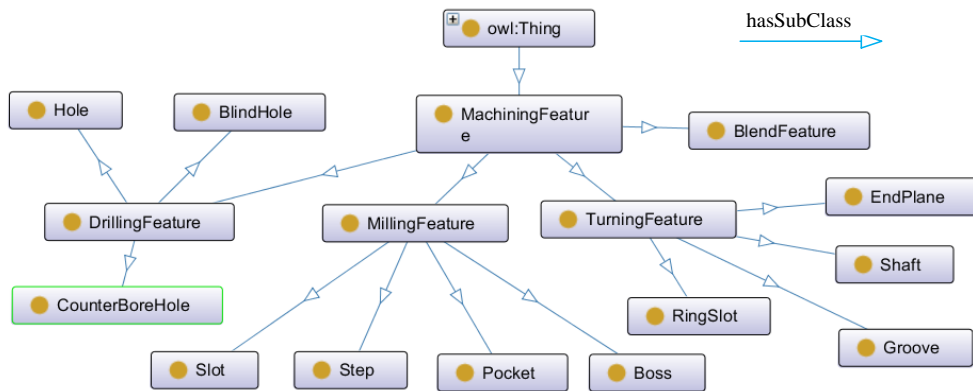


Figure 5: Hierarchy of a part of machining feature concepts.

4.3 Semantic Rules for Machining Features

The ontological concept definition of machining features only provides an explicit definition on their terminologies. Their shape and engineering semantics also need to be explicitly defined so as to recognize machining features from the B-rep model of the part.

As mentioned above, the shape characteristics of the machining faces determine the varieties of machining features. According to above analysis, a machining feature is determined by a set of machining faces and their topological and spatial relations.

The combination of the machining faces is the condition to identify a machining feature. Hence, we can employ SWRL rules to represent machining features. An SWRL reasoning rule can be expressed in the form: Antecedent→Consequent, and the antecedent part and the consequent part can also be represented as the conjunctive formula of atoms: $a_1 \wedge a_2 \wedge \dots \wedge a_n$ and $b_1 \wedge b_2 \wedge \dots \wedge b_m$, respectively. Nevertheless, the relationship between machining faces and machining features is a many to many relationship. In order to represent the mapping knowledge between machining faces and machining features, this paper proposes to map the combination of machining faces to a concept of machining features. The form of defining SWRL rules is as follows:

$$C_1^1 \wedge C_2^1 \wedge C_3^1 \dots C_m^1 \rightarrow M_1,$$

.....,

$$C_1^k \wedge C_2^k \wedge C_3^k \dots C_n^k \rightarrow M_k.$$

In the above formula, M_k represents a newly defined machining feature, and atom C represents a set of machining face combinations which are in either of the forms $C(?x)$ or $P(?x, ?y)$. In the atom combination, if x is an instance of class C , then $C(?x)$ holds; if x is related to y by property P , then $P(?x, ?y)$ holds. The defined machining feature rule base can be represented as:

$$r = \{M_1, M_2, \dots, M_m\}.$$

For example, an island boss feature for milling has a bottom face and a set of closed side faces, which can be represented as the following rule:

$$\text{MachiningFeature}(?x) \wedge \text{hasFace}(?x, ?s) \wedge \text{InnerClosedFace} (?s) \wedge \text{hasFace} (?x, ?b) \wedge \text{BottomFace} (?b) \wedge \text{concaveAdjoin} (?b, ?s) \rightarrow \text{Boss}(x).$$

In the above rule, atom *BottomFace* and *InnerClosedFace* are machining face concepts, which denote a bottom face and a closed side face chain (including a set of closed side faces adjoined to each other), respectively. The side faces adjoin the bottom face concavely.

4.3.1 Classification rules for drilling faces

Drilling operations mainly produce cylindrical surfaces. The geometric characteristic of machining faces and the spatial relation between them determine that a machining face should be classified to which types of specific machining faces. First we define the knowledge rules with SWRL for drilling faces as shown in Table 1.

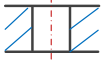
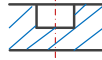
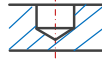
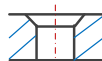
No.	SWRL rule	Explanation
1	$\text{MachiningFace}(\?x) \wedge \text{hasSurface}(\?x, \?s) \wedge \text{InnerCylinder}(\?s) \rightarrow \text{HoleFace}(\?x)$	 A through hole surface
2	$\text{MachiningFace}(\?x) \wedge \text{hasSurface}(\?x, \?s) \wedge \text{InnerCylinder}(\?s) \wedge \text{concaveAdjoin}(\?x, \?y) \wedge \text{hasSurface}(\?y, \?u) \wedge \text{Plane}(\?u) \rightarrow \text{BlindFace}(\?x)$	 Two blind hole surfaces: a cylinder surface and a plane or cone surface
	$\text{MachiningFace}(\?x) \wedge \text{hasSurface}(\?x, \?s) \wedge \text{InnerCylinder}(\?s) \wedge \text{concaveAdjoin}(\?x, \?y) \wedge \text{hasSurface}(\?y, \?u) \wedge \text{Cone}(\?u) \rightarrow \text{BlindFace}(\?x)$	
3	$\text{MachiningFace}(\?x) \wedge \text{hasSurface}(\?x, \?s) \wedge \text{InnerCone}(\?s) \wedge \text{convexAdjoin}(\?x, \?y) \wedge \text{hasSurface}(\?y, \?u) \wedge \text{InnerCylinder}(\?u) \rightarrow \text{CounterBoreFace}(\?x)$	 Two surface: a cone surface and a cylinder surface

Table 1: A part of SWRL rules defining drilling faces.

In Table 1, "hasSurface" is an object property of "MachiningFace" class, which relates a machining face with its geometric faces. "InnerCylinder" is a geometric concept. "InnerCylinder(?s)" denotes that variable s is an inner cylindrical surface, which can be completed by a simple geometric test.

