Detecting 3D Points of Interest Using Projective Neural Networks

Zhenyu Shu, Sipeng Yang*, Shiqing Xin, Chaoyi Pang, Xiaogang Jin, Ladislav Kavan, and Ligang Liu

***Abstract*—Detecting points of interest on 3D shapes is a fundamental research problem in geometry processing. Due to the complicated relationship between points of interest and their geometric features, detecting points of interest on any given 3D shape remains challenging. Due to the lack of training data, previous data-driven methods for detecting 3D points of interest mainly focus on utilizing hand-crafted geometric features to predict the probabilities of each point being a POI, which greatly limits detection performance. In this paper, we propose a novel algorithm for detecting 3D points of interest by using projective neural networks. Our method first projects the labeled training 3D shapes into multiple 2D views and then learns the required features from the 2D views in an end-to-end fashion. The points of interest on test 3D shapes are then automatically detected by applying the learned neural network and our improved density peak clustering. Our method relies neither on hand-crafted feature descriptors nor a large quantity of expensive 3D training data to obtain satisfactory results. Experimental results show significantly superior detection performance of our method over the state-of-the-art methods.**

***Index Terms*—3D shapes, Point of interest, Convolutional neural networks**

I. Introduction

POints of interest (POIs), also known as feature points, are usually defined as distinctive points on the surface of 3D shapes. POIs play a crucial role in many geometry processing tasks, such as viewpoint selection [1], shape enhancement [2], shape retrieval [3], [4], [5], mesh registration [6], [7] and mesh segmentation [8].

POIs can be easily distinguished from other points on 3D shapes by human visual perception. However, it is not easy to accurately define POIs from a geometric perspective, although they are definitely related to geometric features [9], [10]. Therefore, automatically detecting POIs conforming to human visual perception remains a challenging problem.

Zhenyu Shu and Chaoyi Pang are with School of Computer and Data Engineering, NingboTech University, Ningbo, PR China. They are also with Ningbo Institute, Zhejiang University, Ningbo, PR China.
E-mail: shuzhenyu@nit.zju.edu.cn (Zhenyu Shu)

Sipeng Yang is with School of Mechanical Engineering, Zhejiang University, Hangzhou, PR China. Corresponding author.
E-mail: 21825205@zju.edu.cn (Sipeng Yang)

Shiqing Xin is with School of Computer Science and Technology, Shandong University, Jinan, PR China.

Xiaogang Jin is with State Key Lab of CAD&CG, Zhejiang University, Hangzhou, PR China.

Ladislav Kavan is with School of Computing, University of Utah, Salt Lake City, USA.

Ligang Liu is with Graphics & Geometric Computing Laboratory, School of Mathematical Sciences, University of Science and Technology of China, Anhui, PR China.

Manuscript received month day, year; revised month day, year.

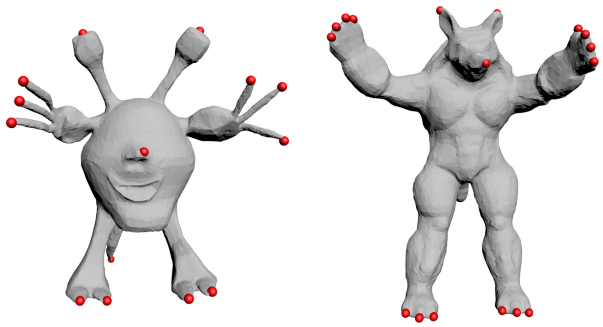


Fig. 1. POIs detected by our approach.

There is a common understanding that judging whether a point is a POI is subjective because different people may have different opinions about POIs. Based on the above observation, data-driven methods are applied to efficiently detect POIs on 3D shapes [9]. With recent advances in related research areas, deep learning has been introduced for detecting POIs on 3D shapes [11] to obtain satisfactory results by learning a complicated mapping between the geometric features and the probability value of being a POI for each point on the surface.

However, the scarcity of 3D training data forces learning-based methods to heavily rely on hand-crafted geometric features instead of directly learning features from raw data because acquiring high-quality training data for 3D shapes is much more expensive than acquiring 2D images. This greatly limits them from gaining better detection performance. In addition, the discrepancy between subjective human visual perception and geometric features on 3D shapes also prevents this kind of method from achieving a more satisfactory performance.

In this paper, we propose a novel algorithm for detecting 3D points of interest on 3D shapes with projective neural networks. Our method learns features in an end-to-end way and can still lead to robust results despite a limited number of training 3D shapes, which is very different from other deep learning-based POI detection methods for 3D shapes. It is worth noting that our approach can accurately capture features that agree with human perception by learning from multiple projected 2D shaded images and depth images, which naturally corresponds to human vision.

Given a training 3D shape, our method generates a probability distribution of each point being a POI on the surface and maps it into multiple 2D views. The probability distribution information is then robustly learned from them to construct a projective neural network in our method. Finally, the POIs on

test 3D shapes are predicted by applying the learned projective neural network and our improved density peak clustering method. Figure 1 shows an example of POIs detected by our method.

Our method is evaluated on the SHREC 2011 non-rigid 3D shape dataset [12]. The dataset contains 600 different 3D shapes, and each one is manually labeled by volunteers to obtain the ground truth data. Extensive experimental results show that our method obtains significantly better performance than the state-of-the-art with various measures, including false negative error (FNE), weighted miss error (WME), false positive error (FPE), area under the ROC curve (AUC), and linear correlation coefficient (LCC).

Our contributions are two-fold:

- Our method is end-to-end, robust, and can achieve better performances than traditional methods; it can obtain a good performance even if only a small number of training samples are provided.
- Our method is completely data-driven, and the performance can be improved further as long as sufficiently many labeled ground truth samples are provided.

The remainder of the paper is organized as follows. In Section II, we introduce the related work on POI detection. The overall workflow of our proposed method is described in Section III, followed by the details of our projective neural networks. After that, we show extensive experimental results, as well as a comparison to the state-of-the-art methods, in Section IV. Finally, we discuss the limitations and future work in Section V and conclude this paper in Section VI.

II. Related work

In this section, we review the different algorithms and technologies related to our work. First, we review the previous research work on POI detection of 3D shapes. Second, we discuss the topic of mesh saliency, which is very similar to POI detection. Third, we introduce some applications of projection-based methods in the digital geometry processing field.

A. POIs detection

3D POI detection aims to extract distinctive salient points aligned with human perception. According to the differences in distinguishing features, previous 3D POI detection algorithms can be divided into two kinds: hand-crafted feature-based algorithms and learning-based algorithms. Early work mostly used hand-crafted features to extract the most representative points on 3D shapes. Recently, machine learning has been introduced into the geometry processing area and shows superior performance in detecting 3D POIs.

Early on, studies relied on various hand-crafted features to detect POIs of 3D shapes, such as shape diameter function (SDF), scale-invariant heat kernel signatures (SIHKS), and wavelet kernel signatures (WKS). For example, Gelfand *et al.* [13] proposed the integral volume descriptor, which is invariant to rotation and translation of the model based on local geometry for each vertex. A small set of vertices can be selected as the POIs according to the uniqueness of the integral volume descriptor. Castellani *et al.* [14] used a statistical descriptor, which contains the 3D saliency measure definition, multi-scale representation, and feature point detection. They extracted sparse salient points by a statistical learning approach using contextual 3D neighborhood information. Zou *et al.* [15] defined another shape descriptor that extracts the POIs with detected scales. In their approach, salient points are detected from the surface of shapes using the difference of Gaussian (DoG) function in geodesic scale space. Instead of computing the descriptors on meshes, Godila *et al.* [16] extracted geometric features of 3D shapes based on the voxel grid. They identified invariant POIs inspired by the scale-invariant feature transform (SIFT) algorithm. DoG function is also used to compute the saliency of 3D shapes.

In recent years, machine learning has been extensively used in digital geometry processing. Creusot *et al.* [17] presented a pioneer learning-based POI detection method. Their research focused on 3D face shapes. By using various geometrical descriptors, the machine learning model can generate the key point score map with respect to the human-labeled POIs. Teran *et al.* [18] proposed an approach to detect POIs by using machine learning. They formulated the POI detection task as a binary classification problem of vertices on the surface. Their method extracted POIs from an unlabeled model by a trained random forest classifier. In particular, the labeled POIs are only a small part of the vertices, and they use resampling to address the imbalanced learning problem. Similar to [18], Shu *et al.* [11] also employed the stacked auto-encoder algorithm. They devised two separate deep neural networks with SAE to learn and predict POIs on 3D shapes. The first deep neural network was trained to predict the category of each shape. The second deep neural network was trained to predict the probability of each vertex being a point of interest. Finally, they used a clustering center detection method to extract the peak points of the high probability areas on the surface as POIs. You *et al.* [19] proposed an automatic aggregation method to detect POIs. The correspondences between the POIs and the vertices on the test 3D shapes can be generated through the minimization of a fidelity loss.

B. Mesh saliency

The mesh saliency detection task is very close to POI detection. The former aims to extract saliency regions on the mesh rather than saliency points. Liu *et al.* [20] classified previous mesh saliency detection methods into two classes: local contrast (LC) methods and global contrast (GC) methods.

Local contrast (LC) methods assume that the salient regions should be distinctive from their surrounding areas [21]. Thus, mesh saliency can be detected by computing geometric feature deviations from adjacent regions. Similar to POI detection methods, Gal *et al.* [22] and Jia *et al.* [23] defined new local surface descriptors to calculate the mesh saliency. Inspired by Harris *et al.* [24], who proposed a feature point detection method for 2D images, [25] and [26] extended the Harris operator into saliency region detection of 3D shapes. Tasse *et al.* [27] proposed a cluster-based mesh saliency detection method that can attain good results without using any topological information.

