Recognition of Free-form Features for Finite Element Meshing using Deep Learning

Hideyoshi Takashima[1] and Satoshi Kanai[2]

[1]Hokkaido University, hideyoshi_takashima@ais-hokkaido.co.jp
[2]Hokkaido University, kanai@ssi.ist.hokudai.ac.jp

Corresponding author: Hideyoshi Takashima, hideyoshi_takashima@ais-hokkaido.co.jp

Abstract. Finite element (FE) models used in large-scale vehicle simulations are usually composed of high-quality FE meshes that comply with in-house meshing specifications. The meshing patterns, location, and resolution are strictly defined for specific free-form features, such as ribs and bosses, in the specifications. However, finding such free-form features on the given computer-aided design (CAD) models requires a great deal of manual operation. This study proposes a deep-learning (DL) approach to recognize free-form features on CAD models for the automatic generation of FE models to address this issue. This approach allows training a deep neural network on point clouds with reasonable recognition accuracy using a dataset with a large variety of free-form feature shapes represented by the point cloud. The dataset is generated from parametric CAD modeling. It classifies the types of the free-form feature shapes on an input product model and label local feature areas on them. The proposed free-form feature recognition method was experimentally verified using two types of complicated product models. First, models where multiple free-form feature shapes are located independently. Second, where multiple feature shapes are smoothly connected and interacted. The labeling verification showed excellent automatic identification of the feature locations and local feature areas on free-form feature shapes. These results suggest that the proposed DL approach for free-form feature recognition effectively generates FE models.

Keywords: Free-Foam Feature Shape Recognition, Classification, Segmentation, Deep Neural Network, PointNet++, Triangular Mesh, Finite Element Method.
DOI: https://doi.org/10.14733/cadaps.2022.677-693

1 INTRODUCTION

As the digitalization of automotive manufacturing increases and as its development process becomes streamlined, large-scale computer-aided engineering (CAE) has been significant in the ever-growing importance of its role. Finite element (FE) models used in large-scale CAE, such as vehicle crash simulations, pedestrian impact simulations, noise vibration, and harshness simulations, must be composed of high-quality FE meshes that comply with in-house meshing specifications. The meshing

specifications are usually defined according to the design policies and past experiences of vehicle manufacturers. The manufacturers and third parties engaging in outsourcing CAE must respect such specifications when creating FE models.

The meshing patterns, location, and resolution are strictly defined for specific free-form features, such as ribs and bosses, in these specifications. For example, as Figure 1 shows, when FE meshes are generated for a cylindrical boss feature, the elements' node points must be placed concentrically and evenly around a medial axis at a specified quantity. A boss or rib is a lightweight substructure that significantly helps maintain the strength and stiffness of the whole product. The load paths going through the feature should be estimated accurately to guarantee CAE accuracy. If the elements in FE meshes are not arranged neatly in these feature shapes, the simulated load path will differ considerably from reality, causing fatal design errors in the product. To avoid this situation, every manufacturer sets their in-house FE model specifications, prescribing how fine and neatly the node points or elements in FE meshes should be arranged and what kind of qualities the mesh should fulfill for a specific type of form feature. Features prescribed in the meshing specifications are not confined to bosses and ribs. Several other feature types exist that FE meshing patterns should be assigned for, such as holes, embosses, fillets, joggles, hemming, and gradually changing plate thickness. Hence, a "feature" in FE meshing specification is a local region on the CAD model whose meshing quality significantly influences the simulation accuracy and is critical to the analysis reliability from the manufacturer's past experiences.

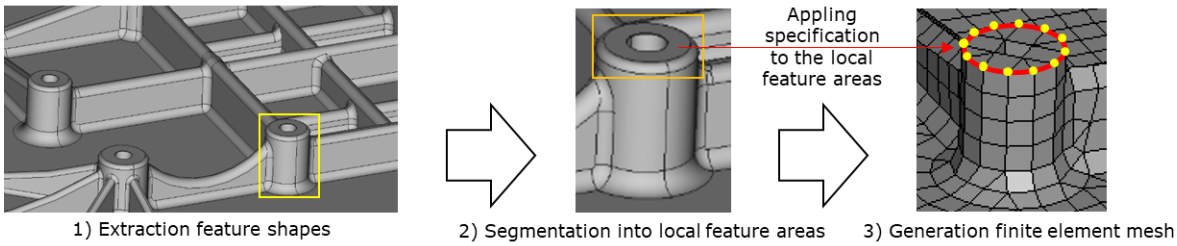

Figure 1: Operations of the feature-compliant finite element meshing.

As shown in Figure 1, feature-compliant FE meshing consists mainly of a series of the following operations:

1. Extracting FE free-form feature shapes such as ribs from a given CAD model.
2. Performing a segmentation of each feature shape into local areas, such as the top, side, and fillet.
3. Generating FE mesh that complies with specific meshing rules defined by recognized feature shapes and areas.

These operations that rely only on the engineer's decisions become incredibly stressful, time-consuming, and error-prone, especially for large and complex CAD model geometry. However, the automatic feature-compliant FE meshing for CAD models is not fully supported in commercial CAE software, requiring a great deal of manual operation, resulting in a high person-hour ratio for the entire CAE process. Such difficulty arises in automatic feature-compliant mesh generation due to the following:

- The CAE model's feature information cannot be exported to the standard file format from the commercial CAD system's native file format. It cannot be imported to a commercial CAE preprocessor. Therefore, feature shape recognition from the CAD model is still needed for FE mesh generation.

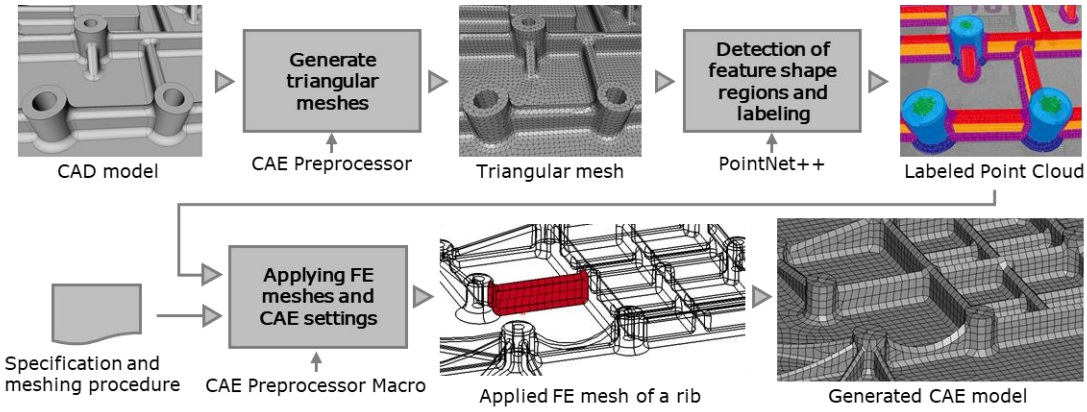

Figure 2: Proposed deep-learning (DL) approach for FE feature recognition system in CAE model generation. The first step is to detect the free-form feature regions from CAD models and label them as a point cloud using PointNet++. The second step is to automatically generate FE mesh, complying with the meshing specifications and procedures defined on each feature type and local feature areas.

