Computers & Graphics 70 (2018) 51–61

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

Special Issue on CAD/Graphics 2017

Random cutting plane approach for identifying volumetric features in a CAD mesh model

Lakshmi Priya Muraleedharan, Shyam Sundar Kannan, Ameya Karve, Ramanathan Muthuganapathy*

Advanced Geometric Computing Lab, Department of Engineering Design, Indian Institute of Technology Madras, Chennai 600036, India

ARTICLE INFO

Article history:
Received 15 June 2017
Revised 14 July 2017
Accepted 15 July 2017
Available online 21 July 2017

Keywords:
Feature recognition
Mechanical features
Volumetric features
Plane cutting
Interacting features
CAD Models

ABSTRACT

This paper presents a method to identify regions that make up features like holes, slots, pockets as well as interacting features in a three-dimensional mesh of a computer-aided design (CAD)/Engineering model. Feature recognition is an important area in the field of CAD/Engineering with applications in model retrieval, creating an analysis model by defeaturing of the designed model for finite element applications, etc. Most feature recognition methods use either a cluster-based decomposition or feature line extraction through solid angles or curvature values, followed by graph-based heuristics. Such approaches require a user parameter for clustering or a threshold value for angle/curvature, neither of which is an easily deterministic one. The proposed algorithm identifies the features using contours generated by random cutting planes, followed by graph traversals (and not using heuristics) and without using parameter/threshold values. The algorithm can identify blind holes, through holes, slots and pockets. The geometry of most of the extracted features has also been identified using Gauss map. Interacting features have also been extracted and separated, which normally pose difficulty for most algorithms. Extensive experiments on CAD models from various benchmarks show that the algorithm is robust. Comparison with different algorithms (of which code was available) shows that our approach performs admirably and in the case of interacting features, the algorithm performs better than the existing ones.

1. Introduction

A polygonal mesh is a collection of vertices, edges and faces used to represent the surface of an object approximated as a polyhedron. The faces usually consist of simple triangles. However, quadrilaterals, convex polygons, and non-convex polygons have also been used.

Creating a computer-aided design (CAD) model is the first step in any design process. A CAD model can then be converted to a mesh model using a CAD software. A mesh model as representation helps in exchange of data as the native format of the CAD data is a proprietary one for the designer. Such a representation has become prominent in Engineering/CAD domain (Fig. 1a shows an example CAD mesh model). Unfortunately, the mesh models do not intrinsically capture any high-level information such as features present in a CAD model. Though there is no unique definition for a feature, the following are some of the ways to look at a feature:

- A feature is any entity used in reasoning about the design, engineering, or manufacturing of a product [29].
- A region of interest on the surface of a part [25]

Volumetric features in a model are regions of interest created by adding or removing volumes to or from it. Examples of such features from a CAD model include through holes, blind holes, pockets, slots etc. (Fig. 1b shows the extracted volumetric features (hereafter, we drop the term 'volumetric') of the CAD model in Fig. 1a). Identification of the features from a CAD mesh model has a number of applications in different fields. Defeaturing, i.e, suppressing the identified features for CAE analysis [5] is one of the most important applications. Model retrieval and classification is another area where identification of these features can be used [9]. Feature identification is also used in mesh enrichment applications [19]. In the area of machining, identifying features can help in determining the operations that have been used to generate a model.

Features need not be of stand-alone type. *Interacting features* are formed when one or more features interact with each other. Examples of interacting features include step cylinder (where one

* Corresponding author.
E-mail addresses: lakshmipriya369@gmail.com (L.P. Muraleedharan), shyamsundar33@gmail.com (S.S. Kannan), ameya.karve@gmail.com (A. Karve), emry01@gmail.com, mraman@iitm.ac.in (R. Muthuganapathy).

http://dx.doi.org/10.1016/j.cag.2017.07.025
0097-8493/© 2017 Elsevier Ltd. All rights reserved.

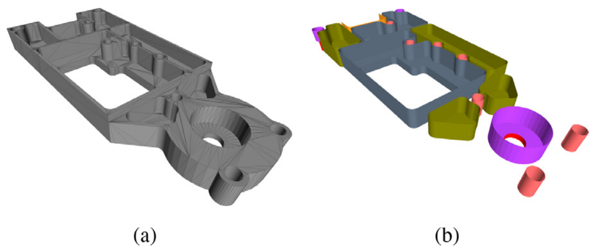


Fig. 1. (a) A CAD mesh model. (b) Identified features.

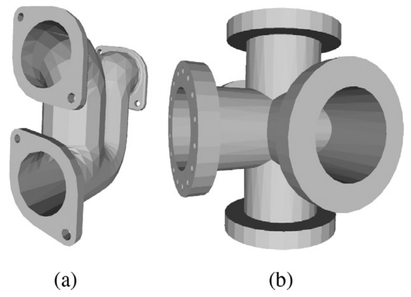


Fig. 2. Models having interacting features (multiple openings).

feature (cylinder) can originate/terminate from one of the faces of the other), intersection of different features creating multiple openings, typically more than two (Fig. 2) etc. Most of the algorithms do not capture the interacting features and have proven to be a challenging one [30]. Though feature recognition is a popular one in the area of CAD, most of the works have focussed on using boundary representation (B-Rep) of the model (e.g., [30], where interacting features have also been handled). On the other hand, only a few works address the recognition of features from a CAD model represented as a mesh.

1.1. Related work

The approaches for feature recognition in mesh models can be broadly classified into

1. *Decomposition-based:* Primitive based clustering approach [7] detects sets of triangles that form a plane, a sphere or a cylinder. Extension of the primitive approach to thin-plate CAD models using parallel planes has been presented in [13]. In [37], a clustering-based approach has been employed to detect the planar, cylindrical and conical surfaces. Tailor [21] tries to cluster parts of the mesh using curvature characterisation and has been shown to work for models with cylinder-like structures. Identifying tubular parts that connect larger portions together has been proposed in [22].
2. *Geometry-based:* In this approach, most of the methods try to identify the feature lines in the body and then subsequently use graph-based heuristics to identify the features. Identification of feature lines is typically done using a threshold on a geometric property viz. dihedral angle across edges [24], Solid angle [15,34], and Gaussian curvature [11,35,38]. Recognition of features for thin parts based on Gaussian and Mean curvatures has been proposed in [31]. Though slicing using planes has been employed in [2], the contours are analysed using threshold on the angles.

1.1.1. Summary

Though various criteria have been used in decomposition (or segmentation) of CAD models by different approaches [3], clustering, a predominantly surface based one, is one of the prominent approaches. However, this requires input parameters to arrive at a result [7,37], a value that is not a deterministic one a priori. Algorithms such as [21,22] can be considered as part-based segmentation as opposed to surface-based. Decomposing a CAD model into volumes using typical mesh decomposition algorithms has been proven to be a challenging one (see [8] for details). As mesh models by themselves do not store volumetric data, but only contain details of the surfaces, identifying the features from a mesh model using decomposition has become that much harder. Also, the entire CAD model is segmented into different parts or patches and not as features. Either the segmented patches have to be merged or parts have to be post-processed for features manually as there does not seem to be an automatic way.

The geometric-based algorithms, though works faster in general, is prone to threshold values for parameters such as dihedral angle or solid angle or Gaussian/Mean curvature. Identifying a threshold value for the desired output is also a cumbersome task as there is no generic way of finding a suitable parameter (often, a trial and error approach is needed). Some of the geometric methods appear to identify only through holes [13,35] (for a recent work on B-Rep models using declarative approach, see [23] and for a work on point-set models, see [20]).

In this paper, an algorithm for identifying the features in a CAD model represented as a mesh (assumed to be watertight) has been developed and implemented. The key approach is to identify the triangles that contribute to the feature. This is based on the fact that a plane passing through a mesh model having features will be surrounded by an outer contour and the triangles of the feature contributing to the inner contour. The methodology proposed in the paper is automatic in the sense that it does not require a geometric parameter such as threshold angle/curvature and hence it is robust. It also does not need parameters such as number of cluster that will decide on the output number of features. The following are our major contributions:

- Identify features without the necessity of tuning a threshold parameter, thereby achieving robustness.
- Planes are chosen in a random manner, avoiding the need to specifically orient them.
- Uses standard graph traversals for extracting features as opposed to graph-based heuristics.
- Algorithm can extract interacting features and separate them.

