İSTANBUL TECHNICAL UNIVERSITY ★ INSTITUTE OF SCIENCE AND TECHNOLOGY

SCALE AND POSE INVARIANT REAL-TIME FACE DETECTION AND TRACKING

Master Thesis by

Mehmet Şerif BAYHAN, B.Sc.

Department : **Computer Engineering**
Programme: **Computer Engineering**

JUNE 2008

İSTANBUL TECHNICAL UNIVERSITY ★ INSTITUTE OF SCIENCE AND TECHNOLOGY

SCALE AND POSE INVARIANT REAL-TIME FACE DETECTION AND TRACKING

Master Thesis By
Mehmet Şerif BAYHAN, B.Sc.
(504051520)

Date of submission : 5 May 2008
Date of defence examination : 11 June 2008

Supervisor (Chairman) : **Prof.Dr. Muhittin GÖKMEN**
Members of the Examining Committee : **Assoc. Prof.Dr. Işın ERER (İ.T.Ü.)**
Asst. Prof. Dr. Mustafa KAMAŞAK (İ.T.Ü.)

JUNE 2008

<u>İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ</u>

ÖLÇEKTEN VE POZDAN BAĞIMSIZ GERÇEK ZAMANLI YÜZ BULMA VE IZLEME

YÜKSEK LİSANS TEZİ

Müh. Mehmet Şerif BAYHAN
(504051520)

Tezin Enstitüye Verildiği Tarih : 5 Mayıs 2008
Tezin Savunulduğu Tarih : 11 Haziran 2008

Tez Danışmanı : **Prof.Dr. Muhittin GÖKMEN**

Diğer Jüri Üyeleri **Doç.Dr. Işın ERER**

Yrd.Doç.Dr. Mustafa KAMAŞAK

HAZİRAN 2008

PREFACE

I am thankful to my advisor Prof. Dr. Muhittin GÖKMEN for his continuous assistance on preparing this thesis. He always encouraged me while I was stuck and shared his precious knowledge and experiences during my researches and allowed times to negotiate on this study and to finalize it.

I appreciate my colleageus and Nortel Networks Netaş for the work flexibility while studying for my master thesis.

I also would like to thank to my family and friends for their material-immaterial supports during my B.Sc. and M.Sc. studies especially my Grandfather Muhittin BAYHAN who had shared his pension with me during my B.Sc. study.

June, 2008 **Mehmet Şerif BAYHAN**

CONTENTS

ABBREVIATIONS

Adaboost	: Adaptive Boosting
CAMSHIFT	: Continuously Adaptive Mean Shift
CVL	: Computer Vision Laboratory of Ljubljana University
FFTS	: Face Finder and Tracker System
FLD	: Fisher's Linear Discriminant
GTAV	: Audio Visual Technologies Group
GUI	: Graphical User Interface
HSV	: Hue-Saturation-Value Color Space
IMM	: Informatics and Mathematical Modeling Department of Denmark University
JPEG	: Joint Photographic Experts Group
OpenCV	: Open Computer Vision Library
PCA	: Principal Component Analysis
PGM	: Portable Gray Map File Format
RIP	: Rotation In Plane
ROP	: Rotation Off Plane
SNoW	: Sparse Network of Winnows
SVM	: Support Vector Machine
YCbCr	: Luminosity - Chrominance of the Blue - Chrominance of the Red Color Space

LIST OF TABLES

Page No

LIST OF FIGURES

LIST OF SYMBOLS

w	**:** width of image
h	**:** height of image
X	**:** positive sample set
$\mathbf{x_i}$	**:** i^{th} positive sample
y	**:** face/non-face indicator
m	**:** number of total positive samples
T	**:** max stage of cascade training
$\mathbf{h_j}$	**:** weak facial classifier
ε	**:** classification error
α_t	**:** an interval value based on classification error
D	**:** probability distribution
$\mathbf{Z_t}$	**:** normalization factor
H(x)	**:** linear combination of the selected weak classifiers
ii	**:** integral image
I	**:** image
x	**:** x coordinate of a 2-dimensional point on an image intensity space
y	**:** y coordinate of a 2-dimensional point on an image intensity space

SCALE AND POSE INVARIANT REAL-TIME FACE DETECTION AND TRACKING

SUMMARY

Face based applications such as lip reading systems, face recognition systems, video surveillance systems have been more popular in Computer Vision community during the past several years. Such systems perform image processing on the human face and produce their throughputs according to their purpose. It can be seen easily that the main step of those applications is detecting the human face in the given video frame or image; hence, face detection is very important for such applications. There have been numerous methods proposed by researches for face detection in the past decades. Despite this, face detection is not a fully solved problem yet. This is because, face is a non-rigid object and its appearance is varied depending on pose of the face, illumination conditions of the environment, scale of the face, facial expressions, aging effects, occlusions, skin color, etc…

The main purpose of this study is developing a face detection and tracking system for multi-view face detections and multiple face tracking. Therefore, published face detection and tracking methods are analyzed. There are mainly four approaches (knowledge based, feature based, template matching based, appearance based) and the most significant one is the appearance based methods. Also, object tracking methods were investigated and skin color based algorithm was determined for face tracking purpose.