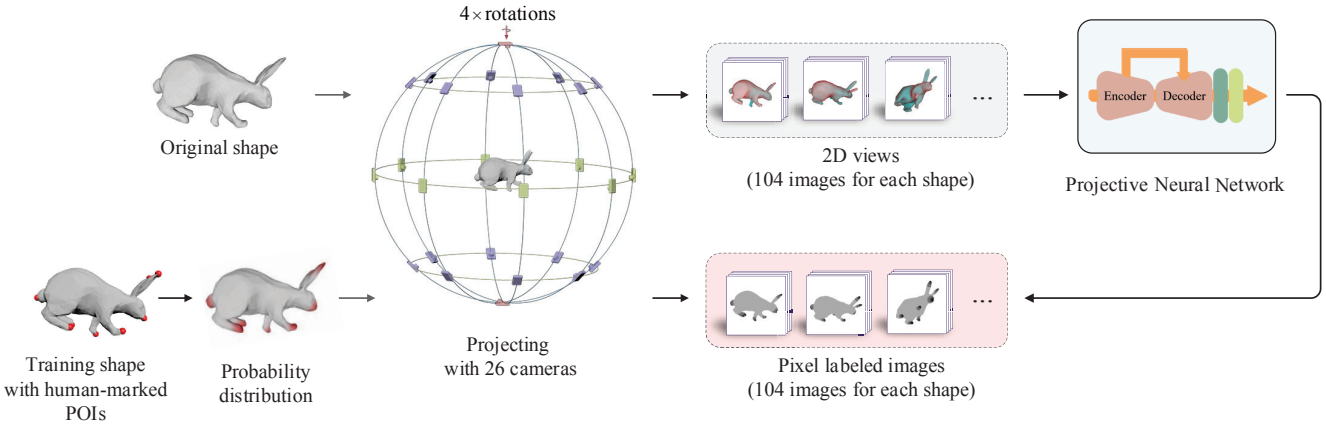


Fig. 2. The overall workflow of our algorithm. In the training phase, the POIs are expanded onto the surface of 3D shapes. The shapes are projected into multiple 2D views, and a deep convolutional neural network is trained to capture the POIs' distribution in the 2D views. In the test phase, our algorithm predicts the probability distribution of each POI on multiple projected 2D views of the test shape. The final predicted POIs on the surface of the test 3D shape are obtained by merging the corresponding predicted results on 2D views.

Global contrast (GC) methods aim to detect salient components from an overall model perspective. Song *et al.* [28] proposed a mesh saliency detection method based on the conditional random field framework. Sipiran *et al.* [29] presented a method to detect key components on 3D meshes. The 3D Harris operator was employed to distinguish the key components and common parts. Jeong and Sim [30] provided a view-independent and view-dependent unified mesh saliency detection method. By using a face-based structure detector, they obtained multi-scale saliency on semi-regular meshes.

C. Projection-based 3D shape analysis

Projection-based methods are important in 3D shape analysis. For example, Su *et al.* [31] proposed a projection-based method for 3D shape recognition by using convolutional neural networks. Their method projects a 3D shape into 12 different views, and a view pooling layer is used to connect the convolutional neural networks. To address the high computational cost of the projection-based method, Yang *et al.* [32] presented a network structure combining a multi-view and extreme learning machine auto-encoder. Their algorithm performs well in both classification accuracy and computational efficiency. Kalogerakis *et al.* [33] introduced a projection-based method for shape segmentation. They used more projections at different distances and angles to train fully convolutional network modules. Depth images containing distance information from cameras to different regions of the model are also used to improve the performance.

III. Method

In this section, we introduce our POI detection method in detail. The overall workflow of our algorithm is shown in Figure 2. Our approach consists of two steps. In the training step, each training 3D shape is mapped into multiple 2D views, and convolutional neural networks are trained to extract features from them. After that, a probability distribution of POIs on a test 3D shape can be predicted by applying the trained neural networks. The POIs on a test 3D shape are finally extracted by applying our improved density peak clustering method.

A. Training data preparation

Providing a sufficient quantity of training data is one of the key factors to obtain satisfactory performance for most of the data-driven approaches, especially for deep learning-based approaches. However, due to the complexity of 3D shapes, preparing 3D training data for geometry processing tasks is usually a very expensive and tedious task. To facilitate extracting the desired features for POI detection with only a few training samples, we use the weights of pretrained convolution layers when building our convolutional neural networks. Similar to Su *et al.* [31], our method projects the training 3D shapes into multiple 2D views and trains the convolutional neural networks in an end-to-end way on the projected 2D views.

The input of projective neural networks. The input images of our projective neural networks are the projections of 3D shapes in different directions. Figure 3 presents our projection process for a Rabbit model. For each shape, we defined 26 virtual cameras in different positions. The cameras are placed at distances of the shape's bounding sphere radius. We randomly set a direction as the initial azimuth and put the first camera (camera 1) at that position. Then, another 7 virtual cameras (cameras 2 to 8) are fixed along the equator at every 45-degree angle. When the longitude of the elevation angle reaches 45 and -45 degrees, another 16 virtual cameras (cameras 9 to 16 and cameras 17 to 24) are placed in the same azimuths as cameras 1 to 8. The last 2 cameras are placed on the poles. In addition, each camera is rotated 4 times in 90-degree intervals to increase the training dataset.

After obtaining the shaded and depth images for each 3D shape, we transform the single-channel images into a three-channel image. As shown in Figure 3, we set the positive depth images to the first channel as the input images for

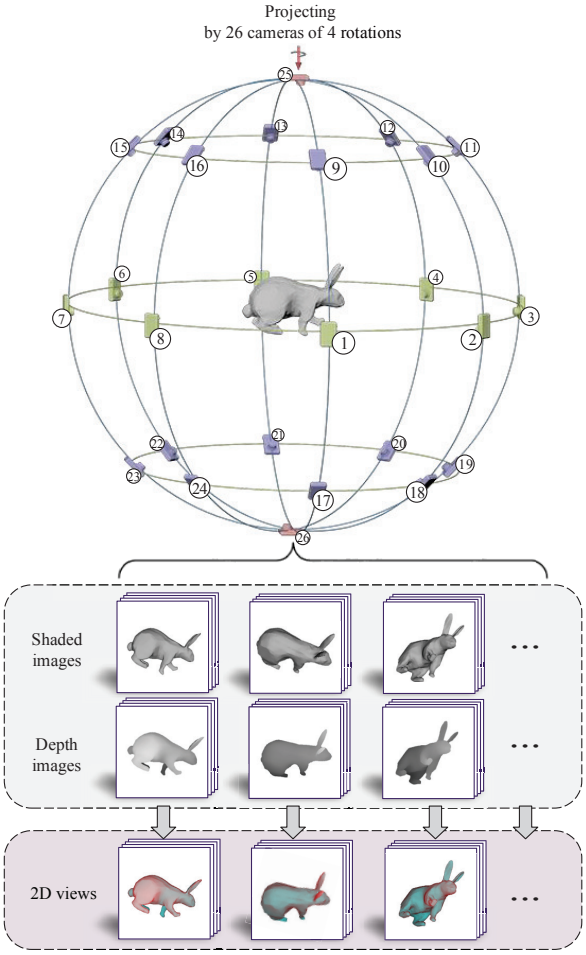


Fig. 3. The projection process. We set 26 cameras around the 3D shape to render shaded images and depth images. Then, the two kinds of single-channel 2D images are combined into three-channel images.

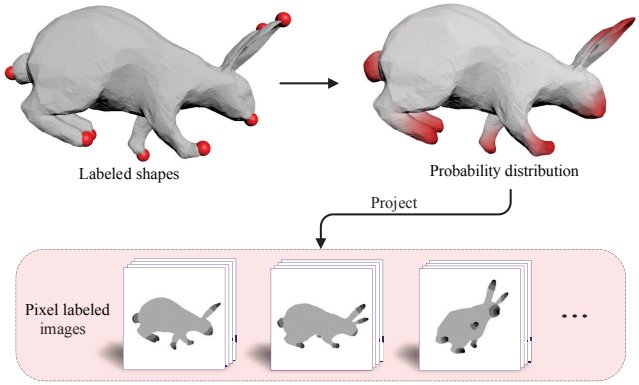


Fig. 4. The smoothing process of POIs. We expand the human-labeled POIs into a probability distribution on the surface and then project it into 2D images.

projective neural networks, set the shaded image to the second channel, and set the negative depth image to the third channel. Examples of the input images generated in this way are shown at the bottom of Figure 3. In this paper, 104 projections are generated for each 3D shape in total, which can cover almost all vertices and facets for each shape in the dataset used in our experiments, although more projections can be used if desired.

The output of projective neural networks. The outputs of projective neural networks are the corresponding projected images of labeled shapes at the same elevation and azimuth. However, human-labeled POIs are usually a few vertices on the mesh. Therefore, directly using the projected images of those labeled shapes in the training process will lead to an extremely imbalanced distribution of positive and negative samples. Thus, we construct a probability distribution P on the surface of each 3D shape based on normal probability density. P is generated as a distribution decaying with the geodesic distances from their nearest POIs. We define P on any vertex v_i as:

$$P(v_i) = \frac{1}{\sigma\sqrt{2\pi}} \exp^{-(d(v_i,p_i))^2/\left(2\sigma^2\right)}, \tag{1}$$

where $d\left(v_i, p_i\right)$ is the geodesic distance from the vertex v_i to its nearest POI p_i. σ is a parameter that can be used to control the decay speed. In this paper, σ is set to one-fifth of the maximum geodesic distance. A visualized example of P can be found in Figure 4. The closer to the POIs, the higher the value of P the vertex has. In our method, the probability distribution P is mapped to all 2D images projected from 3D shapes, which are used as the output of our projective neural networks. Examples of the output images are shown at the bottom of Figure 4.

B. Training and predicting

Training process. In the training process, the projective neural networks take the 2D images of 3D shapes as input and the corresponding labeled images as output. Our neural network tries to learn the mapping from each pixel of input images, which contains shading and depth information of 3D shapes, to the corresponding pixel of output images, which contains probability distribution information of whether a vertex is a POI.