2. Preliminaries

Though the definition of features in [29] is generic, the following are perhaps the more appropriate ones for the indicated features.

Definition 1. A hole is defined as a round recess that is cut using a point tool. Thus, holes are a result of operations such as drilling, reaming or milling. A through hole is one that completely goes through the substrate, whereas a blind hole terminates in the substrate.

Definition 2. A pocket is defined as a recessed area surrounded by walls on all sides.

Definition 3. A slot is defined as a recessed area surrounded by walls on all but one side.

Definition 4. Whenever a feature originates/terminates in another feature, then the resulting structure has multiple openings and channels. This is referred to as interacting feature.

Fig. 3 shows a few examples of features - a through hole, a blind hole, a slot and a pocket. Do note that the features need not be cylindrical or square/rectangle but could be of other shapes such as conical or hexagonal etc. Interacting features have also been

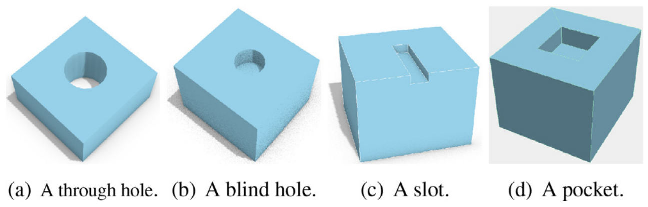


Fig. 3. Examples of features.

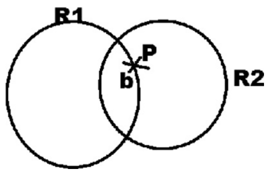


Fig. 4. Figure for Lemma 2.1.

included in this paper. The term *feature*, in this paper, hereafter refers to one of the above mentioned features.

2.1. Theoretical background

In general, topologically, a hole is defined as a structure which prevents the object from being continuously shrunk to a point. When dealing with topological spaces, a disconnectivity is interpreted as a hole in the space [36]. One consequence of such an interpretation is, if a hole does not go through a substrate, then topologically, it does not satisfy the definition of a hole. Thus, blind holes, slots and pockets do not satisfy the topological definition of a hole.

Even though the features are topologically different, in this section, it has been shown that there are some commonalities. Commonality is captured through an interaction between a plane P and the watertight triangular mesh M. The lemmas in this section capture the essence of commonality between the features. It also uses the fact that the contours that are generated due to intersection between the mesh and a plane are always closed.

Definition 5. The closed contours that are generated due to intersection between the mesh and a plane are called as rings. They are labelled as R_1, R_2, R_3 etc

Lemma 2.1. *The rings R_1, R_2 etc. cannot intersect each other.*

Proof. The intersection property is proven by assuming that R_1 and R_2 intersect and disproving the existence of such a case.

It is assumed that R_1 and R_2 intersect. Now if we travel in a counter-clockwise manner, all the points on one side are called as inside the body, and all the points on the other side are called as outside the body. Similarly with R_2, points are marked as inside or outside. Looking at Fig. 4, four distinct cases are possible:

1. Both the regions bound by R_1 and R_2 are inside the body.
2. Both the regions bound by R_1 and R_2 are outside the body.
3. The region bound by R_1 is outside the body and that by R_2 is inside the body.
4. The region bound by R_1 is inside the body and that by R_2 is outside the body.

To prove for cases 1 and 2, we take points P_1 and P_2 very close to P and on opposite sides of it. Then by definition, P_1 and P_2 cannot both lie inside or outside the body because the mesh is watertight, which is contradicted by the statements themselves. For cases 3 and 4, the region b is not clearly defined as being inside the body or outside it, which is not true for watertight meshes. Hence, the rings cannot intersect each other. □

Lemma 2.2. *Let R_1 and R_2 be two rings such that R_2 lies completely inside R_1 such that there is no other ring between R_1 and R_2. Then, the edges that make up R_2 are a part of a feature in the two-dimensional space represented by the plane.*

Proof. By Lemma 2.1, the rings are non-intersecting. For the inner ring R_2, points on one side of the ring are inside the body, and points on the other are outside it. The inner ring, when continuously shrunk, will create a discontinuity or fully shrunk. In either case, this can happen only if the ring is on a feature.

Hence the edges that make up the inner ring belong to a feature. □

Lemma 2.2 indicates that, if there is an inner ring, then there exists a feature. However, not all features will generate inner ring, e.g., a rib on a cuboid does not generate an inner ring. Hence, such ribs/bosses are not handled in this paper. Nevertheless, this approach helps in extracting one of the most difficult ones - interacting features (see Section 3.4.1).

3. The Algorithm

The algorithm consists of the following steps: (a) Identifying rings. (b) Determining Feature Triangles. (c) Extracting Features. and (d) Classification/Identification of Features.

3.1. Identifying rings

With the planes, termed as *cutting planes*, it is easy to see that the rings are an indicator of a feature. Fig 5 shows that, using the cutting planes, rings can be obtained irrespective of the feature type. These rings form the basis to detect the features.

Initially, the intersection between the cutting planes and the given mesh is performed to generate ordered closed loops, i.e., rings, from intersecting edges of the mesh. Let $R_1, R_2, \ldots, R_n$ be all such rings that do not lie inside any other ring. Then, all the other rings must definitely lie inside them as they do not intersect (as a direct consequence from Lemma 2.1). Edges of the interior rings are part of features, and they help in identifying the features in three dimensions.

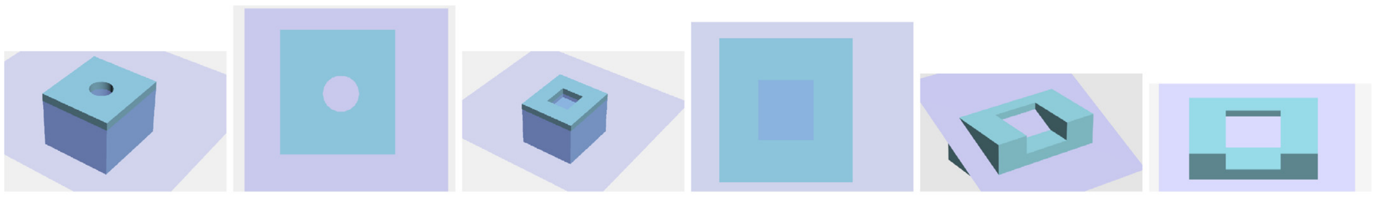
3.1.1. Choosing a cutting plane

An appropriate plane, in general, is not an easy task to identify (for e.g., an 'appropriate plane' for a cylinder would be the one that can give a circle on intersection with the cylinder). Determining such a plane has been shown to be difficult for CAD models and is not that straight-forward [27]. Hence, in this paper, the focus is not on finding an 'appropriate plane' but on the rings generated by any plane. Such an approach also avoids the need to solve an optimisation problem as in [27] or an approximate computation as in [33].

Randomisation of the cutting planes is used in the process of finding the two-dimensional rings. Let a point $P(x, y, z)$ be represented as $x = r\sin\theta\cos\phi$, $y = r\sin\theta\sin\phi$, $z = r\cos\theta$ where θ is the vector from the origin to the point from positive z-axis, and ϕ is the projection of this vector into the $x - y$ plane from positive x-axis.

Using $r = 1$ and randomly generating θ and ϕ ($0° \leq \theta \leq 180°$, $0° \leq \phi \leq 360°$), a cutting plane is generated whose normal direction is same as the normal direction at point P on the unit sphere and the offset for the plane is generated such that the plane cuts through the bounding box, thus increasing the probability that it cuts through the mesh.