In this study, one of the most popular and recent appearance based face detection method is used. It is a combination of Adaboost algorithm, Integral Image and cascading classifiers. Faces are trained for five different poses (left, left+45°, front, right+45° and right). Also, CAMSHIFT algorithm is used for face tracking because of its speed and easy implementation for face. To avoid impact of image analysis's computations on Real-time application, parallel processing methods were used. Two processes (main and child) were created for this purpose. Child process detects faces periodically on the given frame while the main one process all frames and displays the results of child process to the user screen.

In conclusion, our face detection and tracking system (FFTS) has been implemented on OpenCV and Microsoft Visual C++.NET 2005 platforms and it has demonstrated significantly high detection/tracking rates based on the tests on three different image databases and one video database. The system can detect and track more than one face very fast with high accuracy and it can be used as the main step for the aforementioned applications.

ÖLÇEKTEN VE POZDAN BAĞIMSIZ GERÇEK ZAMANLI YÜZ BULMA VE IZLEME

ÖZET

Geçen bir kaç yıl içinde, insan yüzüne dayalı dudak okuma, yüz tanıma ve vidyo ile gözetlemeye dayalı sistemler Bilgisayarla Görü topluluğunda rağbet görmeye başladı. Bu tip sistemler insan yüzü üzerinde görüntü işleme yaparak yapılma amaçlarına göre çıktılar sağlamaktadır. Bu tür sistemlerin dayandığı temel unsurun insan yüzünün bulunması olduğu kolaylıkla görülebilir. Bu yüzden yüzün yerinin bulunması bu uygulamaların olmazsa olmazıdır. Geçmiş senelerden günümüze, görüntü içindeki yüz yerinin tespiti için araştırmacılar tarafından bir çok yöntem önerilmiştir. Buna rağmen insan yüzü yerinin tespiti şu ana kadar tamamen çözülememiştir. Çünkü insan yüzü değişken bir yapıya sahip ve görüntü içindeki görünümü verilen poza, ortamdaki ışığın miktarına, yüzün resimdeki büyüklüğüne, yüzdeki ifadelere, yaşın getiridiği değişimlere, yüzü kapatan nesnelere, yüzün ten rengine, vb. durumlara göre değişebilir.

Bu çalışmanın amacı; çoklu poza dayalı ve verilen görüntüde birden çok insan yüzünü bulup bu yüzleri izlemeye yönelik bir sistem geliştirmektir. Bü yüzden yüz bulma ve izlemeyle ilgili yöntemler incelendi. Ana dört kategori (bilgi tabanlı, özellik tabanlı, şanlon eşleştirme tabanlı, görüntü tabanlı) içinden en kayda değer olan görüntü tabanlı yöntem seçildi. Ayrıca, yüzün izlenmesini sağlayan yöntemler incelendi ve yüzün ten rengine dayalı bir yöntem seçildi.

Bu çalışmada görüntü tabanlı en gözde ve en yeni yöntemlerden biri seçildi. Bu yöntem Adaboost algoritmasına, "Integral Görüntü" tekniğine ve kaskat sınıflandırıcılara dayalıdır. Beş değişik poza (sol, sol+45°, ön yüz, sağ+45° ve sağ) ait insan yüzü bu yöntemle eğitildi. Ayrıca kolay uygulanabilirliğinden ve gerçek zamanlı uygulamalardaki hızından dolayı, yüzün izlenmesi için CAMSHIFT algoritması kullanıldı. Görüntü işlemenin gerçek zamanlı uygulamalara kötü yöndeki etkisinden kaçınmak için paralel programlama gerçeklendi. Bunu sağlamak için iki iplikçik (ana ve çocuk) oluşturuldu. Çocuk iplikçik alınan görüntü çerçeveleri üzerinde yüzleri bulmaya çalışırken, ana iplikçik de gelen tüm görüntüleri çocuk iplikçiden aldığı veriye göre işler ve bunu kullanıcı penceresine basar.

Sonuç olarak, insan yüzlerini bulma ve izleme sistemi (FFTS) OpenCV ve Microsoft Visual C++.NET 2005 platformların da yazıldı ve üç farklı test kümesi ile bir video kümesindeki test sonuçlarına göre çok yüksek başarım oranı sağladığı görüldü. Bu sistem, verilen görüntede hızlı ve doğru bi şekilde birden çok yüzü bulmayı ve izlemeyi sağlamakta vebu yüzden yukarda bahsi geçen yüze dayalı uygulamaların ilk basamağı olarak kullanılabilir.

