Received: 7 December 2020 | Revised: 6 February 2021 | Accepted: 10 February 2021

IET Image Processing

DOI: 10.1049/ipr2.12146

ORIGINAL RESEARCH PAPER

Constructing an efficient and adaptive learning model for 3D object generation

Jiwei Hu[1] | Wupeng Deng[1] | Quan Liu[1] | Kin-Man Lam[2] | Ping Lou[1]

[1] School of Information Engineering, Wuhan University of Technology, Wuhan, China

[2] Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hong Kong

Correspondence
Quan Liu, School of Information Engineering, Wuhan University of Technology, Wuhan, China.
Email: quanliu@whut.edu.cn

Funding information
National Natural Science Foundation of China, Grant/Award Number: 52075404; Application Foundation Frontier Special Project of Wuhan Science and Technology Bureau, Grant/Award Number: 2020010601012176

Abstract
Studying representation learning and generative modelling has been at the core of the 3D learning domain. By leveraging the generative adversarial networks and convolutional neural networks for point-cloud representations, we propose a novel framework, which can directly generate 3D objects represented by point clouds. The novelties of the proposed method are threefold. First, the generative adversarial networks are applied to 3D object generation in the point-cloud space, where the model learns object representation from point clouds independently. In this work, we propose a 3D spatial transformer network, and integrate it into a generation model, whose ability for extracting and reconstructing features for 3D objects can be improved. Second, a point-wise approach is developed to reduce the computational complexity of the proposed network. Third, an evaluation system is proposed to measure the performance of our model by employing various categories and methods, and the error, considered as the difference between synthesized objects and raw objects are quantitatively compared, is less than 2.8%. Extensive experiments on benchmark dataset show that this method has a strong ability to generate 3D objects in the point-cloud space, and the synthesized objects have slight differences with man-made 3D objects.

1 | INTRODUCTION

Learning representation of real-world objects is an extremely essential area of research. Real-world objects are stereoscopic, where the depth information is also necessary. 2D representation only reflect the projection of an object onto a 2D plane, which leads to the loss of depth information. Thus, 2D representation cannot completely represent the real-world objects. Nowadays, 3D representation of real-world objects has become a core concept for vision, robotics, augmented reality and virtual reality applications, for its powerful geometric representation ability [1]. In computer vision and graphics, 3D object modelling can take on various forms of representation [2–4]. Hitherto, most researchers choose the 3D voxel grids or a collection of images to represent 3D object. This data representation results in unnecessary voluminousness [5, 6]. Therefore, in this paper, we consider the representation of 3D objects, in the form of point clouds.

Point clouds are popular for their vectored data representation, as well as their compact encoding of the shape information, optionally embedded with texture. Moreover, point clouds are simple and have a unified structure that avoid the combinatorial irregularities and complexities of meshes, and these make them easier to be learned. Point clouds have been widely used in autonomous driving, autonomous disassembly and other practical tasks. However, the acquisition of point clouds has been a complex and difficult problem, which limits the application of data-driven approach (e.g. deep learning) in practical tasks. There are two main ways to obtain point clouds. One way is to capture the data with range-scanning devices, like Lidar or Kinect [7–10]. However, the point clouds captured by range-scanning devices are often polluted by noise and unable to be recognized. Another way is to reconstruct point clouds by man-made, which can ensure that the data is clean, just like the famous ModelNet40 [11, 12] data set (CAD Model). However, this method consumes plenty of human resources.

In order to solve the point-cloud acquisition problem, many researchers have made impressive progress on the design of generative models via traditional statistical methods [13–15]. Many of these methods synthesize 3D objects by utilizing parts

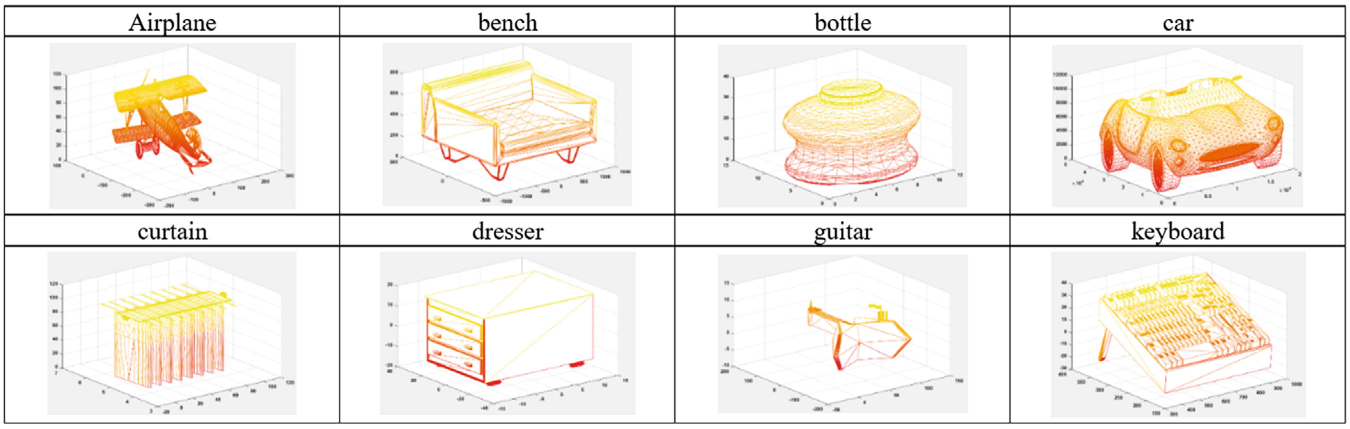

FIGURE 1 Examples of point-cloud models in the ModelNet40 data set

of the objects in existing point-cloud data sets, which requires a great deal of prior knowledge of a certain category. Therefore, these classical methods look theoretically feasible, but not realistic.

Recently, deep learning has been found to be a promising data-driven approach for objects generation. In the application where amount of data is available, deep learning can be employed to achieve better accuracy, compared with those traditional statistical methods. Different from the traditional methods, most of the deep learning approaches do not explicitly model the concept of parts or retrieve a complete object from an object repository, instead they synthesize new objects based on learned object representation. Pioneering deep representations and generative learning models are successful at learning data representation and generating realistic sample from complex underlying distributions, like Auto Encoders (AEs) [16] and Generative Adversarial Networks (GANs) [17, 18]. Both these pioneering methods focus on how to represent and generate 2D data, regardless of the characteristics of the 3D data. 3D data modelling is still a challenging research, and it is much more difficult to reproduce the 3D shape of an object because of higher dimensionality. With the development of deep learning and the availability of benchmark data sets, like the ModelNet40 data set [11, 12], there have been some inspiring attempts in deep generative models based on volumetric space representation [19, 20]. Some of the methods used GANs for 3D point-cloud reconstruction, which achieved promising performance. In [21], a novel framework, namely 3D-GAN, was proposed, which generates 3D objects from a probabilistic space by leveraging recent advances in volumetric convolution networks and GANs. Yang et al. [22] proposed 3D-RecGAN, which reconstructs a complete 3D structure in a high-dimensional voxel space, using GANs. Both these approaches reconstruct 3D objects via a volumetric space representation, where the detailed information is unavailable. However, they all provide a novel approach that GAN can be applied to reconstruct or generate 3D objects independently, which is an inspiration for our research.

In this paper, we aim to tackle the problem of generating 3D objects in the point-cloud space from complex underlying distributions and discriminating 3D object models based on point clouds directly. We propose a novel framework, which consists of a point-cloud generating model and a point-cloud recognition model. Our approach combines the merits of GANs and the point-cloud classification network. In our model, the deconvolution network and spatial transform network [23] are included in the generative model to improve its ability of data generation. Our framework is composed of two parts. The first part is a deep generative network that aims to generate the same 3D point-cloud data as the original 3D point-cloud data. The second part is a deep discriminative network that aims to distinguish between the original 3D point-cloud data and the generated 3D point-cloud data. Based on the proposed network, we design a complete generation flow of 3D objects in the point-cloud space. All of our research subjects come from ModelNet40 data set, as shown in Figure 1.

The main contributions of our research are listed as follows:

First, we utilize the benefits of generative adversarial network and point-cloud recognition model, and propose an effective generalized network to generate 3D object in the point-cloud space directly. Futhermore, we integrate the spatial transformer network with the proposed network to improve the ability of the generation model to extract and reconstruct features for 3D objects in the point-cloud space, where the generation model can take point clouds as input directly. Comprehensive experiments were conducted to evaluate the contributionsof the spatial transformer network to 3D object generation in detail.

Second, owing to the large number of parameters in the proposed network and the limited computing power, the proposed network can only synthesize part of the 3D objects in the point-cloud space. The problem caused by limited computing power often arises, and researchers have provided some methods to solve it, such as using the patch-wise method. In our research, we develop a point-wise approach based on patch-wise, which divides point clouds into several groups and separately trains the groups. By using our approach, a complete 3D object in the point-cloud space can be generated, and the differences between the generated objects and the corresponding raw objects are slight.

Third, we propose a stepwise evaluation method for object generation in the point-cloud space. The first part of this evaluation method compares the performance between different categories, which can assess the generalization of the method; the second part is utilized to evaluate the performance between different methods, regardless of the influence of 3D object representation; the final part measures the difference between the synthesized objects and the corresponding raw objects.