We construct a projective neural network to predict the probability distributions on 2D views of shapes. As shown in Figure 5, we design the convolutional layers (encoder network) as the feature extractor of 2D views. The Conv + BN + ReLU and pooling layers constitute the encoder network. The decoder network is constructed with upsampling and Conv + BN + ReLU layers. The upsampling layers receive the corresponding pooling indices (i.e., positions of the largest values in each pooling operation) from the pooling layers. Each feature value will be set in the position according to the pooling indices to produce sparse feature maps. The Conv layers after the upsampling layers are used to produce dense feature maps. Finally, the feature maps are fed to the softmax layer to classify each pixel independently.

It is worth noting that even though we construct the probability distribution, instead of directly using the projected images (usually binary images) of labeled shapes, we still face imbalanced distributions in training samples. In our experiments, we observe that the positive samples (pixels) are only approximately 7.5 percent of the output images. This negatively impacts the prediction performance of our network. Therefore, we adopt class weighting to balance

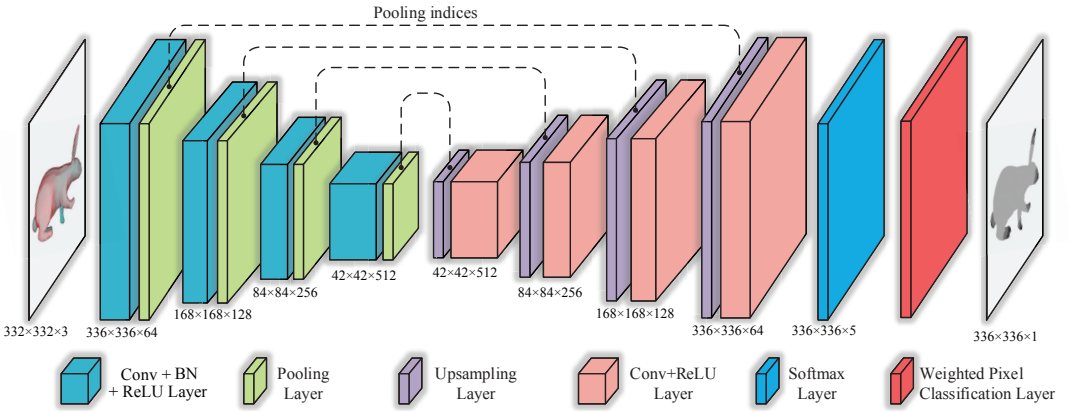


Fig. 5. Our projective neural network. The first four Conv + ReLU and pooling layers can be seen as an encoder network. The four upsampling and Conv + ReLU layers can be seen as a decoder network. The upsampling layers receive corresponding indices from the pooling layers. According to the proportion of different classes, the weighted pixel classification layer is added after the softmax layer for class balancing.

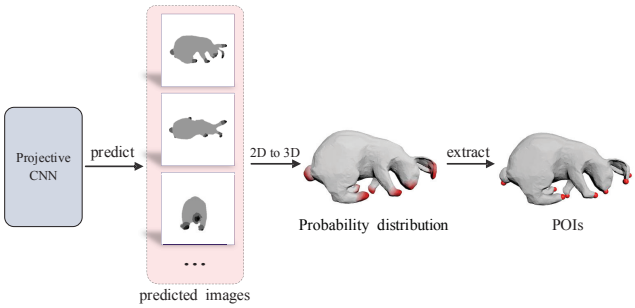


Fig. 6. The predicted probability distribution for a Rabbit shape. A deep convolutional neural network is used to predict the probability distribution of POIs on the 2D views in our algorithm. The corresponding probability distribution of POIs on the surface of 3D shapes is obtained by merging information from all 2D views.

different classes of samples. In our method, the weight w_k for the kth class of samples is defined as:

$$w_k = \frac{1}{W_k / \sum_k W_k}, \tag{2}$$

where W_k represents the number of samples (pixels) in the kth class for an image. Then, the weights w_k are used to establish a weighted pixel classification layer with cross-entropy loss. The loss function is defined as:

$$L = -\sum_m \sum_k w_k t_{km} \log y_{km}, \tag{3}$$

where m is a pixel on the output image, t_{km} is the ground-truth probability that pixel m belongs to the kth class, and y_{km} is the predicted probability that pixel m belongs to the kth class. The last layer of our network, the weighted pixel classification layer, is composed of w_k, i.e., the weights of classes for each pixel.

Prediction process. Given a test 3D shape, our method first projects it into a set of 2D images that contain shading and depth information, as described in Section III. Then, predicted probability distributions are obtained by applying our trained neural network.

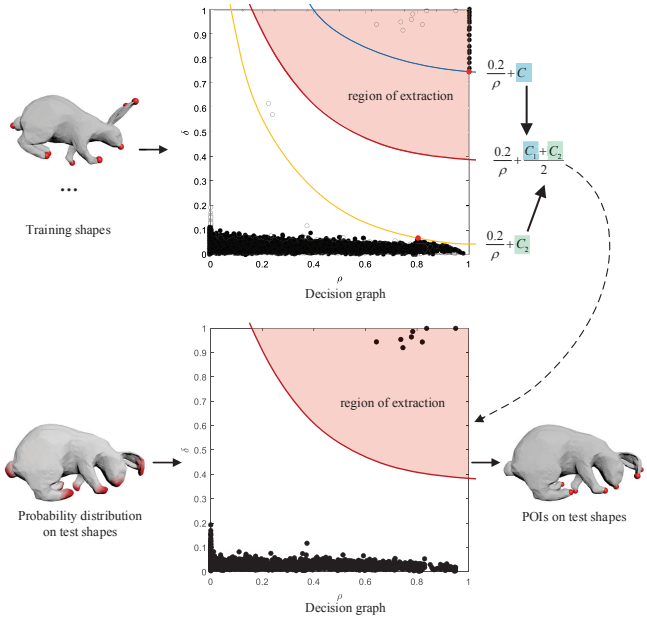


Fig. 7. The extracting process of POIs.

After obtaining probability distributions of whether a vertex is a POI in the form of a set of 2D images, our method directly maps the probability distributions back to the surfaces of 3D shapes according to the relationship between pixels of 2D images and vertices of 3D shapes recorded during the projection. Note that one vertex may occur in multiple projected 2D images. We use the following strategy to determine the predicted probability Q_i of whether the vertex v_i is a POI or not on a 3D shape.

$$Q_i = \begin{cases} 0, & \text{no pixel corresponds to vertex } v_i, \\ \frac{\sum_{j=1}^{n} q_{i,j}}{n}, & n \text{ pixels correspond to vertex } v_i, \end{cases} \tag{4}$$

where $q_{i,j}$ is the predicted value on the jth pixel corresponding to the vertex v_i. Most of the vertices of 3D shapes can be found in at least one projected image, and the corresponding probability can be calculated as the average value of pixels. However, there may still be a few vertices that cannot be

observed in any projected images. Usually, increasing the number of views in the projection process can help to alleviate this situation. Alternatively, we set the corresponding values to 0 by default. An example of the predicted probability distribution Q_i is shown in Figure 6, where the vertices in the red region have high probabilities of being POIs.

C. Detecting the POIs

We obtained the probability distribution of being a POI for each vertex on test shapes. In general, POIs do not contain special features that can be used to distinguish them from ordinary vertices. Our goal is to detect the specific POIs on shapes by the predicted probability distribution. It can be understood as extracting the points with the local highest probability values on the surfaces of 3D shapes. In our method, we propose an improved density peak clustering algorithm to extract the POIs from predicted probability distributions. The main idea of the traditional density peak clustering method [34] is to find those points with local maximal density. However, the method lacks mechanisms to automatically extract those peak points, although it can provide some visual hints to make distinguishing peak points easier. Therefore, we improve the method and build a novel method for automatically extracting the desired peak points in this paper.

For each vertex v_i, let ρ_i denote the corresponding probability value of v_i. On the other hand, we define δ_i as the influence radius of v_i:

$$\delta_i = \log\left(\min_{j:Q_j>Q_i}(d(v_i,v_j))+1\right), \tag{5}$$

where $d(v_i, v_j)$ is the geodesic distance between vertices v_i and v_j. Actually, $\min_{j:Q_j>Q_i}(d(v_i, v_j))$ denotes the geodesic distance between vertex v_i and its nearest vertex v_j, which has a higher probability value than v_i. We map all the vertices on a 3D shape to a two-dimensional decision graph, where the horizontal axis and the vertical axis represent ρ_i and δ_i, respectively. On the decision graph, the point at the upper-right corner that has both large δ and large ρ are regarded as good candidates for being POIs. As shown in Figure 7, a region on the upper-right corner of the decision graph needs to be determined to extract the POIs automatically.

In our method, a data-driven strategy is used to determine the corresponding ranges on the decision graph. In the training phase, we learn a separation curve from the ground truth for each category of training shapes, which can be used to separate the POIs from other ordinary vertices in the decision graph. As shown in Figure 7, two reciprocal functions are used to find the separation curve for POIs and other ordinary vertices (POIs always appear at the upper-right corner, while the other ordinary vertices appear around the two axes). The two functions are defined as:

$$\begin{cases} curve_1 : \delta = \frac{0.2}{\rho} + C_1, \\ curve_2 : \delta = \frac{0.2}{\rho} + C_2, \end{cases} \tag{6}$$

where ρ and δ are the axes. C_1 and C_2 are variables used to move the curves up and down. We determine the variable C_1 by moving down the $curve_1$ as much as possible and making sure it goes through at least one POI. Similarly, we move up $curve_2$ and ensure that it goes through at least one ordinary vertex on the decision graph to determine C_2. We define the separation curve as

$$curve_3 : \delta = \frac{0.2}{\rho} + \frac{C_1 + C_2}{2}. \tag{7}$$

In the testing phase, the learned $curve_3$ can be used to determine the region where POIs appear. The vertices positioned to the right and above of $curve_3$ are regarded as POIs. We map the extracted points back to 3D shapes, and POIs are finally detected.

D. Algorithm

The training and testing process of our POI detection method is executed on each category of shapes. Our method can be summarized as follows.

