Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 93 (2020) 544–549

www.elsevier.com/locate/procedia

53rd CIRP Conference on Manufacturing Systems

Extracting shape features from a surface mesh using geometric reasoning

Torbjørn Langedahl Leirmo[a,*], Oleksandr Semeniuta[a], Ivanna Baturynska[a], Kristian Martinsen[a]

[a]*Department of Manufacturing and Civil Engineering, NTNU – Norwegian University of Science and Technology, Teknologivegen 22, 2815 Gjøvik, Norway*

* Corresponding author. Tel.: +47 480 88 390. *E-mail address:* torbjorn.leirmo@ntnu.no

Abstract

Mesh data is extensively used in CAD/CAM applications to approximate three-dimensional (3D) solid models. The STL file format is one of the key file formats for 3D data transfer in modern manufacturing systems. STL files, however, retain no topological information, which would have been beneficial for subsequent file analysis and manipulation. The ability to extract geometric features from mesh data enables automation and facilitates process planning. This paper describes how geometric primitives may be reconstructed from mesh data by simple heuristics. A case study is presented, and a discussion is made on possible applications.

Keywords: Geometric modelling; Computer-Aided Design (CAD); Feature recognition

1. Introduction

Computer-aided technologies such as computer-aided design (CAD), computer-aided manufacturing (CAM) and computer automated process planning (CAPP) have had a drastic influence on manufacturing systems over the last forty years. The CAD models of components are central in modern manufacturing systems and holds important information with regards to intellectual property, manufacturing processes and capabilities, and quality control and assurance. Because of the widespread use of digital models, a large number of file formats for the representation of part geometries and related properties exist both in open format and software specific formatting.

The STL file format was originally developed to accommodate the specific process planning needs of early additive manufacturing technology with the computational capabilities at that time [1]. The format was made accessible to all and was soon utilized for transferring 3D data between platforms across the computer-aided processes [2]. While the original CAD model retains information about the geometric features of the design which would be useful in subsequent processing stages, this information is not explicit in the STL file [3]. Regaining the lost information about the nominal geometry is necessary whenever the CAD model is unavailable, and the design is needed for operations such as optimizing build preparation in additive manufacturing.

While tessellating a computer model to create an STL file is simple enough, the reverse engineering of shape features from STL data is a much more difficult task. Existing solutions make use of complex mathematical analysis and metaheuristics for mesh segmentation and feature classification, both of which are computationally expensive. As an alternative of low computational complexity, we demonstrate how geometric reasoning may be applied to identify shape features from a triangle mesh as described in the STL file format in four steps:

1. Establish connections to neighboring triangles
2. Identify coplanar facets
3. Identify curved segments
4. Merge curved features to create double-curved surfaces

The important difference from the existing body of knowledge is the deduction of regional geometry based on local topology without engaging in computationally extensive

10.1016/j.procir.2020.02.142

mathematical analysis. The method described herein assumes that the STL file is valid and free from noise, preferably originating from CAD software.

First, a literature review is presented before the fundamentals of the STL file format and relevant geometric methods are provided. Next, the method for extracting shape features using geometric reasoning is described and an example implementation demonstrates the application. Finally, a discussion is made on the prospects of the method before a summary and future work.

2. Literature review

The gap from design to manufacturing operations was recognized at an early stage, hence a number of research efforts have been directed at problems such as the CAD/CAM gap. The work of Henderson and Anderson [4] constitutes an early approach to the problem where machining features such as slots, holes, and pockets were automatically extracted from CAD data and a feature graph was created. Marefat and Kashyap [5] introduced a cavity graph approach to the same problem where prismatic depressions were identified through geometric reasoning. Both [4] and [5] took the CAD model as input and therefore cannot be directly applied to STL files.

