DETECTING FACIAL FEATURES AUTOMATICALLY

A Thesis

by

Uranchimeg OLZVOI

Submitted to the
Graduate School of Sciences and Engineering
In Partial Fulfillment of the Requirements for
the Degree of

Master of Computer Science

in the
Department of Computer Engineering

IŞIK UNIVERSITY June 2013

DETECTING FACIAL FEATURES AUTOMATICALLY

Approved by:

Assist.Professor M.Taner ESKIL, Advisor
Department of Computer Engineering
IŞIK UNIVERSITY

Professor Ercan Solak
Department of Computer Engineering
IŞIK UNIVERSITY

Assoc.Professor Olcay Taner Yildiz
Department of Computer Engineering
IŞIK UNIVERSITY

Date Approved: 2 July 2013

To my mother...

ABSTRACT

There are many algorithms and approaches in object detection world. Many of them are based on Viola Jones algorithm. According to our observations, the features which help to detect an object are very critical for the success of this algorithm. These features are usually created manually. In this thesis we explore automatic extraction of Haar-like features. We describe the design and construction of a completely automated face detector for gray scale images. Finally, we illustrate the performance of our algorithm on various databases.

ÖZETÇE

Obje tespit etmek icin bir çok algoritma ve yaklaşım vardır. Bunların çoğu Viola Jones algoritmasına dayanır. Bizim edindiğimiz tecrübelere göre, obje tespitinde temel konu o objeye ait özniteliklerdir. Bu öznitelikler genellikle manuel olarak oluşturulur. Bu tezde biz Haar-like özniteliklerin otomatik çıkarımları üzerine araştırma yaptık. Gri tonlamalı resimler için tamamıyla otomatikleştirilmiş bir yüz algılayıcısı tasarlayıp bunu uyguladık. Nihayetinde, tasarladığımız algoritmanın farklı veribankaları üzerindeki performansını gösterdik.

ACKNOWLEDGEMENTS

I would like to express my deepest appreciation to M.Taner ESKIL who is my advisor and my life coach during my master education. I am indebted to him for his endless patience.

I want to thank Prof. Dr. Mustafa KARAMAN for his great supports during my graduate life in Şile campus.

I want to give my special thank to my family for their unconditional love and support.

I grateful to Gülce Gül and Serkan Yıldız for their moral support while i am writing my thesis.

I want to very special thank my friends Serol Koşunda, Halil Ibrahim and Erdi Ölmezoğullari for their friendship.

I want to thank Kristin Benli, Asli Cevahir, Didem Agdogan, Efruz Oztop, Sezin Ata and Seda Yildirim for sharing good memories with me.

This research is part of project "Expression Recognition based on Facial Anatomy", grant number 109E061, supported by The Support Programme for Scientific and Technological Research Projects of The Scientific and Technological Research Council of Turkey (TUBITAK).

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Everything has been getting more digital in our lives. We prefer to transfer and keep quite important aspects of our lives online. Many social interactions are now enhanced by multimedia, with immediate access to pictures and videos that share memories, make our points in a discussion, or just to entertain.

Nowdays, 'facebook' and 'youtube' are two largest sites that provide immediate access to multimedia. There are countless social sites that provide enhanced media of self interests. When tasks can be accomplished by transfer of enhanced data over the internet, jobs migrate to digital world through home offices. We prefer to shop for various merchandise, and do bank transactions online. The availability of internet in all aspects of our lives through smart phones, tablets, IP-TV, even household appliances augments our life as we know it with the virtual world.

Through improving technology, transfer and storage of data are more available than last years. Around 2006, memory sticks brought huge improvements in our lives with convenient transfer of data. Now end users are moving away from memory sticks just to avoid the hastle of remembering to take it. We store, share and process various kinds of data on cloud services. Storing a significant portion of our lives on remote servers require huge transfer bandwidth for accessing data.

Also security system demands have been increasing with enlargement on digital world. Security systems have migrated from local, stand-alone architectures to world wide network. This has the advantages of distributed tracking, verification and recognition systems. These systems leverage on large data base of images, real time processing, distributed camera systems and reliable security.

"A picture is worth a thousand words"; an image is more informative than one dimensional data. On the other hand, image processing is a very challenging topic of machine vision. Machine interpretation of images aims to answer questions such as "does the given object exist in the image?".

Human faces are challenging to detect as their appearances vary drastically between individuals (race, beards, eyeglasses, etc.) and through facial expressions. Detection of human faces needs to be performed online for real time analysis of scenes.

Detection of faces is a challenging problem due to their large variation between individuals. The variation is not limited to race, beards, eyeglasses or facial expressions. It is known that a detection system that was tuned for adult faces would be less accurate on infants. The sex, changing hair style, and facial ornaments would adversely affect the detection accuracy of automated systems.

Object detection is a still open research area in the image processing field. Moreover, there is an increasing demand for commercial products that are capable of analyzing scenes. There have been very significant improvements in the face detection field in the last fifteen years (Rowley 1998, Viola Jones 2001)[6, 5].

1.1 Human Face Detection

As a prior step of face recognition, tracking, verification, and facial expression analysis, detection of human faces has attracted interest as a research topic. A face detection algorithm must be robust to interpersonal differences between faces, and possible variations of a human face from one snapshot to another. In this thesis we study one very successful algorithm proposed by Viola and Jones [6]. We also propose an improvement to the Viola Jones algorithm to improve the learning speed and face detection accuracy in general images.

1.2 Motivations

As we have already discussed in section 1.1, the algorithm proposed by Viola-Jones is based on Haar-like features that are iteratively selected and weighted for best detection accuracy. There are however two challenges in implementation of this algorithm:

1. Speed of learning: It is stated [6] that the Adaboost learning implemented in this algorithm takes in the order of weeks[6] for complete training of the face detection system. This procedure is offline and done once, however the time complexity does not allow re-training of the architecture with new faces or for completely new objects.

2. Limitations of Haar-like features: The features used in the Viola-Jones algorithm consist of adjacent rectangular fields of black and white pixels. This puts a constraint on finding the best features that define a human face or an object in general.

   In this research we attempt to lift both constraints in the Viola-Jones algorithm. We evaluate our system on detection of faces, and specifically on the dataset used by Viola and Jones. This decision was made for the sake of comparing the two approaches. Our algorithm is not specific for human faces, i.e. it can quickly be trained and used for detection of any other object in general sceneries.

1.3 Challenges

The biggest challenges in face detection in outdoor images are;

- Speed requirements: The commercial use of face detection algorithm dictates real-time processing of images. This is a challenging task as an 'n' row and 'm' column gray scale image can be considered as an nxm dimensional feature vector.

- Head orientation: Depending on the perspective of the camera and the orientation of a person, the appearance of a human face varies drastically.
- Lighting variations: In an image, ligthing of head produces non-linear effects on the pixels values of an image.
- Expression variations: The appearance of a human face can very drastically with respect to expressions such as blinking, frowning and smiling.
- Variation in scale: The size of the human face varies depending on the distance of the subject and camera parameters.
- False positives: The background scene is often complicated, causing false positives.
- Blockage/occlusion: Eyeglasses, beard, moustache, make-up, and long hair produce adverse affects in detection.
- Variation among individuals: The learning algorithm employed in face detection must be flexible enough to accommodate interpersonal variations.

All these problems that are listed above are known to increase false positives in detection of faces. Most of these challenges can be dealt with a flexible learning algorithm and a database that embeds the related variations. In this research, we use grayscale images of faces and a very large database of non-face objects. Our learning algorithm leverages on finding the best features that eliminate the highest number of non-face images.

1.4 Approach

In this research, we propose a face detection algorithm that is based on Viola and Jones' work. The input to our trained face detection system is a gray scale image that may or may not have a human face on it. We use the features that are extracted

in the training phase to detect possible faces. The output is the positions and sizes of faces that appear in the input image.

Our training algorithm aims to detect best features that represent human faces. These features are based on rectangular black or white regions, that may or may not be adjacent. These features are evaluated with respect to their capacity of eliminating the highest number of non-face objects. Selected features are merged and the training is iterated. Here we explore strategies for selecting the features to merge.

1.5 Structure of Thesis

In this thesis, the challenge of face feature detection has been attempted by using Viola Jones algorithm. A new method for extracting facial features is proposed and implemented. The structure of our thesis is as follows.

Chapter 2: This chapter discusses the techniques and recent developments in the field of computer vision. In subtitles, image processing, object detection and face detection have been reviewed. Selected studies on face detection is discussed for clear understanding of this thesis.

Chapter 3: This chapter focuses on methods of face feature detection. Eigenfaces and neural networks methods are visited.

Chapter 4: This chapter introduces Viola-Jones algorithm. This algorithm has been explained in detail for clear understanding further chapters on our implementation.

Chapter 5: A new algorithm on facial feature extraction has been presented. Coding techniques for face feature detection is introduced. Also some pre-approaches has been given. Databases are reviewed and chosen database has been stated.

Chapter 6: Test results of designed algorithm has been presented. We comment on the performance of the algorithm. Also comparision with other algorithms is given for an objective perspective on success of the algoritm.

Chapter 7: This thesis concludes with the summary of the work reported in this thesis, discussion on some of the open problems and the scope of further research in this area.

CHAPTER II

COMPUTER VISION

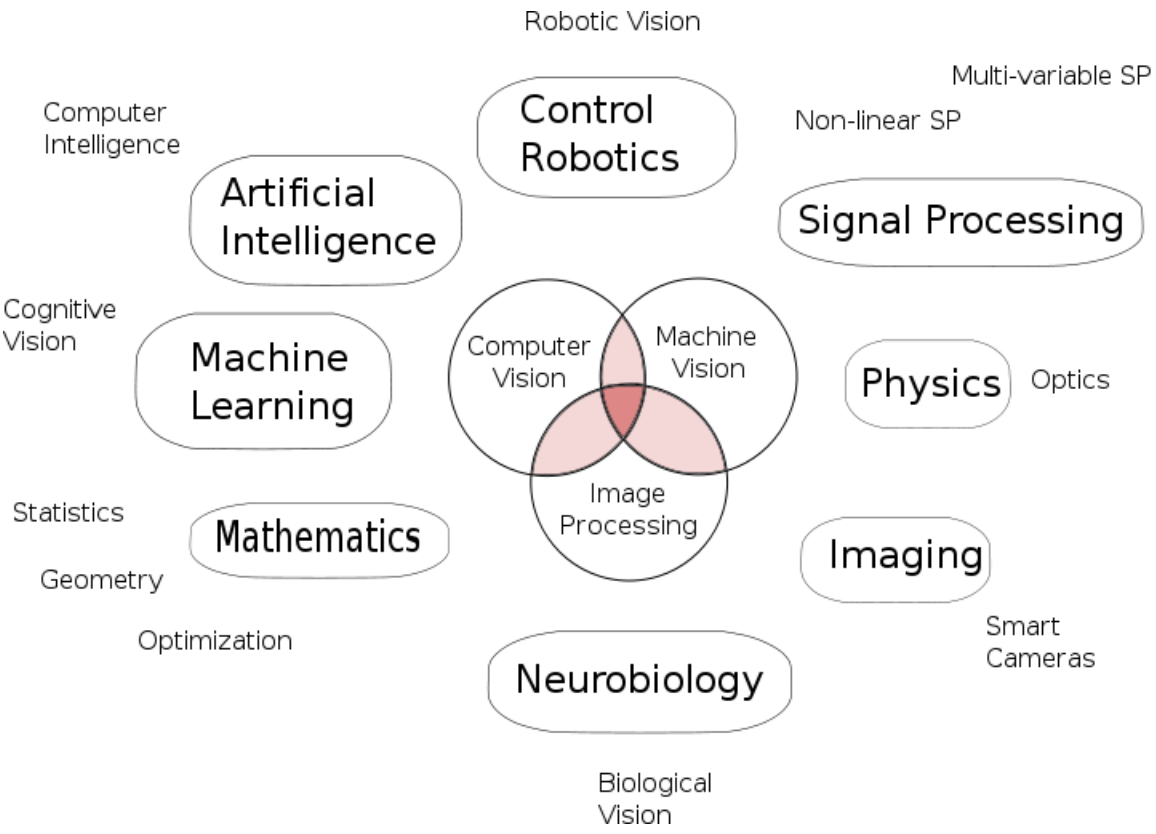


Figure 1: General scheme of Machine Vision.

Computer vision is analysis of visual inputs (figure 1) and it can be effected by several factors like camera model, lighting, color, texture, shape and motion that affect images and videos. A rough structure of computer vision is illustrated in figure 2. It is accepted that , computer vision is a more challenging problem than computer graphics, because of its uncertainties.

