This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2018.2848628, IEEE Transactions on Visualization and Computer Graphics

Detecting 3D Points of Interest Using Multiple Features and Stacked Auto-encoder

Zhenyu Shu, Shiqing Xin*, Xin Xu, Ligang Liu, and Ladislav Kavan

Abstract—Considering the fact that points of interest on 3D shapes can be discriminated from a geometric perspective, it is reasonable to map the geometric signature of a point p to a probability value encoding to what degree p is a point of interest, especially for a specific class of 3D shapes. Based on the observation, we propose a three-phase algorithm for learning and predicting points of interest on 3D shapes by using multiple feature descriptors. Our algorithm requires two separate deep neural networks (stacked auto-encoders) to accomplish the task. During the first phase, we predict the membership of the given 3D shape according to a set of geometric descriptors using a deep neural network. After that, we train the other deep neural network to predict a probability distribution defined on the surface representing the possibility of a point being a point of interest. Finally, we use a manifold clustering technique to extract a set of points of interest as the output. Experimental results show superior detection performance of the proposed method over the previous state-of-the-art approaches.

Index Terms—3D shapes, Point of interest, Multiple features, Stacked auto-encoder

1 Introduction

3D points of interest (POIs), also referred to as feature points or salient points, are distinctive points in visual perception. POIs are found to be very useful in geometry processing tasks, such as mesh segmentation [1], mesh registration [2], shape enhancement [3], shape retrieval [4], viewpoint selection [5] and visual attention guidance [6].

There is a common understanding that POIs can be discriminated from a geometric perspective but the real relationship between POIs and geometric descriptors is quite complicated [7], [8]. A number of research works [9], [10] suggest extracting a feature vector to encode the local geometry for any vertex of the input shape and then selecting a subset of representative vertices as the POIs. However, the resulting POIs are still conspicuously different from manually marked salient points, which reveals that automatically detecting POIs in coincidence with human visual perception still remains a challenging problem.

In spite of the existing challenge, it is reasonable to map the geometric signature of a point p to a probability value encoding to what degree p is a POI. However, judging whether a point is a POI or not is actually a subjective task because different people may have different understanding about POIs. Inspired by recent advances in deep learning [11], [12], we propose a novel algorithm for learning and predicting POIs on 3D shapes by using multiple feature descriptors. Our algorithm requires two separate deep regression neural networks to accomplish the task. One network is trained on the whole dataset to identify the membership of a given 3D shape, and the other network is trained on each category respectively to predict a saliency map according to the extracted geometric signature once the membership is known. Finally, we use a manifold clustering technique to extract a set of POIs as the output. Our deep learning-based method can automatically learn and detect the results in coincidence with humans' expectation as long as the necessary training data is provided. Figure 1 shows an example of POIs detected by our approach.

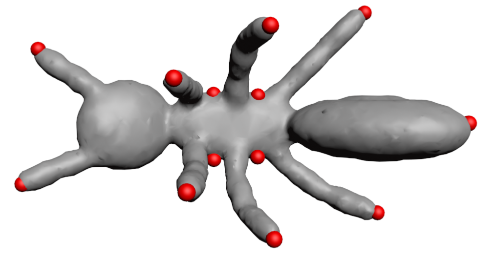


Fig. 1. POIs detected by our approach.

We evaluate our approach on SHREC 2011 non-rigid 3D shape dataset [13], which contains 600 different 3D shapes. The ground truth data on these models are obtained from manually marked data by volunteers. Numerous experimental results show that our approach outperforms the state of the art in terms of various measures, such as False Negative Error (FNE), Weighted Miss Error (WME), False Positive Error (FPE), Area Under the ROC Curve (AUC), Normalized Scanpath Saliency (NSS) and Linear Correlation Coefficient (LCC).

Our contributions are three-fold.

- *Zhenyu Shu is with School of Computer and Data Engineering, Ningbo Institute of Technology, Zhejiang University, Ningbo, PR China. E-mail: shuzhenyu@nit.zju.edu.cn*
- *Shiqing Xin is with School of Computer Science and Technology, ShanDong University, Jinan, PR China. Corresponding author. E-mail: xinshiqing@163.com*
- *Xin Xu is with School of Information Science and Engineering, Ningbo University, Ningbo, PR China.*
- *Ligang Liu is with Graphics & Geometric Computing Laboratory, School of Mathematical Sciences, University of Science and Technology of China, Anhui, PR China.*
- *Ladislav Kavan is with School of Computing, University of Utah, Salt Lake City, USA.*

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

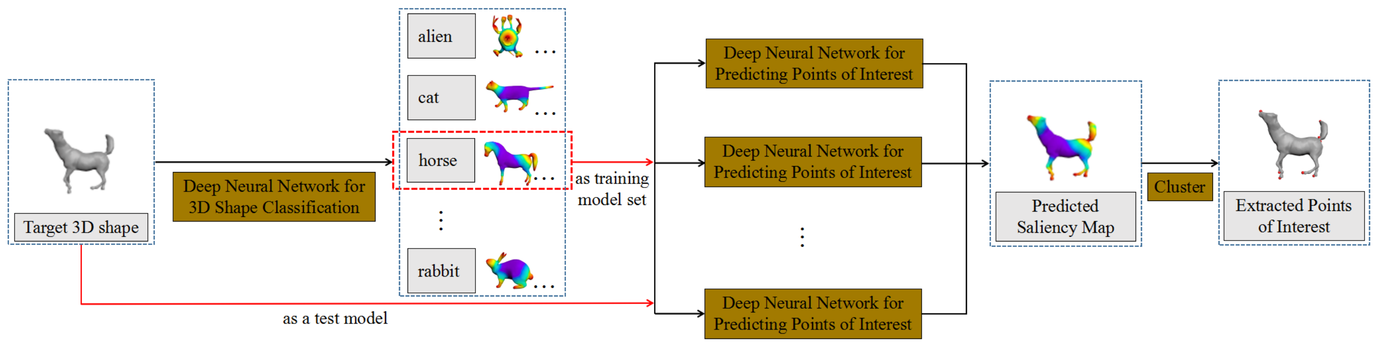

alien
cat
horse
rabbit
Target 3D shape
Deep Neural Network for 3D Shape Classification
as training model set
as a test model
Deep Neural Network for Predicting Points of Interest
Deep Neural Network for Predicting Points of Interest
Deep Neural Network for Predicting Points of Interest
Predicted Saliency Map
Cluster
Extracted Points of Interest

Fig. 2. The overall workflow of the proposed approach for detecting 3D POIs.

- We use deep neural networks to predict a saliency map for a given 3D shape, which is rather different from conventional classification problem.
- We integrate various geometric descriptors into our algorithmic framework. It may achieve better detection results if some new descriptors are considered in this framework.
- Our method is data-driven and users are able to obtain reliable salient points as long as there are sufficiently many labeled ground truth samples.

The remainder of the paper is organized as follows. Section 2 reviews the related work on POI detection and deep learning-based 3D shape analysis. Section 3 presents the overall detection framework followed by detailed configuration of the deep neural networks. The training/predicting techniques are detailed in Section 4. After that, we give the 3D POI detection algorithm in Section 5. In Section 6, we show extensive experimental results, as well as comparison statistics. Finally, we point out limitations and future work in Section 7 and conclude this paper in Section 8.

2 Related Work

At least three topics are highly related to the theme of this paper: detection of POIs, extraction of geometric feature descriptors and deep learning based 3D shape analysis.

2.1 POI detection

POIs or salient points, originated from the area of computer vision, are widely studied in the computer graphics community. Detecting POI is useful on many occasions, for example, performing a shape-based search across distinctive regions [14] or selecting the most informative views of a given 3D model [15].

In the early stages, researchers found 3D POIs from multiple 2D projected views, such as [16], [17], [18], [19]. Since about 10 years ago, researchers turned to detect POIs directly on the input polygonal surface, measuring saliency according to the geometric property in a neighborhood. Depending on the neighborhood size, we can further classify existing methods into two kinds.

The first kind of algorithms measures saliency in a local scale. For example, Koch et al. [20] suggested that the salient regions should be distinctive from their immediate surroundings. Lee et al. [21] defined scale-dependent saliency using a center-surround operator on Gaussian-weighted mean curvatures. Gal et al. [4] constructed salient geometric features to represent the geometry of local regions of the surface by combining low-level features into a high-level one. In fact, spectral analysis techniques can also be used for this purpose. The key idea [22], [23] is to transform the spectral residual in the spectral domain back to the spatial domain.

The other kind of methods [24], [25] measures saliency in a rather different manner. They often need to evaluate global contrast differences and spatial coherence. The central idea is to set up one kind of measurement that makes the eye-catching regions stand out from a global scope. For example, Perazzi et al. [26] conducted contrast-based saliency estimation using high-dimensional Gaussian filters. In addition, priors or more visual cues on foreground or background have been shown to be helpful to saliency detection from recent research results [27], [28].

Technically speaking, detecting POIs is to automatically identify salient information that coincides with human perception. In fact, POIs are related to geometric features but it is hard to find an explicit formula to characterize the relationship between them. Motivated by this observation, in this paper, we propose a novel method based on deep learning techniques. On one hand, this new framework is able to learn from manually labeled data and predict salient points like humans do. On the other hand, it enables us to extract POIs by considering multiple geometric features, rather than only a single feature. New geometric features can be easily integrated into this framework. Finally, it is worthwhile to point out that detecting POIs is a subjective problem, i.e., different people may have different understandings. Our algorithm is data-driven and can guarantee the prediction results to match the training data to the fullest extent.

In [7], the authors also propose a data-driven method to effectively detect Schelling points on 3D mesh models. However, they employ random forests, which is a different strategy from our method, to predict the distribution of Schelling points. The comparison between their method and ours can be found in Section 6.3.