Algorithm: POIs detection method

Inputs:
Training 3D shapes and human-labeled POIs
Outputs:
POIs on test 3D shapes

Training process:
Step 1: Obtain 2D views of 3D shapes and generate corresponding input 2D images by combining shaded images and depth images;
Step 2: Compute probability distribution P of POIs and obtain labels for every pixel on the images;
Step 3: Train the convolutional neural network using the training data prepared in Step 1 and 2.

Testing process:
Step 1: Generate 2D views for test shapes;
Step 2: Predict the probability distributions with the trained neural network;
Step 3: Obtain probability distributions on the surfaces of 3D shapes;
Step 4: Extract POIs on test shapes using our improved density peak clustering algorithm.

IV. Evaluation

In this section, we present experimental validation and analysis of our method compared with other state-of-the-art approaches.

A. Dataset and evaluation metrics

To evaluate the performance of our method, we employ data from the SHREC 2011 non-rigid 3D shape dataset [12], the SHREC 2014 non-rigid 3D human model dataset, and the SHREC 2020 non-rigid shape dataset. The SHREC 2011 dataset is an open dataset that consists of 600 non-rigid 3D shapes in 30 categories and 20 3D shapes in each category.

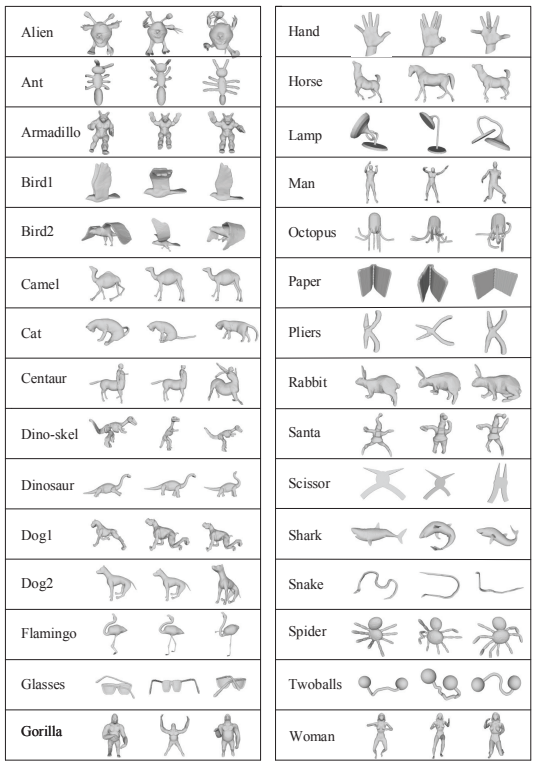


Fig. 8. Some example models of the SHREC 2011 non-rigid 3D shapes dataset.

Some example shapes are shown in Figure 8. The SHREC 2014 dataset consists of 400 3D human shapes that are obtained by scanning real human volunteers. The SHREC 2020 dataset is also obtained by scanning the real objects. It contains 11 partial scans and one full scan of a stuffed toy Rabbit. The SHREC dataset is widely used for benchmarking various geometry processing algorithms, such as 3D model classification [35], segmentation [36], and POI detection [11]. We first test the performance of our algorithm on the SHREC 2011 dataset. Ten shapes are randomly selected as the training shapes for each category of shapes, and the remaining 10 shapes are regarded as the test shapes. The training and testing data in our test are obtained from manual labeling. We asked 5 volunteers to label the POIs on the surface of 3D shapes using a small application we developed.

The evaluation method proposed by Dutagaci *et al.* [10] is adopted to measure the performance of different algorithms. This evaluation method consists of three evaluation metrics, the false negative error (FNE), false positive error (FPE), and weighted miss error (WME). We express the set of ground truth POIs as G and express the set of POIs detected by algorithms as A. Assume that N_g is the number of points in set G, and N_a is the number of points in set A. By setting a localization error tolerance radius r, the "correctly detected" points a by algorithms can be defined as the points in the $r-$neighborhood of ground truth points g. The "corrected detected" point set is expressed as C, i.e., $C(g_i) = \{g_i \in G | d(a, g_i) \leq r\}$, where $d(a, g_i)$ corresponds to the geodesic distance between points a and g_i. N_c denotes the number of "corrected detected" points by algorithms. While n_i subjects have marked the p_i as a POI, those three metrics can be defined as:

$$\begin{cases} FNE(r) = 1 - \frac{N_c}{N_g}, \\ FPE(r) = 1 - \frac{N_c}{N_a}, \\ WME(r) = 1 - \frac{\sum_{i=1}^{N_g} n_i o_i}{\sum_{i=1}^{N_g} n_i}, \end{cases} \tag{8}$$

where

$$o_i = \begin{cases} 1 & \text{if } g_i \text{ is detected by the algorithm,} \\ 0 & \text{otherwise.} \end{cases}$$

FNE and FPE evaluate the algorithms' performance by counting the "correctly detected" points or missing points in the localization error tolerance radius r. WME analyzed the significance level of every ground truth point of human perception. A better detection method achieves lower FNE, FPE, and WME.

In addition, we also adopt another evaluation framework presented in [37]. The authors introduced two metrics for quantitative analysis and comparison of mesh saliency detection methods, including area under the ROC curve (AUC) and the linear correlation coefficient (LCC). The receiver operating characteristic (ROC) curve illustrates the accuracy of true positive POI detection with different discrimination thresholds. AUC is the area under the ROC curve and varies between 0 and 1. A higher AUC value means better performance of the detection method. LCC evaluates the strength of the linear relationship between the shape saliency map of the detection method and its ground truth saliency. Similar to the two above methods, the saliency detection method with a higher LCC value performs better.

B. Experimental results

As mentioned in Section III, our method detects the POIs of 3D shapes in a supervised way. In the first stage, the 3D training shapes and the labeled POIs are projected into 2D views. In the second stage, we train neural networks to predict the probability distribution of being POIs for each vertex. Our networks are trained by mini-batch stochastic gradient descent (SGD). The learning rate is initialized as 0.01. We set the decay rate of the learning rate to 0.9 and set decay steps to every 10 steps. It is worth noting that the initial weights are loaded from the pretrained encoder-decoder model. We first train our projective convolutional neural network using the CamVid [38] dataset. Then, the weights of the corresponding convolution kernels are loaded into the POI detection task model as initial weights. In our experiments, using the pretrained weights can make the model converge faster and achieve better POI detection performance. The POIs are finally extracted by clustering. Figure 9 presents some specific detection results of our method. In the figure, we can see that satisfactory probability distributions are well predicted for test shapes, from which one can easily distinguish the desired POIs from other vertices. POIs on the test shapes are extracted by the learning-based density peak extraction

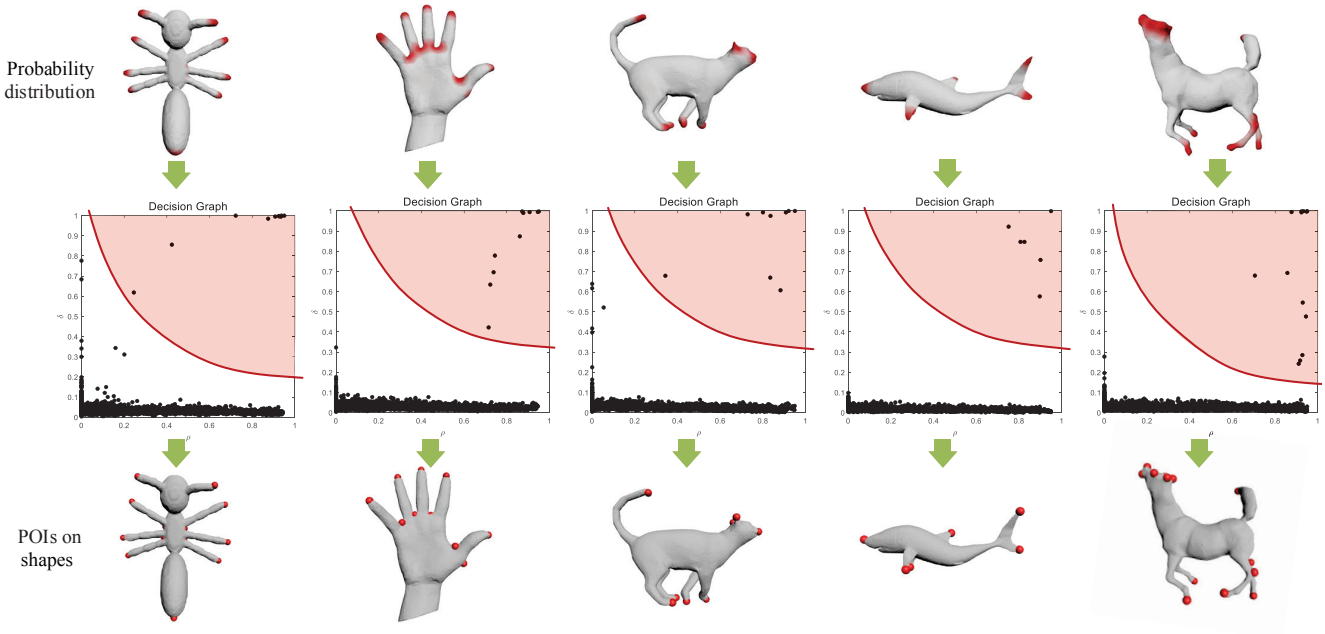


Fig. 9. Some representative POI detection results of our method.

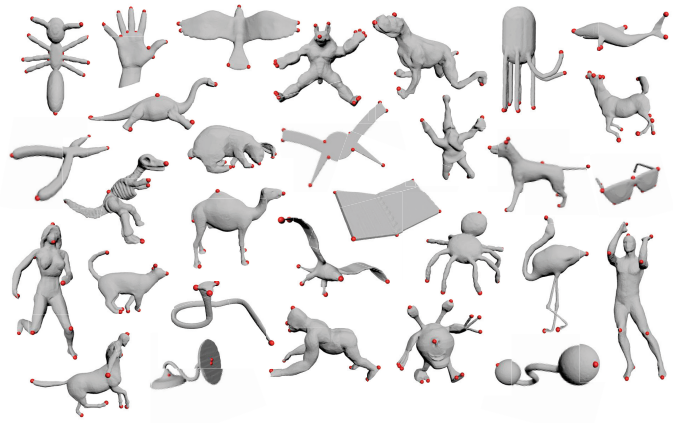


Fig. 10. Some representative POI detection results of our method on the SHREC 2011 dataset.

strategy, which is explained in Section III. Figure 10 shows the representative POI detection results of our method. In the figure, we can see that the results of our method are very consistent with human visual perception.

