Vis Comput (2017) 33:1571–1585
DOI 10.1007/s00371-016-1301-5

ORIGINAL ARTICLE

Scalable 3D shape retrieval using local features and the signature quadratic form distance

Ivan Sipiran[1] · Jakub Lokoč[2] · Benjamin Bustos[3] · Tomáš Skopal[2]

Published online: 9 August 2016

Abstract We present a scalable and unsupervised approach for content-based retrieval on 3D model collections. Our goal is to represent a 3D shape as a set of discriminative local features, which is important to maintain robustness against deformations such as non-rigid transformations and partial data. However, this representation brings up the problem on how to compare two 3D models represented by feature sets. For solving this problem, we apply the signature quadratic form distance (SQFD), which is suitable for comparing feature sets. Using SQFD, the matching between two 3D objects involves only their representations, so it is easy to add new models to the collection. A key characteristic of the feature signatures, required by the SQFD, is that the final object representation can be easily obtained in a unsupervised manner. Additionally, as the SQFD is an expensive distance function, to make the system scalable we present a novel technique to reduce the amount of features by detecting clusters of key points on a 3D model. Thus, with smaller feature sets, the distance calculation is more efficient. Our experiments on a large-scale dataset show that our proposed matching algorithm not only performs efficiently, but also its effectiveness is better than state-of-the-art matching algorithms for 3D models.

Keywords 3D shape retrieval · Local features · Signature quadratic form distance

1 Introduction

Three-dimensional data have become an important resource for many applications in medicine, engineering, entertainment, archeology, and so on. This is because this kind of information provides interesting characteristics which allow us to represent real objects in a more precise manner than other media. Similarly, the emergence of large and publicly available 3D collections such as Google 3D Warehouse brings up the need of storing and manipulating this information in an effective way.

A key aspect in multimedia collections is the content-based search. It aims at using the multimedia information itself to determine the similarity between two objects. Since associated meta-data is not always available, this searching approach is suitable. Moreover, some effective methods have been presented in the context of image searching, for instance. However, these methods are not directly adaptable to other media such as 3D data because it has its own characteristics and analysis tools.

There are several important issues involving content-based search. First, the goal is to provide good effectiveness when a query is given. That is, one wants to retrieve a ranked list of objects from the collection and expects to have similar objects on the first rankings. Second, the search time should be small enough to ensure a practical use of the collection. Finally, the evolution of the retrieval system in time is also important. A retrieval system should be scalable and

✉ Ivan Sipiran
ivan.sipiran@gmail.com; isipiran@dcc.uchile.cl

Jakub Lokoč
lokoc@ksi.ms.mff.cuni.cz

Benjamin Bustos
bebustos@dcc.uchile.cl

[1] Sección Ingeniería Informática, Pontificia Universidad Católica del Perú PUCP, Lima, Peru

[2] SIRET Research Group, Faculty of Mathematics and Physics, Charles University in Prague, Prague, Czech Republic

[3] Department of Computer Science, University of Chile, Santiago, Chile

dynamic. These two aspects imply that the system should support a constant growth. This sounds logical, since the amount of multimedia information is growing rapidly.

We are interested in content-based retrieval of 3D shape collections. Our work consists in using local information from shapes to describe them. Thus, unlike a global approach where a shape corresponds to a single feature descriptor, a 3D shape is represented by a set of local features. Local features extracted from shapes allow us to deal with several problems such as deformable transformations, for instance. However, it also brings up the problem on how to effectively compare two 3D models represented as a set of local features. So far, many methods have been proposed to address the matching of 3D shapes using local information. Most of them have been proposed taking into account only the effectiveness, leaving out other key aspects such as efficiency and scalability.

In this paper, we present a novel method to represent a 3D shape based on local features. Our method finds clusters of key points which correspond to surface regions with an outstanding local structure. Thus, it is possible to reduce the number of features to represent a shape and, therefore, the subsequent process can be efficient. In addition, we apply the signature quadratic form distance (SQFD) [1] to compare two sets of features. This distance has proven effective in the image search domain [2], and we will show that it can enhance 3D model retrieval.

In our opinion, there are several reasons why the SQFD is suitable for our purposes. First, it is a flexible way to compare two multimedia objects represented by feature sets. Coupled with it, if we are able to reduce the size of the representations, we can also get efficiency. Second, the distance is context free as it involves just the feature signatures of the models being compared, not the rest of the database. So, it is easy to add new models into the collection because we just need to extract their representations and store them. This behavior would allow us to build a scalable system for dynamic collections. Third, as SQFD treats the object representations as a black-box feature sets, it is easy to redesign or adjust the entire model without the need of reinventing the mechanism of assessing similarity. Hence, SQFD is a universal distance for comparing object representations based on local features.

The contribution of this paper is threefold. First, we propose the use of key points on meshes to make the local features more discriminative. This step enables the reduction of the amount of information used in the matching. In addition, we present a novel method to detect clusters of key points on 3D meshes. Thus, each object will be represented by a set of features extracted from regions. Second, we apply the signature quadratic form distance to compare two models represented by a set of features. Finally, we evaluate our approach and compare it with techniques from the state of the art. It is worth mentioning that we do not include learning-based approaches in our comparison because we focus our attention in unsupervised retrieval techniques.

2 Related work

Three-dimensional shape retrieval has been an active research area in recent times. As a result, many approaches have been proposed from several perspectives. In general, techniques are divided into global and local feature methods. On the one hand, global methods rely on extracting information regarding the whole shape such as depth views, shape histograms, and so on. Therefore, these representations (possibly after some numerical transformation) can be compared using a distance function. On the other hand, local feature methods consist of describing local portions of a 3D mesh, so the comparison is done between feature sets. In this section, we present a brief review about shape retrieval, with emphasis on methods that use local features.

Initially, the proposals were intended to define a similarity model based on the entire shape. Bustos et al. [13] defined a similarity model where 3D objects were represented as descriptors in high-dimensional spaces. Thus, the storage and manipulation of these descriptors could be efficiently handled with indexing structures. In addition, a comprehensive comparison of several techniques resulted in that techniques based on depth views were the most effective. A weak aspect of the global methods is that they do not support non-rigid transformations and partial retrieval. In the rest of this section, we focus on techniques which use local features for shape retrieval.

When dealing with local features, the problem is to find a compact representation for the object using a set of descriptors. Therefore, the efforts have been concentrated in proposing models for aggregation of local descriptors. Table 1 presents a list of state-of-the-art techniques that used local descriptors and the aggregation approach.

The bag-of-feature approach (BoF) is one of the most known methods in multimedia retrieval. The method can be divided into two steps: computation of visual words and quantization. The first step computes a set of representative descriptors through clustering (commonly k-means). Subsequently, the quantization consists of assigning the local descriptors of a shape to their most similar visual words. The output is a histogram that represents a distribution of the local features with respect to the visual words. Bronstein et al. [3] proposed the Shape Google method using the idea of BoF. The first step is the computation of heat kernel signatures or scale-invariant heat kernel signatures for each vertex in a 3D shape. Then, the authors proposed to applying a soft-quantization to get the shape descriptors. In addition, they proposed a spatially sensitive bag of features to take the spatial relations between descriptors into account in the final

Table 1 Techniques that use aggregation of local descriptors for 3D shape retrieval

Method	Feature	Aggregation
Shape Google [3]	HKS based	BoF
Weighted Heat diffusion [4]	HKS based	BoF
Covariance-based retrieval [5]	Covariance mat.	BoF
Two layer coding [6]	View based	VLAD
Fisher encoding [7]	dFPFH	Fisher encoding
Supervised sparse coding [8]	HKS based	Sparse coding
Locally constrained sparse coding [9]	HKS based	Sparse coding
Deep Shape Descriptor [10]	HKS based	Deep learning
Multi-modal feature fusion[11]	HKS based	Deep learning
Deep Shape [12]	HKS based	Deep learning

representation. Similarly, Abdelrahman et al. [4] proposed a BoF approach to perform the retrieval of 3D shapes considering not only the geometric information but also the textures. The method focuses on a new formulation of the Heat Kernel Signatures to include the color as a weight in edges and points on the surface. Likewise, Tabia et al. [5] proposed a BoF model to work on covariance matrices. The idea behind this technique is the use of simple local features which can be aggregated in covariance matrices in a first level. Then, a set of covariance matrices is considered as local descriptions and a subsequent BoF can, therefore, be applied. In addition to regular BoF, new variations such as VLAD [6] or Fisher encoding [7] have been applied for 3D objects retrieval.

The sparse coding is also a method to aggregate local descriptors. In this case, the approach is able to work in a supervised way for finding a good set of visual words. The sparse coding aims at finding a set of bases and linear coefficients that represents the input data. The term sparse is due to requirement of finding a few coefficients to represent a given descriptor. Litman et al. [8] presented a supervised sparse coding method to learn the shape dictionary based on Heat Kernel Signatures. Once the dictionary was computed, the sparse codes for every local descriptor were calculated using a modified version of the classical regularized pursuit problem. The final quantization was performed by multiplying the matrix of sparse codes with a vector of area-based weights per vertex. Similarly, Liu et al. [9] suggested to apply sparse coding in a patch-level instead of a vertex level. The first step is to segment the 3D shape and describe each segment with the conformal geometry signature, shape diameter function, average geodesic distance and the scale-invariant Heat Kernel Signature. The authors adopted a locally constrained approach to sparse coding in which the similarity between the descriptor and the dictionary atom is involved. This approach enforces the smoothness of the sparse codes in the feature space.

