3D CNN based Partial 3D Shape Retrieval Focusing on Local Features

Wataru Iwabuchi* and Masaki Aono†
Computer Science and Engineering, Toyohashi University of Technology, Aichi, Japan
E-mail: *iwabuchi@kde.cs.tut.ac.jp, †aono@tut.jp

***Abstract*—In this paper, we propose a new method for 3D CNN based partial 3D shape retrieval focusing on local features. A 3D partial shape in our approach is defined by a collection of points on the visible surface projected on the view-screen, during the rendering of a given 3D shape. We construct a voxel from the partial 3D points after extracting the local feature vectors and subsequent dimensional reduction by PCA (Principla Component Analysis) and feeding the reduced feature vectors to 3D CNN. This is a unique approach in contrast to the traditional approach to 3D CNN where the voxels have their values either 0s or 1s (i.e. binary voxels). We conducted experiments with a SHREC2016 partial 3D dataset. Our proposed approach outperformed the VoxNet. We also compared our proposed method with other previous methods for partial 3D shape search.**

I. Introduction

In recent years, 3D models have been used in various fields such as manufacturing industry, medical care, architectural design, education, and entertainment. Accordingly, the amount of 3D shape models available on the Internet is rapidly increasing, so that the need for accurately searching 3D shape models is also increasing. On the other hand, it has been pointed out that 3D-to-3D shape search is not easy unless we have 3D shape models at hand. To alleviate this problem, it is convenient to employ inexpensive 3D scanners such as Kinect [1], in order to acquire a rough 3D shape as a query to 3D-to-3D shape search. Even though it is relatively convenient to use a 3D scanner to obtain a 3D shape, a new problem has arisen, which is caused by the incompleteness of a 3D shape acquired from a scanner. This incomplete shape can be viewed as a partial shape of a complete 3D shape model.

Research on the partial 3D shape search has been conducted by many researchers [4] [6] [8] [16] [19] However, to our knowledge, no research on partial 3D shape retrieval has employed 3D CNN whose input accepts local features directly.

In this paper, we propose a new partial 3D shape retrieval focusing on local features, which are directly fed into 3D CNN. In our proposed method, 3D mesh data obtained by a 3D scanner is first converted to a point cloud, followed by applying pose normalization, extraction of partial shapes, selection of "representative points", and by extracting local feature vectors. Furthermore, we apply Principal Component Analysis (PCA) to local features in order to reduce the dimension of local feature vectors. Finally, the compressed feature vectors are inserted into a voxel. Thus, a voxel is either a zero vector or a real vector having the dimension of the compressed feature vector. Please note that a traditional method such as VoxNet [14] represents a voxel as either 0 (empty) or 1 (an object is occupying the voxel space), which is sometimes referred to as *binary voxel*.

We conducted experiments of our proposed method by using SHREC 2016 partial dataset and demonstrate that our method outperforms the VoxNet.

II. Related Work

3D partial shape retrieval has been studied by many researchers including Furuya et al. who proposed Randomized Sub-Volumes Partitioning (RSVP) [5], and Tran et al. who proposed a composite approach to partial 3D shape retrieval [16].

RSVP is a method of extracting a partial shape from a 3D model. RSVP divides the 3D model into a grid and produces sub-volumes, corresponding to partial shapes. Sub-volumes can be thought of as feature vectors.

Tran et al. proposes a method for partial 3D shape search by combining the BoVW (Bag of Visual Words) features computed from local features called "RootRoPS", and a method called ICP [16]. VoxNet [14] is a representative method for 3D shape similarity search using voxels and 3D CNN. VoxNet represents a voxel as binary value of either 0 or 1. Specifically, in VoxNet, a 3D model is represented by a collection of voxels whose values are initialized as 0s, and if a point sampled from a 3D shape is included in a voxel, the value of the voxel becomes 1.

Typical research on 3D local features includes Fast Point Feature Histogram (FPFH) [17] and 3DMatch [23]. FPFH is computed from the histograms derived from several angular geometric relationships. 3DMatch is a method of computing local features using 3D CNN proposed by Zeng et al. [23]. They employ Siamese Network [11] to constitute their 3D CNN and make it possible to produce local features from their 3D CNN.

III. Proposed Method

The overall procedure of our proposed method is illustrated in Fig. 1. The training and testing stage details (steps (C) and (D)) extracted from Fig. 1 are shown in Figs. 2 and 3, respectively. First of all, Fig. 2 shows the flow of the training stage for the 3D CNN to extract partial shape features, while Fig. 3 is the flow of the testing stage with the trained 3D CNN. The extraction of partial shape features is common to both the training and the testing stages.