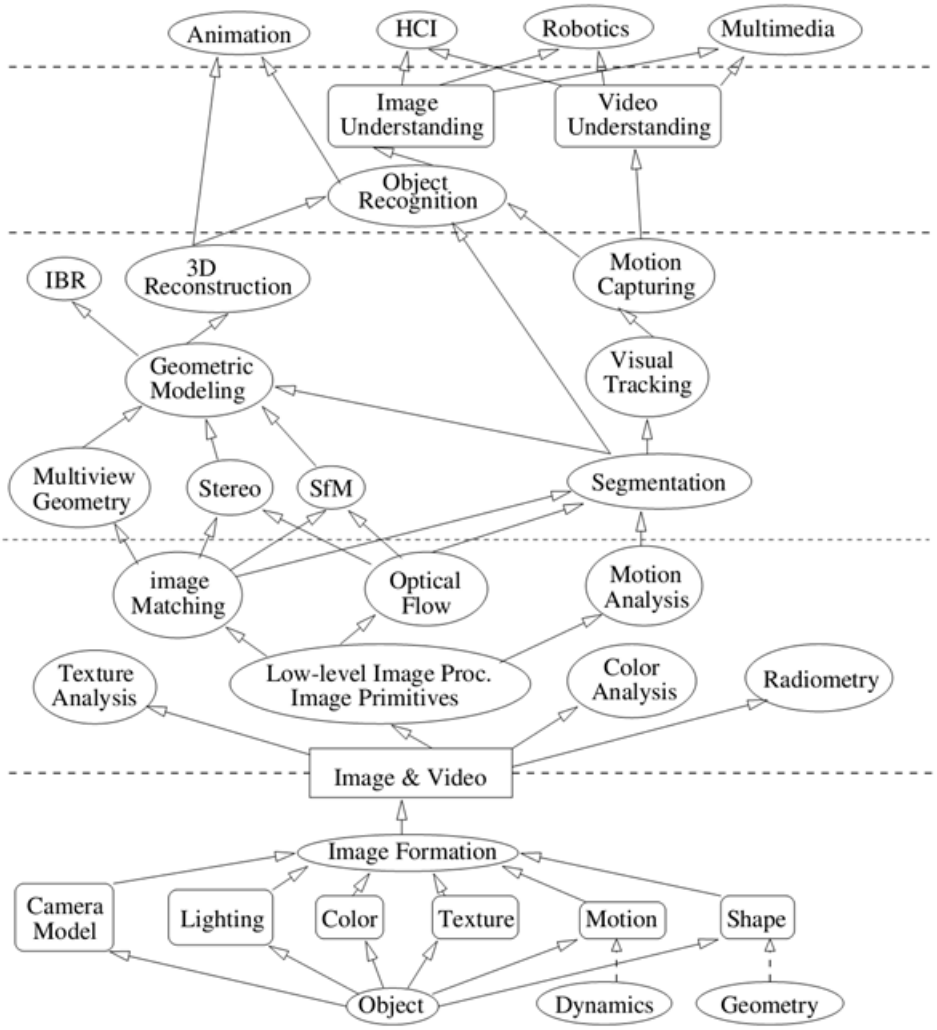


Figure 2: What is computer vision?

One of the primary problems in computer vision is the lack of reliable data. Although image data in some modern digital camera looks extremely appealing to the human eye, even small amount of noise can throw off algorithms rather easily, especially when working with sequences of images (video). Data can be corrupted by many sources of noise which is barely noticeable by humans as shown in [7]. Moreover, the appearance of objects in a scene can change in time due to fluctuations in light, that can be caused by the camera's aperture settings or shutter speed, by

clouds, or due to the natural oscillation of typical AC light source. The stochastic nature of computer vision problems also require suitably styled solutions. Since commonly we approximate badly understood processes linearly, we also require computationally intensive algorithms from linear algebra such as determining the inverse of large matrices, finding eigenvectors, performing singular value decomposition, etc. The complex nature of computer vision problems requires innovative solutions. Algorithms have to be designed in a flexible way, and utilize the available hardware accordingly. When designing computer vision systems, parallelism in both the code and the hardware can be exploited. It is also essential to carefully select an adequate hardware platform for each application.

2.1 Image Processing

In this section, we will cover the basic topics of image processing; histogram equalization, edge detection and smoothing. These procedures are usually termed as preprocessing. The main aim of preprocessing is enhancement of image or elimination of irrelevant data.

2.1.1 Histogram Equalization

Histogram is a graph which shows the number of pixels in an image at each different intensity value found in that image [8, 2]. Histograms can be same for completely different images so it is not a unique identifier of an image. Histograms are mostly used for performing optimal contrast improvement [2], therefore histogram equalization brings better results in many cases. An example of histogram equalization is displayed on Figure 3. In this thesis, Histogram Equalization is used for database normalization.

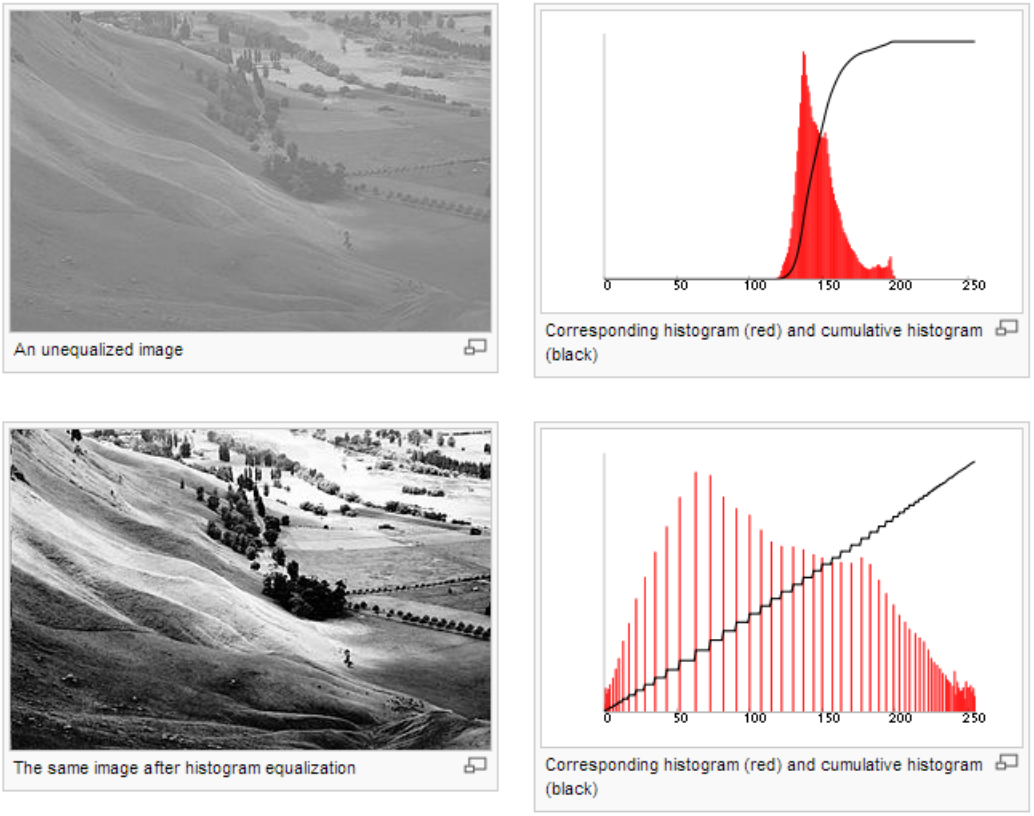


Figure 3: An example of histogram equalization [1]

2.1.2 Smoothing

Smoothing is low pass filtering related to neighborhood operation for the purpose of noise removal. Smoothing is usually implemented in two different ways: first is median filtering and second is averaging. Median filtering is based on selection of the median grey level in a neighborhood. Median filtering is a powerful scheme for smoothing. However it is computationally costly since it makes sorting operation for neighborhood of each pixel. A median filtering example is demonstrated in Figure 4.

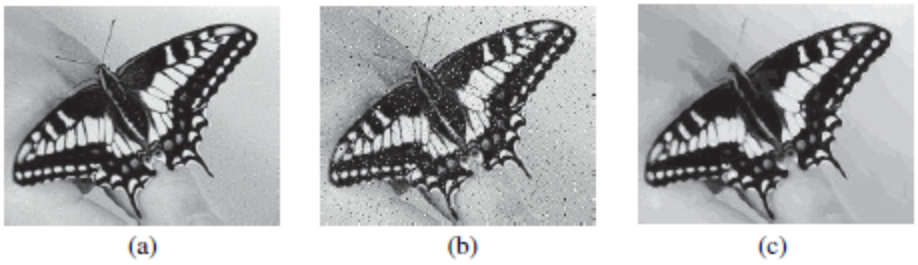


Figure 4: Smoothing with median filtering. a)Original image b)Noisy (Salt and pepper noise) image c)Smoothing of (b) with median filtering [2].

Due to computational considerations, averaging is more preferred than median filtering. Averaging type smoothing is related to convolving the image with a kernel containing the averaging weights. Results of averaging with varying kernels are given in Figure 5.

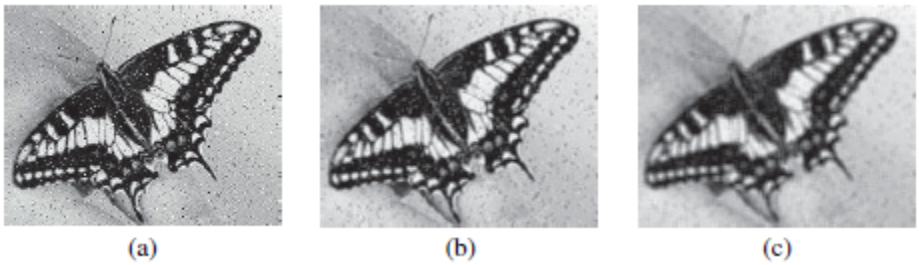


Figure 5: Smoothing with different averaging kernels. a)Original noisy image b)Smoothing with a 3x3 averaging filter c)Smoothing with a 5x5 averaging filter[2].

2.1.3 Edge Detection

Edge detection is used for isolating the edges on an image, and it is one of the most important preprocessing tools in image processing. Recognizing an object and extracting its uniqe identities is generally main aim of image processing and edges are important since they store important information related to objects. Detecting sudden brightness changes in pixels gives edges of images. Edge detection is performed

by taking derivatives of each pixel on horizontal and vertical dimensions. These derivatives provide the magnitude and orientation of edge pixel. There are different kernels that have been developed for calculating the horizontal and vertical derivatives on images. A few examples are demonstrated in Figure 6.

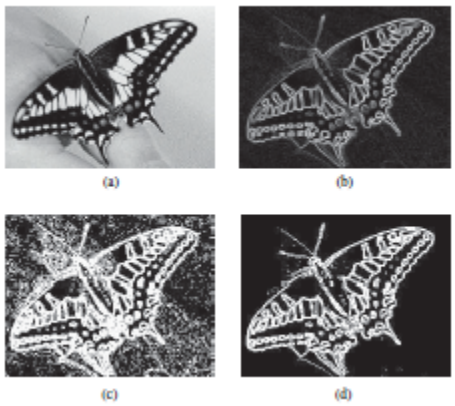


Figure 6: Edge detection using Sobel kernels. a)Original image, b)Edge magnitudes, c)Edge magnitudes thresholded with T=50, d)Edge magnitudes thresholded with T=150)[2].

2.2 Object Detection

Object detection is a difficult research problem, especially when the background is cluttered. Almost all visual systems are interested in foreground objects. Extraction of foreground objects reduces data that are needed to be processed for higher level jobs like tracking or classification. Sudden changes in lighting, shadows or noise makes accurate object detection difficult. On that count, to design a reliable and fast visual system, it is neccesary to precisely extract the foreground objects in images.

Fast and reliable object detection is the main target of next generation image and video processing systems. Object detection is specifically interested on objects like human faces, cars, cats, dogs or bikes. On the other hand, object recognition is related to understanding the identity of a specific object like my face and another

face. For people recognizing other peoples' faces is an easy task. After millions of years of evolution process human brain has been evolved in a perfect capacity for detection and recognition faces. However algorithmic approaches still cannot match up with humans in face recognition.

2.3 Face Detection

Before face recognition is possible, one must be able to reliably find a face and its landmarks. This is essentially a segmentation problem and in practical systems, most of the effort goes into solving this task. Face detection is the first step of face recognition as it automatically detects a face from a complex background for which the face recognition algorithm can be applied. But detection itself involves many complexities such as background, poses, illumination etc. There are two types of face detection problems which are face detection in images and real-time face detection.

2.3.1 Face detection in images

Most face detection systems attempt to extract a specific region of the face, thereby eliminating most of the background and other areas of an individual's head such as hair that are not as informative for the face recognition task. With static images, this is often done by running a 'window' across the image. The face detection system then judges if a face is present inside the window [9]. Unfortunately, with static images there is a very large search space of possible locations of a face in an image. Faces may be large or small and be positioned anywhere from the upper left to the lower right of the image.

