applied sciences

MDPI

Article

A 2D-View Depth Image- and CNN-Based 3D Model Identification Method

Yiyu Hong [1] and Jongweon Kim [2,*]

1 Department of Copyright Protection, Sangmyung University, Seoul 03016, Korea; hongyiyu@cclabs.kr
2 Department of Electronics Engineering, Sangmyung, University, Seoul 03016, Korea
* Correspondence: jwkim@smu.ac.kr

Received: 9 August 2017; Accepted: 22 September 2017; Published: 25 September 2017

Abstract: With the rapid development of three-dimensional (3D) technology and an increase in the number of available models, issues with copyright protection of 3D models are inevitable. In this paper, we propose a 2D-view depth image- and convolutional neural network (CNN)-based 3D model identification method. To identify a 3D model, we first need an adequate number of the modified versions that could be made by copyright infringers. Then, they can be represented by a number of 2D-view depth images that are captured from evenly distributed vertices on a regular convex polyhedron. Finally, a CNN is trained by these depth images to acquire the capability of identifying the 3D model. The experiment carried out with the dataset of Shape Retrieval Contest 2015 (SHREC'15): *Non-Rigid 3D Shape Retrieval* shows the practicability of our method, which yields 93.5% accuracy. The effectiveness of the proposed method is demonstrated via evaluation in the latest standard benchmark SHREC'17 *Deformable Shape Retrieval with Missing Parts*. It clearly shows superior or comparable performance to state-of-the-art methods, shown by the fact that it is in the top three of the 11 participating methods (without counting different runs).

Keywords: 3D-model identification; deep convolutional neural network; depth image

1. Introduction

As three-dimensional (3D) printing [1,2] and modeling grow in popularity, many 3D models are being distributed and circulated through the Internet. Like other digital multimedia (audio, video, image, etc.), copyright infringement [3,4] of 3D models is also unavoidable. There are two general methods [5] that can be used to protect copyright interests from infringement: digital watermarking and fingerprinting.

Digital watermarking is a technique that adds a visible or invisible message to a host´s digital content. If the message is discovered in suspected content, the content can be considered as duplicated, and the ownership of the content can be confirmed. However, on applying digital watermarking techniques, damage to the quality of the host content is inevitable because the method of embedding the message is performed by manipulating the data of the host content. This is why most content producers do not like to use digital watermarking to protect their copyrights. Thus, when applying digital watermarking, controlling the trade-off between robustness and imperceptibility is fairly important.

In contrast to digital watermarking, fingerprinting does not add any message to the host content, which means that no damage occurs. Instead, it extracts unique features from the contents by analyzing the inherent properties of the contents. Ideally, the features are required to be invariant when some modifications are made to the contents. After extracting features from both the host content and the suspect content, the ownership can be proved by comparing the similarity of the two sets of features. Generally, compared with digital watermarking, fingerprinting requires a longer processing time because it has to measure similarity with the information stored in the database. If the database has thousands of millions of items of data, fingerprinting will be time consuming and also imply high financial costs with respect to purchasing and maintaining servers.

In this paper, we propose a method using the fingerprinting technique to identify copyrights for 3D models. Inspired by several previous works [6–19], our method is mainly based on 2D view depth imaging and the convolutional neural network (CNN). Hence, here we give a brief review about methods for view-based 3D model similarity measurement and CNN.

1.1. View-Based 3D Model Similarity Measurement

Basically, view-based methods for measuring the similarity of 3D models follow the idea that "if two 3D models are similar, they also look similar from all viewing angles." View-based methods convert the three-dimensional task into a two-dimensional task so that it can employ and benefit from the research that has been performed previously for 2D images, like image local features [20–22] and image processing [23]. In terms of the initial stage of the view-based methods, the most famous is the light field descriptor (LFD) [6]. The LFD method first takes multiple silhouette images from evenly distributed vertices of a dodecahedron over a hemisphere, and the silhouettes are encoded by its Zernike moments and Fourier coefficients. Then, the dissimilarity between the two 3D models is obtained by exhaustively searching the minimum distance of features that are extracted from all possible situations (60 different camera settings). In more recent research based on the LFD framework, many researchers have studied a variety of pose normalization approaches, strategies taking 2D images, and descriptors to improve performance.

Pose normalization: Generally, a 3D model has arbitrary position, scale, and orientation. Thus, most view-based methods require a proper pose normalization approach such as preprocessing before taking depth images. Scale and position are usually normalized by transferring the center of mass (average of point coordinates) of 3D models to the origin and scaling the maximum polar distance of the points to a certain value. For orientation, there is principle component analysis (PCA)-based [7–9] alignment, symmetry-based alignment [10], projection area-based alignment [11], and so on. Among these approaches, the PCA-based alignment approach is most widely used. These approaches could reduce possible alignments of 3D models to decrease the computation cost of view matching. For instance, Lian et al. [9] employed a PCA-based and rectilinearity-based method to align 3D models into 24 possible pose permutations so that they only need compare 24 matching pairs for two models.

Strategies using 2D images: There is much research [8,9,12–15] utilizing depth images as views for 3D models instead of simple silhouettes. This is because the silhouette is not suitable for discriminating concave geometries, whereas the depth image is capable of carrying useful 3D geometric information. Daras and Axenopoulos [14] utilized both silhouette and depth images to represent a 3D model. With respect to camera settings, the LFD utilized a dodecahedron to set the position of virtual camera, while other authors [8,9] have tried some different platonic solids (octahedron, icosahedron) and their subdivision to take various numbers of images. Papadakis et al. [15] represented 3D models as a set of panoramic views by projecting the 3D model to the lateral surface of a cylinder.

Descriptors: The elevation descriptor [12] extracts six views that contain the altitude information of the 3D model from six directions. The depth line descriptor [13] extracts depth lines from captured depth images and transforms them into sequences. The difference between depth line descriptors is measured by the dynamic programming method. The compact multi-view descriptor [14] describes each captured view by employing 2D polar Fourier transform, 2D Zernike moments, and 2D Krawtchouk moments. With the great success of the 2D local features in the computer vision field, Ohbuchi et al. [8] proposed a view-based method called BF-SIFT, which employs the scale-invariant feature transform (SIFT) [20] with the bag-of-features (BoF) technique [24]. This method first constructs a visual word dictionary (codebook) off-line by k-means clustering of SIFT features extracted from the depth images of 3D models in the training database. Given a 3D model, after taking depth images and extracting SIFT features, the 3D model can be represented by a word histogram that integrates the

SIFT features via a codebook. The distance between the two word histograms of two 3D models is computed by Kullback–Leibler divergence. Lian et al. [9] also proposed a similar view-based method called Clock Matching (CM)-BOF. The major difference is that BF-SIFT represents a 3D model by one histogram, while CM-BOF represents each depth image of a 3D model as one histogram. Although CM-BOF performs better than BF-SIFT, it may require more computation time because of the set-to-set histogram matching strategy for calculating dissimilarity.

