Local Feature Hashing for Face Recognition

Zhihong Zeng, Tianhong Fang, Shishir Shah and Ioannis A. Kakadiaris

***Abstract*— In this paper, we present Local Feature Hashing (LFH), a novel approach for face recognition. Focusing on the scalability of face recognition systems, we build our LFH algorithm on the p-stable distribution Locality-Sensitive Hashing (pLSH) scheme that projects a set of local features representing a query image to an ID histogram where the maximum bin is regarded as the recognized ID. Our extensive experiments on two publicly available databases demonstrate the advantages of our LFH method, including: i) significant computational improvement over naive search; ii) hashing in high-dimensional Euclidean space without embedding; and iii) robustness to pose, facial expression, illumination and partial occlusion.**

I. INTRODUCTION

Face recognition is one of the most widely researched topics in computer vision [26], [3], [23] due to a large variety of applications that require identity management. In contrast to existing efforts in face recognition, this paper is focused on the scalability of face recognition systems and local feature application, specifically, how to build a local feature data structure from a face gallery that can be used to efficiently find the exemplar that is closest to the query.

Recently, local feature analysis has attracted increasing attention from researchers in computer vision, as it provides a powerful way to describe images of objects and scene with the potential to handle affine transformation, deformation, noise and other image conditions ([16], [24]). However, since the number of the local features in an image is usually large and they reside in a high-dimensional space, efficiency of local feature matching becomes an important issue, especially with a large-size gallery.

Several solutions have been proposed for searching local feature matching pairs, including various tree (e.g., [22], [17]) and hash table structures (e.g., [8], [21]). These methods are significantly superior in terms of computational complexity to a naive linear scan where the best match is searched exhaustively.

In this paper we explore the use of a recent hashing technique, the p-stable distribution Locality-Sensitive Hashing (pLSH) scheme [4], which is an approximate nearest neighbor search technique to overcome the query time bottleneck problem. We propose the Local Feature Hashing method (LFH) based on the Scale Invariant Feature Transform descriptors (SIFT) [13] and pLSH, and apply it for face recognition under a less constrained setting (including variation of pose, illumination, facial expression, and partial occlusion).

All authors are with Departments of Computer Science, Electrical and Computer Engineering and Biomedical Engineering, University of Houston, USA. Their emails are {zzeng2, tfang2, sshah, IKakadia}@central.uh.edu.
This work was supported in part by ARL award DWAM80750.

The advantages of our proposed LFH method include: i) significant computational improvement over naive search; ii) hashing ability in high-dimensional Euclidean space without embedding; and iii) robustness to pose, facial expression, illumination and partial occlusion.

The rest of this paper is organized as follows. In the next section, we present related work. In Section III, our approach is presented. We present our experiments on two publicly available challenging databases (FRGC v2 and Multi-PIE) to assess the performance of our algorithm in Section IV. We offer our conclusions in Section V.

II. RELATED WORK

Significant progress has been made in recent years towards automatic face recognition ([26], [3], [23]). However, most existing methods are based on face images under controlled imaging conditions and small-size gallery.

Local-feature-based or patch-based methods have been shown to provide solutions robust to illumination change, facial expression, pose, and other imaging conditions (e.g., [15], [2], [14], [1]). Although local feature methods offer good performance, few or none of these methods address the problem of how to scale to index very large databases. Many existing methods use a naive linear scan mode where a query input of a face image represented by local feature descriptors or parts is matched against all stored exemplars. Some methods focus on face verification where the query image is only compared with one or a few gallery images of the claimed subject (e.g., [14]).

The problem of major importance to these local-feature-based methods is the nearest neighbor searching problem (i.e., how to build a data structure from a face gallery that can be used to efficiently find the exemplar that is closest to the query). For an overview of multidimensional data structures, please see Samet [20]. The nearest neighbor problem for high-dimensional data (e.g., 128D in SIFT descriptors here) remains challenging, since the required space and query time is exponential in the dimension d. Recently, approximate nearest neighbor search methods have been introduced to overcome the query time bottleneck. These approximation algorithms can be used to efficiently obtain a set of nearest neighbor candidates which can be further used to find the closest neighbor. In many applications, a set of the nearest neighbor candidates are better than one exact nearest neighbor due to nonperfect distance measurements and signal noise.

We have recently witnessed the application of the above-mentioned techniques in image retrieval, object recognition, categorization and segmentation. For example, Grauman and

978-1-4244-5020-6/09/$25.00 ©2009 IEEE

Darrell [8][7] used an efficient embedding function and Locality-Sensitive Hashing (LSH) table for indexing sets of feature vectors under partial correspondences. Shakhnarovich *et al.* [21] present a variant of LSH that retrieves examples similar in the body pose space by local feature matching. Silpa-Anan and Hartley [22] used multiple randomized KD-trees in order to improve tree-based local feature searching.

