3D Interest Point Detection via Discriminative Learning

Leizer Teran[1] and Philippos Mordohai[2]

[1] Drexel University, USA
[2] Stevens Institute of Technology, USA

Abstract. The task of detecting the interest points in 3D meshes has typically been handled by geometric methods. These methods, while designed according to human preference, can be ill-equipped for handling the variety and subjectivity in human responses. Different tasks have different requirements for interest point detection; some tasks may necessitate high precision while other tasks may require high recall. Sometimes points with high curvature may be desirable, while in other cases high curvature may be an indication of noise. Geometric methods lack the required flexibility to adapt to such changes. As a consequence, interest point detection seems to be well suited for machine learning methods that can be trained to match the criteria applied on the annotated training data. In this paper, we formulate interest point detection as a supervised binary classification problem using a random forest as our classifier. We validate the accuracy of our method and compare our results to those of five state of the art methods on a new, standard benchmark.

Keywords: 3D computer vision.

1 Introduction

The identification of important points in images has been a long standing problem in computer vision. Once detected, these important, or interest, points are encoded in one of many invariant representations, such as SIFT [19], and are used within a multitude of applications such as image registration, retrieval, object tracking and structure from motion. Note that Lowe [19] presents techniques for detecting and describing interest points, but one can use a different detector and then apply the SIFT descriptor. A similar two-stage approach can be applied to 3D data. However, due to concerns about the reliability of interest point detectors in 3D, in many cases descriptors are computed at uniformly sampled locations of the 3D model [15,9,24]. The reliability of 3D interest point detectors was recently studied by Dutagaci et al. [7] who created a benchmark (Fig. 1) and evaluated several state of the art methods. In this paper, we go beyond pure geometry for 3D interest point detection by learning to detect such points from a corpus of annotated data. Note that descriptor computation is out of scope here.

D. Fleet et al. (Eds.): ECCV 2014, Part I, LNCS 8689, pp. 159–173, 2014.

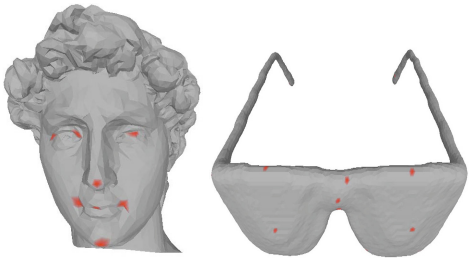


Fig. 1. David and glasses with marked ground truth points

One of the main difficulties in predicting interest points lies in the discrepancy between quantifiable importance and perceived importance. Many methods assume that quantitatively important points, usually found by optimization of a function, correspond to perceptually important points. This assumption works well for vertices that are co-located with sharp changes in the model, e.g. corners. For smoother regions and perceptually ambiguous points, the previous assumption along with a multi-scale approach have been met with varying success. Another layer of difficulty arises when semantic ambiguity is considered. This is due to varying, task-specific requirements and, in the case of our data, due to subjectivity of the annotators.

A successful algorithm should be invariant to rotation, translation and scaling, capture points that are appealing to a large consensus of people and have enough flexibility to deal with ambiguity and subjectivity. One approach to capturing subjectivity, which has not been fully explored in the 3D shape analysis literature, is discriminative learning that attempts to identify patterns associated with the annotators' preferences.

In this paper, we formulate 3D interest point detection as a binary classification problem. We use several geometric detectors to produce attributes which are the inputs to a learning algorithm that gives competitive performance against five state of the art methods on the benchmark of Dutagaci et al. [7]. The peculiarity of this benchmark is that the ground truth interest points were selected by non-expert users who were asked to click on the models. As a result, points with widely varying geometric properties are selected in each case. For example, as seen in Fig. 1, the high curvature of David's hair does not attract the attention of the annotators. In fact in many cases, such as the teddy bear in Fig. 2, the interest points lie on smooth hemi-spheres. Subjectivity and semantics play a large role in feature selection in this data set posing significant challenges to geometric methods. In some sense, our approach aims at encoding the potential subjective criteria of the annotators and then transferring them to unseen meshes in order to detect interest points according to these criteria. The results in Section 4.5 show that our approach is able to cope with this variability to a very satisfactory degree. Furthermore, consistent performance across widely varying 3D models is an indication of our algorithm's generalizability.

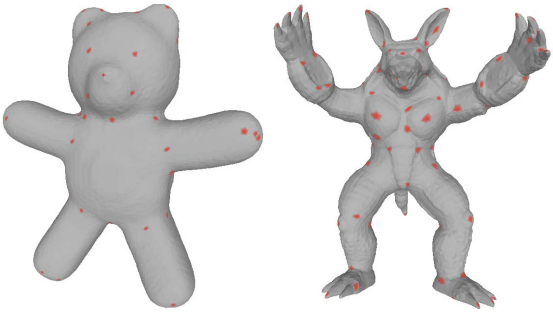


Fig. 2. Teddy bear and armadillo with marked ground truth points

2 Related Work

In a recent survey, Dutagaci et al [7] introduced a benchmark and an evaluation methodology for algorithms designed to predict interest points in 3D. The benchmark comprises 43 triangular meshes and the associated paper evaluated the performance of six algorithms [17,22,2,28,27,12] in interest point detection. Since we also use this benchmark, we focus our attention to these six methods in this section. Other methods for interest point detection on meshes include those based on distinctiveness compared with other local regions [11,10,26], as well as others based on thresholding geometric features [20] or detection in scale-space [16]. We refer readers to recent surveys [21,25,7,18,31] for more details.