Very recently, deep learning has been used for retrieval as well. The idea behind deep learning is the definition of structured learning models that are able to abstract the low-level input data. In this way, one can find high-level representations for the low-level data. It can be somehow considered as an aggregation function which can be applied in local features to make the representation more compact. Bu et al. [11] developed an algorithm that extracts several features such as scale-invariant Heat Kernel Signatures, average geodesic distance, and view-based SIFT descriptors obtained from depth images. Subsequently, a deep neural network is trained for every type of feature. The last layer's output is considered as a feature. The final representation is the concatenated output for each deep neural network. Similarly, Xie et al. [12] proposed an auto-encoder neural network to learn a high-level representation of multi-scale Heat Kernel Signatures. The authors suggested creating a neural network for each scale and the final descriptor is the concatenation of the activation values of the hidden layers. Likewise, Fang et al. [10] trained a many-to-one encoder neural network which forces input from the same class to produce a unique target value. The input for the neural network are the Eigen-shape descriptor and Fisher-shape descriptor, obtained by applying the PCA and Fisher analysis over the distribution of heat kernel signatures in several scales.

The main disadvantage of sparse coding and deep learning techniques is the requirement of having labeled data for the training phase. Although there are some efforts for building labeled large-scale 3D datasets (such as ShapeNet [14] for example), there are still problems that require attention such as unbalanced classes, intra-class variability and so on. These problems make difficult the definition of training and test datasets and, therefore, it prevents the effective use of learning techniques for 3D retrieval in the general case. In this paper, we do not compare our technique against learning-based approaches because we want to provide a fair comparison between unsupervised methods.

From the local features perspective, it is clear that heat kernel signatures and variations are the common choice for current retrieval approaches. This descriptor has desirable properties, in particular for the problem of non-rigid shape retrieval. In this paper, we show that the SQFD distance is an effective complement for the aggregation of HKS descriptors.

3 Background

In this section, we describe the necessary background for the rest of the paper. Briefly, Sect. 3.1 describes the heat kernel signature (HKS) [15], which is a useful analysis and description tool for 3D meshes. In addition, Sect. 3.2 presents the signature quadratic form distance (SQFD).

In addition, at this point it is necessary to clarify some terms. Both methods (HKS and SQFD) make use of the term signature in different meanings. In the HKS, signature is a descriptor for a vertex on the mesh. Differently, in the SQFD, signature is a set of features belonging to an object. To disambiguate them, we will refer explicitly as heat kernel signatures and feature signatures, respectively.

3.1 Heat kernel signatures

The heat diffusion process over a compact manifold S, possibly with boundary, is governed by the heat equation:

$$\Delta_S u(x,t) = -\frac{\partial u(x,t)}{\partial t} \tag{1}$$

where Δ_S is the Laplace–Beltrami operator of S and $u(.,t)$ is the heat distribution over S in time t.

The fundamental solution of Eq. 1 is $K_t(x,y)$ called the heat kernel. This is a solution with a point heat source in x and can be considered as the amount of heat transferred from x to y at time t with a heat source at x. For compact manifolds, the heat kernel can be expressed using the eigenvalues and eigenvectors of the Laplace–Beltrami operator as follows:

$$K_t(x,y) = \sum_{i=0}^{\infty} \exp(-\lambda_i t)\mathbf{v_i}(x)\mathbf{v_i}(y) \tag{2}$$

where λ_i is the ith eigenvalue and $\mathbf{v_i}(\cdot)$ is the ith eigenvector's entry corresponding to a given point.

Sun et al. [15] formally proved that the heat kernel is an isometric invariant, informative, multi-scale, and stable against perturbations on the surface. In addition, by restricting the heat kernel to the temporal domain and fixing the spatial variables, we can obtain a representation for each point on the manifold by computing $K_t(x,x)$. This expression computes the heat distribution in time t for each point in the surface. Figure 1 depicts an example on how the heat distribution changes on every point over time. In practice, the heat kernel signature of a point $x \in S$ is an n-dimensional descriptor vector with each bin corresponding to some value of t: $hks(x) = (hks_1(x), \ldots, hks_n(x))$ where $hks_i(x) = K_{t_i}(x,x)$ (see Fig. 2).

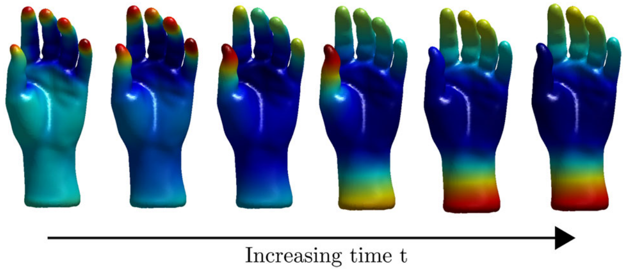

Fig. 1 Heat distribution over time. The *colors* go from *blue* (low values) to *red* (high values). The heat values on every point change according to the time parameter in $K_t(x,x)$

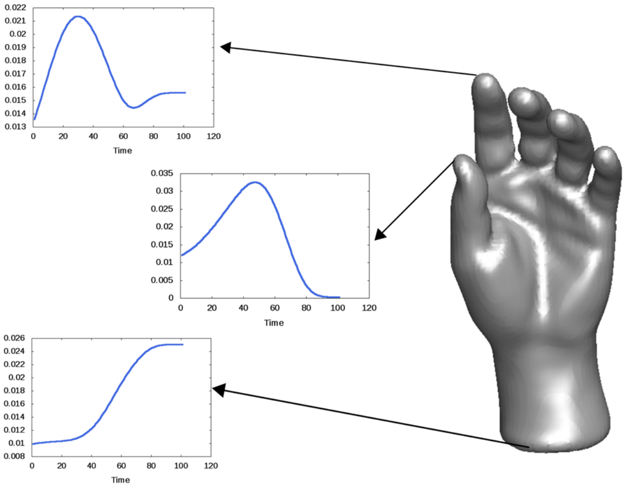

Fig. 2 Heat Kernel Signatures in three different points. Note that the heat distribution over time corresponds to the behavior shown in Fig. 1

To numerically compute the heat kernel signatures over a mesh, we need to calculate the discrete Laplace–Beltrami operator. In this work, we used the discretization proposed by Belkin et al. [16]. This method computes the operator and delivers a symmetric sparse matrix M and a diagonal matrix A with area values associated with each vertex. Both matrices have $N \times N$ elements, where N is the number of vertices on the mesh. Subsequently, we need to solve the generalized eigenvalue problem: $M\mathbf{v} = \lambda A\mathbf{v}$. Furthermore, to compute the heat kernel signatures, we only need to approximate them with a few eigenvalues and eigenvectors. So we can solve the eigenproblem efficiently as the problem is sparse.

A scale-invariant variation of HKS was proposed by Bronstein and Kokkinos in [17]. This variation, hereafter SI-HKS, considers a series of transformations of the HKS function to remove the dependency to scale. The coefficients of low frequency of the final Fourier transformation can be considered as the new descriptor. In all our experiments, we only consider six coefficients.

3.2 Feature signatures and signature quadratic form distance

Generally, histogram-based descriptors aggregating local features to vectors $h_X, h_Y \in R^n$ and the Euclidean distance are frequently used to model a similarity between two objects X, Y. To model also correlations between histogram bins on homogeneous domains, the Euclidean distance can be extended to the quadratic form distance

$$\mathrm{QFD}_A(h_X, h_Y) = \sqrt{(h_X - h_Y)A(h_X - h_Y)^T}, \tag{3}$$

where A is an $n \times n$ positive-definite correlation matrix [18]. Since feature histograms are limited with a fixed number of dimensions for all modeled objects, more flexible representations can be considered to adaptively fit to the complexity of each object. However, the adaptability comes at the cost of mostly incompatible representations for above-mentioned distances. As pointed out by Beecks et al. [19], the incompatibility can be overcome by the signature quadratic form distance adopting the correlation concept from the quadratic form distance.

In our work, feature signatures are used to flexibly aggregate heat kernel-based local features detected on a 3D shape. Let object P be represented by a feature set $F = \{f_i\}$, where $f_i \in$ FS. FS is a feature space of an arbitrary dimension (in our work we use R^n), and let us suppose that $|F| = K$. Furthermore, we need to suppose the existence of a local clustering of $F : C_1, \ldots, C_n$. The feature signature S^P is defined as a set of tuples FS $\times R^+$ as follows:

$$S^P = \left\{(c_i^P, w_i^P), i = 1, \ldots, n\right\} \tag{4}$$

where $c_i^P = \frac{\sum_{f \in C_i} f}{|C_i|}$ and $w_i^P = \frac{|C_i|}{K}$ represent the centroid of ith cluster and a weight, respectively. Note that the size of S^P depends on the local partitioning and it is variable from object to object. Thus, each feature signature may have a different number of tuples according to the clustering process.