These approaches share our goal of efficient image-based search by local feature representation. Specifically, the closely related studies are [8] and [21]. They are based on the original LSH techniques [11][6] which require embedding in order to map l_2 space to l_1 space and then to Hamming space. These embeddings increase the query time and complicate the algorithm. Unlike [8] and [21], we apply the extended LSH technique, p-stable distribution LSH (pLSH) [4], which is able to work directly in p-norm space ($p \in (0,2]$). Our algorithm has the theoretical advantages illustrated in [4] over the studies in [21][8], namely, better query time exponent, direct hashing in Euclidean space without embeddings and easier implementation. The pLSH technique has attracted increasing attention from researchers in super-resolution analysis [25] and subspace analysis [19] [5].

Our contribution is combining local feature and pLSH techniques and applying this combination (we term this technique local feature hashing (LFH)) for face recognition which demands an efficient method to handle large-scale face galleries and variations in face pose, illumination, facial expression and partial occlusion. For the local feature, we choose to use the SIFT feature as the starting point because the literature (e.g., [16]) indicates it has the best results as a distribution-based descriptor. In our implementation, we use multiple randomized pLSH tables while Grauman and Darrell in [8] use only one LSH table.

III. APPROACH

A. Framework

We propose a local feature hashing (LFH) method based on the local feature (SIFT) descriptor and pLSH indexing for 2D face recognition. The framework is illustrated in Figure 1.

The framework includes offline preprocessing and online identification. In the preprocessing step, the gallery images are processed for local feature extraction where the SIFT algorithm [13] is used to obtain stable affine-invariant local features and descriptors. We obtain a local feature gallery where each gallery image is represented by a set of local feature descriptors with ID information. We next apply a subspace technique, Principal Component Analysis (PCA), on the high-dimensional descriptors to further reduce the space and computational cost of the successive process. Based on the pLSH scheme, we use these feature vectors in PCA space to build a set of randomized hash tables that map the feature vectors to a set of buckets where the feature vectors that are close (using Euclidian distance) in high dimensional feature space have a much higher probability of colliding in the buckets than do the feature vectors that are far apart. In order to reduce the time of pLSH indexing,

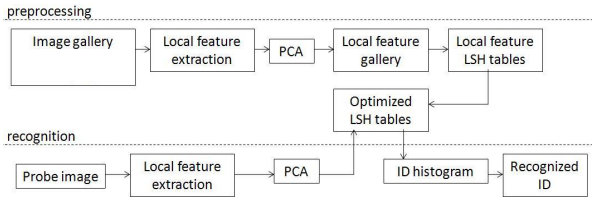

Fig. 1. Depiction of the framework of our approach.

we optimize the pLSH tables by selection of pLSH tables which have relatively uniform indexing distributions.

In the recognition step, given a probe image, we first detect a set of local features and obtain the corresponding descriptors, which are then projected to the PCA representation feature space based on the pre-computed eigenvectors. Through optimized pLSH tables, we hash each feature vector to the buckets that correspond to a set of the gallery feature vectors which are the candidates of nearest neighbors (NN) to the query feature vector. Each NN candidate will produce a set of weighted votes on an ID histogram where the weight is inversely proportional to the distance between it and the query vector. We accumulate all weighted votes in the ID histogram from this set of local feature vectors representing the query image, and the ID with the highest number of votes is the recognized result.

The LFH is a general framework, and its extension to other image search applications is straightforward.

B. Offline preprocessing

1) Local feature detection and PCA representation: We use the SIFT descriptor [13] which has been identified in an evaluation study [16] to generate the most reliable features under common image deformations. However, the framework we propose here is not restricted to SIFT, and other features listed in [16][24] may be applied.

SIFT is a local descriptor of an image which is calculated in two steps: first, difference-of-Gaussian (DoG) filters are used to localize key points with the local scale-space maxima of DoG, and next, each of these key points is used to generate a descriptor represented by a 3D histogram of gradient locations and orientations. This method transforms an image into a set of local descriptors, each of which is robust to image translation, scaling, and rotation and is to some degree invariant to illumination changes and 3D projection due to the quantization of gradient locations and orientations.

Although we can assume that the descriptors of a visually-similar feature from two different images are close in Euclidean distance (l_2 norm, the SIFT matching metric), we examine whether we can build a hash table where they collide with high probability. In addition, the descriptors of visually-dissimilar features (far apart by l_2 distance) should collide in the hash table with low probability. The problem becomes more serious when the input vector is of high dimension rather than of low dimension (called "the curse of dimensionality").

We present two solutions to address this problem. One is the dimensionality reduction of feature descriptors, and the other is to apply the pLSH (described in Section III-B.2). Although the pLSH scheme was proposed to handle high-dimension vectors, the representation in the low dimensional space clearly results in significant space and speed benefits.