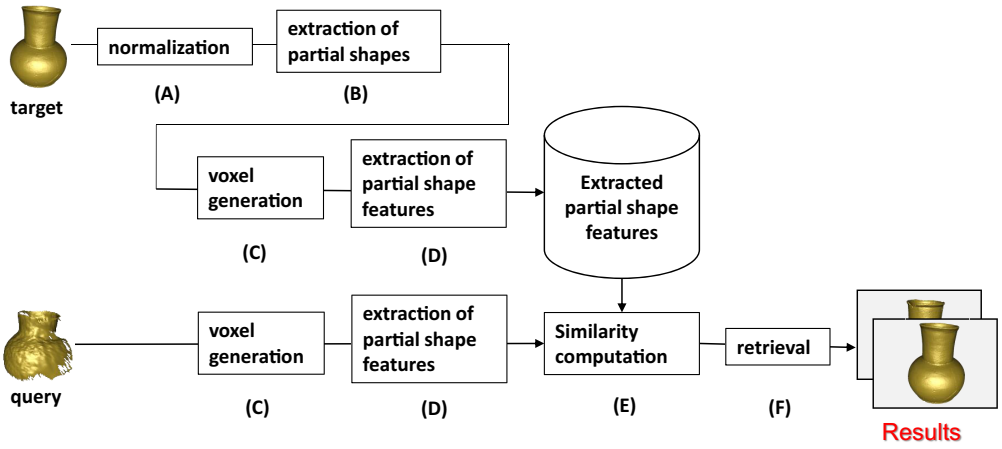

Fig. 1. Overview of our proposed system for partial 3D shape retrieval

Regarding the proposed method, in III-A, normalization of the 3D model will be elaborated, while the extraction of the partial shape model will be elaborated in III-B, generation of voxels with a local feature will be elaborated in the III-C. In III-D, we will explain the partial shape features, the similarity computations in the III-E, and the search in III-F.

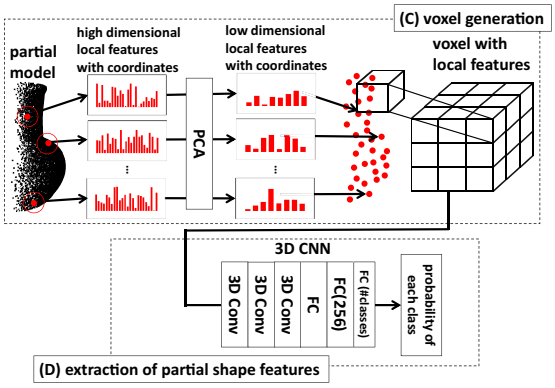

Fig. 2. Flow of training 3D CNN for extracting partial shape feature

A. Normalization of 3D Model

We first normalize the 3D model. This is because 3D shapes generated by anonymous authors have different sizes, locations, and orientations in general.

First, position normalization is performed. The 3D model is translated so that the coordinates of the center of gravity of the model overlap the origin. Next, size normalization is performed. The size is normalized so that the 3D model fits into the unit circle. Calculate the Euclidean distance between the center of gravity and each point, and obtain the maximum value. By dividing the coordinates of each point using the obtained maximum value, the 3D model is converted so as to fit in the unit circle. Next, consider the normalization of the orientation. Normalization of orientation is performed by PointSVD [20] which consists of point cloud generation and singular value decomposition.

Specifically, normalization is performed based on the following equation. First, it can be assumed that the 3D model consists of a triangular mesh without loss of generality. Therefore, we generate random points on the surface of the 3D model as m point clouds [15].

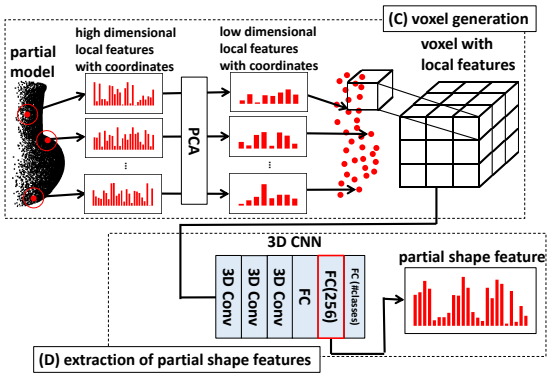

Fig. 3. Flow of a test stage with trained 3D CNN

Here, the coordinates of the points $\mathbf{p}$, uniformly distributed on the surface of the 3D model, are computed from the coordinates of the triangles $\mathbf{a}$,$\mathbf{b}$,$\mathbf{c}$ as shown in the following formula:

$$\mathbf{p} = (1 - \sqrt{r_1})\mathbf{a} + \sqrt{r_1}(1 - r_2)\mathbf{b} + \sqrt{r_1}r_2\mathbf{c}$$

