Eurographics Workshop on 3D Object Retrieval (2013)
S. Biasotti, I. Pratikakis, U. Castellani, T. Schreck, A. Godil, and R. Veltkamp (Editors)

SHREC'13 Track: Retrieval on Textured 3D Models

A. Cerri[†1], S. Biasotti[†1], M. Abdelrahman[2], J. Angulo[3], K. Berger[4], L. Chevallier[5], M. El-Melegy[2], A. Farag[2], F. Lefebvre[5], A. Giachetti[6], H. Guermoud[5], Y.-J. Liu[7], S. Velasco-Forero[8], JR. Vigouroux[5], C.-X. Xu[7], J.-B. Zhang[7]

[1]Istituto di Matematica Applicata e Tecnologie Informatiche "E. Magenes", CNR, Italy
[2]Computer Vision and Image Processing Laboratory (CVIP Lab), University of Louisville, KY, USA
[3]CMM-Centre de Morphologie Mathématique, Mathématiques et Systèmes, MINES ParisTech, France
[4]Oxford e-Research centre, Oxford, UK
[5]technicolor Research & Innovation, Cesson Sévigné, France
[6]Dipartimento di Informatica, Università di Verona, Italy
[7]Department of Computer Science and Technology, Tsinghua University, the People's Republic of China
[8]ITWM Fraunhofer Institute, Germany

Abstract

This contribution reports the results of the SHREC 2013 track: Retrieval on Textured 3D Models, whose goal is to evaluate the performance of retrieval algorithms when models vary either by geometric shape or texture, or both. The collection to search in is made of 240 textured mesh models, divided into 10 classes. Each model has been used in turn as a query against the remaining part of the database. For a given query, the goal was to retrieve the most similar objects. The track saw six participants and the submission of eleven runs.

Categories and Subject Descriptors (according to ACM CCS): H.3.1 [Information Storage and Retrieval]: Content Analysis and Indexing—Abstracting methods;

Introduction

The aim of SHREC is to evaluate the performance of existing 3D shape retrieval algorithms, by highlighting their strengths and weaknesses, using a common test collection allowing for a direct comparison of methods. In this report the results of the SHREC 2013 track: *Retrieval on Textured 3D Models* are presented. The aim of this track of SHREC'13 is to evaluate the performance of retrieval algorithms when models vary either by geometric shape or texture, or both. The novelty of this track is both in the use of textured 3D models and in the choice of deformations of textures and shapes. For each model different textures are considered, while shape perturbations include geometric noise, non-rigid and non-isometric shape deformations.

1. Data Collection and Queries

The dataset is made of 240 watertight mesh models, grouped in ten classes. Each class contains six null models, corresponding to two base meshes endowed with three different textures. Then, four transformations are applied to each null shape, including one non-rigid deformation, two non-metric-preserving deformations and one additive Gaussian noise perturbation. An example of the considered textures and geometric deformations is given in Figure 1.

The whole dataset is pre-classified using a two-level ground truth: if two models share both shape and texture they are highly similar; if they share only shape they are marginally similar; otherwise, they are dissimilar. For the present track, each model has been used in turn as a query against the remaining part of the database, with the goal of retrieving the most similar objects.

2. Participants

Each participant was asked to submit up to 3 runs of her/his algorithm, in the form of dissimilarity matrices: The entry (i, j) of each matrix represents the distance between models i and j in the associated run. Each run could be either the result of a different setting of parameters or the use of a dif-

† Organizer of the track. Dataset and evaluation measures are available at http://wwww.ge.imati.cnr.it/shrec13.

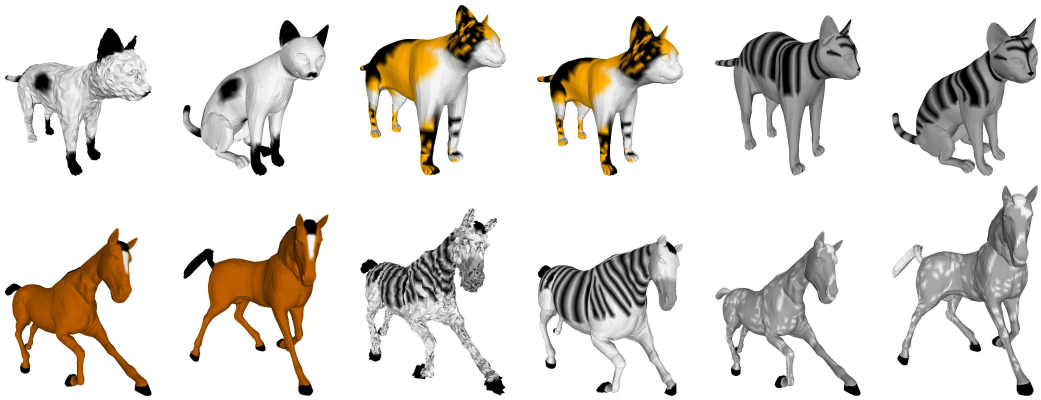

Figure 1: *Some members of the four leg animal class of the proposed dataset. Null models have been processed using MeshLab [Vis] and Remesh [AF06].*

ferent similarity measure. Six groups took part in this track, for a total of 11 submitted runs. More precisely:

1. M. Abdelrahman, M. El-Melegy and A. Farag from the University of Louisville (USA) participated with two runs (**A1** and **A2**). Their method is detailed in Section 3.1;
2. K. Berger, from the Oxford e-Research centre (UK) participated with one run (**Be**). His method is detailed in Section 3.2;
3. J.-B. Zhang, C.-X. Xu and Y.-J. Liu from the Tsinghua University (the People's Republic of China) participated with one run (**Zh**). Their method is detailed in Section 3.3;
4. A. Giachetti from the University of Verona (Italy), participated with one run (**Gi**). His method is detailed in Section 3.4;
5. H. Guermoud, F. Lefebvre, L. Chevallier and JR. Vigouroux from the technicolor Research & Innovation in Cesson Sévigné (France) participated with three runs (**G1**, **G2** and **G3**). Their method is detailed in Section 3.5;
6. S. Velasco-Forero (ITWM Fraunhofer Institute, Germany) and J. Angulo (CMM-Centre de Morphologie Mathématique, Mathématiques et Systèmes, MINES ParisTech, France) participated with three runs (**V1**, **V2** and **V3**). Their method is detailed in Section 3.6.

3. Description of the methods

In this section we describe the methods which were used by the participants in their runs.

3.1. Textured 3D models Classification using Scale Invariant Heat Kernels

We consider the 3D models in the dataset as deformable objects. Modeling these non-rigid shapes is a very challenging problem. It needs more work to compensate for the degrees of freedom resulting from local deformations. Reuter et al. [RWSN09] used the Laplacian spectra as intrinsic shape descriptors, and they employed the Laplace-Beltrami spectra as "shape-DNA" or a numerical fingerprint of any 2D or 3D manifold (surface or solid). They proved that shape-DNA is an isometry-invariant shape descriptor. Recently, Sun et al. [SOG09] proposed heat kernel signatures (HKSs) as a deformation-invariant descriptors based on diffusion of multi-scale heat kernels. HKS is a point based signature satisfying many of the good descriptor properties, but suffers from sensitivity to scale. Bronstein et al. [BK10] solved the HKS scale problem through a series of transformations. The same research group has recently introduced the Shape Google approach [BBGO11] based on the scale-invariant HKS. The idea is to use HKS at all points of a shape, or alternatively at some shape feature points, to represent the shape by a Bag of Features (BoF) vector. Sparsity in the time domain is enforced by pre-selecting some values of the time. In this work we present an approach for shape matching and retrieval based on scale invariant heat kernel signatures (SI-HKS). Sun et al. [SOG09] proposed to use the HKS as a local shape descriptor

$$h(x,t) = H_t(x) = \sum_{i=1}^{\infty} e^{-\lambda_i t} \varphi_i^2(x), \qquad (1)$$

with λ_i and φ_i the eigenvalues and eigenfunctions of the Laplace-Beltrami operator, respectively. The HKS has several desired properties [SOG09]: It is intrinsic and thus isometry-invariant (two isometric shapes have equal HKS), multi-scale and thus captures both local features and global shape structure. Also, it is informative: Under mild conditions, if two shapes have equal heat kernel signatures, they are isometric. The descriptor proposed in this work is based on BoF representation of the HKS in frequency domain combined with the first 15 normalized eigenvalues of the Laplace-Beltrami operator. We propose a novel method to achieve scale-invariance of HK which is shown to be noise-

robust. Scale invariance is a desirable property of the shape descriptor, which can be achieved by many ways. We propose a novel local scale normalization method based on simple operations. It was shown [BBGO11] that scaling a shape by a factor β results in changing $h(x,t)$ to $\beta^2 h(x,\beta^2 t)$. We propose to apply the Fourier transform (Ft) directly. Taking the amplitude of the Ft, the effect of the multiplicative constant β^2 is eliminated by normalizing $|H'(w)|$ by the sum of the amplitudes of the FT components. The amplitudes of the first significant Ft components (we normally use 20) are employed to construct the scale-invariant shape descriptor. This proposed method eliminates the scale effect without having to use the noise-sensitive derivative operation or the logarithmic transformation that both were used in [BBGO11]. Thus our method is simpler, more computational-efficient and more robust to noise. Eventually we use the L_1-norm for classification. We use the first non-trivial Laplace-Beltrami eigenfunction to detect a small number of sparse critical points on the shape surface. These points are robust to the shape class, and their number can itself be used as one of the discriminatory features among the various classes. We use the number of these points to re-arrange the retrieval results. Then we used the color distribution as a third step to re-arrange the results to get similar texture object first.

3.2. An algorithm for spatial and texture retrieval

Description. The key idea is to search for similarities both in the spatial and the texture domain. A special challenge is posed by the fact that the texture is not trivially defined as a 2D map but rather defined as a vertex coloring in a closed 2D mesh embedded in 3D.