- Unlike machining features [5], the features prescribed in the meshing specifications mainly consist of free-form surfaces, having uniquely unshaped geometries, usually bounded by smooth and indistinct boundaries. Therefore, it is difficult to design and implement procedure-oriented or rule-oriented feature-extraction algorithms on a product's CAD model.
- Great diversity exists in the geometries of features for FE meshing. Even ribs and bosses vary substantially in shapes and boundary geometries. Therefore, different recognition algorithms or rules must be designed and implemented for different feature types, especially in procedure- or rule-oriented approaches.
- If the CAD model of a product has product data quality (PDQ) problems, difficulty arises in automatically generating FE meshes, complying with the specifications and simulation accuracy requirements. Similarly, it becomes difficult for the feature shape recognition algorithm to automatically select from a CAD model's boundary faces by relying only on the topological connectivity among faces. Consequently, fixing these poor-quality FE meshes involves substantial time and cost, using CAE preprocessors.

Therefore, automated feature shape extraction and feature area recognition techniques that target feature-compliant FE meshing are strongly required.

Although feature-extraction techniques from CAD models to generate FE meshes have been studied [10][14][16], the following issues still exist.

- The feature-extraction algorithm does not work robustly when the CAD models have product PDQ issues, such as cracked or degenerated geometries.
- The free-form features surrounded by complicated and smooth boundaries commonly found in cast or molded components are challenging to detect when using these techniques.
- The extraction algorithm must be designed in an ad hoc way to work with different feature types and elements with similar shapes. Thus, it is not easy to apply previous feature-extraction techniques when developing a new feature-compliant FE meshing.

Our research group has proposed three-dimensional (3D) shape-descriptor-based FE feature-classification [26] and feature-extraction [27] techniques that use dense point cloud representation to solve these difficulties. However, they offer solutions to only the first operations in the feature-compliant FE meshing process. This study proposes a DL approach to recognize local feature areas

on free-form feature shapes for FE modeling from a product's CAD model, unlike previous studies. PointNet++ [21], a popular multilayer perceptron (MLP) network for 3D point cloud classification and segmentation, was used to recognize free-form feature shapes and local feature areas in our study.

Figure 2 illustrates how the proposed feature shape recognition method works as a part of the FE meshing process. First, the free-form feature regions that comply with FE models' specifications are detected from CAD models. Then, these feature regions are automatically extracted and labeled using our deep neural network that functions directly on the point cloud. Finally, the feature-compliant FE modeling method is applied to the labeled feature regions, generating FE meshes that comply with the FE modeling specifications in which the mesh resolution, node placement constraints, and analysis conditions are defined. However, this study only focuses on how the feature regions are extracted from a target CAD model.

The advantages of the proposed feature shape recognition method are as follows:

- Solid models are transformed into a point cloud with normal as a preprocessing, and it is used for free-form feature recognition based on deep learning (DL) on point clouds. This approach does not rely on any topological connectivity among the faces or edges on the solid models. Consequently, even if the original solid models have PDQ issues, such as cracked or degenerated geometries, point sampling on each triangle can still work, exerting little influence on the extraction process. It improves the stability of the feature recognition process for automatic FE model generation.
- Sometimes, CAE engineers want to perform FE analysis from solid models generated by reverse engineering, where the solid models are generated from the point clouds or triangular meshes measured from the physical model. Here, the CAD (solid) models include more non-uniform rational basis spline surfaces with valuable PDQ problems, such as fine cracks between the surfaces, compared to when the models are generated initially by CAD systems. Conversely, the proposed recognition algorithm only relies on the sampled points on the tessellated meshes. Thus, the processing stability depending on the point sampling distance on the triangles does not degrade whether the model is created using CAD systems or reverse engineering and irrespective of the surface type.
- PointNet++ used in the proposed method is an MLP network for 3D point cloud classification and segmentation. Thus, even if the types of feature shape to be extracted are increased, we only must train by the dataset including new type feature shapes and do not have to modify the recognition algorithm itself.

Given the above-described reasons, the proposed feature shape recognition method can significantly improve the stability and versatility of the feature recognition for FE mesh generation.

The latter part of the paper is organized as follows. In Section 2, related works are reviewed, and issues are clarified. In Section 3, the details of the feature shape recognition algorithms are described. In Section 4, case studies' results are shown. Finally, in Section 5, the conclusion and future work are presented.

2 RELATED WORK

This study is related to forming feature recognition in computer-aided design and computer-aided manufacturing (CAD/CAM), and part-in-whole shape recognition. Their outlines and drawbacks based on free-form feature recognition for FE meshing are discussed in this section.

2.1 Form Feature Recognition

There have been considerable research activities on feature-extraction techniques in CAD and manufacturing fields since the 1990s. Above all, machining feature-extraction from solid models has been intensively studied. An overview of machining feature-extraction techniques has been well-

reviewed [5]. However, they deal with features, such as slots and pockets bounded by sharp and distinct edges. It could be difficult for these techniques to extract features that are surrounded by complex and smooth fillet-like boundaries, commonly found in bosses and ribs on cast or molded parts and are crucial features for FE meshing.

Thus, free-form machining feature recognition from CAD models has been previously studied [2][4][25]. For example, a free-form machining feature recognition method based on a hybrid region segmentation algorithm was proposed by Sunil et al. [25]. In their algorithm, various protrusions and depressions such as bends, beads, and dimples on sheet metal parts can be extracted based on the triangular STL mesh model's curvature properties generated from a B-rep CAD model. However, this approach requires a specific code in an ad hoc manner to fit specified feature classes to be recognized. Therefore, the recognition algorithm is not easily expanded when a new feature class is added.

A rule-based machining feature recognition method from the tessellated B-rep model using the curvature-based region segmentation and region connectivity analysis on the adjacency graph was proposed in a previous study [31]. This method enables the recognition of several STEP-NC features from an unordered set of triangles. However, the technique only extracts features surrounded by sharp edges, such as depressions or protrusions. Also, the extraction ability of features surrounded by complex and smooth fillet-like boundaries is not thoroughly investigated. Moreover, in this method, the extraction algorithm must be implemented as a rule-based algorithm on the adjacency graph. The algorithm must be designed in an ad hoc way to work with different feature types.

Recently, machining feature recognition methods using the heat kernel signature (HKS) have been proposed [6][22], evaluating the heat persistence similarity over the triangulated meshes of products. They can segment any potential machining feature regions from the meshes using a uniform extraction algorithm. Therefore, the methods offer more versatility for feature extraction compared to previous ones. However, they only show the recognition results of the simple-shaped features bounded only with sharp features, such as protrusions or depressions. Also, the recognition possibility of free-form features surrounded by complicated and smooth boundaries remains undiscussed. The most critical problem of this approach is that the feature shapes' types are not directly identified and classified using only HKS. Therefore, the post-processing must be implemented in an ad hoc way to classify the feature types after the extraction. Since the meshing specifications of FE mesh vary by the kind of feature, the lack of feature-classification ability still deteriorates the versatility of the feature extraction in the HKS-based approach.

