Bilkent University

Department of Computer Engineering

Senior Project

Who do you resemble?

Low Level Design Report

Group Members

Merve Soner

Merve Yurdakul

R. Baturalp Torun

Sedef Özlen

Supervisor: Pinar Duygulu Şahin

Jury Members: Selim Aksoy, H. Altay Güvenir

February 25, 2008

Table of Contents

1 Introduction

1.1 Purpose of the system

A facial recognition system is a computer application for automatically identifying or verifying a person from a digital image or a video frame from a video source. One of the ways to do face recognition is by comparing selected facial features from the image and a facial database. It is typically used in security systems and can be compared to other biometrics such as fingerprint or eye iris recognition systems. Popular recognition algorithms include eigenface, fisherface, the Hidden Markov model, and the neuronal motivated dynamic link matching. [7] A newly emerging trend, claimed to achieve previously unseen accuracies, is three-dimensional face recognition. Another emerging trend uses the visual details of the skin, as captured in standard digital or scanned images. Our aim is to find the most accurate working method among these and improve it, by applying it to find the similarities between celebrities. Therefore, we have implemented Lowe's Sift Method "finding match point" algorithm in order to find the interest points and then match those points from a given image among the other images from the database. We have used several constraints in the implementation to eliminate the false matchings. Through this report we will give the implementation details of the implemented algorithm, as well as the results and the efficiency of the algorithm with charts, pictures, etc.

1.2 Definitions

This section contains the definitions of the terms or concepts that are used in the report.

Eigenface: Eigenfaces are a set of eigenvectors used in the computer vision problem of human face recognition. The approach of using eigenfaces for recognition was developed by Matthew Turk and Alex Pentland beginning in 1987, and is considered the first facial recognition technology that worked. [5]

Eigenvector: Eigenvector of a given linear transformation is a vector which is multiplied by a constant called the eigenvalue during that transformation. The direction of the eigenvector is either unchanged by that transformation (for positive eigenvalues) or reversed (for negative eigenvalues). [4]

Eucledean distance: In mathematics, the Euclidean distance or Euclidean metric is the "ordinary" distance between two points that one would measure with a ruler. [6] The Euclidean distance between points P = *($p_1, p_2, \ldots, p_n$)* and *Q* = *($q_1, q_2, \ldots, q_n$)*, in Euclidean *n*-space, is defined as:

$$\sqrt{}((p_1-q_1)^2+(p_2-q_2)^2+\cdots+(p_n-q_n)^2)=\sqrt{}\sum_{i=1}^{n}(p_i-q_i)^2$$

Mean Average Precision: The precision and recall are based on the whole list of documents returned by the system. Average precision emphasizes returning more relevant documents earlier. It is average of precisions computed after truncating the list after each of the relevant documents in turn:

$$\operatorname{Ave}P=\frac{\sum_{r=1}^{N}(P(r)\times\operatorname{rel}(r))}{\text{number of relevant documents}},$$

where *r* is the rank, *N* the number retrieved, *rel()* a binary function on the relevance of a given rank, and *P()* precision at a given cut-off rank. If there are several queries with known relevancies available, the *mean average precision* is the mean value of the average precisions computed for each of the queries separately. [11]

Facial Recognition System (fisherface): A facial recognition system is a computer application for automatically identifying or verifying a person from a digital image or a video frame from a video source. One of the ways to do this is by comparing selected facial features from the image and a facial database**.** [12]

Hidden Markov model: A hidden Markov model (HMM) is a statistical model in which the system being modeled is assumed to be a Markov process with unknown parameters, and the challenge is to determine the hidden parameters from the observable parameters. The extracted model parameters can then be used to perform further analysis, for example for pattern recognition applications. A HMM can be considered as the simplest dynamic Bayesian network. [13]

Interest point detection: Interest point detection is a recent terminology in computer vision that refers to the detection of interest points for subsequent processing. An interest point is a point in the image which in general can be characterized where it has a clear, preferably mathematically well-founded, definition, a well-defined *position* in image space, the local image structure around the interest point is rich in terms of local *information contents*, such that the use of interest points simplify further processing in the vision system. It is *stable* under local and global perturbations in the image domain, including deformations as those arising from perspective transformations (sometimes reduced to affine transformations, scale changes, rotations and/or translations) as well as illumination/brightness variations, such that the interest points can be reliably computed with high degree of *reproducibility*. Optionally, the notion of interest point should include an attribute of *scale*, to make it possible to compute interest points from real-life images as well as under scale changes. [15]

Match point detection: Match point detection is the term used in computer vision that refers to the detection of the matching of the interest points between the given two images.

Three-dimensional face recognition: Three-dimensional face recognition (3D face recognition) is a modality of facial recognition methods in which the three-dimensional geometry of the human face is used. It has been shown that 3D face recognition methods can achieve significantly higher accuracy than their 2D counterparts, rivaling fingerprint recognition. [21]

2 Background

2.1 Interest point

For recognition and retrieval, a good representation is provided by the local image features. These global features are sensitive to variations of images such as pose, viewpoint and illumination. For the recognition of the objects and faces, SIFT algorithm is selected as one of the successful local features. This feature is distinctively invariant among other features. It is a usable technique in reliable matching between different images of a face, an object or a scene. The image scale and rotation are the invariant properties of the features which are considered as the most powerful. On the other hand the illumination and 3D camera viewpoints are considered as the partial invariance of the same features. In the investment of the face authentication, the SIFT approach is the most applicable.

2.2 Geometric constraints

This constraint assumes that the correct matches between images will appear at the similar positions. For example, right eye is expected to be around middle-right of a face in the normalized images.

In this constraint, a set of images is constructed which includes 5 images for each of 10 people. Then for each comparison correct and false matches are assigned in order to use them as training samples running on a quadratic Bayes normal classifier. Then these matching points will be classified as correct or false according to its geometrical distance. The geometrical distance corresponding to i^{th} assignment refers to $\sqrt{}X^2 + Y^2$ where

$$X = \frac{locX\,(i)}{sizeX\,(image1)} - \frac{locX\,(match\,(i))}{sizeX\,(image2)},$$

$$Y = \frac{locY\,(i)}{sizeY\,(image1)} - \frac{locY\,(match\,(i))}{sizeY\,(image2)},$$

and *locX* and *locY* hold *X* and *Y* coordinates of the feature points in the images, *sizeX* and *sizeY* hold *X* and *Y* sizes of the images and *match*(*i*) corresponds to the matched keypoint in the second image of the i^{th} feature point in the first image [18].