1.2. CNN

Over the past few years, use of the deep learning technique has exploded. Much of this has to do with the wide availability of GPUs that make parallel processing ever faster, cheaper, and more powerful. It also has to do with large amounts of digital data (image, text, audio, etc.) practically available. The convolutional neural network (CNN) is a type of deep learning algorithm that has become a hot research topic in many scientific fields, such as image classification [25], face verification [26], and speech recognition [27]. Full CNN architecture can be formed by stacking a number of convolutional layers, pooling layers, and fully-connected layers in an order. It is trained by forward propagation and backward propagation. The forward propagation means that training data go through a series of convolution, activation function, pooling operations, and fully-connected layers to output probabilities. After calculating the total error between target probability and output probability, backpropagation is used to calculate the gradients of the error with respect to all weights in the CNN and uses gradient descent to update the trainable parameters to minimize the output error.

2. Materials and Methods

2.1. The Proposed Method

Let us assume that there are 10 different kinds of 3D models in a database. If one of them is modified and illegally distributed, how will humans identify the 3D model when it is found? In terms of the details of the procedure, first of all, individuals need to remember all 10 three-dimensional models in the database. When a suspected 3D model is found, humans need to look at it from several different viewpoints, and for each view, they will have a virtual prediction of it in the brain. By averaging these predictions, a final decision about whether or not it is the 3D model in the database will be made.

To simulate this procedure, we take depth images uniformly around a 3D model to replace the work of human eyes and let the CNN replace the work of human brain. However, given a 3D model, if we only input the one 3D model for training the CNN, it is hard to identify the 3D model from its modified version, which possibly has significant pose deformation or missing parts. Hence, we arrive at the idea that we could manually modify a 3D model to produce a quantity of possible modified versions, which can be explained as extending data, and then use all of these versions for training a CNN to enable it to identify a 3D model from a modified version not included in the training data. The experimental results in the Section 3, using the datasets of SHREC'15 *Non-rigid 3D Shape Retrieval* [16], proved the practicability of our idea.

Our method can be broken down into a training part and a test part. In the training part, all captured depth images of 3D models in the database and their modified versions with labels are input into the CNN for training. After training, in the test part, depth images of a suspected 3D model are input to the already-trained CNN to output prediction for each view. Finally, the predictions are averaged to derive the identification result by setting a threshold to the maximum predicted 3D model. If the maximum prediction is larger than the threshold, we can consider that the maximum predicted 3D model in the database is identified. Otherwise, the suspected 3D model can be seen as irrelevant. Figure 1a shows the pipeline of our method for 3D model identification. In order to expand experiments, we have participated in the SHREC'17 *Deformable Shape Retrieval with Missing Parts* [28] track by simply considering the output of the network predictions as the feature vector as shown in

Figure 1b. Then, the distance of two 3D models can be computed using the Euclidean distance of the corresponding feature vectors.

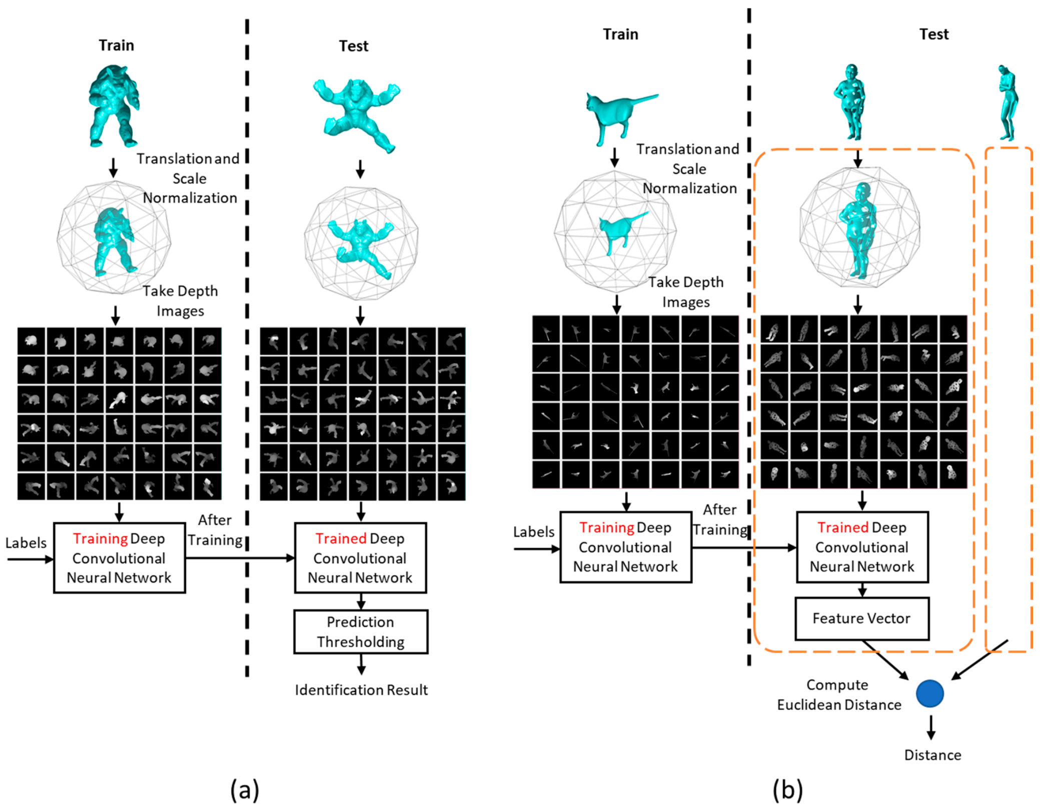
Train
Test
Translation and Scale Normalization
Take Depth Images
Labels
Training Deep Convolutional Neural Network
After Training
Trained Deep Convolutional Neural Network
Prediction Thresholding
Identification Result
Feature Vector
Compute Euclidean Distance
Distance
(a)
(b)

Figure 1. (**a**) The three-dimensional (3D) model identification pipeline of our method experimenting on SHREC'15 *Non-rigid 3D Shape Retrieval*; (**b**) Three-dimensional model retrieval pipeline of our method experimenting on SHREC'17 *Deformable Shape Retrieval with Missing Parts*.

