FORENSIC SKETCH BASED RECOGNITION USING DISCRIMINANT ANALYSIS

A Dissertation submitted towards the partial fulfillment
of the requirement for the award of degree of

Master of Technology
in
Signal Processing & Digital Design

Submitted by

KULVINDER KAUR

2K15/SPD/07

Under the supervision of

Sh. AJAY KUMAR GAUTAM
(Assistant Professor, Department of ECE)

Department of Electronics and Communication Engineering

Delhi Technological University

(Formerly Delhi College of Engineering)
Shahbad, Daulatpur - 110042

2015-2017

DELHI TECHNOLOGICAL UNIVERSITY
Established by Govt. Of Delhi vide Act 6 of 2009
(Formerly Delhi College of Engineering)
SHAHBAD DAULATPUR-110042

CERTIFICATE

This is to certify that the dissertation title "**Forensic Sketch Based Recognition Using Discriminant Analysis**" submitted by **Kulvinder Kaur, Roll. No. 2K15/SPD/07**, in partial fulfilment for the award of degree of Master of Technology in "**Signal Processing and Digital Design(SPDD)",** run by Department of Electronics & Communication Engineering in Delhi Technological University during the year 2015-2017, is a bonafide record of student's own work carried out by him under my supervision and guidance in the academic session 2016-17. To the best of my belief and knowledge the matter embodied in dissertation has not been submitted for the award of any other degree or certificate in this or any other university or institute.

Dr. S INDU
HOD
ECE Department
Delhi Technological University
Delhi-110042

Sh. AJAY KUMAR GAUTAM
Supervisor
Assistant Professor (ECE)
Delhi Technological University
Delhi-110042

DELHI TECHNOLOGICAL UNIVERSITY
Established by Govt. Of Delhi vide Act 6 of 2009
(Formerly Delhi College of Engineering)
SHAHBAD DAULATPUR-110042

DECLARATION

I hereby declare that all the information in this document has been obtained and presented in accordance with academic rules and ethical conduct. This report is my own work to the best of my belief and knowledge. I have fully cited all material by others which I have used in my work. It is being submitted for the degree of Master of Technology in Signal Processing & Digital Design at the Delhi Technological University. To the best of my belief and knowledge it has not been submitted before for any degree or examination in any other university.

Kulvinder Kaur
M. Tech. (SPDD)
2K15/SPD/07

Date: …..JULY, 2017
Place: Delhi Technological University, Delhi

ACKNOWLEDGEMENT

I owe my gratitude to all the people who have helped me in this dissertation work and who have made my postgraduate college experience one of the most special periods of my life.

Firstly, I would like to express my deepest gratitude to my supervisor **Sh. Ajay Kumar Gautam**, Assistant Professor (ECE) for his invaluable support, guidance, motivation and encouragement throughout the period during which this work was carried out.

I also wish to express my heart full thanks to my classmates as well as staff at Department of Electronics & Communication Engineering of Delhi Technological University for their goodwill and support that helped me a lot in successful completion of this project.

Finally, I want to thank my parents, family and friends for always believing in my abilities and showering their invaluable love and support.

Kulvinder Kaur
M. Tech. (SPDD)
2K15/SPD/07

ABSTRACT

In this project, we are addressing the problem of matching sketches (forensic) to mug shot images. In previous researches, forensic sketch matching offered only solutions to highly accurate viewed sketches(sketches that are drawn by looking at the person). The difference between forensic sketches and viewed sketches is that the former is drawn by a police artist with the help of the description provided by an eye witness. We here present a framework called local feature based discriminant analysis(LFDA)to differentiate between various forensic drawings. In LFDA we separately express both drawings and pictures using SIFT feature descriptor and multi-scale local binary patterns(MLBP). We then use multiple discriminant projections on subdivided vectors of feature based representation for least separation matching. On comparison to a leading face recognition system, LFDA provides substantial rectification in comparing forensic drawings to corresponding face images.

Table of Contents

List of Figure

CHAPTER ONE

INTRODUCTION

1 INTRODUCTION

There has been seen tremendous progress in biometric recognition. In law field, additional tools have been used in determining criminal's identity. In addition to circumstantial evidence and DNA reports, if any fingerprint is been found at the scene of investigation or if the surveillance camera captures the image of the face of the suspect, then these clues can be helpful in determing the identity of the culprit using automated biometric identification. But, in many crime cases occur when none of this information can be provided but there is an eye witness is present in the place of crime scene. In such cases, a forensic sketch is drawn by the sketch artist with the help of eye witness with the information provided by the witness. After the successful completion of the sketch, it is distributed among police officers and media in order to be distributed to people with the hope that someone would recognize the suspect. These sketches which are drawn with the help of eye witnesses are called forensic sketches and in this paper we will describe a robust method to match these sketches (forensic sketches) to large mug shot images (database) maintained by law enforcement agencies.

There are two different kinds of face sketches which we are going to discuss here: first, viewed sketch and second, forensic sketch. Viewed sketches are the ones which are drawn as the one making sketch views the photograph of any person or in some cases it could be that he is making sketch while looking at the person only. Forensic sketches are the ones which are drawn with the help of details provided by the eye witness. Many research papers which are published have focused on matching of viewed sketches in spite of the fact that in real world situations forensic sketches are involved. But these both kinds of sketches prove to be a great challenge when it comes to face matching as that sketch image may contain different kinds of textures when compared to the stored photographs with which they are being matched. Still, forensic ones have to face extra challenges as the eye witness may not be able to provide or may forget the exact look of the suspect which in return leads to incomplete and inaccurate sketch.

We focus on two important difficulties here in matching the forensic sketches: firstly, matching across image modalities, and secondly, performing face recognition despite the inaccurate face sketch. So, in order to solve the first problem, we have used LFDA (local-based feature discriminant analysis to perform minimum matching of distances between sketches

drawn and photos available as explained in fig. 1. The second mentioned problem is faced when we match forensic sketches to large mug shot images.

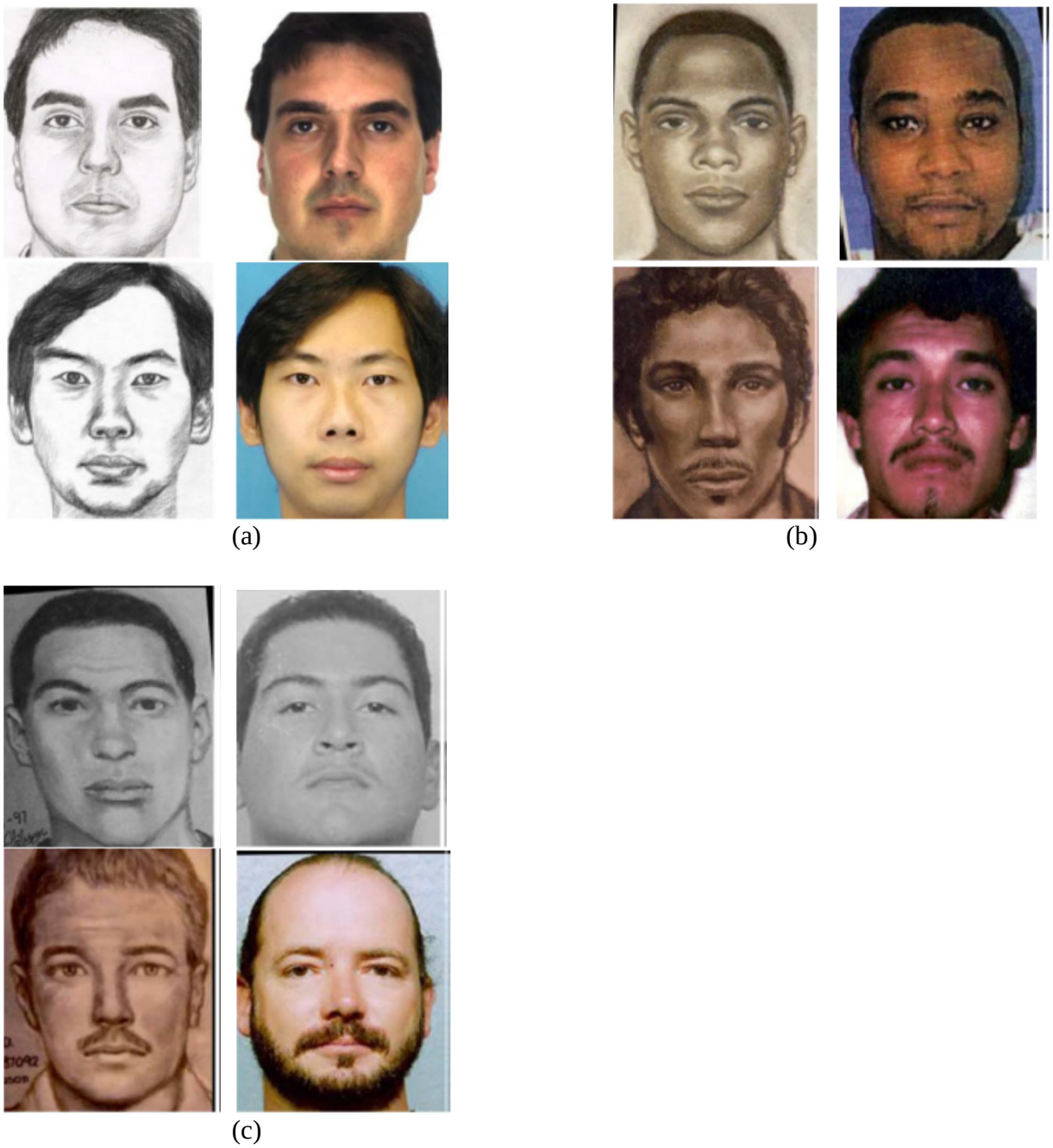


Figure 1. The difference between forensic sketches and viewed sketches. (a) Viewed sketches and their corresponding photographs, (b) two pairs of good quality forensic sketches and the corresponding photographs, and (c) two pairs of poor quality forensic sketches and the corresponding photographs.

CHAPTER TWO

RELATED WORK

2 RELATED WORKS

Compared to photo-based face recognition, there is only a limited amount of research on sketch recognition, and most of the published work is focused on hand drawn sketches.

The initial research in this field focused on matching sketches that are drawn by an artist while looking at the corresponding photograph of the person or the person himself (called viewed sketches). Although both viewed and forensic sketches are drawn by an artist, the difference is that forensic sketches are drawn following the verbal description of an eyewitness or the victim, instead of looking at a person or photograph. During the drawing of a forensic sketch, the witness usually cannot exactly recall the facial appearance of a suspect. Additionally, it is often the case that a disparity exists between the understanding and depiction of facial features between an artist and the eyewitness. Thus, additional challenges are posed when matching forensic sketches against face photographs. The studies on viewed sketches can be grouped into two categories: modal transformation and modal-insensitive feature representation. Approaches in the first category convert images from one modality (e.g. sketch) into a different modality (e.g. photo). Methods for modal transformation include eigen transformation, local linear embedding (LLE), multi-scale Markov Random Fields model, and embedded hidden Markov model (E-HMM) . The merit of these approaches is that traditional face matching algorithms, designed for the target modality, can be used following the modal transformation. However, the synthesized photo (for example) can only be called a pseudo-photo due to its inferred content. In fact, these synthesis methods are often solving a more difficult problem than the recognition task. The second approach to sketch recognition attempts to learn or design feature representations that reduce the intra-class difference caused by modality gap while preserving interclass separability. Representative methods in this category include common discriminant space, coupled spectral regression (CSR), coupled information-theoretic projection (CITP) and partial least squares (PLS).

Major work (on viewed sketches) was performed by *Tang et al.*, these studies shared a common approach i.e., a synthetic photograph is generated from a given sketch (or vice versa method is used) and then a standard face recognition is used to match the synthetic photographs to the ones in the gallery. The different synthesis methods used to recognize the face included an eigen transformation method, local linear embedding and belief propagation on a Markov

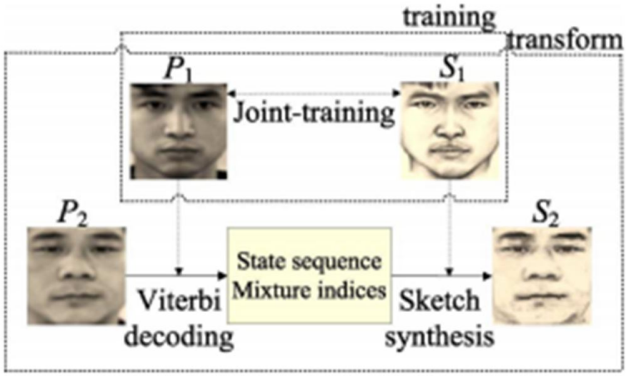
random field. Other methods for synthesis were also proposed as well. Al Nizami studied the impact of the sketches which various artists drew.

Klare and Jain had proposed a method of matching sketches that used same feature based approach that had been successful in many other scenarios of heterogeneous face recognition(especially when matching near infrared face image to visible light). In using SIFT feature descriptors, the intrapersonal variations between the sketch and photo modality were diminished while still maintaining sufficient information for interclass discrimination. Such an approach is similar to other methods of matching near-infrared images (NIR) to visible light images (VIS), where local binary pattern feature descriptors are used to describe both NIR and VIS images. In this paper, we extend our previous feature-based approach to sketch matching. This is achieved by using local binary patterns (LBP) in addition to the SIFT feature descriptor, which is motivated by LBP's success in a similar heterogeneous matching application by Liao et al. Additionally, we extend our feature-based matching to learn discriminant projections on "slices" of feature patches, which is similar to the method proposed by Lei and Li.