4.3.2 Classification rules for milling faces

According to the above analysis, milling faces are mainly classified into bottom faces and side faces. Side faces usually concavely adjoin to a bottom face. The bottom face is completely accessible from its normal direction (i.e., the machining tool is fully accessible). The SWRL knowledge rules for milling are defined in Table 2. There are the following geometric atom concepts:

- The *AccessibleFace* concept

"AccessibleFace (?s)" denotes that variable s is a surface to which machining tool can be fully accessible. In order to implement this concept test, we construct an accessible test function for the geometric surface. In this test function, some geometric computations are needed and if machining tool can be fully accessible the surface, this function returns true else false.

- The *MaxArea* concept

"MaxArea (?s)" denotes that variable s is a surface whose area is the largest than the other adjacent surfaces.

- The *SideFaceChain* concept

"SideFaceChain(?s)" denotes that variable s is a side face chain that consists of a set of side faces adjoined to each other. The *SideFaceChain* concept has two sub-concepts: the *ClosedSideFaceChain* concept and the *OpenSideFaceChain* concept. As shown in Figure 6(a), the *ClosedSideFaceChain* concept is also classified into the *InnerClosedFace* concept and the *OuterClosedFace* concept, which denotes that the closed side faces surround the body of the part and don't surround, respectively. The *OpenSideFaceChain* concept is also classified into the *OneSideFace* concept and the *TwoSideFace* concept. The *OneSideFace* concept denotes the side face chain has one side and the *TwoSideFace* concept denotes the side face chain has two sides as shown in Figure 6(b). If there are two opposite parallel side faces in a side face chain, the side face chain is considered as an instance of the *TwoSideFace* class.

No.	SWRL rule	Explanation
1	$\text{MachiningFace}(?x) \wedge \text{AccessibleFace}(?x) \wedge \text{hasSurface}(?x, ?s) \wedge \text{MaxArea}(?s) \rightarrow \text{BottomFace}(?x)$	If a machining face is accessible, and has a maximum area, the face is a bottom face.
2	$\text{MachiningFace}(?x) \wedge \text{concaveAdjoin}(?x, ?y) \wedge \text{BottomFace}(?y) \rightarrow \text{SideFace}(?x)$	If a machining face concavely adjoins a bottom face, the face is a side face.
3	$\text{SideFace}(?x) \wedge \text{SideFace}(?y) \wedge \text{adjoinTo}(?x, ?y) \wedge \text{differsFrom}(?x, ?y) \rightarrow \text{SideFaceChain}(?c) \wedge \text{addFace}(?c, ?x)$	If a side face adjoins to another side face, a side face chain is created and the face is added into the chain.
4	$\text{MachiningFace}(?x) \wedge \text{MachiningFace}(?y) \wedge \text{adjoinTo}(?x, ?y) \wedge \text{differsFrom}(?x, ?y) \rightarrow \text{FaceChain}(?c) \wedge \text{addFace}(?c, ?x)$	If a machining face adjoins to another machining face, a face chain is created and the machining face is added into the face chain.
5	$\text{FaceChain}(?x) \wedge \text{NoOtherAdjacent}(?x) \rightarrow \text{NoBottomFaceChain}(?x)$	If face chain doesn't adjoin any other machining faces, the face chain is considered as a no bottom face chain.
6	$\text{SideFaceChain}(?x) \wedge \text{ClosedTest}(?x) \rightarrow \text{ClosedSideFaceChain}(?x)$	If a side face chain is closed, the side face chain is relabeled as type of <i>ClosedSideFaceChain</i> class.
7	$\text{SideFaceChain}(?x) \wedge \text{OpenTest}(?x) \rightarrow \text{OpenSideFaceChain}(?x)$	If a side face chain is open, the side face chain is relabeled as type of <i>OpenSideFaceChain</i> class.

Table 2: A part of SWRL rules defining milling faces.

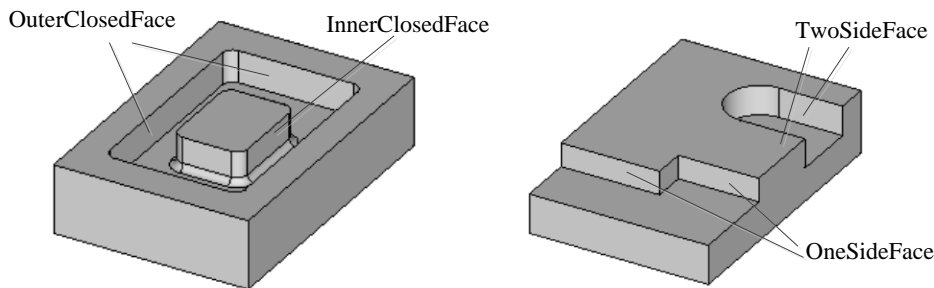


Figure 6: Illustration for four side face concepts: (a) The *OuterCloseFace* concept and *InnerClosedFace* concept, (b) The *OneSideFace* concept and *TwoSideFace* concept.

- The *NoBottomFaceChain* concept

The *NoBottomFaceChain* concept denotes that a face chain that consists of a set of machining faces adjoined to each other doesn't concavely adjoin to any other machining faces. The *NoBottomFaceChain* concept is also classified into the *OpenNoBottomFaceChain* concept and the *ClosedNoBottomFaceChain* concept.

- The *ClosedTest* concept and the *OpenTest* concept

The *ClosedTest* concept and the *OpenTest* concept are two geometric test functions, which are used to test whether a face chain is closed or open.

In addition, due to feature interaction a side face may be split into two or multiple faces. As a result, we use the spatial relation information among side faces and related geometric and topological information to combine some machining faces in a same MFKG into a milling face.

4.3.3 Semantic rules for machining features

The machining feature ontology provided above only defines the concepts or terms and their hierarchical relationships. The shape and engineering semantics of machining features are also needed to be explicitly defined. As mentioned above, the semantics of machining features are determined by machining faces. As a result, we can use SWRL to structure various combination of machining faces so as to explicitly represent machining features. Table 3 shows a part of SWRL semantic rules for machining features.

