Ryutarou Ohbuchi, Masaki Tezuka, Takahiko Furuya, Takashi Oyobe, Squeezing Bag-of-Features for Scalable and Semantic 3D Model Retrieval, *Proc. 8th International Workshop on Content-Based Multimedia Indexing* (*CBMI*) *2010*, 23-25 June 2010, Grenoble, France.

Squeezing Bag-of-Features for Scalable and Semantic 3D Model Retrieval

Ryutarou Ohbuchi[1], Masaki Tezuka[2], Takahiko Furuya[3], Takashi Oyobe[4]
Department of Computer Science and Engineering, University of Yamanashi
4-3-11 Takeda, Kofu-shi, Yamanashi-ken, 400-8511, Japan
[1]ohbuchiAT yamanashi.ac.jp, [2]g08mk022AT yamanashi.ac.jp, [3]snc49925AT gmail.com, [4]g09mk005AT yamanashi.ac.jp

Abstract

We have previously proposed a multiple-view, densely-sampled, bag-of-visual features algorithm for shape-based 3D model retrieval [2]. The method achieved good retrieval performance for moderately sized benchmark datasets (~1,000 3D models), including both rigid and articulated 3D shapes. It is also much faster than the other methods having similar retrieval performance. However, the method does not exploit semantic knowledge. We want the retrieval results to reflect multiple (e.g., ~100) semantic classes. Also, if applied to a larger database (e.g., ~1M models), search through the database can be expensive due to its large feature vector size (e.g., 30k dimensions). This paper proposes a method to "squeeze" its length feature vector by projecting it onto a manifold that incorporates multiple semantic classes. Experimental evaluation has shown that retrieval performances equal or better than original feature can be achieved while reducing feature vector size, e.g., from 30k down to less than 100.

1. Introduction

Effective and efficient content-based retrieval of 3D models, especially by their shape, is about to become an important tool in such diverse areas as mechanical design, medical diagnosis, as well as movie, game, and 3D TV content production [6][16].

We have previously proposed a shape-based 3D model retrieval algorithm that performed quite well for both articulated and rigid 3D models [2][12]. An advantage of the algorithm is its ability to accept many and often mutually incompatible 3D shape representations, such as polygon soup and point set, since the algorithm is appearance based. It extracts tens of thousands of local visual features from a set of range images of a 3D model rendered from dozens of viewpoints. Combination of multiple viewpoint rendering and a rotation invariant local image feature enables the retrieval algorithm to achieve 3D rotation invariance. Another advantage is its invariance to articulation and global deformation of 3D shapes. Invariance to articulation has been dealt with by only a small number of algorithms, and for a limited subset of shape representation, such as manifold mesh. Articulation invariance can be important, for example, to retrieve a human figure in different pose. The algorithm integrates tens of thousands of local visual features into a feature vector per 3D model by using *bag-of-features* (*BoF*) approach. The BoF integration of local features gives the algorithm its articulation invariance. The integration also makes comparison among a pair of 3D models much less expensive. Without the integration, comparison among sets, each containing thousands of features, need to be carried out. Experimental evaluation of the algorithm has shown that the method achieves high retrieval performances for several different benchmark databases. For the *Princeton Shape Benchmark* (*PSB*) [14] containing rigid yet diverse 3D models, retrieval performance measured in R-precision is 56%. For the articulated shapes in the *McGill Shape Benchmark* (*MSB*) [20], R-precision of 76% is achieved.

Despite its high retrieval performance, the algorithm left issues to be solved, namely, (1) incorporation of multi-class semantics, and (2) scalability in searching through a database. Incorporation of semantic knowledge from multi-class semantic labels obviously benefits retrieval. In a classical object recognition setting, classifiers such as Support Vector Machine (SVM) are used to recognize a semantic class. The framework has been extended to recognize multiple classes. However, classification is not necessarily desirable for similarity based 3D shape retrieval. We sometimes want objects that are not in the "correct" class but have similar enough shape and/or semantics. Thus, we do not want a cut-off at the classification boundary. A distance metric space we aim for retrieval is a smooth, continuous one that reflects dozens or even hundreds of semantic classes.

