Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 7584576, 11 pages
https://doi.org/10.1155/2020/7584576

Research Article

Recognition of 3D Shapes Based on 3V-DepthPano CNN

Junjie Yin,[1] Ningning Huang,[2] Jing Tang,[1] and Meie Fang [1]

[1]*School of Computer Science and Cyber Engineering, Guangzhou University, Guangzhou, China*
[2]*School of Computer, Hangzhou Dianzi University, Hangzhou, China*

Correspondence should be addressed to Meie Fang; fme@gzhu.edu.cn

Received 12 August 2019; Accepted 6 January 2020; Published 30 January 2020

Academic Editor: Eckhard Hitzer

This paper proposes a convolutional neural network (CNN) with three branches based on the three-view drawing principle and depth panorama for 3D shape recognition. The three-view drawing principle provides three key views of a 3D shape. A depth panorama contains the complete 2.5D information of each view. 3V-DepthPano CNN is a CNN system with three branches designed for depth panoramas generated from the three key views. This recognition system, i.e., 3V-DepthPano CNN, applies a three-branch convolutional neural network to aggregate the 3D shape depth panorama information into a more compact 3D shape descriptor to implement the classification of 3D shapes. Furthermore, we adopt a fine-tuning technique on 3V-DepthPano CNN and extract shape features to facilitate the retrieval of 3D shapes. The proposed method implements a good tradeoff state between higher accuracy and training time. Experiments show that the proposed 3V-DepthPano CNN with 3 views obtains approximate accuracy to MVCNN with 12/80 views. But the 3V-DepthPano CNN frame takes much shorter time to obtain depth panoramas and train the network than MVCNN. It is superior to all other existing advanced methods for both classification and shape retrieval.

1. Introduction

Three-dimensional shape information provides the important geometric features for identifying 3D objects. Three-dimensional shape analyses have been widely used in computer-aided design/manufacturing/engineering (CAD/CAM/CAE), virtual reality, augmented reality, robotics, and mechanical industrial design. 3D shape analysis methods can generally be classified into two categories: 3D data analysis methods and 2D view learning methods. 3D data analysis methods are primarily based on the geometric characteristics of surfaces or surrounding space of a 3D model recorded in a specific data format. Existing methods include normality and curvature methods represented by histograms or feature sets [1, 2], dense sampling and shape diameters calculation [3], thermonuclear signal representation of polygonal meshes [4, 5], and voxel-based methods [6–8]. Except for the voxel-based methods, all these methods need to extract artificial features at first. Since the effect of artificial features is often limited by the choices of traditional geometric feature descriptors, these methods often do not perform satisfactorily. On the other hand, voxel-based methods face great challenges in neural network computation for complex objects due to the enormous voxel data size. 2D view learning methods obtain a group of 2D views from a 3D shape and then analyze them by neural networks. Two-dimensional views can be 2D projections or panoramic views of a 3D shape [9–13]. Although this type of methods can potentially lose some 3D details as a shape representation, they also have many advantages. For example, 2D data are often more efficient (in terms of memory use) in representing spatial information. 2D views are sometimes better in preserving certain 3D geometric properties, such as noisy surface with holes. Since the approach of learning from 2D views has been widely used in other learning techniques such as in deep learning, some of the existing learning techniques for 2D views may be adopted for our use.

Our approach in this paper is a 2D view-based method. One of the classical 2D view-based methods is proposed by Murase and Nayar [14], which uses a variety of different illuminations to render a 3D model in multiple camera poses, thereby obtaining multiple views for each model. A popular graphics technique, LightField [15], draws the outline of the model from many different perspectives. Macrini et al. [16] split the contour of the model into multiple parts and represent each part by a directed acyclic graph. Cyr and Kimia [17] combine curvatures and 2D views to measure the similarity of 3D shapes. With the recent popularity of deep learning techniques, more and more 3D model recognition methods have begun to apply deep neural networks. Wang et al. [18] propose a sketch-based model retrieval method, which uses a sketch to find the closest view for each model in each class and then uses these views to train a convolutional neural network. Wu et al. [6] train 3D voxel data into a 3D convolution filter.

Shi et al. [12] train the cylindrical panorama of the model. Feng et al. [10] adopt spherical projections to obtain 2D panoramic views. But spherical projection is prone to distortions. Su et al. [13] generate multiple views of the model from multiple angles and then use a parallel CNN to train these views. A view-pooling layer is set in the network to merge the convolution results of different views.

As a typical deep learning method, DeepPano [12] shows clear improvement over other traditional 2D view-based methods. But it adopts only a single view; hence, information at the top and the bottom of a 3D shape can be missed. The MVCNN method [13] utilizes multiple views and applies a parallel multibranch deep network to identify these views. Its recognition rate appears to be the best among the existing methods. However, using too many views may cause information redundancy, and 2D views obtained by planar projection can still lose 3D depth information. In addition, the process of obtaining 12/80 views of rendering images is time-consuming.

To overcome the potential loss of 3D information in 2D view-based methods, we use depth panorama as a view descriptor to store 2.5D information of a 3D shape. Our 2D views are obtained by applying the three-view engineering drawing principle for efficient 2D representations. As in engineering drawing, the three views, i.e., front view, top view, and left view, are designed to efficiently represent the 3D structure of a 3D shape. For a certain class of 3D objects, the 3D shape can be uniquely reconstructed from the three views. The process of obtaining 3 views of depth panoramas is fast. The proposed 3V-DepthPano CNN method adopts this principle and selects the three key views (front, top, and left) to provide maximal projective representation with the minimal number of 2D views. In addition, we include in this representation the depth panorama images formed by cylindrical projection in the directions of the three key views as part of the input to the parallel network for training. Features of the three views are integrated together to represent the 3D shape. A three-branch convolutional neural network, similar to MVCNN [13], is designed to train the model to generate a high-precision descriptor which can be used for model classification and 3D shape retrieval.

This paper is organized as follows. The framework of the proposed 3V-DepthPano CNN method, as well as its comparison with MVCNN, is introduced in Section 2. In Section 3, we describe how to choose key views of 3D shape according to the three-view drawing principle and how to construct cylindrical projections to obtain panoramas of the three views. Section 4 develops the technical details for extracting features of panoramas and for the classification and retrieval of 3D shapes using 3V-DepthPano CNN. Several experiments are shown in Section 5 using several popular datasets to illustrate the effectiveness of 3V-DepthPano CNN for 3D shape recognition. Comparisons of the results with several well-known methods are also discussed. Section 6 concludes this paper with final remarks and future work.