1. INTRODUCTION

In this research, a combination of face detection and tracking approach is presented. Face detection has been an important and challenging research topic for the last decades in the Computer Vision community. It is a challenging problem because faces are non-rigid and have a high degree of variability in size, shape, color, and texture. There are many interesting applications have been developed based on face detection such as face recognition, automatic lip-reading, image database management, multimedia retrieval, human-computer interaction, etc.

Face detection is mainly defined as identifying and locating human faces in an image or video frame regardless of their;

- Pose (in/off plane rotations)
- Illumination conditions
- Scale
- Facial Expression/Accessories
- Aging effects
- Occlusion
- Skin color/tone
- Position
- Presence/absence of structural components

Face detection process causes high computational time hence high speeded and robust methods are required for video applications and for still images. In the past several years, lots of methods have been developed with different goals and for different areas to improve face detection. In this research one of the most popular, fast and robust detection technique that was proposed by Viola and Jones [5] is used. This technique is based on Adaboost (Adaptive Boosting) algorithm [46] and Haar-like features. In this technique, simple Haar-Like features are extracted from faces; face/non-face classification is done by using a cascade of successively more complex classifiers which are trained by Adaboost learning algorithm.

1.1. The proposed approach of the research

In this research, a face detection and tracking system has been implemented based on a cascade of fast, weak classifiers according to the Viola and Jones' approach. This approach is used to determine upright faces for five different poses (left, left+45°, front, right, right+45°) in an image and in video frames. The different five poses are used to cover detecting all the upright faces between left and right profiles. After detection is completed, the detected faces are tracked by CAMSHIFT (Continuously Adaptive Mean Shift) algorithm for real-time multi object tracking [6].

1.2. Outline of proposed approach

This research is organized as follows; in the next section an overview of face detection difficulties, its importance, its application areas and history of the face detection techniques on still images are given. Description of the proposed face detection system is given in Section 3. In section 4, the experimental result of the research is reported in detail. Finally, in Section 5 the main conclusions of the research are outlined, as well as directions for future research.

2. FACE DETECTION OVERVIEW

2.1. Introduction

In this section the importance of the face detection, its application areas and difficulties of the face detection is mentioned. Also, the existing face detection methods are reviewed.

Face detection is the first step of face analysis so it is important and it has many application areas. The main goal the face detector is already explaining the complexity of face detection while explaining its dependencies. Hence, many methods have been designed to detect faces under these circumstances.

2.2. Importance of Face Detection and its Application Areas

Face detection has fundamental importance on model-based video coding, image-databases, content-based image retrievals, human recognition systems, video surveillance systems, image quality of cameras, automatic target recognition or generic object detection/recognition systems [1, 2]. Face detection in real-time processing is an important and first step of many intelligent human-computer interaction applications such as face recognition and video surveillance systems in different areas such as airports, hotels, municipalities, casinos, country border checking and police stations.

Some of the face detection applications and their usage areas can be listed as below;

- Video Surveillance Systems
    - Automatic Security Systems: Monitoring airports, streets, homes in case of any dangerous situations
- Face Recognition applications
    - Entertainments: Video games, virtual realities, human-robot and human-computer interactions
    - Information Security: TV parent controls, personal device logon, desktop logon, application security, database security, file encryption,

intranet security, internet access, medical records, secure trading terminals. All these provides easy of life and security. With all these features, there have been many application developed for communication, defense, health, finance areas. Traditional identity card/documents are incapable of determining personal identification and can not provide sufficient security for banking transactions and etc... It also increases quality of life: To make life more comfortable and throw away unnecessary procedures (Filling identification forms many times, entering passwords, to make read electronic cards etc..) which are time consuming and decrease productivity.

- Smart Cards: Driver's licenses, entitlement programs, immigration, national ID, passports, voter registration. All these provide personal identification without any passwords. There is no need to remember any password or carry any keys for daily used systems. Your face is your password and it is not easy to change it.
- Law enforcement and surveillance: advance video surveillance, CCTV control, portal control, post-even analysis, shoplifting, suspect tracking and investigation

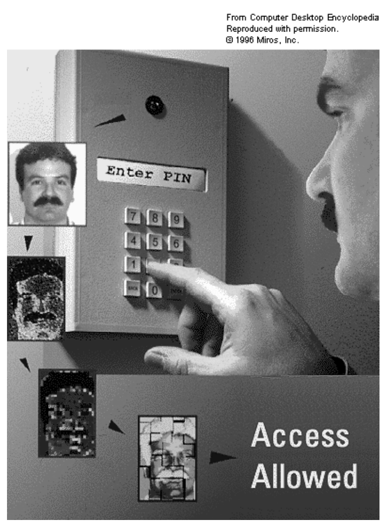

Figure 1: Face Recognition on access control

- Images/video databases management
  - Video indexing and video browsing of databases
- Facial Expression recognition
  - Automatic determination of drunk drivers

- Image quality of cameras
    - Adjusting image capture settings (auto focus, auto exposure, color balance etc.)

