3D Shape Matching via Two Layer Coding

Xiang Bai, *Senior Member, IEEE,* Song Bai, *Student Member, IEEE,* Zhuotun Zhu, and Longin Jan Latecki, *Senior Member, IEEE*

Abstract—View-based 3D shape retrieval is a popular branch in 3D shape analysis owing to the high discriminative property of 2D views. However, many previous works do not scale up to large 3D shape databases. We propose a Two Layer Coding (TLC) framework to conduct shape matching much more efficiently. The first layer coding is applied to pairs of views represented as depth images. The spatial relationship of each view pair is captured with so-called eigen-angle, which is the planar angle between the two views measured at the center of the 3D shape. Prior to the second layer coding, the view pairs are divided into subsets according to their eigen-angles. Consequently, view pairs that differ significantly in their eigen-angles are encoded with different codewords, which implies that spatial arrangement of views is preserved in the second layer coding. The final feature vector of a 3D shape is the concatenation of all the encoded features from different subsets, which is used for efficient indexing directly. TLC is not limited to encode the local features from 2D views, but can be also applied to encoding 3D features. Exhaustive experimental results confirm that TLC achieves state-of-the-art performance in both retrieval accuracy and efficiency.

Index Terms—3D shape matching, shape retrieval, Bag of Features, large scale, Two Layer Coding

1 Introduction

Stepping into the era of big data, there are many large 3D-collections in digital format, being accessed with a common PC or mobile terminals on the Internet. How to efficiently and effectively perform 3D shape matching has become a crucial issue due to many applications such as 3D model retrieval and categorization, 3D reconstruction, CAD, biological analysis, medical imaging, virtual reality and computer game design.

One of the most important challenges in shape matching is to obtain a good shape (dis)similarity measure for comparing a pair of shape instances. Especially for 3D shape matching, both retrieval accuracy and computational efficiency should be urgently improved due to the increasing number of 3D objects on the internet.

Owing to the recent success of image representation and analysis in the Bag-of-Features (BoF) [1], [2] framework, coding-based methods have attracted much attention in shape analysis community. Since the coding framework can efficiently provide a set-to-set correspondence of local shape features in 3D objects, it accelerates 3D shape matching while preserving the discriminative power of the features at the same time. However, unlike image representation, 3D shape, as a high level feature, can not be fully depicted with only local appearance variations under the BoF framework, since the configuration (*i.e.* the spatial arrangement) of local parts is more or less lost in the current coding approaches for 3D shape matching.

In this paper, we propose a novel coding framework for constructing a compact and robust shape representation for 3D shape matching. The proposed representation is obtained based on a set of 2D views in the format of depth buffer rendered from each 3D object. Our method includes two main stages. First, SIFT [3] features collected from pairs of different 2D views are encoded into a single vector with Vector of Locally Aggregated Descriptors (VLAD) [4]. The single vector can be considered as a compact shape feature that contains the shape information of a pair of projections from a given 3D model. In the second stage, we adopt Vector Quantization (VQ) to encode all the features describing view pairs into a single feature vector representing the whole 3D object. Since each 3D model is then represented by a single feature vector, 3D shape matching can be simply accomplished via vector comparison, which is particularly efficient for retrieval and ranking 3D objects in a large database. A brief pipeline of the proposed coding method is given in Fig. 1.

Different from previous view-based 3D shape matching approaches that treat 2D views independently, the proposed idea of combining a pair of views is motivated by an obvious phenomenon: 3D objects can be distinguished more easily based on two different views than a single view. As illustrated in Fig. 2(a), humans may not be able to recognize some 3D objects given just one single view. However, when two different views are presented together, the recognition becomes a much simpler task, *e.g.*, see Fig 2(b). In other words, 3D objects from different

- *X. Bai, S. Bai and Z. Zhu are with the School of Electronic Information and Communications, Huazhong University of Science and Technology, 1037 Luoyu Road, Wuhan, Hubei 430074, P.R. China. E-mail: xbai, songbai, zhuzhuotun@hust.edu.cn.*
- *L.J. Latecki is with the Department of Computer and Information Sciences, Temple University, 1805 North Broad Street, Philadelphia, PA 19122. E-mail: latecki@temple.edu*

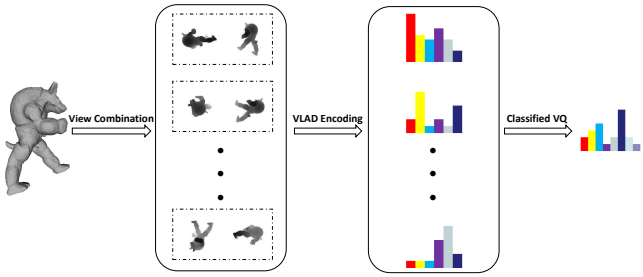


Fig. 1. The pipeline of the proposed coding framework.

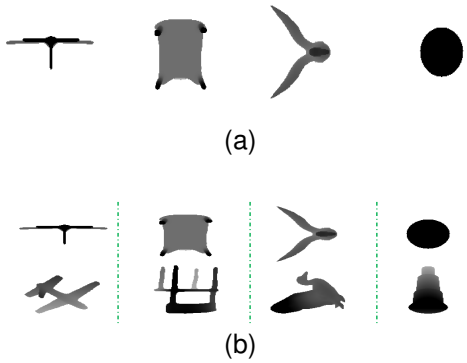


Fig. 2. The difference in the discriminative power between a single view and a view pair. (a) Some examples of unrecognized views; (b) Adding a second view significantly increases the recognition likelihood. This motives our usage of view pairs in encoding 3D shapes.

categories may have similar single views, but they are less likely to have two completely similar view pairs. Another advantage of view pair is the fact that it generates more shape signatures than using single views, which is particularly good for coding methods. For example, given N_v views, the number of combinations of them is $N_v \times (N_v - 1)/2$, and that of permutations is $N_v \times (N_v - 1)$.

Even though the proposed view-pair representation contains more information than the representation of a single view, encoding such features directly is not able to fully describe the spatial arrangement of all the views of a 3D object. Assume that all views of a 3D object are captured on the surface of the unit sphere, then the angle between any two views (treated as projections from the centroid) is known. We utilize this fact to divide view pairs into groups so that any two view pairs in the same group have similar angles. Then the second layer coding is performed for each group, and finally the encoded features from all the groups are concatenated to a single vector. This approach is motivated by the fact that similar objects have similar view pairs that have similar angle, e.g., see Fig. 3. With the above strategy, the spatial relationship of local features (view pairs) is to a large extent preserved during the coding.

To summarize, the proposed method has several merits in comparison with previous 3D shape matching algorithms. First, since the angles between two views of each view pair are invariant to object pose, and no pose normalization with respect to orientation of 3D objects is required. Second, the representation of view pairs is stable and discriminative, since it contains more information than single view, and the angle of a view pair naturally preserves the global spatial configuration of local features in the coding scheme. Third, the proposed two layer coding strategy provides a compact representation for efficient shape matching, which is quite important in large-scale 3D shape retrieval scenarios. Fourth, the feature sampling strategy based on view pair provides a practical description of spatial information of 3D shapes that is preserved during coding. Fifth, the proposed coding framework is not only limited to local features from 2D projections, but also can be directly adopted to depict the spatial configuration of 3D shape signatures on the surface. Extensive experiments on the existing popular 3D shape benchmarks demonstrate that the proposed method achieves state-of-the-art retrieval performance while maintaining high efficiency.

The remainder of this paper is organized as follows. In Section 2, we introduce some related work briefly. The motivation and definition of two layer coding are given in Section 3. In Section 4, experimental results on five benchmark datasets are presented and analyzed. Finally, conclusions are given in Section 5.

2 Related Work

Generally, 3D shape matching methods can be coarsely divided into two categories: model-based methods (*e.g.* [5]) and view-based methods (*e.g.* [6], [7]). For model-based methods, some geometric features are often extracted first, then the correspondence between such features is established by minimizing the matching cost. In the early years, model-based methods have been the main stream, but, how to efficiently perform meaningful feature extraction and how to establish pairwise feature correspondence are the biggest and unsolved obstacles for large-scale 3D model retrieval. View-based methods represent 3D models with a group of 2D projections, which received growing interest in recent years due to their computational efficiency. Besides, fusing multiple complementary features to boost the retrieval accuracy is often adopted, recently. In this paper, we call the approaches on feature fusion as hybrid methods.