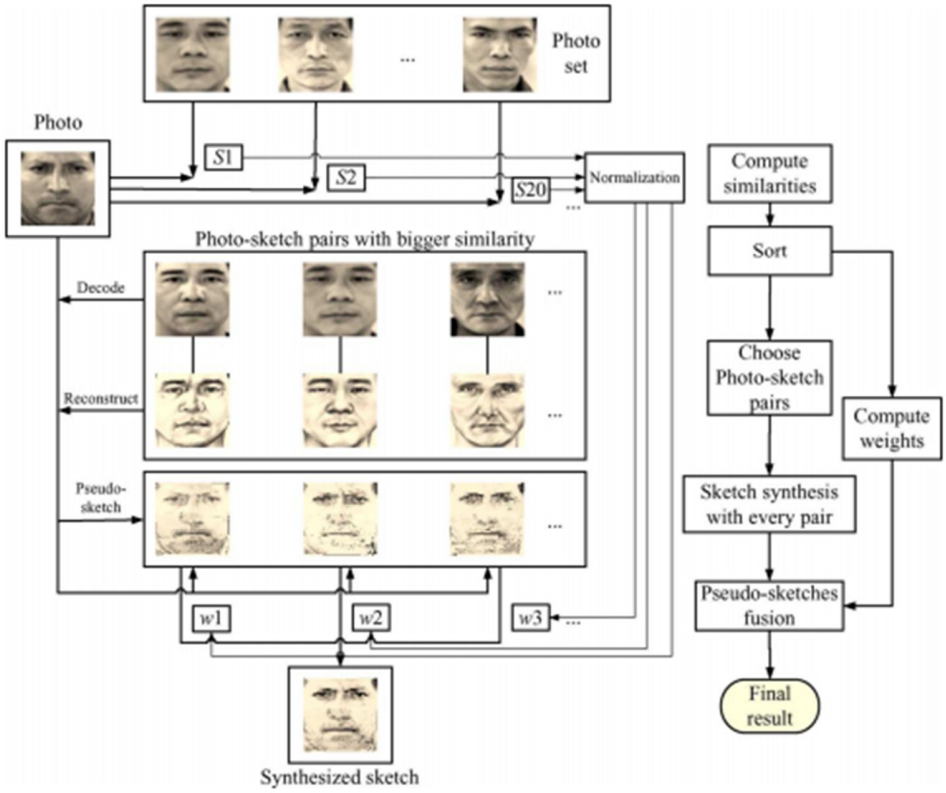
Xiaoou Tang et. al. proposed a method in which automatic retrieval of face images from police mug-shot databases is critically important for law enforcement agencies. It can effectively help investigators to locate or narrow down potential suspects. However, in many cases, the photo image of a suspect is not available and the best substitute is often a sketch drawing based on the recollection of an eyewitness. In this paper, we present a novel photo retrieval system using face sketches. By transforming a photo image into a sketch, we reduce the difference between photo and sketch significantly, thus allowing effective matching between the two. Experiments over a data set containing 188 people clearly demonstrate the efficacy of the algorithm. A novel face sketch recognition algorithm is developed in this paper. The photo-to-sketch transformation method is shown to be an effective approach for automatic matching between a photo and a sketch. Surprisingly, the recognition performance of the new approach is even better than that of human beings. Of course, like most of the regular photo-based face recognition researches, further verification of our conclusions are needed on a larger scale of test. Nevertheless, without considering the absolute recognition accuracy, the relative superior performance of the new method compared to the human performance and the conventional photo based methods clearly demonstrates the advantage of the new algorithm.

Xiaogang Wang et. al. proposed a novel face photo-sketch synthesis and recognition method using a multiscale Markov Random Fields (MRF) model. Our system has three components: 1) given a face photo, synthesizing a sketch drawing; 2) given a face sketch drawing, synthesizing a photo; and 3) searching for face photos in the database based on a query sketch drawn by an artist. It has useful applications for both digital entertainment and law enforcement. We assume that faces to be studied are in a frontal pose, with normal lighting and neutral expression, and have no occlusions. To synthesize sketch/photo images, the face region is divided into overlapping patches for learning. The size of the patches decides the scale of local face structures to be learned. From a training set which contains photo-sketch pairs, the joint photo-sketch model is learned at multiple scales using a multiscale MRF model. By transforming a face photo to a sketch (or transforming a sketch to a photo), the difference between photos and sketches is significantly reduced, thus allowing effective matching between the two in face sketch recognition. After the photo-sketch transformation, in principle, most of the proposed face photo recognition approaches can be applied to face sketch recognition in a straightforward way. Extensive experiments are conducted on a face sketch database including 606 faces. A novel face photo-sketch synthesis and recognition system. Given a face photo (or a face sketch), its sketch (or photo) can be synthesized using a multiscale Markov Random Fields model, which learns the face structure across different scales. After the photos and the sketches have been transformed to the same modality, various face recognition methods are evaluated for the face sketch recognition task. Our approach is tested on a face sketch database including 606 faces. It outperforms existing face sketch synthesis and recognition approaches.

Xinbo Gao et. al. proposed a technique in which sketch synthesis plays an important role in face sketch-photo recognition system. In this manuscript, an automatic sketch synthesis algorithm is proposed based on embedded hidden Markov model (E-HMM) and selective ensemble strategy. First, the E-HMM is adopted to model the nonlinear relationship between a sketch and its corresponding photo. Then based on several learned models, a series of pseudo-sketches are generated for a given photo. Finally, these pseudo-sketches are fused together with selective ensemble strategy to synthesize a finer face pseudo-sketch. Experimental results illustrate that the proposed algorithm achieves satisfactory effect of sketch synthesis with a small set of face training samples.



Framework of sketch synthesis algorithm based on E-HMM.



Framework of the sketch synthesis algorithm based on selective ensemble.

By analyzing the difference between sketches and photos systematically, we have presented a sketch synthesis algorithm based on E-HMM and selective ensemble. First, the E-HMM is adopted to model the nonlinear relationships in sketch-photo pairs, and a series of pseudo-sketches of the same photo are generated using different models. Then those generated pseudo-sketches are fused together with the selective ensemble strategy. Finally, a finer face pseudo-sketch is synthesized. The experimental results show that the proposed method achieves

satisfactory sketch synthesis effect with a small set of face training samples. Whereas, a clear disadvantage of the E-HMM is that it is hard to learn more complex nonlinear relationship. So, we plan to research on how to improve the structure of the E-HMM in the future. The proposed method can be used to set up an automatic face sketch recognition system, which can find various applications in the fields of counter-strike, safety guard, and image or video retrieval based on sketches.

Brendan F. Klare et. al. in this previous research in sketch matching only offered solutions to matching highly accurate sketches that were drawn while looking at the subject (viewed sketches). Forensic sketches differ from viewed sketches in that they are drawn by a police sketch artist using the description of the subject provided by an eyewitness. To identify forensic sketches, we present a framework called local feature-based discriminant analysis (LFDA). In LFDA, we individually represent both sketches and photos using SIFT feature descriptors and multiscale local binary patterns (MLBP). Multiple discriminant projections are then used on partitioned vectors of the feature-based representation for minimum distance matching. We apply this method to match a data set of 159 forensic sketches against a mug shot gallery containing 10,159 images. Compared to a leading commercial face recognition system, LFDA offers substantial improvements in matching forensic sketches to the corresponding face images. We were able to further improve the matching performance using race and gender information to reduce the target gallery size. Additional experiments demonstrate that the proposed framework leads to state-of-the-art accuracys when matching viewed sketches.

CHAPTER THREE

PROPOSED METHOD

3 PROPOSED METHOD

3.1 FEATURE-BASED SKETCH MATCHING

Image feature descriptor describes an image or a part of image using any type of feature vector that usually captures the distinct characteristics of the given image. These kinds of features (image-based) have proven to be successful in face recognition, mostly with the use of LBPs (local binary patterns).

3.1.1 Feature-Based Representation:

We will now be describing the way to represent any face with image-descriptors. As most image-descriptors are not sufficiently verbose to fully describe any face image, the image descriptors are calculated or estimated over a set of evenly distributed sub-regions of the face image. The feature vectors located at sampled regions are summed up together to describe the face. The feature sampling point is chosen using two parameters: *s* as region or patch size and δ as displacement size. The region size denoted as s is defined as the size of the square window over which we compute the image feature. The displacement size denoted as δ denotes the number of pixels the patch is displayed for each sample; hence, (s- δ) is the number of overlapping pixels in two adjacent patches. This is like scanning a window of *s x s* on the complete face image. For any H x W dimensional image, the number of vertical (M) and horizontal (N) sampling locations are given by $M= (H-s)/\delta+1$ and $N= (W-s)/\delta+1$. At each of the N.M patches, we calculate the d-dimensional image feature vector φ. Now we concatenate these image feature vectors into a single (N. M. d) - dimensional image vector φ. Whereas *f(I)* : $I \rightarrow \varphi$ denotes the extraction of a single feature descriptor from an image, sampling multiple features using overlapping patches is denoted as $F(I) \rightarrow \varphi$. Minimum distance sketch matching can be performed directly using this feature-based representation of subjects i and j by computing the normed vector distance $\|F(I^i) - F(I^j)\|$.

In our sketch matching framework, two feature descriptors are used: SIFT and LBP. The SIFT feature descriptor quantizes both the spatial locations and gradient orientations within an s x s-sized image patch, and computes a histogram in which each bin corresponds to a combination

of a particular spatial location and orientation. For each image pixel, the histogram bin corresponding to its quantized orientation and location is incremented by the product of 1) the magnitude of the image gradient at that pixel and 2) the value of a Gaussian function centered on the patch with a standard deviation of s=2. Tri-linear interpolation is used on the quantized location of the pixel, which addresses image translation noise. The final vector of histogram values is normalized to sum to one. It is important to reiterate that because we are sampling SIFT feature descriptors from a fixed grid and we do not use SIFT key-point detection, the SIFT feature descriptor is computed at predetermined locations.

For the local binary pattern feature descriptor, we extended the LBP to describe the face at multiple scales by combining the LBP descriptors computed with radii $r \epsilon \{1; 3; 5; 7\}$. We refer to this as the multi-scale local binary pattern (MLBP). MLBP is similar to other variants of the LBP, such as MB-LBP, but we obtained slightly improved accuracy using MLBP.

The choice of the MLBP and SIFT feature descriptors was based on reported success in heterogeneous face recognition and through a quantitative evaluation of their ability to discriminate between subjects in sketches and photos. Though variants of LBPs have led to substantial success in previous heterogeneous face recognition scenarios, the use of SIFT feature descriptors for this application is quite novel. However, the recent work clearly demonstrates the success of SIFT feature descriptors for viewed sketch recognition. SIFT feature descriptors have also been shown to perform comparatively with LBP feature descriptors in a standard face recognition scenario. These feature descriptors are well-suited for sketch recognition because they describe the distribution of the direction of edges in the face; this is the information that both sketches and photos contain. By densely sampling these descriptors, sufficient discriminatory information is retained to more accurately determine a subject's identity over previously used synthesis methods. The feature-based representation requires each sketch and photo image to be normalized by rotating the angle between the two eyes to 0 degree, scaling the images to a 75 inter-ocular pixel distance, and cropping the image size to 200 by 250 pixels.

3.1.2 Local Feature-Based Discriminant Analysis:

With both sketches and photos characterized using SIFT and MLBP image descriptors, we further refine this feature space using discriminant analysis. This is done to reduce the large

dimensionality of the feature vector ϕ. A straightforward approach would be to apply classical subspace analysis (such as LDA) directly on ϕ, and to extract discriminant features for classification. However, there are several problems with this approach. First, the feature dimensionality is too high for direct subspace analysis. In our experiments, each image is divided into either 154 overlapping patches (for s = 32) or 720 overlapping patches (for s = 16), with each patch producing a 128-dimensional SIFT descriptor or a 236-dimensional MLBP descriptor. The second problem is the possibility of over fitting due to the small sample size (SSS).

In order to handle the combination of a large feature size and small sample size, an ensemble of linear discriminant classifiers called LFDA is proposed. Other discriminant analysis methods have been proposed to handle the SSS problem, such as random sampling LDA, regularized LDA, and direct LDA. However, we chose the proposed LFDA method because it is designed to work with a feature descriptor representation (as opposed to an image pixel representation), and it resulted in high recognition accuracy.

In the LFDA framework, each image feature vector ϕ is first divided into "slices" of smaller dimensionality, where slices correspond to the concatenation of feature descriptor vectors from each column of image patches. Next, discriminant analysis is performed separately on each slice by performing the following three steps: PCA, within class whitening, and between class discriminant analysis. Finally, PCA is applied to the new feature vector to remove redundant information among the feature slices to extract the final feature vector.

To train the LFDA, we use a training set consisting of pairs of a corresponding sketch and photo of n subjects (which are the n training classes). This results in a total of 2n training images with two supports for each subject i: the image feature representation of the sketch $\Phi^i_s = F(I^i_s)$ and the photo $\Phi^i_p = F(I^i_p)$. We combine these feature vectors as a column vector in training matrices and refer to them as $X^p = [\Phi^1_p \Phi^2_p \ldots . \Phi^n_p]$ for the photo, $X^s = [\Phi^1_s \Phi^2_s \ldots . \Phi^n_s]$ for the sketch, and $X = = [\Phi^1_s \Phi^2_s \ldots . \Phi^n_s \Phi^1_p \Phi^2_p \ldots . \Phi^n_p]$ for the sketch and photo combined.

The first step in LFDA is to separate the image feature vector into multiple sub vectors or slices. Given the M x N array of patches consisting of SIFT or MLBP descriptors, we create one slice for each of the N patch columns. With a d-dimensional feature descriptor, each of the N slices is of dimensionality (M. d). We call this a "slice" because it is similar to slicing an image into N pieces. After separating the feature vectors into slices, the training matrices now becomes $X^s_k \in \mathbb{R}^{M.d,n}$, $X^p_k \in \mathbb{R}^{M.d,n}$, and $X_k \in \mathbb{R}^{M.d,2n}$ (k = 1 . . . N), which are all mean-centered.

