Takahiko Furuya and Ryutarou Ohbuchi, Hashing Cross-Modal Manifold for Scalable Sketch-based 3D Model Retrieval, Proc. Int'l Conf. on 3D Vision (3DV) 2014, Tokyo, Japan, Dec. 8~11, 2014.

Hashing Cross-Modal Manifold for Scalable Sketch-based 3D Model Retrieval

Takahiko Furuya, Ryutarou Ohbuchi
Graduate School of Medicine and Engineering
University of Yamanashi
Yamanashi-ken, Japan
g13dm003@yamanashi.ac.jp, ohbuchi@yamanashi.ac.jp

***Abstract*—This paper proposes a novel sketch-based 3D model retrieval algorithm that is scalable as well as accurate. Accuracy is achieved by a combination of (1) a set of state-of-the-art visual features for comparing sketches and 3D models, and (2) an efficient algorithm to learn data-driven similarity across heterogeneous domains of sketches and 3D models. For the latter, we adopted the algorithm [18] by Furuya et al., which fuses, for more accurate similarity computation, three kinds of similarities, i.e., those among sketches, those among 3D models, and those between sketches and 3D models. While the algorithm by Furuya et al. [18] does improve accuracy, it does not scale. We accelerate, without loss of accuracy, retrieval result ranking stage of [18] by embedding its cross-modal similarity graph into Hamming space. The embedding is performed by a combination of spectral embedding and hashing into compact binary codes. Experiments show that our proposed algorithm is more accurate and much faster than previous sketch-based 3D model retrieval algorithms.**

3D shape retrieval; content-based multimedia retrieval; hashing; manifold learning.

I. INTRODUCTION

An important issue in 3D model retrieval is the modality to present a query. Querying by a 3D model has been the most popular modality so far in the literature. However, in reality, a user probably doesn't have a 3D model similar enough for the shape she/he wanted. A promising alternative is *Sketch-Based 3D Model Retrieval* (*SB3DMR*), an easily accessible query modality that allows for specification of shape. With the ubiquity of tablets and other touch-capable devices, this modality has become an attractive alternative. Accuracy and efficiency of SB3DMR algorithm, however, is still insufficient for practical use. *Shape REtrival Contest* (*SHREC*) series of 3DMR contest has been holding a track of SB3DMR for some years (e.g., [2]). But even the best performing of the entrants in the SHREC 2014 track on SB3DMR logged Mean Average Precision (MAP) around 13%. We must improve retrieval accuracy of SB3DMR systems. Also, we must make them scalable so that a large database of 3D models can be searched comfortably.

A difficulty in SB3DMR is that the data to be compared, sketches and 3D models, have different modalities. The most popular approach so far in SB3DMR is to use 2D image as the common ground, and compare sketches with sketch-like renderings of 3D models ([1][9][12][14]). The rendering of a 3D model is done from multiple view orientations so that an image rendered from one of the view orientations would match the sketch, if in-plane orientation is ignored. Still, the comparison is not easy. Sketches are highly varied in their drawing style and abstraction levels. Sketches and rendered images of 3D models may include noise, such as broken strokes or background clutters. Sketches and/or intended retrieval result may be highly influenced by semantics; for example, retrieving realistic human figure based on a stick-figure can be difficult. There is also an issue of computational efficiency and scalability. There will be a large number of 3D models that needed to be searched. Furthermore, multi-view rendering increases, by a factor of 10 or more, the number of matching necessary to compare a sketch with a 3D model.

This paper proposes a novel SB3DMR algorithm that is accurate, can handle semantic labels, and is efficient enough for a database having 100k or more 3D models. We call the algorithm *Sketch-to-3D-model Cross-Modal Manifold Hashing*, or *S3D-CMMH*. The proposed algorithm employs *Cross-Modal Manifold* (*CMM*) of Furuya, et al [18] that effectively bridges a manifold of sketches and a manifold of 3D models for accurate cross-modal comparisons. For an efficient retrieval ranking using a CMM, the proposed algorithm embeds the CMM into Hamming space. The CMM is first mapped into a low-dimensional real-valued vector space by *Laplacian Eigenmaps* (*LE*) [11]. It is then hashed into a set of compact binary feature vectors in Hamming space by using *Iterative Quantization* (*ITQ*) hashing [21]. To construct CMM effectively and efficiently, we use a state-of-the-art feature aggregation method and a hashing algorithm to produce accurate and compact features for sketches and rendered images of 3D models.

Our experimental evaluation shows that the proposed S3D-CMMH algorithm is more accurate and efficient than previous SB3DMR algorithms we have compared against.

We discuss relevant previous work in the next section. We then describe the proposed algorithm in Section III, followed by experiments and their results in Section IV. We conclude the paper in Section V.

II. RELATED WORK

A. Sketch-based 3D model retrieval

Previous algorithms for SB3DMR compare sketches with sketch-like rendering of 3D models for similarity. 3D models

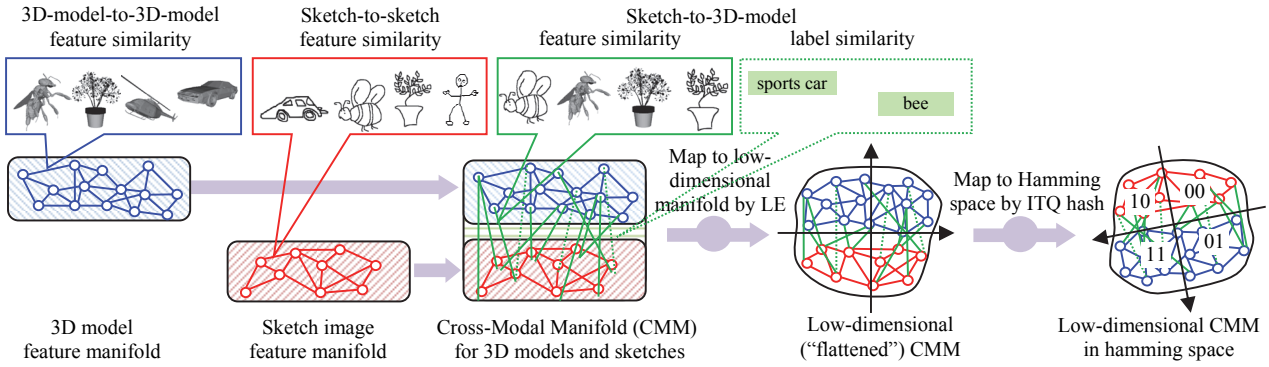


Figure 1. Heterogeneous features for 3D models as well as sketch images are linked into Cross-Modal Manifold (CMM) by their similarities. For retrieval efficiency, CMM is hashed into Hamming space by combining Laplacian Eigenmaps (LE) and Iterative Quantization (ITQ) hashing.