Fig. 6(a) shows a model with two cylindrical holes. Figs. 6(b), 6(c) and (d) show a few of the cutting planes P_1, P_2, and P_3, respectively. Their corresponding rings are shown in Figs. 6(e)–(g).



(a) Hole. (b) Hole contour. (c) Pocket. (d) Pocket contour. (e) Slot. (f) Slot contour.

Fig. 5. Rings for different features through cutting planes.

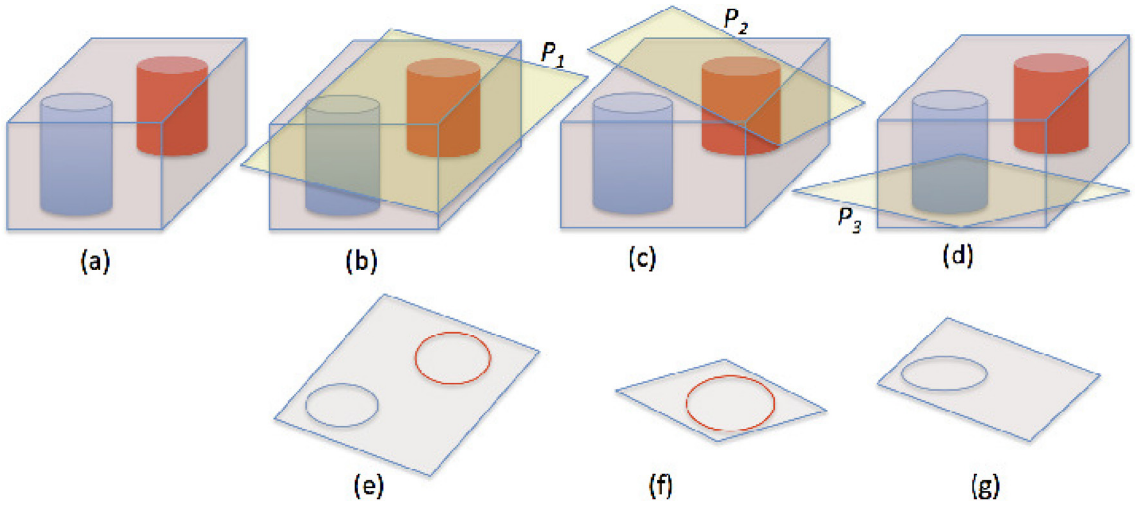


Fig. 6. Cutting planes and rings.

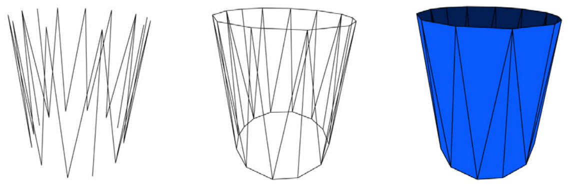
Subsequently, all the rings that are inside other rings are identified using ray tracing technique. As a consequence of Lemma 2.2, these rings are parts of the three-dimensional features. All the corresponding edges in the mesh with respect to the inner rings are added to a set called *forbidden edges*.

3.2. Determining feature triangles

Definition 6. In graph theory, a *cut-edge* is an edge of a graph, whose deletion increases the number of connected components in the graph [10].

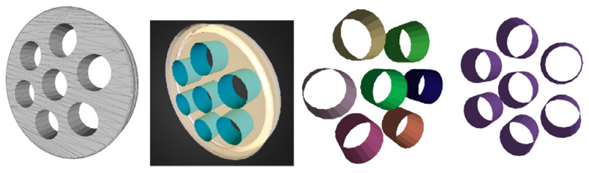
Let $E_{forbidden}$ be the set of all the edges in the mesh that make up the two-dimensional rings in various features. Let T_i be a facet in the mesh such that it does not form a part of any feature. Now, if we traverse the surface of the body from T_i such that we are not allowed to cross over those edges in $E_{forbidden}$, then we will be able to reach all the facets except those which make up the features. Consider the edges and vertices in the mesh as a graph, this problem amounts to identifying cut-edges (Definition 6). Cut-edges can be identified using the depth-first search (DFS) in a graph [4].

At the end of DFS, we have a collection of edges that make up the features. Now, if we have any facet that contains two edges that make up the feature, then the facet will be a part of the feature. Therefore, the third edge will also be a part of the feature. All such edges are added (morphologically, this process is the same as dilation). Fig. 7a shows a set of detected edges of a hole and the result of the dilation process in Fig. 7b. The set of feature triangles is shown in Fig. 7c. For a model in Fig. 8a, the extracted feature triangles are shown in Fig. 8b in cyan. To reiterate, though the triangles have been shown as rendered in Fig. 8b, at this juncture, they are all triangles that contribute to some feature in the model.



(a) Hole edges detected (b) Hole edges after dilation (c) Extracted feature

Fig. 7. Illustration of the algorithm.



(a) (b) (c) (d)

Fig. 8. (a) Input CAD model. (b) Feature triangles (in cyan), rendered. (c) Separation of feature triangle into features. (d) Identified features.

3.3. Separation of feature triangles into features

In this step, we need to separate the features from the set of triangles extracted out of the previous step. Another graph is constructed out of the vertices and edges of the feature triangles. For performing the separation into features, a breadth first search (BFS) [4] procedure is initiated on the graph. BFS will yield the disconnected components of the graph, which are essentially the features themselves. For the model in Fig. 8a–c shows the extracted features (each feature is shown in different colour illustrating the separation of feature triangles shown in cyan in Fig. 8b).

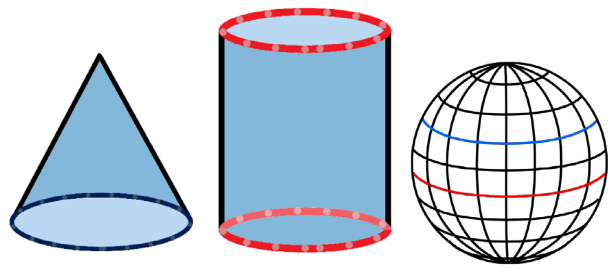

(a) Pocket (b) Through hole (c) Mapping of face normals on Gauss map

Fig. 9. Connected components from outer edges and Gauss map of corresponding shapes.

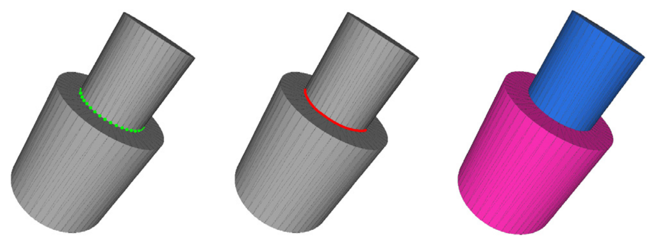

(a) Critical points using negative Gaussian curvature. (b) Edge traversal for determining segment boundary (c) Separation of interacting features.

Fig. 10. Segmentation using negative Gaussian curvature and edge traversal. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article).

🟩 TH - Rectangle	🟩 Po - Rectangle	🟦 BH - Cone
🟪 TH - Cylinder	🟫 Po - Hexagon	🟥 BH - Cylinder
⬛ TH - Unclassified	🟫 Po - Unclassified	🟧 Slot - Unclassified
🟦 IF - Unclassified		

Fig. 11. Colour key for various features detected. TH - Through hole, BH - Blind hole, Po - Pocket, IF - Interacting feature. Terminology of each colour is of 'Feature-Type - GeometryType'.

