Surface Regions of Interest for Viewpoint Selection

George Leifman, Elizabeth Shtrom, and Ayellet Tal

Abstract—While the detection of the interesting regions in images has been extensively studied, relatively few papers have addressed surfaces. This paper proposes an algorithm for detecting the regions of interest of surfaces. It looks for regions that are distinct both locally and globally and accounts for the distance to the foci of attention. It is also shown how this algorithm can be adopted to saliency detection in point clouds. Many applications can utilize these regions. In this paper we explore one such application—viewpoint selection. The most informative views are those that collectively provide the most descriptive presentation of the surface. We show that our results compete favorably with the state-of-the-art results.

Index Terms—Saliency detection, surfaces, point clouds

1 Introduction

Many problems in computer vision and computer graphics benefit from the detection of the most interesting regions of surfaces. Examples include face recognition [1], similarity [2], alignment [2], simplification [3], icon generation [3], abstraction [4], and viewpoint selection [5].

What are these interesting regions? Lee et al. [5] define a measure of surface saliency using a center-surround operator on Gaussian-weighted mean curvatures. Measures of importance have been defined also by [2], [6], [7] and others. These algorithms detect regions where the curvature of a surface is inconsistent with its immediate surroundings. Thus, they take the human vision's tendency to be drawn to differences into account. In [3] surface distinctiveness is based on the similarity between a given surface and similar objects in its class.

We also look for region distinctness (Fig. 1). However, unlike prior approaches, which focus on local distinctness, we take into consideration also global distinctness. This accounts for 3D textures, where local distinctness is high, while global distinctness is low. Additionally, we look for regions of interest (ROI), rather than for isolated points. This consideration follows the human tendency to group close items together.

We propose a novel algorithm that detects regions of interest on surfaces by realizing the considerations above. We assume that the surface is given as a triangulated mesh that consists of vertices and faces. To capture distinctness, we discuss a vertex descriptor that characterizes the geometry in the neighborhood of a vertex. Moreover, we introduce an algorithm that detects surface's extremities, which are typically distinct. To take distance to foci into account, we show how to adjust the distinctness by computing *patch association*.

We also show how our technique can be adopted to handle point clouds, which have been commonly used in recent years. The difference between meshes and point clouds is that point clouds have no connectivity information associated with them. Hence, the existing algorithms that use geodesic distances and simplification cannot be applied to point clouds. Nevertheless, we will show that the main concepts of the approach are still applicable for point clouds.

We demonstrate an extensive evaluation of our results using a benchmark for 3D interest point detection, which has been recently published. The benchmark provides quantitative evaluation for detecting interest points versus human-marked points. We show that our algorithm outperforms all the state-of-the-art methods with released code. The code of some saliency detection algorithms, introduced in recent papers, is not available. Therefore, in these cases, we provide qualitative evaluation by running our code on the same models given in these papers and showing the results side by side.

We demonstrate the utility of our regions of interest in viewpoint selection. The goal is to automatically select the camera position from which the most informative and intuitive view of the shape is seen [5], [8], [9], [10]. We show that our scheme outperforms the state-of-the-art methods. In many applications, such as the creation of thumbnails for huge repositories or catalogs of 3D models, it is necessary to automatically capture an informative image of an object. Good images of 3D objects can also be used for various computer vision problems, such as shape recognition or classification [1], [11], [12].

Our contributions are hence threefold. First, we propose a novel algorithm for detecting the regions of interest of a surface (Sections 3 and 4). Second, we show how our algorithm can be adopted to saliency detection in point clouds (Sections 5). Finally, we present an algorithm for

- *G. Leifman is with the Electrical engineering department, Technion – Israel Institute of Technology and the Media Lab, M.I.T. – Massachusetts Institute of Technology. E-mail: gleifman@mit.edu.*
- *E. Shtrom and A. Tal are with the Electrical engineering department, Technion – Israel Institute of Technology. E-mail: slizas@gmail.com, ayellet@ee.technion.ac.il.*

Manuscript received 9 July 2015; revised 20 Jan. 2016; accepted 23 Jan. 2016. Date of publication 0 . 0000; date of current version 0 . 0000. Recommended for acceptance by D. Forsyth. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below. Digital Object Identifier no. 10.1109/TPAMI.2016.2522437

Fig. 1. The left image shows the regions of interest, where red is the most interesting and blue is the least. The other images present the surface from the two most descriptive viewpoints, as calculated by our algorithm.

viewpoint selection, which utilizes our regions of interest (Sections 6 and 7).

A preliminary version of this paper was an oral presentation in the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'12) [13].

2 Related Work

Our work is closely related to saliency detection, which has attracted a lot of attention in computer vision. Most of the work has concentrated on images and videos. Fewer attempts were made to define saliency of 3D objects and 3D scenes.

2.1 Saliency in Images and Videos

In 1980 Treisman and Gelade [14] proposed a feature-integration theory – a psychological theory that describes how a person pieces together separate features of an object to create a more complete perception of the perceived object. This theory aggregates several feature types, such as color, orientation, and intensity, to explain how the eyes absorb information to somehow "experience" the object one is seeing. Later, Koch and Ullman [15] propose the concept of a *saliency map* as a measure of visual attraction of every point in the scene. This idea was first realized and verified by Itti et al. [16], who proposed a complete model of human attention in images.

Since then, a lot of progress in image saliency has been made. Some of the models use only low-level information [17], [18], [19], while others use high-level object detection [20], [21] or context [22], [23]. The reader is referred to a recent survey on the subject for more details [24].

Most existing video saliency methods are based on image attention models, taking into account motion cues [25], [26], [27], [28]. Other works, such as Cui et al. [29], concentrate on motion saliency only and detect it by using temporal spectral analysis. Rudoy et al. [30] focus on a small number of candidate gaze locations and learn conditional gaze transitions over time. Zhou et al. [31] introduce a motion saliency method that combines various low-level features with region-based contrast analysis, to generate low-framerate videos. Huang et al. [32] propose to use Deep Neural Networks to reduce the semantic gap in saliency prediction. Recently, some new metrics to evaluate saliency were proposed. In [33], a new measure F_β^w is proposed, which offers a unified solution to the evaluation of non-binary and binary maps. Li et al. [34] propose to learn a data-driven metric using Convolutional Neural Network.

2.2 Saliency in 3D data

In this work we focus on 3D objects that are represented by their boundaries. The most common representation of these boundaries is a triangulated mesh. Below we discuss the few previous works in this area.

Lee et al. [5] defined a mesh saliency measure using a center-surround operator on the Gaussian-weighted mean curvatures. Gal and Cohen-Or [2] defined regional clusters on the mesh and estimated the saliency degree of each cluster as a linear combination of the cluster area and the curvature (either the Gaussian curvature or the maximal curvature). Shilane and Funkhouser [3] based surface distinctiveness on the similarity between a given surface and similar objects in its class. Chen et al. [35] investigated "Schelling points" on 3D surface. These points are feature points selected by people in a pure coordination game due to their salience. Wu et al. [36] proposed a novel approach for mesh saliency estimation considering both local contrast and global rarity, which is robust against noise. Song et al. [37] analyzed the spectral attributes of the log-Laplacian spectrum of a mesh. Recently, Tao et al. [38] presented a mesh saliency detection approach based on manifold ranking in a descriptor space.

We also look for region distinctness, however, unlike most prior approaches, which consider only local distinctness, we focus on global distinctness. Additionally, we consider the fact that visual forms may possess one or several centers of gravity about which the form is organized. Moreover, since extremities are considered salient by humans, we look for extreme vertices of meshes.

In this paper we also show how our work can be extended to detect saliency in point clouds. We are aware only of one work on point cloud saliency [39], which proposes an algorithm to cope with large point sets.

3 Detection of Regions of Interest

Given a surface, our goal is to compute its regions of interest. Since people are drawn to differences, we say that a region is *interesting* if it differs from other regions of the mesh. Therefore, we look for vertices that are distinct in their appearance.

In addition, in [40] it is found that *extremities* are considered salient by humans. Indeed, Chen et al. [35] state that tips of protrusions are salient. This is reinforced by observing the ground truth provided in [41]. Hence, we also look for extreme vertices of meshes.

Finally, our goal is to look for regions of interest, rather than for isolated vertices. This consideration follows the human tendency to group close items together. Therefore, we introduce *patch association*, which regards the regions near the foci of attention as more interesting than faraway regions. Hereafter, we present algorithms for realizing each of the above considerations: (1) vertex distinctness (2) shape extremities and (3) patch association. See Fig. 2.