We next reduce the dimensionality of each training slice matrix X_k using the PCA matrix $W_k \in \mathbb{R}^{M.d;r}$ with r eigenvectors. The purpose is to remove the noisy features which are usually associated with the trailing eigenvectors with the smallest eigen values. In our experiments, we use the 100 eigenvectors with the largest eigen values (which preserves about 90 percent of the variance). The discriminant extraction proceeds by generating the mean projected class vectors

$$Y_k = W_k^T(X_k^s + X_k^p)/2 \quad (1)$$

which are used to center the sketch and photo training instances of each class by

$$\ddot{X}_k^s = W_k^T X_k^s - Y_k \quad (2)$$
$$\ddot{X}_k^p = W_k^T X_k^p - Y_k$$

To reduce the intrapersonal variation between the sketch and the photo, a whitening transform is performed. Whitening the within class scatter matrix reduces the large feature dimensions that represent the principal intrapersonal variations, which in this case correspond to intrapersonal differences between sketches and photos. To do so, we recombine the training instances into $\ddot{X}_k = [\ \ddot{X}_k^s\ \ddot{X}_k^p\ ]$. PCA analysis is performed on $\sim X_k$ such that the computed PCA projection matrix $\tilde{V} \in \mathbb{R}^{100,100}$ retains all data variance from $\sim X_k$ Let $A_k \in \mathbb{R}^{100,100}$ be a diagonal matrix whose entries are the eigen values of the corresponding PCA eigenvectors $\tilde{V}_k$. The whitening matrix is $V_k = (\ \Lambda_k^{-1/2}\ V_k^T)^T$.

The final step is to compute a projection matrix that maximizes the intra person scatter by performing PCA on $V^T\ Y_k$ (which is the whitening transform of the mean class vectors). Using all but one of the eigenvectors in the PCA projection matrix, the resultant projection matrix is denoted as $U_k \in \mathbb{R}^{100,99}$. This results in the final projection matrix for slice k:

$$\Psi_k = W_k\ V_k\ U_k \quad (3)$$

With each local feature-based discriminant trained, we match sketches to photos using the nearest neighbor matching on the concatenated slice vectors. We first separate the feature representation of an image into individual slices.

$$\Phi = [\Phi(1)^T\ \Phi(2)^T \quad \Phi(N)^T]^T \quad (4)$$

Where $\Phi(i) \in \mathbb{R}^{M.d}$ is the ith slice feature vector. We then project each slice using the LFDA projection matrix ψ_k, yielding the new vector representation $\psi \in \mathbb{R}^{M.99}$:

$$\Psi = [(\psi_k^T \varphi(1))^T\ (\psi_k^T \varphi(2))^T \ldots\ldots (\ \psi_k^T \varphi(1))^T]^T$$

With the LFDA representation of the sketch Ψ_s and photo Ψ_p, the normed distance $\| \Psi_s - \Psi_p \|$ is used to select the gallery photo with the minimum distance to the probe sketch

The proposed LFDA algorithm is a simple yet effective method. From the results, we can clearly see that LFDA is able to significantly improve the recognition performance over the basic feature-based sketch matching framework First, LFDA is more effective in handling large feature vectors. The idea of segregating the feature vectors into slices allows us to work on more manageable sized data with respect to the number of training images. Second, because the subspace dimension is fixed by the number of training subjects, when dealing with the smaller sized slices, the LFDA algorithm is able to extract a larger number of meaningful features. This is because the dimensionality of each slice subspace is bounded by the same number of subjects as a subspace on the entire feature representation would be.

3.2 TRAINING

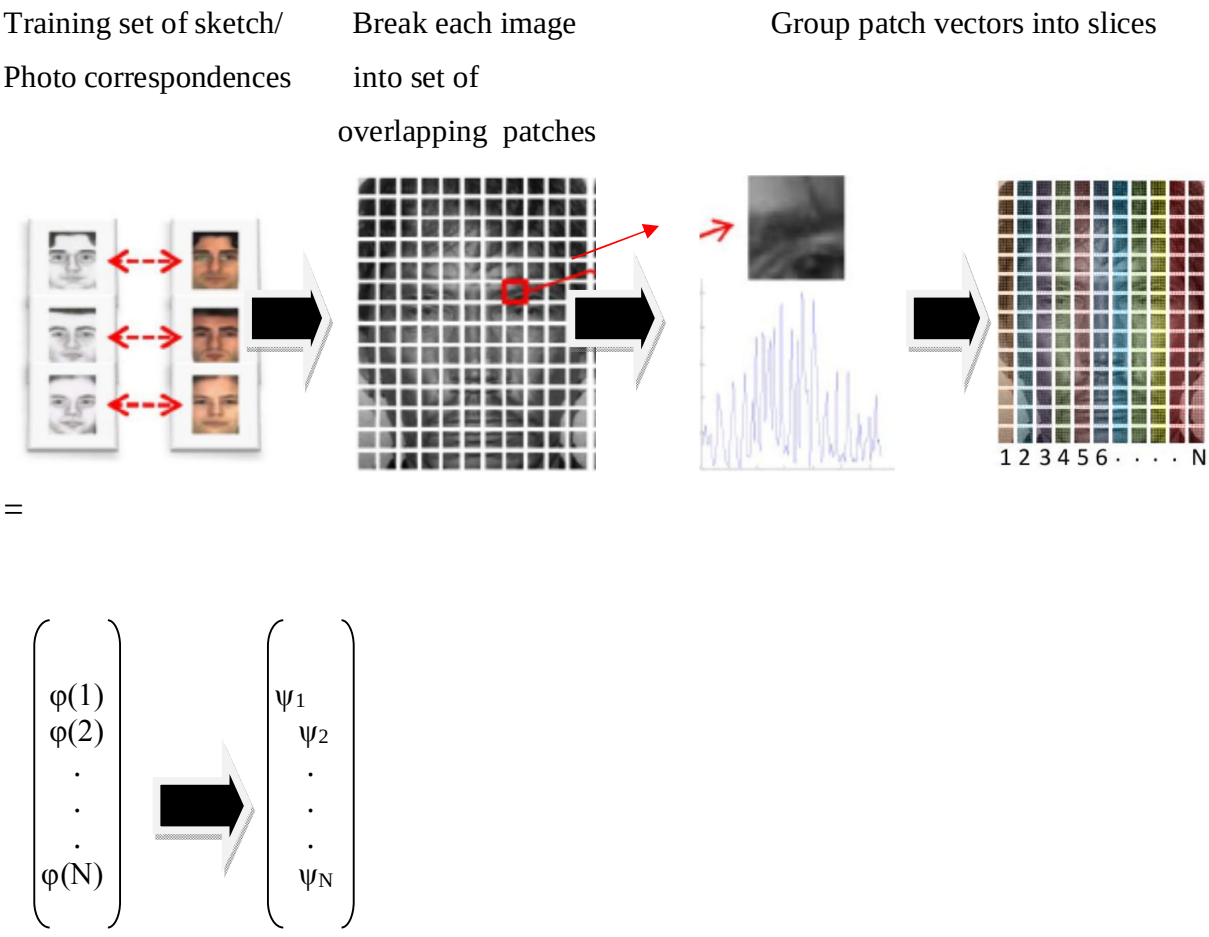


Figure 2: An overview of the training using the LFDA framework.

3.3 MATCHING

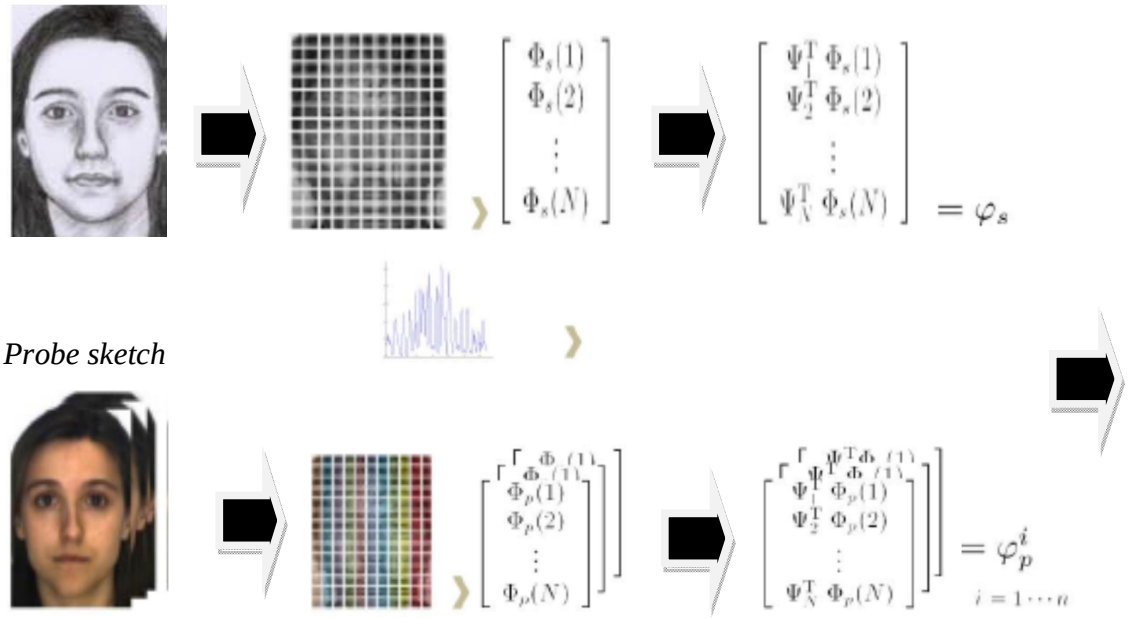
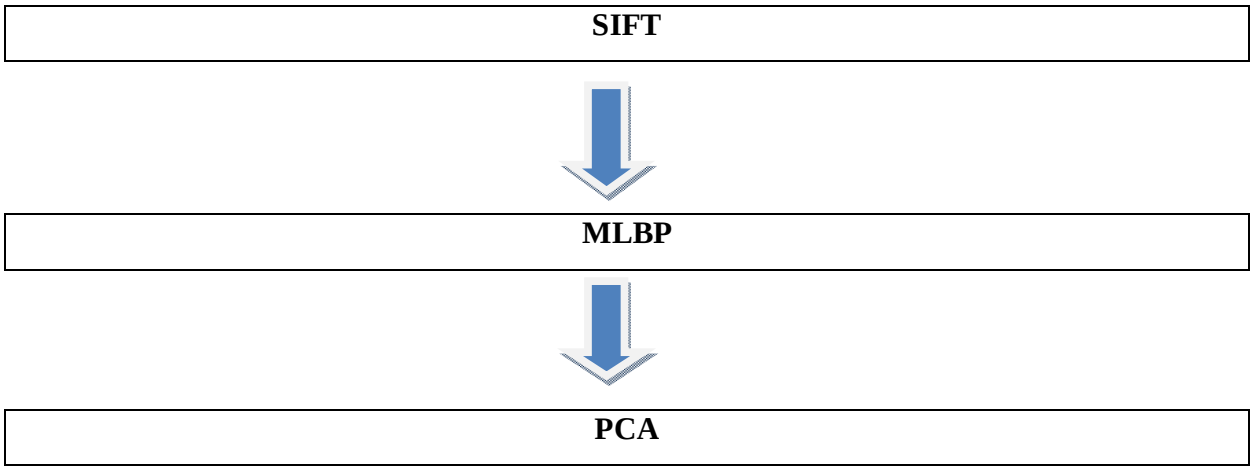


Figure 3: An overview of the recognition using the LFDA framework.

3.4 METHODOLOGY USED



3.4.1 SIFT (Scale Invariant Feature Transform)

Sketch matching is a key aspect in many fields like computer vision which would include tracking of objects, matching of objects or it could be scene recognition. It is widely used in forensic studies for matching of objects. The main step in matching of images is done by extracting features from images or sketches. This is an essential part in matching as we cannot compare complete images or sketches in every case. In case of forensic recognition we have sketches, which are to be compared to either real time images or mug shot images already saved in our database. So in such cases we cannot compare complete images. Hence we extract key features from the sketches to compare them with the test images. We use such properties of the features that will make them suitable for matching different images of an object. The features which we extract do not vary with image scaling and rotation but they are partially invariant in illumination. The feature are well localized in both spatial and frequency domains, thereby reducing the probability of disruption by clutter, occlusion, or noise. Efficient algorithms can help in extracting large number of features from typical images. Highly distinctive features allow matching of single feature with high probability against large data base of features, thereby providing a basis for object recognition. Cascade filtering approach minimizes the cost of extraction of features. There are four major stages of computation in this method which are used to generate the set of image features:

1. **Scale-space extrema detection**: It is the first stage of our method. It searches over all scales and image locations. Difference-of-Gaussian function is used to identify potential interest points that do not vary with scale and orientation.
2. **Key-point localization:** A detailed model is fit to determine the location and scale for each candidate location. Based on measures of their stability, key-points are selected.
3. **Orientation assignment:** Based on local image gradient directions more than one orientation can be assigned to each key-point location. All future operations are performed on image data that has been transformed relative to the assigned orientation, scale, and location for each feature, thereby providing invariance to these transformations.

4. **Key-point descriptor:** we measure local image gradients at selected scales in the each around each key-point. These are transformed into a representation that allows for significant level of local shape distortion and change in illumination.

Since it transforms image data into scale invariant co-ordinates relative to local features, this approach has been named Scale Invariant Feature Transform(SIFT).

An important aspect of the approach is that it produces large numbers of features which densely cover the images over the full range of scale and location. A typical image of size 500*500 pixels will give rise to around 2000 stable features (although this value depends on both images material and choice of various parameters). The quantity of features available are particularly important for object recognition, where ability to detect smaller objects in cluttered background requires that at least 3 features are to be correctly matched with each object for reliable identification.

To do image matching and recognition, SIFT feature is first extracted from a reference pack images and entered in a database. New image is matched by individually comparing the feature from the new image to the previous database and finding the matching features based on Euclidean distance of its feature vectors.

The key point descriptions are highly distinctive, which allows single feature to find its correct match with appreciable probability in a large database of attributes. However, for a cluttered image, many features of the background won't have any correct match in the database, leading to rise in many false matches in addition to the correct answers. The correct matches can be extracted from the full set of values by identifying subsets of key points that agree on the object, its location, scale and orientation of the new image. The probability that some features will agree on such parameters by chance is much lesser than the probability that any individual function match will be in error. The determination of such consistent clusters must be performed rapidly by use of an efficient hash table implementation of available generalized Hough transform.