The scalability issue arises since the dimension of the feature vector is high; in case of the PSB, for example, optimal feature vector size is ~30k dimensions [2]. Computing 10M distances among 10M pairs of such vectors by using L1-norm or Kullback-Leibler divergence would take significant amount of time. In addition to computational cost, (on-memory) storage cost for a high dimensional feature vector is also an issue, especially if we want to recruit CPU cache memory or Graphics Processing Unit (GPU) for an efficient computation. There are studies (e.g., [7]) that address ccompression of BoF feature vectors, for example, via binary quantization or inverted files. While these approaches aim only for compression, we also aim to incorporate semantic knowledge for an improved retrieval results.

This paper addresses the issues of feature compression and semantic similarity-based retrieval (*not* recognition) by means of supervised feature dimension reduction. Empirical evaluation showed that the dimension of the feature may often be reduced from 30k down to about 50 by semi-supervised dimension reduction, while achieving significantly better retrieval performance.

2. Algorithm

We first outline the 3D model retrieval algorithms, the *Bag-of-Features Salient SIFT* (*BF-SSIFT*) [12] and the *Bag-of-Features Dense SIFT* (*BF-DSIFT*) [2]. A feature vector per 3D produced by either of these algorithms then goes through dimension reduction, whose methods are described in Section 2.2.

2.1. Multi-view bag-of SIFT features

Both BF-SSIFT and BF-DSIFT are based on bag-of visual features. After normalizing the model for its position and scale, a set of depth images of the model is generated by using multiple virtual cameras looking inward at the model. From each depth image, dozens of local visual features are extracted.

In the BF-SSIFT algorithm [12], we used a saliency-based local image feature *Scale Invariant Feature Transform* (*SIFT*) by David Lowe [9] with its interest point detector enabled. Typically, a 3D model is rendered into 42 depth images, each one of which then produces 30 to 50 interest points. Thus, a 3D model is described by a set of 1.5k SIFT features. Directly comparing a pair of such feature sets would be too expensive, especially to search through a 3D model database containing a large number of models. To reduce the cost of model-to-model similarity comparison, the set of thousands of visual features is integrated into a feature vector per model by using BoF approach. The BoF approach vector quantizes, or encodes, the SIFT feature into a representative vector, or "visual word", using a previously learned codebook. The visual words are accumulated into a histogram, which becomes the feature vector.

The BF-DSIFT algorithm employs dense and random sampling of each range image without using interest point detector [2]. The "random" sampling actually employs a prior so that the samples are on or near the 3D models to be compared in the images. With the dense sampling, number of SIFT features per range image increased from a few dozens to a few hundreds. That is, number of SIFT features per 3D model increased about tenfold, from thousands to tens of thousands. These features are again integrated into a feature vector by using BoF approach.

To bring down the query processing time, we employed two accelerations. For SIFT feature extraction, we employed a fast GPU-based implementation of SIFT called *SiftGPU* by Wu [19]. We modified the SiftGPU for dense sampling for the BF-DSIFT. We adopted *Extremely Randomized Clustering Tree*, or *ERC-Tree*, by Guerts, et al [3], for both feature set clustering during codebook learning and for vector quantization (VQ) of SIFT features during retrieval. The ERC-Tree is much faster than the combination of *k*-means clustering and naive linear

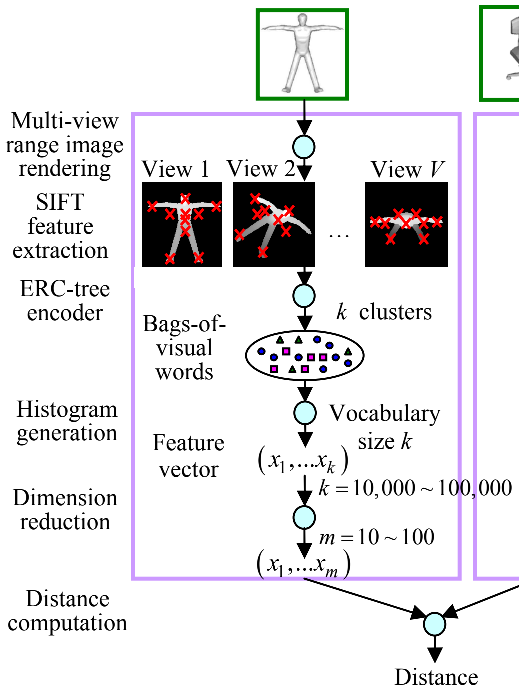


Figure 1. Local visual features from multiple viewpoints are integrated into a vector by using BoF approach. Dimension reduction produces a compact feature vector.

search for the nearest representative vector.