To take depth image, the mass center of the 3D model is translated to the origin, and the maximum polar distance of the points is scaled to one. Then, a regular convex polyhedron in which the vertices are distributed uniformly in all directions is generated, and the center of the polyhedron is also located in the origin. The virtual cameras are set up at each vertex of the polyhedron and pointed at the origin to take depth images of 3D models. We compared numbers of views (6, 18, 42, 66) in our experiments by utilizing an octahedron, icosahedron, and their subdivided versions, as shown in Figure 2.

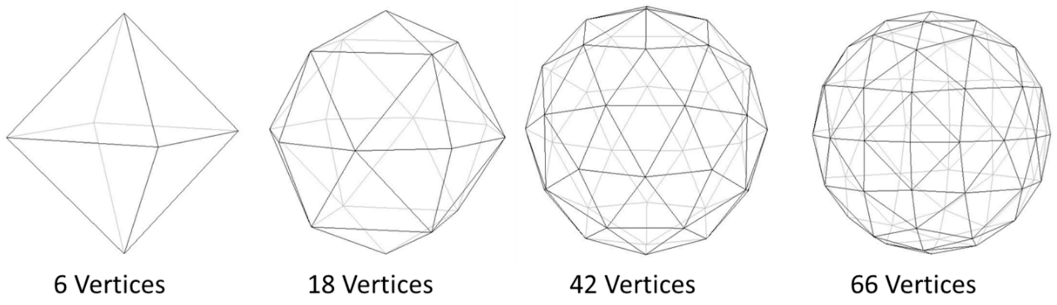

Figure 2. Regular convex polyhedrons that utilized in our method to take depth images of 3D models.

With respect to CNN architecture, we choose the GoogLeNet [17] (also referred to as Inception v1), the winning architecture in the ImageNet 2014 classification task. Although other architectures that perform better than GoogLeNet have been developed, such as Inception-v4 [18] and ResNet [19], our 4-GB GPU used for experiments set a hardware limit on the type of architecture. The GoogLeNet is repeated spatially by the "Inception Module". Each "Inception Module" is made up of 1 × 1 convolutions, 3 × 3 convolutions, 5 × 5 convolutions and 3 × 3 max pooling. The 1 × 1 convolutions decrease the computation cost and also capture the correlated features of an input image in the same region. The 3 × 3 and 5 × 5 convolutions capture features at larger scales. Feature maps which are being produced by all the convolutions are concatenated to form the output. One of the main beneficial aspects of this architecture is that it allows for increasing the number of units at each stage significantly without an uncontrolled blow-up in computational complexity. Hence, the CNN can be designed not only very deeply but can also be efficiently trainable.

2.2. Dataset of SHREC'15 Non-Rigid 3D Shape Retrieval

Since the most common modification to 3D models is pose deformation, we tested our identification method on the dataset of SHREC'15 *Non-Rigid 3D Shape Retrieval*. The benchmark is focused on the comparison of non-rigid 3D shape retrieval methods which have many pose-deformed 3D models. The dataset consists of 1200 3D models derived from 50 original models. For each original model, they used a 3D modeling tool to build its skeleton and then articulate it to generate the 23 deformed versions. Hence, the dataset can be classified into 50 categories, each with 24 models. Figure 3a shows some examples of the 3D models in the dataset. In our identification experiment, for each category, we used 20 for training the CNN and used the remaining four models with the 200 extra 3D models (see Figure 3b), which were randomly collected from *Thingiverse* (a website dedicated to the sharing of user-created digital design files—https://www.thingiverse.com/) to test our method. The 200 extra 3D models are available at http://cclabs.smu.ac.kr/research/datasets-for-2d-view-depth-image-and-cnn-based-3d-model-identification-method/.

As an identification system, it also should have ability to correctly reject irrelevant input queries, and the added 200 extra 3D models play an important role in our identification experiment to verify the ability.

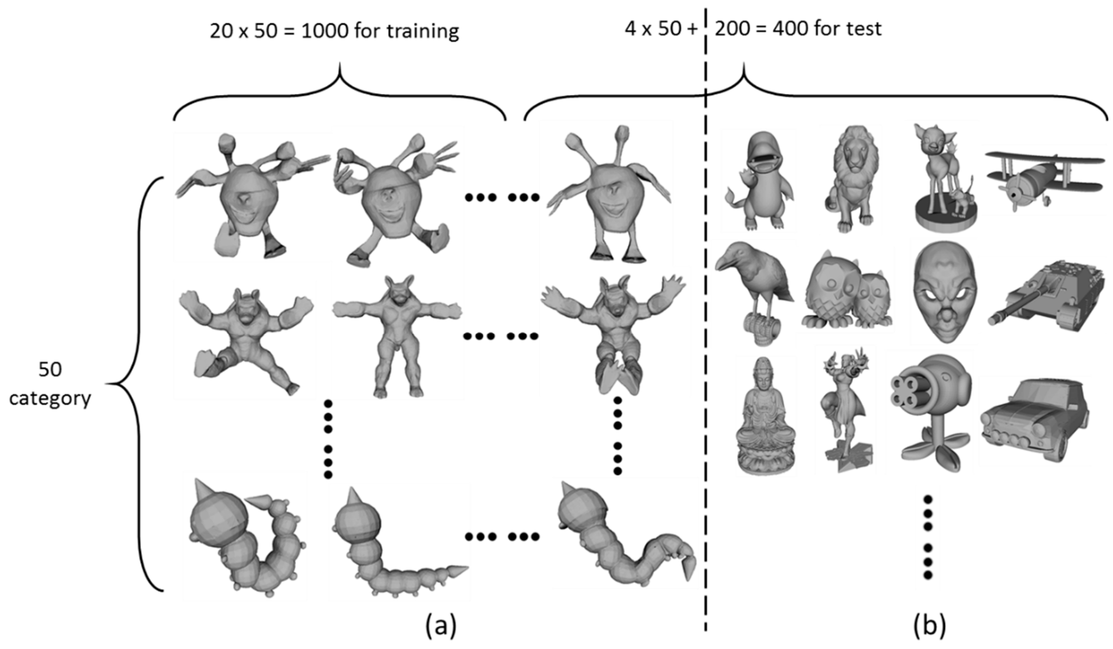

Figure 3. Examples of 3D models used in our identification experiments. (**a**) Examples of 3D models in the SHREC'15 *Non-Rigid 3D Shape Retrieval*; (**b**) Examples of 200 extra 3D models collected from *Thingiverse*.

