Takahiro Matsuda, Takahiko Furuya, Ryutarou Ohbuchi, Lightweight binary voxel shape features for 3D data matching and retrieval, *Accepted*, Proc. First IEEE Int'l Conf. on Multimedia Big Data (BigMM) 2015, 20-22 April 2015, Beijing, China

Lightweight binary voxel shape features for 3D data matching and retrieval

Takahiro Matsuda
Graduate School of Medicine and Engineering
University of Yamanashi
Kofu, Japan
g14mk016@yamanashi.ac.jp

Takahiko Furuya
Graduate School of Medicine and Engineering
University of Yamanashi
Kofu, Japan
g13dm003@yamanashi.ac.jp

Ryutarou Ohbuchi
Graduate School of Medicine and Engineering
University of Yamanashi
Kofu, Japan
ohbuchi@yamanashi.ac.jp

Abstract— **This paper proposes several lightweight local 3D shape features for 3D voxel data that yield compact binary feature vectors. These features are inspired by compact binary features for 2D image, namely, Local Binary Pattern (LBP) [22], BRIEF [6] and ORB [26]. In addition to being compact, extraction of proposed 3D features is inexpensive. Furthermore, these binary feature vectors are very efficient to compare, as their distance in Hamming space can be computed very efficiently. Our experimental evaluation of these features in a shape-based 3D model retrieval setting showed that some of these 3D binary features perform competitively to some of existing features. Depending on benchmark database, proposed features are somewhat less accurate than or about as accurate as the state-of-the-art 3D shape features. However, memory footprint is much more compact, at about 1/10 of the non-binary 3D shape features having comparable retrieval accuracy.**

Keywords- 3D geometric models, 3D shape comparison, local feature, binary feature, local binary patterns, BRIEF, ORB.

I. INTRODUCTION

Increasing number of companies are offering their 3D CAD models of parts, mechanical or electronic, on-line. And increasing number of people are creating and using 3D models for personalized manufacturing by using 3D printers. Kinect and other RGBD cameras are about to realize 3D geometry capture for the masses, or for the industry. Medical scanners routinely capture anatomy in 3D. With these trends in mind, effective and efficient methods to compare, recognize, or retrieve 3D shape have become an important area of study [5, 9, 11, 12, 13, 15, 16, 20, 23, 27, 30, 33, 34].

Initially, a predominant approach for 3D shape retrieval used global shape feature per 3D model. That is, only one feature per 3D model is extracted (e.g., [30]), and the feature is used to compare and retrieve 3D models. Recently, more and more algorithms use a set of large number of local features to describe shape of a 3D model. For example, [5, 9, 12, 20, 23, 31] extract hundreds to tens of thousands of local features per 3D model for 3D model to 3D model comparison. In such a case, the cost of computing and storing local features can be significant. One could argue that storage cost is not an issue as these local features are typically aggregated into a feature vector per 3D model by using Bag-of-Features (BF) [7, 29] approach. However, such is not always the case.

Some of the recent applications of 3D shape comparison demand a large number of local 3D features to be computed and then stored without aggregation. An example is object recognition in a scene, e.g., recognition of a tumor-like anatomy in a volume data acquired by X-ray CT scanner. Or, one may need to find a part having specific shape (e.g., a rotor of an electric motor) from an array of assembled mechanical apparatus (e.g., a dishwasher) in a database. In such a case, a large number of local 3D features extracted from a complex 3D "scene" need to be stored as they are, without aggregation, so that parts of the scene could be matched later with a query. Since location, orientation, and scale of the desired part in the scene are not known, the local features need to be aggregated or compared many times over for the search. Thus, costs of *extracting*, *storing*, and *matching* a large number of these local features, numbering easily in thousands per 3D object, become very important. To this end, traditional local features, such as the 2D image feature SIFT [17] and the 3D shape feature LSF [20] have difficulty satisfying one or more of these requirements. A SIFT feature, for example, is a 128 dimensional (D) floating point vector that requires 512 Byte to store per feature. LSF, a derivative of SPRH by Wahl [34], requires 2,500 Byte per feature for storage. An FPFH [27] feature, a 125D vector, requires 500 Bytes for storage. Furthermore, comparison among LSF or FPFH feature may not be fast enough for repeated re-aggregation required for searching through a large 3D scene for a queried partial shape.

In this paper, we propose and compare several local 3D shape descriptors that are compact and very quick to compare. For the study described in this paper, we assume that 3D shapes are represented by using voxel representation. A voxel data may be captured, for example, by X-ray CT scan or by confocal microscopy. Alternatively, a voxel data can be rendered from a surface-based 3D model, which may or may not define a solid. Our proposed features are inspired by local binary 2D image features BRIEF [6] and ORB [26], whose feature vector are binary bit strings. We experimentally evaluate retrieval accuracy and computational cost of the proposed features by using a 3D shape retrieval scenario. Contribution of this paper can be summarized as follows;

- Proposal of multiple local 3D geometrical features that produce binary feature vector. Binary features are very compact. They are also very fast to compare in Hamming space.
- Experimental evaluation of proposed 3D features by using a shape-based 3D model retrieval scenario. The evaluation showed that the proposed features are very compact and fast to compare, while retaining most of the accuracy of features that have much larger memory footprint.

II. Related Work

In the domain of 2D image analysis, SIFT [17], SURF [4] and other feature descriptors have been used extensively, which is invariant, to varying degrees, to geometrical transformations including scaling, translation, and rotation. However, these descriptors are rather expensive to compute. In addition, their feature vectors have large memory footprints, as they use high dimensional floating point vectors. For example, a SIFT vector having 128D has 512 Byte, or 4,096 bit, memory footprint. For object localization or image retrieval, hundreds to thousands of these features per image are sampled, occupying megabytes of memory.