Unlike machining feature-extraction techniques, few studies exist on the feature extraction aimed at FE meshing. However, recently, there have been increasing studies [1][10][14][16][29][30]. In most studies, graph- and rule-based approaches are used to derive features from a solid model [1][10][16][29]. However, in both approaches, the extraction algorithms or rules are elaborated for specific feature types. Furthermore, a new algorithm or rule must be designed if a new feature type to be extracted must be added.

In contrast, feature-extraction and volume decomposition methods have been developed to automatically generate feature-oriented FE meshes [14][30]. The rule-based approach is used to extract swept features and volumes from a solid model, whose shapes are suitable for automatic hexahedral FE meshing. Furthermore, an automatic mesh generation algorithm using a surface segmentation method based on centroidal Voronoi tessellation has been developed [32][33], which can be applied to shapes with smooth boundaries. However, extraction rules are designed specifically for hexahedral meshing and highly rely on topological loop patterns and local geometric relationships with a solid model. Therefore, the extraction algorithm lacks flexibility and does not work robustly if the CAD model has PDQ issues.

A machine-learning approach based on the traditional neural network has been introduced in machining feature-extraction from CAD models [11][15][17][19][24]. Also, recently, Zhang et al. proposed a machining feature recognition method based on a 3D convolutional neural network [35], which learns the distribution of various manufacturing feature shapes across public 3D model data sets. Also, this approach can recognize particular types of manufacturing features from low-level

geometric data, such as voxels. As in other previous studies, the feature geometries considered in [35] are 2.5-dimensional, consisting only of simple planes and cylinders, and are bounded by sharply concave looped-edges on a B-rep CAD model. However, if the feature is designed on a cast or molded component, the geometry is defined by free-form surfaces. Moreover, the feature is usually bounded by smooth free-form fillet surfaces. Thus, it is doubtful whether previous methods would work well or not for free-form features.

2.2 Part-in-Whole Shape Recognition

Recently, 3D shape retrieval techniques have been researched in computer graphics and 3D mesh processing, as presented in review studies [9][28]. One of these approaches, part-in-whole matching among noisy triangular meshes, has been proposed [8]. This approach can find the local region of a target shape that best matches a reference shape using a probabilistic framework of the feature point and segment similarities. However, it does not necessarily work if the region of the target shape to be found is not congruent to the artifact and has a parametrically deformed relationship with the reference shape. This method can be ineffective in the domain of FE modeling for industrial product design because the method is specifically designed for finding artifacts from the measured noisy point clouds of ancient buildings' walls.

A surface reconstruction method has been developed to create product models of civil structures from laser-scanned point clouds [7]. Here, a part-in-whole matching approach was applied to find a set of representative parts that define major civil structures, such as bridge piers. However, this study only focused on civil structure objects whose shapes are represented by cuboids or cylinders. Thus, it is challenging to apply this method to recognize free-form features on industrial products dealt with in FE meshing.

A similar subpart search technique for FE meshing on solid models has been proposed by Onodera et al. [16], which evaluates the geometric similarities between a reference feature shape and local regions on a target shape. Thus, the partial areas on the target shape, whose geometry is like the archived reference feature shape of proven FE models, are extracted from the novel designed CAD model. However, their subpart search mechanism strongly relies on the topological graphs of a solid model. Therefore, the approach does not work robustly when the CAD model has PDQ issues. This graph-matching approach also fails if two subparts are geometrically similar but have different topological structures.

DL can be a novel and attractive approach for feature recognition in CAD. Recently, some researchers have studied DL on 3D point clouds [3]. As a part of this approach, several neural networks for object classification and part segmentation of 3D shapes represented by point clouds have been actively presented in previous studies [23][34][20][21]. These deep neural network approaches are effective for shape recognition for point clouds [3]. Among them, PointNet++ is one of the most popular and effective methods [21]. However, previous DL approaches, including PointNet++, dealt only with the classification or segmentation of simple-shaped objects, such as tables and chairs. Moreover, the methods' effectiveness has not been fully verified when applied to classify or segment free-form features for FE modeling on industrial products. Furthermore, from the practical perspective, preparing many training examples of free-form features remains a significant issue in DL.

3 RECOGNITION OF FREE-FORM FEATURE SHAPE AND LOCAL FEATURE AREAS

3.1 Basic Concept

Figure 3 outlines the proposed free-form feature recognition pipeline. It automatically labels local feature areas on each feature shape where the feature-compliant FE meshing is needed. In the pipeline, first, the training dataset composed of many point clouds representing feature shapes with feature area labels is generated from CAD models with size variations. These CAD models are automatically generated by parametric modeling of the reference feature shape in advance. Then,

MLP networks for feature classification and part segmentation are trained using the training dataset. Finally, using these trained MLP networks, the feature shapes, such as ribs and bosses, are extracted from an input CAD model, and the local feature areas on each extracted feature shape, such as top and side, are labeled. The targeted features include, but not limited to, free-form ribs and bosses in cast or molded parts.

The proposed recognition method is designed and implemented based on the following concepts.

- The product's geometries are represented by a dense 3D point cloud with normal vectors generated from an input CAD model via triangular mesh for feature extraction. This geometric representation enables stable feature-extraction, even when an input CAD (solid) model contains PDQ-degraded geometries and when complex and smooth fillet-like boundaries surround the feature shapes.
- The machine-learning scheme enables a uniform and portable implementation of the feature-extraction algorithm, regardless of the feature type. Adding a new recognizable category requires only preparing new training examples. It avoids an ad hoc and inefficient implementation of the algorithms for different types.
- Using PointNet++ [21], a DL scheme on point sets can effectively classify and segment 3D point clouds. General and unified machine-learning procedures can be applied to extract free-form feature shapes and to label local feature areas on them.

Figure 3 illustrates the proposed feature recognition method. The recognition pipeline involves seven steps. Steps 1–3 generate the training sets composed of many point clouds expressing feature shapes generated from parametric CAD modeling. Step 4 trains MLP networks for classification and segmentation. Steps 5–7 extract the feature shapes from an input CAD and label the local feature areas on each feature shape. The following subsections describe the details of the pipeline.