Krysl and Ortiz [6] describe a set of algorithms for converting a tessellated surface into boundary representation (B-Rep) where the geometry is described by patches defining the boundary between the interior and the exterior with the use of faces, edges, and vertices. While the STL file is exclusively composed of triangular planar faces, the patches of a B-Rep model are represented by splines and may, therefore, take any form [7]. The authors, however, provide no means to identify any shape features of the part, merely to describe its boundary. A B-Rep model was also created by Chappuis, et al. [8] who demonstrated a diffuse integration method for recognizing features in a surface mesh by calculating the local curvature. More recently, Bénière, et al. [9] proposed a method to reconstruct B-Rep models by fitting primitives based on curvature characteristics of the area around vertices.

Moroni, et al. [10] sliced the STL model with three orthogonal planes and identified cylinders by analyzing closed loops in the resulting contour. The same goal was achieved by Qu and Stucker [11] who presented a method based on the edges between facets to construct closed loops which, after an elimination procedure, constitutes the drilled holes of the part.

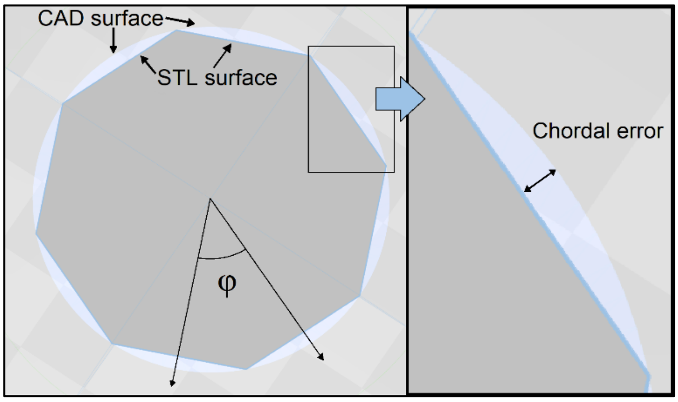
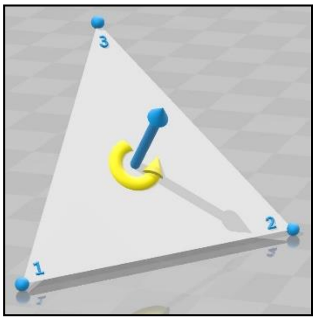


Fig. 1. Illustration of chordal error from STL conversion and the dihedral angle φ.




```
solid <name>
  facet normal <x> <y> <z>
    outer loop
      vertex <x> <y> <z>
      vertex <x> <y> <z>
      vertex <x> <y> <z>
    endloop
  endfacet
.....
(Repeats for each facet)
.....
endsolid <name>
```

Fig. 2. Illustration of a triangular facet with normal vector and vertices (left) and the syntax of an STL file in ASCII format (right).

A somewhat similar method was proposed by Sunil and Pande [12] who identified feature edges by calculating dihedral angles and bounded the identified feature regions. Eight feature types relevant for sheet metal parts could be identified by Gauss and mean curvature calculations. Dihedral angles are widely used in literature for mesh segmentation, and occasionally, they are also utilized for the classification of feature types [13-16].

Hao, et al. [16] demonstrates how the estimation of curvature may be utilized for extracting feature boundaries, but not the feature type. Zhang and Li [17] performed mesh segmentation with regards to local convexity and identified the feature type by analyzing the gaussian image – a technique also applied in [9, 12] where the facet normals are projected on a unit sphere.

3. Theoretic foundations

3.1. Triangle tessellations

Converting a prismatic surface to a triangle tessellation such as the ones present in STL files is unproblematic with regards to accuracy. However, as soon as a curved surface is involved, a deviation known as a chordal error between the original design and the STL model will arise from surface approximations, i.e. the distance from the curved surface of the CAD model to the plane surface of the triangle (Fig. 1) [18]. The magnitude of the chordal errors depends on the resolution of the constructed STL file, i.e. the number of triangles used to represent the part. Typically, the CAD software enables the user to set tolerances for the conversion in terms of maximal chordal error and maximum dihedral angle φ used to represent curved surfaces.

3.2. The STL file format

The STL file contains an unordered list of all the triangles (facets) composing the part, where every facet is represented by a unit normal vector and the coordinates of all three corners. To unambiguously delimit the interior from the exterior, the facet normal points outwards, and the vertices are listed in counterclockwise order as seen from the outside as displayed in Fig. 2 [1, 19].

