BUCUREŞTI

2016s

POLITECHNICA UNIVERSITY OF BUCHAREST

FACULTY OF AUTOMATIC CONTROL AND COMPUTERS

COMPUTER SCIENCE DEPARTMENT

|  |  |
| --- | --- |
| upb | cs |

DIPLOMA PROJECT

Automatic Face Verification System

|  |  |
| --- | --- |
| Thesis supervisor:  Prof. Irina Mocanu | George Leotescu |

BUCHAREST

2016

Table of Contents

[1. Introduction 3](#_Toc265586143)

[3. Related Methods](#_Toc265586146) 16

[4. Application Architecture 4](#_Toc265586147)

The abstract should describe the context in 2-3 phrases and give details about the project functionality and implementation in 3-5 phrases.

1. Introduction

Face recognition is a task which is executed on a daily basis, efficiently and without special efforts by any human being. However, this doesn’t hold in the computer’s world where a face recognition problem like any other computer vision problem is a non-trivial one. Detecting a face from an input image, finding its scale and contour, discovering its shape parameters and rendering the original face into a standard shape format, finding a way to represent the face data in a low dimensional space which allows two faces to be compared one to each other, solving the classification problem, all of these are non-trivial problems, which require a robust and efficient solution in order to implement any modern face recognition system.

Speaking of terminology, the “*Face Recognition*”, “*Face Verification*” and “*Face Identification*” terms are tightly related to each other and they can be easily confused. *Face recognition* represents the general topic, which includes both *Face Verification* and *Face Identification* (also referred to as “*Face Authentication*”). On the one hand, *Face Identification* is concerned with finding the identity of a face from a set of known identities (one to many matching). The key aspect is that a face identification system knows precisely the set of identities with which it is dealing, meaning that all the query faces will be compared against a known set of identities. Obviously in this case, the natural approach is to train the system to learn particular features of the registered identities and search for them in the query photos. On the other hand, face verification deals with validating a claimed identity of a face, by comparing it with a model face (one to one matching). The system doesn’t have any information about the identities of the faces that are going to be compared so an identity specific training cannot be done here. The key aspect is that the system needs to be general enough, in order to be able to deal with different identities, without losing its accuracy. It is obvious that a tradeoff needs to be done between flexibility (do we need to re-train the system when a new identity is added to the system?) and accuracy (how accurate can we be when validating the identity of a person, using only one sample of its face?). Even though we saw that a *Face Verification System* is different comparing to a *Face Identification System*, they have a lot of things in common. There are a lot of methods (image pre-processing, data representation, classification methods) which are employed by both systems. Even though the face recognition research field is a wide one, any improvement brought in any level can have wide ramifications.

In the last few years, the research in the *Face Recognition* field has focused on five main complementary directions: face detection, landmarks key-points detection, face registration, face description and face representation using statistical methods. Any modern face recognition system includes all these steps but in general it is enough to improve only one of them in order to increase the overall system performance.

* 1. Face Detection

*Face detection* is a computer vision technology with applicability in a wide range of domains, whose purpose is to identify human faces in digital images. The *face detection* process is similar to the natural human behavior of localizing faces in a visual scene. Even though detecting a face is a trivial task for any human being, in the computer’s world, it is a high resource consuming task. The main difficulties come from the fact that in unconstrained environments, faces can be found under different conditions, with variations in scale (face size), location, orientation (in plane rotation – the roll angle of the face), pose (out of plane rotation –the yaw angle of the face), facial expressions, facial special characteristics, different illumination conditions etc., as it can be seen in Figure 1. A good face detection system needs to be robust to all these variations.



Fig1. Examples of faces with different shapes, in different positions, orientations, under different lighting conditions.

The research in the Face Detection area has made progresses in special in the past decade. Specifically, the work of Viola and Jones [1] has transformed the face detection problem from an almost untouchable task for the computer’s world, into a feasible one. This emerged immediately into real world applications, integrated directly into video or photo camera devices, or image processing software.

The Viola Jones algorithm has at its core three fundamental concepts: the integral image, classifier learning using AdaBoost and the attentional cascade structure [2]. These were at the moment, innovatory ideas in the face recognition research field, which made Viola Jones a powerful and robust algorithm prepared to be used in real time applications.

* + 1. The integral image

*Integral Image*, also known as *Summed Area Table* was first introduced in 1984 by Crow [3] which used it in mipmaps but hasn’t been used in the computer vision field until 2001 when it was employed by Viola and Jones in their face detection algorithm. The integral image represents a quick and effective way of computing the sum of the pixel values of a sub-rectangle contained in an image. What it does, it precomputes the sum of all the possible sub-rectangles of an image, having the top-left corner in the top-left corner of the image. Having all these sums pre-computed, the pixels sum of any rectangle contained in an image can be obtained in constant space (by decomposing the input rectangle into rectangles for which the pixel’s sum represents a known value). Viola Jones employed this mechanism for computing Haar like features as depicted in Figure 2. Using the integral image mechanism, the Viola Jones algorithm computes an enormous number of Haar features (different scales, different locations in the input image) in a small amount of time [2].

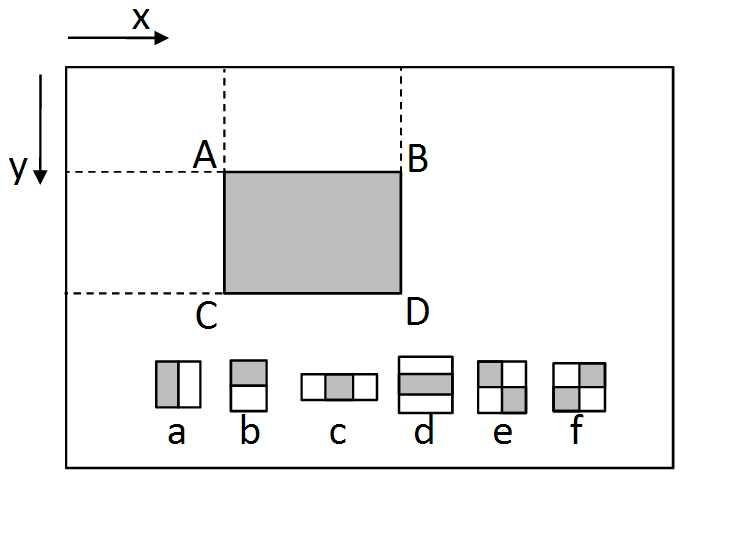


Fig 2. Integral image applied for different types of Haar features (a-f).

* + 1. Classifier learning – Adaboost

Adaboost (Adaptive Boosting) is a Gödel Prize winning machine learning algorithm, formulated by Yoav Freund and Robert Schapire [4]. Even though it’s well known for its contribution to the Viola Jones face detection algorithm, it is a general boosting algorithm which can be used with many other types of machine learning algorithms, in order to improve their performance. Given a set of other learning algorithms (so called “weak classifiers”), AdaBoost combines them into a weighted sum which represents the output of the boosting classifier.