may be rendered in silhouette, occluding contour, suggestive contour [4], and other styles. A sketch image is then compared against rendered images of 3D models. BF-GALIF [12] by Eitz et al. employs Gabor filter-based local feature called GALIF. To compare images, thousands of GALIF feature are extracted and are aggregated into a feature vector per image by using Bag-of-Features (BF) approach (e.g., [8]). Alternatively, one may convert images into other representation, such as a set of line segments [9].

These SB3DMR algorithms, however, may run into trouble if the drawing style of a sketch is different from that of rendered images of 3D models. Or, if the sketch is too abstract, the sketch won't match rendered images of corresponding 3D models. To alleviate these issues, an algorithm may exploit semantic labels. Furuya et al. [17][18] adopts an approach that employs a graph called CMM that connects heterogeneous features of sketches and 3D models by their feature-based similarity and semantic-based (i.e., label-based) similarity for ranking. We will explain CMM in Section II.B as our proposed algorithm is based on CMM. While a SB3DMR system using CMM produced good retrieval accuracy, its high cost for ranking significantly limits CMM's scalability to a large 3D model database. The accuracy reported by Furuya et al [17][18] is also hampered in part by the limited capabilities of the visual features employed.

B. *Cross-modal similarity learning*

Comparison of features having heterogeneous modalities is a difficult problem, especially if semantic influence exists. Previous algorithms for cross-modal feature comparison ([6][15][17][18][20]) employed feature-based similarity, semantic-based (e.g., label-based) similarity, or both. Among these algorithms, [6], [17], [18], and [20] try to capture heterogeneity as well as nonlinearity of data by a graph structure. The graph, which we call CMM, is used to learn similarity metric among data points. For scalability, [6] and [20] perform *k*-means clustering on heterogeneous data sets to approximate a large CMM by a small number of centroids. However, in a large-scale database, small number of centroids fails in approximating the original large CMM graph. Thus retrieval accuracy of these algorithms tend to decrease in a large-scale database. In the field of SB3DMR, the Cross-Domain Manifold Ranking (CDMR) algorithm by Furuya et al. [17][18] employs feature similarity and label-based semantic similarity to form a CMM. To compute ranking of 3D models for a given sketch query, the CDMR algorithm uses Manifold Ranking (MR) by Zhou et al [7]. While producing accurate ranking, the MR algorithm is computationally expensive, especially for a large CMM.

III. Proposed Algorithm

A. *Overview of the algorithm*

Proposed S3D-CMMH algorithm (Fig. 1) combines a novel hashed cross-modal manifold learning algorithm with a set of state-of-the-art visual features designed for SB3DMR. The S3D-CMMH algorithm employs unsupervised learning of three kinds of similarities, that are, among sketches, among 3D models, and between sketches and 3D models, for more accurate similarity ranking. Furthermore, if available, sketch-to-3D-model semantic-label based similarity may also be exploited for semi-supervised learning.

With a large set of sketches and 3D models, the CMM graph becomes quite large, and both its construction and similarity ranking of retrieval results becomes costly. To reduce ranking cost, proposed algorithm first applies spectral embedding a la Laplacian Eigenmaps (LE) [11] to "flatten" a high-dimensional (e.g., 30k dim.) CMM into a low-dimensional (e.g., 1,024 dim.) manifold, then the low dimensional real valued vector is hashed into Hamming space (e.g., 1,024 bit) by Iterative Quantization (ITQ) [21]. In the hamming space, distance between binary features can be computed very quickly for fast retrieval.

We also improve visual feature for sketch-to-3D model comparison. Our proposed visual feature, called *Super Vector-Multi-scale GALIF* or *SV-MGALIF*, is based on the BF-GALIF by Eitz et al. [12]. To improve accuracy, we extract a set of multi-scale local Gabor features and aggregate it by using state-of-the-art Super Vector (SV) coding [19]. To reduce costs of similarity computations for CMM construction, SV-MGALIF is compressed by using Kernel PCA (KPCA) [3] and ITQ hashing.

B. Cross-Modal Manifold Hashing

1) Constructing Cross-Modal Manifold

A CMM is a graph that captures geometry of mutual similarity of multi-modal features. As we have multi-modal feature vectors of sketches and 3D models, CMM is constructed based on mutual similarity, not coordinates, of features. The CMM is represented by a sparse, square matrix $\mathbf{W}_{\mathbf{CMM}}$ of size $(N_s + N_m) \times (N_s + N_m)$, given the number of sketches N_s and the number of 3D models N_m. First N_s rows and N_s columns of the matrix corresponds to features of sketches, and the remaining N_m rows and N_m columns of the matrix corresponds to features of 3D models. Each element of the $\mathbf{W}_{\mathbf{CMM}}$ is one of three similarities, that are, intra-modal sketch-to-sketch similarity (entries of $\mathbf{W}_{SS}$), intra-modal 3D-model-to-3D-model similarity (entries of $\mathbf{W}_{MM}$), and cross-modal similarity between sketches and 3D models (entries of $\mathbf{W}_{SM}$ and $\mathbf{W}_{MS}$). As we assume the similarity between sketches and 3D models are symmetric, $\mathbf{W}_{\mathbf{SM}} = \mathbf{W}_{\mathbf{MS}}^{\mathrm{T}}$. Note that the entries of the matrix $\mathbf{W}_{\mathbf{CMM}}$ is sparse, i.e., most of its elements are zero.

$$\mathbf{W}_{CMM} = \begin{pmatrix} \mathbf{W}_{SS} & \mathbf{W}_{SM} \\ \mathbf{W}_{MS} & \mathbf{W}_{MM} \end{pmatrix} \tag{1}$$

The $\mathbf{W}_{\mathbf{CMM}}$ is formed by connecting k_S sketches and k_M closest 3D models with each sketch or 3D model. It is computationally advantageous for the matrix $\mathbf{W}_{\mathbf{CMM}}$ to be sparse, especially for "flattening" the CMM.