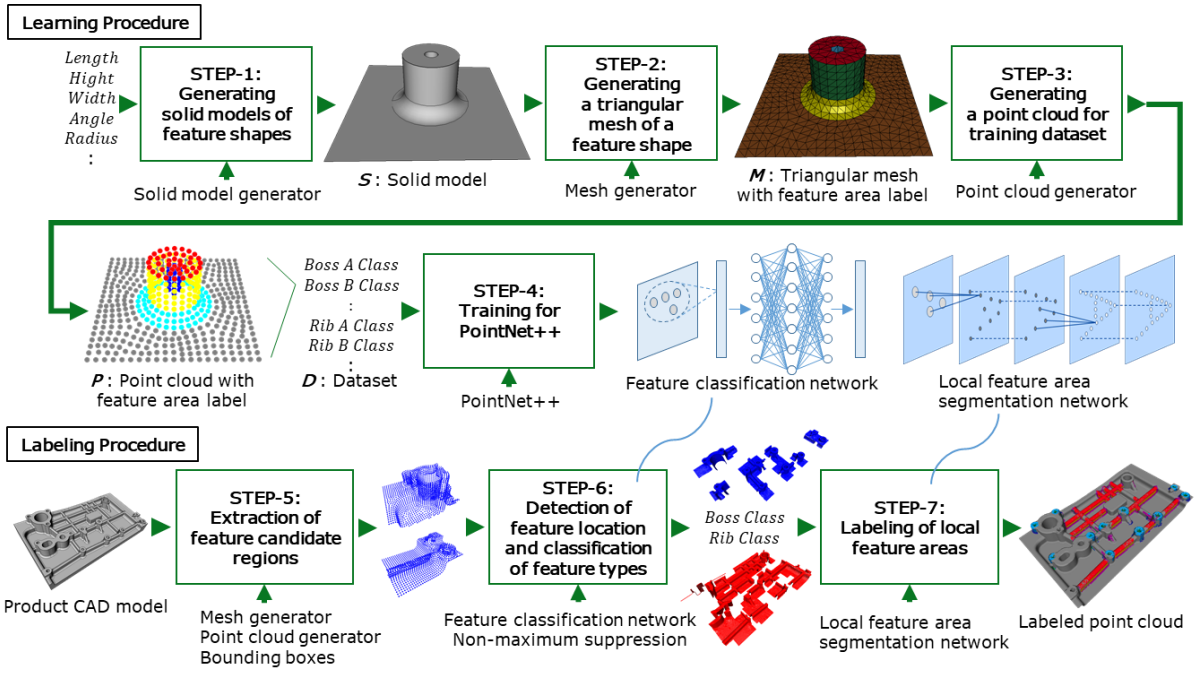

Figure 3: Proposed free-form feature recognition pipeline. Steps 1–3 generate the training dataset composed of many feature shape point clouds, converted from CAD models of feature variants generated by parametric modeling. Step 4 trains MLP networks for feature classification and

segmentation. Steps 5–7 extract the feature shapes from the CAD model and label the local feature areas on each feature shape.

3.2 Step 1: Generating Solid Models of Feature Shapes Using Parametric CAD Modeling

For feature recognition based on DL, an extensive training set of feature shape variants must be prepared to ensure recognition accuracy. However, this study targets the recognition of relatively small feature types, such as ribs and bosses, often appearing in forged or cast automotive parts.

Therefore, the training feature shapes with many size variations using parametric CAD modeling were augmented. A set of solid models can quickly be generated with different size variations in this approach only by changing a set of size parameters defined on a reference shape model. Herein, the reference shape models of rib and boss were prepared (Figures 4 and 5).

Thus, the typical rib and boss shapes on solid models of existing forged and cast automotive parts were observed and analyzed to identify their local feature areas and size parameters, defining feature shapes. Next, many solid models of feature instances with different parameter settings were generated using the parametric deformation function in a CAD system. Each face of these solid model instances is marked with a local feature label (Figure 6). These local feature labels are taken over the point cloud via the triangular mesh in the next step.

3.3 Step 2: Generating a Triangular Mesh of a Feature Shape from the Solid Model

A feature instance's solid model must be finally converted into a 3D point cloud for learning and recognition. Thus, the solid model was first converted into a triangular mesh using a commercial CAE preprocessor. The preprocessor assigns the normal vector and the local feature area label to each vertex of the mesh by referring to the surface information created in step 1 and generates a point cloud for learning and recognition.

3.4 Step 3: Generating a Point Cloud for Training Dataset from the Triangular Mesh

For the DL on point sets, the position and size of the labeled point cloud of the feature variant should be normalized, so the point sets' centroid is translated to the origin, and the maximum length of its axis-aligned bounding box is scaled to one. Finally, the labeled point cloud with normal vectors is generated as a part of the training dataset of PointNet++[21]. Steps 1–3 were repeated for various feature types of bosses and ribs (Figure 5) to prepare the complete training dataset of point clouds of feature variants.

3.5 Step 4: Training for PointNet++

PointNet++ [21] is a deep neural network for classifying and segmenting 3D point clouds. It achieves well-regarded performance using the local feature-learning architecture in which a hierarchical neural network applies PointNet [20] recursively. The training dataset of point clouds of feature variants was used to train PointNet++ and to build two deep neural networks in the training step. The classification network was used to extract feature shapes, and the segmentation network was used to label local feature areas on each extracted feature shape.

3.6 Step 5: Extraction of Feature Candidate Regions

For extracting the feature shape and local feature area, a subset of point clouds belonging to a specific feature class must be first found from an input CAD model's surface. Because a CAD model's surface includes many feature shapes, it is important to extract the local feature candidate regions from the surface and identify their feature types.

For extracting these feature existence regions, an object detection method commonly employed in image processing was applied. First, the input CAD model size was normalized. Next, a dense point cloud from the CAD model was generated using the same operations as steps 1–3. Afterwards, an extensive collection of multi-scale bounding boxes was generated over the dense point cloud,

and a collection of point cloud subsets was cut out from the original dense point cloud with these boxes. For an original point cloud, 8732 multi-scale bounding boxes were generated. Figure 7 illustrates an example of these bounding boxes.

The idea of the multi-scale bounding boxes is like the "default box" of the single-shot multi-box detector (SSD) [12][13]. However, the multi-scale bounding boxes generated over the normalized point cloud were only used to clip point cloud subsets as feature candidate regions. Then, the point cloud subsets, where the number of points reduced and over the threshold were discarded because any feature was unlikely to exist in those areas. These subsets of the feature candidate region were provided for the feature type classification and local feature area labeling of steps 6 and 7.

3.7 Step 6: Detection of Feature Location and Classification of Feature Types

Here, the subsets of point clouds included in each bounding boxes generated in step 5 to the feature-classification network were input and estimated the subsets' feature types, consisting of "rib class" and "boss class." The non-maximum suppression (NMS) [18] was used to eliminate the close overlap among the subsets and adopt the subset point clouds with the highest estimation probabilities. Finally, the selected subsets by the NMS are provided to the segmentation network.

3.8 Step 7: Labeling of Local Feature Areas

Finally, the local feature areas on the subsets of point clouds selected in step 6 are identified using the segmentation network. In the rib features, top, side, fillet, and other areas are identified in each subset of point clouds (Figure 8).

First, each subset of point clouds is input to the segmentation network, and the network assigns a local feature area label to each point in the subset of point clouds. Sometimes, there are intersecting points among the subsets of point clouds because slight overlaps among the subsets of point clouds are allowed in the NMS of step 6. Here, multiple feature area labels might be assigned to a point. The feature area label with the highest estimation is finally adopted to mediate a conflict of these label assignments to the point.

Referring to the point-wise local feature area labels, mesh generation software can identify which meshing specifications should be applied to the feature and automatically generate FE meshes compliant with the specifications.

4 EXPERIMENTAL VERIFICATIONS OF THE FEATURE RECOGNITION

4.1 Recognition Performance of Single Rib and Boss

This verification confirms how well the proposed feature-classification and segmentation networks function for recognizing single free-form features. The trained networks' classification and segmentation performances were evaluated using the datasets, including the rib and boss instances with different size parameters generated from the reference models shown in Figures 4 and 5. A total of 4188 variants with different sizes were generated with various size parameter settings, such as the width and height. Figure 6 shows examples of the local feature areas labeled on these feature models: 3385 and 803 variants were used as the training and testing models, respectively. This dataset was used for verifying both the classification and segmentation networks.