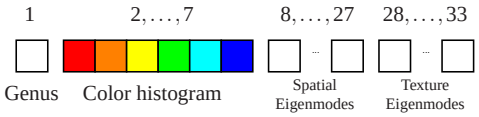

Figure 2: *The feature descriptor consists of* 33 *entries, the first entry depicting the genus, the next 6 entries depicting the hue histogram, the following 20 entries depicting the geometric eigenmodes and the last 6 entries depicting the texture eigenmodes.*

Thus we propose to classify each mesh with a feature descriptor of length $n = 1+6+20+6 = 33$. The first entry simply describes the genus of the mesh, to help discriminating between fundamentally different objects, like a chair (genus 3), a vase with a handle (genus 1) or animals (genus 0). We compute it with

$$\frac{(\#vertices + \#faces - 3/2 * \#faces) - 2}{2} \qquad (2)$$

Next, the entries 2-7 denote the hue histogram of vertex colors. We therefore convert the list of vertex colors from RGB space to HSV space and extract the first channel. It appears reasonable to decide for 6 bins, i.e. 60° spacing between each color bin, in order to discriminate between the shades Red, Orange, Yellow, Green, Blue and Indigo. The color binning helps to distinguish between objects that have clearly different textures, e.g. a brown or green texture, even when the underlying mesh is equal. Afterwards we perform an eigenmode decomposition on the mesh. In order to do so, we compute the Laplacian of the mesh as it is an high pass operator that computes second order derivatives. Specifically we use the distance-type Laplacian. Afterwards the first 20 eigenvalues are derived and stored. This way we get a sinusoidal representation of the mesh, that helps characterizing low- and high-frequent properties.

Finally we seek to classify the texture with respect to its topographic representation on the mesh. In other words, we want to keep relations between colors on the mesh. Thus we look at the vertex connections, namely the faces, in color space, instead of the Euclidean Space. Again, we compute Laplacian and eigenmodes. However, we only keep the six first eigenvalues and store them in the descriptor vector.

We create the dissimilarity matrix by comparing each descriptor with each other using the L_2-norm.

Implementation and computation complexity. We implemented the proposed algorithm in Matlab using the *toolbox graph* [Pey]. The discriptors are precomputed, and the norm-based dissimilarity computation is done in $\mathcal{O}(n^2)$, with n the number of input meshes. It can be seen that the eigenmode computation is the main bottleneck of the algorithm, as it has superlinear complexity. There is a tradeoff between the computation time and the accuracy of the spectral description. Usually, mesh consisting of 50000 vertices might require 9000 basis vectors for a full representation of minor details in the mesh. However, as we seek to have the feature descriptor vector to give an overall description of the main appearance of the mesh, we deem the 20 first basis vectors as being sufficient.

3.3. Measuring Distance between 3D Models Based on Geometry and Color Features

The proposed method can be treated as a simplified version of the one in [LZL*12]. We sample the model using points in geometry and color feature space, then these sampling points are optimally clustered. After calculating the feature histogram using these clustered sampling points, we get the shape distribution of the model. Comparing the shape distributions results in the dissimilarity between two models.

Constructing shape distribution. First, we sample the model in the regions of either geometry-high-variation or color-high-variation. Secondly, we cluster the sampling points into several classes by using a modified ISODATA algorithm. Finally, we calculate the feature histogram of each

model using these clustered sampling points, and we can get a shape distribution for each model. See details in [LZL*12].

We extract geometric features based on curvature information since it is invariant in Euclidean and similarity transformation. For color feature extraction, we use the CIE-LAB color space that is designed to approximate human vision and is more perceptually uniform than RGB and HSV color space.

Suppose that in the model there are c clusters and F_i is the set of feature points in the ith cluster with the number of points in F_i being n_i. For each point $f_{ip} \in F_i$, where $i \neq j$, $i, j = 1,2,...,c$, $p = 1,2,...,n_i$, $q = 1,2,...,n_j$, compute the Euclidean distance $d_{i_p j_q} = \| f_{ip} - f_{iq} \|$ and store all the distances in an array D. Then convert the normalized array D into a histogram to get a shape distribution for each model.

Comparing shape distribution. There are some ways mentioned in [OFCD01] to measure the difference between shape distributions, such as (Minkowski) L_N norms, Kolmogorov-Smirnov distance etc. In our experiments, we choose the L_2 norms of the probability density functions to measure the shape distributions, i.e. $dist(f,g) = \sqrt{\int |f-g|^2}$, which obviously satisfies the three conditions in the definition of metric.

Since the probability density function is represented by the shape distribution generated above (combined with another parameter to create different level of approaching, see details in [OFCD01]), the function is piecewise linear, thus the integration is actually applied on a first degree function (or even a constant), which lead to a direct calculation of a second (first) degree polynomial.

Performance. We run the program on a machine with Intel Core2 Duo T8100, 4GB memory and Windows 7 64-bit OS. The sampling number decides accuracy and efficiency.