2.1 Model-based Methods

The model-based methods usually extract the shape descriptors directly from 3D models with some key point detection/sampling techniques [8], [5], and the (dis)similarity between two 3D shapes is measured by a certain metric in the spatial domain or in the

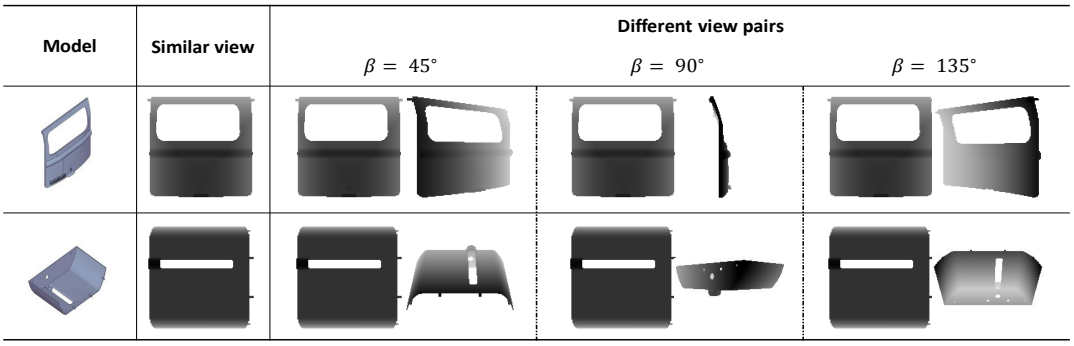


Fig. 3. Two 3D models from the category of "Back Doors" and "Rectangular Housings" from the ESB dataset. As shown in the second column, they have a similar view, but their view pairs that form the same angles are quite different.

spectral domain. In [9], Shape Histogram descriptor is proposed to act as an intuitive and powerful approach for modeling similarity for solid objects. For example, SHELLS is defined as a histogram of distances from the center of mass to points on the surface. Osada *et al*. [10] represent a signature of a 3D object as a shape distribution sampled from a shape function to measure global geometric properties of an object, such as D2 function defined as a histogram of distances between pairs of points on the 3D surface. An enhanced shape function, called the angle distance histogram for inconsistently oriented meshes is proposed in [11]. Spin Images are used in [12] to match surfaces represented as surface meshes. Zaharia *et al*. [13] propose a descriptor capturing the distribution of a shape index over the entire mesh, where a shape index is defined as the angular coordinate of a polar representation of the principal curvature vector. Concrete Radialized Spherical Projection (CRSP) [14] descriptor is proposed to describe a 3D model using a volumetric spherical function, and both Continuous Principal Component Analysis (CPCA) and Normals Principal Component Analysis (NPCA) are used to align the model. Curve analysis is conducted in the 3D surfaces [15] to define a global distance between shapes. In [16], a novel Covariance Descriptor, which uses the covariance of the features instead of the features themselves, is used to perform shape matching. Bronstein *et al*. [17] use multi-scale diffusion heat kernels as geometric words to construct shape descriptors, and spatially close geometric words are considered to create spatially-sensitive bags of features. Partial matching of surfaces represented by triangular meshes is exploited in [18], [19], where Local Surface Descriptors are designed to encode regions of the surface efficiently. Another interesting branch of model-based approaches focuses on establishing the correspondence between the approximate skeletons or medial axes of 3D shapes, which is extremely important for articulation changes of non-rigid shapes [20], [21], [22], [23], [24]. Besides skeleton-based representations, some descriptors extracted on 3D surface [25], [26], [27] are popular as well for handling non-rigid deformations.

Model-based methods can often provide a faithful part correspondence of 3D models, which is quite useful in 3D modeling. However, the computational cost of 3D shape signatures and correspondence matching makes them difficult for 3D shape retrieval in large scale. Though the proposed view pair representations are computed based on 2D projections, they can also be considered as a set of 3D shape signatures collected from a 3D model.

2.2 View-based Methods

View-based methods have been investigated intensively as well in recent years, and it has been demonstrated in [28], [29] that view-based methods achieve competitive overall retrieval performance, due to the existence of the highly discriminative views. Existing view-based methods usually align a given 3D shape to its principal directions with PCA, and project it to several 2D views, which are mostly contours or depth-buffers. Then informative and discriminative features are extracted directly, or learned indirectly, to represent these views. Finally a many-to-many matching strategy, such as the Hungarian method, Dynamic Programming, Shortest Augmenting Path algorithm [30], is adopted to build the correspondence between two sets of view features.

Based on the assumption that if two 3D shapes are similar, they also look similar from all viewing angles, Chen *et al*. [31] propose Light Field Descriptor (LFD), which is composed of Zernike moments and Fourier descriptors, extracted from ten views given by the vertices of a dodecahedron over a hemisphere. Vranic *et al*. [32] design a hybrid descriptor, which is formed using depth buffer images, silhouettes, and ray-extents of a polygonal mesh. Bag of Features (BoF) [1], [2] model, a classical framework widely

used in 2D image representation, is applied to view-based 3D shape analysis in [33] for the first time. In [34], [35], SIFT [3] descriptors are extracted in the projected depth images, and encoded to get a vector representation for each view. The algorithm of Clock-Matching is subsequently investigated to find the best correspondence between two sets of features by considering all possible shape poses. Vectors of Locally Aggregated Tensors (VLAT) [36] are used to encode the visual descriptors by aggregating their tensor products. However, all visual descriptors are put into a single bag, instead of maintaining a separate set of descriptors for each view. Papadakis *et al*. [7] project a 3D shape, normalized by PCA [37], [14], to the lateral surface of a cylinder, and obtain a set of panoramic views represented by 2D Discrete Fourier Transform and 2D Discrete Wavelet Transform. In [38], Compact Multi-View Descriptor (CMVD) is proposed, in which the comparison between 3D shapes is accomplished by the feature matching between selected views using 2D features, such as 2D Polar-Fourier Transform, 2D Zernike Moments, and 2D Krawtchouk Moments.

Besides the aforementioned methods that focus on designing robust descriptors for views, several researchers attach more importance to multi-view matching through some learning-based algorithms. Adaptive Views Clustering (AVC) [39] is proposed to select discriminative views, and the retrieval is performed with a novel Bayesian method.

2.3 Hybrid Methods

Hybrid BoW [40] fuses the feature obtained by Bag-of-Words model without spatial information or a spatially-sensitive descriptor. It achieves good performance on some datasets. The complementarity of 2D and 3D features is observed in [41], where a 2D/3D hybrid descriptor is proposed that consists of 2D features based on depth buffers and 3D features based on spherical harmonics.

Although the hybrid methods perform better in many cases, their main disadvantage is lack of a way to determine the weights of different features in an unsupervised manner, hence many previous algorithms set the weight empirically.

3 Two Layer Coding

Given a query 3D shape with a set of 2D views, the goal of view-based 3D shape retrieval system is to retrieve similar 3D shapes from the database according to some measures defined between two sets of views. The matching between two sets is usually time-consuming. To address this problem, we propose an efficient framework for multi-view matching by coding-based method together with the combination of the spatial arrangement of views. In this section, we introduce the details of the proposed Two Layer Coding (TLC).

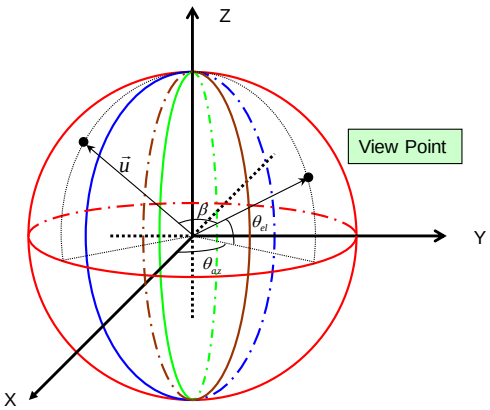


Fig. 4. The illustration of the projection. The two black dots represent two view points. Each projected view is located by the angles θ_{el} and θ_{az}. β is the angle between the two projected views.

3.1 Visual Feature Extraction

Prior to the extraction of visual descriptors, pose normalization is conducted for each 3D shape. However, different from previous algorithms that perform the normalization for rotation invariance using Principal Component Analysis (such as CPCA [37], NPCA [14], we only normalize the scale and translation of the 3D shape in our framework to eliminate their negative influence on the similarity measure between 3D shapes. Specifically, we translate the center of the 3D shape to the origin of the spherical coordinate system, and resize the maximum polar distance of the points on the surface of shape to unit length. Because our method is rotation-invariant, the normalization for rotation is unnecessary.

For a 3D shape $\mathcal{S}$, we create 2D projections (depth images) from N_v view points, which are evenly spaced on the unit sphere. The locations of these view points are determined by the two angles θ_{el} and θ_{az} as illustrated in Fig. 4. These projections constitute the *view set* $\mathcal{V}(\mathcal{S}) = \{v_1, v_2, \ldots, v_{N_v}\}$ for the 3D shape $\mathcal{S}$. Some exemplar 3D shapes and their corresponding projections are presented in Fig. 5. For each view v_i $(1 \leq i \leq N_v)$ in $\mathcal{V}(\mathcal{S})$, several interest points are detected with HarrisLaplace [42] detectors, around which a group of SIFT features [3] is extracted. The collection of SIFT features for the 3D shape $\mathcal{S}$ is denoted by $\mathcal{X}(\mathcal{S}) = \{\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_{N_v}\}$, with $\mathcal{X}_i$ representing the SIFT features extracted in v_i.

3.2 Representation based on View Pairs

Given the local SIFT features, many previous works [33], [35], [36] encode them with some coding strategies to get a feature vector for each view. Then a kind of many-to-many matching procedure, such as Clock-Matching [35], is utilized to measure the similarity between two 3D shapes. However, it is known

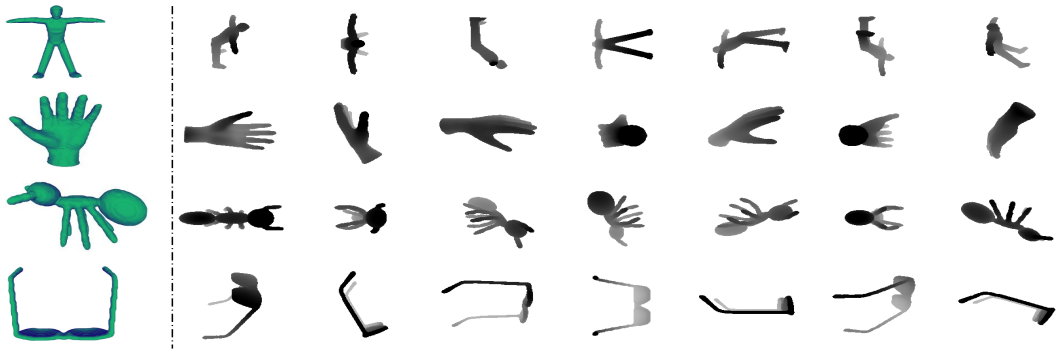


Fig. 5. Some typical 3D models from the McGill dataset (the first column) and their corresponding depth-buffer images. The darker parts in the projections are associated with higher values in depth.

that performing pairwise matching between two sets is time-consuming, which limits the usage of these methods in real-time applications. Furthermore, most of these methods need to compute the principal axis of 3D objects and align it, but the computed principal axis may not be stable in some cases. Some methods, such as VLAT [36], totally ignore the location of the SIFT features, and put all SIFT features from different views in a single bag. While it improves efficiency, the retrieval performance is less accurate due to loss of spatial correlation between views. In contrast, we propose a view-based 3D shape representation that is suitable for large scale 3D shape retrieval and at the same time preserves the spatial arrangement of views.