3.4. Classification of features into various types

The features detected are then classified initially as through holes or pockets. Based on the geometric properties, pockets are classified into rectangle, hexagon, cylinder (blind hole), cone etc. We are also able to find the slots. Similarly, through holes are further classified into rectangular or cylindrical. Step cylinders and other interacting features are then identified. In each of these categories, the features which do not belong to any of the deduced geometric properties are assigned as unclassified. Each such feature discovered is associated with a unique feature identifier. The technique used to discover the geometry of a feature is described below:

- Graph formation from a feature: Let a Graph $G(V, E)$ be formed by V: Set of vertices in the feature, E: Set of edges in the feature
- Outer edge $S(OE)$ detection of the feature: From the graph, a set of outer triangles $S(OT)$ is found i.e. the triangles which have at least one edge which is not shared with any other triangle. Once $S(OT)$ is found, we can extract $S(OE)$ by finding the edges of $S(OT)$ which are not shared with any other triangle. Each such edge will be associated with only a single triangle in the feature.
- Set of loops $S(L)$ which form the outer edges: Loops are discovered by finding the connected components of a set $S(OE)$. This procedure finds the number of connected components in $S(OE)$.

Observation 3.1. A pocket will have a single connected component from the outer edge set $S(OE)$.

Proof. A pocket is open only at a single end and has set of outer triangles only at that end. Hence the outer edges will comprise of edges that form a loop around the open end. □

Fig. 9a shows a single connected component in the set of outer triangles for a conical pocket.

Observation 3.2. A through hole will have two connected components from the outer edge set $S(OE)$.

Proof. If the feature is a through hole we will have two outer loops from each of the open sides and thereby will return two connected components. □

Fig. 9b shows a cylinder having two open ends forming two connected components.

Using observations 3.1 and 3.2, we can identify if the feature is a through hole or a pocket.

Once this is identified, we need to find out the geometry of the feature. This is done by using the Gauss map [12]. Given a surface X lying in R^3, the Gauss map is a continuous map $N: X \rightarrow S^2$ such that $N(p)$ is a unit vector orthogonal to X at p, namely the normal vector to X at p.

We construct a Gauss map for every feature by mapping the face normals of every triangle in the feature onto a Gauss sphere. We further classify the features by taking cues from the Gauss map constructed as well as some of the geometrical properties. A pocket having a cylindrical geometry from the triangles of the edges in the single component is then classified as a *blind hole*. For a slot, it can be observed that not all the edges in the single component of a pocket lie on a plane. Hence a pocket is reclassified as a slot, if some of the edges of the single component are non-planar.

A cylinder will have all the face normals lying on the same plane on the Gauss map as well as form a circle with radius exactly 1 ('red' circle in Fig. 9c). On the other hand, in a cone, the normals while lying on a plane in the unit sphere will always make a circle on the sphere with radius less than 1 ('blue' circle in Fig. 9c). A rectangular hole will have four normals each orthogonal to its neighbour, forming a circle of radius 1 on the Gauss sphere. Using this approach, Fig. 8d shows the identified through holes (cylinders) as features for the model in Fig. 8a.

3.4.1. Interacting feature extraction and separation

Step cylinder, one of the interacting features, will have two connected components, and the normals of the triangles from the edge of the components result in a circle of same radii in the Gauss sphere. However, each component forms circles of different radii in the planes that they lie. This constraint is combined with the fact that not all normals from the feature map to the same circle in the Gauss sphere (i.e., the face connecting the two cylinders in the step cylinder will have normals in a different direction) to arrive at step cylinder. If there are more than two connected components from the outer edge set $S(OE)$, then it is also classified as an *interacting feature*. A segmentation approach is then used to separate the features. Briefly, critical points based on negative Gaussian curvature are identified (green dots in Fig. 10a). Using edge traversal, closed curve from the critical points that forms the boundary of the segments are computed (red in Fig. 10b). The triangles are then grouped to form individual segments (Fig. 10c). Algorithm 1 gives the pseudo-code for extracting feature, given an input of a CAD model represented as a mesh.

Algorithm 1 Feature extraction algorithm.

```
Input: Input mesh data, $S$.
Output: 3D features detected and classified.
 1: Construct graph $\mathfrak{G} = (V, E)$ with Facets as V and mesh edges as E.
 2: $E_{forbidden} \leftarrow \phi$
 3: for 1 to NumberOfPlanes do
 4:     $E_{forbidden} \leftarrow$ inner rings found by cutting planes.
 5: end for
 6: Dilate $E_{forbidden}$.
 7: $T \leftarrow \triangle$ extracted from $E_{forbidden}$ using DFS.
 8: Construct graph $G_d$ from $T$.
 9: Run BFS on $G_d$ to extract the disconnected components.
10: On each component detect $S(OE)$.
11: Find $|S(L)|$ to detect through hole, pocket or interacting feature.
12: Using Gauss Map, classify into a feature.
13: For Interacting features, perform segmentation.
```

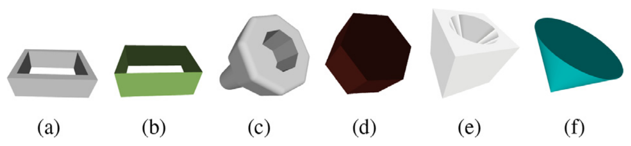


Fig. 12. Test results for rectangular through hole (a) input and (b) output, Hexagonal slot (c) input and (d) output, Conical pocket (e) input and (f) output.

4. Results and discussion

4.1. Benchmark database, models for testing and results

The proposed algorithm (Algorithm 1) is implemented in C++ and CGAL [32], and tested on engineering models from the Engineering Shape Benchmark (ESB) [14], National Design Repository (NDR) [26], and GrabCAD [1]. All the datasets are tuned for Engineering/CAD models. The broad classification of the models in ESB according to [14] is as follows: Solids of revolution, Rectangular-cubic prism or prismatic, and Thin-walled.

To test the algorithm, a dataset consisting of 40 different engineering parts of different shapes and functions are chosen from ESB as well as a few models from NDR and GrabCAD. The identified features were manually verified for all the parts, as test results are typically evaluated qualitatively in the field of feature recognition, i.e. visibly inspecting whether the necessary features (present in the original model) have been identified. The chosen models have varying characteristics viz - prismatic outer having features, solids of revolution having varying sized holes - small, big as well as uniform (almost) sized, some are thin walled etc. In fact, many of the models have features that are not easily visible to the human eye.

Using the colour key in Fig. 11, results of the determined features can be interpreted (the separated interacting features are given arbitrary colours). The term 'unclassified' is used when the geometry was not clearly identified such as bent pipes or when more than one feature are joined together.

Fig. 12 shows a few examples where slots and pockets (a hexagon shaped) have been detected as features. Fig. 12b shows the result for a thin-walled model having through rectangular hole as feature. Fig. 12e is a manually generated input model to demonstrate a conical pocket.

Fig. 13 consists of models that are predominantly having a prismatic outer structure with features interior to it. In Fig. 13b, a slot and a geometrically complex through hole have also been captured. Fig. 13d shows that rectangular shaped pocket features have also been captured. Fig. 13e has one hundred (100) through holes and the algorithm was able to capture all of them (Fig. 13f).

The characteristics of models in Fig. 14 are predominantly having features that are of 'solid of revolution' type from ESB. Moreover, it has both 'big' as well as 'small' holes. It can be noted that the algorithm has performed well for such models.

We are able to classify the features that interacted to form an interacting feature. Interacting features are first extracted as discussed in Section 3.4.1. For example, for the model shown in Fig. 15a, interacting feature is identified as shown in Fig. 15b (in gray). Using segmentation, interacting features are separated (Fig. 15c). Fig. 15 shows the extraction of interacting features (as well as other features) for models picked from various benchmarks demonstrating the algorithm's ability to extract them and then subsequently use segmentation for identifying individual features. The results also demonstrate that the algorithm has the ability to identify large number of features (205 in Fig. 15k).

4.2. Discussion

4.2.1. Number of planes for extracting features

As a random number of planes and directions has been used rather than identifying the planes specifically (since identification has been shown to be difficult [27]), it becomes important to get a feel for the number of planes required to extract features. For all the 40 models used from the ESB (it can be noted that the representative set has been derived from the ESB, a well-classified database for CAD models), we found that using 3000 planes for a model captured all the features in all the models. Using this chosen number of planes, the extraction of features was carried out for all the 42 classes having a total of 801 models and found that the algorithm captured all the features, except in the classification that had curved thin models. The same number of planes were then used for models from NDR and GrabCAD indicating that the number of planes can be fixed across different datasets. Table 1 shows the running time and for the models from ESB, it only took less than 5 s.