Similarity $W_{SS}(i,j)$ of $\mathbf{W}_{SS}$ is computed from the sketch-to-sketch distance $d_{SS}(i,j)$ and a parameter σ_{SS} using equation (2). Similarity $W_{MM}(i,j)$ of $\mathbf{W}_{MM}$ is computed from 3D-model-to-3D-model distance $d_{MM}(i,j)$ and a parameter σ_{MM} using equation (3).

$$W_{SS}(i,j) = \begin{cases} \exp\left(-d_{SS}(i,j)/\sigma_{SS}\right) & i \neq j \text{ and } j \in kNN(i) \\ 0 & otherwise \end{cases} \tag{2}$$

$$W_{MM}(i,j) = \begin{cases} \exp\left(-d_{MM}(i,j)/\sigma_{MM}\right) & i \neq j \text{ and } j \in kNN(i) \\ 0 & otherwise \end{cases} \tag{3}$$

where $kNN(i)$ is a set of nearest neighbor sketches (3D models) of a sketch (3D model) i. Depending on the availability of label-based supervision, similarity $W_{SM}(i,j) = W_{MS}(j,i)$ of $\mathbf{W}_{SM}$ and $\mathbf{W}_{MS}$ are computed as follows;

Unsupervised mode (feature similarity only):

$$W_{SM}(i,j) = \exp\left(-d_{SM}(i,j)/\sigma_{SM}\right) \tag{4}$$

Semi-supervised mode (feature + label similarity):

$$W_{SM}(i,j) = \begin{cases} 1 & C(i) = C(j) \\ \exp\left(-d_{SM}(i,j)/\sigma_{SM}\right) & otherwise \end{cases} \tag{5}$$

For the unsupervised mode, $W_{SM}(i,j)$ corresponds to the inter-feature similarity between the sketch i and the 3D model j, as defined in (4). For the semi-supervised mode, $W_{SM}(i,j)$ is computed by (5), which reflects both semantic similarity and feature similarity. In the equation (5), $C(i)$ and $C(j)$ denote semantic classes of a sketch i and a 3D model j.

2) Hashing Cross-Modal Manifold for Efficiency

The proposed algorithm embeds CMM into Hamming space for efficient inter-feature similarity computation. To do so, the proposed algorithm first maps a high-dimensional data manifold of CMM by spectral embedding using the LE into a low-dimensional real-valued vector space. The LE tries to find geometry of a low dimensional data manifold from a set of high-dimensional data points by first connecting the data points into a graph $\mathbf{A}$, e.g., by using k-nearest neighbor. In our CMM-based algorithm, CMM graph $\mathbf{W}_{\mathbf{CMM}}$ is the graph $\mathbf{A}$ that captures manifold structure. Original LE then eigendecompose its graph Laplacian $\mathbf{L} = \mathbf{I} - \mathbf{A}$ (where $\mathbf{I}$ is an identity matrix) and uses its p smallest eigenvectors as the set of bases for a p-dimensional manifold on which the data points are supposed to lie. Our algorithm directly eigendecompose $\mathbf{W}_{\mathbf{CMM}}$ and uses p largest eigenvectors associated with it. This corresponds to using p smallest eigenvectors associated with a Laplacian of $\mathbf{W}_{\mathbf{CMM}}$.

Eigendecomposition of a *large* CMM matrix $\mathbf{W}_{\mathbf{CMM}}$ can be quite expensive, both computationally and spatially. For a CMM consisting of N_s sketches and N_m 3D models, temporal and spatial costs of eigendecomposition are $O((N_s+N_m)^3)$ and $O((N_s+N_m)^2)$, respectively. That is, LE does not readily scale with the number of 3D models and sketches. To solve this issue, we employ *Randomized SVD* (*RSVD*) proposed by Halko et al. [13]. The RSVD is an approximate algorithm that makes SVD of a very large matrix practical. RSVD first finds a subspace of the matrix by random projection. Then, the original matrix is compressed to the subspace, and the compressed matrix is used to obtain its low-rank factorization. The factorization is back-projected onto the original space to obtain factorization, or eigenvectors of the original matrix.

To perform RSVD, a "compression matrix" $\mathbf{Q} = \left(\mathbf{S}_{\mathbf{CMM}}\mathbf{S}_{\mathbf{CMM}}^{\mathrm{T}}\right)^{q}\mathbf{S}_{\mathbf{CMM}}\mathbf{\Omega}$ is computed from symmetricized version $\mathbf{S}_{\mathbf{CMM}} = \mathbf{W}_{\mathbf{CMM}}\mathbf{W}_{\mathbf{CMM}}^{\mathrm{T}}$ of the CMM $\mathbf{W}_{\mathbf{CMM}}$. Matrix $\mathbf{\Omega}$ is a random matrix of size $n \times r$, in which $n = N_s + N_m$ and $r < n$. We experimented with the power iteration parameter q, and found that by using the RSVD with the value of q in the range tens to hundreds would yield an eigendecomposition about as accurate as the (classical) SVD applied to the original full-size ($n \times n$) matrix. After the power iteration, Gram-Schmidt orthonormalization is applied on $\mathbf{Q}$. An orthonormalized matrix $\mathbf{Q}$ is then used to compress $\mathbf{S}_{\mathbf{CMM}}$ into a compressed matrix $\mathbf{S}'_{\mathbf{CMM}} = \mathbf{Q}^{\mathrm{T}}\mathbf{S}_{\mathbf{CMM}}\mathbf{Q}$ of size $r \times r$. Eigendecomposition of the compressed matrix $\mathbf{S}'_{\mathbf{CMM}}$ yields a set of eigenvectors $\mathbf{U}'$. Back-projecting $\mathbf{U}'$ by $\mathbf{Q}$ produces a set of (approximate) eigenvectors $\mathbf{U} = \mathbf{Q}\mathbf{U}'$ of the CMM matrix $\mathbf{S}_{\mathbf{CMM}}$.