Recently, another method of detecting tactile mesh saliency, where a human is more likely to grasp, press, or touch on a 3D mesh model is proposed in [29]. By mapping a 3D model to multiple depth images, they propose a novel multi-view deep ranking method, which builds a regression between depth images and tactile saliency, to predict the regions where human tend to touch. It is worth to point out that not only the goal of their method, but also the regression constructed is very different from our method where a regression between geometric feature vectors and the probabilities of being POIs for all vertices is constructed. We do not present the comparison of these two methods in this paper due to the very different detection goals.

2.2 Feature descriptors

Feature descriptors are central to POI detection problem. Usually, a feature descriptor is to encode local or global geometric features for a given 3D shape. It can be typically represented by a scalar or vector field on a polygonal mesh, and then converted into a histogram before training the neural network. In recent years, numerous local feature descriptors have been proposed, such as Gaussian curvature (GC), shape diameter function (SDF) ([30]) and average geodesic distance (AGD) ([31]). Bronstein et al. [32] developed a scale-invariant heat kernel descriptor (SIHKS). The construction is based on a logarithmically sampled scale-space in which shape scaling corresponds to a translation. Aubry et al. [33] proposed a shape signature (WKS) which represents the average probability of measuring a quantum mechanical particle at a specific location and very suitable for non-rigid 3D shapes. Knopp et al. [34] presented a local 3D shape descriptor by using Hough-voting. Smeets et al. [35] proposed a four-step algorithm to generate a local shape descriptor for face recognition under expression variations and partial data. In this paper, we use 3 descriptors including SIHKS, WKS, and GC to detect POIs.

2.3 Deep learning-based 3D shape analysis

In recent years, a lot of deep learning-based research work [36], [37], [38] has been proposed to solve various problems in 3D shape analysis, such as shape retrieval, shape segmentation and even shape modeling. The design of the deep learning framework is closely related to geometric representations. Depending on this, we can divide the existing approaches into 3 categories. The first category [39], [40], [41] transforms a given 3D shape into a set of 2D views each of which has a structured representation. However, this kind of methods may suffer from information loss. Therefore, different from the first category of methods, the second category of algorithms [42], [43], [44] directly learns features from uniform or adaptive voxelized data of 3D shapes by introducing a 3D convolution operator. In practice, the voxelization resolution cannot be too high due to the limitation of current computing power. To accelerate the training process, the third category of methods [12], [45], [46], [47], [48], [49], [50] aims at learning high-level features on certain discretizations of surfaces (e.g. triangular meshes, point clouds or range scans). Some of them [12], [45], [46] learn high-level features based on conventional hand-crafted geometric features and can usually achieve desirable performance while tolerating a small amount of training data especially when it is very hard to get sufficiently many training samples. In this paper, we build our deep neural network upon multiple conventional hand-crafted features to automatically detect POIs on 3D shapes, which falls into the third category.

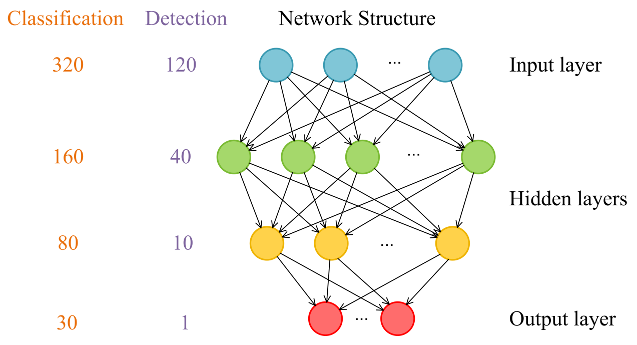


Fig. 3. The architecture of our deep neural network for detecting POIs.

3 Overview

Our algorithm relies on two deep neural networks to robustly detect POIs in a supervised way, as illustrated in Figure 2. One network is to predict the membership of the given 3D model to a specific class, and the other network is to predict the probability of being a POI for any vertex according to its geometric properties. Finally, an extra step is required to extract typical POIs from the resulting saliency map. We achieve this by density peak-based clustering algorithm [51].

It is worth noting that we use deep neural networks to predict a model dependent probability field, rather than a set of labels, which is different from previous shape classification or segmentation problems. For this purpose, we transform manually labeled data to probability fields by constructing a biharmonic distance field. At the same time, we need to extract POIs from probability fields by clustering techniques. This is the key to adapting conventional deep neural networks to the POI detection problem.

4 Background

In this section, we detail the architecture of the two deep neural networks, one used to detect the POIs and the other to predict the class of the target 3D shape. Both of them have four layers in total, concatenated stacked auto-encoders (SAE) [52] and contain a softmax layer, as shown in Figure 3. For the neural network of predicting POIs, the numbers of neurons are experimentally set to 120-40-10-1. For the other neural network, the corresponding setting is 320-160-80-30.

4.1 Auto-encoders

Just as its name implies, SAE is a neural network composed of multiple layers of sparse auto-encoders. As shown in Figure 4, the auto-encoder we used consists of an input layer, a hidden layer, and an output layer. It is able to automatically learn latent features of the input by minimizing the discrepancy between the input features and the reconstruction ones.

In Figure 4, the top mapping represents the stage of the encoder, while the bottom mapping represents that of the decoder. Let N_L and N_G be the numbers of units in

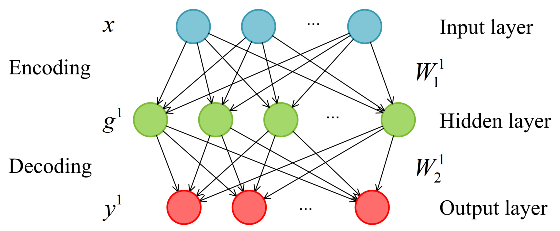


Fig. 4. Illustration of an auto-encoder. Note that the bias parameters b_1 and b_2 are omitted for clarity.

the input layer and the hidden layer respectively. For each training 3D model or each vertex on a training 3D model, different kind of geometric feature descriptors, including GC, SIHKS and wavelet kernel signature (WKS), are used to extract multiple feature vectors (See section 5.2 for details), where each feature vector describes the geometric feature of a 3D model or a vertex from a different aspect. Those multiple feature vectors are then concatenated into one feature vector $\boldsymbol{x} \in R^{N_L \times 1}$. The auto-encoder transforms $\boldsymbol{x}$ to a latent representation $\boldsymbol{g}^1$ by a compound mapping of a linear transformation and a non-linear activation function s as follows:

$$\boldsymbol{g}^1 = s({\boldsymbol{W}_1}^1 \boldsymbol{x} + {\boldsymbol{b}_1}^1), \tag{1}$$

where $\boldsymbol{g}^1 \in R^{N_G}$ is the latent data, ${\boldsymbol{W}_1}^1 \in R^{N_G \times N_L}$ is the encoding weight matrix, ${\boldsymbol{b}_1}^1 \in R^{N_G}$ is the bias vector, and $s(\cdot)$ is the sigmoid function:

$$s(a) = \frac{1}{1+e^{-a}}. \tag{2}$$

Then the latent representation $\boldsymbol{g}^1$ is mapped to a feature vector $\boldsymbol{y}^1 \in R^{N_L}$, which approximately reconstructs the input feature vector $\boldsymbol{x}$ by employing a compound mapping of a linear transformation and a non-linear activation function as follows:

$$\boldsymbol{y}^1 = s(\boldsymbol{W}_2^1 \boldsymbol{g}^1 + \boldsymbol{b}_2^1), \tag{3}$$

where $\boldsymbol{g}^1$ is the latent data, $\boldsymbol{W}_2^1 \in R^{N_L \times N_G}$ is the decoding weight matrix, and $\boldsymbol{b}_2^1 \in R^{N_L}$ is the bias vector.

Given m training feature vectors $\boldsymbol{x}_{(i)}$ (Denoted as $\boldsymbol{X}$), where each one represents the geometric features of a training 3D model or a vertex on a training 3D model, we can learn the underlying features by minimizing the reconstruction error of the cost function:

$$D(\boldsymbol{X}, \boldsymbol{Y}^1) = \frac{1}{2}\sum_{i=1}^{m} \left\| \boldsymbol{x}_{(i)} - \boldsymbol{y}_{(i)}^1 \right\|_2^2, \tag{4}$$

where $\boldsymbol{Y}^1$ denotes all the reconstructed feature vectors, $\boldsymbol{x}_{(i)}$ and $\boldsymbol{y}_{(i)}^1$ represent the ith training feature vector and the ith reconstructed feature vector respectively. Under certain circumstances, some weights may result in over-fitting [53]. Hence, we use a regularization term to avoid over-fitting. And the Equation 4 can be redefined as follows:

$$D(\boldsymbol{X}, \boldsymbol{Y}^1, \boldsymbol{\theta}) = \frac{1}{2}\sum_{i=1}^{m} \left\| \boldsymbol{x}_{(i)} - \boldsymbol{y}_{(i)}^1 \right\|_2^2 + \lambda \|\boldsymbol{\theta}\|_2^2, \tag{5}$$

where λ is the weight decay parameter, $\boldsymbol{\theta} = \{\boldsymbol{W}, \boldsymbol{b}\}$, $\boldsymbol{W}$ and $\boldsymbol{b}$ represent the weights and the biases of the auto-encoder respectively.

To further prevent the auto-encoder from learning some useless information from the training feature vectors, we apply a sparsity constraint on the hidden layer based on the Kullback-Leibler (KL) divergence [54] in this paper. To this end, the optimization problem can be formulated as follows:

$$\arg\min_{\boldsymbol{\theta}} D(\boldsymbol{X}, \boldsymbol{Y}^1, \boldsymbol{\theta}) + \tau \sum_{i=1}^{N_G} KL\left(\rho \| \overset{\wedge}{\rho_i}\right), \tag{6}$$

where,

$$KL\left(\rho \| \overset{\wedge}{\rho_i}\right) = \rho \log \frac{\rho}{\overset{\wedge}{\rho_i}} + (1-\rho)\log\frac{1-\rho}{1-\overset{\wedge}{\rho_i}},$$

τ is the weight of sparsity penalty term, ρ is the sparsity parameter, $\overset{\wedge}{\rho_i} = \frac{1}{m}\sum_{j=1}^{m} (g_i)_j$ is the average activation of the hidden unit g_i (averaged over the all the input training feature vectors), and $(g_i)_j$ represents the corresponding activation of the hidden unit g_i of the jth input training feature vector. From problem (6), we can see that the sparsity penalty term will vanish if $\overset{\wedge}{\rho_i} = \rho$. So the closer $\overset{\wedge}{\rho_i}$ and ρ are, the sparser the hidden layer will be.