Bratley [2] uses pseudo random numbers such as Sobol or Niederreiter for the two random numbers r_1 and r_2 in the above expression. Next, find the rotation matrix Q which determines the center of gravity $\mathbf{g}$ of the 3D model. An average

value of the generated point cloud is obtained, and it is set as the centroid.

$$\mathbf{g} = \frac{1}{m}\sum_{i=1}^{m}\mathbf{p}^{(i)}$$

Then, we translate the point cloud such that the centroid moves to the origin, and denote the point cloud matrix by P.

$$P = \begin{pmatrix} p_x^{(1)} - g_x & p_x^{(2)} - g_x & \dots & p_x^{(m)} - g_x \\ p_y^{(1)} - g_y & p_y^{(2)} - g_y & \dots & p_y^{(m)} - g_y \\ p_z^{(1)} - g_z & p_z^{(2)} - g_z & \dots & p_z^{(m)} - g_z \end{pmatrix}$$

We then apply Singular Value Decomposition (SVD) to the point cloud matrix P.

$$P = U\Sigma W^T$$

where U and W are 3×3 orthogonal matrices, Σ is a 3×3 diagonal matrix, having its singular values in descending order.

The rotation matrix Q is computed by taking the transpose of the left singular vectors as represented by U.

$$Q = \hat{U}^T$$

We compute the reflection matrix F from the rotated point sets P', where $P' = QP$ as follows:

$$Q = \hat{U}^T$$

Finally, we compute the reflection matrix F from the rotated point sets P', where $P' = QP$ as follows:

$$F = \begin{pmatrix} sign(f_x) & 0 & 0 \\ 0 & sign(f_y) & 0 \\ 0 & 0 & sign(f_z) \end{pmatrix}$$

where

$$f_x = \sum_{i=0}^{m} sign(p'^{(i)}_x)(p'^{(i)}_x)^2 \qquad (f_y, f_z \text{ are similarly defined.})$$

The size and location invariance is achieved by transforming the matrix V into V' with k number of points, where V is a matrix already having rotational and reflective invariance as listed below:

$$V = \begin{pmatrix} v_x^{(1)} - g_x & v_x^{(2)} - g_x & \dots & v_x^{(k)} - g_x \\ v_y^{(1)} - g_y & v_y^{(2)} - g_y & \dots & v_y^{(k)} - g_y \\ v_z^{(1)} - g_z & v_z^{(2)} - g_z & \dots & v_z^{(k)} - g_z \end{pmatrix}$$

$$V' = FQV$$

It should be noted that the size invariance is achieved by taking the distance between the centroid and an arbitrary point on the surface divided by the largest distance between the centroid the farthest point on the surface if we put the 3D object in a unit sphere centered at the centroid.

B. Extraction of Partial Shape

In the partial search, the search query is a partial shape model obtained by 3D scanner, and the search target is a perfect model. Since it is difficult to directly compare them, the proposed method extracts the partial shape model from the perfect model and compares the extracted partial shape model with the search query.

Extraction of partial shape model is performed using our prior method [10]. In the proposed method, we simulate the case of looking at the 3D model from a certain viewpoint and extract the visible part. In our previous method,3D partial shape extraction is performed with 66 viewpoints. Here, for the sake of speeding up, a partial shape model is extracted from 38 viewpoints.

For the extracted partial shape model, normalization described in the III-A section is also performed.

C. Voxel Generation

We first randomly pick up the "representative points" on the 3D shape model.

We adopt FPFH (Fast Point Feature Histogram) and 3DMatch as our local features. FPFH was introduced in the mobile robotics field as the local feature of the 3D model [17], while 3DMatch was introduced by 3D object recognition / reconstruction [23]. They are served as local features for partial search of the 3D shape model.

1) Local Angular Features: Here we adopt FPFH as our first local shape feature. FPFH is implemented as follows:

In the FPFH, a 3D local shape is obtained by computing the histograms of the following three angles as shown in :α, ϕ, θ of equation (1)

$$\begin{aligned} \alpha &= \mathbf{v} \cdot \mathbf{n}_t \\ \phi &= \mathbf{u} \cdot \frac{(\mathbf{p}_t - \mathbf{p}_s)}{d} \\ \theta &= \arctan\left(\frac{\mathbf{w} \cdot \mathbf{n}_t}{\mathbf{u} \cdot \mathbf{n}_t}\right) \end{aligned} \tag{1}$$

where $\mathbf{p}_s$ is a representative point, $\mathbf{p}_t$ is its neighbor, $\mathbf{n}_s$ is a normal to $\mathbf{p}_s$, $\mathbf{n}_t$ is a normal to $\mathbf{p}_t$, d is an Euclidean distance between $\mathbf{p}_s$ and $\mathbf{p}_t$.