No	SWRL rule	Illustration
1	$\text{MachiningFeature}(\?x) \wedge \text{hasFace}(\?x, \?s) \wedge \text{InnerClosedFace}(\?s) \wedge \text{hasFace}(\?x, \?b) \wedge \text{BottomFace}(\?b) \wedge \text{concaveAdjoin}(\?b, \?s) \rightarrow \text{Boss}(\?x)$	
2	$\text{MachiningFeature}(\?x) \wedge \text{hasFaces}(\?x, \?b) \wedge \text{BottomFace}(\?b) \wedge \text{hasFace}(\?x, \?s) \wedge \text{TwoSideFace}(\?s) \wedge \text{concaveAdjoin}(\?b, \?s) \rightarrow \text{BlindSlot}(\?x)$	
3	$\text{MachiningFeature}(\?x) \wedge \text{hasFaces}(\?x, \?a) \wedge \text{CounterBoreFace}(\?a) \wedge \text{hasFaces}(\?x, \?b) \wedge \text{CounterBoreFace}(\?b) \wedge \text{differsFrom}(\?a, \?b) \wedge \text{convexAdjoin}(\?a, \?b) \rightarrow \text{CounterBoreHole}(\?x)$	
4	$\text{MachiningFeature}(\?x) \wedge \text{hasFaces}(\?x, \?b) \wedge \text{BottomFace}(\?b) \wedge \text{hasFace}(\?x, \?s) \wedge \text{OuterClosedFace}(\?s) \wedge \text{concaveAdjoin}(\?b, \?s) \rightarrow \text{Pocket}(\?x)$	
5	$\text{MachiningFeature}(\?x) \wedge \text{hasFaces}(\?x, \?b) \wedge \text{BottomFace}(\?b) \wedge \text{hasFace}(\?x, \?s) \wedge \text{SideFace}(\?s) \wedge \text{concaveAdjoin}(\?b, \?s) \wedge \text{hasFace}(\?x, \?w) \wedge \text{SideFace}(\?w) \wedge \text{concaveAdjoin}(\?b, \?w) \wedge \text{differsFrom}(\?s, \?w) \wedge \text{parallelTo}(\?s, \?w) \rightarrow \text{ThroughSlot}(\?x)$	

Table 3: A part of SWRL rules defining machining features.

4.3.4 Transition rules for adjacency relations

Because some shapes cannot be machined at one time, other machining features are also needed. For example, in the slot processing, there is often a fillet transition between the side face and the bottom face. The fillet transition forms the fillet feature, but the adjacent relationship between the side face and the bottom face is cut off. In order to maintain the adjacency relationship, this paper proposes a transition rule method by which in the semantic reasoning recognition system can identify this type of adjacency relations correctly. Table 4 provides two transition rules:

No	SWRL rule	Illustration
1	$\text{MachiningFace}(\?x) \wedge \text{concaveAdjoin}(\?x, \?f) \wedge \text{concaveAdjoin}(\?f, \?b) \rightarrow \text{concaveAdjoin}(\?x, \?b)$	

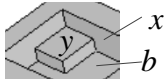
2	$\text{MachiningFace}(?x) \wedge \text{concaveAdjoin}(?x, ?f) \wedge \text{MillingFeature}(?y)$ $\wedge \text{hasFace}(?y, ?f) \wedge \text{hasFace}(?y, ?s) \wedge \text{concaveAdjoin}(?s, ?b)$ $\rightarrow \text{concaveAdjoin}(?x, ?b)$	
---	--	---

Table 4: A part of SWRL rules defining machining features.

4.4 Machining Face Knowledge Graphs of the B-rep Model

4.4.1 The definition of the machining face knowledge graph

The knowledge graph provides a formal and feasible solution to search or find some things of interest from a set of related facts [26]. The geometric faces in a part B-rep model are the known facts on the shape of the part. Furthermore, according to the semantic representation of machining features presented above, it is required to identify and extract the machining faces from the part B-rep model. Geometric face facts can be converted into fewer machining face facts. Hence, we formally model the machining faces of a part into a set of MFKGs.

The MFKG can be represented as a directed graph: $G = \langle V, E \rangle$, where V denotes a set of machining face nodes and E denotes a set of relation edges that link two nodes. A face node corresponds to a machining face. A relation edge corresponds to a type of topological or spatial relations between two machining faces.

According to the topological/spatial relations defined above, we categorize the adjacency relation into five types: *adjoinTo*, *concaveAdjoin*, *convexAdjoin*, *concaveTangent*, and *convexTangent*. The spatial relations are categorized into: *parallelTo*, *coplanarTo*, *coaxialTo*, etc.

Figure 7 provides an illustration on MFKGs. Obviously, the concave adjacency relation and the convex adjacency relation are the most important relationships between two faces, which depend on the concavity and convexity of the edge that connects the two faces and can describe the important shape characteristics of machining features. The following section will elaborate the judgement of the concavity and convexity of an edge that connects two faces.

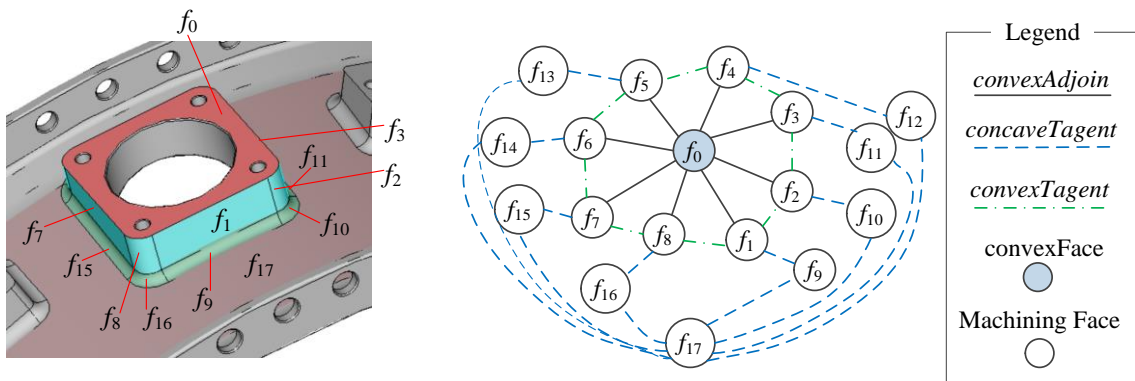


Figure 7: Illustration for the part MFKG: (a) A part of geometric faces, (b) A part of the part MFKG.