With this formulation, we use the back propagation algorithm [55] and the gradient descent approach to train the auto-encoder by optimizing the cost function with respect to $\boldsymbol{\theta}$.

4.2 Stacked auto-encoders

To further learn high-level features, multiple layers of sparse auto-encoders (SAE) are stacked up, where the latent feature vectors learned in the previous auto-encoder are used as the input of the next auto-encoder. The whole training process is carried out in a greedy layer-wise way [56] and better parameters are generated for deep neural networks and desirable results are produced [57]. After training each auto-encoder using the method described in Section 4.1, the outputs $\boldsymbol{W}_2^i$ and $\boldsymbol{y}^i$ of the auto-encoder are discarded and the latent features $\boldsymbol{g}^i$ of the auto-encoder are used to feed the next auto-encoder.

To predict the membership of a 3D model, we add an extra softmax layer to the end of SAE for the neural network of 3D model classification. The predicted membership for each 3D model is obtained by finding the index of the maximal element of the output vector $\boldsymbol{v}$. For the neural network of predicting POIs, we add an extra layer with sigmoid activation function to the end of SAE. Given the output feature vector $\boldsymbol{g}^4$ of the SAE, the output value $\boldsymbol{v} = \frac{1}{1+e^{-\boldsymbol{W}_1^5 * \boldsymbol{g}^4}}$ is the predicted probability of each vertex being a POI.

5 POI DETECTION USING DEEP NEURAL NETWORK

5.1 Training data preparation

Our training data is obtained from manual labeling. For SHREC 2011 dataset, we developed a small and simple application for visually labeling the POIs on training 3D shapes. 5 volunteers were then asked to mark POIs on these models. We adopt the strategies proposed in [7] to avoid and filter possible bad data. The volunteers are asked to select points on the surface of a 3D model likely to

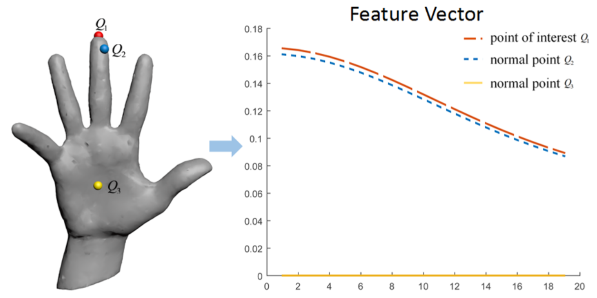


Fig. 5. An example which may confuse the detection neural network and lower its' performance. Three points Q_1, Q_2 and Q_3 on the Hand model and their corresponding feature vectors are shown in the figure. Q_1 is a POI, Q_2 and Q_3 are both normal points. Q_1 and Q_2 have very similar feature vectors but different point type. Q_2 and Q_3 have the same point type but very different feature vectors. Note that the feature vectors shown here are constructed by only using SIHKS for simplicity.

be selected by other volunteers. For the entire interactive session, the cumulative camera rotation is restricted to be less than 36 degrees to maintain approximately the same camera viewpoint. Besides, the average time per click is also not allowed to be less than one second to avoid too hasty clicks.

It is worth to point out, if we directly use the marked data as training data, unsatisfactory results may be produced because the switch between POIs and normal points is too drastic. The main reason is that the corresponding geometry features may be similar for a POI and a normal point while their labels are very different. For example, as shown in Figure 5, Q_1 is a POI, Q_2 and Q_3 are both normal points. Because Q_1 and Q_2 are very close to each other, they have very similar feature vectors (the red and blue curves shown on the right side of the figure) but different types of point. Meanwhile, Q_3 has a very different feature vector (the yellow curve) with Q_2 but the same type. This may decrease the performance of the neural network. To alleviate this problem, we design a function $P(\boldsymbol{v})$ to compute the label for each vertex $\boldsymbol{v}$ of the training 3D shapes. The function P should satisfy the following conditions.

$$\begin{cases} P(\boldsymbol{v}) \in (0,1], \ \forall\, \boldsymbol{v} \in V \\ P(\boldsymbol{c}_i) = 1, \ \text{for } i = 1, 2, \ldots, K \end{cases}$$

where V represents all vertices on a training 3D shape, $\boldsymbol{c}_i$ is a POI and K is the total number of POIs on the training 3D shape. Therefore, we regard $\boldsymbol{c}_i$ as the source vertices and construct a multi-source biharmonic distance field [58] for each training 3D shape. And then we define the function P as

$$P(\boldsymbol{v}) = \exp\left(-\frac{NBHD(\boldsymbol{v}, C)^2}{2\sigma^2}\right),$$

where C is the set of all $\boldsymbol{c}_i$, $NBHD(\boldsymbol{v}, C) = \frac{BHD(\boldsymbol{v}, C)}{\max(BHD(\boldsymbol{v}, C))}$ is the normalized value of the biharmonic distance field $BHD(\boldsymbol{v}, C)$ for $\boldsymbol{v}$, and σ is a factor to control the decreasing speed of the P with regard to BHD, which can be determined by the users.

5.2 Deep neural network training

In the experimental setting, we select three widely known feature descriptors: SIHKS, WKS, and GC, although other feature descriptors can be integrated into our algorithmic framework. These feature descriptors are deemed to have a capability of characterizing the geometric properties well from different perspectives and therefore are widely used for vertex classification, such as in mesh segmentation and other related problems [59], [60], [61], [62], [63].

For our POIs prediction neural network, we extract feature vectors for each vertex on the 3D shapes and feed them to the network for POI detection.

For our 3D shape classification neural network, we extract feature vectors for each 3D shape and feed the feature vectors to the network for classification. For GC that is a scalar field on each patch, it is easy to capture the feature distribution using histograms. For SIHKS and WKS that are vector fields on each patch, we extract the 1D feature distribution by using the well-known bag-of-feature (BoF) technique. The number of bins in the histograms and that of the bags for the bag-of-feature representation are both set to $B = 100$. This way, any feature descriptor can be adapted into our algorithmic framework. After that, we need to concatenate the feature vectors extracted by using different feature descriptors and take them as the input of our 3D shape classification neural network.

Once the deep neural networks are trained, the predicted results can be easily obtained by applying forward propagation to the corresponding vectors of a target 3D shape.

5.3 Point visualization via decision graph

Although it is hard for us to define strictly what a POI is from a mathematical point of view, the following observation is helpful. First, a POI p should have high saliency. Second, points with higher saliency than p are not available in p's neighborhood. The two aspects are crucial to determine a POI. In fact, the clustering technique proposed in [51] inspires us to visualize the points via a decision graph.

For each data point $\boldsymbol{v}_i$, we use $P_i \triangleq P(\boldsymbol{v}_i)$ to denote $\boldsymbol{v}_i$'s saliency. Let $\{d_{ij}\}_{n\times n}$ be the geodesic distance matrix between data points. We use δ_i to denote the influence distance of $\boldsymbol{v}_i$:

$$\delta_i = \min_{j:P_j > P_i} (d_{ij}),$$

Generally speaking, the vertices that are mapped to the upper right corner of the decision graph tend to be POIs. That is to say, it turns out that we can use $\gamma_i = P_i\delta_i$ to measure to what extent a given point is a POI. In Figure 6, we map each vertex of the Hand model (left) to the decision graph (right). We can clearly see that the five tip points are mapped to the upper right corner. However, this is not as easy as it seems. Imagine that there is a large plate with a flat hump in the middle. It's possible that the hump is not so conspicuous to be deemed as a POI by users but may be incorrectly taken as a POI due to its large influence distance δ. Therefore, a quantitative evaluation criterion of a POI is required.

5.4 POI selection via statistical testing

From $\{\boldsymbol{v}_i\}_{i=1}^n$, it is easy to compute $\{\delta_i\}_{i=1}^n$ and $\{\gamma_i\}_{i=1}^n$. In order to select POIs via the statistical testing method, we have to make prior assumptions on $\{\delta_i\}_{i=1}^n$ and $\{\gamma_i\}_{i=1}^n$.

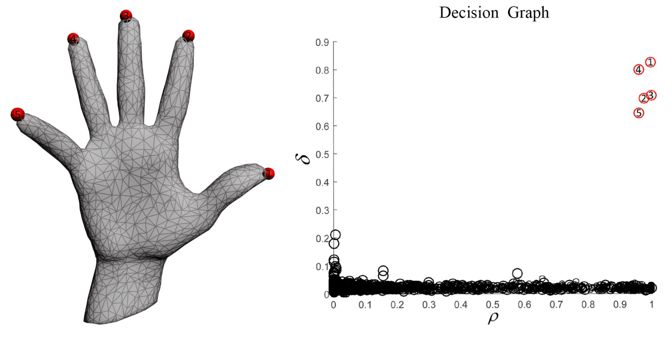


Fig. 6. Detected POIs on Hand model (left) using the corresponding decision graph (right).

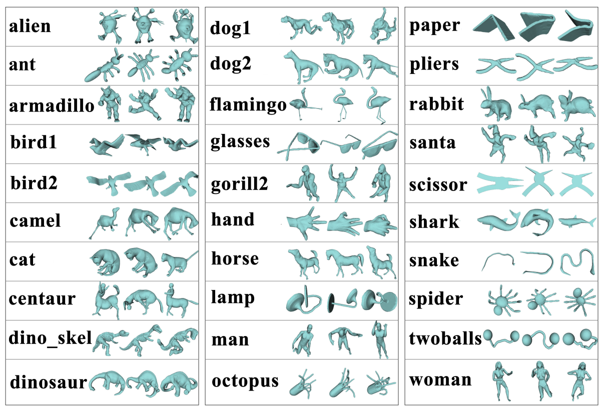


Fig. 7. Some example models of the SHREC 2011 non-rigid 3D models dataset.