Next, we quantitatively evaluate the performance of our method using the metrics *FNE*, *FPE*, and *WME*. Table I presents measured numerical results with different localization error tolerance radius r. Each value in Table I is averaged over all shapes in 30 categories. As mentioned in Subsection IV-A, given a localization error tolerance radius r, lower scores of *FNE*, *FPE*, and *WME* represent the better performance of the detection method. We can see that the three error rates decrease rapidly when the localization error tolerance r increases, although they are relatively high when r is very small. A rapid decrease in the three metrics means our method catches the POIs with a low localization error. This indicates

TABLE I
EVALUATION OF OUR POI DETECTION RESULT ON THE SHREC 2011 DATASET USING THE EVALUATION METRICS *FNE*, *FPE*, AND *WME* WITH DIFFERENT LOCALIZATION ERROR TOLERANCE r.

Tolerance radius r	Average FNE	Average FPE	Average WME
0.00	0.8802	0.9427	0.8936
0.01	0.5623	0.6233	0.4756
0.02	0.2865	0.3895	0.2789
0.03	0.1360	0.3324	0.1865
0.04	0.1050	0.2857	0.1437
0.05	0.1033	0.2450	0.1220
0.06	0.1009	0.2144	0.1014
0.07	0.0974	0.2017	0.0929
0.08	0.0943	0.1985	0.0850
0.09	0.0919	0.1968	0.0805
0.10	0.0831	0.1920	0.0775
0.11	0.0795	0.1918	0.0732
0.12	0.0759	0.1869	0.0690
0.13	0.0751	0.1824	0.0675
0.14	0.0750	0.1786	0.0652

that the POIs detected by our method are very close to the ground truth in most cases. In addition, the three error rates do not decrease significantly when the tolerance radius r is larger than 0.07. The *FNE* decreases steadily to 0.08, i.e., approximately 92% of POIs can be detected by our method even with a small tolerance radius $r = 0.07$.

C. Comparison with other methods

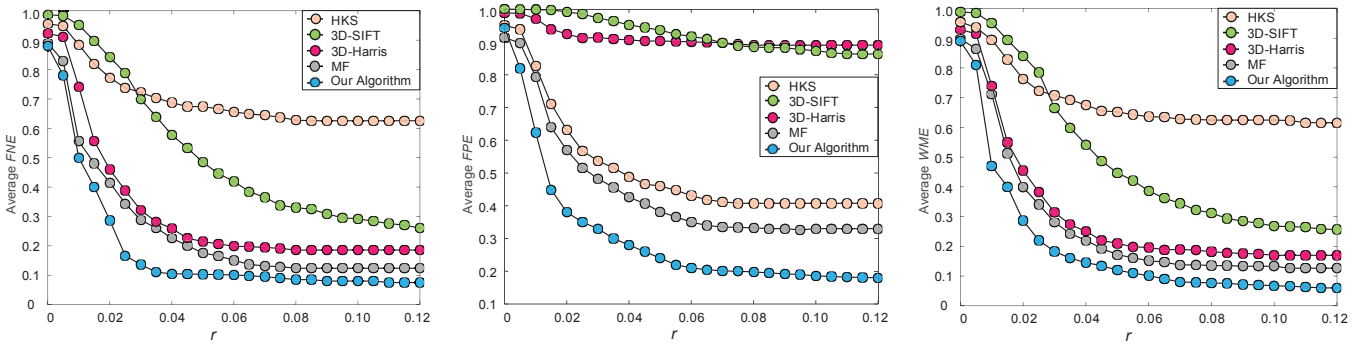
To further evaluate the POI detection performance of our method, we compare it with other methods, including 3D-

TABLE II
THE *AUC* VALUES OF DIFFERENT DETECTION ALGORITHMS ON THE SHREC 2007 NON-RIGID 3D SHAPES DATASET. WE COMPARE OUR METHOD WITH MULTIPLE FEATURES (MF) POIS [11], CLUSTER-BASED POINT SET SALIENCY (CS) [27], SALIENCY OF LARGE POINT SETS (LS) [39], MESH SALIENCY VIA SPECTRAL PROCESSING (MS) [40], AND PCA-BASED SALIENCY (PS) [16]. A HIGHER SCORE REPRESENTS BETTER PERFORMANCE.

Algorithms	Ours	MF	CS	LS	MS	PS
Human	0.6531	**0.6754**	0.5745	0.5893	0.5695	0.5921
Cup	**0.6527**	0.6459	0.6122	0.6192	0.5934	0.6135
Glass	**0.6507**	0.6351	0.5225	0.5727	0.5297	0.5530
Airplane	**0.7043**	0.6625	0.6308	0.6705	0.6160	0.6409
Ant	**0.6651**	0.6524	0.6056	0.6349	0.5696	0.5791
Chair	0.7012	**0.7562**	0.5871	0.6566	0.5505	0.5799
Octopus	0.6235	**0.6694**	0.5480	0.6237	0.5342	0.5726
Table	**0.6918**	0.6738	0.6313	0.6651	0.5802	0.6168
Teddy	0.7411	**0.7584**	0.5671	0.5641	0.5530	0.5682
Hand	**0.6565**	0.6341	0.6060	0.6339	0.5763	0.6022
Plier	0.6264	**0.6537**	0.5997	0.6236	0.5615	0.5754
Fish	0.6312	0.6039	0.6651	0.6717	0.6309	**0.6736**
Bird	**0.6695**	0.6637	0.6010	0.6217	0.5627	0.5984
Spring	0.6241	**0.6351**	0.5512	0.5545	0.5523	0.5339
Armadillo	**0.6936**	0.6825	0.6560	0.6656	0.5996	0.6570
Buste	0.6703	**0.6814**	0.6236	0.6260	0.5620	0.6352
Mechanic	0.6851	**0.7359**	0.6964	0.6932	0.5325	0.7065
Bearing	**0.7085**	0.6537	0.6472	0.6387	0.4986	0.6322
Vase	**0.6809**	0.6328	0.6158	0.6217	0.6058	0.6251
Four-legged	**0.6981**	0.6235	0.6024	0.6168	0.6005	0.6112
Average	**0.6714**	0.6665	0.6072	0.6282	0.5689	0.6083

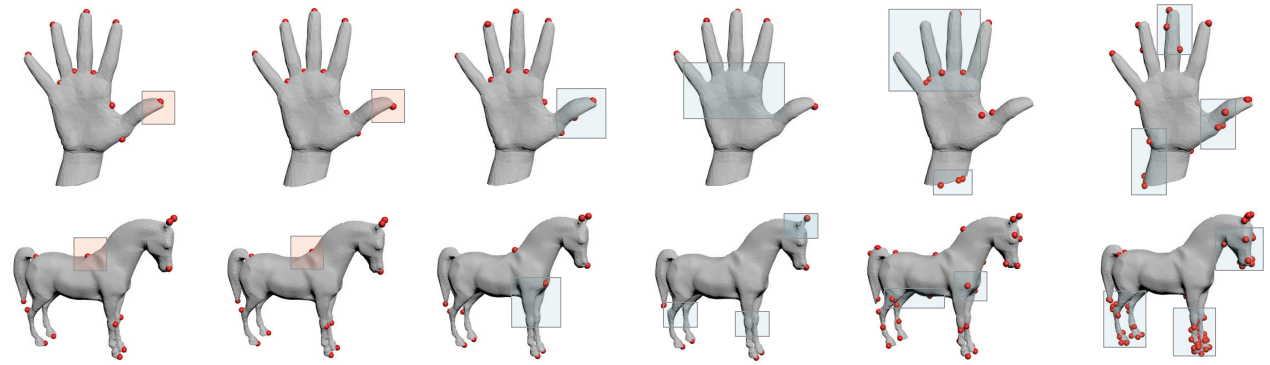
TABLE III
THE *LCC* VALUES OF DIFFERENT DETECTION ALGORITHMS ON THE SHREC 2007 NON-RIGID 3D SHAPES DATASET. WE COMPARE OUR METHOD WITH MULTIPLE FEATURES (MF) POIS [11], CLUSTER-BASED POINT SET SALIENCY (CS) [27], SALIENCY OF LARGE POINT SETS (LS) [39], MESH SALIENCY VIA SPECTRAL PROCESSING (MS) [40], AND PCA-BASED SALIENCY (PS) [16]. A HIGHER SCORE REPRESENTS BETTER PERFORMANCE.