Most face detection systems use an example based learning approach to decide whether or not a face is present in the window at that given instant [10]. A neural network or some other classifier is trained using supervised learning with 'face' and 'non-face' examples, thereby enabling it to classify an image (window in face detection system) as a 'face' or 'non-face'. Unfortunately, while it is relatively easy to find

face examples, it is quite difficult to find a representative sample set of images which represent nonfaces [5]. Therefore, face detection systems using example based learning need thousands of 'face' and 'non-face' images for effective training. Rowley, Baluja, and Kanade [5] used 1025 face images and 8000 non-face images (generated from 146,212,178 sub-images) for their training set.

Another technique for determining whether there is a face inside the face detection system's window is template matching. The difference between a fixed target face pattern and the window is computed and thresholded. If the window contains a pattern which is close to the target face pattern then the window is judged as containing a face. By using several templates of different fixed sizes, faces of different size scales are detected. One implementation of template matching is using a deformable template [11].

A face detection scheme that is related to template matching is image invariants. Here the fact that the local ordinal structure of brightness distribution of a face remains largely unchanged under different illumination conditions is used to construct a spatial template of the face which closely corresponds to facial features. In other words, the average grey-scale intensities in human faces are used as basis for face detection. For example, almost always an individuals eye region is darker than his forehead or nose. Therefore an image will match the template if it satisfies the 'darker than' and 'brighter than' relationships. [10].

2.3.2 Real-time face detection

Real-time face detection involves detection of a face in a series of frames from a video capturing device. While the hardware requirements for such a system are far more stringent, from a computer vision stand point, real-time face detection is actually a simpler process than detecting a face in a static image. This is because unlike most of our surrounding environment, people are usually in motion. Since in real-time face

detection, the system is presented with a series of frames in which to detect a face, by using spatio-temperal filtering (finding the difference between subsequent frames), the area of the frame that has changed can be identified and the individual is detected [12]. Real-time face detection has therefore become a relatively simple problem and is possible even in unstructured and uncontrolled environments using these very simple image processing techniques and reasoning rules.

CHAPTER III

HUMAN FACIAL FEATURE DETECTION

Facial feature extraction is a popular area of image procesing, since it is certainly prework before face recognition and facial feature expresion which are also ongoing popular research areas[13, 14, 9]. This chapter will first present some of the important studies on face detection algorithms.

3.1 Eigenfaces

A set of eigenvectors used in the computer vision problem of human face recognition are called eigenfaces. In eigenfaces approach, faces are points in a vector space called as eigenspace. Face detection is based on computing the distance of image windows to the space of faces. Raw images has many points in a high dimensional space and working with them is impractical for two reasons. First reason, in such a high dimensional spaces, there should be very high computational power and that means high cost. Second reason is, in raw images there are statistically irrelevant datas which downgrades the performance of the system. The basic principal is reducing the dimensionality of space by restricting the attention to some directions which is the scatter is greatest [15]. Thus, principal component analysis is used for projecting the raw face images onto the eigenspace of a representative set of normalized face images, for eliminating redundant and irrelevant information. Figure 7 depicts the standard eigenfaces presented in the eigenfaces demo web page of the MIT Media Lab [16].

Eigenfaces approach is used for both detection and recognition of faces. For the recognition part, the Euclidian distances of a given image to each of the images in the training set is computed in the eigenspace and recognition is performed by choosing the shortest distance image in the training set. For the detection of faces,

the distance to the space of faces (eigenspace) is used. Distance to the space of faces is the reconstruction error, which occurs when an image is projected onto the face space and then projected back to the original space. If the distance of a given image from the eigenspace is less than a predetermined threshold value then the image is accepted as a face image.

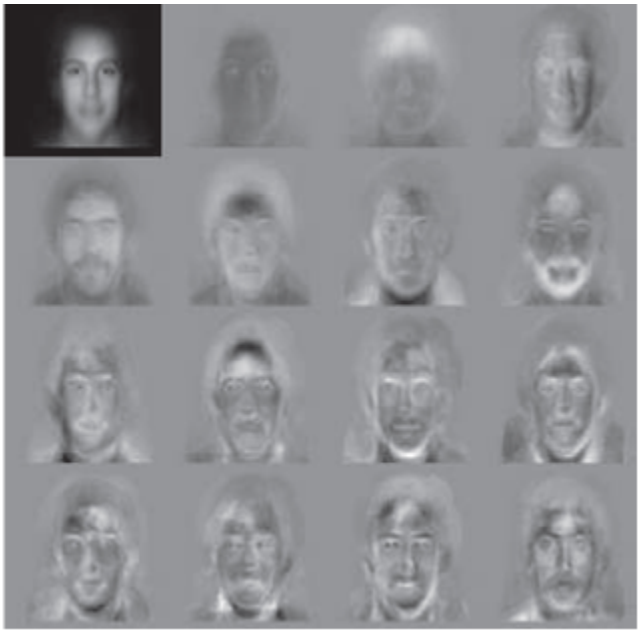


Figure 7: Standard eigenfaces.

3.2 Neural Networks

There are many approaches for face detection such as, colour based, feature based (mouth, eyes, nose), neural network. Neural networks, especially multi layer perceptron, have been used by many researchers for face detection and face recognition, and promising results were reported [17]. The first advanced neural approach which performed reasonably on a large and diffcult dataset was Rowley et al. [18]. The system described in [5] incorporates prior knowledge of face in a retinally connected neural network as shown in figure 8. The system works in two stages: filtering, and

arbitrating. Filtering is performed using neural networks at each location in the image at different scales for the occurrence of a face. Then, arbitrating is performed for merging and eliminating the overlapping detections. The first component of the system is a neural-network based filter that receives a 20x20 pixel region of the image, and signals the existence or nonexistence of a face in the region by outputting a number between -1 and 1.

The system aims multi scale face detection by repeatedly sub-sampling the input image by a factor of 1.2, and re-applying the filter at each scale.The filtering algorithm performs lightning correction and histogram equalization before testing the existence of a face using a neural network. The neural network has retinal connections to its input layer, and the hidden layer contains three different types of hidden units which are chosen to allow the hidden units to represent features that might be important for face detection. As a last step the system merges overlapping detections, and eliminates some of the false detections by employing a simple heuristic.Their system was trained using a very large set of face images at different scales and rotations. They performed tests on a variety of images from newspapers, web, and TV images, and they reported up to % 92.7 face detection rates[5].

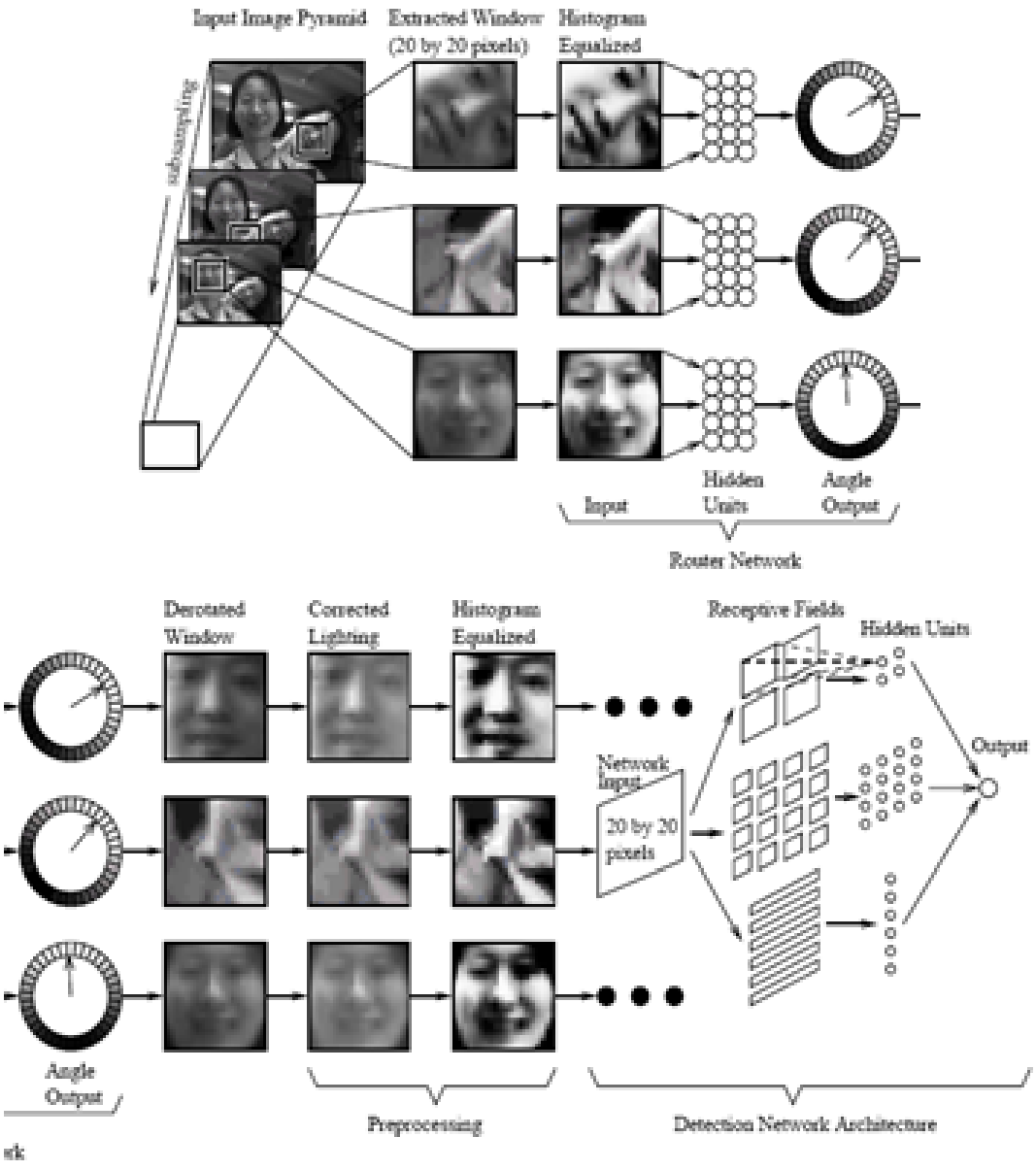


Figure 8: Face detection through neural networks.

CHAPTER IV

VIOLA-JONES

Viola Jones refers to a paper written by Paul Viola and Michael Jones describing a method of machine vision based fast object detection. This method revolutionized the field of face detection. Using this method, face detection could be implemented in embedded devices and detects faces within a practical amount of time. They describe an algorithm that uses a modified version of the AdaBoost machine learning algorithm to train a cascade of weak classifiers (Haar features). Haar features (along with a unique concept, the integral image) are used as the weak classifiers. Weak classifiers are combined using the AdaBoost algorithm to create a strong classifier. Strong classifiers are combined to create a cascade. The cascade provides the mechanism to achieve high classification with a low CPU cycle count cost.

4.1 Introduction

The face detection framework proposed by Viola and Jones has three main concepts.

- First one is called "integral image". They bring a new method which is easy to process on an image.
- Second one is a learning algorithm, based on AdaBoost.
- Third one is a method for combining classifiers in a "cascade".

4.2 Haar Like Features

Haar features are one of the mechanisms used by the Viola -Jones algorithm. Images are made up of many pixels. A 250x250 image contains 62500 pixels. Processing images on a pixel by pixel basis is computationally a very intensive process. In

addition, individual pixel data contains no information about the pixels around it. Pixel data is absolute, as opposed to relative. A side effect of the absolute nature of pixel data is the effect of lighting. Since the pixel data is absolutely effected by the lighting of the image, large variances can occur in the pixel data due only to changes in lighting. Haar features solve both problem related to pixel data (CPU cycles required and relativity of data). Haar features do not encode individual pixel information, they encode relative pixel information. Haar features provide relative information on multiple pixels.

Haar features were originally used in the paper: "A General Framework for Object Detection"[19]. A Haar feature is used to encode both the relative data between pixels, and the position of that data. A Haar feature consists of multiple adjacent areas that are subtracted from each other. Viola - Jones suggested Haar features containing; 2, 3, and 4 regions.