2.3 Unique match constraints

After geometrical constraint is applied, some false matches may still exist. Generally, multiple matches of an interest point and one-way matches cause these false matches. Unique match constraint can eliminate these false matches by ensuring that if an interest point A has a matching with another interest point B in the other image then interest point B also has a matching with interest point A [18].

This method is an alternative for finding similarities between images. In the following steps of the project we will implement this algorithm to see its efficiency. We will try to decrease the database query and retrieval time as much as possible and give the best similarities.

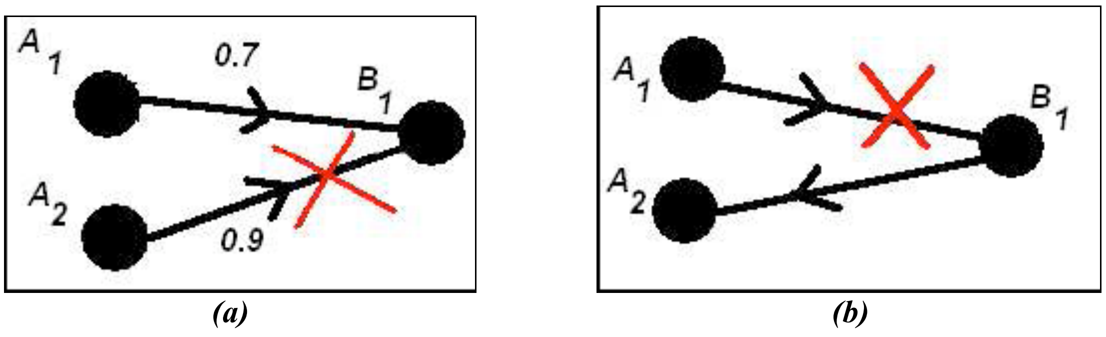


Figure 1: Shows in part (a) that if there are two matches from two point of one image to one point of the second image the point with a longer distance will be eliminated. In part (b) if the match point from the first image to the second image is not matched from the same point of second image to the same point of the first image, then that match is eliminated.

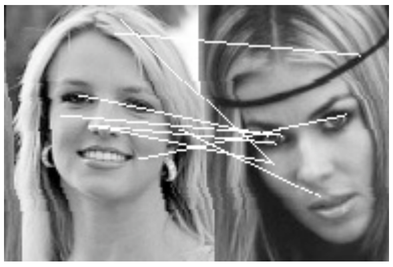
3 Current System

As we mentioned in our high level report, we use Lowe's SIFT algorithm to find interest points in the face images. Then, we use these interest points to find similarities between face images. The algorithm we implement compares the keypoint vectors of the interest points to find matches between face images.

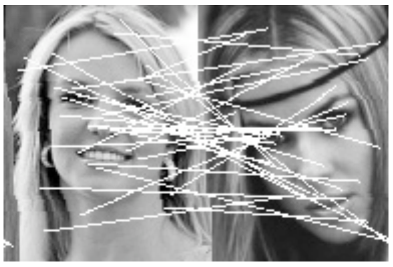
While finding these matches we have decided to apply some constraints to eliminate false and irrelevant matches. One of these constraints is geometric constraint. This constraint is used to compare the location of the interest points in the images with respect to the scale of the images. With this constraint, we tried to find matches in the similar locations of the images so that the wrong matches which match different parts of the faces can be eliminated. For example, this constraint is supposed to eliminate the match which found a similarity between eye of one image and nose of the other.

As mentioned above, geometric distance constraint finds a distance value between two interest points and compares it with a threshold value. This threshold value is found by taking the mean of distances between the match points of the images in the face data set and randomly chosen face images. After this comparison, distances exceeding the threshold value are eliminated. However, this elimination did not work as we expected because the matches between different locations still exist as shown in Figure 2 parts *a* and *b*. Then we change the threshold value with smaller values in order to eliminate existing false matches. Since the threshold value changes from image to image this modification also did not work. In the following steps of the project, we aim to find a better way to implement the geometric constraint by taking the angles between match points into account or finding a better threshold that applies to all face images.

Another constraint that we apply to eliminate false matches is the unique match constraint. This ensures that the found match (interest point B in image 2) for interest point A in image1 is also a match point found for interest point B in image2. Thus, one-way assignments are eliminated. Also, we prevented several matches to same interest point i.e. the multiple assignments are eliminated by choosing the match with the minimum distance.



(a)



(b)

Figure 2: Part (a) shows the matches between two images where both geometric distance and unique match constraints are used in the implementation. In part (b) only the unique match constraint is used not the geometric distance constraint.

4 System test results

In order to test the algorithm developed we constructed a sample database consisting of 160 images of 12 famous people. We gathered these images from our dataset which contains 31000 images. We started with a small part of this database to decrease the testing time. After finding an efficient and good working algorithm, the final system will work with the original dataset.

To see how well the algorithm finds matches of a given image, we run the program and created a table for randomly chosen 7 people in the sample dataset to see how efficient and accurate the program is. These tables have entries for the images in the dataset as seen in Figure 3 and 4. Every row in each table represents the results for one face image comparison with the other images in the sample database.

First column gives the name of the image that we run the program the second column gives the results of the program for the specific picture that we run the program on it. Finally the last column shows the **mean average precision** of the result. We expect that the program should return the images of the same person for the image we have given to the system. The match order goes from best to worst. Therefore 1 represents the best match in the second column and the 160 represent the worst match for the given image among the sample database. The "+" sign indicates the matches of other images of the same person, where as "." indicates the matches for the given image with other people images in the database. By observing the gathering of the "+" signs we can have a conclusion on the output of the program. If the "+" cloud is gathered at the beginning then it means that the program has considerably good result but if it is the opposite then the program does not make proper matches. As you will see in Figure 3 and 4 we have formed two different tables for the same person with the same sample images of that person. One of them represents the results of the matching program with only unique match constraint where as the other table represents the results for both unique match and geometric distance constraints. There are more tables represented in Appendix A for different people from the sample database.