The standard SIFT representation (i.e., descriptor) consists of 128-dimensional vectors. We apply Principal Components Analysis (PCA) to the feature descriptors to obtain low-dimensional vectors. We term our method SIFT_PCA. Note the difference between SIFT_PCA here and PCA_SIFT in [12]. In PCA_SIFT, PCA is applied to the normalized gradient image vector at the stage before the computation of descriptors. Our method is to apply PCA directly on the 128-dimensional SIFT descriptor to obtain low-dimensional descriptors. Our idea is straightforward and is applicable to other descriptors.

2) p-stable distribution Locality Sensitive Hashing scheme (pLSH): The second solution we use to address the problem of hashing of high dimensional vectors is the p-stable distribution Locality Sensitive Hashing (pLSH) scheme, an important technique [4] which is an approximate high-dimensional similarity search scheme with provably sublinear dependence on the data size. Locality-sensitive Hashing (LSH) was introduced to solve the (R,c)-Near Neighbor (NN) problem [11][6].

The fundamental idea of LSH is the following: first, build a family of hash functions such that for each function, the probability of collision is much higher for the high-dimensional points which are close to each other than it is for those which are far apart; next, given a query point, find the nearest neighbors by hashing this point using several hashing function from the built family, and retrieving elements stored in buckets indexed by the query point. The advantages of the pLSH illustrated in [4] include: i) it can achieve a large speedup over several tree-based data structures (e.g., KD-trees); and ii) based on the hashing scheme, it can be extended to a dynamic setting (insertion and deletion) without the complexity of tree structures. The original LSH introduced by [11][6] was intended for Hamming space. In order to handle l_2 space, the original LSH technique requires steps to embed l_2 space into l_1 space, and then into Hamming space.

Recently, Datar *et al.* [4] introduced an extended LSH which works for any l_p norm ($p \in (0,2]$). In this paper, we apply this extended LSH for local feature matching and automatically gain its benefits: simple and easy to implement, working in Euclidean distance space without the embedding needed for l_2-norm space in the original LSH technique, and a better query time exponent than that of the original LSH technique.

The following notation pertains to the extended LSH based on a p-stable distribution (for details, see [4]) and to the processing steps in our study.

We denote by l_p^d the space R^d with the norm l_p (here we use Euclidean norm l_2), and $M = (X,D)$ the metric space with the point $v \in X$ and the distance D. $B(v,r) = \{q \in X | D(v,q) \leq r\}$ is the ball of radius r centered at v.

> DEFINITION: For a domain S of the point set with D, a LSH family $H = \{h : S \rightarrow U\}$ is called (r, cr, p_1, p_2)-sensitive for D if for any $v,q \in S$
> - if $v \in B(q,r)$ then $P_r[h(q) = h(v)] \geq p_1$
> - if $v \notin B(q,cr)$ then $P_r[h(q) = h(v)] \leq p_2$
>
> where $p_1 \geq p_2$ and $c = 1+\varepsilon$.

We build our randomized local feature hash tables as follows. First, we build a (R, cR, p_1, p_2)-sensitive family based on a 2-stable distribution, $H = \{h_1, h_2, \cdots, h_n\}$, where each hash function is $h_j(v) = \lfloor \frac{a \cdot v + b}{r} \rfloor$, $j = 1, \cdots, n$ and v is a feature point. The parameter a is a d-dimensional vector with entries chosen independently from a stable distribution. In this paper we choose a normal distribution which is 2-stable. The parameter b is a real number chosen uniformly from the range $[0,r]$. In other words, the high-dimension vector v is first projected onto a real line which is quantized into equi-width segments of appropriate size r. The vector v is assigned as a hash value based on which segment it projects onto. This hashing function has been proven to be locality-preserving in the sense described above.

Second, in order to amplify the gap between the high probability p_1 and low probability p_2, we concatenate several functions from the pLSH family. We define a new family $G = \{g_1, g_2, \cdots, g_L\}$, each of which is obtained by concatenating k randomly chosen hash functions from H (e.g., $g_i(v) = (h_{i1}(v), h_{i2}(v), \cdots, h_{ik}(v))$, $i = 1, \cdots, L$).

The pLSH algorithm has three main parameters: the width k, the number of hash tables L, and r which is related to the quantization of the projected line in h. For a fixed approximation ratio $c = 1+\varepsilon$ and probabilities p_1 and p_2, set $k = \frac{\log n}{\log 1/p_2}$ and $L = n^{\rho}$, where $\rho = \frac{\log p_1}{\log p_2}$. We can empirically set k, L and r directly as shown in Section IV.

The following performance is then guaranteed:

- preprocessing time: $O(nLkt)$, where t is the time to compute h given an gallery point v,
- space requirement: $O(nL)$, plus the space for storing feature points in the gallery,
- query computational complexity: $O(L(kt + dnp_2^k))$, and
- the algorithm succeeds in finding a point with distance cR from the query point with probability at least $\Omega(\min\{1, Lp_1^k\})$.