On the above definition of MFKGs, we can construct MFKGs of the part to be recognized. The MFKG construction procedures are provided as shown as in Figure 8.

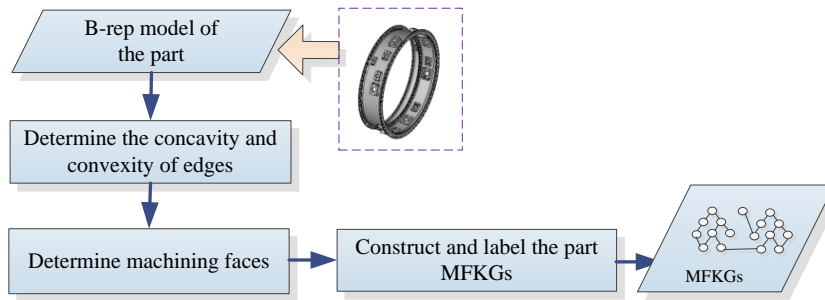


Figure 8: Flow chart of constructing MFKGs.

- Input B-rep model of the part to be recognized.
- Identify the concavity and convexity of all edges in the B-rep model.
- Determine the machining faces from the geometric faces.
- Construct and label the part MFKGs.

The following will elaborate the last three procedures.

4.4.2 Calculation of the concavity and convexity of edges in the B-rep model

An edge is the topological entity that connects two faces. Currently, there are many algorithms to determine the concavity and convexity of edges. Most algorithms implement with vector calculation. However, due to the complexity of the B-rep models, an edge usually corresponds to many types of geometric curves, such as ellipse curves, spline curves, and T curves. If there is no edge direction information, the vector calculation method is very difficult. As far as we know, NX Open API doesn't provide any fin information of edges, which results in the difficulty to obtain the direction information of an edge. Hence, this paper employs a method to determine whether an edge is convex or concave by calculating distances. Its basic principle is to move two points that locate on the two adjacent surfaces of an edge along the normal direction of the two surfaces, and the distance change trend between two points of the convex edge and concave edge is completely different. We use this to judge whether an edge is convex or concave. As shown in Figure 9, the algorithm steps are follows:

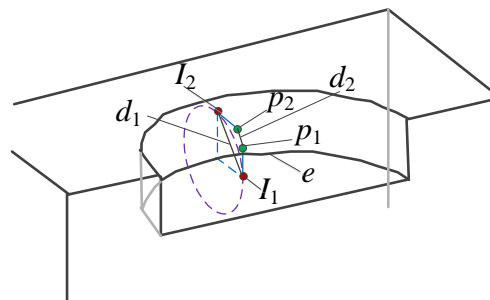


Figure 9: Illustration for calculation of concavity and convexity of edges.

Step1: Select the midpoint of an edge e to be judged and construct a circle whose direction is the cross product direction of two normal directions of adjacent faces of the edge in the midpoint, and whose radius is r .

Step2: Compute the intersection of the circle and two adjacent faces. Let the intersection points as I_1 and I_2 . Compute the distance between I_1 and I_2 and set the calculated distance as d_1 .

Step3: Move intersection I_1 by $0.6*r$ distance along the normal direction of the face where the point I_1 is located, and point p_1 is obtained. Similarly, move intersection I_2 by $0.6*r$ distance along the normal direction of the face where the point I_2 is located, and point p_2 is obtained. The distance between p_1 and p_2 is calculated as d_2 .

Step4: Compare the two distances. If d_2 is less than d_1 , the edge e is a concave edge, otherwise is a convex edge.

In the above algorithm, the radius of auxiliary circle is a very important parameter. Improper setting of the radius value may result in failure of the intersection computation. We employ a progressively setting method to set the radius value.

4.4.3 Determination of the machining faces

As discussed above, machining faces are the basis of recognizing machining features. Before constructing the MFKG, it is necessary to identify the machining face from geometric faces of the B-rep model. Through observation, we summarize the following rules to determine machining faces:

- Adjacent faces of inner loops

All adjacent faces of inner loops (except the face where the inner loop is located) are determined as machined faces.

- The faces that have concave edges

As long as one edge on a geometric face is concave edge, the face is determined as a concave face, accordingly, and is also determined as a machining face.

- Quadric surface with inward normal

All geometric faces that are quadratic surfaces (e.g. cylindrical surfaces, conical surfaces) with inward normal are identified as machining faces.

It should be noted that although some surfaces formed by machining may be completely composed of convex edges, such as the top face of milling, the end face and shaft face of turning, as the shape characteristics of these machining faces are simple, in this paper these machining faces are not selected as machining face nodes of MFKGs.

4.4.4 Constructing and labeling the part MFKGs

According to the definition provided above, the construction of the part MFKGs is to create nodes and relation edges between nodes, and the labelling is to relate an instance node with a class in the ontology model. On the identification of machining faces, the construction and labelling procedures are as follows:

Step1. Create instance nodes of the part MFKG. For each identified machining face in the part B-rep model, an instance of *MachiningFace* class is created and defined as a node of the part MFKG. Suppose there are n instances that have been newly created, the set of nodes is $\{Mf_1, Mf_2, \dots, Mf_n\}$

Step2. Create relation edges of the part MFKG. According to the concavity and convexity of an edge that connects two machining faces, the adjacency relations between them can be obtained, which is also used to create a relation edge that connects two nodes of machining face instances. There are five types of adjacency relations that have been defined as object properties in the machining face class.

Step3. Add spatial relation edges between two machining face nodes. According to the definition provided above, the spatial relation between machining faces is the spatial relation between geometric faces associated by machining faces. The spatial relation mainly is the parallel relation, the coplanar relation, the perpendicular relation and coaxial relation, which is represented by the "*parallelTo*" object property, the "*coplanarTo*" object property, the "*perpendicularTo*" object property, and the "*coaxialTo*" object property, respectively.