For 3D shape comparison and retrieval, the situation could get worse. If the method is view-based, and performs multi-view rendering to achieve 3DOF rotation invariance, a large number of 2D features are produced. For example, in [9], about 10k SIFT features are computed per 3D model. One may use a 3D shape feature densely sampled on a 3D model or a 3D scene. 3D shape features SPRH [34] and its localized derivatives LSF [20], FPFH [27] are computed from oriented point set. The LSF has a dimensionality of 625 and requires 2,500 Byte. FPFH feature, which is very similar to SPRH above, occupies 500 Bytes. 3D shape features VSIFT [23] and 3DSURF [11] are computed from voxel representation of 3D shape. Features of VSIFT and 3DSURF are floating vectors having 1,280D and 162D, respectively.

If an entire 3D model is compared with the whole of the other 3D model, one could aggregate a set of a large number of local features into a feature vector per 3D model, e.g., by using Bag-of-Features (BF) [7, 29] or Fisher Vector [24]. However, if the task is to recognize a sub-part of a 3D model or a scene by using a scheme similar to moving sub-window, local features are often retained as they are for they need to be aggregated repeatedly for each sub-part. In such a case, cost of storing and aggregating thousands or tens of millions of local features becomes critical.

Recently, a class of "compact binary" features are proposed that are lightweight to compute, compact, and efficient to compare. These features employ binary vector having tens to hundreds of bits for their feature. Binary Robust Independent Elementary Features (BRIEF) [6], Oriented FAST and Rotated Brief (ORB) [26] are two of the examples of this class of features. In this paper, we try to extend these binary 2D image features into the domain of 3D shape represented by voxels.

III. Lightweight 3D Shape Features from Voxels

In this paper, we propose three voxel shape features 3D Local Binary Pattern (3DLBP), 3DBRIEF, and 3DORB that produce binary feature vectors. The last, 3DORB, includes four variations, 3DORB-PCA, 3DORB-CG, 3DORBL-CG, and 3DORBL-R-CG, depending on the way rotation normalization is performed and the way features are computed. These features are evaluated by using a 3D shape retrieval scenario explained below.

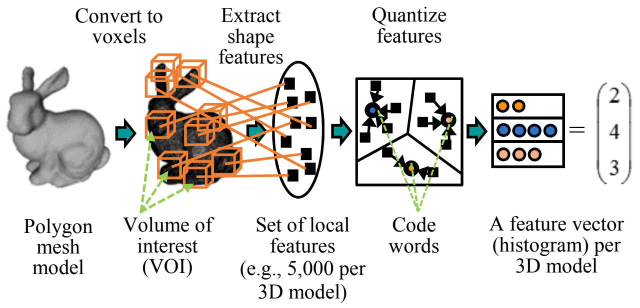


Figure 1. A set of local, voxel-based 3D shape features is extracted from the voxel representation of a 3D model. Local features are aggregated into a feature vector per 3D model for 3D-model-to-3D-model comparison. Proposed binary features use Hamming distance for vector quantization.

A. Surface-based 3D Shape Model Retrieval Pipeline

The steps for 3D model to 3D model shape comparison are as follows. Please refer to Figure 1 for block diagram of the pipeline. Our 3D shape retrieval scenario converts, for feature extraction, input surface-based 3D models into one of two kinds of voxel-based 3D shape models; either *volume voxel model* or *surface voxel model*. A volume voxel model represents volume of a 3D model by voxels. A volume voxel model can only be computed if the original surface-based 3D model defines a solid. If the original surface based 3D model is not solid, e.g., if it is an open mesh or a polygon soup, it can only be converted to a surface voxel model that uses voxels to represent surfaces. For a benchmark database consisting only of solids, retrieval experiments are done using both volume voxel model and surface voxel model of the database. On the other hand, if the database contains non-solid 3D models, retrieval experiments are done using surface voxel model only.

1. **Surface to voxel conversion:** A surface-based 3D model is centered at coordinate origin, and scaled to fit in a unit sphere. It is then converted into a voxel representation having size 128^3 in a voxel buffer of size 150^3. A non-solid surface-based 3D model is converted to a surface-based voxel model. A solid model is converted to either surface voxel model or solid voxel model for comparison.
    (a) **Voxel pyramid generation (optional):** For 3DBRIEF and 3DORB, a voxel scale space, or *voxel pyramid*, is generated by repeated low-pass filtering and 2-to-1 down sampling for multi-scale feature extraction.
    (b) **Gaussian smoothing (optional):** Surface voxel model is blurred by using Gaussian filter prior to feature computation so that voxelized surfaces have thicknesses more than one voxel. A feature works better with these thicker surfaces. Final size of a 3D voxel model after filtering becomes 132^3.
2. **Sample point selection:** Features are extracted from M sample points randomly and densely selected from non-zero

voxels within the voxel buffer. In the experiments described later, we use M=3,000 and 5,000.

3. **Feature extraction:** A cubic *Volume Of Interest* (*VOI*) of size $l \times l \times l$ is set around the sample point, and a feature is computed. For 3DBRIEF and 3DORB, this is done at each resolution level of the pyramid. Details of the feature extraction algorithms will be described later.
   (a) **VOI rotation normalization (optional):** For 3DORB, each VOI is rotation normalized prior to feature extraction. We compare two rotation normalization algorithms that will be described later. The other two features, 3DLBP and 3DBRIEF, do not perform rotation normalization.
4. **Feature aggregation:** A set of M local features are aggregated into a feature vector per 3D model by using *Bag-of-Features* (*BF*) approach.
5. **Ranking:** Database 3D models are ranked against the query 3D model based on the similarity between the aggregated feature vectors of the query and the target (DB) 3D models.

B. Converting Polygonal Surface Model into Voxel Model

Features used in this paper are extracted from voxel, or 3D pixel, representation of 3D shapes. Thus, prior to feature extraction, our algorithm converts input 3D models represented as polygonal mesh into their voxel representations. As described above, for a surface-based 3D model defining a solid, two variations to voxel representations, *surface voxel model* and *volume voxel model*, are generated. Models found in McGill Shape Benchmark [19], for example, define solids. For a non-solid surface-based 3D model, only a surface-based voxel model is generated. A 3D model is converted into a cubic volume having size $n \times n \times n$. Based on preliminary experiments, we chose n=128, inset a voxel buffer of size 150^3.