Given two feature signatures $S^P = \{(c_i^P, w_i^P), i = 1, \ldots, n\}$ and $S^Q = \{(c_j^Q, w_j^Q), j = 1, \ldots, m\}$ and a positive-semidefinite similarity function sim : FS × FS → R, the signature quadratic form distance between S^P and S^Q is defined as

$$\mathrm{SQFD}(S^P, S^Q) = \sqrt{(w^P | - w^Q) \cdot A_{sim} \cdot (w^P | - w^Q)^T} \tag{5}$$

$$A_{sim} = \begin{array}{c|c:c|} & S^P & S^Q \\ \hline S^P & sim(c_i^P, c_j^P) & sim(c_i^P, c_{j-n}^Q) \\ \hdashline S^Q & sim(c_{i-n}^Q, c_j^P) & sim(c_{i-n}^Q, c_{j-n}^Q) \\ \hline \end{array}$$

Fig. 3 Similarity matrix A is composed of four blocks. The intra-dependence blocks (*top left* and *right bottom*) contain the similarity between centroids within the signatures S^P and S^Q, respectively. The inter-dependence blocks (*top right* and *left bottom*) contain the similarity between centroids in S^P and S^Q

where the notation $(w^A | w^B)$ denotes the concatenation of weight vectors. In addition, $A_{\mathrm{sim}} \in R^{(n+m)\times(n+m)}$ is the similarity matrix between centroids of S^P and S^Q, defined as

$$a_{i,j} = \begin{cases} \mathrm{sim}(c_i^P, c_j^P) & \text{if } i \le n \text{ and } j \le n \\ \mathrm{sim}(c_{i-n}^Q, c_j^P) & \text{if } i > n \text{ and } j \le n \\ \mathrm{sim}(c_i^P, c_{j-n}^Q) & \text{if } i \le n \text{ and } j > n \\ \mathrm{sim}(c_{i-n}^Q, c_{j-n}^Q) & \text{if } i > n \text{ and } j > n \end{cases} \tag{6}$$

Originally, Beecks et al. proposed also a coincidence and a quadratic form model for the signature quadratic form distance that are equivalent to the concatenation model. In the concatenation model, vector $(w^P | - w^Q)$ corresponds to the difference of vectors $w'^P, w'^Q \in R^{(n+m)}$, where $w'^P = (w_1^P, \ldots, w_n^P, 0_{n+1}, \ldots, 0_{n+m})$ and $w'^Q = (0_1, \ldots, 0_n, w_{n+1}^Q, \ldots, w_{n+m}^Q)$. The similarity matrix A_{sim} (see Fig. 3) models the correlation between particular dimensions of vectors w'^P and w'^Q. Note that the correlation coefficients $a_{i,j}$ help to match perceptually similar dimensions i, j on homogeneous domains. The heat kernel signatures are descriptions of the heat distribution on the surface of 3D objects, where two similar objects have similar heat distributions. Therefore, it is meaningful to model the correlation between dimensions as a similarity of two centroids corresponding to heat kernel signatures. In our work, we use the Gaussian similarity function

$$\mathrm{sim}_g(c_i, c_j) = \exp(-\alpha d^2(c_i, c_j)), \tag{7}$$

where d : FS × FS → R_0^+ is a metric ground distance function (e.g., the Euclidean distance).

4 3D shape retrieval using local features

We propose three different approaches based on local features and the signature quadratic form distance for 3D shape retrieval. The three approaches heavily rely on a clustering process to determine the feature signatures. We use the clustering algorithm proposed by Leow and Li [20]. This algorithm uses a intra-cluster (β) and inter-cluster (δ) threshold as well as the minimum number of elements per cluster (Nm).

Briefly, the clustering method performs several iterations on the input points. In each iteration, the algorithm looks for groups of points whose distances are below the intra-cluster threshold β. If a point is beyond the inter-cluster threshold δ from any of the current clusters, the algorithm creates a new cluster and starts a new iteration. The algorithm converges after a number of iterations, when the clusters do not change anymore. Note that the clustering algorithm is adaptive because the final number of clusters depends only on the distribution of the points in the space. This flexibility is a key ingredient for the representation power of our method, and it is the main reason of choosing the SQFD distance to assess the similarity.

4.1 First approach: feature signatures on all vertices

In this approach, we aim at partitioning the whole feature space of a 3D object to define the feature signatures. Let S be a 3D model with n vertices. We represent S as a set of heat kernel signatures computed for each vertex on the mesh. We call $\mathrm{FS}(S)$ to this representation and it is formally defined as:

$$\mathrm{FS}(S) = \left\{ \frac{hks(v_i)}{\|hks(v_i)\|} | v_i \in S, i = 1, \ldots, n \right\} \quad (8)$$

The normalization of the heat kernel signatures help us to properly define the intra-cluster and inter-cluster thresholds for the clustering.

Now, we can apply the clustering algorithm over $FS(S)$. The clustering computes a partitioning not necessarily complete, as there are points which do not belong to any cluster. In this respect, each remaining heat kernel signature is assigned to its nearest cluster. Thus, now we have a complete and disjoint partitioning of the feature space. Finally, we represent S as suggested in Eq. 4.

The heat kernel signatures are multi-scale because depending of the t values, we can analyze a local or global behavior in each vertex. We decided to use both behaviors, as they gave good results for clustering and made our descriptors stable to perturbations. Figure 4 shows examples of partitions under non-rigid transformations. It follows that, using the complete signature, the shape is better clustered and the vertices in the same cluster share a spatial region. In Sect. 5, we will detail the configuration for computing the heat kernel signatures.

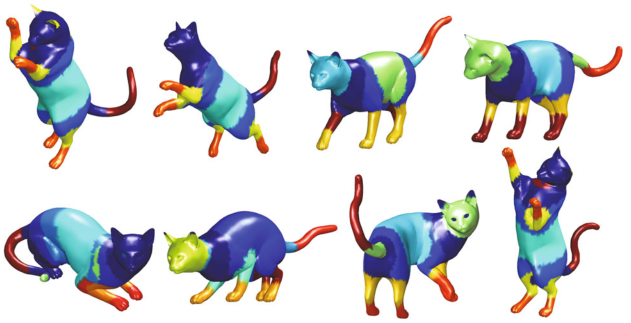

Fig. 4 *Colors* represent the clusters using the HKS descriptors of the entire shape. The clusters are consistent despite the non-rigid transformations. Note: the *colors* are arbitrary and they are only used to show the resulting clusters in a shape

4.2 Second approach: feature signatures on key points

In this approach, we aim at partitioning a sub-set of the feature space of a 3D object to define the feature signatures. The idea behind the use of a sub-set is that the whole feature space is not necessarily discriminative. Our hypothesis is that there is a sub-set which is representative for each model. Therefore, our goal is to find that sub-set and partition it, so the feature signatures represents a partitioning of the most discriminative features.

The task of finding such a sub-set directly from the feature space is difficult. Therefore, at this stage, we apply a method for interest point detection on meshes. By detecting key points on meshes, we aim to select the most relevant vertices on the mesh. Next, we can use the heat kernel signatures on the key points for describing the object, thus obtaining the required sub-set of the feature space.

Given a 3D mesh, we detect key points using the Harris 3D method [21]. Briefly, this method extends the well-known Harris operator to 3D meshes. The idea is to represent a local neighborhood around a vertex as an image. First, the algorithm finds a local patch using an adaptive method making it robust to geometry changes. Second, the local patch is converted into a canonical local system using PCA to fit a plane. Third, a quadratic surface is fitted to the local patch. Fourth, the method takes the derivatives of the fitted surface and it computes smoothed versions of these derivatives by convolving gaussians with them. Finally, the Harris response is calculated using the derivatives which are arranged in a 2 × 2 matrix as usual.

After computing the Harris responses for each vertex on the mesh, we select the 1 % of the total number of vertices with the highest Harris response as key points. Figure 5a shows a 3D model with its key points.

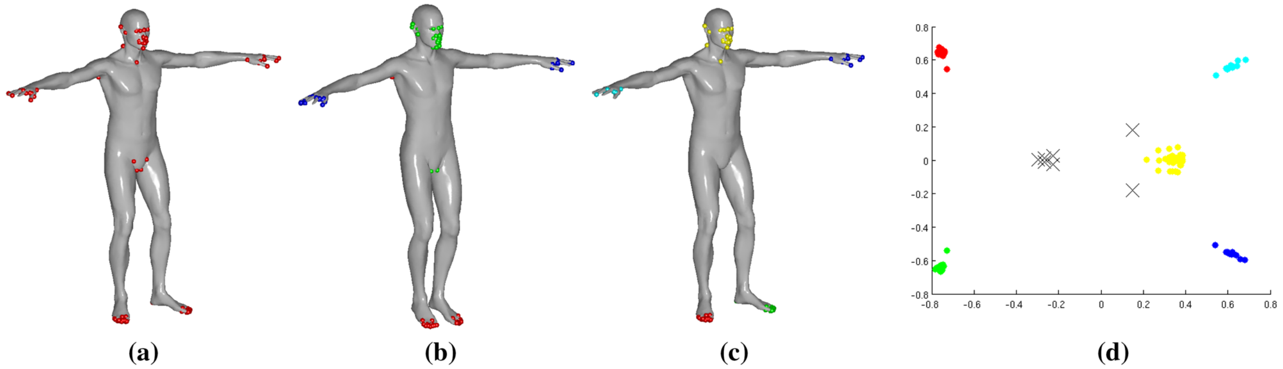

Fig. 5 Local features in our approach: **a** Harris 3D key points, **b** clusters of key points based on their HKS, **c** key points within region of interest, and **d** result of multi-dimensional scaling

The justification to use a key point detector is that it allows us to reduce the amount of information for subsequent tasks. According to our experience, there are vertices which are not representative and, in general, their local geometry is found in all models. Therefore, these vertices are not suitable for discriminating between models from different categories. For example, Harris 3D, as described above, reduces the number of vertices up to a factor of 100, while maintaining the vertices which are highly discriminative in the shapes.