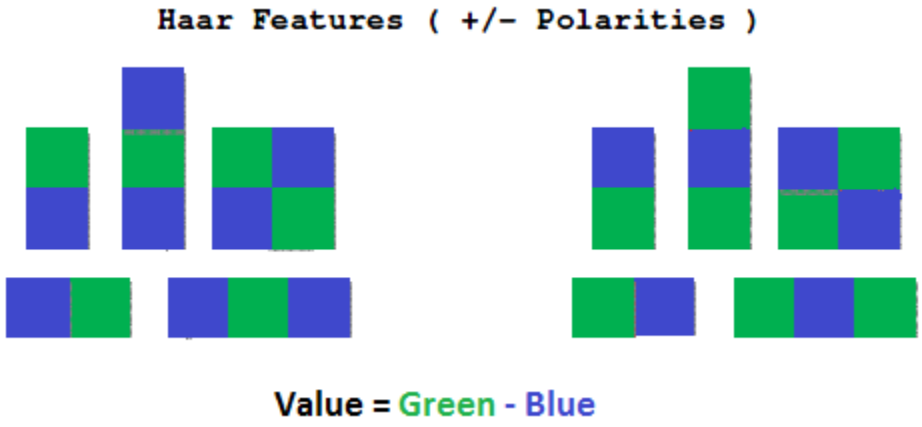


Figure 9: Haar Like 2,3 and 4 areas features.

The value of a Haar feature is calculated by taking the sum of the pixels under the green square and subtracting the sum of the pixels under the blue square.

$$value = \frac{1}{N}(\sum_{i=0}^{N} green_i - \sum_{i=0}^{N} blue_i)$$

By encoding the difference between two adjoining areas in an image, a Haar feature can effectively detects edges. The further the value is from zero, the harder or more distinct the edge. A value of zero indicates the two areas are equal, thus the pixels under the area have equal average intensities (the lack of an edge). It should be noted that although this process can be done on color images, for the Viola Jones algorithm this process is done on grayscale images. In most cases individual pixel values are from 0 to 255, with 0 being black, and 255 being white. The difference of pixel values indicate how harder edge there is. As shown in Figure 10, 0 means 'No edge' and 250 means 'Hard edge'.

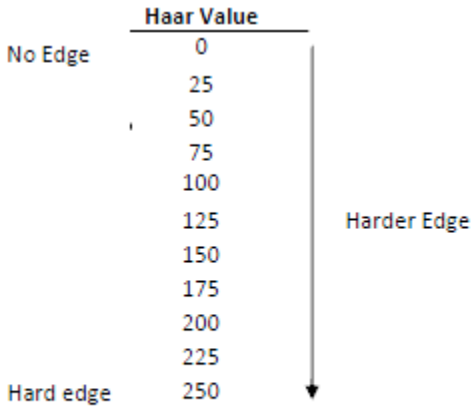


Figure 10: Relation of edge hardness and Haar value.

The values calculated using Haar features require one additional step before being used for object detection. The values must be converted to true or false results. This is done by using thresholding.

4.2.1 Thresholding

Thresholding is the process of converting an analog value into a boolean value. In this case, the analog value is the output of the Haar feature :

$$value = \frac{1}{N}(\sum_{i=0}^{N} green_i - \sum_{i=0}^{N} blue_i)$$

If the value is greater than or equal the threshold, the statement is true, if not it is false.

$$\mathbf{h_j(x)} = \begin{cases} 1 & \textbf{if } \mathbf{p_j f_j(x) < p_j \theta_j} \\ 0 & \textbf{otherwise} \end{cases}$$

Where: $h_j(x)$ - Weak classifier (basically 1 Haar feature)

P_j - Parity

$f_i(x)$ - Haar feature

θ_j - Threshold

As the equation above illustrates, the output of the weak classifier is either true or false. The parity determines the direction of the inequality sign. This will be demonstrated later by examples. The threshold and parity must be set correctly to get the full benefit of the feature. Setting the threshold and parity is not clearly defined in the Viola Jones paper. *"For each feature, the weak learner determines the optimal threshold classification function, such that the minimum number of examples are misclassified"[6].* Many theories have been proposed to calculate the threshold; e.g. minimum, average, standard variation, and average variation.

Figure 11: A three area Haar Feature tries to detect a nose.

Viola Jones found that a 3 area Haar feature across the bridge of the nose provided a better then average probability of detecting a face. The eyes are darker then the bridge of the nose.

$$value = \frac{1}{N}(2\sum_{i=0}^{N} blue_i - \sum_{i=0}^{N} green_i)$$

The value increases if the eye area gets darker, or the bridgeof the nose gets brighter. An example of a 3 area Haar feature illustrated on figure 11.

4.2.2 Parity

The parity variable is used to adjust the value so that it is above 0. An inverted feature

$$value = \frac{1}{N}(-2\sum_{i=0}^{N} blue_i + \sum_{i=0}^{N} green_i)$$

would present a value that is of the same magnitude with a different sign. The parity is used to convert the value of an inverted feature into a positive integer, bringing it above the zero line.

4.3 Integral Image

A key technique is used by Viola and Jones is the integral image. This technique minimizes the number of summations required to calculate the value of a single Haar

feature. A summed area table (also known as an integral image) is an algorithm for quickly and efficiently generating the sum of values in a rectangular subset of a grid [20]. The sum area table is an accumulation of pixel values, starting from the upper left and moving towards the lower right of an image. The value at any point (x, y) in the table is the sum of all the pixels above and to the left of that point. The summed area table can be computed efficiently in a single pass over the image, using the fact that the value in the summed area table at (x, y) is just calculating the sum of a region using a integral image.

$$I[x, y] = S[x + a, y + b] - S[x + a, y] - S[x, y + b] + S[x, y]$$

Where: $I[x, y]$ - Value of integral image on (x,y) coordinate

$S[x, y]$ - Value of summed area table on (x,y) coordinate

Calculation of an integral image is demonstrated on figure 12

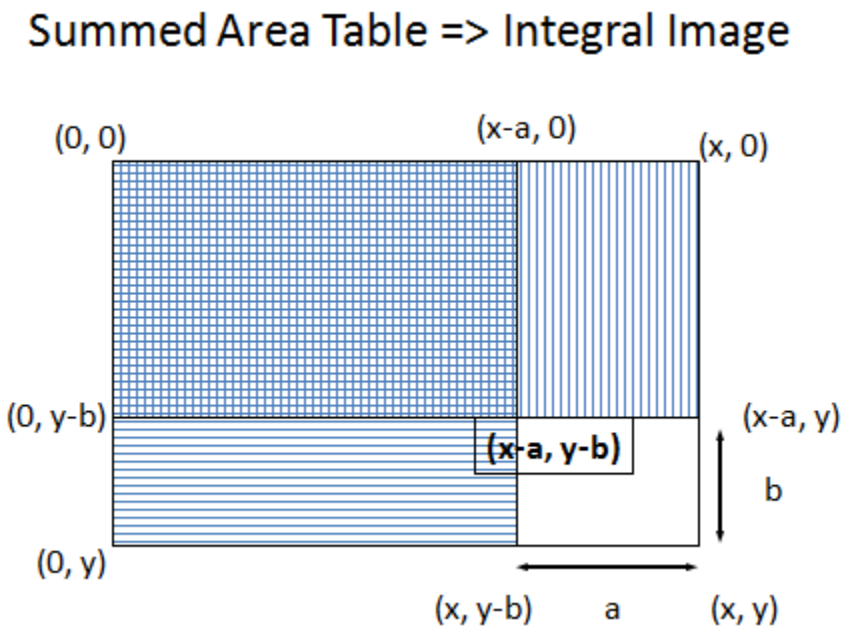


Figure 12: Calculation of an integral image.

4.4 AdaBoost Algorithm

AdaBoost creates a strong classifier by combining weak classifiers. Its a machine learning algorithm that creates a recipe for a strong classifier. The ingredients are the weak classifiers. AdaBoost calculates how much of the result of each weak classifier should be added to the final mix. An illustration of adaboost algorithm is shown in figure 13.

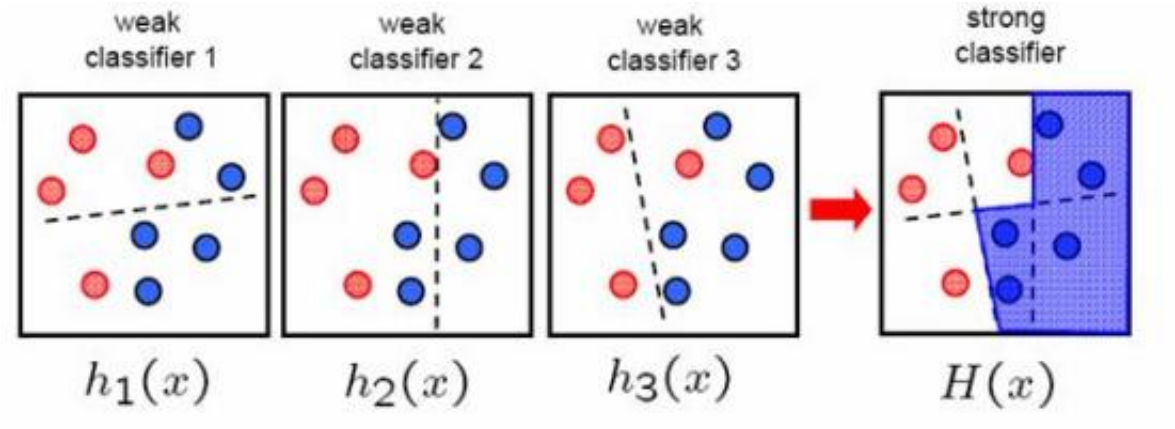


Figure 13: Illustration of adaboost algorithm .

4.4.1 Derivated AdaBoost in the Viola-Jones method

The result of the AdaBoost algorithm is the weights of each weak classifier. The algorithm description can be confusing because the *weight* term is used to describe multiple aspects of the algorithm. For this summary, the term weight is reserved for the final values calculated by the algorithm. AdaBoost learns by testing data (faces) and noise (nonfaces) against the weak classifiers. The data and noise is referred to as a distribution. The result of the test is used to score the distribution. If the weak classifier incorrectly classified an object in the distribution, the score for that result final strong classifier of weak classifier weight calculated by AdaBoost object is increased. If the weak classifier correctly classifies an object, the score for that object is decreased. This process is repeated for all weak classifiers.

4.5 Cascade

A classifier sorts noisy information, letting the desired information pass through, while filtering out the undesired information. Like both electrical and mechanical filters, it can be more efficient to perform the classification in stages. Viola-Jones uses a classifier cascade to increase the computational efficiency of their method. Each stage in the cascade has progressively more Haar features, requiring progressively more computations. The first stage has the least number of weak classifiers, requiring the least number of computations. If the first stage identifies a face, it passes the data (integral image) to the next stage. On the other hand, if the first stage does not see a face it rejects that portion of the integral image and moving on to the next portion. The cascade decreases the number of computation required when scanning across an image, because the majority of the image does not contain faces. It minimizes the number of calculation based on the premise that the majority of the integral image does not contain a face. When a face is detected, the number of calculations that occur, is the same as if a cascade was not used. Rough scheme of cascade of face detection is demonstrated on figure 14

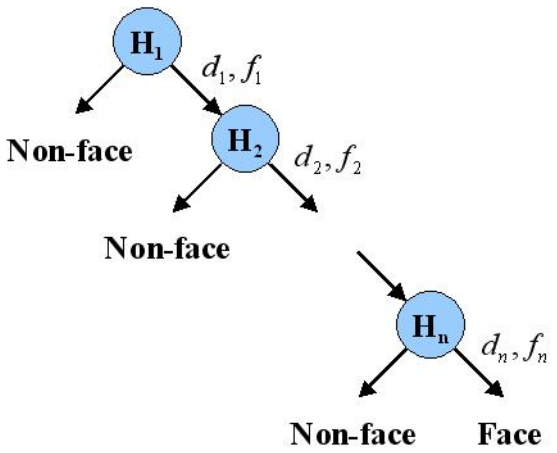


Figure 14: A simple scheme of cascade of face detection.

4.6 Conclusion

Viola-Jones described a method of fast face detection in computer vision by combining four techniques; Haar features, the integral image, Adaboost, and the cascade. The methods described in the Viola-Jones paper made it possible to perform practical face detection with minimal computing power. Haar features are a powerful tool for classifying images like faces. Haar features classify a region of an image based on differences between the pixels of the two symmetrical sides of the feature. This results in a form of 'edge detection'. Combining Haar features together, creates a strong classifier or detector. Boosting is a machine learning method used to determine the best combination of weak classifiers to optimize the resulting strong classifier. Adaboost is a derivative of boosting that scores the result of a weak classifier when used against a distribution. The score is used to calculate a weight that is used to multiply the result of the weak classifier in the final strong classifier. The final strong classifier that results from the Adaboost method is a sum of all the weights multiplied by the results of the weak classifiers. Boosting is only done during the learning phase of the Viola-Jones method. After the weak classifiers have been selected, and the weights calculated, the resulting strong classifier can be used to start filtering desired data out of noise. After the strong classifiers (or filters) have been learned, they are combined into a cascade. Cascade is just another term for multiple stages.

