Computer-Aided Design 163 (2023) 103577

Contents lists available at ScienceDirect

Computer-Aided Design

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

Simplification of 3D CAD Model in Voxel Form for Mechanical Parts Using Generative Adversarial Networks

Hyunoh Lee [a], Jinwon Lee [b], Soonjo Kwon [c], Karthik Ramani [d], Hyung-gun Chi [d], Duhwan Mun [a,*]

[a] School of Mechanical Engineering, Korea University, 145 Anam-ro, Seongbuk-gu, Seoul, 02841, Republic of Korea
[b] Department of Industrial and Management Engineering, Gangneung-Wonju National University, Gangneung, Gangwon, 26403, Republic of Korea
[c] Department of Mechanical System Engineering, Kumoh National Institute of Technology, Gumi, Gyeongbuk, 39177, Republic of Korea
[d] School of Mechanical Engineering, Purdue University, West Lafayette, IN 47907, USA

ARTICLE INFO

Article history:
Received 1 March 2023
Received in revised form 27 April 2023
Accepted 5 June 2023

Dataset link: https://engineering.purdue.edu/cdesign/wp/downloads/

Keywords:
3D CAD model
Convolutional neural networks
Deep learning
Generative adversarial networks
Mechanical parts
Shape simplification

ABSTRACT

Most three-dimensional (3D) computer-aided design (CAD) models of mechanical parts, created during the design stage, have high shape complexity. The shape complexity required of CAD models reduces according to the field of application. Therefore, it is necessary to simplify the shapes of 3D CAD models, depending on their applications. Traditional simplification methods recognize simplification target shape based on a pre-defined algorithm. Such algorithm-based methods have difficulty processing unusual partial shapes not considered in the CAD model. This paper proposes a method that uses a network based on a generative adversarial network (GAN) to simplify the 3D CAD models of mechanical parts. The proposed network recognizes and removes simplification target shapes included in the 3D CAD models of mechanical parts. A 3D CAD model dataset was constructed to train the 3D CAD model simplification network. 3D CAD models are represented in voxel form in the dataset. Next, the constructed training dataset was used to train the proposed network. Finally, a 3D voxel simplification experiment was performed to evaluate the performance of the trained network. The experiment results showed that the network had an average error rate of 3.38% for the total area of the mechanical part and an average error rate of 14.61% for the simplification target area.

1. Introduction

Due to advances in computing power, 3D CAD models are now used for various purposes such as product design, engineering simulations, and virtual prototyping. CAD models have different degrees of shape complexity according to their purpose. 3D CAD models of mechanical parts, created in the design stage, include features such as holes, pockets, chamfers, and fillets and have a high level of detail (LOD). However, when these models are used for manufacturing simulations, design reviews, visualization, and virtual training, the model's connectivity between parts and overall appearance is more important than its detailed shape. Therefore, in real-world applications, there is a need for methods to simplify the shapes of original 3D CAD models according to their purpose. For example, if engineering analysis is performed on a part model consisting of hundreds of features, the computational time will significantly increase. Additionally, rendering and transmitting complex 3D models requires large amounts of storage, computing power, and bandwidth.

Traditional simplification methods evaluate the importance of shape elements, select the elements to be simplified according to importance, and remove the selected elements based on a pre-defined algorithm [1–5]. Such algorithm-based methods produce different simplification results according to the simplification algorithm used. In addition, they have difficulty processing unusual partial shapes not considered in the CAD model. They also have difficulty performing simplification tasks quickly because they require many geometric operations to evaluate the importance of shape elements and remove them. Finally, the algorithms that can be used vary depending on the shape representation method used in the CAD model.

In this study, we propose a learning-based simplification method that uses a generative adversarial network (GAN) [6], which, unlike traditional algorithm-based methods, is a deep learning technique to simplify 3D CAD models of mechanical parts. The proposed simplification network consists of a generator and a discriminator. The generator receives the original CAD model as input and generates a sample simplified CAD model. Then, the discriminator compares this model to real simplified

* Corresponding author.
E-mail addresses: holee0401@korea.ac.kr (H. Lee), jwlee@gwnu.ac.kr (J. Lee), soonjo.kwon@kumoh.ac.kr (S. Kwon), ramani@purdue.edu (K. Ramani), hgchi@purdue.edu (H.-g. Chi), dhmun@korea.ac.kr (D. Mun).

https://doi.org/10.1016/j.cad.2023.103577
0010-4485/

CAD models and classifies the data created by the generator either as generated data or real data. Next, in the learning process, adversarial learning is performed in which the generator and discriminator are developed competitively to train the generator for the process of identifying and removing shapes that must be simplified in the original 3D CAD model. During adversarial learning, the original 3D CAD model and the simplified model are used as training data.

This study has the following differences from preexisting algorithm-based simplification studies. First, in conventional methods, users should determine simplification target shape or LOD. If it is impossible to recognize simplification target shape using predefined algorithm, the shape cannot be simplified. In our proposed method, simplification target shape depends on training dataset unlike the conventional methods. Accordingly, required user intervention can be significantly reduced. By augmenting the training dataset, it is possible to recognize various types of simplification target shapes. Second, the biggest problem of algorithm-based simplification is that if a small change in the model prevents a particular algorithm from recognizing the simplification target shape or region, no change may occur in the model. In other words, if the geometry in a model is the same and the topology is different, one model can be simplified and the other cannot. Our proposed network uses only voxel-based geometry information. Therefore, the network can generate a simplification result regardless of the difference of topological structure.

In this study, a voxel model with a relatively low resolution compared to the original CAD model was used. The boundary representation (b-rep) is the de-facto shape representation in CAD to model solid and sheet objects [7]. The reason why we used voxel instead of b-rep is as follows. B-rep data is composed of various geometric and topological entities that have different sets of parameters, including parametric curves and surfaces. Additionally, there is no one-to-one mapping between a shape and a surface type. As a result, it is not possible to input raw surface information, such as parametric coefficients or spline control points, directly into a neural network because it would not be invariant to the specific parameterization [8]. Therefore, in the field of 3D deep learning, mesh, point cloud, and voxel forms are mainly used. Voxels provide an efficient and regular representation of 3D space that can be processed effectively by CNN. Similar to pixels in an image, voxels are 3D units that are abstracted, with predefined volumes, positions, and attributes, and can be used to structurally represent discrete points in a topologically explicit and information-rich manner [9]. Compared to b-rep, voxels mainly have the advantage of having a uniform resolution over the entire shape, and the representation is very simple, making it robust to small changes or noise in the geometry. Recently, there have been studies where signed distance fields (SDF) have been added to voxel data, improving the performance of CNN-based shape recognition in 3D CAD models [10]. Thus, voxel data's validity has been consistently demonstrated in the CAD/CAM field.

This study's contributions are as follows. First, we propose a learning-based method rather than a conventional algorithm-based method to perform shape simplification of 3D CAD models. Second, we developed a new simplification network that combines variational autoencoders (VAEs) [11] and WGAN (Wasserstein Generative Adversarial Network) [12] to control the shapes of 3D CAD models of mechanical parts. To the best of our knowledge, the proposed network is the first GAN-based network developed for shape simplification of 3D CAD models. Third, we constructed a training dataset that consisted of approximately 1000 pairs of 3D CAD models (CAD models before and after simplification) before augmentation. This dataset can be used for various purposes in 3D deep learning applications.

This paper is organized as follows. Section 2 describes research on the simplification of 3D CAD models and research on deep learning in the field of 3D CAD. Section 3 summarizes the proposed method and proposes a network for simplifying 3D CAD models. Section 4 discusses the results of implementing the proposed deep learning network and performing 3D voxel simplification experiments. Finally, Section 5 presents our conclusions.

2. Related research

2.1. 3D CAD model simplification technology

Studies on 3D CAD model simplification can be classified into one of the following categories according to the model type: Polygon-based methods [13–16], boundary-representation (B-rep) based methods [17–20], and feature-based methods [2,21,22]. Polygon-based methods (Fig. 1(a)), most used in computer graphics, simplify models by reducing the number of triangle meshes. These methods produce good results when used on regular, dense meshes that are used in the field of computer graphics. However, the meshes generated by 3D CAD models are generally not regular or dense. Therefore, the characteristics of the original shape can become distorted if mesh-based simplification is applied to CAD models. B-rep-based simplification methods (Fig. 1(b)) use topology information to simplify shapes and are classified into two types [23]. Dimensional reduction methods convert thin solids into faces or convert long solids into edges. Feature suppression methods simplify shapes by removing shapes such as rounds, fillets, and holes that do not significantly affect engineering analysis. Feature-based methods (Fig. 1(c)) simplify shapes by sequentially removing features with low importance from CAD models [1–5]. Each feature's importance varies according to the usage goals of the model. In B-rep-based simplification methods, the use of feature removal is limited to features recognizable by an individual recognition algorithm, but feature-based simplification methods can use feature removal on all features.

Shape simplification is used to find and remove shape elements that have low importance in a model when the LOD of the model needs to be reduced according to its purpose. In the simplification process, simplification target shape elements are removed, and the rest of shape elements must be kept the same as the original model. For this purpose, the volume of the simplification target shape elements must be removed (e.g. boss) or added (e.g. hole) based on the boundary separated by edges or faces in the CAD model. The importance of shape elements varies according to the evaluation criteria. Accordingly, the simplification result may differ from the designer's intention according to the evaluation criterion and parameter values (e.g. fillet radius, hole diameter). Once the 3D CAD model representation (mesh, point cloud, solid etc.) is changed, the algorithm applicable only to a specific representation method cannot be used. To simplify bosses and holes, a robust feature recognition algorithm is required. However, even if the recognition algorithm is robust, the recognition result may vary depending on the quality of the model to which the algorithm is applied, which may lead to differences in simplification results. For example, it may be difficult to expect identical recognition results in cases where a cylinder is expressed as one surface in a B-rep model, and where it is expressed as two or more surfaces.