2. Overview of 3V-DepthPano CNN

Figure 1(a) shows an overview of the framework of our convolutional neural network. The 3D shape is projected onto a cylinder to obtain three panoramic views in the three key directions. The three preprocessed panoramas are inputted to the network and trained in the branches of the neural network to produce three eigenvectors $d_i, i = 1, 2, 3$. In the blend-pooling layer, three eigenvectors d_i are merged to obtain a unified eigenvector d_p. Finally, d_p is transmitted to the second part of the network to be trained to obtain the classification probability f_p, which is a feature vector. The highest probability in this vector is the final classification category.

Figure 1(b) shows the training pipeline of the MVCNN method [13]. It places a virtual camera at every 30° angle to generate a rendering image on a projection plane, totaling 12 images. Or alternatively, it can obtain 4 planar rendering images at angles 0°, 90°, 180°, and 270° at each surface point of the icosahedron bounding box of a 3D shape model, totaling 80 images. These 2D images are transferred to the CNN with 12 or 80 branches for training. Multiple feature vectors were obtained at the view-pooling layer. The resulting feature vectors are passed to the second CNN for training. The main differences between MVCNN and our 3V-DepthPano CNN are in the following aspects: (1) The input information is different. MVCNN uses 2D rendering images, and 3V-DepthPano CNN uses 2.5D depth panoramas. This leads to different feature descriptors, with the feature descriptor 3V-DepthPano CNN containing more 3D spacial information. (2) 3V-DepthPano CNN uses fewer views than MVCNN: 3 versus 20 or 80. Consequently, 3V-DepthPano CNN requires a much smaller number of branches, which lead to better efficiency. (3) Although the view-pooling layer of MVCNN and the blend-pooling layer of 3V-DepthPano CNN have similar functionality, i.e., fusing multiple feature vectors, the blend-pooling layer of 3V-DepthPano CNN is inserted to a different place to achieve better analysis accuracy. More details will be given in Section 4.1.

3. View and Panorama Generation

In order to generate depth panorama of a 3D shape, we project its depth information to a cylinder surface whose central axis is parallel to the principal axis of the 3D object. According to the three-view drawing principle, we choose x, y, and z axes as the parallel axis to the principal axis of 3D object.

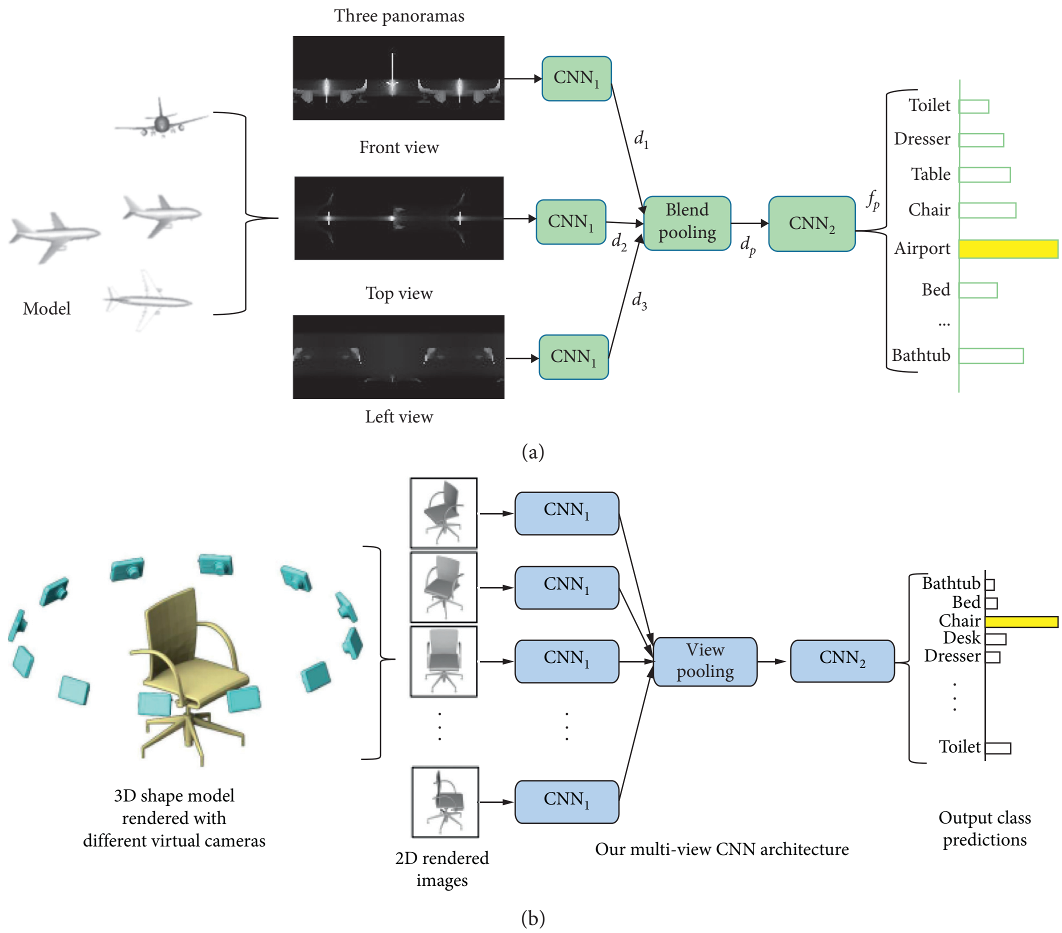


FIGURE 1: Comparison of the frame flow between 3V-DepthPano CNN and MVCNN. (a) Convolutional neural network training process of 3V-DepthPano CNN method. (b) Convolutional neural network training process of MVCNN method [13].

The cylindrical projection in each principal direction is done similarly to what is proposed in [12], though many operational details are quite different. For example, in z-axis projection, as illustrated in Figure 2(a), two groups of coordinate systems are used in the course of cylindrical projection. One group is the Cartesian coordinate xyz-system relative to the 3D object. The other is (θ, h) coordinate system relative to the cylinder surface, where θ represents the polar angle and h represents the z-coordinate. The projection process works as follows:

(1) Compute the central point $C(x_c, y_c, 0)$ of the 3D shape on xoy-plane: x_c, y_c are the weighted averages of x, y coordinates of all central points of the triangular patches on the 3D shape, where the weights are areas of the triangular patches.

(2) Determine the principal axis of the 3D shape: this is the line that is parallel to z-axis and passes through point C.