In machine learning, AdaBoost usually refers to a specific method of training a boosted classifier. A boost classifier is defined as any classifier of the form:

(1)

where, represents a weak learner, which takes as input and determines its object class. The sign of the output value represents the object class while the absolute value gives the confidence of the prediction.

The training algorithm iterates through T steps. At each step , a weak classifier is selected and a corresponding coefficient is determined so that the training sum error of the current step is minimized. For a weak classifier, given its output value (also called the hypothesis) for a sample from the training set , the training sum error of the step ­­­ is given by the formula:

(2)

in which the information belonging to the classifier built on the previous step is embedded as , while denotes the weight assigned to each sample in order to minimize the overall error. The value of the weight is based on the error value calculated in the corresponding sample, at the previous step. These weight values can be used to inform the training of a weak classifier, using as data the given training samples and the associated weights. For example, decision trees can be generated so that they split set of samples using weights values as priorities. The complete AdaBoost algorithm can be described in a few simple steps, as depicted in Figure 3 [2].

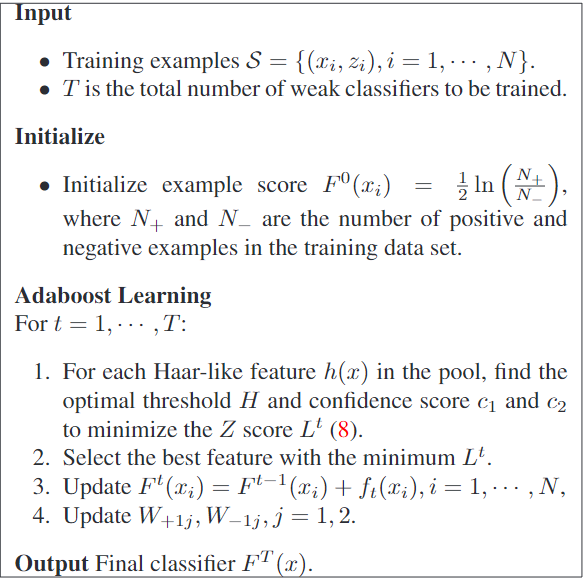


Figure 3. The Adaboost training algorithm

* + 1. The attentional cascade structure

The attentional cascade structure is a key element in the Viola Jones algorithm which allows it to reject negative samples in an incipient phase. In order to do that, a set of smaller but thus more efficient classifiers are used, each of them being trained to reject the negative samples and to keep the positive ones. Thus the whole decision process takes the shape of decision tree, optimized to reject the negative samples as soon as possible. Figure 5 [2] depicts the basic structure of the decision tree, sometimes referred to as a “cascade”. Each node contains a variable number of classifiers, which typically is bigger as the index of the node is bigger relative to its position in the cascade structure (after the obvious negative samples are rejected, it’s more and more difficult to decide whether an image should pass to the next node or should be dropped; so a bigger number of classifiers need to be employed).

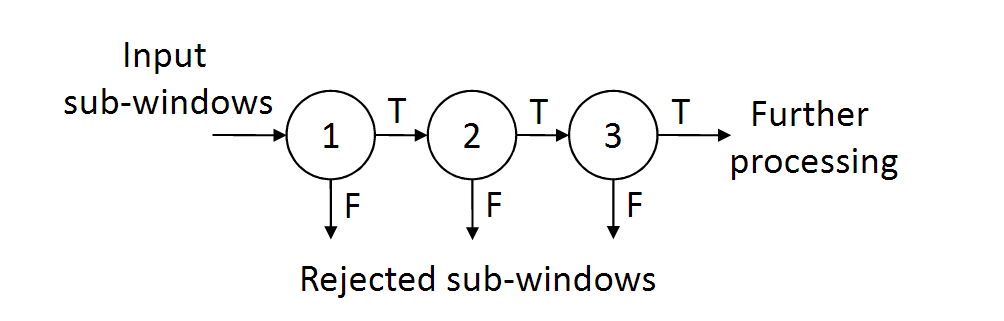


Figure 5. The structure of a cascade classifier, along with the number of classifiers employed by each node

* 1. Facial Landmark Detection

When it comes to face alignment, face analysis, pose estimation or face normalization techniques, detecting facial landmark points is undoubtedly the key stone. Once you have a bounding box around the detected face, the natural follow up is how you detect precisely the shape of the face, its size and special characteristics. For example, one can try to detect the location of different specific features, like the corner of the eyes, the mouth, the tip of the nose, the eyebrows, the full contour of the face. In literature the Facial Landmark Detection problem is also referred to as “Facial Key Point Detection”, “Facial Pose Estimation” or simply “Face Alignment”.

Like any other problem in the Face Recognition research field, a robust and efficient solution for the Facial Landmark Detection problem can improve the solutions of a large set of problems from different research areas.

For example, facial landmarks can be used to align faces to a mean face shape so that after the alignment process is done, specific face characteristics (eyes, mouth, eyebrows) will be approximately in the same place, for all the input faces. This can improve the overall performance of a face recognition system.

Once you have a bounding box around the face, the obvious research problem is to see if you can find the location of different facial features (e.g. corners of the eyes, eyebrows, the mouth, the tip of the nose etc.) accurately. It is natural to think that this will improve the overall performance of a face recognition system and indeed a lot of research has been done in this area and the results confirm it [5][6]. For example, once the location of a small set of landmark points is known, it is possible to estimate the position of the head (the head coordinates in a three dimensional space). In order to do that, an automatic camera calibration system can be employed. To calibrate a camera against a 3D model, implies finding the optimal perspective projection transformation which projects the model from the 3D space into a 2D space in which its shape is as close as possible to the current image shape. By decomposing the projection matrix obtained, one can approximate the head orientation in space [7][8].

As described in [5], after detecting a bounding box around an input face (using an algorithm similar to the Viola Jones algorithm described above) a cascade of regressors is employed.

Each regressor in the cascade is trained to estimate the location of the corresponding key point. An iterative algorithm is employed. At each step, a new prediction is made, using the information obtained at the previous step. Let (x, y) represent the 2D coordinates of the th facial key point of the input image . The vector contains the coordinates of all the key points . We refer to as the estimated shape of and we use to denote the shape of estimated at the step . Given a regressor which can predict a new value for the estimated shape vector S, we can write the following equation:

(3)

which is used to predicted the shape of the image at the step , using the values predicted at step .

The intial estimation is a generic mean shape. It can be easily obtained by aligning and centering the training images to a scale dependent to the size of the bounding box returned by the face detection algorithm. The whole algorithm of learning the regressors cascade, as described in [5], is depicted in Figure 6.