3.1 Vertex Distinctness

We look for vertices whose geometry (i.e., appearance) is unique. This is done by computing, for each vertex, a

Fig. 2. Detection of regions of interest: algorithm outline.

descriptor that characterizes its shape and comparing the descriptors. A vertex is distinct if its descriptor is dissimilar to all other vertex descriptors of the mesh.

Vertex descriptor: We seek a descriptor that has good expressive power of the local shape geometry and is invariant to rigid transformations. Various local descriptors were recently proposed for shape-based retrieval [42], [43], [44], each with its benefits and drawbacks. We have examined some of them [45], [46], [47] and found that the best results were achieved using *spin images* [46], briefly described below.

The spin image is a 2D histogram that encodes the density of oriented points, in our case mesh vertices. Two cylindrical coordinates are defined for each vertex v with a normal n: the radial coordinate r, which is the perpendicular distance to the normal, and the elevation coordinate e, which is the distance to the tangent plane (see Fig. 3). A spin image is created for a vertex by quantizing e and r, creating bins, and counting the number of vertices in each bin. In our implementation, the geometric width of the bins, the *bin size*, is set to the median of the length of the mesh edges. We use a 16×16 histogram.

Dissimilarity measure: We seek a dissimilarity measure, which is robust to small changes in the mesh, such as noise or different triangulations. In addition, the computation should be fast enough, since a quadratic number of comparisons need to be performed.

We use the *diffusion distance*, which models the difference between two histograms as a temperature field and considers the diffusion process on the field [48]. The integration of a norm on the diffusion field over time is used as a dissimilarity measure between the histograms. For computational efficiency, a Gaussian pyramid is used to discretize the continuous diffusion process.

The diffusion distance $D(h_1, h_2)$ is defined as:

$$D(h_1, h_2) = \sum_{l=0}^{L} k(|d_l|), \tag{1}$$

where

$$d_0 = h_1 - h_2$$

$$d_l = [d_{l-1} * \phi(\sigma)] \downarrow_2, l = 1, \ldots, L$$

are different layers of the pyramid. The notation $\downarrow_2$ denotes half size down-sampling. L is the number of pyramid layers and σ is a constant standard deviation for the Gaussian filter

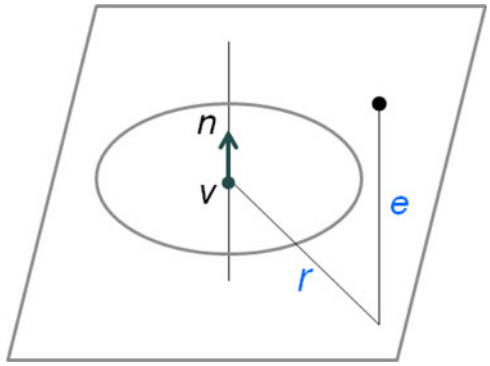


Fig. 3. Spin Image calculation: Two cylindrical coordinates are defined for each vertex v with a normal n: the radial coordinate r, which is the perpendicular distance to the normal, and the elevation coordinate e, which is the distance to the tangent plane.

ϕ (we use $L = 5$ and $\sigma = 0.5$). In our implementation $k(.)$ is the L_1 norm, which makes the diffusion distance a true metric. Note that for the L_1 norm calculation, the $2D$ histogram should be converted into a $1D$ vector.

Note that there are other well-known distances used in the literature. We experimented with L_2, χ^2, Jeffrey distance function, and the Earth Mover's Distance (EMD). We found that the first three are more prone to noise than the diffusion distance, since they are not cross-bin distances, whereas EMD is computationally more expensive.

Distinctness computation: Finally, we need to compute a distinctness value for each vertex, given the dissimilarity values calculated above.

1. SINGLE-SCALE COMPUTATION: As stated above, a vertex v_i is distinct when the distance between the descriptors $D(h(v_i), h(v_j))$ is high $\forall j$. This consideration, however, is insufficient, since the geodesic distances between the vertices are important as well. This is so, since similar vertices that are far away indicate a 3D texture. Thus, a vertex is distinct when the vertices similar to it are nearby and less distinct when the resembling vertices are far away. Hence, the dissimilarity measure should be proportional to the difference in appearance and inverse proportional to the geodesic distance.

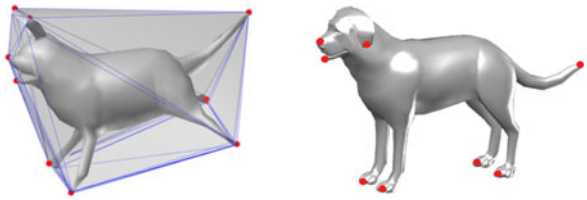
Let $GeodDist(v_i, v_j)$ be the geodesic distance between vertices v_i and v_j, normalized by the largest geodesic distance on the mesh. The geodesic distance is computed according to [49]. Inspired by [23], the dissimilarity measure between v_i and v_j is defined as:

$$d(v_i, v_j) = \frac{D(h(v_i), h(v_j))}{1 + c \cdot GeodDist(v_i, v_j)}, \tag{2}$$

where c = 3 in our implementation.

Vertex v_i is considered distinct when it is highly dissimilar to all other vertices. In practice, it suffices to consider the K most similar vertices, since if they are highly different from v_i, then clearly all the vertices are highly different from v_i as well. Therefore, for every vertex v_i, we search for the K most similar vertices $\{v_k\}_{k=1}^{K}$. Vertex v_i is distinct when $d\ (v_i, v_k)$ is high $\forall k \in [1, K]$. The single-scale distinctness value of vertex v_i is defined as:

$$\mathcal{D}(v_i) = 1 - \exp\left(-\frac{1}{K}\sum_{k=1}^{K} d(v_i, v_k)\right), \tag{3}$$



(a) The MDS-transformed mesh (b) Mesh extremities

Fig. 4. Computation of the shape extremities.

where K is 5 percent of the number of mesh vertices. We have chosen 5 percent empirically, since using $K > 5\%$ increases the computation time and has almost no impact on the result of Equation (3).

2. MULTI-SCALE COMPUTATION: Vertices that belong to 3D textures are likely to have similar vertices at multiple scales. Conversely, distinct vertices may have similar vertices at a few scales, but not at all of them. Therefore, we incorporate multiple scales to further decrease the importance of vertices that belong to 3D textures.

We simplify the given mesh of F faces to meshes having $\{\frac{F}{2}, \frac{F}{4}\}$ faces [50]. We then calculate the distinctness at these three scales $(F, \frac{F}{2}, \frac{F}{4})$.

The multi-scale distinctness value $\mathcal{D}$ is the average of the values of distinctness at all three scales, where the values at the different scales are mapped back to the input mesh. See Fig. 2a for the final distinctness map.

3.2 Shape Extremities

We aim at detecting the extremities of limb-like objects. Given an object, we need to first determine whether it has a limb-like structure, and then find its extremities, if need be (Fig. 2b). This is done in three steps, as follows.

1. MDS TRANSFORMATION: To determine the structure of an object and its extremities, we need to ignore the object's pose. We do it similarly to [51], by transforming the mesh using *multi-dimensional scaling (MDS)*, such that the Euclidean distances between points on the transformed mesh become similar to the geodesic distances between their corresponding points on the input mesh. Hence, the folded parts of the objects are "straightened" (Fig. 4a).

2. DETERMINING THE OBJECT'S STRUCTURE: To decide whether an object has a limb-like structure, we note that the volume of a "round" shape is similar to that of its convex hull, whereas the volume of a (straightened) limb-like object and that of its convex hull, differ. Therefore, we utilize a simple procedure, which works well in practice. We compute the volume of the object V_O and the volume of the convex hull of the MDS-transformed object V_{CH}. If $\frac{V_{CH}}{V_O} > 1.5$, we conclude that the object is limb-like. The constant 1.5 was established empirically.

3. DETECTING EXTREMITIES:Intuitively, an extreme vertex is a vertex that resides on the "tips" of the object. Specifically, we say that a vertex is extreme if it satisfies two conditions: It resides on the convex-hull of the MDS-transformed mesh and it is a local maximum of the sum of the geodesic distance functional [52]. The latter condition can be formally expressed as follows. Given a vertex v, let N_v be its set of neighboring vertices. The local condition that an extreme vertex v should satisfy is that $\forall v_n \in N_v$:

$$\sum_{v_j \in S} GeodDist(v, v_j) > \sum_{v_j \in S} GeodDist(v_n, v_j). \quad (4)$$

This definition derives an algorithm for computing the extreme vertices (Fig. 4). Given a mesh S, the algorithm first computes the convex hull of its MDS-transformed mesh and then finds among the vertices of the convex hull those that satisfy Equation (4).