During our experiment, a sampling number less than about 10.000 can not achieve a good result, while more than 1.000.000 sampling will cost too much time to finish the whole dataset. Considering both sides, we choose 1024×64 sample points to run the program. The program will run about 10s for each pair of the models in the given dataset in average under the condition above.

3.4. Color-weighted Histograms of Area Projection Transform

This method is a simple variation of the Histograms of Area Projection Transform technique proposed in [GL12]. This method is based on computing a spatial map encoding approximated spherical symmetry at different selected scales, called Multiscale Area Projection Transform. This map is obtained by computing for each radius of interest the value

$$\mathrm{APT}(\vec{x}, S, R, \sigma) = Area(T_R^{-1}(k_\sigma(\vec{x}) \subset T_R(S, \vec{n}))) \quad (3)$$

where S is the surface of interest, $T_R(S,\vec{n})$ is the parallel surface if S shifted along the normal vector (in our case only in the inner direction), $k_\sigma(\vec{x})$ is a sphere of radius σ centered in the generic point $\vec{x}$ where the map is computed. Values at different radii are then scaled in order to have a scale-invariant behavior, creating the Multiscale APT map (MAPT):

$$\mathrm{MAPT}(x,y,z,R,S) = \alpha(R)\ \mathrm{APT}(x,y,z,S,R,\sigma(R)) \quad (4)$$

where $\alpha(R) = 1/4\pi R^2$ and $\sigma(R) = c \cdot R \quad (0 < c < 1)$. The transform is easily implemented for surface meshes by sampling points on the model faces, associating a surface value, shifting this point at a distance R along the normal and there adding a normalized surface contribution to the local voxel value. The resulting map is then filtered using the spherical kernel. In [GL12] it is shown that histograms of MAPT computed inside the object can be used with success as global shape descriptors. Map values, ranged in the interval $[0,1]$ are quantized in 12 bins and histograms computed at the different considered scales (radii) are concatenated creating a unique descriptor. Shapes are compared by measuring the histogram distance with the Jeffrey divergence.

Here we propose a new descriptor by concatenating to the previously described histogram, other three similar histograms obtained from "color weighted" APT maps, simply computed multiplying the area contribution of the surface elements by the red, green and blue components respectively (scaled in the interval $[0,1]$). In this way also the texture information should be encoded in the descriptor.

In our test radii and sampling grids are computed as in [GL12]: the isotropic sampling grid is taken proportional to the cubic root of the volume of each model ($s = cbrt(V)/40$), and the sampled radii are integer multiples of s (10 values from $2s$ to $11s$). The radius σ is taken as in the original paper equal to $R/2$ for all the sampled R.

With these choices, the final descriptors of shapes are vectors composed by 480 elements.

3.5. Textured 3D objects retrieval based on 2D multi-view and bag-of-features approach

Three methods are presented for a textured 3D model retrieval, all of them are based on the visual bag-of-features approach. Two steps take a part of the generic framework presented on Figure 3. Offline processing steps correspond to training a visual codebook and indexing. Considering a training 3D model dataset, DAISY [TLF10] and SURF [BETVG08] descriptors are used to extract local features from a range of 2D images rendered from N points of views (in our case $N = 60$) located on vertices of a polyhedron [OOFB08, LGS10]. The visual bag-of-features is clustered to provide a visual dictionary. The latter is used for indexing 3D models of the dataset. For each one of them, local features are extracted and associated to the nearest visual word thanks to the visual dictionary calculated previously.

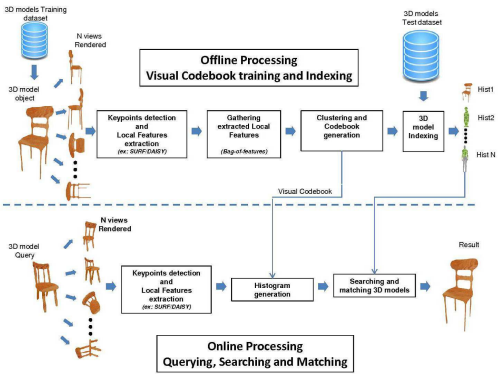

Figure 3: *A sketch of the generic framework described in Section 3.5*

At the end, each 3D model is represented by a histogram of occurrences of visual words. The online processing steps correspond to querying and matching 3D models through the dataset. For a given 3D object to query, local features are extracted and a distribution of visual code word is provided as mentioned previously in the indexing phase. Then, a similarity histogram distance is calculated between the query and the database content. 3D returned models are ranked from the smallest distance to the biggest one.

Pose normalization and Image rendered. Prior to processing cited in the generic framework (Figure 3), a pose normalization is applied to the 3D models contained in the database. Hence, 3D models are centered on their mass center, positioned with respect to there canonical axis and held in unit sphere. Depth and texture representations are used as rendering modes. Ambient light is chosen for textured model representation.