(3) Determine the central axis, radius, and height of the projected cylinder surface: the principal axis is taken as the central axis of the cylinder. The radius and height of this cylinder are determined in such a way that the 3D shape is completely surrounded by the cylinder surface. We first calculate all distances from every vertex of the 3D shape to point C. If the maximum distance is $d_{\max}$, then the radius $R = 2d_{\max} + \delta_1, \delta_1 > 0$. We also define the height of the cylinder as $H + 2\delta_2, \delta_2 > 0$, where H is the height of the 3D shape, as shown in Figure 2(b).

(4) Subdivide the cylinder surface into grid: we uniformly divide the cylinder surface along θ-direction into M_1 intervals and then uniformly divide the surface along h-direction from the top to bottom into M_2 intervals. The values of M_1 and M_2 are typically between 80 and 254, determined by the size and complexity of the 3D shape.

(5) Project depth information onto the cylinder surface grid: as shown in Figure 2(b), for each grid point $P(m, n, h_p)$ on the cylinder surface, we can find a corresponding point $Q(x_c\, y_c\, h_p)$ on the principal axis. The line PQ will intersect with the 3D shape and form a series of intersection points $I_i, i = 1, 2, 3, \ldots$. We define $D_p = \max_i |QI_i|$ as the depth at point P. If $i = 0$ (no intersection point), then $D_p = 0$. We can also define the depth as the minimum or average value of $|QI_i|$. For simplicity, we will only consider the maximum of $|QI_i|$.

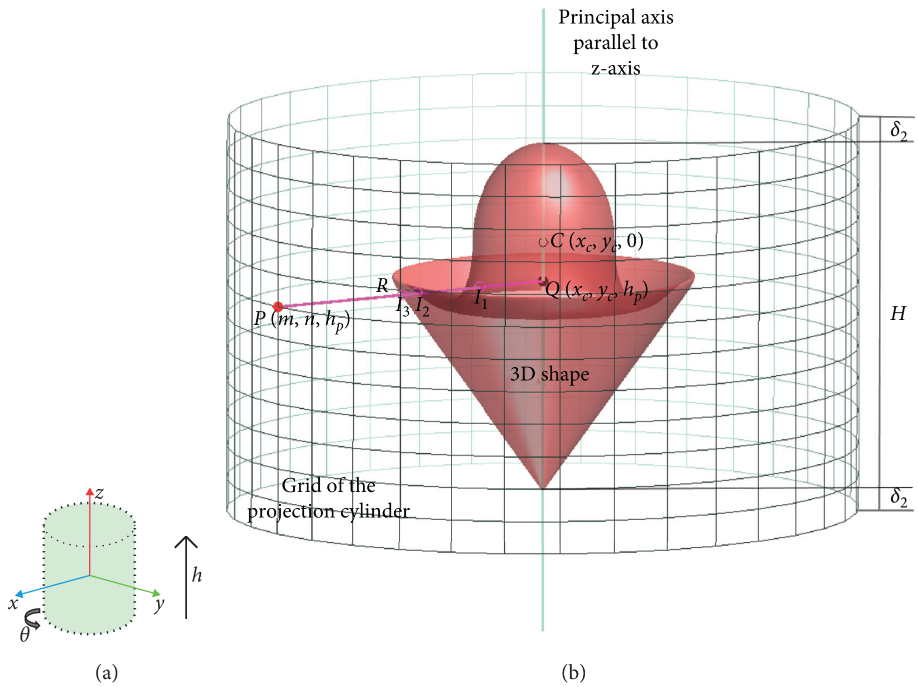


FIGURE 2: Two corresponding coordinate systems and illustration of cylindrical projection.

(6) Unfold the cylinder surface grid into a 2D panorama with depth information: because the cylinder surface is a type of ruled surfaces, it is convenient to be unfolded. We cut it open along $\theta = 0^{\circ}$ and obtain a $(M+1) \times (M+1)$ matrix whose elements are the values of D_p on the corresponding grid point. To normalize the scale of the depth values, we subtract the average and divide standard deviation from the original matrix to obtain the final normalized 2D panorama image whose gray scale pixel intensity represents the normalized depth values.

Apparently, a single cylindrical projection on one view is not sufficient to capture all necessary 3D information. In the proposed 3V-DepthPano method, three panorama views in three key directions are generated to have a more complete coverage of all depth information necessary to describe a 3D object. The depth panorama images of an airplane model in x, y, and z directions are shown in Figure 3. Figure 4 shows the depth panorama images for a number of classical 3D models.

4. 3D Shape Recognition Using 3V-DepthPano CNN

4.1. Extracting Features and Constructing Convolutional Layers. As the depth panoramic views are designed to capture the important 3D information, features extracted from these depth panoramic views can be effective in 3D shape description and retrieval. Although multiple feature vectors extracted from multiple depth panorama images is a more powerful shape descriptor than using a single depth panorama, proper aggregation and fusion of these feature vectors is still critical to the efficiency of the recognition system. Our approach is similar to [13] in using panoramas to training CNN, except that we use three depth panoramas from three key views to train an CNN with three branches. First, the panorama of each direction is transmitted to the corresponding CNN branch. The resulting eigenvectors are aggregated into a feature vector in the blend-pooling layer and then entered into the second part of the network. In the three-view convolutional neural network framework, the parameter settings of all CNN branches of the first part of the network are the same. The fusion of the features from three different views using blend-pooling can help reduce the influence of image distortion in the recognition process. The blend-pooling layer is similar to the max pooling layer and the maxout layer in MVCNN. But the dimensionality of blend-pooling is much lower than that of MVCNN. In addition, their locations in their CNNs are also different. Our experiments show that placing the blend-pooling layer near the final convolutional layer (conv5) leads to the best classification performance.

Our CNN model uses the Alexnet network [19] as a training model, which includes five convolutional layers conv1~conv5, three full-connection layers fc6~fc8, and the last layer is the softmax classification layer. The first two layers of the fully connected layers fc6 and fc7 both have 4096 dimensions, and fc7 outputs the final image descriptor. The Alexnet network uses Relu instead of the sigmoid activation function. The resulting stochastic gradient descent can converge faster. It utilizes a very effective model combination version--Droput, which sets the probability that each hidden neuron's output is zero as 0.5. For each input, neural networks form different structures and these structures share common weights. Because neurons cannot rely on other specific neuronal structures, they are forced to learn more robust features that can effectively prevent overfitting.

4.2. Pretraining Strategies. We first use the large-scale image data set Imagenet to pretrain 3V-DepthPano CNN with three branches. The pretrained CNN model is set as follows:

(1) The size of each image is adjusted to 256×256 pixels. We randomly disturb the order of the images and then extract the subblocks of 227×227 of the image input to the CNN.