Unit vectors $\mathbf{u}$, $\mathbf{v}$, and $\mathbf{w}$, perpendicular to each other, are computed by equation (2).

$$\begin{aligned} \mathbf{u} &= \mathbf{n}_s \\ \mathbf{v} &= \mathbf{u} \times \frac{(\mathbf{p}_t - \mathbf{p}_s)}{\|\mathbf{p}_t - \mathbf{p}_s\|_2} \\ \mathbf{w} &= \mathbf{u} \times \mathbf{v} \end{aligned} \tag{2}$$

In fact, with regard to the three angular features computed from α, ϕ, θ, we calculate all neighboring points within the radius r of the representative point and compute them appropriately. The meaning of the three angles to be converted into the histogram with H bins is as follows: α is the cosine of the angle between the normal vector at the representative

point and the $\mathbf{v}$ axis, ϕ is the cosine of the angle between the neighboring point and the tangent vector $\mathbf{u}$ from the representative point, θ, from the $\mathbf{u}$ axis at the representative point $\mathbf{w}$, representing the elevation angle looking up at the axis. Let this concatenation be SPFH of its representative point.

Finally, we obtain FPFH from SPFH by the following formula. The formula for finding the FPFH of the point $\mathbf{p}_q$ is as follows:

$$\mathrm{FPFH}(\mathbf{p}_q) = \mathrm{SPFH}(\mathbf{p}_q) + \frac{1}{k}\sum_{i=1}^{k}\frac{1}{\omega_i}\cdot \mathrm{SPFH}(\mathbf{p}_i)$$

Here, ω_i represents the distance between $\mathbf{p}_q$ and $\mathbf{p}_i$. k is the number of neighboring points, determined by the radius r. FPFH is defined as the weighted average of k number of SPFH. Please note that FPFH is a $3 \times H$ dimensional vector, where H is the number of bins mentioned above.

2) Local Shape-Pair Feature (LSPF): The second feature we adopt is based on 3DMatch. Since 3DMatch employs "Siamese architecture", i.e. a pair of local patches, we here introduce "local shape-pair feature" (LSPF).The procedure for obtaining this local feature is as follows: First, a cube whose center is a representative point and whose length is l is generated. A point cloud existing inside the generated cube is extracted.

Next, the extracted point cloud is converted into local voxel data. Here, the generated local voxel data is called a patch. With this patch as an input, a pair of two patches are fed into a Siamese. We train this Siamese Network [3]. After training, it can be used as a feature extractor that receives a pair of patches as input and produces a local feature as output.

In our proposed method, we have added a new idea to reduce the influence of the orientation on this local shape pair feature, as illustrated in Fig. 4.With the original definition of 3DMatch, there is a problem that the feature changes depending on the direction in which the patch is extracted. On the other hand, here we would like to judge to see if two patches are approximately equal if by applying shape-preserving transformation, one patch nearly coincides with the other If this is the case, the local shape pair features of the representative point pairs would have similar values. In order to make the above happen, after extracting the point clouds, we have inserted a new process of normalizing orientation using PointSVD mentioned in III-A. Please note that in the FPFH mentioned in the previous section, this process is not performed because similar features can be naturally obtained due to the fact that the FPFH keeps the angular relationship even when the 3D shape is rotated or is translated. The local shape pair feature we extract from the last 3D convolutional layer is of size $1 \times 1 \times 1 \times M$, where M is the number of feature maps, which can be interpreted as an M dimensional vector

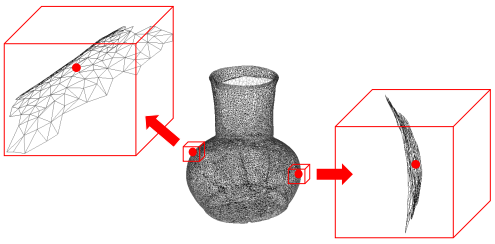

Fig. 4. An example of a pair of patches; we would like to judge to see if two patches are approximately equal if by applying shape-preserving transformation, one patch nearly coincides with the other

3) Voxel Generation from Local Features: Local features defined in the previous two sections cannot be directly fed to 3D CNN. First, we randomly select K representative points on any given 3D model. Once a representative point is selected, local feature vectors at the point are computed. Since the local feature vectors are sparse, we apply Principal Component Analysis (PCA) to feature vectors in order to reduce their dimension. Subsequently, we convert the reduced feature vectors to 3D voxels.