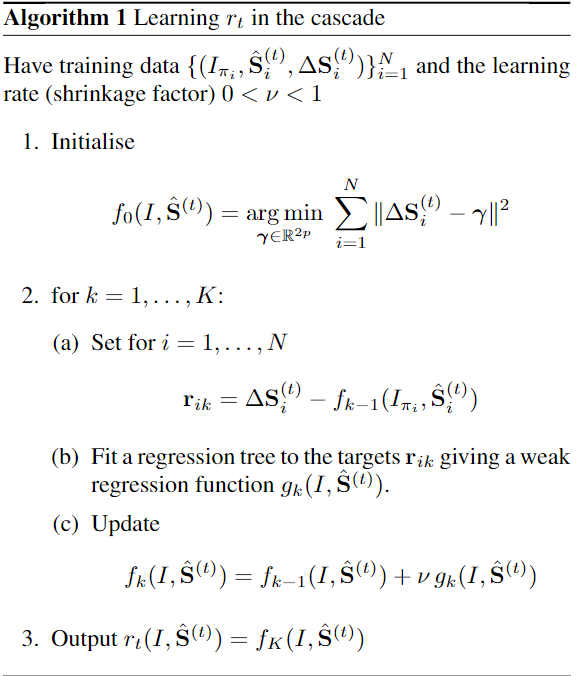


Figure 6. Algorithm for learning a cascade regressor used in facial key points detection

The key component for each regression function is a decision tree. Each tree is fit using as weights, the residual components obtained during the gradient boosting algorithm. At each node in the regression tree, a decision is made by comparing the intensities of the pixels in the input face to the intensities of the correspondent pixels in the mean model face. The decision is made using a threshold obtained during the training phase. Given a pixel in the input face, with the position determined by two vectors , (vectors defined in the query space), the algorithm aims to determine the correspondent pixel in the model shape, its location being defined by and (vectors defined in the model space). In order to solve this problem, a straightforward solution would be to wrap the original input face into the model shape. However, considering that we are dealing with a sparse representation of the image (only the detected key points are relevant) it is easier to wrap only those pixel locations. This can be easily done by computing a global similarity transform (a two dimensional space transformation, based only on a scaling, translation and rotation) and applying it to each point.

* 1. Face Registration

In the Face Recognition field, there are many applications which require robustness against occlusions, variations in shape or size, different illumination conditions, facial expressions etc. The Face Registration problem aims to reduce all these variations by fitting an input image to a general model thus all the query samples being forced to be defined by the same shape, size, or illumination parameters.

A simple Face Registration application can be developed as a straightforward application of the Key Points Detection algorithm described above. When the fitting algorithm finishes, the similarity transform used to map the original input face into the generic model, is known. As described in [9], decomposing a similarity transform, gives us the in plane rotation angle (roll) and the face scaling factor relative to the model shape. This information can be pumped into a simple face alignment algorithm. The original bounding box (as it was returned by the Face Detection algorithm) is shrank using the face scale determine using the method described above. The rectangle is rotated against the face roll angle and cropped from the input image. Thus as cropped and aligned face is obtained from the input image, with a minimum overhead.

Any face alignment algorithm will try to maximize the correlation between an input face and a standard model. In [10] it is described a face alignment method based on gradient descent optimizations. Comparing to the first method, the algorithm employed in [10] determines the correlation coefficients between the two images by computing complex gradients which capture the images structures (orientation, shape) directly, instead of computing average pixel intensities in specific points. The algorithm represents on optimization of the classic Lucas-Kenade algorithm [11] which was the first algorithm of this type. The main characteristic of a gradient descent algorithm is that it is an iterative learning method. At each step, the method solves a least squares problem, having the Hessian matrix pre-computed (it is constant between different iterations). In general, the norm is used to compute the error function. However, it is obvious that optimizations can done on this level in order increase the overall system accuracy. In general, for algorithms in this category, a tradeoff needs to be done between the robustness of the algorithm and its complexity. A robust system replaces the norm with a complex error function. Thus the value of the weights used are updated at each step. In this case the Hessian matrix doesn’t have a constant value across different steps. Its value needs to be determine at each iteration, increasing thus the overall system complexity.

1.4 Face description using statistical methods

In the Face Recognition research field, statistical methods are especially used to learn face representations in a low dimensional space. Given an input image represented in a high dimensional space, a statistical method will encode the image data into a low dimensional space, an euclidean space. In the new space a large set of metric learnings techniques can be applied in order to fit a classifier to the system requirements. In the last few years, the research in this field has focused on finding new image representations or efficient ways of merging two or more consecrate representations (like SIFT, HOG) into a single one, more efficient and robust comparing to any of them.

1.4.1 CCA – Canonical Correlation Analysis

In [11] it is proposed a novel method a fusioning two or more features vectors into a single one which is more discriminative than any of the input feature vectors. The algorithm is based on Canonical Correlation Analysis, a method widely used before to analyze associations between two sets of variables [12].

CCA is a statistical method which uses the mutual relationship between two random variables in order to determine the maximum correlation between them. Generally speaking, it as important as any others statistical methods techniques like PCA (Principal Component Analysis) or LDA (Linear Discriminant Analysis) and it has been used with success in different research fields like signal processing, computer vision, speech recognition etc. The method behind CCA was first introduce by H. Hotelling in 1914. The problem is formulated in the following way: given two random variables, and , centered in zero, find a pair of projection directions named, α and β that maximize the correlation between the two sets of data projected onto them. Specifically, find the value of α and β so that the variables and are as correlated as possible. The pair (,) is called the first pair of canonical variables. The process can continue, finding the second, third and so on, pairs of canonical variables. The key aspect is that, the variables inside a pair of canonical variables are correlated one the each other as much as possible but two different pairs of canonical variables are uncorrelated. Thus, in order to analyze the correlation between two random variables it is enough to analyze their first pairs of canonical variables.

1.4.2 Fisher Vector

The concept of Fisher Kernels represents a powerful framework in the pattern classification research field. The basic idea consists in representing an input signal, using a gradient function determined from a generative mixture model and passing the results to a discriminative classifier. Thus, the concept of Fisher Kernels includes the advantages of two different and complementary approaches: processing data using a generative model and using classifiers trained in a discriminative way. In the last few years, Fisher Kernels have been used successfully in the image classification field. Particularizing the problem, the Fisher Kernels concept can be applied directly using an image as the input signal and a Gaussian Mixture Model as the generative model. The role of the Gaussian Mixture Model is to determine the distribution of the extrinsic characteristics in the input image.

Given a visual vocabulary, the problem of describing an image is reduced to counting how many times each word from the vocabulary appears in the query image. From this point, any classifier can be employed for categorizing the histograms obtained. The main research in this area has focused in finding an accurate visual vocabulary which can describe in an efficient and discriminative way any image received as input. Over time, a lot of algorithms employing different methods have been used, e.g. the k-means algorithm, Gaussian Mixture Models, mean-shift algorithm etc.