(2) Initialize all network parameters randomly. The baseline classifier adopted in this paper is Alexnet [20]. We set the learning rate to 0.01. The momentum is 0.9. The weight attenuation coefficient is 0.0005. And the random drop probability for each layer parameter is set to 0.5. Then, the fine-tuning parameter experiment was carried out. When the learning rate was changed to 0.1 and the weight attenuation coefficient was changed to 0.0008, we can get a satisfactory result. So we set the learning rate to 0.1, the momentum to 0.9, and the weight attenuation coefficient to 0.0008. The random drop probability for each layer parameter is set to 0.5. During training, network parameters are updated using a stochastic gradient descent method. A modified linear unit is used as a nonlinear activation function to prevent the gradient disappearance and the gradient explosion. After the input block is convoluted and pooled multiple times, the system enters the last layer of the full connection. Softmax translates the output into a probability distribution of 1000 classes. Softmax output value is a probability distribution, a real number between 0 and 1. The sum of softmax equals to 1. The cross entropy is used to determine the actual output and desired output proximity. The softmax and cross entropy are often put together to judge the output value and expectation. And taking avantage of the similarity between them can greatly reduce the computational burden of sloving the gradient. Softmax uses cross entropy as a loss function to calculate the difference between probability distributions.

(3) The iteration epoches are set as 100 for the pretraining process (all images are trained once during a epoch of iteration).

The learning rate is updated once every 20 epoches and is reduced to 0.1 times of the learning rate in the last epoch.

4.3. Fine-Tuning Strategies. Because the pretrained image set and the target task set have different image styles and numbers of categories, when the target image set is identified, extracting the image feature using the pretrained CNN model generally cannot achieve optimal performance. Before the extraction of the image features, the pretrained CNN model parameters need to be fine-tuned with the target image set. The process of fine-tuning the CNN model is as follows:

(1) The image of each target image library is adjusted to 256×256 pixels. We randomly disturb the order of the images and then extract 227×227 subblocks from the original images to be inputted to the CNN.

(2) Assuming that the number of categories of the target image data set is t, we change the number of output neurons of the full-connection layer fc8 of the model to t. For cov1~fc7 layers, the parameters are initialized to those obtained in the pretraining process. The parameters of the last layer of the network are randomly initialized by a Gaussian distribution $G(\mu, \sigma)\,(\mu = 0, \sigma = 0.01)$. The method of parameter setting in fine-tuning is similar to that of the 2nd step of the pretraining process. The difference is that different learning rates are set for different network layers in fine-tuning. The initial learning rate of the first 7 layers is set to 0.0001, and the initial learning rate of fc8 is set to 0.01. A small learning rate is set for the first 7 layers to avoid destroying the parameters obtained from the pretrained CNN model. Setting a higher learning rate for fc8 can speed up the convergence of the network to new optimum. The loss function used in the fine-tuning process is the square loss function. 3D shape recognition is essentially a classification problem. Our method is a view-based two-dimensional image classification method. Referring to the existing two-dimensional classification methods, MSE is often used, so equation (1) is adopted as a loss function, and the minimized square loss function can be defined as

$$L = \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{t} \left(\hat{P}_{ik} - P_{ik} \right)^2. \tag{1}$$

(3) The fine-tuning process includes 60 epoches of iterations.

Similarly, the learning rate is updated once for every 20 epoches of iterations and is reduced to 0.1 times of its original value.

5. Experiments

5.1. Datasets. Our experiments use the following data sets:

(1) Princeton ModelNet (http://modelnet.cs.princeton.edu/) is a large 3D CAD model database containing 127,915 CAD models in 662 categories. It is divided into 2 subsets for training and testing.

(i) ModelNet10 consists of 4899 CAD models in 10 categories.
All models in this subset have been aligned properly (the default state is vertical). Among the models, 3,991 are for training, and the remaining 908 are for testing.

(ii) ModelNet40 contains 40 categories with a total of 12311 models, in which 9,843 models are used

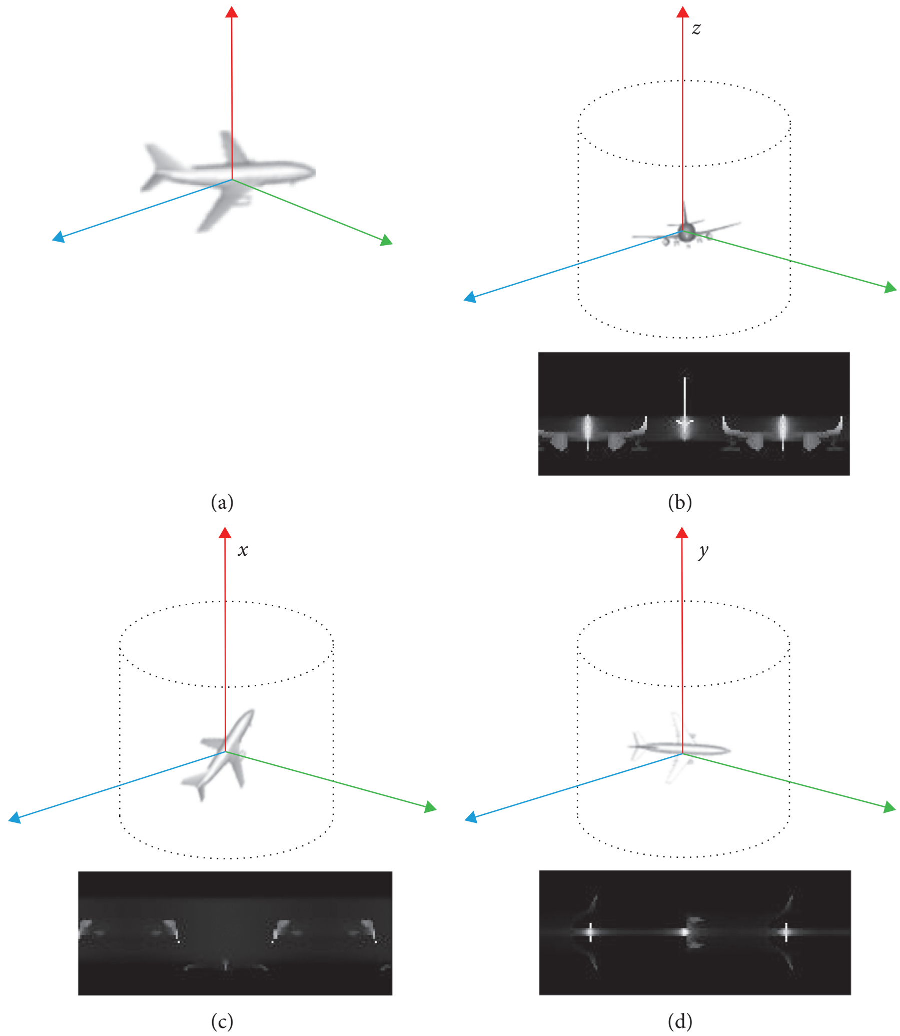


Figure 3: Construction of depth panoramas of a plane model with three views through cylindrical projection.

for training and 2,468 models are used for testing.
Models in ModelNet40 do not have manual alignment, and some model files may exceed the RAM size. So adjustment to the virtual memory size may be necessary.

