Eurographics Workshop on 3D Object Retrieval (2017)
I. Pratikakis, F. Dupont, and M. Ovsjanikov (Editors)

SHREC'17: Deformable Shape Retrieval with Missing Parts

E. Rodolà[1], L. Cosmo[2], O.Litany[3,4], M. M. Bronstein[1,3,4], A. M. Bronstein[3,4,5]
N. Audebert[9], A. Ben Hamza[8], A. Boulch[9], U. Castellani[17], M. N. Do[16], A-D. Duong[14], T. Furuya[6], A. Gasparetto[2], Y. Hong[10], J. Kim[10], B. Le Saux[9], R. Litman[3], M. Masoumi[8], G. Minello[2], H-D. Nguyen[13,15], V-T. Nguyen[13,14], R. Ohbuchi[6], V-K. Pham[13], T. V. Phan[13], M. Rezaei[8], A. Torsello[2], M-T. Tran[13], Q-T. Tran[13], B. Truong[13], L. Wan[7], C. Zou[11,12]

[1]USI Lugano, [2]Università Ca' Foscari Venezia, [3]Tel Aviv University, [4]Intel Perceptual Computing, [5]Technion, [6]University of Yamanashi, [7]Beijing Jiaotong University, [8]Concordia University, [9]ONERA, The French Aerospace Lab, [10]Sangmyung University, [11]Hengyang Normal University, [12]Simon Fraser University, [13]University of Science, VNU-HCM, [14]University of Information Technology, VNU-HCM, [15]John von Neumann Institute, VNU-HCM, [16]University of Illinois at Urbana-Champaign, [17]University of Verona

Abstract

Partial similarity problems arise in numerous applications that involve real data acquisition by 3D sensors, inevitably leading to missing parts due to occlusions and partial views. In this setting, the shapes to be retrieved may undergo a variety of transformations simultaneously, such as non-rigid deformations (changes in pose), topological noise, and missing parts – a combination of nuisance factors that renders the retrieval process extremely challenging. With this benchmark, we aim to evaluate the state of the art in deformable shape retrieval under such kind of transformations. The benchmark is organized in two sub-challenges exemplifying different data modalities (3D vs. 2.5D). A total of 15 retrieval algorithms were evaluated in the contest; this paper presents the details of the dataset, and shows thorough comparisons among all competing methods.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Geometric algorithms, languages, and systems

1. Introduction

Shape retrieval concerns the problem of finding 3D shapes from a shape repository, that are as similar as possible to a given query. It is an active area of research, finding application in several areas of computer vision, graphics, and medical imaging among several others. A challenging setting of 3D shape retrieval occurs when the shapes to be retrieved are allowed to undergo *non-rigid* deformations, which may range from simple changes in pose (near-isometric deformations) to more complicated transformations such as local stretching, intra-class variation, and topological noise. Even more challenging is the setting in which the objects in question have *missing parts* in addition to the aforementioned deformations. In this scenario, the query object is a subset of the full model and the task is to retrieve similar objects with different partiality patterns (or lack thereof), additionally undergoing a non-rigid transformation; vice versa, the query object might be a full model, while the shape repository may contain deformed partial views of objects of the same class.

With this track, we propose a benchmark to evaluate the performance of partial shape retrieval methods, under the *concurrent* presence of:

- Non-rigid deformations;
- Different amounts and types of partiality;
- Topological changes induced by mesh gluing in areas of contact.

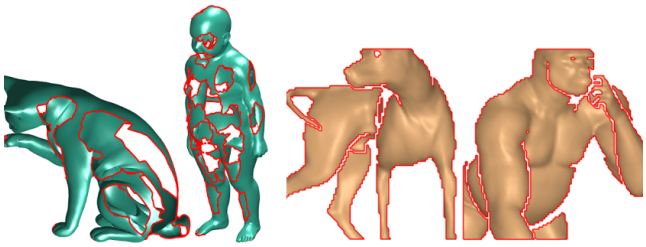


Figure 1: *Example of shapes from the 'holes' (left pair) and 'range' (right pair) challenges. Shape boundaries are colored in red.*

Related benchmarks. Although recent publications in the field of shape analysis [DLL*10, WZZ16, CRA*16, CRM*16] suggest that the community is starting to show interest in the problem of deformable partial shape retrieval, there is a notable lack of benchmarks targeting this task. Similar benchmarks addressing the related problem of correspondence were recently proposed [CRB*16,LRB*16a], but these only include a relatively small number of shapes – a few hundreds as opposed to the several thousands that would be required to provide a meaningful evaluation of retrieval pipelines. Partial shape retrieval was recently addressed in [PSA*16] for *rigid* objects (partial views of vases), and a few

years back in [MPB07, BBC*10] (mostly CAD models, with only 2 classes presenting mild partiality and changes in pose).

By constructing a new, large, specific and challenging dataset with partial deformable shapes, we aim to provide a valuable testbed for the most recent top-performing retrieval methods, which all too often we see "saturate" existing benchmarks that are not specifically designed with partiality in mind. We hope that this new benchmark will foster further interest of the community in this challenging problem.

2. Dataset

The dataset is composed of transformed versions of shapes belonging to the TOSCA repository [BBK08] and to the SHREC'16 Topology [LRB*16a] benchmark. The former consists of 9 shape categories (humans and animals) under various changes in pose; the second includes different poses of a fixed human subject, additionally undergoing topological transformations in the form of mesh 'gluing' in areas of self-contact. The dataset we propose is divided into two subsets, each constituting a different challenge:

- **'holes'**: On each model we consider 70 Euclidean farthest samples. A random subset of at least 20% of these samples is used as seed for an erosion process with random geodesic radius (up to 20% shape diameter per sample). Each resulting partial shape is remeshed by edge contraction [GH97] to a random amount of vertices between 300 and the full mesh resolution, followed by application of a random similarity transformation.
- **'range'**: A virtual orthographic camera is placed in front of 5 randomly rotated versions of each model, at 4 random elevations and resolutions. The resulting 2.5D snapshot is triangulated according to a regular grid, with random maximum allowed edge length (this generates topological shortcuts between parts that are originally disconnected).

See Fig. 1 for examples. Note that the shapes might consist of multiple connected components in both cases. Each set is further divided into a training set, for which (category-level) ground-truth labels are provided, and a challenge set. We consider 'david' and 'michael' from the TOSCA repository as belonging to the same class, while 'victoria' and 'kid' constitute separate classes.

In total, the dataset consists of 1216 training / 1078 test shapes for 'holes', with resolution spanning 323 to 56172 vertices, and 1082 training / 882 test shapes for 'range', with 300 to 7021 vertices. The dataset remains available for download at http://sites.google.com/view/shrec17/.

3. Evaluation measures

The participants were asked to submit a full distance/dissimilarity matrix $\mathbf{D} \in [0,1]^{n \times n}$ for each challenge, where n is the number of shapes in the test set and (d_{ij}) denotes the dissimilarity between shape i and shape j. Participation to both challenges was not mandatory; up to 3 runs per method were allowed.

For the evaluation we adopt the standard measures for shape retrieval according to the Princeton benchmark protocol [SMKF04] using code from [LZC*15]. In particular, we show Precision/Recall curves, and tabled values for nearest neighbor (NN), 1st tier, 2nd tier, and discounted cumulative gain (DCG, at rank equal to n).

4. Methods

The contest saw the participation of 8 research groups, some of which participated with multiple submissions (resulting either from different retrieval methods, or different runs of the same method). In addition, the organizers implemented two baselines (Sections 4.1 and 4.2), both of which are extensions of existing approaches, properly modified to account for missing parts.

4.1. Bag of Words (BoW) [L. Cosmo and E. Rodolà]

As a first baseline, we use a simple BoW approach based on the scale-invariant Heat Kernel Signature (siHKS) [KB10], computed on keypoints extracted as the local maxima of the original HKS [SOG09]. The overall pipeline is composed by the following steps:

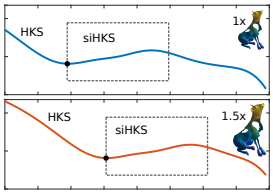
1. For each shape, salient points are extracted using [SOG09], and siHKS features are computed with a variant of [KB10];
2. A dictionary is built by descriptor clustering via k-means;
3. Each shape is represented by a frequency histogram, constructed by assigning each feature to the nearest word in the dictionary;
4. The pairwise distance between shape histograms is computed.