The remainder of this paper is organized as follows. Section 2 gives an overview of some related works. Section 3 describes the architectures of our proposed generation and discrimination networks, the details of the proposed method, and the proposed evaluation method. In Section 4, we present the experiments used to evaluate our model and to compare it to other methods quantitatively and qualitatively. Finally, we give a conclusion in Section 5.

2 | RELATED WORK

2.1 | Deep learning on 3D data

A point cloud represents a 3D object as a set of 3D locations of its surface in the Euclidean coordinate space. From the data-structure viewpoint, a point cloud is an unordered set of vectors, which cannot be applied to the conventional convolutional architecture. However, point clouds have their own merits. They are simple and have unified structures that avoid the combinatorial irregularities and the complexities of meshes for representation, so they are easier to be learned. Meanwhile, point clouds can represent real-life objects in detail, which maximizes the accuracy of data recovery.

There has been a large amount of extended works using deep learning on 3D data for various tasks. In the field of 3D object recognition, [12, 24, 25] represent 3D object models based on the voxelized shapes and apply 3D convolutional neural networks (CNNs) for recognizing 3D objects. However, in the course of space transformation, i.e., from the point-cloud space to the volumetric space, detailed information is lost, which restricts the classification accuracy. Li et al. [26] proposed a special network, namely FPNN, to deal with the data sparsity problem. However, the processing ability of FPNN is still limited owing to the huge size of 3D object in the point-cloud space. In addition to volumetric representation, many researchers focus their attention on the multi-view representation of 3D object. In [27–30], 3D objects are rendered into 2D images, and 2D CNNs are applied for classification. These methods make full use of the merits of existing 2D CNNs. Nevertheless, in the rendering process, information integrity cannot be guaranteed, which limits the development of 3D object recognition. Compared with the volumetric representation and multi-view representation, the point-cloud representation contains more detailed information about 3D object, and it has been widely used in many practical tasks, such as VR and AR. With the advances in 3D deep learning and the enrichment of point clouds, an efficiency network, namely PointNet [7, 8], was developed. PointNet, with point clouds as its input, was designed end to end for 3D object classification, which takes the permutation invariance of the input data points into consideration. With a lack of consideration of local information, the network does not perform well in some categories (e.g. flower_pot), as tested in the ModelNet data set. However, PointNet can still achieve high accuracy. This proves that the method of recognizing 3D objects based on point clouds is feasible, and provide a novel framework to recognize 3D object in the point-cloud space directly and independently.

Based on the discussion of related research on 3D deep learning, the capability of 3D object representation cannot be replaced, and it has been widely used in practical task, where the research of deep learning on point-cloud representation appears more important.

2.2 | 3D object generation

In the field of 3D object generation, existing methods can be divided into two categories: traditional methods and deep learning methods. Most of the traditional works [31, 32] use assemble-based methods to build part-based object models. Zou et al. [33] explored an example-based approach to reconstruct 3D object from a single-view line drawing. Dou et al. [34] presented a coupled-dictionary model for parametric facial shape representation and a two-stage framework for 3D face reconstruction from a single 2D image by using facial landmarks. However, these methods can only reconstruct objects of a specific class with small variations and are computationally intensive, which hinders their use in real-world appliactions. More importantly, these methods ignore the diversity of 3D object shapes from different views. The main reason of the shortcomings above is that the feature extraction and reconstruction during object generation are driven by manmade, and the stability cannot be ensured. Chang et al. [11] proposed a convolutional deep belief network, namely ShapeNet, to represent geometric 3D shapes in the 3D voxel grid. Shi et al. [35] introduced a robust representation of 3D shapes, named DeepPano, which is learned with deep convolutional neural networks on the panoramic view. Qi et al. [27] also applied multi-view CNNs for 3D reconstruction. Pang et al. [36] presented a generic unsupervised method to increase the discriminative power of image vectors obtained from a broad family of deep neural networks for object retrieval. Konstantinos et al. [37] utilized convolutional neural networks to extract features from 2D panoramic view representation of 3D models. In addition, the research of 3D object generation also contains task of multi-object generation. During this kinds of research, traditional methods cannot realize satisfied performance, owing to the huge amount of modelling task. For example, [38–40] researched the reconstruction of smart city based on WebVRGIS, where they proposed peer-to-peer network to achieve 3D visualization. The peer-to-peer network can be also recognized as one of the deep learning methods. All of these methods have two common shortcomings. First, they all use CNNs to extract object features, which are then used to generate the 3D shapes. However, during the process of feature extraction, the 3D object information is lost. As a result, the generated object cannot represent original object accurately. Second, as we have introduced in Section 2.2, the representation methods utilized above still limit the generation accuracy, which cannot meet the requirement for real-world applications.

2.3 | Generative adversarial network

With the advances in CNNs and the gradual improvement of GANs, now, more research is moving to use GANs for 3D object generation and reconstruction. Goodfellow et al. [17] proposed GAN, which incorporates an adversarial discriminator into the procedure of generating objects. The framework has two models: a generative model G that captures the data distribution, and a discriminative model D that estimates the probability that a sample comes from the training data, rather than generated by G. Subsequent studies have witnessed the effectiveness of GAN for different tasks. Recently, more improved versions of GAN have been proposed. Radford et al. [18] introduced a class of CNN, called deep convolutional generative adversarial networks (DCGAN), with the use of deconvolution nets. Experiments demonstrated convincing evidence that DCGAN learns a hierarchy of representations from the object parts of a scene in both the generator and the discriminator. The above GANs and related frameworks are all powerful models in object generation, but they all suffer from training instability, i.e., Nash equilibrium is hard to meet. In order to solve the training problem, Ishaan et al. [41] applied a novel weight-clipping method to GAN, named WGAN, which enables stable training of a wide variety of GAN architecture, with almost no hyper parameter tuning required. Furthermore, Youssef et al. [42] introduced a new family of integral probability metrics for training GAN. These two methods make it feasible to train GANs. All the introduced research in this section above are only performed on 2D object generation, without consideration of 3D object generation. Later, some promising works have employed GAN for 3D generation and reconstruction. Wu et al. [21] proposed a novel framework, named 3D-GAN, which generates 3D objects from a probabilistic space by leveraging the recent advances in volumetric convolutional networks and GANs. Yang et al. [22] proposed an approach, namely 3D-RecGAN, which reconstructs the complete 3D structure of a given object from a single arbitrary depth view using GAN. However, all these methods are constrained by the size of the object, so they require the 3D point clouds to be transformed into the volumetric space, where the detailed information about the 3D object is lost. In the latest studies, researchers began to investigate the 3D object retrieval in the point-cloud space. Panos et al. [43] introduced a deep Auto Encoders (AE) network with state-of-the-art reconstruction and generalization ability, but this method cannot realize 3D recognition tasks without previous shape editing. The autoencoder is also applied in [44], generating high-quality objects. But during the process, the 3D information is transferred into 2D grid information, where the information is lost. Li et al. [45] explored the constrained relationship between point clouds and GANs, and proposed a hierarchical and interpretable sampling process to achieve the direct application of existing GAN algorithm in the point-cloud space. However, the generalization ability still depends heavily on the sampling effect. The main contribution of [46] is that it first uses an reinforcement learning agent to find the correct input to the GAN to generate the best suitable latent space representation of point clouds, but it neglects the defects of the GAN itself in the generation and reconstruction of 3D object in the point-cloud space. Based on the above discussion, a common shortcoming of the GAN-based methods is that they lack the ability to extract and reconstruct feature from 3D objects in the point-cloud space directly.

From all the cited related work, we can draw the following conclusions. First, the representation of 3D objects in the point-cloud space has been widely used in many practical tasks, and the deep learning model on 3D object in the point-cloud space can avoid the influence of information loss. Second, 3D object generation in the point-cloud space driven by deep learning performs better than manmade methods. Third, GANs and related framework provide sufficient evidence that GANs can be applied to generate 3D object. In our research, we explore the method of 3D object generation in the point-cloud space, where the 3D object feature need to be extracted and reconstructed in the point-cloud space directly and independently.

3 | THE ARCHITECTURE AND TRAINING OF PROPOSED NETWORK

3.1 | Overview

In this section, we will introduce the design principle of neural networks and the architecture of our generator and discriminator models in detail.