1.4.2.1 K-means algorithm

K means clustering is a method originally used in signal processing, whose purpose is to cluster n observation variables, into k clusters so that each clustered variable belongs to the cluster with the nearest mean value. Even though this is an NP hard problem, there are efficient heuristic algorithms which converge to the local optimum.

The problem can be formulated in the following way: given a set of observations where each observation is a dimensional variable, the k-means algorithm tries to partition these variables into (with sets, so that each variable corresponds to the set in which it is closest to the set mean value. In other words, the K-means algorithm partitions the variable into different sets so that the within-cluster sum of squares is minimized. For each variable, the algorithm finds the value of the expression:

, (4)

where denotes the mean of the points contained in the set . Figure 7 depicts a simple implementation for the K-means algorithm.

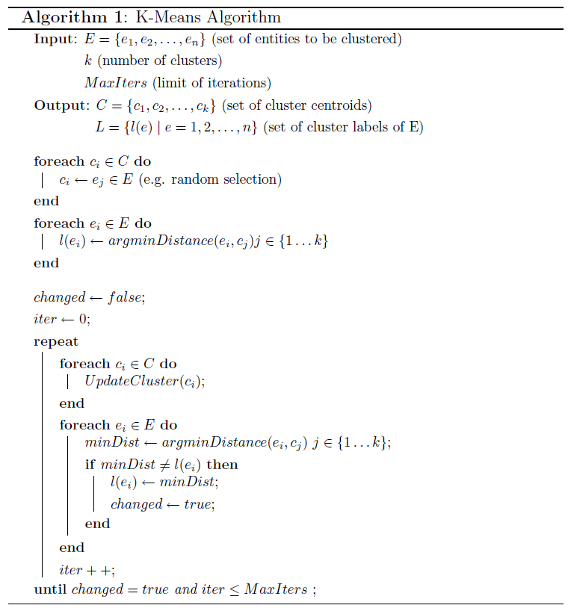


Figure 7. A simple implementation for the K-means clustering algorithm

A general problem consists in finding the initial mean values. A popular solution, as illustrated above consists in randomly choosing samples to represent each sub-set. However it is important to keep in mind that the results produced depend on the initial values of the means so when a bad choice is made there are high chances for a suboptimal result to be found. The standard solution consists in re-iterating the algorithm with a number of different starting points. It is also important to remark that the results depend on the number of clusters,. This is troublesome because we cannot known how many clusters should exist in order to obtain an optimal solution (for example, how you decide if the data returned by a 3-means clustering algorithm is better comparing to a 2-means clustering algorithm?).

1.4.2.2 Gaussian Mixture Models

A Gaussian mixture model is a probabilistic model that assumes all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters. One can think of mixture models as generalizing k-means clustering to incorporate information about the covariance structure of the data as well as the centers of the latent Gaussians.

A general mixture model is a hierarchical model, containing the following components:

1. *N* random variables corresponding to different observations, each assumed to be distributed according to a mixture of *K* components. Each component is assumed to belong to the same family of parameters (e.g. for a Gaussian Mixture Model, all components follow a Gaussian distribution).
2. *N* corresponding random latent variables, each specifying the identity of the mixture component of each observation
3. A set of *K* normalized mixture weights. Each mixture weight represent a probability value between 0 and 1. The sum of all the mixture weights is equal to 1.
4. A set of *K* parameters, each specifying the parameter of the corresponding mixture component. Each parameter represents a set of parametric values. E.g. for a Gaussian Mixture Model, each component is represented by a *mean* and a *variance* value.

Figure 8 depicts the plate notation of a Gaussian Mixture Models, along with the notation used.

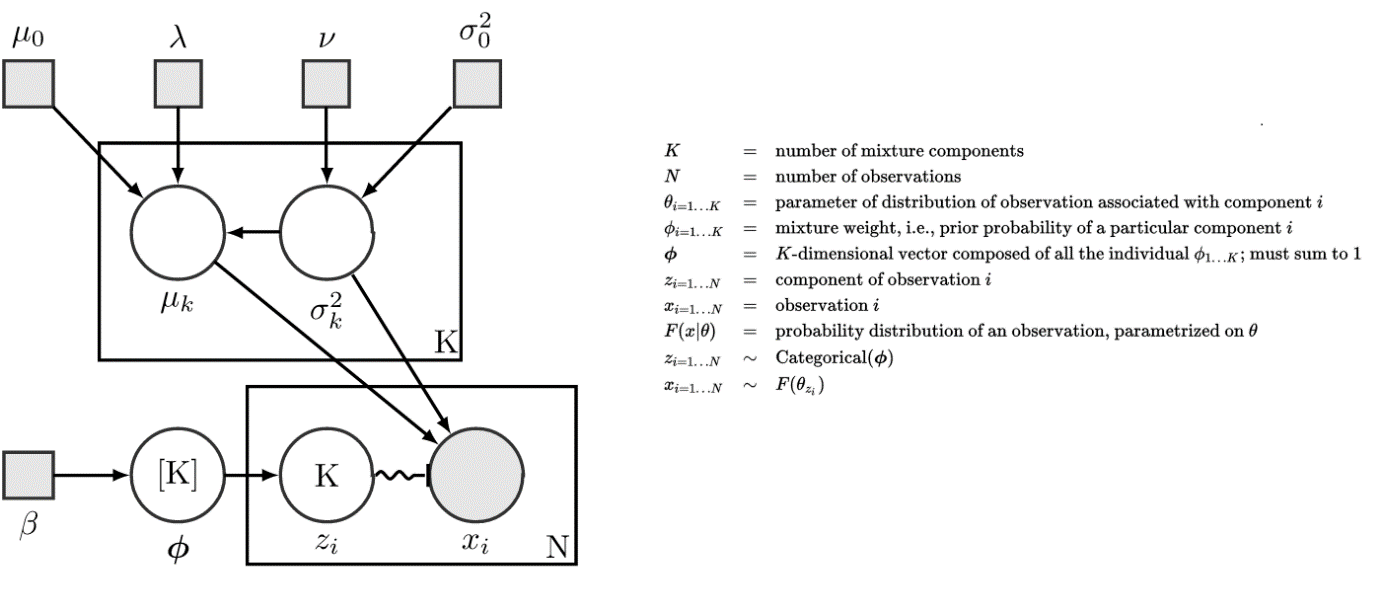


Figure 8. The plate notation of a Gaussian Mixture Model, along with the notations used.

1.4.2.3 Mean Shift

Mean shift is a [non-parametric](https://en.wikipedia.org/wiki/Non-parametric) [feature-space](https://en.wikipedia.org/wiki/Feature_space) analysis technique for locating the maxima of a [density function](https://en.wikipedia.org/wiki/Density_function). It has been introduced long back in 1975, but wasn’t widely used until direct applications of it were found in the computer vision research field. At its core, it is an iterative algorithm which is used for finding modes, clusters etc.