All facets have three adjacent facets which are referred to as neighbors, however, the file contains no information about adjacency relations. In other words, we know that any given facet must have three neighbors, but there is no straight forward way of finding these facets in the file. The three neighbors of a facet are collectively referred to as the neighborhood of the facet in the remainder of this paper and is illustrated in Fig. 4.

The STL file may originate from different sources that influence the contents of the file: (i) export from CAD software generally produces valid files with minimal noise; and (ii) scanned geometries often introduces noise from environmental factors and invalid files are commonplace. Invalid files typically contain intersecting triangles, inverted normal vectors or holes. Moreover, processed meshes may have unpredictable effects on the STL data due to smoothing or simplifications which further complicates file processing.

3.3. Geometric primitives

There are five distinguishable geometric primitives as illustrated in Fig. 3, plane, cylinder, cone, sphere and torus. In constructive solid geometry, these basic shapes constitute the foundation for all designs through Boolean operations such as union and intersection. These primitives are also central in mechanical parts for creating interfacing and functional surfaces which is why it is desired to extract these primitives from the triangle mesh. The orientation of a primitive surface is defined by a feature vector that is perpendicular to plane surfaces and parallel to the axis of single curved surfaces. Note that some surface types are subject to ambiguous feature vectors due to the degrees of freedom associated with the surface type [20]. E.g. a sphere has no identifiable feature orientation without being supplemented with additional information or rules for determining its feature vector.

3.4. Calculating dihedral angles

In the context of a surface mesh, the dihedral angle φ is defined as the angle between the normal vectors of two adjacent facets as illustrated in Fig. 1. The angle θ between two vectors $\mathbf{v}$ and $\mathbf{p}$ in $\mathbb{R}^3$ may generally be calculated as:

$$\theta = \arccos\left(\frac{\mathbf{p} \cdot \mathbf{v}}{|\mathbf{p}| \cdot |\mathbf{v}|}\right) \tag{1}$$

Because the normal vectors of STL files are unit vectors, the denominator will always be one and therefore insignificant. Based on Eq. 1, the dihedral angle may be calculated simply as:

$$\varphi = \arccos(\mathbf{p} \cdot \mathbf{v}) \tag{2}$$

The dihedral angle φ is positive for all normal vectors and will not give any indication of convexity.

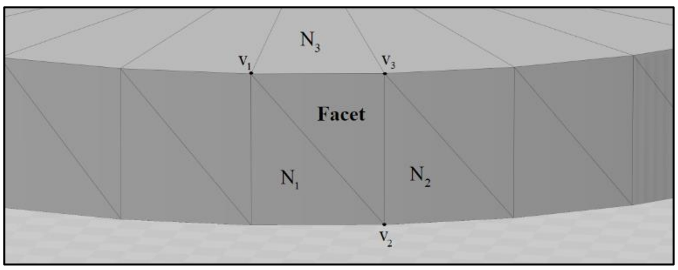


Fig. 4. A facet with vertices v_1, v_2 and v_3 and its neighborhood (N_1, N_2, N_3).

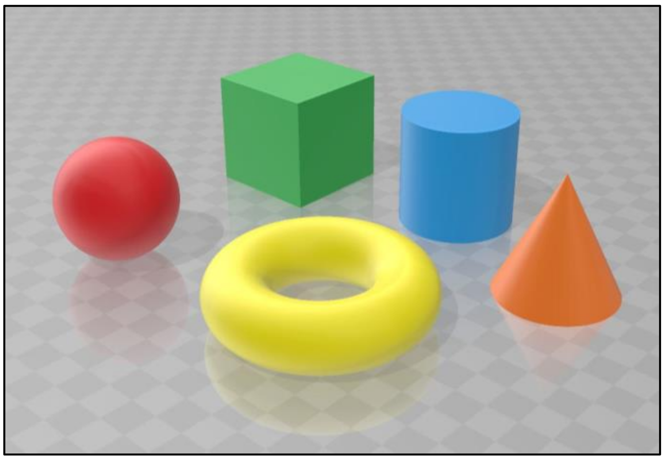


Fig. 3. Illustration of the five geometric primitives: plane (green), cylinder (blue), cone (orange), sphere (red) and torus (yellow).