Considering the advantages of using point clouds to represent 3D real-world objects, we took point clouds from the output of our generator model as the input of our discriminator model. However, GAN cannot extract and reconstruct features from point clouds directly due to three properties. First, different from pixel arrays in 2D image or 3D voxel grids, a point cloud is a set of point coordinates without a specific order. For instance, a point cloud $\binom{P_{11},\ P_{12},\ P_{13}}{\mathcal{Q}_{11},\ \mathcal{Q}_{12},\ \mathcal{Q}_{13}}$ and a point cloud $\binom{\mathcal{Q}_{11},\ \mathcal{Q}_{12},\ \mathcal{Q}_{13}}{P_{11},\ P_{12},\ P_{13}}$, where $(P_{11},\ P_{12},\ P_{13})$ and $(\mathcal{Q}_{11},\ \mathcal{Q}_{12},\ \mathcal{Q}_{13})$ represent two 3D points, which are in a different order, represent the same object. A network, which is input with N 3D data points directly, needs to have the ability to recognize N! point clouds. This greatly increases the computational demand in the generation process and discrimination process for the training samples. Second, data points are not isolated. Neighbouring data points also form local features that represent the geometric characteristic of an object, which requires high capability of local feature extraction and reconstruction of the discrimination and generation network. Third, for real-world objects, spatial transformation, such as rotation and translation, does not change their geometric shapes. This requires the network to perceive the object's spatial information autonomously. From discussion above, a key factor of generating point-cloud model is to improve the network ability of consuminga point clouds directly. Inspired by the point cloud recognition framework in [7] and spatial transformer network [23], where the approximation theory and spatial transformer theory are the main theoretical bases, we propose a novel network that generate 3D object model in the point-cloud space directly. The improved application of two main theories are as follows:

Approximation theory of functions: In order to make our model have the ability to consume 3D point clouds directly, we applied a symmetric function to aggregate the information from all data points. The universality theorem [7] is well known for neural networks, which states that neural networks have a kind of universality. A neural network can approximate any function with a sufficient number of neurons. This universal approximation theory provides the theoretical basis. We defined an approximate function $f(\ldots)$, such that data points can be input directly to a deep network. We used two different functions, $g(\ldots)$ and $h(\ldots)$, to compute the approximate function, i.e.

$$f(\{x_1, \ldots x_n\}) = h(g(x_1), \ldots g(x_n)) \tag{1}$$

where the function $f(\ldots)$ maps n 3D data points to a single value. The $g(\ldots)$ function maps a 3D data point to a single value, and the $h(\ldots)$ function maps the n values to a single value. x_i represents the ith input data point, which contains three coordinates. For the whole function, $g(\ldots)$ realizes feature extraction from 3D point clouds, and $h(\ldots)$ is to perform feature integration. In our designed model, the function of $g(\ldots)$ and $h(\ldots)$ are realized by a set of convolutional layers. This function model outperforms the state-of-the-art methods and the classification performance of our proposed method will be evaluated in the experiment section.

3D spatial transformer network: convolutional neural networks lack the ability to be spatially invariant to the input data in a computational and parameter-efficient manner. Jaderberg et al. [23] proposed a learnable module, named classical spatial transformer network, which allows the spatial manipulation of data within the network. In this paper, we improved the spatial transformer network on 3D point-cloud space and applied it to resolve the spatial alignment problem, caused by the properties of unordering and spatial invariance of 3D data points. In a 3D point cloud, the pointwise transformation is as follows:

$$\begin{bmatrix} x_i^t \\ y_i^t \\ z_i^t \end{bmatrix} = T \begin{bmatrix} x_i^s \\ y_i^s \\ z_i^s \end{bmatrix} = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \\ \theta_{31} & \theta_{32} & \theta_{33} \end{bmatrix} \begin{bmatrix} x_i^s \\ y_i^s \\ z_i^s \end{bmatrix} \tag{2}$$

where $[x_i^t \;\; y_i^t \;\; z_i^t]^T$ and $[x_i^s \;\; y_i^s \;\; z_i^s]^T$ are the target point coordinates and the source point coordinates, respectively, of a point in a 3D point cloud, and T is an affine transformation matrix, which is produced by the localization network. In our transformer network, we only replaced the transformation matrix in the classical spatial transformer network with our proposed matrix T. In the network design, we followed the classical transformer network, and the details of the network can be found in [23]. In the next subsection, we will describe our proposed 3D spatial transformer network in detail.

3.2 | Architecture design

Equation (1) was first used in the design of discrimination model, combined with convolutional neural network, which aimed to extract features from point clouds. Next, in order to improve the ability of consuming point clouds directly, we proposed input alignment model and feature alignment model based on 3D spatial transformer network, and applied them to discrimination model. After the design of discrimination model, we designed generation model based on the mirror discrimination model, which has been proved effective in [21]. While the contribution of 3D spatial transformer network to discrimination model can be seen in [7], we mainly focused on the contribution of 3D spatial transformer network (input alignment model and feature alignment model) to generation model in our experiments. In this paper, four networks were proposed to confirm the contribution of each alignment model: without input alignment and feature alignment (Basic-GAN); with input alignment (Input-GAN); with feature alignment (Feature-GAN); with input alignment and feature alignment (Input & Feature-GAN). The main differences between them are whether proposed network uses alignment model.

Figure 2 shows the basic framework of our proposed network (Basic-GAN). This network consists of a generator (shown in the red frame) and a discriminator (shown in the green frame). The discriminator aims to classify real objects and fake objects synthesized by the generator, and the generator attempts to produce fake objects indistinguishable from real objects. In proposed network, the generator G is used to create point-cloud data to represent a 3D object based on a complex underlying distribution, e.g. Gaussian distribution. The discriminator D outputs the probability of whether the input comes from a real object or a fake object. In the design of our model, two issues are considered. The first one is that the generation process is very difficult, due to the complexity of the 3D data structure. Another issue is that training a GAN is notoriously hard and unstable, where the Nash equilibrium is required to be met. In our method, we employed de-convolutional networks, batch normalization, and the leaky ReLU activation to boost the effectiveness and efficiency in training the model.

In Figure 2, 'Dense (a, b… c)' means a set of fully connected layers, where (a, b… c) represents the size of the respective layers in the fully connected layers. 'Deconv(x, y… z)' means a set of de-convolutional layers, where (x, y… z) represents the height of the output maps (the width of the output maps always equals 1024). 'Conv(x, y… z)' represents a set of convolutional layers, where (x, y… z) also represents the height of the output maps (the width of the output maps always equals 1024). All the layers are followed by the leaky ReLU activation function and a batch normalization layer [47], except for two of the layers that are particularly annotated with the sigmoid activation function. In order to avoid excessive discretization of the generated data, we applied the sigmoid activation function

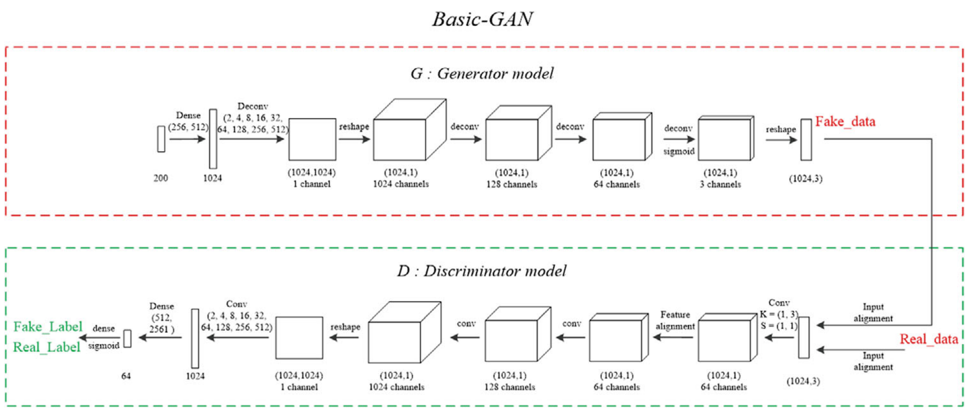

FIGURE 2 The architecture of the proposed network (Basic-GAN)

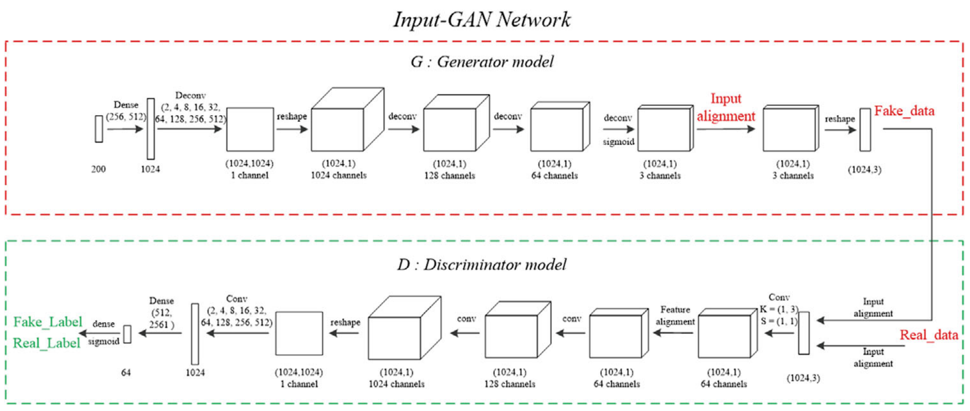

FIGURE 3 The architecture of the proposed network (Input-GAN). Compared with Basic-GAN network, Input-GAN network adds one input alignment on the penultimate layer in generator model. The other settings are all the same as Basic-GAN

to adjust their scope. In the discriminator model, the sigmoid function was used to solve the two-classification problem (Fake-Label or Real-Label). Here, we can also use the softmax function to solve the two-classification problem. However, in order to maintain a mirror image generated by the generator model, which is beneficial to GAN training, we replaced the softmax function with the sigmoid function. In our model, the kernel size and stride size are (1, 1) and (2, 1), respectively. Except for when specially annotated, 'Conv $K = (1, 3)$ $S = (1, 1)$', where Conv represents a convolutional layer, K represents kernel size, and S represents stride size, means that the height and width of the kernel in the convolutional layer are 1 and 3, respectively, and the vertical and horizontal strides in this convolutional layer are both equal to 1. The generator, as shown in the red frame in Figure 2, which aims to synthesize new objects, is composed of three fully connected layers and fifteen deconvolution layers. The input to the generator is a 200D Gaussian noise vector. The output is a series of data points, with a size of m (data capacity). The best result is achieved when the data capacity (m) exceeds the number of data points required to represent an object. However, in considering the huge amount of proposed network and existing computing power, we set m to 1024 in our experiments. The setting of the parameter m is the same as PointNet [7], which has achieved convincing classification performance. In addition, the width of output maps is also the same as m, which represents the number of synthesized points of our proposed network.