Our assumption is based on the following two observations. First, δ_i is non-negative. Second, δ_i of a POI is usually relatively large and the number of POIs is usually relatively smaller than the number of other points on the surface of 3D shapes. Similar to [64], we can therefore assume that $\{\delta_i\}_{i=1}^n$ follows the long-tailed distribution. Furthermore, $\{\gamma_i\}_{i=1}^n$ can also be assumed to follow the long-tailed distribution because δ_i follows the long-tailed distribution and ρ_i is positive.

With the assumption that $\{\delta_i\}_{i=1}^n$ and $\{\gamma_i\}_{i=1}^n$ follow the long-tailed distribution, there exists some $\lambda > 0$, such that the cumulative density function has the following form:

$$F(x) = 1 - L_0(x) \cdot x^{-\lambda},$$

where L_0 is a slowly varying function (for sufficiently large x, L_0 behaves almost like a constant), and the parameter λ denotes the tail index. The key to determining if $\boldsymbol{v}$ is a POI is to make clear whether $\boldsymbol{v}$ is located on the long tail or not according to the distribution of $\{\gamma_i\}_{i=1}^n$.

According to [64], [65], we can transform $\{\gamma_i\}_{i=1}^n$ into an ordered list, $\{\hat{\gamma}_i\}_{i=1}^n$, such that $\hat{\gamma}_1 \geq \hat{\gamma}_2 \geq \cdots \geq \hat{\gamma}_n$, and then select the POIs by checking $\hat{\gamma}_m, \hat{\gamma}_{m-1}, \cdots, \hat{\gamma}_1$ one after another, where m is set to $\lceil 0.1n \rceil$ empirically. Once the point that gives $\hat{\gamma}_k$ ($k \leq m$) is identified as a POI by the following inequality, all the points that give $\hat{\gamma}_1, \hat{\gamma}_2, \cdots, \hat{\gamma}_{k-1}$ are also identified as POIs:

$$\frac{\hat{\gamma}_k}{\hat{\gamma}_{k+1}} \geq [1 - (1-\alpha)^{1/m}]^{-1/(\lambda \cdot k)},$$

where α is the parameter to define the level of significance (5% in the default setting). The tail index λ can be estimated with the modified Hill-type estimator suggested in [65]. The outward statistical testing method enables us to select POIs in an effective and robust manner.

6 Evaluation

In this section, we experimentally validate our algorithm and compare it to previous methods.

6.1 Experimental configuration

Experimental dataset. We test our method on the SHREC 2011 non-rigid 3D model dataset, which is an open dataset originally used for 3D shape classification and retrieval. The SHREC 2011 dataset contains 30 categories of 3D shapes and each category has 20 3D shapes, so that there are 600 3D shapes in SHREC 2011 dataset total. Each shape in the dataset contains about 9500 vertices. We developed a small visual tool and manually marked POIs for each 3D shape in the dataset. For each category, we randomly select 10 shapes as the training shapes, and the left shapes are regarded as test shapes. Note that our regression networks are trained on each category respectively, while our classification network is trained on the whole dataset. Some example models in SHREC 2011 dataset are shown in Figure 7. We show some representatives of our human-marked POIs in Figure 8.

Evaluation metrics. To evaluate our method, we adopt three metrics that are defined by [8], including False Negative Error (FNE), False Positive Error (FPE) and Weighted Miss Error (WME). Let G represents the set of ground truth points, and D be the set of points detected by an algorithm for a 3D shape. A point $\boldsymbol{g}_i \in G$ is considered to be correctly detected if there exists a detected point $\boldsymbol{d} \in D$ such that $\boldsymbol{d}$ is close to $\boldsymbol{g}_i$ but not closer to any other points in G. To measure the error of the detection results, FNE and FPE are defined as $FNE = 1 - N_C/|G|$ and $FPE = 1 - N_C/|D|$ respectively, where N_C is the number of correctly detected points, and $|\cdot|$ represents the size of a set. To take the prominence of g_i into account, WME is used to measure how many subjects are marked by the algorithm within a parameterized geodesic neighborhood of $\boldsymbol{g}_i$. Assuming that n_i subjects have marked a POI within a geodesic neighborhood of radius r around the ground truth point $\boldsymbol{g}_i$, WME is defined as $WME = 1 - \sum_{i=1}^{|G|} n_i \delta_i \Big/ \sum_{i=1}^{|G|} n_i$, where $\delta_i = 1$ if $\boldsymbol{g}_i$ is correctly detected; otherwise, $\delta_i = 0$. For the three metrics, a lower score represents better performance.

We have also adopted three more recent metrics proposed in [66], which are Area Under the ROC Curve (AUC), Normalized Scanpath Saliency (NSS) and Linear Correlation Coefficient (LCC), to measure the performance of our algorithm. A receiver operating characteristic (ROC) curve is a plot illustrating the performance of a binary classifier for different threshold values. The area under the ROC curve is previously widely used to compare saliency models in the 2D case. NSS measures saliency values at points selected by users (ground truth) and defined as a weighted sum of the computational saliency at those points. LCC is the correlation coefficient between two variables. We use it to

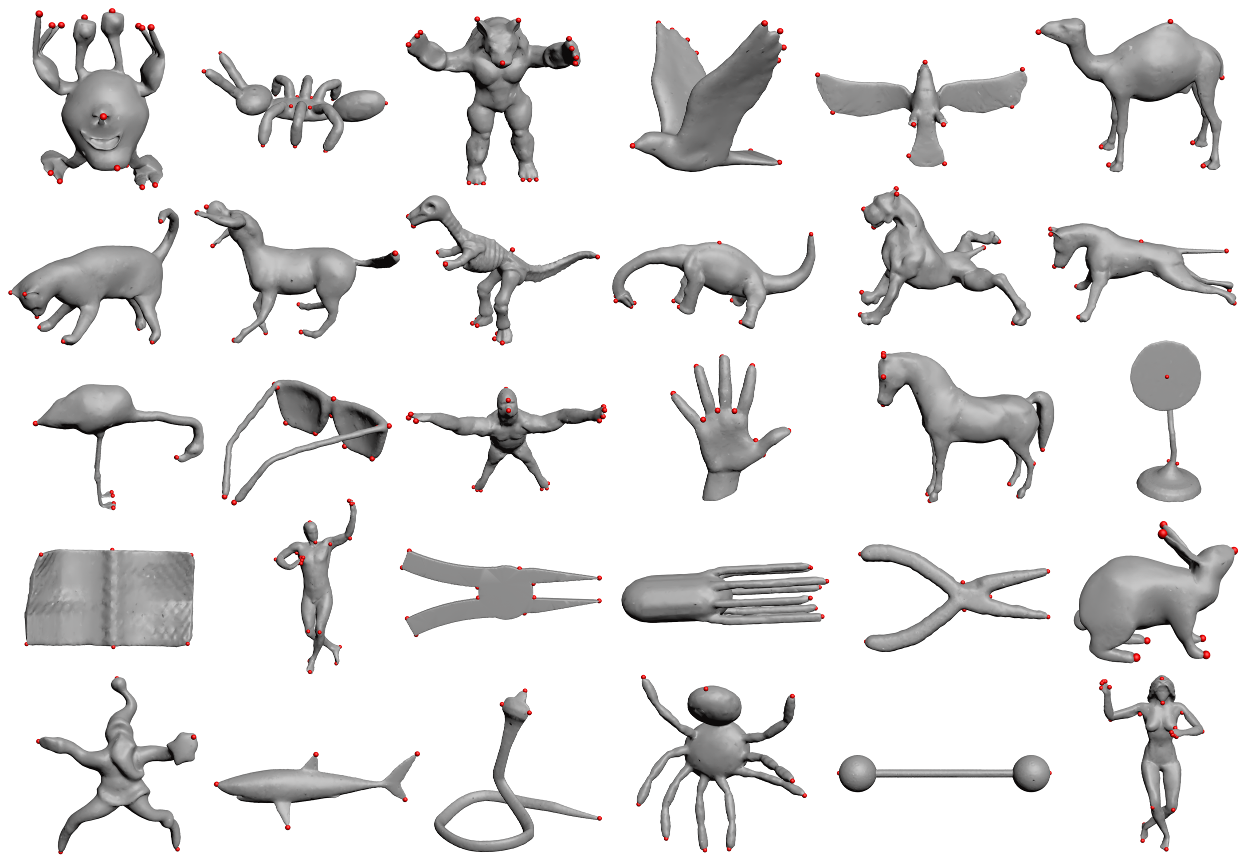


Fig. 8. Some example models with our human-marked POIs on SHREC 2011 dataset.

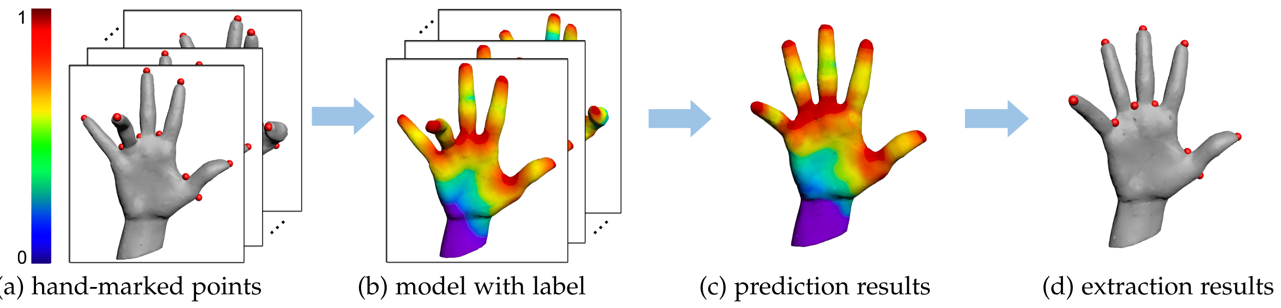


Fig. 9. Hand model POIs prediction: When the inputs are training model's POIs which are hand-marked by human subject (a), we can use the function P to get label value for each vertex (b) and then it is used as a training set for prediction neural network. The POIs for other models are finally predicted by our algorithm (c).