3.5. Analysis of a triangle neighborhood

The relationship between two neighboring facets may be categorized with respect to the dihedral angle φ based on assumptions regarding the tessellation process. Firstly, it is assumed that any round off errors present in the STL file may be contained within a relatively narrow margin of error denoted as ε . Secondly, it is assumed that a limit γ exists for the maximum dihedral angle φ for curved surfaces regardless of which geometric primitive it represents. Finally, it is assumed that ε is small enough to avoid confusion with curved surfaces. These values enable classification of the angular relationship between the facets as displayed in Table 1. We denote the three angular ranges case *A*, *B* and *C* for the remainder of this paper.

Table 1. Ranges of dihedral angles with edge descriptions.

Case	Dihedral angle φ	Description
A	$\varphi \le \varepsilon$	The facets are coplanar
B	$\varepsilon < \varphi \le \gamma$	The edge represents a curved surface
C	$\varphi > \gamma$	The facets are members of separate features

Because each triangle has three neighbors, all of which may represent any of the cases in Table 1, the number of possible combinations constitutes a problem of k-combinations with repetition [21]. This can be expressed as a multiset coefficient as:

$$\left(\!\!\binom{n}{k}\!\!\right) = \binom{n+k-1}{k} = \binom{3+3-1}{3} = \binom{5}{3} = \frac{5\times 4}{2\times 1} = 10 \tag{3}$$

where n is the number of neighbors and k is the number of possible relations (n multichoose k).

The ten possible combinations of neighborhood relations calculated in Eq. 3 correspond to certain local characteristics and may be used in the identification of shape features as tabulated in Table 2. The order of N_1, N_2, and N_3 is irrelevant for this purpose as the number of neighbors corresponding to a surface type is the only information of interest in this regard.

Table 2. Possible combinations of neighborhood relations.

#	N_1	N_2	N_3	Description
1	*A*	*A*	*A*	All neighbors are part of a single large plane
2	*A*	*A*	*B*	All but one neighbor are coplanar
3	*A*	*A*	*C*	All but one neighbor are coplanar
4	*A*	*B*	*B*	One coplanar neighbor and two curved edges
5	*A*	*B*	*C*	One of each category
6	*A*	*C*	*C*	One coplanar facet and two irrelevant neighbors
7	*B*	*B*	*B*	All three edges are curved
8	*B*	*B*	*C*	Two curved edges and one irrelevant neighbor
9	*B*	*C*	*C*	Only one curved edge and two irrelevant neighbors
10	*C*	*C*	*C*	No relevant neighbors (triangular plane detected)

4. Proposed method applying geometric reasoning

The proposed method involves the following four steps:

1. Establish connections to neighboring triangles
2. Identify coplanar facets
3. Identify curved segments
4. Merge curved features to create double-curved surfaces

Establishing the connection between triangles is pivotal for the efficient handling of the triangles in subsequent operations. The details of the data structure created in the first step are not central in the current work and are outside the scope of this paper. The interested reader is referred to [7, 22] for details on possible data structures. The remaining steps are however explained in detail in the following subsections.

4.1. Identifying planes

The first step of feature recognition is to identify all plane surfaces composed of more than one facet. In practice, each facet must be evaluated with respect to the dihedral angles to its neighbors. From Table 2 this would cover all combinations 1–6. Additionally, case 10 indicates a triangular plane surface that needs only a single facet for its representation and hence requires no further processing.

If two facets are found to be coplanar, a recursive neighborhood search is conducted to identify other facets potentially belonging to the same plane. This region-growing continues until the entire plane is identified. If more than two connected facets are coplanar, the feature type may be confirmed as a plane because no other feature type would yield more than two coplanar connected facets.