4.2.2. Multiple/nested inner rings

Fig. 6(e)–(g) show examples of one outer ring with inner rings. However, in practise, the inner rings can be nested ones, i.e., one inner ring can be inside another inner ring. For example, for the DVI pin model shown in Fig. 15a, the pins with its outer casing will form nested inner rings. As the pins are connected to the casing,

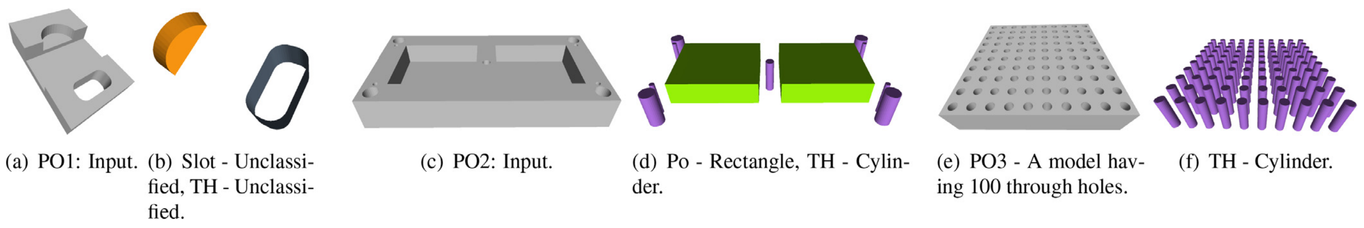
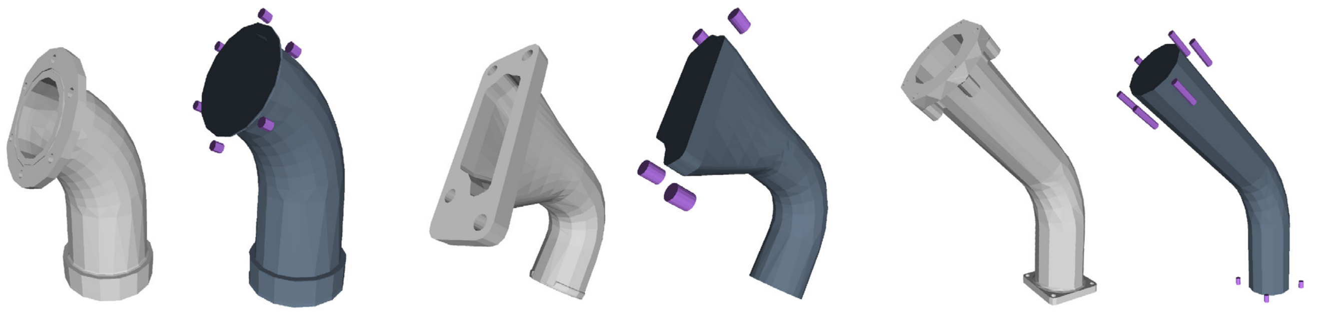

(a) PO1: Input.
(b) Slot - Unclassified, TH - Unclassified.
(c) PO2: Input.
(d) Po - Rectangle, TH - Cylinder.
(e) PO3 - A model having 100 through holes.
(f) TH - Cylinder.

Fig. 13. Test models (prismatic outer) and the identified features.



(a) PM1: Input. (b) TH - Unclassified, TH - Cylinder. (c) PM2: Input. (d) TH - Unclassified, TH - Cylinder. (e) PM3: Input (f) TH - Unclassified, TH - Cylinder.

Fig. 14. Pipe models having big as well as small holes and the identified features.

Table 1
Benchmark for convergence. V - number of vertices in the input mesh model, F - number of facets in the input mesh model, RT(s) - running time in seconds, NF - number of features captured.

Fig.	1 a	13 e	14 a	14 c	14 e	15 a	15 d	15 f	15 h	15 l	15 n	15 p	19 a	19 b
V	2565	4808	844	1899	769	3505	2183	1200	532	1927	828	1793	1324	1200
F	5146	10012	1712	3814	1578	7006	4406	2412	1076	3870	1672	3618	2676	2412
RT(s)	4.83	25.07	1.86	2.44	1.66	22.2	8.67	5.45	1.19	3.31	1.99	3.14	2.25	9.07
NF	17	100	7	5	11	48	18	3	4	10	10	18	7	22

the BFS procedure will give them as a single component. Hence, the multiple inner rings will be detected as an interacting feature (Fig. 15b). The pins are then obtained using segmentation, as shown in Fig. 15c.

4.2.3. Comparison with existing approaches

Comparison was done with the approaches where code was available publicly (SDF [28], WC Seg [16] and HFP [7]). For the models anchor and fork (Fig. 16), results for methods Katz & Tal 03 [18], Katz & Tal 05 [17], Tailor [21] and Plumber [22] are taken from [8] as code was not available. For other models shown in Fig. 17, comparison with SDF [28], WC Seg [16] and HFP [7] has been done.

Decomposition of CAD models has been shown to be a challenging one, and the traditional algorithm for graphical models does not perform well for CAD models [8]. Our approach also does not require the computation of appropriate planes that need to be used for generating rings (as random approach has been followed) as in [27], which works well for graphical models and not for CAD models. For the anchor model in Fig. 16, our method extracted all the features where as the closest one among others is perhaps the HFP (the details contain features which have to be separately obtained). Also, HFP uses visual inspection along with number of clusters as parameter to perform the segmentation. Other approaches mostly under-segment the models, making the extraction of features a difficult task. It should be noted that we use segmentation only to separate the interacting features but not to identifying features from the mesh itself. Moreover, the segmentation uses the boundaries determined from the Gaussian curvature and hence the algorithm does not use threshold value for the segmentation.

For the models 'optical housing', all algorithms under-segment the model, where as for 'DEMO08' and 'CAM101' in Fig. 17, except HFP, other algorithms under-segment them. Under-segmentation makes the capturing of the interacting features a difficult task. For the models 'CADDY02' and 'GEAR38', other approaches tend to over-segment them. Our method has captured all the features in both of them.

As noted in [30], extracting interacting features have posed a challenge for feature recognition algorithms, even from a B-Rep. Algorithms for extracting features of mesh models, both surface-based (e.g. [37]) as well as volume-based (e.g. [2]) have difficulties in handling interacting features as shown in Fig. 17. On the other hand, our approach using plane cutting combined with graph traversal and Gauss map has been able to extract the interacting features.