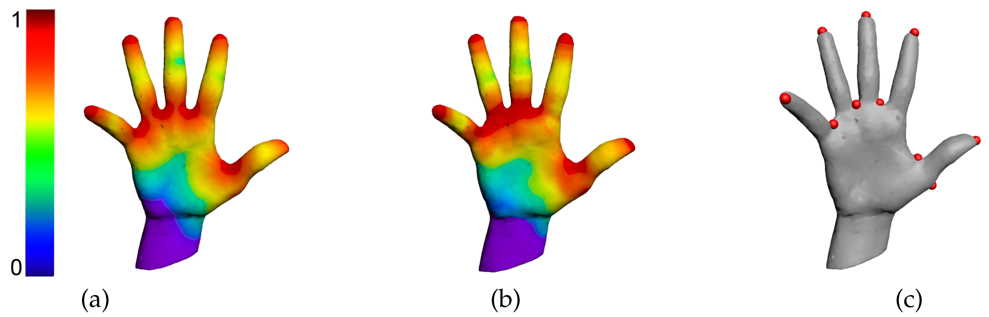


Fig. 10. Comparison of the hand-marked POIs and the result of POI detection for a Hand model: (a) Hand model with hand-marked POIs; (b) The predicted result of POIs by prediction neural network; (c) The POIs extracted from the prediction model.

measure the strength of the linear relationship of the two saliency distributions, which are obtained from an algorithm and the human-marked ground truth on a 3D shape respectively. For all three metrics, a higher score represents better performance.

Parameter settings. In this paper, the coefficient σ of controlling the decreasing speed of the function P is set to 0.05. The geodesic distance between vertices is relatively small, so in order to make the label value of each vertex distributed within the range of 0-1, we set σ to be 0.05, while larger σ will make the label values be less distinguishable. In our experiments, the dimension of the POIs prediction neural network is set to 120, where 1 dimension from GC and 19 dimensions from SIHKS and 100 dimensions from WKS. For our prediction neural network, the numbers of the neurons in each layer are set to 120-40-10-1. The corresponding setting is 300-160-80-30 for our 3D shape classification neural network.

6.2 Results

For SHREC 2011, we train the 3D model classification network and POIs prediction network respectively. To train the POIs prediction network, the hand-marked POIs are selected as ground truth and a probability field P is constructed as described in Section 5.1. As shown in Figure 9, the function P is used to compute the label for each vertex. These labels are then regarded as the ground truth that acts on the whole model and used as the training set for the prediction neural network. As a result, we can get the probability of being a POI for each vertex on the testing model. The detected POIs are finally extracted by employing the clustering technique described in Section 5.3 and 5.4.

Figure 10 shows the results of using the model with each vertex labeled by classification neural network and the prediction neural network to extract the POIs for a Hand model. We can see the results of extraction are fairly obvious and our algorithm gives desirable POIs. Figure 10 also shows the comparison between the human-marked ground truth of POIs (Figure 10 (a)) and our prediction results (Figure 10 (b)) for the Hand model. The comparison also exhibits the power of our algorithm because we can see that they are very similar.

Figure 11 shows the predicted probabilities of the POIs for some representative categories of 3D shapes in SHREC 2011. In some cases, although the distribution of the local probability is not very stable with our algorithm, it is enough for specifying the locations of POIs from the point of view of the overall model. The vast majority of POI detection results from our algorithm are desirable and consistent with our perception. Figure 12 shows more results of our algorithm.

Judging whether a point is a POI or not is a subjective problem. Our approach is data-driven and can naturally adapt to different training data, which is an important advantage. As shown in Figure 13, different predicted results can be obtained when different training data is provided. Regardless of the ridge points between neighboring fingers are considered as POIs or not, our approach can get desirable results as long as the corresponding training data is provided.

Figure 14 shows the results of POI detection by using our method for an Armadillo model with different resolutions. The results are obtained by directly applying our trained predicted neural network to 3D models with lower resolutions. We can see that although the 3D models are greatly simplified, the detected results are still in coincidence with humans' visual perception.

We must point out that the POI detection quality relies on the accuracy of our classification neural network. According to the statistics shown in Table 1, the classification neural network is able to accurately identify the category of the input 3D object, which enables us to predict POIs by referring to the geometrically similar objects in the same category.

6.3 Comparison

We compare our method with three other POI detection algorithms 3D-SIFT [70], 3D-Harris [71] and HKS-based POIs [8], [72] on SHREC 2011 database. For 3D-SIFT, the following strategy is used to identify the POIs on a 3D model. First, a scale space is constructed by applying different layers of 3D Gaussian filters to the voxelized model. Then, the Difference of Gaussian (DoG) for each level is obtained by subtracting the original model from the scaled model. The extrema points are selected by searching the DoG space in both spatial and scale dimensions. Their closest vertices on the original mesh are marked as final POIs. For both 3D-Harris and HKS, the local maxima are first selected by considering the neighborhood of a vertex. For HKS, 2-ring neighborhood is considered and all local maxima are then detected as POIs. However, only 1-ring neighborhood is considered and a certain number of local maxima with highest Harris response are then selected as POIs for 3D-Harris. The results of 3D-Harris and HKS-based POIs are obtained by using the implementation provided by corresponding authors. The results of 3D-SIFT are got from using our own implementation. All the algorithms are compared against human-marked POIs by 16 subjects. Figure 15 shows an example of the POI detection results by our algorithm and three other algorithms. We can see that the results from our algorithm and HKS-based POIs are better than others. We use three metrics provided by [8] for evaluation and the detailed statistics plots are shown in Figure 16, where a quantitative comparison on all models is provided. It can be seen that our POI detection algorithms outperforms the other detection approaches for all three metrics FPE, FNE and WME.

We also compare our method with the state-of-the-art method proposed in [7] (Schelling Point method, SP) on the SHREC 2007 dataset. The dataset contains 20 categories of 3D models and each category has 20 models. Schelling Point method is also data-driven but different from the strategy adopted in this paper – they use a decision tree based regression model (M5P regression trees as provided by Weka) to detect POIs on 3D models. In their work, the authors provide a human-labeled ground truth data set on the SHREC 2007 dataset, which facilitates us to directly train our neural network of predicting POIs. For each category of 3D models, 10 randomly selected models are used for training while the remaining 10 models

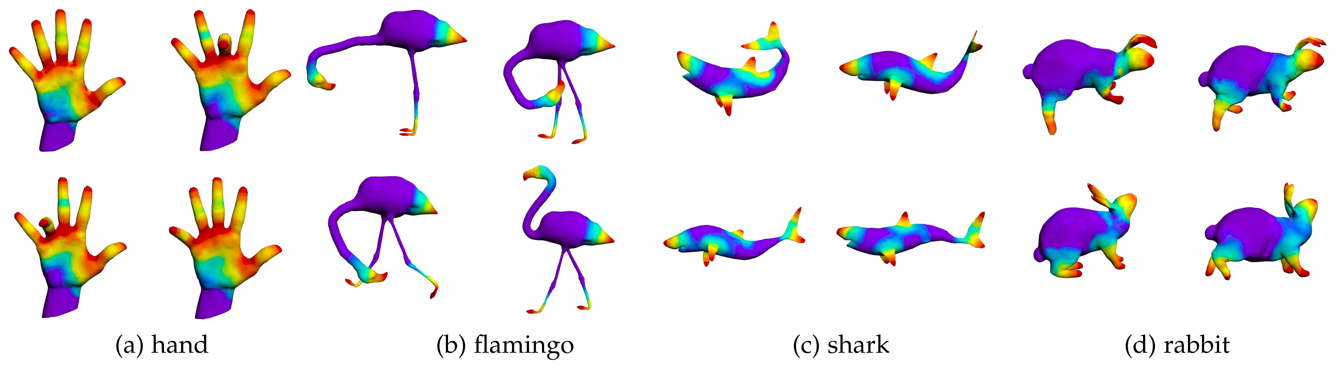


Fig. 11. Examples of the predicted probabilities of the POIs on some representative classes of models:(a)hand models;(b)flamingo models;(c)shark models;(d)rabbit models.

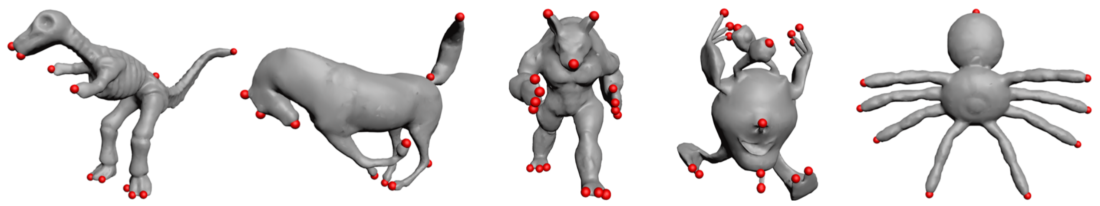


Fig. 12. Some results of our algorithm. POIs in coincidence with humans' expectation are successfully detected by our method.