Mesh Saliency. Lee et al. [17] address interest point detection through the use of local curvature estimates coupled with a center surround scheme at multiple scales. The total saliency of a vertex is defined as the sum of Difference of Gaussian (DoG) operators over all scales. Interest points are selected after non-max suppression on total saliency.

Scale Dependent Corners. Novatnack and Nishino [22] measure the geometric scale variability of a 3D mesh on a 2D representation of the surface geometry given by its normal and distortion maps, which can be obtained by unwrapping the surface of the model onto a 2D plane. A geometric scale-space which encodes the evolution of the surface normals on the 3D model while it is gradually smoothed is constructed and interest points are extracted as points with high curvature at multiple scales. The appropriate scale is automatically selected for each point.

Salient Points. Castellani et al. [2] also adopt a multi-scale approach. DoG filters are applied to vertex coordinates to compute a displacement vector of each vertex at every scale. The displacement vectors are then projected onto the normals of the vertices producing a "scale map" for each scale. Interest points are extracted among the local maxima of the scale maps after an inhibition process.

Heat Kernel Signature. Sun et al. [28] apply the Laplace-Beltrami operator over the mesh to obtain its Heat Kernel Signature (HKS). The HKS captures neighborhood structure properties which are manifested during the heat diffusion process on the surface model and which are invariant to isometric transformations. The time required for heat to diffuse from one part of the model to another can be used to form a signature which is invariant to isometric transformations. The local maxima of the HKS are selected as the interest points of the model.

3D Harris. Sipiran and Bustos [27] generalize the Harris and Stephens corner detector [13] to 3D. The computation is now performed on the rings of a vertex, which play the role of neighboring pixels. A quadratic surface is fitted to the points around each vertex. This enables the use of a filter similar to the Harris operator, the maximal responses of which are selected as interest points.

3D SIFT. Godil and Wagan [12] initially convert the mesh model into a voxel representation. Then, 3D Gaussian filters are applied to the voxel model at various scales as in the standard SIFT algorithm. DoG filters are used to compute the difference between the original model and the model at a particular scale and their extrema are taken as candidate interest points. The final set of interest points are those that also lie on the surface of the 3D object.

Related to our work is the approach of Holzer et al. [14] that detects interest points in range images using a regression forest. The key differences with our approach are that it operates on sequences of RGB-D images instead of meshes, it is designed for real-time processing and thus uses a simpler set of features, and it considers only objective, geometric criteria. Donner et al. [6] propose an approach for detecting class-specific landmarks, such as the finger tips of a human hand, in volumetric data. A similar problem of detecting facial features in meshes is addressed by Creusot et al. [4]. The two latter approaches [6,4] differ from ours in that they aim at detecting different instances of the same feature, while we aim at transferring the knowledge gained from human annotation from a set of meshes to a completely different mesh.

3 Attributes and Learning

In this section, we focus on the geometric attributes that are used as inputs to our classifier. The attributes capture characteristics that intuitively should help discriminate between interest and regular points, such as curvature, saliency compared to neighboring vertices, etc. Since interest points may become salient at different scales, we capture information at multiple scales by applying Difference of Gaussian (DoG) filters on the attributes [17,2]. The DoG filters are an implementation of the center-surround principle for detecting salient regions based on their differences with the surrounding context. All attributes are invariant to rotation, translation and scale. The latter is achieved by normalizing the lengths in each mesh by its diameter.

First, we present the attributes that serve as the basic building blocks for all the others. The motivation behind this basic set is to create descriptors that capture the local geometric properties and context of a given vertex. We attempt to quantify the basic properties of every vertex, v, through the following 10 attributes, the first 7 of which use the 100 nearest Euclidean neighbors, denoted as $\nu(v)$.

3.1 Basic Attributes

The first 5 attributes involve the scatter matrix about a vertex v:

$$S(v) = \sum_{x \in \nu(v)} \exp^{-\frac{||\mathbf{x}-\mathbf{v}||^2}{\tau^2/2}} \frac{(\mathbf{x}-\mathbf{v})(\mathbf{x}-\mathbf{v})^T}{||\mathbf{x}-\mathbf{v}||^2}, \tag{1}$$

with $\mathbf{x}$ being the coordinates of vertex x and $\mathbf{v}$ being the coordinates of vertex v. The parameter τ, was taken to be the radius of $\nu(v)$. The contribution of each neighbor is weighted according to its proximity with v to limit the influence of neighboring points that may be located across discontinuities.

