STEP-NC based reverse engineering of in-process model of NC simulation

Shixin Xú, Nabil Anwer, Charyar Mehdi-Souzani, Ramy Harik, Lihong Qiao

STEP-NC Based Reverse Engineering of In-Process Model of NC Simulation

Shixin XÚ [1,2], Nabil ANWER [1,#], Charyar MEHDI-SOUZANI [1], Ramy HARIK [3] and Lihong QIAO [2]

[1] LURPA, ENS Cachan, Univ. Paris-Sud, 61 avenue du Président Wilson, F-94235 Cachan, France

[2] School of Mechanical Engineering and Automation, Beihang University, Beijing, China, 100191

[3] Department of Mechanical Engineering, University of South Carolina, Columbia, SC, USA

Corresponding Author / E-mail: anwer@lurpa.ens-cachan.fr

Abstract

This paper proposes a comprehensive process for reverse engineering and feature recognition from an in-process model (IPM), represented in triangle meshes which results from NC simulation. IPM can be rather useful in redesign, process planning, machining inspection, etc., in addition to visualizing machining processes. For example, we may carry out in-process corrections accordingly by comparing the rebuilt model from an IPM with the original one. However, until now IPM reverse engineering, especially manufacturing feature based, is seldom researched. First, after systematically summarizing the IPM characteristics and taking advantages of them, a novel region segmentation method based on a shape descriptor — called the shape index, which is close to the visual effect and is capable to identify local shapes accurately, is proposed. The shape index, which is derived from principal curvatures, is calculated by discrete differential geometry methods. The segmentation is carried out gradually from simple surface types, such as planes, cylinders, to more complex surface types; second, the recognition of manufacturing features defined in the ISO 14649 standard is explained in detail. The approach is graph based, and highly relies on the presence of concave edges in the model. In the recognition algorithm Euler characteristic and curvedness of the model are applied, which are novel attempts. Case studies to verify the proposed approaches are provided.

Key words: reverse engineering; region segmentation; feature recognition; in-process model; discrete differential geometry; NC simulation; STEP-NC

1. Introduction

Discrete data geometry processing has gained more attention in recent years in manufacturing since it permits simple representation and yet is capable to consider complex shapes. As a principal component of virtual manufacturing, NC machining simulation depicts the material removal processes by simulating the movements of the machine tool and the cutting tools. Generally machining processes are simulated using a time or displacement discretization of the tool path. Corresponding to each machining process, we can get an in-process model (IPM), as an example, the IPM shown in Fig.1. This model is obtained by the geometric modeling of the engagement of the cutter with the workpiece at discrete intervals along the tool path. There are various modeling techniques for the description of the engagement, such as the solid-model-based, the wire-frame-based, the voxel-/dexel-/z-buffer-based, the point-based [1]. Although the solid modeling techniques (e.g. B-rep) are more precise, discrete meshes (voxel-, dexel-, and z-buffer based) are most commonly used for the representation of cutting tools and workpieces

because the dynamic visualization of the material removal on the part is an essential feature in NC simulation. In the course of simulation, the in-process models can be output as triangle meshes (e.g. in STL/VRML files). Nowadays the majority of commercial NC simulators (such as NCSimul®) provide such an interface.
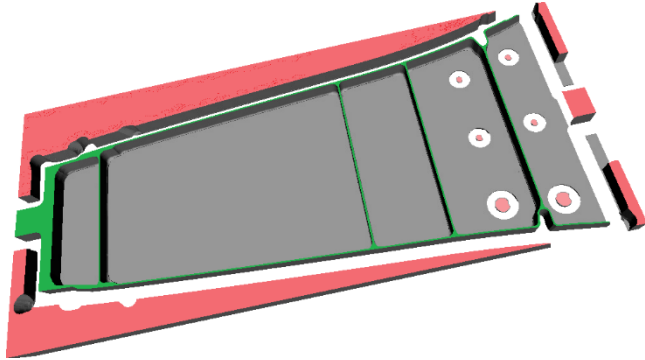

Fig. 1 An IPM output after a roughing process

Although research on feature recognition from regular CAD models has been done extensively, feature recognition and reverse engineering from IPMs are rarely reported and studied so far. Anwer, et al [2] propose a "one surface per triangle method", which uses a special underlying data structure of facets. This data structure is evaluated during the simulation. Their method is efficient and robust when the meshes are not very large.

Until now, the major goals of NC simulation are still confined to check an unproved NC program to avoid potential collisions with fixture elements, making undercuts, or leaving excess material. However, Reverse engineering of the IPM can be rather useful in redesign, process planning, machining inspection, etc. For example, by comparing the reconstructed model from an IPM with the original CAD model, we can know the potential machining deviation of a machining operation, then we may carry out in-process correction. In this work, by making use of the simulated output—the IPM, and by re-establishing a machining feature based model, we can extend the applications of NC simulation to build the feedback link from CAM/CNC systems or shopfloor amendments back to CAM/CAPP/CAD systems. This feedback link is very useful because the part program is often changed by machining engineers according to the simulation results or the shopfloor situation. So the current unidirectional information flow from CAM to CNC can be changed into bidirectional one [3]. The re-established feature model can serve for design improvements since we can compare this model with the original designed model, measure more reliable and accurate dimensions on this model, or use it for FEA, etc.

In addition, in some situations, the original CAD model is neither available nor usable, and the NC program is the only data available for producing a part. This could happen when old CAD archives are incompatible with new systems or machines, or they are damaged or lost, or when parts are designed and machined only at shopfloors, etc. Hence rebuilding the feature based CAD model from the IPM is necessary under these circumstances. It would save much resource than designing a feature based model from scratch.

These practical needs thus prompt to us to develop techniques for feature recognition and reverse engineering of in-process models to promote CAM/CNC integration and bidirectional information transfer between them. The research work reported in this paper is motivated on basis of the above

observation and actual needs. It mainly explains on approaches to IPM segmentation and then recognition of manufacturing features (i.e. STEP-NC features). The remainder of the paper is organized as follows: Section 2 presents briefly the review of literature on mesh segmentation, feature recognition and existing work on reverse engineering NC simulation and G-Code. Section 3 describes the characteristics of the IPM geometric data. Section 4 presents an overview framework of the developed approach. IPM coarse and fine segmentation approaches are elaborated in section 5. IPM coarse segmentation approach is elaborated in section 5. STEP-NC manufacturing features recognition approach and classification-based rules are presented in section 6. Case studies are presented in section 7. Finally, conclusions and perspectives of this research work are discussed in section 8.

2. Related works

The related works mainly involve mesh segmentation and feature recognition. Mesh segmentation partitions the unorganized triangle set into subsets, each of which is supposed to correspond to a particular type of surface. The result is region collections, like a surface model. Feature recognition is to further extract and organize some regions into manufacturing features. The source data is usually a solid model, but it also can be other types of data, such as polygon meshes, point cloud, or a G-code program.

2.1 Mesh segmentation

Mesh segmentation or region segmentation play an important role in the discrete shape modeling domain. Segmentation methods try to exploit geometric information to infer a decomposition of the surface that corresponds to the one that experts would produce manually [4]. When observing a rendered triangle mesh, we can easily perceive a decomposition of the surface into interesting regions, such as primitive shapes, whereas it turns out to be a rather hard task for automation. Segmentation methods have been comprehensively surveyed by Shamir et al [5, 6]. Segmentation for mechanical engineering applications aims at partitioning the discrete part model into regions and each region can be fitted by a single mathematically analyzable shape. The segmentation algorithms for a mechanical part mesh model are broadly identified in two main categories:

Edge-based approaches [7-11] – They try to detect boundaries between surface patches. Some derivative attributes, like normal vectors, curvatures, are commonly used to find the boundaries. Normal vectors are used to determine sharp edges and curvatures for smooth edge detection.

Face-based approaches [12-16] – They cluster points/vertices into connected regions with similar properties and boundaries are then derived. This category works on the global size of point set and so it is more robust to the edge-based one. Bénière, et al [12] have presented and developed a comprehensive process to extract geometric primitives and build a B-rep model from 3D meshes.

In addition, there are some approaches combining edge-based and face-based information to try to overcome limitations of the above two categories of approaches [17, 18]. To segment meshes obtained from point cloud, Zhao, et al [19] use curvedness to detect shape boundary, and use shape index to cluster vertices. This paper's work is a continuation of their approach applied in IPM.

Most of the segmentation approaches are directly based on derivative properties (normal vectors, principal / Gaussian / mean curvatures), which are sensitive to noise. They assume the polyhedral surfaces are noiseless. But IPM data has a kind of "noise"—scallops left on machined surfaces. So there is no very suitable solution reported to region segmentation for in-process models yet.

2.2 Feature recognition

A feature, or a form feature, to be more specifically in our work, usually refers to a portion of a part that bears certain geometric or topological properties, from which we can deduce the manufacturing information for making them. Completeness of form feature set is very subjective and domain dependent. A feature generation model proposed by Nalluri et al [20] attempts to define the completeness of a feature set. They define domain independent form feature as a set of faces with distinct topological and geometric characteristics. They model creation of a form feature as addition or subtraction of feature-solid to/from based-solid. They define feature "type" based on the local topology of participating base-solid faces and "shape" based on shape of the feature-solid. Based on these definitions they enumerate and classify form features. They enumerate 94 sweep form features and provide proof that those 94 types are complete for sweep feature-solid.

In this paper we are interested in machining features, which are concerned with both the geometric shapes and manufacturing operations, needing more parameters in their definition. For example, we may either use the name "pocket" to refer to a swept cut on the boundary of a part model, or to a trace left on the part boundary by a specific machining operation. Machining features, as an important subset of manufacturing features, can be regarded as the volume swept by a cutting tool, which is always a negative (subtracted) volume. Work on features can be divided into two rough categories: design-by-features [21] and feature recognition [22]. In design-by-features feature structures are introduced directly into a model using particular operations or by sewing in shapes. On the other hand, the goal of feature recognition is to extract automatically higher level entities (e.g. machining features) from lower level elements (e.g. surfaces, edges, etc.) of a CAD model.

Automatic feature recognition is regarded as an ideal solution to automate design and manufacturing processes. This is the part of the research that has attracted much of the attention. Another important application of feature recognition is for manufacturability evaluation. A feature recognition system should be able to interpret the design differently based on alternative features and feedback the manufacturability and cost of those interpretations to the designer.

There are different feature recognition techniques that have been proposed for CAD/CAM integration and process planning. Han et al [22] provides a detailed analysis of some of the existing approaches. The most common methods range from graph-based algorithms [23] to hint-based and volumetric decomposition techniques. In the graph-based feature recognition, a graph showing the topology of the part (connection of faces) is created. The graph is often attributed (e.g., the edges are marked as concave or convex), it is then analyzed to extract subsets of nodes and arcs that match with a predefined template.

Graph-based approaches are criticized for several shortcomings. They fail to account for manufacturability of the recognized features due to their strong reliance on topological patterns rather than geometry. The intersection of features causes an explosion in the number of possible feature patterns that spoils attempts to formulate feature patterns. To address these difficulties, Vandenbrande et al [24] proposed to search for "minimum indispensable portion of a feature's boundary", called hints, rather than complete feature patterns. For example, presence of two opposite planar faces is a hint for potential existence of a slot feature. Hints are not necessarily restricted to the part geometry. They can be extracted from design attributes as well. For example, "a thread attribute may be taken as a hole hint". This approach has been more successful in recognizing intersecting features. Some authors [25, 26] are in favor of using a hybrid of graph-based and hint-based feature recognition to improve the efficiency of hint-based reasoning. In the hybrid approach, graph-based reasoning is used to find out those areas of the part that certainly lead to valid features when used by the hint-based reasoner.

However, building feature recognition systems that function effectively on real industrial products has been elusive. A real product with thousands of faces and edges brings almost all the above approaches to a halt due to computational complexity. Furthermore, the features studied in these approaches are usually over simplified. The majority of feature recognition literature normally handles 2½D features. The work done by Sundararajan [27] is focused on free form surfaces, but it is limited in application. Some issues such as the presence of filleted edges and free form surfaces in the model are studied by Rahmani and Arezoo [25].

Few commercial feature recognition systems are also available. They have effectively adopted feature recognition technology for recreating the feature tree from imported models so that even the imported models can be edited as if it were a native model. Major 3D CAD modelers have feature recognition to convert imported 3D models into native feature based models. CAM software and design for manufacturing software are also built using this technology. Few CAD/CAM software have used commercially available third-party feature recognition library, which recognizes various features from B-rep models [28]. Separate libraries are available for design, manufacturing and sheet metal applications [28]. However, manufacturing features such as 3/4/5-axis feature recognition are generally unavailable in such commercial systems.

Many researches focused on reconstruction of surfaces or B-Rep solid model from triangular meshes, but only few went further to recognize features. The work done by Sunil et al [29] enables to perform feature recognition from a mesh model of sheet metal parts. Kiswanto et al [30] discussed feature identification by employing information from normal vectors of faceted model. This information together with curvature information are used to detect flat, saddle, concave, convex and closed bounded features on a sculptured surface. Extracting features, especially STEP-NC features, from a faceted model still needs to be investigated further.

Very limited works has been reported on recognition manufacturing/machining features from an in-process model of NC simulation [2], to our knowledge. In particular, no region segmentation work that uses shape-index has been reported. If developed the NC simulation functions can be extended. The emphasis of our present work is on the reverse engineering and feature recognition of in-process models. Various design and realization issues are addressed in the sections to follow.