2.3. Face Detection Difficulties

The main difficulties of the face detection are listed in its definition as mentioned before. These are the pose of the face (in/off plane rotations), illumination conditions of the image, scale of face, facial expressions/accessories, aging effects on face, partial/full occlusion, different skin colors, position of face in the image, extremely high dimensional space of face image etc…

2.3.1. Pose of the Face

A face can be in different poses (profile, frontal, upside down etc…) in the image and this make the detection harder. The pose of the face is depending on two types of rotations; RIP (rotation in plane) and ROP (rotation off plane). RIP defined as rotating the face (or head) between left and right sides while ROP is rotating the face from between down and up sides.

Figure 2: RIP of a Face

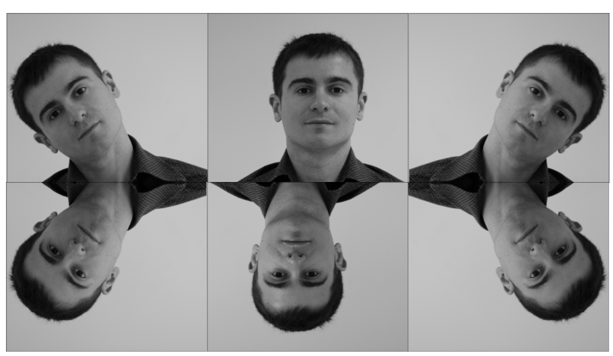

Figure 3: ROP of a Face

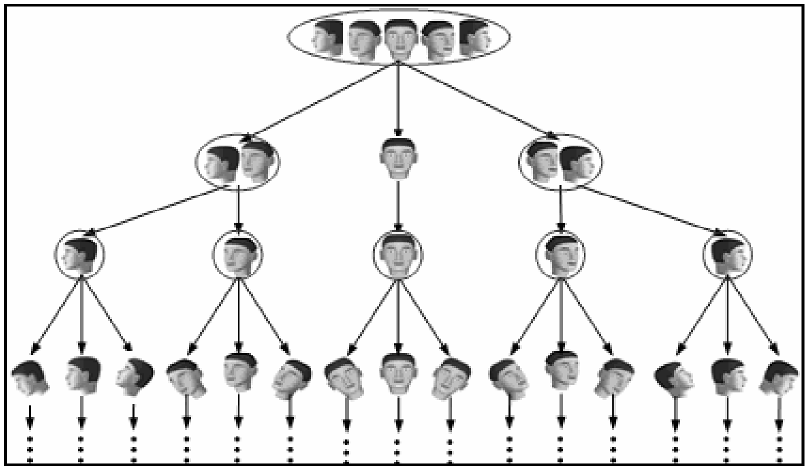

Figure 4: Combination of RIP and ROP of Face [3]

All these rotation types effect on the features and intensity values of the face which make the detection harder and add new situations. Because of these reasons many researches have been done mostly on frontal images.

2.3.2. Illumination Conditions of the Image

Images taken in different lighting conditions, camera characteristics and resolution add to the variation in face pictures. Because of this reason, the performance of face detection is dramatically affected by changes in the illumination conditions of image capture or frame. In general the variation between images of different faces captured in the same conditions is smaller than that of the same face taken in a variety of environments.

The following figure shows that a variation in illumination conditions can seriously alter the appearance of a face in the image plane to the extent where the images on

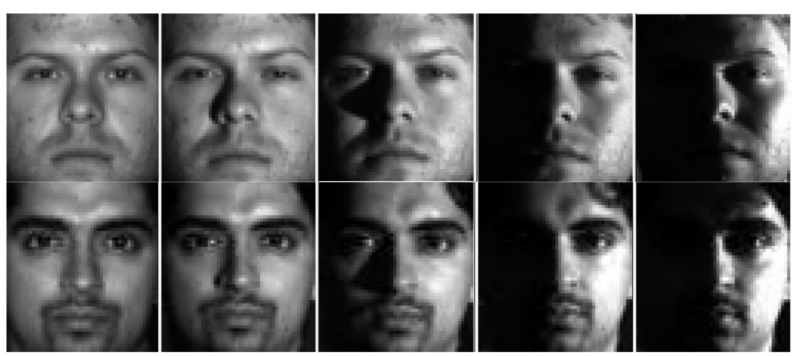

Figure 5: Faces with varying illumination taken from the Yale B database

the far right appear more similar to one another than to their respective frontally illuminated faces [4].

2.3.3. Scale of face

Size of the face in the image can vary and it has different effects to different detection approaches. Face features are searched in the given image. Search is started from a default scale and the scale is increased until the image size is reached. This affects the detection time. Also, the distances between geometric structures of the face features (nose, mouth etc…) are changed and this can cause loosing some face features.

Figure 6: Different scale of faces for the same image size

2.3.4. Facial expressions/accessories

Facial expression such as smiling, frowning, winking change the intensity values, features and geometric relationship of face features of the face. Also, facial accessories such as glasses, make-up has the same effects and so make the detection of the faces harder.