DAISY and SURF descriptors. DAISY descriptor was introduced in [TLF10], finding inspiration from SIFT [Low04] and GLOH [MS05]. It is extracted from local regions. In order to be efficiently computed at every pixel location, the weighted sum of gradients norms used for SIFT and GLOH are replaced by convolutions of the gradients in specific directions with Gaussian filters. This leads anyhow to the same invariance as SIFT and GLOH histogram building.

The SURF [BETVG08] descriptor was designed to be faster than SIFT. It relies on hessian matrix computed on the integral image to detect maxima, keypoints, by the detector and on the sums of 2D Haar wavelet responses calculated on a window that surrounds the detected keypoints.

Methods and Implementations. In the first method, the 3D model is rendered in depth mode and a range of 2D images ($N = 60$) are captured from vertices of a truncated icosahedron. We use the OpenCv bag-of-features workflow [Ope], salient keypoints were provided by SURF detector and described with a DAISY descriptor. Prior to that, a visual codebook (dictionary) was trained offline and contained 2048 bins. Thus, each view of a given 3D model is described by a histogram of occurrence of codeword of the dictionary. Bin to bin histogram summation of the N views provides a single fixed bit length signature for each 3D model of the database. Dissimilarity matrix of distances between models of the database is calculated following the rules announced in the SHREC 2013 contest. The Bhattacharya distance is used to compare two 3D models. In the second and third method, distance calculation between two models is a weighted summation of two distances $d1$ and $d2$. Distance $d1$ is obtained from 3D model textured rendering and distance $d2$ is issued from 3D model depth rendering. SURF descriptor is used in the second method while DAISY descriptor is used in the third one. The same OpenCv workflow is used as explained above.

3.6. 3D Shape + Texture Retrieval

The basic idea of this approach is to compute two features: A shape- and a color-based descriptor. Denote a given colored shape $\mathsf{S} = (\mathcal{S}, \mathbf{S})$, where $\mathcal{S}$ is the mesh information and $\mathbf{S}$ is its color vertices information.

Shape descriptor. 3D shapes are represented by a geodesic distance matrix (GDM). Following [SFH*09] the first 40 eigenvalues of the GDM are used as shape descriptor and then compared using the mean normalized Manhattan distance. We denote this “mesh" distance between two shapes as $dist_{shape}(\mathcal{S}_1, \mathcal{S}_2)$.

Colour-Texture feature. The basic idea is to compute the average Earth mover’s distance between RGB histograms for two given shapes.

Earth mover’s distance (EMD). The earth mover’s distance (also called Wasserstein distance) between two functions $\mathbf{p}$ and $\mathbf{q}$ (where it is assumed that the area under the graph of both functions is the same) is the least work that is required to move the region lying under the graph of $\mathbf{p}$ to that of $\mathbf{q}$ [RTG00]. This can be formalized in the case of two histograms having the same range (m-bins) as a linear programming task, and implemented accordingly. In our experiments, we have used the fast robust implementation given by [PW09]. Thus, for two shapes, we define the associated colour distance as $dist_{RGB}(\mathbf{S}_1, \mathbf{S}_2) = \sum_k \mathrm{EMD}(hist(\mathbf{S}_1^k), hist(\mathbf{S}_2^k))/3$, where $hist$ denote the histogram and the power in $\mathbf{S}^k$ is the colour channel.

Shape and Texture descriptor. Given two shapes $\mathsf{S}_1 = (\mathcal{S}_1, \mathbf{S}_1)$ and $\mathsf{S}_2 = (\mathcal{S}_2, \mathbf{S}_2)$, three measures have been de-

fined:

$$D_1(S_1,S_2) = \frac{dist_{shape}(\mathcal{S}_1,\mathcal{S}_2) + dist_{RGB}(\mathbf{S}_1,\mathbf{S}_2)}{2}$$
$$D_2(S_1,S_2) = \min(dist_{shape}(\mathcal{S}_1,\mathcal{S}_2), dist_{RGB}(\mathbf{S}_1,\mathbf{S}_2))$$
$$D_3(S_1,S_2) = \sqrt{dist_{shape}(\mathcal{S}_1,\mathcal{S}_2) * dist_{RGB}(\mathbf{S}_1,\mathbf{S}_2)}$$

We note that both shape and color distance have been normalized to have a maximum equal to one.

4. Evaluation measures and results

The retrieval performance of each submitted run has been evaluated according to a ternary relevance scale: If a retrieved object shares both shape and texture with the query, then it is highly relevant; if it shares only shape, it is considered marginally relevant; otherwise, it is not relevant. The evaluation process has been thus based on a 2-level ground-truth, by using several evaluation measures: Average precision-recall curves, Nearest Neighbor (NN), First Tier (FT), Second Tier (ST), Normalized Discounted Cumulated Gain (NDCG) and Average Dynamic Recall (ADR).

Note that, because of the multi-level relevance assessment of each query, most of the evaluation measures have been split up as well. "Highly relevant" evaluation measures are based on the highly relevant items only, while "relevant" evaluation measures are based on all the relevant items (highly relevant items + marginally relevant items). We provide in what follows a brief explanation of each evaluation measure, together with the associated evaluation results. The runs of all the track's participants are labeled as specified in Section 2.