Given local feature vectors from the gallery, we can build a set of randomized pLSH tables where a query feature vector can be hashed to a set of buckets by which we can find the nearest neighbor candidates with the corresponding ID. The framework of building pLSH tables is illustrated in Fig. 2.

One important step in our pLSH table building is the optimization of pLSH tables. As in all hashing techniques, the query time of our algorithm depends on the pLSH table occupancy and index distribution. If the size of the pLSH table is of the order of the feature points in the gallery

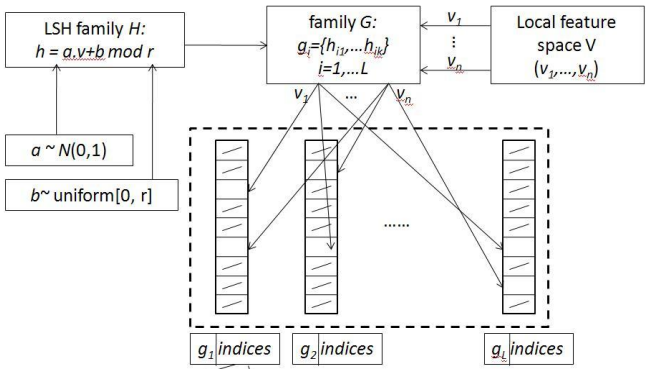

Fig. 2. Building Local feature pLSH.

and the distribution is uniform, the access complexity will be equal to $O(1)$. If, on the other hand, the pLSH table is small or all of the feature points are hashed into only a few buckets, the access time of pLSH table may be dominated by the number of feature points in the table. The nonuniform occupancy of the pLSH hash results in buckets with a large number of entries. To ensure this condition, a desirable pLSH table should have a uniform distribution of the entries. Thus, we apply an optimization step: to select the pLSH tables described above that have the flat index distribution. In this paper we choose the pLSH tables having a small average number of entries for buckets. This step significantly reduces the query time in our experiments.

The offline enrollment process is shown in Algorithm 1.

C. Identification

The identification is illustrated in Algorithm 2.

Algorithm 1 Offline enrollment

Given: An image gallery where each subject is represented by one or multiple images $\{X_1, \cdots, X_N\}$:

1. Apply the SIFT algorithm to each image $X_i (i = 1$ to $N)$ to obtain a set of SIFT descriptors $\{v_j, j = 1, \cdots, m_j | v_j \in R^d\}$.
2. Compute the eigenspace to represent SIFT descriptors.
3. Use all gallery feature points to build a set of randomized pLSH tables. For each feature point v_j,
   - Project it to a set of real lines which are quantized into equal-width segments, according to $h(v_j) = \left\lfloor \frac{a \cdot v_j + b}{r} \right\rfloor$,
   - Concatenate k bits to form hash buckets: $g_i(v_j) = (h_{i1}(v_j), h_{i2}(v_j), \cdots, h_k(v))$, $i = 1, \cdots, M$, and
   - Store the indices of elements of the buckets.
4. Optimize the pLSH tables by selecting L tables with flat index distribution.

Algorithm 2 Identification

Given: A probe face image represented by a set of local feature vectors :

1. Detect SIFT features and obtain a set of SIFT descriptors $Q = \{q_1, \cdots, q_n\}$.
2. Use the pre-computed eigenspace to project SIFT features from 128 dimensional space to a low-dimensional space.
3. Compute hash keys for each feature: $g_j(q_i) = (h_1(q_i), h_2(q_i), \cdots, h_k(q_i))$, $i = 1, \cdots, n$.
4. Query hash buckets in the optimized pLSH tables to obtain m NN candidates.
5. Compute the l_2 distance between these NN candidates and the query point, and sort these distances.
6. Use the s $(s \leq m)$ NN candidates with the smallest distances to give the weighted votes in the ID histogram. The weight is inversely proportional to their normalized distances (normalized by the sum of distances of the s NN candidates).
7. Accumulate the votes in the ID histogram from all feature points representing the query image. The ID with the largest number of votes is the recognized result.

IV. EXPERIMENTS

We evaluated our LFH algorithm on two publicly available databases: FRGC v2 [18] and the Multi-PIE database [9]. The experiments on the FRGC v2 database include the comparison between our LFH and naive exhaustive search, the effect of parameters on the performance, and partial occlusion performance. The experiments on the Multi-PIE database include the comparison beteen our LFH and the PittPatt [10], a commercial face recognition system with different poses, expressions and illumination conditions. We selected this popular commercial 2D face recognition system as the baseline to ensure that we have a strong and realistic comparison with the-state-of-the-art methods.

