Neurocomputing 510 (2022) 181–192

Contents lists available at ScienceDirect

Neurocomputing

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

A robust scheme for copy detection of 3D object point clouds

Jiaqi Yang [a], Xuequan Lu [b], Wenzhi Chen [a,*]

[a] College of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China
[b] School of Information Technology, Deakin University, Australia

ARTICLE INFO

Article history:
Received 11 August 2021
Revised 16 August 2022
Accepted 4 September 2022
Available online 8 September 2022
Communicated by Zidong Wang

Keywords:
3D point cloud copy detection
3D shapes
3D watermarking
GMM
Similarity

ABSTRACT

Most existing 3D geometry copy detection research focused on 3D watermarking, which first embeds "watermarks" and then detects the added watermarks. However, this kind of methods is non-straightforward and may be less robust to attacks such as cropping and noise. In this paper, we focus on a fundamental and practical research problem: judging whether a point cloud is plagiarized or copied to another point cloud in the presence of several manipulations (e.g., similarity transformation, smoothing). We propose a novel method to address this critical problem. Our key idea is first to align the two point clouds and then calculate their similarity distance. We design three different measures to compute the similarity. We also introduce two strategies to speed up our method. Comprehensive experiments and comparisons demonstrate the effectiveness and robustness of our method in estimating the similarity of two given 3D point clouds.

1. Introduction

The copy detection of 3D geometric data has received increasing attention in recent years. Several specific laws and regulations, such as EU GDPR [50], pay great attention to copyright issues, including 3D geometric data. It is significant to copyright valuable, high-quality 3D models like digital cultural heritage [1]. Complex 3D models, as shown in Fig. 1, are often obtained through a series of steps such as scanning, smoothing, hole filling, surface reconstruction, etc. Those steps are non-trivial and could be time-consuming and cost-expensive. As a result, it is more challenging to capture 3D models than 2D images [28]. Thus, it is very significant to value the owners' efforts and detect 3D models' copies.

This work focuses on a fundamental research problem: to judge if two point clouds (a 3D representation with a set of unordered points) are similar or not (e.g., partly similar or the same point cloud models), especially for complex models. The problem comes from the situation in which a user uses a 3D model, and we need to judge if this model is a manipulated duplication of a "ground-truth" model at hand (e.g., rotated copy, noisy copy, and cropped copy). If they are judged to be highly similar, the user's model is most likely a copy, and the user may violate the copyright of that model.

Existing methods mainly focused on borrowing the "watermarking" concept from digital watermarking [6,28,44]. The pipeline is very similar to digital watermarking on images: first adding watermarks and then detecting watermarks for recognition. However, the watermarking techniques depend greatly on watermarks which may be fragile to attacks like noise and cropping. For instance, if the 3D model is corrupted with considerable noise, the watermarks can hardly be extracted successfully. It is non-straightforward to add watermarks and then detect watermarks. An alternative is using 3D shape retrieval methods to compare the similarity of two 3D shapes. Nevertheless, 3D shape retrieval methods aim to search models of the same category as the query model, while our target in this work is 3D object copy detection (i.e., judging if two models are similar regardless of the categories, even in the presence of attacks).

We are motivated by the above analysis and propose a novel approach for 3D point cloud copy detection. Our core idea is first to align the two point clouds and then calculate three different similarity distances, revealing the similarity degree of the two point clouds.

In particular, we employ an effective point set registration algorithm - Coherent Point Drift (CPD) [31] for the alignment. We discover that the probability matrix depicts the relationship between the two point clouds and is very useful for calculating the similarity. To achieve this, we perform the Robust Principal Component Analysis (RPCA) [53] on the matrix and obtain its low-rank component, representing the vital information for this matrix. We design the low-rank measure like the mean of the low-rank matrix ele-

* Corresponding author.
E-mail addresses: yangjiaqi@zju.edu.cn (J. Yang), xuequan.lu@deakin.edu.au (X. Lu), chenwz@zju.edu.cn (W. Chen).
URL: http://www.xuequanlu.com (X. Lu).

https://doi.org/10.1016/j.neucom.2022.09.008
0925-2312/© 2022 Elsevier B.V. All rights reserved.

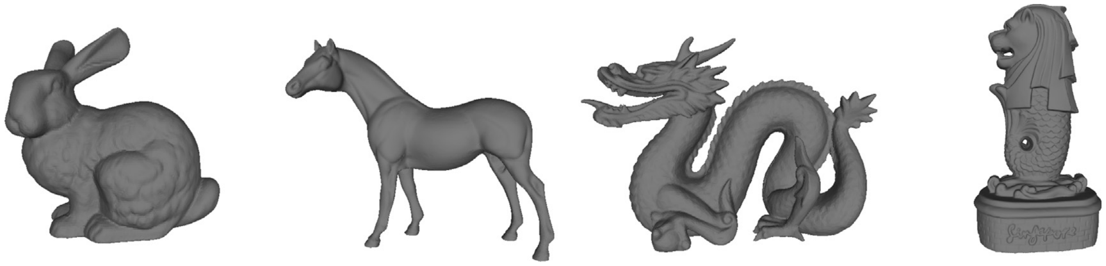


Fig. 1. Some high-quality and complex 3D shapes used in this work. From left to right: Bunny, Horse, Dragon, Merlion.

ments, which is sufficient to account for the similarity degree. In addition, we also design two other measures for speed and comparison purposes: the Kurtosis measure and the Correlation measure. Finally, we design two acceleration strategies to speed up the computation. The main contributions of this paper are as follows.

- We present a novel technique of 3D point cloud copy detection that avoids using the watermarking concept.
- We design three different but effective distance measures to calculate the similarity degree between two point clouds.
- We conduct extensive experiments to validate our method. We compare our method with watermarking methods. Furthermore, we also compare our method with recent 3D shape retrieval approaches in the shape retrieval setting. We finally perform some additional studies. Results show that our method is effective and robust in estimating the similarity of two 3D point clouds in the presence of various attacks.

2. Related Work

This section mainly concentrates on the techniques that are most related to 3D point cloud copy detection, including 3D shape watermarking and 3D shape retrieval. Finally, we look back upon some related applications of the Gaussian mixture model.

2.1. 3D Shape Watermarking

Like 2D image watermarking, 3D shape watermarking adds negligible watermarks on 3D geometric models and then extracts watermarks through specifically designed algorithms. It generally includes robust and fragile schemes [6]. The robust watermarking aims to endure malicious attacks and thus protect the copyright, while the fragile watermarking intends to check the authenticity and the integrity of the 3D models [28]. And we only consider robust watermarking, which generally requires a sophisticated process and obscure mapping relationships to ensure the robustness and transparency of the watermark. Transparency estimates the privacy of the embedded watermarks, and the robustness focuses on immutability, namely bit error rate (BER) and correlation.3D shape watermarking deals with the spatial domain [5,2,48,24,17,23] and the spectral domain [34,15,14,11], depending on either the geometry and connectivity information or the spectral information [28]. Wang et al. [51] concluded that intrinsic properties of 3D meshes (i.e., chaotic topology and unpredictable sampling of 3D meshes) and diversity of malicious distortion on watermarked meshes cause 3D watermarking technology to be more awkward compared to the digital image processing field.

Amar et al. [2] quantified and deformed Euclidean distances from all vertices to the mass center of the 3D model. There is a mapping relationship between watermarks and the parity of quantization value of Euclidean distances: an odd number corresponds to watermark value 0, and an even number corresponds to watermark value 1. And the vertex position is modified to reduce the quantization value in the watermark embedding phase, while the watermark is determined according to therelationship mentioned above in the watermark extraction phase.

Hamidi et al. [15] established a codebook with the private key, watermark, and wavelet coefficient vector (WCV) of coarsest meshes obtained through multi-resolution wavelet decomposition and reconstructed meshes using the WCV modified by the codebook. As for watermark extraction, the watermark should make the codeword in the codebook the closest to the WCV norm. After that, Hamidi et al. [14] improved the performance by using the vertices in saliency rather than all vertices in the mesh model.

Several additional processing techniques are devoted to watermarking for 3D printed models, which naturally introduce distortion during printing and scanning. Hou et al. [17] estimated the print axis by analyzing the layering artifact and added a sinusoidal frequency signal to the vertex coordinates calculated from the watermark. Delmotte et al. [7] computed the norm histogram continuously over the entire surface instead of a discrete set of vertices and shifted the mean of each bin of the norm histogram to indicate the watermark value (0 or 1). These methods eliminate the adverse effects of sampling in the scanning process.

2.2. 3D Shape Retrieval

3D shape retrieval targets to query 3D shapes close to the given 3D model. In essence, 3D shape retrieval is to extract and compare the feature of a shape with that of the query model. Different data representations and application scenarios exacerbate the complexity of shape retrieval methods [55,21]. We simply cover the structure-based methods and view-based methods here.

Structure-based approaches. Rich surface and hidden geometric/graph structure amply depict the discrepancy among shapes. The shape descriptors mentioned in [57] are practical in retrieval for polygon meshes and point clouds, e.g., global information [35], local features [12,47,42,13], Zernike moment [33], distribution [36,30], skeleton [40], topology [4,45]. Recently, deep learning methods for 3D shape retrieval based on shape structure have been proposed. For example, Furuya et al. [13] introduced the DLAN to extract rotation-invariant local 3D features and aggregated these local features into global descriptors. Feng et al. [10] calculated the spatial and structural descriptors of all polygon faces and obtained the global descriptors through the combination of descriptors and neighbor aggregation operations. Zhang et al. [58] proposed PCAN, which can predict the significance of each local point feature based on point context.