Every such cluster of 3 or greater features that agree upon an object and its pose are then subject to further depth verification. First, a least-squared calculation is made for an affine approximation for the given object poses. Also, any other image features consistent with the pose is identified, and outliers are to be discarded. Finally, detailed computations are to made of the

probability that a given set of features indicates for the presence of an object, given the suitability of fit and cases of probable false matches. Object matches that shall pass all these tests can be selected as correct with high probability and confidence.

3.4.1.1 Detection of scale-space extrema:

As described above, we will detect key points by using a cascade filtering approach that makes use of efficient algorithms to identify candidate's locations that are examined in further detail. Now, the first stage of key-point detection is to find such locations and scales that can be repetitively assigned with differing views of the same entity. Detecting locations that are invariant to scale change of the image may be accomplished by searching for suitable features across all possible scales, using continuous function of scale called as scale space.

It has been seen that that for a variety of reasonable assumptions, the only possible type of scale-space kernel is the Gaussian function. So, the scale space for an image is defined as a function, $L(x, y, \sigma)$ that is produced using the convolution of a variable-scale Gaussian statement, $G(x, y, \sigma)$, with an input image, $I(x, y)$:

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)$$

where * is the convolution operation for x and y, and

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}$$

To detect efficiently stable key-point locations in a scale space, we have proposed a method using a scale-space extrema in the difference-of-Gaussian function given with the image, D(x, y, σ), which can be computed from difference of two nearby scales which have been separated by a constant multiplicative factor referred *k*:

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma) \end{aligned} \quad (1)$$

There are many reasons for choosing this type of function. Firstly, it is a particularly efficient type of a function for computing as the smoothed images, *L*; needs to be computed as in any case for scale space feature description. Similarly, *D* can also be computed by feature of simple image subtraction.

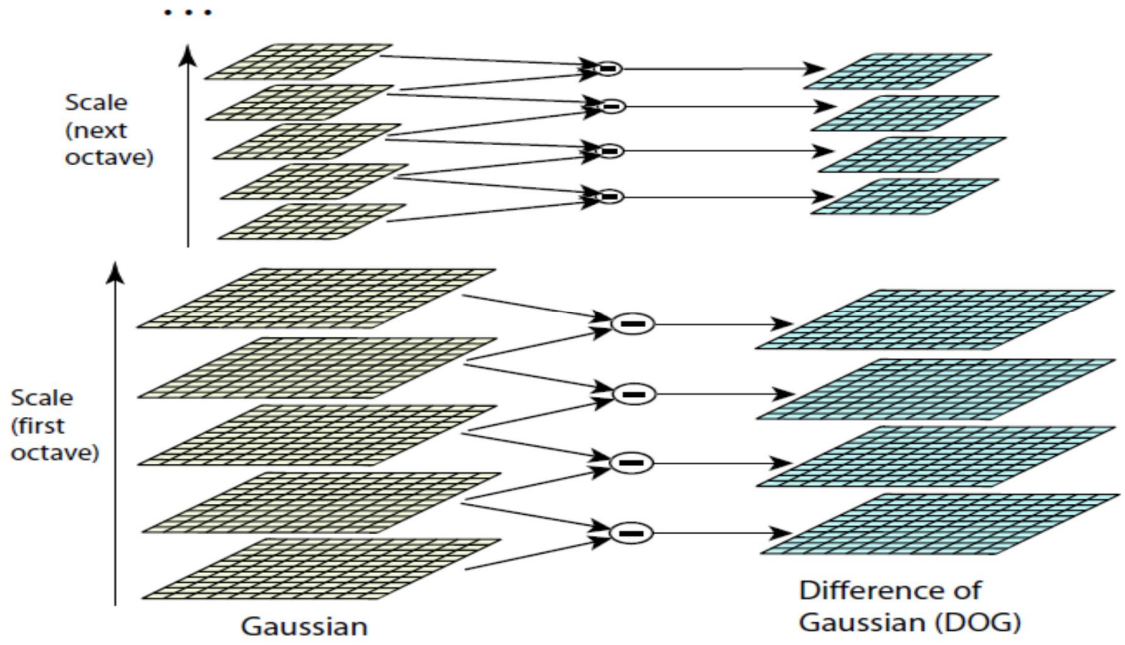


Figure 4 : For each octave of scale space, the initial image is repeatedly convolved with Gaussians to produce the set of scale space images shown on the left. Adjacent Gaussian images are subtracted to produce the difference-of-Gaussian images on the right. After each octave, the Gaussian image is down-sampled by a factor of 2, and the process repeated.

In addition, a difference-of-Gaussian function gives a close approximation to the scale normalized Laplace of Gaussian, $\sigma^2\nabla^2 G$ as studied and given by Lindeberg. Lindeberg explained that the normalization of the Laplacian with the factor σ^2 which is selected for true scale invariance. In detailed experiments, Mikolajczyk found that maxima and minima of $\sigma^2\nabla^2 G$ produces more stable images compared to a range of other possible image functions, such as gradient function Hessian or Harris corner function.

The relationship between D and $\sigma^2\nabla^2 G$ can be evaluated using the heat diffusion equation (parameterized in terms of σ rather than more usual $t = \sigma^2$.

$$\frac{\partial G}{\partial \sigma} = \sigma\nabla^2 G$$

From this, we see that, $\nabla^2 G$ can be computed from the finite difference approximation to $\partial G / \partial \sigma$, using the difference of nearby scales at kσ and σ.

$$\frac{\partial G}{\partial \sigma}=\sigma\nabla^2 G \approx \frac{G(x,y,k\sigma)-G(x,y,\sigma)}{k\sigma-\sigma}$$

and therefore,

$$G(x,y,k\sigma)-G(x,y,\sigma)\approx(k-1)\sigma^2\nabla^2 G$$

The above equation shows that when the difference-of-Gaussian function is having scales differing by a constant value factor it already incorporates the σ^2 scale normalization required in by the scale-invariant Laplacian. The factor *(k-1)* in the shown equation is a constant value on over all scales and hence does not give any influence on extrema location points. The approximation error tends to go to zero as k tends to 1, but we have found that approximate value results have almost no impact upon the stability and results of extrema detection for significant differences in scale, like in case of k=$\sqrt{2}$.

A more efficient approach is the construction of D(x, y, σ) which is shown in Figure. Initially, value is incrementally convolved with Gaussians for producing images separated using a constant factor k in scale space, as shown arranged in the left column stack. We have chosen to divide each octave of scale space (i.e., doubling of σ) into a integer number, *s*, in intervals, so $k = 2^{1/s}$. Using this, we shall produce $s + 3$ images in the stack which will consist of blurred images for each of the octaves, so that final extrema detection will cover an entire octave. Adjacent image scales are to subtracted in order to produce a difference-of-Gaussian images which are available on the right. Once a complete octave gets processed, we undergo a re-sampling process of the Gaussian image that will again have twice the initial value of σ. (it will be made up of 2images starting from the top of the stack) by taking into consideration every second pixel in each of the rows and columns. The accuracy of taking the samples is relative to σ is not different than for the start of the previous taken octave, whereas the computation is greatly reduced to a certain extent.

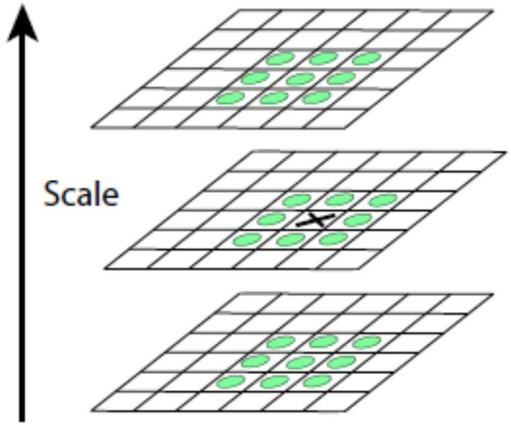


Figure 5: *Minima and maxima of the difference-of-Gaussian images are detected by comparing a pixel (marked with X) to its 26 neighbors in 3x3 regions at the current and adjacent scales (marked with circles).*

(a) Local extrema detection: For doing the detection of the local maxima and minima of D(x; y; σ) each of the sample point is to compared with respect to its eight neighbors in the current available image and nine neighbors in the scale below and above. It is to be selected only in case if it is larger than all of its neighbors or relatively smaller than all of them under consideration. The cost of this check is comparatively low due to the fact that a lot of the sample points will get eliminated in the first few checks of the evaluation process.

An important issue that may be faced is to calculate the frequency of sampling in the images and scale domains that are needed to reliably detect the value of the extrema. Unfortunately, it results that there is no minimum spacing in the samples that will be able to detect all extrema, since the extrema can randomly close together. This can be seen by assuming a white circle over a black background, which would have a single scale of space maximum where the circular positive region of difference-of-Gaussian function matches to the size and location of the circle. For an elongated ellipse, there shall be two maxima near to each end of an ellipse. As these locations of maxima is a continuous function of the images, for some ellipse of intermediate elongation there will be a transition stage from a single maximum to two with the maxima closing arbitrarily to each other near the next transition.

Therefore, we are required to settle for a solution that trades off the efficiency completely. In fact, it might be expected and is confirmed in our experiments, extrema that are closer together are generally unstable to small perturbations of the images. We can determine the

best options experimentally by studying upon a range of sampling frequencies and using them to provide the most reliable results for a realistic simulation of the given matching task.

3.4.1.2 Accurate Key-point Localization:

Once a key point has been found on comparing a pixel to its neighbors, the next task is to perform a detailed fit for the nearby data for location, scale along with ratio of principal curvatures. This information enables points to be rejected that have lower contrast (and are therefore quite sensitive to noise) or are poorly localized along the given edge.

The initial implementation of such approach (Lowe, 1999) simply located key-points at a given location and scale of the central sample point of the analysis. However, recently Brown developed a method (Brown and Lowe, 2002) for fitting a given 3D quadratic function using the local sample points to determine the interpolated locations of the maximum. His experiments showed that it provides a substantial improvement for matching and stability and of the points. His approach uses the facts of Taylor expansion (till quadratic terms) of the scale-space function, D(x, y, σ) shifted so that origin coincides to the sample point:

$$\boldsymbol{D(x) = D + \frac{\partial D^T}{\partial x} x + \frac{1}{2} x^T \frac{\partial^2 D}{\partial x^2} x}$$

where D and its derivatives are calculated at the sample point and x = (x; y; σ)^T is an offset from this point of consideration. The location of the extremum, ^x, is calculated by taking the derivative of this given function with respect to x and setting it as zero, giving

$$\ddot{x} = -\frac{\partial^2 D^{-1}}{\partial x^2} \frac{dD}{dx} \quad (3)$$

As suggested by Brown, the Hessian and derivative of D are calculated by using differences

of neighboring sampling points. Hence, the resulting 3x3 linear system can be solved within a minimal cost. If the offset ^x is larger than 0.5 in any of the dimension, then it would mean that the extremum will lies closer to another different sampling point. In this case, the sample point gets changed and the interpolation shall be performed about that new point. The final offset ^x is to added to the location of its sample point to make the interpolated estimate for these location of the extremum.

The functional value at the extremum, D(^x), will be useful for rejecting unstable extrema with low contrast. Now, it can be obtained by substituting equation (3) in (2), giving

$$D(\ddot{x}) = D + \frac{1}{2}\frac{dD^T}{dx}\ddot{x}$$

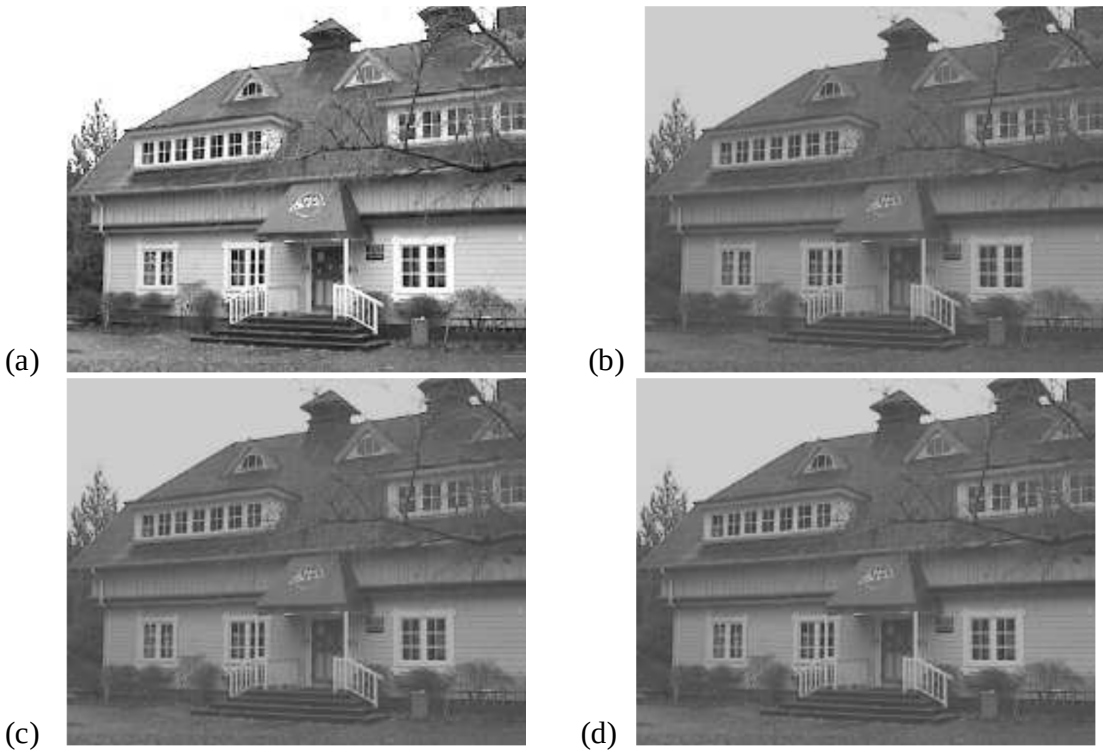


Figure 6: *This figure shows the stages of key-point selection. (a) The 233x189 pixel original image. (b) The initial 832 key point locations are at maxima and minima of the difference-of-Gaussian function. Key-points are indicated as vectors indicating scale, orientation along with location. (c) After applying threshold on minimum contrast, 729 key-points remain. (d) The final 536 key-points that exist following an additional threshold on ratio of principal curvature.*

For the experiments mentioned in this paper, all extrema with a value of jD(^x)j lesser than 0.03 were being discarded (as before, here we have assumed image pixel value lies in the range [0,1]).