We use r eigenvectors of dimension n as the set of basis for the low-dimensional manifold having dimension r. Projection of an n-dimensional feature vector onto the r-dimensional set of basis produces a dimension-reduced feature vector. Numerically, computing RSVD is rather tractable; they deal either with an operation with large but sparse matrix or multiplication with a dense yet low-rank matrix. Thus, RSVD can be computed with reasonable costs

on a CPU. For a little bit more speed, we employ a GPU running a sparse linear computation library cuSPARSE, which is available as a part of CUDA, for RSVD.

The ITQ hashing is used to hash the low-dimensional manifold having real-valued vector at its nodes. The ITQ searches for an optimal $r \times r$ rotation matrix $\mathbf{R}$ for data points that minimizes a loss in accuracy in converting real vector $\mathbf{x}$ to binary vector $\mathbf{b} = sgn(\mathbf{xR})$. Once converted to a binary vector, a distance between a sketch and a 3D model can be computed very quickly as a Hamming distance.

C. Similarities for constructing CMM

1) Sketch-to-3D-model similarity

Similarity between a sketch image and a 3D model is computed as a similarity between the sketch image and a set of multi-view sketch-like renderings of the 3D model (Fig. 2). To render the 3D model, we adopt the method in [18]. The 3D model is first normalized for scale and position. It is then rendered from 20 viewpoints placed uniformly in the solid angle into 20 sketch-like images of 256×256 pixels each. We combine silhouette and suggestive contour [4] for the rendering. We then normalize, using the method described in [18], orientations of sketch images and the rendered images of 3D models by using responses of multi-orientation Gabor filters.

The algorithm then extract a set of local features based on the BF-GALIF [12] from the images. A GALIF feature, by using oriented, single-scale Gabor filter, captures orientation of gray scale gradient in the images. BF-GALIF densely samples GALIF feature at grid points, and aggregates these features into a vector per image by BF approach. To improve invariance to scale variation, we extract GALIF at three Gabor filter scales. After the rotation normalization, a set of Gabor filters are applied to each image. With the following 3 sets of parameters and four orientations {0, π/4, π/2, 3π/4} [rad], we have 12 distinct Gabor filters to apply; $(\sigma_x, \sigma_y, \omega) \in \{(2.0, 6.65, 0.065), (4.0, 13.3, 0.13), (8.0, 26.6, 0.26)\}$. After applying the filters, on each image, feature points are sampled densely at square grid points having an interval of 12 pixels. For each feature point, three GALIF features are computed from three square region-of-interests whose sizes are 86, 114 and 142 pixels. Overall, about 4,000 MGALIF features are extracted per image.

Our algorithm aggregates the 4,000 MGALIF features extracted from an image into a feature vector per image by using SV coding [19]. We first reduce dimensionality of MGALIF feature from 64 down to 16 by using PCA. We observed a small improvement in accuracy due to the dimension reduction. Using randomly sampled dimension-reduced 250,000 MGALIF features, a codebook is learned by using Gaussian Mixture Model (GMM) clustering algorithm as with [16]. After SV aggregation, a SV feature has dimension of $k(d+1)$, in which d is the dimensionality of the PCA-reduced MGALIF and k is the number of code words. In this paper, we set d=16 and k=4,000, resulting in 68,000 dimensional SV-MGALIF.

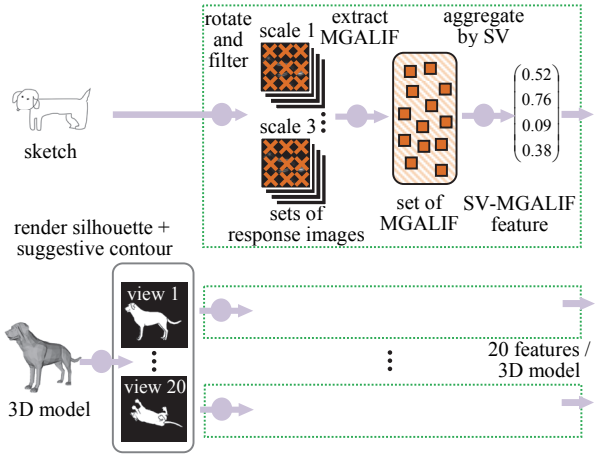


Figure 2. Feature extraction from a sketch and a 3D model.

While SV coded features yield high matching accuracy, their comparison is costly as they are high-dimensional real-valued vectors. We reduce the cost of comparison by converting the SV-MGALIF into binary feature vector called *binarized SV-MGALIF* (*bSV-MGALIF*). The conversion is performed by non-linear dimensionality reduction using KPCA followed by hashing into Hamming space by using ITQ [21]. The KPCA reduces dimensions of SV-MGALIF from 68,000 down to 1,024. It is then hashed into a 1,024 bit binary vector by using the ITQ. To train the KPCA and the ITQ, we use a set of SV-MGALIF features extracted from 5,000 randomly selected sketches. (We tried using rendered images of 3D models for KPCA training, but it resulted in a lower retrieval accuracy.) Rotation matrix for the ITQ is learned with 20 iterations. In preliminary experiments, we found that converting high dimensional SV-MGALIF to compact bSV-MGALIF using the method described doesn't decrease sketch-to-3D model matching accuracy.

Assume that a 3D model is rendered from 20 viewpoints into 20 sketch-like images. Then, a similarity between a sketch image and a 3D model is computed as a maximum similarity among a bSV-MGALIF vector of a sketch image and a set of 20 bSV-MGALIF features of the 20-view rendered images of the 3D model.

2) Sketch-to-sketch similarity

Similarities among sketch images are computed by using SV-MGALIF feature but without rotation normalization and binarization. We didn't use rotation normalization for comparison among sketch images since most of the hand-drawn sketches are drawn in natural "canonical" orientations. We didn't use binarization since sketch-to-sketch comparison is not the computational bottleneck. The number of sketches (e.g., thousands to tens of thousands) is much smaller than the number of multi-view rendered images of 3D models (e.g., tens of thousands to millions).

We use number of clusters k=2,000, and the dimension of PCA-processed MGALIF feature is d=16. Thus, the dimension of SV-MGALIF for sketch-to-sketch comparison

is 34,000. We reduce dimension of SV-MGALIF down to 200 by KPCA and employ Cosine similarity for comparison.