(2) ShapeNetCore (https://www.shapenet.org/) is a subset of the ShapeNet database, covering 55 common object categories and approximately 51,300 unique 3D models. The popular 3D computer vision benchmark dataset, PASCAL 3D+, is also extracted from ShapeNetCore.

5.2. Implementation Details. We implemented the network within the TensorFlow framework on Linux platform. The panorama is built separately in MATLAB running on a PC with Intel Core i5 CPU, NVS 315 GPU, with 8 GB of RAM. The rendering of a panorama of each 3D shape takes less than one second using CPU only. More speedup can be achieved if a GPU is used. In our experiment, the axis parallel to the 3D shape's upright direction is defined as the z-axis. According to the Cartesian coordinate system, each 3D shape is projected from the cylinder parallel to the x, y, and z directions to obtain three panoramas. The size of each panorama is then normalized into a 65×160 image.

5.3. Classification of 3D Shapes. 3D models are classified using a classification probability vector based on the network output. The last layer of the CNN structure is the softmax layer, which outputs an N-dimensional classification probability vector, and the item with the highest probability in the vector is taken as the class the model is classified into.

In order to evaluate the efficiency of the proposed classification method, the training data are randomly initialized and then used to train the neural network. The trained network outputs the classification probabilities from its softmax layer, and the class with the highest probability is used as the prediction result. This method is then compared with LightField descriptors [15] (LFD, 4700 dimensions), spherical harmonic descriptors [8] (SPH, 544 dimensions), 3DShapeNets [6] (4000 dimensions), MVCNN [13] (4096 dimensions), and DeepPano [12] (4096 dimensions) in

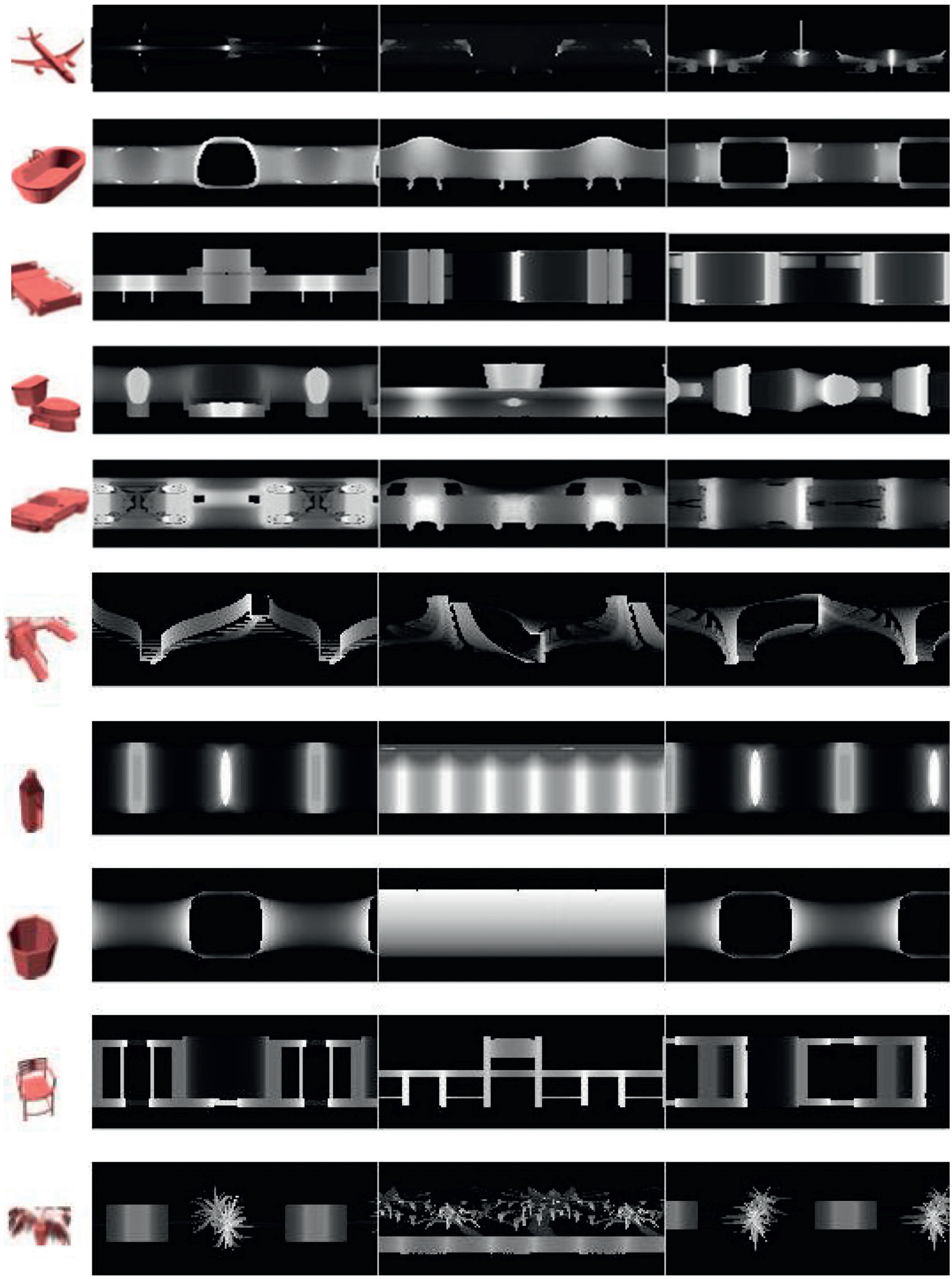


Figure 4: Some 3D shapes from data sets and their depth panoramas on three key views.

terms of classification accuracy. The results are shown in Table 1. The classification accuracy indicates the correct rate of classification.