To generate a surface voxel model, we use *Out-Of-Core Sparse Voxel Octree* (OOC-SVO) algorithm by Baert et al [3]. It exploits octree data structure for fast voxelization. As an option, a surface voxel model may be Gaussian smoothed using a kernel having size 5^3 to produce a surface voxel model having surface thickness more than 1 voxels. (The Gaussian smoothing makes a voxel model to expand to the size 132^3.) To generate a solid voxel model, we use the algorithm by Abdellah et al [1]. The algorithm first generates a voxel model of closed surface, then run a flood-fill to convert it into a filled, solid voxel model.

C. Dense Local Voxel Feature Extraction

In case of 2D image features, features are often extracted at interest points detected by such interest point detectors as FAST [25] or Difference of Gaussian (DoG) detector in SIFT [17]. Interest point detection is a must if matching of local features among images (or 3D models) is the objective. For object recognition and retrieval, however, dense sampling of features often outperforms interest-point based feature sampling. As our objective is retrieval and recognition of 3D object, we assume dense sampling of 3D features on or near the 3D object. We randomly and densely select voxels having non-zero value, and use them as "forced" (v.s. detected) keypoints. Then, at each keypoint, one of the following features is computed.

1) 3DLBP

This feature, called *3D Local Binary Pattern* (*3DLBP*), is a 3D counterpart to a variation of *Local Binary Pattern* (*LBP*) feature for 2D images [21, 22]. A 3DLBP feature is computed from a cubic *Volume Of Interest* (*VOI*) of size $l \times l \times l = l^3$. Within the VOI, multiple *Sub-Local Binary Patterns* (*SLBPs*) are computed at every (i.e., l^3) voxel in the VOI. Each SLBP, a 6 bit string, is computed by comparing the value of the SLBP center voxel with the values of its 6 neighboring voxels using a fixed voxel traversal order (Figure 2). The bit corresponding to a neighboring voxel becomes '1' ('0') if its value is less than (greater or equal to) the center voxel. Total of l^3 SLBP bit strings are aggregated into a 3DLBP feature vector for the VOI having 64 dimensions by counting population of 64 (0~63) possible values of the 6 bit SLBPs. A 3DLBP is thus not a binary feature but a 64D integer-valued vector. This aggregation per VOI for the 3DLBP may be considered as a very simple form of Bag-of-Features with a fixed 64 word vocabulary.

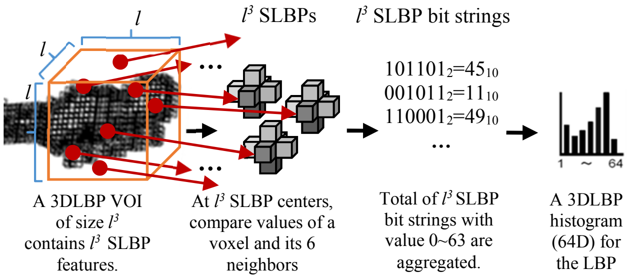

l
l
l
l^3 SLBPs
l^3 SLBP bit strings
$101101_2=45_{10}$
$001011_2=11_{10}$
$110001_2=49_{10}$
…
1 ~ 64
A 3DLBP VOI of size l^3 contains l^3 SLBP features.
At l^3 SLBP centers, compare values of a voxel and its 6 neighbors
Total of l^3 SLBP bit strings with value 0~63 are aggregated.
A 3DLBP histogram (64D) for the LBP

Figure 2. The *3D Local Binary Pattern* (*3DLBP*) feature computes a binary string based on comparison of local voxel values within the volume of interest about the feature center.

The number of SLBPs per 3DLBP feature, or the size of VOI, l^3 is a parameter that affects performance and computational cost of the 3DLBP feature. It is possible to use neighborhood size of SLBP larger than 6, e.g., 18, but the dimensionality of the aggregated 3DLBP feature per the VOI quickly becomes too large (e.g., 2^{18}=262,144) with little or no gain in accuracy.

To gain certain degree of invariance to scale, we apply the 3DLBP to a multi-scale voxel pyramid. To form a multi-scale pyramid of voxels, we simply down sample the object having 128^3 resolution by half in each axis, producing 4 levels of representations having voxel size 128^3, 64^3, 32^3, and 16^3 (Figure 3), before applying 3DLBP at each resolution level.

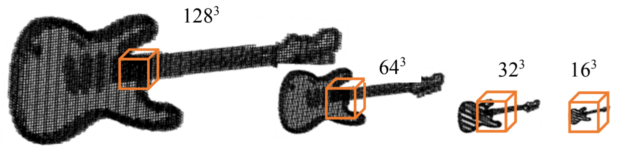


Figure 3. Feature extraction from multi-scale (4 levels) voxel pyramid.

2) 3DBRIEF

3DBRIEF is a simple 3D binary feature based on the BRIEF feature [6], and is very lightweight to extract. An N bit BRIEF feature vector is produced by comparing values of N randomly selected pairs of voxels. If a pair of voxels at locations **a** and **b** is selected, their values $vox(\mathbf{a})$ and $vox(\mathbf{b})$ are compared according to equation (1) to yield a bit of the N bit feature vector. Typically, number of bits N of the descriptor is in the range tens to hundreds of bits. In the experiments that follow in this paper, we used N=256 based on the preliminary experiments.

$$\tau(a,b) = \begin{cases} 1, & if\ vox(\mathbf{a}) < vox(\mathbf{b}) \\ 0, & otherwise \end{cases} \tag{1}$$

3DBRIEF does have invariance against translation, but it lacks invariances against rotation and scaling. We extract 3DBRIEF on a voxel image pyramid to provide for certain invariance against scaling (but not rotation). All the features extracted from multiple scales are later aggregated together into a feature vector by using BF approach.

3DBRIEF is very fast to compare, as it is a binary vector and the comparison is made by computing Hamming distance. Hamming distance between a pair of binary strings can be computed very efficiently by using a combination of (SIMD) an XOR instruction and a population count instruction.

3) 3DORB

Adding rotation invariance to 3DBRIEF by 3DOF orientation normalization produces *3D ORiented Brief* or *3DORB*. The rotation normalization is applied to each local VOI, not for the entire voxel model.