Given a set of views of a 3D shape, how do humans perceive them? We list some views from Watertight Models Track of SHREC2007 [43] in Fig. 2(a). We observe that these objects are hard to recognize with just a single view, unless additional information is given. But if two views are combined with each other, it is easy for us to distinguish the objects as illustrated in Fig.2(b). For example, the one in the top left corner is a view from the back of a plane, and the one in the top right corner is a view from the bottom of a bearing. As can be seen, two views are much more informative in visual concept than just one view. This example inspires us to use view pairs instead of individual views to represent a 3D shape.

Given a view pair $p_{i,j}$ which is composed of two single views v_i and v_j, the problem is how to design a proper descriptor $f_{i,j} = \mathcal{G}(p_{i,j})$ according to a mapping function $\mathcal{G}$. Note that we have extracted two bags of visual descriptors $\mathcal{X}_i$ and $\mathcal{X}_j$ for v_i and v_j, respectively. Coding-based methods can be utilized to encode these visual descriptors to get the vector representation for $p_{i,j}$. In this paper, Vector of Aggregated Local Descriptors (VLAD) [4] is adopted.

In fact, alterative ways, such as Vector Quantization (VQ) and Fisher Kernel [44], could also be utilized. But compared with other coding strategies, VLAD has some good properties to encode the visual descriptors in the specific situation. First, compared with VQ that usually uses an extremely large codebook, VLAD often needs coarser clusters (typically 32 clusters in our experiments) to accumulate the residual vectors. Hence some operations during the coding procedure, such as NN search, are conducted in a smaller feature space, which reduces the computation time. Second, compared with FK, VLAD is a simplified version that is fast to implement. Third, VLAD aggregates visual descriptors based on the locality criterion in the codebook, and it can preserve the information of codebook, which is beneficial for our second layer coding.

We propose two ways to aggregate the SIFT descriptors, *i.e.* , a joint way and an individual way. The joint way merges two bags $\mathcal{X}_i$ and $\mathcal{X}_j$ into a single bag, and aggregates them jointly, while the individual way considers the two bags separately, and encodes them individually. The two ways are associated with two representations of the view pair, which we call Joint-Pair (J-Pair) and Individual Pair (I-Pair). The definitions of J-Pair and I-Pair are presented next.

3.2.1 J-Pair

Joint-Pair (J-Pair) does not distinguish which view a certain SIFT descriptor belongs to. Two bags of local descriptors from both views v_i and v_j are merged together simply to get its corresponding bag pair

$$\mathcal{Y}_{i,j} = \mathcal{X}_i \bigcup \mathcal{X}_j, \tag{1}$$

where the number of visual descriptors in $\mathcal{Y}_{i,j}$ is equal to the sum of those in $\mathcal{X}_i$ and $\mathcal{X}_j$.

Let $\mathcal{B} = \{b_1, b_2, \ldots, b_K\}$ be a codebook of SIFT descriptors of size K learned off-line with the standard K-means [45] algorthim. For the view pair $p_{i,j}$, the response value of VLAD for the k-th $(1 \leq k \leq K)$ quantization index is a sub-vector $f_{i,j}^k$, defined as the sum of the residual vector, *i.e.* , the difference between the local descriptors and their corresponding visual word:

$$f_{i,j}^k = \sum_{y \in \mathcal{Y}_{i,j}, q(y)=b_k} y - b_k, \tag{2}$$

where function $q(.)$ returns the nearest visual word in the codebook for the input feature. Let $f_{i,j}$ be the concatenation of all the aggregated residuals:

$$f_{i,j} = [f_{i,j}^1 \ f_{i,j}^2 \ \dots \ f_{i,j}^K] \in \mathbb{R}^{d \times K}, \tag{3}$$

where d is the dimension of the local descriptor. The vector representation $f_{i,j}$ for the view pair $p_{i,j}$ is then L_2 normalized. Since the order of the two views is not considered, *i.e.* , $\mathcal{Y}_{i,j} = \mathcal{Y}_{j,i}$, we have

$$\mathcal{G}_J(p_{i,j}) = \mathcal{G}_J(p_{j,i}). \tag{4}$$

The view pair set of the given shape $\mathcal{S}$ is obtained by enumerating the combinations of all view pairs

$$\mathcal{P}_J(\mathcal{S}) = \{p_{i,j} \mid 1 \leq i < j \leq N_v\}, \tag{5}$$

and the corresponding J-Pair feature set is given by

$$\mathcal{F}_J(\mathcal{S}) = \{f_{i,j} | f_{i,j} = \mathcal{G}_J(p_{i,j}), p_{i,j} \in \mathcal{P}(\mathcal{S})\}. \tag{6}$$

Hence the number of view pairs defined by J-Pair is equal to the combinations of two views, which means

$$|\mathcal{P}_J(\mathcal{S})| = |\mathcal{F}_J(\mathcal{S})| = \binom{N_v}{2} = \frac{N_v \times (N_v - 1)}{2}, \tag{7}$$

where the function $|.|$ calculates the set size.

3.2.2 I-Pair

Individual Pair (I-Pair) encodes the visual descriptors from different views individually, i.e., we concatenate the encoded features of two views to represent the view pair.

More specifically, the encoded feature $f_i \in \mathbb{R}^{d \times K}$ for view v_i is computed by encoding the SIFT descriptors in $\mathcal{X}_i$ using VLAD. Then the I-Pair feature $f_{i,j}$ of the view pair $p_{i,j}$ is the concatenation of f_i and f_j

$$f_{i,j} = [f_i, f_j] \in \mathbb{R}^{2 \times d \times K}. \tag{8}$$

In this case, the view pair $p_{i,j}$ is not the same as $p_{j,i}$ under the representation of I-Pair as

$$\mathcal{G}_I(p_{i,j}) \neq \mathcal{G}_I(p_{j,i}), \tag{9}$$

for the I-Pair is sensitive to the order of the two views.

The view pair set described with I-Pair can be obtained through enumerating the permutations of two views

$$\mathcal{P}_I(\mathcal{S}) = \{p_{i,j} \mid 1 \leq i, j \leq N_v, i \neq j\}, \tag{10}$$

and the corresponding I-Pair set $\mathcal{F}_I(\mathcal{S})$. For the shape $\mathcal{S}$, the number of view-pairs defined by I-Pair is equal to the permutations of two views, which means

$$|\mathcal{P}_I(\mathcal{S})| = |\mathcal{F}_I(\mathcal{S})| = \begin{bmatrix} N_v \\ 2 \end{bmatrix} = N_v \times (N_v - 1). \tag{11}$$

For I-Pair description, other methods or some man-made features, such as Zernike moments [46] are also suitable, since it represents a view pair by just concatenating two features of two views that make up the pair. On the contrary, J-Pair can be only adopted with coding-based methods, as it is generated by directly merging two bags of visual descriptors from two views into a single bag. Consequently, J-Pair is invariant to the order of views.

Encoding the visual descriptors to get an encoded feature for a view pair is the **first layer** coding in our Two Layer Coding (TLC) framework. Unlike many previous algorithms that only encode a single view one by one, we consider the collaborative representation of a view pair. In Section 3.3, we present how we handle the second layer coding, applied to both J-Pair and I-Pair, to get the final representation for the 3D shape with a novel angular division that considers the spatial distribution of views and keeps our method invariant to rotation.

3.3 Coding with Angular Division

To simplify the notation, we remove the subscript "J" or "I" in the mapping function $\mathcal{G}$, the view pair set $\mathcal{V}(\mathcal{S})$, and the feature set $\mathcal{F}(\mathcal{S})$ in this section. All definitions next are applied to both J-Pair and I-Pair.

After computing the encoded feature $f_{i,j}$ for each element $p_{i,j}$ in the view pair set $\mathcal{P}(\mathcal{S})$, the upcoming problem is how to organize these encoded features to get a compact representation for the whole 3D shape. A straightforward solution is to encode these features directly again via some coding algorithms, such as Vector Quantization.

By using the codebook constructed by the standard K-means clustering, Vector Quantization deems that two view pairs generate a successful match, when they fall into the same cluster. Therefore, they are assigned to a same visual word in the coding procedure, and contribute equally to the final representation of the 3D shape.

However, when Vector Quantization is applied directly, the spatial relation of views is not considered. Note that all views lie on the surface of the unit sphere. Let $\vec{u_i}$ denote the unit vector from the centroid to the view point where v_i is rendered. We define

$$\beta_{i,j} = \arccos < \vec{u_i}, \vec{u_j} > \tag{12}$$

as the angle between v_i and v_j, which make up a view pair $p_{i,j}$. The illustration of the unit vectors , the angle β, and the view points are presented in Fig 4. The angle is an important attribute of a view pair, and we call it the "eigen-angle" of a view pair. We assume that two shapes are similar when the view pairs from each shape with similar eigen-angles are also similar. Although it is possible that some shapes from different categories may have similar views, the likelihood of having similar view pairs with a similar eigen-angle is extremely small, as illustrated in Fig. 3.