Attributes 1 to 3 are ratios of eigenvalues of $S(v)$:

$$F_1(v) = \lambda_{1,v}/\lambda_{2,v} \tag{2}$$
$$F_2(v) = \lambda_{1,v}/\lambda_{3,v} \tag{3}$$
$$F_3(v) = \lambda_{2,v}/\lambda_{3,v}, \tag{4}$$

where $\lambda_{i,v}$ is the i-th largest eigenvalue of $S(v)$. These capture properties of the surface, such as planarity in which case $S(v)$ is almost rank deficient, or corners that have three eigenvalues of similar magnitude.

We look at the differences between the eigenvalues for the next two attributes, as follows:

$$F_4(v) = \lambda_{2,v} - \lambda_{1,v} \tag{5}$$
$$F_5(v) = \lambda_{3,v} - \lambda_{2,v}. \tag{6}$$

Attribute 6 is the vertex density about the point, whereas attribute 7 is the average inner product of vertex v's normal with the normals of its 100 nearest neighbors:

$$F_6(v) = \frac{100}{\mathrm{Vol}(\nu(v))} \tag{7}$$
$$F_7(v) = \frac{\sum_{x \in \nu(v)} \mathbf{n}(v) \cdot \mathbf{n}(x)}{100} \tag{8}$$

Where $\mathbf{n}(x)$ is the vertex normal of vertex x.

Attributes 8 and 9 are the principal curvatures at vertex v and the 10^{th} attribute is its Gaussian curvature.

$$F_8(v) = \kappa_1(v) \tag{9}$$
$$F_9(v) = \kappa_2(v) \tag{10}$$
$$F_{10}(v) = \kappa_1(v)\kappa_2(v) \tag{11}$$

To compute the principal curvatures, we look at the one-ring of the current vertex. Then, directional curvatures along the edges are approximated and used to compute the tensor of curvature [30].

3.2 DoG Attributes

The basic attributes described above are generally not sufficient to detect all interest points, since they may become salient at different scales. Inspired by the success of the multi-scale approach adopted by other algorithms [17,2], we compute a set of DoG attributes that are functions of the basic attributes.

The DoG attribute computations were performed within Euclidean neighborhoods of radius r centered at vertex v, which will be referred to as $N(v, r)$. We compute the Gaussian weighted average within the $\delta, 2\delta, 4\delta$ and 6δ neighborhoods of vertex v, for each basic attribute. We use 0.3% of the model diameter as δ. In addition, we compute the Gaussian weighted neighborhood averages of the mean curvature for vertex v, which was not included in the set of basic attributes because it is a linear combination of the principal curvatures.

Attributes 11 through 20 are the DoGs between the δ and 2δ neighborhoods at each vertex for each basic attribute.

$$G_{\delta,j}(v) = \frac{\sum_{x \in N(v,\delta)} F_j(x) \exp^{-||x-v||^2/(2\delta^2)}}{\sum_{x \in N(v,\delta)} \exp^{-||x-v||^2/(2\delta^2)}} \tag{12}$$

$$F_{j+10}(v) = |G_{2\delta,j}(v) - G_{\delta,j}(v)| \ , j \in \{1...10\}. \tag{13}$$

Attribute 21 is the DoG of the mean curvature for the δ and 2δ neighborhoods:

$$\mu_\delta(v) = \frac{\sum_{x \in N(v,\delta)} C(x) \exp^{-||x-v||^2/(2\delta^2)}}{\sum_{x \in N(v,\delta)} \exp^{-||x-v||^2/(2\delta^2)}} \tag{14}$$

$$F_{21}(v) = |\mu_{2\delta} - \mu_\delta| \tag{15}$$

with $C(x)$ being the mean curvature at vertex x.

Attributes 22 through 31 are DoGs for the basic attributes but use the 2δ and 4δ neighborhoods instead. The 32^{nd} attribute is the mean curvature DoG for the 2δ and 4δ neighborhoods. Finally, we look at the DoGs for the 4δ and 6δ neighborhoods to define the 33^{rd} through 43^{rd} attributes in the same way.

To summarize, each vertex has a total of 43 attributes, denoted by $F_i(v)$ with $i \in 1, ..., 43$. The attribute set for each vertex can be broken down into a set of basic attributes $\{F_1(v)...F_{10}(v)\}$ and a set of DoG attributes $\{F_{11}(v)...F_{43}(v)\}$ that are functions of the basic attributes at varying scales. Having defined the inputs to our classifier, we now shift our attention to the random forest.

3.3 Random Forest

Random forest classifiers are ensembles of classification and regression trees that have gained popularity due to their high accuracy and ability to generalize [1,5].