2.2. 3D deep learning in the field of 3D CAD

2.2.1. Classification and part segmentation

Object classification is a technology that recognizes the object types represented in 3D models, such as chairs or tables; various

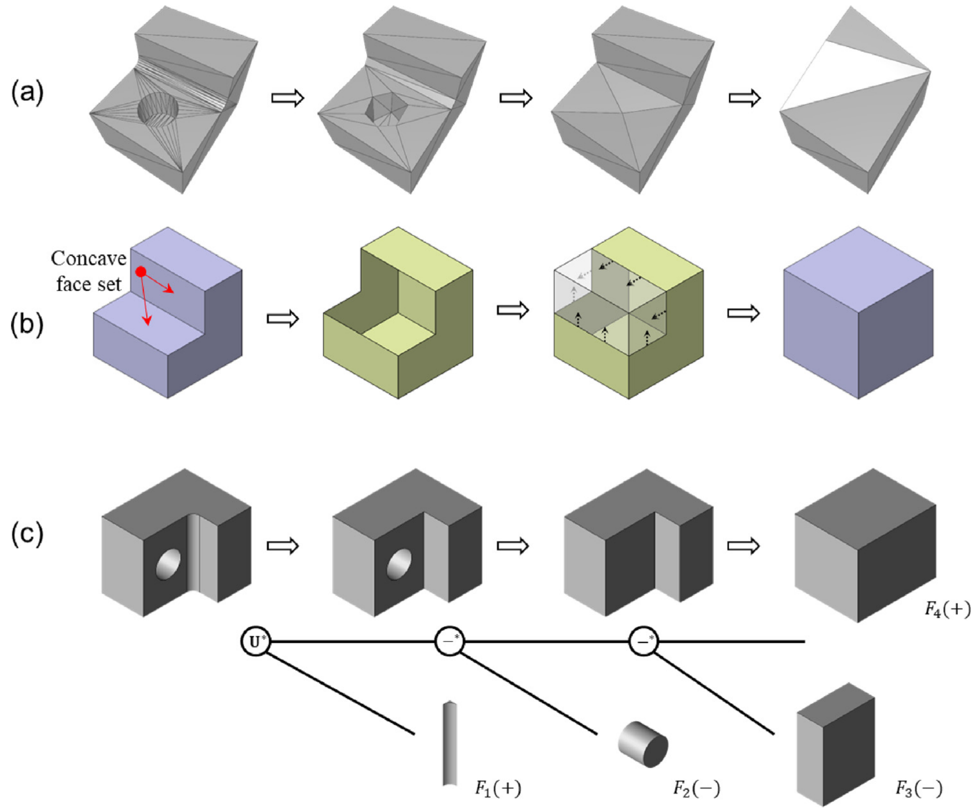

Fig. 1. 3D CAD model simplification methods: (a) polygon-based simplification method; (b) B-rep-based simplification method; and (c) feature-based simplification method.

approaches that use deep neural networks (DNNs) have been proposed in previous related studies. CNNs have shown remarkable performance in 2D image classification. Therefore, research that targets 3D data is now being undertaken and can be classified into multiview-based methods [24–26], volumetric methods [27–30], and point-based methods [31–33].

Shape segmentation is a technology that divides the subparts that constitute an input 3D shape or semantically segments the 3D shape. Recently, researchers have developed deep learning models for segmenting 3D shapes, which are represented in various ways, including volumetric grids [28,29], point clouds [34–36], multi-view rendering [37], and surface meshes [38,39].

2.2.2. Shape generation and simplification

Shape reconstruction is a technology that compresses 3D shapes into latent space with a 3D encoder, uses this as input, and generates 3D shapes using a 3D decoder. Unlike traditional multi-view stereo algorithms, deep learning models can encode prior knowledge of the space of 3D shapes, which can help resolve ambiguities in the input data [40]. Shape reconstruction technology can be classified into voxel-based reconstruction [41–45], point cloud-based reconstruction [26,46–48], and mesh-based reconstruction [49,50] according to the input data representation method.

As the image generation ability of GAN has been proven through many experiments, studies on the generation of 3D models have also been proposed. Li et al. [51] uses conditional GAN to generate voxel-based 3D models from a single image. Yu et al. [52] uses Point Encoder GAN to inpaint 3D point clouds. Liu et al. [53] uses MapGAN to reconstruct 3D models from a single image. Yang et al. [54] uses X2CT-GAN to generate 3D spine data from simulated bi-planar 2D X-ray images.

Mesh and point cloud simplification methods aim to reduce the complexity of 3D models while retaining visual quality and relevant salient features. Potamias et al. [55] proposed a method of simplifying point clouds using a graph neural network. In their follow-up study, they proposed a method to learn the mesh connection distribution in an unsupervised learning manner using a graph neural network [56].

2.2.3. Data translation

Image-to-image translation is the task of changing certain aspects of the original image. With the introduction of GAN, this task has allowed for improvements in areas such as changing hair color [57], reconstructing images from edge maps [58], and performing style transfers on certain images [59].

For example, Pix2Pix was trained to perform an image translation task under supervised learning using conditional GANs (CGANs) [58]. Pix2Pix requires paired data samples because it combines adversarial loss and L1 loss. To resolve the problem of having to obtain data pairs, unpaired image-to-image translation frameworks have been proposed [57,60,61]. UNIT [60] created a GAN framework by combining variational autoencoders (VAEs) [11] and CoGAN [62]. The two generators that make up this framework share weights to train for the joint distribution of images in cross domains.

CycleGAN [61] and DiscoGAN [57] preserve the key attributes between input images and translated images by applying cycle consistency loss. However, these frameworks can only be trained on the relationships between two different domains at a time.

3. Proposed network

3.1. Background

In this section, we present our baseline models and explain the theory of each model. The baseline models for this study are

Pix2Pix [58] and Shape Inpainting [63]. Isola et al. [58] proposed a network that uses conditional GAN for image-to-image translation [64]. They used a U-net-based generator and a PatchGAN-based discriminator. To perform image translation, the network must be trained to translate the input data into the ground truth. By combining the input image and the ground truth image, they supplied auxiliary information corresponding to the condition to be translated. Wang et al. [63] proposed a network that uses GAN and recurrent convolutional neural networks (RCNN) to perform shape completion on corrupted 3D scan data. They used GAN to reconstruct low-resolution voxels from corrupted 3D scan data, while RCNNs were used to generate high-resolution voxels from the low-resolution voxels.

3.1.1. Variational autoencoder

VAE [11] consists of a network that encodes a data sample x into latent variables z and a network that decodes the latent variables into a reconstructed data $\tilde{x}$ (Eq. (1)).

$$z \sim Enc\left(x\right) = q\left(z|x\right), \tilde{x} \sim Dec\left(z\right) = p(x|z) \tag{1}$$

The main limitation of VAE is that the generated samples tend to be blurry. This limitation is caused by imperfect elementwise measures, such as squared error and injected noise.

3.1.2. Generative adversarial network

GAN trains two models simultaneously. The generator G generates the sample $G(z)$ from the prior $p(z)$ to mislead the discriminator D. The discriminator is trained to distinguish the generated samples from the given ground truth data. The objective function of GAN [6] is shown in Eq. (2).

$$\mathcal{L}_{GAN}\left(G, D\right) = E_{x\sim p_{data}(x)}\left[logD\left(x\right)\right] + E_{z\sim p_z(z)}\left[\log(1 - D\left(Gz\right))\right] \tag{2}$$

GAN has disadvantages in that the training process is generally unstable and the generated samples may be unnatural. Currently, many studies are being conducted to improve the stability of the training and the quality of the generated samples.

3.1.3. Wasserstein generative adversarial network

Adversarial learning of traditional GAN is formulated as minimizing the Jenson–Shannon divergence between the probability distributions of the real data and the generated data. However, this is a major cause of GAN's instability. To improve upon this method, Wasserstein GAN [12] substitutes Earth-Mover's distance (EMD) for Jenson–Shannon divergence (Eq. (3)).

$$W\left(X_r, X_g\right) = \inf_{\gamma \sim \Pi(X_r, X_g)} \mathbb{E}_{(X_r, X_g)\sim\gamma}[\|X_r - X_g\|] \tag{3}$$

$\Pi(X_r, X_g)$ represents all joint distributions of the real data's distribution X_r and the generated data's distribution X_g as defined in the original GAN. γ(x, y) is the mass that must be moved to transform distribution X_r to distribution X_g. The EM distance is the cost of the optimal transport plan. The loss function of WGAN is shown in Eq. (4).

$$\mathcal{L}_{WGAN} = \mathbb{E}_{x\sim p_g(x)}\left[D\left(x\right)\right] - \mathbb{E}_{x\sim p_r(x)}\left[D\left(x\right)\right] \tag{4}$$

Where p_r represents the probability distribution of the real samples. p_g represents the probability distribution of the samples generated by generator. $\mathbb{E}$ is a mathematical expectation that indicates the average value is calculated multiple times during the actual operation.

In addition to variational autoencoders, generative adversarial networks, and Wasserstein generative adversarial networks, researchers have also proposed networks that combine VAE and GAN, such as VAE–GAN [65] and adversarial autoencoders [66]. The studies that have been mentioned in Section 3.1 up to this point have significantly influenced the development of our simplification network.

3.2. 3D CAD model simplification network

Our objective is to implement a network that simplifies 3D CAD models of mechanical parts using a learning-based method. In this study, the word "simplification" means the recognition and removal of simplification target shapes included in the original 3D CAD model. The input model and the output model used by the proposed network are both expressed as voxel grids. To represent the 3D CAD model, we only used occupancy information comprising of 1s and 0s. Here, 1 indicates an occupied cell, and 0 indicates an empty cell. The resolution of the voxels is 64 × 64×64.

Our network consists of a generator and a discriminator. This network was defined by combining VAE and GAN [65]. In several studies, by utilizing VAE–GAN, they showed good performance not only in image fields [67,68] but also in 3D data [45,69,70]. Our goal is to implement a network that can recognize and remove simplification target area from input voxel. In the voxel simplification experiment, in the case of autoencoder (AE) and VAE, the error rate for the part total area was low, but the error rate for the simplification target area was relatively high. In this study, we were able to overcome this issue through adversarial learning using WGAN. The results of the voxel simplification experiment are mentioned in Section 4.1.1.