The discriminator aims to distinguish whether the 3D point-cloud data is plausible or not. As shown in the green frame in Figure 2, the discriminator mostly mirrors the generator, and was designed as a binary discriminator to classify fake data against real data. The discriminator takes real point-cloud data and synthesized point-cloud data as the input, and outputs the classification label. Based on the label, our discriminator can differentiate the data source. In our experiments, we set label '1′ to represent real data and label '0′ for synthesized data.

Based on the proposed Basic-GAN, we then applied input alignment model and feature alignment model to Generator model in Basic Network, and proposed three networks: Input-GAN (as shown in Figure 3), Feature-GAN (as shown in Figure 4) and Input & Feature-GAN (as shown in Figure 5).

In the four networks above, 'Feature alignment' and 'Input alignment', which are named as 3D alignment networks collectively, are all achieved by using the spatial transformer networks. The 3D alignment networks are designed on the basis of the 3D spatial transform networks, as mentioned above. Here, we use the input alignment network, as shown in Figure 6, as an example to illustrate the architecture. The structure of the feature alignment network is similar to the input

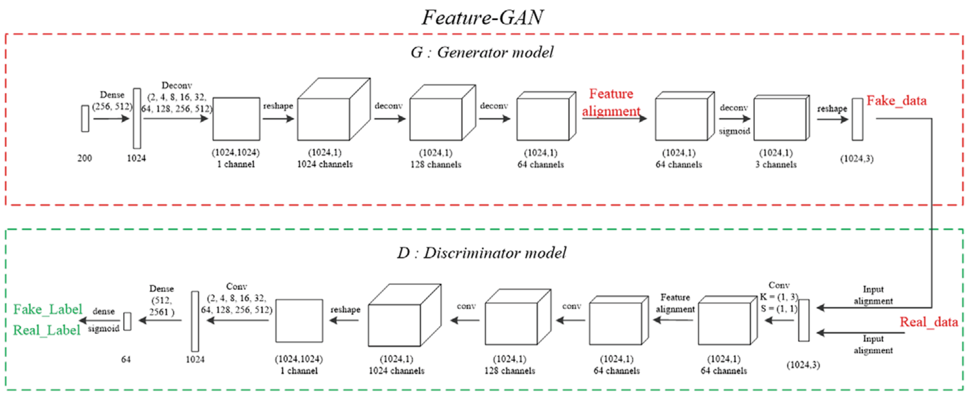

FIGURE 4 The architecture of the proposed network (Feature-GAN). Feature-GAN network adds one feature alignment on the penultimate third layer in generator model of Basic-GAN network. The other settings are all the same as Basic-GAN

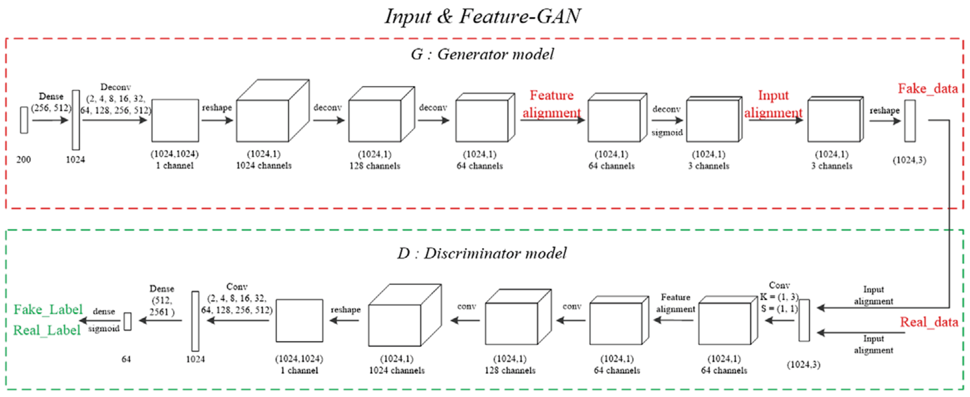

FIGURE 5 The architecture of the proposed network (Input and Feature-GAN). Input and Feature-GAN network adds one input alignment and one feature alignment on the penultimate and penultimate third layer in generator model of Basic-GAN network. The other settings are all the same as Basic-GAN

alignment network. In the feature alignment network, we only changed the size of the transformation matrix T.

As shown in Figure 6, the alignment network mainly consists of three modules: localization network, 3D parameterised sampling network, and 3D sampling network. The localization network, as shown in the red frame, is a regression network, composed of several convolutional layers and fully connected layers, followed by ReLU activation. The function of this module is to calculate $K \times K$ angle values (e.g. θ_{11}, θ_{12}, ...). The 3D parameterised sampling network, shown in the purple frame in Figure 6, uses the $K \times K$ angle values to compute the transformation matrix T, which has been mentioned in Section 3.1. During training, the transformation matrix can be optimized by using a final regression layer in our model. Finally, the aligned data points can be obtained by projecting the original data points into the optimized transformation matrix. The training of alignment network is integrated with the complete network.

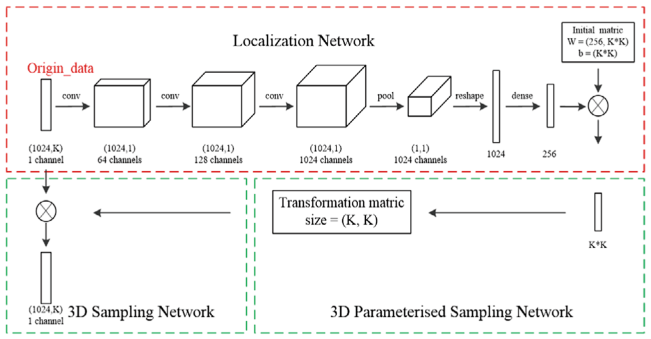
Localization Network
Origin_data
conv
pool
reshape
dense
Initial matric W = (256, K*K) b = (K*K)
Transformation matric size = (K, K)
3D Sampling Network
3D Parameterised Sampling Network

FIGURE 6 The architecture of the alignment network

3.3 | Training details

The training details in four proposed networks are the same as each other. We adapted an end-to-end training procedure for the whole model to synthesized object in the point-cloud space from a 200D vector following a Gaussian distribution over 0 to 1. As we have mentioned previously, GAN belongs to a zero-one game model, where the aim of the G and D modules are opposite. The generator aims to synthesize fake data that cannot be distinguished by the discriminator. Meanwhile, the discriminator was dedicated to classifying real data and fake data. Based on the targets of the G and D modules, the objective function was set as Equation (3). The optimization criteria are to minimize the loss in training the generator and maximize the loss in training the discriminator.

$$Loss = \log D(r) + \log(1 - D(G(z))) \tag{3}$$

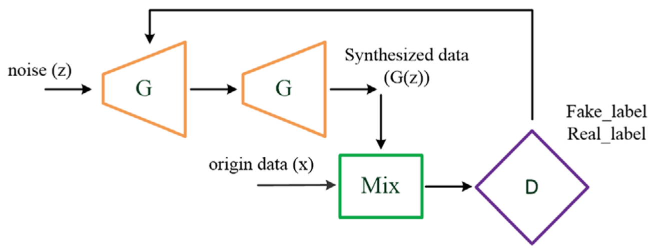

FIGURE 7 The flow diagram for one round of training

where r is a real object in the point-cloud space, and z is a randomly sampled noise vector from a 200D vector following a Gaussian distribution over 0 to 1, and $D()$ and $G()$ represent the output of the generator and discriminator model, respectively. The objective function has two parts with completely opposite training target, which is hard to optimize in a single function. The loss function was separated for G and D, as shown in Equations (4) and (5), respectively. The loss can be optimized in an end-to-end manner for the network, as follows:

$$Loss_G = -\log(D(G(z))) \tag{4}$$

$$Loss_D = -\log(D(r)) + \log(D(G(z))) \tag{5}$$

In training the model, the $Loss_G$ for the G network and the $Loss_D$ for the D network were minimized. Both the generator and the discriminator were trained with the same batch. However, during training, we found that the discriminator usually learns faster than the generator, possibly because generating 3D point-cloud data is more complicated than differentiating between real and synthesized objects, as illustrated in [17, 18, 21]. In order to tackle this problem, we updated G twice, but D only once, in each round of training. The flow diagram is shown in Figure 7.

We set both the learning rate of G and D to 10^{-3}, and used RMSProp [48] for optimization with the rate = 0.99. As the process of 3D point-cloud data generation and the discrimination between fake and real data are very complicated, the model is huge, taking up 10.6G of graphics memory, which limits our batch size to 1.

3.4 | 3D object generation in point-cloud space

In Section 3.3, we have introduced the principle and architecture of our proposed network. However, there are still three straitened circumstances in generating a complete 3D point-cloud object, regardless of the application of 3D alignment networks.