3.3 Patch Association

Visual forms may possess one or several centers of gravity about which the form is organized. Therefore, regions that are close to the foci of attention should be more interesting than faraway regions.

We model this effect as follows. We define a fraction (20 percent in our implementation) of vertices with the highest distinctness values as *focus points*. Let $GeodFoci(v_i)$ be the geodesic distance between vertex v_i and its closest focus point, normalized to the range [0, 1]. Let $\mathcal{D}_{foci}(v_i)$ be the distinctness value of this focus point. The association of vertex v_i is defined as:

$$\mathcal{A}(v_i) = \mathcal{D}_{foci}(v_i) e^{-\frac{GeodFoci^2(v_i)}{2\sigma^2}}, \quad (5)$$

where $\sigma = 0.05$.

Similarly, an extreme vertex is considered a focus point. Therefore, for each mesh vertex v_i, we compute its geodesic distance to the closest extreme vertex $GeodExt(v_i)$, normalized to [0, 1]. The extremity of v_i is defined as:

$$E(v_i) = e^{-\frac{GeodExt^2(v_i)}{2\sigma^2}}. \quad (6)$$

Naturally, for non limb-like objects, $E(v_i) = 0 \; \forall i$.

Finally, we integrate the results obtained by the different phases of the algorithm (Fig. 2c). The degree of interest $\mathcal{I}(v_i)$ of vertex v_i is defined as the maximum of the distinctness and the extremity of the vertex, taking into account patch association:

$$\mathcal{I}(v_i) = \max\left(\frac{\mathcal{D}(v_i) + \mathcal{A}(v_i)}{2}, E(v_i)\right). \quad (7)$$

4 REGIONS OF INTEREST: RESULTS

4.1 Qualitative Evaluation

We ran our algorithm on a broad set of object categories, including animals (10 models), humans (6), creatures (7), sculptures (8), ancient artifacts (5), cars (4), other vehicles (7), furniture (8), tools (7) and miscellaneous accessories and instruments (17). The number of objects per class differs, depending on the object variance in the class.

Fig. 5 shows the results of our algorithm for a representative model per class. It can be seen that our algorithm usually detects the expected regions of interest. For example, for the dog, our algorithm finds the facial features, the feet, and the tail interesting, where the facial features are the most interesting. Similarly, for the chess set, the chess pieces are more interesting than the board, yet the unique

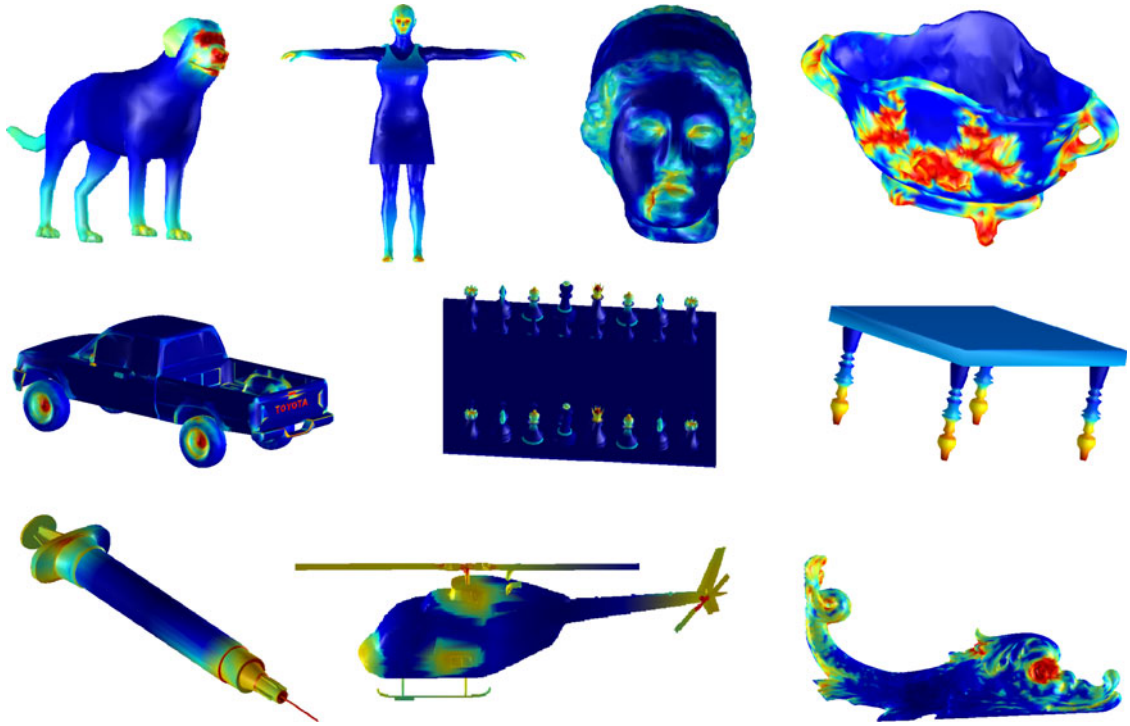


Fig. 5. Regions of interest of representative objects, as computed by our algorithm.

pieces, like the kings and the queens, are more interesting than the regular pawns, since there are many of them.

We also compare our results to works that explore saliency or class-distinctiveness of surfaces (Fig. 6). Since their implementations are unavailable, we ran our code on the same models given in these papers and show the results side by side. Fig. 6a compares a couple of results from [5] with ours. Our algorithm is less influenced by local changes of the curvature, when they happen frequently. For instance, the long body of the dragon is not considered interesting by our algorithm, whereas many small regions on it are salient according to [5], since the local changes are large. In addition, the lack of patch association in [5] is noticeable. Conversely, our algorithm detects large regions, rather than small, more isolated ones. For instance, the whole dinosaur head, and not only its facial features, are interesting (yet, the facial features are more interesting).

Fig. 6b compares our results with those of [2]. While in [2], the saliency is more distributed over the surface, as it detects local changes of curvature, our results are more focused on several regions of interest. For instance, the distinct facial features are emphasized in all three examples. Moreover, the shape extremity consideration is noticeable in the animals' feet and hump.

Fig. 6c compares our results with those of [3]. It is important to note that the goals are different. There, the goal is to find regions that distinguish a shape from objects in a different class, while we aim at locating the regions of interest regardless of the class it belongs to. Consequentially, while in [3] the whole head of the horse is marked, we detect mostly the facial features, and similarly for the horse's legs. In the bunny, both algorithms mark the ears as important. However, while the bunny's tail was found to be interesting by our algorithm, it is not distinctive according to [3].

Fig. 6d compares our results with those of [37]. Both algorithms detect the facial features and limbs of the animals. However, unlike [37], we detect the importance of the camel's hump. Moreover, we treat all four dog's legs similarly and do not prefer only the rear ones as in [37].

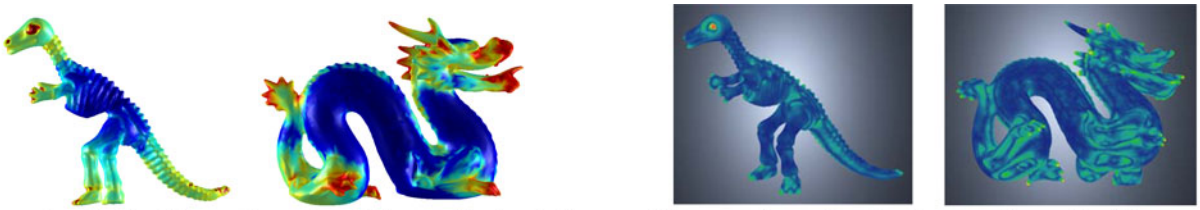
Fig. 6e compares our results with those of [36]. Our algorithm is more specific, detecting the specific facial features on both statues.

Fig. 7 demonstrates the robustness of our method to various mesh manipulations. Simplifying the model from 100K faces to 10K faces has almost no impact on the results. Similarly, remeshing does not change the resulting regions of interest significantly. Finally, to demonstrate the robustness of our method to noise, we added a synthetic Gaussian noise of zero mean and a variance of 0.5 percent of the diagonal of the model's bounding box. Despite the visible distortion of the noisy model, the results are very stable. We would like to mention that semantic manipulation, such as removing parts of the model, might change the resulting regions of interest significantly.