The generator's baseline is that of a variational autoencoder (VAE) [11]. We configured the generator's layers by referencing our previous study [71]. In our proposed method, the simplification target shape included in the original 3D CAD model is expressed as an empty cell, and the same shape included in the ground truth is expressed as an occupied cell. To simplify the 3D CAD model, the simplification target shape area included in the original model must be outputted as an occupied cell. At the same time, the area that is not the simplification target shape must be the same as the original model in the output. To do this, during the generator training, mean square error (MSE) is used to compare the generated model and the ground truth and calculate loss. Also, the discriminator compares the difference in the data distribution of the generated model and the ground truth to calculate loss. To perform this kind of adversarial learning, it is necessary to have a paired dataset that pairs original 3D CAD models and simplified models.

An overview of the proposed 3D CAD model simplification network is shown in Fig. 2. First, when the original 3D CAD model x is provided as input to the generator G, the average (μ) and variance (σ) are produced as output by the 3D encoder. Then, epsilon (ε) is created by a sampling function following Gaussian probability distribution. Epsilon is multiplied by the variance (σ), and this is added to the average (μ) to determine the value of the latent vector. Next, the 3D decoder generates a shape from the latent vector. Here, the 3D decoder must generate a shape in which the simplification target shape is removed from the original model. The model (G(x)) generated by the decoder is used to calculate the MSE loss and the original GAN's adversarial loss, along with the simplified model (y) that corresponds with the ground truth. After this, G(x) and y are provided as input to the discriminator D to calculate the EM distance.

3.2.1. Objective function

To intentionally mislead the discriminator, the generator must create a voxel model similar to the ground truth(y). The mean square error (MSE) loss for training the generator is shown in Eq. (5).

$$\mathcal{L}_{MSE} = \frac{1}{m}\sum_{i=1}^{m} \|G\left(x^{(i)}\right) - y^{(i)}\|^2 \tag{5}$$

m is of model index. $G(x)$ is the model that the generator creates by simplifying the original 3D CAD model x. $G(x)$ is entered in the MSE loss calculation formula along with the ground truth y.

The proposed network's objective function is shown in Eq. (6).

$$\mathcal{L}_{simplification} = \mathcal{L}_{WGAN}(D) + \mathcal{L}_{MSE}(G) + \mathcal{L}_{GAN}(G) \tag{6}$$

In the equation above, the generator receives the original 3D CAD model x as input and generates the simplified model G(x). EM distance is calculated to train the discriminator D. This calculation trains the network to ensure that the distribution of G(x), which was created by the generator, is similar to the distribution of the ground truth. After this, to train the generator G, the MSE between G(x) and the ground truth is calculated, and the original GAN's adversarial loss is calculated. These calculations directly compare the ground truth and the G(x) that was generated by the generator, and the network is trained to minimize the error. In addition, weights are updated to minimize the sum of the EM distance, MSE, and the adversarial loss of the original GAN. By using this objective function, we could train the generator to recognize the simplification target shape included in the original model and generate a simplified model.

In WGAN, adversarial loss is calculated using data distribution to train a generator. When this method was used in our network, blurry models were generated, or mode collapse occurred. In addition, performing calculations by combining L1 distance and adversarial loss, like the Pix2Pix objective function, is not suitable for generator training. As such, the proposed method uses WGAN's adversarial loss to train the discriminator, and it uses MSE and the original GAN's adversarial loss to train the generator.

3.2.2. Model architecture

The structure of the proposed network is as follows. The formats of the input model and the output model are both voxel formats, and they have one channel that represents known space (occupied space) and unknown space (empty space). Known space refers to the area occupied by voxels, while unknown space refers to the area devoid of voxels. The generator consists of an encoding layer, latent space, and decoding layer. The discriminator consists of an encoding layer.

A 3D voxel with a size of 64 × 64 × 64 is inputted into the generator's encoding layer. First, the voxel goes through a dropout layer with a probability of 0.5. Next, a low-dimensional latent space is generated through four 3D convolutional layers. In the first convolutional layer, the kernel size is set to 4 × 4 × 4, and the stride is set to 2 × 2 × 2. In the second to fourth convolutional layers, the kernel size is set to 3 × 3 × 3, and the stride is set to 2 × 2 × 2. The numbers of channels in the layers are set to 64, 128, 256, and 512. As a result, 512 channels with a size of 3 × 3 × 3 are created via the convolutional layers. After this, in order to flatten the feature map into a one-dimensional vector with a length of 13,824, a fully connected layer with the same length is used.

In the generator's decoding layer, simplification of the 3D model is performed as a latent space that includes the features of the input shape passes through a 3D deconvolution layer as input. The kernel sizes, stride, layer sizes, and numbers of channels follow the reverse order of the encoding layer. In addition, a skip connection similar to that of the U-net architecture [72] was added between the encoding layer and the decoding layer. In doing so, it was possible to significantly improve upsampling performance by transferring the intermediate output of the encoding layer to the decoding layer.

In the encoding layer of the discriminator, the distribution of the input data is calculated using EM distance, and this value is minimized. The kernel sizes, stride, layer sizes, and numbers of channels are the same as in the generator's encoding layer.

The network training process is as follows. First, the original 3D CAD models and the simplified models that correspond to each original model are loaded from the training dataset. In the training of the discriminator, the EM distance that was defined in WGAN is used to calculate the difference in the distributions of the ground truth and the model generated by the generator. This value is used to update the weights of the discriminator. After this, the weights are clamped by the clipping parameter. In the training of the generator, the MSE of the ground truth and the model generated by the generator is calculated. After this, the adversarial loss of the original GAN is calculated. These values are used to update the generator's weights. The detailed training algorithm is shown in List 1.

List 1. Network training algorithm

Algorithm Training algorithm. All experiments in the paper used the default values α=0.00001, c=0.01, m=6, e=350.

Require: G, generator. D, Discriminator. α, the learning rate. c, the clipping parameter. m, the batch size. e, epochs.
Require: w_0, initial discriminator parameters. θ_0, initial generator's parameters.
for e **do**
  Sample $\{\boldsymbol{x}^{(i)}\}_{\boldsymbol{i=1}}^{\boldsymbol{m}} \sim X_o$ a batch from the original 3D CAD model.
  Sample $\{\boldsymbol{y}^{(i)}\}_{\boldsymbol{i=1}}^{\boldsymbol{m}} \sim X_r$ a batch from the simplified 3D CAD model.
  $D_w \leftarrow [\frac{1}{m}\sum_{i=1}^{m} D_w\big(y^{(i)}\big) - \frac{1}{m}\sum_{i=1}^{m} D_w\big(G_\theta(x^{(i)})\big)]$
  $w \leftarrow w + \alpha \cdot Adam(w, D_w)$
  $w \leftarrow clip(w, -c, c)$
  Sample $\{\boldsymbol{x}^{(i)}\}_{\boldsymbol{i=1}}^{\boldsymbol{m}} \sim X_o$ a batch of original 3D CAD data
    $G_\theta \leftarrow [\frac{1}{m}\sum_{i=1}^{m} \|G_\theta(x^{(i)}) - y^{(i)}\|^2 + \frac{1}{m}\sum_{i=1}^{m} \log\left(1 - D\big(G_\theta(x^{(i)})\big)\right)]$
  $\theta \leftarrow \theta - \alpha \cdot Adam(\theta, G_\theta)$
end for

3.3. Construction of network training dataset

The purpose of our network is to translate the input data. To train the network for data translation, adversarial learning must be performed using the input data and the ground truth data of the domain to be translated. Our goal is to simplify the original 3D CAD model. As such, there is a need for a paired dataset that comprises the original 3D CAD models, which correspond to the input data, and the simplified models, which correspond to the ground truth data. We constructed a training dataset for the proposed network using the process shown in Fig. 3. The small ridges seem to be present in some of the models with cylindrical shapes, as shown in Fig. 3. However, the original models have smooth cylinders without any ridges. This is because the voxel model's resolution was reduced during the process of voxelizing the original model. Our dataset consisted of voxel models with a grid size of 64 × 64×64. The training dataset was constructed using open datasets that consisted of 3D CAD models of mechanical parts. To construct the dataset, we used the 3D CAD models provided by ESB [73] and DSB [74]. However, additional modifications were performed because these 3D CAD models included many cases where the simplification target area was too small or large. We performed remodeling under the following conditions. First, simplification target shape's types are limited to holes and pockets for efficient learning. Second, considering that the voxel size is 64 × 64×64, size of simplification target shape was modeled with an increase of at least 5% compared to the overall model size. Finally, if there are holes or pockets in the original model, the size is modified to meet the size conditions above. In addition, we also modeled simplified models (y) that corresponded to each of the original 3D CAD models (x). As a result, 985 CAD model pairs (CAD models before and after simplification) were prepared. Of these, the 98 CAD model pairs that were to be used in the simplification experiments in Section 4 were excluded. The CAD model pairs that were to be used in the simplification experiments were selected by selecting one model from among several models with the same part type.

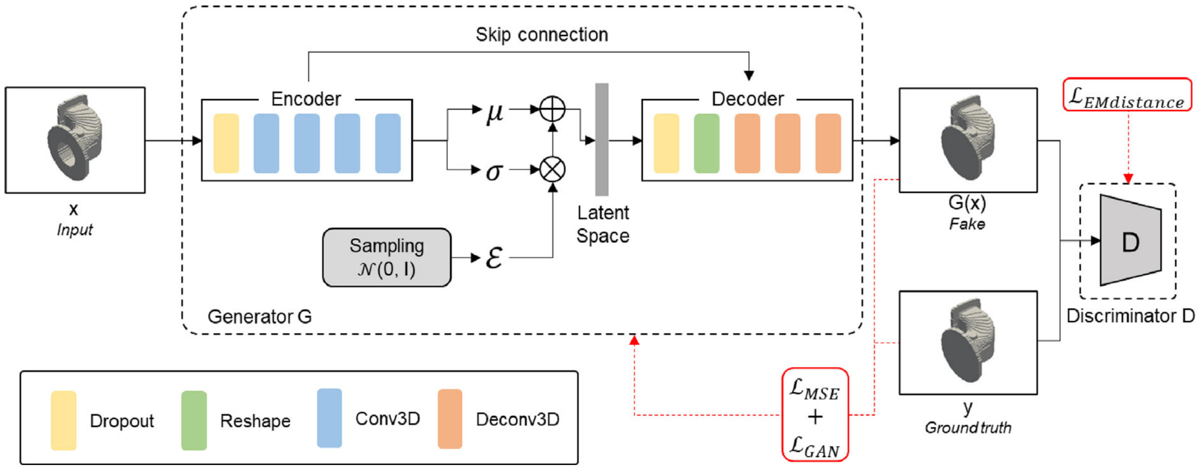

Fig. 2. Overview of 3D CAD model simplification network.