2.3 Reverse engineering of NC simulation and G-code

The NC simulation geometric result, the in-process model, can be used to rebuild a part's CAD model. To our knowledge, reported work on this problem is scarce. Karunakaran, et al [31] proposed an algorithm for octree-to-B-rep conversion, which can deal with the reverse engineering of the simulation whose geometric output is octree representation. Anwer, et al [2] proposed two methods: "one surface per triangle" method and discrete shape segmentation method for processing mesh-output of NC simulation. The former one turns the mesh into a polygonal B-rep model, by attaching such information as color indicating a tool, cut number, surface index to triangle meshes in the course of simulation. Each triangle is turned into a face of the B-rep model. This method was realized in cooperation with a developer of NC simulation software. The latter one is based on differential geometry and the pair shape index, curvedness. This method decompose vertices into a group of regions that can be represented by primary shapes. Xu, et al [32] continued the above work to build a machining feature based model from the IPM.

A part program G-code can also be used to rebuild the part's CAD model by supplementing cutting tool and rawpiece information. Cioană, et al [33] developed an add-on application for SolidWorks® to import toolpath data from G code, together with cutting tool data, to build a CAD model interactively. Yan et al [34] developed a method for feature recognition from G-code based on Z-map representation of forms

(workpiece and tools). For each cutting operation, their system creates and updates the *Z*-map of the machined area and extracts cut contours, cutting parameters and milling feature. Milling feature recognition is done by comparing *Z* levels and checking the continuity of the cut contours. Shin et al [35] developed a system called G2STEP to convert G-code into STEP-NC. In their approach turning G-code programs are handled in detail, and turning feature recognition are mentioned. Zhang et al [3] proposed a method to re-use the process knowledge embedded in G-code part programs with different manufacturing resources. Their method can recognize 2½D features from milling G-code mainly by tool types, toolpath boundary and the rawpiece geometry.

3. In-process model characteristics

An in-process model is the geometric output data of an NC simulation. In virtual machining it is used for visualization of the machining process. The in-process model can be output by the simulation system as a collection of unordered set of triangles, which are usually in STL file format. In terms of triangle shapes within a triangle mesh, there are two distinct types of triangle meshes. One type is as shown in Fig. 2a (type I). Within a required surface deviation, this type, which is usually used for visualization purpose, uses as few triangles as possible to discretize a part's surfaces in order to present rapidly the dynamic variations of the workpiece being machined. The size, shape and distribution density of triangles are quite different. The other type is shown as in Fig. 2b (type II). The shapes of triangles are roughly similar, but the sizes of triangles can be quite different, and triangles are usually densely distributed in surface zones or along curved boundaries, which means the number of triangles per unit area is more than other zones. This type of triangulation is usually used for analysis, such as FEA. The factors to control the sizes and distributions of triangles are the maximum triangle's edge lengths, element number per curvature radius of the part's surface, etc. One common method to get type II triangulation is the Delaunay triangulation, which maximize the minimum angle of all the angles of the triangles, tending to avoid skinny triangles. In terms of the quality of the triangle mesh data, type II is generally better than type I. Besides, the triangle mesh obtained after the processing of scanned cloud points in reverse engineering belongs to this type since a huge number of the dense points are used to construct the mesh. The NC simulation requires rapid display and update of the geometric variation of the workpiece being cut, so the IPM adopts the former type of triangle mesh. It tries to use the least number of triangles for better performance.

By observing type I triangle mesh, we can find that, except for the planar/cylindrical zones, the characteristics of triangles within the surface zone are quite similar to those of type II. Therefore we have to devise an IPM-based region segmentation method that can efficiently handle triangle mesh type I as well as type II considering that a general mechanical part is composed of planar and cylindrical surfaces and possibly many other types of surfaces.
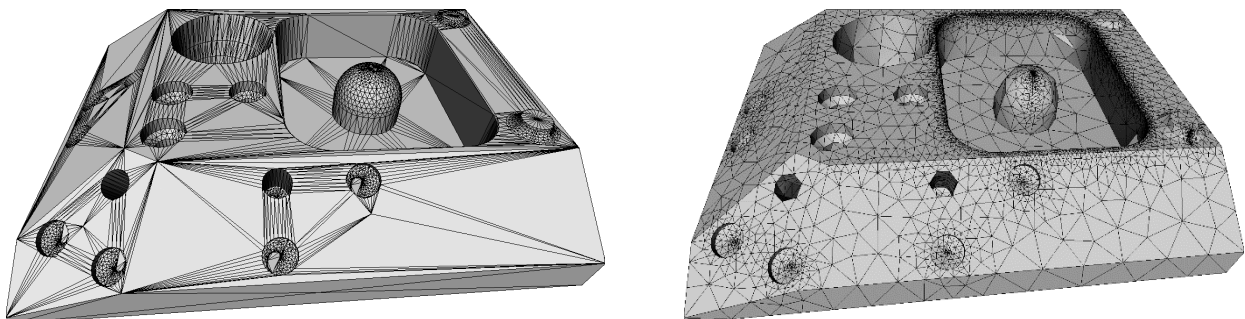
Fig. 2 Two types of triangle mesh: a) type I; b) type II

One distinctive characteristics of IPM is that there are often scallops and stepover traces left on machined surfaces, as seen on the wall faces in Fig.3. The part is machined by an endmill cutter. These scallops and traces, which will disturb the true underlying surface type recognition, can be considered as a kind of noise.

Another characteristic of the IPM of NC simulation is that it often has offcut components, some of which are even connected to the finished part thru a thin slice, like the offcut colored in red as shown in Fig.3.
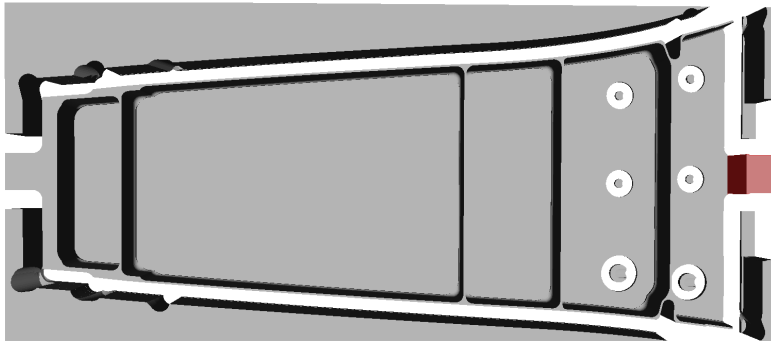

Fig. 3 An IPM (The connected offcut is indicated by red color.)

Another characteristic of IPM is that it usually has plenty of clusters of narrow triangles in planar and cylindrical regions (Fig. 2a). These narrow triangles result from the discretization of cylinder shapes or ruled surfaces on the mechanical part, and their connections have certain regular patterns. Large triangles are present in planar faces whose line boundary edges are not discretized.

Hence, an in-process model of NC simulation is a model with non-uniform vertex/triangle distributions. It is mainly composed of three types of triangles: narrow triangles, relatively small uniform triangles and large triangles.

4. Overview of the method

In fact the work is to establish a high-level geometric structure based on the unstructured triangle mesh. Considering these characteristics of NC simulation, and taking advantage of them, we have devised an efficient method for segmentation and feature recognition, which is uniquely adapted to in-process model. Fig. 4 shows the overall strategy of the method which consists of the following steps: (1) Polyhedral B-rep model construction: this is the pre-processing of the IPM, including offcut removal, IPM data error correction, the building of a manifold polyhedral B-rep model; Computation of vertex normal vectors, vertex curvatures, edge-xities, some attributes of triangle mesh, etc. These will be used for subsequent processing and for determining some thresholds. (2) Coarse segmentation: identification of sharp edges on the part's boundary, identification of narrow triangle clusters, segmentation of planar and most of cylindrical faces; (3) Shape index based segmentation and refinement: segmentation of non-ruled faces. Nine surface types are defined based on shape index. Every vertex of the shape is assigned a surface type label during local surface type recognition. Then connected regions are generated from

these clustered vertices; (4) STEP-NC feature recognition: feature class identification and its defining components, such as its profile(s), parameters.
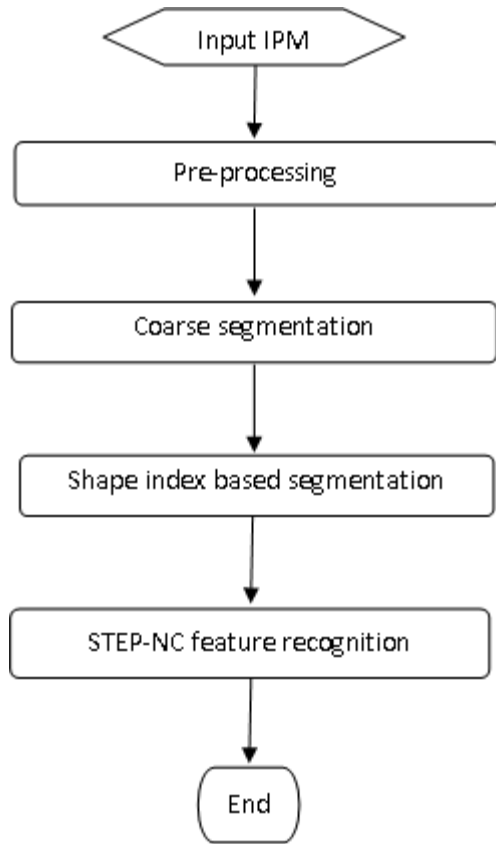

Fig. 4 Flowchart of the method

The pre-processing step includes removing offcuts, building a polyhedral B-rep model and compute some geometric values (like facet normal vectors, dihedral angles between two facets). The NC simulation software, such as NCSimul®, generally outputs IPM as an STL file. This original mesh data usually contains offcuts, isolated vertices/edges usually resulting zero-area triangles, incorrectly oriented triangles, holes, etc. For example, the IPM shown in Fig.3 has a part of the offcuts that is still connected to the finished part.

For amending the various defects of the IPM, approaches are briefed as follows: Isolated or non-manifold vertices/edges/faces are detected and corrected automatically (For example, if a zero-area triangle is found, an isolated vertex or edge is detected, then this triangle can be removed.); incorrectly oriented triangles are reversed automatically according to the orientation of their neighboring triangles; holes are filled interactively in consideration that the model can be a non-solid mesh, and a real boundary and a hole's boundary cannot be identified automatically; the offcuts are cleaned interactively since it is easy to make out them for a user. If an offcut is connected to the part, there might be some new holes left on the IPM after a user deletes it to clean the model, a mesh splitting operation with a polygon section is used and make sure the holes healed.

Therefore after repairing the model, in our case, we build a watertight polyhedral B-rep model, in which each triangle is built as an oriented face, and meanwhile the IPM data is corrected [2]. We adopt the half-edge data structure for the B-rep model. This procedure include elimination of offcut data, constructing/healing model topology, computing geometric values of the model and storing them to the extended attributes of the relevant entities (faces/edges/vertices). The pre-processed IPM can be saved as an alternative mesh file format, OFF, which is index based, and hence much smaller in size compared with the corresponding STL file. After finishing this step the IPM data has no duplicated vertices, no wrongly oriented normal vectors, no degenerated faces, no non-manifold entities, and no invalid face indices.

5.   In-process model segmentation

In this section, a new segmentation method is developed to identify machining region from triangular meshes of the in-process model of NC Simulation. After the identification of sharp edges and grouping narrow triangles, then small regular triangles are also grouped. From these groups, region boundaries are identified. To recognize the surface types for the groups of small regular triangles, shape index based segmentation is applied.