First, in consideration of the distribution of real point-cloud data, it is hard to directly synthesize point clouds with the real distribution. Here, we proposed a new regularization scheme based on the principles of averaging and regularization, as shown in Equations (6)–(8), to preprocess the real data points.

$$x_i^t = \frac{1}{1 + e^{\left(-\left(x_i^s - \frac{1}{m}\sum_{i=1}^{m}(x_i^s)\right)\right)}} \tag{6}$$

$$y_i^t = \frac{1}{1 + e^{\left(-\left(y_i^s - \frac{1}{m}\sum_{i=1}^{m}(y_i^s)\right)\right)}} \tag{7}$$

$$z_i^t = \frac{1}{1 + e^{\left(-\left(z_i^s - \frac{1}{m}\sum_{i=1}^{m}(z_i^s)\right)\right)}} \tag{8}$$

where (x_i^s, y_i^s, z_i^s) are the coordinates of the ith source 3D point, m represents the number of data points, and (x_i^t, y_i^t, z_i^t) are the coordinates of the ith processed data point. In Equations (6)–(8), the processed data point (x_i^t, y_i^t, z_i^t) is generated from the corresponding source data point, by shifting it by the mean of all the source data points, makes each of the coordinates of (x_i^t, y_i^t, z_i^t) have its value between [0, 1]. This can solve the problem, if the data span is too huge, and can help speed up the training.

Second, due to the limitation of the graphics memory, we can only generate m ($m = 1024$) point-cloud data points, which cannot reproduce the real point-cloud data in the ModelNet40 data set, where the size of each point cloud is well over m. In order to tackle this issue, we designed the point-wise approach based on patch-wise and for training. We first randomly divided the real data into 100 groups, with each group containing m points. Based on this point cloud of m real points, we synthesized m fake data points. Finally, we combined these 100 groups of data to form a synthesized point cloud model.

Thirdly, the synthesized data lies between 0 and 1. After obtaining the entire synthesized data, we performed reverse regularization, using Equation (9). As we have described in Equation (6), sigmoid regularization was applied to the input data. The synthesized data can be fully recovered as follows:

$$f_i = \log(s_i) - \log(1 - s_i) \tag{9}$$

where $s_i = (x_i^s, y_i^s, z_i^s)^T$, i.e. the three coordinates of the ith data point synthesized by proposed network, and f_i is the final data point generated by our proposed method. Equation (9) is the reverse process of sigmoid regularization.

The complete flow of generating a 3D point cloud object is composed of data regularization, grouping, proposed network, combination, and data recovery. As shown in Figure 8, the procedure is divided into two modules: training module and inference module. The aim of training module is to train the proposed generative adversarial network (as we have illustrated in Section 3.2), and then the trained network is used in inference module to generate point cloud model. The detailed procedure is as follows:

In training module, the original 3D point cloud data is randomly divided into 100 groups, where each group has m data points. Next, for the m points in each group, regularization is performed, based on Equations (6)–(8), to promote the modelling capability. Then, the normalized data and Gaussian noise

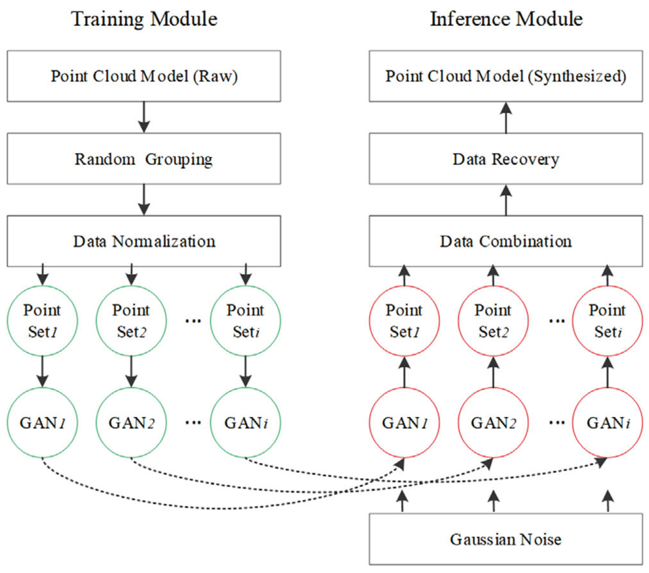

FIGURE 8 The flow chart of point-cloud model generation

are fed to the proposed network to train the proposed generative adversarial network (as we named GAN$_i$ in the green circle of Figure 8). During training, the output of each proposed network is point clouds containing 1024 points (as we named Point Set$_i$ in the green circle of Figure 8). After training, the trained proposed networks are transferred to inference module (as we named Point Set$_i$ in the red circle of Figure 8) respectively to synthesize point clouds (as we named Point Set$_i$ in the red circle of Figure 8), and then the synthesized point clouds from all the groups are combined. Finally, the synthesized point clouds are recovered by using Equation (9). After the above generating procedure, a new 3D point cloud model is synthesized. This can expand the data capacity of the original data set.

3.5 | Evaluation criteria

An important contribution of this paper is that we proposed a new evaluation system, which includes three aspects. The evaluation system can be used to compare proposed networks' performance between different categories, different existing methods, and it can be also applied to measure the morphological difference between synthesized data and raw data.

In the evaluation system, we first proposed a measure, aiming at comparing the performance difference between different categories for one network. In our work, the size of point clouds representing synthesized object is different with the size of point clouds representing raw object, where *JSD* [43] and *D2F* [45] which lack the ability to compare point clouds with different size cannot be used. Here, 95% Hausdorff distance, denoted as $d_{H,r}(R,S)$, was proposed. Following is a brief description of how the 95% Hausdorff distance is calculated:

Step1: Calculate the directed 95% Hausdorff distance from raw data points to synthesized data points, denoted as $\vec{d}_{H,r}(R,S)$, which is the 95th percentile distance over all the distances from every raw data point to the corresponding closest synthesized data points. This distance is calculated as follows:

$$\vec{d}_{H,r}(R,S) = K_r\left(\min_{y\in S} d(x,y)\right), \forall x \in R. \tag{10}$$

Step2: Calculate the directed 95% Hausdorff distance from synthesized data points to raw data points, denoted as $\vec{d}_{H,r}(S,R)$, which is the 95th percentile distance over all distances from every synthesized data point to the corresponding closest raw data points. This distance is calculated as follows:

$$\vec{d}_{H,r}(S,R) = K_r\left(\min_{x\in R} d(y,x)\right), \forall y \in S. \tag{11}$$

In Equations (10) and (11), K_r represents the 95% percentile distance, $d(,)$ represents the Euclidean distance, R represents the set of all the raw data points, and S represents the set of all the synthesized data points.

Step3: Calculate the undirected 95% Hausdorff distance $d_{H,r}(R,S)$, as follows:

$$d_{H,r}(R,S) = \frac{\vec{d}_{H,r}(R,S) + \vec{d}_{H,r}(S,R)}{2} \tag{12}$$

After the comparison on different categories, we took full advantages of the relationship between generation model and discrimination model in GAN, and proposed an unsupervised classification model to evaluate proposed method's ability of generating 3D objects, which can be used to compare with different existing methods. In the process of designing GAN, the relationship between generation model and discrimination model is mutual game, and the target of training GAN is to reach a balance point where generation model has the strongest object generation ability and discrimination model has the strongest object classification ability. Inspired by the principle of GAN, we applied discrimination model of proposed network to extract feature for an input 3D object, and the last three fully connected layers were removed, replaced by a linear SVM for classification. The inspiration of this approach is from [21, 43, 45]. The classification accuracy reflects the generation ability.

Finally, in order to verify the difference between synthesized objects and raw objects when used (e.g. recognition, reconstruction, etc.), we applied the state-of-the-art point cloud recognition models (PointNet [7] and PointNet++ [8]) to confirm proposed networks' ability. The work is as follows: we trained point cloud recognition models on train data of ModelNet40 data set. Next, we tested trained recognition models on test data of ModelNet40 data set and synthesized data, recorded the accuracy of these two methods as well. At the same time, we trained point cloud recognition models on synthesized data and test trained recognition models on test data of ModelNet40 data set.

4 | EXPERIMENTS AND COMPARISON

Experiment set-up, analyses and experiment results are given in this section. First, we explored the contribution of input alignment and feature alignment in proposed network, and determined the most effective network among proposed. Then, we compared the difference based on different values of group. In the next three subsections, we evaluated the performance of proposed network and method in three aspects, as illustrated in Section 3.5.

4.1 | Experiment data

As we have mentioned in the Section 1, all of experiment data come from ModelNet40 data set. The ModelNet40 data set provides a comprehensive collection of clean 3D object models, containing 40 categories of object models, and has become the most popular data set in the field of 3D deep learning. The ModelNet40 data set record 3D spatial coordinates, as well as the edge point coordinates of each plane for each 3D model.

4.2 | Contribution of 3D alignment network

As we have mentioned in Section 3, we proposed four different networks: Basic-GAN, Input-GAN, Feature-GAN, and Input & Feature-GAN. The main difference of these four networks is the application of 3D alignment network (feature alignment model and input alignment model). In this part, we explored the contributions of these models to object generation. The experiment setting of these four networks all follows Section 3.3. We first trained our networks, and then tested the four networks on each category in the data set, separately. Next, we recorded the 95% Hausdorff distance, and compared the difference between four networks for each category, as shown in Table 1 and Figure 9. In our paper, we random select 10 categories to illustrate experiment results.