View-based approaches. Inspired by the intuitive perception of 3D shapes, researchers proposed to convert 3D shapes to two-dimensional planes (i.e., depth maps [9], projection [46,3,19]), thus facilitating the application of mature two-dimensional retrieval techniques for 3D shapes. Su et al. [46] extracted 2-dimensional features from different projection rendering views, which were computed by rotating the virtual camera for each shape. Later they aggregated these features into a global descriptor for the entire 3D shape. VGG neural networks pre-trained on ImageNet were used in their work. Instead of aggregation, Song et al. [3] matched the features of each projection with counterparts in the retrieval database one by one, applying the re-ranking component to process the matching results. Recently, there were also 3D shape retrieval studies related to metric learning. He et al. [16] proposed a triplet-center loss for view-based techniques to improve retrieval performance. Jing et al. [20] proposed a method to learn modal-invariant and view-invariant features from different modalities.

2.3. Gaussian Mixture Model

Gaussian Mixture Model (GMM) [39] is a weighted sum of M Gaussian components, which interprets complex abstract problems as data fitting problems. In general, the Expectation–maximization (EM) algorithm [29] is applied to estimate GMM parameters. Because of its powerful capability, GMM serves in several fields extensively, such as rigid and non-rigid point set registration [32,31,8], compressive sensing [56], speech recognition [37], model denoising [27], model reconstruction [38] and skeleton learning [25,26]. For instance, Preiner et al. [38] proposed a hierarchical EM algorithm to quickly reduce the number of model points, preserving the utmost details. Lu et al. [27] fitted GMM centroids (representing the filtered points of the noisy model) to the data (the noisy model), achieving robust feature-preserving point set filtering.

3. Method

This section explains how to detect the copies of the original 3D point cloud via our introduced method. We first give an overview of our approach and then explain each step of our method: alignment and similarity distance measure. Finally, we introduce acceleration strategies to speed up our method.

3.1. Overview

Our method consists of two steps to realize the copy detection of 3D object point cloud data. The first step is to align two point clouds, which ensures the fair computation of the "distance" between the two input models in the second step. We design three quantitative metrics to evaluate the "distance" between them. Also, we design strategies to speed up the computation. Fig. 2 illustrates the overview of the proposed method.

3.2. Alignment

Our first step is point set registration which aligns two point clouds to a similar pose. We employ CPD [31] since our main observations and the proposed method are formulated upon the probability matrix in CPD. Note that other ways like deep learning [46,10,18,43] do not provide this essential information, and thus our method is not a deep learning technique. Moreover, CPD is a robust and effective method for 3D registration. This work only considers the rigid registration of two point clouds.

For two point sets $\mathbf{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_N\}^T$ and $\mathbf{Y} = \{\mathbf{y}_1, \cdots, \mathbf{y}_M\}^T$, we assume $\mathbf{X}$ is the sample data set generated by a GMM with each point in $\mathbf{Y}$ acting as a centroid of a Gaussian distribution. The probability of $\boldsymbol{x}_n$ is defined as Eq. (1):

$$\begin{aligned} p(\boldsymbol{x}_n) &= \omega\tfrac{1}{N} + (1-\omega)\sum_{m=1}^{M} p(\boldsymbol{y}_m) g(\boldsymbol{x}_n|\boldsymbol{y}_m, \sigma) \\ g(\boldsymbol{x}_n|\boldsymbol{y}_m, \sigma) &= \tfrac{1}{(2\pi\sigma^2)^{D/2}} e^{-\frac{\|\boldsymbol{x}_n - \boldsymbol{y}_m\|^2}{2\sigma^2}}, \end{aligned} \qquad (1)$$

where $p(\boldsymbol{y}_m) = \frac{1}{M}, D = 3$ and σ is the equal isotropic covariance of all Gaussian components. And ω is the weight of the uniform distribution, which is to explain noise and outliers.

$\mathbf{y}_m$ is constrained to be rigid and observes the following form:

$$\mathbf{y}_m = s\mathbf{R}\mathbf{y}_m^{ori} + \mathbf{t}, \qquad (2)$$

where $\mathbf{y}_m^{ori}$ denotes the point of $\boldsymbol{Y}$ without any rigid transformation. s is a scaling factor, $\mathbf{R}_{3\times3}$ is a rotation matrix and $\mathbf{t}_{3\times1}$ is a translation vector.

To achieve a GMM that best explains the relationship between the two point clouds, we rewrite Eq. (1) adding rigid transformation, and minimize the negative log-likelihood $L(\theta)$.

$$\begin{aligned} L(\theta) &= -\sum_{n=1}^{N} log \sum_{m=1}^{M} p(\boldsymbol{x}_n|\theta) \\ p(\boldsymbol{x}_n|\theta) &= \omega\tfrac{1}{N} + (1-\omega)\sum_{m=1}^{M}\tfrac{1}{M} g(\boldsymbol{x}_n|\theta) \\ g(\boldsymbol{x}_n|\theta) &= \tfrac{1}{(2\pi\sigma^2)^{D/2}} e^{-\frac{\|\boldsymbol{x}_n - (s\boldsymbol{R}\boldsymbol{y}_m^{ori} + \boldsymbol{t})\|^2}{2\sigma^2}}, \end{aligned} \qquad (3)$$

where $\theta = (s, \boldsymbol{R}, \boldsymbol{t}, \sigma)$.

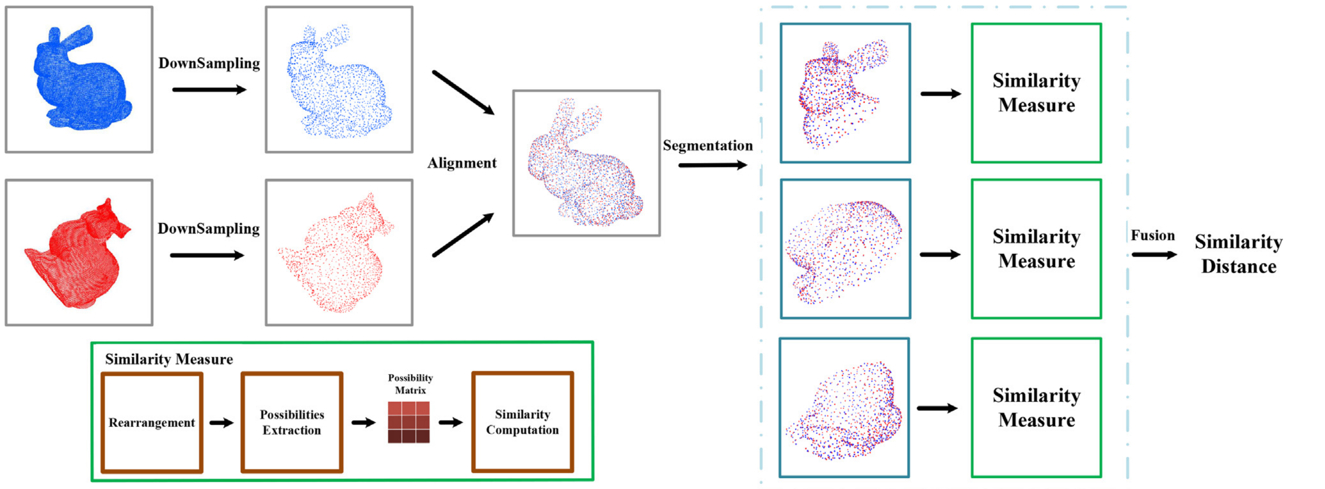
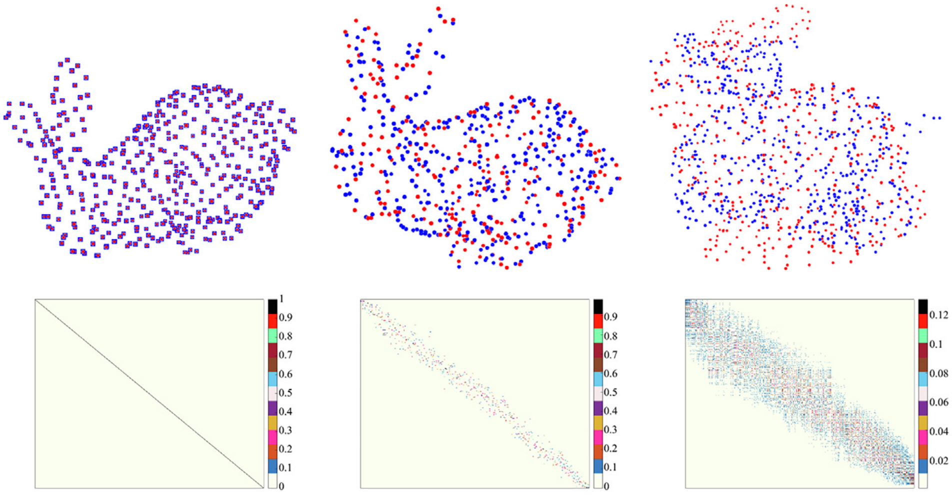


Fig. 2. Overview of our method. Two models are colored in red and blue, respectively. The final similarity distance is achieved after sampling, alignment, segmentation, and similarity calculation (details are explained in Section 3).



(a) b453(X) b453(Y) (b) b227(X) b378(Y) (c) b453(X) d521(Y)

Fig. 3. The registration results (in the first row) and the probability matrices (***P***) (in the second row) of 3 pairs of point clouds. b453 stands for the Bunny model with 453 points, d521 represents the Dragon model with 521 points, and the same below. See color bars for the value ranges of each element in the probability matrix.