The code for this baseline is available on the track website.

Keypoints and descriptors. Salient points of a mesh $\mathcal{M}$ are extracted using the method proposed in [SOG09]: given a diffusion time t and the associated self-diffusivity H_t, we consider a point $x \in \mathcal{M}$ as a salient point if $H_t(x) > H_t(x')$ for all x' in the one-ring neighborhood of x. Boundary vertices are ignored.

The descriptor is then transformed according to:

$$\dot{H}_t = \log(H_{t+1}) - \log(H_t)\,. \tag{1}$$



Scaling of the original shape corresponds to a shift of $\dot{H}$. Conversely from the original method [KB10], we do not use the DFT of $\dot{H}$ to attain scale-invariance; instead, we restrict each descriptor to within a small window starting from the first local minimum/maximum (see inset figure).

BoW representation and matching. Once salient points descriptors are computed for each shape, a dictionary is constructed by clustering all the descriptors with a k-means algorithm, and by considering their centroids as words of a dictionary $C = (c_0, \ldots, c_k)$.

A frequency histogram is then computed for each shape $\mathcal{M}$, assigning each descriptor $d_i^{\mathcal{M}}$ to the nearest word in the dictionary:

$$\mathbf{b}_{\mathcal{M}} = \sum_i \mathbf{e}_{\arg\min_j d(d_i^{\mathcal{M}}, c_j)}\,, \tag{2}$$

where $\mathbf{e}_j$ is 1 at position j and 0 at all other indices, and $d(c_i, c_j) = \sum_t |c_i^t - c_j^t|$ is the cityblock distance.

Finally, the distance between two shapes $\mathcal{M}$ and $\mathcal{N}$ is computed using the cosine similarity:

$$d_{\mathcal{M},\mathcal{N}} = 1 - \frac{\mathbf{b}'_{\mathcal{M}}\mathbf{b}_{\mathcal{N}}}{\sqrt{(\mathbf{b}'_{\mathcal{M}}\mathbf{b}_{\mathcal{M}})(\mathbf{b}'_{\mathcal{N}}\mathbf{b}_{\mathcal{N}})}}\,. \tag{3}$$

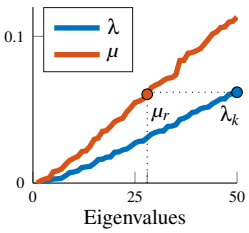
Implementation details. Even if the descriptor used is scale-invariant, one needs to consider comparable diffusion times to be able to retrieve the relative scale; to this end, the shapes were normalized by the area of their bounding box. Moreover, for increased robustness to partiality and to avoid boundary effects, small diffusion times must be used for the computation of the HKS. For this step we replaced the standard spectral implementation of the heat kernels [SOG09] with the robust Chebyshev-Padé method introduced in [Pat13]. This proved to be crucial for an accurate evaluation of the descriptors under large amounts of partiality.

The baseline comes in two variants, namely BoW-siHKS (described above) and BoW-HKS, where the logarithm of the HKS is used instead of siHKS. Diffusion times were sampled logarithmically at 80 samples between 2^{-16} and 2^{-6}, and the window size for the scale-invariant feature was set to 20 time samples. The dictionary size was set to $k = 128$.

4.2. Partial ShapeDNA (DNA) [E. Rodolà and L. Cosmo]

The second baseline is an extension of the ShapeDNA global shape descriptor described in [RWP06]. This approach is based upon the observation that, assuming lack of partiality (i.e., in the standard full shape setting), the spectrum of the Laplace-Beltrami operator of a shape acts as an isometry-invariant "fingerprint" for recognition. Recently, Rodolà et al. [RCB*16] showed how the removal of shape parts modifies the Laplacian spectrum, providing a strong prior that was subsequently used for the purpose of shape matching.

In particular, consider a full shape $\mathcal{M}$ and a subset $\mathcal{N} \subseteq \mathcal{M}$ (or an isometrically deformed version of $\mathcal{N}$), and denote by $(\lambda_j)_{j=1}^k$ and $(\mu_i)_{i=1}^k$ the increasingly ordered sequences of the first k Laplacian eigenvalues of $\mathcal{M}$ and $\mathcal{N}$ respectively. Then, the equality $\mu_i \approx \lambda_j$ holds (approximately) for $i \leq j$ and $i = 1,\dots,r$, where $r \approx \lfloor k \frac{\text{area}(\mathcal{M})}{\text{area}(\mathcal{N})} \rfloor$ [RCB*16]. See the inset figure below for an illustration. Note that the standard setting where $\mathcal{N}$ is a full isometric deformation of $\mathcal{M}$, assumed in the original ShapeDNA method [RWP06], is obtained with $i = j$ and $r = k$. Litany et al. [LRB*16b] later showed that the part-to-full analysis of [RCB*16] can be extended with little modifications to the part-to-part setting.



One may use this knowledge for the task of *partial* shape retrieval by comparing the spectra (μ_i) and (λ_j) at the pairs of indices $\mathcal{I} = \{(i, \lfloor i\frac{k}{r} \rfloor) \mid i = 1,\dots,r\}$, to compute a dissimilarity score $\sigma(\mathcal{N},\mathcal{M}) = \sum_{(i,j)\in\mathcal{I}} |\mu_i - \lambda_j|$. The approach works under the assumption that the two shapes $\mathcal{M},\mathcal{N}$ have *the same scale*; however, in this challenge, the test shapes are randomly rescaled so that $\mathcal{N} \subseteq \alpha\mathcal{M}$, where $\alpha \in \mathbb{R}$ is *unknown*. In particular, this induces a transformation of the eigenvalues $\mu_i \mapsto \frac{1}{\alpha^2}\mu_i$.

In order to account for the unknown scale, the score $\sigma(\mathcal{N},\alpha\mathcal{M})$ is computed for a fixed range of scales $\alpha \in [10^{-1},\dots,10]$ sampled at $N = 10^3$ equally spaced values; then, the minimum score is kept for the given pair of shapes. We refer to Section 6 for further considerations on the lack of a consistent scale across the dataset.

4.3. Deep aggregation of localized statistical features (DLSF) [T. Furuya and R. Ohbuchi]

The algorithm aims at extracting 3D shape descriptors that are robust both against non-rigid deformations and noise, such as irregular holes on the surfaces. It is built upon previous studies on local 3D geometric features [OOFO12] and deep learning-based aggregation of local features [FO16]. Fig. 2 describes the processing pipeline of DLSF. DLSF takes as its input a set of low-level local 3D geometric features. The low-level feature is inherently invariant against geometric transformations including translation (1DOF), uniform scaling (1DOF), and rotation (3DOF) of 3D shapes. DLSF produces a compact, high-level feature per 3D model for efficient and effective matching among non-rigid 3D models. The algorithm consists of the following four steps.

Generating oriented point set. Given a polygon-based 3D model, it is first converted into a 3D oriented point set by sampling the surface using the algorithm by Osada et al. [OFCD02]. This sampling process gives DLSF a certain invariance against different shape representations. The algorithm randomly and uniformly samples points on the surface; each point is associated with the normal vector of the triangle on which the point is sampled. For each model, 3K oriented points are sampled. The oriented point set of the 3D model is uniformly scaled to fit a sphere having unit diameter.