We experimented with two approaches for orientation normalization; one is called 3DORB-CG for its use of *Center of Gravity* (*CG*) for orientation normalization, and the other is called 3DORB-PCA for its use of *Principal Component Analysis* (*PCA*). There are two variants to feature computation method. 3DORB compares a pair of individual voxel values to derive feature vector, as in 3DBRIEF. A variant, called 3DORBL with suffix "L", uses a smoothed (averaged) value computed from a small (e.g, 3×3×3) *Volume of Smoothing* (*VOS*) of size $k \times k \times k$, $k << l$, for the comparison. 3DORBL with smoothing exhibits higher accuracy than 3DORB, albeit at a higher computational cost. 3DORBL-CG has a subtype called 3DORBL-R-CG. The 3DORBL-R-CG has less cost of rotation normalization than the 3DORBL-CG. Figure 4 illustrates these variants.

2D image version of the ORB [26] employs a simple learning strategy to determine most effective N pairs of pixels for its N bit descriptor. We did not employ this learning, as our preliminary experiments showed that the learning did not improve accuracy. We simply select N randomly selected pairs of voxels for 3DORB signature.

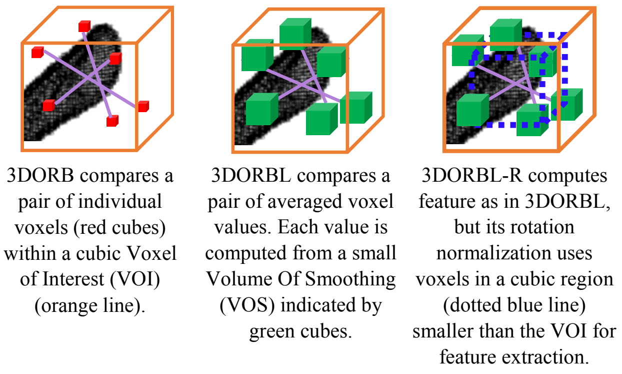

3DORB compares a pair of individual voxels (red cubes) within a cubic Voxel of Interest (VOI) (orange line).
3DORBL compares a pair of averaged voxel values. Each value is computed from a small Volume Of Smoothing (VOS) indicated by green cubes.
3DORBL-R computes feature as in 3DORBL, but its rotation normalization uses voxels in a cubic region (dotted blue line) smaller than the VOI for feature extraction.

Figure 4. Three variants to 3DORB.

a) 3DORB-CG

The 3DORB-CG employs the same strategy as 2D version of the ORB for rotation normalization. We first compute 0th and 1st order moment m_{pqr} of the voxels in a cubic VOI, whose length of an edge is n voxels. Moment m_{000} is 0th order, that is, sum of all the voxel values in the VOI [25]. Three 1st order moments m_{100}, m_{010}, m_{001} are 1st order moments along x, y, and z axes, respectively [10].

$$m_{pqr} = \sum_{x,y,z} x^p y^q z^r vox(x,y,z) \tag{2}$$

$$(\bar{x}, \bar{y}, \bar{z}) = \left(\frac{m_{100}}{m_{000}}, \frac{m_{010}}{m_{000}}, \frac{m_{001}}{m_{000}}\right) \tag{3}$$

Then, we compute a vector from the CG to the center of VOI as a rotational axis, and align the VOI along the direction. We then find and compensate for the rotation about this axis, by finding a farthest non-zero voxel (the green sphere in Figure 5), and rotate the VOI about the axis. Note that the rotation normalization is not effective if the estimated displacement vector from the center of VOI to the CG is unreliable. In our implementation, we forego the rotation normalization if the length of computed displacement vector is less than 1/5 of the size n of side of the VOI.

After the rotation normalization, a feature vector, in the form of a bit string, is extracted in the manner similar to 3DBRIEF. As with the case of 3DBRIEF, for scale invariance, features are extracted from multi-scale pyramid of voxels. Based on our preliminary experiment, number of bits N=256 for the 3DBRIEF is used in the following.

b) 3DORB-PCA

For rotation normalization, the 3DORB-PCA applies PCA to the 3×3 covariance matrix of voxel values within the VOI. Principal axes obtained from the PCA indicate orientation, but directions of the vectors are left undefined. Direction of the axes is disambiguated by comparing the center of gravity and the (geometric) center of the VOI. After the rotation normalization, feature vector is extracted in the manner similar to 3DORB-CG (or 3DBRIEF). We use, in the following experiment, the feature vector length N=256 bit.

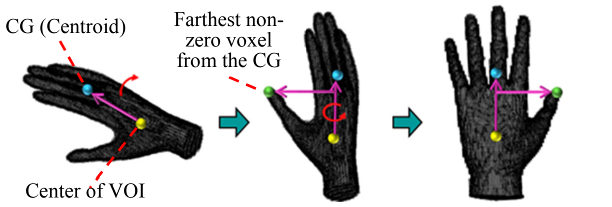


Figure 5. Rotation normalization of voxels for the 3DORB-CG feature. This rotation normalization is performed for each local VOI, not for the entire 3D model. (Best viewed in color.)

4) 3DORBL-CG and 3DORBL-R-CG

Comparing a pair of individual voxels, as in BRIEF, 3DBRIEF and 3DORB, is somewhat prone to noise. To improve robustness against noise, we tried a variation of 3DORB that uses an average of voxel values in a small cubic region called Volume of Smoothing (VOS) for the comparison. It amounts to box-filtering voxel values, and we employ the (3D) integral image for fast computation of local sum of voxel values in a VOS. We identify the smoothing by adding letter "L" to the name of the methods, as in 3DORBL-CG and 3DORBL-R-CG.

For 3DORBL-CG, rotation normalization is done by using all the voxels in the VOI of size $l \times l \times l$ for feature computation. For a larger VOI size, this leads to a rather high cost of rotation normalization. We thus created a variation of 3DORB-CG called 3DORBL-R-CG ("R" stands for "reduced cost") that has reduced cost of rotation normalization. In 3DORBL-R-CG, the (cubic) region used for rotation normalization is smaller than that of the VOI for feature computation.