Figure 10 illustrates a part model and its MFKGs. The created MFKGs also can be represented as the following set form:

$$G_1 = \{MachiningFace\{F1, F2, F3, F4, F5, F6, F7, F8, F9\}, concaveAdjoin\{(F1, F2), (F1, F3), (F1, F4), (F1, F5), (F1, F6), (F1, F7), (F1, F8), (F1, F9)\}, parallelTo\{(F7, F9), (F6, F8), (F2, F5), (F3, F4)\}, coplanarTo\{(F9, F8), (F2, F3), (F4, F5), (F6, F7)\}\}$$

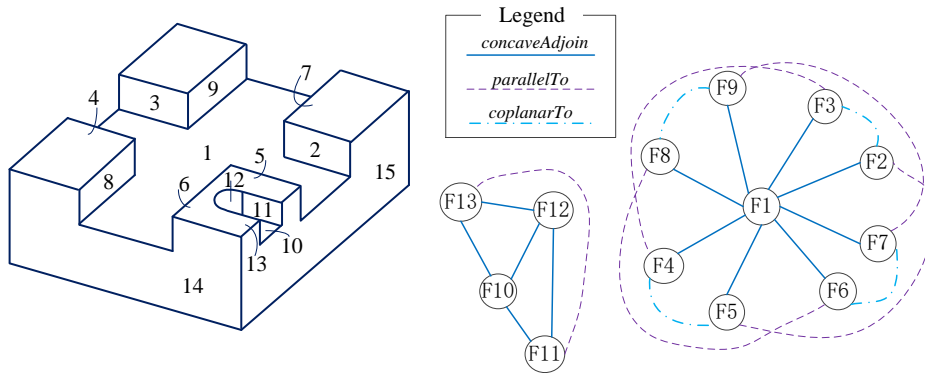
$$G_2 = \{MachiningFace\{F10, F11, F12, F13\}, concaveAdjoin\{(F10, F11), (F10, F12), (F10, F13)\}, parallelTo\{(F11, F13)\}\}$$


Figure 10: Illustration for a part model and its MFKGs: (a) The part model and (b) MFKGs of the part model.

5 USER-DEFINED MACHINING FEATURES RECOGNITION

5.1 Overview for the Recognition Approach

The so-called feature recognition is identification for the geometric surface based on the cognition of the shape and engineering meaning of the machining features. In this paper, we represent this cognition and engineering meaning as recognition knowledge of machining features characterized by a set of concepts and explicit semantic rules. Hence, the feature recognition is converted into an issue on semantic reasoning for MFKGs using machining feature knowledge.

The semantic reasoning mechanism is to determine the semantic satisfiability of machining feature rules for the constructed MFKGs. Suppose we have the knowledge base: $T \cup G \cup R$. T denotes the ontology, G denotes the facts of geometric faces in the part, and R denotes the knowledge rules (including explicitly defined feature recognition knowledge rules and semantic rules in ontology). The main reasoning procedure is divided as two steps as shown in Figure 11.

The first step is to reason the MFKG constructed by the method provide in Section 4.4. The machining face node in the MFKG is performed classification reasoning according to the ontology definition of machining faces and SWRL knowledge rules. Specifically, a machining face denoted by a node is continually classified as a more specific machining face. For example, if a machining face is tested and confirmed as an accessible face, the face is classified as a bottom face and the face node is relabeled as an instance of the *BottomFace* class. In addition, some face nodes may be regrouped as a new type of nodes or may be removed.

The second step is to reason the MFKG processed by first step and recognize machining features. After the first step, the processed MFKG more tends to the conceptual description of specific machining features, which lays a foundation for subsequent machining feature recognition based on semantic reasoning. The details of the two steps are elaborate in the following sections.

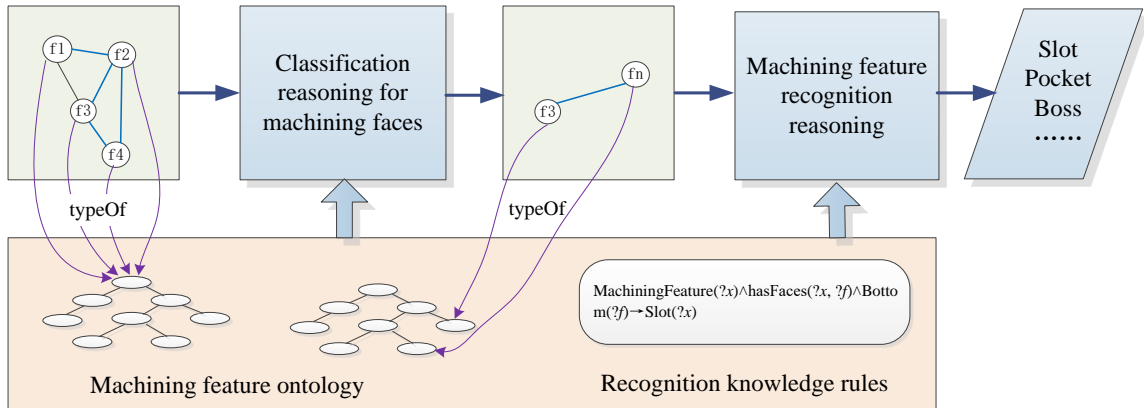


Figure 11: Reasoning procedure for machining feature recognition.

5.2 Classification Reasoning for Machining Faces Combining Rules and Geometric Computation

As analyzed above the machining faces are formed by cutting tools and their movement so the definition of machining faces are associated to the machining methods. Hence, the instance of the *MachiningFace* class in the MFKG needs to be relabeled by more specific machining face classes. The classification reasoning combines the SWRL knowledge rules and geometric test computation. The following provides two kinds of classification reasoning methods.