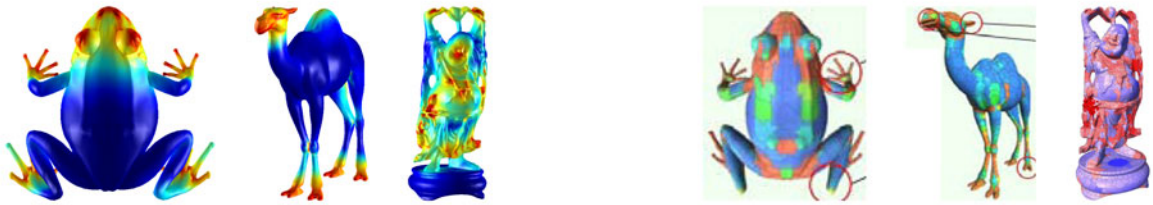
4.2 Quantitative Evaluation

We evaluated our algorithm on the *Benchmark for 3D Interest Point Detection Algorithms* [41]. This benchmark uses points of interest on 3D objects, which have been marked by human subjects, to provide a quantitative evaluation for interest point detection algorithms. The data set contains 43 models for which at least 16 human subjects have marked the interest points. We compare our algorithm to the six techniques of the benchmark: mesh saliency [5], salient points defined by [53], 3D-Harris [54], 3D-SIFT [55], scale-dependent corners [56], and Heat Kernel Signature (HKS) [57].

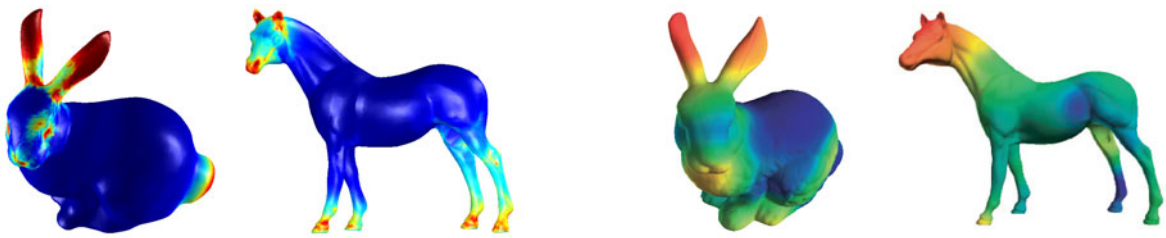
In this benchmark, two criteria were being used while constructing the ground truth: the radius of an interest



(a) Comparison to [5]: Our results are less influenced by local changes of the curvature, when they happen frequently. Moreover, thanks to patch association, our algorithm detects large salient regions.



(b) Comparison to [2]: The regions of interest detected by our algorithm are more focused on the head, feet, and hump.



(c) Comparison to [3]: Notice the differences in the facial features, the legs and the tail.

(d) Comparison to [37] (right). Our algorithm detects the hump of the camel and all four legs of the dog.

(e) Comparison to [36]: Our algorithm is more specific, detecting the specific facial features on both statues.

Fig. 6. We compare our results (left) with five state-of-the-art methods (right). Since their implementations are unavailable, we ran our code on the same models given in these papers and show the results side by side. Our approach considers global distinctness, which allows us to detect specific facial feature ignoring repeating patterns, (e.g., the dragon (a) and gargoyle (e)). Our special algorithm for extremity detection allows us to treat all the limbs equally (without preferring rear legs to the front ones (d)). Finally, our patch association allows us to detect large regions, rather than small, more isolated ones (e.g., the frog and the camel (b)).

region σ and the number of users n, who marked a point within this interest region. Interest points whose geodesic distances to each other were less than 2σ were grouped together. If the number of points in the set was less than n, this set was discarded. Otherwise, a representative from the set was selected and set as a ground truth interest point.

Three measures were used to evaluate the performance of the different algorithms: the False Negative Error (FNE), the False Positive Error (FPE), and the Weighted Miss Error (WME). Let us denote the set of ground truth points for model M as $\mathcal{G}(n,\sigma)$ and the set of interest points detected by an algorithm as $\mathcal{A}$. For an interest point g in a set $\mathcal{G}$, a geodesic neighborhood of radius r is defined as $\mathcal{C}_r(g) = \{p \in M \mid d(g,p) \leq r\}$, where $d(g,p)$ corresponds to the geodesic distance between points g and p. The parameter r controls the localization error tolerance. A point g is considered to be "correctly detected" if there exists a detected point $a \in \mathcal{A}$ in $\mathcal{C}_r(g)$, such that a is not closer to any other point in $\mathcal{G}$.

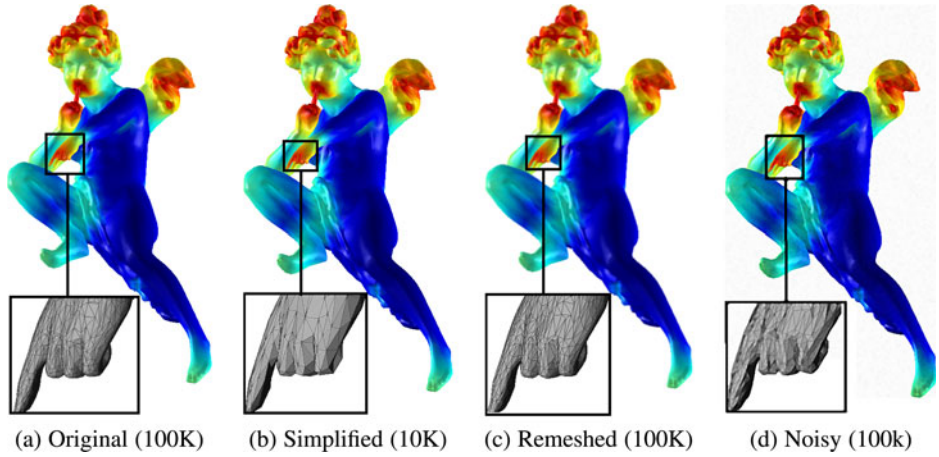


Fig. 7. Robustness. Our method is robust to various mesh manipulations, including remeshing, simplification and adding noise.

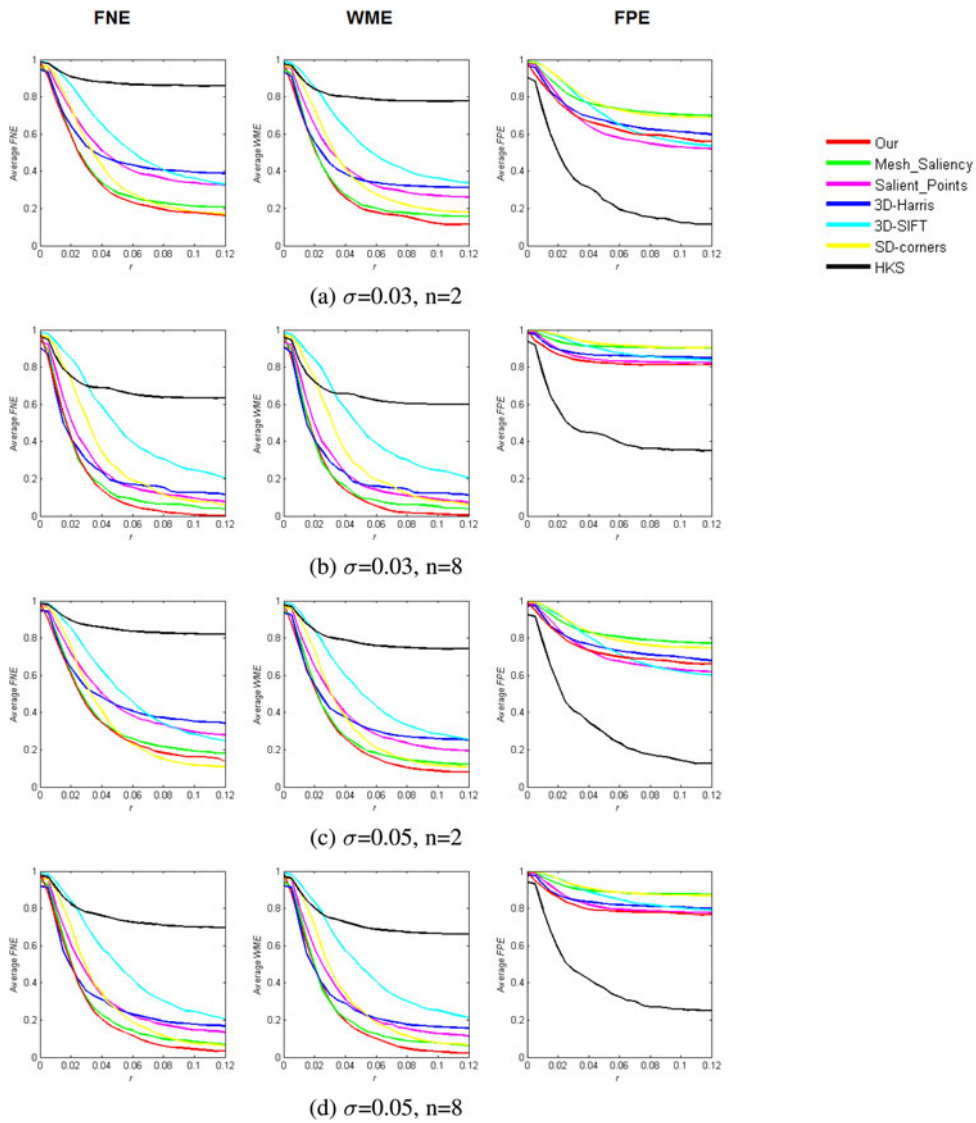


Fig. 8. Evaluation on the *Benchmark for 3D Interest Point Detection Algorithms* [41]: Our algorithm's performance graph is colored in red. The lower the graph, the better the result. The left and the middle columns show that the FNE and the WME drop faster with our algorithm, compared to the other methods. All the algorithms, except for HKS, find more interest points than the human users. For the FPE measure (right column), almost all the methods, including ours, produce similar results (except for HKS, since it detects very few key points).