We uniformly divide $[0, \pi]$, the range of eigen-angles, into $\mathcal{L}$ bins $\{U_1, U_2, \dots, U_{\mathcal{L}}\}$, where U_l is de-

fined as

$$U_l = \{\beta | \frac{l-1}{\mathcal{L}}\pi \leq \beta \leq \frac{l}{\mathcal{L}}\pi\}. \quad (13)$$

A binary indicator vector $I_{i,j} = \{I^1_{i,j}, I^2_{i,j}, \ldots, I^{\mathcal{L}}_{i,j}\} \in \mathbb{R}^{\mathcal{L}}$ defined as

$$I^l_{i,j} = \mathcal{H}(p_{i,j}) = \begin{cases} 1 & if \quad \beta_{i,j} \in U_l \\ 0 & otherwise \end{cases} \quad (14)$$

is used to determine which eigen-angle bin the view pair $p_{i,j}$ belongs to. Each view pair $p_{i,j}$ is represented by a tuple $(f_{i,j}, I_{i,j})$, where $I_{i,j}$ can be interpreted as a spatial description of feature $f_{i,j}$. The matching between two view pairs can be valid, only if they look similar in the feature space and their eigen-angles lie in the same bin.

The eigen-angle $\beta_{i,j}$ reveals the spatial relationship between v_i and v_j, and such a relative spatial description is rotation-invariant, *i.e.* , no matter how we rotate the 3D shape $\mathcal{S}$, the eigen-angle between two views remains unchanged. Many previous methods, like [35], [7], achieve the rotation invariance by applying PCA-based technique and aligning the 3D shape to the principal axis. However, different PCA-based techniques lead to different canonical coordinate frames of a 3D shape. Moreover, it is usually unstable to define the principal axis for many shapes like a ball.

3.3.1 Codebook Learning

We apply Classified Vector Quantization (CVQ) [47] that utilizes classified K-means for codebook learning, since we not only consider the distribution of view pairs in the feature space, but also attach importance to the distribution of the view pairs in the angular space. While CVQ in [47] is applied to encoding images, we employ it to encode the view pairs of 3D shapes. In our approach, the discriminator function is based on the eigen-angle division. First the whole training set is divided into $\mathcal{L}$ subsets according to the eigen-angle bins. Then, a standard K-means is used to construct the visual vocabulary for each subset. In the phase of coding, the features with eigen-angles belonging to different bins are never assigned to the same visual word, since they are encoded with different sub-codebooks.

Given a set of view pairs $\mathcal{P}(\mathcal{S})$ of a 3D shape $\mathcal{S} \in \mathcal{S}'$, where $\mathcal{S}'$ is the shape database, we define the indicator vector set as

$$I(\mathcal{S}) = \{I_{i,j} | I_{i,j} = \mathcal{H}(p_{i,j}), p_{i,j} \in \mathcal{P}(\mathcal{S})\}. \quad (15)$$

Then, the collection of the features

$$\mathcal{F}(\mathcal{S}') = \{f_{i,j} | f_{i,j} \in \mathcal{F}(\mathcal{S}), \mathcal{S} \in \mathcal{S}'\} \quad (16)$$

can be computed. Based on the division of the eigen-angle, the l-th$(1 \leq l \leq \mathcal{L})$ subset of $\mathcal{F}_l(\mathcal{S}')$ is represented by

$$\mathcal{F}^l(\mathcal{S}') = \{f_{i,j} | f_{i,j} \in \mathcal{F}(\mathcal{S}'), I^l_{i,j} = 1\}. \quad (17)$$

Standard K-means is applied to divide $\mathcal{F}^l(\mathcal{S}')$ into M informative regions, and the codebook $\mathcal{C}^l = \{c^l_1, c^l_2, \ldots, c^l_M\}$ is learned for $\mathcal{F}^l(\mathcal{S}')$. The final codebook is obtained as $\mathcal{C} = \bigcup_{l=1}^{\mathcal{L}} \mathcal{C}^l$.

3.3.2 Coding

With the sub-codebook learned on each subset, we need to encode the feature set $\mathcal{F}(\mathcal{S})$ to give a final vector representation for a shape $\mathcal{S}$. Given a 3D shape $\mathcal{S}$, the l-th subset of features is given by

$$\mathcal{F}^l(\mathcal{S}) = \{f_{i,j} | f_{i,j} \in \mathcal{F}(\mathcal{S}), I^l_{i,j} = 1\}. \quad (18)$$

Let

$$z_{i,j} = \mathcal{T}(f_{i,j}) = [z^1_{i,j}, z^2_{i,j}, \ldots, z^M_{i,j}] \in \mathbb{R}^M \quad (19)$$

be the encoded feature for $f_{i,j}$, and $z^m_{i,j}$ be the response value of $f_{i,j}$ with respect to the m-th visual word in the sub-codebook C^l. For each $f_{i,j} \in \mathcal{F}^l(\mathcal{S})$, VQ assigns it to the nearest visual word in the corresponding codebook $\mathcal{C}^l$ so that $z^m_{i,j}$ satisfies

$$z^m_{i,j} = \begin{cases} 1 & if \quad q(f_{i,j}) = c^l_m \\ 0 & otherwise \end{cases} \quad (20)$$

However, as analyzed in [48], vector quantization offers a coarse estimation to the real distance between two features, *i.e.* , zero if assigned to the same visual word, and infinite otherwise. This may lead to severe quantization errors, especially for features located at the boundary of several visual words. In order to alleviate the problem, we use a soft-assignment strategy [48] to assign a descriptor to more than one visual words, defined as

$$z^m_{i,j} = \frac{\exp(-\alpha \cdot \hat{d}(f_{i,j}, c^l_m))}{\sum_{k=1}^{M} \exp(-\alpha \cdot \hat{d}(f_{i,j}, c^l_k))}, \quad (21)$$

where

$$\hat{d}(f_{i,j}, c^l_m)) = \begin{cases} d(f_{i,j}, c^l_m)) & if \quad c^l_m \in q_k(f_{i,j}) \\ 0 & otherwise \end{cases} \quad (22)$$

and α is the smoothing factor that controls the softness of the assignment. The function $q_k(.)$ yields the set of visual words containing the first k nearest neighbors of the input feature.

The sum-pooling method defined as

$$\mathcal{W}^l = \sum \{z_{i,j} | z_{i,j} = \mathcal{T}(f_{i,j}), f_{i,j} \in \mathcal{F}^l(\mathcal{S})\} \in \mathbb{R}^M \quad (23)$$

is used to count the number of occurrences of corresponding visual word.

The final representation of the 3D shape $\mathcal{S}$ is the concatenation of all $\mathcal{W}^l (1 \leq l \leq \mathcal{L})$, and it can be determined as

$$\mathcal{W}(\mathcal{S}) = [\mathcal{W}^1 \ \mathcal{W}^2 \ \ldots \ \mathcal{W}^{\mathcal{L}}] \in \mathbb{R}^{M \times \mathcal{L}}, \quad (24)$$

which is subsequently L_2 normalized.

Let $\mathcal{Q} = [\mathcal{Q}^1, \mathcal{Q}^2, \ldots, \mathcal{Q}^{\mathcal{L}}]$ and $\mathcal{D} = [\mathcal{D}^1, \mathcal{D}^2, \ldots, \mathcal{D}^{\mathcal{L}}]$ represent two 3D shapes, where $\mathcal{Q}^i = [q^i_1, q^i_2, \ldots, q^i_M]$

denotes the feature vector of ith eigen-angle bin of $\mathcal{Q}$, and similarly $\mathcal{D}^i = [d_1^i, d_2^i, \ldots, d_M^i]$. We use the L_2 distance between $\mathcal{Q}$ and $\mathcal{D}$ computed as

$$d(\mathcal{Q}, \mathcal{D}) = \sum_i^{\mathcal{L}} d(\mathcal{Q}^i, \mathcal{D}^i)_2 = \sum_i^{\mathcal{L}} \sum_j^{M} |q_j^i - d_j^i|_2 \quad (25)$$

to measure the dissimilarity of the two shapes, where $|.|_2$ denotes the L_2 distance. The distance function d preserves the spatial information of eigen-angle, for it only compares the feature vectors belonging to the same eigen-angle bins, which makes the function d more discriminative as demonstrated by the experimental results presented in the next section.

4 EXPERIMENTS

In this section, we evaluate the proposed dissimilarity function on five 3D shape datasets and perform a comprehensive comparison to state-of-the-art algorithms. We also study the influence of parameters on the retrieval performance in Section 4.4.1. Comparisons with three baseline methods are presented in Section 4.4.2. The robustness against noise is discussed in Section 4.4.3. The proposed TLC framework is extended to 3D feature in Section 4.5. The average matching time is discussed in Section 4.6.

4.1 Datasets and Evaluation Tools

The datasets used here are the Princeton Shape Benchmark test dataset (PSB) [28], the Engineering Shape Benchmark (ESB) [49], the McGill dataset [23], the Watertight Models track of SHape REtrieval Contest 2007 dataset (WM-SHREC07) [43], and the SHape REtrieval Contest 2014 Large Scale Comprehensive Track Benchmark (SHREC14LSGTB) [50].