2.3. The Dataset of SHREC'17 Deformable Shape Retrieval with Missing Parts

The task of the SHREC'17 *Deformable Shape Retrieval with Missing Parts* track is to retrieve similar 3D shapes with different partiality patterns, additionally undergoing a non-rigid transformation. Eight research groups participated, including us, and the result shows our method performed quite well among the 11 methods in total (without counting different runs) that were evaluated in the contest. There are two challenges. One is called "irregular holes," and the other is "range data." The 3D models used for irregular holes have a random scale, random position in space, and span several resolutions, especially with irregular holes on the surface of 3D models. In range data, the 3D models are triangulated from 2.5D snapshot by a virtual orthographic camera which is placed around the original 3D models. See Figure 4 for examples of two kinds of 3D models. Both datasets fall into 10 categories. The irregular holes dataset consists of 1216 training 3D models and 1078 test 3D models, and the range data dataset consists of 1082 training 3D models and 882 test 3D models. The ground-truth labels are provided for the training set.

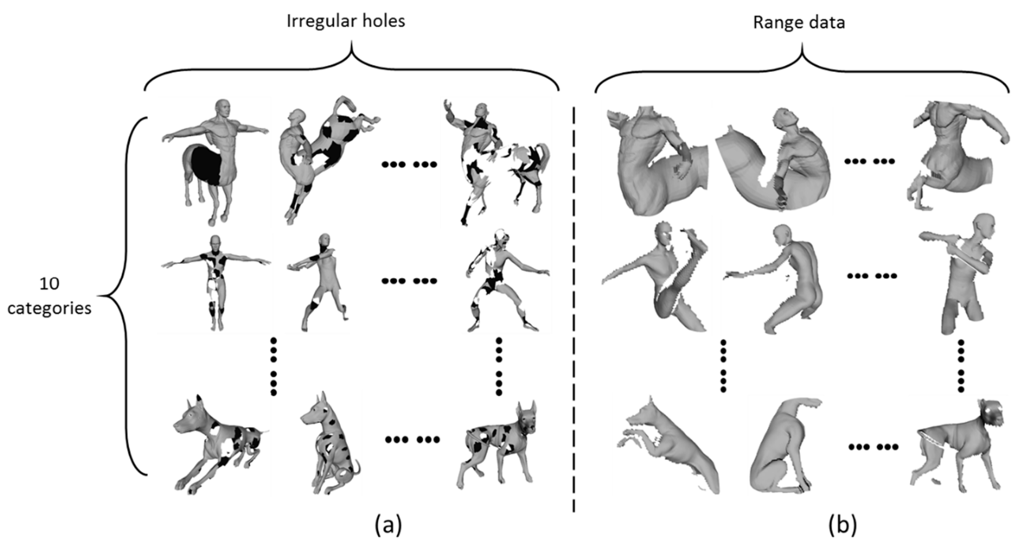

Figure 4. Examples of 3D models in SHREC'17 *Deformable Shape Retrieval with Missing Parts*. (**a**) Irregular holes; (**b**) Range data.

3. Experimental Results

In this section, we show the experimental results of proposed method on the datasets that are mentioned above. There are mainly two parts of experiments, one is for 3D model identification and the other is for deformable 3D model retrieval with missing parts. The latter is to perform a comparative evaluation with state-of-the-art methods. Note that the experimental results of this part are taken from SHREC'17 *Deformable Shape Retrieval with Missing Parts* where we submitted our test results; the contest organizer received test results of all participants and reported the results of the combination.

If not specified, we adopt the following setup throughout our experiments. The number of depth images of each 3D model is 42, captured depth image resolution is 224×224, the pixel value of depth image is zero-centered and then scaled by its standard deviation as a pre-processing step, and the batch size is set to 64. With respect to CNN, the weights in each layer are initialized from a zero-mean normal truncated distribution with standard deviation 0.02, and the neuron biases are all initialized with constant 0. Ten percent of the training data is used for validation, and the training process is stopped when the validation accuracy has no further growth at around 30~60 epochs.

All the experiments below were run on a desktop with a 3.60-GHz i7-4790 CPU, 8 GB of DDR3 RAM, and a GeForce GTX 960 4 GB Graphic Card. GoogLeNet was built by TFLearn, which is a deep learning library featuring a higher-level API for Tensorflow at https://github.com/tflearn built on top of Tensorflow, an open-source software library for machine intelligence (https://www.tensorflow.org/). Depth images were taken with the MATLAB 3D Model Renderer (Matlab function for rendering textured 3D models—http://www.openu.ac.il/home/hassner/projects/poses/).

3.1. 3D Model Identification

Here, we test the 3D model identification performance of our method on the data that is described in Section 2.2. We will first show the identification results on default settings of our method, and then we will demonstrate how the number of depth images and the number of 3D models in each category that used for training affect identification results.

As described in Section 2.2, from 50 categories in dataset of SHREC'15 *Non-Rigid 3D Shape Retrieval*, we use 20 pose-deformed 3D models for each category, which are 1000 models in total, to train the GoogLeNet. Each 3D model is represented by 42 depth images, and each depth image is assigned the label of the corresponding category. These depth images and corresponding labels are put into the GoogLeNet, for which the last softmax layer is changed to output 50-way prediction values. The training used momentum with a decay of 0.9, a learning rate of 0.001, and every epoch is decayed using an exponential rate of 0.96 with a dropout ratio in the last fully connected layer set to 0.5.

