

# Paintings at an Exhibition

## EE368 Group 17 Project Report

Mithun Kamat  
Stanford University  
*mkamat at stanford dot edu*

**Abstract** — An algorithm is developed and implemented to recognize paintings on display at the Cantor Arts Center based on snapshots taken with a camera-phone. The training images were used to create a database of low dimensional fisher images. The image processing was carried out in three steps. First, the training images were gray scaled and down sampled from the high resolution color patterns while preserving the key features. The images were also adjusted for brightness by normalization. Second, the training images were segmented to identify edges corresponding to the frames of the paintings. The location of the frames of the paintings was used to crop and resize the training images to a consistent scale. Finally, the training images were transformed to a lower order database using Fisher Linear Discriminant (FLD) transformation. The test image is compared against the database using Euclidean distance nearest neighbor similarity matching. The algorithm demonstrated reasonably good computational efficiency and accuracy.

This algorithm was developed as part of the project requirements for EE368 class at Stanford University. The algorithm was selected by trading off the computational and accuracy requirements for the project. The code was implemented in Matlab™ version 6.5 using the Image Processing Toolbox.

This work could be useful as part of an electronic museum guide wherein the users would point their camera-phone at a painting of interest and would hear commentary based on the recognition result.

**Keywords** — augmented reality, eigen images, FLD, fisher images, image processing, painting recognition

### I. INTRODUCTION

IMAGE RECOGNITION is a complex problem due to the difficulty in recognizing and matching image features in the presence of variable orientation, illumination, and scale among other things. Additional complexity is involved in the case of

recognition of paintings because no two paintings are fundamentally alike unlike faces where certain facial features (e.g. nose, eyes) tend to be similar between different individuals. The applications for image recognition are numerous and growing. For example, an electronic museum guide could be envisioned where the user would point their camera-phone at a painting of interest and would hear commentary based on the recognition result. Such an application would involve processing of the image and comparing with a known database of library images. The images could vary from paintings, art displays to historical displays such as World War II artifacts. The matched image could provide the user the relevant audio-visual information about the image at hand. Applications of this kind are usually referred to as "augmented reality" applications. Implemented on hand-held mobile devices, they are called "mobile augmented reality."

The task of the project [1] is to focus on the image recognition part of the above applications. Photographs were taken in the European Gallery of the Cantor Arts Center in California. Three photos were taken of each of 33 paintings. The photos were taken with a Nokia N93 with a resolution of 3 mega pixels. The paintings did not represent any certain type and included portraits as well as landscapes. An image recognition algorithm will perform the necessary processing on the "training" images such that a new image from the same set of paintings could be identified accurately in a computationally efficient manner.

### II. ALGORITHM DESCRIPTION

#### A. ALGORITHM SELECTION

A literature search was performed to identify the most relevant image processing algorithm. A large body of literature exists for image processing related to face recognition [2-5]. The most cited algorithms include the eigen-faces or fisher-faces method that adopt Principal

Components Analysis (PCA) and Fisher Linear Discriminant Analysis (FLD). PCA techniques, also known as Karhunen-Loeve methods [2], choose a dimensionality reducing linear projection that maximizes the scatter of all training images. A drawback of this approach is that the scatter being maximized is due not only to the between-class scatter that is useful for classification, but also to the within-class scatter that, for classification purposes, is unwanted information [3]. Fisher's Linear Discriminant (FLD) [6] is an example of a class specific method, in the sense that it tries to shape the scatter in order to make it more reliable for classification. This method selects a transformation of the raw images such a way that the ratio of the between-class scatter and the within-class scatter is maximized. Both FLD and PCA methods are not invariant to image alignment and scaling [7]. Hence, preprocessing would be involved before applying these methods. More recently, a more robust approach has been developed for feature identification. The Scale Invariant Feature Transform (SIFT) algorithm [8] detects key features that are invariant to image scale and rotation, and are shown to provide robust matching across a substantial range of affine distortion, change in 3D viewpoint, addition of noise, and change in illumination.

The simplest algorithm that can be chosen for this particular problem is to do a nearest neighbor search of pair wise correlation of the image with every image in the training database. Obviously such an algorithm will require very high computational space and may be too sensitive to disturbances. Selection of the image processing algorithm involved consideration to the following elements.

- i. The images presented challenges due to low-light conditions. The algorithm had to adjust for illumination and brightness.
- b. The images were taken with a hand held camera and were quite noisy and motion blurs and/or defocus.
- c. The algorithm had to be invariant to rotation or alignment, and scaling as the images were taken from different vantage points and orientation views.
- d. The algorithm had to be computationally inexpensive as the turn around on matching had to be on the order of less than a minute.