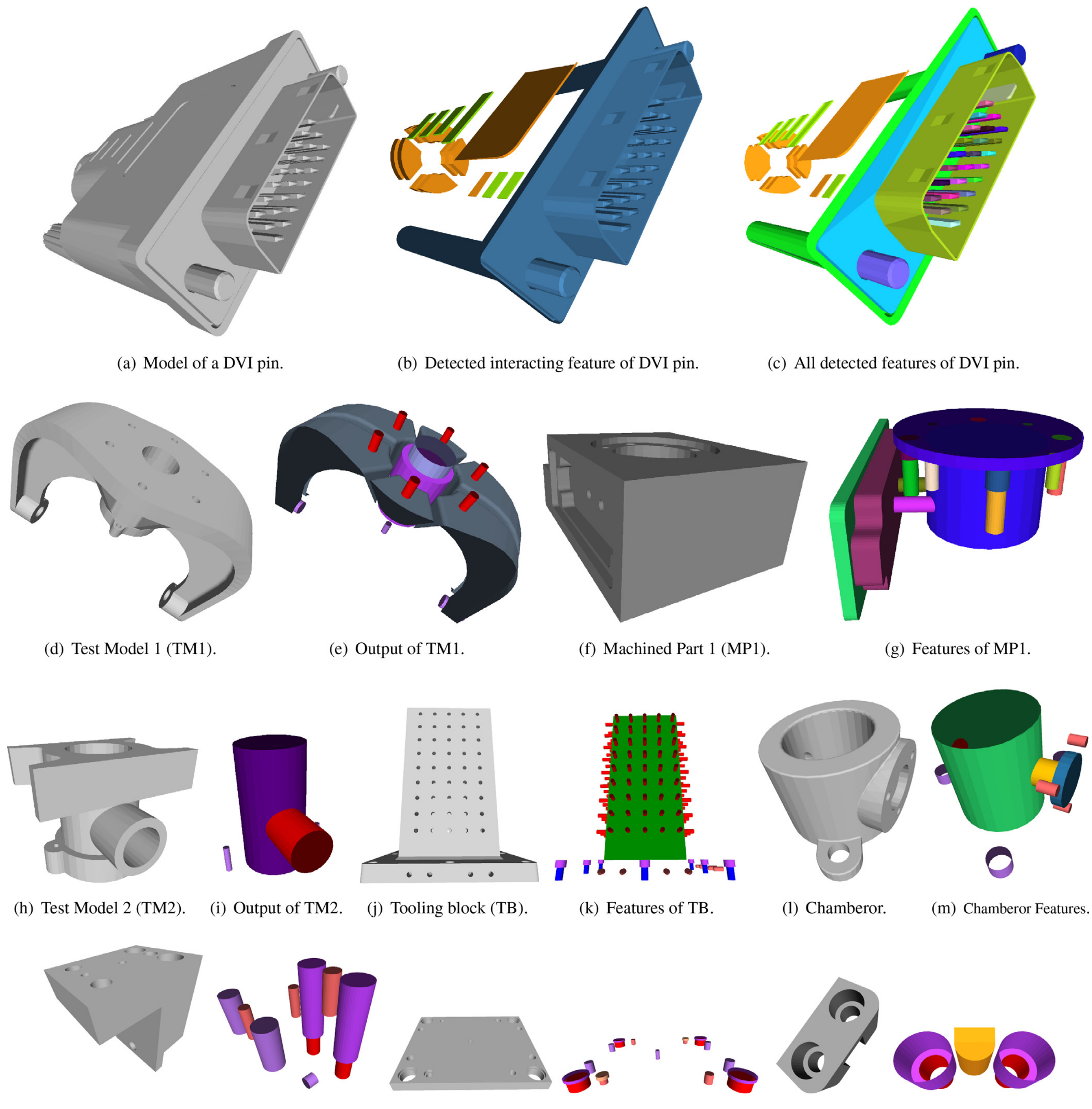
In [37], Gauss map has been used for clustering, whereas we use it only for identifying the feature type. The algorithm in [35] determines a plane to be used by merging the triangles belonging to the same plane. This may pose a restriction when the model has only cylindrical features and they do not seem to handle blind features as well.

Our algorithm does not depend on the value for angle threshold used in many approaches to distinguish cylindrical and conical features. On a similar note, our algorithm does not require the number of clusters to determine the features. The primitive-based approaches are restricted to a plane, a cylinder and a sphere where as our algorithm can capture a variety of features (for example, the differently shaped pockets/slots in Fig. 12). Declarative approach [23] can handle a wide variety of features as long as declarative definitions are available, which can be a complex task.

Our algorithm uses graph traversal instead of graph-based heuristics, and hence enabling the extraction of differently shaped features. It also uses randomised plane (as opposed to using parallel planes with threshold values in [2]) and their generated rings as a consequence of intersection with the mesh. The algorithm is robust as it is not error prone to threshold, that is typically dependent on various factors such as mesh density, and hence different models require different thresholds. Moreover, identifying a threshold value that can work for all models is not a feasible task.

4.2.4. Noisy input

As a mesh model can be imported from different sources, it is possible that there can be noise in the data. To test the algorithm for noisy inputs, ReMESH 2.1 [6] (an editor for manifold triangle meshes) was used to generate them. Noise is varied by distributing the Gaussian noise over the model in the normal direction and it is performed by increasing the percentage of bounding ball radius (BBR) of the mesh model [6]. Fig. 18 shows the results for models with varying noise specified by BBR. For BBR = 10 to



(a) Model of a DVI pin. (b) Detected interacting feature of DVI pin. (c) All detected features of DVI pin.

(d) Test Model 1 (TM1). (e) Output of TM1. (f) Machined Part 1 (MP1). (g) Features of MP1.

(h) Test Model 2 (TM2). (i) Output of TM2. (j) Tooling block (TB). (k) Features of TB. (l) Chamberor. (m) Chamberor Features.

(n) Spacer. (o) Spacer Features. (p) P3: Input. (q) P3 Features. (r) Retainer. (s) Retainer Features.

Fig. 15. Extracting interacting features and separating them using segmentation for models from various benchmarks.

200, Fig. 18(a)–(d)) show that the algorithm has extracted all the features. For the pipe model, the algorithm was able to extract all the features for noise from BBR 10 to 100 (Figs. 18(e)–(g)). For BBR = 200, all features were extracted but contains extra triangles (Fig. 18(h)). The geometry information of the features is not obtained due to the presence of noise that altered the Gauss map information.

4.2.5. Limitations

Fig. 19 shows a few examples of extracted interacting features but not segmented into individual features. Segmentation was not able to separate them as the joints between them have a complex boundary. A feature such as a rib/boss on a model, when it lies externally, though has cross-sections from plane intersection, need not have inner rings. Such a feature would not be captured by our algorithm as it depends on the presence of inner rings. For noisy models, geometry information of the extracted features is not determined.

4.2.6. Future course of work

As mentioned in the limitation (see Section 4.2.5), there is a need to separate the interacting features having joined together by a complicated boundary (see Fig. 19b). This is the key area where the algorithm has to be improved upon. Fig. 20 shows that number of planes (NOP) varies with respect to Gaussian curvature K. Hence, It might be worth exploring using properties such as K to

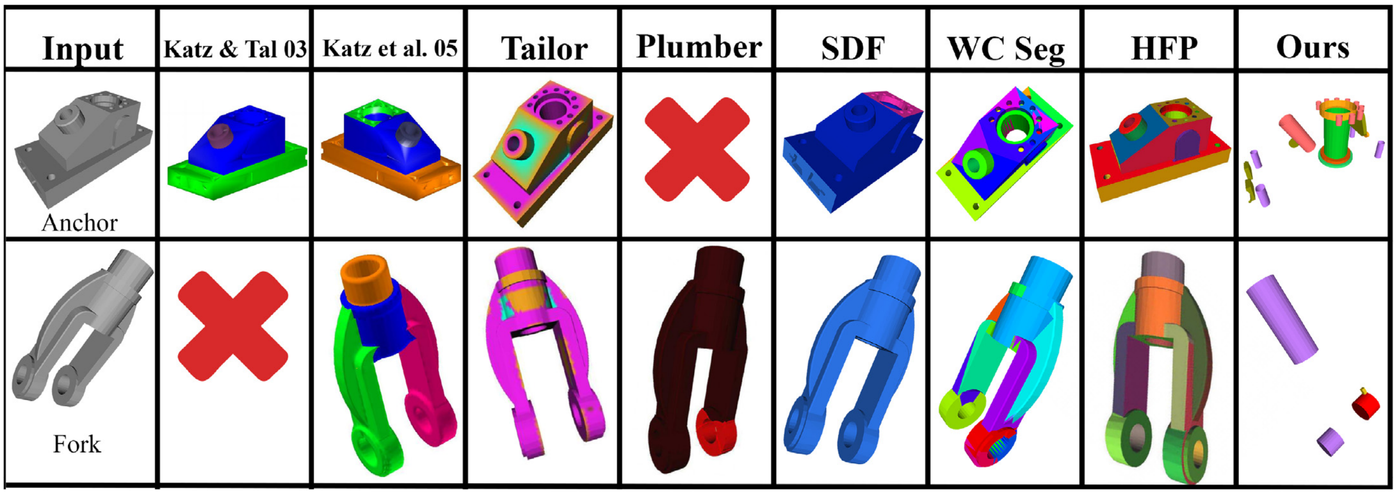


Fig. 16. Comparison with other approaches. Code was available only for SDF [28], WC Seg [16] and HFP [7]. Results for methods Katz & Tal [18], Katz & Tal 05 [17], Tailor [21] and Plumber [22] are taken from [8].