2.2. Dimension Reduction Algorithms

For dimension reduction, we compared three dimension reduction algorithms (Figure 2) below. They are adopted from our earlier work [9].

(1) Unsupervised Dimension Reduction (UDR): Extract a k-dimensional feature from each model in the unlabeled set of 3D models M_{UL} of size p for learning manifold for the UDR. Given the set of p unlabeled features, an UDR algorithm learns the l-dimensional subspace S_{UDR} spanned by the set. The resulting S_{UDR} maps an input k-dimensional feature to the interim, l-dimensional feature in which $l < k$. For the LLE [13], we must approximate the S_{UDR} so that it is defined everywhere in the input space. We used RBF-network algorithm for the approximation, as proposed by He, et al [5].

(2) Supervised Dimension Reduction (SDR): Extract k-dimensional features from the set of labeled 3D models M_{SL} for learning manifold for the SDR. Note that the size q of the M_{SL} is typically much smaller than the size p of the M_{UL}. A SDR algorithm then learns categories from the q labeled features in a batch and encodes the knowledge into the m-dimensional subspace S_{SDR} to be used for later SDRs. The manifold maps k-dimensional input feature onto salient, m-dimensional feature used for retrieval. We used *Supervised Locality Preserving Projections* (*SLPP*) [4] for the SDR.

(3) Semi-Supervised Dimension Reduction (SSDR): Semi-supervised dimension reduction performs the UDR and the SDR in tandem. For each k-dimensional input feature of all the models in the database M_D, employ the UDR and SDR in succession to produce m-dimensional salient feature that incorporates semantic concepts learned from the labeled models in the training set M_{SL}.

For retrieval, store the salient feature together with the corresponding 3D model for later retrieval. As the dimension m of a salient feature is much smaller than the dimension k of the corresponding input feature, cost of distance computation and feature storage are significantly reduced.

For the UDR, we used the *LLE* code available in the Statistical Learning Toolbox for the MatLab. For the LLE, we used the 0/1 weights (as opposed to the heat kernel) on the edges during the mesh construction step. The algorithm employs RBF-network to continuously approximate the discrete map of the LLE. We used the implementation of RBF-network included in the *Neural Network Toolkit* for the MatLab. Parameters of the RBF-network, such as neighbourhood size, are

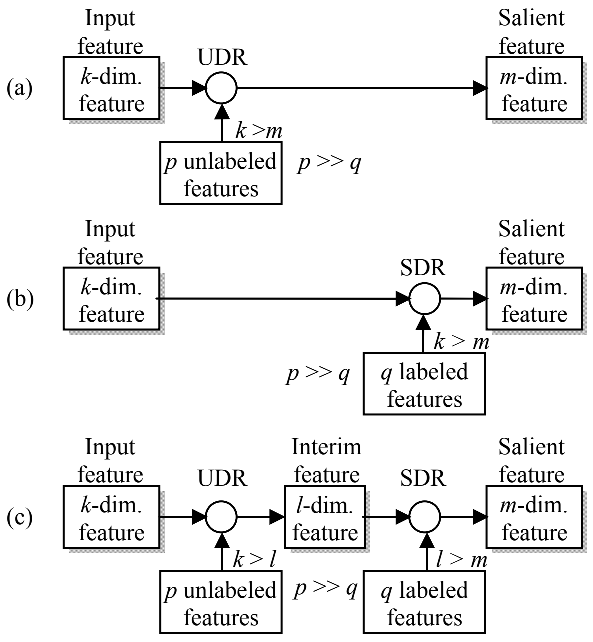


Figure 2. Three dimension reduction approaches used in this paper; *Unsupervised Dimension Reduction* (*UDR*) (a), *Supervised Dimension Reduction* (*SDR*) (b), and *Semi-Supervised Dimension Reduction* (*SSDR*) (c).

found experimentally. We used the code provided by Xaofei He at his web site [4] for the SLPP.