D. Local Feature Aggregation by using Bag-of-Features

Local features described above are aggregated into a feature vector per 3D model by using the *Bag-of-Feature* (*BF*) approach [7, 29]. The BF approach learns, at its preprocessing stage, a set of vocabulary, or codebook, from the 3D shape descriptors extracted from a set of 3D models either identical or similar to the retrieval target. The codebook is typically learned by clustering the features, e.g., by using *k*-means clustering algorithm. Since the features above are binary bit string, instead of *k*-means in *Euclidian* space (i.e., using *L2-norm*), we run *k*-means in *Hamming* space (i.e., Hamming distance.). We modified the code of *k*-means++ [2], which performs clustering in Euclidian space in its original form, so that the clustering in Hamming space can be computed.

After the dictionary is learned, a descriptor is vector quantized, in Hamming space, into a word from the dictionary. Histogram of words, whose dimension is the number of words in the dictionary, becomes the descriptor for the 3D model. The size of *k*, or the number of words in the codebook, significantly affects retrieval accuracy. A feature vector per 3D model thus created is a histogram whose element is an integer, and most of the elements are zero.

E. 3D Model-to-3D Model Dsitance Computation

A distance among a pair of per-3D-model feature vectors (that is, features after aggregation) is computed, per database, by using either *L1-norm*, *L0.5-norm*, or (symmetricized) *Kullback-Leibler Divergence* (*KLD*). For each database, we try all the three distances, and choose the best performing one to derive retrieval accuracy indices for the database. (The KLD may face divide-by-zero error as a BF-aggregated feature vector would contain many elements with value zero. To circumvent the error, we replace a zero with a very small non-zero value, e.g., 1×10^{-20} .)

$$d_{L1}(x_i, x_j) = \sum_{a=1}^{n} | x_{ia} - x_{ja} | \tag{4}$$

$$d_{L0.5}(x_i, x_j) = \left(\sum_{a=1}^{n} \sqrt{x_{ia} - x_{ja}} \right)^2 \tag{5}$$

$$d_{KLD}\left(x_i, x_j\right) = \sum_{a=1}^{n} \left(x_{ia} - x_{ja}\right) \log\left(\frac{x_{ia}}{x_{ja}}\right) \tag{6}$$

IV. Experiments and Results

In this section, we evaluate accuracy, computational cost, and memory footprint of the proposed binary features by using four benchmarks for shape-based 3D model retrieval. We used MSB [19] for highly articulated but less detailed shapes defined as solid. The MSB consists of 255 models in 10 classes. The MSB includes such articulated shapes as "humans", "octopuses", "snakes", "pliers", and "spiders". We used PSB [28] test set for a set of non-solid 3D models (many of them polygon soup model) having quite diverse but mostly rigid shapes. The PSB test-set contains 907 models partitioned into 92 classes. The PSB contains a very diverse, mostly rigid 3D shapes, including cars, chairs, potted plants, airplanes, range-scanned (open-mesh) faces, insects, etc. For the MSB and PSB, query models are drawn from the retrieval targets, and every pair of models in the database are queried to compute retrieval accuracy. As the third benchmark, we used ESB [8] database, which consists of 867 mechanical CAD models of machine parts, including gears, pipes, screws, flanges, etc., divided into 45 classes. Fourth benchmark is SHREC 2011 Non-Rigid (SH11NR) [14], which contains 600 non-rigid, solid 3D models in 30 classes extracted from MSB and other dataset. As with the MSB, ESB and PSB, in SH11NR, query models are drawn from the retrieval targets, and every pair of models in the database are queried to compute retrieval accuracy. Examples of 3D models from these four benchmark databases are shown in Figure 6.

As performance indices, we use *Mean Average Precision* (*MAP*) [18].

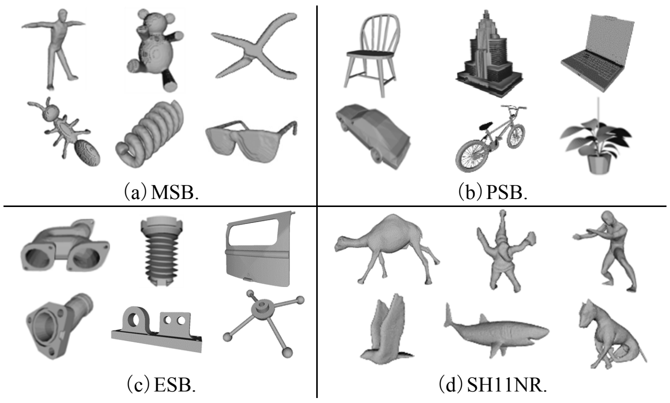


Figure 6. Example 3D models from the four benchmark databases, MSB, PSB, ESB, and SH11NR.

A. Retrieval accuracy of features

Figure 7 compares retrieval accuracy of 8 features as recall-precision plots. Bar graphs of Figure 8 compare retrieval accuracy of 8 features measured as MAP [%] by using the four benchmarks. 3D models of MSB, ESB, and SH11NR define solids. These models are converted to both solid voxel and surface voxel representations in order to compare performances of features among these two representations. Then, two of the features, 3DORBL-CG and 3DORBL-R-CG, are tested on both solid voxel and surface voxel representations. (3D models in PSB can only be converted to surface voxel representations.) Consequently, there are 10 "algorithms" listed in the graph. Top two algorithm, LSF [20], and BF-DSIFT [9], are used as baseline.