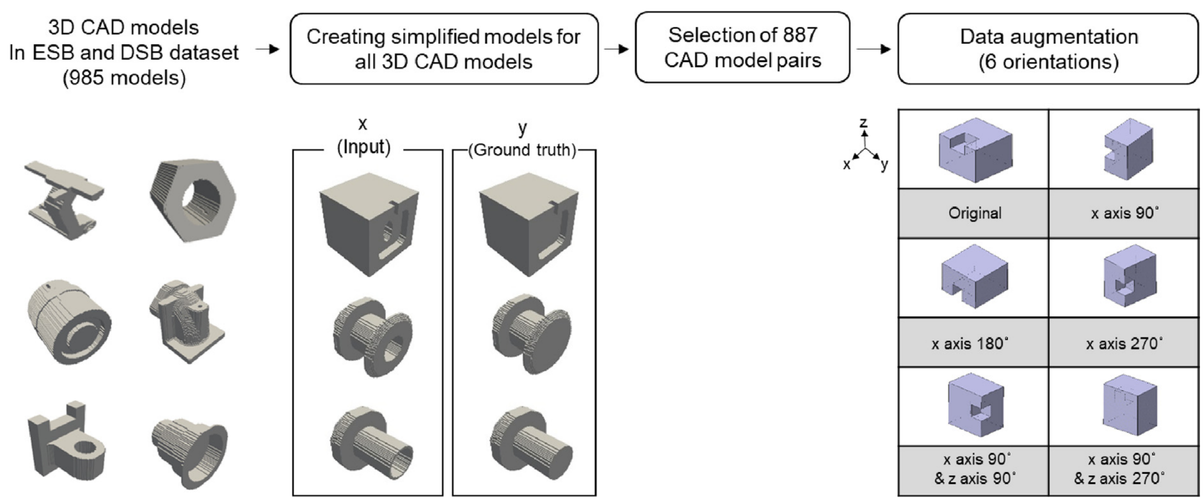

Fig. 3. Construction of 3D CAD model dataset for network training.

After this, data augmentation was performed on the remaining 887 CAD model pairs.

In the data augmentation process, we allowed the 3D CAD model to be rotated around the *x* and z axes and set up a total of 6 orientations. For example, "x-axis 90° and *z*-axis 270°" means rotating the model 90° counter-clockwise around the *x*-axis and then rotating it 270° counter-clockwise around the *z*-axis. This process expanded the number of CAD model pairs in the training dataset to 5322. After this, these files were converted to HDF5 [75]. HDF5 is a data format generally used to manage large-scale data. To reduce the data size of the HDF files, one-bit bools were used in the voxel representation.

4. Implementation and experiments

To train the deep learning model, Python 3.7.7 and the TensorFlow 2.2.0 library were used. All experiments were performed on a computer with 128 GB memory, 2 Nvidia GeForce RTX 3090 GPUs and an Intel Core i9-10900K CPU (3.7 GHz). The 3D CAD model simplification network was implemented using the configuration described in Section 3. Adam was used for gradient descent optimization [76]. The epoch was set at 350. The batch size was set to six. The learning rate was set to 1e-05.

As shown in Fig. 4, 2 test cases were used in the 3D voxel simplification experiments. Test Case A consists of 98 models, which were not used for network training in the dataset built in the way of remodeling the ESB and DSB datasets in Section 3.3. The ESB's 3D model covers a wide range of geometries with many real-world engineering artifacts. They classify 3D models into three super-classes: solids of revolution, rectangular–cubic prism or prismatic, and thin-walled. The DSB dataset contains various types of data, including synthetic datasets composed mainly of primitive shapes and actual artifact datasets composed of mechanical engineering parts. We calculated the average error rate of the simplification target area under various configurations for Test Case A to determine the optimal hyperparameters and loss function. The sensitivities of the hyperparameters are shown in Fig. 5, and the performance variations according to different types of loss functions are shown in Table 1.

Test Case B consists of a total of 12 models, with four models selected from the CADNET dataset [77], MCB dataset [78], and Fusion360 dataset [79], respectively. The CADNET dataset includes engineering CAD models with 42 classes, such as elbows, bearings, flanges, nuts, etc. The MCB dataset consists of mechanical components with 68 classes, such as caps, flanges, nozzles, bolts, etc. 3D models from the Fusion 360 dataset were modeled using sketch and extrude, including washers, pegs, plates, etc. The 3D CAD models that made up Test Case B did not include simplified models. Therefore, the experiments results were confirmed by visualizing the simplified models.

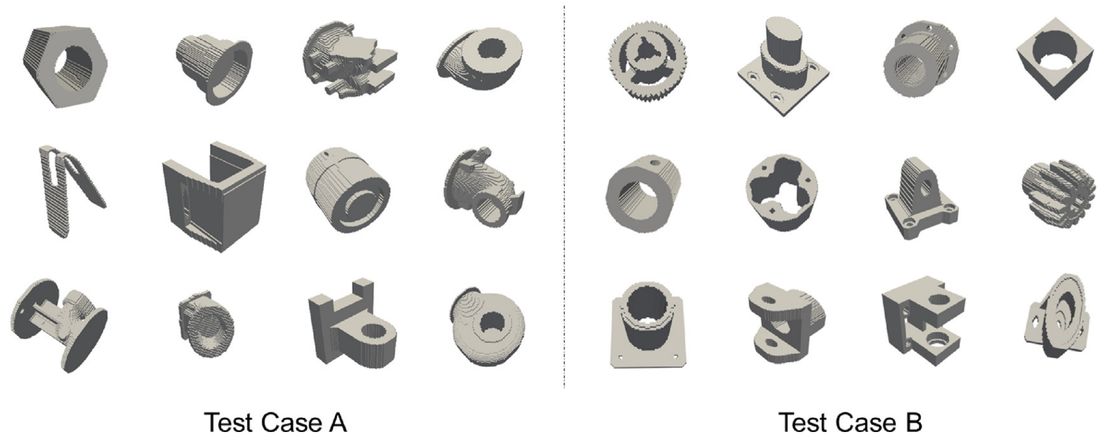

Fig. 4. Results of 3D voxel simplification according to various parameter configurations.

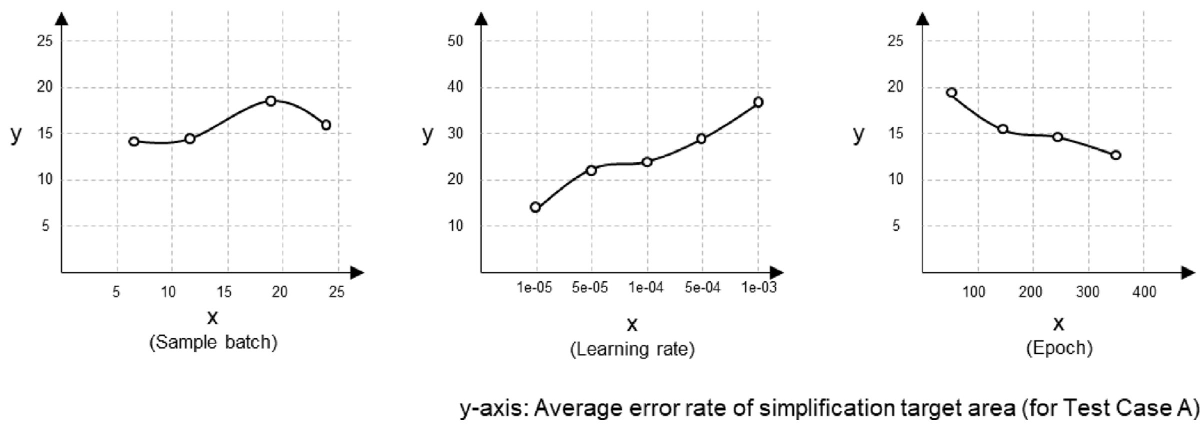

Fig. 5. Composition of test cases.

Table 1
Average error rate of simplification target area according to types of loss functions.

Loss function	Average error rate of simplification target area (%)
Mean Squared Error	14.61
Mean Absolute Error	15.53
Huber	16.38

4.1. 3D voxel simplification

This section describes the method for calculating the proposed network's 3D voxel simplification performance via the average error rate. Then, we compared the results of performing 3D voxel simplification experiments using several GANs including our proposed network.

Studies on the 3D reconstruction based on deep learning use dice similarity coefficient, Jaccard similarity coefficient, overlap volume, and structural similarity index measures as performance evaluation indicators [54]. These methods are often used to compare the similarity of data contributions. However, in our proposed method, since we have ground truth that can be directly compared with the model generated by the generator in the test dataset, we used a method of directly comparing the voxels of the generated model and the ground truth. Furthermore, in order to measure simplification performance, it is necessary to obtain an error rate for simplification target area rather than the part total area of the voxel model. For the reasons above, the average error rate for the part total area of part and the average error rate for the simplification target area were used as performance evaluation indicators.

3D voxel simplification performance is calculated by comparing the ground truth and the model that the generator created by simplifying the input model. The average error rate is defined by Eq. (7).

$$Average\ error\ rate = \frac{100}{N * q}\sum_{i=1}^{N}\sum_{j=1}^{q} {V_j^{(i)}}_{(GT)} - {V_j^{(i)}}_{(Gen)} \tag{7}$$

N is the total number of test models that are used in the voxel simplification experiment. q is the total number of voxels that make up the input data. i is the index for the test models. j is the index for the voxels included in the ith test model. $V_{j(GT)}$ is the occupancy information of the jth voxel of the ground truth. $V_{j(Gen)}$ is the occupancy information of the jth voxel of the model created by the generator. Because the occupancy information that we use is expressed as 1 or 0, the error rate can be calculated through subtraction between voxels that are in the same location.