The key idea during training is to generate decisions trees that partition the attribute space in a way that separates the training data according to their labels, interest and non-interest points in our case. In the training stage, a new training set is created for each tree by sampling with replacement (bootstrapping) from the original training set. Each node performs randomly generated tests on random subsets of the full attribute set. The attribute and threshold value that optimize a function of the input samples is selected and the data are divided to the node's children. The Gini gain and the information gain are standard functions used for the selection. We use the former in our analysis.

Once the forest has been trained, the test set vertices are fed to each trained tree in the forest. The current vertex is run down each tree and decisions are made at every node based on the optimal splits computed during training. This process continues until the terminal node is reached and a decision is made about the current vertex's class label. The class label that is output by the majority of the trees in the forest is assigned to the vertex. We use Scikit-Learn [23] to implement the random forests.

The performance of the forest is controlled by its parameters: the depth of each tree, the number of attributes at each node and the the number of trees in the forest. Following Breiman's recommendation [1], we do not prune the trees and allow them to grow until each leaf contains one example. In general, the full attribute set is not used while sampling at each node. This is done to keep the trees in the forest as uncorrelated as possible. Following common practice, we use $\sqrt{p}$ attributes, where p is the total number of attributes, as a guideline to search for the optimal number of attributes used at each node. As discussed in Section 4, we use cross-validation to determine the number of trees.

3.4 Imbalanced Classes

Imbalanced classes present a challenge for most classifiers. Poor predictive performance arises because the standard implementation aims to reduce the overall error rate. As a consequence, the random forest can afford to misclassify almost all the minority class examples and still achieve a very low error rate. To make matters worse, the minority class may not even be selected during bootstrapping and therefore may be missing almost entirely during the training process. For our problem, the ratio of interest to non interest points can range anywhere from 1:100 to even 1:240 within our training data set.

There are a few strategies to deal with the misrepresentation of classes. We chose a technique proposed by Chen et al [3] where the dominant class is down-sampled, while the minority class is over-sampled. For a set of labelled vertices, we randomly select n interest points, where n is one half of the total interest points. During training each tree is given a balanced set of vertices that have a ratio of k non interest vertices to n interest vertices, where $k \geq n$. The parameter k and the bootstrap ratio will be discussed in Section 4.

4 Ground Truth and Experiments

In this section we describe the data we used, the experimental setup and our results.

4.1 Ground Truth

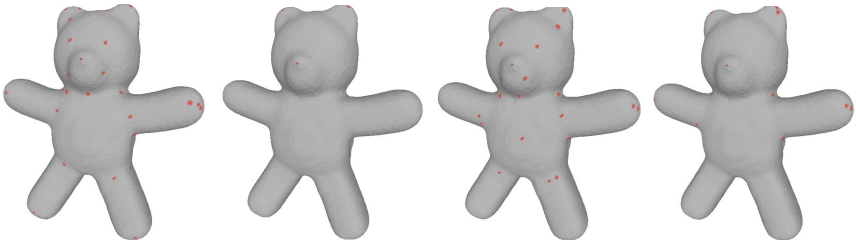
Dutagaci et al. [7] used a web-based application to collect user clicks on 43 mesh models. These models were organized in two overlapping data sets, Data Set A and Data Set B, consisting of 24 and 43 triangular mesh models respectively. Through the web-based application, a user was shown the models from a data set one at a time and was allowed to freely click on them. Data Set A was annotated by 23 human subjects while Data Set B was annotated by 16 human subjects. The positions of the individual user clicks showed some variability as well as some consensus. The variability may be due to imprecise clicking or the subjective nature of interest points.

In order to determine user consensus and remove outliers, the authors considered two criteria while constructing each set's ground truth. The first is the radius, σd_M, of an interest region and the second is the number of users n that clicked within the region. The radius of the interest region is model-specific with d_M denoting the diameter of the model and σ is a parameter in [0.01, 0.02, 0.03, ... , 0.1]. Individual user clicks are clustered together if their geodesic distances are less than $2\sigma d_M$. If the number of clicks in a cluster is less than n that cluster is ignored. If not, the point that minimizes the sum of geodesic distances to the other points is chosen as the representative of the cluster and included in the ground truth. In case the distance between cluster representatives is less than $2\sigma d_M$, the clusters are merged and the representative with the highest number of cluster points is chosen as the final representative.

The parameters, σ and n, affect the number of ground truth points. For a fixed σ, fewer ground truth points are observed as n increases since a higher consensus among users is needed. With small σ and large n, there tends to be better localization around the points and typically fewer ground truth points are observed. As can be seen in Fig. 3, as σ increases, more ground truth points are observed. This trend continues until the clusters are large and close enough to be merged with adjacent clusters, consequently decreasing the overall number of ground truth points for large σ.

4.2 Experiments

We use the human generated ground truth to compare the random forest with the five following methods: Mesh Saliency [17], Salient Points [2], Heat Kernel Signature [28], 3D Harris [27] and Scale Dependent Corners [22], across Data Sets A and B. There are a few models within these data sets that are not watertight and therefore do not allow volumetric representations. As a consequence, the output of 3D SIFT for these models was not provided at the benchmark website.