After repeating the computation of local features at the representative points, we generate voxels of size $L\ timesL\ timesL$. It should be noted that when a voxel is generated, ordinarily we expect each voxel has one local feature. However, there are two cases needed for special attention: (1) a voxel has two or more local features, and (2) a voxel has no local features. For case (1), we take the average of the local features for each voxel, and the result is set as the local feature. For case (2), we set the voxel value to 0 vector.

D. Partial Shape Feature

The partial shape feature is defined as the output of the second to the last fully connected layer (FC) of 3D CNN after training, where 3D CNN is a 3D extension of the Convolutional Neural Network (CNN) used for images [9].

E. Similarity Computation

Let $\mathbf{q}$ be the feature vector of the search query, and let T be the set of feature vectors given by multiple viewpoints. Here we compute the cosine similarity between $\mathbf{q}$ and the feature vector $\mathbf{t} \in T$, which is the feature vector from the database to be searched. The computed cosine similarities are sorted in descending order, where we define the final similarity $S(\mathbf{q}, T)$ as shown in equation (3).

$$S(\mathbf{q}, T) = \max_{\mathbf{t}\in T} s(\mathbf{q}, \mathbf{t}) \tag{3}$$

$$s(\mathbf{q}, \mathbf{t}) = \frac{\mathbf{q}\cdot\mathbf{t}}{|\mathbf{q}||\mathbf{t}|}$$

We perform the ensemble of local angle and local shape pair features. For each local feature, the similarity between the search query and the search target is computed. Final similarity is defined by the average of all the similarities.

F. Retrieval

The similarity computation in Section III-E is performed for all of the search targets. We then sort the results in descending order. The search result is the output of the sorting.

IV. Experiments

In this section, we describe datasets in Section IV-A, the implementation details in Section IV-B, the evaluation measure in Section IV-D, the 3D CNN training method in Section IV-C, the comparison of our proposed method with previous methods in Section IV-E, and the results in Section IV-F.

A. Dataset

The dataset we use was SHREC 16 Partial dataset [16]. The dataset is divided into 6 classes, and contains 383 perfect 3-dimensional models as search targets, 192 queries. The query dataset has three major different types: Artificial (Fig. 5), Breuckmann (Fig. 6), and Kinect (Fig. 7).

Artificial is a complete model made by humans, where partial shapes of Artificial are generated by artificially cut the complete part. Thus, each partial shape has a clear cross section. Artificial is divided into two groups, Q25 and Q40. Q25 has a partial shape of 25% of the overall shape and 40% of Q40. Here we conducted an experiment using only Q40. Breuckmann is generated by SmartScan's Breuckmann Scanner which is a highly accurate range scanner. Kinect is generated by Microsoft's Kinect V2 sensor, which is a low accuracy range scanner. Breuckmann and KINECT data provided with three different view groups; View 1, View 2, and View 3. In our experiments, we employed only View 1.

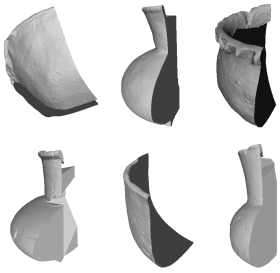

Fig. 5. An example of Artificial data. Cross section is artificially severed.

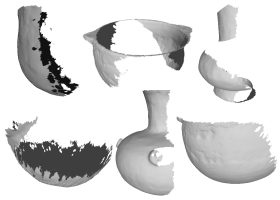

Fig. 6. An example of Breuckmann data acquired by high resolution 3D scanner

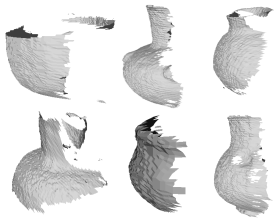

Fig. 7. An example of Kinect data. Scanned mesh is coarser than Breuckmann

B. Implementation Details

In this experiment, we pick up 500 "representative points", i.e. $k = 500$. For FPFH, we set 11 bins for the histogram of each angle, which amounts to 33 dimension, because we have three angles α, ϕ, and θ . For the radius parameter of FPFH, we set the radius $r = 3$ empirically to search for neighboring points.

On the other hand, for LSPF, we employed Analysis-by-Synthesis [21], 7Scenes [18], SUN3D [22], RGB-D Scenes v.2 [13], and Halber et al.[7], for training the 3D CNN on which 3DMatch is dependent. For the dimension of LSPF, we set $M = 512$, and the patch length $l = 0.3$.