5.1 Identification of sharp edges

The edge-xity, which is a measure of an edge's concavity/convexity, is used to decide whether an edge is an obviously sharp edge or not. For a shared edge of two surfaces, the edge-xity m is defined as (Fig. 5) [36]:

$$m = (\mathbf{n_1} \times \mathbf{n_2}) \cdot \mathbf{a_1} \tag{1}$$

Where, $\mathbf{n_1}$, $\mathbf{n_2}$ are unit normal vectors of the two surfaces F_1 and F_2, $\mathbf{a_1}$ is their shared edge e's direction, represented as a unit vector.
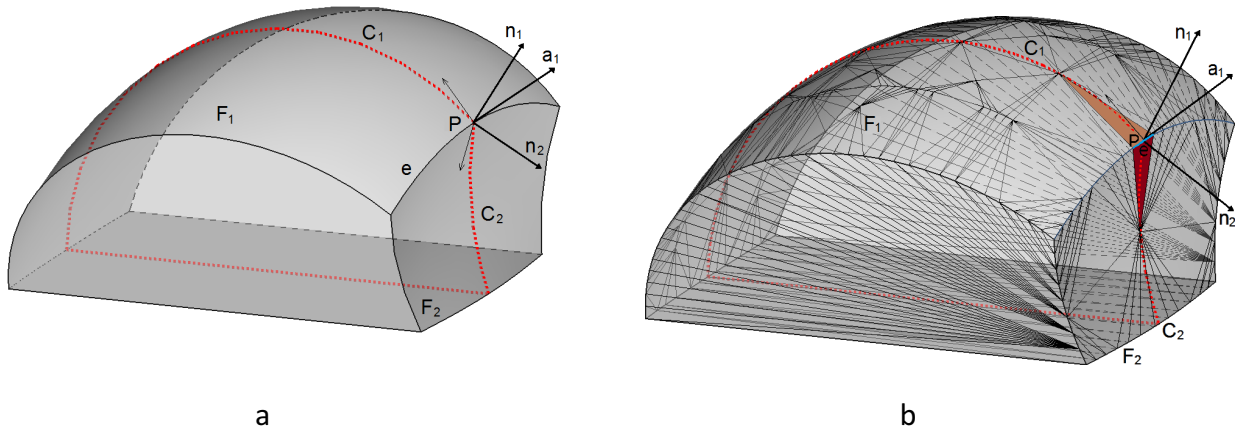

Fig. 5 Definition of the edge-xity: a) smooth model; b) discrete model

Here we suppose that if $|m| \geq \varepsilon_1$ (Where ε_1 is a small value, which can be adjusted by users.), then the edge is considered as a sharp edge. This constraint corresponds to the dihedral angle of the two facets. For example, when $\varepsilon_1 = 0.5$, it is equivalent to have a constraint on the dihedral angle between 30° and 150°. Fig.6 shows the obviously sharp edges filtered out in green.
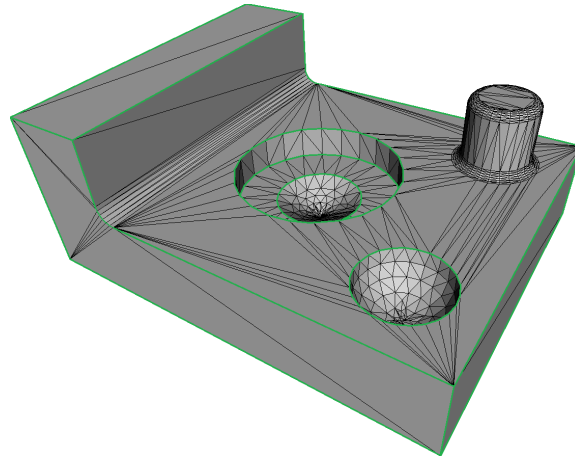
Fig. 6 Sharp edges filtered by edge-xities

5.2 Identification and grouping of narrow triangles

First we define the shape attribute of a narrow triangle. In a triangle, if one of its angle $\alpha < \varepsilon_2$, and no other angle greater than $(90°+\varepsilon_2)$, then the triangle is considered as a narrow triangle. Here it is called a pin triangle. Besides the triangle's shape, the grouping criteria also takes account the adjacency relationship and the size of the triangles. This step is to group the connected pin triangles in the mesh into regions. We have noticed that a pin triangle's head edge (i.e. the opposite edge of the triangle's smallest angle, the shortest edge in the triangle) is usually resulted from the discretization of a curved edge.

The grouping operation is performed as follows: For a pin triangle, if one of its side edges, which form the smallest angle, is not a sharp edge, and the adjacent triangle is an axe triangle, then add the adjacent triangle to this group. Do the same for the pin triangle's another side adjacent triangle. Repeat the grouping operation for the newly added triangle(s). In the group pin triangles are connected only by their side edges, and their head edges' lengths along a certain direction are similar. By the end of grouping operation, if there are less than 3 pin triangles in a group, we drop this group since they are unlikely resulted from a cylinder. Fig.7 shows a result after grouping pin triangles. (Here we set $\varepsilon_2=15°$ in the experiments.)

Next to determine the underlying surface types of the group regions of the pin triangles. The surface type should be planar or cylindrical in most cases. First, use the linear least square method to determine whether a group region can be fitted by a plane or not within a given tolerance. If the group region cannot be fitted by a plane, then use the Gauss image feature of a cylinder (which is mapping from a point on a surface to the unit normal of the surface at this point) to determine whether the region is a cylinder or not. The Gaussian image of a cylinder is a great circle on the Gaussian sphere, i.e. the normal of the facets forming the cylinder form a part of the great circle on the Gaussian sphere. Take 3 connected facets to start, if (1) there is a constrained plane possible passing thru the origin of Gauss sphere such that the distance of the 3 normal points from the constrained plane is below a threshold value and (2) the circle fitted on these 3 points forms a part of great circle on Gauss sphere (i.e. center = (0, 0), radius = 1), then the 3 connected facets form a cylinder. Add adjacent facet if it satisfies the rules. Mark identified cylindrical facets as convex (ridge) or concave (valley) depending on whether the facet normals are diverging or converging. If the region is not a cylinder, then set the region as undetermined surface type. For this case, the region probably be other type of ruled surface, or a part of a torus, or a

part of a surface. The region's surface type will be determined by the subsequent curvature-based segmentation method.
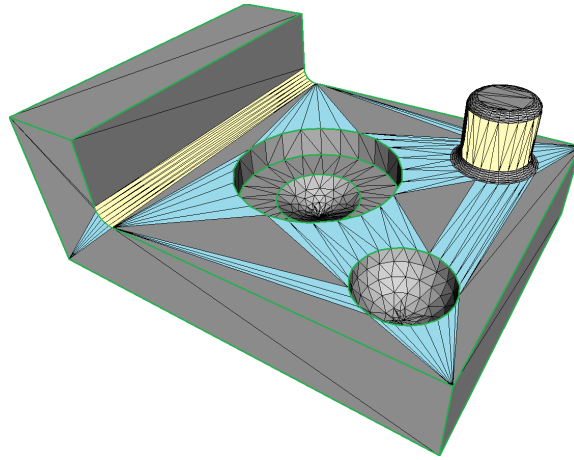

Fig. 7 Narrow triangle identification and grouping

5.3 Grouping small regular triangles

Small regular triangles are usually discretized by a general surface in an IPM. They have relatively uniform sizes and shapes, and they are dense in a region. The grouping purpose is to control the size of sub-components, which possibly makes the segmentation less complicated. The grouping operation tries to start from boundary triangles of a region being formed, and is performed as follows: select a seed triangle that is not yet grouped and shares an edge with a pin triangle's head edge. If there is no pin triangle in the IPM, from the ungrouped triangles, select a seed triangle that has a sharp edge. Check the seed's adjacent triangle, if the adjacent triangle is similar to the seed in terms of shape and area, then add it to the group. Again check the newly-added triangle, gradually expand the group with triangles that are similar to their adjacent ones until no more can be added. The grouping operations should not cross sharp edge boundaries.

In these groups, there are probably still some planar and/or cylindrical regions left, use the same methods in 4.2 to identify them. The result for this step is shown in Fig.8.
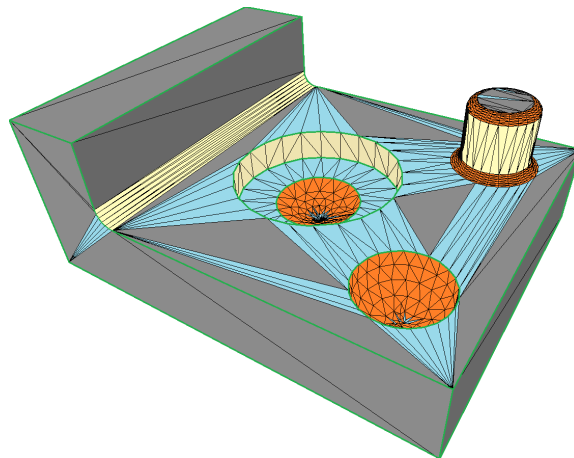

Fig. 8 Small regular triangle grouping

5.4 Group merging and identification of region boundaries

This step is to merge adjacent regions of the same types to reduce the number of regions obtained in the previous steps. For planar type of regions, if several groups and their surrounding ungrouped triangles are coplanar, they can be merged into one region. If two adjacent cylindrical type of regions have same axis and radius, they can be merged into one region as well. If several ungrouped big triangles are coplanar, they can be grouped as a region. After these expansion and merge operations, there probably are still some isolated triangles (like the ones in the bottoms of the surfaces as shown in Fig.8). If the adjacent triangles of the isolated one belong to a small regular triangle group, add the isolated triangle to this group.

After the above operations, each triangle belongs to a planar or cylindrical region, or a regular triangle group. In a region its external edges form its boundary. A region's boundary, like a face, possibly consists of one or more edge loops. A loop's direction keeps the direction of halfedges in boundary triangles. In a planar region, the loops can be marked as outer or inner loops. If a loop's normal vector conforms to the region's normal vector, then it is an outer loop; if a loop's normal vector is opposite to the region's normal vector, then it is an inner loop. In order to facilitate subsequent applications, if several consecutive boundary edges can form a circle/arc, they are labeled. Fig.9 shows the parts' regions after coarse segmentation. Regions are demarcated by blue boundaries. The underlying surface types of those groups in orange color will be decided in the next further segmentation procedure using discrete differential geometry methods.
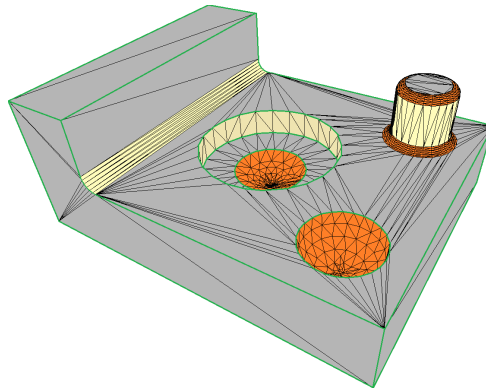

Fig. 9 Results of coarse segmentation

5.5 Shape index based region segmentation

A regular triangle group obtained above might be composed by several surface patches underneath. This phase is to identify the triangle set of these patches. The segmentation is performed on the regular triangle groups. Specifically, it is operated on the set of vertices of triangles within a group. A value could be associated with each vertex in the vertex set which somehow encapsulates the characteristics of the locality of the vertex. The definition of segmentation is the partition of vertices such that each region consists of connected vertices which have the same value within a tolerance.

In this phase the first step is to identify flat (planar) type of vertices based on the normal vectors of triangles. The method is as follows: for a vertex of a triangle in the regular triangle group, compute the normal vectors of the vertex's one-ring triangles, compare these vectors, if their directions are equal

within a tolerance, then mark vertex's type as flat. This simple yet effective method bypasses curvature since curvature calculation needs normal vector information as well.

The second step is to compute the curvatures. The principal curvatures and principal directions for 3D meshes are computed from the vertex's neighborhood vertices [37-39]. The method adopted here is based on the work reported in [37], which was improved by adding weights to the discrete shape operator matrix and modifying neighbor region [2]. Special attention should be paid to vertices on the model boundary. The boundary is the intersection of adjacent surfaces, which implies that the geometric property of each vertex on a boundary should be calculated from its surrounding neighborhood within one surface. If the neighborhood vertices are chosen across the boundary, high deviation of the principal curvatures of the vertex would occur. To lower the deviation, the anisotropic neighborhood is advised here. Based on the coarse segmentation result, which has marked the region's boundary edges, we choose the two-ring neighborhood triangles of a vertex exclusively within the region (whereas, for an inner vertex, choose only one-ring neighborhood triangles.), and therefore more accurate curvatures would be obtained. For example, in Fig.10, to compute the curvatures of vertex V, only the 7 neighboring vertices of V in the orange region are used, the other 4 vertices in the light orange region are not used. The first ring vertices are indicated by blue color, and the second ring vertices are indicated by light blue color. When there are scallop noises usually caused by a ball-end cutting tool machining in the IPM data, we can choose two or more neighborhood triangles of a vertex to compute the vertex's curvatures. This can ease the disturbance of the scallops.
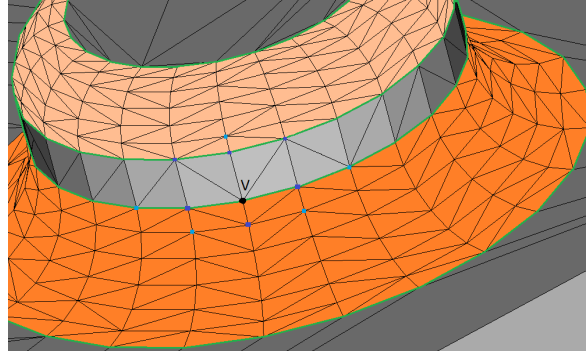

Fig. 10 Neighborhood vertex selection for computing curvatures of a vertex on region boundary

The third step is to compute the shape index value for the non-flat type vertices and classify them accordingly. Shape index, a single value for describing local shape, is selected as the mathematical basis for region segmentation [40]. The shape index, ranged [-1, 1], is calculated by the principal curvatures.

Shape index is defined as:

$$s = -\frac{2}{\pi}\arctan(\frac{\kappa_{max}+\kappa_{min}}{\kappa_{max}-\kappa_{min}})$$

(2)

Where, κ_{max} and κ_{min} are the maximum and minimum principal curvatures of a vertex, respectively.