Among the five datasets, PSB dataset is the first widely-used generic shape benchmark, and ESB dataset is more relevant to the mechanical engineering domain for it consists of 3D CAD models. Different from PSB dataset that only contains rigid shapes, McGill dataset contains non-rigid models. WM-SHREC07 and SHREC14LSGTB are two shape datasets for competition held each year, and SHREC14LSGTB is the latest and the biggest one, but shares some common models with PSB, ESB and McGill. SHREC14LSGTB is also the most challenging dataset so far, in which the models exhibit more diversity than those in any previous datasets. The details of these datasets, including the total number of models $\#Model$, the total number of categories $\#Category$, the average number of models per category $\#Aver$ and the maximum number of models per category $\#Max$, are presented in Table 1.

To quantify the performance, we employ the following evaluation tools:

TABLE 1
The details of the five datasets used in this paper.

Dataset	#Model	#Category	#Aver	#Max
PSB	907	92	10	50
ESB	867	45	19	58
McGill	255	10	26	31
WM-SHREC07	400	20	20	20
SHREC14LSGTB	8987	171	53	632

- Nearest Neighbor (NN): the percentage of the closest matches that belongs to the same class as the query.
- First Tier (FT) and Second Tier (ST): the recall for the top $(k-1)$ and $2(k-1)$ matches in the ranked list respectively, where k is the number of shapes in the category that query belongs to.
- Discounted Cumulative Gain (DCG): a statistic that attaches more importance to the correct results near the front of the ranked list than the correct results at the end of the ranked list, under the assumption that a user is less likely to consider elements near the end of the list.

All the evaluation scores range from 0% to 100%, and a higher score indicates a better performance. Refer to [28] for more details about the definitions of NN, FT, ST and DCG.

We also give Precision-Recall curves of different methods to visualize the performance of Two Layer Coding against other methods. "Recall" (the horizontal axis) is defined as the ratio of correct retrieved models to the total number of models in the category, while "Precision" (the vertical axis) is the ratio of correct retrieved models to the total number of retrieved models. The more shifted up a P-R curve is, the better of the retrieval performance for a given algorithm.

4.2 Implementation Details

If not stated otherwise, we adopt the following setup for all experiments.

Projection and feature extraction: After pose normalization, each 3D shape $\mathcal{S}$ is projected into $N_{\mathcal{V}} = 64$ views. Each view is a depth image of size 200×200. We extract about 200 SIFT descriptors around the interest points per view. We use the the variant of RootSIFT [51] in all experiments.

Codebook learning: We randomly choose 1M SIFT features to learn the codebook $\mathcal{B} = \{b_1, b_2, \ldots, b_K\}$ in the first layer using the standard K-means. The codebook size K is set to 32. A randomly selected set of VALD features (500K) is used to construct the codebook $\mathcal{C}^l = \{c_1^l, c_2^l, \ldots, c_M^l\}$ $(1 \leq l \leq \mathcal{L})$ in the second layer, with the bin number $\mathcal{L}$ set to 4, and the codebook size M for each bin set to 500. Considering that SHREC14LSGTB is an extremely large dataset, K is set to 160, and M is set to 1000 for this dataset.

Coding: We use VLAD in the first layer coding, and VQ in the second layer coding. The number of visual

words in the soft-assignment strategy of VQ is set to 3, and the smoothing factor α is set to 1.

Metric: Euclidean distance, also known as L_2 metric, is utilized to compute the distance between two shapes.

Additional technique: The speed of retrieval is crucial in industrial applications. Considering the sparsity of our learned representations for 3D shapes, inverted file, an index data structure widely used in the document and image retrieval, can be used to boost the retrieval speed significantly.

Experimental platform: Experiments are carried out on a desktop machine with an Intel(R) Core(TM) i5-2320K CPU (3.00GHz) and 12GB memory.

4.3 Comparative Evaluation

We compare the performance of the proposed method to several state-of-the-art algorithms listed below:

- Covariance Method [16]: a newly developed descriptor using covariance matrices of features instead of the features themselves. Two ways (one is matching method and the other is an extension to BoW) are used to compute the similarity between 3D models based on the Riemannian metric.
- PANORAMA [7]: a novel 3D shape representation that uses a set of panoramic views of the 3D model well aligned via PCA technique. The views are described with the combination of the 2D Discrete Fourier Transform and the 2D Discrete Wavelet Transform.
- 2D/3D Hybrid [41]: a hybrid descriptor composed of 2D features based on depth buffers and 3D features based on spherical harmonics. The feature compactness is achieved via scalar feature quantization, and further compressed by Huffman coding.
- Light Field Descriptor (LFD) [31]: a classical descriptor that uses the orthogonal projections of the 3D object. These projections are encoded both by Zernike moments and Fourier descriptors as features for later retrieval.
- DESIRE [32]: a composite 3D shape feature vector which is combined of depth buffer images, silhouettes, and ray-extents of a polygonal mesh.
- SH-GEDT [52]: a rotation invariant representation of the Gaussian Euclidean Distance Transform descriptor.

As shown in Table 2, it is evident that TLC exhibits encouraging discriminative power in shape matching, and outperforms other methods significantly. Our TLC consistently achieves state-of-the-art performance for all four evaluation metrics (NN, FT, ST and DCG) in PSB dataset, ESB dataset, McGill dataset, and WM-SHREC07 dataset. Another phenomenon is that the results achieved by I-Pair are overall sightly better

TABLE 2
The performance of different algorithms on five standard datasets.

(a) PSB

Methods	NN	FT	ST	DCG
SH-GEDT [52]	0.556	0.309	0.411	0.584
LFD [31]	0.657	0.380	0.487	0.643
DESIRE [32]	0.665	0.403	0.512	0.663
2D/3D Hybrid [41]	0.742	0.473	0.606	-
PANORAMA [7]	0.753	0.479	0.603	-
PANORAMA + LRF [7]	0.752	0.531	0.659	-
TLC + J-Pair	**0.776**	0.555	0.700	**0.775**
TLC + I-Pair	0.763	**0.562**	**0.705**	0.773

(b) ESB

Methods	NN	FT	ST	DCG
SH-GEDT [52]	0.803	0.401	0.536	-
LFD [31]	0.820	0.404	0.539	-
DESIRE [32]	0.823	0.417	0.550	-
2D/3D Hybrid [41]	0.829	0.465	0.605	-
PANORAMA [7]	0.865	0.494	0.641	-
PANORAMA + LRF [7]	0.870	0.499	0.658	-
TLC + J-Pair	0.874	**0.570**	0.718	**0.821**
TLC + I-Pair	**0.882**	0.568	**0.720**	**0.821**

(c) McGill

Methods	NN	FT	ST	DCG
2D/3D Hybrid [41]	0.925	0.557	0.698	0.850
Hybrid BoW [40]	0.957	0.635	0.790	0.886
PCA-based VLAT [36]	0.969	0.658	0.781	0.894
Covariance Method [16]	0.977	0.732	0.818	0.937
Graph-based [53]	0.976	0.741	0.911	0.933
TLC + J-Pair	**0.988**	0.795	0.921	**0.956**
TLC + I-Pair	0.980	**0.807**	**0.933**	**0.956**

(d) WM-SHREC07

Methods	NN	FT	ST	DCG
SH-GEDT [52]	0.870	0.447	0.585	-
LFD [31]	0.923	0.526	0.662	-
Tabia *et al.* [15]	0.853	0.527	0.639	0.719
DESIRE [32]	0.917	0.535	0.673	-
Hybrid BoW [40]	0.918	0.600	0.740	0.847
Covariance Method [16]	0.930	0.623	0.737	0.864
2D/3D Hybrid [41]	0.955	0.642	0.773	-
PANORAMA [7]	0.957	0.673	0.784	-
PANORAMA + LRF [7]	0.957	0.743	0.839	-
TLC + J-Pair	**0.993**	0.815	0.917	0.952
TLC + I-Pair	0.988	**0.831**	**0.935**	**0.957**

(e) SHREC14LSGTB

Methods	NN	FT	ST	DCG	Dim
VM-1SIFT [54]	0.732	0.282	0.380	0.688	-
DBNAA-DERE [55]	0.817	0.355	0.464	0.731	-
BF-DSIFT [54]	0.824	0.378	0.492	0.756	-
ZFDR [56]	0.838	0.386	0.501	0.757	-
KVLAD [50]	0.605	0.413	0.546	0.746	32k
DBSVC [50]	0.868	0.438	0.563	0.790	270k
TLC + J-Pair	0.871	0.447	0.577	0.799	**4k**
TLC + I-Pair	**0.879**	**0.456**	**0.585**	**0.804**	**4k**

than J-Pair. This is reasonable, as the encoding procedure of J-Pair puts the local descriptors from two views into a same visual word, but I-Pair never.

For the comparison on SHREC14LSGTB dataset, we collected the leading results in the Shrec competition 2014 from the survey paper [50]. As shown in Table 2 (e), our method also achieves the best performance. Notice that the competitive methods including BF-DSIFT, KVLAD and DBSVC are coding-based methods, as is TLC. The dimension of the final features obtained by KVLAD, DBSVC is 32k and 270k respectively. Such high dimensional features are not a proper choice for large scale 3D shape retrieval, due to large memory consumption and low retrieval efficiency. By contrast, our features are much more compact with advanced discriminatory power, which is more suitable for 3D shape retrieval in large scale dataset. The numerical comparison of average matching time will be presented in Section 4.6.

It has been proven that utilizing contextual information in the data manifold through some learning methods, such as Local Relevance Feedback (LRF) [7], Diffusion Process [57], [58], *etc.* , can improve the retrieval performance. However, our method, without considering the data manifold, even surpasses PANORAMA with LRF markedly, which confirms its superior discriminative ability. It is expected that TLC can achieve a better performance if contextual information is taken into consideration.