Table 1 quantitatively shows the point-generation ability of four different proposed networks on different categories in terms of the 95% Hausdorff distance. The average value of Input & Feature-GAN is the lowest (20), and the value of Input-GAN (33.9) and Feature GAN (34.4) occupies the middle level, the value of Basic-GAN is the highest (53.8). In Figure 9, we can see that the values of Input & Feature-GAN is lower than the other three networks in all the categories. Based on the comparison above, we confirmed that the application of 3D alignment model is effective, and the Input & Feature-GAN performs best, which was determined as the final version of proposed network in our experiment.

TABLE 1 The 95% Hausdorff distance of each proposed network on ten categories

	Basic-GAN	Input-GAN	Feature-GAN	Input & Feature-GAN
airplane	100	52.3	55.7	44.7
bench	88.8	62.4	58.1	33.8
car	101	101	66.6	52.3
chair	12.8	12.8	8.23	2.09
cup	14.8	14.8	9.67	4.17
curtain	240	170	190	102
lamp	34.3	13.3	15.4	7.86
laptop	395	281	303	220
sofa	40.9	11.5	12.3	5.88
table	11.5	16.9	9.87	4.72
ModelNet40	53.8	33.9	34.4	20

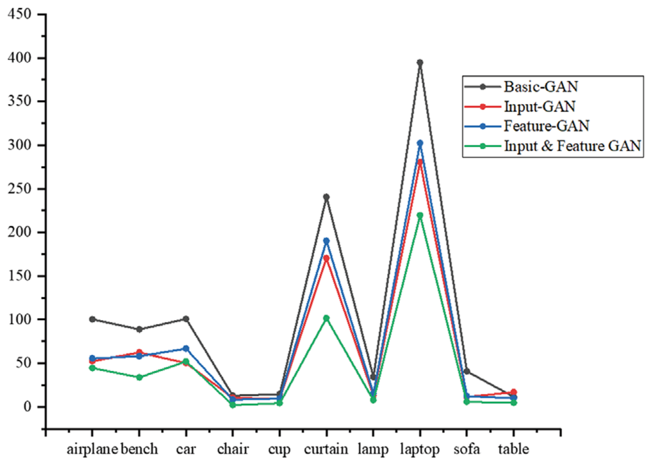

FIGURE 9 The comparison of 95% Hausdorff distance on different proposed networks

4.3 | 3D object generation in point-cloud space

In the former section, we evaluated the performance of four different networks on each category, and determined the structure of proposed network (Input & Feature-GAN). In this part, we first analysed the point clouds generated by proposed network, as shown in Figure 10. And then, we compared the output of proposed method (illustrated in Section 3.4) with raw object in ModelNet40 data set, as shown in Figure 11.

Figure 10 records the point coordinates synthesized by proposed network, which can only synthesize part of the raw object. As shown in Figure 10, the shape of synthesized object by proposed network for one time cannot represent the raw object, and be recognized by any existing method, which is mainly because the size of synthesized points ($m = 1024$) is lower than that of raw object in the point-cloud space. In addition, based on the comparison of two examples of synthesized points for

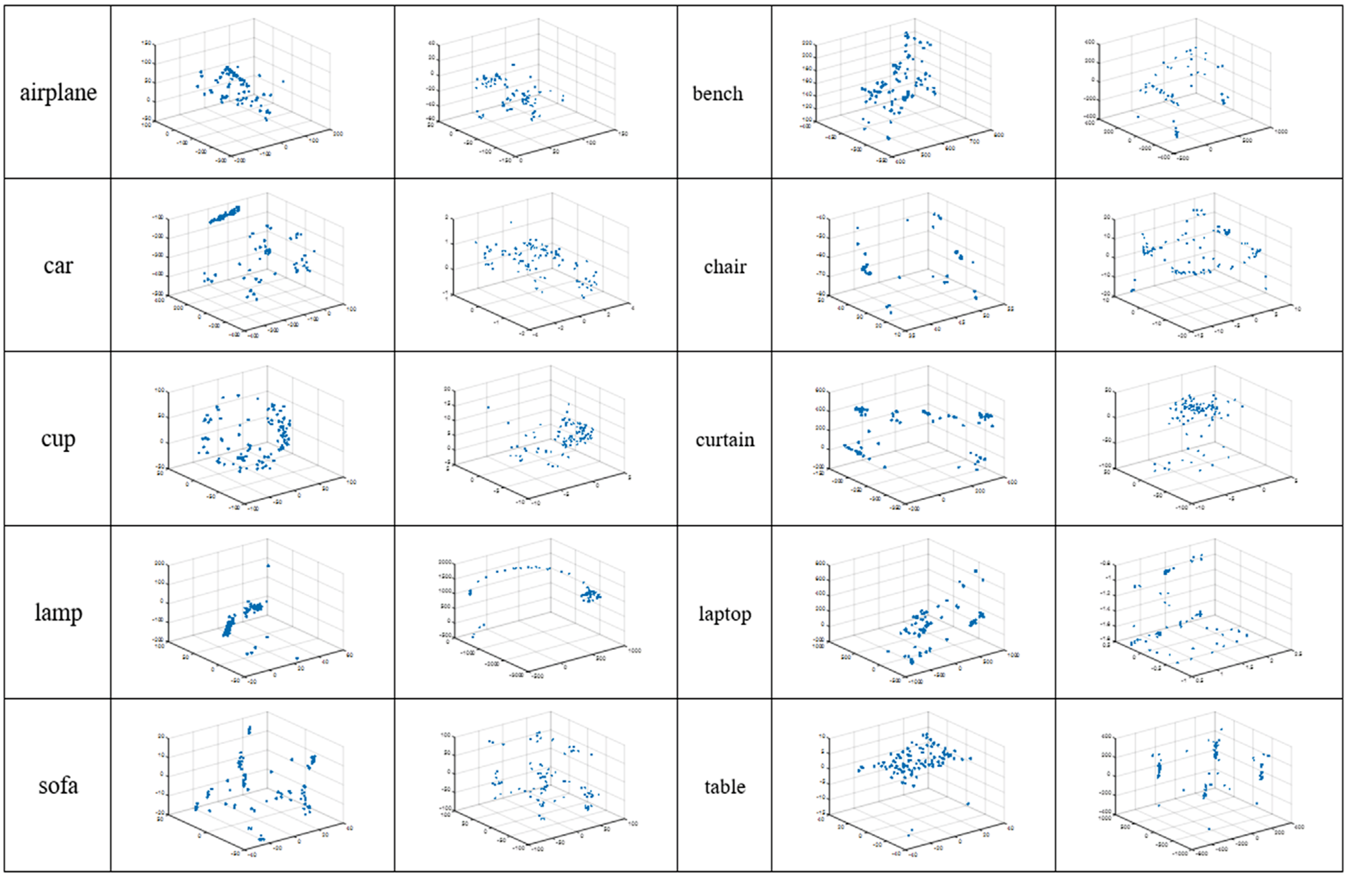

FIGURE 10 The shape of synthesized point clouds by Input & Feature-GAN

each category, we found that the points generated by proposed network are different each time, and they are all a small fraction of the 3D object. In order to tackle this issue, we introduced a new approach based on the proposed network, as illustrated in Section 3.4. The input of the introduced approach are the set of synthesized points, as shown in Figure 10. The synthesized objects in the point-cloud space by proposed approach are shown in Figure 11.

As shown in Figure 11, the synthesized objects can be easily classified, and the synthesized objects can represent the 3D object completely. In addition, the synthesized objects for each category are different from each other, which confirm that the mechanism of proposed network and method is learning and prediction rather than memory, which is a significant improvement of our research compared with existing methods. Next, the quantitative evaluation between raw objects in ModelNet40 data set and synthesized objects was carried out based on proposed evaluation system.

4.4 | Performance evaluation

In the previous section, we analysed the experimental performance from the perspective of visual effect, where the quantitative analysis is needed. As we have illustrated, the generation process is a two-stage framework: proposed network (Input & Feature-GAN) and proposed method (Figure 8). In this section, we evaluated the performance of these two aspects, respectively.

TABLE 2 Comparison of the classification accuracy of proposed network and other state-of-the-art methods

Method	Representation	Accuracy
T-L Network [19]	Volumetric space	74.4%
VConv-DAE [49]	Volumetric space	75.5%
3D-GAN [22]	Volumetric space	83.3%
FoldingNet [44]	Point-cloud space	88.4%
Point-Cloud GAN [45]	Point-cloud space	87.5%
Proposed network	Point-cloud space	88.9%

First, the evaluation of proposed network was carried out. We applied unsupervised classification accuracy to demonstrate the generation ability, and the detailed information has been introduced in Section 3.5. This evaluation method enables our proposed network to be compared with existing methods, which is very important. At the same time, this kind of method eliminates the barrier when comparing existing methods, where the format of synthesized object are different, such as volume-space.