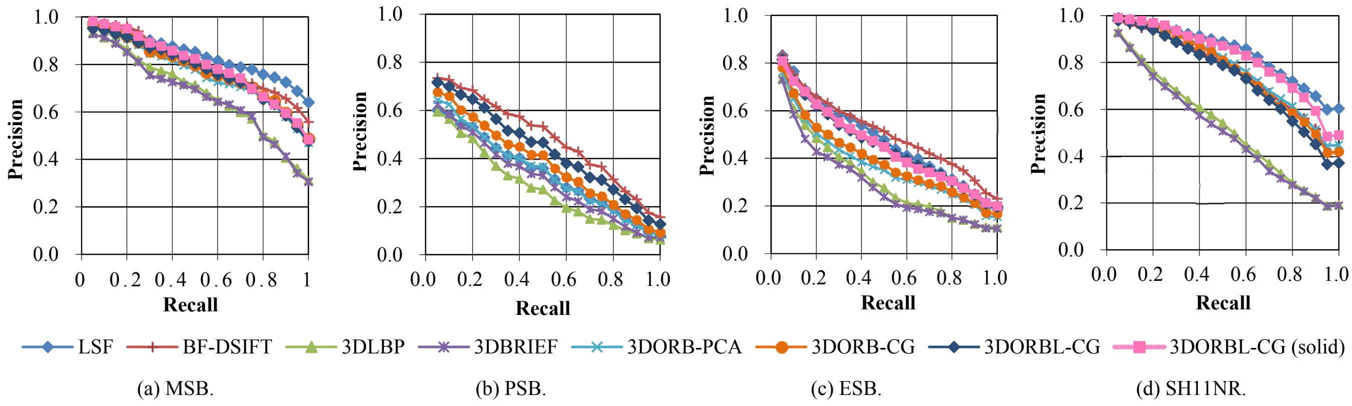


Figure 7. Recall-precision plots for four benchmark databases.

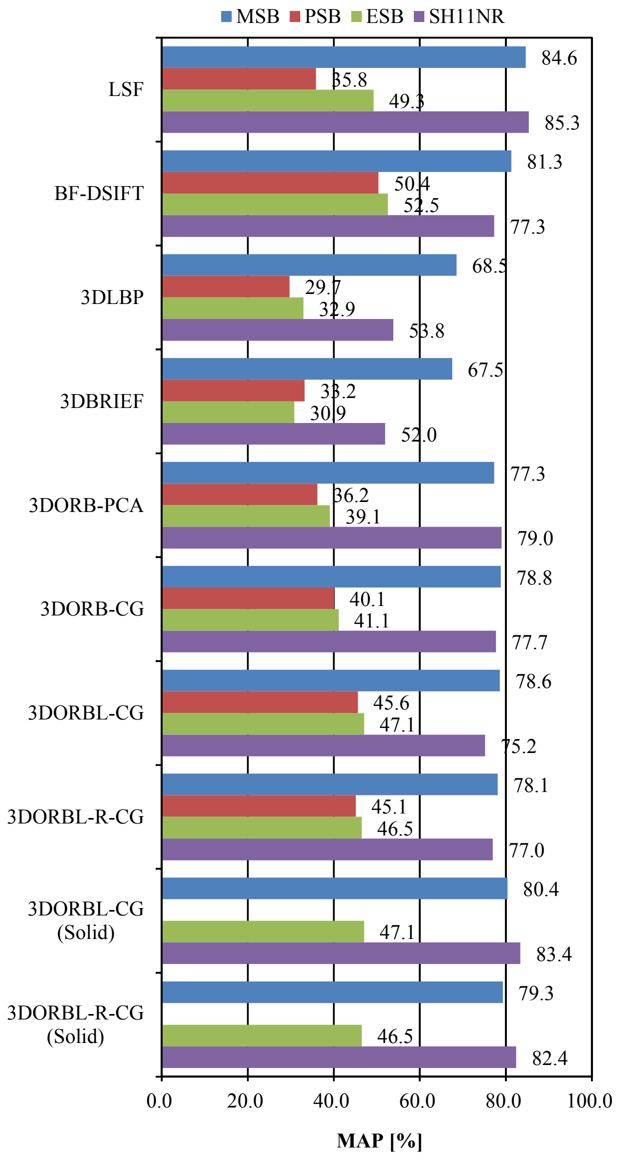


Figure 8. Comparison of retrieval accuracy in MAP [%] by using four benchmarks, eight algorithms, and two voxelization methods.

According to some SHREC tracks [15], the BF-DSIFT is one of better performing features, especially for rigid polygon soup models of PSB as well as for non-rigid models of MSB. Overall, view-based BF-DSIFT performs the best for PSB containing many polygon soup models. LSF, which captures 3D structure by sampling surface by points, does well for ESB, SH11NR and MSB.

3DORB-CG, with its rotation normalization, clearly outperforms BRIEF and 3DLBP, as well as 3DORB-PCA. Of rotation normalized 3DORB-CG variations, 3DORBL-CG, which compares locally smoothed voxel values, did better. 3DORBL-R-CG, a variation of 3DORBL-CG with less stringent but faster rotation normalization, did slightly worse than 3DORBL-CG for the ESB, but almost as well as on the other two benchmarks.

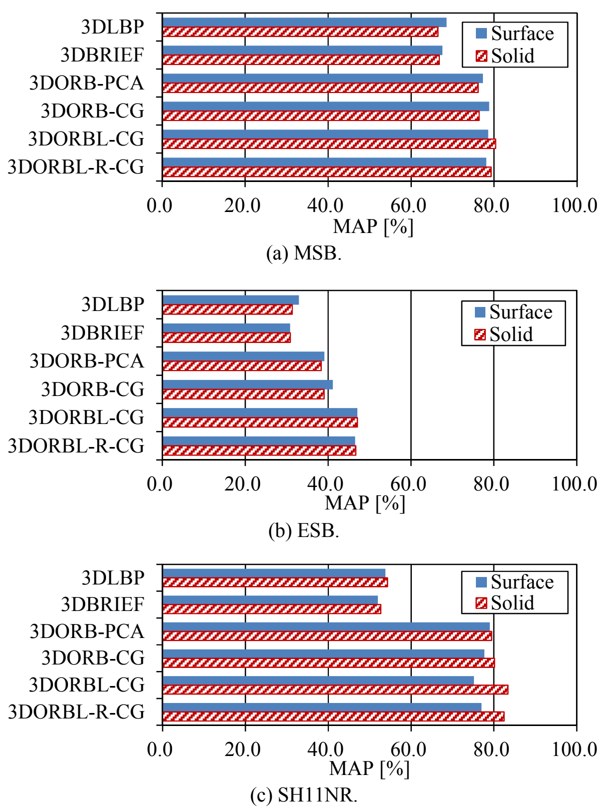
Figure 9 compares, using 3 benchmarks consisting of solid 3D models, 6 proposed algorithms for their retrieval accuracy on solid voxel and surface voxel representations of 3D shapes. Overall, for all the three benchmarks, there is only small difference between solid voxel and surface voxel representations.