For the training of PCA, we used 3d pottery dataset [12]. It should be noted that PCA is applied to both FPFH and LSPF. In case of FPFH, we use PCA to reduce dimension from 33 to 8, while in case of LSPF, we use PCA to reduce dimension from 512 to 8. In both cases, we confirm that the cumulative contribution rate becomes more than 95 %.

In Fig. 8, the red graph corresponds to the FPFH, while the blue graph corresponds to the LSPF.

On the other hand, we set voxel size $L = 8$. For the dimension of the vector we extract from 3D CNN, we chose the layer which was the second to the last, and its dimension was 256. At this time, L2 normalization was performed on the obtained partial shape features.

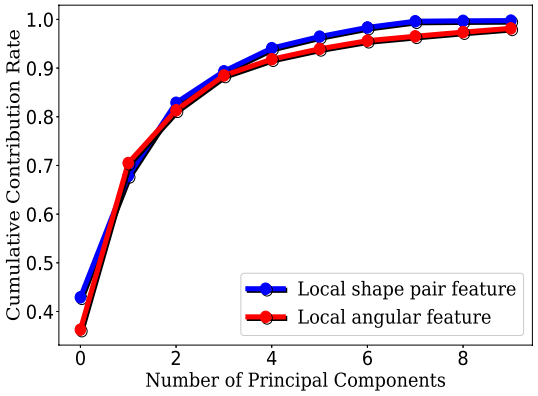

Fig. 8. Cumulative contribution rate

C. Training 3D CNN

Training of 3D CNN was performed using 3dpottery dataset. The 3dpottery dataset contains 36 classes, 1012 models. Of these classes, excluding "other" classes, we select classes that contain 5 or more models, and we randomly extract five models from those classes. As a result, we train 3D CNN using 22 class 110 models. Classes with extremely small numbers of data were eliminated, and in addition, the imbalance in the number of data for each class was resolved. For the data used for training, the same processing (i.e. (A), (B), (C), and (D) in Fig. 1) as the search target was performed. Since 38 partial shape models are extracted from one model, the final number of training data is 4,180.

The learning rate was 0.001, the optimizer was Adam, and the epoch was 10.

D. Evaluation Measure

As evaluation measure, we employ Nearest Neighbor (NN), First Tier (FT), Second Tier (ST), Discounted Cumulative Gain (DCG).

NN is referred to as P@1, and can be computed by

$$\text{Nearest Neighbor (NN)} \quad = \quad rel(1)$$

Here, $rel(k)$ is the number of 3D objects belonging to the same class as the search query included in the search high-order k items. NN is an evaluation measure indicating the relevance ratio of the top search result.

FT is also called R-precision. The formulae of FT and ST are as follows.

$$\text{First Tier (FT)} \quad = \quad \frac{rel(c-1)}{c-1}$$

$$\text{Second Tier (ST)} \quad = \quad \frac{rel\left\{2(c-1)\right\}}{c-1}$$

Here, c is the number of 3D models belonging to the same class as the search question. par DCG is an evaluation measure showing how much the ranking of correct answer data can be reproduced including rank. As the DCG value increases, more often relevant data are appearing around the top of the ranking.

$$\text{DCG}(i) \quad = \quad \begin{cases} G(1) & (i=1) \\ \text{DCG}(i-1) + \frac{G(i)}{\log_2(i)} & (otherwise) \end{cases}$$

$$\text{DCG@N} \quad = \quad \frac{\text{DCG(N)}}{1 + \sum_{j=2}^{N} \frac{1}{\log_2(j)}}$$

Here, i is the rank, G is the relevant data list, and N is the total number of 3D models.

E. Comparison with Previous Methods

For the previous methods for comparison, VoxNet [14], Tran et al [16], RSVP [5] are used, where VoxNet had $32 \times 32 \times 32$ binary voxels. To match the condition of VoxNel with our proposed method, when we use VoxNel, the processes (C) and (D) of Fig. 1 are replaced by VoxNet.

F. Experimental Results

Experimental results using Artificial is summarized in Table I, while experimental results using Breuckmann is summarized in Table II. Experimental results using Kinect is summarized in Table III.

G. Discussion

As shown in Tables II and III, we outperformed the previous methods in terms of FT, ST, and DCG@383. Meanwhile, for Artificial data, our method was not performed very well. We speculate that the reason for the low precision of Artificial lies in the extraction method of parts. Our proposed method for extracting partial shapes is best suited for a 3D object obtained by 3D scanners, because with such a device the partial shape is exactly corresponding to the visible area taken from the camera attached to these devices.