Denoting the number of correctly detected points in $\mathcal{G}$ as N_C and the number of points in $\mathcal{G}$ as N_G, the false negative error rate at localization error tolerance r is defined as:

$$FNE(r) = 1 - \frac{N_C}{N_G}. \quad (8)$$

To calculate the false positive error rate, each correctly detected point $g \in \mathcal{G}$ corresponds to a unique a, the closest point to g among the points in $\mathcal{A}$. All the points in $\mathcal{A}$, excluding a correspondence in $\mathcal{G}$, are considered as false positives. Then, the number of false positives, denoted as N_F , is set to $N_F = N_A - N_C$, where N_A is the number of interest points detected by the algorithm. The false positive error rate at localization error tolerance r is then defined as:

$$FPE(r) = \frac{N_F}{N_A}. \quad (9)$$

To incorporate the prominence of an interest point into the evaluation, another miss error measure is defined, denoted as the Weighted Miss Error. Suppose that within a geodesic neighborhood of radius r around the ground truth point $g_i \in \mathcal{G}$, n_i subjects have marked an interest point. The WME is defined as

$$WME(r) = 1 - \frac{1}{\sum_{i=1}^{N_G} n_i} \sum_{i=1}^{N_G} \delta_i n_i, \quad (10)$$

where

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

An algorithm gets a low WME if the points it detects were voted by many human subjects. Hence, this error intends to measure the ability of an algorithm to detect the semantically significant interest points.

To extract key points from our regions of interest we follow [41], where candidate interest points are selected from the local maxima of the interest values. A vertex is marked as a local maximum if its interest value is higher than those of all its neighboring vertices. Then, the final interest points are those with a interest value higher than the average interest value.

Fig. 8 shows the performance of the six methods tested in [41] and ours, with respect to the localization error tolerance r. The lower the graphs, the better the results. With an ideal interest point detection algorithm, the errors are expected to drop very quickly with respect to r. A rapid drop in FNE means that the algorithm finds the interest points with a low localization error, while a rapid drop in FPE indicates that the algorithm does not return excessive interest points.

As the graphs show (the left and the middle columns), for most of the settings of σ and n, the FNE and WME drop faster with our algorithm, compared to the other methods. As the localization error tolerance increases, our method detects more ground-truth interest points than the competing methods, having lower FNE and WME. Note that all the algorithms, except for HKS [57], mark more interest points than the human users. For the FPE measure, almost all the methods, including ours, produce similar results. Only HKS outperforms them significantly, since it detects very few key points.

4.3 Complexity Analysis

The complexity of the distinctness computation depends on the computation of the K-nearest neighbors and on the geodesic distance computation, which is $O(n^2 \log n)$, where n is the number of vertices. Finding the extremities depends on the MDS computation, which is $O(n^2)$. The other operations are convex hull construction $O(n \log n)$ and volume computation $O(n)$.

In the actual implementation, the time-consuming steps are performed on simplified meshes. The running time, for instance, for the Buddha that has 540,000 vertices, is 102 seconds. The experiments were preformed on an Intel Xeon E5630 2.53 GHz CPU with 8 GB memory, using the same set of configuration parameters.

5 Region of Interest Detection for Point Clouds

Due to the fast development of 3D data acquisition techniques, 3D point clouds have become widely spread. This section discusses the adaptation of our algorithm to operate on point clouds. In particular we show how to adjust the algorithm's building blocks; vertex distinctness, shape extremities and patch association, to this data.

These three building blocks require geodesic distances between points. Since point clouds lack connectivity information, we approximate the distances as follows. First, we build a sparse graph, in which edges are created between each point and its K Euclidian nearest neighbors (we use K=8). The weight of each edge is the Euclidian distance between the corresponding points. Then, we use Johnsons's search algorithm for sparse graphs to create a distance matrix [58].

For vertex distinctness computation we use the Spin Image descriptor. Edge lengths are needed for determining the bin size, which is essential for the descriptor calculation. Thus, we use the median of the edge lengths in the sparse graph described above.

Shape extremities calculation employs geodesic distances between pair of points. We use approximate geodesic distances instead, as follows. First, we use the distance as input to multi-dimensional scaling, which "straightens" the folded parts of the objects. Then, the approximated geodesic distances are used to determine whether a point is an extremity (Equation (4)).

Finally, for the patch association, the approximated geodesic distances are used to detect the points that are close to the foci of attention and consider them more interesting than faraway points. The approximated geodesic distances are used both in Equation (5) and in Equation (6). Fig. 9 demonstrates the results of each stage of the algorithm, when applied to the point cloud representation of the object from Fig. 2, (i.e., considering only the vertices of the model).

To assess the the quality of the algorithm applied to point clouds, we performed the same quantitative evaluation, as described in Section 4. The point clouds were obtained by distributing random points on the surface of each model, where the number of points on each mesh face is proportional to the area of the face. It can be seen in Fig. 10 that for all three measures (FNE, WME and FPE), similar results for meshes and for

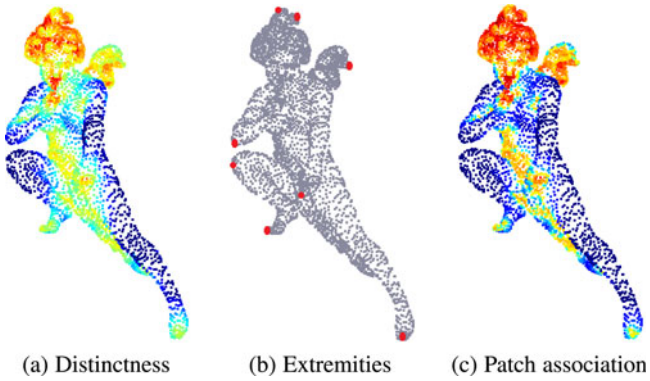


Fig. 9. Detection of regions of interest on point clouds.

point clouds are obtained. The results for other sets of parameters ($\sigma \in [0.03, 0.05]$, $n \in [2, 8]$) are similar as well.

6 Viewpoint Selection

Given a surface, our aim is to automatically determine the set of the most informative views, which jointly describe the surface well. The key idea of the algorithm is to maximize the area of the viewed regions of interest. The algorithm consists of four steps, outlined in Fig. 11 and explained thereafter.

Initially, we generate a set of candidate viewpoints P_s. This is done by uniformly sampling a sphere that bounds the object [59]. We use a sphere whose radius is twice as large as that of the tight bounding sphere. In practice, we sample the sphere with 200 points.

Next, we evaluate the quality of each viewpoint $p_i \in P_s$ according to the regions of interest it views:

$$\bar{I}(p_i) = \sum_{v_j \in V} \mathcal{I}(v_j) w_i(v_j), \tag{11}$$

where V is the set of vertices of mesh S, $\mathcal{I}(v_j)$ is defined in Equation (7), and $w_i(v_j)$ is a weight defined as follows. The weight should be high if the area the region occupies in the surface's projection is large. Let β_{ij} be the angle between the surface normal at vertex v_j and the viewing direction $\overrightarrow{p_i - v_j}$. If v_j is visible from p_i, then $w_i(v_j) = \sqrt{\cos \beta_{ij}}$, otherwise $w_i(v_j) = 0$.

In Stage 3, a few descriptive viewpoints are selected. Since it is expected that points close to each other will view similar regions, it is insufficient to simply choose the best ones. Instead, we select viewpoints that collectively describe different regions of interest of the mesh.