Above SWRL face knowledge rules provide a basis for classification reasoning. It can be observed that an SWRL rule can be divided into two parts. The right-hand side is the reasoning result, which denotes the variable instance should be relabeled by which more specific machining face class. The left hand side is an antecedent with its associated semantics. Each atom will match not only all instance nodes in a MFKG but will also match individuals who are entailed by the ontology to be individuals of that class. The reasoning process is shown as in Algorithm 1:

Algorithm 1: Classification reasoning for machining faces

Input: A set of classification rules: $R = \{R_1, R_2, \dots, R_n\}$ and a set of instances of the *MachiningFace* class in a MFKG of the part: $MF = \{f_1, f_2, \dots, f_m\}$

```

1: for  $i \leftarrow 1$  to  $n$  do
2:   for  $j \leftarrow 1$  to  $m$  do
3:      $k \leftarrow 1$ ;
4:     while  $k < q$  do
5:       Test antecedent of rule  $R_i (a_1 \wedge a_2 \wedge \dots \wedge a_q)$  by  $f_j$ ;
6:       if  $a_k = \text{false}$  then
7:         break;
8:       end if
9:        $k \leftarrow k + 1$ ;
10:    end do
11:    if  $k = q$  then
12:      relabel  $f_j$  as the class denoted by the consequent of rule  $R_i$ 
13:    end if
14:  end do
15: end do

```

5.3 Machining Feature Recognition Reasoning

After the classification reasoning for machining faces introduced above, the part MFKGs become more concise and specific. For example, the side face chain includes a group of sides faces (machining surfaces), which lays a foundation for machining feature recognition. Same as above, machining feature recognition reasoning starts from the defined semantic rules. From the defined rules in Table 3, we can observe that consequent of rules are the concept of machining features. Hence, for each rule we can assume that the consequent of the rule holds, namely the consequent feature has been found, we only needs to find a variable solution to make the antecedent of the rule hold true.

The so-called solution is a configuration of variables for the reasoning rules. In a solution, a variable in the rule is bound to a value (a value can be an instance or a data type value). If all the bounded values in a solution can make all the atoms in the rule body hold true, this solution is a feasible solution and can be output as the reasoning result. Hence, we employ a backward chained reasoning approach to search the instances in the part MFKGs. The backward chained reasoning approach is a well-known form of goal-directed reasoning with conventional Horn rules and can help users find implicit instances or fillers for some defined classes or properties.

In the solution search process, to efficiently search for and match the instances in the MFKG, we employ a well-known Prolog derivation tree as the searching data structure. Each obtained feasible solution forms a leaf of the derivation tree. The details on search process can be referred in [13]. The machining features recognition reasoning process is shown as in Algorithm 2.

Algorithm 2: User-defined machining features recognition

Input: A set of rules: $R \{R_1, R_2, \dots, R_n\}$ and a MFKG of the part : G .

Output: A set of recognized machining features: $F \{f_1, f_2, \dots, f_m\}$

```

1: for  $i \leftarrow 1$  to  $n$  do
2:    $m \leftarrow 0$ ;
3:    $k \leftarrow 1$ ;
4:   while  $k < q$  do
5:     Construct a variable solution  $s$  by  $G$ ;
6:     Test antecedent of rule  $R_i(a_1 \wedge a_2 \wedge \dots \wedge a_q)$  by  $s$ ;
7:     if  $a_k = 0$  then
8:       break;
9:     end if
10:     $k \leftarrow k + 1$ ;
11:   end do
12:   if  $k = q$  then
13:     $m \leftarrow m + 1$ ;
14:    Create a machining feature  $f_m$  by the consequent of rule  $R_i$ ;
15:    Append  $s$  into  $f_m$ ;
16:    Add  $f_m$  to  $F$ ;
17:   end if
18: end do

```

6 CASE STUDY

The approach presented in this paper is implemented by programming using visual C++ in NX 12.0. The B-rep model of the part can be visited by UG OPEN API. The machining feature ontology and semantic rules are defined under the protégé tool and saved in OWL / XML format. In the recognition system, we developed a parsing program using TinyXML parser to load and parse ontology and rules.

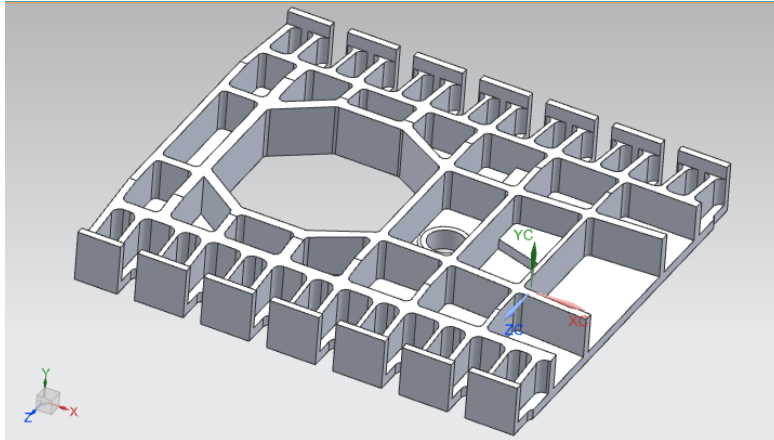


Figure 12: A part to be performed machining feature recognition.

In order to demonstrate the presented approach, we take the machining feature recognition of an aeronautical part shown in Figure 12 as case study. This part is a thin wall structural part that is mainly milled from a titanium alloy blank. From Figure 12, we can see that there are multiple machining features to be recognized. Using the user-defined approach presented in this paper the implementation procedure is as follows:

6.1 Defining the Semantic Description for Newly Added Machining Features

Firstly, we analyze the structure shape of this part. After considering the processing technology and current tools, it is decided to add four new machining features on the basis of the original machining feature library, which are the U-shaped slot, the no bottom U-shaped slot, the step boss and the circular boss. The main procedures are the following two steps.

Step1: Define the concepts (terms) for the newly added machining features. Open the ontological file in the feature library and load into the protégé tool. On the *MachiningFeature* class, we define the *U-shapedSlot* class, the *NoBottomU-shapedSlot* class, the *StepBoss* class, and the *CircularBoss* class, respectively, which are shown in Figure 13(b). The object properties of the four kinds of features are also defined. As a result, the concepts (terms) on the newly added machining features and their hierarchical relationships have been set up.

Step2: Define explicit SWRL rules for the newly added machining features. Analyze the structure shape of the newly added machining features as follows:

- U-shaped Slots

As shown in Figure 13(a), the U-shaped slot has a bottom face and a chain of side faces which concavely adjoin to the bottom. The chain of side faces is adjoined to each other and is open. In addition, as there are two opposite parallel side faces in the face chain, we use *TwoSideFace* concept to represent them.