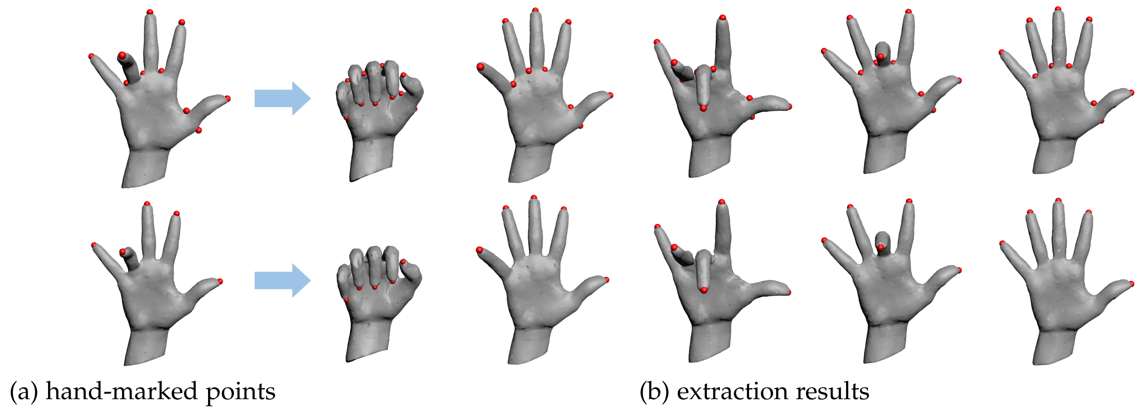


Fig. 13. Results of our method trained with different input points. The top row shows the detected results where the ridge points between neighboring fingers are considered and marked as POIs in the training data. As a comparison, the bottom row shows the detected results where the ridge points between neighboring fingers are not regarded as POIs in the training data.

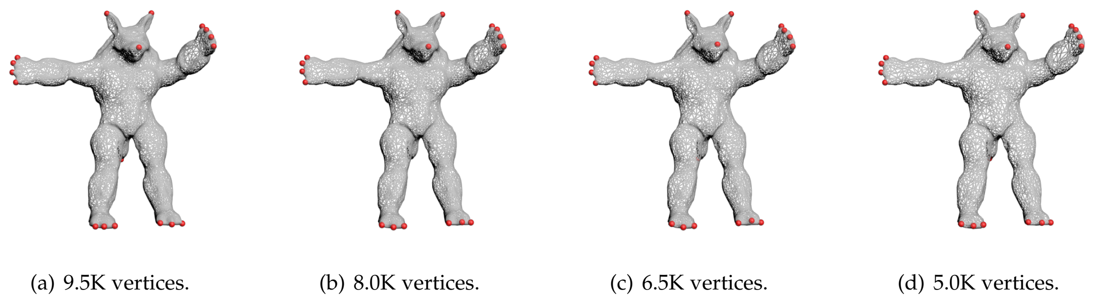
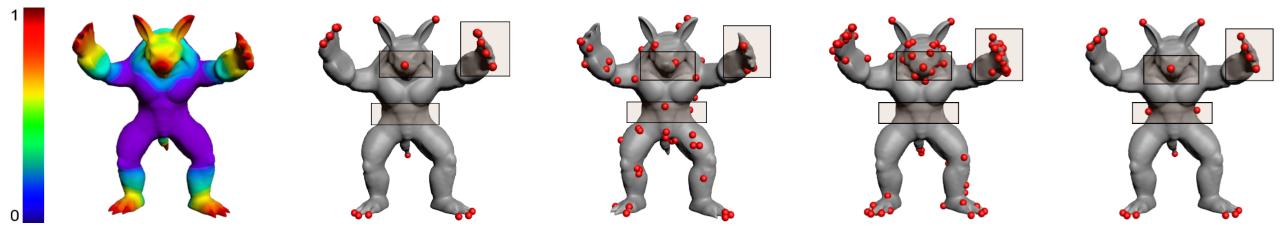
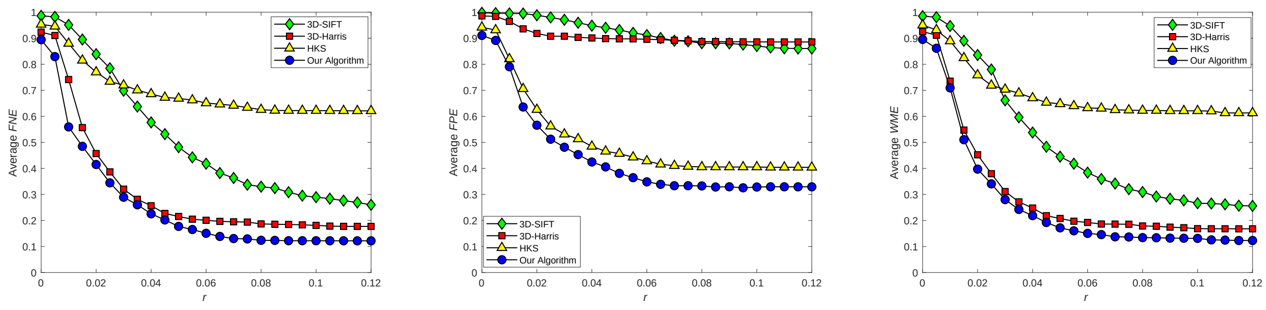


Fig. 14. The results of POI detection by using our method for an Armadillo model with different resolutions.



(a) predicted probabilities (b) our algorithm (c) 3D-SIFT (d) 3D-Harris (e) HKS

Fig. 15. Comparison between results from our algorithm and the other three feature point detection algorithms. (a) shows the predicted probabilities of the POIs using our algorithm. The approaches used here include (b) Our algorithm; (c) 3D-SIFT; (d) 3D-Harris; (e) HKS.



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

Fig. 16. Comparison between our algorithm with state-of-the-art feature points detection algorithms on all models using False Negative Error, False Positive Error, and Weighted Miss Error. The data for each metric is obtained from averaging over all models on SHREC 2011 datasets.

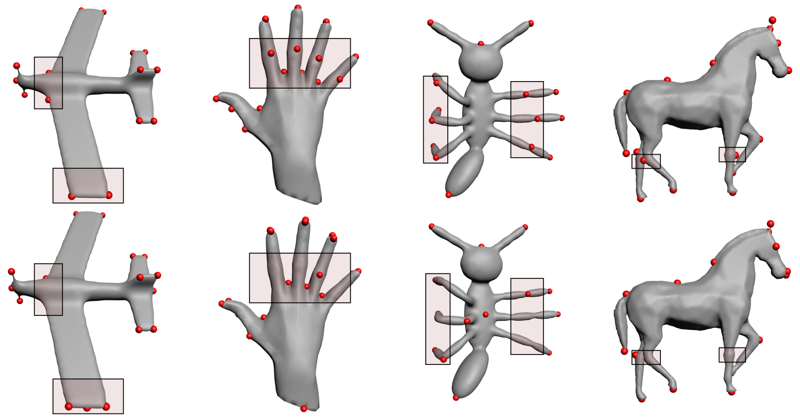


Fig. 17. The comparison between our method and the method proposed in [7]. The top line shows some examples of detected POIs by using our method. The bottom line shows the corresponding examples of detected POIs by using the method proposed in [7].

TABLE 1
The accuracy rates of our classification neural network for each category of models in the SHREC 2011 dataset.

Object categories	Alien	Ants	Armadillo	Bird1	Bird2	Camel	Cat	Centaur	Dino_ske	Dinosaur
Accuracy rate	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
Object categories	Dog1	Dog2	Flamingo	Glasses	Gorilla	Hand	Horse	Lamp	Laptop	Man
Accuracy rate	100%	100%	100%	90%	100%	100%	100%	100%	100%	100%
Object categories	MyScissor	Octopus	Pliers	Rabbit	Santa	Shark	Snake	Spiders	Two_balls	Woman
Accuracy rate	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%

TABLE 2
The comparison of AUC scores from our algorithm, applying model trained from SHREC 2011 (SH2011), Schelling Point (SP, [7]), Cluster-based point set saliency (CS, [67]), Saliency of large point sets (LS, [68]), Mesh saliency via spectral processing (MS, [69]) and PCA-based saliency (PS, [66]) on the SHREC 2007 non-rigid 3D models dataset. A higher score represents better performance.

Algorithms	Our algorithm	SH2011	SP	CS	LS	MS	PS
Human	**0.6754**	0.6383	0.6311	0.5745	0.5893	0.5695	0.5921
Cup	**0.6459**	–	0.5864	0.6122	0.6192	0.5934	0.6135
Glass	**0.6351**	0.6018	0.6097	0.5225	0.5727	0.5297	0.5530
Airplane	0.6625	0.6597	0.6609	0.6308	**0.6705**	0.6160	0.6409
Ant	**0.6524**	0.6468	0.6029	0.6056	0.6349	0.5696	0.5791
Chair	**0.7562**	–	0.6492	0.5871	0.6566	0.5505	0.5799
Octopus	**0.6694**	0.6592	0.5679	0.5480	0.6237	0.5342	0.5726
Table	**0.6738**	–	0.6162	0.6313	0.6651	0.5802	0.6168
Teddy	**0.7584**	–	0.6596	0.5671	0.5641	0.5530	0.5682
Hand	**0.6341**	–	0.5668	0.6060	0.6339	0.5763	0.6022
Plier	**0.6537**	0.6182	0.6128	0.5997	0.6236	0.5615	0.5754
Fish	0.6039	0.5902	0.6562	0.6651	0.6717	0.6309	**0.6736**
Bird	**0.6637**	0.6390	0.6169	0.6010	0.6217	0.5627	0.5984
Spring	**0.6351**	–	0.5751	0.5512	0.5545	0.5523	0.5339
Armadillo	0.6825	**0.6859**	0.6792	0.6560	0.6656	0.5996	0.6570
Buste	**0.6814**	–	0.5757	0.6236	0.6260	0.5620	0.6352
Mechanic	**0.7359**	–	0.6548	0.6964	0.6932	0.5325	0.7065
Bearing	**0.6537**	–	0.6063	0.6472	0.6387	0.4986	0.6322
Vase	**0.6328**	–	0.5540	0.6158	0.6217	0.6058	0.6251
Four-legged	**0.6235**	0.6140	0.6056	0.6024	0.6168	0.6005	0.6112
Average	**0.6665**	–	0.6144	0.6072	0.6282	0.5689	0.6083

TABLE 3
The comparison of LCC scores from our algorithm, applying model trained from SHREC 2011 (SH2011), Schelling Point (SP, [7]), Cluster-based point set saliency (CS, [67]), Saliency of large point sets (LS, [68]), Mesh saliency via spectral processing (MS, [69]) and PCA-based saliency (PS, [66]) on the SHREC 2007 non-rigid 3D models dataset. A higher score represents better performance.