When comparing Breuckmann and Kinect which is data obtained from a 3D scanner, Breuckmann overall has higher accuracy. This is because the difference between the obtained local feature and the local feature obtained from the search object is large because Kinect's data has a rough mesh, such differences did not occur in Breuckmann. Ensemble was effective in many cases, but it was not effective when extreme difference in accuracy exists between the local angle feature and the local shape pair feature. We speculate that this is due to the fact that the results on the side with lower precision have been pulled.

TABLE I
RESULTS WITH ARTIFICIAL (Q40)

Method	NN	FT	ST	DCG@383
KAZE+VLAD	0.76	0.50	**0.74**	0.80
Tran et al	**1.00**	**0.52**	0.71	**0.82**
RSVP	0.90	0.49	0.71	**0.82**
VoxNet	0.33	0.36	0.65	0.73
Proposed(LAF)	0.62	0.48	**0.74**	0.81
Proposed(LSPF(w/o PointSVD))	0.24	0.19	0.37	0.62
Proposed(LSPF)	0.24	0.21	0.36	0.64
Proposed(Ensemble)	0.48	0.36	0.58	0.74

TABLE II
RESULTS WITH BREUCKMANN (VIEW1)

Method	NN	FT	ST	DCG@383
KAZE+VLAD	0.24	0.29	0.58	0.69
Tran et al	**0.56**	0.32	0.52	0.69
RSVP	0.36	0.33	0.55	0.68
VoxNet	0.36	0.38	0.67	0.75
Proposed(LAF)	0.48	**0.45**	**0.70**	**0.78**
Proposed(LSPF(w/o PointSVD))	0.40	0.35	0.60	0.73
Proposed(LSPF)	0.44	0.39	0.63	0.75
Proposed(Ensemble)	0.52	**0.45**	0.69	**0.78**

TABLE III
RESULTS WITH KINECT (VIEW1)

Method	NN	FT	ST	DCG@383
KAZE+VLAD	0.20	0.24	0.55	0.66
Tran et al	**0.60**	0.40	0.61	0.76
RSVP	0.08	0.21	0.49	0.62
VoxNet	0.48	0.37	0.61	0.75
Proposed(LAF)	0.44	0.38	0.63	0.74
Proposed(LSPF(w/o PointSVD))	0.32	0.35	0.57	0.72
Proposed(LSPF)	0.48	0.37	0.59	0.74
Proposed(Ensemble)	0.48	**0.43**	**0.67**	**0.78**

V. CONCLUSION

In this paper, we propose a 3D partial shape retrieval method using 3D CNN with two different local feature vectors as input after dimensional reduction with PCA. Experimental results demonstrate that our proposed method is particularly effective for 3D partial shape model2 acquired by 3D scanners.

Future research includes enhancement of the network structure that can deal with 3D volumetric data in addition to the current 3D CNN we have employed, exploration of additional local features, and the additional experiments using other 3D datasets such as mechanical parts and 3D scenes where "partial 3D shape retrieval" is highly required.

ACKNOWLEDGMENT

A part of this research was carried out with the support of the Grant-in-Aid for Scientific Research (B) (issue number 17H01746).

REFERENCES

[1] Kinect. https://en.wikipedia.org/wiki/Kinect.

[2] Paul Bratley and Bennett L. Fox. Algorithm 659: Implementing sobol's quasirandom sequence generator. *ACM Trans. Math. Softw.*, 14(1):88–100, March 1988.

[3] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a "siamese" time delay neural network. In *Proceedings of the 6th International Conference on Neural Information Processing Systems*, NIPS'93, pages 737–744, San Francisco, CA, USA, 1993. Morgan Kaufmann Publishers Inc.

[4] H. Dutagaci, A. Godil, C. P. Cheung, T. Furuya, U. Hillenbrand, and R. Ohbuchi. SHREC'10 Track: Range Scan Retrieval. In Mohamed Daoudi and Tobias Schreck, editors, *Eurographics Workshop on 3D Object Retrieval*. The Eurographics Association, 2010.

[5] T. Furuya, S. Kurabe, and R. Ohbuchi. Randomized sub-volume partitioning for part-based 3d model retrieval. In *Proceedings of the 2015 Eurographics Workshop on 3D Object Retrieval*, 3DOR, pages 15–22, Aire-la-Ville, Switzerland, Switzerland, 2015. Eurographics Association.