A. FRGC v2

From FRGC v2 database we select a cohort of 248 subjects with multiple races, genders and ages. These subjects attended image collections in both Fall 2003 and Spring 2004 sessions. We use one image per subject from the Fall 2003 session (total 248 images which produced 121,018 local feature vectors) as gallery data and one image per subject (total 248 images) from the Spring 2004 session as probe data.

Computation comparisons between naive exhaustive search and our LFH method are illustrated in Table I and Table II. Table I shows, for one query feature vector, the average nearest neighbor searching time (in seconds) in the gallery of 121,018 feature vectors. The computational time is measured on a computer with 1.8GHz CPU and 2GB RAM, and MATLAB implementation. The naive exhaustive search method requires 121018 comparisons (distance computations) between the query point and gallery points to

find the NN, while LFH method only requires comparison with the feature candidates that collide in the buckets in the pLSH tables with the query feature. Table II presents the comparison percentage and failure percentage of LFH hashing. The comparison percentage is the ratio of examined features and total features in the gallery after LFH indexing. The failure percentage is used to measure the LFH error rate of NN searching, which is the ratio of occurrences of failure to find NN and total searching occurrences. The ground truth of a probe's nearest neighbors is obtained by exhaustive search. These results are based on the parameters $k = 30$ and $L = 20$.

The results in Table I and Table II demonstrate that the LFH algorithm based on pLSH can dramatically decrease the number of comparisons (distance computations) and the NN searching time to find the nearest neighbor with an acceptable failure percentage.

We also notice that the NN searching time per feature of LFH on the original SIFT shown in Table I is close to the search time of LFH with SIFT_PCA. It is also shown in Fig. 4 where the image query time of SIFT and SIFT_PCA with $k = 30$ and $L = 20$ are very close to each other. The reason behind this phenomenon is the following: although the dimension reduction of SIFT_PCA reduces the distance computation, it takes additional computation to project the SIFT descriptors to the PCA subspace. However, under other parameter setting ($k = 10$ and 20) shown in Fig. 4, the SIFT_PCA processing can significantly reduce the query time. More details of the effect of parameters on the performance will be discussed below.

Figures 3, 4, 5 and 6 show the influence of parameters (k, L and the number of principal components) on the rank-1 identification rate and average face image query time of LFH methods with SIFT and SIFT_PCA descriptors. Fig. 3 shows the results of rank-1 identification rate with k and L on FRGC v2 by using SIFT and SIFT_PCA methods. The corresponding query time for face identification is shown in Figure 4. We chose from 100 randomized pLSH tables, L pLSH tables which have relative flat indexing distribution. These experiments suggest that using multiple pLSH tables has a significant advantage of accuracy over using one single pLSH table, although the computational cost increases with the number of tables. Fig. 3 shows that $10 \leq L \leq 20$ is a very good choice with over 90% rank 1 accuracy. For $L > 10$, the rank-1 rate increases slightly but the computational cost increases nearly linearly. The choice of different k has small impact on the identification rate but large impact on the computation cost. Another parameter in our LFH method is r which defines the segments of the projected line described in Section III-B.2. The influence of r on the LFH performance is nearly the same as k. We set $r = 10$ segments in the projected line. In addition, compared with the LFH algorithm based on traditional 128D SIFT descriptor, the LFH algorithm based on SIFT_PCA has little loss of identification rate but gains significant computation and storage benefit. Thus, the LFH method based on SIFT_PCA is a better solution than that based on the traditional SIFT descriptor in terms of computational complexity and storage space.

The identification rates and query times for different numbers of Principal Components (PC) for SIFT_PCA are shown in Fig. 5 and 6. These figures indicate that the identification rate increases with the number of PCs until 40 PCs. The query time increases with the number of PCs. Thus, we use 40 PCs in our experiments.

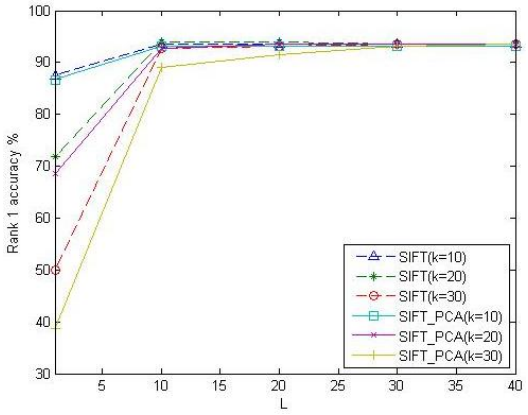

Fig. 3. The rank-1 identification rate of SIFT and SIFT_PCA methods on FRGC v2 with k and L, using 40 principle components.