Algorithms	Ours	MF	CS	LS	MS	PS
Human	**0.5506**	0.4503	0.2544	0.1895	0.2580	0.3292
Cup	0.3343	**0.3769**	0.2924	0.3478	0.2444	0.3039
Glass	**0.5543**	0.5020	0.1201	0.4837	0.4120	0.3353
Airplane	0.6116	0.6328	**0.6936**	0.6357	0.4876	0.6601
Ant	0.6972	0.6584	0.7356	**0.7658**	0.3509	0.3357
Chair	0.7076	**0.7326**	0.5152	0.6995	0.2723	0.3652
Octopus	**0.7701**	0.6066	0.2228	0.4654	0.2467	0.3565
Table	0.6774	0.5947	0.5831	**0.7111**	0.2727	0.3076
Teddy	0.4564	**0.5861**	0.1184	0.1246	0.1224	0.1188
Hand	**0.6269**	0.5542	0.5439	0.5372	0.3077	0.3687
Plier	**0.7138**	0.6447	0.5978	0.6591	0.2945	0.3214
Fish	**0.6972**	0.6422	0.6967	0.6349	0.4638	0.5889
Bird	**0.6966**	0.5633	0.5578	0.5690	0.4099	0.5067
Spring	**0.6807**	0.5764	0.4838	0.5227	0.3977	0.1859
Armadillo	**0.7591**	0.5249	0.5495	0.4083	0.2627	0.5589
Buste	**0.4867**	0.4138	0.2540	0.2657	0.1240	0.2755
Mechanic	0.6336	**0.7561**	0.4064	0.4237	0.0548	0.5756
Bearing	0.4286	**0.4836**	0.2676	0.3701	0.0316	0.2933
Vase	**0.5439**	0.4713	0.3673	0.4228	0.3133	0.3285
Four-legged	**0.6206**	0.5062	0.3819	0.3896	0.3682	0.2861
Average	**0.6124**	0.5639	0.4321	0.4813	0.2848	0.3701



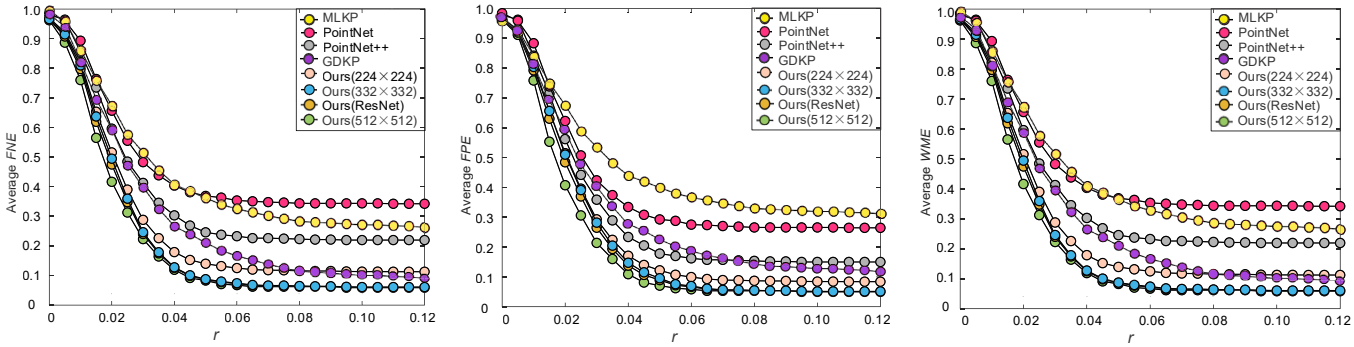
(a) FNE (b) FPE (c) WME

Fig. 11. Comparisons of POI detection results between our method and previous methods on the SHREC 2011 dataset. The evaluation metrics *FNE*, *FPE*, and *WME* are employed to evaluate different methods.



(a) ground truth (b) ours (c) MF (d) HKS (e) 3D-SIFT (f) 3D-Harris

Fig. 12. The comparisons of POI detection results between our method and previous methods on the hand and horse model.



(a) FNE (b) FPE (c) WME

Fig. 13. Comparisons of POI detection results between our method and state-of-the-art methods on the SHREC 2014 dataset.

SIFT [16], 3D-Harris [25], HKS-based POIs [41], and multiple features (MF) POIs [11], on the SHREC 2011 dataset using the evaluation metrics provided by [10].

As shown in Figure 11, the *FNE*, *FPE*, and *WME* values of all algorithms are plotted. We can observe the following facts from the figure. First, the curves of our method are lower than others. This means that while the localization error tolerance radius $r \in [0, 0.12]$ (which is the effective tolerance radius), our method performs better than the other four methods with smaller false negative and false positive errors. Second, on the left side of the graphs where the localization error tolerance radius $r \in [0, 0.2]$, our method exhibits a steep descent in all three charts. This indicates that there are smaller distances between our POIs and ground truth POIs. Our method achieves low error rates in a small tolerance radius r. Therefore, the positions of our POIs are more precise. Third, according to the definitions, *FNE* denotes the rate of undetected POIs and *FPE* denotes the rate of POIs by false detection. Detecting more POIs may lead to higher *FPE*, while detecting insufficient POIs may result in higher *FNE*. Therefore, it is not easy to obtain low *WME*, *FNE*, and *FPE* values simultaneously. In Figure 11, one can see that 3D-SIFT and 3D-Harris have lower *FNE* but higher *FPE*. HKS has lower *FPE* and higher *FNE*. However, our method has lower *WME*, *FPE*, and *FNE* values than other methods,

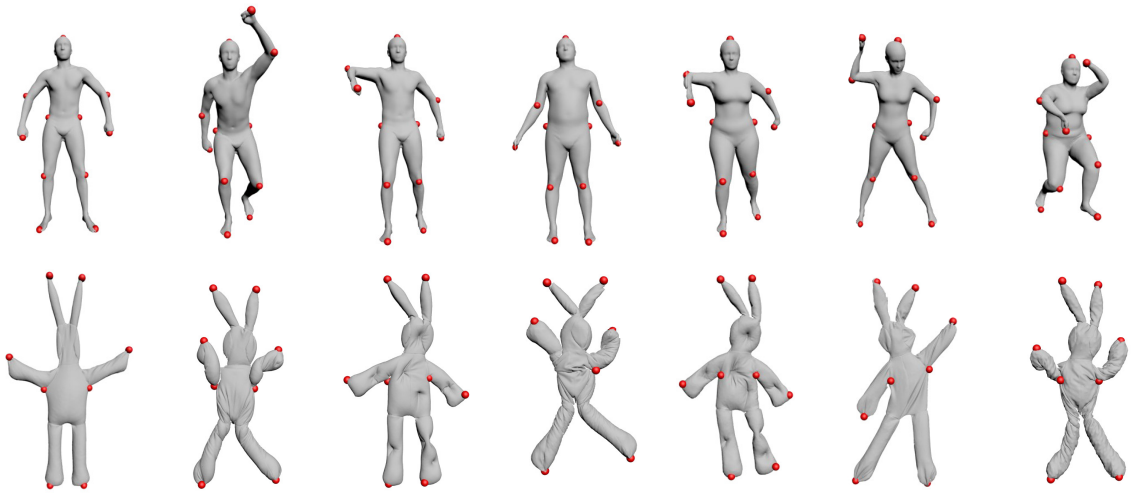


Fig. 14. Some representative POI detection results of our method on the SHREC 2014 and 2020 datasets.

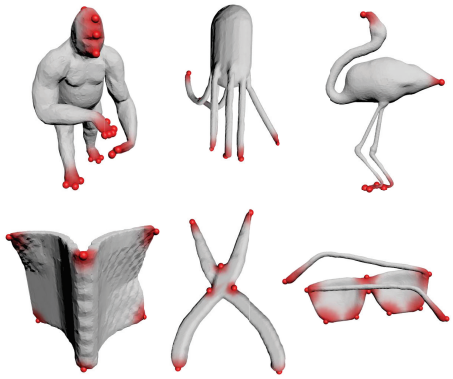


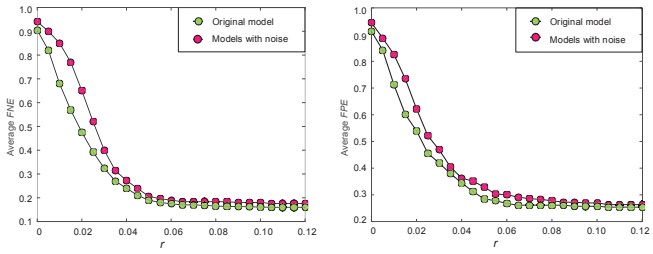
Fig. 15. The saliency distribution in an intermediate step of our algorithm.

which indicates the best performance our method achieves.

Figure 12 shows a visual comparison of representative POI detection results among ours and the other methods mentioned above. We can see that the POI detection results of learning-based methods (MF and ours) are obviously closer to human-marked POIs. More importantly, these two methods have fewer false detected POIs and duplicated detected POIs. In addition, the positions of our detected POIs are more accurate than those of MF. This result is consistent with the conclusions in Figure 11.

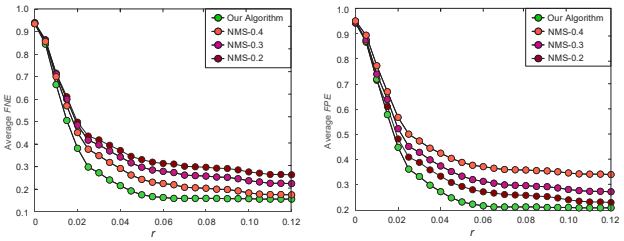
The SHREC 2011 dataset consists of synthetic 3D shapes. To evaluate the feasibility of detecting 3D POIs of our method on actual data, we test our method on the SHREC 2014 dataset and the SHREC 2020 dataset, which are both scanned from the real world. For the SHREC 2014 dataset, 100 shapes are split into the training set, and the remaining 300 shapes are split into the test set. For the SHREC 2020 dataset, the full scan is used as the training set, and the 11 partial scans are used as the test set. On the SHREC 2014 dataset, we compare the performances of our proposed method with other methods, including MLKP [17], PointNet [42], PointNet++ [43], and GDKP [19]. As shown in Figure 13, our method achieves the lowest error curves of WME, FPE, and FNE. More performance tests about the projective network and the resolution of 2D views are also shown in Figure 13. The visualized POI detection results of our method can be found in Figure 14. Akilan *et al.*[44] introduced the residual blocks in the encoder-decoder network. To verify whether introducing residual blocks to our projective neural network can increase the performance, we add the residual blocks to our projective neural network by replacing the original convolutional layer sequence and test the corresponding performance. In addition, we test the performance of our method with higher-resolution images as input. All the experimental results are shown in Figure 13. It can be found that the residual blocks are not significantly helpful for improving the performance of our method while using higher-resolution 2D views does make our method perform slightly better. However, higher resolution also means more computational resource consumption.