After the GoogLeNet is trained, the remaining four pose-deformed 3D models in each category and 200 irrelevant 3D models are used for testing. Each test 3D model is also represented by 42 depth images and input to the trained GoogLeNet, which will output 42×50 prediction values. These are averaged on each dimension to generate a 50-dimensional final prediction. Then, by setting a threshold to the final prediction, we can yield an identification result.

Regarding the experiment in this section as from the point of view of the 3D model identification system for copyright protection, it can be considered that there are 50 original copyrighted 3D models in the database of the identification system. Thus, in the following article, we define the query 3D model that is deformed version of an original copyrighted 3D model as Q_c and define the query that is irrelevant to the original copyrighted 3D models as Q_i.

To evaluate performance, we use True Positive (tp), True Negative (tn), False Positive (fp), False Negative (fn), and Accuracy (Acc).

- True Positive: If a query 3D model is a deformed version of an original copyrighted 3D model (Q_c) and the identification system correctly identifies the query 3D model.
- True Negative: If a query 3D model is irrelevant (Q_i) and the identification system correctly identifies it as irrelevant.
- False Positive: If a query 3D model is irrelevant (Q_i) but the identification system wrongly identifies it as one of the 3D models in the database, or if a query 3D model is a deformed version of an original copyrighted 3D model (Q_c), but the identification system identifies it as a wrong one in the database, which means the query is not the deformed version of the 3D model that was identified by the system.
- False Negative: If a query 3D model is a deformed version of an original copyrighted 3D model (Q_c), but the identification system identifies it as irrelevant.
- Accuracy: $Acc = \frac{tp+tn}{tp+tn+fp+fn}$

The calculation process is written in pseudo code in Table 1. Table 2 shows the identification results that using the 400 test 3D models, and Figure 5 shows the corresponding graph between accuracy and threshold. From the table and graph, we can see that the method works best when the prediction threshold is 0.7, yielding 93.5% accuracy: 183 3D models are correctly identified from the 200 Q_c 3D models and 191 irrelevant 3D models are correctly rejected from the 200 Q_i 3D models. If users do not want to miss any deformed versions of their 3D models from a target database, they can set a lower threshold, such as 0.4, which correctly identified 199 out of the 200 deformed 3D models. However, there would be more false positive cases. In other words, without the premise of an ideal 3D model identification system existing, users are free to select the threshold to adjust the performance of the identification system to meet their own needs. From the results, we can conclude that the proposed identification method works well on the 400 test 3D models.

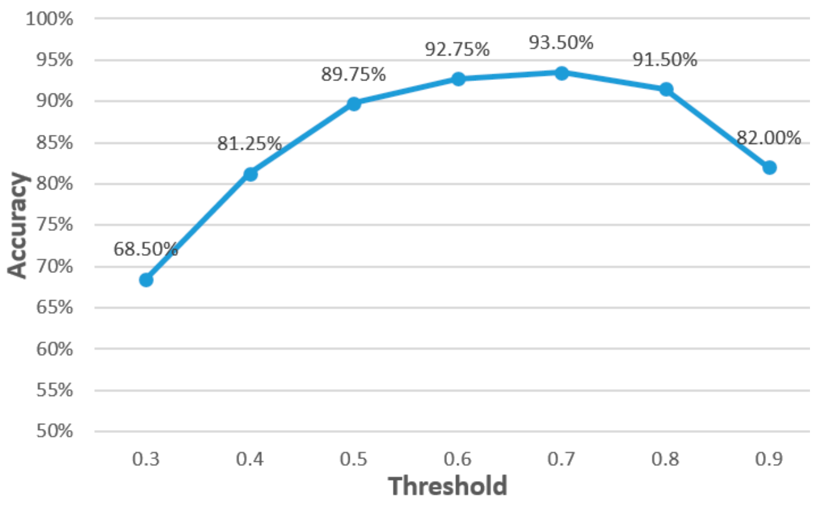

Figure 5. Graph of 3D model identification results with default settings and different thresholds.

Table 1. Pseudo code for evaluating the performance of 3D model identification.

```
1:  **Input** :$Q_c$
2:  $p = avg(GoogLeNet(Q_c))$ /* get prediction values*/
3:  **if** $max(p) > Threshold$
4:    **if** $argmax(p) == Ground\ Truth\ Label$
5:      $tp = tp + 1$     /*correctly identified the 3D model*/
6:    **else**
7:      $fp = fp + 1$   /* wrongly identified*/
8:    **end**
9:  **else** /* copyrighted one is missed by the system*/
10:   $fn = fn + 1$
11: **end**
12:
13: **Input** :$Q_i$
14: $p = avg(GoogLeNet(Q_i))$
15: **if** $max(p) < Threshold$
16:   $tn = tn + 1$   /*correctly rejected irrelevant 3D model*/
17: **else**
18:   $fp = fp + 1$   /* wrongly identified*/
19: **end**
```

Table 2. Table of 3D model identification results with default settings and different thresholds.

Threshold	True Positive	True Negative	False Positive	False Negative	Accuracy
0.3	199	75	125	1	68.50%
0.4	199	126	74	1	81.25%
0.5	198	161	39	2	89.75%
0.6	193	178	22	7	92.75%
0.7	183	191	9	17	93.50%
0.8	172	194	6	28	91.50%
0.9	131	197	3	69	82.00%

The Figure 6 shows how the number of depth images influence the identification accuracy. As described in Section 2.1, we took 6, 18, 42, and 66 depth images from vertices of regular convex polyhedrons, and the same numbers of depth images were used in both the training and test parts to yield identification accuracy. From the graph, we can see that the accuracy is improved by the increasing number of depth images. The improvement also slows down as the number of depth images increases and there is nearly no gap between 42 and 66 images when they are at the highest accuracies.

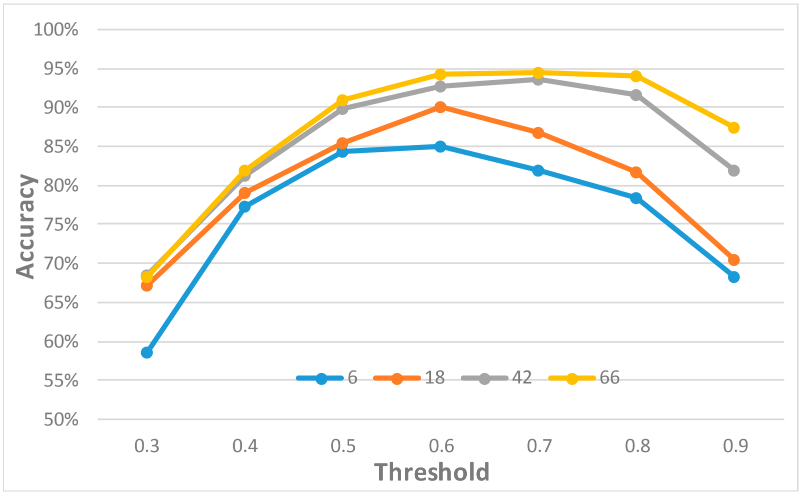

Figure 6. The influence of number of depth images on the identification accuracy.