A SIFT based image recognition method [8] would automatically cover a – c. However, the algorithm is patented (not readily usable in a commercial setting), and is not computationally efficient in Matlab without the usage of external interfaces such as MEX files, compared with other available methods. On the other hand, FLD approach seemed to be computationally very efficient and academically free to

use. Based upon the discussion provided in [1], FLD seemed more relevant than PCA. However, both those algorithms are not invariant to scaling and alignment and hence preprocessing would be required before applying the FLD transformation.

### B. PROCEDURE

The proposed algorithm is made up of three steps. The step wise procedure is described in Figure 1.

```
0100090000037800000002001c00000000004000000030108
00050000000b0200000000050000000c02f9021a0404000000
2e0118001c000000fb021000070000000000bc0200000000010
2022253797374656d00021a0400008d5c0000a835110070838
23928258b030c02000004000002d01000004000000201010
01c000000fb02ceff0000000000009001000000000440001254
696d6573204e657720526f6d616e0000000000000000000000
00000000000040000002d010100050000000902000000020d
000000320a2d0000000100040000000001a04f70220d41600
040000002d010000030000000000
```

Figure 1: Step-wise Image Processing Algorithm for detection of paintings

In the first step, the raw training images are pre-processed to account for the variations in lighting, noise, blur and scaling. Also, images are down sampled in order to reduce the computational time. In the second step, the images are decomposed initially based upon the PCA method (Karhunen Loeve Transform – KLT) to a reduced order dimension. This step avoids singularity of the within class scatter matrix in FLD. Subsequently, the reduced order images are used to compute the transformation matrix using the FLD transformation approach. The transformation is applied on the training images and the reduced order training images are stored in the database for similarity matching. In the final step, the test images are processed using the similar approach identified in the first step, transformed using the transformation matrix identified in step two, and finally compared with the reduced order image database using the nearest neighbor Euclidean distance similarity matching method. The algorithm is coded in Matlab™ version 6.5. Commands from the Image Processing Toolbox within the Matlab library are used heavily in this algorithm.

### C. IMAGE PRE-PROCESSING

#### 1) Convert to Gray Scale

The training images obtained from the library were in the in the form of RGB array. The image is converted into gray scale with pixel values depicting intensities between 0 to 255. Figure 2 shows one of the original paintings in gray scale. The gray scale image retained the key features of the color images

without compromising the efficiency of the algorithm. The gray scale image has significant computational advantage due to lower storage requirement.

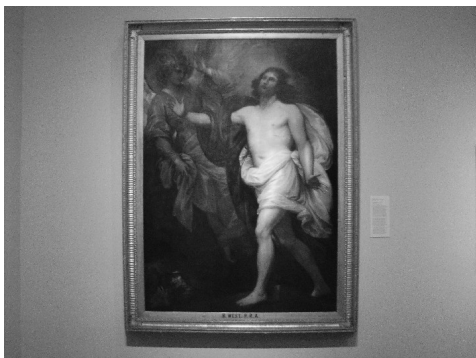


Figure 2: Gray-scale image of “The Resurrection”.

### 2)Down Sample

The gray scale image is then down sampled to 15% of the original image size. This step was performed to avoid the computational burden of a high resolution image on a desktop PC. As will be described below, the KLT method involves the calculation of covariance matrix. Such matrix multiplication operations were found to be computationally extremely expensive especially on a desktop PC. With regard to the objective of this work, it was found that a down sampled image retained the features to a sufficient level provided the correct down sampling scheme is adopted. In this work, the image was down sampled using a bilinear interpolation. Figure 3 shows one of the paintings down sized by a factor of 0.15 using the bilinear interpolation.

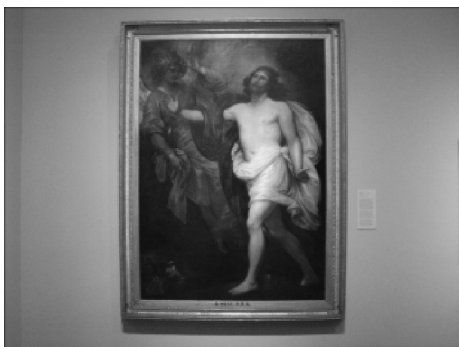


Figure 3: Down sampled version of “The Resurrection”

A visual inspection of Figure 3 shows that the down sampled image has retained the key features of the original image.