The goal of using multiple strong classifiers organized into stages is to decrease the total number of computations required by the filter. The first stage is a prefilter, designed to catch the majority of noise (non-faces). It has the least number of weak classifiers (haarfeatures) requiring the least number of computations. The following stages progressively increase the number of weak classifiers, but also filter out more noise than the previous stage. After the resulting strong classifiers have been learned using Adaboost, and organized into a cascade, it can be used to start filtering out noise to find the desired data. For face detection, the data is face, and the noise is

anything that is not a face. After the cascade has been learned, the classification process is computationally simple. Steps are as shown below.

- Convert the image into an integral image.
- Slide the detector full of Haar features over the integral image.
- For each position, calculate all the Haar features required by the first stage of the cascade, and multiply each of them by their corresponding weights that resulted from the Adaboost algorithm.
- Threshold the result into a true or false (face under detector, or face not under detector).
- If the first stage returns a true, then use the integral image to calculate the Haar features in the second stage of the cascade.

This process is repeated until a stage either determines that no face is under the detector, or the final stage determines that a face is under the detector. Sample scheme of Viola-Jones algorithms demonstrated on figure 15.

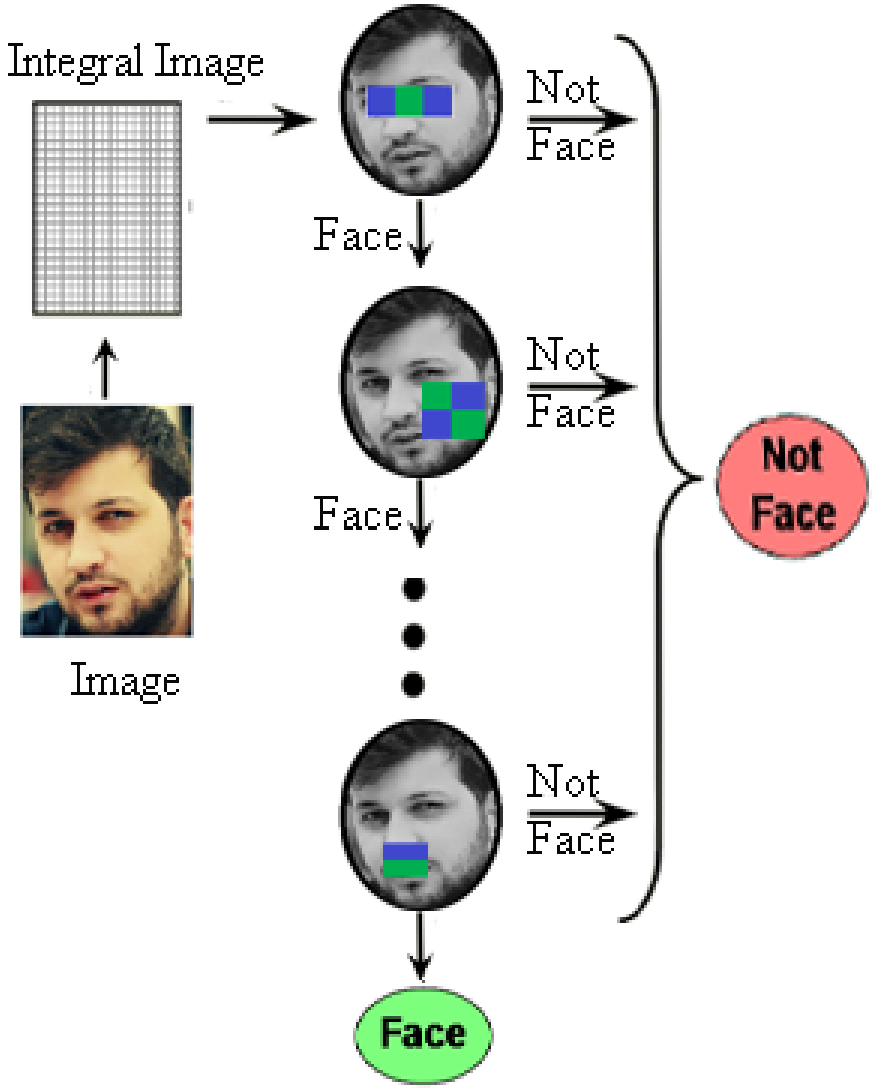


Figure 15: Sample scheme of Viola-Jones algorithms.

CHAPTER V

IMPLEMENTATION

Our first step was detection of circle object. In this case we used only 1-2 derivated haar box features which created by manually. One of our first sample is displayed on figure 16 and figure 17.

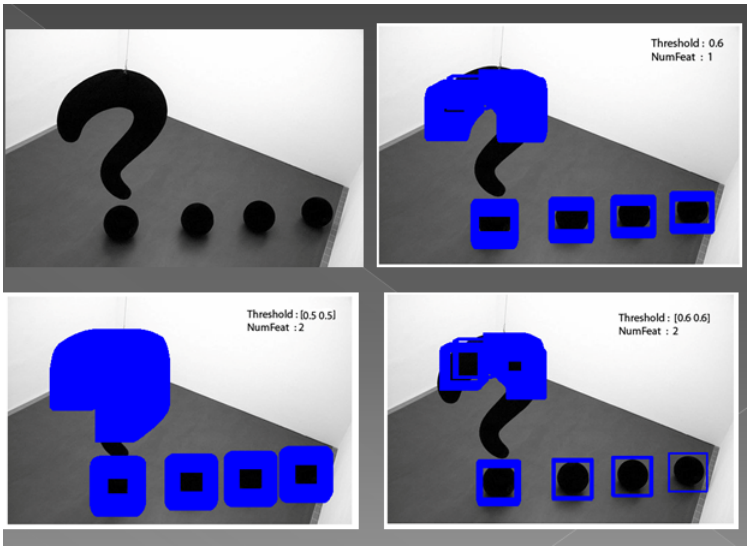


Figure 16: Detecting a circle with various thresholds.

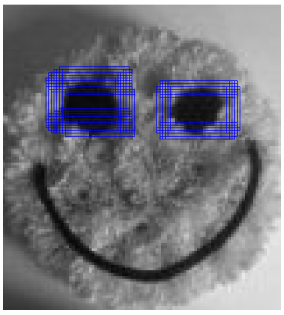


Figure 17: A sample of detecting circle. 2 features are used.

In this progressing, we realized the importance of 'thresholding'. Determining

a reliable threshold is one of the most challenging parts of our thesis. First goal is exploring a new method that can determine a feature and a threshold which can detect a face more accurately. Our second goal is implementation of the new algorithm that should be easy and fast. In this process we focused on relationship of the features. We tried to catch the patterns of facial features and merge them. Our third goal is creating as less as possible number of 'super' features. In Viola-Jones training almost all possible scale and positions of feature are calculated. The training with such large number of features takes several weeks. Our fourth goal is, keeping the training time as short as possible.

5.1 Coding

In this section, we illustrate main parts of our algorithm step by step. Preprocessing is displayed on figure 18.

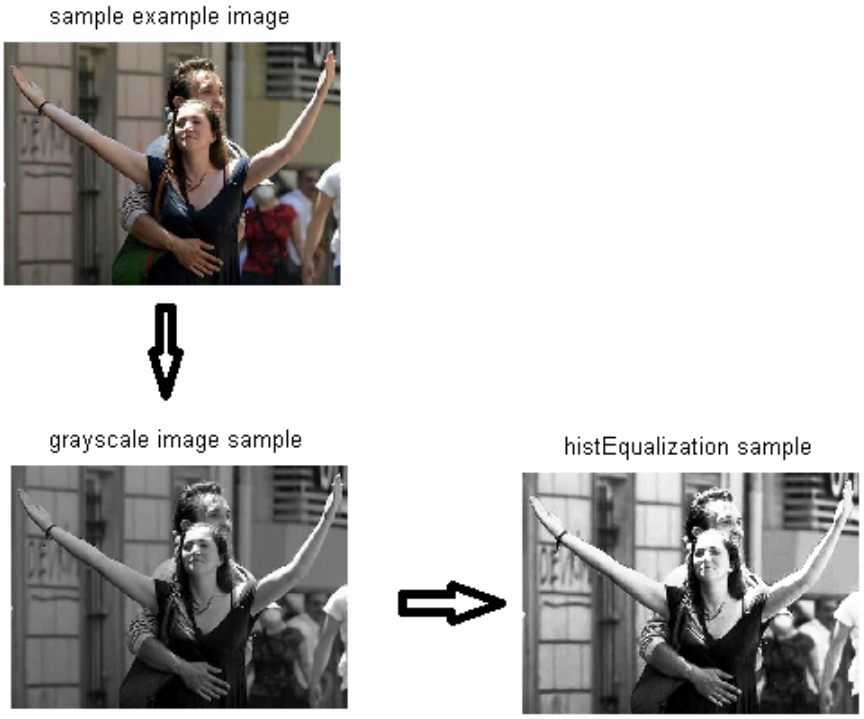


Figure 18: Preprocessing steps of our algorithm.

Our implementation code is written in Matlab. We haven't used any package algorithm.

5.1.1 Preprocessing

In preprocessing part, we do a couple of operations on images for decreasing the complexity of images. After the preprocessing part, datasets get more suitable and the algorithms can return better results.

1. Convert to Grayscale: We use 'rgb2gray()' matlab command which converts the truecolor image RGB(Red Green Blue) to the grayscale intensity image. rgb2gray() converts RGB images to grayscale by eliminating the hue and saturation information while retaining the luminance[21]. A sample is ilustrated in figure 18. Usually an image presented [0x255]x3 channels (RGB) on the digital envoirenment. After grayscale processing, the RGB 3 channels are reduced to one channel. The matrix which present an image gets [0x255]x1. It equals to [0x255].

2. Histogram equalization : We discussed about histogram equalization on subsection 2.1.1. We benefit from 'histeq()' matlab command which enhances the contrast of images by transforming the values in an intensity image. After histeq(), values of the image matrix are between {0 255} still. Sample example is ilustrated on figure 18.

3. Normalization: We decided to do a normalization on our datasets for more distinguishable threshold. Our normalization sets up the image matrix from {0 255} to {-1 1}.

4. Integral Image: Integral image topic is described on section 4.3. A sample is ilustrated in figure 19. There is not a matlab command for this procedure. We can calculate a summed area matrix when 'cumsum()' function is used twice.

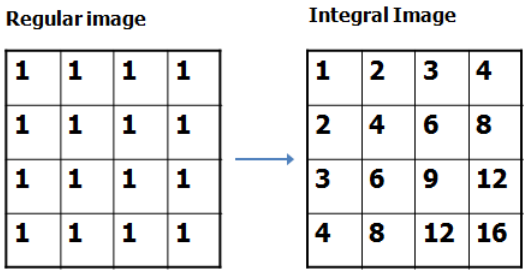


Figure 19: sample of Integral Image calculation.

5.1.2 Face Detection Algorithm

1. Create Random Features:

   In this part, we create white and black boxes in given size. As an example, assume that the size of the main image is 20x20. We can select size of subwindow as 1x1, 2x2, ...5x5 . These subwindows are our pure random features. We call them 'random' because in future steps they will be merged automatically based on merged feature success. Some example subwindows are shown in figure 20, 21.

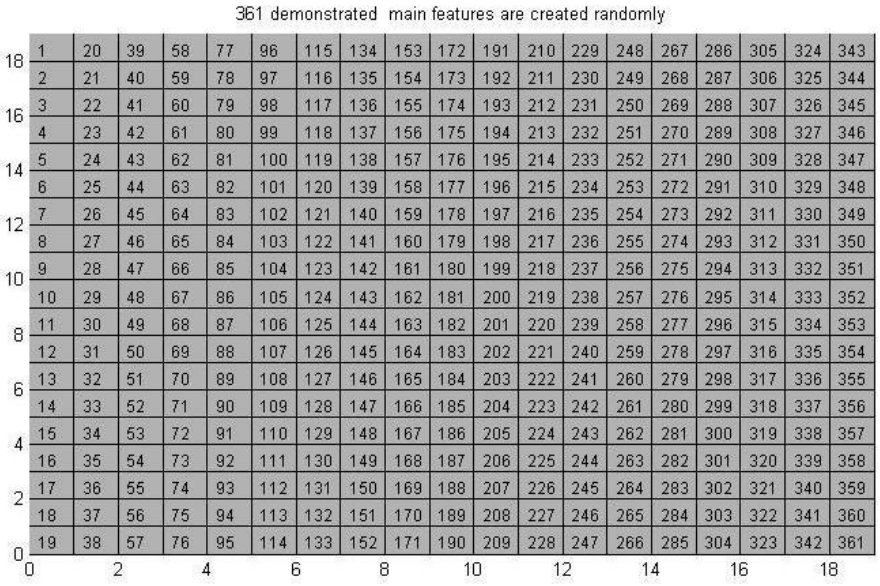


Figure 20: Samples of subwindow. (1x1 subwindow in 19x19 image)

In figure 20, the main image has 19*19 = 361 pixels. size of subwindow is chosen as 1x1. It means there are 361 random features created initially.