4.2. Identifying curved surfaces

Curved surfaces appear in many forms in STL files and require a much more thorough analysis compared to the planar surfaces. The candidate list for curved surfaces includes all single facets remaining after step 1 except those subject to case 10. Additionally, all plane surfaces composed of only two facets must be considered because pairs of coplanar triangles are sometimes present on curved surfaces (see Fig. 5).

Another useful piece of information for guiding the identification of curved surfaces is that they are often

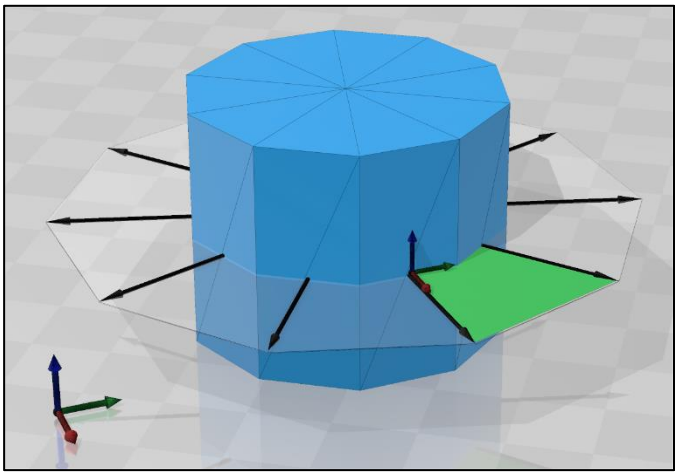


Fig. 5. Illustration of how all facet unit normal vectors lie in the same plane to which the axis of the cylinder is perpendicular.

represented by triangles of roughly the same dimensions. Consequently, if a neighboring triangle is much larger or smaller than the facet of interest, the chance of the neighbor belonging to a different feature is substantial. However, the area of facets should only be used to guide the feature growing, not to determine membership. This is because facets may be of similar size without necessarily belonging to the same feature.

When two neighboring facets are candidates for a curved surface (i.e. a relationship of case *B*, and roughly the same size), the first step is to check if the facets are part of a cylindrical surface. This is accomplished by identifying the direction of the axis of the potential cylinder and then testing the hypothesis on the next neighbors for validation. The direction of the axis of the potential cylinder may be defined as a vector perpendicular to the normal vectors of both facets as illustrated in Fig. 5. The hypothesis is tested by simply checking for perpendicularity between the cylinder axis and the facet normal vectors of the next neighboring facets. Note that because of possible numerical imprecisions in the STL file, all calculations must consider a margin of error.

If none of the next neighbors meet the criteria for cylinders, a similar test is performed to check for cone. This test requires a third facet that must be acquired from the neighborhood with a unique facet normal vector. Because the unit normal vectors are of equal length, the endpoints of the vectors may be used to define a plane that will have a normal vector parallel to the axis of the cone, thus defining the direction of the cone axis as illustrated in Fig. 6. The axis of the cone may be defined as the normal vector of the plane defined by the endpoint of all three facet normal vectors. The apex angle can easily be calculated as twice the angle between the facet normals and the axis. The hypothesis is confirmed if a fourth facet is found that is connected to the existing members with a dihedral angle within the range of case *B* and complies with the apex angle. If the surface is confirmed as a cylinder or cone, a recursive neighborhood search is conducted to collect all member facets.

4.3. Merging from single- to double-curved features

Because of the discretization of continuous surfaces, all surfaces have been decomposed into planes, cylinders, and cones after the previous section. The tessellation process turns

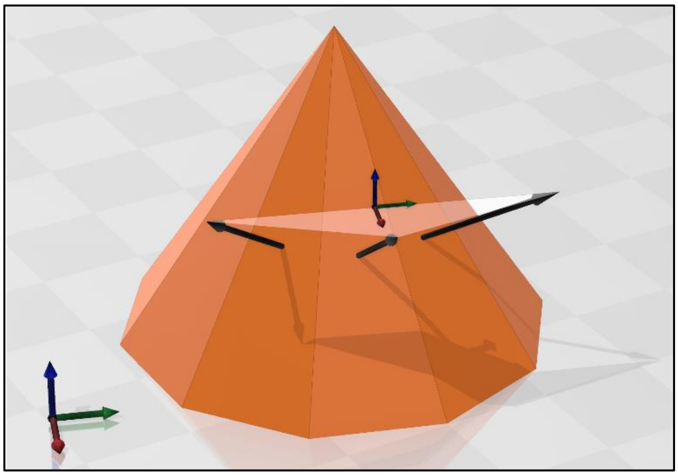


Fig. 6. Illustration of how the endpoints of the facet unit normal vectors may define a plane to which the axis of the cone is perpendicular.