### 3)Normalization

In order to eliminate differences with regard to contrast and brightness, the images are normalized using a dynamic range expansion. For example, if the intensity range of a gray scale image is 30 to 190, then the pixel values are first subtracted by 30 and then scaled by a factor of 255/190 in order to achieve normalization from 0 to 255. Figure 4 shows image “The Resurrection” after normalization.

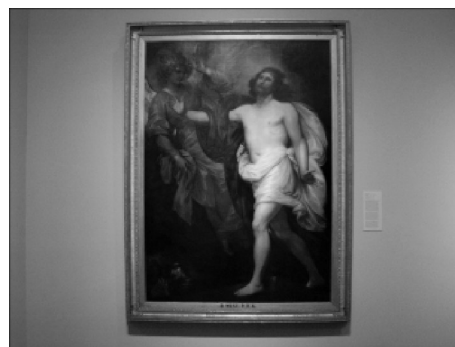


Figure 4: Normalized “The Resurrection” image.

### 4)Isolation of the painting

This is one of the most important steps in the pre-processing algorithm. The original images were taken at different vantage points. See Figure 5a and 5b for two images of the same painting taken from a different distance and angle from each other. Clearly, the images could possibly contain the paintings as well as other background information such as corners in the building, clips of other paintings or objects. First, the key edges in the images are detected using an edge detection algorithm. In this work, the canny edge detection algorithm is used as it optimizes the ability of several types of edge detectors and also demonstrates invariance to noise. The masked image with edges is then dilated using a structuring element of a 3x3 matrix of ones in order to clearly segment the edges of the painting. Note that the edges of the painting are basically the frames of the paintings. Second, the edges corresponding to the frames of the paintings are used to segment the painting itself from the rest of the background. This is performed using a simple segmentation operation using labeling. Figure 5c and 5d show the segmented images after edge detection and labeling.

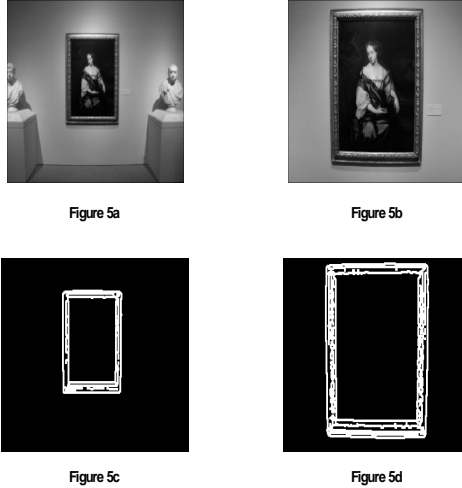


Figure 5: [a, b] Two gray-scale images of the same painting; [c, d] segmented images using Canny edge detection and segmentation using labeling.

#### 5) Image scaling

The location of the four corners of the edges isolated in the previous step are computed and then the gray scaled image from Step 3 is cropped to the size of the box corresponding to the edges. The cropped image is then resized to the original size thereby ensuring that all images are of the same scale and size.



Figure 6a



Figure 6b

Figure 6: [a, b] Cropped and rescaled images of the same painting

Figure 6 shows the scaled images of a painting that were taken from two different angles. This step ensures that the scale invariant weakness of the eigen images algorithm is not exploited. As will be described in the Results section below,

this step proves to be quite robust to changes in dimensions, orientation as well as scale.

### D. PROJECTION TRANSFORMATION

#### 1) Karhunen Loeve Pre-Transform

The Karhunen Loeve Transform method is used to perform a Principal Components Analysis (PCA) on the images. The PCA analysis maximizes the scatter between the images thus extracting distinct features from the image. Assuming a set of images in vector form with each image of size  $M \times N$ ,

$$F = \{f_1, f_2, \dots, f_L\}$$

The KLT transformation can reduce the dimensionality of the matrix  $F$  of size  $M \times N \times L$  to  $J \times L$  by representing the image by  $J$  coefficients.

$$C = WF$$

Where  $W$  is the transformation matrix. The transformation matrix is computed using the Sirovich and Kirby method [9] of computing eigen images corresponding to the  $L_0 \leq L$  largest eigen values according as  $Sv_i$  where  $S$  is the training set matrix,  $S = \{f_1 - \mu, f_2 - \mu, \dots, f_L - \mu\}$ , and  $v_i$  are the eigen vectors of  $S^T S$ . Ideally,  $L_0$  would be chosen to minimize the number of eigen images required to represent the training set. However, since our primary objective is to perform a Fisher transform, we retain the first  $n$  eigen images, where  $n$  represents the number of class of images.