3) 3D-model-to-3D-model similarity

For 3D-model-to-3D-model comparison, we combine two heterogeneous features, SV-DSIFT and LL-MO1SIFT described in [16]. Note, however, that the method used in this paper for combing the two features (vector concatenation) is different from the method described in [16].

SV-DSIFT is invariant to articulation of 3D model. It densely extracts SIFT [5] features from 42 depth-images rendered from 42 viewpoints. About 300 SIFT features per image are extracted, and a 3D model is described by a set of about 13k SIFT features. The set of SIFT features is aggregated by SV coding. As the dimensionality of SIFT is 128 and we set the number of code words (clusters) to 2,500, dimensionality of SV-DSIFT is 322,500. LL-MO1SIFT is sensitive to articulation. For each of 42 viewpoints, the rendered image of the 3D model is in-plane rotated to 16 different orientations. Then, a SIFT feature is extracted from each image without rotation normalization. A set of 672 SIFT features ($672=16 \times 42$) extracted from a 3D model is aggregated by using *Locality-constrained Linear* (*LL*) coding [10] into a feature vector per 3D model. As we set the number of clusters to 9,000, dimensionality of LL-MO1SIFT is 9,000.

Finally, two feature vectors of SV-DSIFT and LL-MO1SIFT are concatenated, resulting in a 331,500 dimensional vector. Dimension of the concatenated vector is reduced to 200 by using KPCA. Comparison among dimension-reduced fused features is done by cosine similarity.

IV. Experiments and Results

We experimentally evaluated the proposed algorithm for SB3DMR, both in terms of effectiveness and efficiency. As numerical indices of retrieval accuracy, we use *Nearest Neighbor* (*NN*) [%] and *Mean Average Precision* (*MAP*) [%] as with [1] and [2].

There are a few notational conventions used. In this section, our proposed S3D-CMMH is denoted by CMMH for saving spaces. A feature set FS $\{f_{SM}, f_{SS}, f_{MM}\}$ consists f_{SM} for comparing sketches with 3D models, f_{SS} for comparing sketches, and f_{MM} for comparing 3D models. We use two feature sets; FS1 = {BF-fGALIF, BF-fGALIF, BF-DSIFT} is the combination of features used in [17] and [18], and FS2 = {bSV-MGALIF, SV-MGALIF, SV-DSIFT+LL-MO1SIFT} is the combination proposed in this paper. The notation CMMH(US) stands for unsupervised CMMH, and CMMH(SS) stands for semi-supervised CMMH.

A. Benchmark databases

We use three benchmarks in the following experiments; the *S-PSB* benchmark [12], the benchmark used for *SHREC 2014 Extended Large Scale Sketch-Based 3D Shape Retrieval* [2] (*SH14*), and artificially inflated SH14X benchmark we have created based on SH14 that contains 100,000 3D models. Fig. 3 shows examples of sketches and 3D models of the S-PSB and SH14 benchmarks.

S-PSB: The S-PSB is partitioned into train set and test set for supervised learning, each of which contains a set of 907 sketch queries and a set of 907 retrieval target 3D models. Each of the train set of sketches and the train set of 3D models is partitioned into corresponding 90 semantic classes. Test set of sketches and test set of 3D models are also partitioned into 92 classes each.

SH14: We use SH14 to evaluate both unsupervised mode and densely-labeled semi-supervised mode of the CMMH. The SH14 benchmark contains 13,680 sketches divided into 171 semantic classes and 8,987 retrieval target 3D models. To evaluate unsupervised CMMH, all the 13,680 sketches and 8,987 3D models are used. To evaluate the semi-supervised mode of the CMMH, the sketches in the SH14 are already partitioned into 8,550 train set of sketches and 5,130 test set of sketches. We thus use these sets for both training and testing. As for the 3D models, we randomly partition ground-truth labeled 8,987 3D models of the SH14 into two disjoint subsets of 4,493 models each. A subset is used for training, and the other subset is used for testing. To obtain a retrieval accuracy figure, 10 retrieval runs using the random partitioning method are performed, and their results are averaged.

SH14X: We want to evaluate scalability of the proposed algorithm on a benchmark much larger than SH14. However, there has been no such benchmark. To create a SH14X benchmark containing 100k 3D models, we artificially created 91,013 "imposter" 3D models by randomly scaling and rotating 3D models in the SH14. The SH14X is thus a union of the 8,987 SH14 models and 91,013 imposter models. The set of 13,680 sketches of the SH14X is the same as the set of sketches for SH14. The SH14X is used for evaluating CMMH in its unsupervised mode. For the experiment, we form a CMM by using 13,680 sketches and 100k 3D models. To evaluate retrieval accuracy, the SH14 test set of 5,130 sketches is used as the set of queries. In computing retrieval accuracy, all the imposters are excluded, and only the set of 3D models that existed in the SH14 is used.

Table I summarizes parameters used for the CMMH(US) algorithm with FS2 feature set. These parameters were determined through preliminary experiments so that retrieval accuracy is the highest among the combinations we tried.

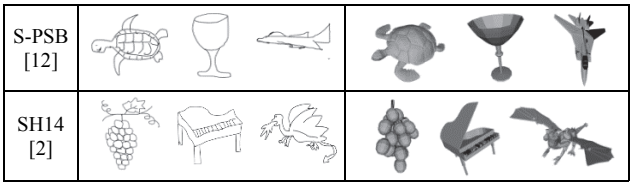


Figure 3. Examples of hand-drawn sketches and 3D models.