Our observations is that, at voxel resolution of 128^3, solid voxel and surface voxel models become nearly identical, especially for ESB whose mechanical parts, e.g., flanges, panels, gears, or tubes have thin walls. In the case of SH11NR, with its solid models of animals etc. having thick body, solid voxel representation worked for 3DORBL-CG and 3DORBL-R-CG features.

B. Spatial and Temporal Costs

Table 1 shows execution timings, per query, of the features for the 3D model retrieval task using PSB benchmark. The cost of feature extraction includes Gaussian filtering and multi-scale voxel pyramid construction. For each algorithm (row), timing for voxelization (VX), feature extraction (FE), vector quantization (VQ), distance computation (DC), and total time for retrieval (RET) are shown in seconds [s]. Time for voxelization includes Gaussian filtering, if applicable, and feature extraction includes time for voxel pyramid construction

In terms of feature extraction, 3DBRIEF is very fast, requiring only 0.12s per 3D model, while 3DORB-CG and 3DORBL-CG requires about 10s per 3D model. The latter two spend most of their time performing rotation normalization. In



(a) MSB.

(b) ESB.

(c) SH11NR.

Figure 9. Retrieval accuracy in MAP for 3D model benchmark databases that consists only of solid models.

Table 1. Execution timings for processing a query. Timings are measure for PSB benchmark by using surface voxel representation. Retrieval accuracy in MAP [%] for PSB is also shown.

Algorithm	VX [s]	FE [s]	VQ [s]	DC [s]	RET [s]	MAP [%]
LSF		1.54	6.11	0.002	7.65	35.8
BF-DSIFT (CPU)		33.06	0.02	0.030	33.11	50.4
BF-DSIFT (GPU)		1.12	0.02	0.030	1.17	50.4
3DLBP	0.05	0.60	0.78	0.002	1.43	29.7
3DBRIEF	0.05	0.12	0.24	0.002	0.41	33.2
3DORB-PCA	0.05	2.49	0.24	0.002	2.78	36.2
3DORB-CG	0.05	9.36	0.24	0.002	9.65	40.1
3DORBL-CG	0.05	10.35	0.24	0.002	10.64	45.6
3DORBL-R-CG	0.05	0.76	0.24	0.002	1.05	45.1

Table2. Memory footprints of features in typical usage.

Features	Size per feature [Byte]	# features / 3D model	Memory footprint per 3D model [Byte]
LSF	2,500	2,048	5,120k
BF-DSIFT (CPU)	512	12,600	6,451k
BF-DSIFT (GPU)	512	12,600	6,451k
3DLBP	256	3,000	768k
3DBRIEF	32	5,000	160k
3DORB-PCA	32	5,000	160k
3DORB-CG	32	5,000	160k
3DORBL-CG	32	5,000	160k
3DORBL-R-CG	32	5,000	160k

comparison, 3DORBL-R-CG is quite economical, requiring only 0.76s for both rotation normalization and feature extraction. Total execution time of 3DORBL-R-CG is also fast among the features compared. Yet, as noted in the previous section, retrieval accuracy of 3DORBL-R-CG is only slightly worse than much slower 3DORBL-CG.

Table 2 compares memory footprints of the features. Here the comparison is not as easy, since the best performing number of features per 3D model varies depending on the benchmark database, etc. Still, it is apparent that 3DBRIEF and 3DORB are compact, requiring only 32 Bytes per feature. In terms of memory footprint per 3D model, which assumes all the local features need to be retained, binary features of 3DORB and 3D BRIEF require much less (1/10) space than floating point feature vectors of LSF, etc.

V. Summary and Conclusion

With increasing popularity of 3D shape models, shape similarity search of such 3D models is about to become an important tool. In this paper, we have proposed local, lightweight, binary 3D shape features. We have experimentally evaluated their retrieval accuracy, computational cost, and storage cost. Our evaluation experiments are set in the context of 3D shape retrieval. According to the experiments, one of the proposed features, 3DORBL-CG, produced very compact feature and performs only slightly worse than some of the state-of-the-art 3D model retrieval algorithms. An appearance based retrieval algorithm, BF-DSIFT [9], outperforms proposed 3DORBL-CG for retrieval accuracy for the PSB benchmark. However, BF-DSIFT feature requires memory footprint of 6.4 MByte, while that of 3DORBL-CG is only 1/10 at 160 Kbyte.

In the future, we intend to improve the proposed binary features both in terms of computational cost and retrieval accuracy. For example, we plan to explore a faster rotation normalization algorithm and GPU implementation. We also would like to try Fisher Vector for binary features [32] for an aggregation method better than Bag-of-Features.

References

[1] M. Abdellah, voxelizer2. https://github.com/marwan-abdellah/ voxelizer-2/, (accessed 2015-02-21).

[2] D. Arthur, S. Vassilvitskii, "k-means++: The Advantages of Careful Seeding," *Proc. ACM-SIAM Symp. on Discrete Algorithms 2007*, pp.1027-1035, (2007).

[3] J. Baert, A. Lagae, P. Dutr´e, "Out-of-core construction of sparse voxel octrees," *Proceedings of the 5th High-Performance Graphics Conference. ACM, 2013*. pp. 27-32. (2013).

[4] H. Bay, T. Tuytelaars, L. Van Gool, "SURF: Speeded Up Robust. Features," *Proc. ECCV'06*, pp. 404-417, (2006).

[5] A.M. Bronstein, M.M.Bronstein, L.J. Guibas, M. Ovsjanikof, Shape google: Geometric words and expressions for invariant shape retrieval, *ACM TOG*, **30**(1), pp.1-20, (2011).

[6] M. Calonder, V. Lepetit, C. Strecha, P. Fua, "BRIEF: Binary robust independent elementary features," Proc. ECCV 2010, 778-792, (2010).

[7] G. Csurka, C.R. Dance, L Fan, J Willamowski, C. Bray, "Visual Categorization with Bags of Keypoints," *Proc. European Conference on Computer Vision '04 workshop on Statistical Learning in Computer Vision*, pp. 59-74, (2004).