#### 2) Fisher Linear Discriminant Transform

The KL transform maximizes scatter within the linear subspace over the entire image set regardless of the classification task at hand. However, the set of training images includes multiple images of the same paintings. Specifically, three images of 33 paintings are provided in total. The FLD algorithm is designed to exploit the information with each class of painting. The FLD approach, first proposed in 1936, maximizes the between class scatter while minimizing the within class scatter.

Assume  $R_B$  and  $R_W$  as the between class and within class scatter matrices.

$$R_B = \sum_{i=1}^c N_i (\mu_i - \mu)(\mu_i - \mu)^H$$

$$R_W = \sum_{i=1}^c \sum_{f_i \in \text{Class}(i)} (f_i - \mu_i)(f_i - \mu_i)^H$$

The FLD approach involves calculating the eigen values and eigen vectors corresponding to the following solution

$$R_B w_i = \lambda_i R_W w_i$$

Where  $w_i$  are the eigen vectors.

The KL transform was used to generate a reduced order set of 33 eigen images. This ensures that for cases where  $L \ll MN$ , the  $R_W$  matrix is not singular. The eigen values and eigen

vectors of the matrix,  $(R_W^{-1} R_B)$  were computed using singular value decomposition. The eigen values of the FLD approach were visually inspected to decide the reduced order dimension of the FLD transformation matrix.

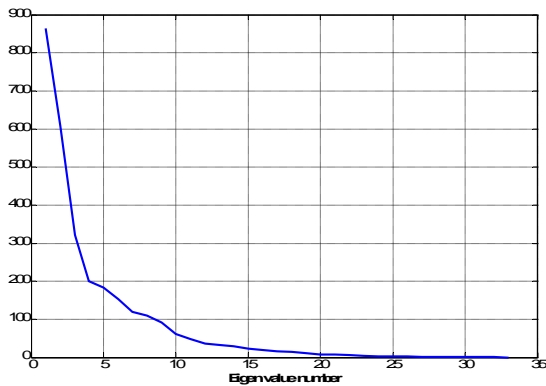


Figure 7: Eigen values of the reduced image. X axis denotes the number of eigen values.

Figure 7 shows the different eigen values of the FLD approach. Clearly, most of the high energy content of the images is condensed in the first 10 eigen values. A sensitivity analysis was performed by varying the dimension of the transformation matrix from eight to 15, and it was found that the dimension of 10 is sufficient to preserve the high energy characteristics while maintaining algorithm accuracy. Therefore, the first 10 sorted eigen vectors constitute the FLD transformation matrix.

### 3) Database Generation and Storage

The PCA and FLD transforms are applied in series to images that were pre-processed using steps outlined in Step 1. Since the FLD transform is applied on the mean images within each class, the final size of the database of images was 10x33 where 10 represents the dimension of the eigen (fisher) images and 33 represent the number of types of paintings in the database. Thus, the FLD transform was successful in reducing the dimension of an image of size  $MN$  to a vector of 10 fisher coefficients.

## E.SIMILARITY MATCHING

### 1) Test Image Transformation

The test image is subjected to the preprocessing outlined in the Step 1 of the previous section. The processed image is projected on to a lower dimensional space of size 10 by convolution with the FLD transformation matrix

### 2) Minimal Euclidean distance matching

The lower dimensional vector of test image is compared to every image in the training database by computing the Euclidean distance between the test image and every training image. The values are sorted based upon the nearest neighbor distance.

$$\xi^2 = \min(\|q - p_k\|^2)$$

Where  $q$  is the reduced test image, and  $p_k$  is the reduced training image in the database,  $k = 1$  to  $L$  (number of images)

### 3) Output

A two step approach is utilized to match the painting. First the nearest neighbor value is used to detect a possible matching candidate in the database. Next, the nearest neighbor Euclidean distance is compared with a pre-set value of  $\theta$ .

$$\theta = \frac{1}{2} \max(\|p_i - p_j\|)$$

For  $i, j = 1$  to  $L$  ( $L =$  number of images)

The following rejection system is applied.

if  $\xi > \theta$ , painting not recognized  
else, painting recognized

The following functional interface provided the results

$[painting\_title] = identify\_painting(img)$

Where

$img$ : height-by-width-by-3 uint8 array representing a 24-bit RGB image obtained through  $imread()$ .

$painting\_title$ : A character array indicating the title of the dominant painting contained in the input image.