(a) $\sigma = 0.03, n = 2$ (b) $\sigma = 0.03, n = 8$ (c) $\sigma = 0.05, n = 2$ (d) $\sigma = 0.05, n = 8$

Fig. 3. Ground truth points for the teddy bear for four σ/n combinations

This led to its exclusion from our analysis, but partial results can be seen in [7]. (3D SIFT is not among the top performing methods on the benchmark.)

Data Sets A and B are treated as distinct experiments where we measure the effects of user consensus, n, and ground truth localization, σ, on algorithmic performance. Adhering to the evaluation protocol [7], we adopt the following σ/n combinations for Data Set A: 0.03/2, 0.03/11, 0.05/2 and 0.05/11. For Data Set B we use the following combinations: 0.03/2, 0.03/8, 0.05/2 and 0.05/8. In other words, for a given mesh model M, we assess how well the algorithms perform in detecting ground truth points agreed upon by at least two subjects or by at most one half of the subjects labeling each set.

4.3 Training and Predicting

For all experiments, we apply three-fold cross validation for both Data Sets A and B. Specifically, Data Set B was partitioned into three disjoint sets consisting of 14 models each (igea was removed since igea and bust2 are duplicates). Data Set A was split into three disjoint sets consisting of eight models each. Once the splits are established for a given set, we train the random forest, using the attributes from Section 3 as the classifier input, on the first two folds and predict the interest points of the last fold (test set), then the folds are rotated between the test and training sets. The test data are never seen by the classifiers and are not used for parameter selection.

Every vertex of every model in the training set of a given fold has a set of labels. Because the ground truth points vary by σ and n, we have to make a compromise between the number of ground truth points available and high consensus among the annotators. For Data Set B, the representative of clusters σ/n, $n \in [11...22]$, are placed in the positive class for all values of σ provided. All other vertices are placed in the negative class. Likewise, for Data Set A we place the representative of clusters σ/n, $n \in [8...15]$, in the positive class for every available value of σ. The remaining vertices are considered members of the negative class.

Our classifier has three main parameters. The first is the bootstrap ratio of interest vertices to non-interest vertices that is used to balance the training set.

The second is the number of attributes sampled during the node splitting process and the third is the number of trees. We use a 1:1 ratio of interest vertices to non-interest vertices while sampling and a random sample of 5 attributes at every node for all 100 unpruned trees in the forest. These parameters are found via cross validation.

Examining the resulting random forests reveals that the most important features on Data Set B are: the DoG of Gaussian curvature at the largest scale (f_{42}), the average inner product with the 100 neighbors (f_7), Gaussian curvature (f_{10}), the DoG of mid-scale Gaussian curvature (f_{31}), maximum principal curvature (f_9) and two of the eigenvalue ratios (f_1 and f_2). Most feature types, at some scale, contribute to the classifier.

In general, the random forest returns a large number of interest points, as nearby vertices that have similar attributes form clusters. To address this, non-max suppression is performed on the test set vertices that are chosen by the forest. The suppression is done within the $c\psi$ Euclidean neighborhood of the chosen vertices, where c is found to be 5 for Data Set A and 2 for Data Set B via cross validation. ψ is the average minimum geodesic distance between ground truth vertices of the training set.

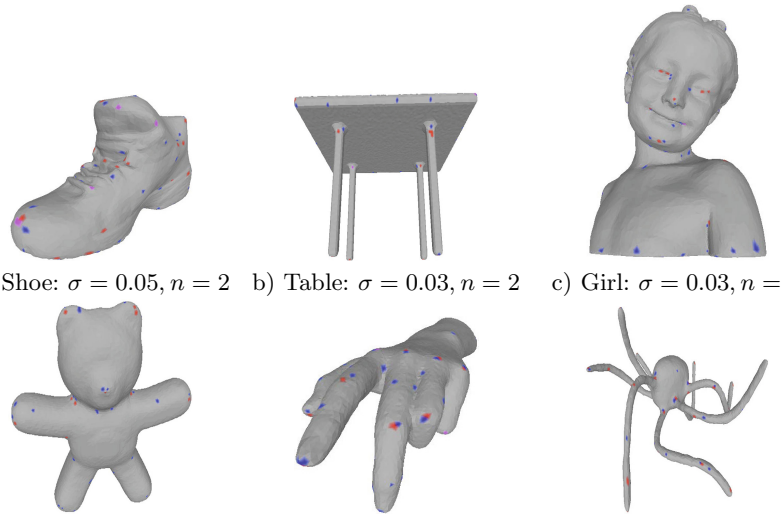
Note, that the competing methods cannot be trained. Adjusting some threshold is their only means for adaptation. Their outputs were downloaded from the website provided by the authors of [7] and compared with our method's output on each test set. A detailed explanation of the test set scoring is given in the following section.