The EM algorithm is used for optimization. In the E-step, the posterior probability can be computed as:

$$\begin{aligned} p_{m,n} &= p(\mathbf{y}_m|\mathbf{x}_n) \\ &= \frac{e^{-\frac{1}{2}\left\|\frac{\mathbf{x}_n-(s\boldsymbol{R}\boldsymbol{y}_m^{old}+\boldsymbol{t})}{\sigma}\right\|^2}}{\sum_{j=1}^{M} e^{-\frac{1}{2}\left\|\frac{\mathbf{x}_n-(s\boldsymbol{R}\boldsymbol{y}_j^{old}+\boldsymbol{t})}{\sigma}\right\|^2}+c}, \end{aligned} \tag{4}$$

where c is a constant independent of rigid transformation. In the M-step, we solve for $s, \boldsymbol{R}, \boldsymbol{t}$ and σ which are discussed in detail in [31].

$$\begin{aligned} SP = \sum_{n=1}^{N}\sum_{m=1}^{M} p_{m,n}, \mu_x = \tfrac{1}{SP}\boldsymbol{X}^T\boldsymbol{P}^T\boldsymbol{1}, \mu_y = \tfrac{1}{SP}\boldsymbol{Y}^T\boldsymbol{P1}, \\ \hat{\boldsymbol{X}} = \boldsymbol{X} - \boldsymbol{1}\mu_x^T, \hat{\boldsymbol{Y}} = \boldsymbol{Y} - \boldsymbol{1}\mu_y^T, \\ \boldsymbol{U}\Sigma\boldsymbol{V}^T = SVD(\hat{\boldsymbol{X}}^T\boldsymbol{P}^T\hat{\boldsymbol{Y}}), \boldsymbol{C} = d(1,\cdots,1,det(\boldsymbol{UV}^T)), \\ \boldsymbol{R} = \boldsymbol{UCV}^T, s = \frac{tr(\hat{\boldsymbol{Y}}^T\boldsymbol{P}\hat{\boldsymbol{X}}\boldsymbol{R})}{tr(\hat{\boldsymbol{Y}}^T d(\boldsymbol{P1})\hat{\boldsymbol{Y}})}, \boldsymbol{t} = \mu_x - s\boldsymbol{R}\mu_y, \end{aligned} \tag{5}$$

where ***P*** is the matrix formed from $p_{m,n}$, and ***1*** is a column vector of all ones. $det(\cdot)$ is used to calculate the determinant and $tr(\cdot)$ is used to calculate the trace. $d(\cdot)$ is a diagonal matrix formed from the column vector and $SVD(\cdot)$ denotes singular value decomposition.

3.3. Similarity Distance

This step is to estimate the distance between the two aligned point clouds. As opposed to most research that utilized point positions to compute the similarity distance, we define three distance measures based on the probability matrix (***P***).

Fig. 3 visualizes three probability matrices between ***Y*** and ***X***. Additionally, to combat the impact of disorder, it is necessary to reorganize points' coordinates with the rearrangement component shown in Fig. 2. This component rearranges the model in the ascending order of the x, y, and z coordinate values of the 3D model points (i.e., first x, then y, last z). Therefore, a colored diagonal band in Fig. 3 can express the relationship between 2 point clouds.

Apparently, the colored diagonal band of ***P*** is tightly bounded up with the overlap of ***X*** and ***Y***. The posterior probability $p_{m,n}$ defined in Eq. (4) indicates the coincidence between $\boldsymbol{y}_m$ and $\boldsymbol{x}_n$. For the probabilities in a column where $\boldsymbol{x}_n$ is fixed, they follow that $\sum_{m=1}^{M} p_{m,n} = 1$. Ideally, the probability $p_{m^*,n} = 1$ proves that x_n only corresponds to the y_{m^*}. In the case depicted in Fig. 3(a), this narrowest diagonal band shows a one-to-one correspondence between ***X*** and ***Y***, and the probabilities inside and outside the diagonal band have the greatest difference.

In the following, we design three different distance measures based on geometry and statistics, respectively.

3.3.1. Low-rank measure

The measure attempts to characterize the diagonal band. We strip the diagonal band from the probability matrix ***P***, keeping or eliminating the diagonal band. A significantly effective elimination method is that the higher probabilities are considered as noise in ***P***, and the low-rank matrix does not retain this information, which has been widely applied in image denoising.

Robust principal component analysis (RPCA) [53], as a low-rank representation solver, performs a vital function in compressive sensing and sparse representation against traditional methods such as PCA. The key is the decomposition of the complex matrix ***C*** into a low-rank matrix ***A*** and a sufficiently sparse error matrix ***E*** shown in Eq. (6).

$$\min_{\boldsymbol{A},\boldsymbol{E}} ||\boldsymbol{A}||_* + \lambda||\boldsymbol{E}||_1, \qquad \textit{subject to } \boldsymbol{C} = \boldsymbol{A} + \boldsymbol{E}, \tag{6}$$

where $||\cdot||_*$ denotes the nuclear norm, $||\cdot||_1$ denotes the sum of the absolute value of matrix elements, and λ is a weighting parameter. There are various open-source RPCA variants and their evaluations[1]. Note that Lin et al. [22] proposed a variant of RPCA based on the inexact augmented Lagrange multiplier (IALM) method – a trade-off between higher precision and less storage/time, which works best for our method in practice.

Fig. 4 shows the low-rank components of the three probability matrices in Fig. 3. For the complete or almost similar Bunny mod-

[1] The RPCA variant in our work is maintained by the Perception and Decision Lab at the University of Illinois at Urbana-Champaign and Microsoft Research Asia in Beijing. You can find the methods and brief introductions we use at the following website: https://people.eecs.berkeley.edu/~yima/matrix-rank/sample_code.html

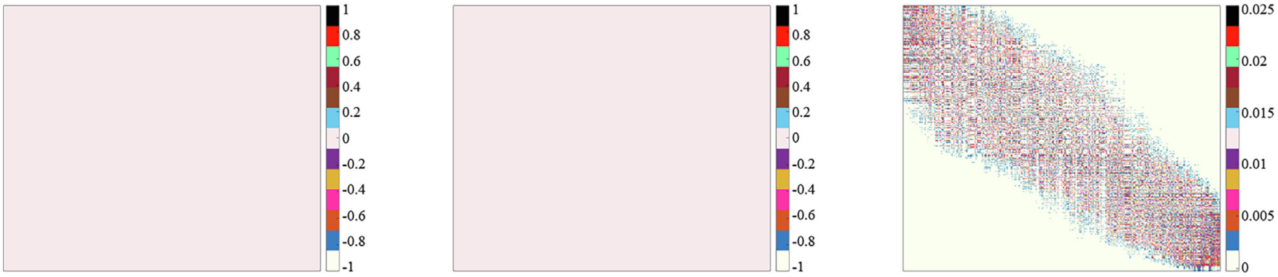


Fig. 4. The low-rank matrices. For two almost consistent models, the low-rank matrix is even close to the all-zero matrix, which proves the role of the IALM method in model comparison.

els, IALM characterizes the $\boldsymbol{P}$ as an all-zero matrix, while the low-rank matrix $\boldsymbol{A}$ corresponding to the registration of the Bunny model and the Dragon model retains most of the values.

To facilitate the measurement of the low-rank matrix, we define a distance named LR based on IALM in Eq. (7).

$$LR = \frac{1}{M}\frac{1}{N}\sum_{i=1}^{M}\sum_{j=1}^{N}[IALM(\boldsymbol{P}^T)]_{i,j}, \tag{7}$$

where $IALM(\cdot)$ denotes the function of obtaining low-rank matrix mentioned in Eq. (6), and $(\cdot)_{i,j}$ denotes the elements in the i-th row and the j-th column of a matrix.

3.3.2. Kurtosis measure

This measure calculates the similarity between two point clouds in terms of the probability distribution of individual points. The colored bands in Fig. 3 indicate that $\boldsymbol{y_m}$ usually has a strong correlation with a short-range continuous $\boldsymbol{x_n}$. Therefore, if each column of $\boldsymbol{P}$ is viewed as a data sample set, these values should follow certain patterns in the continuous space to some extent.

Fig. 5 describes the distribution of $p(\boldsymbol{y}_m|\boldsymbol{x}_{200})$, where $\boldsymbol{y}_m$ can be any point in $\boldsymbol{Y}$. And these distributions are generally low at both ends and high in the middle. For the entirely aligned Bunny models shown in Fig. 5a) and Fig. 5(b), only $p_{200,200} = 1$ while the rest elements are 0. Namely, closer pairs of points $(\boldsymbol{x_m}, \boldsymbol{y_n})$ will induce larger probabilities, leading to a steeper distribution. As for the registration of the Bunny and the Dragon models, scattered outliers lead to a sharp decline in deviation (2 orders of magnitude).

We define the distance KURT based on the kurtosis in Eq. (8), which reveals and aggregates the sharpness of local distributions.

$$KURT = \frac{1}{N}\sum_{j=1}^{N} Kurt(\boldsymbol{P}_j), \tag{8}$$

where $\boldsymbol{P}_j$ denotes the j-th column of $\boldsymbol{P}$, and $Kurt(\cdot)$ denotes the kurtosis coefficient of the given data calculated according to Eq. (9).

$$Kurt(\mathbf{x}) = \frac{\mu_4}{\sigma_4} = \frac{\frac{1}{M}\sum_{i=1}^{M}(x_i - \bar{x})^4}{\left(\frac{1}{M}\sum_{i=1}^{M}(x_i - \bar{x})^2\right)^2}, \tag{9}$$

where μ_4 denotes the fourth central moment, and σ denotes the standard deviation.