We also evaluate our algorithm by comparison with mesh saliency methods using the metrics AUC and LCC. This is meaningful because mesh saliency aims at detecting important regions according to the visual perception of 3D shapes, while POI detection methods attempt to extract the points of interest. Thus, we saved the results of the intermediate step in our algorithm (see Figure 15) and compared them against five other state-of-the-art methods on the SHREC 2007 dataset. These methods include multiple features (MF) POIs [11], cluster-based point set saliency (CS) [27], saliency of large point sets (LS) [39], mesh saliency via spectral processing (MS) [40], and PCA-based saliency (PS) [16]. The comparison results are shown in Table II and Table III. AUC evaluates how many saliency regions of the ground truth are detected by the algorithm. MF and our method achieve better performance on AUC. LCC evaluates whether the saliency regions of the ground truth and the algorithm have peaks at the same place. We find that our method achieves the best performance on LCC, which means that the peaks of saliency regions detected by our method are closest to the ground truth when compared to those obtained from other methods.



(a) FNE (b) FPE

Fig. 16. The *FNE* and *FPE* curves of our method executed on the original models and the models with noise.



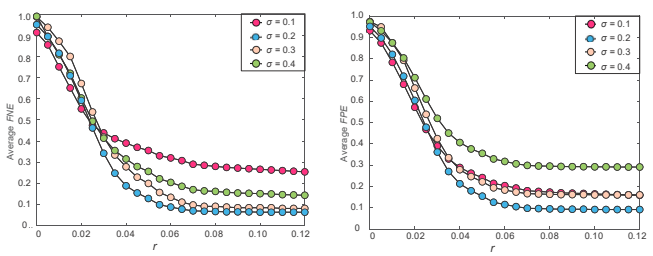
(a) FNE (b) FPE

Fig. 17. Comparisons of POI detection results between applying our improved density peak clustering algorithm and using the naive NMS method with different distance thresholds (0.2, 0.3, and 0.4).

D. Robustness and ablation study

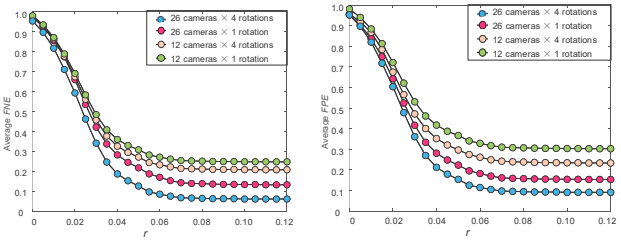
To further assess the robustness of our algorithm, we test it on five categories of 3D shapes containing noise, including Alien, Ant, Armadillo, Hand, and Man. For each shape, we add white Gaussian noise to the coordinates of each vertex on the surface. We test our algorithm on original shapes and noisy shapes. The performance of our method is measured on the two kinds of shapes by using *FNE* and *FPE*. Figure 16 shows the corresponding results. We can see that the performance measured on the original 3D shapes and the noisy shapes is very close. Especially when the tolerance radius r is greater than 0.04, the two error curves appear to almost overlap. This indicates that the noise added to the shapes only has little impact on the performance of our algorithm. We can also see that with the low tolerance radius $r \in [0, 0.04]$, both the *FNE* and *FPE* error curves of noisy shapes are higher than those of original shapes. The results show that the rugged surfaces make the POIs detected by our algorithm deviate from the best positions slightly. In general, our algorithm is robust to noise because projecting 3D shapes into multiple 2D images reduces the impact of noise added to shapes.

To show the effectiveness of our improved density peak clustering for improving the total detection performance, we compare the results from our method and those from using the naive non-maximum suppression (NMS) algorithm [45]. In our experiments, our improved density peak clustering algorithm and the naive NMS algorithm are applied to the probability distributions predicted by the projective neural network. The threshold of the distance is set to 0.2, 0.3, and 0.4 for the NMS algorithm. The results are shown in Figure 17. We can see that our algorithm achieves the lowest error rates for both *FNE* and *FPE*.



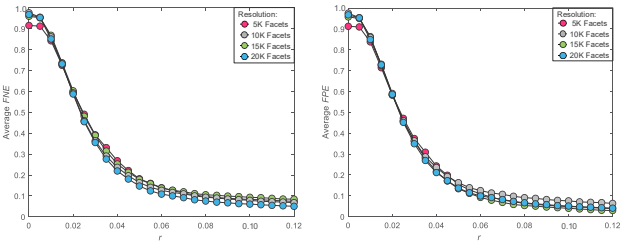
(a) FNE (b) FPE

Fig. 18. The comparisons among POI detection results of our method with different values of σ.



(a) FNE (b) FPE

Fig. 19. The comparisons among POI detection results of our method using different numbers of 2D views as the input of the projective convolutional neural network.



(a) FNE (b) FPE

Fig. 20. The comparisons among POI detection results of our method under different resolutions of 3D shapes.

The parameters σ in Equation (1) determine the probability distributions of POIs and may affect the performance of our method. Smaller values of σ will make the distribution more concentrated. Figure 18 shows the POI detection performances under the different values of σ, where $\sigma = 0.2$ denotes σ is set to one-fifth of the maximum geodesic distance, 0.1 means one-tenth, and so on. The results show that setting $\sigma = 0.2$ can achieve lower error rates. We also compare the performance of our method using different numbers of 2D views as input. As shown in Figure 19, using more 2D views can make the observation of 3D shapes more comprehensive and result in better performance. Therefore, when the POI detection results are not satisfactory, we recommend using more 2D views of different perspectives.

We further test the performances of our method for 3D shapes with different resolutions. For each shape in the SHREC 2014 dataset, we first simplify it from containing 20K facets to containing 15K, 10K, and 5K facets, respectively, using the classic QEM algorithm [46]. Then the performances are measured with FNE and FPE for 3D shapes with different resolutions. The experimental results are shown in Figure 20. From the Figure, we can see that our method achieves very similar performance on the shapes of 5K, 10K, 15K, and 20K facets and is robust to different resolutions of 3D shapes.

TABLE IV
THE AVERAGE RUNNING TIME ON EACH CATEGORY OF 3D SHAPE FOR OUR METHOD, WHERE EACH CATEGORY CONTAINS 20 SHAPES AND EACH SHAPE CONSISTS OF 12∼25K VERTICES.

Steps	Preparation of training data (15 shapes)	Training of the neural network	POI extraction (every shape)
Time (minutes)	10∼15	120	0.3∼0.5

E. Performance

We implement our algorithm in Matlab and C++. The performance of our approach is measured on a PC with a 3.20 GHz CPU, 32 GB RAM, and NVIDIA GeForce GTX 1080 Ti GPU. The measured performance is shown in Table IV. On average, it takes less than 1 minute to transform a 3D shape into 2D images, 2 hours to train the neural network, and less than 0.5 minutes to predict and extract the POIs. The inference process of one image takes approximately 50 milliseconds. The computational time of our method is mainly spent in the training process, which takes over 80% of the total time.

V. LIMITATION AND FUTURE WORK

First, the training and testing shapes need to be classified in advance. We train the convolutional neural network for each category of shapes and input the test shapes to the neural network. Second, the training of convolutional neural networks is very time consuming. Finding a more efficient structure of neural networks may reduce the training time further, which is one of our future plans. Third, our method relies on geodesic distances to measure the distances between any two vertices on the manifold. However, geodesic distance computation on 3D meshes with high resolutions may consume considerable computational time. In this case, 3D mesh simplification algorithms may be applied to the input 3D meshes in advance to reduce their resolutions and the corresponding computational burden.

VI. CONCLUSION

In this paper, we propose a novel learning-based method for POI detection of 3D shapes. Instead of computing POI geometrical features on the meshes, we detect POIs using multiple projections of 3D shapes, including shaded and depth images. Our method is well-aligned with human visual perception. The key idea is to predict a saliency map to encode the probability of being a POI and then extract typical POIs by clustering. Our method is data-driven and can predict POIs in a similar way as humans. Extensive experimental results show that our method outperforms state-of-the-art approaches.

ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China (61872321, 62025207, 61772016, 61572022, 62036010), National Science Foundation (IIS-1617172, IIS-1622360, IIS-1764071), Open Research Projects of Zhejiang Lab (2019NB0AB03), the Ningbo Major Special Projects of the "Science and Technology Innovation 2025" (2020Z005, 2020Z007), and Ningbo Innovative Team (2016C11024).