Algorithms	Our algorithm	SH2011	SP	CS	LS	MS	PS
Human	**0.4503**	0.4382	0.4047	0.2544	0.1895	0.2580	0.3292
Cup	**0.3769**	–	0.2606	0.2924	0.3478	0.2444	0.3039
Glass	**0.5020**	0.4834	0.4689	0.1201	0.4837	0.4120	0.3353
Airplane	0.6328	0.6317	**0.7107**	0.6936	0.6357	0.4876	0.6601
Ant	0.6584	0.6487	0.5944	0.7356	**0.7658**	0.3509	0.3357
Chair	**0.7326**	–	0.5749	0.5152	0.6995	0.2723	0.3652
Octopus	**0.6066**	0.6019	0.3385	0.2228	0.4654	0.2467	0.3565
Table	0.5947	–	0.4505	0.5831	**0.7111**	0.2727	0.3076
Teddy	**0.5861**	–	0.4616	0.1184	0.1246	0.1224	0.1188
Hand	**0.5542**	–	0.1366	0.5439	0.5372	0.3077	0.3687
Plier	0.6447	0.6256	**0.6602**	0.5978	0.6591	0.2945	0.3214
Fish	0.6422	0.6381	0.6914	**0.6967**	0.6349	0.4638	0.5889
Bird	0.5633	0.5233	**0.6248**	0.5578	0.5690	0.4099	0.5067
Spring	**0.5764**	–	0.3178	0.4838	0.5227	0.3977	0.1859
Armadillo	0.5249	0.5195	**0.6942**	0.5495	0.4083	0.2627	0.5589
Buste	**0.4138**	–	0.1716	0.2540	0.2657	0.1240	0.2755
Mechanic	**0.7561**	–	0.7434	0.4064	0.4237	0.0548	0.5756
Bearing	**0.4836**	–	0.3275	0.2676	0.3701	0.0316	0.2933
Vase	**0.4713**	–	0.2815	0.3673	0.4228	0.3133	0.3285
Four-legged	**0.5062**	0.4983	0.4264	0.3819	0.3896	0.3682	0.2861
Average	**0.5639**	–	0.4670	0.4321	0.4813	0.2848	0.3701

for testing. The saliency of each vertex on the testing 3D models is then predicted by applying the regression model. Figure 17 shows a comparison between the results from the two algorithms. We can see that our method is able to achieve better detection results due to the use of deep neural networks. A more detailed quantitative comparison between our algorithm and Schelling Point method on the SHREC 2007 dataset is presented in Table 2, Table 3 and Table 4. The measures used for comparison include AUC, LCC, and NSS – a higher score indicates better performance. It can be seen that our method has a superior performance for most of categories of 3D models. Besides, our method obtains better scores when averaging all the categories.

Furthermore, we compare our method with four other state-of-the-art methods including Cluster-based point set saliency (CS) [67], Saliency of large point sets (LS) [68], Mesh saliency via spectral processing (MS) [69] and PCA-based saliency (PS) [66] on SHREC 2007 dataset. The resulting AUC, LCC, and NSS metrics are respectively shown in Table 2, Table 3 and Table 4. From the comparison tables, it can be seen that our algorithm outperforms the other algorithms for most of the categories. Although our algorithm is not always the best, the average metric score of our algorithm is stably higher than the other algorithms.

We further test the performance of applying our neural network trained from SHREC 2011 dataset to SHREC 2007

TABLE 4
The comparison of NSS scores from our algorithm, applying model trained from SHREC 2011 (SH2011), Schelling Point (SP, [7]), Cluster-based point set saliency (CS, [67]), Saliency of large point sets (LS, [68]), Mesh saliency via spectral processing (MS, [69]) and PCA-based saliency (PS, [66]) on the SHREC 2007 non-rigid 3D models dataset. A higher score represents better performance.

Algorithms	Our algorithm	SH2011	SP	CS	LS	MS	PS
Human	**0.9034**	0.8587	0.6743	0.6417	0.5829	0.4724	0.7152
Cup	**0.9247**	–	0.4852	0.6196	0.7968	0.4520	0.6268
Glass	**0.7067**	0.6813	0.6808	0.2157	0.5977	0.3762	0.4087
Airplane	1.2334	1.1824	**1.2798**	1.1604	1.1828	0.8449	1.2208
Ant	**1.4279**	1.4008	0.8294	1.1379	1.2840	0.5587	0.5216
Chair	**1.7978**	–	1.2605	0.9017	1.4085	0.4513	0.7007
Octopus	**1.0351**	1.0142	0.5490	0.4190	0.7800	0.3694	0.5070
Table	1.4562	–	1.0164	1.2281	**1.5990**	0.5905	0.8030
Teddy	**1.8746**	–	1.2297	0.4124	0.4369	0.3890	0.4154
Hand	**1.0989**	–	0.2433	0.9133	1.0030	0.5136	0.6396
Plier	**1.1473**	1.1205	0.9109	0.7901	0.9528	0.5398	0.4843
Fish	1.0056	0.9313	**1.2753**	1.2276	1.1726	0.8235	1.1782
Bird	**1.1832**	0.9719	0.9136	0.8240	0.8987	0.5259	0.8021
Spring	**1.0531**	–	0.5095	0.6492	0.7025	0.5274	0.3560
Armadillo	1.9101	1.8944	**2.0643**	1.7027	1.3418	0.7206	1.6718
Buste	**1.1397**	–	0.4797	0.8002	0.8041	0.3266	0.8587
Mechanic	**2.3988**	–	2.1069	1.4833	1.5650	0.2216	2.0954
Bearing	**0.8752**	–	0.5736	0.7923	0.8141	0.0201	0.7677
Vase	**1.0906**	–	0.4997	0.7767	0.8447	0.6087	0.7222
Four-legged	**1.0717**	0.9452	0.7934	0.8440	0.8887	0.7277	0.6792
Average	**1.2667**	–	0.9188	0.8770	0.9828	0.5030	0.8087

TABLE 5
The average running time of each step for our neural network of predicting POIs.

Steps	Data preprocessing	Training of neural network	POIs extraction
Time (s)	66.446	518.537	46.301

dataset. The results are also shown in Table 2, Table 3 and Table 4. It can be observed that comparable performances are gained. It is worth pointing out that only 10 categories of 3D shapes are selected and tested because the other categories have no geometric similarities.

6.4 Performance

We implemented the proposed algorithm in Matlab and C++. In average, our algorithm takes around 10 minutes to process a single model on a PC with 2.60GHz CPU and with 128GB RAM. As shown in Table 5, the bottleneck lies in the training step that spends over 80% of the total compute time.

7 Limitation and future work

First, the detection performance of our method depends on the quality of manually labeled data. A reliable training dataset is necessary to guarantee good results.

Second, a classification step is pre-computed before we predict POIs on the input 3D model. This is due to the lack of training data. If we had a sufficiently large training data set, our algorithm would not need to include the classification step.

Finally, the current implementation of the training process is very time-consuming. In the future, we plan to explore parallelized implementations to reduce the time complexity and facilitate the application of our method in practice. Employing specialized hardware for deep networks (such as Tensor Processing Units) may be very helpful.

8 Conclusion

In this paper, we propose a novel supervised method for detecting POIs on 3D shapes based on deep learning techniques. The key idea is to predict a saliency map to encode the possibility of being a POI and then extract typical POIs by clustering. Our method is data-driven and able to predict POIs in a similar way as human observers. Extensive experimental results show that our method outperforms state-of-the-art approaches and reveals how the distribution of POIs depends on geometric features.

Acknowledgments

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions. This work is supported by National Natural Science Foundation of China (11226328, 61672482, 11626253, 61772016, 61572022), National Science Foundation (IIS-1617172, IIS-1622360), Natural Science Foundation of Zhejiang Province (LY17F020018), Ningbo Leader and Top-notch Talent Training Project, Ningbo Innovative Team: The intelligent big data engineering application for life and health (2016C11024), the Fundamental Research Funds of Shandong University, and the Open Project Program of the State Key Lab of CAD & CG (Grant No. A1702), Zhejiang University.

References

[1] 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.

[2] 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.

[3] 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.

[4] 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.

[5] 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.

[6] Y. Kim and A. Varshney, "Persuading visual attention through geometry," *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 4, pp. 772–782, 2008.

[7] 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.

[8] 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.

[9] S. Wang, N. Li, S. Li, Z. Luo, Z. Su, and H. Qin, "Multi-scale mesh saliency based on low-rank and sparse analysis in shape feature space," *Computer Aided Geometric Design*, vol. 35, pp. 206–214, 2015.

[10] P. Tao, J. Cao, S. Li, X. Liu, and L. Liu, "Mesh saliency via ranking unsalient patches in a descriptor space," *Computers & Graphics*, vol. 46, pp. 264–274, 2015.

[11] 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.

[12] K. Guo, D. Zou, and X. Chen, "3D mesh labeling via deep convolutional neural networks," *ACM Transactions on Graphics*, vol. 35, no. 1, pp. 1–12, 2015.

[13] 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.

[14] P. Shilane and T. Funkhouser, "Distinctive regions of 3D surfaces," *ACM Transactions on Graphics*, vol. 26, no. 2, p. 7, 2007.

[15] G. Leifman, E. Shtrom, and A. Tal, "Surface regions of interest for viewpoint selection," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 414–421.