Once we have a set of key points, we need to compute the feature signatures for the 3D models. Let P be a 3D model with n vertices. Let $\mathrm{FS}(P)$ be the feature space of P as defined in Eq. 8. After the detection step, we have a set of vertices $\mathrm{IP}(P) = \{v | v \in P\}$ with m key points. So the feature sub-set induced by the set of key points is defined as

$$\mathrm{FS}_{\mathrm{IP}}(P) = \left\{ \frac{hks(v)}{\|hks(v)\|} | v \in \mathrm{IP}(P) \right\} \tag{9}$$

The final step is to apply the clustering algorithm over the set $\mathrm{FS}_{\mathrm{IP}}(P)$. Then, we proceed in a similar way as described in the previous section.

In addition to represent a 3D model with the most discriminative features, the use of key points allows us to reduce the computations in two ways. First, we need to compute less heat kernel signatures. Second, the clustering algorithm obviously takes less time. This issue is important because it is possible to improve the effectiveness while improving the efficiency. Figure 5b presents a 3D model and its key points after clustering. Vertices with the same color correspond to key points in the same cluster.

4.3 Third approach: feature signatures on clusters of key points

We noted that key points detected by the Harris 3D method tend to cluster, that is, they are placed in discriminative regions of the 3D object. For example, this can be observed in Fig. 5a. Thus, we investigated the possibility of taking advantage of this fact. For example, by finding regions with high concentration of key points, we are able to discard the isolated ones, as key points not belonging to any cluster could be noise on the mesh. By discarding them, we ensure that these points do not interfere in the matching process.

For this approach, we need to find clusters in the geodesic space of a surface. The geodesic distance between two points on a mesh is the shortest distance between these points going through points on the mesh. We use geodesic distances to perform the clustering because we want that the local descriptors are invariant to non-rigid transformations. In this work, we used the fast marching algorithm proposed by Kimmel et al. [22] to compute geodesic distances.

For detecting the regions of interest on a mesh, we transform the set of detected key points under the geodesic metric into a new 2D Euclidean space using the multi-dimensional scaling method (MDS). This method transforms a set of points into a low-dimensional space preserving a metric defined in the original set. Commonly, it is used for visualization of high-dimensional spaces. We propose using MDS because with this method the distances between points in the projected space correspond as close as possible to the distances in the original space. We mapped to two dimensions because it facilitates choosing a centroid for the clustering. This is important because the centroid defines the geodesic region (i.e., cluster) of a group of key points. We also mapped to two dimensions because geodesic distances are measured on a 2D manifold embedded in a 3D space. Thus, we expect that distances in the projected 2D space are good approximations of the original geodesic distances. In this paper, we use the SMACOF algorithm for computing the multi-dimensional scaling [23].

After the MDS step, we obtain a set of 2D points which approximates the geodesic distances in an Euclidean sense. So, we apply the clustering algorithm for detecting clusters of key points. Note that the clustering algorithm discards points not belonging to any cluster (noisy key points). Finally,

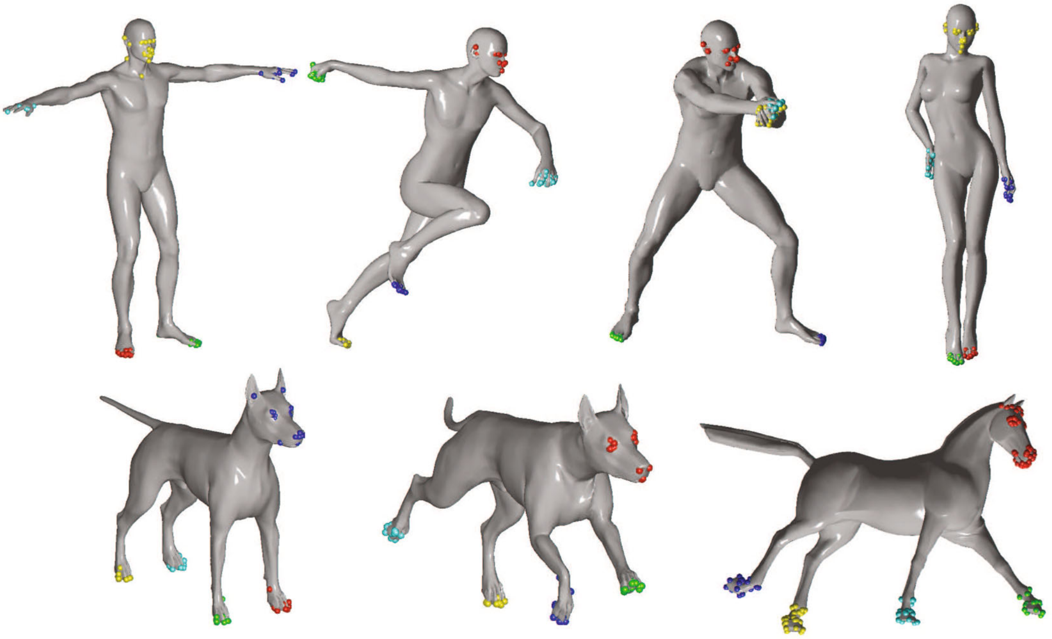

Fig. 6 Clusters of key points detected on several 3D shapes

each region is composed by the interest points inside the region. It is worth noting that one could have directly used a clustering algorithm like k-medoids in the geodesic space. The disadvantages of such an approach are that k-medoids would not allow us to discard noisy key points, and it would require that the center of each cluster is an actual key point. We do not have these restrictions by first mapping the key points using MDS and then performing the clustering.

Once having the set of detected regions, we proceed to describe them. Let R be the set of regions, with $|R| = n$, from a shape P. Each region contains a set of key points which are inside the region. Then, we compute the heat kernel signature for each point. At this stage, we have a set of regions represented by the normalized descriptors of their corresponding key points. Finally, we compute a feature signature for the 3D object, where each region is represented by a unique descriptor and a weight as in Eq. 4. In the feature signature, the element c_i^P is the average of the normalized heat kernel signatures in the region, and the weight w_i^P is proportional to the number of elements in R_i with respect to the total number of key points after the clustering process.

Figure 5c shows a 3D shape with key points belonging to the same cluster. In addition, Fig. 5d shows the result of applying the multi-dimensional scaling over the key points. Colors in Fig. 5d correspond to colors in Fig. 5c. We present additional results of our technique for finding clusters of key points in Fig. 6.

The guarantee of our method to obtain meaningful mid-scale features is the evidence found in a previous study [24]. In that paper, we showed that regions computed from agglomerations of small-scale features are highly repeatable. The experiments even revealed that strong local deformations can be overcome by detecting the regions that contain many local protrusions.

5 Effectiveness evaluation

In this section, we present the dataset and the experimental results obtained to evaluate the effectiveness of our method. Experimental results are divided into two parts. First, we evaluate the three presented approaches and the application of the SQFD. Second, we compare our technique with methods from the state of the art. Our experiments ran on a workstation Intel Core 2 Duo 3.0 Ghz and 4 GB RAM. To evaluate our experiments, we used common retrieval measures such as precision–recall graphs, R-precision, nearest neighbor, and mean average precision (MAP).

5.1 Dataset

We built a dataset containing deformable shapes from two commonly used datasets for shape retrieval tasks: the TOSCA dataset [25] and the Sumner dataset [26]. The complete dataset has 310 models in 16 classes ranging from human models to different kinds of animals. We discarded classes with less than three models per class within the TOSCA dataset, as they could bias the results. In addition, in the Sumner dataset, we discarded models that are the product of collapse animations, as they do not necessarily represent deformable shapes. Furthermore, all models were pre-processed to have approximately 10,000 vertices and surface area of 1.0.

5.2 Implementation issues

To compute the descriptors, we first computed the Laplace–Beltrami operator. Then, we computed the 300 largest eigenvalues and their associated eigenvectors. The dimension of the resulting heat kernel signatures was 100, that is, we took 100 values for t in the interval $[t_{\min}, t_{\max}]$ where

$$t_{\min} = \log |(4 \log 10)/\lambda_{300}| \quad \text{and}$$
$$t_{\max} = \log |(4 \log 10)/\lambda_{2}|$$

With respect to the Harris 3D key points, we used adaptive neighborhoods with size 0.01 of the diagonal of the bounding box of the model. In addition, we set $k = 0.04$ and we took the 1 % of vertices with the highest response as key points.