3. Experiments and Results

We evaluate, for UDR, SDR, and SSDR, the relationship between feature dimension reduction and retrieval performance. We also evaluate, for comparison, effectiveness of a simple quantization level reduction of BoF histogram for feature compression.

Evaluation experiments were performed using three benchmarks, the *McGill Shape Benchmark* (MSB) [20] for highly articulated but less detailed shapes, the SHREC 2006 (SHREC) [17] for a set of diverse and detailed shapes, and the SHREC 2007/2008 CAD track (CAD) [10][18] for mechanical parts. Examples of 3D models found in these databases are shown in Figure 3. The MSB consists of 255 models categorized into 10 classes. The MSB include such highly articulated shapes as “humans”, “octopuses”, “snakes”, “pliers”, and “spiders”. The SHREC 2006 uses the 1814 Princeton Shape Benchmark (PSB) [14] models as the target, and has 30 out-of-sample queries. The SHREC 2007 CAD track employs the Engineering Shape Benchmark (ESB)[7], which includes 867 models divided into 45 classes. The SHREC 2007/2008 CAD track uses 45 out-of-sample queries for evaluation. The SHREC benchmark models (that are, those in the PSB) are more diverse in shape and are generally more detailed than those in the CAD and the MSB benchmarks.

The same database is used for VQ codebook training and VQ during retrieval experiments. That is, the VQ codebook generated by using the ESB, for example, is used to query the ESB in the SHREC 2007 CAD track benchmark. We used the training set size $N_t = 50,000$ of SIFT features extracted from multi-view images of models in a database. Size of each range image is 256×256 for both codebook training and for query experiments.

We use *R-precision* as the performance index, which is a ratio, in percentile, of the models retrieved from the desired class C_k (i.e., the same class as the query) in the top R retrievals, in which R is the size of the class $|C_k|$.

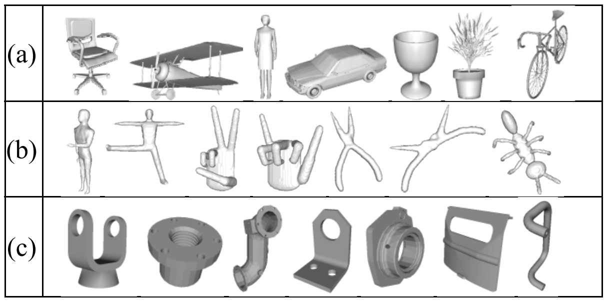


Figure 3. Examples of 3D models from the Princeton Shape Benchmark (a) [12], the McGill Shape Benchmark (b) [20], and the Engineering Shape Benchmark (c) [7][10].

3.1. Feature compression via quantization level reduction

In the first experiment, we evaluated simple feature vector compression via reduction in quantization levels of each bin of the BoF histogram. The method simply clips a value in a histogram bin to the maximum a given number of bits could represent. That is, if 4 bit is allotted to the bin, any counts larger than 15 will be clipped to 2^4=15.

Figure 4 shows the effect of number of bits per bin on retrieval performance (in R-Precision). We used BF-DSIFT feature having 30,125 dimensions for this experiment. To maintain retrieval performance of full quantization levels (8bit/bin), the PSB requires 3 bit/bin, while the MSB requires 2 bit/bin. Higher diversity and/or higher model complexity of the PSB than the MSB could explain this discrepancy.

As the original feature used a 32bit integer for a bin, a feature having 30,125 dimensions required 964 kbits to represent. For the PSB, this simple quantization level reduction produced a feature of size 90 kbits, or about 10 to 1 compression, without losing retrieval performance.

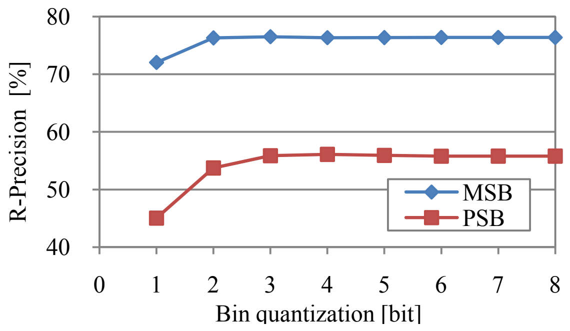


Figure 4. Number of histogram bin quantization bits and retrieval performance for BF-DSIFT.