In addition, we also plot the precision-recall curves for the five datasets to visualize the performance of different algorithms in Fig. 6. Consistent with the previous analysis, our algorithm also performs best among these algorithms.

4.4 Discussion

4.4.1 To what extent does TLC improve the baseline

In order to show the superiority of two layer coding, we implement three baseline methods which focus on different aspects of TLC. The default settings described in Section 4.2 are used for the implementation of TLC.

1) One layer coding: Encoding the visual local features from all views in just one layer is a straightforward way, which has been proven feasible in [36]. However, it ignores the spatial arrangement of different views, and considers all the features extracted from different views equally. We set the codebook size as 32 in order to keep the same dimension of output features with TLC for fair comparison.
2) Single view coding: To demonstrate that view pair representation is more stable and discriminative than representation using single view, we directly encode all the VLAD features of individual views into a histogram using vector quantization. The codebook size is set to 2,000 to keep the final feature of the same length with TLC.
3) TLC without angular division: This baseline method is used to prove the effect of angular division proposed in TLC. In this setting, angular division is not performed in the second layer coding, and all the view pairs are directly encoded by vector quantization. Two results of the baseline method are reported, associated with I-Pair and J-Pair respectively.

The comparison is conducted on PSB dataset and WM-SHREC07 competition dataset, and the results are presented in Table 3. It can be easily seen that TLC, with either J-Pair or I-Pair, performs much better than direct one layer coding for all four metrics, which confirms that the way we use the spatial information can generate much more discriminative and robust features for shape matching. The inferior performance of single view coding indicates that it is very beneficial to utilize the stable representation of view pairs. The performance of TLC without angular division is also presented. As it suggests, exploiting the spatial arrangement of views can improve the performance further.

4.4.2 Parameter Analysis

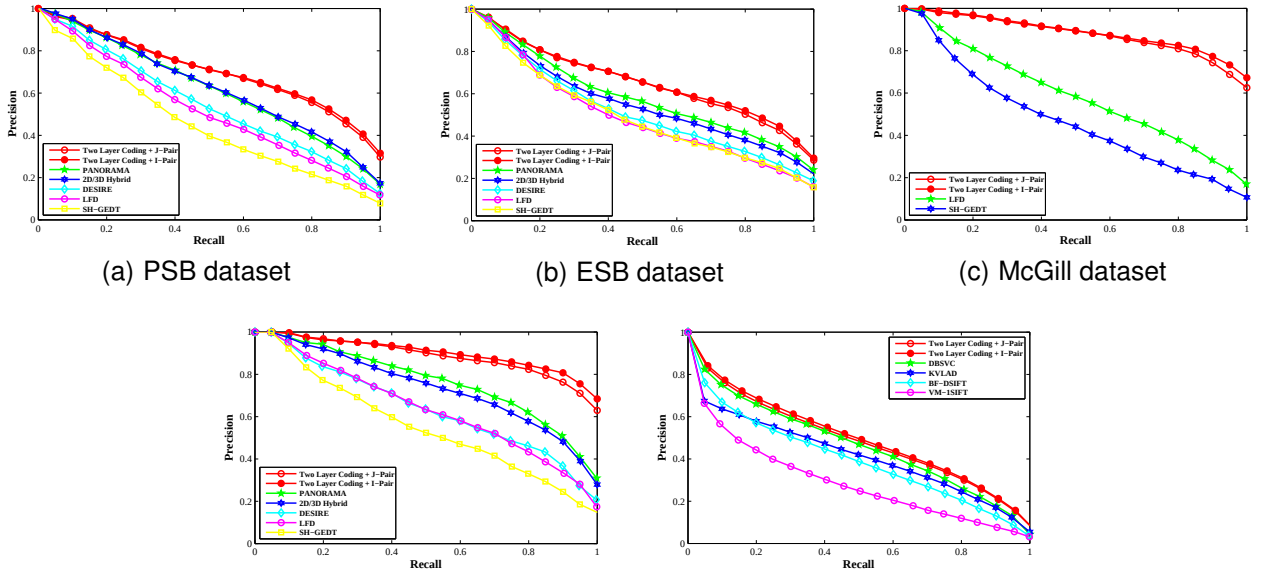
In the section, we discuss the impact of the parameters on the retrieval performance. The most important parameters in our method are the size K of codebook learned for the SIFT features in the first layer coding, and the size M of sub-codebooks learned for the feature pairs (J-Pair or I-Pair) in the second layer coding.

It has been proven experimentally in image retrieval and classification that the performance improves with the codebook size increasing, but gets saturated when the codebook size reaches a critical point. The phenomenon is also known as the so-called overfitting effect, which results in the plateau of performance curves.

The performance of TLC with J-Pair using codebooks of different sizes in PSB dataset is reported in Fig. 7. We first fix the size of sub-codebook in the second layer to 500, and plot the performance under different sizes of codebook used in the first layer in Fig. 7(a). There is no increase in retrieval performance when the codebook size arrives at 64. We also evaluate the influence of the sub-codebook size used in the second layer coding in Fig. 7(b), when the codebook size in the first layer is constantly set to 32. It can be found that the saturation point of the sub-codebook size in the second layer is 750.

4.4.3 Robustness Against Noise of View Sampling

In real cases, the view points are not necessarily located evenly around the 3D model. In order to evaluate the robustness of TLC against view point changes, we



(a) PSB dataset (b) ESB dataset (c) McGill dataset

(d) WM-SHREC07 dataset (e) SHREC14LSGTB dataset

Fig. 6. The Precision-Recall curves of TLC and other comparative approaches on five datasets.

TABLE 3
The comparison with baseline methods.

	PSB dataset				WM-SHREC07 competition			
Method	NN	FT	ST	DCG	NN	FT	ST	DCG
One layer Coding	0.735	0.508	0.638	0.749	0.935	0.692	0.822	0.900
Single view coding	0.703	0.468	0.586	0.714	0.962	0.691	0.773	0.893
TLC+J-Pair (No angular division)	0.770	0.549	0.691	0.771	0.975	0.740	0.844	0.918
TLC+I-Pair (No angular division)	0.763	0.546	0.688	0.767	0.967	0.790	0.887	0.937
TLC+J-Pair	**0.776**	0.555	0.700	**0.775**	**0.993**	0.815	0.917	0.952
TLC+I-Pair	0.763	**0.562**	**0.705**	0.773	0.988	**0.831**	**0.935**	**0.957**

add a random noise during view sampling procedure. The noise follows a Gaussian distribution with zero mean and standard deviation σ.

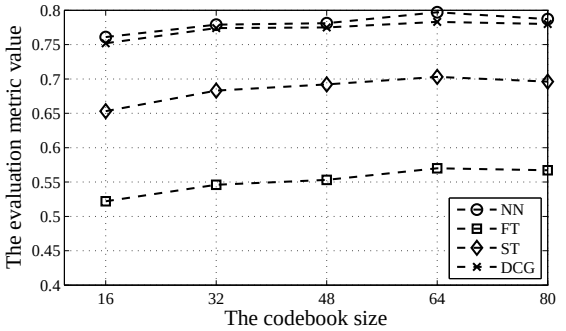
Fig. 8 depicts the retrieval performances of TLC with I-Pair with regard to the noise ratio σ. As Shown in Fig. 8, TLC is stable to view point changes, and adding extra noise does not impair the retrieval performances too much. It also demonstrates that view pair provides much more stable representation than the individual view for coding approaches.

4.5 Extension to 3D Feature

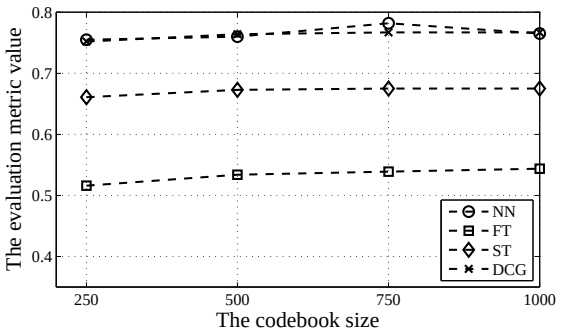
Though the proposed method is based on 2D projections, TLC can be easily extended to encode 3D shape features on surface, since feature sampling and coding strategies with view pair are general. We encode 3D features with TLC as follows: 1) extract the 3D features on the vertices of 3D models; 2) For a certain view (determined by θ_{el} and θ_{az}), the 3D feature of the visible vertices are collected and encoded using vector quantization like Shape Google [17] into a histogram to represent that view. With the above simple operations, the proposed TLC can be applied to encode 3D features.

To demonstrate the effectiveness of TLC for 3D features, we compare it to two recent popular model-based approaches, eg., Shape Google [17] and Intrinsic Spatial Pyramid Matching [59], which are designed for encoding 3D shape signatures, and are a good fit for non-rigid shapes. For fair comparison, we adopt Heat Kernel Signature (HKS) [26] as the input feature for TLC, consistent to [17], [59]. Notice that though HKS and Scale-invariant Heat Kernel Signature (SIHKS) [27] are popular descriptors, they require 3D models containing manifold structure as input, which is not fulfilled in the five datasets we used. As a result, we use SHREC 2011 non-rigid dataset [60] following Intrinsic Spatial Pyramid Matching [59] for the comparison. Consistent with the parameter setting of HKS in [59], the first 150 eigenvalues and eigenvectors are used; the diffusion time is formulated as $t = t_0\alpha^{\tau}$ where t_0 and α are set to 0.01 and 4 respectively; τ ranges from 0 to 5 with a step size 0.25; the codebook size for HKS is set to 32.