In addition, it is worth mentioning how we used the dataset for obtaining the results. We computed the feature signatures for each model in the dataset. Next, we simulated a content-based system in which each model of the collection was presented as query. Thus, we re-computed the feature signatures for each query and retrieved a ranked list applying the proposed methods. The queried shape was removed for the collection before searching. All our evaluations consider the average effectiveness and efficiency.

5.3 Results

In this section, we present the obtained results of the three proposed approaches: feature signatures on all vertices (SQFD-All), feature signatures on key points (SQFD-IP), and feature signatures on clusters of key points (SQFD-Cluster). Table 2 shows the configuration of parameters for the clustering used for each method.

To decide the best configuration for the SQFD, we experimented with the ground distance and the Gaussian similarity function (which has proven to be effective in image retrieval [19]) using the SQFD-Cluster approach as baseline. The results can be seen in Fig. 7 and Table 3. For the rest of the experiments, we use the L_2 distance and the Gaussian similarity function with $\alpha = 0.9$.

Table 2 Clustering parameters for each proposed technique

Method	β	δ	Nm	Iter
SQFD-All	0.1	0.2	30	10
SQFD-IP	0.1	0.2	10	10
SQFD-Cluster	0.1	0.2	10	10

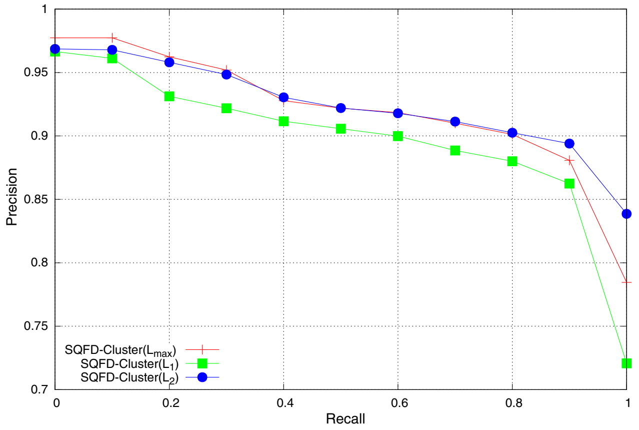

Fig. 7 Recall–precision plots for each variant regarding the ground distance for the SQFD

Table 3 Evaluation measures for the several ground distances applied on SQFD-Cluster

Ground distance	NN	RP	MAP
L_1	0.9607	0.8715	0.8984
L_2	0.9651	**0.9068**	**0.9253**
L_∞	**0.9738**	0.8951	0.9221

Bold values show the best result per measure

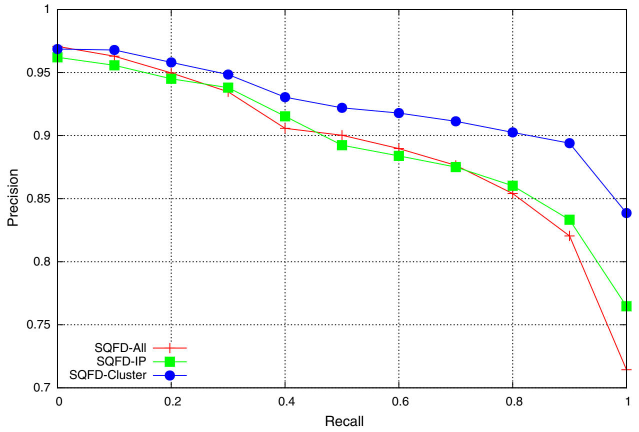

Fig. 8 Recall–precision plots for each of our proposed methods

Figure 8 shows the recall–precision plots for each variant of our method. From the figure, it is clear that SQFD-Cluster outperforms the other two variants, which systematically drop after recall 0.5. However, it is difficult to figure out which is the best between SQFD-All and SQFD-IP. For that reason, Table 4 presents a summary of the results with the other criteria. The variant SQFD-IP is slightly better than SQFD-All taking into account the ranking list (RP and MAP). However, for the nearest neighbor measure, SQFD-All is as good as SQFD-Cluster.

The SQFD-Cluster variant shows an interesting improvement with respect to SQFD-All and SQFD-IP. In our opinion,

Table 4 Evaluation measures on our approaches

Method	NN	RP	MAP
SQFD-All	0.9651	0.8423	0.8906
SQFD-IP	0.9476	0.8491	0.8941
SQFD-Cluster	**0.9651**	**0.9068**	**0.9253**

Bold values show the best result per measure

Table 5 First column: average number of feature signatures, second column: average time for extracting the feature signatures to a query object, third column: average time for computing the SQFD from the query object to all shapes in the collection

Method	Num. Feat.	Sig. (s)	Dist. (s)
SQFD-All	14	1.1623	0.1103
SQFD-IP	5	0.3093	0.0148
SQFD-Cluster	9	1.0333	0.0511

there are two reasons for this behavior. First, SQFD-Cluster considers spatial information for obtaining the feature signatures. The process of detecting the cluster of key points ensures that the signatures represent similar and close local features. Second, despite SQFD-IP is applied on key points, the clustering is still done in the feature space, that is, there could be key points representing different parts of a model belonging to the same feature signature. Therefore, SQFD-Cluster combines the information contained in local descriptors, while effectively represents a spatial region of the mesh by regarding their proximity.

Next, we present results regarding the query time. Table 5 shows the average time for extracting the feature signatures from a query, and subsequently applying the SQFD for obtaining the distance to all shapes within the collection. In addition, we report the average number of features per signature for each variant.

The scenario of query time is favorable to SQFD-IP. Obviously, this method considers a reduced number of descriptors in the clustering, so the computation of the feature signatures is fast. In addition, as it contains the least average number of signatures, the SQFD is computed quickly. In contrast, SQFD-All took more time due to the large number of descriptors (one per vertex) used in the clustering. On the other hand, SQFD-Cluster took a considerable time to compute the signatures. This is because this variant considers additional tasks such as calculation of geodesic distances and multi-dimensional scaling. Therefore, the use of some variant depends on the application and the required efficiency or effectiveness.

5.4 Comparison with state-of-the-art methods

Our method was also compared to methods from the state of the art. We chose two representative techniques: bag-of-feature approach with heat kernel signatures (similar in spirit to the Shape Google technique proposed by Bronstein et al. [3]), and ShapeDNA which is an effective global technique for deformable shapes. Following, we briefly describe each technique and present the parameter configuration.

- *Bag-of-features for local aggregation* Bronstein et al. [3] proposed to use a bag-of-feature approach with heat kernel signatures extracted from a shape. In this work, we used vocabulary sizes of 32 (BoF 32) and 48 (BoF 48) calculated via k-means clustering.
- *ShapeDNA* Reuter et al. [27] proposed a descriptor invariant to deformable shapes. The method computes the Laplace–Beltrami operator for a shape, and subsequently, it performs an Eigen-decomposition. A few of the eigenvalues with the lowest magnitude are considered as descriptor for the whole shape. In this work, we consider 10 eigenvalues for each shape.

Figure 9 shows the recall–precision plots of our method and the aforementioned techniques. In addition, Table 6 presents results with other criteria. Clearly, SQFD-Cluster outperforms BoF and Shape DNA methods.

With respect to the query time, Table 7 shows the average query time for computing a ranked list from the collection. Note that in the rest of this section, we provide results for naïve implementation of feature signatures extraction and

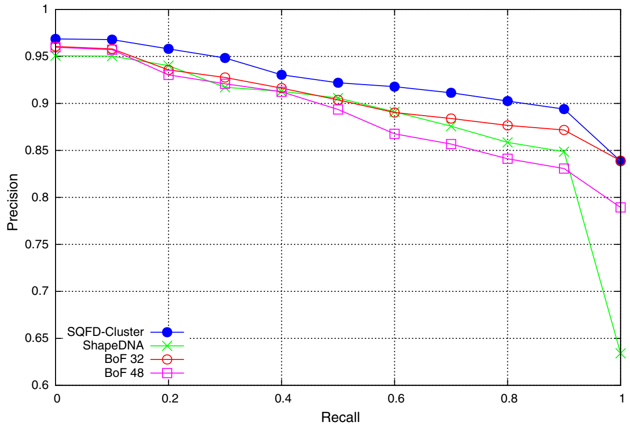

Fig. 9 Recall–precision plots comparing our method with state-of-the-art methods

Table 6 Evaluation measures of the compared methods

Method	NN	RP	MAP
ShapeDNA	0.9301	0.8559	0.8906
BoF 32	0.9576	0.8721	0.9010
BoF 48	0.9476	0.8500	0.8827
SQFD-Cluster	**0.9651**	**0.9068**	**0.9253**

Bold values show the best result per measure

Table 7 Query time for each compared method

Method	Query time
ShapeDNA	0.0014
BoF 32	0.5468
BoF 48	0.6551
SQFD-Cluster	1.0844

querying using simple sequential scan. A scalable implementation of our approach is discussed in the next section.