We determined that there was a limit to simplifying the target local shape while maintaining the overall shape of the model only with VAE. In GAN, in terms of 3D shape simplification, in order to fool discriminator, generator trains by repeating the process of recognizing and removing local shape area. Since this training method was thought to be effective in shape simplification of 3D CAD models, our network was constructed by combining VAE and GAN. In this study, simplification target shapes of training dataset were limited to machining features (hole, pocket, chamfer, fillet). It is assumed that simplification is performed only when the simplification target shape is smaller than a certain size.

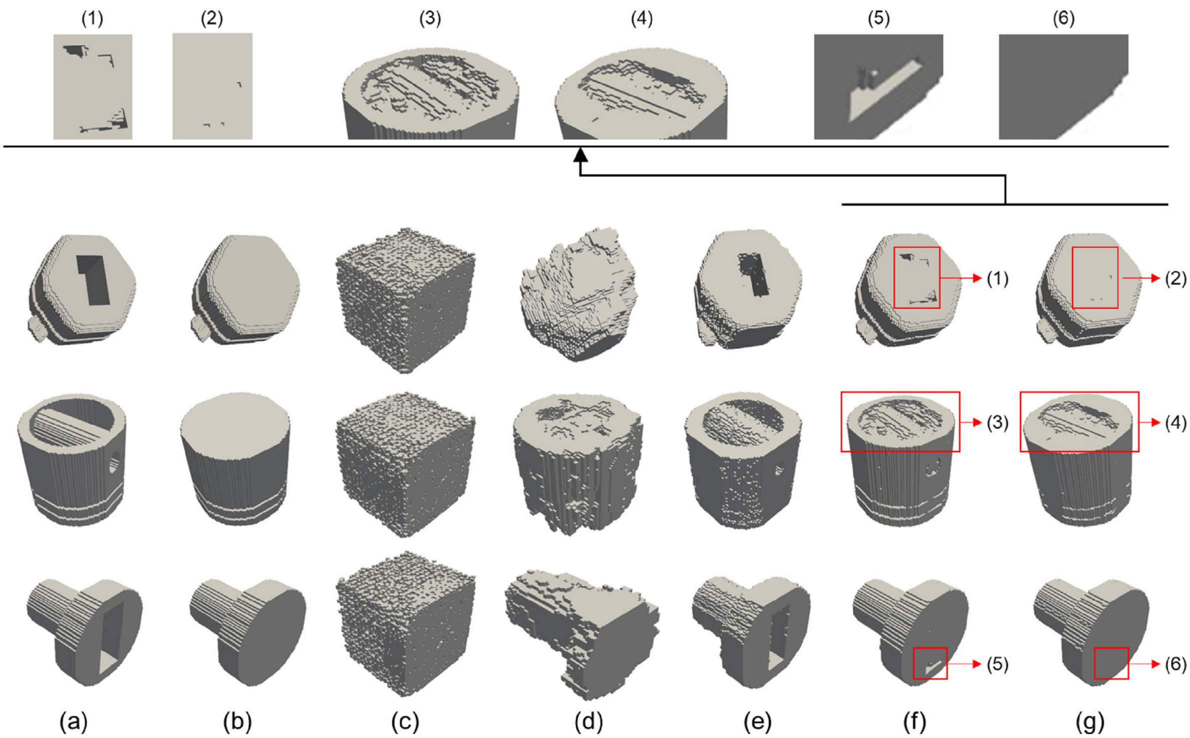

Fig. 6. Results of 3D voxel simplification for Test Case A: (a) input model; (b) ground truth; (c) original GAN; (d) Shape inpainting; (e) WGAN; (f) Pix2PixGAN; and (g) proposed network.

To verify effectiveness of the proposed network, a 3D voxel simplification experiment was conducted using AE and VAE structures. As shown in Table 2, when the AE and VAE were used, the average error rate of part total area was calculated to be approximately 3%. However, the average error rate of simplification target area was calculated to be 23% or higher. Except for our study, AE and Pix2PixGAN methods showed good performance in terms of the average error rate of part total area. These methods compress the input data into latent space without sampling. Comparing the simplification performance, the average error rate of simplification target area was calculated to be about 10% higher than our method. Through 3D voxel simplification experiments, we confirmed that VAE/GAN has a better simplification performance than AE or Pix2PixGAN.

Table 2
Average error rate obtained by various network models for Test Case A.

Dataset		Method	Average error rate of part total area (%)	Average error rate of simplification target area (%)
Test Case A	Auto encoder	VAE [11]	3.72	30.49
		AE [80]	3.36	25.32
	Generative Adversarial Networks	Original GAN [6]	51.15	80.83
		Shape inpainting [63]	16.23	26.51
		WGAN [12]	12.37	41.31
		Pix2PixGAN [58]	3.94	27.45
		Ours	3.38	14.61

Table 2 shows the error that occurred when different networks were used to simplify the 3D voxels of Test Case A. The "average error rate of part total area" in Table 2 is the average value of the error rate that was calculated for voxels in the area of the overall shape of the test model. The "average error rate of simplification target area" is the average value of the error rate calculated only for voxels in the area of the test model's simplification target shape. The experiment results show that the proposed network exhibited the best performance with an "average error rate of part total area" of 3.38% and an "average error rate of simplification target area" of 14.61%.

4.1.1. GAN

Fig. 6 shows the results of performing 3D voxel simplification experiments on 3 random models from among the 3D CAD models of Test Case A. Fig. 6(c) shows the original GAN's 3D voxel simplification results, in which mode collapse occurred and only voxels for a cubic shape were generated. Mode collapse is an error in which the generator only generates one sample or a set of very similar samples. Fig. 6(d) shows the Shape inpainting's 3D voxel simplification results. Fig. 6(e) and (f) are the 3D voxel simplification results of WGAN and Pix2Pix, respectively. Of these networks, Pix2Pix's "average error rate of part total area" was similar to that of the proposed network. However, when the "average error rate of simplification target area" values were compared to that of the proposed network, it was found that Pix2PixGAN's value was 12% higher and WGAN's value was 26% higher. Fig. 6(g) shows the model that was simplified using the proposed network, and it had the lowest error rate in the area of the simplification target shape in the test model. Fig. 6(1)–(6) are enlargements of the simplification results produced by Pix2PixGAN and the proposed network for the simplification target area in the 3D voxel model. Fig. 7 shows the results of using the proposed method to perform 3D voxel simplification experiments on Test Case B. The experiment results confirmed that shape simplification could also be performed on 3D CAD models that were not used to train the network. However, for some of the models, shape simplification was not performed completely. As seen in Fig. 7, area (a) in Model 3 was not recognized as the simplification target shape. In Models 4, 6, and 9, the area around the outer boundary was not completely simplified. It takes approximately 2 s to create each simplified model.

Test Case A and Test Case B mainly consist of 3D CAD models of mechanical parts, including real-world engineering components. The simplification outcomes for Test Case A and Test Case

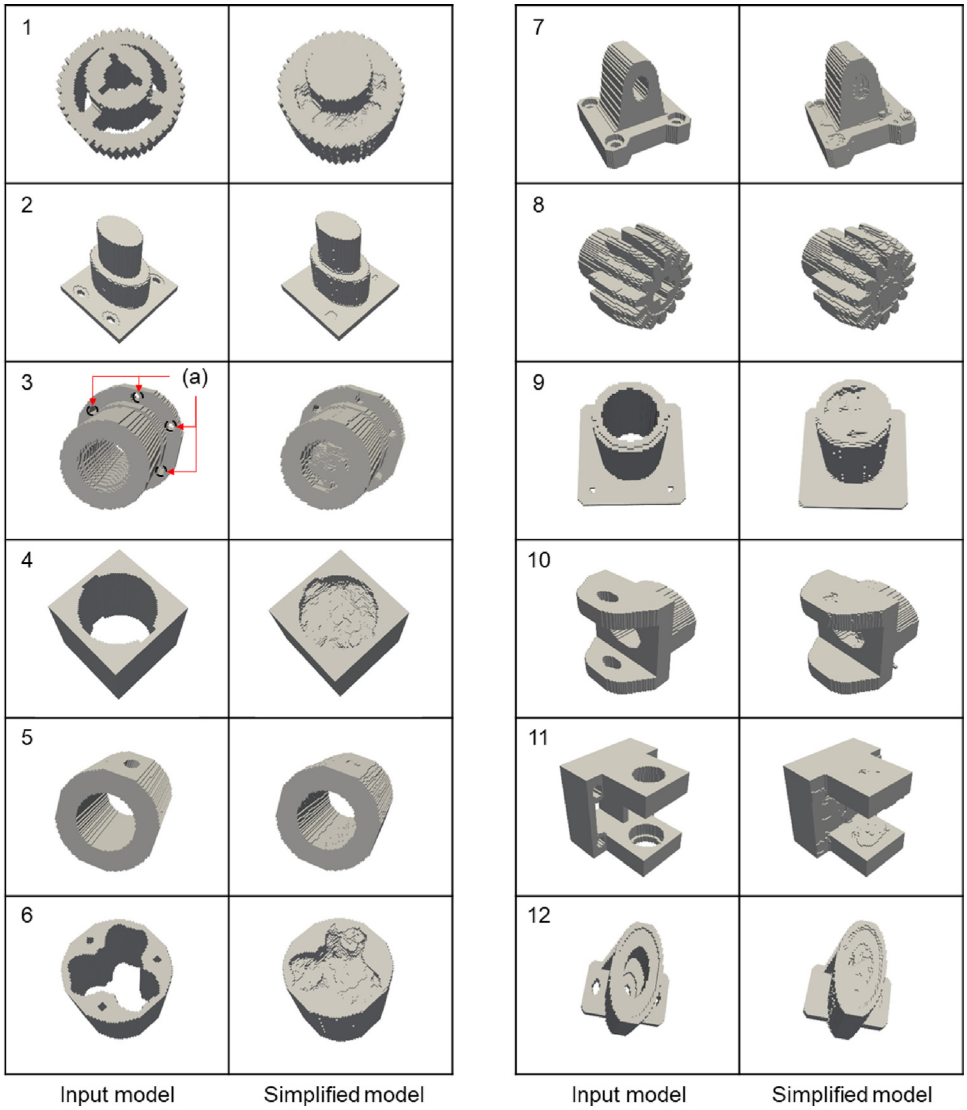

Fig. 7. Results of 3D voxel simplification for Test Case B.