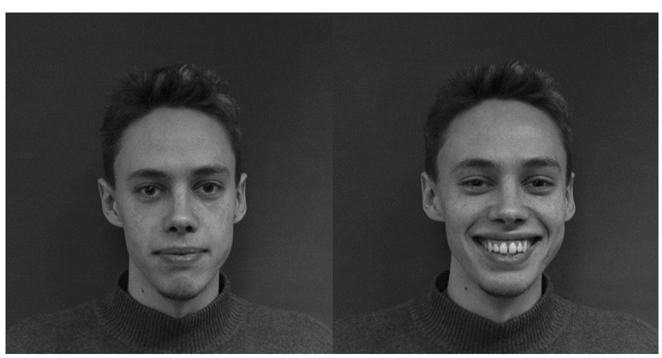

Figure 7: Different facial expression of the same face

2.3.5. Aging effects

Morphology of the faces is changed year by year and so younger faces are smoother than older faces. Older humans have more wrinkled faces and illumination changes increases on them also. All of these changes affect the faces features and the intensity value of images.

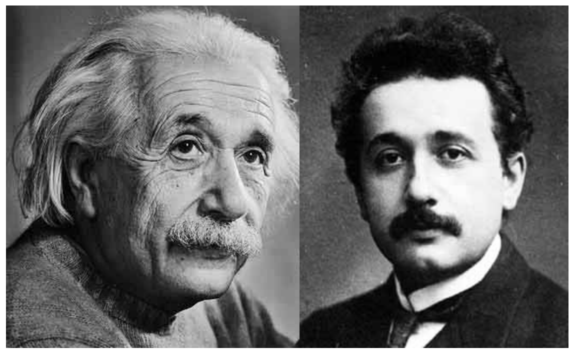

Figure 8: Older and younger Albert Einstein

2.3.6. Occlusion

Faces can be occluded by some objects such as hat, scarf, other objects in front of the face etc… In this situation the whole face is not seen so most of the face detection algorithms fail to find all the features (mouth, nose, eyebrows etc…) of the face. This makes the detection of faces harder.

Figure 9: An occluded face from AR Face Database

2.3.7. Skin color/tone

Peoples can be from different origins and so their color skins are varied. Most of the current researches have been done based on white skinned humans. Skin color has significant effects on the face image so it is hard to implement face detection on all skins.

Figure 10: Top model Naomi Campbell and Tom Cruise

2.3.8. Position of the face

Faces can be in any position of the given image. Face detector must have search all of the image space to find the position of the image and this can take much time according to the selected search method.

Figure 11: Many faces in different positions

2.3.9. Presence or absence of structural components

Facial features such as beard, mustache, and glasses may or may not be present and there is a great deal of variability among these components including shape, color, and size [1]. The intensity values are changed directly so this affects the face features.

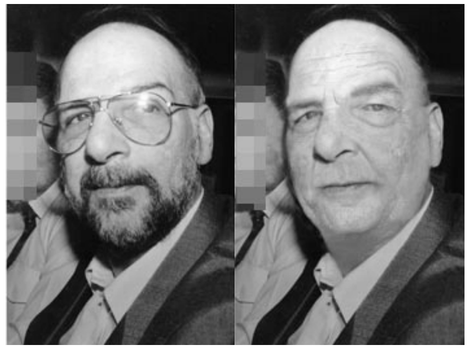

Figure 12: Effects of glasses/beard/mustache on a face

2.4. Review of Existing Face Detection Methods

2.4.1. Introduction

The face detection problem is pretty old and there are numerous methods have been proposed to detect faces in a single image of intensity or color images. Actually, the first research on automatic machine recognition/detection of faces started in the 1970's after the seminal work of Kanade [9] and Kelly [10]. Over the past thirty eight years extensive research has been conducted by neuroscientists, psychophysicists and engineers on various aspects of face recognition by humans and machines.

Several surveys have been written for face recognition and detection by different researchers. One of the surveys of face recognition methods was written by Samal and Iyengar before 1991 [7]. Chellapa et al. wrote a more recent survey on face recognition and some detection methods in 1995 [8]. Another survey is prepared by Ming-HsuanYang for face detection methods in 2002 [1].

The existing techniques of face detection have been categorized under four different methods in those surveys. However some of the detection techniques can overlap category boundaries. The detection techniques are categorized as follows;

- Knowledge-based methods
- Feature invariant approaches
- Template matching methods
- Appearance-based methods

Among the above methods, appearance based methods are presenting excellent results so it has been attracted by many researches. Performance of appearance based methods are mostly depended on the training set so the training set should be as much as wide to contains different faces in varied styles (pose, illumination, structural components, skin color, age, facial expression, occlusion, etc.). Face detector has to be able to recognize that the given image does not include any faces. Unfortunately, non-faces object space is very wide and complex so it is hard to do it. This is because of the high dimensional space of face and the complexity of the non-face objects.

The past and recent face detection techniques will be explained in detail in the methods described above.