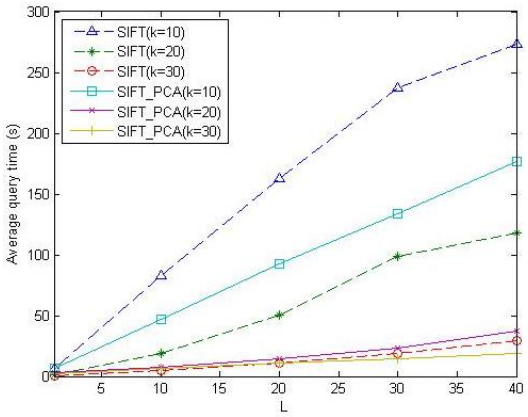

Fig. 4. The query time of SIFT and SIFT_PCA methods on FRGC v2 with k and L, using 40 principle components.

TABLE I

NN SEARCH TIME (IN SECONDS) OF NAIVE SEARCH AND LFH INDEXING PER FEATURE. PCS INDICATES PRINCIPAL COMPONENTS

Time (s)	SIFT	SIFT_PCA with varying PCs			
		10	20	30	40
Naive	.335	.049	.079	.104	.131
LFH	.004	.003	.004	.004	.004

TABLE II

COMPARISON AND FAILURE PERCENTAGE (%) OF LFH

LFH (%)	SIFT	SIFT_PCA with PCs			
		10	20	30	40
comparison percentage	4.2	5.1	4.0	6.0	5.7
failure percentage	3.5	7.4	9.4	8.0	6.0

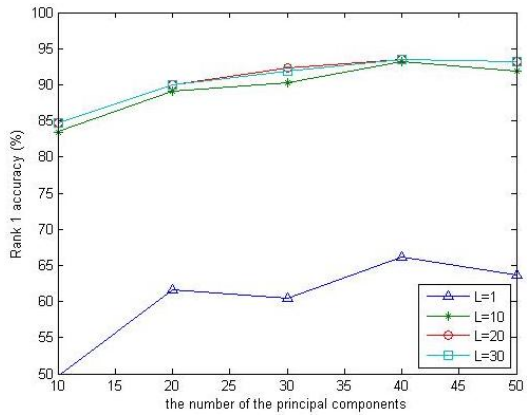

Fig. 5. The rank-1 identification rate of SIFT_PCA on FRGC v2 for different numbers of Principal Components (PCs).

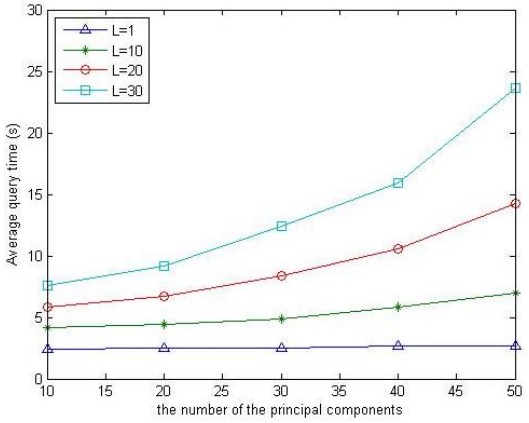

Fig. 6. The query time of SIFT_PCA on FRGC v2 for different numbers of Principal Components (PCs).

We also evaluate our algorithm on face images with partial occlusion. We use a black square to occlude mouth, nose and right eye as illustrated in Fig. 7. Table III shows that the occlusions of mouth and right eye result in minor performance loss, while the nose occlusion results in significant decrease. Although the local feature method has the ability to handle occlusion, the effect of occlusion is dependent on whether the occluded area has a significant number of the discriminant features.

B. Multi-PIE

In order to demonstrate the advantages of our method over existing methods under variation of pose, expression and illumination, we use the Multi-PIE database [9] for evaluation. We choose as our compared method a popular commercial system, PittPatt face recognition system [10], which is a state-of-the-art representative method in the face recognition field.

Fig. 7. Sample of occlusion on FRGC v.2 data.

TABLE III
IDENTIFICATION RATES (%) UNDER PARTIAL OCCLUSION

Occlusion	Rank (%)				
	1	2	3	4	5
mouth	90.73	93.15	95.16	95.16	96.37
nose	76.61	82.66	83.47	84.27	85.48
r_eye	90.73	91.94	92.74	93.95	94.35

The Multi-PIE database was collected at CMU and contains data of 337 subjects with 4 acquisition sessions over a span of six months, and with a large variety of poses, illumination and expressions. In our experiments, we chose the data of 129 subjects who attended all of the four sessions, and under 7 poses from 0° to +90° yaw rotation angles, 2 lighting condition (top frontal flash, top left flash), and 3 expressions (neutral, surprise and squint).