4.4 Evaluation Criteria

Dutagaci et al. [7] evaluated the performance of the algorithms based on the following definitions and criteria. Let A_M be the set of vertices selected by the algorithm for a given mesh model, M, with diameter d_M. A ground truth point, $g \in G$, is correctly identified if there exists $a \in A_M$ such that the geodesic distance between them is less than some error tolerance, ϵ, and that no other point in G is closer to a. Or in other words, g_0 is correctly identified by the algorithm if: $g_0 = \text{argmin}_{g \in G}(d(a, g) \leq \epsilon)$ for some $a \in A_M$. The following error tolerances are used: $\epsilon = rd_M$ with $r \in [0, 0.1, 0.2...0.12]$.

This definition allows each correctly detected point in the ground truth set to be in correspondence with a unique $a \in A$. The number of false positives (FP) is then $N_A - N_C$ and the false negatives (FN) are $N_G - N_C$ where N_A, N_C and N_G are the number of algorithm selected points, correctly identified points and number of ground truth points respectively. The geodesic distances were computed using publicly available software [29].

In addition to the evaluation criteria proposed by the authors of the benchmark, we also adopt the Intersection Over Union (IOU) criterion, which has been used to evaluate object detection in images [8]. We choose the IOU as our main metric because FN and FP rates can be misleading in isolation. For a mesh model, M, and algorithm, A, we use FP, FN and TP (true positives) as defined above to compute the IOU of an algorithm over a mesh model as:



a) Shoe: $\sigma = 0.05, n = 2$ b) Table: $\sigma = 0.03, n = 2$ c) Girl: $\sigma = 0.03, n = 8$

d) Teddy: $\sigma = 0.05, n = 8$ e) Hand: $\sigma = 0.03, n = 8$ f) Octopus: $\sigma = 0.03, n = 2$

Fig. 4. Random Forest predictions for six Set B models. Ground truth points for various σ/n pairs in red. Random forest predictions in blue. Purple points are ground truth vertices that are predicted exactly.

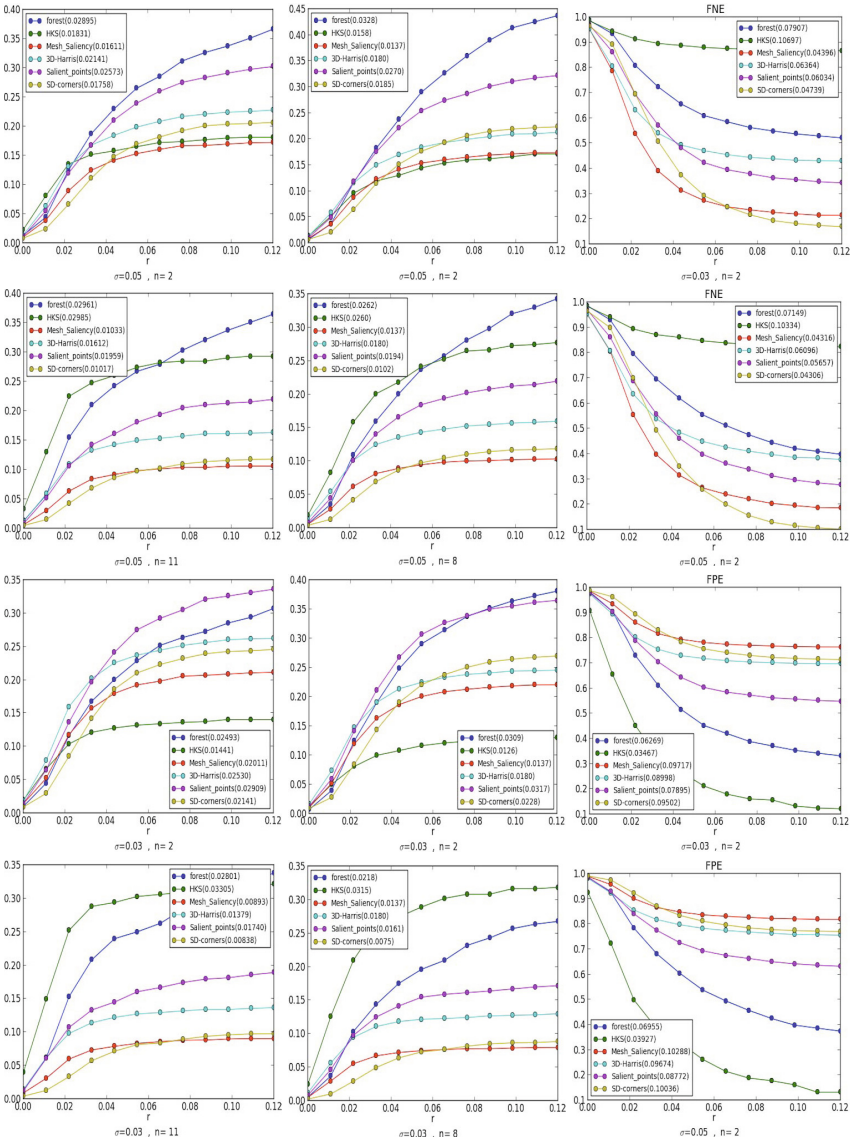
$$\text{IOU}_{A,M} = \frac{\text{TP}}{\text{FP} + \text{TP} + \text{FN}} \tag{16}$$