TABLE I. Parameters For CMMH (FS2, US)

benchmark	σ_{SS}	σ_{MM}	σ_{SM}	k_S	k_M	q
S-PSB	0.065	0.080	0.160	10	5	150
SH14	0.080	0.200	0.150	10	5	100
SH14X	0.080	0.800	0.150	10	25	150

B. *Similarity ranking on CMM*

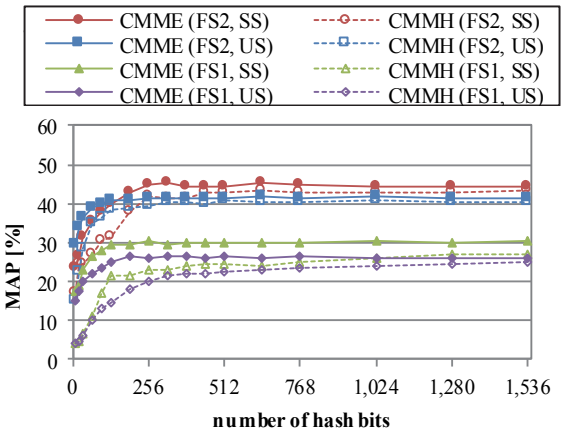
1) *Accuracy*

Fig. 4 plots, for the S-PSB and for the SH14, the impact binary code length has on retrieval accuracy. In Fig. 4, "CMME" indicates accuracies without ITQ hashing, i.e., sketches and 3D models are compared in the LE-projected space by cosine similarity.

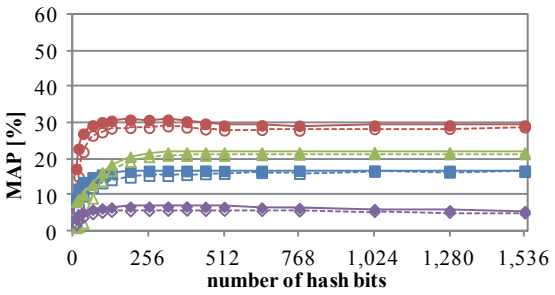
For the two benchmarks, semi-supervised versions (with "SS") of the CMME and the CMMH using the new feature set FS2 showed the highest retrieval accuracies when a relatively long binary code (e.g., 512 bits) is used. With enough bits, loss of retrieval accuracy due to ITQ hashing is contained to within 2~3%. We take this as an acceptable trade-off for speed we gain. These results show that our proposed method, which learns hash functions on the CMM by combining both improved feature set and semantic labels, is effective for SB3DMR. It is also clear, from comparing plots of unsupervised methods (with "US") that the newly proposed feature set FS2 is superior to the conventional feature set FS1 used by Furuya, et al. [17].

2) *Efficiency*

We evaluated spatial and temporal computational costs of the CMMH algorithm. We run the experiment on a PC having a pair of Intel Xeon E5-2650 v2 CPUs, 256GB of RAM, and an Nvidia GeForce GTX 770 GPU having 4GB of memory.



(a) S-PSB.



(b) SH14.

Figure 4: Number of hash bits vs. retrieval accuracy.

Table II shows computation time per query for the CMMH and the CDMR [17]. '*Feat.*' is time needed to extract feature from a sketch query for sketch-to-sketch and sketch-to-3D model comparison. '*Graph*' is time required to embed the query into the CMM graph. Graph embedding is computed by sketch-to-sketch feature similarity and sketch-to-3D model feature similarity to connect the sketch query to its neighboring sketches and 3D models. '*Hash*' is time for hashing the sketch query by using CMMH. '*Dist.*' is time for distance computation between the sketch query and all the 3D models in the database. In this table, the CMMH used the feature set FS2 and the CDMR used the feature set FS1. For the CMMH, all the computational steps are run on CPU. For the CDMR, time for relevance diffusion on the CMM, the '*Dist*' step, is run on the GPU, while the other steps are run on the CPU. As the table shows, our proposed CMMH takes only about 1 second per query for all the benchmarks we run. Even for the SH14X that has 100,000 models, retrieval using CMMH takes only 1.25s. Despite using linear search in Hamming space, the time required for distance computation, '*Dist*', is so small that the algorithm is expected to scale to a much larger database.

Table III summarizes spatial cost for the CMMH algorithm. In Table III, each column indicates memory usage in MBytes for storing each data. Columns "SV-MGALIF" and "bSV-MGALIF", respectively, shows space needed for the sets of dimension-reduced (200 dimensional) SV-MGALIF and 1,024 bit bSV-MGALIF features. The column "CMMH codes" is for storing 512 bit long hashed codes for 3D models, and the column "codebooks & matrices" is for codebooks, projection matrices, and rotation matrices required for feature extraction and hashing of a sketch query. Table III shows that a retrieval using CMMH requires hundreds of MBytes of memory. Memory consumption for storing feature vectors is relatively small since features are compressed by using dimensionality reduction and hashing. On the other hand, storing codebooks and some matrices is costly for CMMH.

Table IV compares hash function learning methods. Hash function is learned by eigendecomposition of the CMM and ITQ in a low-dimensional manifold. Note that hash function learning is pre-processing, i.e., its computational time is not

TABLE II. COMPUTATION TIME PER QUERY [S]

benchmark	method	*Feat.*	*Graph*	*Hash*	*Dist.*	Total
S-PSB	CMMH	1.06	0.04	0.003	0.00002	**1.10**
	CDMR	0.23	0.07		2.42	2.72
SH14	CMMH	1.06	0.16	0.012	0.00014	**1.23**
	CDMR	0.23	2.59		124.73	127.55
SH14X	CMMH	1.06	0.18	0.013	0.00116	**1.25**

TABLE III. MEMORY CONSUMPTION FOR RETRIEVAL USING CMMH ALGORITHM [MBYTES]

bench-mark	SV-MGALIF	bSV-MGALIF	CMMH codes	codebooks & matrices	Total
S-PSB	1.4	4.4	0.1	304.1	310.0
SH14	10.4	21.9	0.6	341.2	374.1
SH14X	10.4	244.1	6.1	519.0	779.6

TABLE IV. COMPARISON OF CMMH LEARNING METHODS

benchmark	learning time [s]		MAP [%]	
	RSVD	SSVD	RSVD	SSVD
S-PSB	110.6	119.0	40.6	40.9
SH14	229.9	3865.9	15.9	15.6
SH14X	437.4		13.1	

included in the retrieval stage. The column 'RSVD' indicates numbers obtained by using the RSVD, in which its power iteration is accelerated by GPU, while eigendecomposition of small matrix is computed on CPU. The column 'SSVD' in Table IV indicates numbers when eigendecomposition of CMM is run entirely on GPU by using a standard SVD algorithm. We couldn't evaluate computational complexity of SSVD for the SH14X as its in-core version of the algorithm didn't fit in the GPU's memory.