Figure shows the effects of key point selection for a natural image. In order to avoid this large amount of clutter, a low-resolution 233 by 189 pixel image shall be used and key points can be shown as vectors indicating the location, scale along with orientation of each of the key point. Figure (a) shows the original image of our analysis, which is shown at a reduced contrast behind the subsequent images. Figure (b) shows the 832 key-points at all detected maxima and minima of the difference-of-Gaussian function, while (c) shows the 729 key-points that remain following

removal of those points with a value of jD(^x)j less than 0.03. Part (d) will be explained in the following subsequent section to follow:

Eliminating edge responses: For stability, it is insufficient to reject the key-points with low contrast. The difference-of-Gaussian function will result a strong response along the edges, even if the locations along the edges are poorly determined. Hence, they are unstable to small amounts of noise.

A poorly defined peak result in the difference-of-Gaussian function will generate a large principal curvature across its edge but having a small one in its perpendicular direction. The principal curvatures can be computed from a 2x2 Hessian matrix, H, computed at the location and scale of the key-point:

$$H = \begin{bmatrix} Dxx & Dxy \\ Dxy & Dyy \end{bmatrix} \quad (4)$$

The derivatives are calculated by taking the differences of the found neighboring sample points. As the eigen values of **H** are proportional to the principal curvatures of D. Borrowing from the methods used by Harris and Stephens (1988), we can always on our part avoid explicitly computing up the eigen values, as here we are only concerned with its ratio. Let α be the chosen eigen value with the largest magnitude and let β be the smaller one. Now, we will compute the sum value of the eigen values using the trace of **H** and its product from the determinant function:

$$Tr(H) = Dxx + Dyy = \alpha + \beta$$

$$Det(H) = DxxDyy - (Dxy)2 = \alpha\beta$$

In an unlikely event that the determinant result is negative, the curvatures will have different signs so the point shall be discarded as not being an extremum value. Let r be the computed ratio between the largest magnitude eigen value and smaller one, so that $\alpha = r\beta$. Then,

$$\frac{\mathrm{Tr(H)2}}{\mathrm{Det(H)}} = \frac{(\alpha+\beta)^2}{\alpha\beta} = \frac{(r\beta+\beta)^2}{r\beta^2} = \frac{(r+1)^2}{r};$$

Which will ultimately depend only on the value of the ratio of the eigen values rather than its individual values. The quantity (r+1)2=r gives its minimum when the two eigen values are taken equal and it increases with factor r. Hence, to check the ratio value of principal curvatures below some threshold r, we are required to only check

$$\frac{\mathrm{Tr(H)2}}{\mathrm{Det(H)}} < \frac{(r+1)^2}{r}$$

This is very easy to compute, with lesser than 20 floating point operations needed in order to test each key point accurately. The experiments done in this paper use a value of r = 10, which further eliminates key points which are having ratio in between the principal curvatures greater than 10. The transition from Figure 5 (c) to (d) clearly shows the effects of this operation.

3.4.1.3 Orientation assignments:

By assigning a consistent orientation to each of the key-point based upon local image properties, the key-point descriptor can be portrayed relative to this orientation and henceforth achieve invariance in image rotation. This approach works in contrast with the orientation invariant descriptors given by Schmid and Mohr (1997), in which each of the image properties are based on a rotationally invariant measure value. The disadvantage of this method of approach is that it limits the descriptors which however can be used and discards the generated image information by not requiring all measures that are to be based upon a consistent rotation concept.

Following experimentation with a number of approaches pertaining to assigning a local orientation, the study showed that following approach was found to produce the most stable results. The scale of a key point is used for choosing the Gaussian smoothed image, L, with the nearest scale, so that all the above computations are performed on a scale-invariant manner. For each image sample available, L(x; y), at this scale for the gradient magnitude, m(x; y) having orientation, θ(x, y) is pre-evaluated using the differences in the pixel:

$$\mathrm{m(x,y)} = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2}$$

$$\theta(x, y) = \tan^{-1}((L(x,y+1) - L(x,y-1)/(L(x+1,y) - L(x-1,y))$$

An orientation histogram is generated from the gradient orientations of available sample points within a enclosed region around the key-points. The orientation histogram is having 36 bins covering the entire 360 degree range of orientations. Each of the values of the sample added to the histogram is weighted by its magnitude gradient and by a Gaussian-weighted having circular window with a σ that is exactly 1.5 times the scale of the key-point.

Peaks in these orientation histogram corresponds to dominant directions for the local gradients. Once the highest peak of the histogram is detected, afterwards any other local peak

that lies within 80% of the highest peak result is used to also create a key-point within that orientation. Henceforth, for locations having multiple peaks for similar magnitudes, there will be existing multiple key-points created at the same location and for the same scale but having a different orientation. Only about 15% of such points are assigned with multiple orientations, but these however contribute significantly towards the stability of matching. Finally, a parabola is made to fit the 3 histogram values closest to each of the peak values to interpolate the peak positions, thereby resulting in better accuracy.

3.4.1.4 The local image descriptor:

The previous operations have been assigned an image location, scale along with orientation to each key-point. These parameters impose a repetitive local 2D coordinate system in which we can describe the local image region. Thereby, provide invariance to these parameters. The next step involves the computation of a descriptor for the image region which are highly distinctive yet are seen as invariant as possible to existing variations, such as changes in illumination or 3D view point.

One such obvious approach would be sampling the local image intensities around the available key point at the appropriate scales, so as to match these using up normalized correlation measure. However, a simple correlation of image patches is extremely sensitive to changes which may cause mis-registration of samples, such as affine or 3D viewpoint changes or non-rigid deformations. A better approach had been demonstrated by Edelman, Intrator along with Poggio(1997). Their proposed representation was subjected upon a model of biological vision, that in particular of complex neurons is primary visual cortex. Hence, these complex neurons respond actively to a gradient at a particular orientation apart from the spatial frequency. Since the location of these type of gradient on the retina is mostly allowed to shift over a very small receptive field instead being precisely localized. Edelman *et al.* virtualized that the function of these types of complex neurons was actually to allow for matching and identification of 3D objects from a range of large viewpoints. They have performed detailed experiments using 3D computer modeling of objects and animal shapes showing that matching gradients would be allowing for a shift in these position and their results in better classification under 3D rotation system. Say for example, recognition accuracy for any 3D object rotated in depth by approximately 20 degrees increased by nearly 35% for correlation of the gradients to 94% using

a complex cell model. Our method of implementation described below was in relation to the context by this idea, but allows us for positional shift using a different method of computational mechanism.

4.1 Descriptor representation: Figure illustrates the calculation of the key-point descriptor. Firstly, the image gradient magnitude and orientations are sampled in and around the key-point location, using the given scale of the key point in selecting the level of Gaussian blur for an image. In order to achieve such orientation**,** the Gaussian weighting function with invariance results, the coordinates of the descriptor and its gradient orientations are to be rotated relative its key-point orientation. For increased efficiency, the gradients are pre-computed at all levels of the pyramid. These are portrayed with small arrows at each and every sample location.

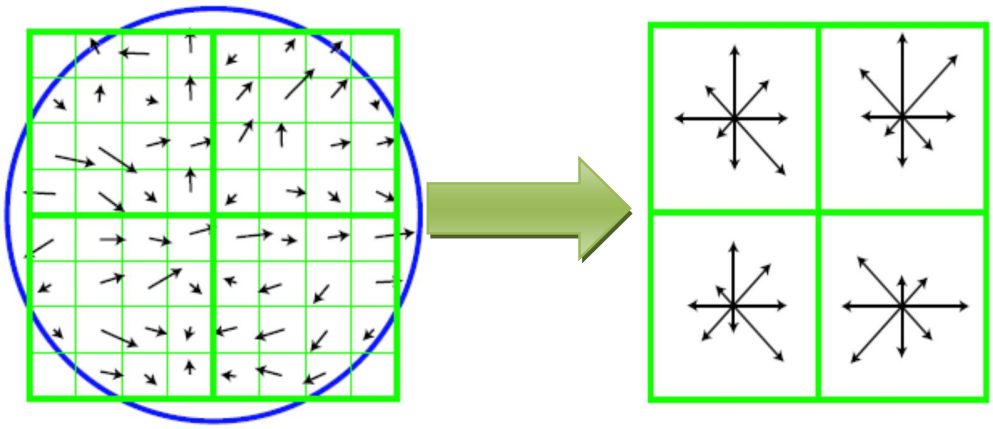


Figure 7*: A key-point descriptor is created by first computing the gradient, magnitude and orientation at each stage of image sample point in a region in and around the key point location which has been shown.*

These are weighted in a Gaussian window, indicated by the overlaid circle as shown. These samples are then arranged into orientation histograms summarizing these contents over 4x4 sub regions as shown with the length of each such arrow corresponding to a sum of the gradient magnitude near that direction within that region. This figure shows about a 2x2 descriptor array evaluated from a 8x8 set of samples whereas the experiments in this document use a 4x4 descriptors computed from a 16x16 sample array value.

A Gaussian weighting function with σ equal to 0.5 times the width of the descriptor window is used to assign a magnitude for each sample point. This is demonstrated with a circular window on the left side, although, the weight falls downwards smoothly. The purpose of this

kind of Gaussian window is to avoid sudden changes in descriptor with small changes in the position of window and giving less emphasis to gradients which are far from the center of the descriptor, since these are mostly affected by wrong registration errors.

The key-point descriptor is shown on the right side. It allows a significant shift in gradient locations by creating orientation histogram over 4x4 sample regions. The figure indicates eight directions for each orientation histogram, with the length of each arrow being the magnitude of its histogram entry. A gradient sample on the left side may shift up to 4 sample locations while still contributing to the same histogram on right side, thereby achieving an objective of allowing for larger positional shifts. It is intended to avoid all boundary affects wherein the descriptor abruptly changes as the sample shifts from being within one histogram to another one or from one orientation to another one. Hence, tri-linear interpolation is applied to distribute the value of all gradient samples into adjacent histogram. In a way, each entry is multiplied by a corresponding value of weight of 1-d for each dimension. Here d is the distance of the sample from the mid value as measured in units of the histogram spacing.

The descriptor is formed using a vector containing the values for all the orientation histogram values, corresponding to the length of the arrows. The figure shows a 2x2 array of orientation histograms, while our experiments below show that the appreciable results are achieved with a 4x4 array with 8 orientation bins in each. Hence, the experiments in this paper use a 4*4*8 =128 element feature vector for each of the key-point. Lastly, the feature vector is modified in order to reduce the effects of illumination changes. Firstly, the vector is normalized to a unit length vector. Changes in image contrast in which each and every pixel value is multiplied by a constant value of gradients by the same constant, so this contrast change shall be canceled by use of vector normalization. A brightness change where a constant is added to its entire images pixel will not change the gradient values, as these are computed incorporating pixel differences. So, the descriptor is invariant to changes in an illumination. However, non-linear illumination change may occur due to camera color saturation or due to its illumination changes that may change 3D surfaces with differing orientations by unequal amounts.

These effects may cause a huge difference in relative magnitudes for little number of gradients, but are less likely to cause any change in the gradient orientations. So, we prefer to reduce the influence of such large gradient magnitudes by keeping the values in the unit feature vector to each be not greater than 0.2 and then normalizing it to unit length. It would mean that matching

the magnitudes overlarge gradients is now not as much important and that too the distribution of orientations involves greater emphasis. The value of 0.2 was found experimentally by using the available images containing differing illuminations for a same 3D object.

3.4.2 MLBP (Multi-scale Local Binary Pattern)

LBP is the most commonly used feature descriptor in computer vision field such as in biometrics, face recognition, pattern recognition etc. It is seen as a powerful tool in texture classification it has further been determined that when LBP is combined with the Histogram of oriented gradients (HOG) descriptor, it improves the detection performance considerably on some datasets.

The LBP feature vector, in its simplest form, is created in the following manner:

1. Divide the examined window into cells (e.g. 16x16 pixels for each cell).
2. For each pixel in a cell, compare the pixel to each of its 8 neighbors (on its left-top, left middle, left-bottom, right-top, etc.). Follow the pixels along a circle, i.e. clockwise or counterclockwise.
3. Where the center pixel's value is greater than the neighbor's value, write "0". Otherwise, write "1". This gives an 8-digit binary number (which is usually converted to decimal for convenience).
4. Compute the histogram, over the cell, of the frequency of each "number" occurring (i.e., each combination of which pixels are smaller and which are greater than the center). This histogram can be seen as a 256-dimensional feature vector.

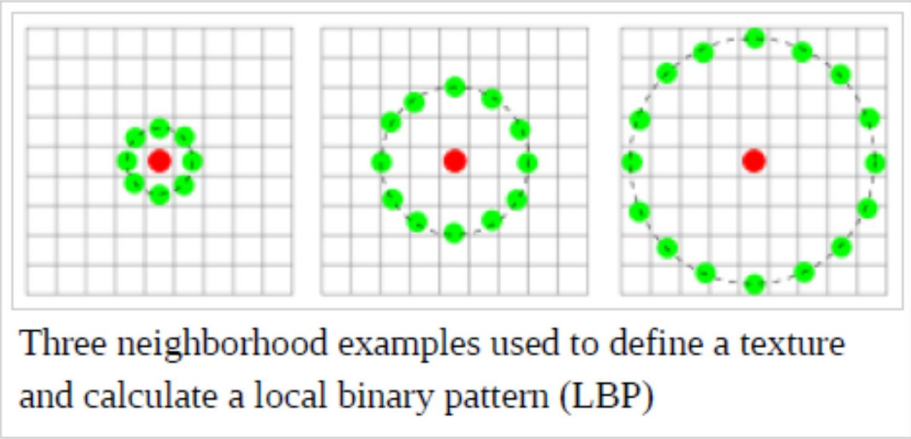


Figure 8: Three neighborhood example

5. Optionally normalize the histogram.

6. Concatenate (normalized) histograms of all cells. This gives a feature vector for the entire window.

The feature vector can now be processed using the Support vector machine or some other machine-learning algorithm to classify images. Such classifiers can be used for face recognition or texture analysis.