Then for the non-flat types of vertices, nine surface types (spherical cup, trough, rut, saddle rut, saddle, spherical cap, dome, ridge and saddle ridge) are defined based on their shape index values (Fig. 11). Note: we add an isolate value 2, which is not calculated by the shape index formula and does not belong to the

shape index interval, for labeling flat vertices. Every vertex of the shape is assigned a surface type label during local surface type recognition.
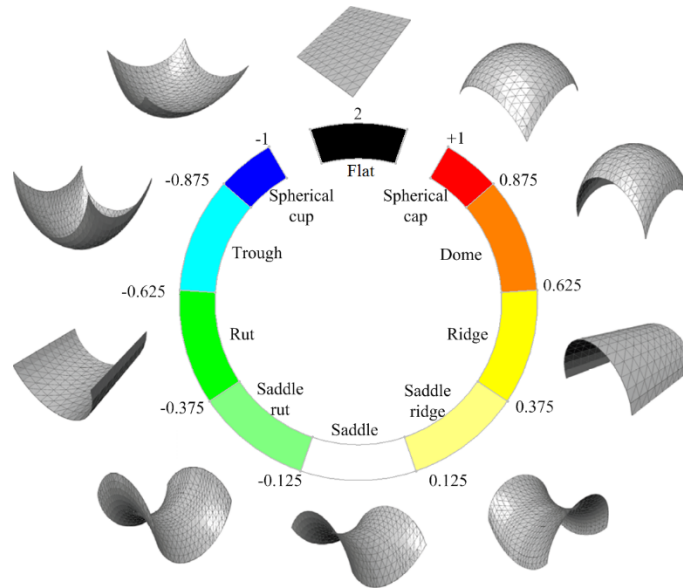

Fig. 11 Surface types, shape index scales and color scales

The fourth step is to generate connected regions by clustering vertices which have the same surface type. Then two operations are performed [19]: connected region growing to generate initial segmentation result, and region refining, which aims to reduce over-segmented regions and to improve the segmentation result. When a vertex yet unassigned with a region label is met, the vertex is marked with a region label by its neighbor condition. With this associated region label, the vertex is added to an existing region or creates a new region.

The steps for the region segmentation of a regular triangle group are illustrated in Fig. 12.

```
        ┌──────────────────────────────────────────────┐
        <  Set of vertices of a regular triangle group   >
        └──────────────────────────────────────────────┘
                            │
                            ▼
        ┌──────────────────────────────────────────────┐
        │         Identify flat type of vertices         │
        └──────────────────────────────────────────────┘
                            │
                            ▼
        ┌──────────────────────────────────────────────┐
        │ Identify surface types of non-flat vertices by │
        │                 shape indices                  │
        └──────────────────────────────────────────────┘
                            │
                            ▼
        ┌──────────────────────────────────────────────┐
        │   Vertex clustering and connected region       │
        │                 labeling                       │
        └──────────────────────────────────────────────┘
                            │
                            ▼
                  ╱─────────────────╲        N    ┌────────────────────────────┐
                 <  Partition satisfying? >──────▶│ Region merging and refining│
                  ╲─────────────────╱             └────────────────────────────┘
                            │ Y                                  │
                            ▼                                    │
        ╱─────────────────────────────────╲                     │
       ╱      Region segmentation results   ╲◀──────────────────┘
       ╲─────────────────────────────────────╱
```
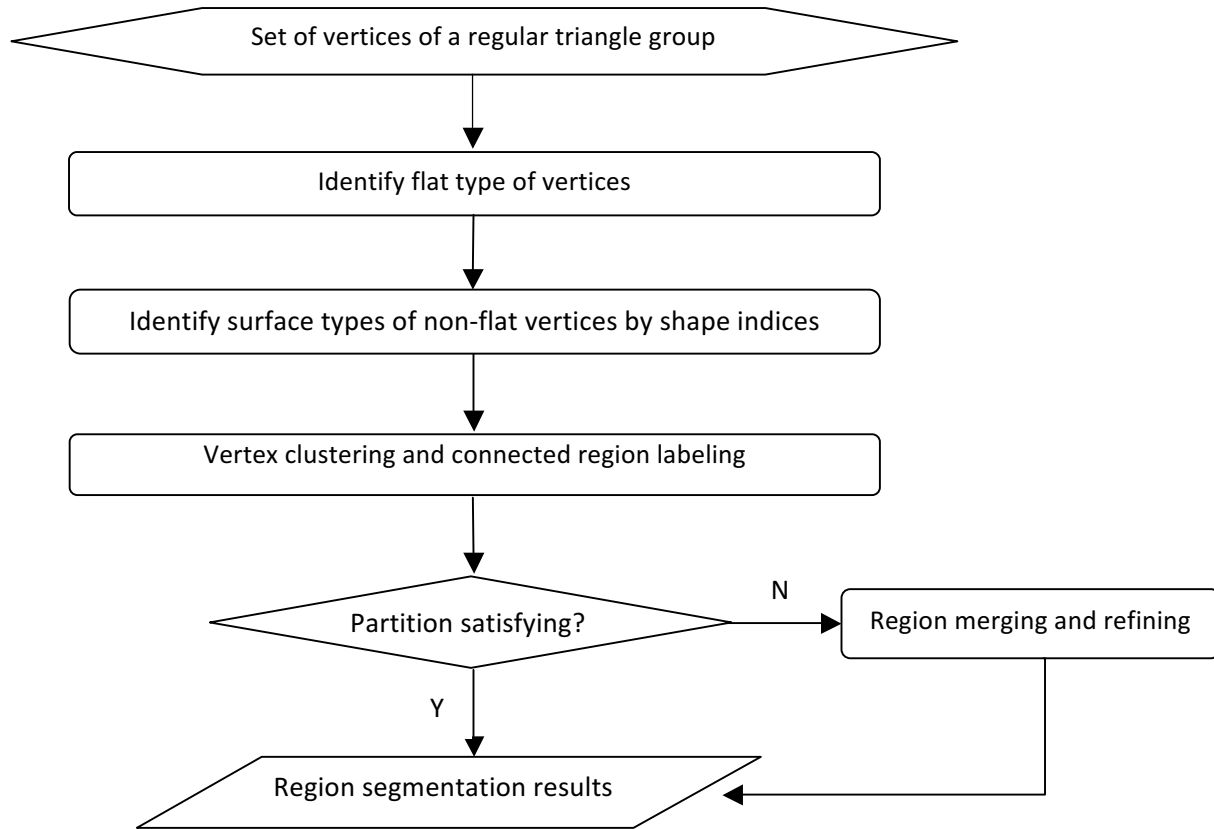
Fig. 12 Steps for the region segmentation of a regular triangle group

The IPM region segmentation established a higher structure on the triangle mesh. The result is a list of regions, and each region is given the following information: all facets (triangles) of the region, its surface type, and its boundary edges and its type (line, arc …).

6. Manufacturing feature recognition

This section elucidates STEP-NC manufacturing features recognition approach and classification-based rules. After a brief introduction of STEP-NC features, we presents rules only based on topological characteristics to classify four basic types of form features, which covers the majority of STEP-NC features. Then detailed explanation to recognize various types of STEP-NC features are given.

6.1 STEP-NC features

The manufacturing features defined in the ISO 14649 (STEP Data Model for Computerized Numerical Controllers) standard are often called STEP-NC features. ISO 14649 is a model of data transfer between CAD/CAM systems and CNC machines [41, 42]. It aims at standardizing the data formats used at the machine level, one key link in the entire process chain in manufacturing enterprises. It specifies machining processes rather than machine tool motion, using an object-oriented data structure. The

essential entity of the standard, "machining_workingstep", specifies the association between a manufacturing feature and a machining operation completed to that feature. It encapsulates design information (i.e. manufacturing feature) and manufacturing information, such as cutting tool, feedrate, cutting speed, coolant status. STEP-NC creates an exchangeable, workpiece-oriented data model for CNC machine tools, supports the direct use of product data from ISO 10303. The data model of the recent STEP AP242 [43] for machining form feature is consistent with STEP AP214 and STEP-NC.

Fig. 13 shows the STEP-NC manufacturing feature, which is an abstract supertype of region, transition feature and 2½D manufacturing feature. This figure indicates that a STEP-NC feature is defined implicitly by a set of parameters. A feature with a profile, like pocket, slot, step, is defined using an open/closed boundary with a depth or a travel path. For example, information of feature's depth, placement, 'feature boundary' and 'bottom condition' is used to define a closed pocket. The 'feature boundary' (a closed and non-self-intersecting profile) of a closed pocket is defined in the XY plane of the feature coordinate system. The pocket is formed by moving the profile a distance of the depth along the negative Z-axis direction. The 'bottom condition' specifies the pocket's possible bottom shape. On the pocket's bottom there might exist islands (bosses), which are not cut during the milling of the pocket. A pocket may have other parameters: slope (defining the angle between side faces and Z-axis), 'planar radius' (defining the edge fillet between side faces and the bottom) and 'orthogonal radius' (defining the edge fillet among side faces). Holes, slots, steps, profile features, etc. are defined similarly.

A transition features, describing the type of connection between adjacent machining features, can be defined only on the border of the two features. Chamfer (defined by the angle and a dimension) and 'edge round' (defined by a radius and optionally 2 dimensions) are two types of transition features in STEP-NC. A compound feature is a group of connected two or more features. 'Counterbore' consisting of 2 holes with different diameters, and 'countersunk' consisting of a tapered hole and a cylindrical hole, are two types of compound features. Their bottoms of the non-thru holes with larger diameters are the starting surfaces for the thru holes with smaller diameters.
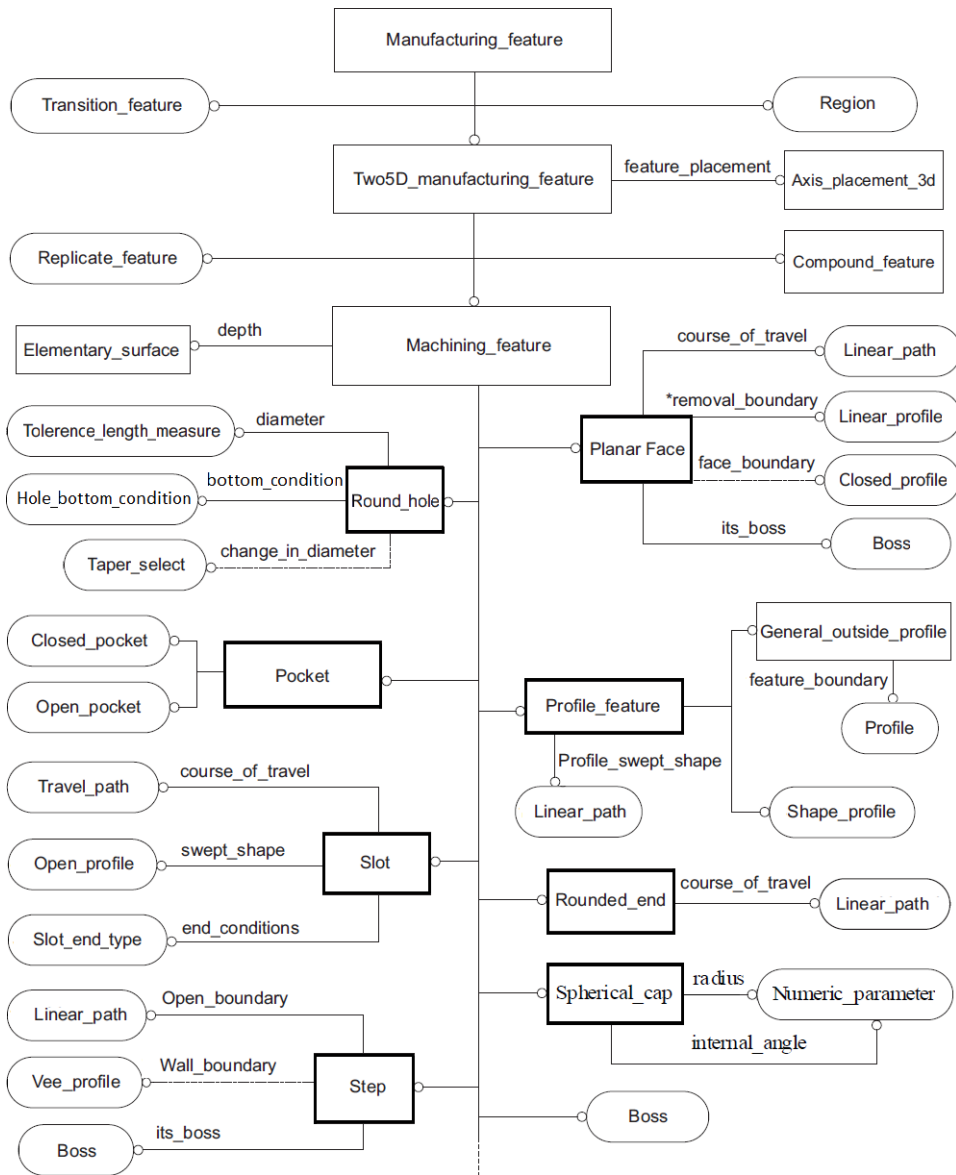
Fig. 13 STEP-NC features [41]

6.2 Feature classification

We classify form features based on the purely topological characteristics and we use graph theory to analyze them [36]. Form features imply potential machining information that makes their forms. Their geometric scope is generally larger than those of machining features. For each defined category the corresponding STEP-NC features are given. So the subsequent feature extraction algorithms can exploit and match the graph properties of each category.