As shown in Table IV, learning time for SSVD drastically increases for a larger-scale database. On the other hand, learning by RSVD is much faster than SSVD while keeping MAP scores. For the SH14X which contains 100,000 models, hash functions can be learned in about 7 minutes.

C. Comparison with the other algorithms

Table V shows, for the S-PSB and the SH14 benchmarks, comparison of retrieval accuracy of the proposed CMMH algorithm with other algorithms. CMMH, CDMR, and SCMR-OPHOG employ similarity metric learning. Other algorithms also compare sketch query with rendered images of 3D models by using image features. However, similarity among these features are computed without using any similarity metric learning.

For the benchmarks we tested, accuracy of our proposed CMMH algorithm is higher than the other algorithms. In case of the SH14, our unsupervised CMMH (CMMH(FS2,US)) is more accurate than the most accurate of the algorithms that entered the contest, the SCMR-OPHOG, by Tatsuma et al. [2]. Our algorithm is also more accurate and much faster than the CDMR [17] in both unsupervised and semi-supervised mode. Fig. 5 shows examples of sketch queries and retrieval results that compares our proposed CMMH algorithm with BF-fGALIF [17] and the CDMR [17].

V. CONCLUSION

In this paper, we proposed a Sketch-Based 3D Model Retrieval (SB3DMR) algorithm called *Sketch-to-3D Cross-Modal Manifold Hashing* (*S3D-CMMH*) for accurate and efficient SB3DMR. The proposed algorithm employs a Cross-Modal Manifold (CMM) that bridges a manifold of sketches and a manifold of 3D models for an accurate cross-modal comparison. The CMM is constructed by using three kinds of feature similarity, which are, sketch-to-3D-model, sketch-to-sketch, and 3D model-to-3D-model similarities. If available, semantic label similarity can also be used for the CMM. To significantly speed up retrieval result ranking, a Hamming-space embedding of the CMM graph is learned by a combination of Laplacian Eigenmaps [11] and Iterative Quantization [21] hashing. Given a sketch query, it is hashed into a Hamming space and compared with binary codes of 3D models for efficient ranking. For accurate and efficient CMM construction, we also proposed an improved sketch-to-3D model comparison method that uses a state-of-the-art feature aggregation method and hashing into compact binary code.

TABLE V. COMPARISON OF RETRIEVAL ACCURACY [%] AMONG SB3DMR ALGORITHMS

algorithms	S-PSB		SH14	
	NN	MAP	NN	MAP
BF-fGALIF [17]	29.7	17.4	11.5	4.4
bSV-MGALIF	37.4	22.0	16.1	5.3
CDMR(FS1, US) [17]	32.6	27.9	11.2	9.1
CMMH(FS2, US)	**41.7**	**40.6**	**21.5**	**15.9**
OPHOG [2]			16.0	6.1
SCMR-OPHOG [2]			16.0	13.1
CDMR(FS1, SS) [17]	35.0	31.1	25.9	23.8
CMMH(FS2, SS)	**39.2**	**42.7**	**28.8**	**27.9**

Experimental results showed that S3D-CMMH outperforms state-of-the-art SB3DMR algorithms both in terms of accuracy and speed. With sparsely labeled database, semi-supervised S3D-CMMH yielded even higher accuracy. In addition, S3D-CMMH is efficient both in terms of space and time, allowing it to handle a 3D model database containing 100k or more 3D models at an interactive speed.

ACKNOWLEDGMENT

This research is supported by *JSPS Grant-in-Aid for Scientific Research on Innovative Areas* #26120517 and *JSPS Grants-in-Aid for Scientific Research* (*C*) #26330133.

REFERENCES

[1] B. Li, Y. Lu, A. Godil, T. Schreck, B. Bustos, A. Ferreira, T. Furuya, M.J. Fonseca, H. Johan, T. Matsuda, R. Ohbuchi, P.B. Pascoal, and J.M. Saavedra, A comparison of methods for sketch-based 3D shape retrieval, *Computer Vision and Image Understanding* (*CVIU*), **119**, pp.57–80, 2014.

[2] B. Li, Y. Lu, C. Li, A. Godil, T. Schreck, M. Aono, M. Burtscher, H. Fu, T. Furuya, H. Johan, J. Liu, R. Ohbuchi, A. Tatsuma, C. Zou, SHREC' 14 Track: Extended Large Scale Sketch-Based 3D Shape Retrieval. *Eurographics Workshop on 3D Object Retrieval 2014* (*3DOR 2014*), pp.121–130, 2014.

[3] B. Shölkopf, A. Smola, and K-R. Müller, Nonlinear Component Analysis as a Kernel Eigenvalue Problem, *Nueral Computation*, **10**(5), pp.1299–1319, 2006.

[4] D. DeCarlo, A. Finkelstein, S. Rusinkiewicz, and A. Santella, Suggestive Contours for Conveying Shape, *ACM TOG*, **22**(3), pp.848–855, 2003.

[5] D.G. Lowe, Distinctive Image Features from Scale-Invariant Keypoints, *IJCV*, **60**(2), pp.91–110, 2004.

[6] D. Zhai, H. Chang, Y. Zhen, X. Liu, X. Chen, and W. Gao, Parametric Local Multimodal Hashing for Cross-View Similarity Search, *Proc. IJCAI 2013*, 2013.

[7] D. Zhou, O. Bousquet, T.N. Lal, J. Weston, and B. Schölkopf, Learning with Local and Global Consistency, *Proc. NIPS* 2003, pp.321–328, 2003.

[8] G. Csurka, C.R. Dance, and L. Fan, J. Willamowski, and C. Bray, Visual Categorization with Bags of Keypoints, *Proc. ECCV '04 workshop on Statistical Learning in Computer Vision*, pp.59–74, 2004.