The algorithm considers the image feature space as an empirical probability density function. An image represents a set of points so the mean shift algorithm considers them as being sampled by the density function which needs to be find out. By analyzing the probability function, if dense regions are encountered, they correspond to the local maxima value of the function (also referred to as the function mode). For each point in the data set, mean shift associates it with the nearby peak of the function. A window is defined around it and the algorithm computes the mean value of the data point. After that, the center of the window is shifted to the mean value and the same steps repeat until the algorithm converges. This can be visualized as a window moving step by step to denser and denser regions of the dataset, until it is centered in the peak value.

Therefore, the algorithm can be described in 4 simple steps:

1. Compute a window w around each point in the dataset
2. Compute the mean shift vector
3. Translate the estimated window with
4. Repeat until the algorithm converges

The algorithm described above is a time consuming algorithm ( complexity, *n* representing the number of points in the data set). However many improvements can be made in order to make the algorithm to converge faster. A possible solution would consist in allowing the bandwidth parameter (the window size) to vary for each data point. For example the value of the bandwidth for a given point could be found by applying a K Nearest Neighbor algorithm. Another solution would be to alter the data points during the execution of the mean shift algorithm, for example by using a Gaussian Kernel.

As a conclusion, even though K Means is not a parametric algorithm, it requires the value of the bandwidth parameter to be known. Even tough, in order to find it, different solutions can be employed (like the K Nearest Neighbor Algorithm), it is important to notice that the choice of the bandwidth parameter will influence directly the number of clusters generated and the convergence rate of the algorithm. When applying the K Nearest Neighbor Algorithm, the size of K will influence directly the bandwidth parameter generated. As a common rule, it is recommended to increase the value of k with the size of the input data.

1.4.2.4 Methods used in generating the Visual Vocabulary

Analyzing the research methods in this field, one can observe that even in the case of a small database, with a limited number of classes, the best results were obtained when a visual vocabulary with a large number of entries was employed. Obviously using a big visual vocabulary implies a higher computational cost, because given a base vocabulary and an input object, the cost of creating its histogram depends directly on the vocabulary size. This is why a lot of research has been done in this area, trying to find a way of generating a visual vocabulary, as compact as possible without losing the robustness and overall performance of the system. A possible solution consists in organizing the vocabulary in a tree structure (Randomized Tree Structure). Even though the lookup time for a visual term will decrease in time, this solution doesn’t scale. Another solution is based on the Information Bottleneck Principle. The e Information Bottleneck Principle finds the best tradeoff between accuracy and the size of the visual vocabulary employed. The method represents a probabilistic algorithm rather than a statistical one. Given a distance metric between two words in the visual vocabulary and a distance matrix, it computes the transition probabilities between two sample pairs. Given the transition probabilities, one can compute the probability for each sample word of being included in the final visual vocabulary.

Even though both methods described above fulfilled their expectations (for example by reducing five times the size of the visual vocabulary, without any loss in the system accuracy) the fundamental aspect is that the vocabulary obtained is not a universal one. The problem that emerged came from the fact that the system wasn’t flexible enough. In order to add new classes, the whole system needed to be retrained. Based on these, it became obvious that the idea of generating a vocabulary which was both compact and universal, was almost an impossible one. However a pseudo-solution appeared, based on the K Means algorithm. It consisted in building a visual vocabulary for each different class employed. Even though it proved to be a flexible approach (the system doesn’t need to be re-trained) it’s doesn’t necessarily reduce the overall computational complexity of the system.

In [14] it is described a method for combining the advantages of a universal vocabulary with the advantages of using a set of smaller vocabularies, one per each class. Thus the K Means problem is transformed into a 2 Means problem. For each class, the universal vocabulary is combined with the corresponding one-class vocabulary. Given an input image, a histogram is computed for all the combined vocabularies. The resulted histograms show for each class if the image is best modelled by the universal vocabulary or by the in-class vocabulary.

1.4.2.5 Principles of Fisher Kernels

The main idea behind the Fisher Kernels consists in representing an image using a model based on a probability function which characterizes the process that might have generated the input signal. Gradient based representations which can be found at the core of the Fisher Kernels, have the great advantage, comparing to the classic methods which employ visual vocabularies (e.g. Bag of Visual Words), that they can use linear classifiers bigger in size. An important decision that needs to be made when designing the generative model behind the Fisher Kernels consists in whether or not to use the class information during the training phase. Specifically, the decision consists in choosing between a Visual Vocabulary learned using a supervised method and a vocabulary learned using a non-supervised one. Definitely, choosing a supervised method will increase the system overall performance but it will also impose limitations on its flexibility (in order to add a new class, the whole system needs to be re-trained).

When speaking about pattern classification methods, there are two distinct directions: generative learning methods and discriminative learning methods. While the methods in the first category are based on modeling a density probabilistic function which describes the input object or the process that generated it, the methods in the second category handle directly the classification methods, for example by using different metric learning techniques. All these things being said, the generative methods still have a set of properties which make them attractive. For example, the possibility of using data with variable dimensions.

From the beginning, Fisher kernels were designed so that they could combine the advantages of a discriminative learning technique with the advantages of generative one. Using the notation introduce in [14], let be the density probabilistic function and, a set of samples used during training. The log-likelihood gradient of is given by the formula:

(5)

Intuitively speaking, the log-likelihood gradient describes the directions in which the parameters of the mixture model should be changed so that it will optimally describe the input data. Thus, the input data is transformed from a vector variable in size into a vector fixed in length, whose size depends only on the number of the parameters of the employed model. As the next step, this gradient can be classified using any discriminative classifier. In [14] Fisher Kernels are applied directly on Visual Vocabularies, represented by the means values of a Gaussian Mixture Model.

1. Related methods

During the last few years, Face Recognition has been one of the main research fields in the computer’s world. Along time, a lot of techniques have been employed, starting from geometrical hand-crafted algorithms to statistical methods and deep learning. Even though the techniques employed vary significantly from one method from another in general any Face Recognition System deals with the same set of problems. One can consider that any classic Face Recognition System solve the following key-stone problems:

1. *The Data Representation Problem.* An image is stored in memory as a sequence of pixels, each pixel being represented by 3 numbers (the RBG model). Even though this works for images, it is obvious that this model cannot be used to represent faces. The first problem is that a box containing a face has a lot of noise information in it. The first source of the noise is the face background. This can be reduced but we still remain with a pixel-based representation which doesn’t help in highlighting the discriminative information of the face. A good representation model should keep only the relevant information of the face, those characteristics that discriminate it from other faces. Another problem comes from the fact that a Face Recognition System should bring all the input faces to the same denominator, before comparing them. Any Face Recognition System will have better results when the input faces are aligned, resized to the same scale, frontalized etc.