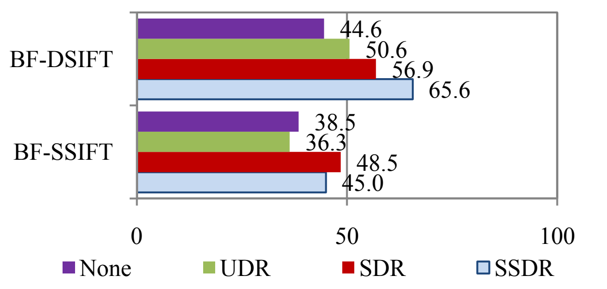
3.2. Feature compression via dimension reduction

Second set of experiments evaluated the effectiveness of the proposed feature compression using manifold-based dimension reduction algorithms. Bar graphs in Figure 5 shows, for both BF-SSIFT and BF-DSIFT, retrieval performances due to three dimension reduction methods for three retrieval benchmarks. Figure 5(a), Figure 5(b), and Figure 5(c) show, respectively, the result obtained by using the SHREC 2006 ("generic" and diverse models), the MSB (articulated models), and the SHREC 2007 CAD track (mechanical parts models). Table 1 shows the same performance figures for three databases, but with dimension of features used. Overall, the dimension reduction is able to compress feature significantly, while at the same time improving retrieval performance. Several observations can be made of these results.

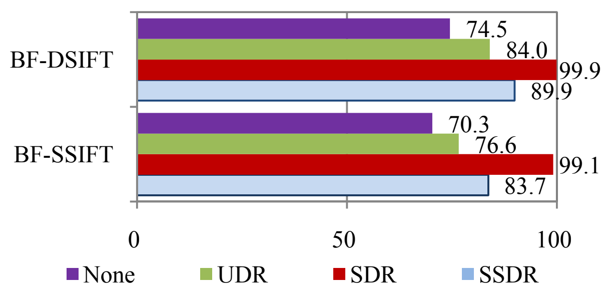
The first observation is that, for most of the cases, dimension reduction resulted in quite significant performance gain.

If supervision is not possible, the UDR can be used to compress feature vector dimension while improving or at least maintaining the retrieval performance. For the MSB, the UDR produced 5 to 10% gain in R-Precision, while decreasing the dimension nearly by about 1/10. For the CAD benchmark and for the SHREC benchmark, small performance loss over the original feature is observed for the BF-SSIFT. For the BF-DSIFT, however, the UDR either improved or maintained the retrieval performance of the original feature vector. The UDR often suffers from insufficient training set. It is very likely that the 867 models of the CAD benchmark is not sufficient for the LLE to learn the structure of feature manifold. The SHREC benchmark has twice as many model, 1,814, as the CAD benchmark. But the SHREC database is more diverse. Thus learning manifold for the SHREC benchmark could have been more difficult than the CAD benchmark. The MSB with it 255 models could also be too small for the LLE, but lack of diversity in the database might have saved the LLE. The MSB appears to lack diversity in visual vocabulary; for the BF-SSIFT, the number of vocabulary, i.e., feature dimension for the MSB is the smallest among the three (See Table 1.)

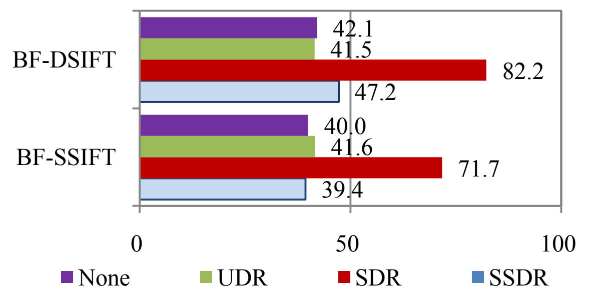
If supervision is allowed, the SDR using SLPP [4] produced very good retrieval performance for the MSB and CAD benchmarks. In case of the MSB, SDR using SLPP produced near perfect retrieval score of R-Precision over 99%. In case of the CAD, R-Precision is 82.2% for the BF-DSIFT processed with the SDR. This figure compares favorably with the best score of the SHREC 2008 CAD track with R-Precision=78.2%. (There was no CAD track for the SHREC 2009.) For the SHREC benchmark, results are a bit complex; the SDR worked the best for the BF-SSIFT, while the



(a) SHREC benchmark.