Extracting local statistical features. A set of 100 Spheres-Of-Interest (SOIs) is sampled from the oriented point set of the 3D model. Position and radius of each SOI are chosen randomly. Each SOI is then described by Localized Statistical Feature (LSF) [OOFO12], which is formed as a 625-dimensional vector. The reasons for employing LSF as descriptor for SOIs are two-fold. First, LSF has invariance against 3D rotation of points within a local region, a desirable property for attaining robustness to non-rigid deformations. In the SHREC 2015 non-rigid 3D shape retrieval track [LZC*15], LSF, combined with an unsupervised local feature aggregation algorithm, is quite capable in comparing non-rigid 3D models. Secondly, LSF tends to be robust against geometrical and topological noise caused by irregular holes on the surface. A local feature similar to LSF [RBB09] has successfully been applied to the task of 3D registration, where 3D models generated by 3D range scanners present surface cracks and holes.

Aggregating local features. A deep neural network (DNN) is employed to aggregate the set of LSFs into a single feature per 3D model. The DNN consists of two blocks; (1) E-block, performing *Encoding* of LSF, and (2) AC-block, performing *Aggregation* of the encoded LSFs into a feature per 3D model, followed by *Compression* of the aggregated feature. The E-block comprises three fully-connected layers, each having 3072, 2048, and 512 neurons, respectively. The AC block pools, by averaging, the set of encoded mid-level local features into a single feature per 3D model. The subsequent three fully-connected layers, with their 1024, 512, and 128 neurons, compress the aggregated feature to produce a compact and salient feature per 3D model. ReLU is used as activation function for the fully-connected layers.

Comparing aggregated features. Finally, a compact (128-dimensional) aggregated feature of the query model is efficiently

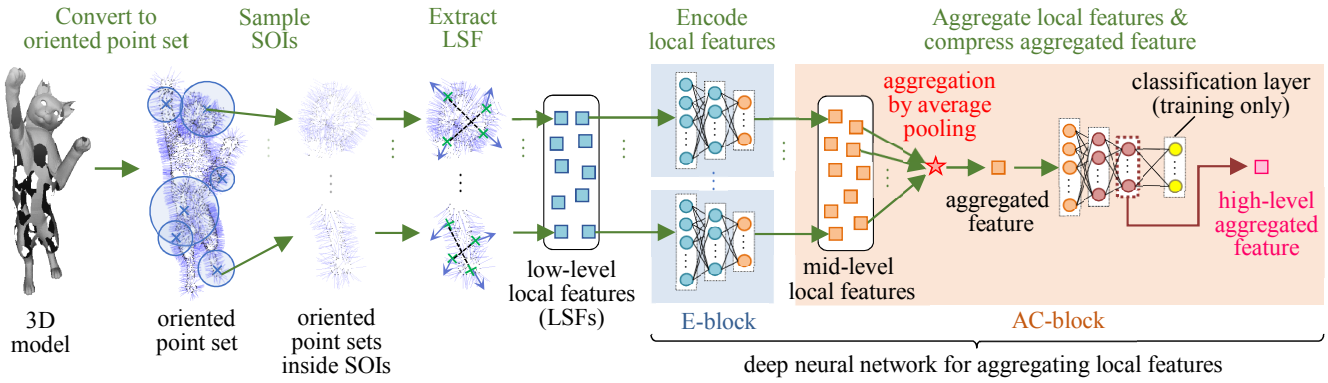


Figure 2: *Processing pipeline of DLSF (Section 4.3). The 3D model is represented as a set of low-level, local 3D geometric features having invariance against 3D rotation in the SOI. The deep neural network aggregates the local features to a compact and salient feature per 3D model for effective matching among non-rigid 3D models.*

compared against the aggregated features of the targets in the database. Cosine similarity is used for this step.

The DNN is trained with a two-step process. The E-block is pre-trained first by using a large set of labeled SOIs. Then, the entire network is trained by using a set of labeled 3D models.

Pre-training E-block. To learn expressive local feature better suited for accurate 3D shape retrieval, the E-block is pre-trained by using a large number of labeled SOIs so that it could predict object categories of the SOIs. To do so, SOIs are sampled at random position and scale from the labeled training 3D models. The label for each SOI is identical to the label for the 3D model from which the SOI is sampled. A total of 3M labeled SOIs are collected from the training set. The classification layer is appended at the output end of the E-block. Parameters in the E-block, i.e., connection weights among neurons, are randomly initialized and cross-entropy loss is minimized by using AdaGrad [DHS11] with initial learning rate equal to 0.1. A 20% dropout is performed for all the hidden layers. The pre-training is iterated for 10 epochs.

Training whole network. After supervised pre-training of the E-block, the entire network is trained including both the E-block and the AC-block by using the labeled 3D models from the training set. The parameters in E-block inherit as their initial values the result of the pre-training. The parameters in the AC-block are randomly initialized. Training of the entire DNN is done by minimizing cross-entropy loss by using AdaGrad with initial learning rate equal to 0.1 and 20% dropout. The training is iterated for 100 epochs.

DLSF has per-query runtime of 0.063s, of which 0.061s are needed for feature extraction, and 0.002s for comparison (measured on a machine with Intel Core i7-6700 CPU, Nvidia GeForce GTX 1080 GPU and 64GB DRAM).

4.4. Sparse reconstruction (SR) [L. Wan et al.]

Wan et al. [WZZ16] propose to measure the similarity between two shapes based on sparse reconstruction of shape descriptors. For each shape, its local descriptors and sparse dictionary are computed. The similarity between two shapes is then defined by the error incurred when reconstructing one's descriptor set using the other's dictionary. No training set is needed in the whole process.

In this method, the computation of the HKS [SOG09] descriptors is modified on the following aspects: (1) The descriptors are calculated on the largest connected component for a disconnected shape, while some descriptors of the boundary vertices and their 1-ring neighbors are excluded; (2) The diffusion time scales are adaptively set for each shape, rather than some fixed values. The modified descriptors are called I-HKS descriptors. To compute these descriptors for a shape, the eigenvalues $\lambda_0,\ldots,\lambda_{99}$ and the corresponding eigenfunctions $\phi_0,\ldots,\phi_{99}$ of the Laplace-Beltrami operator are used. After setting $t_{\min}=4\ln 10/\lambda_1$ and $t_{\max}=4\ln 10/\lambda_{99}$ according to [SOG09], the diffusion time scales are chosen from $t_{\text{start}}=t_{\min}$ to $t_{\text{end}}=t_{\min}+(t_{\max}-t_{\min})/10$. The time scales are then formulated as:

$$t_i = 10^{\lg t_{\text{start}} + \frac{\lg t_{\text{end}} - \lg t_{\text{start}}}{n-1}(i-1)}, \quad i=1,\ldots,n. \tag{4}$$

Next, sparse dictionary learning [MBPS10] is utilized to compute a dictionary for each shape. For a shape $\mathcal{S}_A$, after its I-HKS set $\boldsymbol{F}_A=\{\boldsymbol{f}_j^A \mid j=1,\ldots,N_A\}$ is computed, its dictionary $\boldsymbol{D}_A$ can be obtained by solving the constrained optimization problem

$$\tilde{\boldsymbol{D}}_A = \arg\min_{\boldsymbol{D}_A} \frac{1}{N_A}\sum_{j=1}^{N_A} \|\boldsymbol{f}_j^A - \boldsymbol{D}_A\boldsymbol{\gamma}_j^A\|_2^2 \quad \text{s.t. } \|\boldsymbol{\gamma}_j^A\|_0 \le T, \tag{5}$$

where $\boldsymbol{\gamma}_j^A$ consists of sparse coefficients and T is a sparsity threshold.