- No bottom U-shaped slots

Except that there is no bottom face, the structure of the no bottom U-shaped slot is the same as that of the above-mentioned U-shaped slot. As a result, we use the *OpenNoBottomFaceChain* concept to describe their side faces.

- Circular bosses

The circular boss has only a face in cylindrical surface which concavely adjoins to the bottom face.

- Step bosses

The step boss is a boss adjoined to side faces. As a result, the step boss has a bottom and its side face chain is open and concavely adjoins to the bottom. We use the *OpenSideFaceChain* concept to describe its side faces.

Based on above analysis, we find that we can define new SWRL rules only by using the defined machining face concepts. The machining feature rule is defined in the SWRL tab window of the protégé tool. The defined SWRL rules are listed in Table 5.

No	SWRL rule
1	$\text{MachiningFeature}(\?x) \wedge \text{hasFaces}(\?x, \?b) \wedge \text{BottomFace}(\?b) \wedge \text{hasFace}(\?x, \?s) \wedge \text{TwoSideFace}(\?s) \wedge \text{concaveAdjoin}(\?b, \?s) \rightarrow \text{U-shapedSlot}(\?x)$
2	$\text{MachiningFeature}(\?x) \wedge \text{hasFaces}(\?x, \?s) \wedge \text{OpenNoBottomFaceChain}(\?s) \rightarrow \text{NoBottomU-shapedSlot}(\?x)$
3	$\text{MachiningFeature}(\?x) \wedge \text{hasFaces}(\?x, \?b) \wedge \text{BottomFace}(\?b) \wedge \text{hasFace}(\?x, \?s) \wedge \text{SideFace}(\?s) \wedge \text{concaveAdjoin}(\?b, \?s) \wedge \text{hasSurface}(\?s, \?g) \wedge \text{CylindricSurface}(\?g) \rightarrow \text{CircularBoss}(\?x)$
4	$\text{MachiningFeature}(\?x) \wedge \text{hasFaces}(\?x, \?b) \wedge \text{BottomFace}(\?b) \wedge \text{hasFace}(\?x, \?s) \wedge \text{OpenSideFaceChain}(\?s) \wedge \text{concaveAdjoin}(\?b, \?s) \wedge \text{hasFaces}(\?x, \?t) \wedge \text{convexAdjoin}(\?t, \?s) \rightarrow \text{StepBoss}(\?x)$

Table 5: The SWRL rules defining four newly added machining features.

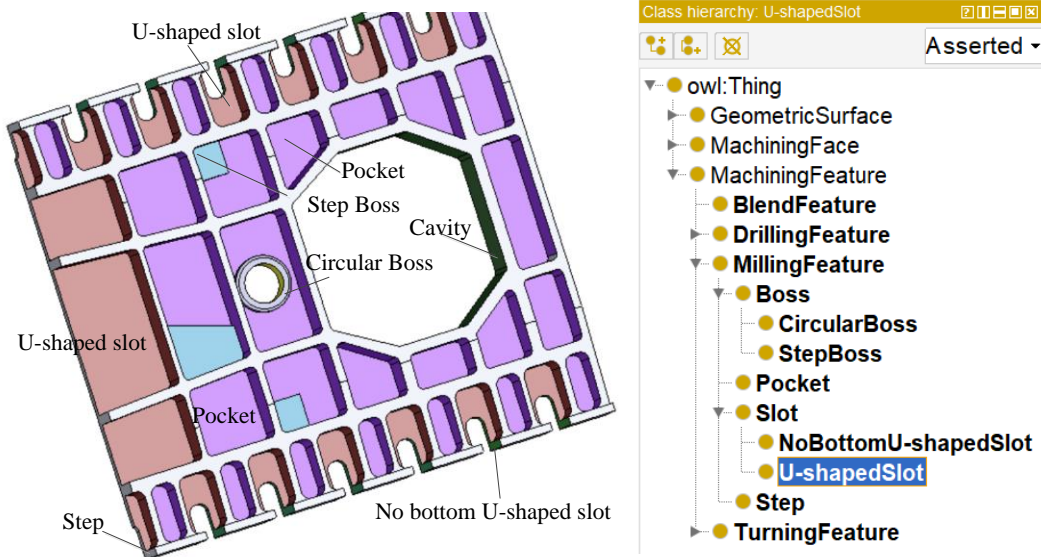


Figure 13: A machining feature recognition case: (a) The recognition results, (b) User-defined machining features.

6.2 Constructing MFKGs of the Part and Perform Feature Recognition

After open the part file and load into NX, the B-rep model of this part can be visited. Construct the MFKGs of the part using the constructing approach provided in Section 4.4. A set of MFKGs are constructed, which includes 60 MFKGs.

Perform feature recognition based on semantic reasoning for the constructed MFKGs and 60 machining features are recognized. The recognized result is listed in Table 6.








No.	Machining Feature	Number	Illustration
1	U_shapedSlot	12	
2	U_shapedThroughSlot	12	
3	Pocket	29	
4	CircularBoss	1	
5	StepBoss	3	
6	Step	2	
7	Cavity	1	

Table 6: The machining feature recognition results of the study part.

7 CONCLUSION

Owing to the diversity of product structures, machining methods, equipment and tools, there is a great need for an open, shareable and extensible knowledge representation and recognition methods for machining feature recognition. Users can define machining features with an open pattern according to their own needs, and the recognition system can accurately recognize them from CAD models accordingly. This paper proposes an approach based on semantic representation and reasoning to realize the recognition of custom machining features from the B-rep model. Using ontology and SWRL knowledge rules users can define the machining features to be recognized. The recognition system converts the part B-rep model into a set of MFKGs labeled by ontology, and then performs semantic reasoning for the created MFKGs according to the defined SWRL rules. Finally a set of machining features are recognized based on reasoning results. The presented recognition approach has been realized using C++ programming in NX system, and a group of mechanical parts have been tested. The recognition results demonstrate feasibility and good openness of the presented method. This work provides an attempt for intelligently solving engineering problems with open knowledge driven methods.

Due to the loss of geometric information after feature intersection, it is difficult to describe the change of shape based on logical symbol methods. This work still falls short on the capabilities to semantically describe, recognize, and understand complex intersecting features. In the future work, we will integrate machine learning methods to help solve this problem.