Average precision-recall curves. Precision and recall are common measures to evaluate information retrieval systems. Precision is the fraction of retrieved items that are relevant to the query. Recall is the fraction of the items relevant to the query that are successfully retrieved. Being A the set of all the relevant objects and B the set of all the retrieved object,

$$\text{Precision} = \frac{|A \cap B|}{|B|}, \quad \text{Recall} = \frac{|A \cap B|}{|A|}. \tag{5}$$

Note that the two values always range from 0 to 1. For a visual interpretation of these quantities it is useful to plot a curve in the reference frame recall vs. precision. We can interpret the resulting curve as follows: The larger the area below such a curve, the better the performance under examination. In particular, the precision-recall curve of an ideal retrieval system would result in a constant curve equal to 1.

For each query, we thus have a precision-recall curve. By taking the average on all the queries, we get the average precision-recall curve. Figure 4 shows the performances of all the runs with respect to the average precision-recall curve, both "relevant" and "highly relevant".

	Relevant			Highly Relevant			
Run	**NN**	**FT**	**ST**	**NN**	**FT**	**ST**	**ADR**
A1	0.963	0.588	0.681	0.515	0.553	0.710	0.374
A2	0.958	0.603	0.720	0.508	0.561	0.730	0.380
Be	0.083	0.135	0.229	0.019	0.175	0.209	0.173
Zh	0.342	0.238	0.353	0.174	0.135	0.214	0.104
Gi	0.971	0.574	0.715	0.788	0.658	0.748	0.470
G1	**1.00**	**0.708**	**0.873**	0.417	0.526	0.799	0.379
G2	0.992	0.575	0.708	0.898	0.733	0.893	0.508
G3	0.983	0.632	0.801	0.519	0.579	0.772	0.415
V1	0.871	0.422	0.583	0.807	0.511	0.633	0.413
V2	0.996	0.480	0.606	0.879	**0.764**	**0.904**	**0.520**
V3	0.971	0.476	0.634	**0.909**	0.733	0.863	0.511

Table 1: *Retrieval performances on the whole dataset. For each evaluation measure, best results are in bold text.*

Nearest Neighbor, First tier and Second tier. These evaluation measures aim at checking the fraction of models in the query's class also appearing within the top k retrievals. Here, k can be 1, the size of the query's class, or the double size of the query's class. Specifically, for a class with $|C|$ members, $k = 1$ for the nearest neighbor (NN), $k = |C| - 1$ for the first tier (FT), and $k = 2(|C| - 1)$ for the second tier (ST). The final score is an average over all the models in the database. Not that all these values necessarily range from 0 to 1. Table 1 reports the performances for all the runs according to these measures, with respect to the "relevant" and "highly relevant" classifications.

Average dynamic recall. The idea is to measure how many of the items that should have appeared before or at a given position in the result list actually have appeared. The average dynamic recall (ADR) at a given position averages this measure up to that position. Precisely, for a given query let A be the set of highly relevant classified items, and let B be the set of the relevant items. Obviously $A \subseteq B$. Also, for our dataset $|B|$ is always equal to 24. The ADR is computed as:

$$\text{ADR} = \frac{1}{|B|} \sum_{i=1}^{|B|} \frac{r_i}{i},$$

where r_i is defined as

$$r_i = \begin{cases} \frac{|\{\text{highly relevant items in the first } i \text{ retrieved items}\}|}{i}, & \text{if } i \leq |A|; \\ \frac{|\{\text{relevant items in the first } i \text{ retrieved items}\}|}{i}, & \text{if } i > |A|. \end{cases}$$

For all participants, the last column of Table 1 reports the ADR measure averaged on all queries.

Normalized discounted cumulated gain. It is first convenient to introduce the *discounted cumulated gain (DCG)*. Its definition is based on two assumptions. First, highly relevant items are more useful if appearing earlier in a search engine result list (have higher ranks); Second, highly relevant items are more useful than marginally relevant items, which are in turn more useful than irrelevant items.