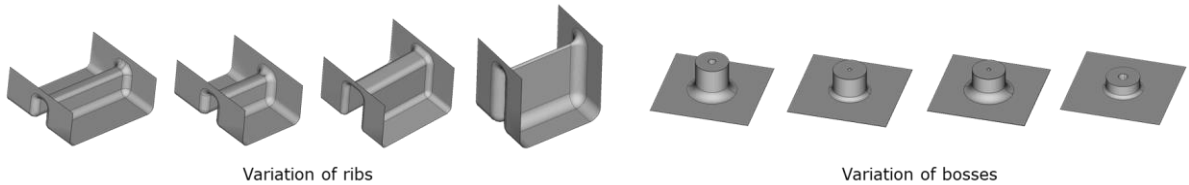

Figure 4: Feature instance examples of ribs and bosses with different parameter settings.

Figure 8 shows a part of the recognition results of the single ribs and bosses. The mean intersection-over-union (mIoU) evaluated per local feature areas of the rib and boss classes were 0.988 and 0.997, respectively, and the total achieved was 0.992. The results showed that both proposed neural networks provided sufficient recognition accuracy for recognizing free-form feature shapes and local feature areas when the feature exists as a separate entity.

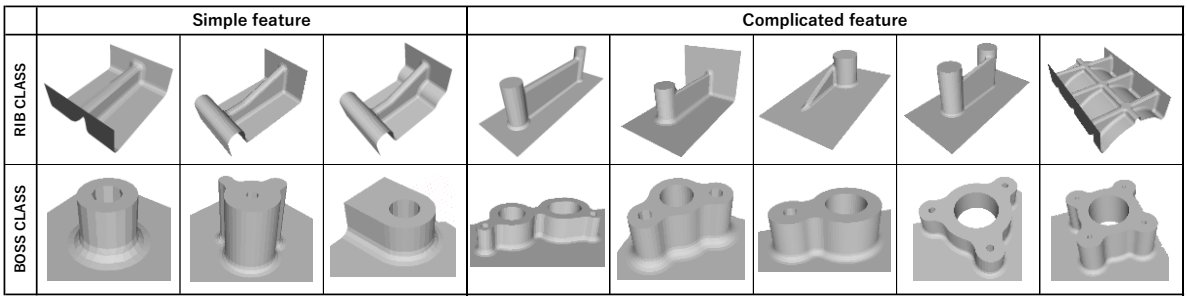

Figure 5: Feature shapes used for the training and verification: The simple feature shapes were used for the training in Section 4.1.3, as "simple" training sets. Both the simple and the complicated feature shapes were used for the training in Sections 4.1, 4.2.1, and 4.2.2, as "complicated" training sets. The recognition performances' differences were compared when using "simple" and "complicated" training sets, in Section 4.2.3.

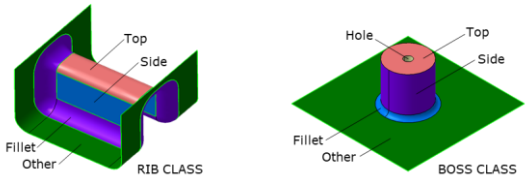

Figure 6: Local feature areas for the rib and boss types.

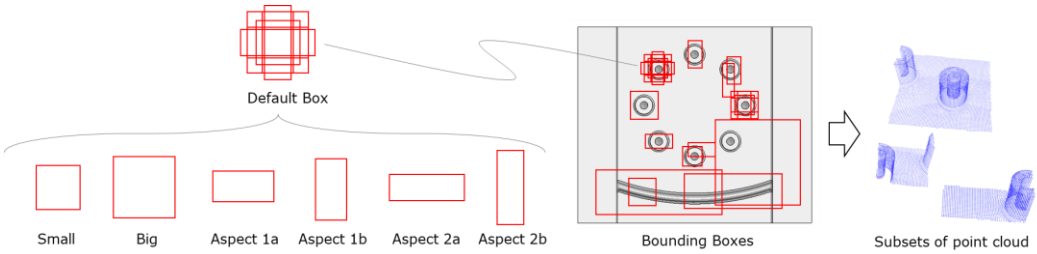

Figure 7: Example of the multi-scale bounding boxes to clip the point cloud subsets.

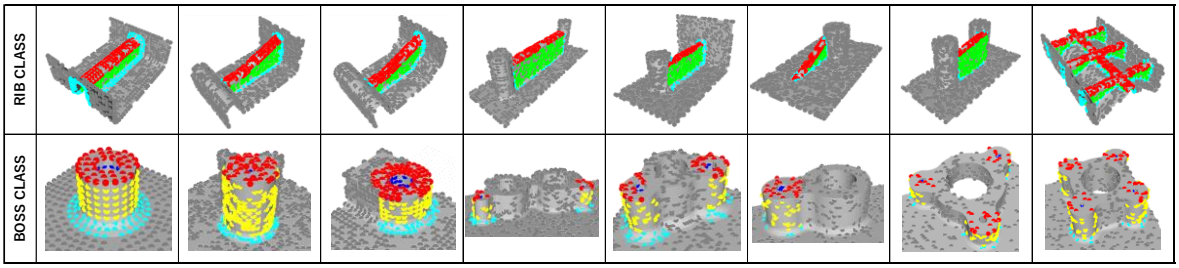

Figure 8: Recognition examples of feature shapes and local feature areas.

4.2 Recognition Performances of Multiple Ribs and Bosses on a CAD Model

This verification investigates how the proposed feature recognition method works when multiple rib and boss features exist in a CAD model. Two complicated models, including ribs and bosses, were used for feature recognition. In the first model, different ribs and bosses were located separately on a CAD model. The second model was more challenging than the first one, where ribs and bosses were smoothly connected and interacted on a model.

To improve the feature recognition accuracy at these two complicated models, the number of feature shapes in the training dataset was increased by adding more feature shape variants than in Section 4.1 using parametric CAD modeling. Also, the point cloud density on the CAD model was increased ~three times. Consequently, the number of models in the new training dataset reached 22380 and was used to retrain the classification and segmentation networks. In the dataset, 17904 and 4476 models were used for training and testing, respectively. The mIoU of feature recognition using this training and test sets reached 0.970. The result sufficiently shows the potential of the point set-based DL approach for free-form feature recognition.

Finally, this retrained neural network was applied to recognize features with multiple ribs and bosses located on two complicated models. These results are summarized in Sections 4.2.1 and 4.2.2. Also, the effect of the point sampling densities and feature shape complexities of the training data set and the DL network structure on the recognition performance was investigated in Section 4.2.3.

4.2.1 Recognition of multiple ribs and bosses located separately

To verify complex features' detection performance, an input model (model A) was created (left of Figure 9), where eight bosses and a rib were arranged independently. For simplicity, multi-scale bounding boxes' heights to generate the feature candidate region were fixed constant because all features in this model were arranged on the same plane. In the extraction of feature candidate regions (step 5), 8732 bounding boxes used for object detection of images by SSD [13] were generated to clip point cloud subsets. Moreover, point cloud subsets where the number of points reduced and exceeded the threshold were removed because of the unlikeliness of any feature existence in such areas.