3.3.3. Correlation measure

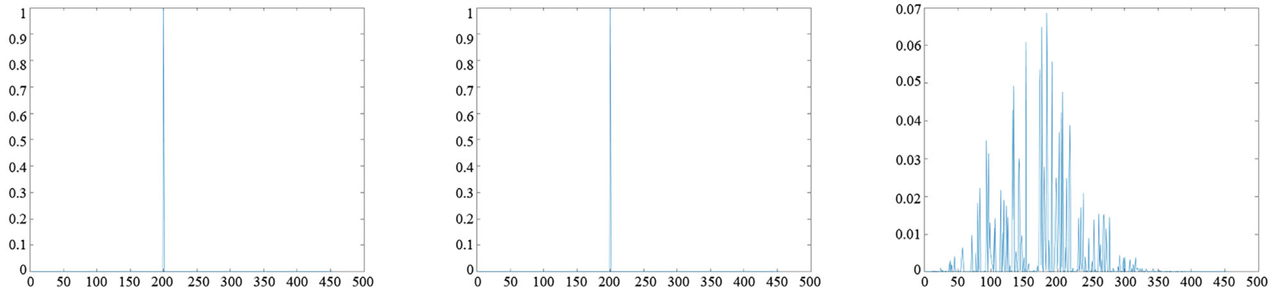
This measure focuses on the correspondences of point clouds in geometric space. Correspondences is a collection of point pairs $(\boldsymbol{y}_m, \boldsymbol{x}_{n^*})$, which indicates $\boldsymbol{x}_{n^*}$ is most likely to $\boldsymbol{y}_m$ after point set registration. Consequently, $\boldsymbol{x}_{n^*}$ and $\boldsymbol{y}_m$ are observed as characteristics of $\boldsymbol{X}$ and $\boldsymbol{Y}$ in the geometric space, which is of great significance for judging the similarity of various models. In short, the correspondences reconstruct $\boldsymbol{X}$ in order to simulate $\boldsymbol{Y}$.

The process is as follows: a) generate matrices $X^*_{M\times 3} = \{\boldsymbol{x}_{n^*}\}$ and $Y^*_{M\times 3} = \{\boldsymbol{y}_m\}$ respectively according to the correspondences, where the position of $\boldsymbol{x}_{n^*}$ in X^* is the same as the position of $\boldsymbol{y}_m$ in Y^*; b) uses the Pearson distance to measure the distance between X^* and Y^*. As a result, the irrelevance included in $\boldsymbol{X}$ and disturbance such as different numbers of points are eliminated.

We define the distance CORR based on Pearson distance in Eq. (10).

$$CORR = \frac{\sum_{i=1}^{M}\sum_{j=1}^{3}\delta_{X^*_{i,j}}\delta_{Y^*_{i,j}}}{\sqrt{\left[\sum_{i=1}^{M}\sum_{j=1}^{3}\delta^2_{X^*_{i,j}}\right]\left[\sum_{i=1}^{M}\sum_{j=1}^{3}\delta^2_{Y^*_{i,j}}\right]}}, \tag{10}$$

where $\delta_{X^*_{i,j}} = X^*_{i,j} - \bar{X}^*, \delta_{Y^*_{i,j}} = Y^*_{i,j} - \bar{Y}^*, (\cdot)_{i,j}$ denotes the element in the i-th row and the j-th column of a matrix. $\bar{X}^*$ and $\bar{Y}^*$ denote the mean of all elements in X^* and Y^*, respectively.



(a) b453(X) b453(Y) (b) b227(X) b378(Y) (c) b453(X) d521(Y)

Fig. 5. Distribution of the 200th column of the three probability matrices shown in Fig. 3.

Remark. Considering performance and needs, we design three different metrics to measure the similarity between two aligned point clouds. The LR distance is more stable in various attack scenarios, focusing on the essential information of the probability matrix. However, because of complex operations such as SVD operation, its required resources and running time will become unaffordable with the increase of data points. For this reason, we design the KURT distance and the CORR distance to calculate similarity quickly. Meanwhile, the CORR distance is naturally compact with the correlation, which is the popular metric for evaluating the robustness of watermarking methods. It offers plausible comparisons with watermarking methods. However, the accuracy of the CORR distance is related to the number of points and alignment accuracy. When the number of points is insufficient, the CORR may fail to express the correlation accurately. Furthermore, two different models may still cause considerable overlap, which may induce a "misleading" CORR measure.

3.4. Acceleration Strategies

The registration and RPCA involve expensive computation. As depicted in [31], the CPU time occupied by CPD increases exponentially as points soar. Meanwhile, the IALM method becomes unstable, and its performance drops sharply when the models are extraordinarily mismatched or have excessive points. Hence, we design speeding-up strategies in this section.

3.4.1. Downsampling

As shown in Fig. 2, the point cloud maintains the original geometric characteristics after downsampling but reduces the amount of data. This way alleviates the stress of calculation. Compared to random downsampling, the hierarchical expectation maximum (HEM) [38] algorithm tends to aggregate points level by level and provides a more meaningful representation for the original point cloud. The HEM algorithm is shown below. We utilize the GMM to characterize the 3D point clouds, applying only one initial EM iteration on the point cloud and then sequentially shrinking the mixture by hierarchically applying EM on Gaussians rather than points. Specifically, we pick up 1/3 of the Gaussian components obtained in the current round as the Gaussian components used in the next iteration, and the remaining 2/3 components are regarded as "virtual samples". Moreover, we only merge the Gaussian components in the same neighborhood when executing the EM procedure. In our implementation, the neighborhood radius is determined by the diagonal length of the minimum bounding box of the point cloud and a customized weight.

3.4.2. Segmentation and fusion

Another way to reduce the overhead caused by IALM is segmentation and fusion. Segmentation and fusion can effectively decrease the runtime. Generally, parts of two registered point clouds should be aligned, respectively. With this premise, we design the segmentation and fusion strategy below. First and foremost, we put two point clouds in the same three-dimensional coordinate system after registration and arrangement and then determine the longest axis from the x, y, and z-axis where the point clouds have the most extensive projection range. Later, the models are partitioned into $\max(\lceil M/T \rceil, \lceil N/T \rceil)$ parts evenly along the longest axis, in the sense that the number of points in each part is roughly T. Moreover, the last segment covers the remaining points to maintain the integrity of the point cloud. Finally, we calculate the similarity distance of each corresponding parts pair and merge their distances. This strategy simplifies the RPCA problem when processing relatively large point clouds.

4. Experimental Results

In this section, we illustrate the effectiveness of our method in the copy detection of 3D object point clouds. We compare our method with 3D watermarking and 3D shape retrieval methods most relevant to this work. We adopt the attacks introduced by [52] in our experiments.

4.1. Comparison With 3D Model Watermarking

This section compares our method with the watermarking algorithm based on Euclidean distance deformation (EU) [2].

Considering that this paper's core problem is copy detection, we only focus on the robustness of our method and the watermarking methods in detecting copies suffering from different attacks.

It now boils down to the comparability of the watermark's correlation/BER and similarity distance. We determine the CORR distance threshold (0.9951) according to the principle of maximizing the delta of true positive rate (TPR) and false positive rate (FPR), which will be introduced in Section 4.3.1. Moreover, the CORR distance is greater than or equal to the threshold, indicating that the test model is a copy of the given model. Our method can hardly determine the threshold for the watermarking method because of its high sensitivity to attack intensity and the lack of statistics. However, it seems plausible that we use the FPR of the CORR threshold to determine the threshold of the watermarking method. For a 64-bit watermark, if a misjudgment of 4% (approximately the FPR of the CORR threshold) is allowed, the BER of watermarks should not be higher than 0.22.

Table 1 shows three models for this experiment. The homologous models are generated by the benchmark developed in [52].

Table 2 lists the comparisons between our method and the EU method against three geometric attacks, including noise, quantization, and smoothing, and three connectivity attacks, including cropping, simplification, and subdivision. Our method outperforms the EU method against geometric attacks since almost all CORR reaches the maximum value of 1. With the intensifying of the attacks, BER rises considerably and exceeds the threshold. For example, the BER of the cow model with 0.5% noise reaches 0.31. Regarding the subdivision attack, our method is also superior to the EU method. Although the BER of the EU method watermark does not exceed the above threshold of 0.22 except for the dragon model suffering from the $\sqrt{3}$ subdivision, CORR almost perfectly reaches the maximum value (1). Furthermore, both methods have their own merits against simplification attacks. The EU method is more suitable for the cow and hand models, while our method resists all the simplified scenarios of the dragon model. As for cropping attacks, our method performs poorly. However, the results indicate that our method is more robust than the EU method in most cases.

4.2. Comparison with 3D Shape Retrieval

We compare our method with MVCCN [46], MeshNet [10] and Unique Shape Context (USC) [47]. The first two are deep learning methods. For fair comparisons, we design our homologous model

Table 1
Models used for 3D watermarking comparison.

Name	vertices	faces
Dragon	50,000	100,000
Cow	2,904	5,804
Hand	36,619	72,958

Table 2
Comparison of our method (CORR) with the EU approach (BER). The up/down arrows at the metrics heading indicate the direction of the better result: a smaller EU indicates a better result, and a larger CORR means a better result. The attack types are abbreviations whose corresponding relationship is shown in Table 3.