Lets take an example on LBP, which is operating on a fixed 3x3 neighborhood pixels as showen below:

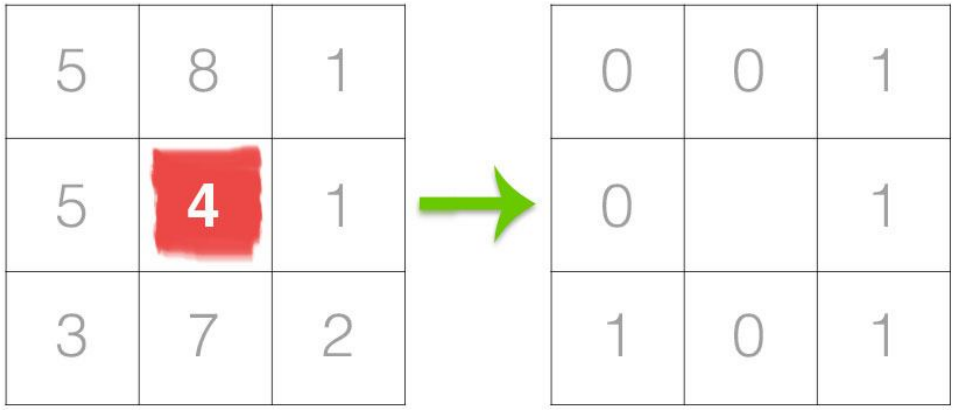


Figure 9. The first step in constructing a LBP is to take the 8 pixel neighborhood surrounding a center pixel and threshold it to construct a set of 8 binary digits.

In the above figure shown, we have taken a center pixel(red in color). We will now threshold it against the 8 neighborhood pixels. If the intensity of the center pixel is greater than or equal to the the intensity of the neighbor, then we will set the value of that neighbor as 1 else it will be set as 0. With the 8 pixels which are surrounding the center pixel , we will be having a total of *2^8= 256* possible combinations of LBP codes.

From this, we have to calculate the LBP value of the center pixel. We have the choice that we can start from any pixel according to our own choice, this can be either clockwise or anti-clockwise, but this way of ordering should be kept *consistent* for each and every pixel of our image as well as all the images in the dataset. We get a 8-bit binary number according to the method mentioned above, this number can be converted to decimal as follows:

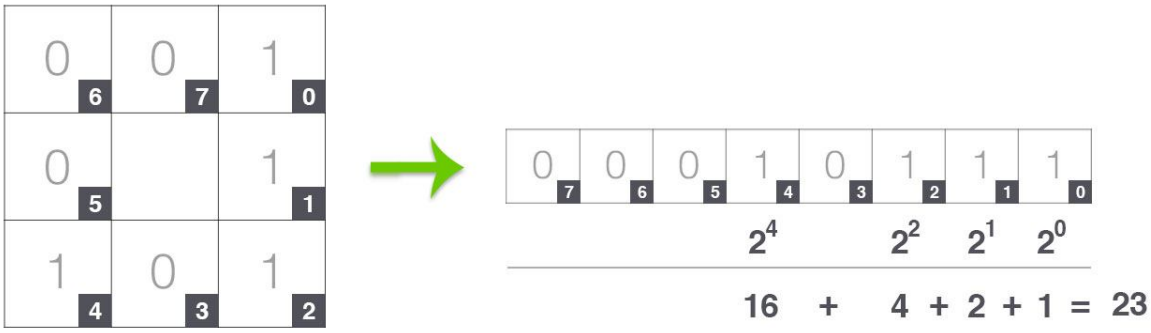


Figure 10: Taking the 8-bit binary neighborhood of the center pixel and converting it into a decimal representation.

In the given example, we have started from the top right corner and moving in clockwise direction gathering the binary numbers along the way. We will then convert the binary number into decimal, which results in decimal value of 23 in our case.

This value of 23 which we get is stored as the output of the LBP 2-D array, which can be seen as below:

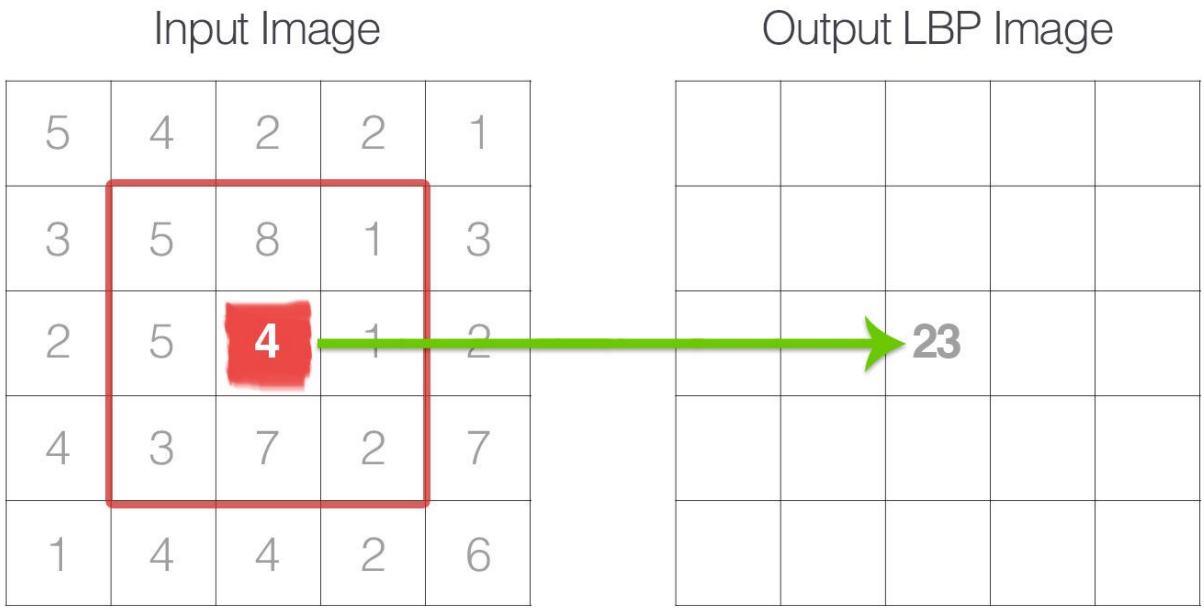
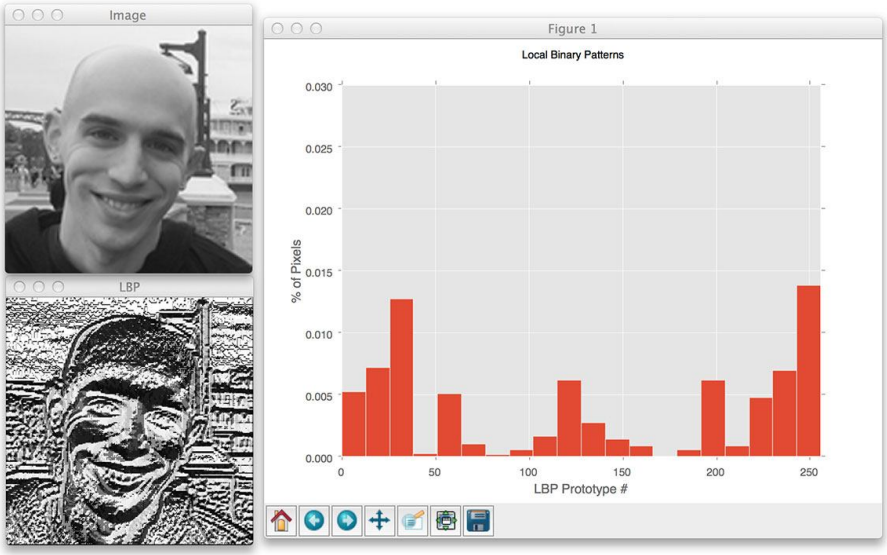


Figure 11 : The calculated LBP value is then stored in an output array with the same width and height as the original image.

This process of firstly taking a threshold, accumulating binary digits, and storing the output decimal value in the LBP array is repeated for each and every pixel in the input image.

Here is an example of computing and visualizing a full LBP 2D array:

The last step of finding LBP is to compute a histogram which is computed over the output LBP array. Since a *3 x 3* neighborhood will have *2 ^ 8 = 256* possible patterns, hence our LBP 2D array will have a *minimum value of 0* and a *maximum value of 255*, therefore allowing us to construct a 256-bin histogram of LBP codes which will be our final feature vector



The primary benefit of this original LBP implementation will be that we can capture extremely fine-grained details in the image. However, capturing details at such a small scale is also the *biggest drawback to the algorithm* — here we cannot capture details at varying scales, only the fixed *3 x 3* scale!

Hence to account for variable neighborhood sizes, new methods were introduced which involved two parameters as follows:

1. The number of points *p* in a circularly symmetric neighborhood to consider.
2. The radius of the circle *r*, which allows us to account for different scales.

Below follows a visualization of these parameters:

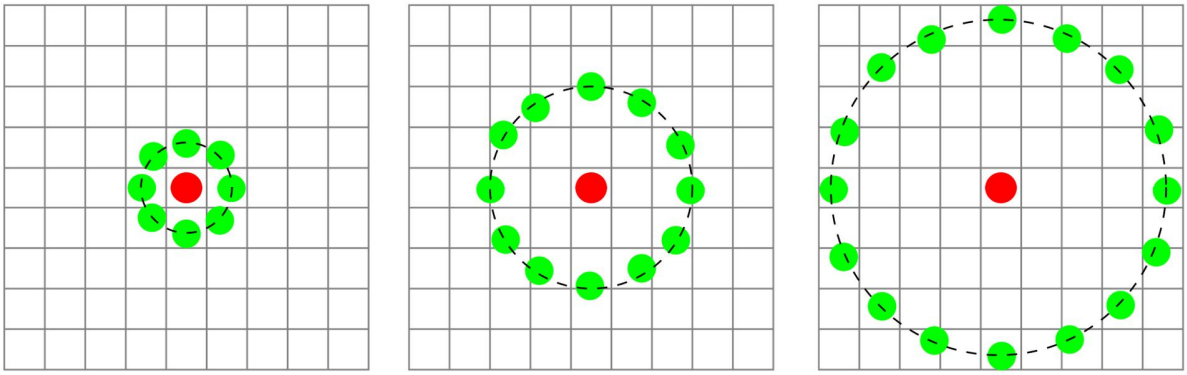


Figure 12. Three neighborhood examples with varying *p* and *r* used to construct Local Binary Patterns.

Lastly, it is very important to consider the concept of LBP ***uniformity***. LBP is considered to be uniform if it has ***at-most*** two *0 to 1* or *1 to 0* transitions. For example, the pattern 00001000 (it has 2 transitions) and 10000000(it has 1 transition) are both considered to be **uniform patterns** since they are having at most two *0 to 1* and *1 to 0* transitions. On the other hand, the pattern 01010010 will *not be* considered a uniform pattern since it has six *0 to 1* or *1 to 0* transitions.

The number of uniform models in a LBP is totally dependent on the number of points *p*. As the value of *p* increase, the dimensionality of our resulting histogram will also increase. For

given number of points *p* in the LBP, there are *p + 1* **uniform patterns**. The final dimensionality of the histogram will thus be *p + 2*,

Uniform LBP just adds an extra level of *rotation and grayscale invariance*, hence they are commonly used when extracting LBP feature vectors from images.

3.4.3 Principal Components Analysis

In principal components analysis (PCA) and factor analysis (FA) one wishes to extract from a set of *p* variables a reduced set of *m* components or factors that accounts for most of the variance in the *p* variables. In other words, we wish to reduce a set of *p* variables to a set of *m* underlying superordinate dimensions.

These underlying factors are inferred from the correlations among the *p* variables. Each factor is estimated as a weighted sum of the *p* variables. The i^{th} factor is thus

$$F_i = W_{i1}X_1 + W_{i2}X_2 + \ldots + W_{ip}X_p$$

One may also express each of the *p* variables as a linear combination of the *m* factors,

$$X_j = A_{1j}F_1 + A_{2j}F_2 + \ldots + A_{mj}F_m + U_j$$

where U_j is the variance that is unique to variable *j*, variance that cannot be explained by any of the common factors.

Goals of PCA and FA

One may do a PCA or FA simply to reduce a set of *p* variables to *m* components or factors prior to further analyses on those *m* factors. For example, Ossenkopp and Mazmanian (*Physiology and Behavior*, *34*: 935-941) had 19 behavioral and physiological variables from which they wished to predict a single criterion variable, physiological response to four hours of cold-restraint. They first subjected the 19 predictor variables to a FA. They extracted five

factors, which were labeled Exploration, General Activity, Metabolic Rate, Behavioral Reactivity, and Autonomic Reactivity. They then computed for each subject scores on each of the five factors. That is, each subject's set of scores on 19 variables was reduced to a set of scores on 5 factors. These five factors were then used as predictors (of the single criterion) in a stepwise multiple regression.

One may use FA to discover and summarize the pattern of inter correlations among variables. This is often called Exploratory FA. One simply wishes to group together (into factors) variables that are highly correlated with one another, presumably because they all are influenced by the same underlying dimension (factor). One may also then operationalize (invent a way to measure) the underlying dimension by a linear combination of the variables that contributed most heavily to the factor.

If one has a theory regarding what basic dimensions underlie an observed event, e may engage in Confirmatory Factor Analysis. For example, if I believe that performance on standardized tests of academic aptitude represents the joint operation of several basically independent faculties, such as Thurstone's Verbal Comprehension, Word Fluency, Simple Arithmetic, Spatial Ability, Associative Memory, Perceptual Speed, and General Reasoning, rather than one global intelligence factor, then I may use FA as a tool to analyze test results to see whether or not the various items on the test do fall into distinct factors that seem to represent those specific faculties.