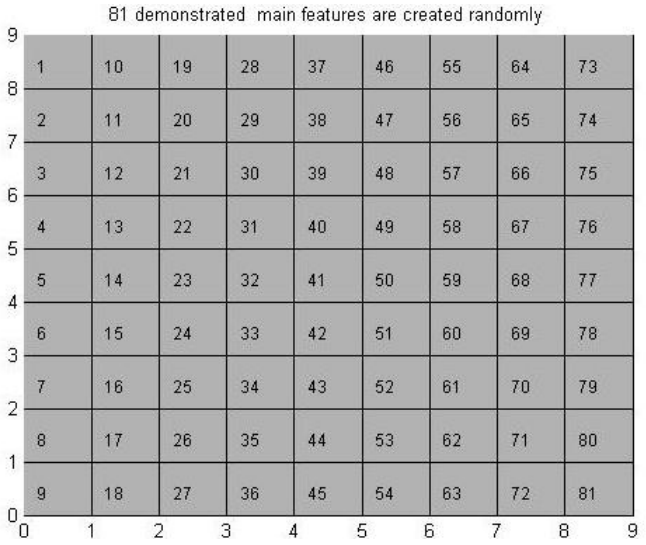


Figure 21: Samples of subwindow. (2x2 subwindow in 19x19 image)

In figure 21, the main image has 19*19 = 361 pixels. size of subwindow is chosen as 2x2 .In this case there are 81 random features created initially.

2. Threshold Filtering:

   Usually a threshold is used for labelling and classification. We used thresholds for classification. It can help to classify face and non-face depending on a feature value. Here feature value presents a result value which is calculated by integral image of a given feature.

   A threshold value can be between {-1 1}. If the value is near to -1, a chosen feature is opposite with a subimage. This situation tells us the feature is unsuitable for specified position of the image. If the value is near to 1, a chosen feature is suitable with a subimage. It shows the feature is valuable.

   When we discuss about a threshold, confusion matrix need to be known. There

are four term of confusion matrix which are FP,TP,FN and TN. Confusion matrix scheme is more understandable which is shown on figure 22.

		Predicted Label	
		positive	negative
Known Label	positive	TP	FN
	negative	FP	TN

Figure 22: A scheme of confusion matrix

Confusion Matrix:

TP - True Positive - Correctly detected

TN - True Negative - Correctly detected

FP - False Positive - False alarm

FN - False Negative - Miss call

- Realistic: As shown in figure 23 In other word, we can call 'mean' threshold filtering. The mean threshold is the average of the value for images of positive database which contains a face. This is done by summing a pure feature values obtained in the same position over the face on positive database, and dividing by the number of images.

$$Thres_{realistic} = \frac{1}{N+M}(\sum_{i=1}^{N+M} h_{j(i)})$$

Where: h_j - a feature value in same position on all images

i - single image

N - Number of positive images

M - Number of negative images

Realistic threshold filtering is near to real life. It can have FP.

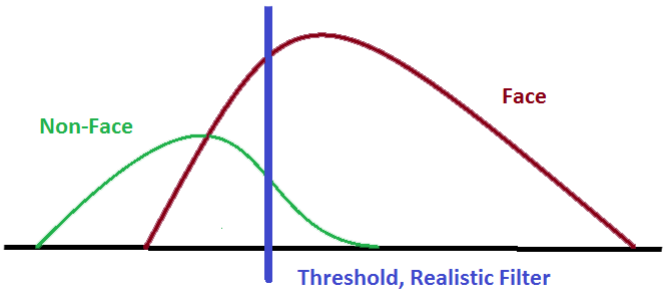


Figure 23: Threshold, Realistic Filter

- Pessimistic: As we can understand from 'pessimistic' vocabulary, the pessimistic threshold filtering acts very harshly on non-face. It doesn't allow any false-positives on the training set. (FP=0). It can detect all negative samples (non-face). This is done by taking maximum value of non-face datasets.

$$Thres_{pessimistic} = \max(h_{j(i)}|_1^M)$$

Where: h_j - a feature value in same position on all images

i - single image

M - Number of negative images

Pessimistic threshold filtering ilustration is shown on figure 24.

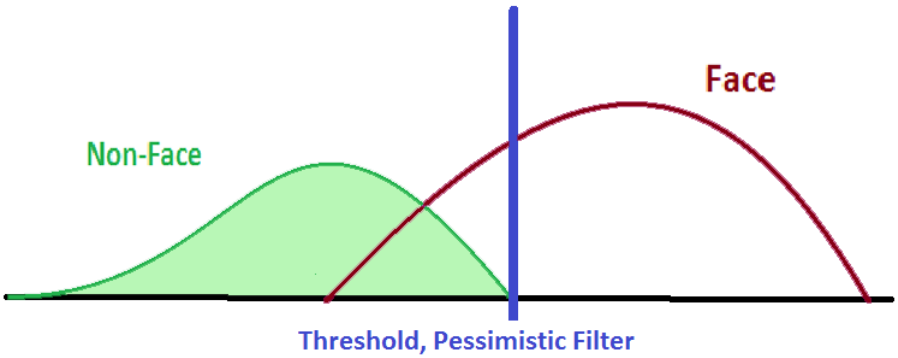


Figure 24: Threshold, Pessimistic Filter

- Optimistic: Optimistic threshold filtering acts like all faces have to be detect correctly. There can be 'miss call' which a non-face is detected as face. But every face is more valuable than miss-call for optimistic threshold filtering. In this case FN=0. This is done by taking minimum value of all face datasets.

$$Thres_{optimistic} = \min(h_{j(i)}|_1^N)$$

Where: h_j - a feature value in same position on all images

i - single image

N - Number of positive images

Shown on figure 25.

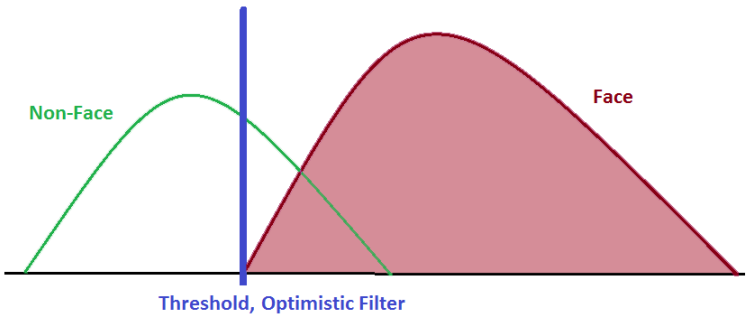


Figure 25: Threshold, Optimistic Filter

3. Majority on Random Feature Polarity:

In our work, A random feature has polarity which can be white (1) or black (-1). Initial polarity of the features is white. We update polarity of the features by majority. Which one (white or black polarity) is more suitable on place of chosen feature, it can be polarity of the feature. A sample is ilustrated on figure 26. In here gray pixels present white polarity and black pixels present black polarity. Increasing of size of subwindow (feature) is not preferred. For our work, 2x2 is the most suitable. 1x1 is the best choice. But 1x1 brings more

computational cost in training process.

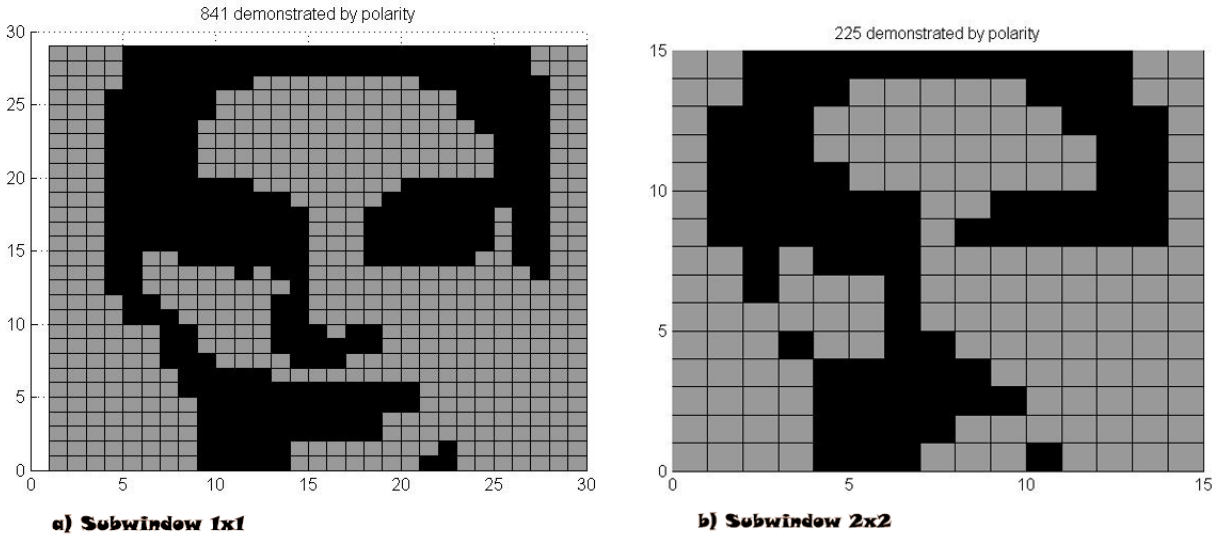


Figure 26: Polarity of features. a) Subwindow 1x1 b) Subwindow 2x2

4. Merging:

   Merging concept is our proposed method. Our random features are white and black boxes. The merging algorithm is explained below:

   (a) Choose two random features by some our proposed method.

   - BruteForce : Training process needed some days. We seeked other methods which can help to avoid bruteforce.
   - Covariance
   - XOR logic operation
   - OR logic operation
   - AND logic operation

   (b) Assume to the two features merged and create a new merged feature.

   (c) Calculate performance of a new merged feature through all negative and positive datasets.

(d) Check the merging condition which depends on chosen threshold filtering. When a new merged feature provides the merging condition, we can accept it as a new feature and name it as 'super' feature.

(e) Merging options. We have two merging options which are 'adding' and 'non-adding'. Merging option is demonstrated on figure 27.

- Adding : A new super feature is added in our exist feature pool.
- Non-Adding : A new super feature is added in our exist feature pool and the two features which are merged are deleted.

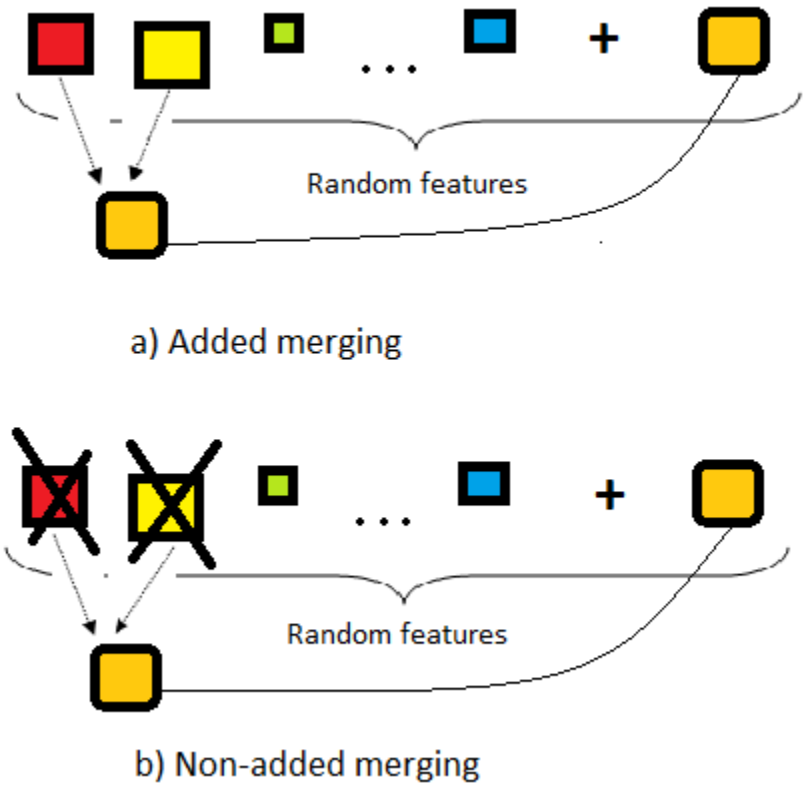
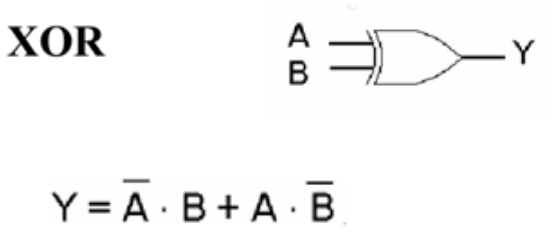


Figure 27: A merging option. a) Adding merge b) Non-adding merge

5.2 Implementation Challenges

5.2.1 XOR Logic Operation

XOR gate is a digital logic function which have two or more inputs and one output. The output of XOR gate is true only if one of its input is true. If both inputs are true then the output is false, or both inputs are false then output of XOR is again false. The truth table and general sysmbol representation for an XOR gate with two inputs appears at the below on figure 28.