REFERENCES

[1] M. Feixas, M. Sbert, and F. Gonz A Lez, "A unified information-theoretic framework for viewpoint selection and mesh saliency," *ACM Transactions on Applied Perception*, vol. 6, no. 1, pp. 1–23, 2009.
[2] Y. Kim and A. Varshney, "Saliency-guided enhancement for volume visualization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 5, pp. 925–932, 2006.
[3] J. Xie, G. Dai, and Y. Fang, "Deep multimetric learning for shape-based 3D model retrieval," *IEEE Transactions on Multimedia*, vol. 19, no. 11, pp. 2463–2474, 2017.
[4] J.-Y. Chen, C.-H. Lin, P.-C. Hsu, and C.-H. Chen, "Point cloud encoding for 3D building model retrieval," *IEEE Transactions on Multimedia*, vol. 16, no. 2, pp. 337–345, 2013.
[5] S. Bu, Z. Liu, J. Han, J. Wu, and R. Ji, "Learning high-level feature by deep belief networks for 3-D model retrieval and recognition," *IEEE Transactions on Multimedia*, vol. 16, no. 8, pp. 2154–2167, 2014.
[6] G. K. Tam, Z.-Q. Cheng, Y.-K. Lai, F. C. Langbein, Y. Liu, D. Marshall, R. R. Martin, X.-F. Sun, and P. L. Rosin, "Registration of 3D point clouds and meshes: A survey from rigid to nonrigid," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 7, pp. 1199–1217, 2013.
[7] Q. Zhen, D. Huang, Y. Wang, and L. Chen, "Muscular movement model-based automatic 3D/4D facial expression recognition," *IEEE Transactions on Multimedia*, vol. 18, no. 7, pp. 1438–1450, 2016.
[8] S. Katz, G. Leifman, and A. Tal, "Mesh segmentation using feature point and core extraction," *The Visual Computer*, vol. 21, no. 8, pp. 649–658, 2005.
[9] X. Chen, A. Saparov, B. Pang, and T. Funkhouser, "Schelling points on 3D surface meshes," *ACM Transactions on Graphics*, vol. 31, no. 4, p. 29, 2012.
[10] H. Dutagaci, C. P. Cheung, and A. Godil, "Evaluation of 3D interest point detection techniques via human-generated ground truth," *The Visual Computer*, vol. 28, no. 9, pp. 901–917, 2012.
[11] Z. Shu, S. Xin, X. Xu, L. Liu, and L. Kavan, "Detecting 3D points of interest using multiple features and stacked auto-encoder," *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 8, pp. 2583–2596, 2019.
[12] Z. Lian, A. Godil, B. Bustos, M. Daoudi, J. Hermans, S. Kawamura, Y. Kurita, G. Lavoué, H. Nguyen, and R. Ohbuchi, "SHREC'11 track: shape retrieval on non-rigid 3D watertight meshes," *Eurographics Workshop on 3D Object Retrieval*, vol. 11, pp. 79–88, 2011.
[13] N. Gelfand, N. J. Mitra, L. J. Guibas, and H. Pottmann, "Robust global registration," in *Proceedings of the Third Eurographics Symposium on Geometry Processing*. Aire-la-Ville, Switzerland: Eurographics Association, 2005.
[14] U. Castellani, M. Cristani, S. Fantoni, and V. Murino, "Sparse points matching by combining 3D mesh saliency with statistical descriptors," *Computer Graphics Forum*, vol. 27, no. 2, pp. 643–652, 2008.
[15] G. Zou, J. Hua, M. Dong, and H. Qin, "Surface matching with salient keypoints in geodesic scale space," *Computer Animation and Virtual Worlds*, vol. 19, no. 3-4, pp. 399–410, 2008.
[16] A. Godil and A. I. Wagan, "Salient local 3D features for 3D shape retrieval," in *SPIE Electronic Imaging*. International Society for Optics and Photonics, 2011, pp. 1–8.
[17] C. Creusot, N. Pears, and J. Austin, "A machine-learning approach to keypoint detection and landmarking on 3D meshes," *International journal of computer vision*, vol. 102, no. 1-3, pp. 146–179, 2013.
[18] L. Teran and P. Mordohai, "3D interest point detection via discriminative learning," in *European Conference on Computer Vision*. Springer, 2014, pp. 159–173.
[19] Y. You, Y. Lou, C. Li, Z. Cheng, L. Li, L. Ma, C. Lu, and W. Wang, "Keypointnet: A large-scale 3D keypoint dataset aggregated from numerous human annotations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 13 647–13 656.

[20] X. Liu, L. Liu, W. Song, Y. Liu, and L. Ma, "Shape context based mesh saliency detection and its applications: A survey," *Computers & Graphics*, vol. 57, pp. 12–30, 2016.
[21] C. Koch and S. Ullman, "Shifts in selective visual attention: Towards the underlying neural circuitry," in *Matters of intelligence*. Springer, 1987, pp. 115–141.
[22] R. Gal and D. Cohen-Or, "Salient geometric features for partial shape matching and similarity," *ACM Transactions on Graphics*, vol. 25, no. 1, pp. 130–150, 2006.
[23] S. Jia, C. Zhang, X. Li, and Y. Zhou, "Mesh resizing based on hierarchical saliency detection," *Graphical Models*, vol. 76, no. 5, pp. 355–362, 2014.
[24] C. G. Harris, M. Stephens *et al.*, "A combined corner and edge detector," in *Alvey Vision Conference*, vol. 15, no. 50, 1988, pp. 10–5244.
[25] I. Pratikakis, M. Spagnuolo, T. Theoharis, and R. Veltkamp, "A robust 3D interest points detector based on Harris operator," in *Eurographics Workshop on 3D Object Retrieval*, vol. 5, 2010.
[26] I. Sipiran and B. Bustos, "Harris 3D: a robust extension of the harris operator for interest point detection on 3D meshes," *The Visual Computer*, vol. 27, no. 11, p. 963, 2011.
[27] F. P. Tasse, J. Kosinka, and N. Dodgson, "Cluster-based point set saliency," in *IEEE International Conference on Computer Vision*, 2015, pp. 163–171.
[28] R. Song, Y. Liu, Y. Zhao, R. R. Martin, and P. L. Rosin, "Conditional random field-based mesh saliency," in *2012 19th IEEE International Conference on Image Processing*. IEEE, 2012, pp. 637–640.
[29] I. Sipiran and B. Bustos, "Key-components: Detection of salient regions on 3D meshes," *The Visual Computer*, vol. 29, no. 12, pp. 1319–1332, 2013.
[30] S.-W. Jeong and J.-Y. Sim, "Saliency detection for 3D surface geometry using semi-regular meshes," *IEEE Transactions on Multimedia*, vol. 19, no. 12, pp. 2692–2705, 2017.
[31] H. Su, S. Maji, E. Kalogerakis, and 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.
[32] Z.-X. Yang, L. Tang, K. Zhang, and P. K. Wong, "Multi-view CNN feature aggregation with ELM auto-encoder for 3D shape recognition," *Cognitive Computation*, vol. 10, no. 6, pp. 908–921, 2018.
[33] E. Kalogerakis, M. Averkiou, S. Maji, and S. Chaudhuri, "3D shape segmentation with projective convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3779–3788.
[34] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, no. 6191, pp. 1492–1496, 2014.
[35] H. Dutagaci, A. Godil, P. Daras, A. Axenopoulos, G. Litos, S. Manolopoulou, K. Goto, T. Yanagimachi, Y. Kurita, S. Kawamura *et al.*, "SHREC'11 track: Generic shape retrieval," in *Proceedings of the 4th Eurographics conference on 3D Object Retrieval*. Eurographics Association, 2011, pp. 65–69.
[36] G. Lavoué, J.-P. Vandeborre, H. Benhabiles, M. Daoudi, K. Huebner, M. Mortara, and M. Spagnuolo, "SHREC'12 Track: 3D Mesh Segmentation," in *Eurographics Workshop on 3D Object Retrieval*. The Eurographics Association, 2012.
[37] F. P. Tasse, J. Kosinka, and N. A. Dodgson, "Quantitative analysis of saliency models," in *SIGGRAPH ASIA 2016 Technical Briefs*. New York, USA: ACM, 2016, pp. 19:1–19:4.
[38] G. J. Brostow, J. Fauqueur, and R. Cipolla, "Semantic object classes in video: A high-definition ground truth database," *Pattern Recognition Letters*, vol. 30, no. 2, pp. 88–97, 2009.
[39] E. Shtrom, G. Leifman, and A. Tal, "Saliency detection in large point sets," in *IEEE International Conference on Computer Vision*, 2013, pp. 3591–3598.
[40] R. Song, Y. Liu, R. R. Martin, and P. L. Rosin, "Mesh saliency via spectral processing," *ACM Transactions on Graphics*, vol. 33, no. 1, pp. 1–17, 2014.
[41] J. Sun, M. Ovsjanikov, and L. Guibas, "A concise and provably informative multi-scale signature based on heat diffusion," *Computer Graphics Forum*, vol. 28, no. 5, pp. 1383–1392, 2009.
[42] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 652–660.
[43] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *Advances in Neural Information Processing Systems*, 2017, pp. 5099–5108.
[44] T. Akilan, Q. J. Wu, and W. Zhang, "Video foreground extraction using multi-view receptive field and encoder–decoder DCNN for traffic and surveillance applications," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 10, pp. 9478–9493, 2019.
[45] A. Neubeck and L. Van Gool, "Efficient non-maximum suppression," in *International Conference on Pattern Recognition*, vol. 3. IEEE, 2006, pp. 850–855.
[46] M. Garland and P. S. Heckbert, "Surface simplification using quadric error metrics," in *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, 1997, pp. 209–216.

Zhenyu Shu earned his PhD degree in 2010 at Zhejiang University, China. He is now working as a professor at NingboTech University. His research interests include computer graphics, digital geometry processing, and machine learning. He has published over 30 papers in international conferences or journals.

Sipeng Yang is currently a postgraduate in the School of Mechanical Engineering at Zhejiang University. His research interests include computer graphics and machine learning.

Shiqing Xin is an associate professor at the School of Computer Science and Technology at Shandong University. He received his PhD degree in applied mathematics at Zhejiang University in 2009. His research interests include computer graphics, computational geometry, and 3D printing.

Chaoyi Pang is a senior member of ACM. He earned his PhD degree from the University of Melbourne (1999), advised by Prof. Gouzhu Dong and Prof. Rao Kotagiri. After receiving a PhD degree, he worked in the IT industry as an IT engineer and consultant in 1999-2002 and CSIRO as a research scientist in 2002-2014 in Australia. Currently, he is the Dean of the School of Computer and Data Engineering, Zhejiang University (NIT). He is a distinguished professor at Zhejiang University (NIT), Hebei Academy of Sciences, and Hebei University of Economics and Business. His research interests lie in algorithms, stream data compression and processing, data warehousing, data integration, database theory, graph theory and e-health.

Xiaogang Jin received a BSc degree, an MSc degree, and a PhD degree from Zhejiang University, P.R. China, in 1989, 1992, and 1995, respectively. He is a professor in the State Key Laboratory of CAD&CG, Zhejiang University. His current research interests include traffic simulation, collective behavior simulation, cloth animation, virtual try-on, digital face, implicit surface modeling and applications, creative modeling, computer-generated marbling, sketch-based modeling, and virtual reality. He received an ACM Recognition of Service Award in 2015 and the Best Paper Awards from CASA 2017 and CASA 2018. He is a member of the IEEE and the ACM.

Ladislav Kavan received his Mgr. degree from Charles University in Prague in 2003 and his PhD degree in 2007 from Czech Technical University in Prague. He is currently an associate professor at the University of Utah, USA. His research interests include computer animation, physics-based simulation, and geometry processing.

Ligang Liu received a BSc degree in 1996 and a PhD degree in 2001 from Zhejiang University, China. He is a professor at the University of Science and Technology of China. Between 2001 and 2004, he was at Microsoft Research Asia. Then, he was at Zhejiang University during 2004 and 2012. He paid an academic visit to Harvard University during 2009 and 2011. His research interests include geometric processing and image processing.