2.4.2. Knowledge-Based Methods

Knowledge-Based methods are described in this section and several previously done researches with current research are mentioned.

2.4.2.1. Introduction

Researchers use the biometric structure of face and derive some rules according to this structure and provide develop knowledge-based methods. Human face is simple to extract rules according to the relationship of the face features. Features are related which each other according to their relative distance and their positions. Top-down approach is used in this method. Some of the basic rules are listed below;

- Faces have two eyes which are symmetric
- Eyebrows reside above the eyes
- There is a nose and a mouth
- Etc.

In this approach, facial features are extracted from the given image and pre-coded rules are run over the image to determine some areas which are similar to the face. These areas are called as face candidates and to make sure if it is really a face, verification process is applied. [1]

The rules of this approach should be well defined otherwise some faces may not be detected. The rules shouldn't be more general since it causes high false positives rates. This approach present high accuracy for uncluttered frontal poses but it is not efficient for other poses.

2.4.2.2. Overview of Knowledge-Based Researches

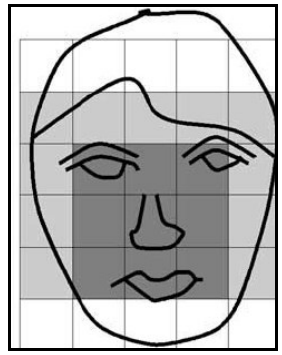

Figure 13: Simulation of face for knowledge-based method [1]

In 1994, an exemplary study was done by Yang and Huang and a three level hierarchical type of rules was used [12]. All possible face candidates are found in the results of first level of rule. The rule is applied to all locations of the given image by shifting a sub-window. The second and third level rules are used to find the features of the face. For this purpose, sub-sampling and averaging is applied to the given image and this sequence is continued until four images with different resolution are created as below;

Figure 14: Sub-sampling and averaging applied on an image [1]

The first rule is applied to the image with lowest resolution one. Face candidates are determined according to the following rules in this level;

- The center part of the face (the dark shaded parts in Fig.13) has four cells with a basically uniform intensity
- The upper round part of a face (the light shaded parts in Fig. 13) has a basically uniform intensity

➢ The difference between the average gray values of the center part and the upper round part is significant

The found face candidates are then processes in images which has finer resolution. Histogram equalization and edge detection is applied to the face candidates in level 2. Then, the survived candidates are moved to level 3 and face features related rules are applied to them. Focus of attention and coarse-to-fine strategies are used to decrease the computation time. This strategy is used in later face detection methods [11]. The system's performance is 50 correctly detected faces over 60 faces but with 28 false positives.

In 1997, Kotropoulos and Pitas presented a rule-based localization method which is similar to [12]. In this method horizontal and vertical projections of the image is obtained and local minimums are used to determine face features. Horizontal projection determines the left and right side of the head while vertical projection is used to determine the mouth, lips, nose and eyes. The detection rate was 86.5 % on the European ACTS M2VTS database [13].

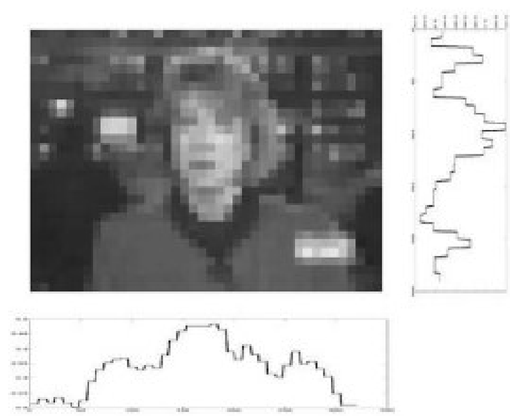

Figure 15: Horizontal and vertical projection graphs of he image [1]

In 2000, Chiunhsiun Lin and Kuo-Chin Fan presented a rule based face detection system [15]. The proposed system consists of two main parts. The first part is to search the potential face regions that are gotten from the triangles based on the rules of "the combination of two eyes and one mouth". The second part of the proposed system is to perform the face verification task. Experimental results demonstrate that an approximately 98% success rate is achieved and the relative false detection rate is very low.

In 2006, Yi-Ting Huang and Cheng-Chin Chiang presented a rule based real-time face detector [14]. The process of the proposed algorithm starts from the skin-color pixels extraction through rules built upon a quadratic polynomial model using the normalized color coordinate. The algorithm then verifies the extracted eyes and lips using a set of rules derived from the common spatial and geometrical relationships of facial features.

In 2008, Kevin Ho et al. presented a rule based new face model for detecting human faces in still images [16]. It was developed by observing the relative geometric positions of facial features and the variations in the pixel values. These observations were converted into a set of rules which were mathematically expressed. The proposed method is simple, flexible, accurate and efficient.

2.4.3. Feature-Based Methods

Feature based methods are described in this section and several previously done researches with current research are mentioned.