Here form features are represented by the adjacency graphs, whose nodes denote faces, and arcs with concavity/convexity values denote edges [36]. For example, the form feature shown in gray in Fig. 14a is represented by the adjacency graph G shown in Fig. 14b. Removing the convex edges in G, we get the graph G$^+$, as shown in Fig. 14c.
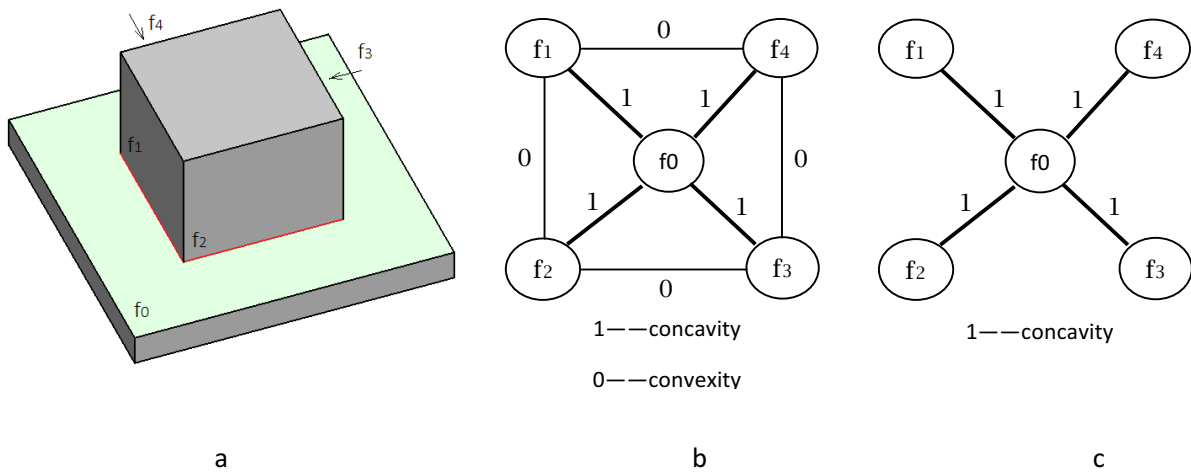
Fig. 14 Adjacency graph of a feature: a) a form feature; b) graph G; c) graph G^+

(1) Passage. A passage has no bottom, and consists of a set of faces forming a wall. The wall faces are connected by concave or convex edges. A concave edge is the one whose edge-xity m<0; whereas a convex edge is the one whose edge-xity m>0. If there are convex edges, their number are generally less than the number of concave edges.

A passage has the following properties (i or ii) (Fig. 15&16):

i)   The graph G^+ is a cycle and $G = G^+$;
ii)   The graph G^+ is one or several chains and graph G is a cycle.
    Where, G is the adjacency graph, G^+ is the graph after removing the convex edges in G.

The STEP-NC features that belong to the passage class are thru round holes and thru closed pockets (Fig.17).
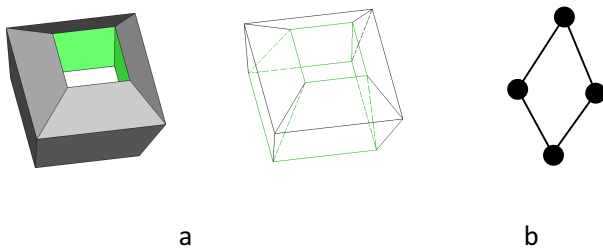
Fig. 15 Example of a passage: a) the passage; b) G / G^+
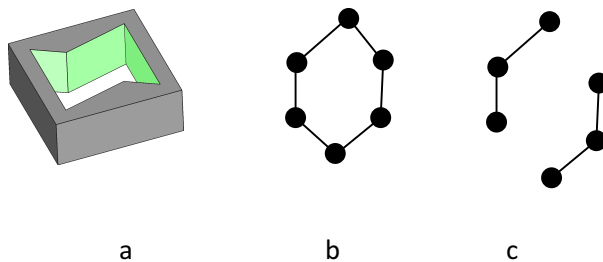
Fig. 16

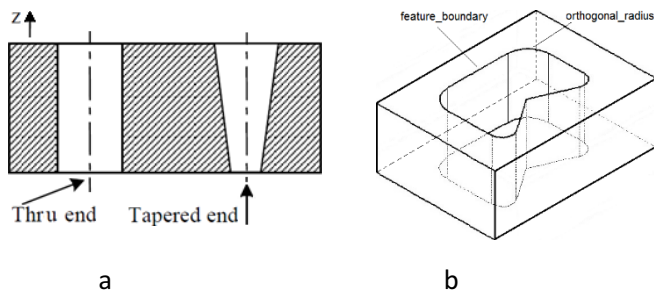Fig. 16 Another example of a passage: a) the passage; b) G; c) G^+

a

b

Fig.17 STEP-NC passage features: a) thru holes; b) thru closed pocket

(2) Depression. A depression has one access of entry and one bottom face connecting its wall faces via concave edges. It has the following properties (i and ii) (Fig.18):

i) Graph G^+ has a vertex v whose degree deg(v) = $\lambda(G^+)$, where $\lambda(G^+)$ is the number of cocylcles (i.e. number of vertices minus number of connected components, which is 1 in this paper.);

ii) Graph G-v obtained by suppression vertex v of G, is a cycle, whose concave edges is more than the convex edges in number.

The STEP-NC depression features are blind round holes, blind closed pockets, and slots with two ends.
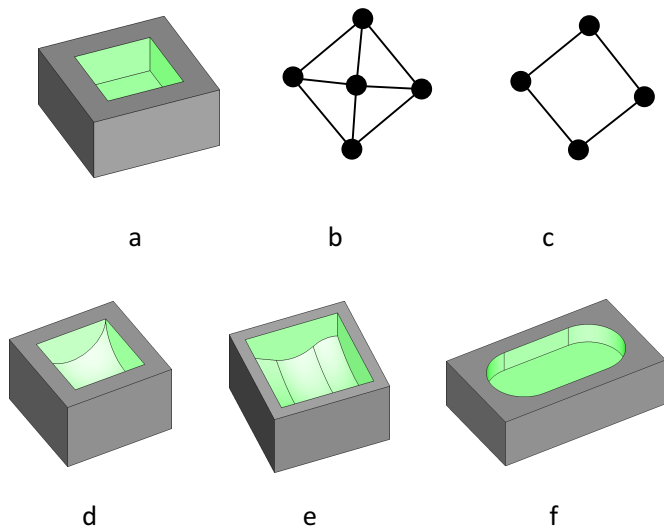

a

b

c

d

e

f

Fig. 18 STEP-NC depression features: a) blind closed pocket, planar bottom; b) G^+; c) G-v; d) blind closed pocket, radiused bottom; e) blind closed pocket, general bottom; f) slot with 2 radiused ends

A degenerated depression is the one whose bottom face degenerates to an edge or a vertex. Like the depression in Fig. 19, its bottom is vertex. In this case, we can add an isolated element v to the graph. If the graph G+v has the above properties, then the feature is a depression.

Fig. 19 A degenerated depression: a) the feature; b) G+v

(3) K-protrusion. A k-protrusion has one or several accesses of entry, and has one base face on which protrusions are attached via concave edges. It has the following properties (i and ii and iii) (Fig.20):

i) Graph G^+ has a vertex v whose degree $deg(v) = \lambda(G^+)$;

ii) Graph G-v contains k disjoint components.

iii) A component $(G-v)_i$, $(1 \leq i \leq k)$, it is a cycle whose concave edges is less than the convex edges in number.

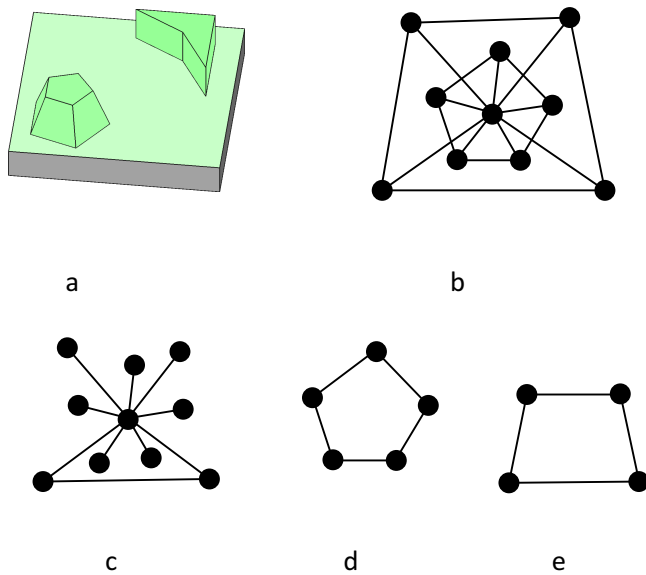The STEP-NC k-protrusion features are bosses, profile features and spherical cap.

a   b

c   d   e

Fig. 20 Example of k-protrusion: a) the k-protrusion; b) G; c) G^+; d) $(G-v)_1$; e) $(G-v)_2$

(4) K-slot. A k-slot has one or several accesses of entry, and has one base face connecting one or several open wall faces. It has the following properties (i and ii) (Fig.21):

i) Graph G^+ has a vertex v whose degree $deg(v) = \lambda(G^+)$;

ii) Graph G-v contains k disjoint chains $(G-v)_1, \ldots, (G-v)_j$, and/or isolated vertices $s_1, \ldots, s_k$.

The STEP-NC k-slot features are blind open pockets, slots that has at least one end is open, and steps.
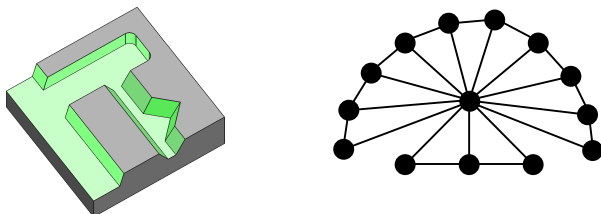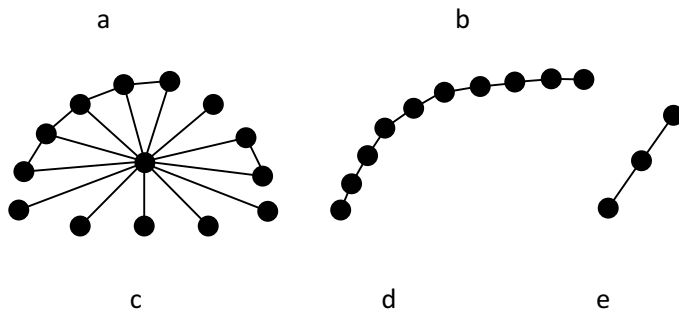
Fig.21 Example of k-slot: a) the k-slot; b) G; c) G^+; d) $(G-v)_1$; e) $(G-v)_2$

(5) Compound class. This type of form feature is composed by several simple features defined above. For example, a depression with an island on its bottom is a compound feature. The basic processing approach is decomposition the graph. Fig. 22 shows a compound and the graph decomposition.
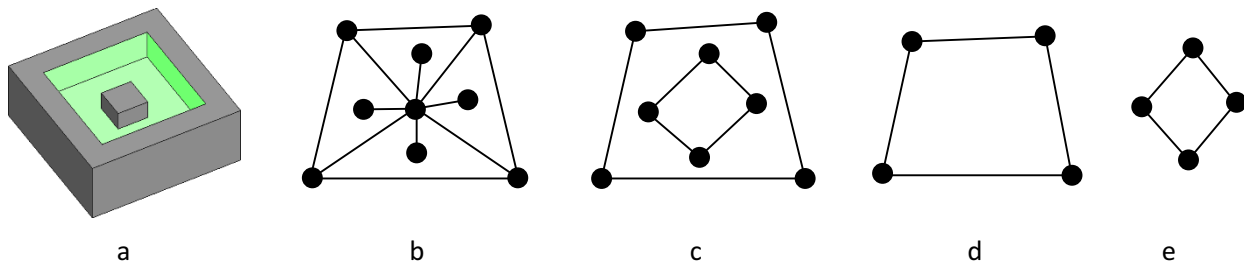

Fig.22 Example of a compound: a) the feature; b) G^+; c) G-v; d) $(G-v)_1$ (depression); e) $(G-v)_2$ (k-protrusion)

6.3 STEP-NC feature recognition

6.3.1 Flowchart of the algorithm

Generally speaking, the feature recognition is done in two phases. One phase is to identify the form feature class (passage, depression, k-protrusion, or k-slot), and then determine the corresponding STEP-NC feature class; the other phase is to extract the STEP-NC feature parameters according to the standard's specifications. Fig.23 shows the flowchart of the feature recognition algorithm.
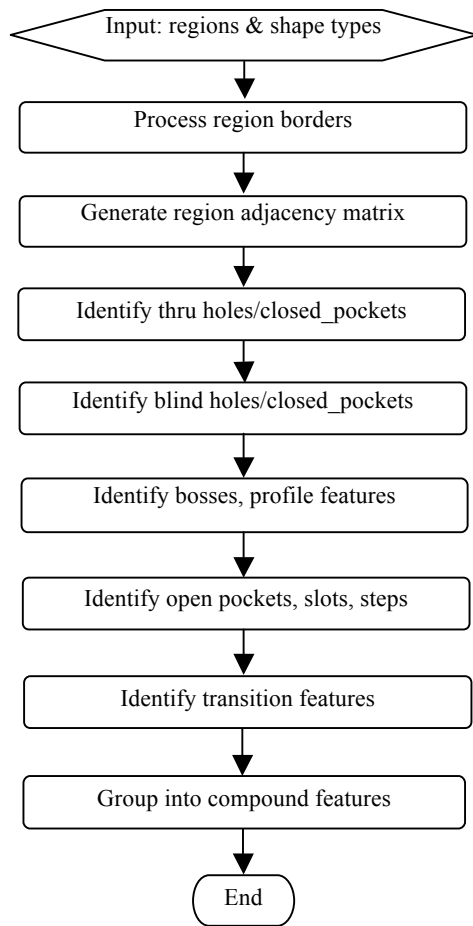
Fig. 23. Flowchart of the feature extraction algorithm

6.3.2 Process of region borders

After region segmentation, most of region borders are smooth, but in the small regular triangle regions, there are some segments whose borders are jagged. To keep the topological relationship of mesh unchanged, we adopt the method in [44], to find the shortest path between two border vertices that are currently connected with jagged edges. In Fig. 24 the jagged border edges of segment S consist of e_1-e_2, e_3-e_4. By searching the shortest path on the border, the alternative edges are found as e'_1, e'_3-e'_4, respectively.
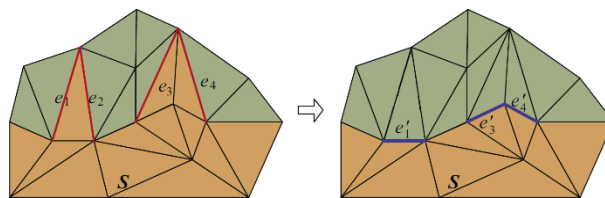

Fig. 24 Region boundary smoothing

Determine outer/inner loops of region borders. If two triangles sharing an edge are not in same region, the shared edge is a border edge. The border's direction is in conformance to the edge direction of the border triangle. For a planar region, the normal vector of a border is calculated according to its direction. If the border's vector is in the same direction as the region's, then the border is the outer loop of the region; If the border's vector is in the opposite direction to the region's, then the border is an inner loop the region.