The shape similarity measure is slightly different to [WZZ16]. Given two shapes $\mathcal{S}_A$ and $\mathcal{S}_B$, if using $\mathcal{S}_A$'s dictionary $\boldsymbol{D}_A$ to sparsely code $\mathcal{S}_B$'s I-HKS $\boldsymbol{f}_j^B$, the reconstruction error is expressed as:

$$E(\boldsymbol{f}_j^B, \boldsymbol{D}_A) = \min \|\boldsymbol{f}_j^B - \boldsymbol{D}_A\boldsymbol{\gamma}_j^B\|_2^2 \quad \text{s.t. } \|\boldsymbol{\gamma}_j^B\|_0 \le T. \tag{6}$$

Therefore, the average reconstruction error for coding the I-HKS set $\boldsymbol{F}_B$ is formulated as:

$$E(\boldsymbol{F}_B, \boldsymbol{D}_A) = \frac{1}{N_B}\sum_{j=1}^{N_B} E(\boldsymbol{f}_j^B, \boldsymbol{D}_A). \tag{7}$$

Similarly, if using $\mathcal{S}_B$'s dictionary $\boldsymbol{D}_B$ to represent $\mathcal{S}_A$'s I-HKS set $\boldsymbol{F}_A$, the average reconstruction error is $E(\boldsymbol{F}_A, \boldsymbol{D}_B)$. The distance between $\mathcal{S}_A$ and $\mathcal{S}_B$ is finally defined as:

$$Dist(\mathcal{S}_A, \mathcal{S}_B) = \min(E(\boldsymbol{F}_B, \boldsymbol{D}_A), E(\boldsymbol{F}_A, \boldsymbol{D}_B)). \quad (8)$$

All the parameter settings are the same as [WZZ16].

This method assumes that a partial shape has a large connected component which still keeps most of the corresponding full shape. So, if a partial shape is broken into many pieces or misses too much surface information, this method may be not suitable.

4.5. Geodesic multi-resolution framework (GMR) [M. Masoumi et al.]

The framework of [MLH16, MH17] is based on the eigensystem of the Laplace-Beltrami operator, which is invariant to isometric transformations. A geodesic multi-resolution descriptor is defined by incorporating the vertex area into the definition of spectral graph wavelet [LB13] in a bid to capture more geometric information and, hence, further improve its discriminative ability. A Mexican hat wavelet is utilized as a generating kernel, which considers all frequencies equally important as opposed to the cubic spline kernel [LB13]. Furthermore, in order to capture the spatial relations between features, the geodesic multi-resolution descriptor is weighted by a geodesic exponential kernel.

The framework consists of four main steps. The first step is to represent each 3D shape in the dataset $\mathcal{D}$ by a spectral graph wavelet signature matrix $\mathbf{S}$, where $\mathbf{S} = (\mathbf{s}_1, \ldots, \mathbf{s}_m) \in \mathbb{R}^{p \times m}$, $\mathbf{s}_i$ is the p-dimensional local descriptor at vertex i, and m is the number of mesh vertices. In the second step, the area-weighted spectral graph wavelet signatures $\mathbf{s}_i$ are mapped to high-dimensional mid-level feature vectors using the soft-assignment coding step of the BoF model, resulting in a $k \times m$ matrix $\mathbf{U} = (\mathbf{u}_1, \ldots, \mathbf{u}_m)$ whose columns are the k-dimensional mid-level feature codes. In the third step, the $k \times k$ geodesic multi-resolution matrix $\mathbf{F}$ is computed using the mid-level feature codes matrix and a geodesic exponential kernel as:

$$\mathbf{F} = \mathbf{U}\mathbf{K}\mathbf{U}^{\intercal}, \quad (9)$$

where $\mathbf{U}$ is a $k \times m$ matrix of geodesic multi-resolution codes (i.e. mid-level features), and $\mathbf{K} = (\kappa_{ij})$ is a $m \times m$ geodesic exponential kernel matrix whose elements are given by

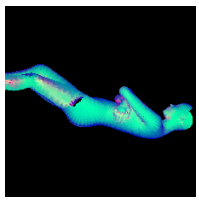
$$\kappa_{ij} = \exp\left(-\frac{d_{ij}^2}{\varepsilon}\right), \quad (10)$$

with d_{ij} denoting the geodesic distance between vertices v_i and v_j, and ε is a positive, carefully chosen parameter that determines the width of the kernel. Matrix $\mathbf{F}$ is reshaped into a k^2-dimensional global descriptor $\mathbf{x}_i$. In the fourth step, the geodesic multi-resolution vectors $\mathbf{x}_i$ of all n shapes in the dataset are arranged into a $k^2 \times n$ data matrix $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$. Finally, a query $\mathbf{x}$ is compared to all data points in $\mathbf{X}$ using the ℓ_1-distance. The lower the value of this distance, the more similar the shapes.

The experiments were conducted on a desktop computer with an Intel Core i5 3.10 GHz CPU and 8 GB RAM; the algorithms were implemented in MATLAB. In this setup, a total of 201 eigenvalues and associated eigenfunctions of the LBO were computed. The resolution parameter was set to $R = 2$ (i.e., the spectral graph wavelet signature matrix is of size $5 \times m$, where m is the number of vertices) and the kernel width to $\varepsilon = 0.1$. The parameter of the soft-assignment coding is computed as $\alpha = 1/(8\mu^2)$, where μ is the median size of the clusters in the vocabulary.

4.6. SnapNet [A. Boulch et al.]

The objective of the approach is to learn a classifier that will produce similar outputs for the same shapes with different poses. The training set is composed of 10 different classes. For a new shape, the classification vector obtained with this new shape is used as a description of the shape.



Training dataset. The training dataset is generated by taking snapshots around the 3D model (see inset) [Gra14]. In order to create visually consistent snapshots, the point cloud is remeshed using [MRB09]. The snapshots are 3-channel images: the first channel encodes the distance to the camera (i.e., depth map), the second is the normal orientation to the camera direction, and the third channel is an estimation of the local noise in the point cloud (ratio of eigenvalues of the neighborhood covariance matrix).

The adopted CNN is a VGG16 [SZ14] with a final fully-connected layer with 10 outputs. The weights are initialized with the model trained on the ILSVRC12 contest; the network is then fine-tuned using a step learning rate policy.

Distance computation. The classifier is then applied to images and produces images classification vectors $\mathbf{v}_{im}$. For each model, a prediction vector $\mathbf{V}_M$ is computed based on the images :

$$\mathbf{V}_M = \frac{\sum_{im \in M} \mathbf{v}_{im}}{||\sum_{im \in M} \mathbf{v}_{im}||_2}.$$

The distance matrix $\mathbf{X}$ contains the pairwise ℓ_2 distances between the $\mathbf{V}_M$. Each line is then normalized using a soft max :

$$\mathbf{X}_{i,j} = \frac{\exp(\mathbf{X}_{i,j})}{\sum_j \exp(\mathbf{X}_{i,j})}.$$

Note that matrix $\mathbf{X}$ is not symmetric. Finally, a symmetric distance matrix is defined $\mathbf{D} = \mathbf{X}^T\mathbf{X}$. The values of $\mathbf{D}$ are clipped according to the 5^{th} and 50^{th} percentiles and then re-scaled in $[0, 1]$.

The method was implemented Python and C++, using the deep learning framework *PyTorch*. The experiments were performed on a CPU Intel Xeon(R) E5-1620 3.50GHz. The training part was operated on a Nvidia Titan X Maxwell GPU and the test part (predictions) on a Nvidia GTX 1070. Generating the snapshot took around 10 seconds per model. The training took around 8 hours. The prediction vectors were generated in 2 seconds per model and the dissimilarity matrix was computed in less than 10s.

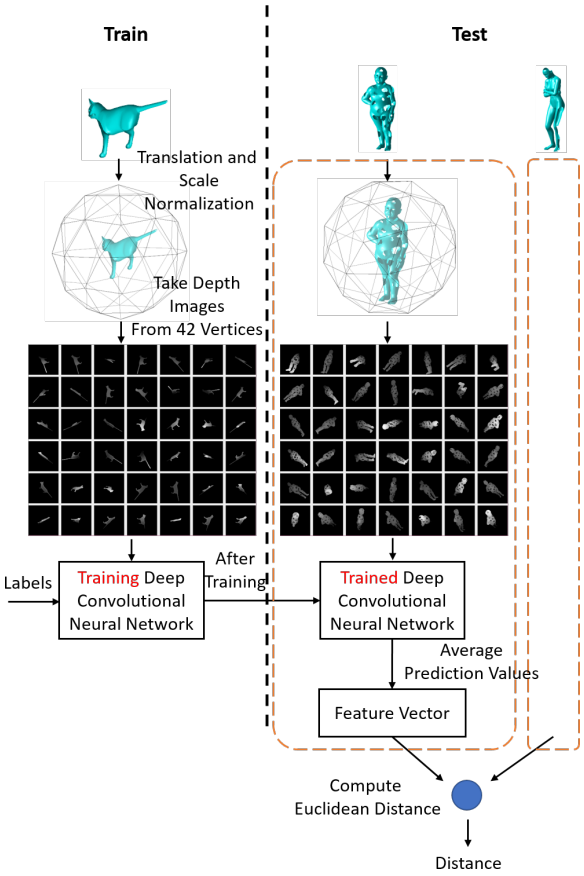


Figure 3: *Pipeline of the 2VDI-CNN method (Section 4.7).*

4.7. 2D-view depth image and CNN-based method (2VDI-CNN) [Y. Hong and J. Kim]

This method makes use of 2D-view depth images [PTK06, LGS10] and a convolutional neural network (CNN); Fig. 3 shows the general pipeline.