[6] A. Godil, H. Dutagaci, B. Bustos, S. Choi, S. Dong, T. Furuya, H. Li, N. Link, A. Moriyama, R. Meruane, R. Ohbuchi, D. Paulus, T. Schreck, V. Seib, I. Sipiran, H. Yin, and C. Zhang. Range scans based 3d shape retrieval. In *Proceedings of the 2015 Eurographics Workshop on 3D Object Retrieval*, 3DOR, pages 153–160, Aire-la-Ville, Switzerland, Switzerland, 2015. Eurographics Association.

[7] Maciej Halber and Thomas A. Funkhouser. Structured global registration of RGB-D scans in indoor environments. *CoRR*, abs/1607.08539, 2016.

[8] Binh-Son Hua, Quang-Trung Truong, Minh-Khoi Tran, Quang-Hieu Pham, Asako Kanezaki, Tang Lee, HungYueh Chiang, Winston Hsu, Bo Li, Yijuan Lu, Henry Johan, Shoki Tashiro, Masaki Aono, Minh-Triet Tran, Viet-Khoi Pham, Hai-Dang Nguyen, Vinh-Tiep Nguyen, Quang-Thang Tran, Thuyen V. Phan, Bao Truong, Minh N. Do, Anh-Duc Duong, Lap-Fai Yu, Duc Thanh Nguyen, and Sai-Kit Yeung. RGB-D to CAD Retrieval with ObjectNN Dataset. In Ioannis Pratikakis, Florent Dupont, and Maks Ovsjanikov, editors, *Eurographics Workshop on 3D Object Retrieval*. The Eurographics Association, 2017.

[9] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(1):221–231, January 2013.

[10] Y. Kobayashi and M. Aono. Three-dimensional model retrieval method and three-dimensional model retrieval system, March 2017. Japanese Patent Application No.2017-029425, (in Japanese).

[11] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. 2015.

[12] Anestis Koutsoudis, George Pavlidis, Vassiliki Liami, Despoina Tsiafakis, and Christodoulos Chamzas. 3d pottery content-based retrieval based on pose normalisation and segmentation. *Journal of Cultural Heritage*, 11(3):329 – 338, 2010.

[13] K. Lai, L. Bo, and D. Fox. Unsupervised feature learning for 3d scene labeling. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3050–3057, May 2014.

[14] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Pittsburgh, PA, September 2015.

[15] Robert Osada, Thomas Funkhouser, Bernard Chazelle, and David Dobkin. Shape distributions. *ACM Trans. Graph.*, 21(4):807–832, October 2002.

[16] I Pratikakis, MA Savelonas, Fotis Arnaoutoglou, G Ioannakis, Anestis Koutsoudis, T Theoharis, MT Tran, VT Nguyen, VK Pham, HD Nguyen, et al. Shrec16 track: Partial shape queries for 3d object retrieval. *Proc. 3DOR*, 1(8), 2016.

[17] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. Fast point feature histograms (fpfh) for 3d registration. In *Proceedings of the 2009 IEEE International Conference on Robotics and Automation*, ICRA'09, pages 1848–1853, Piscataway, NJ, USA, 2009. IEEE Press.

[18] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon. Scene coordinate regression forests for camera relocalization in rgb-d images. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2930–2937, June 2013.

[19] I. Sipiran, R. Meruane, B. Bustos, Tobias Schreck, H. Johan, B. Li, and Y. Lu. SHREC'13 Track: Large-Scale Partial Shape Retrieval Using Simulated Range Images. In Umberto Castellani, Tobias Schreck, Silvia Biasotti, Ioannis Pratikakis, Afzal Godil, and Remco Veltkamp, editors, *Eurographics Workshop on 3D Object Retrieval*. The Eurographics Association, 2013.

[20] A. Tatsuma, Y. Seki, M. Aono, and R. Ohbuchi. Shape similarity search of three-dimensional model based on multiple fourier spectrum representation. *IEICE Transactions*, 91-D(1):23–36, jan 2008. (in Japanese).

[21] Julien P. C. Valentin, Angela Dai, Matthias Nießner, Pushmeet Kohli, Philip H. S. Torr, Shahram Izadi, and Cem Keskin. Learning to navigate the energy landscape. *CoRR*, abs/1603.05772, 2016.

[22] J. Xiao, A. Owens, and A. Torralba. Sun3d: A database of big spaces reconstructed using sfm and object labels. In *2013 IEEE International Conference on Computer Vision*, pages 1625–1632, Dec 2013.

[23] Andy Zeng, Shuran Song, Matthias Nießner, Matthew Fisher, Jianxiong Xiao, and Thomas Funkhouser. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In *CVPR*, 2017.