Fit region borders with lines, arcs or spline curves. The profiles of a STEP-NC feature consist of lines, arcs and spline curves. This fitting process will facilitate to extract a machining feature and decide its type and its parameters.

6.3.3 Region adjacency matrix

In order to improve the efficiency of the algorithm, a data scheme originally used in image segmentation [45] — region adjacency graph (RAG) is adopted. This algebraic structure contains a set of nodes and a set of edges. Each nodes represents a connected region (i.e. a connected subset of the mesh), and each edge represents an adjacency between two regions. The algorithm input is the region segmentation result of an in-process model, which are regions represented by a collection of triangles and their respective shape types. Here the input is structured as a RAG. Then after finishing some relevant calculations (such as the convexity/concavity for region borders.), the RAG are transformed as a region adjacency matrix (which is similar to a face adjacency matrix). The adjacency matrix is a square symmetric matrix and its dimension is equal to the number of regions. The elements a_{ij} of this matrix describe the dependencies between regions:

$a_{ij}=$
0: region i and region j are not in contact

type: diagonal elements indicate a region type

1: regions i and j share a concave edge

-1: regions i and j share a convex edge

-2: regions i and j share a tangent edge

The extraction algorithm checks these relations frequently, so the creation of this matrix accelerate the calculations.

6.3.4 Extraction of thru holes and thru, closed pockets

These features belong to the passage class. Here we propose a novel approach to recognize them. First we calculate the number of passages in the in-process model by using Euler characteristic (also called Euler number). Then if the Euler number indicates that there are passages in the model, normal searching procedures are performed. This will certainly increase the efficiency of the algorithm.

Here is the explanation on how to calculate the Euler characteristic, which is a topological invariant, a number that describes a topological space's shape regardless of the way it is bent. The Euler characteristic, commonly denoted by χ, was classically defined for the surfaces of polyhedra, according to the formula [46]

$$\chi = V - E + F \qquad\qquad (3)$$

Where V, E, and F are respectively the numbers of vertices, edges and faces in the given polyhedron.

On the other hand, the Euler characteristic of a closed manifold shape can be calculated from its genus g, which is number of passages, intuitively.

$$\chi = 2 - 2g \tag{4}$$

After pre-processing, the IPM becomes a manifold polyhedral B-rep triangle mesh model. Moreover its face number and edge number has the following relationship.

$$E = 3F/2 \tag{5}$$

Combining (1), (2) and (3), the number of passages is as under:

$$g = 1 + F/4 - V/2 \tag{6}$$

The number of triangles F and the number of vertices V can be easily got from the pre-processed IPM. If g>0 then perform the following detections.

A usual approach to extract thru holes and thru, closed pockets is to analyze region's inner loops with convex edges only [47]. Such a loop indicates the existence of a hole or a closed pocket. Starting from such a loop, the feature's sides (wall faces) are collected. The collection ends when side regions encounter a common region having an inner loop with convex edges. The two inner loops confirm that the hole or the pocket is bottomless. If the type of side regions is "rut", then the feature is a thru hole; otherwise it is a thru closed pocket. But as Fig.15 shows, the thru pocket is not originated from an inner loop. In this case the above "loop" hint cannot be used, but we can be sure that there is a passage in the part by applying formula (6), then the graph matching of the topological properties of a passage can be used to search the passage. So the Euler number can help find passages in some situation.

Next determine the feature's parameters. The location of the region containing the inner loop with bigger z-value is used to determine the location (x, y, z) and two vectors describing the orientation of the feature. The first vector normal to the region decides Z-axis, while the vector laying on the region decide X-axis. The region with this inner loop is the starting face of the feature. If the feature is a hole as decided above, the inner loop is in fact a circle, and the hole's radius equals to the circle's. If there are arcs between two adjacent loop edges forming concave angle, the radius of the arcs is used to store the value of orthogonal_radius parameter (Fig.17b).

6.3.5 Extraction of blind holes and blind, closed pockets

Extracting blind holes and blind, closed pockets is also done by analyzing region's inner loops with convex edges. These depression features have bottom regions with concave border edges. A side wall of the feature starts from the inner loops and ends at the bottom region thru a concave edge. The feature's location, orthogonal_radius of the feature can be decided in the same way as thru, closed pockets. To find the bottom, the side region having the greatest length in the direction of the vector normal to the region with the inner loop is selected. If such determined side region contacts with the region whose type is cylindrical (rut), or saddle, or spherical cup, and these two regions form a concave angle, the radius of such cylindrical or cup regions is selected as the value of planar_radius parameter and also pocket/hole depth is revised. Note that the feature's bottom region can be inclined. After the bottom region is selected, the pocket/hole bottom type must be determined. For example, a blind hole can have 4 bottom types: flat, conical, flat with radius, or spherical bottom (Fig.25). The bottom type can be easily decided by the bottom region's surface type. If the region representing the bottom has inner loops with concave edges only, the pocket has bosses. Boss is the part of the material which must be not cut during the machining. The pocket bottom region also can have inner loop with convex edges, determining other concave feature, for example next pocket lying on this bottom. All the side regions and the bottom region forming the feature are marked as recognized. Such regions are no longer analyzed by the extracting procedure. The starting region for the pocket or hole is not marked as recognized because such regions can be the part of other feature.

According to STEP-NC feature specifications, some slots can be considered as the special types of pockets. If the appropriate conditions are satisfied, the pocket is classified as slot. For example, if the pocket profile (the inner loop) consists of two parallel lines and two arcs, like a running track, and width of the pocket is twice the value of the radius on the pocket ends, then the pocket is a slot with two radiused ends (Fig.18f).
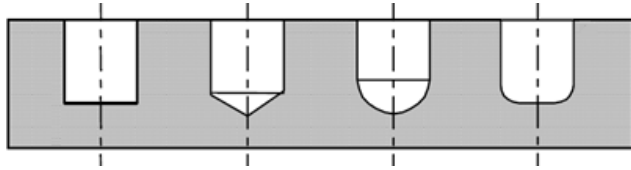

Fig.25 STEP-NC blind hole types

6.3.6 Extraction of bosses, profile features, spherical cap

In ISO 14649 a boss, as a dependent machining feature, having no separate workingstep assigned to it, can occur on features of planar face, blind closed/open pocket, and step. An outside profile (Fig.26a) is the removal volume of arbitrary shape from the outside shape of a workpiece. These types of features can be extracted by analyzing a region's inner loop with concave edges only, and the region is the base face. For a profile feature this loop is the "feature_boundary"; for a boss from this loop its boundary, which is defined at the top of the boss, can be inferred. Based on the length of the side regions of the boss or profile feature, measured in the direction of the vector normal to the base face of the feature, determines the parameter of Height of a boss or Depth of a profile feature. Regions forming the boss or profile feature are marked as recognized.

A spherical cap is circular about an axis of rotation. It consists of all points at a given distance from a point constituting its center. The center is defined by its placement, which is in the sphere's center. Fig. 26b shows a spherical cap sitting on a boss. When we found out all the connected regions originating from an inner loop with concave edges, we identify a region whose surface type is "spherical cap" (surface index≈1), then this region is obviously a spherical_cap feature. The feature's parameters, radius and internal_angle can be obtained by fitting the region with a sphere surface.
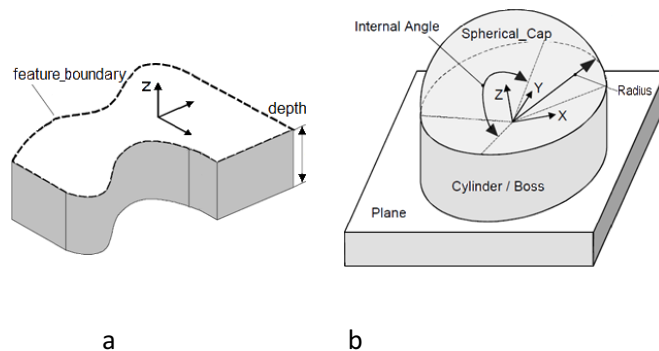

a   b

Fig 26. a) Profile feature; b) spherical cap

6.3.7 Extraction of open pockets, slots and steps

Search a region whose outer border has most number of connected concave edges. The number should be equal or greater than 2. This region is selected as the bottom face of the new blind feature (pocket or

slot). The connected regions sharing the concave edges with the bottom is included as the side faces. The side face with the greatest dimension in the direction of the vector normal to the bottom is used to decide the pocket depth. Side regions are marked as recognized. Again search a region whose outer border has most number of connected concave edges, and repeat the operation. This region might be same as the one previously selected since there might exist several disconnected open pockets based on same face.

Slot is regarded as a special pocket type under certain geometric constraints. The above operations have extracted blind open pockets. Among them there might exist such a slot whose one end is open and the other end is woodruffed, or radiused, or flat, as defined by ISO 14649 (Fig.21a). The specific slot type can be determined by checking the relevant geometric constraints. Note that slots with two ends being woodruffed/radiused/flat, or slots with loop end types, have already been extracted in 7.5 since they are in fact blind close pockets.

The topology of a thru, open pockets is face chain connected by concave edges. So region chains connected by concave border edges are searched among unrecognized regions. These regions form the side faces of a thru, open pocket, and from them the open_boundary parameter can be inferred. This method can only extract automatically thru, open pockets whose side faces have concave edges. For those whose side faces without concave edges, user intervention is needed.

Thru slots and steps (Fig.21a) are special thru, open pockets satisfying certain geometric constraints. A step feature is a volume of material removed from the top and the sides of the workpiece. A step has two planar faces defined by a V-profile, and connected by a concave edge. As for the thru slot, one common shape has a bottom face and two parallel side faces, connected by two concave edges. In the extracted thru, open pockets there might include some thru slots and/or steps. We can use these geometric characteristics to sort out thru slots and steps from the extracted thru, open pockets.

6.3.8 Extraction of transition features

ISO 14649 defines two types of transitions: chamfers and edge rounds. The rule for extracting chamfers is a chamfer's geometric traits plus its manufacturing knowledge. A chamfer is always an outer chamfer, which means all its edges are convex, as only this can be generated in a separate machining operation. Its length/width ratio is high and it has the normal vector nor parallel or perpendicular to Z-axis of the part coordinate system in general. Checking these properties on unmarked planar regions, chamfers can be selected. The chamfer's parameter angle_to_plane is the angle between one of its adjacent regions and the chamfer region usually with value of 45°or 30° from manufacturing point of view.

Rounding/blending features including edge rounds are extracted here using a unique approach developed by the authors. Curvedness map of the model is first computed, and then unmarked regions with high curvedness are identified as rounding features.

Curvedness is a positive value that specifies the amount or intensity of the surface curvature [40]. It is defined as:

$$c = \sqrt{\frac{\kappa_{max}^2 + \kappa_{min}^2}{2}}$$

(7)

Where κ_{max} and κ_{min} are the maximum and minimum principal curvatures of a vertex, respectively.

We use the curvedness value to detect rounding region. Vertices on smooth edges and transition regions have higher curvedness value, as thus a threshold can be given to detect them. This method is easy to be

realized and to operate. The curvedness range of a discrete model is dependent on its shape size. The threshold can be given as the average of the model's maximum curvedness and minimum curvedness. Fig. 27b shows the results of a rounded pocket's curvedness map. In most cases the identified rounding regions have surface types of rut/ridge/spherical_cup/spherical_cap. As for the ISO edge round feature, according to the definition, it is always an outer fillet, which can be manufactured in an extra machining operation. This means that only the rounding region filtered by the curvedness with surface type 'ridge' can be an edge round feature. The edge round feature's parameter 'radius' can be calculated by fitting the 'ridge' region with a cylindrical surface.
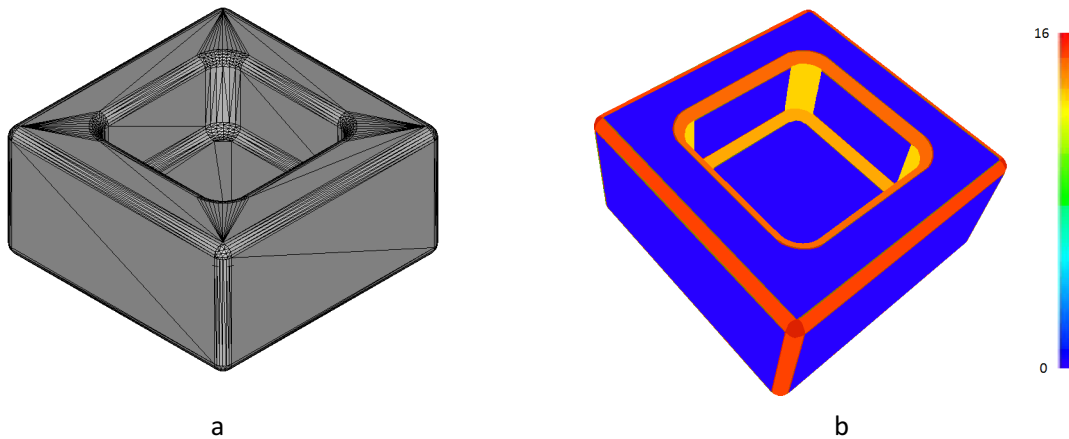

Fig. 27 a) Rounded pocket; b) Curvedness

6.3.9 Grouping into compound features

A compound feature is a feature composed of two or several simple features. Unlike a replicate feature, there is no regular spacing between the elements of a compound feature. Two examples of compound features in STEP-NC are counterbore_hole and countersunk_hole as shown in Fig.28.