spherical and toric surfaces into segments of connected cones and cylinders (Fig. 7). Hence, the identification of spheres and tori may be accomplished by checking if adjacent cones are coaxial and with apex angles deviating from each other with an angle within the tolerance of curved surfaces, i.e. same as case *B* for dihedral angles (Fig. 7a). Likewise, adjacent cylindrical segments with axes deviating with an angle within the range of case *B* may also be combined to form spheres or tori as illustrated in Fig. 7b and c. To avoid the features merging into unrelated connected surfaces, the direction of axial offset should be constrained. One solution to this problem is to define a plane on which the axes of potential candidate cylinders should lie. For spheres and tori alike, an extra check for the sizes of triangles within candidate features should be conducted to avoid features growing out of bounds.

5. Example implementation

To demonstrate the feasibility of the proposed method, the approach is exemplified on a ball joint which is a simple geometry that embodies several of the geometric primitives. Fig. 8a depicts a plain representation of the triangle mesh, and Fig. 8b illustrates the geometric primitives comprising the ball joint with the color scheme introduced in Fig. 3. The component is designed in SolidWorks 2018 and exported as an STL file with the resolution option "fine" which resulted in 8044 triangles. Due to the rounded edges of the part, no connected triangles form dihedral angles large enough to clearly distinguish separate features. With reference to Table 1, this means that only cases *A* and *B* are present in the mesh which consequently leaves only combinations 1, 2, 4 and 7 from Table 2.

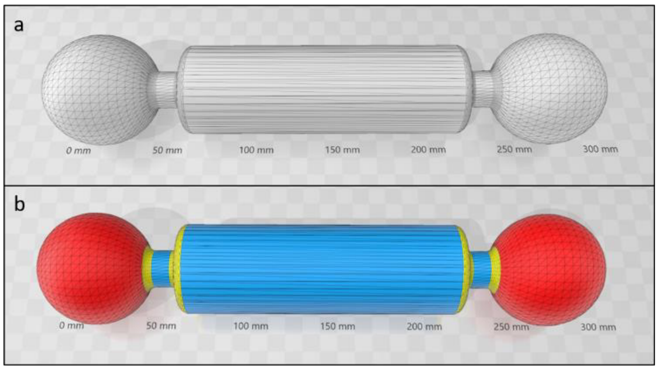


Fig. 8. Ball joint represented as a) raw STL file, b) with color coded surfaces with respect to the geometric primitives.

After the adjacency relations have been established, the first step is to extract coplanar adjacent facets by pairwise comparison of facet normals. Following Table 2, no regions correspond to neighborhood combination 1. Combination 2 is present only in the planar sections visible in Fig. 9. Because three connected coplanar facets may be found by investigating the neighbors of a single facet, the sections are immediately recognized as planar features. Combination 4 is present in most of the mesh. In fact, apart from the planar features already identified, all but the outermost sections of the spheres are classified as combination 4. All instances of combination 4 are included in the subsequent search for curved surfaces.

The second step identifies cylinders and cones by investigating the facet neighborhood. Consider the large cylinder in the middle of the part. The curved surface is prevented from growing into the filleted edges because of the proportional size difference. However, the normal vectors of all the triangles constituting the cylinder lie in the y-z-plane and thus the axis of the cylinder must be parallel with the x-axis as exemplified in Fig. 5). As soon as this knowledge is obtained, the remainder of the cylinder is identified by finding the dihedral angle of the next neighbor recursively and making sure it is perpendicular to the axis of the cylinder. Similarly, segments of the toric and spherical surfaces are identified in this step. Note that depending on the particular implementation and order of facets in the list, the exact results from this step may vary. However, with perfectly defined threshold values and a thorough exploration of cylinders before going forth with identifying cones, one would end up with the cylindrical segments displayed in Fig. 9a. A similar implementation with an emphasis on cones would give the results in Fig. 9b.