The feature recognition result is illustrated in Figure 9. The result showed that all feature shapes were detected, and local feature areas were almost labeled with enough accuracy. The mIoU evaluated per local feature areas of this feature result was 0.915. This verification confirmed that the proposed point set-based DL approach works well for recognizing independently arranged free-form features on the model.

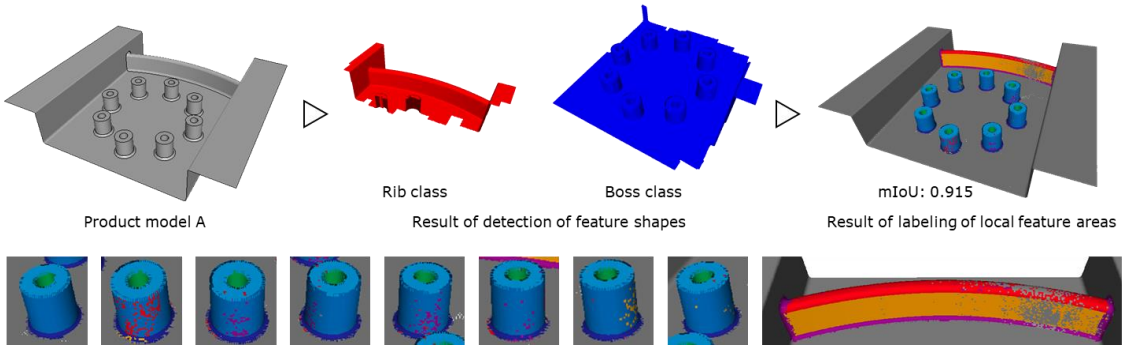

Figure 9: Recognition results of multiple ribs and bosses located separately: Original solid model on which the bosses and rib features are located separately (Left). Point cloud subset's feature detection results clipped from the original model (red: rib feature type, blue: boss feature type) (middle). Result of local feature area labeling (light blue: top of a boss, blue: side of a boss, dark blue: fillet of a boss, green: hole of a boss, red: top of a rib, orange: side of a rib, and purple: fillet of a rib) (right).

4.2.2 Recognition of smoothly connected multiple ribs and bosses

Finally, the complex feature detection performance was verified where the feature shapes are closer to actual forged or cast parts. Two CAD models (models B and C) were created in which the feature shapes are smoothly connected and interacted (Figures 10 and 11). To clip point cloud subsets like the model of Section 4.2.1, 8732 bounding boxes were generated and used.

The feature recognition results from these models are shown in the middle and right of Figures 10 and 11. The mIoUs evaluated per local feature area reached 0.626 and 0.588, respectively, in models B and C, confirming that the proposed feature recognition method also works for the models where feature shapes are smoothly connected.

The recognition accuracies of models B and C were worse than those of model A (Figures 10 and 11). Significantly, the recognition accuracy near the boundaries between features degraded (Figure 12) due to the improper selection of the bounding boxes' size and arrangement. The recognition can be improved by choosing the bounding boxes' size variations more suitable for the features and the input model.

So far, the feature recognition pipeline ends up assigning the labels of local feature areas to each point of the dense point clouds of the product. However, if the assigned labels are projected onto the solid model's face at each point, the feature type of each face on the CAD (solid) model can be determined using the voting scheme. The projection ensures that the feature boundary shape coincides with the face boundary shape on the CAD model. Therefore, if the point cloud's labeling accuracy is reliable to some degree, the feature area boundaries on the dense point clouds do not have to be detected smoothly and precisely.

The verification results show that the proposed free-form feature shape recognition method worked to some degree effectively for automatic FE model generation. However, it can still be improved when feature shapes are smoothly connected. This improvement remains to be solved in our future work.

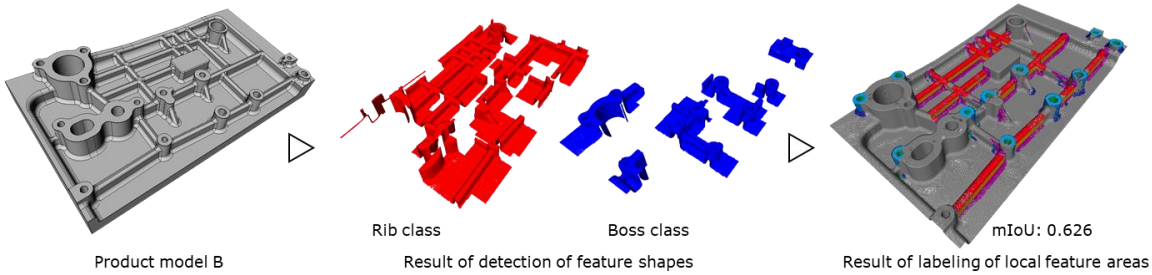

Figure 10: Feature recognition result of the product model B.

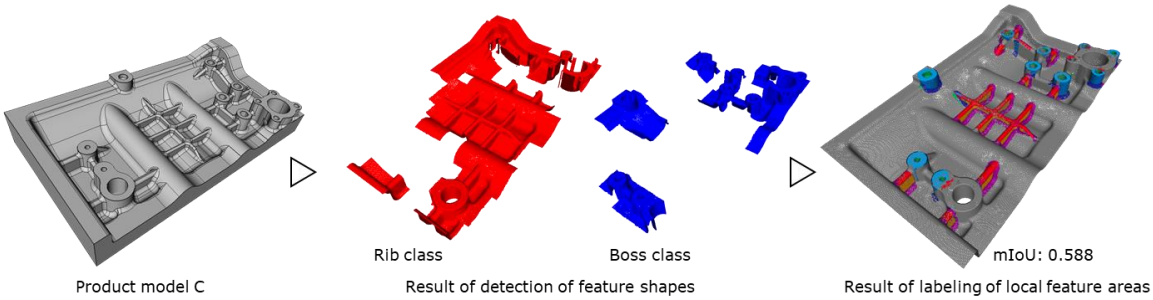

Figure 11: Feature recognition result of the product model C.

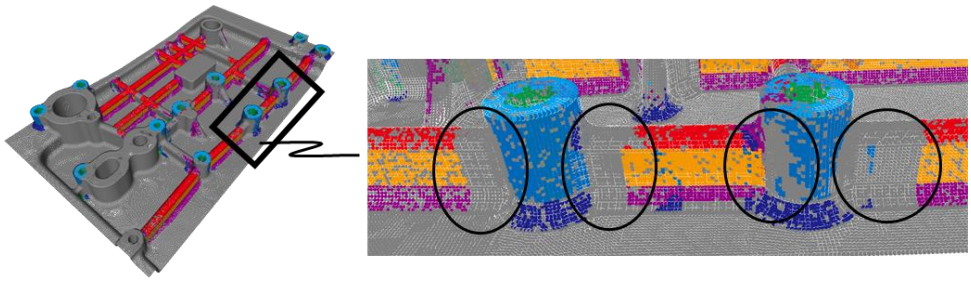

Figure 12: Example of poor recognition accuracy areas around boundaries between feature shapes.