A given 3D shape is represented by 42 depth images which are used for training. In order to take depth images, the center (average of vertices coordinates) of the 3D shape is translated to the origin and the maximum polar distance of the vertices is scaled to one. Then, 42 depth images are taken on the vertices of a Pentakis icosidodecahedron (subdivided icosahedron), which is a convex polyhedron with 80 triangular faces and 42 vertices.

The training shapes are labeled to 10 categories, and each depth image is assigned the label of the corresponding class. These depth images and corresponding labels are put into a CNN for training. The chosen architecture is GoogLeNet [SLJ*15] (also referred to as Inception v1), the winning architecture on ImageNet 2014. Although there are other architectures that perform better than GoogLeNet, such as Inception v4 [SIVA16] and ResNet [HZRS16], the 4GB GPU used for the contest set a hardware limit on the type of architecture. GoogLeNet's last softmax layer was changed to output 10-way prediction values.

In the training step, a total of $1216 \times 42 = 51072$ ('holes') and $1082 \times 42 = 45444$ ('range') depth images (of size 224×224) were used as training data, where 10% of these were used for validation. The training used momentum with a decay of 0.9, learning rate of 0.01, and every epoch was decayed using an exponential rate of 0.96 with dropout ratio set to 0.1. All training data was zero-centered and then scaled by its standard deviation as a preprocessing step; batch size was set to 64. Training for 'holes' was stopped after 38 epochs, where the validation accuracy reached 84.49% with no further growth. Training for 'range' was stopped after 34 epochs, with 86.47% validation accuracy and no further growth.

In the test step, for a given 3D shape, a 10-dim. feature vector is extracted to represent the model. As in the training step, 42 depth images are taken from the 3D shape and input to the trained GoogLeNet, which outputs 10 prediction values per depth image, making for a total of 42×10 prediction values. These are averaged on each dimension to generate a 10-dim. vector, which is taken as the final feature vector for the test shape. The final score between two 3D shapes is given by the Euclidean distance of the corresponding feature vectors.

GoogLeNet was built by TFLearn (http://tflearn.org) on top of Tensorflow. Depth images were taken with MATLAB 3D Model Renderer (http://www.openu.ac.il/home/hassner/projects/poses/). The method was implemented on a 3.60 GHz i7-4790 CPU, 8GB DDR3 RAM, GeForce GTX 960 4GB rig. On this system, the average runtimes for taking depth images and generating the predictions for 10 shapes is 3.25s and 0.31s (on GPU) respectively. Calculating the similarity between a query and 882 shapes takes only 0.005s. Better performance could possibly be achieved by tuning the hyperparameters for training and by using better CNN architectures.

4.8. Non-parametric spectral model (NPSM) [A. Gasparetto et al.]

The method described in [GMT15] is a supervised approach for the construction of a generative model based on the spectral decomposition of the Laplace-Beltrami operator. The idea is to define a statistical framework that models a shape as two independent generative models for the eigenvector and the eigenvalue components of the spectral representation of the Laplacian. In particular, it is assumed that the spectral embedding space of the eigenvector part is a set of independent observations which follows an unknown distribution. The underlying distribution is estimated in a non-parametric way, through kernel density estimation. In particular, the posterior probability $P\left(\Phi^M | \Theta^\Phi\right)$ can be computed by solving the problem:

$$\max_{\mathcal{O}\in\mathbb{O}(d)} \max_{S\in\{\pm 1\}^d} (Nh^d)^{-n} \prod_{i=1}^{n} \sum_{j=1}^{N} e^{-\frac{\|\mathcal{O}S\phi_i^M - \theta_j^\Phi\|^2}{2h^2}}, \qquad (11)$$

where d is the embedding dimension. For the results produced for this contest, $d = 50$, hence the first 50 smallest eigenvalues (and corresponding eigenvectors) have been used to build the models and to infer them.

The two most important benefits brought by this point-wise ap-

proach regard the robustness (and somehow, the scalability) of the algorithm and its acquired invariance to vertex correspondences.

Two alignment steps are defined and applied to the eigenvector model in order to take care of the residual rotations between eigenvector matrices. The first one deals with the non-uniqueness of the retrieved eigenvectors, problem which often takes the name of sign ambiguity. A heuristic-based method is used to solve this problem, even if it does not guarantee the identification of all the sign flips that should be performed.

The second alignment step deals with the mixing-eigenspaces problem. The problem concerns the retrieval of eigenvectors that correspond to very close eigenvalues, which could result in a swap between them. To solve it, a rotation matrix which aligns the eigenvector matrices is introduced into the model computation process. Such rotation is computed by casting the problem as an Orthogonal Procrustes problem, in which the orthonormal transformation which maximizes a certain probability (defined in terms of Parzen-Rosenblatt kernel density estimator) is sought.

On the other hand, in order to define a descriptor that is robust to small non-isometric perturbations, the eigenvalues are assumed to be log-normally distributed:

$$P(\Lambda^M|\Theta^\Lambda) = (2\pi)^{\frac{d}{2}} \prod_{i=1}^{d} \frac{1}{\lambda_i \sigma_i} e^{\frac{-(\ln\lambda_i - \mu_i)^2}{2\sigma_i^2}} . \tag{12}$$

Finally, the posterior probability of a mesh with respect to a certain class is computed as the combination of posterior probabilities of both models as follows:

$$P(M|\Theta) = P(\Lambda^M|\Theta^\Lambda) P(\Phi^M|\Theta^\Phi) . \tag{13}$$

The posterior probabilities of a mesh with respect to each class of the training set are treated as components of a feature vector which characterizes a mesh. The distance (in this contest, Chebyshev's distance) between the feature vectors of two meshes represents the score between them.

4.9. A statistical model of Riemannian metric variation (RMVM) [A. Gasparetto and A. Torsello]

The method proposed in [GT15] consists in a supervised technique to learn a statistical model build on the Riemannian metric variations on deformable shapes based on the spectral decomposition of the Laplace-Beltrami operator. Similarly to NPSM (Section 4.8), the method employs a statistical framework that models a shape as two independent models for the eigenvectors and for the eigenvalues. The eigenvector matrices of a set of discrete representations (i.e., meshes representing the shape in different poses) are assumed to be points on the manifold of special orthogonal matrices $\mathcal{SO}(n)$. Here the model is assumed to follow a Γ-distribution over the manifold geodesic distances from a manifold centroid Φ_0

$$d_g(\phi, \phi_0) \approx 2n - Tr\left(\phi^T \phi_0\right) + O(\Theta_i^4) , \tag{14}$$

where Θ_i are the angles of the residual rotation $\phi^T \phi_0$.