ACKNOWLEDGEMENTS

This work is supported by the National Science Foundation of China (Grant No. 51775081). The authors thank the anonymous reviewers for their helpful suggestions on this study.

Yingzhong Zhang, <http://orcid.org/0000-0003-3584-7239>

REFERENCES

- [1] Shah, J. J.; Anderson, D.; Kim, Y. S.; Joshi, S.: A discourse on geometric feature recognition from CAD models, *Journal of Computing and Information Science in Engineering*, 1, 2001, 41–51.
- [2] Han, J.; Pratt, M.; Regli, W. C.: Manufacturing feature recognition from solid models: A status report, *IEEE Transactions on Robotics and Automation*, 16 (6), 2000, 782–796.
- [3] Wan, N.; Du, K.; Zhao, H.; Zhang, S.: Research on the knowledge recognition and modeling of machining feature geometric evolution, *Journal of Advanced Manufacturing Technology*, 79(1), 2015, 491–501.
- [4] Verma, A. K.; Rajotia, S.: A review of machining feature recognition methodologies, *International Journal of Computer Integrated Manufacturing*, 23(4), 2010, 353–368. [https://doi: 10.1080/09511921003642121](https://doi.org/10.1080/09511921003642121).
- [5] Joshi, S.; Chang, T. C.: Graph-based heuristics for recognition of machined features from 3D solid model, *Computer-Aided Design*, 20 (2), 1988, 58–66.
- [6] Singh, R. D.; Mittal, A.; Bhatia, R. K.: 3D convolutional neural network for object recognition: a review, *Multimedia Tools and Applications*, 78, 2019, 15951-15955. [https://doi: 10.1007/s11042-018-6912-6](https://doi.org/10.1007/s11042-018-6912-6)
- [7] Zhang, Z.; Jaiswal, P.; Rai, R.: FeatureNet: machining feature recognition based on 3D convolution neural network, *Computer-Aided Design*, 101, 2018, 12-22. [https://doi: 10.1016/j.cad.2018.03.006](https://doi.org/10.1016/j.cad.2018.03.006)
- [8] Shi, P.; Qi, Q.; Qin, Y.; Scott, P. J.; Jiang X.: A novel learning-based feature recognition method using multiple sectional view representation, *Journal of Intelligent Manufacturing*, 31, 2020, 1291-1309. <https://doi.org/10.1007/s10845-020-01533-w>
- [9] Charles, R. Q.; Su, H.; Kaichun, M.; Guibas L. J.: Pointnet: Deep learning on point sets for 3d classification and segmentation, *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, 77-85.
- [10] Ma, Y.; Zhang, Y.; Luo, X.: Automatic recognition of machining features based on point cloud data using convolution neural networks, *ACM International Conference Proceeding Series*, 2019, 229-235.
- [11] Stefano, P. D.; Bianconi, F.; Angelo, L. D.: An approach for feature semantics recognition in geometric models, *Computer-Aided Design*, 36 (10), 2004, 993–1009.
- [12] Wang, Q.; Yu, X.: Ontology based automatic feature recognition framework, *Computers in Industry*, 65(7), 2014, 1041–1052.
- [13] McGuinness, D. L.; Harmelen, F. V.: OWL Web ontology language overview, 2004. <http://www.w3.org/TR/owl-features/> February 20
- [14] Zhang, Y.; Luo, X.; Zhang, B.; Zhang, S.: Semantic approach to the automatic recognition of machining features, *The International Journal of Advanced Manufacturing Technology*, 89(1-4), 2017, 417–437.
- [15] Sanfilippo, E. M.; Borgo, S.: What are features? An ontology-based review of the literature, *Computer-Aided Design*, 80, 2016, 9–18.
- [16] Gupta, R. K.; Gurumoorthy, B.: Feature-based ontological framework for semantic interoperability in product development, *Advanced Engineering Informatics*, 48, 2021, 101260.
- [17] Mandorli, F.; Borgo, S.; Wiekak, P.: From form features to semantic features in existing MCAD: An ontological approach, *Advanced Engineering Informatics*, 44, 2020, 101088.
- [18] Sanfilippo, E. M.: Feature-based product modelling: an ontological approach, *International Journal of Computer Integrated Manufacturing*, 31(11), 2018, 1097–1110.
- [19] Gaines, D. M.; Hayes, C. C.: Custom-Cut: a customizable feature recognizer, *Computer-Aided Design*, 31(2), 1999, 85–100.
- [20] Li, S.; Shah, J. J.: Recognition of user-defined turning features for mill/turn parts, *Journal of Computing and Information Science in Engineering*, 7(3), 2007, 223-235.
- [21] Liu, C.; Li, Y.; Wang, P.; Hao, X.: A user defined method for machining features in NC programming of complex structural parts, *Hangkong Xuebao/Acta Aeronautica et Astronautica Sinica*, 38(6), 2017, 420-735.

- [22] Ma, H.; Zhou, X.; Liu, W; Niu, Q.; Kong C.: A customizable process planning approach for rotational parts based on multi-level machining features and ontology, *The International Journal of Advanced Manufacturing Technology*, 108, 2020, 647–669.
<https://doi.org/10.1007/s00170-020-05437-0>
- [23] Wu, M. C., Liu, C. R.: Analysis on machined feature recognition techniques based on B-rep, *Computer-Aided Design*, 28(8), 1996, 603–616.
- [24] Musen, M. A.: The Protégé project: A look back and a look forward, *AI Matters*, Association of Computing Machinery Specific Interest Group in Artificial Intelligence, 1(4), June 2015.
<https://doi.org/10.1145/2757001.2757003>
- [25] Horrocks, I.; Patel-schneider, P. F.; Boley, H.; Tabet, S.; Grosf, B.; Dean, M.: SWRL: a semantic Web rule language combining OWL and RuleML, 2004,
<http://www.w3.org/Submission/SWRL>.
- [26] Jia, J.; Zhang, Y.; Saad, M.: An approach to capturing and reusing tacit design knowledge using relational learning for knowledge graphs, *Advanced Engineering Informatics*, 51, 2022,101505.