2.4.3.1. Introduction

In this method, it is intended to find invariant face features (edge, intensity, texture, shape, color, etc.) for face detection. Bottom-up approach is used. Humans can detect faces and objects in different poses and lighting conditions easily. Hence it is assumed that there must exist properties or features which are invariant over this variability.

Facial features (eyebrows, eyes, nose, mouth and hair-line) are firstly extracted from the image mostly by using edge detectors and then the face itself obtained according to these features. Extracted features are combined and a statistical model is built to describe their relationships and to verify the existence of a face.

There are several issues that are observed during face detection for this method. In some cases, face features can be corrupted due to noise, occlusion or illumination conditions. Also, shadows have negative impact on grouping the facial features for the detection since they cause numerous strong edges.

2.4.3.2. Overview of Feature-Based Researches

In 1993, Sirohey proposed a research which considers segmentation and identification of human faces from grey scale images with clutter [17]. The segmentation developed utilizes the elliptical structure of the human head. It uses the information present in the edge map of the image and through some preprocessing separates the head from the background clutter. The identification procedure finds feature points in the segmented face through a Gabor wavelet decomposition and performs graph matching. This algorithm achieves 80 percent accuracy on a database of 48 images with cluttered backgrounds.

In the same year, Chetverikov and Lerch proposed a method of detecting faces in images based on detection of blobs and streaks [18]. Eyes are dark features so dark blobs, cheekbones and a nose are light features so light blobs, and the outlines of eyebrows, a mouth, and a face are linear edges which are known as streaks. The mentioned features are extracted from the given image and faces are detected among the spatial relationship of these features. Blobs are extracted from a low-resolution of the given image easily. After that, these features are detected in finer resolution of the same image with high precision. This speeds up the detection process.

In 1995, Graf et al. developed a method to locate faces and facial features in gray scale images [19]. Face features which has high intensities are obtained after band pass filtering and some morphological operations. Peaks are seen in the histogram of the image after those operations and these peaks are used to generate binarized images. Face candidates are obtained according to binarized images and with some classifiers for verification of faces.

In 1995, a technique is proposed by Leung et al. to find the location of faces according to local features and random graph matching [20]. Totally five facial features (nostrils, eyes, and nose/lip junction) are used here. The algorithm works by coupling the local feature detectors with a statistical model of the mutual distances between facial features it is invariant with respect to translation, rotation (in the plane), and scale and can handle partial occlusions of the face.

In 1996, identification a face-like texture method is proposed by Dai and Nakano [23]. This method was developed by Augusteijn and Skufca in 1993 [24]. However, Dai and Nakano combined this method with color information and enhanced it. The modified method has ability of detecting non-upright faces regardless of presence/absence of structural face components (beards, mustaches, glass).

In 1997, Yow and Cipolla presented a feature-based method that uses a large amount of evidence from the visual image and their contextual evidence [21]. Gaussian filter is applied to the given image and local maximums are assumed as facial feature candidates to be used. Then, the edges around these candidates are examined to be grouped. A feature vector is used to store measurements of a region's characteristics. Facial features are trained and the mean and covariance matrix of each facial feature vector are computed. Mahalanobis distance of the image regions' are calculated with regarding to the corresponding features and they are determined as a valid feature if the distance is smaller than a threshold. Validated features are then grouped according to the knowledge of where they should normally occur. This method has ability to detect faces with different poses and orientations.

In 1998, a morphology-based technique is proposed by Han et al. to extract eye-analogue segments for face detection [22]. They used eye and eyebrow for facial features detection with assuming that eyes and eyebrows are most salient and stable features of human face. The feature candidates are found with some morphological operations as closing, clipped difference and thresholding. The face candidates are then verified with a neural network.

In 2006, Lai and Li proposed a new method for detecting faces in color images with complex backgrounds [25]. In this method, image color space is transformed to the chrominance color space (YCbCr). Then, a Gaussian model is fitted on the transformed image and the likelihood of skin and image is calculated. After this process, to segment skin pixels and form a binary skin map, a threshold is applied to the likelihood image. Face candidates are contained in the binary skin map. In the last step, a verification process is run to determine whether the candidates are a actually a face or not.

In 2006, Zhang and Izquierdo proposed a two-step face detection technique [26]. In the first step a conventional skin detection method based on Gaussian mixture model in the YCbCr color space is used to extract possible face regions. In the next step, false positives are removed among the face candidate regions based on applying a learning approach using multiple additional features and a suitable metric in multi-feature space.

In 2008, Kevin Ho et al. presented a new feature based face detection system which is a combination of knowledge-based and feature-based approach [16]. It has been mentioned in section of knowledge-based methods.

2.4.4. Template Matching Methods

Template matching methods are described in this section and several previously done researches with current research are mentioned.

2.4.4.1. Introduction

A standard face pattern which can be parameterized by a function or can be defined manually is created in this method. It is simple to implement this approach. Correlation values are calculated for the face contour, mouth, eyes, and nose according to the standard face pattern for the given image. Based on the calculated correlation values, faces are detected in the input image.