1. Generate candidate viewpoints.
2. Compute the viewed ROI from each candidate.

Iterate on:

3. Select an informative viewpoint.
4. Improve the selection in the local neighborhood.

Fig. 11. Viewpoint selection algorithm.

Specifically, the first viewpoint selected is the one having the maximal $\bar{I}(p_i)$. Then, we iteratively add a new viewpoint, which jointly with the previously-selected viewpoints, maximize the viewed regions of interest. Let $w_{max}(v_j)$ be the highest weight assigned to v_j by one of the viewpoints selected so far. And, let $\delta(p_i)$ be the added degree of interest contributed by p_i, defined as:

$$\delta(p_i) = \sum_{v_j \in V} \mathcal{I}(v_j) \max\big(w_i(v_j) - w_{max}(v_j), 0\big). \tag{12}$$

We add to the set of the selected viewpoints the candidate viewpoint that maximizes δ.

The number of viewpoints is dynamic and depends on the object's geometry. We keep adding viewpoints to the set until one of the following conditions is satisfied. First, the computed interest of the viewed vertices is at least 60 percent of the total computed interest over the whole mesh: $0.6 \cdot \sum_v \mathcal{I}(v) < \bar{I}(p_0) + \sum_{p_i \in set} \delta(p_i)$. Second, no new viewpoint p adds large-enough viewed region of interest: $\delta(p) \leq 0.1 \cdot \sum_{v_j \in V} \mathcal{I}(v_j)$.

When a new viewpoint is selected, Stage 4 attempts to refine its location by searching its neighborhood for a better viewpoint. The neighborhood's size is the initial distance between the sampled candidate viewpoints. This neighborhood is uniformly sampled, every three degrees. The point that maximizes the weighted viewed region of interest (Equations (11), (12)) is chosen.

In order to avoid generating two symmetric views, as a pre-processing step we apply an algorithm that detects reflective symmetries [60]. The regions of interest on one side of the symmetric plane are zeroed.

7 Viewpoint Selection: Results

We ran our algorithm on the 79 meshes from Section 4. For 42 models our algorithm generated a single viewpoint, for 32 models two viewpoints, and for five models three viewpoints.

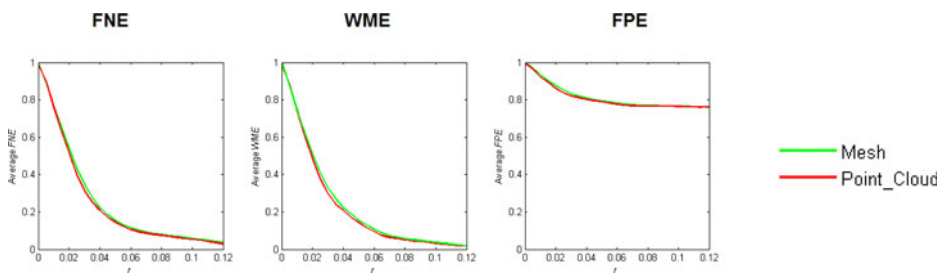


Fig. 10. Region of interest detection on a mesh versus a point cloud: Quantitative evaluation based on [41] compares the results of our technique for meshes and for point clouds. The Evaluation was performed on the *Benchmark for 3D Interest Point Detection Algorithms* [41] with parameters σ=0.05, n=8. For all three measures we get similar results for meshes and for point clouds.

Fig. 12. The most informative viewpoints. For each model, the selected viewpoints are shown in descending order of informativeness, from left to right.

Fig. 13. The most informative viewpoint.

Fig. 12 shows all the viewpoints generated for some models, each drawn in descending order, from the most informative view to the least informative view. For instance, our algorithm generated three viewpoints for the Buddha, which are indeed appropriate for this case, since every view reveals additional interesting details. Fig. 13 shows only the most informative view for additional models. For the piano and the pickup truck, our algorithm resulted in a single view. The most informative viewpoints for all the meshes can be found in the supplementary material, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPAMI.2016.2522437.

Comparison to other methods. The results of our algorithm competes favorably with those of the state-of-the-art methods. Fig. 14 compares our results with [8]'s, where saliency is based on class distinctiveness. In [8], the number of viewpoints is static and set to 4. We compare only the first viewpoint. As can be seen, our selected viewpoints are often more "natural".

Fig. 15 compares our most informative viewpoint with those of [5], [9], [10]. Our selected viewpoint of David is the best according to our user study (described next). Moreover, it is close to the classic three-quarter frontal view (i.e., the oblique view between frontal view and profile view), which is considered good in human vision.

Some recent approaches for viewpoint selection incorporate user input, either as an interactive machine learning system [61] or as a high level factor [62]. Our approach is fully automatic. Still, as demonstrated in Figs. 16, 17 our results are comparable and often outperform those presented in [61] and [62].

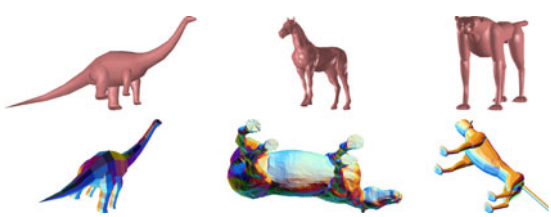


Fig. 14. Comparison of our results (top) with some of the results of [8] (bottom).

Evaluation using a user study: Since there is no available ground truth for the best viewpoints, we conducted a user study. The goal was to learn which views are considered the most informative.

For each of our 79 models, we produced 12 images, each taken from a different viewpoint. The viewpoints were created by uniformly sampling the bounding sphere. We decided to use 12 images as a compromise between the accuracy of the survey (requiring a large number of viewpoints) and our wish to avoid overloading the evaluators (requiring a small number of viewpoints). We asked the evaluators to mark the most informative views of the object—those that let the observer understand its shape. Each screen included 12 randomly-ordered viewpoints of a single object. The number of informative views that could be marked was unlimited. We collected results for over two months, from 195 evaluators, 57 percent men and 43 percent women, at the age 15-65. We obtained 68 evaluations per model on average.

Fig. 18a shows a typical distribution of the evaluation. In this example, there are three views considered informative by the evaluators. Therefore, rather than defining our ground truth to be a single view, we define it as a set of views, consisting of the highest-ranked views before the largest decrease in the histogram. Fig. 18c shows the ground truth—a set of three informative views. Our best view is given in Fig. 18b. All the viewpoint results and their comparison to the user study can be found in the supplementary material, available online.

To assess the results of our algorithm, we compared the view(s) selected by our algorithm to the ground truth. Since the evaluators could choose between 12 viewpoints only, our result is considered correct if it is closer (angularly) to a view of the ground truth than to any other view. For 75 out of 79 models (94.9 percent), the most informative view selected by our algorithm matched the ground truth. If we take into account also the other informative views generated by our algorithm, the views of 78 out of 79 models (98.7 percent)

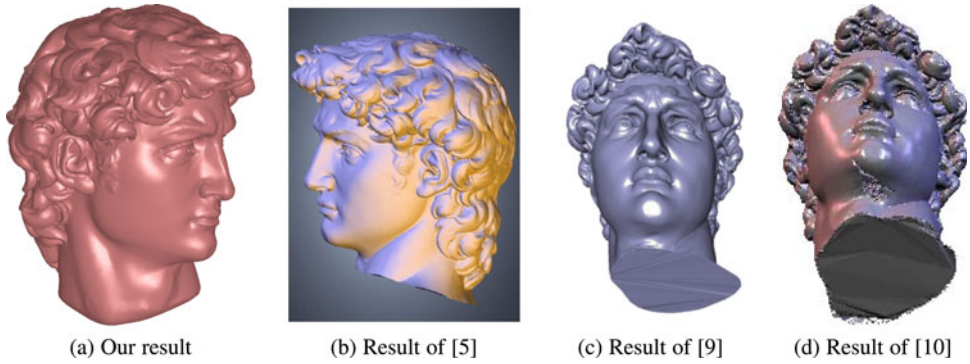


Fig. 15. Comparison of the best views of David. Our view is close to the classic three-quarter frontal view. (Images are taken "as is" from the corresponding papers.)

Fig. 16. Comparison of our results (top) with some of the results of [62] (bottom). It can be seen that our results are comparable and even, in some cases (David's head), slightly better, even though user input is provided in [62].

matched the ground truth. When operating on point clouds the success rates are 93.7 and 97.5 percent, respectively.

Complexity analysis: The complexity of the viewpoint selection algorithm depends on the computation of the regions of interest, which is $O(n^2 \log n)$, where n is the number of mesh vertices. Uniform sampling of the sphere with k samples (candidates) is done in $O(k^3)$. For each candidate we compute its visible vertices using HPR operator ($O(nlogn)$), leading to $O(knlogn)$. The rest of the iterative algorithm has the complexity of $O(k^2 n)$. The total complexity is $O(k^3 + k^2 n + \ kn^2 \log n)$, and since $k \ll n$, we get $O(kn^2 \log n)$.