B, as shown in Figs. 6 and 7, respectively, were quite similar. For instance, the model in the second row of Fig. 6 and models 7 and 9 in Fig. 7 have a cylindrical shape and include the simplification target shape. Our method successfully identified the inside of the cylinder as the simplification target shape in all models.

4.1.2. VAE and AE

To compare the simplification performance of VAE, AE and GAN, we defined each network and performed a 3D voxel simplification experiment. In VAE and AE, MSE (L2 loss) was used for loss calculation. The network structure of VAE is defined identically to the encoder structure of the proposed GAN network, as shown in Fig. 8(a). Unlike the network structure of VAE, the network structure of AE creates a latent space without sampling, as shown in Fig. 8(b).

Fig. 9 shows the results of the 3D voxel simplification experiment for Test Case A (Model 1–Model 3) and Test Case B (Model 4–Model 6) in VAE, AE and GAN. According to the experimental results, the reconstruction performance of the original model was similar, but the simplification performance of GAN was higher. As shown in Fig. 9(a) and (b), VAE and AE performed worse than GAN in recognizing and removing the simplification target area. In Fig. 9(c) and (d), the reconstruction performance of VAE and AE was also worse than GAN. In particular, in the case of Fig. 9(d), the voxels expressed as empty space in the original model were reconstructed as occupied space.

4.1.3. Algorithm-based method

To compare simplification performances of the proposed method and conventional algorithm-based simplification methods, a simplification experiment was performed using method proposed by Kwon et al. [23]. They used B-rep model-based simplification method using volume composition algorithm. For the experiment, one test model was selected in Test Case A and two test models were selected in Test Case B. These models were remodeled in B-rep format for use in algorithm-based simplification methods. As shown in Fig. 10, in the case of Model 1, when the conventional method was used, the ellipse in the center and the four shapes on the lower surface were simplified and showed good performance, as shown in Fig. 10(a). However, in the case of Model 2, simplification proceeded without preserving the appearance of the model as shown in Fig. 10(b). Furthermore, holes were not removed. Similarly, in the case of Model 3, the same problem as in Model 2 occurred when the simplification was performed, as shown in Fig. 10(c). The reason for the limitation of this conventional method is that the simplification results are

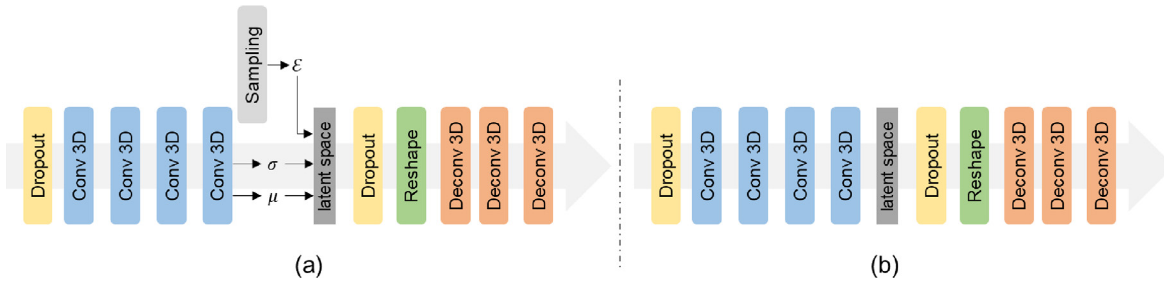

Fig. 8. Network architecture ((a) variational autoencoder, (b) autoencoder).

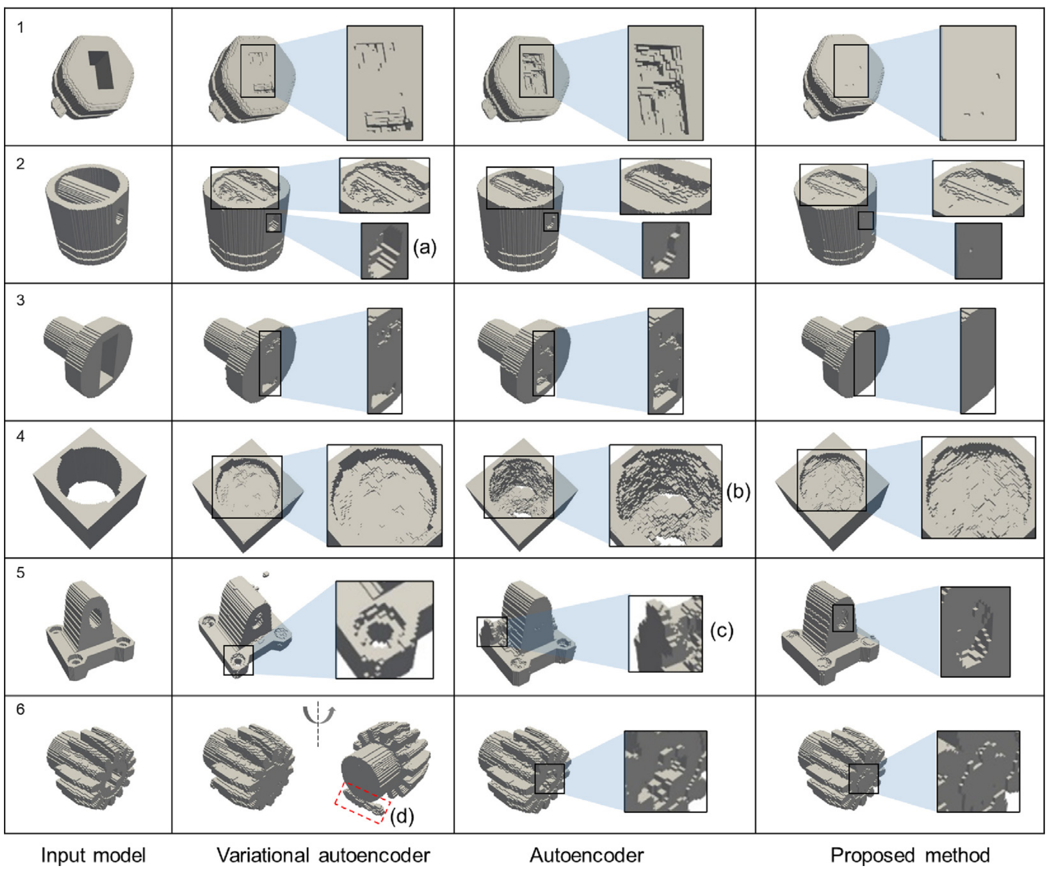

Fig. 9. Results of 3D voxel simplification by the proposed method, VAE, and AE.

greatly affected by the characteristics of the algorithm designed and human intervention (level of detail input by a user). Contrary to the conventional method, the proposed method has an issue that some of the target shapes were not clearly simplified, as shown in Fig. 10(d) and (e). It is mainly caused by the use of voxels as input. Therefore, this problem can be solved if the proposed network is adapted to input a B-rep model directly.

The proposed learning-based shape simplification method can be used in the following fields. First, it can be used in collaborative projects that share CAD models [81,82]. Second, it can be used when generating virtual models from digital twins [83]. Third, it can be used for 3D CAD models that are used on mobile devices. In addition, it can also be used in fields such as manufacturing simulations, design reviews, visualization, and virtual training where it is necessary to process 3D CAD models.

5. Conclusions

We have proposed a learning-based shape simplification method for 3D CAD models of mechanical parts. The proposed method uses adversarial learning to recognize the simplification target shape in the original model and removes the recognized shape. To implement the network, we defined a model architecture and an objective function. Then we constructed a 3D CAD model dataset for training the network. Finally, the effectiveness of the proposed network was verified through 3D voxel simplification experiments. The experiment results showed that the proposed method exhibited excellent performance in comparison to other networks, with an average error rate of 3.38% for the part total area and an average error rate of 14.61% for the simplification target area.

In this study, we confirmed the feasibility of a learning-based shape simplification method. To the best of our knowledge, the proposed method is the first study to use a GAN-based network for shape simplification of 3D CAD models. However, the error rate for the simplification target area is still fairly high at 14.61%. Therefore, simplification performance will be improved in the future by segmenting and separately labeling voxels that correspond to the simplification target shape and developing a network that uses the proposed simplification method.

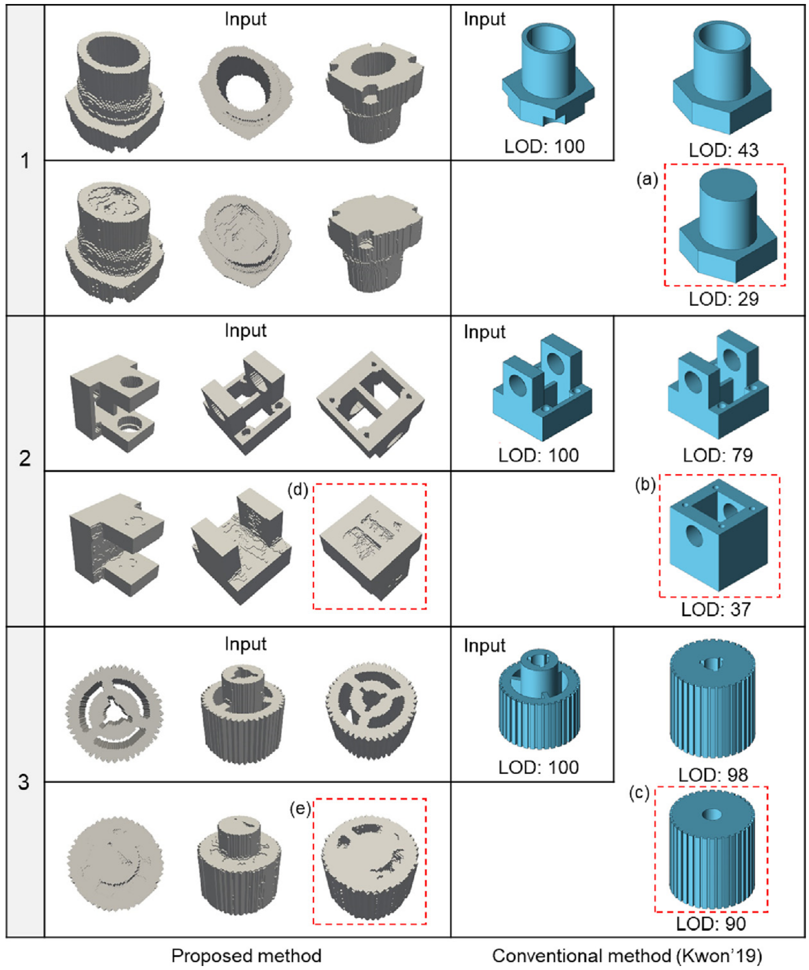

Fig. 10. Results of 3D voxel simplification using the proposed method (left) and conventional method (right).