As can be seen, ShapeDNA is the fastest method, as it considers only to apply a distance function (for instance, L_2) between descriptors. On the other hand, SQFD-Cluster takes approximately twice the time taken by BoF methods. Obviously, most of the time taken by BoF is to compute the bag of features, and subsequently to perform the distance computations. With respect to SQFD-Cluster, there is a trade-off between effectiveness and efficiency. Of course, one could use SQFD-Cluster to gain effectiveness at cost of efficiency issues. However, if the efficiency is a matter, SQFD-IP is a good choice too. This variant is the fastest of our proposed method and its MAP is only slightly below than BoF-32 effectiveness.

There are two remarkable aspects with respect to efficiency. First, BoF requires to construct a visual vocabulary. This phase is performed off-line and considers a clustering task over the complete set of descriptors of a dataset. In our experiments, this process took approximately two hours. In contrast, our proposals do not require any kind of preprocessing tasks. Second, there is a concern regarding the efficiency in memory. BoF requires to store the complete set of descriptors in memory for clustering. In this paper, we used 310 models with approximately 10,000 descriptors per model, so the number of descriptors is approximately 3.1 M. Each descriptor was represented by 100 float numbers, so the clustering process needed 2.3 GB of memory. Although in our experiments it was possible to meet this memory requirement, clearly it is an issue when the collection grows.

Finally, we claim that our methods can be potentially used for large collections due to the high effectiveness, and their efficiency is good in memory and comparable in time. In addition, our methods allow us to have a dynamic system, where new models can be easily added into the system.

6 Scalability evaluation

The representation of 3D objects by feature signatures can bring benefits in terms of effectiveness; however, such representation requires also a more complex similarity function. Since we model the similarity by the signature quadratic form distance that has quadratic time complexity (depending on the number of tuples in feature signatures), the retrieval system cannot rely on a simple sequential search of the database. During the last decade, there have been proposed several approaches to efficiently process similarity queries when using models based on feature signatures and the SQFD. The approaches rely either on exact [28,29] or approximate [30] search strategies. We do not consider approximate search strategies as they trade precision for effectiveness and in such cases the ShapeDNA method could provide better trade-off. In our work, we focus on exact search strategies employing an index suitable for SQFD.

6.1 Metric space approach

Since SQFD was proved to be a metric distance, the metric space model of database indexing can be used to speedup the similarity search tasks over large datasets of feature signatures [28]. Generally, the metric space properties (identity, positiveness, symmetry, triangle inequality) of any distance function can be used for efficient similarity search under that distance. More specifically, using metric properties tight lower bounds of SQFD can be cheaply determined (instead of computing the original expensive distance) and used to filter a majority of the irrelevant feature signatures while searching. Let $\mathbb{U}$ be the universe of valid feature signatures, and let δ denote the SQFD distance. Given feature signatures $o, p, q \in \mathbb{U}$ and their precomputed SQFD distances $\delta(o, p), \delta(q, p)$, the distance between objects o and q is lower bounded by expression $\text{LB}_{\triangle}(\delta(o, q)) = |\delta(o, p) - \delta(q, p)|$. As a consequence, if (q, r) represents a range query and $\text{LB}_{\triangle}(\delta(o, q)) > r$, then object o can be safely filtered without the explicit evaluation of expensive distance $\delta(o, q)$.

Let $\mathbb{S} \subset \mathbb{U}$ be a set of objects, and let $\mathbb{P} \subset \mathbb{S}$ be a set of reference objects (so called pivots). The lower bounding technique requires an indexing phase of $\mathbb{S}$, where distances between database objects $o_i \in \mathbb{S}$ and the reference objects $p_j \in \mathbb{P}$ have to be evaluated and organized in an indexing structure. Although there have been designed many metric indexing techniques employing lower bounding, in this paper, we demonstrate scalability of our new models using a simple yet efficient structure called Pivot Table [31].

In the Pivot Table index, the whole distance matrix between all database objects and all pivots is precomputed and stored. When using Pivot Table to efficiently process a similarity query, first the distances between query object and all pivots have to be evaluated. Then the lower bound distance between each database object o_i and query object q is evaluated as $\text{LB}_{\triangle}(\delta(o_i, q)) = \max_{\forall p_j \in \mathbb{P}} |\delta(o_i, p_j) - \delta(q, p_j)|$. The lower bound distance can be used directly to improve efficiency of range queries; however, k-NN queries with unknown initial query radius require more sophisticated query processing strategies. The strategies try to find the

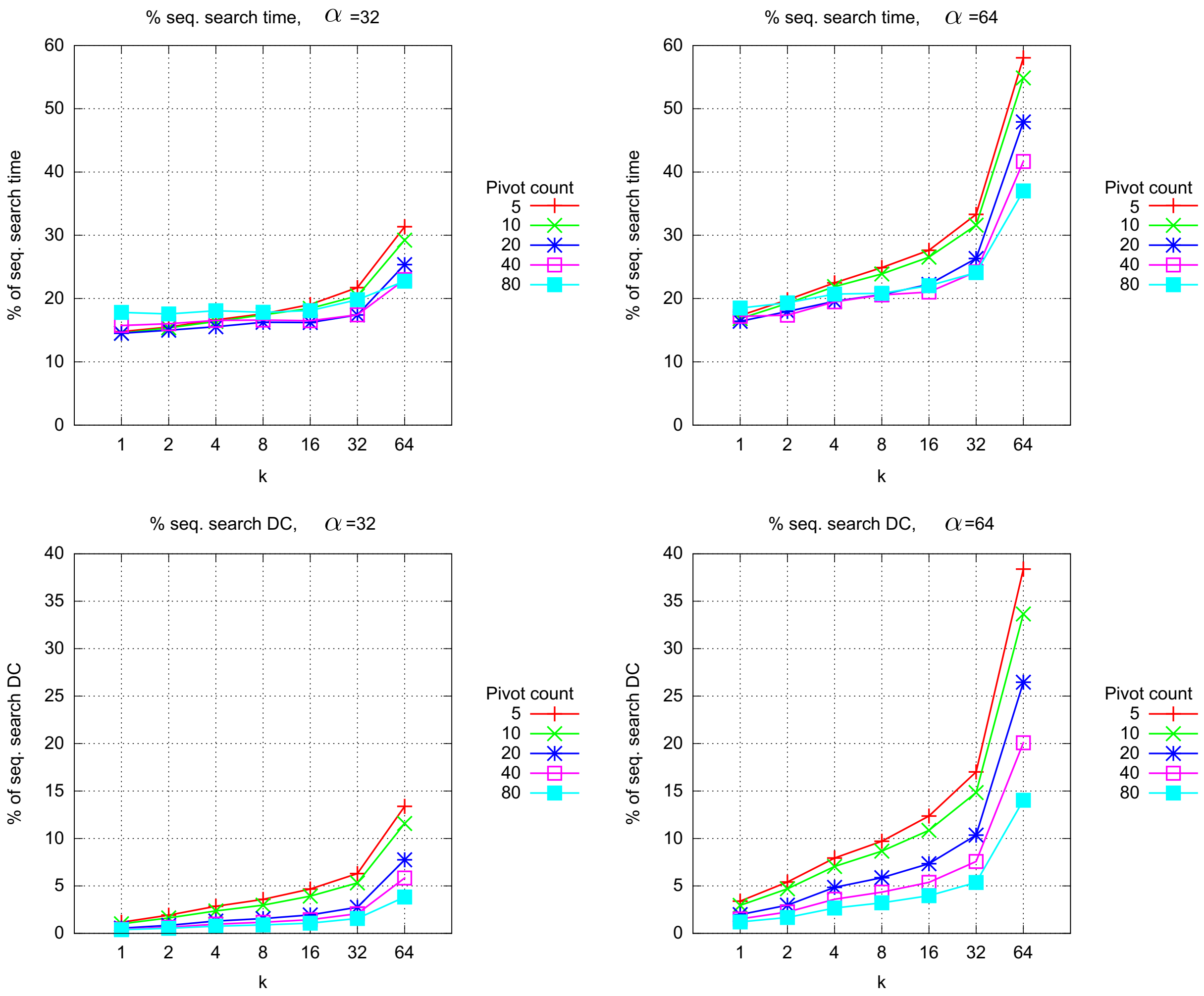

Fig. 10 PivotTable k-NN search efficiency for SQFD-All-6d feature signatures. The *top plots* show the percentage between PivotTable time and sequential search time. The *bottom plots* show the percentage between distance calculations with PivotTable and the distance calculation with sequential search. Low values of percentage is better

best candidates from the database as soon as possible and thus to update/decrease the actual query radius more rapidly. In our experiments, we use the strategy that sorts the database objects in ascending order according to their lower bound distance to the query object. More details about the Pivot Table index can be found in Navarro [31] and Mico et al. [32].

6.2 Experimental results

The experiments are based on the dataset of the SHREC'15 Track: scalability of non-rigid shape retrieval [33]. The dataset consists of more than 90,000 3D objects, from which 229 are annotated and classified. For our experiments we selected 80,000 3D objects from the dataset, including all 229 annotated objects. The annotated objects were used as query objects. At each query. the actually considered query object was removed from the database before performing it.