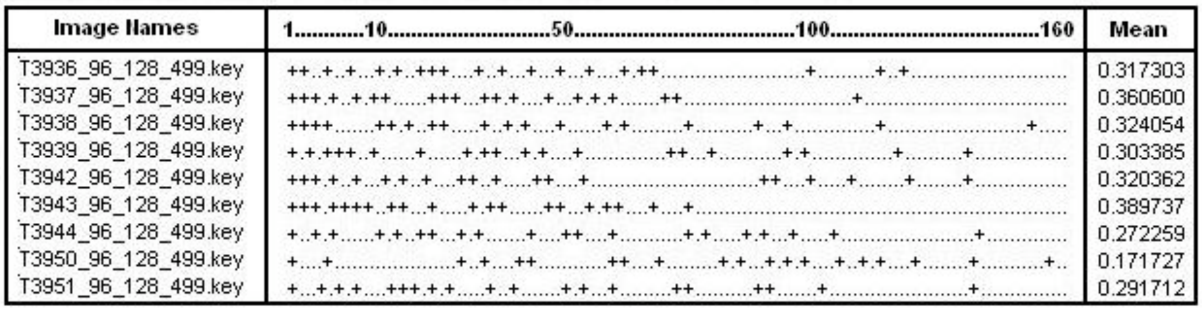


Image Names	1..........10..........................50....................................100..................................160	Mean
T3936_96_128_499.key	++..+..+...+.+..+++....+..+...+...+...+....+.++.........................+..........+..+............................	0.317303
T3937_96_128_499.key	+++.+..+.++......+++...++.+....+...+.+.+.......++............................+....................................	0.360600
T3938_96_128_499.key	++++.......++.+..++.....+..+.+....+......+.+.........+..........+...+.............+........................+.....	0.324054
T3939_96_128_499.key	+.+.+++..+......+......+.++...+.+....+.............++...+...........+.+.............+..........+................	0.303385
T3942_96_128_499.key	+++.+..+...+.+..+....++..+.....++....+...........................++....+.....+........+........+................	0.320362
T3943_96_128_499.key	+++.++++..++...+.....+.++......++...+.++....+....+..	0.389737
T3944_96_128_499.key	+..+.+......+.+..++...+.+.......+....++....+...........+.+.....+.+...+.....+.......................+..............	0.272259
T3950_96_128_499.key	+....+.....................+..+....++...........++....+.........+.+...+.+.+....+..+.+....+........+...........+..	0.171727
T3951_96_128_499.key	+...+.+.+....+++.+.+.....+..+.......+.+...+.........++..........++.......+........................+...............	0.291712

Figure 3: Table shows sample images for Britney Spears and their result for the program that runs only with unique match constraint.