The Figure 7 shows the influence of the number of 3D models used for training on identification accuracy. In this experiment, the 400 test 3D models remained unchanged, but we reduced the training 3D models in each category by 5, 10, and 15, representing 250, 500, and 750 training 3D models in total, respectively. From the graph, it can be observed that the identification accuracy is very low when using five 3D models. However, there is a great improvement when the number of 3D models for training jumps from 5 to 10. The highest accuracy improves about 3% from 10 to 15 models, but no improvement is produced from 15 to 20. This experiment indicates that, for a given original 3D model, 15 properly pose-deformed 3D models may be enough for our method to learn its other pose deformations, and this number can be used for 3D model identification.

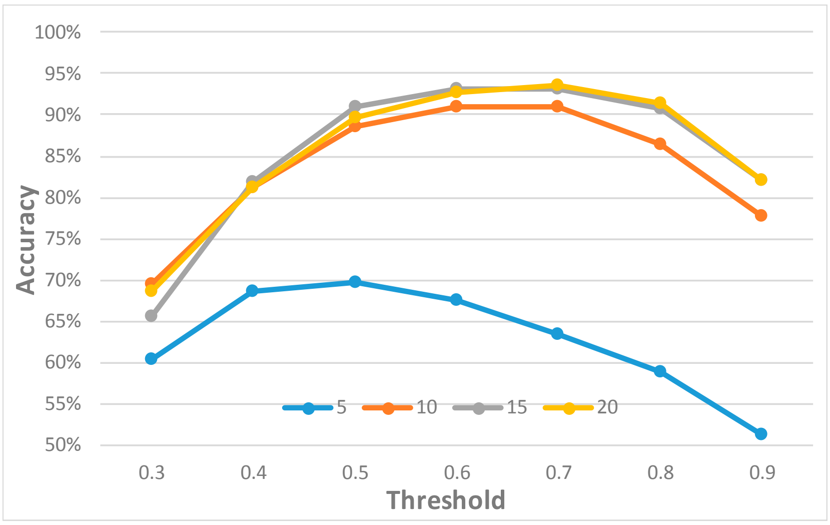

Figure 7. The influence of the number of 3D models used for training on the identification accuracy.

3.2. Deformable 3D Model Retrieval with Missing Parts

In this section, we present the comparative performance of the proposed method on SHREC'17 *Deformable Shape Retrieval with Missing Parts.* We named our method the 2D-view depth image- and CNN-based method (2VDI-CNN) and participated this test.

As described in Section 2.2, there are two challenges in the contest: irregular holes and range data, and both have 10 categories. Hence, in the training part, for each challenge we trained a GoogLeNet and also used default settings of our method, except the last softmax layer of the GoogLeNet was changed to output 10-way prediction values. The training used momentum with a decay of 0.9 and a learning rate of 0.01, and every epoch is decayed using an exponential rate of 0.96 with the dropout ratio in the last fully connected layer set to 0.1. In the test part, as the test is about comparing retrieval accuracy, each 3D model is supposed to be represented by feature vectors, and it is simply generated in our method by averaging prediction values of 42 depth images of a 3D model in each dimension. Thus, a 3D model is represented by a 10-dimensional feature vector, and the dissimilarity between two 3D models is given by the Euclidean distance of the corresponding feature vectors. In the end, for each challenge we were asked to calculate the dissimilarity between each pair of 3D models in the test set to construct a dissimilarity matrix $\left(d_{ij}\right)_{n\times n}$, where n is the number of 3D models in the test set and d_{ij} denotes the dissimilarity between the 3D model i and 3D model j.

With the matrices, retrieval performance is compared based on the precision-recall (P-R) curve and four quantitative measures: nearest neighbor (NN), first tier (1-Tier), second tier (2-Tier) and discounted cumulative gain (DCG).

- Nearest neighbor: The percentage of best matches that belong to the query's class.
- First tier and second tier: The percentage of models belonging to the query's class that appear within the top $(K-1)$ and $2(K-1)$ matches where the number of models in the query's class is K.
- Discounted cumulative gain: A statistic that weights correct results near the front of the list more than correct results later in the ranked list.
- Precision-recall curve: Precision is the ratio of retrieved models that are relevant to a given query, while recall is the ratio of relevant models to a given query that have been retrieved from the total number of relevant models. Thus, a higher P-R curve indicates better retrieval performance.

The four quantitative measures range from 0 to 1, and higher values indicate better retrieval performance. For more details about the metrics, we refer readers to [29].

In the contest, there are 11 participating methods (if including runs with different parameters, there are totally 15 algorithms), where six are supervised learning-based and the others are not learning-based. These methods also can be roughly divided into the Laplace–Beltrami operator-based method, the bag-of-word framework-based method, and the 2D view-based method. Readers can refer to [28] for a detailed description of each method.

Tables 3 and 4 shows the four quantitative measures for irregular holes and range data, respectively. Figure 8 shows P-R curves for all methods on the two challenges. In the irregular holes challenge, from the left side P-R curve graph in Figure 8, we can obviously see the top three performing methods among 11 competing methods (without counting different runs): DLSF is the best, followed by BoW+RoPS and our 2VDI-CNN. From Table 3, it can be observed that in terms of 1-tier, 2-tier, and DCG, 2VDI-CNN is ranked third after DLSF and BoW-RoPS. The NN is ranked second. In the range data challenge, from the right side P-R curve graph in Figure 8, we can see that 2VDI-CNN and BoW+RoPS are the best-performing methods among the six competing methods. From Table 4, it can be observed that 2VDI-CNN achieves top scores in terms of NN, 2-tier and DCG, except the 1-tier metric take the second highest score. In summary, the retrieval performance of our method is ranked third in the irregular holes challenge among 11 competing methods and ranked first in the range data challenge among six competing methods.