A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

Figure 28: XOR operation.

We use XOR logic operation as sorting filter which helps to avoid from bruteforce. The chosen features are as inputs of XOR operation. Our idea is in here, to understand XOR relationship of two different placed features.

5.2.2 AND Logic Operation

AND gate is a digital logic function which has two or more inputs and one output. The output of AND gate is true only if all inputs are true and the output is false if one or more inputs are false. The truth table and general sysmbol representation for an AND gate with two inputs appears at the below on figure 29.

AND

$Y = A \cdot B$

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

Figure 29: AND operation.

5.2.3 OR Logic Operation

OR gate is a digital logic gate which has two or more inputs and one output. The output of OR gate is true only if one or more inputs are true. The output is false only if all inputs are false. The truth table and general sysmbol representation for an OR gate with two inputs appears at the below on figure 30.

OR

$Y = A + B$

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

Figure 30: OR operation

5.2.4 Covariance

Covariance is defination of the relation of two random variables. Lets say there are two random variable named X and Y and their expected values are as below:

$$\mu_x = E(X)$$

$$\mu_y = E(Y)$$

Then we can define the covariance of these two random variable is as below:

$$cov(X, Y) = E[(X - \mu_x)(Y - \mu_y)]$$

If these two random variable is independent then their covariance is zero.

$$cov(X, Y) = 0$$

These general formulation is an review of covariance. In this thesis, covariance approach has been tested for facial feature detection.

5.3 Database

Database means a collection of organized data. Databases used as many different approaches which can be classification, training a system, searching, tracking, booking etc. Thus, databases are an indispensable tool for modern life. People in modern life get in touch with databases in their daily life for different purposes. For example, if we go to the bank to ask how much money left we have, if we try to book a hotel or airline ticket online, if we go to a library and try to search catalog, or if we pay something on the internet. Within all these activities we are connected and we use databases somehow. Databases are always used to store a particular type of datas.

Database is as like human memory. We store all information into brain as we can. When we need a specific information, we track our memory or brain searching through stored information. Database and brain memory are one of the most important parts of a system and human life. In machine learning, database is as like human brain memory. All machine learning systems are trained by a database.

This chapter presents a small survey of databases generally used for facial detection and recognition. We used several different databases in this thesis which are stated below.

5.3.1 The Importance of a Database

In face detection database is as important as algorithms.Accuracy of a system is also dependent directly or indirectly to chosen databases. In computer vision, databases are mostly used for extracting common feature from a chosen database.

5.3.2 Our chosen Databases

In face detection system, there is needed sizable databases of face images for training the system and reliably test face detection algorithms. There are many databases that are presented on the internet for noncommercial purposes and mostly they are free of charge.The choice of an appropriate database should be based on the task given (aging, expressions, lighting etc). Also another criteria to choose database could be property that wanted to be tested. (e.g. how algorithm behaves when given images with lighting changes or images with different facial expressions). On the other hand, if an algorithm needs to be trained with more images, database should be chosen based on this perspective. In the following section we have presented a brief review of some available databases which are widly known and used. Some of them are listed below:

1. FEI
2. CBCL
3. A database which is used by Rowley [5]

5.3.2.1 FEI Database

The FEI face database had been created by Brazilian scientists between June 2005 and March 2006 at the Artificial Intelligence Laboratory of FEI in So Bernardo do Campo, So Paulo, Brazil [3]. This dataset had been collected from 200 individuals. Individuals have different hair stylies and appearances and aged between 19 and 40. We named this datasets as ‘database 1’.

- Training set: 100 faces, 130 non-faces
- Test set: 100 faces, 130 non-faces

Random image of FEI database shown on figure 31.

Figure 31: Before/after normalization image of FIE database[3].

5.3.2.2 CBCL Database

The MIT Center for Biological and Computation Learning distributes training and test database. We named this datasets as 'database 2'.

- Training set: 2,429 faces, 4,548 non-faces
- Test set: 472 faces, 23,573 non-faces

The database is available for download on the internet. Random image of the database shown on figure 32.

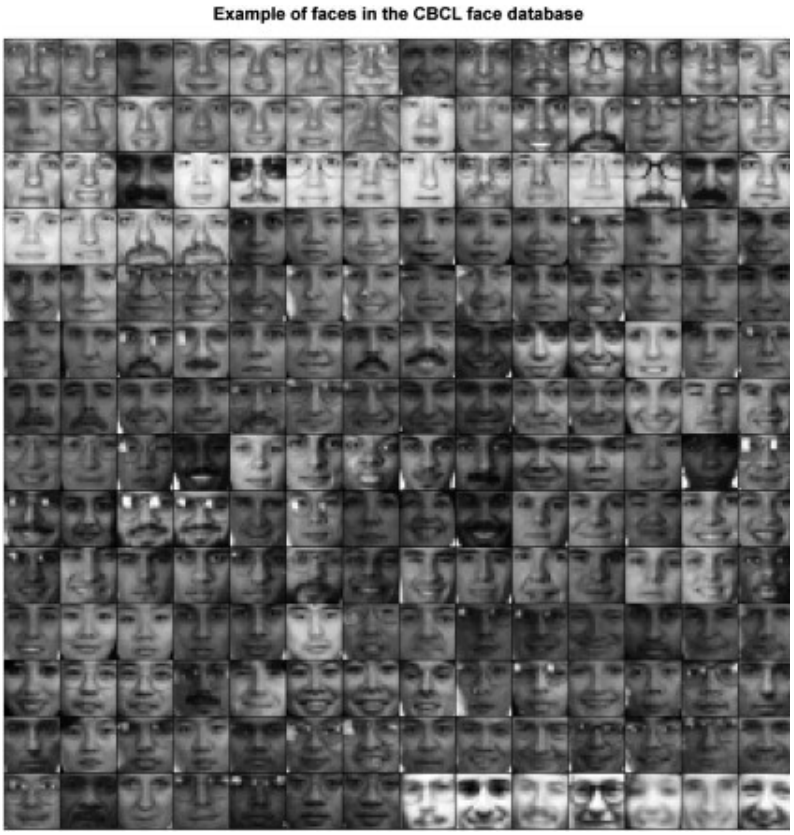


Figure 32: CBCL database[4].

5.3.2.3 Rowley Database

The testing MIT+CMU Rowley database [5], including 508 faces, as figure [5], total 130 test images. We have used just test datasets.

Figure 33: A database which is used by Rowley [5].

CHAPTER VI

RESULTS

In this chapter, we will present outcomes of our experimental results. We have suggested and tested several approaches. We gave the results as given in the overview of the thesis. In each step we commented on the results and we judged them over their performance rate.

6.1 Bruteforce

Firstly, we can discuss about bruteforce method. We were able test bruteforce method only with few number of features. Here we used bruteforce method for calculating all possible dual merging of existed features and choosing the best features. We aim to avoid from bruteforce and explore new algorithms which bring us the features which present best merging.

6.2 Covariance

One of these approaches is 'covariance'. We tried to merge features based on covarience value of features on positive samples. We expected this procedure to automatically explore distinct features such as eye balls, however the results were not as expected. Because of that we have experimented on many approaches. In further steps we will look the results of each approach closely.

6.3 Super Feature

At next we tried on 'super' feature approach. The most important characteristic of the 'Super' feature is distinguishing the face and non-face images clearly. This means TP(true positive) = 100, TN(true negative)=100, FP(false positive)=0, FN(false

negative)=0. In other words, this feature is an ideal feature. First, we worked on small databases which are not mentioned at the database section of the thesis. The database is consisted of 26 clear faces and approaximately 50 non-face images. The size of images are [30x30]. We explored the images by applying features which are 5x5 and 6x6 sized subwindows. We reached up to %91 face detection rate over the chosen database. But we couldn't get good results when we run our algorithm over database2. The database2 is very general and challenging database. Thus we prooceded to improve our algorithm.

6.4 Best N Feature

After that we went over our 'BestNFeat' named algorithm. The results of this algorithm are presented in figures 34 and 35. The pseudocode of this algorithm is as follows:

- Calculate the performances of random features on the dataset.
- Sort the performance rates in increasing order.
- Detecting the first N merged features which provides the merging condition.

In this approach the merging condition is defined as follows.

- Take two features named A and B which are wanted to be merged.
- Say merged of these two features is C.
- If we look at the performance rates of these three (A,B,C) features on positive database.
- If we get the higher performance from C then we say that the merging condition is guaranteeded.

This procedure is repeated for the best 200 features as shown in figure 34. If we look at the results, it is enough to take 83 featues to get 0 false rate. However we wanted to get as less as possible false negative rate over the negative database so using between the first 90 and 100 merged features was suitable for this goal. The results on this test set were good enough. But we could not obtain satisfactory results when we ran the algorithm over our more tough database2.The main reason for that is the mis-alignment of many face images in database2. The test results from database2 is shown in figure 35.

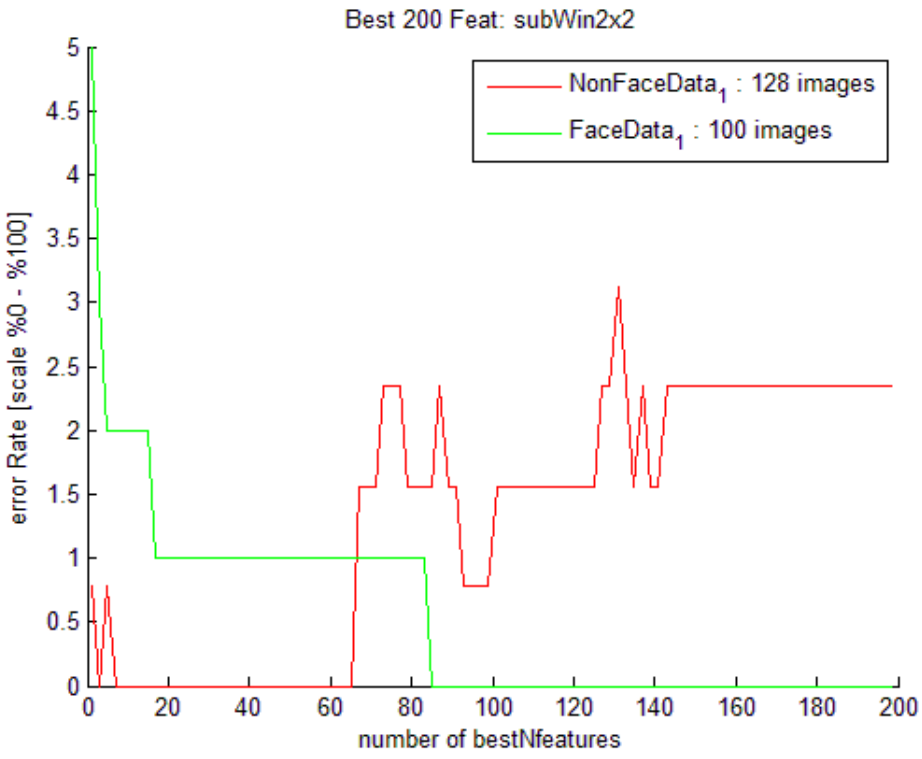


Figure 34: A result of BestNFeat algorithm with dataset-1.