The definition for the IOU given above is for a mesh model. To find the IOU score of an algorithm over a set of models, as is done in the experiments, a running total of the false positives, false negatives and true positives were kept over all vertices in the test set and then used to compute a set-wise IOU. We compute the set-wise IOU score for the σ/n combinations in the experiments section. For a given σ/n combination the IOU scores are found for values of $r \in [0, 0.1, 0.2...0.12]$.

4.5 Results

In this section, we present the results of the algorithm evaluations for Data Set A and Data Set B under the IOU metric. Figure 4 contains some visualizations of the results, while Fig. 5 shows the set-wise IOU scores averaged over the three folds as the radius r varies. The legends show the Area Under the Curve (AUC) for the different values of r. In addition, we compare the methods at $\sigma = 0.03/n = 2$ and $\sigma = 0.05/n = 2$ using the evaluation criteria proposed by [7] in the last column of Fig. 5. Here, FNE and FPE are defined as $1 - N_C/N_G$ and FP/N_A respectively. It is important to note that the points detected by the methods are constant when σ and n change.

Figure 5 shows that the random forest is either first or second across Data Set B's σ/n settings. It performs especially well when the localization is relaxed, as



a) Data Set A b) Data Set B c) FNE/FPE

Fig. 5. Column 1 shows the IOU curves for Data Set A at various σ/n pairs. Column 2 contains the IOU curves for Data Set B and Column 3 has the FNE and FPE rates for Data Set B with $\sigma = 0.03, n = 2$ and $\sigma = 0.05, n = 2$. The parameter r is mentioned in Section 4.4. The AUC for each method is provided in the legend.

can be seen when $\sigma = 0.05$. One possible cause for this is that new ground truth points emerge when σ increases for $n = 8$. This is indicative of an ambiguous region within the model, as seen in the teddy bear's neck in Fig. 4. The random forest captures these regions more effectively than the other methods.

Small values of n result in large numbers of ground truth interest points, favoring aggressive methods, such as Salient Points. On the other hand, when n is large, conservative methods are able to identify the truly salient points without producing too many false positives. In this sense, the HKS algorithm is an outlier compared to the rest of the algorithms, including ours. HKS can reliably detect a very small number of very salient points but is unable to detect even slightly more ambiguous points. The last column of Fig. 5 contains FPE and FNE curves for Data Set B with $n = 2$. These curves reveal how aggressive or conservative the algorithms are. The random forest performs well according to all criteria over the parameter range.

Data Set A is expected to be more challenging for a learning-based method, since the training set is smaller. Nevertheless, our algorithm is the top performing one according to IOU. To reach an overall ranking, we average the IOU AUCs over all settings. The results are summarized in Table 1.

Table 1. Average IOU AUCs for all methods on both data sets

Method	Data Set A	Data Set B
Random Forest	0.0279	0.0279
HKS	0.0239	0.0215
Salient Points	0.0230	0.0236
3D Harris	0.0192	0.0180
SD Corners	0.0144	0.0148
Mesh Saliency	0.0139	0.0137

5 Conclusions

In this paper, we have presented a discriminative learning approach to 3D interest point detection that gives competitive performance over state of the art methods. Experiments on a new, publicly available benchmark demonstrate that our method handles variability in the ground truth, or the desirable output, more steadily than other methods. A closer look at Figs. 1, 2 and 4 reveals the diversity of the data, which combined with the relatively small number of models on which our classifier is trained, serves as evidence of generalizability. This translates to an increased ability to cope with human subjectivity in these experiments, and it is equivalent to the ability to adapt to different task-specific requirements imposed on the algorithm. This is they key difference between our work and other approaches that rely on purely geometric criteria for detecting interest points.