In Table 4, we compare the performance of HKS-based TLC with Shape Google [17] and Intrinsic Spatial Pyramid Matching [59]. We can observe that TLC is also suitable for 3D features, and it outperforms the ISPM by 0.17, and ShapeGoogle by 0.26 in DCG.



(a)



(b)

Fig. 7. The influence of the codebook size in each layer. (a) The influence of the codebook size in the first layer for the four evaluation metrics. The size of each sub-codebook in the second layer is set to 500. (b) The influence of codebook size in the second layer for the four evaluation metrics. The size of codebook in the first layer is set to 32.

TABLE 4
The comparison on SHREC 2011 non-rigid dataset.

Method	NN	FT	ST	DCG
Shape Google [17]	0.982	0.637	0.732	0.881
ISPM [59]	-	-	-	0.890
Single view coding	0.987	0.652	0.719	0.881
TLC+J-Pair (HKS)	0.985	0.703	0.797	0.907
TLC+I-Pair (HKS)	0.983	0.694	0.791	0.901
TLC+J-Pair (SIFT)	0.982	0.864	**0.941**	**0.964**
TLC+I-Pair (SIFT)	**0.990**	**0.865**	0.933	0.963

The single view coding with HKS as a baseline is also reported in Table 4, which performs much worse than TLC. This again demonstrates that view pair representation is more stable than single view. We also reported the results of TLC that deals with SIFT on the 2D projections, and they yield the best performance in Table 4.

Note that TLC is designed for generic 3D shape retrieval, and it seems not proper for non-rigid deformation. However, as demonstrated by the competitive results on McGill dataset and SHREC 2011 non-rigid dataset, TLC is also robust to such pose variation for two reasons. First, the SIFT descriptor with interest point detectors is invariant to pose variations. The Harris detector can more or less capture the local parts of a non-rigid object even though its pose changes. Second, TLC is partly robust to pose variations due to the inborn characteristic of histogram feature. Vector quantization, especially the soft assignment version, can tolerate some minor changes of the encoded information from the view pairs.

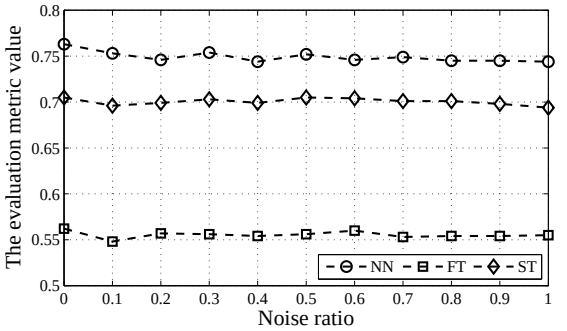


Fig. 8. The retrieval performance of TLC on PSB dataset is reported when adding Gaussian noise with standard deviation σ increasing from 0 to 1 during view sampling procedure.

4.6 The Average Matching Time

As the proposed method is mainly designed for 3D shape retrieval in large scale, we show the time cost of TLC and compare it with several efficient methods on PSB dataset in Table 5. The average pairwise matching time is defined as the time of retrieving the whole database dividing by the maximum possible comparison number. Many existing view-based matching methods are mainly designed for establishing the accurate correspondence of multiple views. In contrast, TLC generates a single vector representation for a 3D shape, which is a sparse vector with many zero elements. Consequently, inverted file [61] can de adopted to significantly speed up the matching process. As shown in Table 5, TLC only takes microseconds to finish a retrieval task for one query. On the largest SHREC14LSGTB dataset, the average matching time of TLC is about $2.10us$, while the time decreases to $0.22us$ if inverted file is utilized. Moreover, it can be expected that the speed of retrieval can be further boosted with the usage of more advanced indexing techniques, such as Product Quantization [62], KD-tree [63].

5 Conclusion

In this paper, we propose a Two Layer Coding (TLC) framework for effectively retrieving 3D shapes with high time efficiency. The proposed method does not need PCA for orientation alignment due to its

TABLE 5
The average pairwise matching time of different algorithms on PSB dataset.

Algorithms	Time(us)	Algorithms	Time(us)
LFD	1300	2D/3D Hybrid	170
Panoramic Views	230	TLC	**1.09**

rotation-invariant property. For the first layer, two representations of view pairs are proposed to encode the local features around the interest points in the depth views. For the second layer, view-pair representations can be considered as a set of local features collected from a given 3D shape, which are encoded into a final feature vector for shape comparison. The angular division of view pairs facilitates the second layer coding that preserves the relative spatial relationship between views.

Instead of establishing the view correspondence explicitly, the final signatures of TLC are merely sparse vectors that can be directly used for shape matching using L_2 distance, and the sparsity property is a particularly good fit for large scale 3D shape retrieval with inverted file. Our method not only has advantage in the time efficiency, but also consistently achieves state-of-the-arts retrieval performance on several standard datasets. In addition, our framework is not limited to encoding features from 2D projections, but also works well on encoding 3D surface descriptors.

In the future, we will study the feature fusion of complementary descriptors and more advanced coding methods [36], [64] in our two layer coding framework. Also, more delicate methods can be explored for modeling the spatial relation between views.

ACKNOWLEDGMENT

The authors would like to thank Iasonas Kokkinos and Chunyuan Li for providing us their softwares for the performance comparison on SHREC2011 dataset. This work was primarily supported by National Natural Science Foundation of China (No. 61222308), and in part supported by Program for New Century Excellent Talents in University (No.NCET-12-0217).

REFERENCES

[1] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints," in *ECCV*, 2004.
[2] F.-F. Li and P. Perona, "A bayesian hierarchical model for learning natural scene categories," in *CVPR*, 2005.
[3] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *IJCV*, vol. 60, no. 2, pp. 91–110, 2004.
[4] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," in *CVPR*, 2010.
[5] M. Körtgen, G. J. Park, M. Novotni, and R. Klein, "3d shape matching with 3d shape contexts," in *The 7th Central European Seminar on Computer Graphics*, 2003.
[6] X. Bai, C. Rao, and X. Wang, "Shape vocabulary: A robust and efficient shape representation for shape matching," *TIP*, vol. 23, pp. 3935–3949.
[7] P. Papadakis, I. Pratikakis, T. Theoharis, and S. J. Perantonis, "Panorama: A 3d shape descriptor based on panoramic views for unsupervised 3d object retrieval," *IJCV*, vol. 89, no. 2-3, pp. 177–192, 2010.
[8] F. Tombari, S. Salti, and L. di Stefano, "Performance evaluation of 3d keypoint detectors." *IJCV*, vol. 102, no. 1-3, pp. 198–220, 2013.
[9] M. Ankerst, G. Kastenmüller, H.-P. Kriegel, and T. Seidl, "Nearest neighbor classification in 3d protein databases," in *ISMB*, 1999.
[10] R. Osada, T. A. Funkhouser, B. Chazelle, and D. P. Dobkin, "Matching 3d models with shape distributions," in *SMI*, 2001.
[11] R. Ohbuchi, T. Minamitani, and T. Takei, "Shape-similarity search of 3d models by using enhanced shape functions," *IJCAT*, vol. 23, no. 2-4, pp. 70–85, 2005.
[12] A. E. Johnson and M. Hebert, "Using spin images for efficient object recognition in cluttered 3d scenes," *TPAMI*, vol. 21, no. 5, pp. 433–449, 1999.
[13] T. Zaharia and F. J. Preteux, "3d-shape-based retrieval within the mpeg-7 framework," in *Photonics West 2001-Electronic Imaging*, 2001.
[14] P. Papadakis, I. Pratikakis, S. J. Perantonis, and T. Theoharis, "Efficient 3d shape matching and retrieval using a concrete radialized spherical projection representation," *Pattern Recognition*, vol. 40, no. 9, pp. 2437–2452, 2007.
[15] H. Tabia, M. Daoudi, J.-P. Vandeborre, and O. Colot, "A new 3d-matching method of nonrigid and partially similar models using curve analysis," *TPAMI*, vol. 33, no. 4, pp. 852–858, 2011.
[16] H. Tabia, H. Laga, D. Picard, and P.-H. Gosselin, "Covariance descriptors for 3d shape matching and retrieval," in *CVPR*, 2014.
[17] A. M. Bronstein, M. M. Bronstein, L. J. Guibas, and M. Ovsjanikov, "Shape google: Geometric words and expressions for invariant shape retrieval," *TOG*, vol. 30, no. 1, p. 1, 2011.
[18] R. Gal and D. Cohen-Or, "Salient geometric features for partial shape matching and similarity," *TOG*, vol. 25, no. 1, pp. 130–150, 2006.
[19] Y. Wang, M. Gong, T. Wang, D. Cohen-Or, H. Zhang, and B. Chen, "Projective analysis for 3d shape segmentation," *TOG*, vol. 32, no. 6, p. 192, 2013.
[20] M. Hilaga, Y. Shinagawa, T. Komura, and T. L. Kunii, "Topology matching for fully automatic similarity estimation of 3d shapes," in *SIGGRAPH*, 2001.
[21] H. Sundar, D. Silver, N. Gagvani, and S. J. Dickinson, "Skeleton based shape matching and retrieval," in *SMI*, 2003.
[22] N. D. Cornea, M. F. Demirci, D. Silver, A. Shokoufandeh, S. J. Dickinson, and P. B. Kantor, "3d object retrieval using many-to-many matching of curve skeletons," in *SMI*, 2005.
[23] K. Siddiqi, J. Zhang, D. Macrini, A. Shokoufandeh, S. Bouix, and S. J. Dickinson, "Retrieving articulated 3-d models using medial surfaces," *Mach. Vis. Appl.*, vol. 19, no. 4, pp. 261–275, 2008.
[24] C. Li and A. B. Hamza, "Skeleton path based approach for nonrigid 3d shape analysis and retrieval," in *Combinatorial Image Analysis - 14th International Workshop, IWCIA 2011, Madrid, Spain, May 23-25, 2011. Proceedings*, 2011.
[25] I. Kokkinos, M. M. Bronstein, R. Litman, and A. M. Bronstein, "Intrinsic shape context descriptors for deformable shapes," in *CVPR*, 2012.
[26] J. Sun, M. Ovsjanikov, and L. Guibas, "A concise and provably informative multi-scale signature based on heat diffusion," vol. 28, no. 5, pp. 1383–1392, 2009.
[27] M. M. Bronstein and I. Kokkinos, "Scale-invariant heat kernel signatures for non-rigid shape recognition," in *CVPR*, 2010.
[28] P. Shilane, P. Min, M. M. Kazhdan, and T. A. Funkhouser, "The princeton shape benchmark," in *SMI*, 2004.
[29] B. Bustos, D. A. Keim, D. Saupe, T. Schreck, and D. V. Vranic, "Feature-based similarity search in 3d object databases," *ACM Comput. Surv.*, vol. 37, no. 4, pp. 345–387, 2005.
[30] R. Jonker and A. Volgenant, "A shortest augmenting path algorithm for dense and sparse linear assignment problems," *Computing*, vol. 38, no. 4, pp. 325–340, 1987.
[31] D. Y. Chen, X. P. Tian, Y. T. Shen, and M. Ouhyoung, "On visual similarity based 3d model retrieval," *Comput. Graph. Forum*, vol. 22, no. 3, pp. 223–232, 2003.
[32] D. V. Vranic, "Desire: a composite 3d-shape descriptor," in *ICME*, 2005.