The eigenvalues are assumed to be log-normally distributed for the same stability considerations presented by Aubry et al. [ASC11]. The shape centroid is computed as follows:

$$\underset{\phi_0, \mathcal{R}_i \in \mathcal{SO}(p)}{\operatorname{argmax}} \quad \operatorname{Tr}\left(\sum_i^N \mathcal{R}_i \phi_i^T \phi_0\right) , \tag{15}$$

where the rotation matrix $\mathcal{R}_i$ is introduced in order to align the eigenvectors of the Laplacian of a mesh i, since its embedding is defined up to an isometry. This is solved by separately optimizing for ϕ_0 and $\mathcal{R}_i$ in an iterative process

$$\Phi_0 = \underset{\Phi_0 \in \mathcal{O}(n)}{\operatorname{argmax}} \quad \operatorname{Tr}\left(\left(\sum_i^N \mathcal{R}_i \Phi_i^T\right) \Phi_0\right) \tag{16}$$

$$\mathcal{R}_i = \underset{\mathcal{R}_i \in \mathcal{SO}(n)}{\operatorname{argmax}} \quad \operatorname{Tr}\left(\left(\sum_i^N \Phi_i^T \Phi_0\right) \mathcal{R}_i\right) \tag{17}$$

where both optimizations can be solved exactly through Singular Value Decomposition.

The two statistical models are combined to compute the posterior probability of a mesh to belong to a certain class:

$$p(j) = \left(\prod_i \log \mathcal{N}(x_{ij}; \mu_i, \sigma_i)\right) \Gamma(k, \theta)(g_{d_j}) . \tag{18}$$

The main drawback of this method concerns the manifold in which the eigenvector matrices lie. Indeed, to compute the eigenvector centroid, the eigenvector matrices must share a common intrinsic space. Hence, the meshes are assumed to have both the same number of vertices and a vertex-to-vertex correspondence (at least for the training set). Unfortunately, this is not the case for this contest. The issue is addressed by (sub)sampling the meshes to a common number of vertices, and approximating the correspondence map by casting the problem as an assignment problem. The same process, which is explained in detail in the original work [GT15], is applied to the meshes to be classified.

From a practical point of view, the embedding dimension d is set to 50, i.e., the 50 smallest eigenvalues (and corresponding eigenvectors) are used to build the models and to infer them. The score between two meshes is computed as the Euclidean (RMVM-Euc) and Spearman's (RMVM-Spe) distance between the score vectors whose components are the probability density computed with respect to each class of the training set.

4.10. Bag of Words framework with RoPS (BoW+RoPS) [M-T. Tran et al.]

In this method, the similarity of two query objects q_i and q_j is not measured directly, but via an intermediate domain consisting of the training set. Similar objects are retrieved per-query from the training set; then, the two queries are compared indirectly (see Fig. 4).

Let $R(q_i)$ and $R(q_j)$ be the rank lists of training objects corresponding to the query objects q_i and q_j, respectively. Let $n_K(q,c)$ be the number of training objects in category c appearing among the top K of rank list $R(q)$. Define $s_K(q,c) = n_K(q,c)/K$ to represent the score of a query object q being in category c. This way, each query q is encoded as a normalized 20-dim. vector $\mathbf{v}(q) = (s_K(q,1), s_K(q,2)..., s_K(q,20))$. To achieve high precision, only the top 1 or 2 best score values are kept in the feature vector

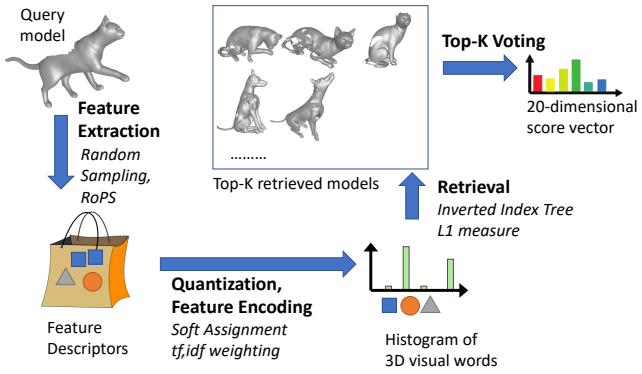


Figure 4: *Encoding a 3D object using BoW (Section 4.10).*

$\mathbf{v}(q)$; other elements in $\mathbf{v}(q)$ are zeroed. Finally, the similarity of q_i and q_j is defined as the dot product of $\mathbf{v}(q_i)$ and $\mathbf{v}(q_j)$.

The framework for 3D object retrieval [PSA*16] is further enriched using RoPS features [GSB*13] to retrieve similar objects in the training set corresponding to a query object. The method is based on the Bag-of-Words (BoW) scheme for visual object retrieval [SZ03, NNT*15].

First, each 3D object is normalized to fit in a unit cube. Then, uniform random samples of $5\% \leq p_{Sampling} \leq 20\%$ of the total number of vertices are taken for each object. RoPS [GSB*13] is used as a local feature, restricted to a Euclidean sphere with radius $r \in [0.01, 0.2]$ around each sampled vertex.

The extracted features are used to build a codebook with size equal to 10% of the total number of features in the corpus, using Approximate K-Means. Soft-assignment [PCI*08] with 3 nearest neighbors is used to reduce quantization error. As a query can be part of a training object and vice versa, an ℓ_1 asymmetric distance [ZJS13] is used to measure the dissimilarity of each pair of objects.

For the 'holes' challenge: since shapes have irregular mesh, they are subdivided to reduce significant difference in vertex density between different parts of the surface. A random sampling $p_{Sampling} = 10\%$ and support radius $r = 0.05$ are used for RoPS, and the codebook size is set to 180K. Three runs are provided:

- Run1: The top $K = 1$ in the rank list is considered.
- Run2: The top $K = 9$ are considered, with a voting scheme for the best category.
- Run3: The top $K = 9$ are considered, with a voting scheme for the two best categories.

For the 'range' challenge: topological noise is removed by deleting triangles with abnormally long edge. Random sampling $p_{Sampling} = 10\%$ is used. Due to low mesh quality, a support radius $r = 0.1$ is used for RoPS. Since the grid-like topology induces a large amount of vertices, the codebook size is increased to 1.2M. Three runs are provided as above, with Run2 and Run3 using $K = 5$ instead of $K = 9$.

The codebook training module was implemented in Python 2.7 and run on a 2.4 GHz Intel Xeon CPU E5-2620 v3, 64 GB RAM. The 3D feature extraction and description module was written in C++ and run on a 2GHz Intel Xeon CPU E5-2620, 1GB RAM. The retrieval process was written in Matlab R2012b, where feature quantization and distance calculation were performed on a 2.2GHz Intel Xeon CPU E5-2660, 12 GB RAM. The average time to calculate model features is 1-2s, whereas it takes on average 0.1s to compare a test object against the entire training set.

4.11. Supervised bag-of-features (SBoF) [R. Litman et al.]

This is an implementation of the method proposed in [LBBC14], with some slight modifications made specifically for this challenge. The first and major change is the selection of point descriptors, where the original intrinsic descriptors were replaced by the extrinsic SHOT [STDS14] descriptor, due to its good performance in other recent challenges dealing with partial shapes [RCB*16, CRB*16]. The second change is due to the higher dimensionality of the SHOT descriptor, where a bigger dictionary of size 256 is adopted as opposed to the original 32.

The rest of the hyper-parameters were selected in a manner similar to [PSR*14], by cross validation over the training set. First, as a pre-processing step, all shapes from the 'holes' set were downsampled to 15K faces. Sparse coding was done over unit ℓ_2 norm SHOT descriptors, with regularization value set to $\lambda = 0.25$. Each shape's sparse codes were pooled into a single histogram using average-pooling, and compared using the ℓ_1 metric. The triplet loss margin was selected to be $\mu = 0.2$.

The method was implemented in MATLAB, with parts in C++ for the computation of SHOT features. Dictionary learning took about 5h on a 12-core 2.4GHz XEON processor with 48GB of memory. At test time, shapes were ranked according the the ℓ_1 similarity of their 256 histogram descriptor, which took less than 100ms.

5. Results

Precision/Recall curves for all methods on the two challenges are shown in Fig. 5; NN, 1-Tier, 2-Tier and DCG scores are reported in Tables 1 and 2.