Finally, adjacent single curved segments are compared with respect to the orientation of their axes. Again, the relative sizes of the facets constituting the surfaces may be used as a guide for avoiding features growing out of bounds. If two surfaces

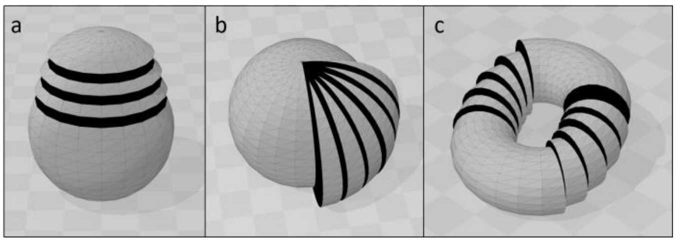


Fig. 7. Segmentations of double curved surfaces. a) cones of a sphere, b) cylinders of a sphere, and c) cylinders of a torus.

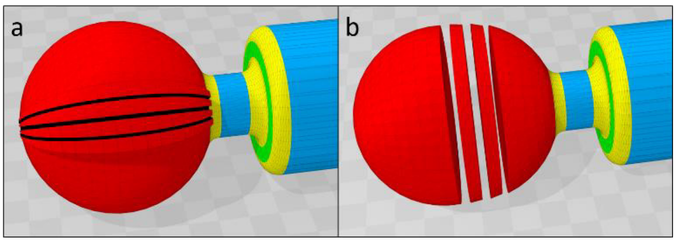


Fig. 9. Ball joint with a) cylindric segments, and b) segments of conic form.

are found to be compatible, they are joined to form the relevant double-curved surface (sphere or torus).

6. Discussion

There are some prerequisites for the proposed method to be feasible in an industrial setting. Firstly, the STL file must be free from holes and intersecting triangles for the adjacency relations to be established correctly. This is important because the method relies on the neighboring triangles being readily available for efficient execution. Next, the method assumes smooth surfaces in the sense that no shape feature contains surface areas deviating from the shape primitive more than the errors induced by the tessellation and numerical imprecision. In practice, this means that the method is not suitable for processing 3D scanned surfaces without preprocessing such as smoothing operations to reduce the inherent noise. Such operations should, however, be used with care, especially in automatic applications, as they can easily distort the geometry.

The presented method is geared towards STL files originating from CAD software without any form of re-meshing and may, therefore, be infeasible in many real-world applications. Certain adaptations must be considered before the method may be successfully applied to organic geometries and alternative file origins. However, the reasoning described in the current work constitutes a logic foundation that is viable for extracting shape features from valid STL files. The proposed method may provide a starting point or otherwise support more advanced feature recognition techniques.

A current trend in manufacturing is the increasing geometric complexity of components, motivated by sustainability in terms of cost savings as well as environmental concerns [23]. Additive manufacturing promises complexity for free, but despite the organic shapes created through topology optimization, the functional surfaces are still primitive. Naturally, the proposed method will perform poorly on freeform surfaces because only small pieces of primitive shapes will be recognized.

7. Summary and further work

The current work described how geometric reasoning may be applied to extract geometric primitives from a triangle mesh. The presented method involves four steps; (i) establishing adjacency relations, (ii) identifying planes, (iii) recognizing single curved segments, and (iv) joining single curved segments to double-curved surfaces. An example implementation was presented to demonstrate the progression of the method. Further validation and demonstration by application on industrial components is planned for future work and a C++ implementation is being developed.

The current work constitutes a computationally inexpensive framework that establishes a foundation for rule-based geometric analysis. The logic presented may be used as a starting point for more advanced feature recognition methods, or as support for computer-aided operations such as process planning, quality assessment, and design optimization. Future work should include the integration of the method with computer-aided technologies. Furthermore, the reverse engineering of solid models from STL data is a possible extension of the current work along with the identification of solid features.