Limitations: The failure case of our algorithm is the tank in Fig. 19, for which our algorithm generated a single view. This failure can be explained by the lack of high level factors. People tend to prefer "natural" positioning of objects. Therefore, our evaluators preferred a side view. However, most of the details of the tank (turret, cannon) are on the top and therefore our algorithm computed a top view.

Similarly, for the lamp in Fig. 19, our algorithm chose the view of the ground truth as the second-most informative

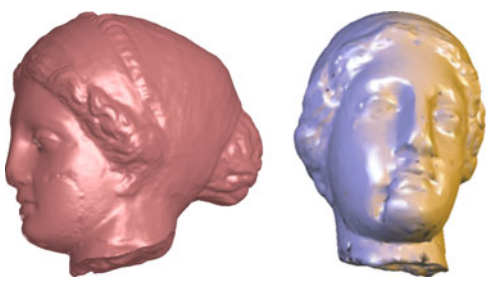


Fig. 17. Comparison to [61]. According to our user study for Igea, our best (left) view matches the second best view picked by the users, which is only slightly worse than the best view matching the view picked by [61]. (right)

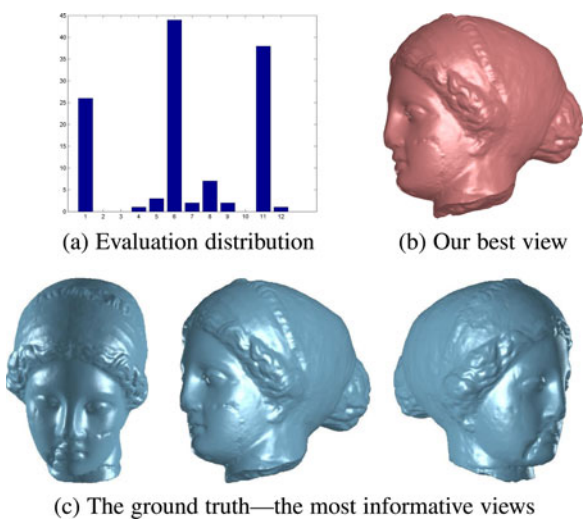


Fig. 18. Informative views. Three views are considered informative by the evaluators (a). Our computed most-informative view (b) is one of those that belong to the ground-truth (c).

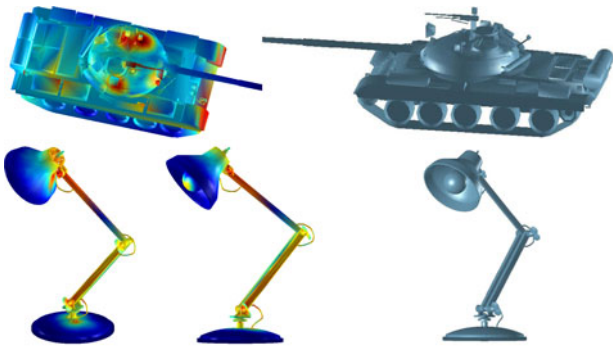


Fig. 19. Limitations. High-level factors are not considered by our algorithm. Therefore, it may choose views with many details, rather than more "natural" views.

view. As the most informative view our algorithm chose a back view, which contains many distinct details, such as wires and screws. In contrast, the evaluators preferred the front view, where the light bulb is visible.

8 Conclusion

This paper has studied the detection of surface regions of interest. We discuss two considerations—vertex distinctness and patch association—and methods that realize them. We propose an algorithm that operates on surfaces represented by meshes. We also show how our technique can be adopted to handle point clouds, which have been commonly used in recent years. The regions of interest can benefit many computer vision and geometry processing applications. We explore one such application—the selection of the most informative viewpoints of objects. We show state-of-the-art results, which are reinforced by a user study.

In the future, high-level factors can be added to our algorithm. A notable example is the class distinctness of [3], which detects regions that distinguish a shape from shapes in other classes.

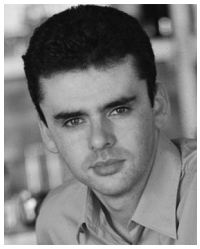
Acknowledgments

The models were generously provided by the AIM@SHAPE Shape Repository, The Technion's CG&M Lab, Stanford 3D Scanning Repository and the Princeton Benchmark. This research was funded in part by the Intel Collaborative Research Institute for Computational Intelligence (ICRI-CI), Omek Consortium under the MAGNET program of the ministry of economy, the Technion Funds for Security Research, and the Ollendorff foundation.

References

[1] J. Lee, B. Moghaddam, H. Pfister, and R. Machiraju, "Finding optimal views for 3D face shape modeling," in *Proc. IEEE 6th Int. Conf. Autom. Face Gesture Recog.*, 2004, pp. 31–36.
[2] R. Gal and D. Cohen-Or, "Salient geometric features for partial shape matching and similarity," *ACM Trans. Graph.*, vol. 25, no. 1, p. 150, 2006.
[3] P. Shilane and T. Funkhouser, "Distinctive regions of 3D surfaces," *ACM Trans. Graph.*, vol. 26, no. 2, p. 7, 2007.
[4] Y. Yang, T. Lu, and J. Lin, "Saliency regions for 3D mesh abstraction," in *Proc. 10th Pacific Rim Conf. Multimedia: Adv. Multimedia Inf. Process.*, 2009, pp. 292–299.
[5] C. Lee, A. Varshney, and D. Jacobs, "Mesh saliency," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 659–666, 2005.
[6] N. Gelfand, N. Mitra, L. Guibas, and H. Pottmann, "Robust global registration," in *Proc. 3rd Eurograph. Symp. Geometry Process.*, 2005, p. 197.
[7] X. Li and I. Guskov, "Multi-scale features for approximate alignment of point-based surfaces," in *Proc. 3rd Eurograph. Symp. Geometry Process.*, 2005, p. 217.
[8] H. Laga, "Semantics-driven approach for automatic selection of best views of 3D shapes," in *Proc. 3rd Eurograph. Conf. 3D Object Retrieval*, 2010, pp. 15–22.
[9] M. Mortara and M. Spagnuolo, "Semantics-driven best view of 3D shapes," *Comput. Graph.*, vol. 33, no. 3, pp. 280–290, 2009.
[10] H. Yamauchi, W. Saleem, S. Yoshizawa, Z. Karni, A. Belyaev, and H.-P. Seidel, "Towards stable and salient multi-view representation of 3D shapes," in *SMI*, 2006, p. 40.
[11] P. Hall and M. Owen, "Simple canonical views," in *Proc. Brit. Mach. Vis. Conf.*, 2005, vol. 1, pp. 7–16.
[12] F. Mokhtarian and S. Abbasi, "Automatic selection of optimal views in multi-view object recognition," in *Proc. Brit. Mach. Vis. Conf.*, 2000, pp. 272–281.
[13] G. Leifman, E. Shtrom, and A. Tal, "Surface regions of interest for viewpoint selection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2012, pp. 414–421.
[14] A. Treisman and G. Gelade, "A feature-integration theory of attention," *Cognitive Psychol.*, vol. 12, no. 1, pp. 97–136, 1980.
[15] C. Koch and S. Ullman, "Shifts in selective visual attention: Towards the underlying neural circuitry," in *Matters of Intelligence*. New York, NY, USA: Springer, 1987, pp. 115–141.
[16] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 11, pp. 1254–1259, Nov. 1998.
[17] D. Walther and C. Koch, "Modeling attention to salient proto-objects," *Neural Netw.*, vol. 19, no. 9, pp. 1395–1407, 2006.
[18] N. Bruce and J. Tsotsos, "Saliency based on information maximization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2006, vol. 18, p. 155.
[19] J. Harel, C. Koch, and P. Perona, "Graph-based visual saliency," in *Proc. Adv. Neural Inf. Process. Syst.*, 2007, vol. 19, p. 545.
[20] X. Hou and L. Zhang, "Saliency detection: A spectral residual approach," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2007, pp. 1–8.
[21] T. Judd, K. Ehinger, F. Durand, and A. Torralba, "Learning to predict where humans look," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2009, pp. 2106–2113.
[22] R. Margolin, A. Tal, and L. Zelnik-Manor, "What makes a patch distinct?" in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2013, pp. 1139–1146.
[23] S. Goferman, L. Zelnik-Manor, and A. Tal, "Context-aware saliency detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 10, pp. 1915–1926, Oct. 2012.
[24] A. Borji and L. Itti, "State-of-the-art in visual attention modeling," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 185–207, Jan. 2013.
[25] W. Kim, C. Jung, and C. Kim, "Spatiotemporal saliency detection and its applications in static and dynamic scenes," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 4, pp. 446–456, Apr. 2011.
[26] V. Mahadevan and N. Vasconcelos, "Spatiotemporal saliency in dynamic scenes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 1, pp. 171–177, Jan. 2010.
[27] S. Mathe and C. Sminchisescu, "Dynamic eye movement datasets and learnt saliency models for visual action recognition," in *Proc. Eur. Conf. Comput. Vis.*, 2012, pp. 842–856.
[28] H. J. Seo and P. Milanfar, "Static and space-time visual saliency detection by self-resemblance," *J. Vis.*, vol. 9, no. 12, p. 15, 2009.
[29] X. Cui, Q. Liu, and D. Metaxas, "Temporal spectral residual: Fast motion saliency detection," in *Proc. 17th ACM Int. Conf. Multimedia*, 2009, pp. 617–620.
[30] D. Rudoy, D. B. Goldman, E. Shechtman, and L. Zelnik-Manor, "Learning video saliency from human gaze using candidate selection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2013, pp. 1147–1154.
[31] F. Zhou, S. Kang, and M. Cohen, "Time-mapping using space-time saliency," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2014, pp. 3358–3365.
[32] X. Huang, C. Shen, X. Boix, and Q. Zhao, "Salicon: Reducing the semantic gap in saliency prediction by adapting deep neural networks," in *Proc. Int. Conf. Comput. Vision*, 2015, pp. 262–270.