Table 2 compares the unsupervised classification accuracy with five the state-of-the-art methods tested on ModelNet40

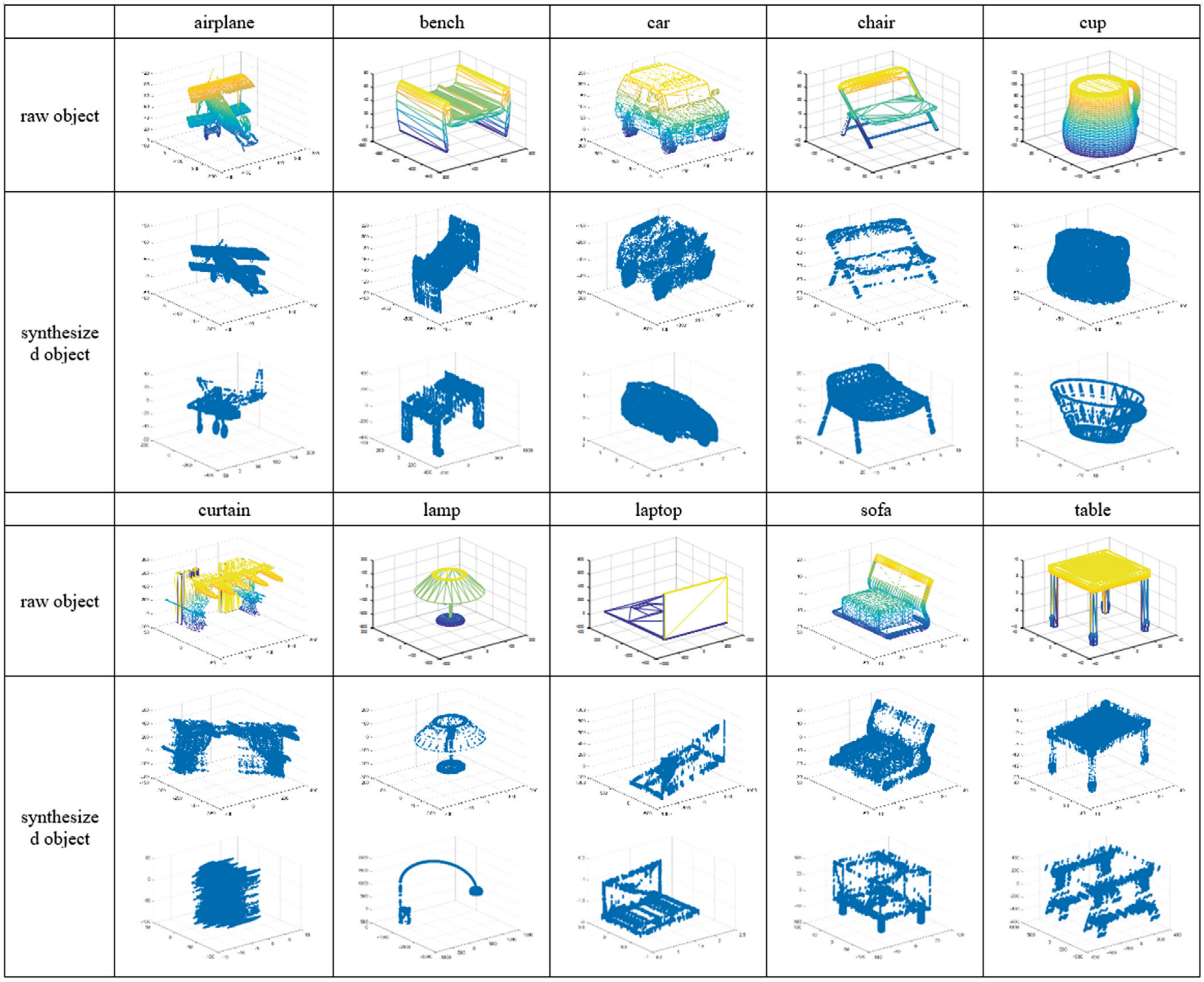

FIGURE 11 The comparison between synthesized objects by proposed method and raw objects in ModelNet40 data set

data set. These can be divided into two parts. The object representation of first part (T-L Network, VConv-DAE and 3D-GAN) is volumetric space, where the highest classification accuracy is 83.3%. The second part synthesized object in the point-cloud space, and our proposed network achieved the highest classification accuracy (88.9%) in two kinds of representation. As we have introduced in the relationship between generator model and discriminator model in GAN, the highest unsupervised classification accuracy also means that our proposed network performs better than the state-of-the-art methods in terms of object generation. However, the unsupervised classification accuracy only reflects the performance of our proposed network (Input & Feature-GAN), and the synthesized point clouds cannot represent a complete 3D object in the point-cloud space, where more processes are needed, as shown in Figure 8.

In this part, we mainly compared the difference between complete synthesized object and raw object based on the achievements of point-cloud recognition. In the field of point-cloud recognition, PointNet [7] and PointNet++ [8] have attracted great attention, and they have been accepted as the effective point-cloud recognition methods. In our research, we first made a point-cloud model data set by proposed method. The settings of this data set is the same as ModelNet40 test data set. And then, we designed three experiments based on PointNet and PointNet++. The experiment settings are as follows:

First, we trained PointNet and PointNet++ on ModelNet40 train data set, and then evaluated classification accuracy of PointNet and PointNet++ on ModelNet40 test data set (Experiment-1), which is also the benchmark test of utilized framework. Second, we trained PointNet and PointNet++ on ModelNet40 data set, and then evaluated their performance on synthesized data set (Experiment-2). Third, we trained PointNet and PointNet++ on synthesized object data set, and then evaluated their performance on ModelNet40 test data set

TABLE 3 The comparison between synthesized objects and raw objects based on the classification accuracy of the state-of-the-art methods

Method	Experiment-1	Experiment-2	Experiment-3
PointNet	89.2%	90.3%	86.4%
PointNet++	91.9%	92.2%	89.9%

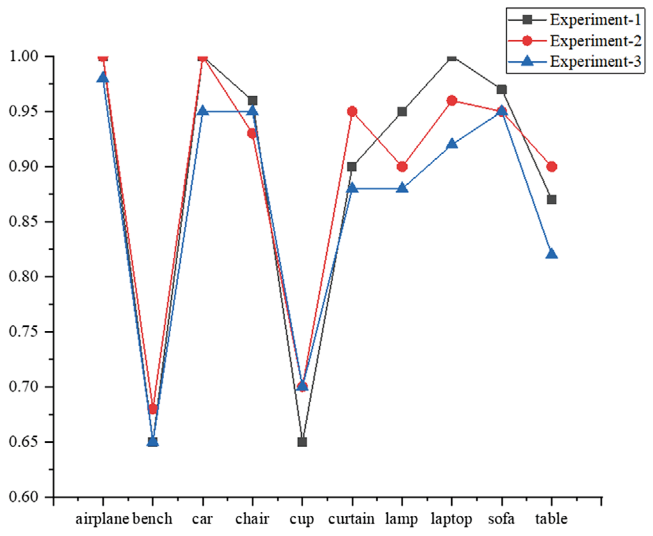

FIGURE 12 The classification accuracy based on PointNet

(Experiment-3). All the experiments setting are the same as the original experiments setting (given by [7, 8]).

Table 3 compares the mean classification accuracy between three different experiments based on the PointNet and PointNet++. In terms of PointNet, the classification accuracy are very similar, and the maximum difference between these three accuracy is 3.9%. In terms of PointNet++, the classification accuracy are also similar, and the maximum difference between these three accuracy is 2.3%. The values of these differences are within the normal range of fluctuation. At the same time, by comparing the accuracy difference between Experiment-1 and other two experiments, we can find the maximum difference is 2.8%, which proves that synthesized object by our method has a high degree of similarity to the raw object. In addition, we also compared the accuracy differences in each category, as shown in Figures 12 and 13. Figures 12 and 13 compare the classification accuracy of three different experiments in ten categories based on PointNet and PointNet++, respectively. The highest differences are 0.07 and 0.05, confirming that synthesized objects are similar to raw objects for each category.

The comparison of the three experiments demonstrates that the objects generated by our proposed method in the point-cloud space have slight difference with man-mad 3D object in the point-cloud space, and the synthesized objects can also be applied for 3D deep learning. Our proposed method has excellent performance.

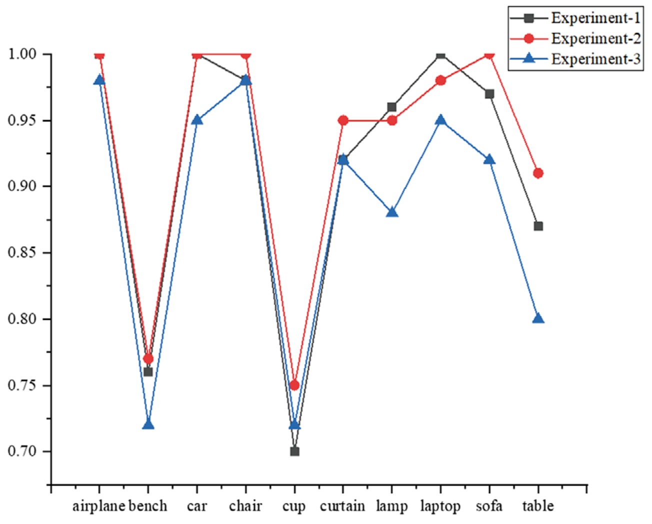

FIGURE 13 The classification accuracy based on PointNet++

5 | CONCLUSION

In this paper, we have proposed a new method for 3D object generation in the point-cloud space, which combines and improves the generative adversarial networks, spatial transformer networks and the point-cloud recognition network. We have shown that our method can synthesize 3D objects from a complex underlying distribution, and the synthesized objects are highly lifelike. We also proposed a comprehensive evaluation system, which can be applied to evaluate the performance of various generating methods from three aspects. From the experiment results, we can draw the three conclusions: the application of 3D spatial transformer networks improves the capacity of object generation in the point-cloud space, our proposed method can generate more accurate 3D object than the state-of-the-art methods in the point-cloud space, and proposed evaluation system can fully quantify the performance of object generation method.