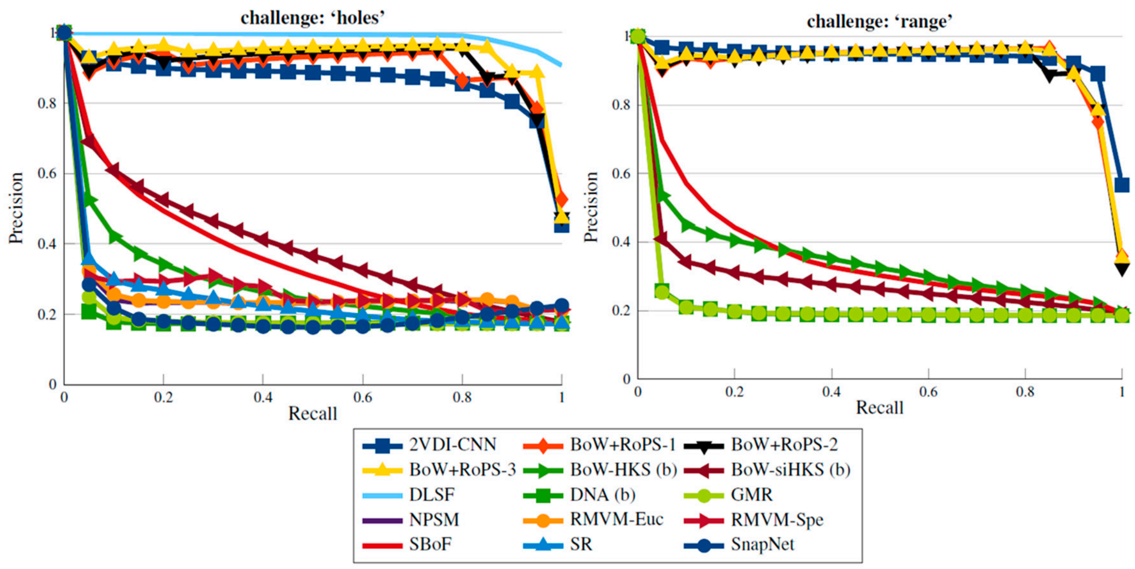

Figure 8. Precision-recall curves for all methods in the two challenges. Left graph is for irregular holes, right graph is for range data. 2VDI-CNN: 2D-view depth image- and CNN-based method.

Table 3. Retrieval performance for the *Irregular holes* challenge, sorted decreasingly by the neural network (NN) score. Best results are reported in bold. DCG: discounted cumulative gain.

Method	NN	1-Tier	2-Tier	DCG
DLSF	1.000	0.971	0.999	0.998
2VDI-CNN (ours)	0.906	0.818	0.937	0.954
SBoF	0.815	0.326	0.494	0.780
BoW-siHKS	0.710	0.370	0.566	0.790
BoW+RoPS-3	0.607	0.918	0.970	0.968
BoW+RoPS-1	0.597	0.877	0.963	0.956
BoW-HKS	0.578	0.261	0.436	0.725
RMVM-Euc	0.392	0.226	0.402	0.679
BoW+RoPS-2	0.380	0.894	0.965	0.955
NPSM	0.347	0.222	0.395	0.676
RMVM-Spe	0.251	0.228	0.410	0.676
SR	0.241	0.225	0.395	0.676
GMR	0.186	0.172	0.343	0.642
SnapNet	0.117	0.172	0.349	0.641
DNA	0.078	0.163	0.348	0.632

Table 4. Retrieval performance for the range data challenge, sorted decreasingly by NN score. Best results are reported in bold.

Method	NN	1-Tier	2-Tier	DCG
2VDI-CNN (ours)	0.969	0.906	0.977	0.980
SBoF	0.811	0.317	0.510	0.769
BoW+RoPS-2	0.643	0.910	0.962	0.962
BoW+RoPS-3	0.639	0.908	0.964	0.965
BoW-HKS	0.519	0.326	0.537	0.736
BoW+RoPS-1	0.515	0.915	0.959	0.960
BoW-siHKS	0.377	0.268	0.485	0.699
GMR	0.178	0.184	0.371	0.640
DNA	0.130	0.183	0.366	0.640

4. Conclusions

Aiming at copyright protection for 3D models, in this paper we propose a 2D-view depth image- and CNN-based 3D model identification method. The key idea is that if a 3D model needs to be protected, a sufficient number of modified versions of the 3D model that might be produced is first required. Then, they are represented by several depth images that are captured from evenly distributed vertices on a regular convex polyhedron. Next, those depth images with labels are used to train CNN to identify the 3D model.

The experiment carried out on the dataset of SHREC'15 *Non-Rigid 3D Shape Retrieval* and 200 irrelevant 3D models proved the practicability of our method for identifying pose-deformed 3D models which yield a 93.5% accuracy with default settings. Additional experiments indicate that with an increasing number of depth images, the identification accuracy is improved, though the improvement speed is slower. The experiment for investigating identification accuracy with the number of pose deformed 3D models used for training shows that 15 properly pose-deformed 3D models may be enough for our method to learn its other pose deformations. Moreover, the experiment carried out on SHREC'17 *Deformable Shape Retrieval with Missing Part* shows that the performance of our method is superior or comparable to the state-of-the-art methods.

In the future, we will study more advanced deep learning techniques to improve 3D model identification accuracy and expand test settings with other deformations like cropping, noise, etc.

Acknowledgments: This research was supported by the Institute for Information & Communications Technology Promotion (IITP) funded by the Ministry of Science, ICT and Future Planning (No. R0126-15-1024).