Reading the P/R curves, the best performing method for the 'holes' challenge is DLSF (Section 4.3), followed closely by BoW+RoPS (Section 4.10). In terms of NN, the best performing methods are DLSF, 2VDI-CNN (Section 4.7), and SBoF (Section 4.11). DLSF, 2VDI-CNN, and BoW+RoPS also exhibit the best scores in the 1-Tier, 2-Tier, and DCG measures.

The 'range' challenge received less submissions (9 against the 15 for 'holes', baselines included). In terms of P/R, the best performing methods are BoW+RoPS and 2VDI-CNN, which also receive top scores in the 1-Tier, 2-Tier, and DCG measures. The highest NN scores are attained again by 2VDI-CNN and SBoF. Overall, we note a consistent behavior for methods that participated in both challenges. In particular, SBoF has a higher NN score than BoW+RoPS in both challenges, despite the latter having a significant advantage in terms of P/R.

Method	NN	1-Tier	2-Tier	DCG
2VDI-CNN	**0.969**	0.906	**0.977**	**0.980**
SBoF	0.811	0.317	0.510	0.769
BoW+RoPS-2	0.643	0.910	0.962	0.962
BoW+RoPS-3	0.639	0.908	0.964	0.965
BoW-HKS (baseline)	0.519	0.326	0.537	0.736
BoW+RoPS-1	0.515	**0.915**	0.959	0.960
BoW-siHKS (baseline)	0.377	0.268	0.485	0.699
GMR	0.178	0.184	0.371	0.640
DNA (baseline)	0.130	0.183	0.366	0.640

Table 1: *Retrieval accuracy for the* 'range' *challenge, sorted decreasingly by NN score. Best results are reported in bold.*

Method	NN	1-Tier	2-Tier	DCG
DLSF	**1.000**	**0.971**	**0.999**	**0.998**
2VDI-CNN	0.906	0.818	0.937	0.954
SBoF	0.815	0.326	0.494	0.780
BoW-siHKS (baseline)	0.710	0.370	0.566	0.790
BoW+RoPS-3	0.607	0.918	0.970	0.968
BoW+RoPS-1	0.597	0.877	0.963	0.956
BoW-HKS (baseline)	0.578	0.261	0.436	0.725
RMVM-Euc	0.392	0.226	0.402	0.679
BoW+RoPS-2	0.380	0.894	0.965	0.955
NPSM	0.347	0.222	0.395	0.676
RMVM-Spe	0.251	0.228	0.410	0.676
SR	0.241	0.225	0.395	0.676
GMR	0.186	0.172	0.343	0.642
SnapNet	0.117	0.172	0.349	0.641
DNA (baseline)	0.078	0.163	0.348	0.632

Table 2: *Retrieval accuracy for the* 'holes' *challenge, sorted decreasingly by NN score. Best results are reported in bold.*

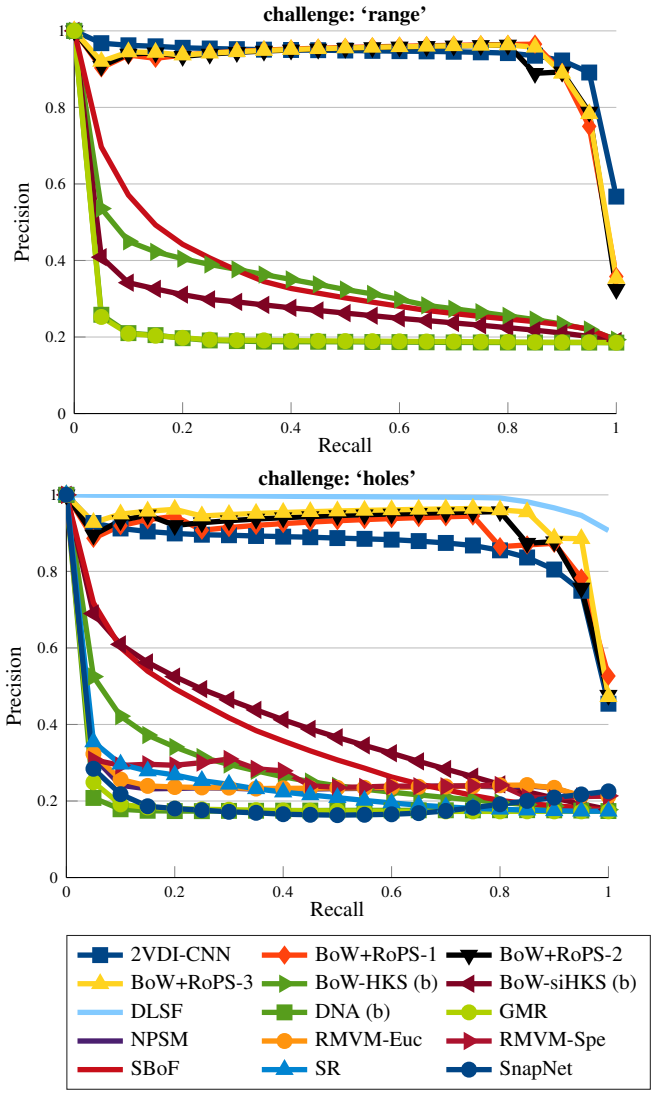


Figure 5: *Precision/Recall curves for all methods in the two challenges. In the legend, '(b)' identifies the baselines.*

6. Discussion and conclusions

Out of the 11 participating methods (without counting different runs), 6 were supervised learning techniques and the remaining 5 were non learning-based. Despite coming from separate research groups, we note a surprising consistency in the general approach: the adoption of BoW-like frameworks on one hand, and the use of synthetic 2D views on the other. In both cases, we observe a wide variance in the accuracy among the different submissions, suggesting that the old adagio "the devil is in the details" is as valid for learning-based approaches as it is for axiomatic modeling. At the same time, we did not observe a significant difficulty bias across different shape categories or deformations.

Given the non-rigid nature of the dataset, it is interesting to note that the best performing methods only rely on extrinsic, *not* deformation-invariant quantities. We conjecture that this is in part due to the piecewise-rigid transformations, and the presence of similar shapes in the training and test sets. In particular, the only purely intrinsic spectral approach (DNA) achieved the worst scores in both challenges. We further note the lack of intrinsic CNN-based methods such as [BMRB16].

Finally, a notable difficulty is the lack of a consistent scale in the 'holes' dataset, due to the random similarity transformation being applied to each shape. This has a detrimental effect on all methods relying on spectral (Laplacian-based) quantities and heat diffusion. Defining local, deformation- and scale-invariant features remains an open challenge tackled by few.

References

[ASC11] AUBRY M., SCHLICKEWEI U., CREMERS D.: The wave kernel signature: A quantum mechanical approach to shape analysis. In *Proc. CVPR Workshops* (2011), pp. 1626–1633. 7

[BBC*10] BRONSTEIN A. M., BRONSTEIN M. M., CASTELLANI U., ET AL.: SHREC 2010: Robust large-scale shape retrieval benchmark. In *Proc. 3DOR* (2010). 2

[BBK08] BRONSTEIN A., BRONSTEIN M., KIMMEL R.: *Numerical Geometry of Non-Rigid Shapes*. Springer, 2008. 2

[BMRB16] BOSCAINI D., MASCI J., RODOLÀ E., BRONSTEIN M. M.:

Learning shape correspondence with anisotropic convolutional neural networks. In *Proc. NIPS* (2016), pp. 3189–3197. 9

[CRA*16] COSMO L., RODOLÀ E., ALBARELLI A., MÉMOLI F., CREMERS D.: Consistent partial matching of shape collections via sparse modeling. *Computer Graphics Forum* (2016). 1

[CRB*16] COSMO L., RODOLÀ E., BRONSTEIN M. M., ET AL.: SHREC'16: Partial matching of deformable shapes. In *Proc. 3DOR* (2016). 1, 8

[CRM*16] COSMO L., RODOLÀ E., MASCI J., TORSELLO A., BRONSTEIN M.: Matching deformable objects in clutter. In *Proc. 3DV* (2016), pp. 1–10. 1

[DHS11] DUCHI J., HAZAN E., SINGER Y.: Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res. 12* (2011), 2121–2159. 4