Psychometricians often employ FA in test construction. If you wish to develop a test that measures several different dimensions, each important for some reason, you first devise questions (variables) which you think will measure these dimensions. For example, you may wish to develop a test to predict how well an individual will do as a school teacher. You decide that the important dimensions are Love of Children, Love of Knowledge, Tolerance to Fiscal Poverty, Acting Ability, and Cognitive Flexibility. For each of these dimensions you write several items intended to measure the dimension. You administer the test to many people and FA the results. Hopefully many items cluster into factors representing the dimensions you intended to measure. Those items that do not so cluster are rewritten or discarded and new items

are written. The new test is administered and the results factor analyzed, etc. etc. until you are pleased with the instrument. Then you go out and collect data testing which (if any) of the factors is indeed related to actual teaching performance (if you can find a valid measure thereof) or some other criterion (such as teacher's morale).

There are numerous other uses of FA that you may run across in the literature. For example, some researchers may investigate the differences in factor structure between groups. For example, is the factor structure of an instrument that measures socio-politico-economic dimensions the same for citizens of the U.S.A. as it is for citizens of Mainland China? Note such various applications of FA when you encounter them.

A Simple, Contrived Example

Suppose I am interested in what influences a consumer's choice behavior when e is shopping for beer. I ask each of 20 subjects to rate on a scale of 0-100 how important e considers each of these qualities when deciding whether or not to buy the six pack: low COST of the six pack, high SIZE of the bottle (volume), high percentage of ALCOHOL in the beer, the REPUTATion of the brand, the COLOR of the beer, nice AROMA of the beer, and good TASTE of the beer. Here are the contrived data, within a short SAS program that does a PCA on them:

```
DATA BEER;
INPUT COST SIZE ALCOHOL REPUTAT COLOR AROMA TASTE;
CARDS;
------- see the data in the file "factbeer.sas"
PROC FACTOR;
```

Checking For Unique Variables

Aside from the raw data matrix, the first matrix you are likely to encounter in a FA is the correlation matrix. Here is the correlation matrix for our data:

	COST	SIZE	ALCOHOL	REPUTAT	COLOR	AROMA	TASTE
COST	1.00	.83	.77	-.41	.02	-.05	-.06
SIZE	.83	1.00	.90	-.39	.18	.10	.03
ALCOHOL	.77	.90	1.00	-.46	.07	.04	.01
REPUTAT	-.41	-.39	-.46	1.00	-.37	-.44	-.44
COLOR	.02	.18	.07	-.37	1.00	.91	.90
AROMA	-.05	.10	.04	-.44	.91	1.00	.87
TASTE	-.06	.03	.01	-.44	.90	.87	1.00

Unless it is just too large to grasp, you should give the correlation matrix a good look. You are planning to use PCA to capture the essence of the correlations in this matrix. Notice that there are many medium to large correlations in this matrix, and that every variable, except reputation, has some large correlations, and reputation is moderately correlated with everything else (negatively). There is a statistic, Bartlett's test of sphericity, that can be used to test the null hypothesis that our sample was randomly drawn from a population in which the correlation matrix was an identity matrix, a matrix full of zeros, except, of course, for ones on the main diagonal. I think a good ole Eyeball Test is generally more advisable, unless you just don't want to do the PCA, someone else is trying to get you to, and you need some "official" sounding "justification" not to do it.

If there are any variables that are not correlated with the other variables, you might as well delete them prior to the PCA. If you are using PCA to reduce the set of variables to a smaller set of components to be used in additional analyses, you can always reintroduce the unique (not correlated with other variables) variables at that time. Alternatively, you may wish to collect more data, adding variables that you think will indeed correlate with the now unique variable, and then run the PCA on the new data set.

One may also wish to inspect the Squared Multiple Correlation coefficient (SMC or R^2) of each variable with all other variables. Variables with small R^2 s are unique variables, not well correlated with a linear combination of the other variables.

Partial correlation coefficients may also be used to identify unique variables. Recall that the partial correlation coefficient between variables X_i and X_j is the correlation between two residuals,

$$\left(X_i - \hat{X}_{i.12..(i)..(j)..p}\right) \text{ and } \left(X_j - \hat{X}_{j.12..(i)..(j)..p}\right)$$

A large partial correlation indicates that the variables involved share variance that is not shared by the other variables in the data set. Kaiser's Measure of Sampling Adequacy (MSA) for a variable X_i is the ratio of the sum of the squared simple *r*'s between X_i and each other X to (that same sum plus the sum of the squared partial *r*'s between X_i and each other X). Recall that squared *r*'s can be thought of as variances.

$$MSA = \frac{\sum r_{ij}^2}{\sum r_{ij}^2 + \sum pr_{ij}^2}$$

Small values of MSA indicate that the correlations between X_i and the other variables are unique, that is, not related to the remaining variables outside each simple correlation. Kaiser has described MSAs above .9 as marvelous, above .8 as meritorious, above .7 as middling, above .6 as mediocre, above .5 as miserable, and below .5 as unacceptable.

The MSA option in SAS' PROC FACTOR [Enter PROC FACTOR MSA;] gives you a matrix of the partial correlations, the MSA for each variable, and an overall MSA computed across all variables. Variables with small MSAs should be deleted prior to FA or the data set supplemented with additional relevant variables which one hopes will be correlated with the offending variables.

For our sample data the partial correlation matrix looks like this:

	COST	SIZE	ALCOHOL		REPUTAT		COLOR
	AROMA	TASTE					
COST	1.00	.54	-.11	-.26	-.10	-.14	.11
SIZE	.54	1.00	.81	.11	.50	.06	-.44

ALCOHOL	-.11	.81	1.00	-.23	-.38	.06	.31
REPUTAT	-.26	.11	-.23	1.00	.23	-.29	-.26
COLOR	-.10	.50	-.38	.23	1.00	.57	.69
AROMA	-.14	.06	.06	-.29	.57	1.00	.09
TASTE	.11	-.44	.31	-.26	.69	.09	1.00
MSA	.78	.55	.63	.76	.59	.80	.68

OVERALL MSA = .67

These MSA's may not be marvelous, but they aren't low enough to make me drop any variables (especially since I have only seven variables, already an unrealistically low number).

Extracting Principal Components

We are now ready to extract principal components. We shall let the computer do most of the work, which is considerable. From *p* variables we can extract *p* components. This will involve solving *p* equations with *p* unknowns. The variance in the correlation matrix is "repackaged" into *p* eigenvalues. This is accomplished by finding a matrix V of eigenvectors. When the correlation matrix R is premultiplied by the transpose of V and postmultiplied by V, the resulting matrix L contains eigenvalues in its main diagonal. Each eigenvalue represents the amount of variance that has been captured by one component.

Each component is a linear combination of the *p* variables. The first component accounts for the largest possible amount of variance. The second component, formed from the variance remaining after that associated with the first component has been extracted, accounts for the second largest amount of variance, etc. The principal components are extracted with the restriction that they are orthogonal. Geometrically they may be viewed as dimensions in *p*-dimensional space where each dimension is perpendicular to each other dimension.

Each of the *p* variable's variance is standardized to one. Each factor's eigenvalue may be compared to 1 to see how much more (or less) variance it represents than does a single variable. With *p* variables there is *p* x 1 = *p* variance to distribute. The principal components extraction will produce *p* components which in the aggregate account for all of the variance in the *p* variables. That is, the sum of the *p* eigenvalues will be equal to *p*, the number of variables. The proportion of variance accounted for by one component equals its eigenvalue divided by *p*.

For our beer data, here are the eigenvalues and proportions of variance for the seven components:

COMPONENT	1	2	3	4	5	6	7
EIGENVALUE	3.31	2.62	.57	.24	.13	.09	.04
PROPORTION	.47	.37	.08	.03	.02	.01	.01
CUMULATIVE	.47	.85	.93	.96	.98	.99	1.00

Deciding How Many Components to Retain

So far, all we have done is to repackage the variance from *p* correlated variables into *p* uncorrelated components. We probably want to have fewer than *p* components. If our *p* variables do share considerable variance, several of the *p* components should have large eigenvalues and many should have small eigenvalues. One needs to decide how many components to retain. One handy rule of thumb is to retain only components with eigenvalues of one or more. That is, drop any component that accounts for less variance than does a single variable. Another device for deciding on the number of components to retain is the scree test. This is a plot with eigenvalues on the ordinate and component number on the abscissa. Scree is the rubble at the base of a sloping cliff. In a scree plot, scree is those components that are at the bottom of the sloping plot of eigenvalues versus component number. The plot provides a visual aid for deciding at what point including additional components no longer increases the amount of variance accounted for by a nontrivial amount.

For our beer data, only the first two components have eigenvalues greater than 1. There is a big drop in eigenvalue between component 2 and component 3. On a scree plot, components 3 through 7 would appear as scree at the base of the cliff composed of components 1 and 2. Together components 1 and 2 account for 85% of the total variance. We shall retain only the first two components.

With SAS one can specify the number of components to be retained by adding NFACT = n, where n is the desired number, to the PROC FACTOR command. One may specify the total amount of variance to be accounted for by the retained components by adding P = p, where p = the proportion or percentage desired. One can specify the minimum eigenvalue for a retained component with MIN = m. I used MIN = 1 for the beer data.

Loadings, Unrotated and Rotated

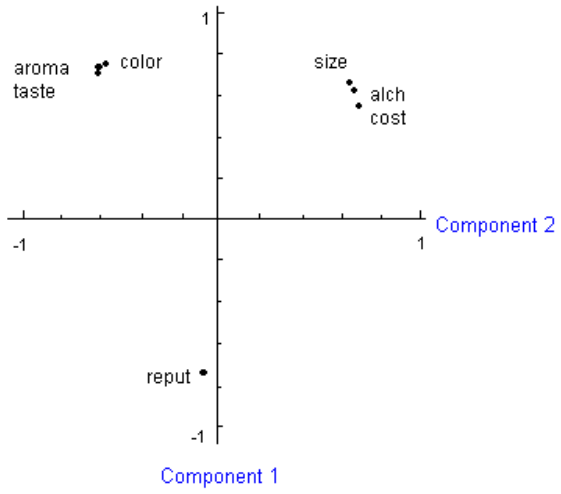
Another matrix of interest is the loading matrix, also known as the factor pattern matrix. This matrix is produced by postmultiplying the matrix of eigenvectors by a matrix of square roots of the eigenvalues. We are retaining only two components, so we shall get a 7 x 2, variables x components, matrix.

Here is the loading matrix for our beer data:

COMPONENT	1	2
COST	.55	.73
SIZE	.67	.68
ALCOHOL	.63	.70
REPUTAT	-.74	-.07
COLOR	.76	-.57
AROMA	.74	-.61
TASTE	.71	-.61

The entries in this matrix, loadings, are correlations between the components and the variables. Since the two components are orthogonal, they are also beta weights, that is, $X_j = A_{1j}F_1 + A_{2j}F_2 + U_j$, thus A_1 equals the number of standard deviations that X_j changes for each one standard deviation change in Factor 1. As you can see, almost all of the variables load well on the first component, all positively except reputation. The second component is more interesting, with 3 large positive loadings and three large negative loadings. Component 1 seems to reflect concern for economy and quality versus reputation. Component 2 seems to reflect economy versus quality.

Remember that each component represents an orthogonal (perpendicular) dimension. Fortunately, we retained only two dimensions, so I can plot them on paper. If we had retained more than two components, we could look at several pairwise plots (two components at a time).

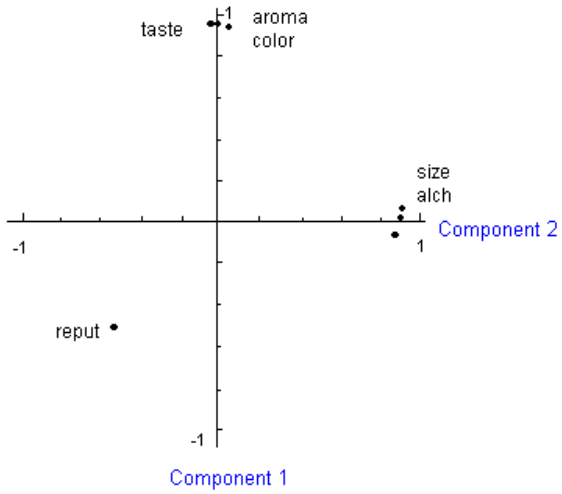


For each variable I have plotted on the vertical dimension its loading on component 1, on the horizontal dimension its loading on component 2. Wouldn't it be nice if I could rotate these axes so that the two dimensions passed more nearly through the two major clusters (COST, SIZE, ALCH and COLOR, AROMA, TASTE). Imagine that the two axes are perpendicular wires joined at the origin (0,0) with a pin. I rotate them, preserving their perpendicularity, so that the one axis passes through or near the one cluster, the other through or near the other cluster. The number of degrees by which I rotate the axes is the angle PSI. For these data, rotating the axes -40.63 degrees has the desired effect.

After rotating the axes I need recompute the loading matrix. This is done by postmultiplying the unrotated loading matrix by a orthogonal transformation matrix. The orthogonal transformation matrix for this two dimensional transformation is

$$\begin{bmatrix} \text{COS PSI} & \text{-SIN PSI} \\ \text{SIN PSI} & \text{COS PSI} \end{bmatrix} = \begin{bmatrix} .76 & .65 \\ -.65 & .76 \end{bmatrix}$$

The rotated loading matrix, with the variables reordered so that first come variables loading most heavily on component 1, then those loading most heavily on component two, is:



COMPONENT	1	2
TASTE	.96	-.03
AROMA	.96	.01
COLOR	.95	.06
SIZE	.07	.95
ALCOHOL	.02	.94
COST	-.06	.92
REPUTAT	-.51	-.53

The rotated loadings plot is shown to the left.

All of the statistics and plots we have discussed so far can be produced by SAS with this command:

PROC FACTOR CORR MSA SCREE REORDER MIN=1 ROTATE=VARIMAX PREPLOT PLOT;

Number of Components in the Rotated Solution

I generally will look at the initial, unrotated, extraction and make an initial judgment regarding how many components to retain. Then I will obtain and inspect rotated solutions with that many, one less than that many, and one more than that many components. I may use a "meaningfulness" criterion to help me decide which solution to retain – if a solution leads to a component which is not well defined (has none or very few variables loading on it) or which just does not make sense, I may decide not to accept that solution.