For the ModelNet10 database, the classification accuracy of our proposed 3V-DepthPano CNN is 89.83%, which is better than all other methods. Compared with LFD and SPH, our classification accuracy is significantly higher because automatically extracted features by deep learning embodies more powerful recognition ability than manually extracted features by LFD and SPH. Compared with 3DShapeNets using deep confidence network, our method performs better because of the power of the 3-branched CNN. Our method also performs better than DeepPano because we utilize more complete depth information in three key views while DeepPano only uses one view.

For the ModelNet40 database, the classification accuracy of the proposed 3V-DepthPano CNN is 88.95%. In this case, the classification accuracy of other methods is not only lower, but the differences are also more significant than in ModelNet10. This is because classification in a larger and more complex shape database requires the use of more complete 3D information, which is the advantage of our 3V-DepthPano CNN method.

As illustrated in Figure 1(b), the MVCNN method uses virtual cameras to obtain planar rendering images of a 3D shape from 12 or 80 views, which are not only cumbersome but also redundant. We believe the 3-view approach taken in 3V-DepthPano CNN is more balanced and efficient in shape feature extraction. From Table 1, we can see that the classification accuracy of 3V-DepthPano CNN is very close to those of MVCNN+12 and MVCNN+80, but with only 3 views instead of 12 or 80 views. Fewer views can lead to less complexity in data collection and processing, simpler training branches and networks, lower feature space dimension, and shorter training and classification time.

Table 1: Comparison results of classification accuracies for different methods.

Method	View	Accuracy (%)	
		ModelNet10	ModelNet40
SPH [8]	—	79.79	68.23
LFD [15]	—	79.87	75.47
3D ShapeNets [6]	—	83.54	77.32
DeepPano [12]	1	88.66	82.54
MVCNN+12 [13]	12	—	89.90
MVCNN+80 [13]	80	—	90.10
3 views-Pano	3	89.83	88.95

5.4. Retrieval of 3D Shapes. We also applied the proposed method to 3D shape retrieval. Figure 5 illustrates a pipeline of this process. During the training process, each model in the training set is first projected onto a cylindrical surface to form a depth panoramic set, which is then used to fine tune a pretrained 3-view neural network. During the course of retrieval, the depth panorama of the target model is matched by similarity to generate a sorted list of matching results.

Use the pretrained 3V-DepthPano CNN to extract feature descriptors fc6~fc8 from each panorama through the 2nd layer of the CNN. We can also obtain the other three feature descriptors, Ft-fc6, Ft-fc7, and Ft-fc8, using the fine-tuning techniques discussed in Section 4.3. For similarity measures, as in [13], the distance $d(A,B)$ between two models A, B is defined as

$$d(A,B) = \frac{1}{6}\left(\sum_{j=1}^{3}\min_{i=1,2,3}\left\|A_i - B_i\right\|_2 + \sum_{i=1}^{3}\min_{j=1,2,3}\left\|B_j - A_j\right\|_2\right), \tag{2}$$

where $\|A_i - B_j\|_2$ represents the l_2 distance between the feature vector of i-th panorama of model A and the feature vector of j-th panorama of model B.

Table 2 shows the retrieval results by different features of the panoramic database.

As shown in Table 2, the accuracies of 3V-DepthPano CNN using the features fc6, fc7 and fc8 are 4% to 34% higher than other methods for dataset ModelNet10.

For dataset ModelNet40 using features fc6, fc7, and fc8, our accuracies are 10% to 35% higher than other methods except MVCNN which performs only slightly (1.5% to 4.4%) better than our method. But our method uses a fewer number of views and is therefore more efficient in both data processing and computational time. For example, MVCNN needs about 100 seconds to construct the rendering images for all 80 views, while 3V-DepthPano CNN needs less than 1 second to obtain depth panoramas for its three views. Since our CNN only has three branches, the training time needed is also much shorter.

Table 2 also shows that the accuracy of our method using features ft-fc6, ft-fc7, and ft-fc8 is higher than those without fine-tuning. They are also higher than those of MVCNN+12 and MVCNN+80. This clearly demonstrates the effectiveness of fine-tuning 3V-DepthPano CNN model.

Figure 6 shows the comparison of precision-recall curves for the above 12 features. The precision-recall curve illustrates the tradeoff between the fraction of retrieved instances that are positive (precision) and the fraction of retrieved instances to all positive instances (recall). The larger the area under the curve, the better the classification performance. Figure 6(a) shows the comparison of the precision-recall curve using the proposed fc6 of 3V-DepthPano and those using other features for dataset ModelNet10; Figure 6(b) shows the comparison of the precision-recall curves using the proposed six features of 3V-DepthPano extracted from the proposed method for dataset ModelNet10. Figure 6(c) shows the comparison of the precision-recall curve using the proposed fc6 of 3V-DepthPano and those using other features for dataset ModelNet40; Figure 6(d) shows the comparison of the precision-recall curves using the proposed six features of 3V-DepthPano and that using MVCNN for dataset ModelNet40.

From Figures 6(a) and 6(c), we can see that the performance of the proposed method is better than of other advanced methods except MVCNN which requires a much larger number of views. In Figure 6(b), feature ft-fc7 of six