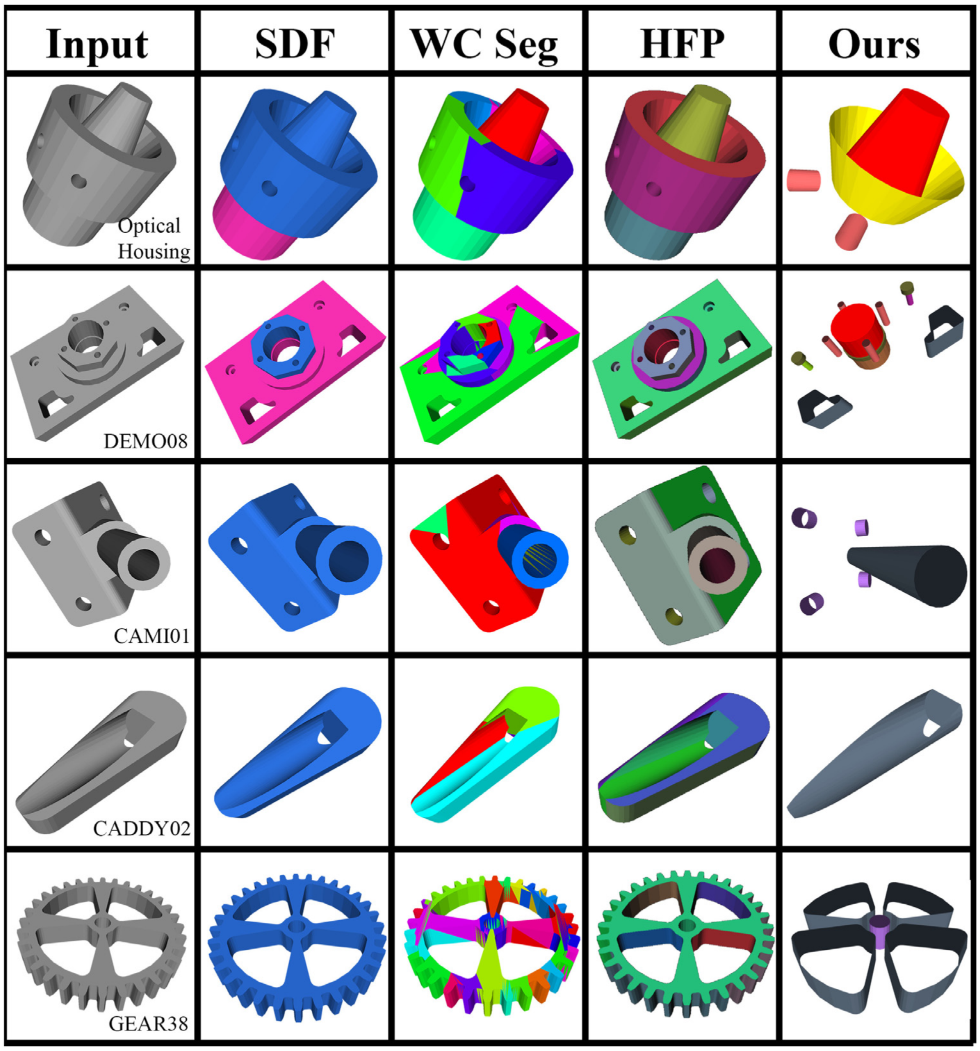


Fig. 17. Code was available only for SDF [28], WC Seg [16] and HFP [7] and comparison with them for various models show that our method was able to extract all the features including the interacting features better than others.

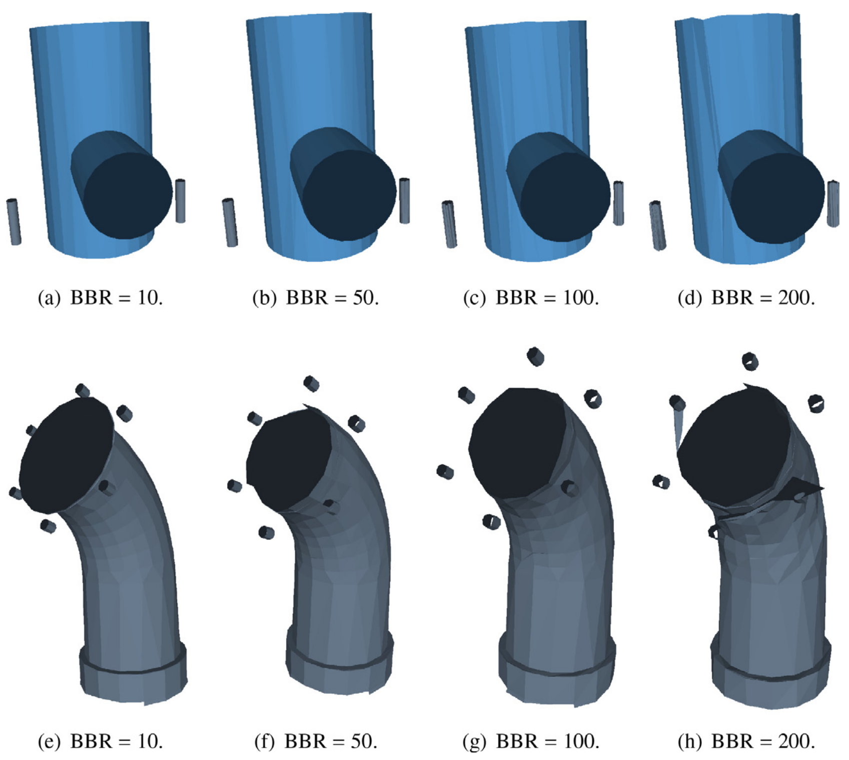


Fig. 18. Test models having varying noise in the input mesh. Noise generated using [6] specified by bounding ball radius (BBR).

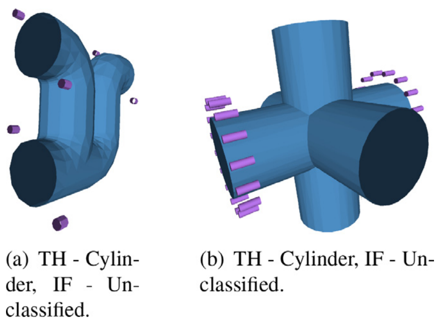

(a) TH - Cylinder, IF - Unclassified.
(b) TH - Cylinder, IF - Unclassified.

Fig. 19. Complex joints in Interacting features.

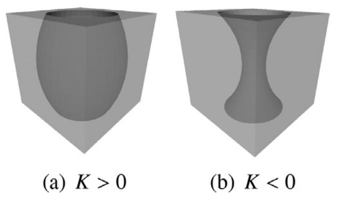


Fig. 20. For $K > 0$, NOP = 500, and for $K < 0$, NOP went up to 1500.

determine NOP as opposed to random generation. Applications of this work in the areas such as matching and retrieval etc. are also in the pipeline.

5. Conclusions

In this paper, a simple algorithm for detecting features in a mesh representation of a CAD model has been proposed and demonstrated. The approach uses random plane cutting as opposed to a harder task of determining appropriately computed planes. This led to the idea of using contours/rings along with graph traversal rather than graph-based heuristics, which also eliminates the need to use threshold values. The algorithm is very simple to implement and experiments over large number of models indeed showed that it not only runs quite fast but also captures all the features. Extraction of interacting features as well as separating them demonstrate the ability of the algorithm to handle them. The failure cases have also been delineated along with possible future directions.