4.2.3 Comparison with different training datasets and different deep neural networks

Generally, object recognition performance using deep neural networks is influenced by selecting training datasets and the network structure. Because sampled point clouds represent the feature shape in our algorithm, the choice of point sampling densities used in the training datasets might also affect recognition performances.

Considering these factors, the feature recognition performances were compared when selecting four training datasets with different point sampling densities and feature shape complexities (Table 1 (1)–(4)). We also compared the performance with that derived from the DL network other than PointNet++. The same CAD models (models A, B, and C), as in Sections 4.2.1 and 4.2.2 were used for the performance comparison.

As for the point sampling densities, the training dataset of *"dense"* and *"sparse"* point clouds was created, where the average point-to-point distance in sampling was set to 1 and 3mm, respectively. Whereas for the feature shape complexities, two training datasets using six simple and ten complicated feature geometries were created (Figure 5). The *"simple"* training set is only composed of the simple features in Figure 5. The *"complicated"* training set is composed of the simple and complicated feature geometries, in Figure 5. The recognition results were compared among the following four cases: dense and complicated, dense and simple, sparse and complicated, and sparse and simple. Under the four cases, the features from models A, B, and C introduced in Section 4.2 were extracted and validated.

Table 1 summarizes the difference in the recognition performances among the four cases. As expected, the result in the first case (*"dense"* and *"complicated"*) provided the best results. Also, the differences in the recognition performances at different point sampling densities were generally not negligible. However, the results of Model C showed that the dense sampling density was not that critical in improving the recognition performance and the change in the feature shape complexities significantly affected the performance than that in the point sampling densities.

Overall, it was found that training the network with denser point sampling and more complicated feature shapes provide better recognition results. Since the recognition examples introduced in Sections 4.2.1 and 4.2.2 were performed under the first case, the results in those sections offered the best among the four cases. Of course, the detailed dependency of the feature recognition performances on the sampling point density of the CAD model should be more investigated but remain as our future work.

However, since our feature-extraction method uses a deep neural network (PointNet++[21]), the choice of the deep neural network structure might influence the recognition accuracy. It is fair to compare the recognition result with that from other deep neural networks. Therefore, we replaced PointNet++ with the original PointNet [20] and compared their recognition results. PointNet++ improved the recognition accuracy of local shapes of PointNet by introducing the hierarchical structure network. By comparing columns (1) and (5) in Table 1, the recognition performance (mIoUs) obtained from PointNet ++ were significantly better than those of PointNet. Of course,

although there are still other state-of-the-art deep neural networks, this comparison indicates that the proposed method using PointNet++ works better than the other one using a different deep neural network from PointNet++.

Setting		(1)	(2)	(3)	(4)	(5)
Training conditions	Point sampling densities	Dense	Dense	Sparse	Sparse	Dense
	Feature shape complexities	Complicated	Simple	Complicated	Simple	Complicated
	Deep neural network	PointNet++	PointNet++	PointNet++	PointNet++	PointNet
mIoU results	Model A	0.915	0.831	0.887	0.660	0.564
	Model B	0.626	0.400	0.560	0.386	0.477
	Model C	0.588	0.527	0.585	0.472	0.507

Table 1: Comparison of recognition performance with different training datasets and different deep neural networks: For the point cloud densities, the average point-to-point distance was set to 1 mm for the "Dense" point cloud and 3 mm for the "Sparse" point cloud. For the feature shape complexities, the "simple" was trained the neural networks using only the simple features in Figure 5, and the "complicated" was trained using both the simple and complicated feature geometries in Figure 5.

5 CONCLUSIONS

This study proposed a DL-based free-form feature recognition method to develop an intelligent FE model generation system. The proposed feature recognition method uses the dense point set representation of an original CAD model and feature shapes and DL on point sets for feature classification and labeling. The training dataset was effectively generated using parametric CAD modeling. Two neural networks of feature type classification and local feature area labeling based on PointNet++ were trained and used for feature recognition. The bounding boxes and NMS techniques were combined with the networks for feature detection and labeling. The verification results of several examples showed that the proposed free-form feature shape recognition method functioned effectively for automatic FE model generation. However, it can still be improved when feature shapes are smoothly connected and when the accuracy degrades close to the boundary between features.

In future work, we will attempt to improve recognition accuracy around the boundary between different features. Also, the dependency of the feature recognition performances on the sampling point density of the CAD model will be more investigated, and we will determine the guidelines for the best sampling density to achieve the best recognition accuracy. We will also compare our method with any rule-oriented feature-extraction algorithm.

Moreover, we will develop an integrated pipeline for the automated feature-compliant FE model generation. First, we will inherit the feature type and local feature area labels of the faces on an original CAD model from the labeled point clouds. We will also create the feature-compliant FE mesh from the labeled solid model by selecting the proper FE meshing operations according to the company-specific meshing specifications. Then, we will develop a method to automatically set up constraints and contact conditions based on the labeled CAD model. These developments remain as our future works.

Hideyoshi Takashima, https://orcid.org/0000-0002-7158-6059
Satoshi Kanai, https://orcid.org/0000-0003-3570-1782

REFERENCES

[1] Boussuge, F.; Leon, J.-C.; Hahmann, S.; Fine, L.: Idealized models for FEA derived from generative modeling processes based on extrusion primitives, Engineering with Computers, 31(3), 2015, 513-527. http://doi.org/10.1007/s00366-014-0382-x
[2] Cai, N.; Bendjebla, S.; Lavernhe, S.; Mehdi-Souzani, C.; Anwer, N.: Freeform machining feature recognition with manufacturability analysis, Procedia CIRP, 72, 2018, 1475-1480. https://doi.org/10.1016/j.procir.2018.03.261
[3] Guo, Y.; Wang, H.; Hu, Q.; Liu, H.; Liu, L.; Bennamoun, M.: Deep Learning for 3D Point Clouds: A Survey, IEEE Transactions on Pattern Analysis and Machine Intelligence, 2020. https://doi.org/10.1109/TPAMI.2020.3005434
[4] Gupta, R.K.; Gurumoorthy, B.: Automatic extraction of free-form surface features (FFSFs), Computer-Aided Design, 44(2), 2012, 99-112. https://doi.org/10.1016/j.cad.2011.09.012
[5] Han, J.; Pratt, M.; Regli, W.C.: Manufacturing feature recognition from solid models: a status report, IEEE Transactions on Robotics and Automation, 16(6), 2000, 782-796. https://doi.org/10.1109/70.897789
[6] Harik, R.; Shi, Y.; Beak, S.: Shape Terra: mechanical feature recognition based on a persistent heat signature, Computer-Aided Design and Applications, 14(2), 2017, 206-218. http://dx.doi.org/10.1080/16864360.2016.1223433
[7] Hidaka, N.; Michikawa, T.; Yabuki, N.; Fukuda, T.; Motamedi, A.: Creating product models from point cloud of civil structures based on geometric similarity, The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, 40(4), 2015, 137-141. https://doi.org/10.5194/isprsarchives-XL-4-W5-137-2015
[8] Itskovich, A.; Tal, A.: Surface partial matching and application to archaeology, Computers & Graphics, 35(2), 2010, 334-341. https://doi.org/10.1016/j.cag.2010.11.010
[9] Iyer, N.; Jayanti, S.; Lou, K.; Kalyanaraman, Y.; Ramani, K.: Three-dimensional shape searching: state-of-the-art review and future trends, Computer-Aided Design, 37(5), 2005, 509-530. https://doi.org/10.1016/j.cad.2004.07.002
[10] Lai, J.-Y.; Wang, M.-H.; Song, P.-P.; Hsu, C.-H.; Tsai, Y.-C.: Recognition and Decomposition of Rib Features in Thin-shell Plastic Parts for Finite Element Analysis, Computer-Aided Design and Applications, 15(2), 2018, 264-279. https://doi.org/10.1080/16864360.2017.1375678
[11] Lankalapalli, K.; Chatterjee, S.; Chang, T.C.: Feature recognition using ART2: A self-organizing neural network, Journal of Intelligent Manufacturing, 8(3), 1997, 203-214. https://doi.org/10.1023/A:1018521207901
[12] Limaye, A.; Mathew, M.; Nagori, S.; Swami, P.K.; Maji, D.; Desappan, K.: SS3D: Single Shot 3D Object Detector. arXiv:2004.14674v2
[13] Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C.: SSD: Single Shot MultiBox Detector, European Conference on Computer Vision, 2016, 21-37. https://doi.org/10.1007/978-3-319-46448-0_2
[14] Lu, Y.; Gadh, R.; Tautges, T.J.: Feature based hex meshing methodology: feature recognition and volume decomposition, Computer-Aided Design, 33(3), 2001, 221-232. https://doi.org/10.1016/S0010-4485(00)00122-6
[15] Nezis, K.; Vosniakos, G.: Recognizing 2 1/2D shape features using a neural network and heuristics, Computer-Aided Design, 29(7), 1997, 523-539. https://doi.org/10.1016/S0010-4485(97)00003-1