[DLL*10] DEY T., LI K., LUO C., RANJAN P., SAFA I., WANG Y.: Persistent heat signature for pose-oblivious matching of incomplete models. *Computer Graphics Forum 29*, 5 (2010), 1545–1554. 1

[FO16] FURUYA T., OHBUCHI R.: Deep aggregation of local 3d geometric features for 3d model retrieval. In *Proc. BMVC* (2016). 3

[GH97] GARLAND M., HECKBERT P. S.: Surface simplification using quadric error metrics. In *Proc. SIGGRAPH* (1997), pp. 209–216. 2

[GMT15] GASPARETTO A., MINELLO G., TORSELLO A.: Non-parametric spectral model for shape retrieval. In *Proc. 3DV* (2015), pp. 344–352. 6

[Gra14] GRAHAM B.: Spatially-sparse convolutional neural networks. *CoRR abs/1409.6070* (2014). 5

[GSB*13] GUO Y., SOHEL F. A., BENNAMOUN M., LU M., WAN J.: Rotational projection statistics for 3D local surface description and object recognition. *IJCV 105*, 1 (2013), 63–86. 8

[GT15] GASPARETTO A., TORSELLO A.: A statistical model of riemannian metric variation for deformable shape analysis. In *Proc. CVPR* (2015), pp. 1219–1228. 7

[HZRS16] HE K., ZHANG X., REN S., SUN J.: Deep residual learning for image recognition. In *Proc. CVPR* (2016), pp. 770–778. 6

[KB10] KOKKINOS I., BRONSTEIN M. M.: Scale-invariant heat kernel signatures for non-rigid shape recognition. pp. 1704–1711. 2

[LB13] LI C., BEN HAMZA A.: A multiresolution descriptor for deformable 3D shape retrieval. *The Visual Computer 29* (2013), 513–524. 5

[LBBC14] LITMAN R., BRONSTEIN A. M., BRONSTEIN M. M., CASTELLANI U.: Supervised learning of bag-of-features shape descriptors using sparse coding. In *Computer Graphics Forum* (2014), vol. 33, pp. 127–136. 8

[LGS10] LIAN Z., GODIL A., SUN X.: Visual similarity based 3d shape retrieval using bag-of-features. In *Proc. SMI* (2010), pp. 25–36. 6

[LRB*16a] LÄHNER Z., RODOLÀ E., BRONSTEIN M. M., ET AL.: SHREC'16: Matching of deformable shapes with topological noise. In *Proc. 3DOR* (2016). 1, 2

[LRB*16b] LITANY O., RODOLÀ E., BRONSTEIN A. M., BRONSTEIN M. M., CREMERS D.: Non-rigid puzzles. *Computer Graphics Forum 35*, 5 (2016), 135–143. 3

[LZC*15] LIAN Z., ZHANG J., CHOI S., ET AL.: SHREC'15 track: Non-rigid 3d shape retrieval. In *Proc. 3DOR* (2015). 2, 3

[MBPS10] MAIRAL J., BACH F., PONCE J., SAPIRO G.: Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research 11* (2010), 19–60. 4

[MH17] MASOUMI M., HAMZA A. B.: Spectral shape classification: A deep learning approach. *Journal of Visual Communication and Image Representation 43* (2017), 198–211. 5

[MLH16] MASOUMI M., LI C., HAMZA A. B.: A spectral graph wavelet approach for nonrigid 3d shape retrieval. *Pattern Recognition Letters 83* (2016), 339–348. 5

[MPB07] MARINI S., PARABOSCHI L., BIASOTTI S.: Shape retrieval contest 2007: Partial matching track. In *Proc. SMI* (2007). 2

[MRB09] MARTON Z. C., RUSU R. B., BEETZ M.: On Fast Surface Reconstruction Methods for Large and Noisy Datasets. In *Proc. ICRA* (2009). 5

[NNT*15] NGUYEN V., NGO T. D., TRAN M., LE D., DUONG D. A.: A combination of spatial pyramid and inverted index for large-scale image retrieval. *International Journal of Multimedia Data Engineering and Management 6*, 2 (2015), 37–51. 8

[OFCD02] OSADA R., FUNKHOUSER T., CHAZELLE B., DOBKIN D.: Shape distributions. *ACM Trans. Graph. 21*, 4 (2002), 807–832. 3

[OOFO12] OHKITA Y., OHISHI Y., FURUYA T., OHBUCHI R.: Non-rigid 3d model retrieval using set of local statistical features. In *Proc. ICME Workshops* (2012), pp. 593–598. 3

[Pat13] PATANÉ G.: wFEM heat kernel: Discretization and applications to shape analysis and retrieval. *Comput. Aided Geom. Des. 30*, 3 (2013), 276–295. 3

[PCI*08] PHILBIN J., CHUM O., ISARD M., SIVIC J., ZISSERMAN A.: Lost in quantization: Improving particular object retrieval in large scale image databases. In *Proc. CVPR* (2008). 8

[PSA*16] PRATIKAKIS I., SAVELONAS M. A., ARNAOUTOGLOU F., ET AL.: SHREC'16 track: Partial shape queries for 3d object retrieval. In *Proc. 3DOR* (2016). 1, 8

[PSR*14] PICKUP D., SUN X., ROSIN P. L., ET AL.: Shrec'14 track: Shape retrieval of non-rigid 3d human models. In *Proc. 3DOR* (2014). 8

[PTK06] PASSALIS G., THEOHARIS T., KAKADIARIS I. A.: Ptk: A novel depth buffer-based shape descriptor for three-dimensional object retrieval. *Vis. Comput. 23*, 1 (2006), 5–14. 6

[RBB09] RUSU R. B., BLODOW N., BEETZ M.: Fast point feature histograms (fpfh) for 3d registration. In *Proc. ICRA* (2009), pp. 3212–3217. 3

[RCB*16] RODOLÀ E., COSMO L., BRONSTEIN M. M., TORSELLO A., CREMERS D.: Partial functional correspondence. *Computer Graphics Forum* (2016). 3, 8

[RWP06] REUTER M., WOLTER F.-E., PEINECKE N.: Laplace-Beltrami spectra as 'shape-DNA' of surfaces and solids. *Comput. Aided Design 38*, 4 (2006), 342–366. 3

[SIVA16] SZEGEDY C., IOFFE S., VANHOUCKE V., ALEMI A.: Inception-v4, inception-resnet and the impact of residual connections on learning. *CoRR abs/1602.07261* (2016). 6

[SLJ*15] SZEGEDY C., LIU W., JIA Y., SERMANET P., REED S., ANGUELOV D., ERHAN D., VANHOUCKE V., RABINOVICH A.: Going deeper with convolutions. In *Proc. CVPR* (2015), pp. 1–9. 6

[SMKF04] SHILANE P., MIN P., KAZHDAN M., FUNKHOUSER T.: The princeton shape benchmark. In *Proc. SMI* (2004). 2

[SOG09] SUN J., OVSJANIKOV M., GUIBAS L.: A concise and provably informative multi-scale signature based on heat diffusion. *Computer Graphics Forum 28*, 5 (2009), 1383–1392. 2, 3, 4

[STDS14] SALTI S., TOMBARI F., DI STEFANO L.: Shot: Unique signatures of histograms for surface and texture description. *Computer Vision and Image Understanding 125* (2014), 251–264. 8

[SZ03] SIVIC J., ZISSERMAN A.: Video google: A text retrieval approach to object matching in videos. In *Proc. ICCV* (2003), pp. 1470–1477. 8

[SZ14] SIMONYAN K., ZISSERMAN A.: Very deep convolutional networks for large-scale image recognition. *CoRR abs/1409.1556* (2014). 5

[WZZ16] WAN L., ZOU C., ZHANG H.: Full and partial shape similarity through sparse descriptor reconstruction. *The Visual Computer* (2016), 1–13. 1, 4, 5

[ZJS13] ZHU C., JEGOU H., SATOH S.: Query-adaptive asymmetrical dissimilarities for visual object retrieval. In *Proc. ICCV* (2013), pp. 1705–1712. 8