[33] R. Margolin, L. Zelnik-Manor, and A. Tal, "How to evaluate foreground maps," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2014, pp. 248–255.
[34] J. Li, C. Xia, Y. Song, S. Fang, and X. Chen, "A data-driven metric for comprehensive evaluation of saliency models," in *Proc. IEEE Int. Conf. Comput. Vision*, 2015, pp. 190–198.
[35] X. Chen, A. Saparov, B. Pang, and T. Funkhouser, "Schelling points on 3D surface meshes," *ACM Trans. Graph.*, vol. 31, no. 4, p. 29, 2012.
[36] J. Wu, X. Shen, W. Zhu, and L. Liu, "Mesh saliency with global rarity," *Graphical Models*, vol. 75, no. 5, pp. 255–264, 2013.
[37] R. Song, Y. Liu, R. R. Martin, and P. L. Rosin, "Mesh saliency via spectral processing," *ACM Trans. Graph.*, vol. 33, no. 1, p. 6, 2014.
[38] P. Tao, J. Cao, S. Li, X. Liu, and L. Liu, "Mesh saliency via ranking unsalient patches in a descriptor space," *Comput. Graph.*, vol. 46, pp. 264–274, 2015.
[39] E. Shtrom, G. Leifman, and A. Tal, "Saliency detection in large point sets," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2013, pp. 3591–3598.
[40] Y. Kim, A. Varshney, D. Jacobs, and F. Guimbretière, "Mesh saliency and human eye fixations," *ACM Trans. Appl. Perception*, vol. 7, no. 2, pp. 1–13, 2010.
[41] H. Dutagaci, C. Cheung, and A. Godil, "Evaluation of 3D interest point detection techniques via human-generated ground truth," *Visual Comput.*, vol. 28, pp. 901–917, 2012.
[42] B. Bustos, D. Keim, D. Saupe, T. Schreck, and D. Vranić, "Feature-based similarity search in 3D object databases," *ACM Comput. Surveys*, vol. 37, no. 4, pp. 345–387, 2005.
[43] J. Tangelder and R. Veltkamp, "A survey of content based 3D shape retrieval methods," *Multimedia Tools Appl.*, vol. 39, no. 3, pp. 441–471, 2008.
[44] O. van Kaick, H. Zhang, G. Hamarneh, and D. Cohen-Or, "A survey on shape correspondence," *Comput. Graph. Forum*, vol. 30, no. 6, pp. 1681–1707, 2011.
[45] A. Bobenko and P. Schröder, "Discrete Willmore flow," in *Proc. 3rd Eurograph. Symp. Geometry Process.*, 2005, pp. 101–110.
[46] A. Johnson and M. Hebert, "Using spin images for efficient object recognition in cluttered 3d scenes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 5, pp. 433–449, May 1999.
[47] J. Koenderink and A. van Doorn, "Surface shape and curvature scales," *Image Vision Comput.*, vol. 10, pp. 557–565, 1992.
[48] H. Ling and K. Okada, "Diffusion distance for histogram comparison," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2006, vol. 1, pp. 246–253.
[49] R. Kimmel and J. A. Sethian, "Computing geodesic paths on manifolds," *Proc. Nat. Acad. Sci. USA*, vol. 95, pp. 8431–8435, 1998.
[50] M. Garland and P. Heckbert, "Surface simplification using quadric error metrics," in *Proc. 24th Annu. Conf. Comput. Graph. Interactive Techn.*, 1997, pp. 209–216.
[51] A. Elad and R. Kimmel, "On bending invariant signatures for surfaces," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 10, pp. 1285–1295, Oct. 2003.
[52] S. Katz, G. Leifman, and A. Tal, "Mesh segmentation using feature point and core extraction," *Visual Comput.*, vol. 21, no. 8, pp. 649–658, 2005.
[53] U. Castellani, M. Cristani, S. Fantoni, and V. Murino, "Sparse points matching by combining 3d mesh saliency with statistical descriptors," *Comput. Graph. Forum*, vol. 27, no. 2, pp. 643–652, 2008.
[54] I. Pratikakis, M. Spagnuolo, T. Theoharis, and R. Veltkamp, "A robust 3d interest points detector based on Harris operator," in *Proc. Eurograph. Workshop 3D Object Retrieval*, 2010, pp. 7–14.
[55] A. Godil and A. I. Wagan, "Salient local 3d features for 3d shape retrieval," *Proc. IS&T/SPIE Electron. Imaging*, vol. 7864, pp. 78640S-1–78640S-8, 2011.
[56] J. Novatnack and K. Nishino, "Scale-dependent 3d geometric features," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2007, pp. 1–8.
[57] J. Sun, M. Ovsjanikov, and L. Guibas, "A concise and provably informative multi-scale signature based on heat diffusion," *Comput. Graph. Forum*, vol. 28, no. 5, pp. 1383–1392, 2009.
[58] D. B. Johnson, "Efficient algorithms for shortest paths in sparse networks," *J. ACM*, vol. 24, no. 1, pp. 1–13, 1977.
[59] J. Mitchell, "Discrete uniform sampling of rotation groups using orthogonal images," *SIAM J. Sci. Comput.*, vol. 30, pp. 525–547, 2007.
[60] M. Kazhdan, B. Chazelle, D. Dobkin, A. Finkelstein, and T. Funkhouser, "A reflective symmetry descriptor," in *Proc. 7th Eur. Conf. Comput. Vis.*, 2002, pp. 642–656.
[61] T. Vieira, A. Bordignon, A. Peixoto, G. Tavares, H. Lopes, L. Velho, and T. Lewiner, "Learning good views through intelligent galleries," *Comput. Graph. Forum*, vol. 28, no. 2, pp. 717–726, 2009.
[62] A. Secord, J. Lu, A. Finkelstein, M. Singh, and A. Nealen, "Perceptual models of viewpoint preference," *ACM Trans. Graph.*, vol. 30, no. 5, p. 109, 2011.



George Leifman received the BSc degree (Summa cum Laude) in computer engineering in 2001 and the MSc degree in electrical engineering in 2003 both from the Technion. He received the PhD degree in electrical engineering from Technion. He is currently a postdoctoral fellow at the Media Lab, MIT.

Elizabeth Shtrom received the BSc (Summa cum Laude) degree in computer science from the Technion in 2008. After graduating, she was an engineer in the IBM search development solutions team for three years. During this period, she held various development positions, starting from user interface, and ending with back-end development. During her last year at work, she started the MSc (Cum Laude) degree in electrical engineering in the Technion and received the degree in 2013. Her research included saliency detection in 3D surfaces and point clouds and saliency-based summarization techniques. Starting from 2014, she is a software engineer in Google.

Ayellet Tal received the BSc degree (Summa cum Laude) in mathematics and computer science and the MSc degree (Summa cum Laude) in computer science from Tel-Aviv University. She received the PhD degree in computer science from Princeton University. She is a professor in the Department of Electrical Engineering, Technion and the founder in the Laboratory of Computer Graphics and Multimedia.