[33] T. Furuya and R. Ohbuchi, "Dense sampling and fast encoding for 3d model retrieval using bag-of-visual features," in *CIVR*, 2009.
[34] Z. Lian, A. Godil, X. Sun, and J. Xiao, "Cm-bof: visual similarity-based 3d shape retrieval using clock matching and bag-of-features," *Mach. Vis. Appl.*, vol. 24, no. 8, pp. 1685–1704, 2013.
[35] Z. Lian, A. Godil, X. Sun, and H. Zhang, "Non-rigid 3d shape retrieval using multidimensional scaling and bag-of-features," in *ICIP*, 2010.
[36] H. Tabia, D. Picard, H. Laga, and P. H. Gosselin, "Compact vectors of locally aggregated tensors for 3d shape retrieval," in *3DOR*, 2013.
[37] D. V. Vranić, "3d model retrieval," *University of Leipzig, Germany, PhD thesis*, 2004.
[38] P. Daras and A. Axenopoulos, "A 3d shape retrieval framework supporting multimodal queries," *IJCV*, vol. 89, no. 2-3, pp. 229–247, 2010.
[39] T. F. Ansary, M. Daoudi, and J.-P. Vandeborre, "A bayesian 3-d search engine using adaptive views clustering," *IEEE Transactions on Multimedia*, vol. 9, no. 1, pp. 78–88, 2007.
[40] G. Lavoué, "Combination of bag-of-words descriptors for robust partial shape retrieval," *The Visual Computer*, vol. 28, no. 9, pp. 931–942, 2012.
[41] P. Papadakis, I. Pratikakis, T. Theoharis, G. Passalis, and S. J. Perantonis, "3d object retrieval using an efficient and compact hybrid shape descriptor," in *3DOR*, 2008.
[42] K. Mikolajczyk and C. Schmid, "An affine invariant interest point detector," in *ECCV*, 2002.
[43] D. Giorgi, S. Biasotti, and L. Paraboschi, "Shape retrieval contest 2007: Watertight models track," *SHREC competition*, vol. 8, 2007.
[44] F. Perronnin, Y. Liu, J. Sánchez, and H. Poirier, "Large-scale image retrieval with compressed fisher vectors," in *CVPR*, 2010.
[45] S. P. Lloyd, "Least squares quantization in pcm," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–136, 1982.
[46] A. Khotanzad and Y. H. Hong, "Invariant image recognition by zernike moments," *TPAMI*, vol. 12, no. 5, pp. 489–497, 1990.
[47] B. Ramamurthi and A. Gersho, "Classified vector quantization of images," *Communications, IEEE Transactions on*, vol. 34, no. 11, pp. 1105–1115, 1986.
[48] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Lost in quantization: Improving particular object retrieval in large scale image databases," in *CVPR*, 2008.
[49] S. Jayanti, Y. Kalyanaraman, N. Iyer, and K. Ramani, "Developing an engineering shape benchmark for cad models," *Computer-Aided Design*, vol. 38, no. 9, pp. 939–953, 2006.
[50] B. Li, Y. Lu, C. Li, A. Godil, T. Schreck, M. Aono, Q. Chen, N. Chowdhury, B. Fang, T. Furuya *et al.*, "Shrec14 track: Large scale comprehensive 3d shape retrieval," *3DOR*, pp. 131–140, 2014.
[51] R. Arandjelovic and A. Zisserman, "Three things everyone should know to improve object retrieval," in *CVPR*, 2012.
[52] M. M. Kazhdan, T. A. Funkhouser, and S. Rusinkiewicz, "Rotation invariant spherical harmonic representation of 3d shape descriptors," in *Symposium on Geometry Processing*, 2003.
[53] A. Agathos, I. Pratikakis, P. Papadakis, S. J. Perantonis, P. N. Azariadis, and N. S. Sapidis, "Retrieval of 3d articulated objects using a graph-based representation," in *3DOR*, 2009.
[54] R. Ohbuchi and T. Furuya, "Distance metric learning and feature combination for shape-based 3d model retrieval," in *Proceedings of the ACM workshop on 3D object retrieval*, 2010.
[55] Q. Chen, B. Fang, Y.-M. Yu, and Y. Tang, "3d cad model retrieval based on the combination of features," *Multimedia Tools and Applications*, pp. 1–19, 2014.
[56] B. Li and H. Johan, "3d model retrieval using hybrid features and class information," *Multimedia tools and applications*, vol. 62, no. 3, pp. 821–846, 2013.
[57] M. Donoser and H. Bischof, "Diffusion processes for retrieval revisited," in *CVPR*, 2013.
[58] X. Bai, X. Yang, L. J. Latecki, W. Liu, and Z. Tu, "Learning context-sensitive shape similarity by graph transduction," *T-PAMI*, vol. 32, no. 5, pp. 861–874, 2010.
[59] C. Li and A. B. Hamza, "Intrinsic spatial pyramid matching for deformable 3d shape retrieval," *International Journal of Multimedia Information Retrieval*, vol. 2, no. 4, pp. 261–271, 2013.
[60] Z. Lian, A. Godil, B. Bustos, M. Daoudi, J. Hermans, S. Kawamura, Y. Kurita, G. Lavoué, H. V. Nguyen, R. Ohbuchi, Y. Ohkita, Y. Ohishi, F. Porikli, M. Reuter, I. Sipiran, D. Smeets, P. Suetens, H. Tabia, and D. Vandermeulen, "SHREC '11 track: Shape retrieval on non-rigid 3d watertight meshes," in *3DOR*, 2011.
[61] D. Nistér and H. Stewénius, "Scalable recognition with a vocabulary tree," in *CVPR*, 2006.
[62] H. Jégou, M. Douze, and C. Schmid, "Product quantization for nearest neighbor search," *TPAMI*, vol. 33, no. 1, pp. 117–128, 2011.
[63] J. H. Friedman, J. L. Bentley, and R. A. Finkel, "An algorithm for finding best matches in logarithmic expected time," *ACM Trans. Math. Softw.*, vol. 3, no. 3, pp. 209–226, 1977.
[64] S. Bai, X. Wang, C. Yao, and X. Bai, "Multiple stage residual model for accurate image classification," in *ACCV*, 2014.

Xiang Bai received the B.S., M.S. and Ph.D. degrees from Huazhong University of Science and Technology (HUST), Wuhan, China, in 2003, 2005, and 2009, respectively, all in Electronics and Information Engineering. He is currently a Professor with the School of Electronic Information and Communications, HUST. He is also the Vice-Director of National Center of Anti-Counterfeiting Technology, HUST. His research interests include object recognition, shape analysis, scene text recognition and intelligent systems. He is a senior member of IEEE.

Song Bai received the B.S. degree in Electronics and Information Engineering from Huazhong University of Science and Technology (HUST), Wuhan, China in 2013. He is currently working toward the Ph.D. degree in the School of Electronic Information and Communications, HUST. His research interests include shape analysis, image classification and retrieval, object detection. He is a student member of IEEE.

Zhuotun Zhu is currently working toward his bachelor degree in the School of Electronic Information and Communications in Huazhong University of Science and Technology (HUST), Wuhan, China. He was awarded Young Microsoft Fellow in 2014, National Scholarship twice in 2013 and 2012. His research interests include deep learning and shape retrieval.

Longin Jan Latecki is a professor at Temple University. His main research interests include computer vision and pattern recognition. He has published 200 research papers and books. He is an editorial board member of Pattern Recognition and International Journal of Mathematical Imaging. He received the annual Pattern Recognition Society Award together with Azriel Rosenfeld for the best article published in the journal Pattern Recognition in 1998.