[9] J.M. Saavedra, B. Bustos, M. Scherer, and T. Schreck, STELA: sketch-based 3D model retrieval using a structure-based local approach, *Proc. ICMR 2011*, **26**, pp.1–8, 2011.
[10] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong, Locality-constrained Linear Coding for Image Classification, *Proc. CVPR 2010*, pp.3360–3367, 2010.
[11] M. Belkin and P. Niyogi, Laplacian Eigenmaps for dimensionality reduction and data representation, *Neural Computation*, **15**(6), pp.1373–1396, 2003.
[12] M. Eitz, R. Richter, T. Boubekeur, K. Hildebrand, and M. Alexa, Sketch-Based Shape Retrieval, *ACM TOG*, **31**(4), 31:1–31:10, 2012.
[13] N. Halko, P.G. Martinson, and J.A. Tropp, Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions, *SIAM Review*, **53**(2), pp.217–288, 2011.
[14] S. Kanai, Content-based 3D mesh model retrieval from hand-written sketch, *IJIDeM*, **2**(2), pp.87–98, 2008.
[15] S. Kumar and R. Udupa, Learning Hash Functions for Cross-View Similarity Search, *Proc. IJCAI 2011*, pp.1360–1365, 2011.
[16] T. Furuya and R. Ohbuchi, Fusing Multiple Features for Shape-based 3D Model Retrieval, *Proc. British Machine Vision Conference 2014* (*BMVC 2014*), 2014.
[17] T. Furuya and R. Ohbuchi, Ranking on Cross-Domain Manifold for Sketch-based 3D Model Retrieval*, Proc. CyberWorlds*,pp.274–281, 2013.
[18] T. Furuya and R. Ohbuchi, Similarity Metric Learning for Sketch-based 3D Object Retrieval, *Multimedia Tools and Applications* (*MTAP*), DOI: 10.1007/s11042-014-2171-3, 2014.
[19] X. Zhou, K. Yu, T. Zhang, and T.S. Huang, Image Classification using Super-Vector Coding of Local Image Descriptors, *Proc. 11th ECCV* 2010, pp.141–154, 2010.
[20] X. Zhu, Z. Huang, H.T. Shen, and X. Zhao, Linear cross-modal hashing for efficient multimedia search, *Proc. ACM Multimedia 2013*, pp.143–152, 2013.
[21] Y. Gong and S. Lazebnik, Iterative quantization: A procrustean approach to learning binary codes, *Proc. CVPR* 2011, pp.817–824, 2011.

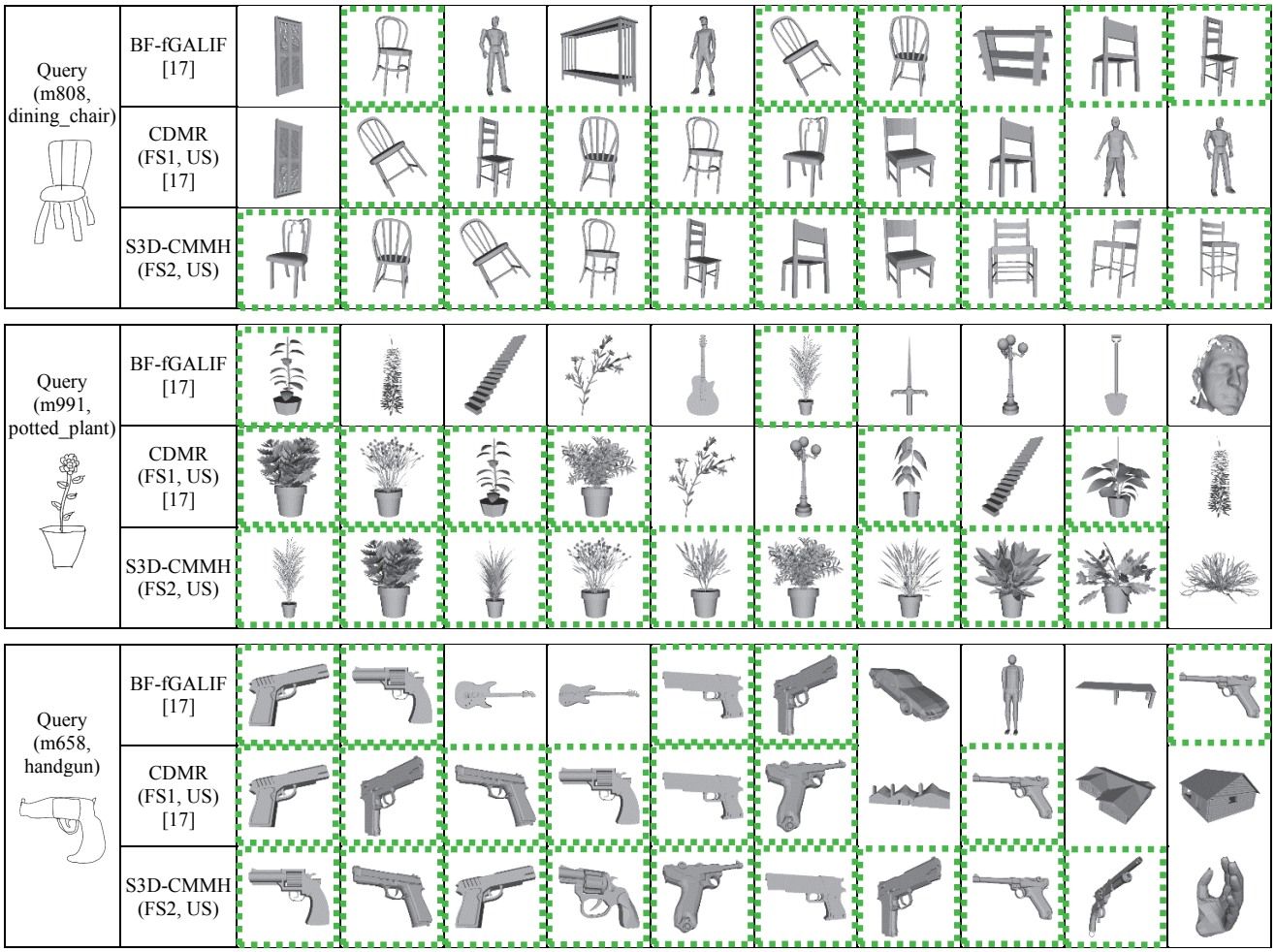


Figure 5. Examples of sketch queries and their retrieval results obtained by using BF-fGALIF [17] (based on [12]), CDMR [17], and our S3D-CMMH for the SPSB benchmark [12]. For all the queries in this figure, our S3D-CMMH produces better results than other algorithms. The S3D-CMMH takes much less time than the CDMR for processing a query.