We use the data in the session 1 as gallery data and the data in the session 2 as probe data to evaluate the system robustness across different acquisition situations. In order to build a system with the ability to handle a large range of pose variation, for each subject we use three images as the gallery data that contain one frontal face image (0° yaw angle), one side-view image (+45° yaw angle) and one profile image (+90° yaw angle). All of images in the gallery are of neutral expression under top frontal flash illumination. The number of the total gallery images is 129*3=387. The number of probe images for each test situation (pose, illumination, and expression) is 129. Thus, in total, we used 129 (subjects) * 7 (poses) * 2 (illuminations) * 3 (expressions) = 5418 images for our identification experiment.

The identification experimental comparison on the Multi-PIE data between our LFH and PittPatt are shown in Fig. 9, 10, 11, and 12. Fig. 9 indicates that LFH has the same performance as PittPatt at the 0° yaw angle, and provides improved results on non-frontal-view face recognition. The significant advantage of our LFH method is the good dynamic setting (e.g., it can be easily extended to achieve this extension by simply adding a few exemplar images).

Fig. 10 and Fig. 11 show the LFH and PittPatt performance on the Multi-PIE data with different expressions (surprise and squint expression, respectively). Fig. 12 shows the LFH and Pittpatt performance on the Multi-PIE data with different lighting conditions. These experiments demonstrate that although the PittPatt has better performance on the nearly-frontal-view face images than our LFH, our LFH significantly outperforms the Pittpatt on the non-frontal-view face recognition.

V. CONCLUSIONS

We introduced the Local Feature Hashing (LFH) method which is based on the local feature representation (SIFT) and the *p*-stable distribution Locality-Sensitive Hashing scheme,

Fig. 8. Sample data from the Multi-PIE database. (a) sample data with neutral expression and top frontal flash on; (b) sample data with surprise expression and top frontal flash on; (c) sample data with squint expression and top frontal flash on; (d) sample data with neutral expression and top left flash on.

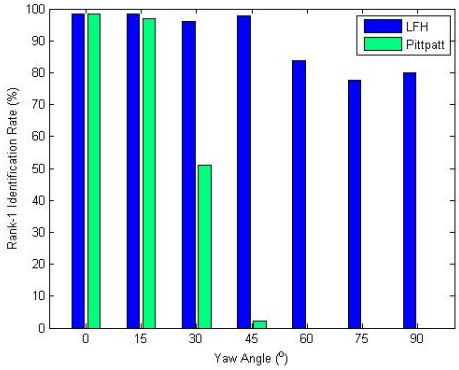

Fig. 9. The identification rate (%) of our LFH and PittPatt on the Multi-PIE database with neutral expression and top frontal flash on.

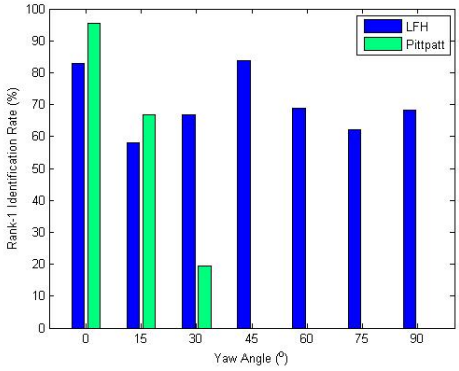

Fig. 10. The identification rate (%) of our LFH and PittPatt on the Multi-PIE database with surprise expression and top frontal flash on.

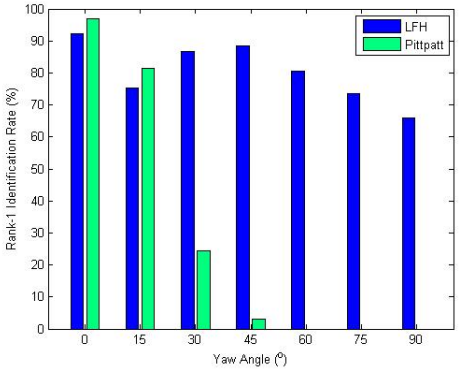

Fig. 11. The identification rate (%) of our LFH and PittPatt on the Multi-PIE database with squint expression and top frontal flash on.

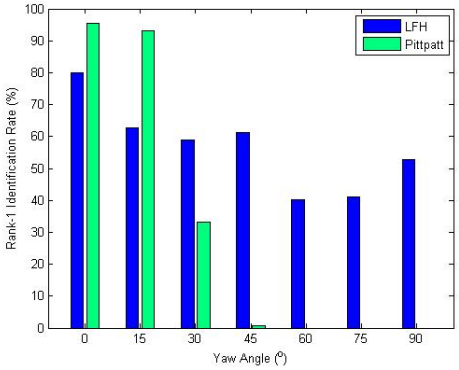

Fig. 12. The identification rate (%) of our LFH and PittPatt on the Multi-PIE database with neutral expression and left frontal flash on

and applied it in the face recognition scenario. Our extensive experiments on two publicly available face databases demonstrated the advantages of our proposed method: it is fully automatic, robust to variation of pose, partial occlusion and facial expressions, easy to implement for Euclidean space without embedding, and suitable to scale to a large-size gallery. The LFH framework is quite general and can be applied to other image search applications. Our future work will include integration of multiple local features (e.g., LBP [1]) to increase identification accuracy, and speed up the algorithm by C++ implementation.