Author Contributions: Both authors contributed to the research work. Both authors designed the new method and planned the experiments. Jongweon Kim led and reviewed the research work. Yiyu Hong performed the experiments and wrote the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ishengoma, F.R.; Mtaho, A.B. 3D Printing Developing Countries Perspectives. *Int. J. Comput. Appl.* **2014**, *104*, 30–34. [CrossRef]
2. Pirjan, A.; Petrosanu, D.-M. The Impact of 3D Printing Technology on the Society and Economy. *J. Inf. Syst. Oper. Manag.* **2013**, *7*, 360–370.
3. Weinberg, M. What's the Deal with Copyright and 3D Printing? Available online: https://www.publicknowledge.org/files/What%27s%20the%20Deal%20with%20Copyright_%20Final%20version2.pdf (accessed on 7 August 2017).
4. Gupta, D.; Tarlock, M. 3D Printing, Copyright Challenges, and the DMCA. *New Matter* **2013**, *38*, 1–16. Available online: http://www.fbm.com/files/Uploads/Documents/3D%20Printing%20Copyright%20Challenges%20and%20the%20DMCA.pdf (accessed on 24 September 2017).
5. Milano, D. Content Control: Digital Watermarking and Fingerprinting. Available online: http://www.rhozet.com/whitepapers/Fingerprinting-Watermarking.pdf (accessed on 7 August 2017).
6. Chen, D.; Tian, X.; Shen, M. On Visual Similarity based 3D Model Retrieval. *Eurographics* **2003**, *22*, 223–232. [CrossRef]
7. Papadakis, P.; Pratikakis, I.; Perantonis, S.J.; Theoharis, T. Efficient 3D Shape Matching and Retrieval using A Concrete Radialized Spherical Projection Representation. *Pattern Recognit.* **2007**, *40*, 2437–2452. [CrossRef]
8. Ohbuchi, R.; Osada, K.; Furuya, T.; Banno, T. Salient Local Visual Features for Shape-Based 3D Model Retrieval. In Proceedings of the IEEE International Conference on Shape Modeling and Applications (SMI'08), Stony Brook, New York, NY, USA, 4–6 June 2008; pp. 93–102. [CrossRef]
9. Lian, Z.; Godil, A.; Sun, X. Visual Similarity based 3D Shape Retrieval using Bag-of-Features. In Proceedings of the IEEE International Conference on Shape Modeling and Applications (SMI'10), Aix-en-Provence, France, 21–23 June 2010. [CrossRef]
10. Chaouch, M.; Verroust-Blondet, A. Alignment of 3D Models. *Graph. Models* **2009**, *71*, 63–76. [CrossRef]

11. Johan, H.; Li, B.; Wei, Y. 3D Model Alignment based on Minimum Projection Area. *Vis. Comput.* **2011**, *27*, 565–574. [CrossRef]
12. Shih, J.L.; Lee, C.H.; Wang, J. A New 3D Model Retrieval Approach based on the Elevation Descriptor. *Pattern Recognit.* **2007**, *40*, 283–295. [CrossRef]
13. Chaouch, M.; Verroust-Blondet, A. 3D Model Retrieval based on Depth Line Descriptor. In Proceedings of the 2007 IEEE International Conference on Multimedia and Expo, Beijing, China, 2–5 July 2007. [CrossRef]
14. Daras, P.; Axenopoulos, A. A Compact Multi-View Descriptor for 3D Object Retrieval. In Proceedings of the 2009 Seventh International Workshop on Content-Based Multimedia Indexing, Chania, Greece, 3–5 June 2009. [CrossRef]
15. Papadakis, P.; Pratikakis, I.; Theoharis, T.; Perantonis, S. PANORAMA: A 3D Shape Descriptor Based on Panoramic Views for Unsupervised 3D Object Retrieval. *Int. J. Comput. Vis.* **2010**, *89*, 117–192. [CrossRef]
16. Lian, Z.; Zhang, J.; Choi, S.; ElNaghy, H.; El-Sana, J.; Furuya, T.; Giachetti, A.; Guler, R.A.; Lai, L.; Li, C.; et al. SHREC'15 Track: Non-rigid 3D Shape Retrieval. In Proceedings of the Eurographics Workshop on 3D Object Retrieval, Zurich, Switzerland, 2–3 May 2015. [CrossRef]
17. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going Deeper with Convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015. [CrossRef]
18. Szegedy, C.; Ioffe, S.; Vanhoucke, V.; Alemi, A. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, San Francisco, CA, USA, 4–10 February 2017.
19. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016. [CrossRef]
20. Lowe, D.G. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110. [CrossRef]
21. Milolajczyk, K.; Schmid, C. A Performance Evaluation of Local Descriptors. *IEEE Trans. Pattern Anal. Mach. Intell.* **2005**, *27*, 1615–1630. [CrossRef] [PubMed]
22. Heinly, J.; Dunn, E.; Frahm, J.-M. Comparative Evaluation of Binary Features. In *Computer Vision–ECCV*; Springer: Berlin/Heidelberg, Germany, 2012; Volume 7573, pp. 759–773. [CrossRef]
23. Gonzalez, R.C.; Woods, R.E. *Digital Image Processing*, 3rd ed.; Pearson Prentice Hall: Upper Saddle River, NJ, USA, 2006.
24. Csurka, G.; Dance, C.R.; Fan, L.; Willamowski, J.; Bray, C. Visual Categorization with Bags of Keypoints. In Proceedings of the Workshop on Statistical Learning in Computer Vision (ECCV'04), Prague, Czech Republic, 11–14 May 2004.
25. Krizhevsky, A.; Sutskever, I.; Hinton, G. ImageNet Classification with Deep Convolutional Neural Networks. In Proceedings of the Advances in Neural Information Processing Systems, Lake Tahoe, NV, USA, 3–6 December 2012.
26. Taigman, Y.; Yang, M.; Ranzato, M.; Wolf, L. DeepFace: Closing the gap to human-level performance in face verification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014. [CrossRef]
27. Abdel-Hamid, O.; Mohamed, A.-R.; Jiang, H.; Deng, L.; Penn, G.; Yu, D. Convolutional Neural Networks for Speech Recognition. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2014**. [CrossRef]
28. Rodola, E.; Cosmo, L.; Litany, O.; Bronstein, M.M.; Bronstein, A.M.; Audebert, N.; Hamza, A.B.; Boulch, A.; Castellani, U.; Do, M.N.; et al. SHREC'17: Deformable Shape Retrieval with Missing Parts. In Proceedings of the Eurographics Workshop on 3D Object Retrieval, Lisbon, Portugal, 23–24 April 2017. [CrossRef]
29. Shilane, P.; Min, P.; Kazhdan, M.; Funkhouser, T. The Princeton Shape Benchmark. In Proceedings of the Shape Modeling International (SMI'04), Genova, Italy, 7–9 June 2004. [CrossRef]