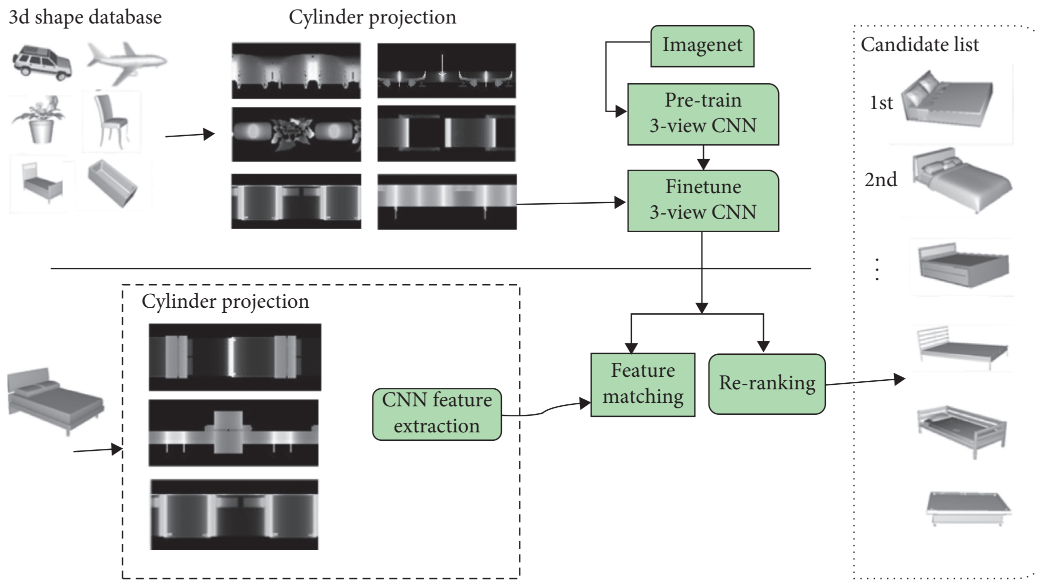
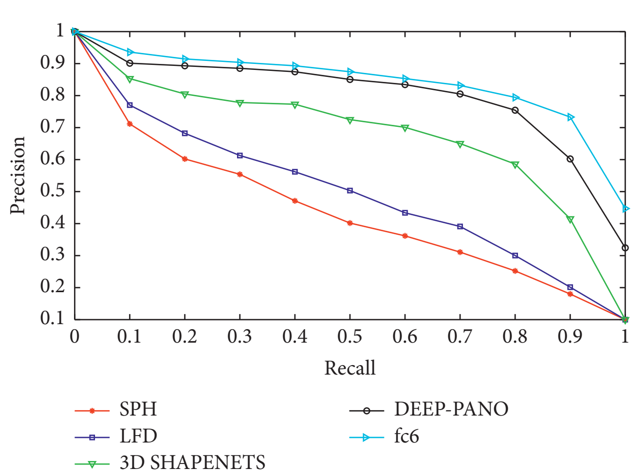


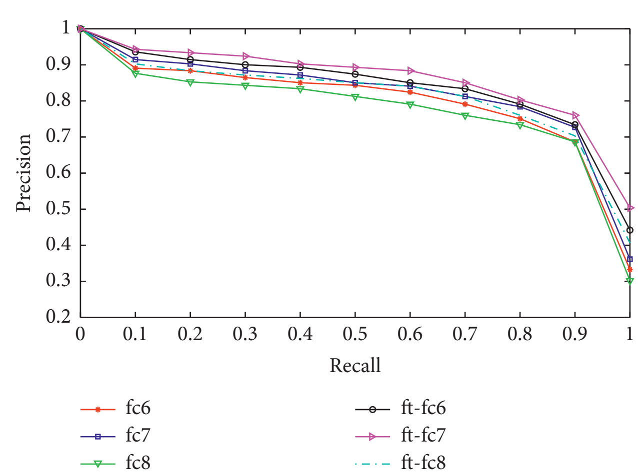
Figure 5: The retrieval flow of 3D shape by 3V-DepthPano CNN.

Table 2: Comparison results of average accuracy rate for 3D shape retrieval.

Feature		View	MAP (%)	
			ModelNet10	ModelNet40
SPH [8]		—	44.05	33.26
LFD [15]		—	49.82	40.91
3D ShapeNets [6]		—	68.26	49.23
DeepPano [12]		1	69.88	56.14
MVCNN+12 [13]		12	—	70.13
MVCNN+80 [13]		80	—	70.41
3V-DepthPano	fc6	3	74.38	65.62
	fc7	3	76.12	68.59
	fc8	3	75.23	69.56
	ft-fc6	3	75.57	67.81
	ft-fc7	3	78.89	71.74
	ft-fc8	3	77.92	72.14



(a)



(b)

Figure 6: Continued.

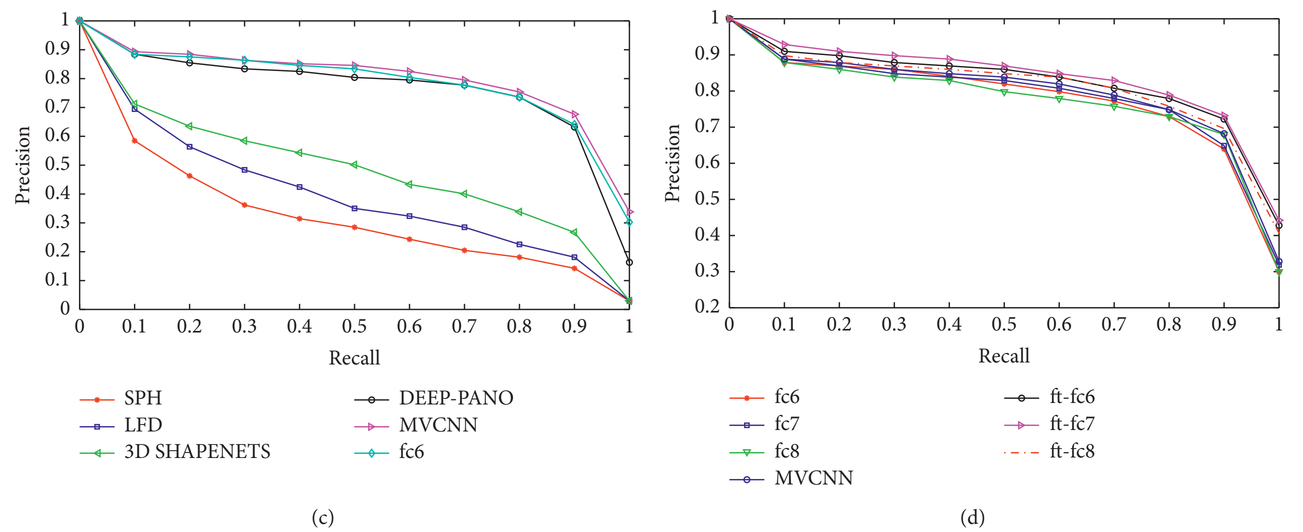


Figure 6: Comparison of precision-recall curves for the 12 features listed in Table 2.

features extracted under the 3V-DepthPano method has the best retrieval performance. In Figure 6(d), the retrieval performances of fc6, fc7, and fc8 extracted during pretraining of 3V-DepthPano CNN are slightly lower than that of the MVCNN method. But after fine-tuning, the retrieval performances of ft-fc6, ft-fc7, and ft-fc8 are better than that of the MVCNN method. This again indicates that our fine-tuning training strategy is effective.

A survey published in [21] over five leading-edge methods of 3D shape retrieval compares their experimental results based on database ShapeNetCore55. The five methods are MVCNN, GIFT, ViewAggregation, CCMLT, and DB-FMCD-FUL-LCDR, which all belong to 2D-view methods. To compare with this survey result, we also applied our 3V-DepthPano on database ShapeNetCore55. Table 3 lists MAPs of the six methods for 3D shape retrieval. The table shows that the performance of 3V-DepthPano CNN is very stable, and its MAP is close to that of MVCNN and clearly better than those of other four methods.