## III.RESULTS

The algorithm was tested in a series of three steps. First, the training images used in the algorithm were used as test images to test the self-reliability and computational speed. As expected, the algorithm successfully identified 100% of the input images. Each call to the function approximately took 33 seconds for processing. Second, ten of the training images were altered by inducing a combination of noise, blur and alignment changes. The algorithm successfully detected 95%

of the input images. The images that were not successfully recognized were the ones that were aligned differently (vertical angle was altered). This provided feedback on the weakness of the approach to changes in image orientation. Finally, to test the limits of the algorithm, a set of 38 test images [10] completely different from the training database were tested. Some of the test images indeed were very challenging because the camera used to capture the image was a low resolution camera and the vantage points were deliberately chosen to be extremely stringent.



Figure 8: [a] Gray-scale test image (courtesy [xx]); [b] Processed test image using the algorithm

Figure 8 shows a test image and the output of the preprocessing of the test image. The algorithm accurately detected 78% of the test images. Also, the average time for the computation of the results was only 3.35 seconds. The difference in the computational speed between the test images and the training images was primarily attributed to the size of the input image. The test images were generated with a low resolution camera and hence were of smaller size. The result showed that the algorithm indeed was robust to most of the variations in the input images including blurring, noise, scaling, and brightness. A study of the cases where the algorithm failed to detect the test image revealed the common theme of the issue with detection of unaligned images.

#### IV.CONCLUSIONS

An image processing algorithm was developed as part of the project requirements for EE368 class at Stanford

University. The algorithm processed images of paintings of different types, scale and orientation. The images were transformed into a reduced dimensional space using Fisher Linear Discriminant (FLD) transformation. The algorithm demonstrated reasonably good computational efficiency and accuracy. The algorithm suffered from accuracy issues related to image orientation. The success rate of the algorithm in matching images needs to be improved further before the code could be implemented in commercial software for image recognition of paintings. Future work has been identified to improve the features of the algorithm.

#### V.LIMITATIONS AND FUTURE WORK

The algorithm was not designed to be fully robust against image alignment. FLD and PCA based methods are known to be susceptible to scale invariant issues. The testing of the algorithm against the test images revealed this flaw. There are two proposals for future work with this algorithm. First, the pre-processing of the images before the FLD transformation should involve some sort of alignment of the images. One method is to combine SIFT based feature detection with fisher images. One can use the SIFT based algorithm to detect key features including the scale and the orientation. This information could be used to detect the major orientation of the image and the image could then be rotated to adjust for the orientation. Second, the computational speed of the algorithm could be improved by creating C code for some of the major sub-routines (e.g. scaling sub routine). The C code could be converted into a Matlab MEX file, which will improve the computational efficiency of the overall algorithm.

#### ACKNOWLEDGMENT

The author would like to thank Prof. Bernd Girod and EE368 Spring 2006-07 Teaching Assistants for their help throughout the course and especially during trouble shooting. The author also acknowledges Mr. Jacob Mattingley, a fellow student, for graciously sharing test images from the Canton Arts Center. Those images were used to evaluate the limits of the algorithm.

#### REFERENCES

- 1] B. Girod, *Lecture notes for EE368: Digital Image Processing*, Spring 2007, Stanford University.
- 2] Javier Ruiz-del-Solar, and Pablo Navarrete, "Eigenspace-based face recognition: A comparative study of different approaches", *Applications and Reviews*, vol. 35, no. 3, Aug. 2005.
- 3] P. Belhumeur, J. P. Hespanha, and D. Kriegman, "Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection", *IEEE transactions on PAMI*, July 1997
- 4] R. Chellappa, C. Wilson, and S. Sirohey, "Human and machine

- recognition of faces: A survey”, *Proceedings of the IEEE*, vol. 83, no. 5, pp. 705-740, 1995.
- 5] A. Samal and P. Iyengar, “Automatic recognition and analysis of human faces and facial expressions: A survey”, *Pattern recognition*, vol. 25, pp. 65-77, 1992.
  - 6] R. Fisher, “The use of multiple measures in taxonomic problems”, *Ann. Eugenics*, vol. 7, pp. 179-188, 1936.
  - 7] M. Turk and A. Pentland, “Eigenfaces for recognition,” *J. Cogn. Neurosci.* vol. 3, no. 1, pp. 71–86, 1991.
  - 8] David G. Lowe, “Distinctive image features from scale-invariant feature keypoints”, *International journal of computer vision*, 2004.
  - 9] L. Sirovitch and M. Kirby, “Low-dimensional procedure for the characterization of human faces”, *J. Optical Soc. of America A*, vol. 2, pp. 5190524, 1987.
  - 10] J. Mattingley, test images from the Canton Arts Center, clicked on personal camera, Spring 2007.