Attack		Cow		Dragon		Hand	
Type	Intensity	EU↓	CORR↑	EU↓	CORR↑	EU↓	CORR↑
NA	0.05%	0.08	1.0000	0	0.9999	0	1.0000
	0.1%	0.16	1.0000	0.04	0.9999	0.01	1.0000
	0.3 %	0.23	1.0000	0.07	0.9999	0.27	1.0000
	0.5%	0.31	1.0000	0.20	0.9999	0.26	1.0000
QU	10-bits	0.02	1.0000	0.07	0.9999	0.10	1.0000
	9-bits	0.14	1.0000	0.16	0.9999	0.32	1.0000
	8-bits	0.22	1.0000	0.20	0.9999	0.26	1.0000
	7-bits	0.35	1.0000	0.26	0.9998	0.10	1.0000
SM	5	0.10	0.9999	0.13	0.9999	0.01	1.0000
	10	0.16	0.9999	0.16	0.9999	0.10	1.0000
	30	0.23	0.9997	0.20	0.9999	0.20	1.0000
	50	0.36	0.9996	0.23	0.9999	0.23	1.0000
CR	10%	0	0.9935	0	0.9930	0.01	0.9922
	30%	0.16	0.9655	0.13	0.9452	0.10	0.9713
	50%	0	0.9316	0.20	0.8684	0.13	0.7967
SI	50.0%	0.16	0.9879	0.23	0.9995	0.20	0.9913
	70.0%	0.22	0.9881	0.29	0.9993	0.16	0.9933
	90.0%	0.26	0.9875	0.46	0.9990	0.16	0.9985
	95.0%	0.13	0.9796	0.53	0.9983	0.26	0.9953
	97.5%	0.20	0.9571	0.56	0.9981	0.30	0.9676
SU	Loop	0.07	0.9999	0.07	0.9999	0.20	1.0000
	Midpoint	0	0.9999	0.08	0.9999	0	1.0000
	$\sqrt{3}$	0.02	0.9999	0.23	0.9999	0.04	1.0000

set HM25 based on the Princeton ModelNet40 [54] and evaluate these approaches.

4.2.1. Homologous Model Dataset

Previously, researchers gathered 40 categories of routinely accessed objects online to construct the ModelNet40 containing 12,311 shapes. However, we construct the HM25, focusing on the above attacks rather than categories. We yield some homologous models with the attacks in [52]. And the homologous models are obtained by imposing diverse attacks on the same original model. Table 3 lists the attacks and parameters. And Fig. 6 shows eight attacked models of the Bunny model (i.e., Bunny's homologous models).

We pick up 625 objects in 25 categories from ModelNet40 as the source models and obtain 21 attacked models from each source model. Each category takes 25 source models and is split by a training-test ratio of 8:2.

It should be noted that this dataset generation procedure should comply with several principles. a) The source training and test models are from the train set and test set of ModelNet40, respectively; b) All models, including homologous models, must not exceed 4096 faces considering MeshNet, which aggregates features of all faces; c) The attacks are only applicable to two-dimensional orientable manifold meshes which are not guaranteed by ModelNet40. At last, we created the HM25, containing 13,750 3D models of 25 categories.

Table 3
Type and intensity of attacks on homologous models. Intensity implies various units/properties perchance.

Attack Type	Abbr.	Intensity	Number
Crop	CR	5%/10%	1
Noise Addition	NA	0.1%/0.3%/0.5%	1
Quantization	QU	9/8/7	1
Reorder	RE	/	1
Simplification	SI	10%/20%/40%	1
Smooth	SM	10/30/50	1
Similarity Transformation	ST	/	3
Subdivision	SU	loop/$\sqrt{3}$/midpoint	1

4.2.2. Experimental Setup

We utilize HM25 to fine-tune MVCNN with 12 views pre-trained on ImageNet and MeshNet pre-trained on the simplified version of ModelNet40 developed in [10]. The hyperparameters are consistent with the original papers.

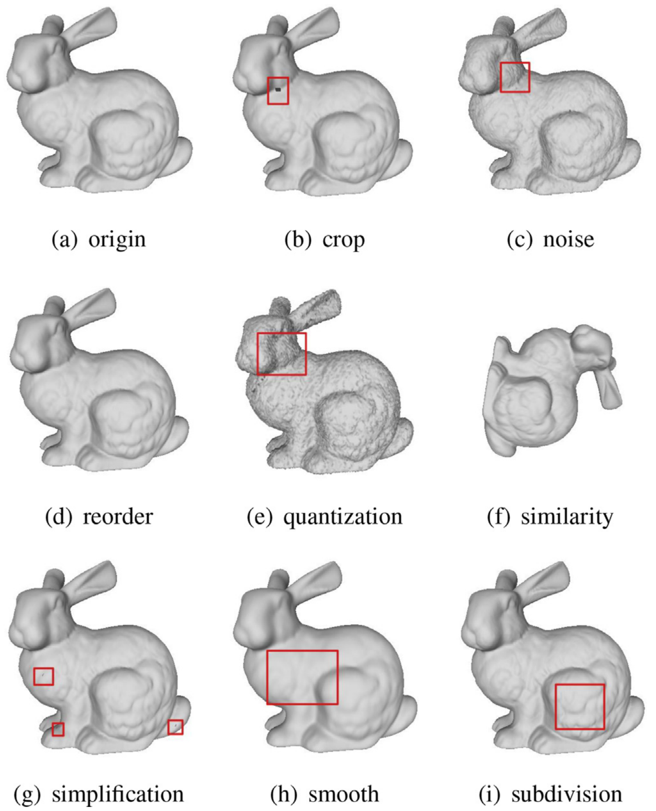
Then we use the HM25 test set for comparison experiments, with the original model as the query and the attacked models as the retrieval library. We extract the descriptors of query and retrieval models with MVCNN [46], MeshNet [10], and USC [47], respectively, and measure the similarity of models with Euclidean distance.

MVCNN [46] takes the output of the relu7 layer of CNN2 as the descriptor, and MeshNet [10] takes the output of the penultimate layer of the perception layer as the descriptor. The USC [47] algorithm uses the maximum pooling result of the feature vectors of all points as the descriptor. We implement USC with Point Cloud Library (PCL) [41], where the radius parameter is the mean of the 20th nearest neighboring distance in the point cloud. Table 5 shows the information on the three methods.

In the experiment, we evaluate the performance of our method as well as 3D shape retrieval algorithms. Firstly, we divide the HM25 test dataset by attack types, leading to 21 query sub-datasets and a target sub-dataset ($\mathscr{T}$) containing the whole source models. Each sub-dataset contains 125 models. After that, we calculate the retrieval rate Top_K of a given query database $\mathscr{Q} = \{q_1, \cdots, q_S\}$ on the target database $\mathscr{T}$. For the i-th model in $\mathscr{Q}$, we calculate the similarity between q_i and the model in the $\mathscr{T}$ one by one (our method uses LR, KURT, and CORR distances, and other methods use the L_2 distance), and arrange the models in $\mathscr{T}$ in the descending/ascending order of similarity. The set of K models most similar to q_i is denoted as res_i. Top_K is defined as:

$$\begin{gathered} Top_K = \frac{1}{S}\sum_{i=1}^{S}\sum_{k=1}^{K} Res_{i,k} \\ Res_{i,k} = \begin{cases} 1, & q_i \in \mathscr{H}(res_{i,k}) \\ 0, & q_i \notin \mathscr{H}(res_{i,k}) \end{cases}, \end{gathered} \tag{11}$$

where S denotes the number of models in $\mathscr{Q}$ ($S = 125$), $res_{i,k}$ is the k-th model in res_i, and $\mathscr{H}(x)$ denotes the collection of homologous models of the source model x.



(a) origin (b) crop (c) noise

(d) reorder (e) quantization (f) similarity

(g) simplification (h) smooth (i) subdivision

Fig. 6. Examples of the Bunny's homologous models. The red box marks the difference between the currently attacked and the original models.

4.2.3. Results

Table 4 summarizes the experimental results of shape retrieval on HM25. Our method outperforms other methods in combating similarity transformation. Compared with the USC[47] method (20.0%), the Top_1 retrieval rate of our method (60.8%) is even three times higher than it. Besides, our method is robust against noise addition, quantization, and reorder attacks, where Top_5 results of CORR measure reach above 99%, although somewhat inferior to the deep learning methods. However, there is a gap between our method and deep learning methods when facing connectivity attacks that adjoin or slit some points or edges. As for the cropping attack, the Top_1 accuracy of our method based on CORR measure is as low as 36.0% while MVCNN[46] achieves 92.8% accuracy. However, the CORR measure is up to 94% in the Top_5 results against the connectivity attack. In short, the results demonstrate that our method performs remarkably in general, better than USC, and slightly poorer than MVCNN[46] and MeshNet[10], in particular, Top_2 and Top_5 results.

Regarding the performance of our method in Table 4, we have the following observation. Our method may be weaker than MVCNN and MeshNet, which certainly benefit from the powerful deep learning. However, we would like to stress that deep learning retrieval methods learn a large amount of data and the 'category' information in order to search models of the same category of the given query model. By contrast, our method judges whether two 3D models are similar or not (e.g., incomplete or complete model) in a more flexible manner regardless of the categories. It does not require a large amount of data and takes as input two models only.

To further illustrate the results intuitively, Fig. 7 presents some of the retrieval results of the queries based on similarity transformation according to the CORR measure. As shown in the third row of Fig. 7, compared with the wardrobe and its original model, the wardrobe is more similar to the bottle. Several factors contribute to this unreasonable phenomenon. First, the wardrobe lacks enough points to describe its shape, and the CORR measure becomes invalid actually. Second, the CPD algorithm itself is not working well for the alignment.