In the future, it may be possible to increase the detail of 3D models by using training dataset consisting of pairs of original model and simplified model. If a deep learning network for model detailing can be constructed based on our training dataset, it will be more useful in terms of flexibly in adjusting the model detail. The limitation of our research is that the 3D model generated from the network is in voxel form. One solution to this issue is to convert the voxel model to a mesh and then convert this model back to the b-rep model [84]. In the future, as another solution, we plan to use a graph neural network to use b-rep models directly.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Duhwan Mun reports financial support was provided by National Research Foundation of Korea. Duhwan Mun reports financial support was provided by Korea Ministry of Trade Industry and Energy. Duhwan Mun reports financial support was provided by Korea Ministry of Science and ICT.

Data availability

The datasets in this study were constructed by modifying original 3D CAD models from ESB [73] and DSB [74] dataset. ESB dataset is available at https://engineering.purdue.edu/cdesign/wp/downloads/. DSB dataset is available from the corresponding author [74] on reasonable request.

Funding

This research was supported by the Basic Science Research Program [grant number NRF-2022R1A2C2005879] through the National Research Foundation of Korea (NRF) funded by the Korean government (MSIT), by the Carbon Reduction Model Linked Digital Engineering Design Technology Development Program [grant number RS-2022-00143813] funded by the Korean government (MOTIE), and by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korean government (MSIT) [grant numbers 2022-0-00431, 2022-0-00969].

References

[1] Kim BC, Mun D. Enhanced volume decomposition minimizing overlapping volumes for the recognition of design features. J Mech Sci Technol 2015;29:5289–98. http://dx.doi.org/10.1007/s12206-015-1131-9.

[2] Kwon S, Kim BC, Mun D, Han S. Simplification of feature-based 3D CAD assembly data of ship and offshore equipment using quantitative evaluation metrics. Comput Aided Des 2015;59:140–54. http://dx.doi.org/10.1016/j.cad.2014.03.003.

[3] Kwon S, Kim BC, Mun D, Han S. Graph-based simplification of feature-based three-dimensional computer-aided design models for preserving connectivity. J Comput Inf Sci Eng 2015;15. http://dx.doi.org/10.1115/1.4030748.

[4] Kwon S, Mun D, Kim BC, Han S. Feature shape complexity: a new criterion for the simplification of feature-based 3D CAD models. Int J Adv Manuf Technol 2017;88:1831–43. http://dx.doi.org/10.1007/s00170-016-8937-1.

[5] Kwon S, Kim H, Mun D. Multiobjective evolutionary optimization for feature-based simplification of 3D boundary representation models. Int J Adv Manuf Technol 2020;110:2603–18. http://dx.doi.org/10.1007/s00170-020-06004-3.
[6] Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, et al. Generative adversarial nets. Adv Neural Inf Process Syst 2014;27.
[7] Jayaraman PK, Lambourne JG, Desai N, Willis KD, Sanghi A, Morris NJ. Solidgen: An autoregressive model for direct b-rep synthesis. 2022, arXiv preprint arXiv:2203.13944.
[8] Jayaraman PK, Sanghi A, Lambourne JG, Willis KDD, Davies T, Shayani H, Morris N. UV-Net: Learning from boundary representations. In: Proc Cvpr IEEE. 2021, p. 11698–707. http://dx.doi.org/10.1109/Cvpr46437.2021.01153.
[9] Xu YS, Tong XH, Stilla U. Voxel-based representation of 3D point clouds: Methods, applications, and its potential use in the construction industry. Autom Constr 2021;126:103675. http://dx.doi.org/10.1016/j.autcon.2021.103675.
[10] Hilbig A, Vogt L, Holtzhausen S, Paetzold K. Enhancing 3D-CNN-based geometric feature recognition for adaptive additive manufacturing: A SDF data approach. J Comput Des Eng 2023;qwad027.
[11] Kingma DP, Welling M. Auto-encoding variational bayes. 2013, arXiv preprint arXiv:1312.6114.
[12] Arjovsky M, Chintala S, Bottou L. Wasserstein generative adversarial networks. In: International conference on machine learning. PMLR; 2017, p. 214–23.
[13] Hoppe H. Progressive meshes. In: Proceedings of the 23rd annual conference on computer graphics and interactive techniques. 1996, p. 99–108.
[14] Veron P, Léon J-C. Shape preserving polyhedral simplification with bounded error. Comput Graph 1998;22:565–85. http://dx.doi.org/10.1016/S0097-8493(98)00063-6.
[15] Sheffer A. Model simplification for meshing using face clustering. Comput Aided Des 2001;33:925–34. http://dx.doi.org/10.1016/S0010-4485(00)00116-0.
[16] Liang YQ, He FZ, Zeng XT. 3D mesh simplification with feature preservation based on whale optimization algorithm and differential evolution. Integr Comput-Aided Eng 2020;27:417–35. http://dx.doi.org/10.3233/Ica-200641.
[17] Koo S, Lee K. Wrap-around operation to make multi-resolution model of part and assembly. Comput Graph 2002;26:687–700. http://dx.doi.org/10.1016/S0097-8493(02)00124-3.
[18] Zhu H, Menq C-H. B-rep model simplification by automatic fillet/round suppressing for efficient automatic feature recognition. Comput Aided Des 2002;34:109–23. http://dx.doi.org/10.1016/S0010-4485(01)00056-2.
[19] Seo J, Song Y, Kim S, Lee K, Choi Y, Chae S. Wrap-around operation for multi-resolution CAD model. Comput-Aided Des Appl 2005;2:67–76. http://dx.doi.org/10.1080/16864360.2005.10738354.
[20] Foucault G, Cuillière J-C, François V, Léon J-C, Maranzana R. Adaptation of CAD model topology for finite element analysis. Comput Aided Des 2008;40:176–96. http://dx.doi.org/10.1016/j.cad.2007.10.009.
[21] Lee SH, Lee K. Simultaneous and incremental feature-based multiresolution modeling with feature operations in part design. Comput Aided Des 2012;44:457–83. http://dx.doi.org/10.1016/j.cad.2011.12.005.
[22] Kang Y, Kim BC, Mun D, Han S. Method to simplify ship outfitting and offshore plant equipment three-dimensional (3-d) computer-aided design (CAD) data for construction of an equipment catalog. J Mar Sci Tech-Japan 2014;19:185–96. http://dx.doi.org/10.1007/s00773-013-0239-9.
[23] Kwon S, Mun D, Kim BC, Han S, Suh HW. B-rep model simplification using selective and iterative volume decomposition to obtain finer multi-resolution models. Comput Aided Des 2019;112:23–34. http://dx.doi.org/10.1016/j.cad.2019.03.003.
[24] Su H, Maji S, Kalogerakis E, Learned-Miller E. Multi-view convolutional neural networks for 3D shape recognition. In: IEEE I conf comp vis. 2015, p. 945–53. http://dx.doi.org/10.1109/Iccv.2015.114.
[25] Wang C, Pelillo M, Siddiqi K. Dominant set clustering and pooling for multi-view 3d object recognition. In: Proceedings of British machine vision conference. 2017.
[26] Yu LQ, Li XZ, Fu CW, Cohen-Or D, Heng PA. PU-Net: Point cloud upsampling network. In: Proc Cvpr IEEE. 2018, p. 2790–9. http://dx.doi.org/10.1109/Cvpr.2018.00295.
[27] Maturana D, Scherer S. Voxnet: A 3d convolutional neural network for real-time object recognition. In: 2015 IEEE/RSJ international conference on intelligent robots and systems. IEEE; 2015, p. 922–8.
[28] Riegler G, Ulusoy AO, Geiger A. OctNet: Learning deep 3D representations at high resolutions. In: Proc Cvpr IEEE. 2017, p. 6620–9. http://dx.doi.org/10.1109/Cvpr.2017.701.
[29] Wang PS, Liu Y, Guo YX, Sun CY, Tong X. O-CNN: Octree-based convolutional. ACM Trans Graph 2017;36. http://dx.doi.org/10.1145/3072959.3073608.
[30] Wu Z, Song S, Khosla A, Yu F, Zhang L, Tang X, et al. 3D shapenets: A deep representation for volumetric shapes. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2015, p. 1912–20.
[31] Joseph-Rivlin M, Zvirin A, Kimmel R. Momen (e) t: Flavor the moments in learning to classify shapes. In: Proceedings of the IEEE/CVF international conference on computer vision workshops. 2019, p. 4085–94.
[32] Qi CR, Yi L, Su H, Guibas LJ. PointNet plus plus : Deep hierarchical feature learning on point sets in a metric space. Adv Neural Inf 2017;30.
[33] Zaheer M, Kottur S, Ravanbakhsh S, Poczos B, Salakhutdinov RR, Smola AJ. Deep sets. Adv Neural Inf Process Syst 2017;30.
[34] Huang Q, Wang W, Neumann U. Recurrent slice networks for 3d segmentation of point clouds. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2018, p. 2626–35.
[35] Klokov R, Lempitsky V. Escape from cells: Deep kd-networks for the recognition of 3d point cloud models. In: Proceedings of the IEEE international conference on computer vision. 2017, p. 863–72.
[36] Zhang DJ, He FZ, Tu ZG, Zou L, Chen YL. Pointwise geometric and semantic learning network on 3D point clouds. Integr Comput-Aided Eng 2020;27:57–75. http://dx.doi.org/10.3233/Ica-190608.
[37] Kalogerakis E, Averkiou M, Maji S, Chaudhuri S. 3D shape segmentation with projective convolutional networks. In: Proc Cvpr IEEE. 2017, p. 6630–9. http://dx.doi.org/10.1109/Cvpr.2017.702.
[38] Wang PY, Gan Y, Shui PP, Yu FG, Zhang Y, Chen SL, et al. 3D shape segmentation via shape fully convolutional networks. Comput Graph-Uk 2018;70:128–39. http://dx.doi.org/10.1016/j.cag.2017.07.030.
[39] Yi L, Su H, Guo XW, Guibas L. SyncSpecCNN: Synchronized spectral CNN for 3D shape segmentation. In: Proc Cvpr IEEE. 2017, p. 6584–92. http://dx.doi.org/10.1109/Cvpr.2017.697.
[40] Mescheder L, Oechsle M, Niemeyer M, Nowozin S, Geiger A. Occupancy networks: Learning 3D reconstruction in function space. In: Proc Cvpr IEEE. 2019, p. 4455–65. http://dx.doi.org/10.1109/Cvpr.2019.00459.
[41] Brock A, Lim T, Ritchie JM, Weston N. Generative and discriminative voxel modeling with convolutional neural networks. In: Advances in neural information processing systems, workshop on 3D learning. 2016.
[42] Dai A, Qi CR, Niessner M. Completion using 3D-encoder-predictor CNNs and shape synthesis. In: Proc Cvpr IEEE. 2017, p. 6545–54. http://dx.doi.org/10.1109/Cvpr.2017.693.
[43] Liu JC, Mills S, McCane B. Variational autoencoder for 3D voxel compression. In: Int conf imag vis. 2020, http://dx.doi.org/10.1109/IVCNZ51579.2020.9290656.
[44] Sharma A, Grau O, Fritz M. VConv-DAE: Deep volumetric shape learning without object labels. Lecture Notes in Comput Sci 2016;9915:236–50. http://dx.doi.org/10.1007/978-3-319-49409-8_20.
[45] Wu JJ, Zhang CK, Xue TF, Freeman WT, Tenenbaum JB. Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling. Adv Neural Inf 2016;29.
[46] Achlioptas P, Diamanti O, Mitliagkas I, Guibas L. Learning representations and generative models for 3D point clouds. In: Pr mach learn res, vol. 80. 2018.
[47] Fan HQ, Su H, Guibas L. A point set generation network for 3D object reconstruction from a single image. In: Proc Cvpr IEEE. 2017, p. 2463–71. http://dx.doi.org/10.1109/Cvpr.2017.264.
[48] Song J, Lee J, Ko K, Kim W-D, Kang T-W, Kim J-Y, et al. Unorganized point classification for robust NURBS surface reconstruction using a point-based neural network. J Comput Des Eng 2021;8:392–408. http://dx.doi.org/10.1093/jcde/qwaa086.
[49] Ranjan A, Bolkart T, Sanyal S, Black MJ. Generating 3D faces using convolutional mesh autoencoders. Lecture Notes in Comput Sci 2018;11207:725–41. http://dx.doi.org/10.1007/978-3-030-01219-9_43.
[50] Wang NY, Zhang YD, Li ZW, Fu YW, Liu W, Jiang YG. Pixel2Mesh: Generating 3D mesh models from single RGB images. Lecture Notes in Comput Sci 2018;11215:55–71. http://dx.doi.org/10.1007/978-3-030-01252-6_4.
[51] Li HS, Zheng YP, Wu XQ, Cai Q. 3D model generation and reconstruction using conditional generative adversarial network. Int J Comput Int Syst 2019;12:697–705. http://dx.doi.org/10.2991/ijcis.d.190617.001.
[52] Yu YK, Huang ZT, Li F, Zhang HD, Le XY. Point encoder GAN: A deep learning model for 3D point cloud inpainting. Neurocomputing 2020;384:192–9. http://dx.doi.org/10.1016/j.neucom.2019.12.032.
[53] Liu XJ, Zhou SB, Wu S, Tan D, Yao R. 3D visualization model construction based on generative adversarial networks. Peerj Comput Sci 2022;8. http://dx.doi.org/10.7717/peerj-cs.768.
[54] Yang CJ, Lin CL, Wang CK, Wang JY, Chen CC, Su FC, et al. Generative adversarial network (GAN) for automatic reconstruction of the 3D spine structure by using simulated bi-planar X-ray images. Diagnostics 2022;12. http://dx.doi.org/10.3390/diagnostics12051121.
[55] Potamias RA, Bouritsas G, Zafeiriou S. Revisiting point cloud simplification: A learnable feature preserving approach. Lecture Notes in Comput Sci 2022;13662:586–603. http://dx.doi.org/10.1007/978-3-031-20086-1_34.
[56] Potamias RA, Ploumpis S, Zafeiriou S. Neural mesh simplification. In: Proc Cvpr IEEE. 2022, p. 18562–71. http://dx.doi.org/10.1109/Cvpr52688.2022.01803.
[57] Kim T, Cha M, Kim H, Lee JK, Kim J. Learning to discover cross-domain relations with generative adversarial networks. In: International conference on machine learning. PMLR; 2017, p. 1857–65.