A possible solution could be to extract patches at different scales, which don’t overlap and use a universal vocabulary to create the image histogram. Another solution could be to divide the input image into cells of pixels, and to find the local pattern of each cell (for example by extracting feature based characteristics like SIFT or HOG).

1. Data Encoding Problem. This step deals with noise removal, how to find discriminative representations, how to project data from a high dimensional space to a euclidean space etc. Methods like PCA (Principal Component Analysis), LDA (Linear Discriminant Analysis) have been successfully employed.
2. Classification problem. Given two image representations, one needs to compare them and to decide, for example, whether the faces come from the same person or not. The easiest solutions consist in projecting the input data on a low dimensional space and use as metric, the euclidean distance. Other approaches apply metric learning techniques, for example classifiers based on Bayesian Networks or Support Vector Machines.

In the rest of this section, we are going to present three state of the art systems and the methods employed by each of them.

* 1. Face verification system based on feature fusion techniques

In [12] there is described a fully automated Face Verification System, based on feature fusion techniques (Canonical Correlation Analysis). This system works in the following way: before any processing is done, an input image is normalized (both in-plane and out-of-plane normalization being employed); an Active Appearance Model (AAM) is used in order to fit the input face to a standard model. Given the normalized face, different types of features are extracted from it (GABOR, SIFT, HOG) and fused together into a single feature vector (using Canonical Correlation Analysis – CCA). As an intermediate step, Principal Component Analysis (PCA) is applied in order to reduce the size of the extracted feature vectors. Finally, a linear classifier based on Euclidean distance is employed.

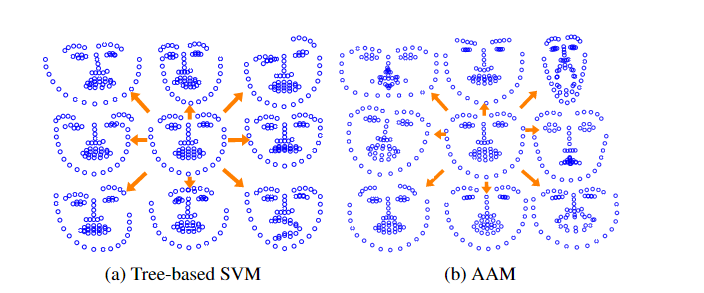
The motivation behind the used methods, comes from the fact that, in general, the biggest number of discrepancies between two faces are generated by external factors (for example different illumination conditions, different head orientations, different scales etc.). Therefore these variables will be eliminated from an early point. In order to do that, the systems performs in-plane and out-of-plane normalization and merges features vectors extracted using different methods (each method is robust to a different set of external perturbations). The in-plane normalization consists in finding the roll angle of the face and rotating the input image against it, while the out-of-plane normalization consists in registering the input face against a standard frontalized 3D model. Even though the system fulfilled its expectations, it can be improved. The face were registered against a fixed 3D model; even though this helps, because same type of features will be found in the same position, the solution is not a robust one. When dealing with faces which deeply vary in shape (for example the distance between the eyes is different comparing to an average value), the performance of the system will decrease drastically. It is natural to think that when a face with an unusual shape is registered against a standard model, the discriminative information is lost.

For Landmark Points Detection, it is employed the method described in [15]. It is based on Parts Model, a method widely used in the Object Detection research field.

2.1.1 Parts Model

Part-based models are built on the original idea of Fischler and Elschlager of using the relative position of a few templates matched into the query image in order to predict whether the input image contains a specific type of object or not. For example in order to detect a face, one can try to detect smaller pieces of the face (e.g. nose, mouth, eyebrows) and using the relative positions of the detected pieces to decide whether they are part of a real face or not. The model employed has an arborescent structure, using a shared pool of parts , each part being trained to capture deformations in location, shape, size of a given landmark point .

From this point, dynamic programming algorithms can be employed to find the globally optimal solutions. Figure 9 shows a 2D model, deformed using the method described above (a) versus a classic method (b) which uses a joint Gaussian Model, a standard solution commonly used in Active Appearance Model – AAM.

Figure 9. Deformations of a face model based on Parts Model vs. the classic solutions based on joint Gaussian Models

2.1.2 Histogram of Oriented Gradients – HOG

Histogram of oriented gradients (HOG) is a feature description algorithm, widely used in computer vision and image processing, with the main application in object detection. From a high level perspective, HOG is similar to other feature extraction techniques (like edge orientation histograms, scale invariant feature transform, etc.). The key aspect, different comparing to other existing methods is that HOG is computed on a dense grid of overlapping cells and uses local contrast normalization techniques to improve accuracy.

Hog is based on the fact that the shape of an object can be described the distribution of intensity gradients or edge directions. The image is divided into cells of pixels (each cell containing for example 4x4 pixels), and for the pixels within each cell, a histogram of gradient directions is compiled. The final cell descriptor is represented by is the concatenation of these histograms. To improved accuracy, the local histograms can be contrast-normalized by calculating a measure of the intensity across a larger region of the image, called a block, and then using this value to normalize all cells within the block. It has been shown that this normalization algorithm makes the system robust to illumination variations.

The HOG descriptor has a few key advantages over other descriptors. It operates on local cells so it is invariant to geometric transformations (rotations, translations). However it is not robust to object orientation variations. This is the main reason why pose normalization techniques are employed. Figure 10 depicts the graphical representation for a HOG descriptor extracted from a human face.

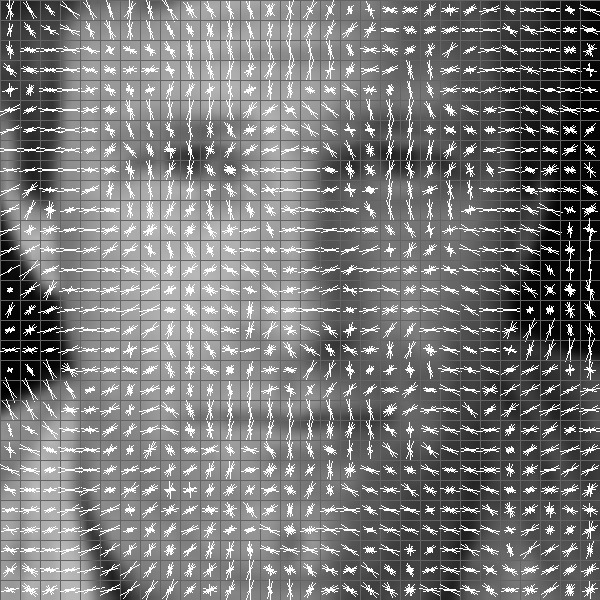


Figure 10. The visual representation of the HOG descriptor applied to a face image

A key aspect when using HOG in Face Recognition is to apply it for aligned faces. Because HOG operated on pixel cells, it is important to know that corresponding cells store correlated information. In [12], after the normalization process was done, HOG descriptors were employed using cells of 4x4 pixels and 9 orientations.