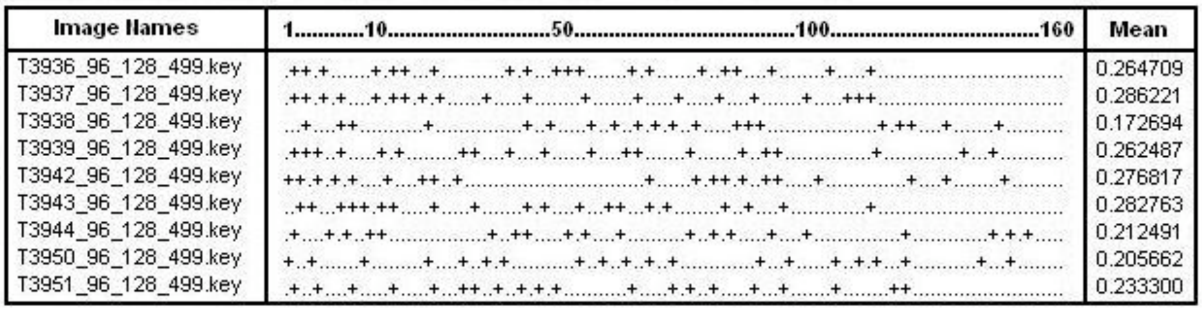


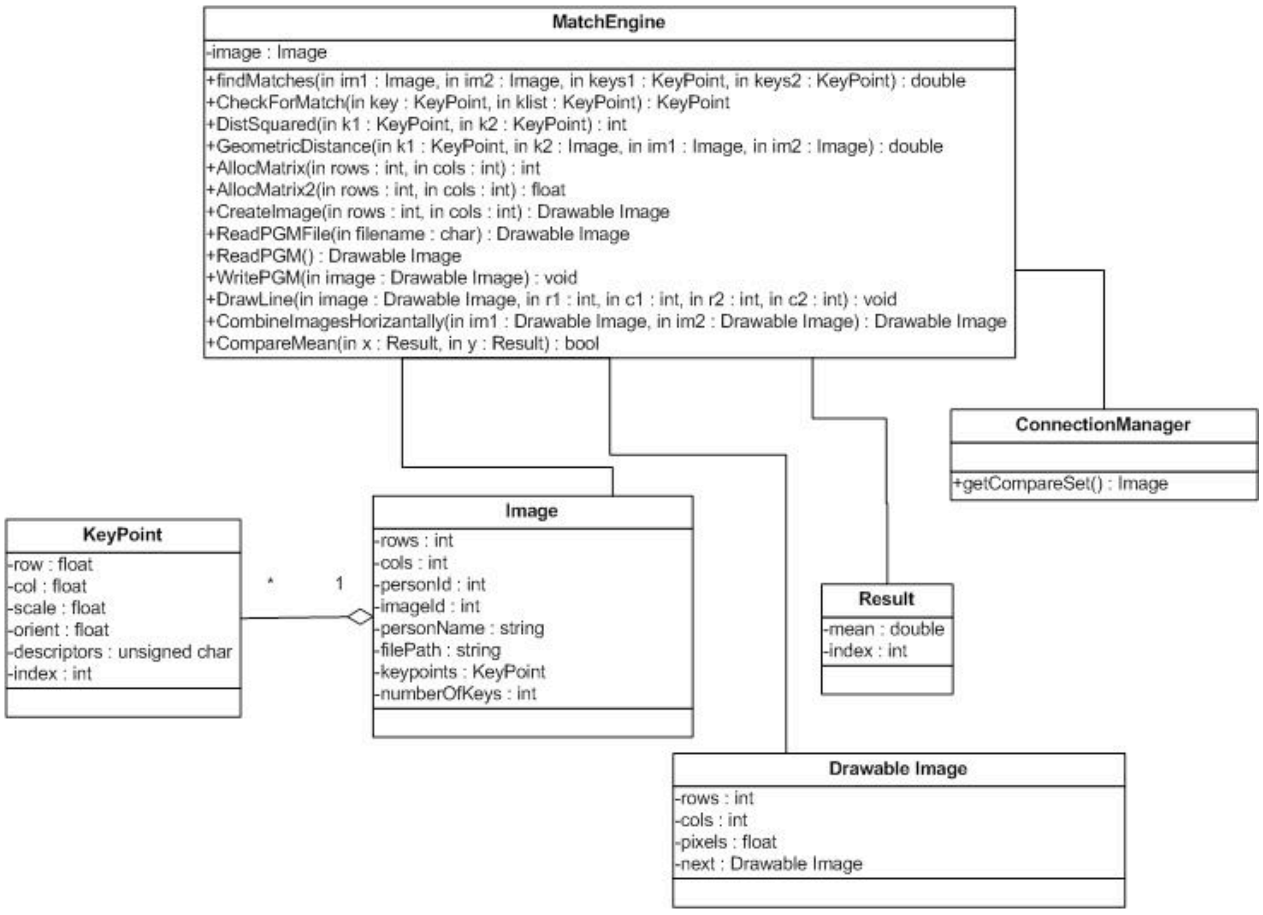
Image Names	1..........10..........................50....................................100..................................160	Mean
T3936_96_128_499.key	.++.+.......+.++...+...........+.+...+++.......+.+.......+..++....+........+.....+................................	0.264709
T3937_96_128_499.key	.++.+.+....+.++.+.+......+......+.........+.......+.....+.....+....+.......+.....+++...............................	0.286221
T3938_96_128_499.key	...+....++...........+...............+..+.....+..+..+.+.+..+.....+++..................+.++....+.......+..........	0.172694
T3939_96_128_499.key	.+++..+.....+.+.........++....+....+......+....++.......+.......+..++..............+.............+...+...........	0.262487
T3942_96_128_499.key	++.+.+.+....+....++..+.............................+......+.++.+..++.....+..............+....+........+.........	0.276817
T3943_96_128_499.key	..++...+++.++.....+......+.......+.+....+...++...+.+..........+..+....+.............+................................	0.282763
T3944_96_128_499.key	.+....+.+..++................+..++.....+.+....+..........+..+.+.....+....+...............+............+.+.+.....	0.212491
T3950_96_128_499.key	+..+.......+.........+....+..+.+...........+..+..+..+..+.............+...+......+..+.+...+...........+...+........	0.205662
T3951_96_128_499.key	.+..+....+.....+.....+...++..+..+.+.+...........+......+.+..+.....+...+.......+........++..........................	0.233300

Figure 4: Table shows sample images for Britney Spears and their result for the program that runs both with unique match and geometric distance constraints.

As you will see above figures 3 and 4 that Figure 3 results are better than Figure 4 results even though the Figure 4 has both unique match and geometric distance constraint. Normally we expect figure 4 to have better results. Nevertheless we did not get the results we expected, even though geometric distance constraint eliminates the irrelevant matches. The reason for this is that the geometric constraint is not set in the implementation properly. Although it eliminates the irrelevant matches it also eliminates some necessary matches as well as storing some irrelevant ones. Our next step will be to improve the geometric constraint in the implementation to get more accurate results.

5 Implementation details

5.1 Class diagram



MatchEngine
-image : Image
+findMatches(in im1 : Image, in im2 : Image, in keys1 : KeyPoint, in keys2 : KeyPoint) : double
+CheckForMatch(in key : KeyPoint, in klist : KeyPoint) : KeyPoint
+DistSquared(in k1 : KeyPoint, in k2 : KeyPoint) : int
+GeometricDistance(in k1 : KeyPoint, in k2 : Image, in im1 : Image, in im2 : Image) : double
+AllocMatrix(in rows : int, in cols : int) : int
+AllocMatrix2(in rows : int, in cols : int) : float
+CreateImage(in rows : int, in cols : int) : Drawable Image
+ReadPGMFile(in filename : char) : Drawable Image
+ReadPGM() : Drawable Image
+WritePGM(in image : Drawable Image) : void
+DrawLine(in image : Drawable Image, in r1 : int, in c1 : int, in r2 : int, in c2 : int) : void
+CombineImagesHorizantally(in im1 : Drawable Image, in im2 : Drawable Image) : Drawable Image
+CompareMean(in x : Result, in y : Result) : bool
ConnectionManager
+getCompareSet() : Image
KeyPoint
-row : float
-col : float
-scale : float
-orient : float
-descriptors : unsigned char
-index : int
*
1
Image
-rows : int
-cols : int
-personId : int
-imageId : int
-personName : string
-filePath : string
-keypoints : KeyPoint
-numberOfKeys : int
Result
-mean : double
-index : int
Drawable Image
-rows : int
-cols : int
-pixels : float
-next : Drawable Image

Figure 5: Shows the class diagram of the existing program.

MatchEngine: This is the main class that performs the matching operation

Properties:

image: This is the given image to be compared.

Methods:

FindMatches: Takes two image files and their keypoints in order to find matches.

CheckForMatch: Finds the best match in the klist (keypoint vector) by calling DistSquared function.

DistSquared: Finds the distance between two keypoints

GeometricDistance: Given the images, finds the geometric distance between two keypoints

AllocMatrix: Allocates integer matrix in order to draw lines between match points.

AllocMatrix2: Allocates float matrix in order to check unique match constraint.

CreateImage: Creates the images that the matching points are shown

ReadPGM: Takes the file name and calls the ReadPGMFile function

ReadPGMFile: Initializes images with respect to their features like size, pixels etc.

WritePGM: Creates pgm files

DrawLine: Draws line between match points of the face images.

CombineImagesHorizantally: Combines two images horizantally in order to see the match points

compareMean: It is used to sort the Result vector in ascending order with respect to their minimum distance values.