Table 3: Comparison of six kinds of cutting-edge methods for 3D shape retrieval.

Method	View	MAP (%)
MVCNN	12/80	81.7
GIFT	64	74.0
ViewAggergation	12	71.1
CCMLT	36	71.1
DB-FMCD-FUL-LCDR	1	59.6
3V-DepthPano	3	79.8

6. Conclusion and Future Work

In this paper, we use the three-view drawing principle to obtain three key 2.5D depth panoramas of a 3D shape by cylindrical projection and establish a three-branch convolutional neural network for 3D shape recognition, including classification and retrieval. The proposed method only adopts three views and performs nearly as well as MVCC which requires 12 or 80 views and much longer processing and training time. Except MVCC, our method outperforms all other existing methods. We believe our 3V-DepthPano CNN strikes a good balance between performance and system complexity because our baseline network is also MVCNN. MVCNN has 12 branches for feature extracting, while ours has only 3 branches, so the ratio of the time complexity for two methods is 4 : 1. But our network is getting better performance.

Similar to other 2D view-based methods, 3V-DepthPano CNN also needs to determine the principal axis of 3D models, which can sometimes influence the recognition results. In this paper, we only consider x-, y-, and z-axis as principal axes of the 3D object. How to determine the principal axes is still a common challenge to all 2D view-based methods. In addition, we choose the maximum distance as the stored depth information during cylinder projection. It is not clear whether other types of depth information such as minimum distance or mean distance could perform better. Furthermore, how to determine the best number of views for optimal performance is a question that is worthy of further study. In the future, we will perform research upon these problems and adopt more measures to further improve the recognition accuracy.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The work described in this article was partially supported by the National Natural Science Foundation of China (61772164) and Provincial Key Platforms and Major

Scientific Research Projects in Universities and Colleges of Guangdong (2017KTSCX143).

References

[1] X. G. Liu, R. J. Su, S. B. Kang et al., "Directional histogram model for three-dimensional shape similarity," in *Proceedings of Computer Vision and Pattern Recognition*, pp. 813–820, IEEE Computer Society Press, Los Alamitos, CA, USA, 2003.

[2] B. K. P. Horn, "Extended Gaussian images," *Proceedings of the IEEE*, vol. 72, no. 12, pp. 1671–1686, 1984.

[3] S. Chaudhuri and V. Koltun, "Data-driven suggestions for creativity support in 3D modeling," *ACM Transactions on Graphics*, vol. 29, no. 6, p. 183, 2010.

[4] A. M. Bronstein, M. M. Bronstein, L. J. Guibas, and M. Ovsjanikov, "Shape google," *ACM Transactions on Graphics*, vol. 30, no. 1, pp. 1–20, 2011.

[5] I. Kokkinos, M. M. Bronstein, R. Litman et al., "Intrinsic shape context descriptors for deformable shapes," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 159–166, IEEE Computer Society Press, Washington DC, USA, 2012.

[6] Z. Wu, S. Song, A. Khosla et al., "3d shapenets: a deep representation for volumetric shapes," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1912–1920, Boston, MA, USA, June 2015.

[7] D. Maturana and S. Scherer, "VoXNet: a 3D convolutional neural network for real-time object recognition," in *Proceedings of the Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on. IEEE*, pp. 922–928, Hamburg, Germany, October 2015.

[8] M. Kazhdan, T. Funkhouser, and S. Rusinkiewicz, "Rotation invariant spherical harmonic representation of 3d shape descriptors," *Symposium on Geometry Processing*, vol. 6, pp. 156–164, 2003.

[9] A. Kurenkov, J. Ji, A. Garg et al., "DeformNet: free-form deformation network for 3D shape reconstruction from a single image," in *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 858–866, Lake Tahoe, NV, USA, March 2018.

[10] Y. Feng, M. Xia, P. Ji et al., "Deep spherical panoramic representation for 3D shape recognition," *Journal of Computer-Aided Design and Computer Graphics*, vol. 29, no. 9, pp. 1689–1695, 2017.

[11] P. Papadakis, I. Pratikakis, T. Theoharis, and S. Perantonis, "PANORAMA: a 3D shape descriptor based on panoramic views for unsupervised 3D object retrieval," *International Journal of Computer Vision*, vol. 89, no. 2-3, pp. 177–192, 2010.

[12] B. Shi, S. Bai, Z. Zhou, and X. Bai, "Deeppano: deep panoramic representation for 3-d shape recognition," *IEEE Signal Processing Letters*, vol. 22, no. 12, pp. 2339–2343, 2015.

[13] H. Su, S. Maji, E. Kalogerakis et al., "Multi-view convolutional neural networks for 3d shape recognition," *Proceedings of the IEEE International Conference on Computer Vision*, pp. 945–953, Santiago, Chile, December 2015.

[14] H. Murase and S. K. Nayar, "Visual learning and recognition of 3-D objects from appearance," *International Journal of Computer Vision*, vol. 14, no. 1, pp. 5–24, 1995.

[15] D.-Y. Chen, X.-P. Tian, Y.-T. Shen, and M. Ouhyoung, "On visual similarity based 3D model retrieval," *Computer Graphics Forum*, vol. 22, no. 3, pp. 223–232, 2003.

[16] D. Macrini, A. Shokoufandeh, S. Dickinson et al., "View-based 3-D object recognition using shock graphs," *Proceedings of International Conference on Pattern Recognition*, vol. 3, pp. 24–28, IEEE Computer Society Press, Los Alamitos, CA, USA, March 2002.

[17] C. M. Cyr and B. B. Kimia, "A similarity-based aspect-graph approach to 3D object recognition," *International Journal of Computer Vision*, vol. 57, no. 1, pp. 5–22, 2004.

[18] F. Wang, L. Kang, and Y. Li, "Sketch-based 3d shape retrieval using convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1875–1883, IEEE Computer Society Press, Los Alamitos, CA, USA, June 2015.

[19] I. J. Goodfellow, D. Warde-Farley, M. Mirza et al., "Maxout networks," *Computer Science*, vol. 28, pp. 1319–1327, 2013.

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *NIPS*, vol. 1, pp. 1106–1114, 2012.

[21] M. Savva, F. Yu, H. Su et al., "SHREC'16 track large-scale 3D shape retrieval from ShapeNet Core55," in *Proceedings of the EG 2016 Workshop on 3D Object Recognition*, Goslar, Germany, 2016.