4.3. Additional Studies

4.3.1. Threshold

This section calculates the *TPR* and *FPR* of the similarity distance to determine the distance threshold. *TPR* is the probability of perfectly detecting the duplicate and the source model, and *FPR* is the probability of mistaking a non-duplicate as a copy of the source version. To this end, we calculate the *TPR* and *FPR* of the HM25 data set. Firstly, we obtain the positive sample set $\mathscr{P}$ and the negative sample set $\mathscr{N}$, where $\mathscr{P} = \{(X,Y)|C(X,Y) = 1, X \in HM25, \quad Y \in HM25\}, \mathscr{N} = \{(X,Y)|C(X,Y) = 0, X \in HM25, Y \in HM25\}$. $C(X,Y) = 1$ denotes that X is the copy of Y; otherwise, they are not a copy relationship. Note that the subdivision attack is not considered here. Subsequently, we calculate the similarity distance sets $\mathscr{PD}$ and $\mathscr{ND}$ of the model pairs in $\mathscr{P}$ and $\mathscr{N}$, respectively, and compute the *TPR* and *FPR* of the $\mathscr{PD}$ and $\mathscr{ND}$ according to multiple preset thresholds. Given the threshold t, *TPR* and *FPR* of the LR distance and CORR distance are calculated as:

$$\begin{aligned}
TPR_{LR} &= \frac{|\{x|x \leqslant t, x \in \mathscr{PD}\}|}{|\mathscr{PD}|},\\
FPR_{LR} &= \frac{|\{x|x \leqslant t, x \in \mathscr{ND}\}|}{|\mathscr{ND}|},\\
TPR_{CORR} &= \frac{|\{x|x \geqslant t, x \in \mathscr{PD}\}|}{|\mathscr{PD}|},\\
FPR_{CORR} &= \frac{|\{x|x \geqslant t, x \in \mathscr{ND}\}|}{|\mathscr{ND}|},
\end{aligned} \tag{12}$$

where $|\cdot|$ denotes the number of elements in the set. Ultimately we decide the optimal threshold in line with maximizing $TPR - FPR$. Fig. 8 illustrates the trend of $TPR_{LR}, FPR_{LR}, TPR_{CORR}$, and FPR_{CORR} with varying the threshold. When $(TPR_{CORR}, FPR_{CORR}) = (0.9259, 0.0369)$ and $(TPR_{LR}, FPR_{LR}) = (0.8365, 0.0413)$, the difference between *TPR* and *FPR is maximum*. In this way, we empirically find the optimal

Table 4
Comparison of our method with the other three retrieval methods. The first four categories are geometric attacks (ST, NA, SM, QU), and the latter two are connectivity attacks (CR, SI). NA1 denotes the 0.1% noise attack (the rest of the symbols and so on). See Table 3 for abbreviations and Intensities. Bold indicates the highest performance.

Attack	Top_1						Top_2						Top_5					
	LR	KURT	CORR	[10]	[46]	[47]	LR	KURT	CORR	[10]	[46]	[47]	LR	KURT	CORR	[10]	[46]	[47]
ST	43.2	56.0	**60.8**	28.8	26.4	20.0	50.4	**64.0**	60.8	46.4	32.8	20.8	56.0	67.2	**68.0**	59.2	39.2	24.8
NA1	55.2	83.2	94.4	99.2	**100**	94.4	60.8	94.4	95.2	**100**	**100**	94.4	63.2	96.0	99.2	**100**	**100**	94.4
SM10	60.0	67.2	76.8	94.3	**97.6**	87.2	67.2	77.6	80.0	96.8	**99.2**	88.0	70.4	87.2	93.6	**100**	**100**	89.6
QU7	84.8	78.4	93.6	93.6	**99.2**	84.0	89.6	90.4	95.2	98.4	**100**	84.8	92.0	95.2	99.2	**100**	**100**	85.6
CR5	68.0	46.4	36.0	87.2	**92.8**	62.4	81.6	60.0	51.2	92.0	**95.2**	65.6	88.0	74.4	94.4	98.4	**97.6**	71.2
SI10	54.4	75.2	80.0	90.4	**94.4**	89.6	61.6	85.6	84.8	92.0	**95.2**	90.4	65.6	94.4	93.6	**96.8**	96.0	91.2
RE	59.2	82.4	94.4	92.0	**100**	94.4	64.8	94.4	95.2	98.4	**100**	94.4	66.4	96.0	99.2	**100**	**100**	94.4

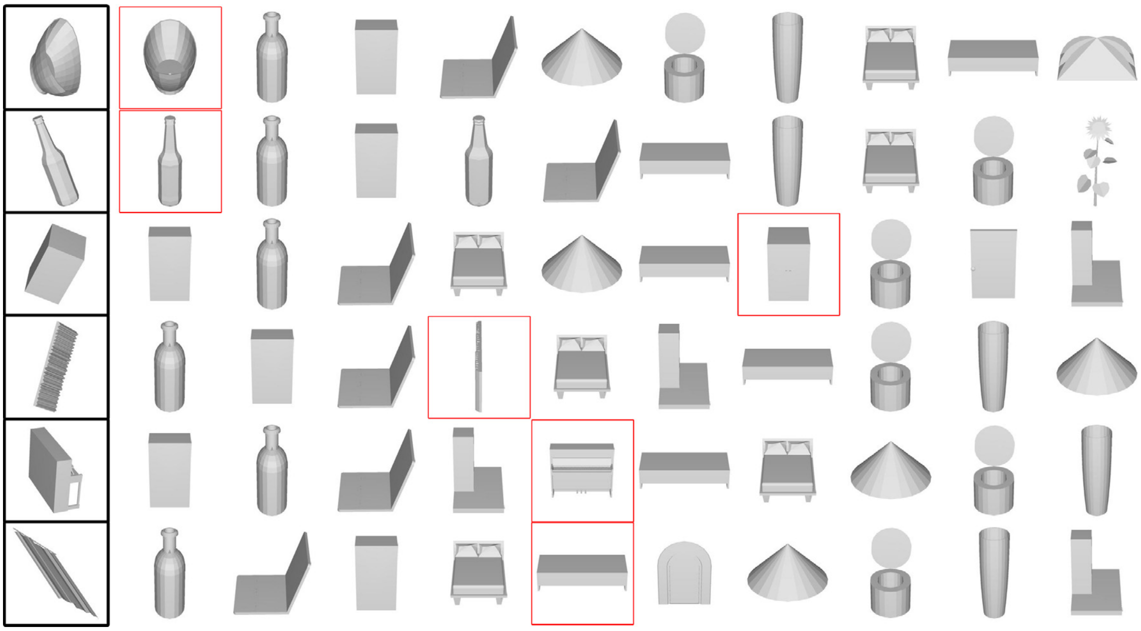
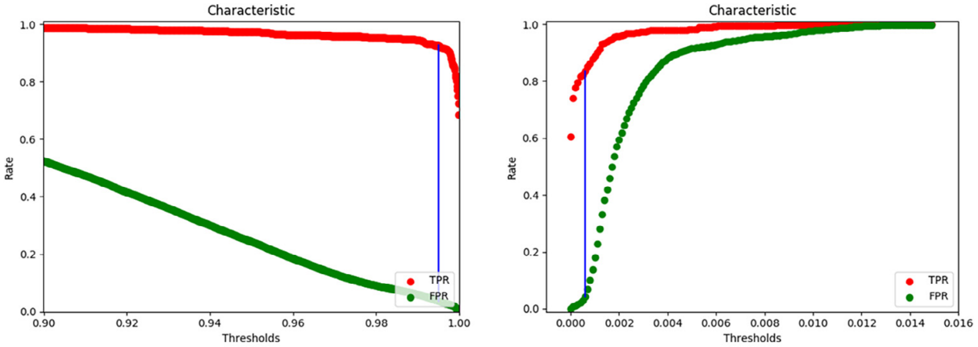


Fig. 7. Top ten original models queried by the homologous models based on the similarity transformation attack. The similarity is calculated by the CORR measure. And the black box is the query for similarity transformation, and the red box indicates the best-matched original model.



(a) threshold of CORR

(b) threshold of LR

Fig. 8. The curve of *TPR* and *FPR* of LR and CORR varying with the threshold. The blue line indicates the maximum $TPR - FPR$ under this threshold.

threshold of CORR distance $t_{CORR} = 0.9951$, and the optimal threshold of LR distance $t_{LR} = 0.0006$.

4.3.2. Downsampling

In this section, we prove that downsampling improves the efficiency of point set registration. To this end, we choose the Bunny model and its 5% noisy version as the experimental objects, both of which have 34,835 points and 69,666 faces, and conduct the following experiments. Firstly, we calculate the measures between the two models as the baseline. We then perform HEM downsampling [38] and random downsampling on the two models to obtain point clouds with different numbers of points. The downsampling parameters are adjusted to ensure the same number of points in both sampling schemes. Finally, we calculate the similarity distances and registration runtime of the point clouds above.

Table 6 shows the effectiveness of the downsampling strategies. It is observed that the higher the sampling rate is, the greater the advantage of HEM over random downsampling appears. Downsampling significantly reduces the CPD algorithm's running time (i.e., point set registration) because of the plunge of point number. Meanwhile, it has little impact on LR and CORR. The maximum modification of only 10^{-7} indicates the best robustness of the LR measure against downsampling. The CORR measure has larger variations than the LR distance, with a tiny difference of 0.03. KURT is sensitive to downsampling (i.e., number of points), but it is still doable in judging models with the same number of points.

Table 7 compares the runtime of HEM and random downsampling. It is feasible to control the approximate sampling rates for the Horse model with 112,642 points and the Bunny model with 34,835 points. The HEM algorithm generally demands more extra time than the random downsampling. However, HEM is still fast , considering point set registration.

Table 5
Information for the compared methods.

Methods	Descriptor	Size	Measurement
MVCNN,12×	Relu7	86528	Euclidean Distance
MeshNet	Concat_MLP	1024	Euclidean Distance
USC	Maxpool	1960	Euclidean Distance

Table 6
Evaluation of downsampling strategy on our method.

parameters		random			HEM			benchmark
X	size	13073 × 3	2263 × 3	97 × 3	13073 × 3	2263 × 3	97 × 3	34835 × 3
	sampling rate(%)	37.528	6.496	0.278	/	/	/	/
	layers	/	/	/	3	3	10	/
	amendatory factor	/	/	/	2	4	10	/
	size	15283 × 3	2931 × 3	115 × 3	15283 × 3	2931 × 3	115 × 3	34835 × 3
	sampling rate(%)	43.873	8.414	0.330	/	/	/	/
	layers	/	/	/	3	3	10	/
	amendatory factor	/	/	/	2	4	10	/
	time of CPD(s)	336.35	12.09	0.28	374.95	18.27	0.63	1329.44
metrics	*LR*	0	0	9.9481×10^{-7}	0	0	0	0
	KURT	11618.84	1926.231	81.92	12632.84	2388.49	101.9082	34770
	CORR	0.99984	0.99861	0.97295	0.99982	0.99929	0.98486	0.99999

Table 7
Runtime of HEM and random downsampling.