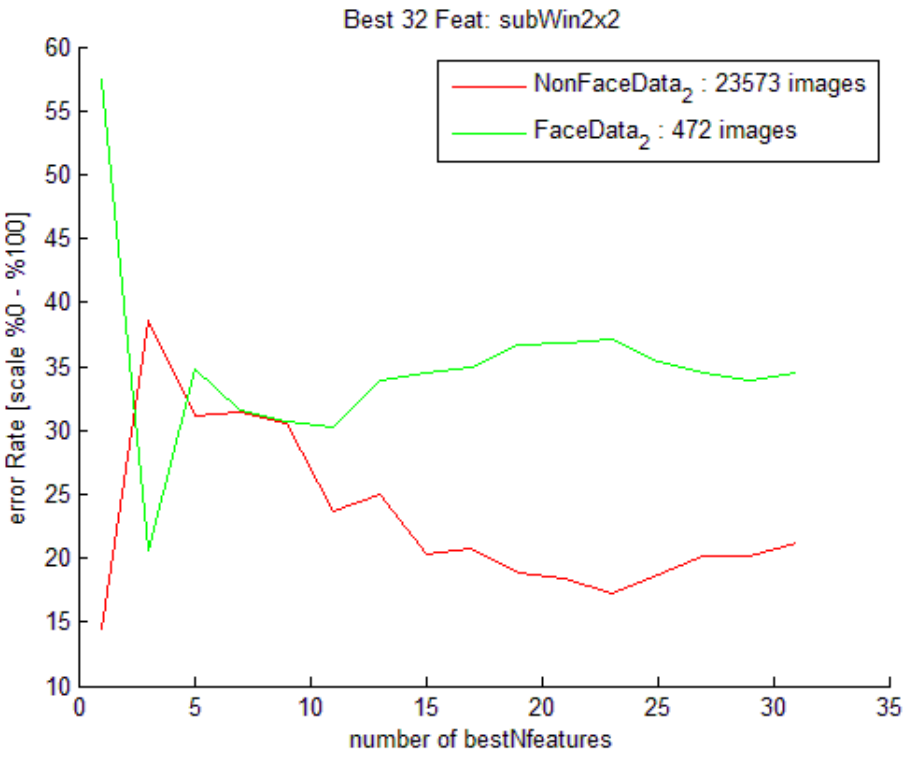


Figure 35: A result of BestNFeat algorithm with dataset-2.

6.5 BestNFeat + Majority

BestNFeat algorithm presents face detection when a feature in an image has to provide certain thresholds which are calculated in certain conditions. Threshold determining algorithm is challenging because images are so sensitivie to illumination. So we improved our BestNFeat approach by adding majority voting. Its test results over database1 is shown figure 36. Here detection error decreases with increasing merged features as expected. Hovewer we still cannot get effective results over database2.

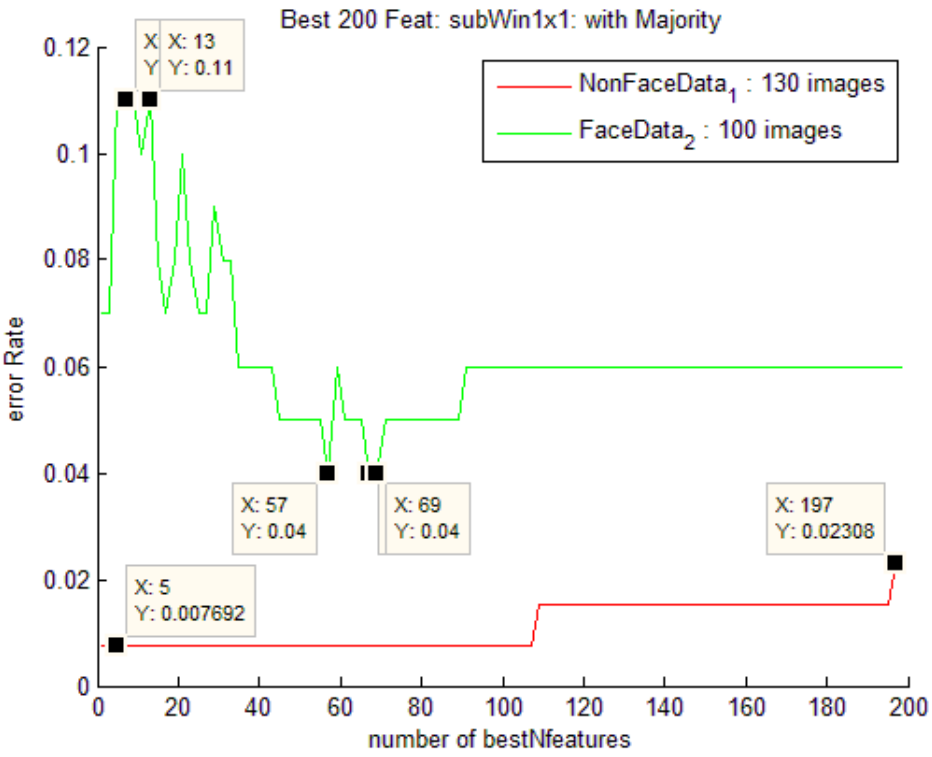


Figure 36: Result extended BestNFeat algorithm with Majority approach.

6.6 AND-OR-XOR

Next approach is AND. We wanted to test the results of merging features which are similar to each other. One of the test result is presented in figure 37. The features in figure 37, has been trained with train-database-1 and improved with validation-database-1 finally tested with test-database-1.

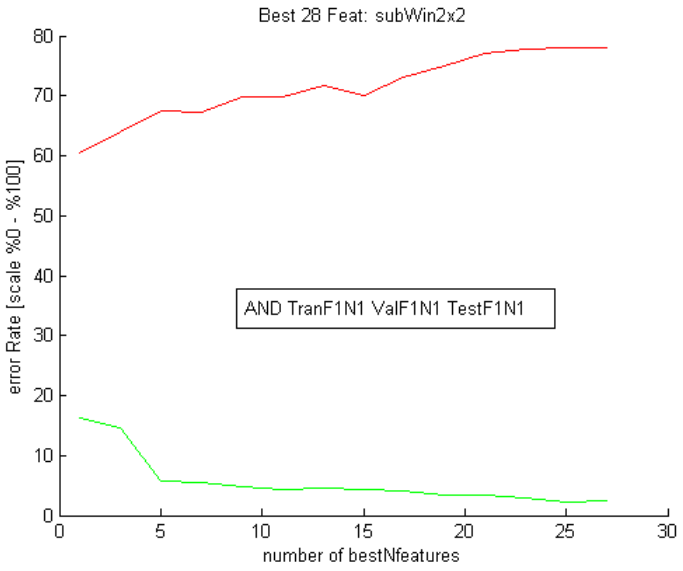


Figure 37: Results of AND approach with BestNFeat figures

We tried XOR approach similar with AND. The results are shown in figure 38

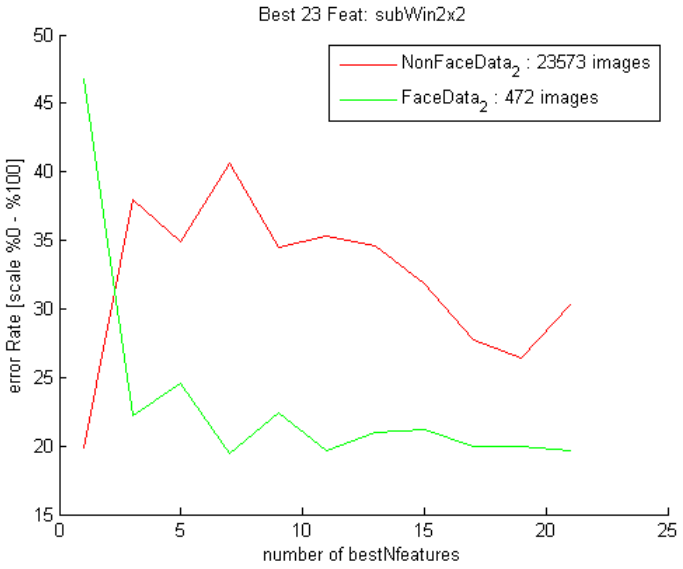


Figure 38: Results of XOR approach with BestNFeat figures

6.7 Conclusion

Experimental results in the training part showed that both XOR,AND,OR and covariance merging strategies. As the iterations grow, number of merged boxes are increased. Hence, it is possible to overfit training data and this yields decrease in generalization abilities as the iterations grow.

CHAPTER VII

DISCUSSION AND CONCLUSIONS

7.1 Discussion and Conclusions

In this study, an automatic facial feature extraction algorithm is proposed. The algorithm first create features randomly on given database. (Sizes of images in our database are the same.) After preprogressing, the images are converted to gray-scale and ready for the training progress of features. Training progress detects the pattern of face features iteratively. The iteration process continues until we reach the highest performance with the least number of 'super' features. In order to detect faces with different sizes, size of merged 'super' features are scaled with scaling coefficients and slided on the image. Usually we have taken the scaling coefficients between 1.2 and 1.5. In this case, we assume that the extracted features are core features. The features seek a face on the image starting with the smallest sized feature(core feature) to the higher sized scales. This search allows the algorithm to detect faces in multiple scales. The best size of face window is 18x24 pixels, which is reported to be the smallest resolution at which human beings can perform recognition. Our algorithm performs the search using an 19x19 pixels face window, which is close to the ideal size. In order to measure our algorithm performance, we have chosen 2 different face datasets. One of them is properly aligned and other one is roughly aligned.

The algorithm is tested on facial image databases of FEI, CBCL and the datasets which are used in Rowley[5] paper. As shown in figures of chapter 6, the results of our approaches could not reach the performance of Viola Jones algorithm. All approaches that we suggested, with the exception of the brute force approach, detect facial features automatically, decreasing the training time. Even when we get good

results over properly aligned datasets , our results on the more challenging dataset were not good enough.

7.2 Future Work

It is hard to choose a proper dataset in general for object detection. Our automatic feature extraction ideas can be improved with new extensions. However, this requires the availability and use of non-commercial, high quality and properly aligned datasets.

We believe we can get better resultswith these approaches when they are used in combination. This is a future direction we plan to take.

Automatic extraction of facial features is one the most challenging topics in object detection. All approaches are tested by face images. Our algorithm may perform better on detection of other, solid objects.

Bibliography

[1] W. Online, "Histogram equalization — Wikipedia, the free encyclopedia," 2006. http://en.wikipedia.org/wiki/Histogram_equalization.

[2] N. Efford, *Digital Image Processing: a practical introduction using JavaTM*. Pearson Education Limited, 2000.

[3] D. C. E. Thomaz-Online, "Fei face database," last update 2012. http://fei.edu.br/~cet/facedatabase.html.

[4] C. for Biological, C. L. at MIT, and MIT-Online, "Cbcl face database #1," 2000. http://cbcl.mit.edu/projects/cbcl/software-datasets/FaceData1Readme.html.

[5] H. A. Rowley, S. Baluja, and T. Kanade, "Neural network-based face detection," *IEEE Transactions On Pattern Analysis and Machine intelligence*, vol. 20, pp. 23–38, 1998.

[6] P. Viola and M. Jones, "Robust real-time object detection," in *International Journal of Computer Vision*, 2001.

[7] T. Brodsky, "Face recognition in poor-quality video," *American Psychological Society*, 1999.

[8] V. H. M. Sonka and R. Boyle, *Image Processing Analysis and Machine Vision*. Brooks/Cole, second ed., 1999.

[9] R. Brunelli and T. Poggio, "Face recognition: features versus templates," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, no. 10, pp. 1042–1052, 1993.

[10] K. K. Sung and T. Poggio, "Example based learning for view-based human face detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, pp. 39–51, 1995.

[11] A. L. Yuille, P. W. Hallinan, and D. S. Cohen, "Feature extraction from faces using deformable templates.," *International Journal of Computer Vision*, 1992.

[12] J. Y. A. Wang and E. H. Adelson, "Spatio-temporal segmentation of video data," 1994.

[13] M. J. Black and Y. Yacoob, "Recognizing facial expressions in image sequences using local parameterized models of image motion," *International Journal of Computer Vision*, vol. 25, pp. 23–48, 1997.

[14] G. Donato, M. S. Bartlett, J. C. Hager, P. Ekman, and T. J. Sejnowski, "Classifying facial actions," *IEEE Transactions on Pattern Analysis an Machine Intelligence*, vol. 21, no. 10, pp. 974–989, 1999.

[15] P. E. H. R. O. Duda and D. G. Stork, *Pattern Classification*. John Wiley and Sons, Inc, second ed., 2001.

[16] . B. Moghaddam MIT Media Lab, "Photobook eigenfaces demo," 2006. `http://vismod.media.mit.edu/vismod/demos/facerec/basic.html`.

[17] A. S. Pandya and R. R. Szabo, *Introduction to Face Recognition,Neural Networks For Face Recognition*, ch. 7, pp. 287–314. CRC Press, 1999.

[18] E. Hjelmas and B. K. Low, "Face detection: A survey," *Computer Vision and Image Understanding*, vol. 83, pp. 236–274, 2001.

[19] C. P. Papageorgiou, M. Oren, and T. Poggio, "A general framework for object detection," in *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)*, pp. 555–562, Jan. 1998.

[20] W. Online, "Summed area table — Wikipedia, the free encyclopedia," 2008. `http://en.wikipedia.org/wiki/Summed_area_table`.

[21] I. 1994-2013 The MathWorks, "Matlab documentation center." `http://www.mathworks.com/help/matlab/`.

[22] E. Gregori, "Viola jones simplified," *Robot Magazine*, May 2011.

VITA

Uranchimeg Olzvoi was born on 28 Oct 1982, in Ulaanbaatar/Mongolia. She received her B.S. degree from Electrical and Electronic Engineering Department of Istanbul University, Istanbul, Turkey in 2010.