As we now want to study the scalability issue, for this experiment we only consider the SQFD-All approach, which was the approach that computes a descriptor for each vertex on the mesh. As the dimensionality of the descriptors is an important parameter for studying the scalability of the techniques, we extracted two different sets of feature signatures from all the 3D objects:

- SQFD-All-100d signatures: consist of 100-dimensional centroids (high dimensionality). The local descriptors are heat kernel signatures which were computed following the same guidelines as Sect. 5.2.
- SQFD-All-6d signatures: consist of six-dimensional centroids (low dimensionality). The local descriptors are Scale-invariant Heat Kernel Signatures as described by Bronstein and Kokkinos [17].

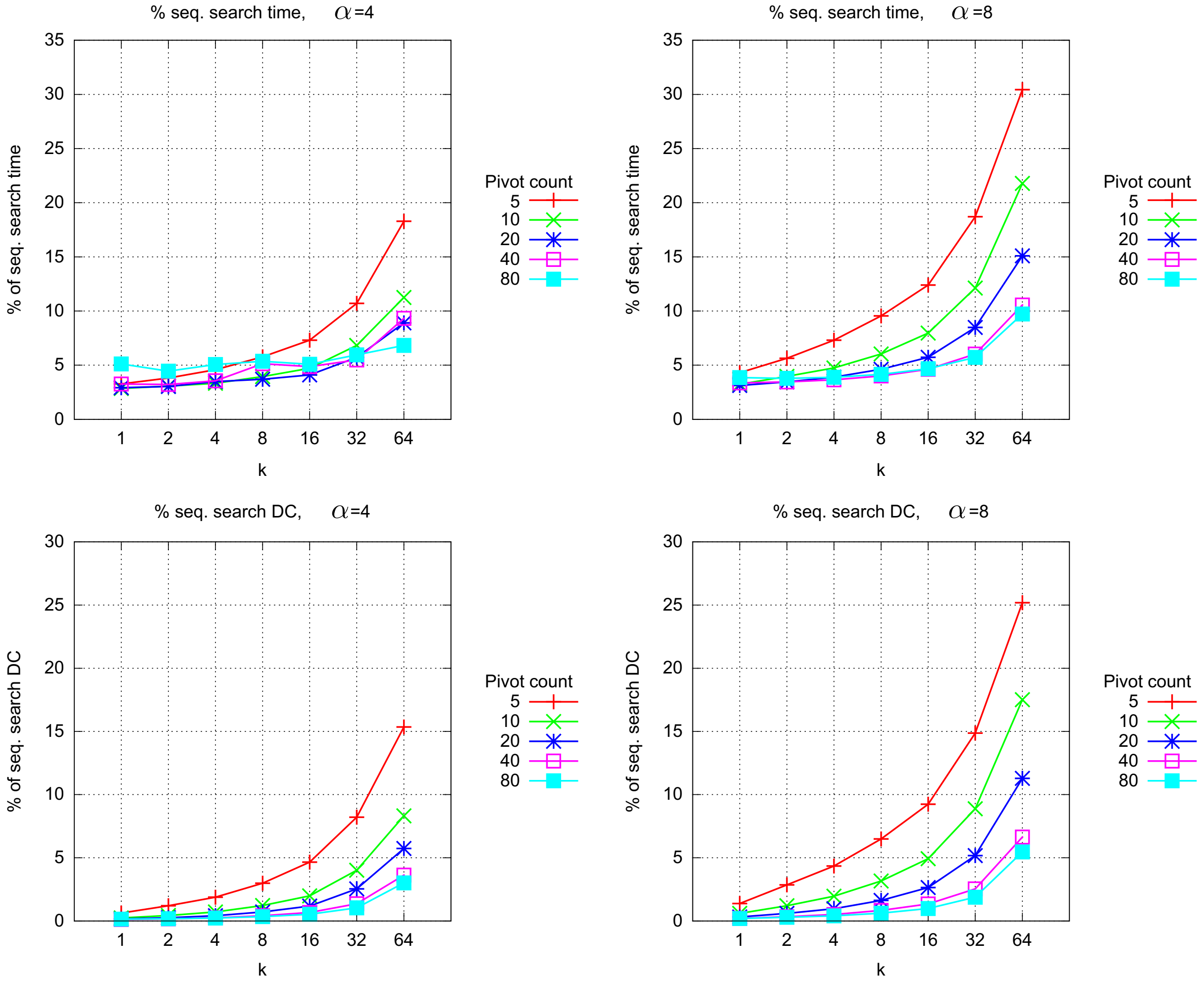

Fig. 11 PivotTable k-NN search efficiency for SQFD-All-100d feature signatures. The *top plots* show the percentage between PivotTable time and sequential search time. The *bottom plots* show the percentage between distance calculations with PivotTable and the distance calculation with sequential search. Low values of percentage is better

For each set, we have investigated the effect of the SQFD α parameter (see Sect. 3.2) on precision, considering just the annotated objects. For all the experiments in this section, we use the Gaussian similarity function for the SQFD. Whereas for SQFD-All-6d the optimal value was $\alpha = 64$, for SQFD-All-100d the optimal value was $\alpha = 8$. As reported by Beecks et al. [28] in the image retrieval domain, the optimal value of α often results in high intrinsic dimensionality that negatively affects the performance of metric indexes. We have observed similar behavior in our experiments: using lower values of α results in slightly worse effectiveness but also in lower intrinsic dimensionality. Therefore, we have considered also values $\alpha = 32$ for SQFD-All-6d and $\alpha = 4$ for SQFD-All-100d feature signatures.

For SQFD-All-6d feature signatures, the sequential search took on average about 370 ms and for SQFD-All-100d feature signatures, the sequential search took on average about 1990 ms. To create variants of metric PivotTables, we have randomly selected sets of pivots. Although a high number of pivots improve the filtering power of PivotTable, it also increases the lower bound calculation time. Therefore, we have considered different numbers of pivots for the PivotTable: 5, 10, 20, 40 and 80. In the experiments, we have investigated the efficiency of the kNN search, where k ranged from 1 to 64. The experiments were run on a 64-bit Windows Server 2008 R2 Standard with Intel Xeon CPU X5660, 2.8 GHz.

Figure 10 shows the performance of PivotTable index on SQFD-All-6d feature signatures. The figure shows that the number of pivots and the parameter α affect the efficiency of the retrieval. We may observe that for queries with lower k, lower number of pivots is sufficient. For queries with higher k (i.e., with higher radius), the filtering power of PivotTable decreases. In such cases, the higher number of pivots can help to better estimate the lower bounds and thus to better estimate the real query radius and filter more non-relevant objects.

We may observe similar effect also for distance spaces with higher intrinsic dimensionality ($\alpha = 64$), where higher number of pivots improves efficiency. Note also the difference between time and distance calculations. Whereas distance calculations decrease with higher number of pivots (in the bottom line), the time considering also lower bound estimation overhead shows worse results for cheap queries (e.g., for $k = 1$). Since the SQFD distance is less expensive for SQFD-All-6d feature signatures, the retrieval time can be reduced just to 15–20 % of the sequential search time for queries with $k \leq 16$ (speed-up factor $>5\times$).

Figure 11 shows the efficiency results of k-NN search for SQFD-All-100d feature signatures. The general observations are similar as for SQFD-All-6d feature signatures. Since the SQFD distance is more expensive for SQFD-All-100d feature signatures, the PivotTable overhead is less significant than the number of distance computations. We may observe that the retrieval time can be reduced up to 3–5 % of the sequential search for queries with $k \leq 16$ (speed-up factor $>20\times$).

We conclude that even the simple PivotTable index can be used to significantly speed up k-NN search with our new descriptors and the SQFD distance.

7 Conclusions

In this paper, we presented a novel approach for 3D shape retrieval using local features. Our method to find clusters of key points has shown to have a positive impact in retrieval. This fact can be noted in the high effectiveness achieved by the variant SQFD-Cluster. Likewise, the SQFD has proven to be effective in our framework. So the comparison of two 3D objects represented as a set of features can be addressed efficiently with this distance. In addition, the use of the SQFD has enabled the possibility to implement our method in dynamic collections, hence making a retrieval system scalable.

An important aspect of our results is that, depending of the requirements for a retrieval system, our method offers several characteristics. On the one hand, if we are interested in the retrieval results, SQFD-Cluster can be used because of its high effectiveness. On the other hand, if we are interested in the query time, SQFD-IP performed better in this case. Anyway, our approach (in its different variants) performed better than state-of-the-art methods. Finally, we also show how to make the method scalable using a Pivot Table, taking advantage of the fact that the SQFD is a metric distance.

Acknowledgements This work has been partially supported by Programa Nacional de Innovación para la Competitividad y Productividad, INNOVATE Perú, Grant Nr. 280-PNICP-BRI-2015. This work has been also supported by Charles University projects P46 and SVV-2016-260331. Benjamin Bustos has been funded by FONDECYT (Chile) Project 1140783 and the Millennium Nucleus Center for Semantic Web Research, Grant Nr. NC120004.