After simple features in the part model are extracted, grouping simple features into compound features can be done by checking their geometric patterns. For example, a counterbore feature consists of two co-axial holes. One blind hole has a larger diameter than the other. The larger hole's bottom connects the small hole. To extract a counterbore, first select 2 co-axial holes: one is blind and the other is thru, then compare their diameters and connection relationship. If the two fit the pattern, then they are grouped into a counterbore feature.
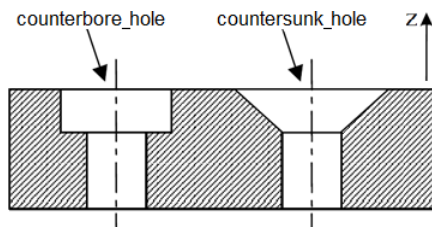

Fig.28 Counterbore and countersunk holes

In some situation a feature is broken due to its intersection with other feature(s). An example is shown as in Fig.29. The thru hole intersects with the thru pocket on the side faces. Because of this, two separate holes are extracted. The linking approach is similar to the one for compound feature grouping. During

linking of broken holes, the co-axial relationship and the same value of selected parameters (the hole diameter in this case) is used as the linking criteria.

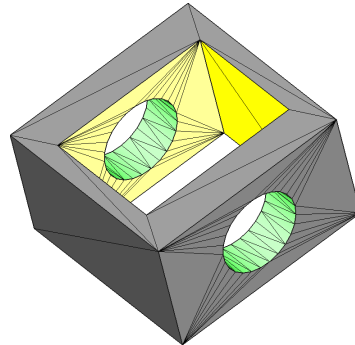In addition, the approach to group simple features into replicate features is similar.

Fig.29 Linking broken features

7. Case study

The proposed methods for feature recognition and reverse engineering of IPM were verified by developing the software. The input data, the part's IPM, was generated by NCSimul 9.0, which uses a G-code program running on the Siemens 840D CNC controllers. The IPM was output after the NC simulation as an STL file, and then was converted into an OFF file, which unified duplicated vertices and therefore was much smaller in size. The software was developed with Microsoft Visual Studio 2012 (C++). Its interface was created with Qt, and OpenGL graphic library was used for the visualization. For processing the triangle mesh, OpenMesh and CGAL developing libraries were used. The software was run on a computer with an *Intel Core CPU 2.4GHz and 4Gb RAM*.

For the example part shown in Figures 6-9, its IPM has 631 vertices and 1256 triangles. The total processing time after feature recognition is 23ms, and the result is displayed in Fig. 30.

a   b
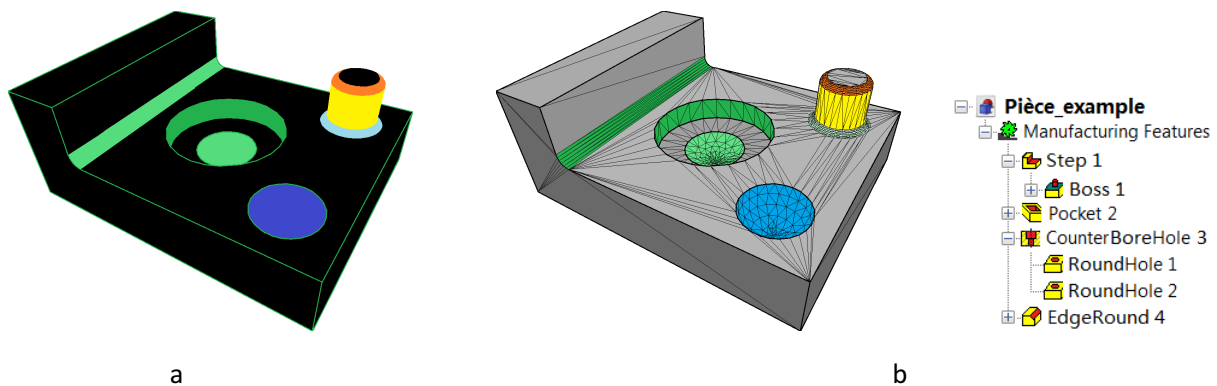
Fig. 30 Result for the example part: a) segmentation result; b) features got from the recognition

The test software's interfaces and another test part are shown in Fig. 31. The IPM treatment functions are under menu "Meshes". The machining feature recognition functions are under menu "NC-Features". This part's IPM shown in Fig. 2a has 1556 vertices and 3112 triangles. The total processing time is 57ms. The region segmentation result for the test case is shown in Fig. 31. The mesh segments of the part "Pièce_test" include 1 spherical cap, 1 ridge, 2 saddles, 9 spherical cups, 28 ruts and 19 planes.
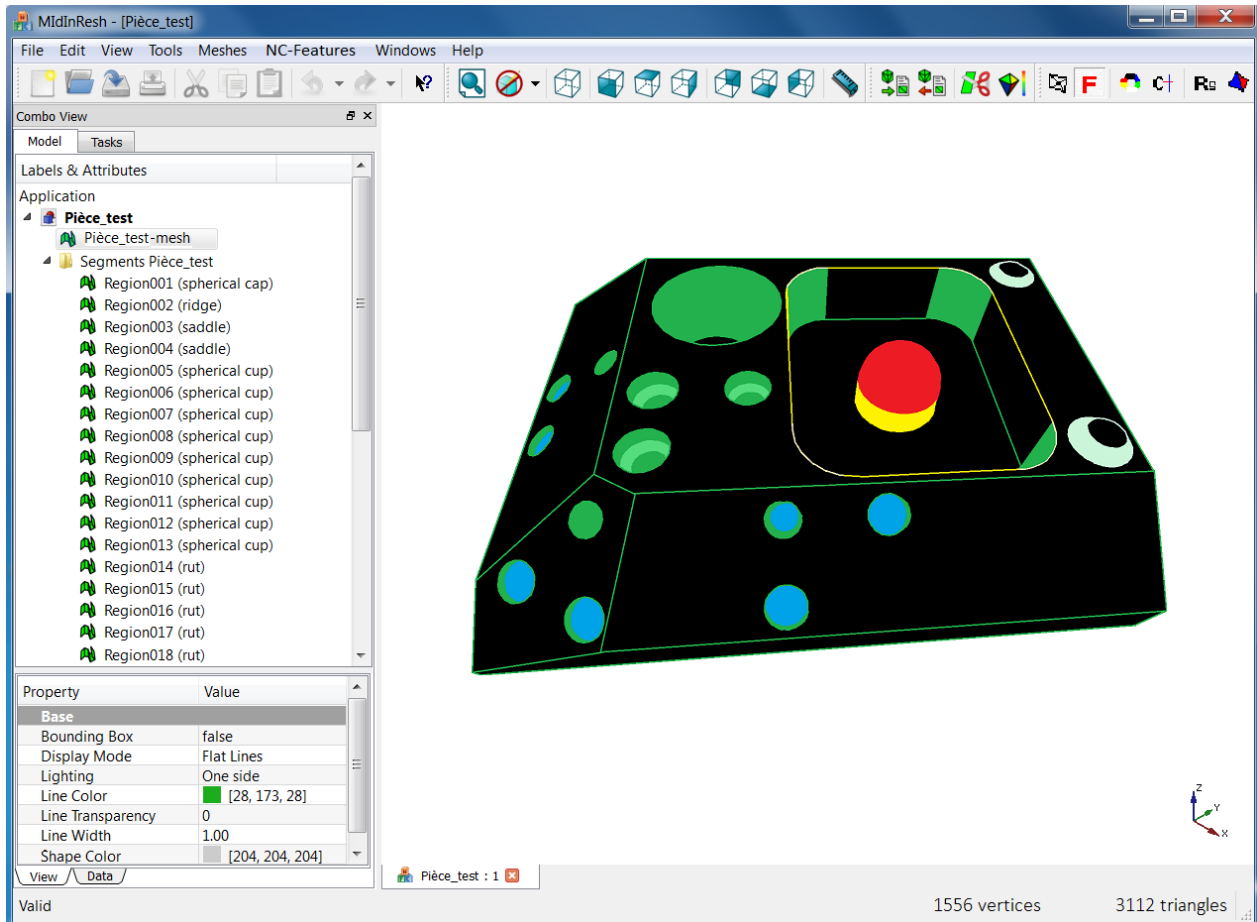

Fig. 31 Screenshot of the software for region segmentation of the test part

After the feature recognition, we can know that the model has 20 features in total, presented on the left tree list (Fig.32)—1 square pocket with 1 boss (spherical cap); 1 three-axis pocket; 3 counterbore holes; 1 planar face (i.e. the part's top face) with 2 bosses; 1 chamfer on the square pocket's entrance border; 9 holes with different bottom conditions on the part's inclined faces; and 1 slot with woodruff ends on a side face. When the feature's name in tree list is clicked, the feature's parameters are shown in the bottom-right table.
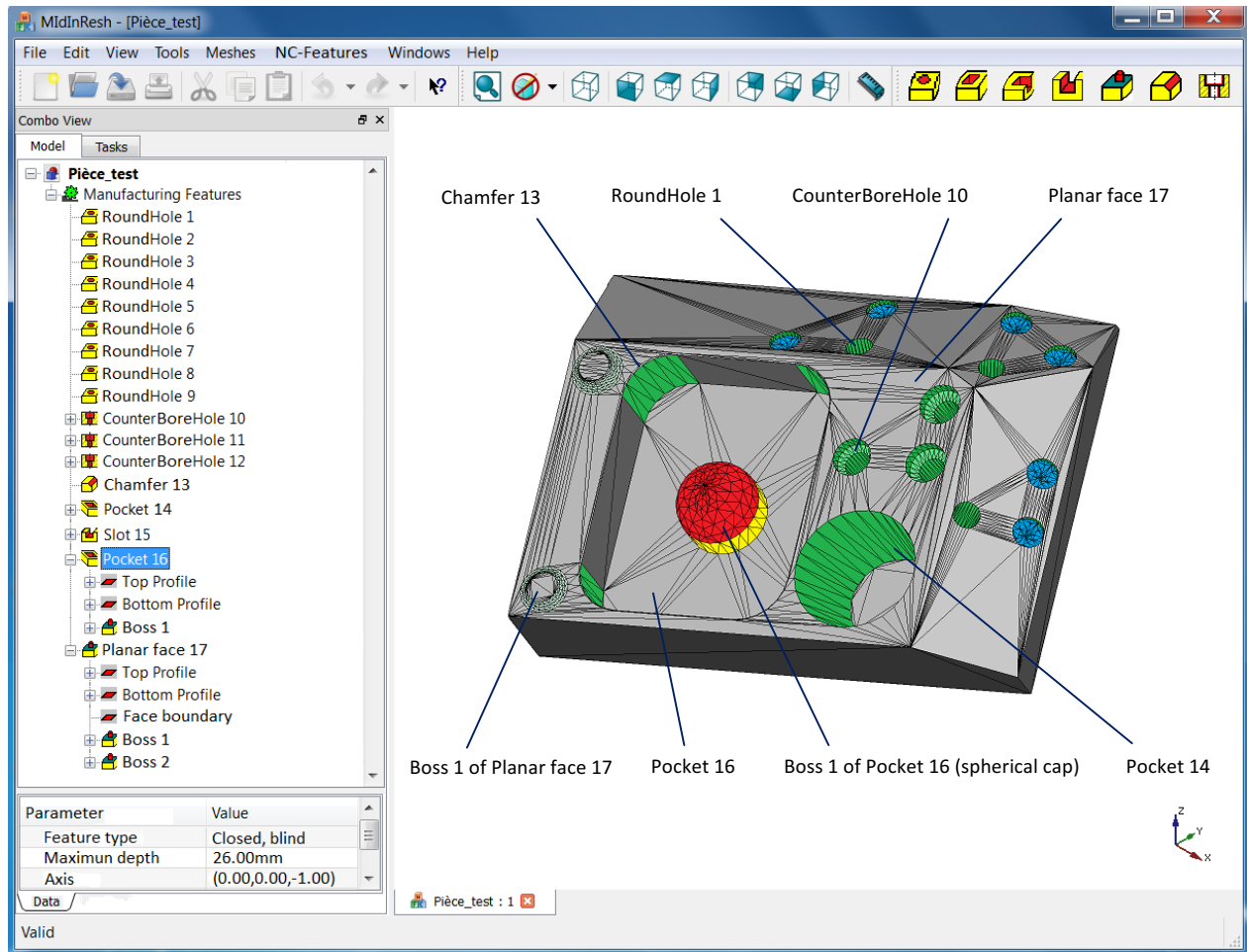
Fig.32 Screenshot of the software for feature recognition of the test part (annotations added manually)