REFERENCES

[1] T. Ahonen, A. Hadid, and M. Pietikainen. Face description with local binary patterns: Application to face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(12):2037–2041, 2006.

[2] A. Albiol, D. Monzo, A. Martin, J. Sastre, and A. Albiol. Face recognition using HOG-EBGM. *Pattern Recognition Letters*, 29(10):1537–1543, July 2008.

[3] K. Bowyer, K. Chang, and P. Flynn. A survey of approaches and challenges in 3D and multi-modal 3D+2D face recognition. *Computer Vision and Image Understanding*, 101(1):1–15, 2006.

[4] M. Datar, N. Immorlica, P. Indyk, and V. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *Proc. ACM Symposium on Computational Geometry*, pages 253–262, Brooklyn, NY, 2004.

[5] Y. Fu, Z. Li, J. Yuan, Y. Wu, and T. S. Huang. Locality versus globality: Query-driven localized linear models for facial image com-

puting. *IEEE Transactions Circuits and Systems for Video Technology*, 18(12):1741–1752, 2008.

[6] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In *Proc. Conference on Very Large Databases*, pages 518–529, San Francisco, CA, USA, 1999.

[7] K. Grauman. *Matching sets of features for efficient retrieval and recognition*. PhD thesis, MIT, 2008.

[8] K. Grauman and T. Darrell. Pyramid match hashing: Sub-linear time indexing over partial correspondences. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, Minneapolis, Minnesota, USA, 2007.

[9] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker. Multi-PIE. In *Proc. International Conference on Automatic Face and Gesture Recognition*, pages 1–8, Amsterdam, The Netherlands, 2008.

[10] http://www.pittpatt.com/face_recognition/.

[11] P. Indyk and R. Motwani. Approximate nearest neighbor: Towards removing the curse of dimensionality. In *Proc. ACM Symposium on Theory of Computing*, pages 604–613, Dallas, Texas, USA, 1998.

[12] Y. Ke and R. Sukthankar. PCA_SIFT: A more distinctive representation for local image descriptors. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 506–513, Washington, DC, USA, 2004.

[13] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[14] S. Lucey and T. Chen. A viewpoint invariant, sparsely registered, patch based, face verifier. *International Journal of Computer Vision*, 80(1):58–71, 2008.

[15] J. Luo, Y. Ma, E. Takikawa, S. Lao, M. Kawade, and B.-L. Lu. Person-specific sift features for face recognition. In *Proc. IEEE Conference on Acoustics, Speech and Signal Processing*, pages 593–596, Honolulu, HI, Apr. 15-20 2007.

[16] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, page 257, 2003.

[17] D. Nistr and H. Stewnius. Scalable recognition with a vocabulary tree. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 2161–2168, 2006.

[18] J. P. Phillips, T. W. Scruggs, A. J. O'Toole, P. J. Flynn, K. W. Bowyer, C. L. Schott, and M. Sharpe. FRVT 2006 and ICE 2006 large-scale results. *National Institute of Standards and Technology Internal Report*, 7408, 2007.

[19] T. H. R. Basri and L. Zelnik-Manor. Approximate nearest subspace search with applications to pattern recognition. In *Proc. Computer Vision and Pattern Recognition*, Minneaplis, Mnneasota, USA, 2007.

[20] H. Samet. *Foundations of Multidimensional and Metric Data Structures*. Elsevier, 2006.

[21] G. Shakhnarovich, P. Viola, and T. Darrell. Fast pose estimation with parameter-sensitive hashing. In *Proc. International Conference on Computer Vision*, pages 750–757, Nice, France, 2003.

[22] C. Silpa-Anan and R. Hartley. Optimised KD-trees for fast image descriptor matching. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, Anchorage, AK, USA, June 2008.

[23] X. Tan, S. Chen, Z. Zhou, and F. Zhang. Face recognition from a single image per person: A survey. *Pattern Recognition*, 39(9):1725–1745, 2006.

[24] T. Tuytelaars and K. Mikolajczyk. Local invariant feature detectors: A survey. *Foundations and Trends in Computer Graphics and Vision*, 3(3):177–280, 2008.

[25] Q. Wang, X. Tang, and H. Shum. Patch based blind image super resolution. In *Proc. International Conference on Computer Vision*, pages I: 709–716, Beijing, China, 2005.

[26] W. Zhao, R. Chellappa, P. Phillips, and A. Rosenfeld. Face recognition: A literature survey. *ACM Computing Surveys*, 35(4):399–458, 2003.