(b) MSB benchmark.



(c) CAD benchmark.

Figure 5. Dimension reduction methods and retrieval performance for three features applied to three benchmarks. Horizontal axis indicates retrieval performance in R-Precision.

SSDR worked the best for the BF-DSIFT. Simple SDR using SLPP worked for the MSB and the CAD benchmarks as they are "easier" than the SHREC benchmark; the former two have smaller number of ground truth classes and larger number of models per class than the PSB. For the SHREC database having more diverse set of 3D models, and has smaller class (e.g., only 4 models/class), the SSDR appears to work better. Note that the SSDR worked for the BF-DSIFT, but not for BF-SSIFT. A possible explanation for this is the interference of interest point detector in BF-SSIFT with the manifold estimation of the LLE.

Figure 6 shows, for the case of SDR using SLPP, retrieval performance of BF-DSIFT for three databases as a function of compressed feature dimension. A plot with "baseline" shows the performance of the original feature vector. For both MSB and CAD, dimension reduced feature wins over the original everywhere. For the MSB benchmark, the SDR produced very high score regardless of feature dimension. The plot for the CAD benchmark has a clear peak. The plot for the SHREC benchmark has many dips and peaks, a possible indication of its difficult nature.

Table 1. Dimensions of compressed features and retrieval performance.

Data-base	Dimension reduction	BF-SSIFT		BF-DSIFT	
		Dim	R-P	Dim	R-P
SHREC 2006	SSDR	30	45.0	60	65.6
	SDR	140	48.5	110	56.9
	UDR	50	36.3	50	50.0
	None	1500	38.7	32236	44.6
MSB	SSDR	5	83.7	8	89.9
	SDR	10	99.1	10	99.8
	UDR	10	76.6	10	84.0
	None	900	70.3	75581	74.5
SHREC 2007 CAD	SSDR	15	39.4	40	47.2
	SDR	70	71.7	40	82.2
	UDR	25	41.6	100	41.5
	None	1200	40.0	32362	42.1

The second observation is that, as Table 1 shows, very significant reduction in feature size has been achieved. Such reduction should reduce the feature storage feature comparison costs. For example, for the SHREC benchmark, 32,236 dimensions of the BF-DSIFT is reduced to only 50 using the UDR with performance gain of 5%, from 44.6% to 50.0%. When supervision is allowed, the SSDR produced a 60 dimensional feature having a significantly better R-Precision of 65.6%. Note that the dimension reduction produces floating point numbers, so that a 60D feature would consume $60 \times 32 = 1{,}920$ bit. However, this is still significantly smaller than $32{,}236 \times 3 = 96{,}708$ bit of the original, uncompressed feature encoded using reduced quantization levels of 3 bit/bin.

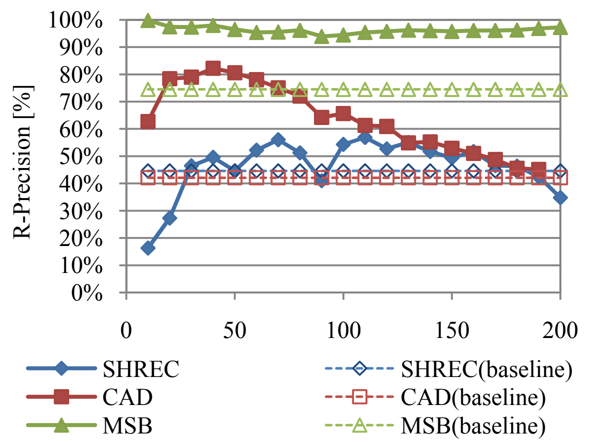


Figure 6. Dimensions after supervised dimension reduction (SDR) using the SLPP [4] and retrieval performance. (BF-DSIFT)

4. Conclusion and Future Work

In this paper, we proposed, in a similarity-based 3D shape retrieval setting, a method to reduce size of feature vector produced by bag-of-features approach while incorporating semantics provided by multiple class labels. We wanted to reduce the feature size for computational efficiency in searching though a large database. We also wanted the reduced feature vector to have better retrieval performance by exploiting semantics. We employed supervised and semi-supervised dimension reduction algorithms for the task.