A quantitative analysis to the reconstruction was performed. We chose some parameters from this part's CAD model as a standard reference and compared them with the recovered ones by the proposed approach. Three features in the CAD model shown in Fig. 33 are selected for the comparison. From Table 1, we can see the recovered values are very close to those of the CAD model, especially for features placed horizontally or vertically. In column "Error" of Table 1, the error is computed as the distance of two positions; and for two vectors, the error is computed as their angle in radian. These errors are deemed to be resulted mainly from the simulation discretization, and not from the reverse approach.
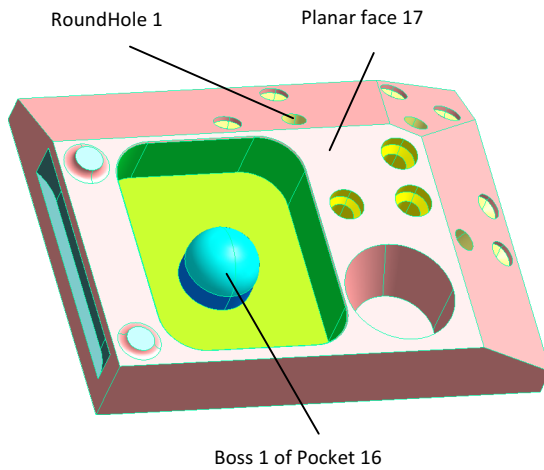
Fig. 33 CAD model of the test part

Table 1: Quantitative comparison to the reconstruction

Features	Checked values	Values got from CAD model	Values got by RE of IPM	The error
Boss 1 of Pocket 16	Center (x,y,z)	(-34, -16.5, 20)	(-34.000, -16.501, 20.000)	0.001
	Radius	16	16.000	0.000
Planar face 17	(n_x,n_y,n_z)—normal vector	(0, 0, 1)	(0.000, 0.000, -1.000)	0.000
	p— distance from origin to plane $(xn_x + yn_y + zn_z = p)$	36	36.000	0.000
RoundHole 1	Axis vector (n_x,n_y,n_z)	(0,0.5,0.866)	(0.000,0.498,0.867)	0.014 rad
	Axis position (x,y,z)	(12, 46.8,31.5)	(12.000, 46.789,31.504)	0.012
	Diameter	10.3	10.369	0.069
	Depth	32	31.886	0.114

8. Conclusions

The presented method in this research for feature recognition and reverse engineering of an in-process model has the following features: (1) It uses shape index to segment a mesh for the first time, to our best knowledge. To decide the local surface type of a mesh, only one value, the shape index, is sufficient, whereas other methods usually use a combination of two or more types of curvatures. (2) It adopts a progressive segmentation method according to the IPM characteristics. In this way each region of the IPM can be partitioned using the most effective algorithm. (3) The proposed form feature classification approach is only depended on the solid's topology, and it can cover a rather large scope of forms. Degenerated and compound form features can also be handled. Based on the classified forms, STEP-NC features can easily be recognized. (4) Two novelties are presented in the feature recognition approach: Euler characteristic is applied to aid the extraction STEP-NC passage features (thru holes and closed

pockets, etc.), and the model's curvedness, a shape descriptor, is used to extract transition features and edge rounds. The recognition input data is the segmentation result－regions with their shape types. The output is STEP-NC features, including the feature types, their defining components, such as profiles, placements, scalar parameters.

The key advantage of our method comes from the adoption of the shape index derived from the two principal curvatures for mesh segmentation. Other curvature based methods usually use the Gaussian and mean curvatures, or the principal curvatures, or combination of them. Some problems of other methods are: size and local shape are coupled; rules to classify the local shape is complex; surface shape variation is hard to be shown by only one type of curvature in a color scale. Whereas, the shape index is scale invariant, and is quite consistent to the intuitively distinct shapes, which may facilitates a user interface design by this single parameter instead of two. Besides, parabolic curves on a surface can neatly be distinguished, but the Gaussian or mean curvatures cannot.

The future work for improvements of the method may imply the following: (1) improving curvature accuracy of vertices on boundaries. Vertices of the mesh should be first classified as vertices of the form, vertices on edges and vertices inside faces by applying tensor voting theory [13]; (2) region boundary smoothing approach. Both global direction and local direction of the boundary should be considered, especially at the intersecting point of several boundaries. Currently only local direction is handled; (3) refining classification rules of form features, and detailing the graph's topological properties for degenerated and compound form features, as well as the relevant graph operation methods. For example, the number of concave or convex edges of graph G-v decides a depression or protrusion, which is not precise in some rare cases. Consider using edge-xity instead of number of concave edges; (4) refining approaches to extract open pockets and profile features, and employment of manufacturing knowledge to support the extraction algorithm. An open pocket can be rather complex, needing further manufacturing information to decide its exact STEP-NC feature type. Similar situation exists for transition features.

REFERENCES

1. Altintas, Y., Kerting, P., Biermann, et al, "Virtual process systems for part machining operations", CIRP Annals – Manuf. Tech., Vol. 63, pp. 585-605, 2014.

2. Anwer, N., Yang, Y., Zhao, H., and Coma, O., "Paul, J.: Reverse engineering for NC machining simulation", IDMME'2010-Visual Concept 2010, Bordeaux, France, pp. 1-8, 2010.

3. Zhang, X., Nassehi, A., Newman, S.T., "Feature recognition from CNC part programs for milling operations", Int J Adv Manuf Technol, Vol. 70, pp. 397–412, 2014.

4. Attene, M., Falcidieno, B. and Spagnuolo M., "Hierarchical mesh segmentation based on fitting primitives", Visual Comput., Vol. 22, pp. 181-193, 2006.

5. Shamir, A., "A survey on mesh segmentation techniques", Computer Graphics Forum, Vol. 27, No. 6, pp. 1539-1556, 2008.

6. Shamir, A., "A formulation of boundary mesh segmentation", 3DPVT, Vol. 4, pp. 82-89. 2004.

7. Benko, P. and Varady, T., "Segmentation method for smooth point regions of conventional engineering objects", Computer-Aided Design, Vol. 36, pp. 511-523, 2004.

8. Sunil, B. and Pande, S., "Automatic recognition of features from freeform surface CAD models", Computer-Aided Design, Vol. 40, pp. 502–517, 2008.

9. Xiao, D., Lin H., Xian C. and Gao S., "CAD mesh segmentation by clustering", Computers & Graphics, Vol. 35, pp.685-691, 2011.

10. Kim, S.-K. and Kim, C.-H., "Finding ridges and valleys in a discrete surface using a modified MLS approximation", Computer-Aided Design, Vol. 37, pp. 1533-1542, 2005.

11. Sun, Y., et al. "Triangle mesh-based edge detection and its application to surface segmentation and adaptive surface smoothing", Proceedings of International Conference on Image Processing. Vol. 3. IEEE, pp. 825-828, 2002.

12. Bénière, R., Subsol, G., Gesquière, G., Breton, F. and Puech, W. "A comprehensive process of reverse engineering from 3D meshes to CAD models", Computer-Aided Design, Vol. 45, pp. 1382-1393, 2013.

13. Kim, H.-S., Choi, H.-K. and Lee K.-H., "Feature detection of triangle meshes based on tensor voting theory", Computer-Aided Design, Vol. 41, pp. 47-58, 2009.

14. Wu, J. and Kobbelt L., "Structure recovery via hybrid variational surface approximation", Eurographics 2005, Vol. 24, pp. 277-284.

15. Besl, J. and Jain R., "Segmentation through variable order of surface fitting," Transactions of IEEE on Pattern Analysis and Machine Intelligence, Vol. 10, No. 2, pp. 167-192, 1988.

16. Yan, D.-M., Liu, Y. and Wang, W. "Quadric surface extraction by variational shape approximation," Geometric Modeling and Processing-GMP. Springer Berlin Heidelberg, pp. 73-86, 2006.

17. Lavoué, G., Dupont F. and Baskurt, A., "A new CAD mesh segmentation method, based on curvature tensor analysis," Computer-Aided Design, Vol. 37, pp. 975-987, 2005.

18. Razdan, A. and Bae, M.S., "A hybrid approach to feature segmentation of triangle meshes", Computer-Aided Design, Vol. 35, pp. 783-789, 2003.

19. Zhao H., Anwer N. and Bourdet P., "Curvature-based registration and segmentation for multisensor coordinate metrology", Procedia CIRP, Vol. 10, pp. 112-118, 2013.

20. Nalluri, S., "Form feature generation model for features technology", Ph.D. thesis, Indian Institute of Science, 1994.

21. Shah, J.J. and Rogers M.T., "Expert form feature modeling shell", Computer-Aided Design, Vol. 20, pp. 515–524, 1988.

22. Han, J.H., Pratt M. and Regli W.C, "Manufacturing feature recognition from solid models: a status report", IEEE Trans. on Robotics and Automation, Vol. 16, No. 6, pp. 782-796, 2000.

23. Joshi, S., "Graph-based heuristics for recognition of machined features from a 3D solid model", Computer-Aided Design, Vol. 20, pp. 58–66, 1988.

24. Vandenbrande, J.H. and Requicha, A.A.G., "Spatial reasoning for the automatic recognition of machinable features in solid models", IEEE Trans. Pattern Anal. Machine Intell., Vol. 15, pp. 1-17, 2000.

25. Rahmani, K. and Arezoo, B., "Boundary analysis and geometric completion for recognition of interacting machining features", Computer-Aided Design, Vol. 38, pp. 845-856, 2006.

26. Rahmani, K. and Arezoo, B., "A hybrid hint-based and graph-based framework for recognition of inter acting milling features", Computers in Industry, Vol. 58, No. 4, pp. 304-312, 2007.

27. Sundararajan, V. and Wright, P.K., "Volumetric feature recognition for machining components with free from surfaces", Computer-Aided Design, Vol. 36, pp. 11-25, 2004.

28. Geometric Ltd., http://feature.geometricglobal.com.

29. Sunil, B. and Pande, S., "Automatic recognition of features from freeform surface CAD models", Computer-Aided Design, Vol. 40, pp. 502–517, 2008.

30. Kiswanto G. and Azka M., "Automatic part primitive feature identification based on faceted models", IJCSI Int. J. of Computer Science Issues, Vol. 9, Issue 5, No. 2, pp. 126-132, 2012.

31. Karunakaran, K. P. and R. Shringi. 2007. "Octree-to-BRep conversion for volumetric NC simulation" *Int J Adv Manuf Technol.* 32:116-131

32. S. Xu, N. Anwer, and C. Mehdi-Souzani, "Machining Feature Recognition from In-Process Model of NC Simulation," Comput. Aided. Des. Appl., vol. 12, no. 4, pp. 383–392, 2015.

33. CIOANĂ, C., D. STAN, C. COSMA, and V. ȚUȚ. 2010. "Development of virtual objects using NC programs" Proceedings of the 2nd International Conference on Manufacturing Engineering, Quality and Production Systems (ISBN: 978-960-474-220-2) pp.133-138

34. Yan, X., K. Yamazaki and J. Liu. 2000. "Recognition of machining features and feature topologies from NC programs" Computer-Aided Design, 32:605-616

35. Shin, S. J., S. H. Suh, and I. Stroud 2007. "Reincarnation of G-code based part programs into STEP-NC for turning applications." Computer-Aided Design 39 (1):1–16

36. Anwer, N., "Méthodologie d'analyse de raisonnement pour la génération automatique des gammes d'usinage en fraisage", Ph.D. dissertation, ÉNS de Cachan, France, pp. 96, 2000.

37. Cohen-Steiner, D. and Morvan J.-M., "Restricted Delaunay triangulations and normal cycle", Proceedings of the 19th annual symposium on computational geometry. ACM, pp. 312-321, 2003.

38. Razdan, A. and Bae, M.S., "Curvature estimation schemes for triangle meshes using biquadratic Bézier patches", Computer-Aided Design, Vol. 37, pp. 1481-1491, 2005.

39. Kalogerakis, D., Simari, P., Nowrouzezahrai, D. and Singh, K., "Robust statistical estimation of curvature on discretized surfaces", Symposium on Geometry Processing, pp. 13-22, 2007.

40. Koenderink J. and Doorn A., "Surface shape and curvature scales", Imaging and Vision Compting, Vol. 10, No. 8, pp. 557-565, 1992.

41. ISO 14649 Part 10: General process data, 2002

42. ISO 14649 Part 11: Process data for milling, 2002

43. ISO 10303-242: 2014: Application protocol: Managed model-based 3D engineering.

44. Wang J. and Yu Z., "Surface feature based mesh segmentation", Computer & Graphics, Vol. 35, pp. 661-667, 2011.

45. Saarinen, K., "Color image segmentation by a watershed algorithm and region adjacency graph procesing", IEEE Int. Conf. Image Process, Austin, TX, USA, pp. 1021-1024, 1994.

46. Richeson, D.S., "Euler's Gem: The Polyhedron Formula and the Birth of Topology", Princeton University Press, 2008.

47. Pobożniak J., "Alogorithm for ISO 14649 (STEP-NC) feature recognition", Management and Production Engineering Review, Vol. 4, No. 4, pp. 50-58, 2013