[58] Isola P, Zhu J-Y, Zhou T, Efros AA. Image-to-image translation with conditional adversarial networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2017, p. 1125–34.
[59] Choi Y, Choi M, Kim M, Ha J-W, Kim S, Choo J. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2018, p. 8789–97.
[60] Liu MY, Breuel T, Kautz J. Unsupervised image-to-image translation networks. Adv Neural Inf 2017;30.
[61] Zhu J-Y, Park T, Isola P, Efros AA. Unpaired image-to-image translation using cycle-consistent adversarial networks. In: Proceedings of the IEEE international conference on computer vision. 2017, p. 2223–32.
[62] Liu MY, Tuzel O. Coupled generative adversarial networks. Adv Neural Inf 2016;29.
[63] Wang W, Huang Q, You S, Yang C, Neumann U. Shape inpainting using 3d generative adversarial network and recurrent convolutional networks. In: Proceedings of the IEEE international conference on computer vision. 2017, p. 2298–306.
[64] Mirza M, Osindero S. Conditional generative adversarial nets. 2014, arXiv preprint arXiv:1411.1784.
[65] Larsen ABL, Sønderby SK, Larochelle H, Winther O. Autoencoding beyond pixels using a learned similarity metric. In: International conference on machine learning. PMLR; 2016, p. 1558–66.
[66] Makhzani A, Shlens J, Jaitly N, Goodfellow I, Frey B. Adversarial autoencoders. In: International conference on learning representations. 2016.
[67] Ding B, Fan YF, He YJ, Zhao J. Enhanced VAEGAN: a zero-shot image classification method. Appl Intell 2022. http://dx.doi.org/10.1007/s10489-022-03869-7.
[68] Xian YQ, Sharma S, Schiele B, Akata Z. f-VAEGAN-D2: A feature generating framework for any-shot learning. In: Proc Cvpr IEEE. 2019, p. 10267–76. http://dx.doi.org/10.1109/Cvpr.2019.01052.
[69] Kingkan C, Hashimoto K. Generating mesh-based shapes from learned latent spaces of point clouds with VAE-GAN. In: Int C patt recog. 2018, p. 308–13. http://dx.doi.org/10.1109/ICPR.2018.8546232.
[70] Yang S, Liu JY, Wang WJ, Guo ZM. TET-GAN: Text effects transfer via stylization and destylization. In: AAAI conf artif inte. 2019, p. 1238–45.
[71] Lee H, Lee J, Kim H, Mun D. Dataset and method for deep learning-based reconstruction of 3D CAD models containing machining features for mechanical parts. J Comput Des Eng 2022;9:114–27. http://dx.doi.org/10.1093/jcde/qwab072.
[72] Ronneberger O, Fischer P, Brox T. U-net: Convolutional networks for biomedical image segmentation. In: International conference on medical image computing and computer-assisted intervention. Springer; 2015, p. 234–41.
[73] Jayanti S, Kalyanaraman Y, Iyer N, Ramani K. Developing an engineering shape benchmark for CAD models. Comput Aided Des 2006;38:939–53. http://dx.doi.org/10.1016/j.cad.2006.06.007.
[74] Bespalov D, Ip CY, Regli WC, Shaffer J. Benchmarking CAD search techniques. In: Proceedings of the 2005 ACM symposium on solid and physical modeling. 2005, p. 275–86.
[75] HDFGroup. Hierarchical data format version 5. in. 1997.
[76] Kingma DP, Ba J. Adam: A method for stochastic optimization. 2014, arXiv preprint arXiv:1412.6980.
[77] Manda B, Bhaskare P, Muthuganapathy R. A convolutional neural network approach to the classification of engineering models. IEEE Access 2021;9:22711–23. http://dx.doi.org/10.1109/ACCESS.2021.3055826.
[78] Kim S, g. Chi H, Hu X, Huang Q, Ramani K. A large-scale annotated mechanical components benchmark for classification and retrieval tasks with deep neural networks. In: European conference on computer vision. Springer; 2020, p. 175–91.
[79] Willis KDD, Pu YW, Luo JL, Chu H, Du T, Lambourne JG, Solar-Lezama A, Matusik W. Fusion 360 gallery: A dataset and environment for programmatic CAD construction from human design sequences. ACM Trans Graph 2021;40. http://dx.doi.org/10.1145/3450626.3459818.
[80] Rumelhart DE. Learning internal representations by error propagation, in parallel distributed processing. In: Explorations in the microstructure of cognition. 1986, p. 318–62.
[81] Kim BC, Mun D, Han S. Retrieval of CAD model data based on web services for collaborative product development in a distributed environment. Int J Adv Manuf Technol 2010;50:1085–99. http://dx.doi.org/10.1007/s00170-010-2571-0.
[82] Kim BC, Lee H, Mun D, Han S. Lifecycle management of component catalogs based on a neutral model to support seamless integration with plant 3D design. J Comput Des Eng 2021;8:409–27. http://dx.doi.org/10.1093/jcde/qwaa087.
[83] Lee H, Noh C, Kim S, Kim BC, Park J, Mun D, et al. Neutral model-based interfacing of 3D design to support collaborative project management in the process plant industry. J Comput Des Eng 2021;8:824–35. http://dx.doi.org/10.1093/jcde/qwab017.
[84] Andujar C, Brunet P, Chica A, Navazo I, Vinacua A. Solid modelling for manufacturing: From Voelcker's boundary evaluation to discrete paradigms. Comput Aided Des 2022;152:103370. http://dx.doi.org/10.1016/j.cad.2022.103370.