One can err in the direction of extracting too many components (overextraction) or too few components (underextraction). Wood, Tataryn, and Gorsuch (1996, *Psychological Methods, 1*, 354-365) have studied the effects of under- and over-extraction in principal factor analysis with varimax rotation. They used simulation methods, sampling from populations where the true factor structure was known. They found that overextraction generally led to less error (differences between the structure of the obtained factors and that of the true factors) than did underextraction. Of course, extracting the correct number of factors is the best solution, but it might be a good strategy to lean towards overextraction to avoid the greater error found with underextraction.

Wood et al. did find one case in which overextraction was especially problematic – the case where the true factor structure is that there is only a single factor, there are no unique variables (variables which do not share variance with others in the data set), and where the statistician extracts two factors and employs a varimax rotation (the type I used with our example data). In this case, they found that the first unrotated factor had loadings close to those of the true factor, with only low loadings on the second factor. However, after rotation, factor splitting took place – for some of the variables the obtained solution grossly underestimated their loadings on the first factor and overestimated them on the second factor. That is, the second factor was imaginary and the first factor was corrupted. Interestingly, if there were unique variables in the data set, such factor splitting was not a problem. The authors suggested that one include unique variables in the data set to avoid this potential problem. I suppose one could do this by including "filler" items on a questionnaire. The authors recommend using a random number generator to create the unique variables or manually inserting into the correlation matrix variables that have a

zero correlation with all others. These unique variables can be removed for the final analysis, after determining how many factors to retain.

Explained Variance

The SAS output also gives the variance explained by each component and each variable's communality estimates, both before and after the rotation. The variance explained is equal to the sum of squared loadings (SSL) across variables. For component 1 that is ($.76^2$ + $.74^2$ +...+ $.67^2$) = 3.31 = its eigenvalue before rotation and ($.96^2$ + $.96^2$ +...+ $-.51^2$) = 3.02 after rotation. For component 2 the SSL's are 2.62 and 2.91. After rotation component 1 accounted for 3.02/7 = 43% of the total variance and 3.02 / (3.02 + 2.91) = 51% of the variance distributed between the two components. After rotation the two components together account for (3.02 + 2.91)/7 = 85% of the total variance.

Naming Components

Now let us look at the rotated loadings again and try to name the two components. Component 1 has heavy loadings (>.4) on TASTE, AROMA, and COLOR and a moderate negative loading on REPUTATION. I'd call this component AESTHETIC QUALITY. Component 2 has heavy loadings on large SIZE, high ALCOHOL content, and low COST and a moderate negative loading on REPUTATION. I'd call this component CHEAP DRUNK.

Communalities

Let us also look at the SSL for each variable across factors. Such a SSL is called a communality. This is the amount of the variable's variance that is accounted for by the components (since the loadings are correlations between variables and components and the components are orthogonal, a variable's communality represents the R^2 of the variable predicted from the components). For our beer data the communalities are COST, .84; SIZE, .90; ALCOHOL, .89; REPUTAT, .55; COLOR, .91; AROMA, .92; and TASTE, .92.

The SSL's for components can be used to help decide how many components to retain. An after rotation SSL is much like an eigenvalue. A rotated component with an SSL of 1

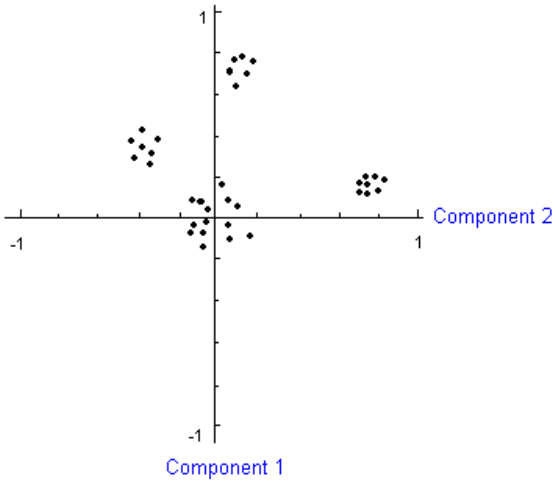
accounts for as much of the total variance as does a single variable. One may want to retain and rotate a few more components than indicated by the MIN = 1 criterion. Inspection of the retained components' SSL's after rotation should tell you whether or not they should be retained. Sometimes a component with an eigenvalue > 1 will have a postrotation SSL < 1, in which case you may wish to drop it and ask for a smaller number of retained components.

You also should look at the postrotation loadings to decide how well each retained component is defined. If only one variable loads heavily on a component, that component is not well defined. If only two variables load heavily on a component, the component may be reliable if those two variables are highly correlated with one another but not with the other variables.

Orthogonal Versus Oblique Rotations

The rotation I used on these data is the VARIMAX rotation. It is the most commonly used rotation. Its goal is to minimize the complexity of the components by making the large loadings larger and the small loadings smaller within each component. There are other rotational methods. QUARTIMAX rotation makes large loadings larger and small loadings smaller within each variable. EQUAMAX rotation is a compromise that attempts to simplify both components and variables. These are all orthogonal rotations, that is, the axes remain perpendicular, so the components are not correlated with one another.

It is also possible to employ oblique rotational methods. These methods do not produce orthogonal components. Suppose you have done an orthogonal rotation and you obtain a rotated loadings plot that looks like this:
The cluster of points midway between axes in the upper left quadrant indicates that a third component is present. The two clusters in the upper right quadrant indicate that the data would be better fit with axes that are not orthogonal. Axes drawn through those two clusters would not be perpendicular to one another. We shall return to the topic of oblique rotation later.


1
-1
1
-1
Component 2
Component 1

CHAPTER FOUR

RESULTS

4 RESULTS

Comparison

For Forensic sketches

	Rank-1 accuracy (%)	Rank-10 accuracy (%)	Rank-50 accuracy (%)
With LFDA	24.09	32.71	47.20
Without LFDA	18.02	21.14	39.00
FaceVACS	2.04	4.08	8.16

Forensic sketch recognition performance using the 159 forensic sketch images (probe set) and 10,159 mug shot images (gallery) will now be presented. In these matching experiments, we use the local feature-based discriminant analysis framework. Our matching uses the sum of score fusion of MLBP and SIFT LFDA, as this was the highest performing method for matching viewed sketches. We evaluated our method using viewed sketches from the CUHK data set This data set consists of 606 corresponding sketch/photo pairs that were drawn from three face data sets: 1) 123 pairs from the AR face database, 2) 295 pairs from the XM2VTS database , and 3) 188 pairs from the CUHK student database. Each of these sketch images was drawn by an artist while looking at the corresponding photograph of the subject.

The performance of matching sketches classified as good and poor. There is a substantial difference in the matching performance of good sketches and poor sketches. Despite the fact that poor sketches are extremely difficult to match.

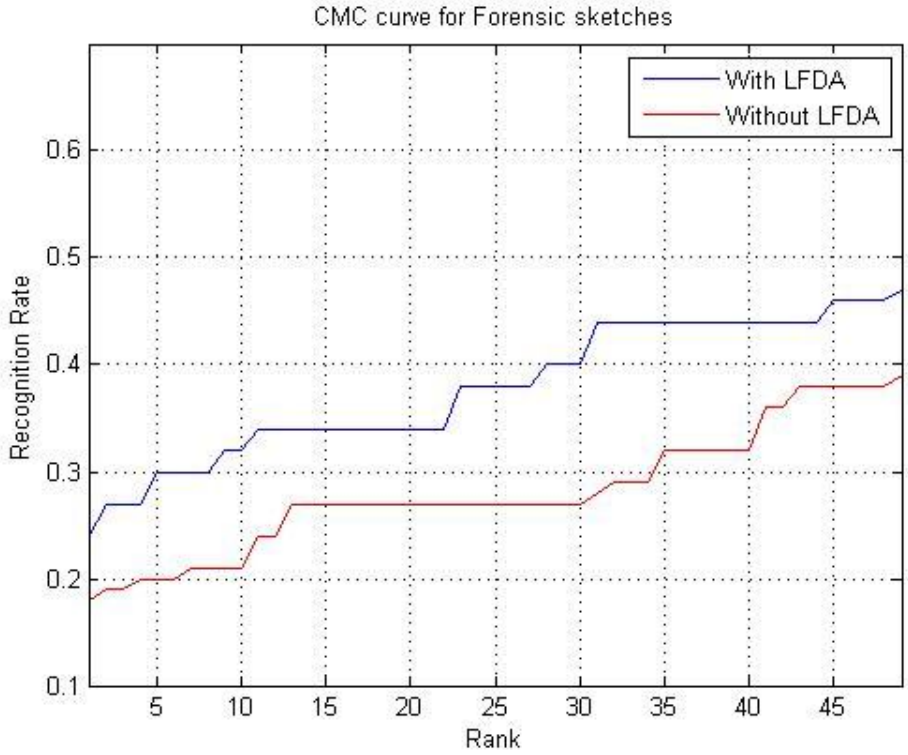


Figure 13 : CMS curve for Forensic sketches

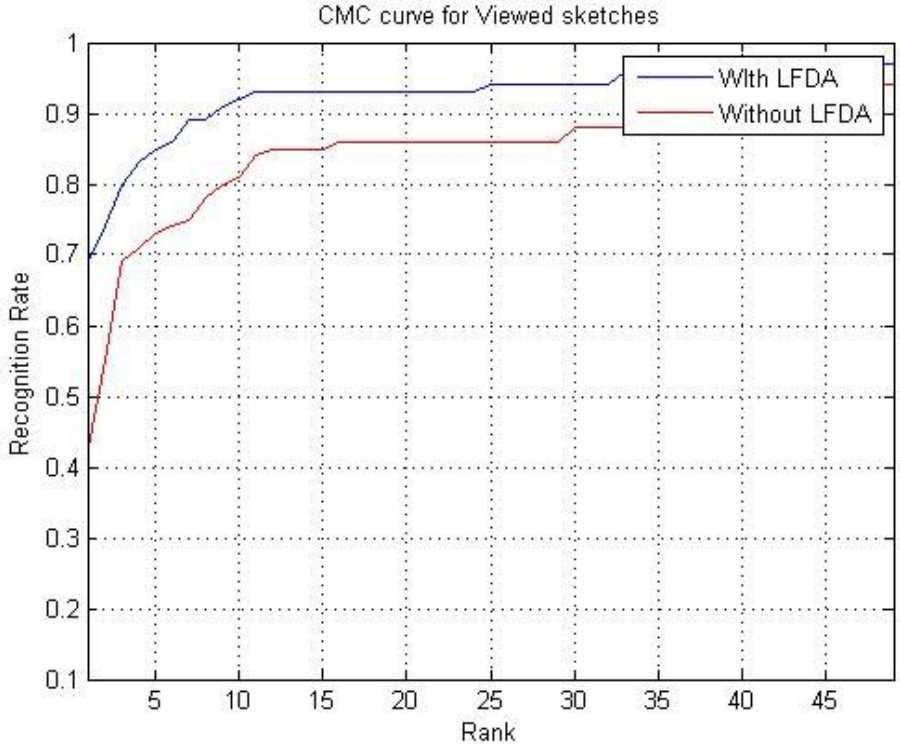


Figure 14 : CMS curve for Viewed sketches

CHAPTER FIVE

CONCLUSION & FUTURE WORK

5 CONCLUSIONS & FUTURE WORK

5.1 CONCLUSION

We have presented methods and experiments in matching forensic face sketches to photographs. Matching forensic sketches is a very difficult problem for two main reasons:

1) Forensic sketches are often an incomplete and poor portrayal of the subject's face.
2) We must match across image modalities since the gallery images are photographs and the probe images are sketches.

One of the key contributions of this method is using SIFT and MLBP feature descriptors to represent both sketches and photos. We improved the accuracy of this representation by applying an ensemble of discriminant classifiers, and termed this framework local feature discriminant analysis. The LFDA feature-based representation of sketches and photos was clearly shown to perform better on a public domain-viewed sketch data set than previously published approaches. Another major contribution is the large-scale experiment on matching forensic sketches. While previous research efforts have focused on viewed sketches, most real-world problems only involve matching forensic sketches.

Continued efforts on matching forensic sketches are critical for assisting law enforcement agencies in apprehending suspects. A larger data set of forensic sketches and matching photographs needs to be collected to further understand the nature and complexity of the problem.

5.2 FUTURE WORK

Continued efforts on matching forensic sketches are critical for assisting law enforcement agencies in apprehending suspects. A larger data set of forensic sketches and matching photographs needs to be collected to further understand the nature and complexity of the problem.

6 BIBLIOGRAPHY

1. David G. Lowe. Distinctive image features from scale-invariant key-points. International journal of computer vision, 60, 2004.
2. http://www.pyimagesearch.com/2015/12/07/local-binary-patterns-with-python-opencv/
3. A.K. Jain, S.C. Dass, and K. Nandakumar, “Soft Biometric Traits for Personal Recognition Systems,” Proc. Int’l Conf. Biometric Authentication, 2004.
4. B. Xiao, X. Gao, D. Tao, and X. Li, “A New Approach for Face Recognition by Sketches in Photos,”
5. X. Wang and X. Tang, “Face Photo-Sketch Synthesis and Recognition,” IEEE Trans. Pattern Analysis and Machine Intelligence.
6. B. Klare and A. Jain, “Sketch to Photo Matching: A Feature-Based Approach,”
7. B. Klare and A. Jain, “Heterogeneous Face Recognition: Matching NIR to Visible Light Images,”
8. T. Ojala, M. Pietika¨inen, and T. Ma¨enpa¨a¨, “Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns,” IEEE Trans. Pattern Analysis and Machine Intelligence,
9. B. Klare, Z. Li, and A. Jain, “On Matching Forensic Sketches to Mug Shot Photos,”
10. S. Raudys and A. Jain, “Small Sample Size Effects in Statistical Pattern Recognition: Recommendations for Practitioners,”
11. X. Tang and X. Wang, “Face Sketch Recognition,” IEEE Trans. Circuits and Systems for Video Technology, vol. 14, no. 1, pp. 50-57, Jan. 2004.