DCG originates from a more primitive measure called *cu-*

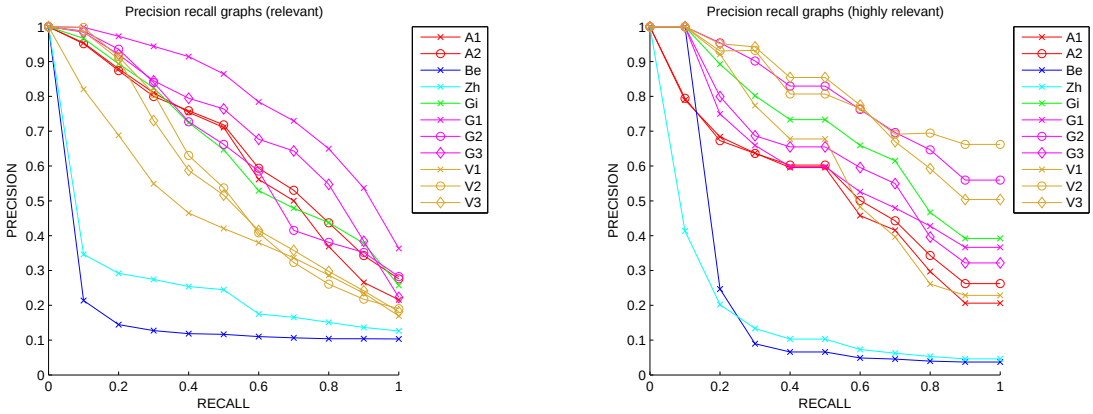

Figure 4: *Performances of all the runs with respect to the average precision-recall curve, both relevant and highly relevant.*

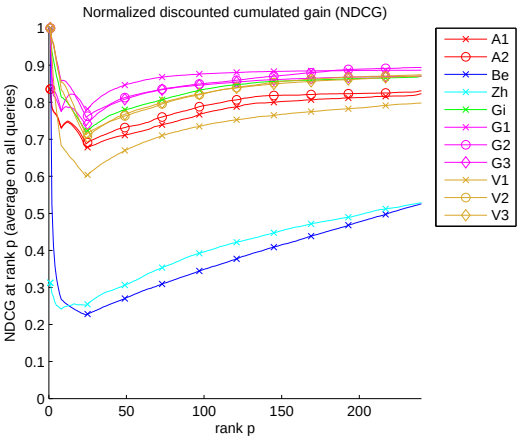

Figure 5: *Performances of all the runs with respect to the NDCG measure.*

mulated gain (CG). The CG at a position p is defined as

$$\mathrm{CG}_p = \sum_{i=1}^{p} \mathrm{rel}_i,$$

with rel_i the graded relevance of the result at position i. The value computed with the CG function is unaffected by changes in the ordering of search results. Based on the two assumptions made above about the usefulness of search results, DCG is used in place of CG for a more accurate measure. Precisely, the DCG at a position p is defined as:

$$\mathrm{DCG}_p = \mathrm{rel}_1 + \sum_{i=2}^{p} \frac{\mathrm{rel}_i}{\log_2(i)}.$$

Obviously, the DCG is query-dependent. To overcome this problem, we normalize the DCG to get the *normalized discounted cumulated gain (NDCG)*. This is done by sorting elements of a retrieval list by relevance, producing the maximum possible DCG till position p, also called *ideal DCG (IDCG)* till that position. For a query, the NDCG is computed as

$$\mathrm{NDCG}_p = \frac{\mathrm{DCG}_p}{\mathrm{IDCG}_p}.$$

In the present evaluation, the NDCG values for all queries are then averaged to obtain a measure of the average performance for each submitted run. Note that for an ideal run, the DCG_p will be the same as the IDCG_p producing an NDCG of 1. All NDCG calculations are then relative values on the interval 0 to 1 and so are cross-query comparable. Figure 5 shows the performance evaluation for all runs according to the NDCG measure as a function of the rank p.

5. Discussion

The experimental results offer several hints for discussion.

NDCG and ADR provide an *overall* evaluation of the capability owned by the proposed methods in interpreting the 2-level classification of the dataset. On the one hand, the NDCG results in Figure 5 show encouraging results from almost all the runs submitted to the track, thus revealing that the current scenario about retrieval methods for textured 3D models is very lively and promising. On the other hand, the ADR results in Table 1 emphasize that the dataset was challenging and supplies for performance improvements. Indeed, the best ADR scores fluctuate around 0.5, the best possible ADR value being 1.

Results in Table 1 about the nearest neighbor measure (NN) interestingly reveal that all runs degrade passing from the “relevant” to the “highly relevant” evaluation. This means that, as for the first retrieved item, the proposed methods foster in general the texture-based retrieval instead of the geometry-based one, which is actually in contrast with the

relevance scale underlying the dataset classification. Nevertheless, such a behavior is progressively redressed when considering a larger number of retrieved models, that is, passing from the NN to the second tier (ST) measure through the first tier (FT) one, see once more Table 1. Indeed, along this path we can observe that, as for the "highly relevant" evaluation, more and more runs outperform their performances in the "relevant" situation.

Finally, looking at the precision-recall curves in Figure 4, we can deduce that:

- The strategy used for runs **V1**, **V2** and **V3** to deal with color information (see Section 3.6) appears to be promising to improve the retrieval performance by taking models' texture into account;
- The bag-of-feature approach proposed with runs **G1**, **G2** and **G3** (see Section 3.5) seems to be flexible and modular enough to accomplish good performances both in generic and textured 3D models retrieval;
- The HKS-based strategy exploited by runs **A1** and **A2** (see Section 3.1) obtained noticeable results even in the presence of non-isometric model deformations, and appears to be scalable with small efforts to the case of textured 3D models retrieval;
- The color weighted MAPT (run **Gi**, see Section 3.4) emerges as a valuable technique to get shape descriptors which incorporate *a priori* the texture information. As mentioned by the author, performances could be improved by optimizing scale selection, binning or histogram comparison methods according to the task at hand.