References

[1] GrabCAD. GrabCAD Free CAD Models 2017 https://grabcad.com/library.
[2] Adhikary N, Gurumoorthy B. A slice basedapproach to recognize and extract free-form volumetric features in a cad mesh model. Comput-Aided Des Appl 2016;13(5):587–99. doi:10.1080/16864360.2016.1150703.
[3] Agathos A, Pratikakis I, Perantonis S, Sapidis N, Azariadis P. 3D mesh segmentation methodologies for cad applications. Comput-Aided Des Appl 2007;4(6):827–41. doi:10.1080/16864360.2007.10738515.
[4] Aho AV, Hopcroft JE, Ullman J. Data structures and algorithms. 1st. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.; 1983. ISBN 0201000237.
[5] Armstrong CG. Modelling requirements for finite-element analysis.. Comput-Aided Des Appl 1994;26(7):573–8.
[6] Attene M, Falcidieno B. ReMESH: an interactive environment to edit and repair triangle meshes.. In: SMI. IEEE Computer Society; 2006. p. 41. ISBN 0-7695-2591-1.
[7] Attene M, Falcidieno B, Spagnuolo M. Hierarchical mesh segmentation based on fitting primitives. Vis Comput 2006a;22(3):181–93. doi:10.1007/s00371-006-0375-x.
[8] Attene M, Katz S, Mortara M, Patane G, Spagnuolo M, Tal A. Mesh segmentation–a comparative study. In: Proceedings of the IEEE international conference on shape modeling and applications 2006. SMI '06. Washington, DC, USA: IEEE Computer Society; 2006b. p. 7. ISBN 0-7695-2591-1. doi:10.1109/SMI.2006.24.
[9] Bespalov D, Regli WC, Shokoufandeh A. Local feature extraction and matching partial objects. Comput-Aided Des Appl 2006;38(9):1020–37. Shape Similarity Detection and Search for CAD/CAE Applications.
[10] Bollobas B. Modern graph theory. Graduate texts in mathematics. New York: Springer; 1998. ISBN 9780387984889.
[11] Chen L, Georganas ND. An efficient and robust algorithm for 3D mesh segmentation. Multimed Tools Appl 2006;29(2):109–25.
[12] do Carmo MP. Differential geometry of curves and surfaces. Prentice Hall; 1976. ISBN 978-0-13-212589-5.
[13] Geng C, Suzuki H, Yan D-M, Michikawa T, Sato Y, Hashima M, et al. A Thin-plate CAD mesh model splitting approach based on fitting primitives Theory and practice of computer graphics. Collomosse J, Grimstead I, editors.

The Eurographics Association; 2010. ISBN 978-3-905673-75-3. doi:10.2312/LocalChapterEvents/TPCG/TPCG10/045-050.
[14] Jayanti S, Kalyanaraman Y, Iyer N, Ramani K. Developing an engineering shape benchmark for CAD models.. Comput-Aided Des Appl 2006;38(9):939–53.
[15] Jiao X, Heath MT. Feature detection for surface meshes. In: Proceedings of the 8th international conference on numerical grid generation in computational field simulations. Honolulu HI; 2002. p. 705–14.
[16] Kaick OV, Fish N, Kleiman Y, Asafi S, Cohen-OR D. Shape segmentation by approximate convexity analysis. ACM Trans Graph 2014;34(1) 4:1–4:11. doi:10.1145/2611811.
[17] Katz S, Leifman G, Tal A. Mesh segmentation using feature point and core extraction. Vis Comput 2005;21(8):649–58. doi:10.1007/s00371-005-0344-9.
[18] Katz S, Tal A. Hierarchical mesh decomposition using fuzzy clustering and cuts. ACM Trans Graph 2003;22(3):954–61. doi:10.1145/882262.882369.
[19] Lai JY, Wang MH, Hsu CH, Tsai YC. Recognition and decomposition of bosses on CAD models for hexahedral meshes generation in CAE analysis. In: Proceedings of international conference on applied system innovation (ICASI); 2016. p. 1–4. doi:10.1109/ICASI.2016.7539747.
[20] Li Y, Wu X, Chrysathou Y, Sharf A, Cohen-Or D, Mitra NJ. Globfit: consistently fitting primitives by discovering global relations. ACM Trans Graph 2011;30(4) 52:1–52:12. doi:10.1145/2010324.1964947.
[21] Mortara M, Patané G, Spagnuolo M, Falcidieno B, Rossignac J. Blowing bubbles for multi-scale analysis and decomposition of triangle meshes. Algorithmica 2004a;38(1):227–48.
[22] Mortara M, Patané G, Spagnuolo M, Falcidieno B, Rossignac J. Plumber: a method for a multi-scale decomposition of 3D shapes into tubular primitives and bodies. In: Proceedings of the 9th ACM symposium on solid modeling and applications. Eurographics Association; 2004b. p. 339–44.
[23] Niu Z, Martin RR, Langbein FC, Sabin MA. Rapidly finding CAD features using database optimization. Comput-Aided Des 2015;69:35–50. http://dx.doi.org/10.1016/j.cad.2015.08.001.
[24] Owen SJ, White DR. Mesh-based geometry: a systematic approach to constructing geometry from a finite element mesh. In: Proceedings of the 10th international meshing roundtable. California: Newport Beach; 2001. p. 83–96.
[25] Pratt M, Wilson P. Requirements for support of form features in a solid modelling system. Comput Aided Manuf Int; 1988.
[26] Regli WC, Foster C, Hayes E, Ip CY, Mcwherter D, Peabody M, Shapirsteyn Y, Zaychik V. National Design Repository Project: A Status Report 2001.
[27] Sellamani S, Muthuganapathy R, Kalyanaraman Y, Murugappan S, Goyal M, Ramani K, et al. PCS: prominent cross-sections for mesh models. Comput-Aided Des Appl 2010;7(4):601–20.
[28] Shapira L, Shamir A, Cohen-Or D. Consistent mesh partitioning and skeletonisation using the shape diameter function. Vis Comput 2008;24(4):249. doi:10.1007/s00371-007-0197-5.
[29] Sreevalsan PC, Shah JJ. Unification of form feature definition methods.. In: Proceedings of IntCAD. In: IFIP Transactions, B-4. North-Holland; 1991. p. 83–106.
[30] Sunil V, Agarwal R, Pande S. An approach to recognize interacting features from b-rep CAD models of prismatic machined parts using a hybrid (graph and rule based) technique. Comput Ind 2010;61(7):686–701. doi:10.1016/j.compind.2010.03.011.
[31] Sunil VB, Pande SS. Automatic recognition of features from freeform surface CAD models. Comput Aided Des 2008;40(4):502–17. doi:10.1016/j.cad.2008.01.006.
[32] The CGAL Project. CGAL User and Reference Manual. 4.9. CGAL Editorial Board; 2016. http://doc.cgal.org/4.9/Manual/packages.html.
[33] Umetani N, Schmidt R. Cross-sectional structural analysis for 3D printing optimization. SIGGRAPH asia 2013 technical briefs. SA '13, New York, NY, USA: ACM; 2013. 5:1–5:4. ISBN 978-1-4503-2629-2.
[34] Vidal V, Wolf C, Dupont F. Robust feature line extraction on CAD triangular meshes.. In: Proceedings of the international conference on computer graphics theory and applications; 2011. p. 106–12.
[35] Wang Y, Liu R, Li F, Endo S, Baba T, Uehara Y. An effective hole detection method for 3D models. In: Proceedings of the 20th european signal processing conference (EUSIPCO). IEEE; 2012. p. 1940–4.
[36] Weisstein E.W. Hole. From MathWorld—A Wolfram Web Resource. URL http://mathworld.wolfram.com/Hole.html.
[37] Xiao D, Lin H, Xian C, Gao S. CAD mesh model segmentation by clustering. Comput Graph 2011;35(3):685–91. Shape Modeling International (SMI) Conference 2011, http://dx.doi.org/10.1016/j.cag.2011.03.020.
[38] Zhang C, Chen T. Efficient feature extraction for 2D/3D objects in mesh representation. In: Proceedings 2001 international conference on image processing, 3. IEEE; 2001. p. 935–8.