	Horse		Bunny	
layers	5	/	5	/
amendatory factor	12	/	6.7	/
sampling rate (%)	/	0.471	/	1.521
size	532 × 3	530 × 3	546 × 3	530 × 3
time(s)	10.48	0.03	1.6359	0.02

Table 8
Evaluation of segmentation strategy on our method. The points item shows the number of points in the first $(n-1)$ block/ the number of points in the last block.

	Merlion			
threshold	base	3000	4000	5000
blocks	1	6	5	4
points	17705	2950/2955	3541/3541	4426/4427
IALM time(s)	13.3664	4.14514	4.03534	4.5393
memory	> 20G	> 1G	> 1G	> 1G
LR	0	0	0	0

Table 9
Evaluation of Gaussian noise attack.

μ	σ	LR	KURT	CORR
0	0.01	0	4392.32	0.99961
0	0.05	3.00×10^{-11}	1906.25	0.99575
0	0.1	6.26×10^{-10}	1264.37	0.98665
0	0.5	2.08×10^{-31}	526.64	0.94794
0	1	1.50×10^{-19}	277.18	0.94108

4.3.3. Segmentation

In this section, we evaluate the impact of the segmentation strategy on the LR measure. The Merlion model with 17,705 points and 35,414 faces and its homologous model after similar transformation are used for the experiments. The registered point clouds are first divided into multiple segments, and the LR distance is then computed for each segment pair. Finally, we merge the LR results of each segment and compare the effects of segment numbers on our method.

Table 8 reveals that more segments would lead to less time and memory but hardly impact LR distance. $> x$ means the value fluctuates around x within a stable and observable duration. The Merlion models in this experiment achieve good alignment through CPD, and thus the time required for the IALM operation is relatively short without the segmentation strategy. The memory is related to the size of the input matrix of the IALM algorithm, and the running time of the IALM is related to the complexity of the matrix.

4.3.4. Gaussian noise attack

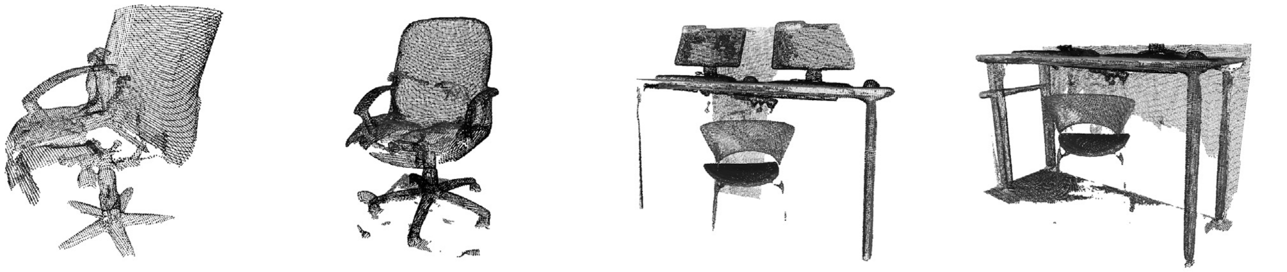
We conduct experiments on Gaussian noise attacks. We first take the Bunny model as the original model and then apply a Gaussian noise attack to it.

The attack process of the point cloud $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$ is as follows: first, we grab a $N \times 3$ matrix $\{\Delta\boldsymbol{r}_1, \ldots, \Delta\boldsymbol{r}_N\}$, each column of which is generated from a normal distribution with mean μ and standard deviation σ, followed by yielding a point cloud $\boldsymbol{Y} = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_N\}$ via translating the vertices of $\boldsymbol{X}$ with above displacements.

$$\boldsymbol{y}_i = (x_{i,1} + \Delta r_{i,1}, x_{i,2} + \Delta r_{i,2}, x_{i,3} + \Delta r_{i,3}), \tag{13}$$

where $\boldsymbol{x}_i = (x_{i,1}, x_{i,2}, x_{i,3}))$ and $\Delta\boldsymbol{r}_i = (\Delta r_{i,1}, \Delta r_{i,2}, \Delta r_{i,3})$.

Table 9 shows the results of our method under different levels of Gaussian noise attacks. As the noise increases, the performance decreases generally. The KURT decreases with a faster speed, but LR and CORR exhibit superior resistance. The LR metric is much



(a) ground truth chair (b) perturbed chair (c) ground truth table (d) perturbed table with background

Fig. 9. Some point clouds of the ScanObjectNN dataset.

Table 10
Evaluation result of scanned point clouds.

point cloud	LR	KURT	CORR
chair	4.81×10^{-5}	226.34	0.97472
table	5.29×10^{-5}	48.19	0.95327

smaller than the LR threshold (0.0006) mentioned in Section 4.3.1 as the noise increases from 0.01 to 1. Although the Gaussian attack may greatly affect the establishment of point-to-point correspondences, the LR metric performs well.

4.4. Scanned Objects

We demonstrate our performance on real-world scanned object point clouds. We select several point clouds from the ScanObjectNN dataset [49], a new real-world point cloud object dataset based on scanned indoor scene data. Fig. 9 shows some point clouds of the ScanObjectNN dataset.

Table 10 shows our results on chairs and tables shown in Fig. 9, with computing the distances between 9a and 9b and the distances between 9c and 9d, respectively. The LR distance can well resist attacks such as noise and rigid transformation that may exist in real data, while the CORR and KURT distances are less robust.

5. Conclusion

This work presents a robust method for copy detection of 3D object point clouds. It first registers two point clouds and then computes different distance measures between them. The distance measures reveal the similarity degree of the two point clouds, enabling the judgement of two same or different models. Extensive experiments show that our method generally achieves better outcomes than the state-of-the-art watermarking techniques and comparable performance to current 3D shape retrieval methods. We believe our work will inspire more insights in terms of copyrighting 3D shapes.

One main limitation of our work is the efficiency due to the complex computation in point set registration. In other words, our method is more suitable for offline processing. In the future, we would like to design efficient techniques like deep learning to enable fast processing (inference). We focused on 3D shapes rather than scene data in this work, and would like to investigate this kind of data in the future.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] E. Agosto, L. Bornaz, 3d models in cultural heritage: approaches for their creation and use, International Journal of Computational Methods in Heritage Science (IJCMHS) 1 (2017) 1–9.
[2] Y.B. Amar, I. Trabelsi, N. Dey, M.S. Bouhlel, Euclidean distance distortion based robust and blind mesh watermarking, IJIMAI 4 (2016) 46–51.
[3] S. Bai, X. Bai, Z. Zhou, Z. Zhang, Q. Tian, L.J. Latecki, Gift: Towards scalable 3d shape retrieval, IEEE Transactions on Multimedia 19 (2017) 1257–1271.
[4] V. Barra, S. Biasotti, 3d shape retrieval using kernels on extended reeb graphs, Pattern Recognition 46 (2013) 2985–2999.
[5] J.W. Cho, R. Prost, H.Y. Jung, An oblivious watermarking for 3-d polygonal meshes using distribution of vertex norms, IEEE Transactions on Signal Processing 55 (2006) 142–155.
[6] C.M. Chou, D.C. Tseng, Technologies for 3d model watermarking: A survey, International Journal of Computer Science and Network Security 7 (2007) 328–334.
[7] A. Delmotte, K. Tanaka, H. Kubo, T. Funatomi, Y. Mukaigawa, Blind 3d-printing watermarking using moment alignment and surface norm distribution, IEEE Transactions on Multimedia (2020).
[8] J. Fan, J. Yang, D. Ai, L. Xia, Y. Zhao, X. Gao, Y. Wang, Convex hull indexed gaussian mixture model (ch-gmm) for 3d point set registration, Pattern Recognition 59 (2016) 126–141.
[9] J. Feng, Y. Wang, S.F. Chang, 3d shape retrieval using a single depth image from low-cost sensors, in: 2016 IEEE Winter Conference on Applications of Computer Vision (WACV), IEEE, 2016, pp. 1–9.
[10] Y. Feng, Y. Feng, H. You, X. Zhao, Y. Gao, Meshnet: Mesh neural network for 3d shape representation, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2019, pp. 8279–8286.
[11] F.A. Ferreira, J.B. Lima, A robust 3d point cloud watermarking method based on the graph fourier transform, Multimedia Tools and Applications 79 (2020) 1921–1950.
[12] A. Frome, D. Huber, R. Kolluri, T. Bülow, J. Malik, Recognizing objects in range data using regional point descriptors, in: European conference on computer vision, Springer, 2004, pp. 224–237.
[13] T. Furuya, R. Ohbuchi, Deep aggregation of local 3d geometric features for 3d model retrieval., in: BMVC, 2016. p. 8.
[14] M. Hamidi, A. Chetouani, M. El Haziti, M. El Hassouni, H. Cherifi, A robust blind 3-d mesh watermarking technique based on scs quantization and mesh saliency for copyright protection, in: International Conference on Mobile, Secure, and Programmable Networking, Springer, 2019, pp. 211–228.
[15] M. Hamidi, M. El Haziti, H. Cherifi, D. Aboutajdine, A robust blind 3-d mesh watermarking based on wavelet transform for copyright protection, in: 2017 International Conference on Advanced Technologies for Signal and Image Processing (ATSIP), IEEE, 2017, pp. 1–6.
[16] X. He, Y. Zhou, Z. Zhou, S. Bai, X. Bai, Triplet-center loss for multi-view 3d object retrieval, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 1945–1954.
[17] J.U. Hou, D.G. Kim, H.K. Lee, Blind 3d mesh watermarking for 3d printed model by analyzing layering artifact, IEEE Transactions on Information Forensics and Security 12 (2017) 2712–2725.
[18] X. Huang, G. Mei, J. Zhang, Feature-metric registration: A fast semi-supervised approach for robust point cloud registration without correspondences, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 11366–11374.
[19] Z. Huang, Z. Zhao, H. Zhou, X. Zhao, Y. Gao, Deepccfv: Camera constraint-free multi-view convolutional neural network for 3d object retrieval, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2019, pp. 8505–8512.
[20] L. Jing, Y. Chen, L. Zhang, M. He, Y. Tian, Self-supervised modal and view invariant feature learning, 2020. arXiv preprint arXiv:2005.14169.
[21] B. Li, Y. Lu, C. Li, A. Godil, T. Schreck, M. Aono, M. Burtscher, Q. Chen, N.K. Chowdhury, B. Fang, et al., A comparison of 3d shape retrieval methods based on a large-scale benchmark supporting multimodal queries, Computer Vision and Image Understanding 131 (2015) 1–27.
[22] Z. Lin, M. Chen, Y. Ma, The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices, 2010. arXiv preprint arXiv:1009.5055.
[23] J. Liu, Y. Yang, D. Ma, A blind 3d point cloud watermarking algorithm based on azimuth angle modulation, in: 2018 11th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), IEEE, 2018, pp. 1–7.
[24] J. Liu, Y. Yang, D. Ma, W. He, Y. Wang, A novel watermarking algorithm for three-dimensional point-cloud models based on vertex curvature. International Journal of Distributed Sensor Networks 15, 2019. 1550147719826042.
[25] X. Lu, H. Chen, S.K. Yeung, Z. Deng, W. Chen, Unsupervised articulated skeleton extraction from point set sequences captured by a single depth camera, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2018.
[26] X. Lu, Z. Deng, J. Luo, W. Chen, S.K. Yeung, Y. He, 3d articulated skeleton extraction using a single consumer-grade depth camera, Computer Vision and Image Understanding 188 (2019) 102792.
[27] X. Lu, S. Wu, H. Chen, S.K. Yeung, W. Chen, M. Zwicker, Gpf: Gmm-inspired feature-preserving point set filtering, IEEE transactions on visualization and computer graphics 24 (2017) 2315–2326.
[28] N. Medimegh, S. Belaid, N. Werghi, A survey of the 3d triangular mesh watermarking techniques, International Journal of Multimedia 1 (2015).
[29] T.K. Moon, The expectation-maximization algorithm, IEEE Signal processing magazine 13 (1996) 47–60.
[30] M. Moyou, K.E. Ihou, A. Peter, Lbo-shape densities: Efficient 3d shape retrieval using wavelet density estimation, in: 2014 22nd International Conference on Pattern Recognition, 2014, pp. 52–57.
[31] A. Myronenko, X. Song, Point set registration: Coherent point drift, IEEE transactions on pattern analysis and machine intelligence 32 (2010) 2262–2275.
[32] A. Myronenko, X. Song, M.A. Carreira-Perpinán, et al., Non-rigid point set registration: Coherent point drift, Advances in neural information processing systems 19 (2007) 1009.
[33] M. Novotni, R. Klein, Shape retrieval using 3d zernike descriptors, Computer-Aided Design 36 (2004) 1047–1062.
[34] R. Ohbuchi, S. Takahashi, T. Miyazawa, A. Mukaiyama, Watermarking 3d polygonal meshes in the mesh spectral domain, Graphics interface (2001) 9–17.