2.1.3 Gabor filters

In image processing, a Gabor filter, is a linear filter used in general for edge detection. It has been found that the output of a GABOR filter applied to an image, resembles mostly the way in which an image is stored and depicted in the human brain. From a mathematical point of view, a 2D Gabor filter is a Gaussian kernel function, modulated by a sinewave.

Gabor filters are directly related to Gabor wavelets, the way in which Gabor filters are generally used in the computer vision field. A Gabor wavelets is basically a filter bank containing a set of Gabor filters, in different scales and orientations. The main advantage of Gabor wavelets is that they ensure robustness against geometrical transformations (scaling, translations) variations in illumination conditions and noise.

[12] uses a Gabor wavelet containing 40 different Gabor filters (5 scales and 8 orientations). Figure 11 depicts the result of the filter bank applied to a face image.

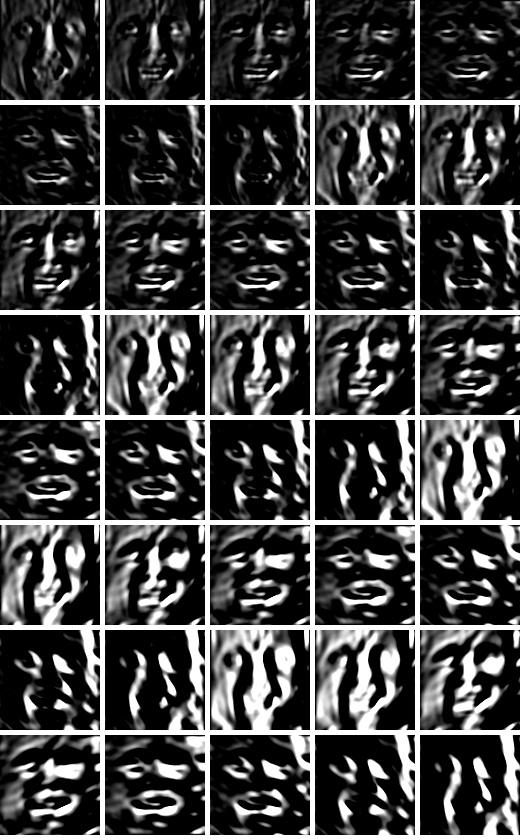


Figure 11. The output of a Gabor wavelet containing filters in 5 scales and 8 different orientations, when applying it to a face image

2.1.4 Feature fusion

Given the two descriptor vectors (the first one obtained using HOG and the other one using GABOR), the method described in [12], merges the two feature vectors into a single one. As an intermediary step, Principal Component Analysis is applied in order to reduce the noise and to shrink the size of the two vectors. The method employed is based on Canonical Correlation Analysis (CCA), a statistical method which finds the associations between two sets of variables. More details about it can be found in the Section 1 of this document. Given the resulted fusion vector, it is passed to a linear classifier, which uses as metric the Euclidean distance.

2.1.5 Experimental results

The system has been trained and tested in 3 different setup conditions.

1. *FERET* database – a face recognition database, containing 2200 images of 200 subjects (11 images / subject). The faces are represented in different poses, with different facial expressions. The system obtained state-of-the-art results, an overall accuracy of over 91.5%.
2. *CMU-PIE* database – a smaller face recognition database, containing faces of 68 subjects in 13 different positions. The system registered 95% accuracy, in the conditions in which the poses of the faces varied between -45 and 45 degrees.
3. *LFW* – the standard face recognition database, containing faces in unconstrained environments. The database contains 13233 faces of 5749 subjects with 1680 subjects having more than one sample. The system registered an accuracy over 87% which is better comparing with other methods applied in these conditions.
   1. Face verification system based on Fisher Vectors and Linear Discriminant Analysis

[16] Presents an original approach which combines three key concepts in the Face Recognition Research field: Scale Invariant Feature Transform (SIFT), Fisher Vectors and Linear Discriminant Analysis. Speaking form a high level perspective, the algorithm divides the input image into a large number of overlapping patches. For each patch the corresponding feature is extracted (using SIFT) and all the feature vectors are encoded into a single feature vector using Fisher Vectors as the encoding method. At this point, a Mahanalobis distance is learned by applying Linear Discriminant Analysis (LDA) to the feature vectors of the training set images. The LDA algorithm has two functions: first of all it reduces the size of the input vector and secondly it learns a discriminative metric.

* + 1. Algorithm overview

The whole system can be reduced to the following steps:

1. *Feature Extraction*. The input image is divided into a large number of overlapping patches. In [16] the size of a patch was set to 24x24 pixels with a stride value of 1 pixel. The process repeats 5 times, in different scales, using a scaling factor of. This results in approximatively 26k patches. The algorithm applies SIFT to each patch, resulting a 128 dimension feature vector. Concatenating all these feature vectors into a single one, it results a 26k x 128 feature vector.
2. *Features Encoding.* PCA is applied in order to reduce noise. A SIFT feature vector is reduced from 128 dimensions to 64. Thus the image descriptor is reduced in size to 26k x 64. In the next step, a Gaussian Mixture Model is employed in order to create a histogram of the extracted patches. More details about Fisher Vectors can be found in the first section of this document.
3. *Location augmentation.* Each patch is augmented with its location in the input image. Given a patch with the location (x, y) in the original image, its final representation is given by (S, x, y), where S represents the SIFT feature of the given patch.
4. *Classification*. A linear classifier based on Euclidean distance is used.
   * 1. Experimental Results

The system has been trained and tested on LFW (Labeled Faces in the Wild), the standard database used in training and testing systems designed for unconstrained environments. LFW contains 13233 faces of 5749 subjects with 1680 subjects having more than one sample. The database was divided in 10 disjoint sets, pumped into a cross validation system. When a stride of 2 pixels was used, the system registered an accuracy of 89%, while when the stride was reduced to a single pixel and a Bayesian classifier was trained, the accuracy increased to 93%.

* 1. Face verification system based on Discriminative Multimanifold Analysis

[17] Describes a novel approach which combines two other state-of-the-art techniques in the Face Recognition research: Manifold Learning and Discriminant Analysis.

* + 1. Algorithm overview

The training images are associated with labels, corresponding to N different classes. Each class is viewed as a different manifold. Existing methods divide each manifold into a number of sub-sets, each sub-set being described by a linear space. Thus each manifold will be described by a collection of linear sub-spaces. However it has been proven that a linear space is a weak representation for a subset so another approaches have been developed.