KeyPoint: This class holds the key point values of the interest points.

Proporties:

row: Y position of the interest point.

col: X position of the interest point.

scale: Scale of the interest point.

orient: Orientation of the interest point.

descriptors: Gray scale color range of the interst point (between 0-256).

index: Index of the interest point among all interest points.

Image: This class represents the image object that is brought from the database.

Proporties:

rows: Y dimension of the image.

cols: X dimension of the image.

personId: ID of the person that the image belongs to.

imageId: Id of the image.

personName: Name of the person that the image belongs to.

fileName: Filepath of the image.

keypoints: Keypoints of the image.

numberOfKeys: Number of keypoints.

Result: This class represents the result of the comparison between the given image and the images in the database.

Properties:

mean: Mean of the mininum distances of matching points.

index: Index of the result image.

Drawable Image: This class is used to draw the lines between matching points.

Properties:

rows: Y position of the image.

cols: X position of the image.

pixels: Pixel matrix of the image.

next: Refers to next drawble image.

ConnectionManager: This class is used to get the dataset from the database

Methods:

getCompareSet: This function returns the images in the database in Image class format.

5.2 Database Structure

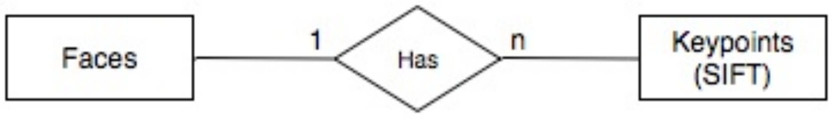


Figure 6: ER diagram that shows the database structure of the system

Database structure consists of two main tables which are faces and keypoints. In faces table, each row represents a normalized face image with some additional information of corresponding person. In keypoints table, each row represents an interest point, a feature length of 128, which is extracted by David Lowe's SIFT Keypoint Detector. Original image files are stored in file system within folders by their ID numbers. These image files are not used during execution of the program, once features have been extracted. (See Figure 6)

PK = Primary Key, FK = Foreign Key

faces(id, personId, name, image)

id: (PK, int) ID # of the face image

personId: (int) ID # of corresponding person

name: (string) full name of person

image: (string) filename of corresponding image file in file system

keypoints_lowe(id, imageId, row, column, scale, orientation, feature[128])

id: (PK, int) ID # of interest point

imageId: (FK, int) ID # of face image which references to *faces* table

row: (float) y position of the interest point

column: (float) x position of the interest point

scale: (float) scale of the interest point

orientation: (float) orientation of the interest point in range [-PI, PI]

feature[128]: (int) feature descriptor vector of interest point in range [0, 255]

6 Application Details

Our main aim in this project, is to find an algorithm which can appropriately find similarities between faces. After we have improved our algorithm, and start to get good result for most of the inputs, we will develop our application using this algorithm. We are planning to include the following features in our application:

1.Finding Celebrities: Our application will find the celebrities that resemble the given photo the most.

2. Comparing faces: When given two face images, the application will be able to find how much they resemble each other.

3. Finding the Identity: When given an old or even a childhood photo of a famous person that exist in our database, our application will tell who that famous person is.

4. Matching Part: When a certain part (for example: only the eye) of a celebrity is given, our application will match it to the concerned celebrity from the database.

7 Appendix

7.1 Appendix A

Here are the tables that shows the results of the program that is run for different images of randomly chosen 7 people. Each table represents one person. Each person represented with a unique ID. By this way they are differentiated.

Person 221:

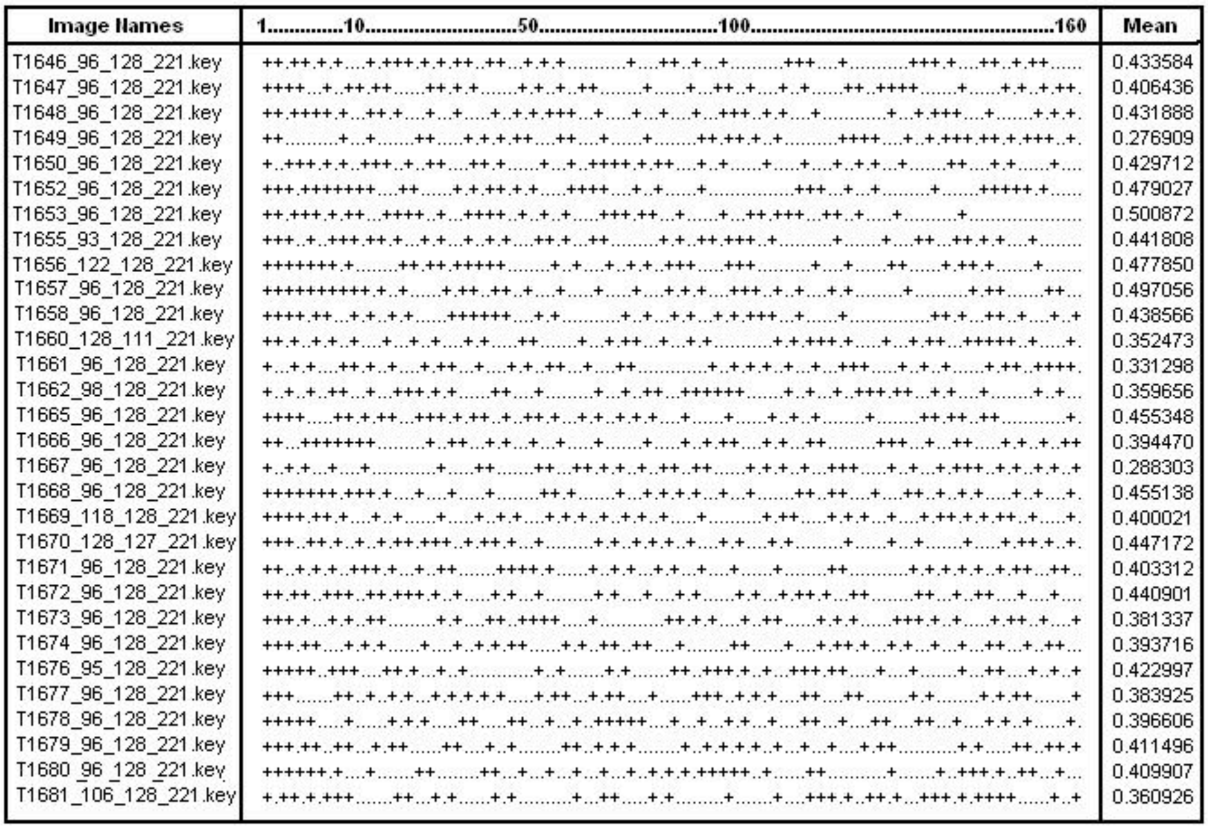


Image Names	1............10..........................50................................100...160	Mean
T1646_96_128_221.key	++.++.+.+....+.+++.+.+.++..++...+.+.+..........+....++..+...+.........+++....+..........+++.+....++..+.++......	0.433584
T1647_96_128_221.key	++++...+..++.++......++.+.+.......+.+..+..++........+......+...++..+....+..+......++..++++.......+......+.+..+.++.	0.406436
T1648_96_128_221.key	++.++++.+...++.+....+...+.....+..+.+.+++...+.....+...+....+...+++..+.+....+...........+...+.+++....+.......+.+.+.	0.431888
T1649_96_128_221.key	++..........+...+.......++.....+.+.+.++.....++....+......+........++.++.+..+..........++++....+..+.+++.++.+.+++..+.	0.276909
T1650_96_128_221.key	+..+++.+.+..+++..+..++....++.+.......+...+..++++.+.++....+..+......+......+....+..+.+...+.......++....+.+.......+....	0.429712
T1652_96_128_221.key	+++.+++++++....++......+.+.++.+.+.....++++....+..+......+................+++...+...+..........+.......+++++.+......	0.479027
T1653_96_128_221.key	++.+++.+.++...++++..+...++++..+..+..+.....+++.++...+.......+...++.+++...++..+.....+..........+...................	0.500872
T1655_93_128_221.key	+++..+..+++.++.+...+.+...+..+.+...++.+...++.........+.+..++.+++..+..........+.......+....++...++.+.+....+........	0.441808
T1656_122_128_221.key	+++++++.+........++.++.+++++........+..+....+..+.+..+++.....+++..........+....+.....++......+.++.+.......+.......	0.477850
T1657_96_128_221.key	++++++++++.+..+......+.++..++..+....+.....+.....+....+.+.+....+++..+..+....+.+.........+..........+.++.......++...	0.497056
T1658_96_128_221.key	++++.++...+.+..+.+......++++++....+.+..........+..+...+.+...+.+.+++...+......+..............++.+...++..+....+..+	0.438566
T1660_128_111_221.key	++.+..+.+..+....+...+..+....+.+.....++.......+...+.++...+...+.+..........+.+.+++.+.....+...+.++...+++++..+.....+.	0.352473
T1661_96_128_221.key	+...+.+....++.+..+....+.++...+....+.+..++...+....++............+..+.+.+..+...+...+++.....+..+..+......+.++..++++.	0.331298
T1662_98_128_221.key	+..+..+..++...+...+++.+.+......++....+.........+...+..++...++++++.......+..+...+..+++.++...+.+....+........+..+...	0.359656
T1665_96_128_221.key	++++.....++.+.++...+++.+.++..+..++.+...+.+..+.+.+....+......+......+..+..+........+........++.++..++...........+.	0.455348
T1666_96_128_221.key	++...+++++++.........+..++...+.+...+...+.....+........+.....+..+.++....+.+...++.........+++...+...++......+.+..+..++	0.394470
T1667_96_128_221.key	+..+.+...+....+............+......++.......++...++.+.+..+..++..++......+.+.+..+...+++.....+..+...+.+++..+.+..+.+..+	0.288303
T1668_96_128_221.key	+++++++.+++.+....+....+.....+........++.+.......+...+.+.+.+...+...+.......++..++....+....++..+..+.+......+..+....+.	0.455138
T1669_118_128_221.key	++++.++.+....+..+......+.....+..+.+....+.+.+..+..+.+..+.....+..........+.++.....+.+.+...+....+.++.+.+.++..+.....+.	0.400021
T1670_128_127_221.key	+++..++.+..+..+.++.+++..+.++.+...+.........+.+..+.+.+..+....+.+.....+.+.........+......+...+.......+......+.++.+..+.	0.447172
T1671_96_128_221.key	++..+.+.+..+++.+...+..++.......++++.+......+..+.+...+.+...+....+......+........++...........+.+.+.+.+..+.++...++..	0.403312
T1672_96_128_221.key	++.++..+++..++.+++.+..+.....+.+...+.........+.+....+.....+.+......+.+...+.++.+...++........++...+..++.....+.....+....	0.440901
T1673_96_128_221.key	+++.+...+.+..++.........+.+....++..++++.....+...........++.+.+.....+..++......+.+.+......+++.+..+......+.++..+.....+	0.381337
T1674_96_128_221.key	+++.++.....+.+.+......+....+..+.+.++......+.+..++..++....+........++......+.....+.++.+...+.+...+...+...++...+..++...	0.393716
T1676_95_128_221.key	+++++..+++.....++.+...+..+...........+..+.......+.+.......++..+++.+..+...+++.++.....+.....+.......+...++.....+..+..+	0.422997
T1677_96_128_221.key	+++.......++..+..+.+...+.+.+.+.+.....+.++...+.++....+......+++..+.+.+....++.....++.........+.+.........+.+.++.......+	0.383925
T1678_96_128_221.key	+++++.....+......+.+.+.....++.....++....+....+..+++++....+....+....+.+...+.....++....+.....++.....++....+.....+.+..+......+.	0.396606
T1679_96_128_221.key	+++.++..++...+.++......++....+..+.........++..+.+.+.........+..+.+.+.+..+...+...+...+.++..........+.+......++..++.+	0.411496
T1680_96_128_221.key	++++++.+....+.......++.........++...+...+...+...+...+..+.+.+.+++++..+.......++...........+........+..+++.+..++...+...	0.409907
T1681_106_128_221.key	+.++.+.+++.......++...+.+......+.+..........+...++.....+.+.........+........+....+++.+..++.+...+++.+.++++.......+..+	0.360926

Person 478:

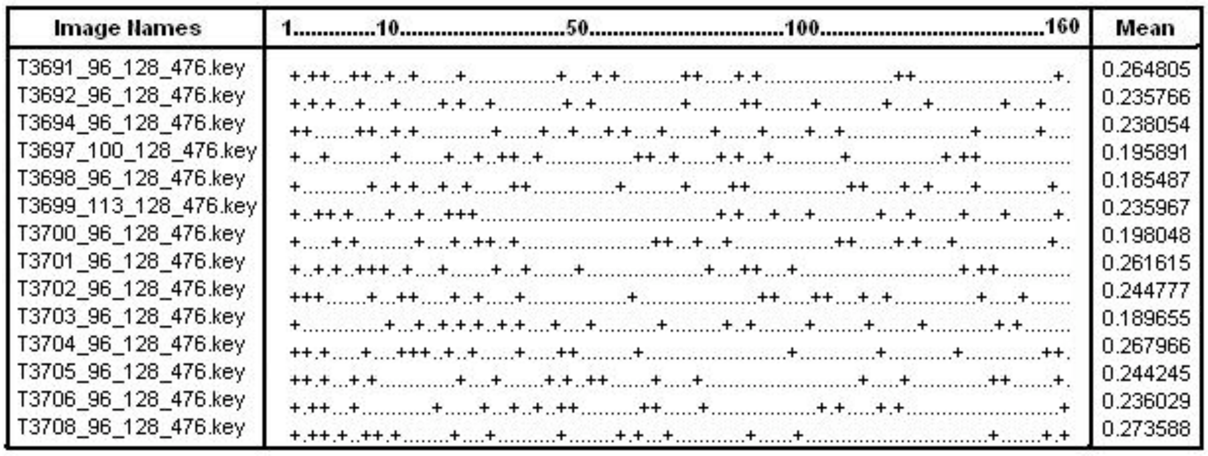


Image Names	1............10..........................50................................100.....................................160	Mean
T3691_96_128_476.key	+.++...++..+..+.......+...............+....+.+..........++.....+.+........................++.......................+.	0.264805
T3692_96_128_476.key	+.+.+...+.....+......+.+...+...........+..+.............+.........++........+..........+........+...........+.....+....	0.235766
T3694_96_128_476.key	++........++..+.+...........+.......+....+....+.+....+........+.......+.......+...+.........................+.........+....	0.238054
T3697_100_128_476.key	+...+...........+........+...+..++..+................++..+.......+.+...+...........+...............+.++...............	0.195891
T3698_96_128_476.key	+...........+..+.+...+..+......++..............+..........+.......++..................++.....+..+.......+...........+..	0.185487
T3699_113_128_476.key	+..++.+.....+...+...+++...+.+.....+.....+..........+...+.......+.....+.......+.	0.235967
T3700_96_128_476.key	+......+.+..........+.....+..++..+....................++...+...+..................++......+.+....+................+..	0.198048
T3701_96_128_476.key	+..+.+..+++..+.....+.......+...+.......+.....................+.....++.....+...........................+.++............	0.261615
T3702_96_128_476.key	+++.......+...++......+..+.....+................+...................++......++....+..+..............+.....+.......	0.244777
T3703_96_128_476.key	+.............+...+..+.+.+..+.+.....+.....+..........+........+..........+........+.......+..........+.+.........	0.189655
T3704_96_128_476.key	++.+......+.....+++..+..+......+......++.........+.........................+.............+...........+............++.	0.267966
T3705_96_128_476.key	++.+...+.+.............+.....+.......+.+..++.......+......+..........................+.....+.............++.......+.	0.244245
T3706_96_128_476.key	+.++...+...........+.......+...+..+..+..++..........++.......+..................+.+.....+.+...........................+	0.236029
T3708_96_128_476.key	+.++.+..++.+........+.....+..........+.........+.+...+............+.......+.................................+.......+.+	0.273588

Person 499:

Image Names	1..........10..........................50....................................100...................................160	Mean
T3936_96_128_499.key	++..+..+...+.+..+++....+..+...+...+...+....+.++..............................+............+..+................................	0.317303
T3937_96_128_499.key	+++.+..+.++......+++...++.+....+...+.+.+......++...................................+...	0.360600
T3938_96_128_499.key	++++......++.+..++.....+..+.+....+......+.+.........+.........+...+............+....................+.....	0.324054
T3939_96_128_499.key	+.+.+++..+......+......+.++...+.+....+...............++...+..........+.+.............+.........+...........	0.303385
T3942_96_128_499.key	+++.+..+...+.+..+....++..+.....++....+...................................++....+.....+........+........+.............	0.320362
T3943_96_128_499.key	+++.++++..++...+.....+.++......++...+.++....+.....+..	0.389737
T3944_96_128_499.key	+..+.+......+.+..++...+.+.......+....++....+............+.+.....+.+...+.....+.........................+..........	0.272259
T3950_96_128_499.key	+....+........................+..+....++..........++....+........+.+...+.+.+....+..+.+....+.......+..........+..	0.171727
T3951_96_128_499.key	+...+.+.+....+++.+.+.....+..+.......+.+...+.........++..........++.......+..........................+............	0.291712

Person 569:

Image Names	1..........10..........................50....................................100.........................160	Mean
T4473_96_128_569.key	+.++..............................+.+.......+..........................+.........................+.....+.................................. .	0.154612
T4474_96_128_569.key	+......+++......... ....+...........................+.......+..................+.............................+... .	0.137828
T4475_96_128_569.key	+..+...........+...........+...........+..+..........+.........................+......................................+......... .	0.138287
T4476_96_128_569.key	++.+.....+.......+.........+..........................+.......................+.............................+...................... .	0.178070
T4477_96_128_569.key	++..........+...........+...+.........+.....................+.........+.....+... .	0.154882
T4481_96_128_569.key	+..+......+..........+...................+.......+..+...+....................+...... .	0.078851

Person 666:

Image Names	1..........10..........................50....................................100.........................160	Mean
T5286_96_128_666.key	++...+.+........+.............+.....+.....+......+..........+..................................+............................	0.198843
T5289_96_128_666.key	+.....+..+.+..+..............+.....+...........................++......+........+...	0.175240
T5290_96_128_666.key	+.++.....+...+...+...+..+....+.......+..+.......................	0.219380
T5291_92_128_666.key	+........+..+.....+......++........+...........+........+...+..+..........	0.155713
T5292_96_128_666.key	+..+..++..+..+....+..+...........................+.+............+...	0.218427
T5293_96_128_666.key	+..+...+..........+....+.....................+.+...++...........+.+..	0.172883
T5294_96_128_666.key	+.++.+...+.....+...+..+...+...+..+.............	0.203451
T5295_96_128_666.key	+.+++.....+......+.+.....+.........................+...............................+......................................+.........	0.218441
T5296_96_128_666.key	+..+.++....+...............+..+.+....+......+..........+...	0.207141

Person 681:

Image Names	1..........10..........................50....................................100.........................160	Mean
T5442_96_128_681.key	+..+..+...+....++..........+.............................++.........+.............+.	0.101430
T5443_96_128_681.key	++.........+....+..............+...........+...+.+..+.+............................+...	0.165390
T5444_96_128_681.key	+..+............+.......+....+...............+.................+.+.+..............+................................+	0.147448
T5445_96_128_681.key	+...................................+.....+.....+.....................+.........+.....+..........+..+..+....+............	0.096397
T5447_87_128_681.key	+....+.........................+.....+..+.....+.........+.......+..+........+...+.	0.129886
T5448_96_128_681.key	+.......+..+...........+.........+.......+...........+..........+.........+.........+...........................+	0.141904
T5449_96_128_681.key	+...............++........+..++.................................+...+.................+.................+..+..........	0.134683
T5451_85_128_681.key	+......+..........+...........+..............+........+......+...++......+...................................+..........	0.130205
T5452_96_128_681.key	+.......+..............++.....+...+....+..+.....+.......+..............+..........	0.131673

Person 770:

Image Names	1...........10..........................50..................................100.....................................160	Mean
T6260_96_128_770.key	+.......................+..+.....................+.............................++..+.....+.	0.091117
T6263_96_128_770.key	++.......+.......+..+............................+.....+........+............	0.137156
T6264_85_128_770.key	+..+..+..........+..............+..+........+..........+............	0.134563
T6265_91_128_770.key'	+...................................+.........+..+.........+..+...+...........+..	0.087642
T6267_96_128_770.key	++...+..................++++......................+......	0.101690

8 Reference

[1] Grgic, Mislav, and Kresimir Delac. "General Info." Face Recognition Homepage. 2005. 21 Oct. 2007 <http://www.face-rec.org/general-info/>.

[2] Gross, Ralph. “Chapter 13. Face Databases” Handbook of Face Recognition Ed. Stan Z. Li, Ed. Anil K. Jain New York: Springer, 2005. 301-328

[3] Viola, P.; Jones, M., "Rapid Object Detection Using a Boosted Cascade of Simple Features", *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, ISSN: 1063-6919, Vol. 1, pp. 511-518, December 2001

[4] “Eigenvector.” Wikipedia. 29 Dec. 2007. Wikimedia Foundation. <http://en.wikipedia.org/wiki/Eigen_vector>.

[5] “Eigenface.” Wikipedia. 29 Dec. 2007. Wikimedia Foundation. <http://en.wikipedia.org/wiki/Eigenface>.

[6] “Euclidean Distance.” Wikipedia. 29 Dec. 2007. Wikimedia Foundation. <http://en.wikipedia.org/wiki/Euclidean_distance>.

[7] "Face Detection and Recognition." Betaface. 17 July 2007. 20 Oct. 2007 <http://www.betaface.com/>.

[8] "Face Recognition Demo." My Heritage. 2006. 20 Oct. 2007 <http://www.myheritage.com/FP/Company/tryFaceRecognition.php?lang=TR>.

[9] "Face Recognition." Electronic Privacy Information Center. 5 Sept. 2007. EPIC. 20 Oct. 2007 <http://www.epic.org/privacy/facerecognition/>.

[10] "Facial Recognition System." Wikipedia. 18 Oct. 2007. Wikimedia Foundation.

19 Oct. 2007
<http://en.wikipedia.org/wiki/Facial_recognition_system>.

[11] "InformationRetrieval." Wikipedia. 20 Feb. 2008. Wikimedia Foundation.
< http://en.wikipedia.org/wiki/Information_retrieval#Average_precision>.

[12] "Fisherface." Wikipedia. 29 Dec. 2007. Wikimedia Foundation.
<http://en.wikipedia.org/wiki/Fisherface>.

[13] "Hidden Markov Model." Wikipedia. 29 Dec. 2007. Wikimedia Foundation.
<http://en.wikipedia.org/wiki/Hidden_Markov_model>.

[14] "How Facial Recognition Systems Work." How Stuff Works. 2007. 19 Oct. 2007
<http://computer.howstuffworks.com/facial-recognition.htm>.

[15] "Interest Point Detection." Wikipedia. 29 Dec. 2007. Wikimedia Foundation.
<http://en.wikipedia.org/wiki/Interest_point_detection>.

[16] "Linear Discriminant Analysis." Wikipedia. 18 Oct. 2007. Wikimedia Foundation.
19 Oct. 2007
< http://en.wikipedia.org/wiki/Linear_discriminant_analysis>

[17] Lowe, David. Distinctive Image Features From Scale-Invariant Keypoints. University of British Columbia. Vancouver: University of British Columbia, 2004. 2. Nov. 2007 <http://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf>.

[18] Ozkan, Derya. A Graph Based Approach for Finding People in News. Bilkent University. 2007. 24-28. Nov.-Dec. 2007
<http://www.cs.bilkent.edu.tr/%7Eduygulu/Thesis/DeryaOzkanThesis.pdf>.

[19] Pissarenko, Dimitri. "Eigenface-Based Facial Recognition." (2003). Nov. 2007
<http://openbio.sourceforge.net/resources/eigenfaces/eigenfaces-html/facesOptions.html>.

[20] "Principal Components Analysis." Wikipedia. 18 Oct. 2007. Wikimedia Foundation.

19 Oct. 2007
< http://en.wikipedia.org/wiki/Principal_components_analysis>.

[21] "Three-dimensional Face Recognition." Wikipedia. 29 Dec. 2007. Wikimedia Foundation.
<http://en.wikipedia.org/wiki/Three-dimensional_face_recognition>.