[8] Engineering Shape Benchmark, <http://shapelab.ecn.purdue.edu/.>

[9] T. Furuya, R. Ohbuchi, "Dense sampling and fast encoding for 3D model retrieval using bag-of-visual features," *Proc. ACM CIVR 2009*, *Article No. 26*, (2009).

[10] M. K. Hu. "Visual pattern recognition by moment invariants," *IRE Trans. Information Theory*, **8**(2), pp. 179–187, (1962).

[11] J. Knopp, M. Prasad, G. Willems, R. Timofte, L. Van Gool, "Hough transform and 3D SURF for robust three dimensional classification," *Proc. ECCV'10*, 589-602, (2010).

[12] G. Lavoué, "Bag of Words and Local Spectral Descriptor for 3D Partial Shape Retrieval," *Proc. EG Workshop on 3D Object Retrieval 2011* (3DOR 2011), pp.41-48, (2011).

[13] B. Li, Y. Lu, C. Li, A. Godil, T. Schreck, M. Aono, Q. Chen, N.K. Chowdhury, B. Fang, T. Furuya, H. Johan, R. Kosaka, H. Koyanagi, R. Ohbuchi, A. Tatsuma, "SHREC'14 Track: Large Scale Comprehensive 3D Shape Retrieval", *Proc. EG Workshop on 3D Object Retrieval 2014* (3DOR 2014), pp.131-140, (2014).

[14] Z. Lian, A. Godil, B, Bustos, M, Daoudi, J. Hermans, S. Kawamura, D. Vandermeulen, SHREC'11 Track: Shape Retrieval on Non-rigid 3D Watertight Meshes. 3DOR, 11, 79-88, (2011).

[15] 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, D. Vandermeulen, "A comparison of methods for non-rigid 3D shape retrieval," *Pattern Recognition*, **46**(1), January, (2013).

[16] Y. Liu, Z. Zha, H. Qin, Shape Topics, "A Compact Representation and New Algorithms for 3D Partial Shape Retrieval," *Proc. IEEE CVPR*, Vol.2, pp.2025-2032, (2006).

[17] D.G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *IJCV*, 60(2), (2004).

[18] D. Manning, P. Raghavan, H. Schutze, *"Introduction to Information Retrieval*, Cambridge University Press (2008).

[19] McGill 3D Shape Benchmark, <http://www.cim.mcgill.ca/~shape/benchMark/.>

[20] Y. Ohkita,Y. Ohishi,T. Furuya, R. Ohbuchi, "Non-rigid 3D Model Retrieval Using Set of Locasl Statistical Features," *Proc. IEEE ICME 2012 Workshop on Hot 3D*, pp. 593-598, (2012).

[21] T. Ojala, M. Pietikäinen, and D. Harwood, "Performance evaluation of texture measures with classification based on Kullback discrimination of distributions", Proceedings of the 12th IAPR International Conference on Pattern Recognition (ICPR 1994), vol. 1, pp. 582 - 585 (1994).

[22] T. Ojala, M. Pietikäinen, and D. Harwood, "A Comparative Study of Texture Measures with Classification Based on Feature Distributions", Pattern Recognition, vol. 29, pp. 51-59 (1996).

[23] K. Osada, T. Furuya, R. Ohbuchi, "SHREC'08 entry: Local volumetric features for 3D model retrieval," *Proc. IEEE Shape Modeling and Applications* (SMI 2008), *IEEE International Conference on*, pp. 245 – 246, (2008).

[24] F. Perronnin, J. Sánchez, and T. Mensink. "Improving the fisher kernel for large-scale image classification", *In Proc. ECCV, 2010*, pp.143-156. (2010).

[25] E. Rosten, T. Drummond, "Fusing points and lines for high performance tracking," *Proc. ICCV 2005*, Vol. 2: 1508–1511, (2005).

[26] E. Rublee, V. Rabaud, K. Konolige, G. Bradski, "ORB: an efficient alternative to SIFT or SURF," *Proc. ICCV* '11, pp. 2564–2571, (2011).

[27] R. B. Rusu, N. Blodow, M. Beetz, Fast point feature histograms (FPFH) for 3D registration, *Proc. IEEE International Conference on Robotics and Automation* (*ICRA*) 2009, pp. 1848-1853, (2009).

[28] P. Shilane, P. Min, M. Kazhdan, and T. Funkhouser, "The Princeton Shape Benchmark," *Proc. SMI '04*, 2004, pp. 167-178, <http://shape.cs.princeton.edu/benchmark/.>

[29] J. Sivic, B. Russell, A. Efros, A. Zisserman, and W. Freeman, "Discovering objects and their location in images," *Proc. IEEE ICCV 2005*, *Vol. 1*, pp. 370-377, Oct. (2005).

[30] J.W.H. Tangelder, R.C.Veltkamp, "A survey of content based 3d shape retrieval methods," *Multimedia Tools and Application*s, vol. 39, pp. 441–471, (2008).

[31] R. Toldo, U. Castellani, A. Fusiello, "Visual vocabulary signature for 3D object retrieval and partial matching," *Proc. EG Workshop on 3D Object Retrieval 2009* (3DOR 2009), pp.21-28, (2009).

[32] Y. Uchida, S. Sakazawa, "Image Retrieval with Fisher Vectors of Binary Features," *Proc. 2nd IAPR Asian Conference on Pattern Recog ition* (*ACPR*) 2013, pp. 23-28, (2014).

[33] R.C. Veltkamp, F.B. ter Harr, "SHREC 2007 3D Shape Retrieval Contest," *Dept. of Info. and Comp. Sci.*, Utrecht University, *Technical Report UU-CS-2007-015*, (2007).

[34] E. Wahl, U. Hillenbrand, G. Hirzinger, "Surflet-Pair- Relation Histograms: A Statistical 3D-Shape Represen- tation for Rapid Classification," *Proc. of the Fourth Intern'l Conf. on 3-D Digital Imaging and Modeling* (*3DIM 2003*), 474-481, Banff, Canada, (2003).