[16] G. Guy and G. Medioni, "Inference of surfaces, 3D curves, and junctions from sparse, noisy, 3D data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 11, pp. 1265–1277, 1997.

[17] H. Yee, S. Pattanaik, and D. P. Greenberg, "Spatiotemporal sensitivity and visual attention for efficient rendering of dynamic environments," *ACM Transactions on Graphics*, vol. 20, no. 1, pp. 39–65, 2001.

[18] L. Itti, C. Koch, E. Niebur, and Others, "A model of saliency-based visual attention for rapid scene analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 11, pp. 1254–1259, 1998.

[19] R. Mantiuk, K. Myszkowski, and S. Pattanaik, "Attention guided mpeg compression for computer animations," in *Proceedings of the 19th Spring Conference on Computer Graphics*. ACM, 2003, pp. 239–244.

[20] C. Koch and S. Ullman, "Shifts in selective visual attention: towards the underlying neural circuitry," in *Matters of intelligence*. Springer, 1987, pp. 115–141.

[21] C. H. Lee, A. Varshney, and D. W. Jacobs, "Mesh saliency," *ACM Transactions on Graphics*, vol. 24, no. 3, pp. 659–666, 2005.

[22] X. Hou and L. Zhang, "Saliency detection: A spectral residual approach," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2007, pp. 1–8.

[23] 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. 6:1–6:17, 2014.

[24] M.-M. Cheng, N. J. Mitra, X. Huang, P. H. Torr, and S.-M. Hu, "Global contrast based salient region detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 569–582, 2015.

[25] L. Duan, C. Wu, J. Miao, L. Qing, and Y. Fu, "Visual saliency detection by spatially weighted dissimilarity," in *2011 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2011, pp. 473–480.

[26] F. Perazzi, P. Kr A Henb U Hl, Y. Pritch, and A. Hornung, "Saliency filters: Contrast based filtering for salient region detection," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 733–740.

[27] A. Borji, M.-M. Cheng, H. Jiang, and J. Li, "Salient object detection: A benchmark," *IEEE Transactions on Image Processing*, vol. 24, no. 12, pp. 5706–5722, 2015.

[28] Y. Wei, F. Wen, W. Zhu, and J. Sun, "Geodesic saliency using background priors," in *European Conference on Computer Vision*. Springer, 2012, pp. 29–42.

[29] M. Lau, K. Dev, W. Shi, J. Dorsey, and H. Rushmeier, "Tactile mesh saliency," *ACM Transactions on Graphics*, vol. 35, no. 4, p. 52, 2016.

[30] L. Shapira, A. Shamir, and D. Cohen-Or, "Consistent mesh partitioning and skeletonisation using the shape diameter function," *The Visual Computer*, vol. 24, no. 4, pp. 249–259, 2008.

[31] L. Shapira, S. Shalom, A. Shamir, D. Cohen-Or, and H. Zhang, "Contextual part analogies in 3D objects," *International Journal of Computer Vision*, vol. 89, no. 2, pp. 309–326, 2010.

[32] M. M. Bronstein and I. Kokkinos, "Scale-invariant heat kernel signatures for non-rigid shape recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 1704–1711.

[33] M. Aubry, U. Schlickewei, and D. Cremers, "The wave kernel signature: A quantum mechanical approach to shape analysis," in *IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*. IEEE, 2011, pp. 1626–1633.

[34] J. Knopp, M. Prasad, and L. Van Gool, "Orientation invariant 3D object classification using Hough transform based methods," in *Proceedings of the ACM Workshop on 3D Object Retrieval*, 2010, pp. 15–20.

[35] D. Smeets, J. Keustermans, D. Vandermeulen, and P. Suetens, "meshsift: Local surface features for 3D face recognition under expression variations and partial data," *Computer Vision and Image Understanding*, vol. 117, no. 2, pp. 158–169, 2013.

[36] E. Rodolà, S. Bulò, T. Windheuser, and M. Vestner, "Dense non-rigid shape correspondence using random forests," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 4177–4184.

[37] D. Boscaini, J. Masci, E. Rodolà, M. M. Bronstein, and D. Cremers, "Anisotropic diffusion descriptors," *Computer Graphics Forum*, vol. 35, no. 2, pp. 431–441, 2016.

[38] D. Boscaini, J. Masci, E. Rodolà, and M. M. Bronstein, "Learning shape correspondence with anisotropic convolutional neural networks," in *Neural Information Processing Systems*, 2016, pp. 3189–3197.

[39] Z. Zhu, X. Wang, S. Bai, and C. Yao, "Deep learning representation using autoencoder for 3D shape retrieval," in *International Conference on Security, Pattern Analysis, and Cybernetics*, 2014, pp. 279–284.

[40] A. Sinha, J. Bai, and K. Ramani, "Deep learning 3D shape surfaces using geometry images," in *European Conference on Computer Vision*. Springer, 2016, pp. 223–240.

[41] D. Ezuz, J. Solomon, V. G. Kim, and M. Ben-Chen, "GWCNN: A metric alignment layer for deep shape analysis," *Computer Graphics Forum*, vol. 36, no. 5, pp. 49–57, 2017.

[42] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and 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.

[43] A. Brock, T. Lim, J. M. Ritchie, and N. Weston, "Generative and discriminative voxel modeling with convolutional neural networks," in *Neural Information Processing Conference*, 2016, pp. 1–9.

[44] P.-S. Wang, Y. Liu, Y.-X. Guo, C.-Y. Sun, and X. Tong, "O-CNN: octree-based convolutional neural networks for 3D shape analysis," *ACM Transactions on Graphics*, vol. 36, no. 4, p. 72, 2017.

[45] Y. Fang, J. Xie, G. Dai, M. Wang, F. Zhu, T. Xu, and E. Wong, "3D deep shape descriptor," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 2319–2328.

[46] L. Luciano and A. B. Hamza, "Deep learning with geodesic moments for 3D shape classification," *Pattern Recognition Letters*, vol. 105, pp. 182–190, 2017.

[47] J. Masci, D. Boscaini, M. Bronstein, and P. Vandergheynst, "Geodesic convolutional neural networks on Riemannian manifolds," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2015, pp. 37–45.

[48] F. Monti, D. Boscaini, J. Masci, E. Rodolà, J. Svoboda, and M. M. Bronstein, "Geometric deep learning on graphs and manifolds using mixture model CNNs," in *Computer Vision and Pattern Recognition*, 2017, pp. 5115–5124.
[49] D. Boscaini, J. Masci, S. Melzi, M. M. Bronstein, U. Castellani, and P. Vandergheynst, "Learning class-specific descriptors for deformable shapes using localized spectral convolutional networks," in *Eurographics Symposium on Geometry Processing*, 2015, pp. 13–23.
[50] 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.
[51] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, no. 6191, pp. 1492–1496, 2014.
[52] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," *Advances in Neural Information Processing Systems*, vol. 19, pp. 153–160, 2007.
[53] T. Dietterich, "Overfitting and undercomputing in machine learning," *ACM Computing Surveys*, vol. 27, no. 3, pp. 326–327, 1995.
[54] F. Perez-Cruz, "Kullback-leibler divergence estimation of continuous distributions," in *IEEE International Symposium on Information Theory*, 2008, pp. 1666–1670.
[55] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, 1986.
[56] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.
[57] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.
[58] Y. Lipman, R. M. Rustamov, and T. A. Funkhouser, "Biharmonic distance," *ACM Transactions on Graphics*, vol. 29, no. 3, pp. 27:1–27:11, 2010.
[59] E. Kalogerakis, A. Hertzmann, and K. Singh, "Learning 3D mesh segmentation and labeling," *ACM Transactions on Graphics*, vol. 29, no. 4, pp. 1–12, 2010.
[60] R. Hu, L. Fan, and L. Liu, "Co-segmentation of 3D shapes via subspace clustering," *Computer Graphics Forum*, vol. 31, no. 5, pp. 1703–1713, 2012.
[61] M. Meng, J. Xia, J. Luo, and Y. He, "Unsupervised co-segmentation for 3D shapes using iterative multi-label optimization," *Computer-Aided Design*, vol. 45, no. 2, pp. 312–320, 2013.
[62] Z. Xie, K. Xu, L. Liu, and Y. Xiong, "3D shape segmentation and labeling via extreme learning machine," *Computer Graphics Forum*, vol. 33, no. 5, pp. 85 – 95, 2014.
[63] Z. Shu, C. Qi, S. Xin, C. Hu, L. Wang, Y. Zhang, and L. Liu, "Unsupervised 3D shape segmentation and co-segmentation via deep learning," *Computer Aided Geometric Design*, vol. 43, pp. 39–52, 2016.
[64] G. Wang and Q. Song, "Automatic clustering via outward statistical testing on density metrics," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 8, pp. 1971–1985, 2016.
[65] C. Schluter and M. Trede, "Identifying multiple outliers in heavy-tailed distributions with an application to market crashes," *Journal of Empirical Finance*, vol. 15, no. 4, pp. 700–713, 2008.
[66] 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.
[67] F. P. Tasse, J. Kosinka, and N. Dodgson, "Cluster-based point set saliency," in *IEEE International Conference on Computer Vision*, 2015, pp. 163–171.
[68] E. Shtrom, G. Leifman, and A. Tal, "Saliency detection in large point sets," in *IEEE International Conference on Computer Vision*, 2013, pp. 3591–3598.
[69] 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.
[70] 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.
[71] 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.
[72] 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.

Zhenyu Shu got his Ph.D. degree in 2010 at Zhejiang University, China. He is now working as an associate professor at Ningbo Institute of Technology, Zhejiang University. His research interests include computer graphics, digital geometry processing and machine learning. He has published over 30 papers in international conferences or journals.

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

Xin Xu is currently a postgraduate in the Faculty of Electrical Engineering and Computer Science at Ningbo University. Her research interests include digital geometry processing and machine learning.

Ligang Liu received the BSc degree in 1996 and the Ph.D. 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.

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