Thus the concept of “affine hulls” has been introduced. A spectral clusterization algorithm is used to divide a set of images into multiple sub-sets. Each subset represents an affine hull and the clusterization algorithm ensures us that for each affine, a representative and discriminative mean vector can be obtained. The clusterization algorithm builds the Laplacian matrix using as input similarities between image pairs contained in each manifold and uses its first eigen-vectors to clusterize the input data.

Given the affine hulls computed by the clusterization algorithm, a Mahanlobis metric is learned, applying Linear Discriminant Analysis (LDA).

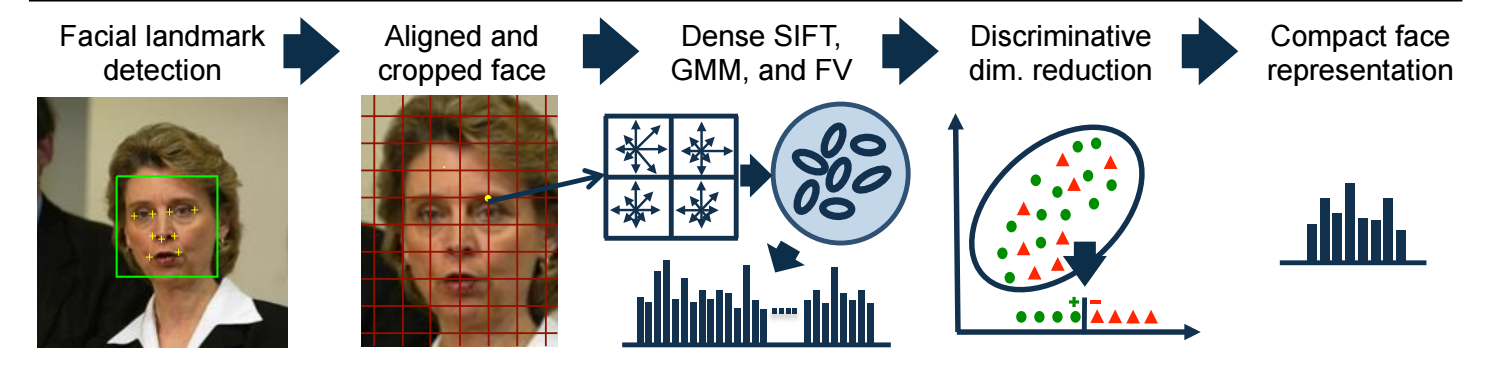
* + 1. Experimental Results

The system has been trained and tested in two different setup conditions.

1. YALE B database – a Face Recognition database, containing 2414 images of 38 different persons. Faces contain variations in pose, facial expressions and shapes. The accuracy registered by the system varied deeply, between 42% and 99%, depending on the sets of data used.
2. *CMU-PIE* database – a smaller face recognition database, containing faces of 68 subjects in 13 different positions. The accuracy registered by the system varied deeply, between 59% and 100%, depending on the sets of data used.

Even though the system fulfilled its expectations and the method has been successfully validated, it is obvious that the system is not robust enough and improvements needs to be done.

1. The Application Architecture



bibliography

[1] P. Viola, M. Jones – Robust Real-time Face Detection, International Journal of Computer Vision, Vol. 57, Issue 2, pp. 137-154, 2004

[2] C. Zhang, Z. Zhang - A Survey of Recent Advances in Face Detection, Technical report, Microsoft Research, 2010

[3] Lewis, J.P – Fast template matching, Proc. Vision Interface. pp. 120–123, 1995

[4] A decision-theoretic generalization of on-line learning and an application to boosting, Journal of Computer and System Sciences 55, 119-139, 1997

[5] V. Kazemi, S. Sullivan – One millisecond face alignment with an ensemble of regression trees, [IEEE Conference on Computer Vision and Pattern Recognition](http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=6909096), pp. 1867-1874, 2014

[6] X. Cao, Y. Wei, F. Wen, and J. Sun. Face alignment by explicit shape regression. In CVPR, pages 2887-2894, 2012

[7] T. Hassner – Viewing Real-World Faces in 3D,  [IEEE International Conference on Computer Vision](http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=6750807), pp. 3607-3614, 2013

[8] T. Hassner and R. Basri. Example based 3D reconstruction from single 2D images. In Proc. Conf. Comput. Vision Pattern Recognition, 2006

[9] S. Umeyama Least-squares estimation of transformation parameters between two point patterns, IEEE Transactions on Pattern Analysis and Machine Intelligence (Volume:13 , Issue: 4 ), pp. 376 – 380, 1991

[10] [G. Tzimiropoulos](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Georgios%20Tzimiropoulos.QT.&newsearch=true) , [S. Zafeiriou](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Stefanos%20Zafeiriou.QT.&newsearch=true) – Roboust and efficient parametric face alignment, [International Conference on Computer Vision](http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=6118259), pp. 18471-854, 2011

[11] B.D. Lucas andT. Kanade, An iterative image registration technique with an application to stereo vision, Interna-tional joint conference on artificial intelligence, pp. 674–679, 1981

[12] M. Haghighat , M. Abdel-Mottaleb, W. Alhalabi,  Fully automatic face normalization and single sample face recognition in unconstrained environments, [Expert Systems with Applications](http://www.sciencedirect.com/science/journal/09574174), [Volume 47](http://www.sciencedirect.com/science/journal/09574174/47/supp/C), pp. 23–34, 2016

[13] Q. Seun, S. Zeng, Y. Liu, P. Heng, D. Xia, A new method of feature fusion and its application in image recognition, [Pattern Recognition](http://www.sciencedirect.com/science/journal/00313203), 2015

[14] F. Perronnin – Fisher Kernels on Visual Vocabularies for Image Categorization,  [IEEE Conference on Computer Vision and Pattern Recognition](http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=4269955), pp. 1-8, 2007

[15] D. Ramanan - Face detection, pose estimation, and landmark localization in the wild,  IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2879-2886, 2012

[16] K. Symonian, O. Parki, A.Vedaldi, A. Zisseraman – Fisher Vectors in the Wild, British Machine Vision Conference, 2013

[17] J. Lou, Y. Tan, G. Wang – Discriminative  Multimanifold Analysis for Face Recognition from a Single Training Sample per Person, [IEEE Transactions on Pattern Analysis and Machine Intelligence](http://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=34), Vol 35, pp. 39-51, 2012

[1] C. Soviany, Embedding Data and Task Parallelism in Image Processing Applications, PhD Thesis, Technische Universiteit Delft, 2003

[2] A. Mauthe, D. Hutchison, G. Coulson and S. Namuye, “Multimedia Group Communications Towards New Services”, in Distributed Systems Eng., vol. 3, no. 3, Sept. 1996, pp. 197-210

[3] V. I. Arnold, Metodele matematice ale mecanicii clasice (Mathematical models of classic mechanics), Editura Ştiinţifică şi Enciclopedică, Bucureşti, 1980

[4] \*\*\* COSMOS/M – Finite Element System, User Guide, 1995.