6. Conclusions

In this paper, the new track of SHREC'13 on *Retrieval on Textured 3D Models* is introduced, describing how the dataset was built and the kind of deformations made on a set of textured models. This is the first time that a track of SHREC specifically focuses on the performance evaluation of retrieval algorithms when models vary either by geometric shape or texture, or both. The experimental results show that the current scenario about retrieval methods for textured 3D models is promising. Indeed, the submitted runs obtained in general encouraging results. Finally, it is to be hoped that this new benchmark will promote further investigation on the comparison and retrieval of textured models.

Acknowledgments. This work has been developed in the CNR research activity ICT.P10.009, and partially supported by VISIONAIR, European project "FP7 INFRASTRUCTURES" (2011-2015).

References

[AF06] ATTENE M., FALCIDIENO B.: Remesh: An interactive environment to edit and repair triangle meshes. In *Proc. SMI'06* (2006), IEEE Computer Society, p. 41. 2

[BBGO11] BRONSTEIN A. M., BRONSTEIN M. M., GUIBAS L. J., OVSJANIKOV M.: Shape google: Geometric words and expressions for invariant shape retrieval. *ACM Trans. Graph. 30*, 1 (2011), 1:1–1:20. 2, 3

[BETVG08] BAY H., ESS A., TUYTELAARS T., VAN GOOL L.: Speeded-up robust features (surf). *Comput. Vis. Image Underst. 110*, 3 (June 2008), 346–359. 4, 5

[BK10] BRONSTEIN M. M., KOKKINOS I.: Scale-invariant heat kernel signatures for non-rigid shape recognition. In *Proc. CVPR* (2010), pp. 1704–1711. 2

[GL12] GIACHETTI A., LOVATO C.: Radial symmetry detection and shape characterization with the multiscale area projection transform. *Comp.Graph.Forum 31*, 5 (2012), 1669–1678. 4

[LGS10] LIAN Z., GODIL A., SUN X.: Visual similarity based 3d shape retrieval using bag-of-features. In *Proc. SMI '10* (2010), IEEE Computer Society, pp. 25–36. 4

[Low04] LOWE D. G.: Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision 60*, 2 (2004), 91–110. 5

[LZL*12] LIU Y.-J., ZHENG Y.-F., LV L., XUAN Y.-M., FU X.-L.: 3d model retrieval based on color + geometry signatures. *Vis. Comput. 28*, 1 (2012), 75–86. 3, 4

[MS05] MIKOLAJCZYK K., SCHMID C.: A performance evaluation of local descriptors. *IEEE Trans. PAMI 27*, 10 (2005), 1615–1630. 5

[OFCD01] OSADA R., FUNKHOUSER T., CHAZELLE B., DOBKIN D.: Matching 3d models with shape distributions. In *Proc. SMI'01* (2001), IEEE Computer Society, pp. 154–166. 4

[OOFB08] OHBUCHI R., OSADA K., FURUYA T., BANNO T.: Salient local visual features for shape-based 3d model retrieval. In *In SMI* (2008). 4

[Ope] http://opencv.willowgarage.com/documentation/cpp/object_recognition.html. 5

[Pey] PEYRE G.: Toolbox graph - a toolbox to process graph and triangulated meshes. http://www.ceremade.dauphine.fr/~peyre/matlab/graph/content.html. 3

[PW09] PELE O., WERMAN M.: Fast and robust earth mover's distances. In *Computer Vision, 2009 IEEE 12th International Conference on* (2009), pp. 460 –467. 5

[RTG00] RUBNER Y., TOMASI C., GUIBAS L. J.: The earth mover's distance as a metric for image retrieval. *Int. J. Comput. Vision 40*, 2 (2000), 99–121. 5

[RWSN09] REUTER M., WOLTER F.-E., SHENTON M., NIETHAMMER M.: Laplace-beltrami eigenvalues and topological features of eigenfunctions for statistical shape analysis. *Comput. Aided Des. 41*, 10 (2009), 739–755. 2

[SFH*09] SMEETS D., FABRY T., HERMANS J., VANDERMEULEN D., SUETENS P.: Isometric deformation modelling for object recognition. In *Proc. CAIP '09* (Berlin, Heidelberg, 2009), Springer-Verlag, pp. 757–765. 5

[SOG09] SUN J., OVSJANIKOV M., GUIBAS L.: A concise and provably informative multi-scale signature based on heat diffusion. In *Proc. SGP '09* (2009), Eurographics Association, pp. 1383–1392. 2

[TLF10] TOLA E., LEPETIT V., FUA P.: Daisy: An efficient dense descriptor applied to wide baseline stereo. *IEEE TRANS. PAMI 32*, 5 (2010). 4, 5

[Vis] VISUAL COMPUTING LAB – ISTI – CNR: Meshlab. http://meshlab.sourceforge.net/. 2