References

1. Beecks, C.: Distance-based similarity models for content-based multimedia retrieval. In: Dissertation, Fakultt fr Mathematik, Informatik und Naturwissenschaften, RWTH Aachen University (2013)
2. Beecks, C., Uysal, M.S., Seidl, T.: Signature quadratic form distances for content-based similarity. In: Proc. ACM Int. Conf. on Multimedia, MM '09, pp. 697–700. ACM, New York (2009)
3. Bronstein, A., Bronstein, M., Guibas, L., Ovsjanikov, M.: Shape google: geometric words and expressions for invariant shape retrieval. ACM Trans. Comput. Graph. **30**(1), 1:1–1:20 (2011)
4. Abdelrahman, M., Farag, A., Swanson, D., El-Melegy, M.T.: Heat Diffusion over weighted manifolds: a new descriptor for textured 3D non-rigid shapes. In: Proc. IEEE Conf. Comput. Vision and Pattern Recognit., CVPR, pp. 187–195 (2015)
5. Tabia, H., Laga, H., Picard, D., Gosselin, P.H.: Covariance descriptors for 3d shape matching and retrieval. In: Proc. IEEE Conf. Comput. Vision and Pattern Recognit., pp. 4185–4192. IEEE Computer Society, Washington, DC (2014)
6. Bai, X., Bai, S., Zhu, Z., Latecki, L.: 3D shape matching via two layer coding. IEEE Trans. Pattern Anal. Mach. Intell. **37**(12), 2361–2373 (2015)
7. Savelonas, M.A., Pratikakis, I., Sfikas, K.: Partial 3D object retrieval combining local shape descriptors with global fisher vectors. In: Pratikakis, I., Spagnuolo, M., Theoharis, T., Gool, L.V., Veltkamp, R. (eds.) Proc. Eurographics Workshop on 3D Object Retr., pp. 23–30. The Eurographics Association (2015)
8. Litman, R., Bronstein, A.M., Bronstein, M.M., Castellani, U.: Supervised learning of bag-of-features shape descriptors using sparse coding. Comput. Graph. Forum **33**(5), 127–136 (2014)
9. Liu, Z., Bu, S., Han, J.: Locality-constrained sparse patch coding for 3d shape retrieval. Neurocomputing **151, Part 2**, 583–592 (2015)
10. Fang, Y., Xie, J., Dai, G., Wang, M., Zhu, F., Xu, T., Wong, E.: 3D deep shape descriptor. In: Proc. IEEE Conf. Comput. Vision and Pattern Recognit., pp. 2319–2328 (2015)
11. Bu, S., Cheng, S., Liu, Z., Han, J.: Multimodal feature fusion for 3D shape recognition and retrieval. IEEE Multimed. **21**(4), 38–46 (2014)
12. Xie, J., Fang, Y., Zhu, F., Wong, E.: DeepShape: deep learned shape descriptor for 3D shape matching and retrieval. In: Proc. IEEE Conf. Comput. Vision and Pattern Recognit., CVPR, pp. 1275–1283 (2015)
13. Bustos, B., Keim, D.A., Saupe, D., Schreck, T., Vranic, D.V.: Feature-based similarity search in 3D object databases. ACM Comput. Surv. **37**(4), 345–387 (2005)
14. Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., Yu, F.: Shapenet: An information-rich 3d model repository. CoRR arxiv:1512.03012 (2015)
15. Sun, J., Ovsjanikov, M., Guibas, L.J.: A concise and provably informative multi-scale signature based on heat diffusion. Comput. Graph. Forum **28**(5), 1383–1392 (2009)
16. Belkin, M., Sun, J., Wang, Y.: Discrete laplace operator on meshed surfaces. In: Proc. Symposium on Comput. Geom., pp. 278–287. ACM (2008)
17. Bronstein, M., Kokkinos, I.: Scale-invariant heat kernel signatures for non-rigid shape recognition. In: Proc. IEEE Conf. Comput. Vision and Pattern Recognit., pp. 1704–1711 (2010)
18. Hafner, J., Sawhney, H.S., Equitz, W., Flickner, M., Niblack, W.: Efficient color histogram indexing for quadratic form distance functions. IEEE Trans. Pattern Anal. Mach. Intell. **17**(7), 729–736 (1995). doi:10.1109/34.391417
19. Beecks, C., Uysal, M.S., Seidl, T.: Signature quadratic form distance. In: Proc. ACM Int. Conf. on Image and Video Retr., CIVR '10, pp. 438–445. ACM, New York (2010)

20. Leow, W.K., Li, R.: The analysis and applications of adaptive-binning color histograms. Comput. Vis. Image Underst. **94**, 67–91 (2004)
21. Sipiran, I., Bustos, B.: Harris 3D: a robust extension of the Harris operator for interest point detection on 3D meshes. Vis. Comput. **27**, 963–976 (2011)
22. Kimmel, R., Sethian, J.A.: Computing geodesic paths on manifolds. In: Proc. Natl. Acad. Sci. USA, pp. 8431–8435 (1998)
23. Borg, I., Groenen, P.: Modern multidimensional scaling: theory and applications. Springer, Berlin (2005)
24. Sipiran, I., Bustos, B.: Key-components: detection of salient regions on 3d meshes. Vis. Comput. **29**(12), 1319–1332 (2013). doi:10.1007/s00371-013-0870-9
25. Bronstein, A., Bronstein, M., Kimmel, R.: Numerical geometry of non-rigid shapes, 1st edn. Springer Publishing Company, Berlin (2008)
26. Sumner, R.W., Popović, J.: Deformation transfer for triangle meshes. ACM Trans. Graph. **23**, 399–405 (2004)
27. Reuter, M., Wolter, F.E., Peinecke, N.: Laplace-Beltrami spectra as Shape-DNA of surfaces and solids. Comput. Aided Des. **38**(4), 342–366 (2006)
28. Beecks, C., Lokoč, J., Seidl, T., Skopal, T.: Indexing the signature quadratic form distance for efficient content-based multimedia retrieval. In: Proc. 1st ACM Int. Conf. on Multimedia Retr., pp. 24:1–24:8. ACM, New York (2011)
29. Hetland, M., Skopal, T., Lokoč, J., Beecks, C.: Ptolemaic access methods: challenging the reign of the metric space model. Inf. Syst. **38**, 989–1006 (2013)
30. Lokoč, J., Grošup, T., Skopal, T.: On scalable approximate search with the signature quadratic form distance. In: Brisaboa, N., Pedreira, O., Zezula, P. (eds.) Proc. 7th Int. Conf. on Similarity Search and Applications, Lecture Notes in Computer Science, vol. 8199, pp. 312–318. Springer, Berlin, Heidelberg (2013)
31. Navarro, G.: Analyzing metric space indexes: what for? In: Proc. 2nd Int. Workshop on Similarity Search and Applications, pp. 3–10. IEEE Computer Society (2009)
32. Mico, M.L., Oncina, J., Vidal, E.: A new version of the nearest-neighbour approximating and eliminating search algorithm (aesa) with linear preprocessing time and memory requirements. Pattern Recognit. Lett. **15**(1), 9–17 (1994)
33. Sipiran, I., Bustos, B., Schreck, T., Bronstein, A.M., Bronstein, M., Castellani, U., Choi, S., Lai, L., Li, H., Litman, R., Sun, L.: Scalability of Non-Rigid 3D Shape Retrieval. In: Pratikakis, I., Spagnuolo, M., Theoharis, T., Gool, L.V., Veltkamp, R. (eds.) Proc. Eurographics Workshop on 3D Object Retr. The Eurographics Association (2015)

Ivan Sipiran is a researcher in the Section of Informatics at the Pontificia Universidad Católica del Perú-PUCP in Lima, Perú. Previously, I held a Post-doc position at the University of Konstanz. I got a PhD in the Department of Computer Science at the University of Chileunder the advice of Dr. Benjamin Bustos. In addition, I was a research assistant in PRISMA Research Group. I received the Bachelor degree in Computer Science from the Computing School at National University of Trujillo, Perú in 2005.

Jakub Lokoč received the doctoral degree in software systems from the Charles University in Prague, Czech Republic. He is an assistant professor in the Department of Software Engineering at the Charles University in Prague, Faculty of Mathematics and Physics, Czech Republic. He is a member of SIRET research group and his research interests include metric access methods, multimedia retrieval and exploration, and similarity modeling.

Benjamin Bustos is an Associate Professor at the Department of Computer Science, University of Chile. He is the head of the PRISMA Research Group. He leads research projects in the domains of multimedia retrieval, video copy detection, sketch-based image retrieval, and retrieval of handwritten documents. His interests include similarity search, multimedia retrieval, 3D object retrieval, and (non)-metric indexing. He has a doctoral degree in natural sciences from the University of Konstanz, Germany (2006).

Tomáš Skopal is an Associate Professor in the Department of Software Engineering at the Charles University in Prague, Faculty of Mathematics and Physics, Czech Republic. His research interests are metric access methods, database index structures, multimedia databases, and similarity modeling. He has published extensively in the area of metric and nonmetric similarity search. He has a doctoral degree in computer science and applied mathematics (2004) from the Technical University of Ostrava, Czech Republic.