ACKNOWLEDGMENTS
This work was supported in part by the National Natural Science Foundation of China under Grant 52075404, and in part by the Application Foundation Frontier Special Project of Wuhan Science and Technology Bureau under Grant 2020010601012176.

AUTHOR CONTRIBUTIONS

J.H.: Funding acquisition; supervision; writing-original draft; writing-review and editing. W.D.: Investigation; methodology; writing-original draft; writing-review and editing. Q.L.: Supervision; writing-review and editing. K.-M.L.: Writing-original draft; writing-review and editing. P.L.: Funding acquisition; supervision

ORCID
Wupeng Deng https://orcid.org/0000-0001-5917-2294

REFERENCES

1. Wei, H., Wang, L.: Understanding of indoor scenes based on projection of spatial rectangles. Pattern Recogn. 81, 497–514 (2018)
2. Turhan, C.G., Bilge, H.S.: Class-aware single image to 3D object translational autoencoder. IET Image Process. 14(13), 3046–3053 (2020)
3. Rivera, P., et al.: Trilateral convolutional neural network for 3D shape reconstruction of objects from a single depth view. IET Image Process. 13(13), 2457–2466 (2019)
4. Derdar, S., et al.: Localization of topological features using 3D object representations. IET Image Process. 11(8), 578–586 (2017)
5. Zou, Y., et al.: BroPH: An efficient and compact binary descriptor for 3D point clouds. Pattern Recogn. 76, 522–536 (2018)
6. Nurunnabi, A., Sadahiro, Y., Laefer, D.: Robust statistical approaches for circle fitting in laser scanning three-dimensional point cloud data. Pattern Recogn. 81, 417–431 (2018)
7. Charles, Q., et al.: PointNet: Deep learning on point sets for 3D classification and segmentation. In: Proceedings of CVPR, Honolulu, HI, pp. 77–85 (2017)
8. Charles, Q., et al.: PointNet++: Deep hierarchical feature learning on point sets in a metric space. In: Proceedings of NIPS, Long Beach, CA, pp. 5100–5109 (2017)
9. Yang, C., Cheung, G., Stankovic, V.: Estimating heart rate and rhythm via 3D motion tracking in depth video. IEEE Trans. Multimedia 19(7), 1625–1636 (2017)
10. Wang, C., Liu, Z., Chan, S.: Superpixel-Based hand gesture recognition with kinect depth camera. IEEE Trans. Multimedia 17(1), 29–39 (2015)
11. Zhou, Y., et al.: 3D shape classification and retrieval based on polar view. Inf. Sci. 474, 205–220 (2019)
12. Wu, Z., et al.: 3D ShapeNets: A deep representation for volumetric shapes. In: Proceedings of CVPR, Boston, MA, pp. 1912–1920 (2015)
13. Liu, A., et al.: Multi-view hierarchical fusion network for 3D object retrieval and classification. IEEE Access 7, 153021–153030 (2009)
14. Kang, L., et al.: Two-view underwater 3D reconstruction for cameras with unknown poses under flat refractive interfaces. Pattern Recogn. 69, 251–269 (2017)
15. Li, B., et al.: 3D-ReConstnet: A single-view 3D-object point cloud reconstruction network. IEEE Access 8, 83782–83790 (2020)
16. Romain, L., et al.: Information constraints on auto-encoding variational Bayes. In: Proceedings of NIPS, Montreal, Canada, pp. 6114–6125 (2018)
17. Goodfellow, L., et al.: Generative adversarial nets. In: Proceedings of NIPS, Montreal, Canada, pp. 2672–2680 (2014)
18. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv: 1511.06434 (2016)
19. Girdhar, R., et al.: Leaning a predictable and generative vector representation for objects. In: Proceedings of ECCV, Amsterdam, Netherlands, pp. 484–499 (2016)
20. Su, H., et al.: Render for cnn: Viewpoint estimation in images using CNNS trained with rendered 3D model view. In: Proceedings of ICCV, Santiago, Chile, pp. 2686–2694 (2015)
21. Wu, J., Zhang, C., Xue, T., et al.: Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling. In: Proceedings of NIPS, Barcelona, Spain, pp. 82–90 (2016)
22. Yang, B., et al.: 3D object reconstruction from a single depth view with adversarial learning. Proceedings of ICCVW, Venice, Italy, pp. 679–688 (2017)
23. Jaderberg, M., et al.: Spatial transformer networks. In: Proceedings of NIPS, Montreal, Canada, pp. 2017–2025 (2015)
24. Maturana, D., Scherer, S.: Voxnet: A 3d convolutional neural network for real-time object recognition. In: Proceedings of IROS, Hamburg, Germany, pp. 922–928 (2015)
25. Minto, L., Zanuttigh, P., Pagnutti, G.: Deep learning for 3D shape classification based on volumetric density and surface approximation clues. In: Proceedings of VISIGRAPP, Funchal, Madeira, Portugal, pp. 337–324 (2018)
26. Li, Y., et al.: Fpnn: Field probing neural networks for 3D data. In: Proceedings of NIPS, Barcelona, Spain, pp. 307–315 (2016)
27. Qi, C.R., Su, H., Niebner, M., et al.: Volumetric and multi-view CNNS for object classification on 3D data. In: Proceedings of CVPR, Las Vegas, NV, pp. 5648–5656 (2016)
28. Su, H., Maji, S., Kalogerakis, E., et al.: Multi-view convolutional neural networks for 3D shape recognition. In: Proceedings of ICCV, Santiago, Chile, pp. 945–953 (2015)
29. EI-Sayed, E., et al.: Plane detection in 3D point cloud using octree-balanced density down-sampling and iterative adaptive plane extraction. IET Image Process. 12(9), 1595–1605 (2018)
30. Yu, T., Meng, J., Yuan, J.: Multi-view harmonized bilinear network for 3D object recognition. In: Proceedings of CVPR, Salt Lake City, UT, pp. 186–194 (2018)
31. Kalogerakis, E., et al.: A probabilistic model for component-based shape synthesize. ACM Trans. Graphics 31(4), 1–11 (2012)
32. Huang, H., Kalogerakis, E., Marlin, B.: Analysis and synthesize of 3d shape families via deep-learned generative models of surfaces. Comput Graph Forum 34(5), 25–38 (2015)
33. Zou, C., et al.: An example-based approach to 3D man-made object reconstruction from line drawings. Pattern Recogn. 60, 543–553 (2016)
34. Dou, P., et al.: Monocular 3D facial shape reconstruction from a single 2D image with coupled-dictionary learning and sparse coding. Pattern Recogn. 81, 515–527 (2018)
35. Shi, B., et al.: DeepPano: Deep panoramic representation for 3D shape recognition. IEEE Signal Proc. Lett. 22(12), 2339–2343 (2015)
36. Pang, S., et al.: Building discriminative CNN image representation for object retrieval using the replicator equation. Pattern Recogn. 83, 150–160 (2018)
37. Konstantinos, S., Ioannis, P., Theoharis, T.: Ensemble of PADORAMA-based convolutional neural networks for 3D model classification and retrieval. Comput. Graph 71, 208–218 (2018)
38. Lv, Z., et al.: BIM big data storage in WebVRGIS. IEEE Trans. Ind. Inf. 16(4), 2566–2573 (2020)
39. Lv, Z., et al.: Virtual reality smart city based on WebVRGIS. IEEE Internet Things 3(6), 1015–1024 (2016)
40. Lv, Z., et al.: Managing big city information based on WebVRGIS. IEEE Access 4, 407–415 (2016)
41. Ishaan, G., et al.: Improved training of wasserstein GANs. In: Proceedings of NIPS, Long Beach, CA, pp. 5768–5778 (2017)
42. Youssef, M., Tom, S., Vaibhava, G.: McGAN: Mean and covariance feature matching GAN. In: Proceedings of ICML, Sydney, Australia, pp. 3885–3899 (2017)
43. Panos, A., et al.: Learning representations and generative models for 3D point clouds. In: Proceedings of ICML, Stockholm, Sweden, pp. 67–85 (2018)
44. Yang, Y., et al.: FoldingNet: Point cloud auto-encoder via deep grid deformation. In: Proceedings of CVPR, Salt Lake City, UT, pp. 206–215 (2018)
45. Li, C., et al.: Point cloud GAN. In: Proceedings of ICLR, New Orleans, LA (2019)
46. Sarmad, M., Lee, H.J., Kim, Y.M.: RL-GAN-Net: a reinforcement learning agent controlled GAN network for real-time point cloud shape completion. arXiv: 1904.12304 (2019)
47. Gu, J., et al.: Recent advances in convolutional neural networks. Pattern Recogn. 77, 354–377 (2018)
48. Kang, L., et al.: Two-view underwater 3D reconstruction for cameras with unknown poses under flat refractive interfaces. Pattern Recogn. 69, 251–269 (2017)
49. Sharma, A., Grau, O., Fritz, M.: VConv-dae: Deep volumetric shape learning without object labels. In: Proceedings of the ECCV, Amsterdam, Netherlands, pp. 236–250 (2016)

How to cite this article: Hu J, Deng W, Liu Q, Lam K-M, Lou P. Constructing an efficient and adaptive learning model for 3D object generation. *IET Image Process.* 2021;15:1745–1758. https://doi.org/10.1049/ipr2.12146