[16] Onodera, M.; Hariya, M.; Kongo, C.; Shintani, M.; Ka, K.; Watanuki, K.: Development of high precise similar sub-part search technique for automatic mesh generation reusing proven models, Transactions of The Japan Society of Mechanical Engineers, 85(880), 2019, 19-138. https://doi.org/10.1299/transjsme.19-00138
[17] Onwubolu, G.C.: Manufacturing features recognition using backpropagation neural networks, Journal of Intelligent Manufacturing, 10(3), 1999, 289-299. https://doi.org/10.1023/A:1008904109029
[18] Pham, T.Q.: Non-maximum Suppression Using fewer than Two Comparisons per Pixel, Advanced Concepts for Intelligent Vision Systems, 2010. https://doi.org/10.1007/978-3-642-17688-3_41
[19] Prabhakar, S.; Henderson, M.R.: Automatic form-feature recognition using neural-network-based techniques on boundary representations of solid models, Computer-Aided Design, 24(7), 1992, 381-393. https://doi.org/10.1016/0010-4485(92)90064-H
[20] Qi, C.R.; Su, H.; Mo, K.; Guibas, L.J.: PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation, Proceedings of the IEEE Conference on CVPR, 2017, 652-660. https://doi.ieeecomputersociety.org/10.1109/CVPR.2017.16
[21] Qi, C.R.; Yi, L.; Su, H.; Guibas, L.J.: PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space, Proceedings of NIPS, 2017, 5105-5114. https://arxiv.org/abs/1706.02413
[22] Shi, Y.; Zhang, Y.; Baek, S.; Backer D.W.; Harik, R.: Manufacturability analysis for additive manufacturing using a novel feature recognition technique, Computer-Aided Design and Applications, 15(6), 2018, 941-952. https://doi.org/10.1080/16864360.2018.1462574
[23] Su, H.; Jampani, V.; Sun, D.; Maji, S.; Kalogerakis, E.; Yang, M. H.; Kautz, J.: SPLATNet: Sparse Lattice Networks for Point Cloud Processing, Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2018, 2530-2539. https://doi.org/10.1109/CVPR.2018.00268
[24] Sunil, V.B.; Pande, S.S.: Automatic recognition of machining features using artificial neural networks, International Journal of Advanced Manufacturing Technology, 41, 2009(9-10), 932-947. https://doi.org/10.1007/s00170-008-1536-z
[25] Sunil, V.B.; Pande, S.S.: Automatic recognition of features from freeform surface CAD models, Computer-Aided Design, 40(4), 2008, 502-517. https://doi.org/10.1016/j.cad.2008.01.006
[26] Takaishi, I.; Kanai, S.; Date, H.; Takashima, H.: Free-Form feature classification for finite element meshing based on shape descriptors and machine learning, Computer-Aided Design and Applications, 17(5), 2020, 1049-1066. https://doi.org/10.14733/cadaps.2020.1049-1066
[27] Takashima, H.; Kanai, S.: Shape Descriptor-Based Similar Feature Extraction for Finite Element Meshing, Computer-Aided Design and Applications, 18(5), 2021, 1080-1095. https://doi.org/10.14733/cadaps.2021.1080-1095
[28] Tangelder, J.W.H.; Veltkamp, R.C.: A survey of content-based 3D shape retrieval methods, Multimedia Tools Applications, 39, 2008, 441-471. https://doi.org/10.1007/s11042-007-0181-0
[29] Wang, M.-S.; Lai, J.-Y.; Hsu, C.-H.; Tsai, Y.-C.; Huang, C.-Y.: Boss Recognition Algorithm and Application to Finite Element Analysis, Computer-Aided Design and Applications, 14(4), 2017, 450-463. http://doi.org/10.1080/16864360.2016.1257187
[30] Wu, H.; Gao, S.: Automatic swept volume decomposition based on sweep directions extraction for hexahedral meshing, Procedia Engineering, 82, 2014, 136-148. https://doi.org/10.1016/j.proeng.2014.10.379
[31] Xu, S.; Anwer, N.; Mehdi-Souzani, C.: Machining Feature Recognition from In-Process Model of NC Simulation, Computer-Aided Design and Applications, 12(4), 2015, 383–392. http://dx.doi.org/10.1080/16864360.2014.997634
[32] Yu, Y.; Liu J.G.; Zhang Y.J.: HexDom: Polycube-Based Hexahedral-Dominant Mesh Generation. arXiv:2103.04183v2

[33] Yu, Y.; Wei, X.; Li, A.; Liu, J.G.; He, J.; Zhang, Y.J.: HexGen and Hex2Spline: Polycube-based Hexahedral Mesh Generation and Spline Modeling for Isogeometric Analysis Applications in LS-DYNA. arXiv:2011.14213v1
[34] Yuan, W.; Khot, T.; Held, D.; Mertz, C.; Hebert, M.: PCN: Point Completion Network,2018 International Conference on 3D Vision, Verona, 2018, 728-737. https://doi.org/10.1109/3DV.2018.00088.
[35] Zhang, Z.; Jaiswal, P.; Rai, R.: FeatureNet: Machining feature recognition based on 3D Convolution Neural Network, Computer-Aided Design, 101, 2018, 12-22. https://doi.org/10.1016/j.cad.2018.03.006