Acknowledgments

This research is funded by the Norwegian Ministry of Research and Education and is associated with SFI Manufacturing funded by the Norwegian Research Council.

References

[1] Hull CW, Spence ST, Albert DJ, Smally DR, et al. Cad/cam stereolithographic data conversion. US patent no. US182830. 1989.
[2] Gibson I, Rosen DW, Stucker B. Additive manufacturing technologies. 2nd ed. New York: Springer; 2015.
[3] Moroni G, Petrò S, Polini W. Geometrical product specification and verification in additive manufacturing. CIRP Ann Manuf Technol. 2017;66(1):157–60.
[4] Henderson MR, Anderson DC. Computer recognition and extraction of form features: A CAD/CAM link. Comput Ind. 1984;5(4):329–39.
[5] Marefat M, Kashyap RL. Geometric reasoning for recognition of three-dimensional object features. IEEE Trans Pattern Anal Mach Intell. 1990;12(10):949–65.
[6] Krysl P, Ortiz M. Extraction of boundary representation from surface triangulations. Int J Numer Methods Eng. 2001;50(7):1737–58.
[7] Bærentzen JA, Gravesen J, Anton F, Aanæs H. Guide to Computational Geometry Processing: Foundations, Algorithms, and Methods. Springer, London; 2012.
[8] Chappuis C, Rassineux A, Breitkopf P, Villon P. Improving surface meshing from discrete data by feature recognition. Eng Comput. 2004;20(3):202–9.
[9] Bénière R, Subsol G, Gesquière G, Le Breton F, Puech W. A comprehensive process of reverse engineering from 3D meshes to CAD models. Comput Aided Des. 2013;45(11):1382–93.
[10] Moroni G, Syam WP, Petrò S. Towards Early Estimation of Part Accuracy in Additive Manufacturing. Procedia CIRP. 2014;21:300–5.
[11] Qu X, Stucker B. Circular hole recognition for STL-based toolpath generation. Rapid Prototyp J. 2005;11(3):132–9.
[12] Sunil VB, Pande SS. Automatic recognition of features from freeform surface CAD models. Comput Aided Des. 2008;40(4):502–17.
[13] Yang S, Shu S. Robust Feature Extraction for the Composite Surface Mesh from STL File. In: The 9th International Conference for Young Computer Scientists: IEEE; 2008:1373–8.
[14] Shamir A. A survey on Mesh Segmentation Techniques. Comput Graph Forum. 2008;27(6):1539–56.
[15] Bespalov D, Shokoufandeh A, Regli WC, Sun W. Local Feature Extraction Using Scale-Space Decomposition. In: Computers and Information in Engineering Conference; Salt Lake City, Utah, USA: ASME; 2004:1–10.
[16] Hao J, Fang L, Williams RE. An efficient curvature-based partitioning of large-scale STL models. Rapid Prototyp J. 2011;17(2):116–27.
[17] Zhang J, Li Y. Region segmentation and shape characterisation for tessellated CAD models. Int J Comput Integr Manuf. 2016;29(8):907–15.
[18] Zha W, Anand S. Geometric approaches to input file modification for part quality improvement in additive manufacturing. J Manuf Process. 2015;20:465–77.
[19] Szilvśi-Nagy M, Mátyási G. Analysis of STL files. Math Comput Model. 2003;38(7):945–60.
[20] Martinsen K. Vectorial tolerancing for all types of surfaces. In: Advances in Design Automation; Albuquerque, USA: ASME; 1993.
[21] Stanley RP. Enumerative Combinatorics. Springer, Boston, MA; 1986.
[22] Guo K-B, Zhang L-C, Wang C-J, Huang S-H. Boolean operations of STL models based on loop detection. Int J Adv Manuf Technol. 2007;33(5):627–33.
[23] Thompson MK, Moroni G, Vaneker T, Fadel G, et al. Design for Additive Manufacturing: Trends, opportunities, considerations, and constraints. CIRP Ann Manuf Technol. 2016;65(2):737–60.