[35] R. Osada, T. Funkhouser, B. Chazelle, D. Dobkin, Matching 3d models with shape distributions, in: Proceedings International Conference on Shape Modeling and Applications, IEEE, 2001, pp. 154–166.

[36] R. Osada, T. Funkhouser, B. Chazelle, D. Dobkin, Shape distributions, ACM Transactions on Graphics (TOG) 21 (2002) 807–832.

[37] D. Povey, L. Burget, M. Agarwal, P. Akyazi, K. Feng, A. Ghoshal, O. Glembek, N.K. Goel, M. Karafiát, A. Rastrow, et al., Subspace gaussian mixture models for speech recognition, in: 2010 IEEE International Conference on Acoustics, Speech and Signal Processing, IEEE, 2010, pp. 4330–4333.

[38] R. Preiner, O. Mattausch, M. Arikan, R. Pajarola, M. Wimmer, Continuous projection for fast l1 reconstruction, ACM Trans. Graph. 33 (2014), 47–1.

[39] Reynolds, D.A., 2009. Gaussian mixture models. Encyclopedia of biometrics 741.

[40] M. Rezaei, E. Gunpı̇nar, A k-means clustering based shape retrieval technique for 3d mesh models, Selçuk Üniversitesi Mühendislik, Bilim ve Teknoloji Dergisi 6 (2018) 114–128.

[41] R.B. Rusu, S. Cousins, 3d is here: Point cloud library (pcl), in: 2011 IEEE international conference on robotics and automation, IEEE, 2011, pp. 1–4.

[42] S. Salti, F. Tombari, L. Di Stefano, Shot: Unique signatures of histograms for surface and texture description, Computer Vision and Image Understanding 125 (2014) 251–264.

[43] V. Sarode, X. Li, H. Goforth, Y., Aoki, R.A. Srivatsan, S. Lucey, H. Choset, Pcrnet: Point cloud registration network using pointnet encoding, 2019. arXiv preprint arXiv:1908.07906.

[44] D. Shukla, M. Sharma, Watermarking schemes for copy protection: A survey, International Journal of Computer Science and Engineering Survey 3 (2012) 65.

[45] A. Som, K. Thopalli, K. Natesan Ramamurthy, V. Venkataraman, A. Shukla, P. Turaga, Perturbation robust representations of topological persistence diagrams, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 617–635.

[46] H. Su, S. Maji, E. Kalogerakis, E. Learned-Miller, Multi-view convolutional neural networks for 3d shape recognition, in: Proceedings of the IEEE international conference on computer vision, 2015, pp. 945–953.

[47] Tombari, F., Salti, S., Di Stefano, L., 2010. Unique shape context for 3d data description, in: Proceedings of the ACM workshop on 3D object retrieval, pp. 57–62.

[48] Y.Y. Tsai, Y.S. Tsai, I. Chi, C.S. Chan, et al., A vertex-based 3d authentication algorithm based on spatial subdivision, Symmetry 10 (2018) 422.

[49] M.A. Uy, Q.H. Pham, B.S. Hua, T. Nguyen, S.K. Yeung, Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data, in: Proceedings of the IEEE/CVF international conference on computer vision, 2019, pp. 1588–1597.

[50] P. Voigt, A. Von dem Bussche, The eu general data protection regulation (gdpr). A Practical Guide, 1st Ed., Cham: Springer International Publishing 10, 2017. 10–5555.

[51] K. Wang, G. Lavoué, F. Denis, A. Baskurt, Three-dimensional meshes watermarking: Review and attack-centric investigation, in: International Workshop on Information Hiding, Springer, 2007, pp. 50–64.

[52] K. Wang, G. Lavoué, F. Denis, A. Baskurt, X. He, A benchmark for 3d mesh watermarking, in: 2010 Shape Modeling International Conference, IEEE, 2010, pp. 231–235.

[53] J. Wright, A. Ganesh, S. Rao, Y. Peng, Y. Ma, Robust principal component analysis: Exact recovery of corrupted low-rank matrices via convex optimization, Advances in neural information processing systems (2009) 2080–2088.

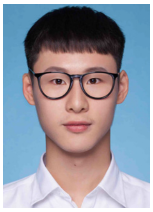
[54] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, J. Xiao, 3d shapenets: A deep representation for volumetric shapes, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 1912–1920.

[55] Y.P. Xiao, Y.K. Lai, F.L. Zhang, C. Li, L. Gao, A survey on deep geometry learning: From a representation perspective, Computational Visual Media 6 (2020) 113–133.

[56] J. Yang, X. Yuan, X. Liao, P. Llull, D.J. Brady, G. Sapiro, L. Carin, Video compressive sensing using gaussian mixture models, IEEE Transactions on Image Processing 23 (2014) 4863–4878.

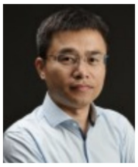
[57] L. Zhang, M.J. da Fonseca, A. Ferreira, C.R.A. e Recuperaçao, Survey on 3d shape descriptors. FundaÃgao para a Cincia ea Tecnologia, Lisboa, Portugal, Tech. Rep. Technical Report, DecorAR (FCT POSC/EIA/59938/2004) 3, 2007.

[58] W. Zhang, C. Xiao, Pcan: 3d attention map learning using contextual information for point cloud based retrieval, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 12436–12445.



Jiaqi Yang received a master's degree in computer science and technology from the School of Computer Science, Zhejiang University. His current research interests include computer graphics and other fields.

Xuequan Lu is an Assistant Professor at the School of Information Technology, Deakin University, Australia. He spent more than two years as a Research Fellow in Singapore. Before that, he earned his Ph.D. at Zhejiang University (China) in June 2016. His research interests mainly fall into visual computing, for example, geometry modeling, processing and analysis, animation/simulation, 2D data processing, and analysis. More information can be found at http://www.xuequanlu.-com.



Wenzhi Chen (Member, IEEE) received a Ph.D. degree from the College of Computer Science and Engineering, Zhejiang University. He is currently a Professor with the College of Computer Science and Technology, Zhejiang University, and the Director of the Information Technology Center, Zhejiang University. He used to be the Vice Dean of the College of Computer Science and Technology. His current research interests include embedded systems and their application, computer architecture, computer system software, and information security. He is a member of ACM and the ACM Education Council.