Empirical evaluation showed that the proposed approach often boosted retrieval performance while compressing the feature very significantly. Reduced size of compressed features would allow efficient us of CPU cache or Graphics Processing Units (GPU) memory. The reduced feature also showed significantly improved retrieval performance.

A weakness in the proposed algorithm is the scalability (or lack of thereof) of learning algorithms for dimension reduction. The dimension of original feature and/or the number of training samples determine space and time complexity of these algorithms. We are currently experimenting with methods to make these learning algorithms more scalable.

References

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

[2] T. Furuya, R. Ohbuchi, Dense sampling and fast encoding for 3D model retrieval using bag-of-visual features, *Proc. ACM CIVR 2009*, (2009).

[3] P. Guerts, D. Ernst, L. Wehenkel, Extremely randomized trees, Machine Learning, 2006, **36**(1), 3-42, (2006)

[4] X. He, P. Niyogi, Locality Preserving Projections, *Advances in Neural Information Processing Systems*, **16**, Vancouver, Canada, (2003). http://people.cs.uchicago.edu/~xiaofei/LPP.html

[5] X. He, W-Y. Ma, H-J. Zhang, Learning an Image Manifold for Retrieval, *Proc. ACM Multimedia 2004*, 17-23 (2004)

[6] M. Iyer, S. Jayanti, K. Lou, Y. Kalyanaraman, K. Ramani, Three Dimensional Shape Searching: State-of-the-art Review and Future Trends, *Computer Aided Design*, **5**(15), 509-530, (2005).

[7] S. Jayanti, Y. Kalyanaraman, N. Iyer, K. Ramani, Developing An Engineering Shape Benchmark For CAD Models, *Computer-Aided Design*, **38**(9), pp. 939-953, (2006).

[8] H. Jégou, M. Douze, C. Schmid, Packing Bag-of-features, *Int'l Conf. on Computer Vision 2009*, (2009).

[9] D.G. Lowe, Distinctive Image Features from Scale—Invariant Keypoints, Int'l Journal of Computer Vision, **60**(2), (2004)

[10] R. Muthuganapathy, K. Ramani, Shape REtrieval contest 2008: CAD models, *Proc. SMI 2008*, 221-222 (2008)

[11] R. Ohbuchi, A. Yamamoto, J. Kobayashi, Learning semantic categories for 3D model retrieval, *Proc. ACM MIR 2007*, 31-40, (2007).

[12] R. Ohbuchi, K. Osada, T. Furuya, T. Banno, Salient local visual features for shape-based 3D model retrieval, *Proc. SMI '08*, 93-102, (2008).

[13] S.T. Roweis, L.K. Saul, Nonlinear Dimensionality Reduction by Locally Linear Embedding, *Science*, **290**(5500), 2323-2326, (2000)

[14] P. Shilane, P. Min, M. Kazhdan, T. Funkhouser, The Princeton Shape Benchmark, *Proc. SMI '04*, 167-178, (2004). http://shape.cs.princeton.edu/search.html

[15] J. Sivic, A. Zisserman, Video Google: A text retrieval approach to object matching in Videos, Proc. ICCV 2003, Vol. 2, 1470-1477, (2003).

[16] J.W.H. Tangelder, R. C. Veltkamp: A survey of content based 3D shape retrieval methods. Multimedia Tools Appl. **39**(3): 441-471 (2008)

[17] R. C. Veltkamp, et al., SHREC2006 3D Shape Retrieval Contest, *Dept of Info and Comp. Sci., Utrecht University, Technical Report* UU-CS-2006-030. (2006)

[18] 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)

[19] C. Wu, SiftGPU: A GPU Implementation of David Lowe's Scale Invariant Feature Transform (SIFT) , http://cs.unc.edu/~ccwu/siftgpu/

[20] Zhang, J., Kaplow, R., Chen, R., Siddiqi, K., The McGill Shape Benchmark (2005). http://www.cim.mcgill.ca/~shape/benchMark//