References

1. Breiman, L.: Random forests. Machine Learning 45(1), 5–32 (2001)
2. Castellani, U., Cristani, M., Fantoni, S., Murino, V.: Sparse points matching by combining 3D mesh saliency with statistical descriptors. Computer Graphics Forum 27(2), 643–652 (2008)
3. Chen, C., Liaw, A., Breiman, L.: Using random forest to learn imbalanced data. Tech. rep., University of California, Berkeley (2004)
4. Creusot, C., Pears, N., Austin, J.: A machine-learning approach to keypoint detection and landmarking on 3D meshes. IJCV 102(1-3), 146–179 (2013)
5. Criminisi, A., Shotton, J., Konukoglu, E.: Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning. Found. Trends. Comput. Graph. Vis. 7(2-3), 81–227 (2012)
6. Donner, R., Birngruber, E., Steiner, H., Bischof, H., Langs, G.: Localization of 3D anatomical structures using random forests and discrete optimization. In: Menze, B., Langs, G., Tu, Z., Criminisi, A. (eds.) MICCAI 2010. LNCS, vol. 6533, pp. 86–95. Springer, Heidelberg (2011)
7. Dutagaci, H., Cheung, C., Godil, A.: Evaluation of 3D interest point detection techniques via human-generated ground truth. The Visual Computer 28, 901–917 (2012)
8. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The pascal visual object classes (VOC) challenge. IJCV 88(2), 303–338 (2010)
9. Frome, A., Huber, D., Kolluri, R., Bülow, T., Malik, J.: Recognizing objects in range data using regional point descriptors. In: Pajdla, T., Matas, J. (eds.) ECCV 2004, part III. LNCS, vol. 3023, pp. 224–237. Springer, Heidelberg (2004)
10. Gal, R., Cohen-Or, D.: Salient geometric features for partial shape matching and similarity. ACM Transactions on Graphics 25(1), 130–150 (2006)
11. Gelfand, N., Mitra, N.J., Guibas, L.J., Pottmann, H.: Robust global registration. In: Proceedings of the Third Eurographics Symposium on Geometry Processing (2005)
12. Godil, A., Wagan, A.I.: Salient local 3D features for 3D shape retrieval. arXiv 1105.2796[cs.CV] (2011)
13. Harris, C., Stephens, M.: A combined corner and edge detector. In: Proceedings of the 4th Alvey Vision Conference, pp. 147–151 (1988)
14. Holzer, S., Shotton, J., Kohli, P.: Learning to efficiently detect repeatable interest points in depth data. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part I. LNCS, vol. 7572, pp. 200–213. Springer, Heidelberg (2012)
15. Johnson, A., Hebert, M.: Using spin images for efficient object recognition in cluttered 3D scenes. PAMI 21(5), 433–449 (1999)
16. Knopp, J., Prasad, M., Willems, G., Timofte, R., Van Gool, L.: Hough transform and 3D SURF for robust three dimensional classification. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part VI. LNCS, vol. 6316, pp. 589–602. Springer, Heidelberg (2010)
17. Lee, C.H., Varshney, A., Jacobs, D.: Mesh saliency. ACM Transactions on Graphics 24(3), 659–666 (2005)
18. Lian, Z., Godil, A., Bustos, B., Daoudi, M., Hermans, J., Kawamura, S., Kurita, Y., Lavoué, G., Van Nguyen, H., Ohbuchi, R., et al.: A comparison of methods for non-rigid 3D shape retrieval. Pattern Recognition 46(1), 449–461 (2013)

19. Lowe, D.: Distinctive image features from scale-invariant keypoints. IJCV 60(2), 91–110 (2004)
20. Matei, B., Shan, Y., Sawhney, H.S., Tan, Y., Kumar, R., Huber, D., Hebert, M.: Rapid object indexing using locality sensitive hashing and joint 3D-signature space estimation. PAMI 28(7), 1111–1126 (2006)
21. Mian, A.S., Bennamoun, M., Owens, R.A.: On the repeatability and quality of keypoints for local feature-based 3D object retrieval from cluttered scenes. IJCV 89(2-3), 348–361 (2010)
22. Novatnack, J., Nishino, K.: Scale-dependent 3D geometric features. In: ICCV (2007)
23. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research 12, 2825–2830 (2011)
24. Rusu, R., Blodow, N., Beetz, M.: Fast point feature histograms (FPFH) for 3D registration. In: ICRA, pp. 3212–3217 (2009)
25. Salti, S., Tombari, F., Di Stefano, L.: A performance evaluation of 3D keypoint detectors. In: 3DIMPVT, pp. 236–243 (2011)
26. Shilane, P., Funkhouser, T.: Distinctive regions of 3D surfaces. ACM Transactions on Graphics 26(2) (2007)
27. Sipiran, I., Bustos, B.: Harris 3d: a robust extension of the harris operator for interest point detection on 3D meshes. Visual Computer 27(11), 963–976 (2011)
28. Sun, J., Ovsjanikov, M., Guibas, L.: A concise and provably informative multi-scale signature based on heat diffusion. In: Proceedings of the Symposium on Geometry Processing, pp. 1383–1392 (2009)
29. Surazhsky, V., Surazhsky, T., Kirsanov, D., Gortler, S.J., Hoppe, H.: Fast exact and approximate geodesics on meshes. ACM Transactions on Graphics 24(3), 553–560 (2005)
30. Taubin, G.: Estimating the tensor of curvature of a surface from a polyhedral approximation. In: ICCV, pp. 902–907 (1995)
31. Yu, T.H., Woodford, O.J., Cipolla, R.: A performance evaluation of volumetric 3D interest point detectors. IJCV 102(1-3), 180–197 (2013)