This approach is not more effective under varied pose, illumination, scale and so it is not appropriate for face detection. To overcome on this circumstances some different templates are derived which are known as deformable, multi-scale, sub-templates, multi-resolution, etc.

2.4.4.2. Overview of Template Matching Methods

In 1969, Sakai et al. proposed a technique to detect frontal images from photographs [27]. In this technique some predefined templates are used for face features (eyes, nose, mouth, etc...). The greatest gradient changes are sought in the given input images and their correlations with the on hold templates are computed to detect face candidates. The first template comparisons provide the area to be focused on for face candidates and the other templates applied on these areas.

In 1987, Craw et al. proposed a technique for automatic extraction of face features based on a frontal face shape template [28]. Edges are extracted from the image with a Sobel filter and they are grouped and compared with the face templates to detect face contours. Then, face features are extracted for face detection.

In 1989 and following years, Govindaraju et al. presented hypotheses on face modeling and a face detection technique which consists of two stages [29]. The research covers the face detection on normal photographs and on newspapers photographs. Their face detector shows high performance only on frontal faces with no occlusion and rotations. To acquire edge map of the given image, Marr-Hildreth edge operator is applied. Hair-line curve and curves of the left and right side of the face are used as feature. Non-face regions of the edge map are filtered and related contours are linked. Corners on the contours are suggested as intersected feature curves and face detection is evaluated according to this feature groups.

In 1992, Yuille et al. proposed a method for detecting and describing the features of faces using deformable templates [33]. Features (e.g. eyes) are described by a parameterized template. An energy function is defined to link edges, peaks, and valleys in the given image to corresponding properties of the deformable template. The Energy function is minimized by changing some parameters to provide dynamic interaction of the template with the given image. By doing this, template is deformed itself to find the best fit and detect the faces.

In 1994, Tsukamoto et al. presented a new method for detecting a human face, and estimating its pose while tracking it in real image sequences [30]. The used face samples are divided to template blocks and parameterized qualitative features derived for detection process. The template blocks are traversed over the input image to compute the face probability and the region is labeled as face if the face probability exceeds a predefined threshold.

In 1995, Samal and Iyengar used silhouette templates for face localization instead of intensity images [31]. They presented a new approach based on principal component analysis to obtain a set of basis face silhouettes which are represented by an array of bits. Then, generalized Hough transform is used to detect faces with help of the silhouette templates.

In 1999, Miao et al. proposed a hierarchical template matching method for face detection [32]. This detector is able to detect slanted faces ($-25° \sim 25°$) and faces with much horizontal rotating angles ($-45° \sim 45°$) and vertical rotating angles ($-30° \sim 30°$). Different resolution of the given image is produced and Laplacian operator is applied to extract the facial features. Then heuristics are applied and faces are detected.

In 2007, Jin et al. proposed a face detection technique with using skin color and template matching together [34]. In the first step, skin pixels of the given color image are detected with the help of luminance-conditional distribution model of skin color. Secondly, skin-region rectangles are extracted via morphological operations and then linear transformation based template matching is used to detect face in the skin-region rectangles.

2.4.5. Appearance-Based Methods

Appearance based methods are described in this section and several previously done researches with current research are mentioned.

2.4.5.1. Introduction

Appearance-based methods are learned from a collected face and non-face images. The method is based on statistical analysis and machine learning algorithms that are used to find relevant characteristics of face and non-face images. These characteristics can be discriminant functions or distribution models. To make the performance of the method faster and robust, the dimensions of the samples are decreased.

Most of the appearance-based methods are based a probabilistic distribution model. In this approach; an image or a feature vector derived from an image is viewed as a random variable x, and this random variable is characterized for faces and non-faces by the class-conditional density functions p(x|face) and p(x|non-face) [1].

The other appearance-based methods are based on finding a discriminant function between face and non-face classes. The discriminant function can be a decision surface, a separating hyper-plane or a threshold function and it is usually based on distance metrics hence low dimensional image patterns are used for classification.

2.4.5.2. Overview of Appearance-Based Methods

There are many types of appearance based methods so the recent and most popular ones will be given in this section Some of these types can be listed as Eigenfaces, Distribution-based methods, Neural Networks, Support Vector Machines (SVM), Sparse Network of Winnows, Adaboost, etc.

- Eigenfaces

In 1989, Kohonen proposed a sample usage of eigenvectors with a simple neural network for face recognition [35]. The face is described by a simple neural network which is based on approximating the eigenvectors of the image's autocorrelation matrix.

A new recent technique has been proposed by Jie et al. for human face detection and recognition system on color images in 2008 [36]. Skin color and motion analysis are used to find the faces in the given input images. Face verification is done based on a SVM and by locating facial features via Generalized System Transform. The face pattern features are extracted with PCA (Principal Component Analysis) method. The system achieves high detection and recognition rates under reasonable illumination conditions.

- Distribution-based methods

In 1998, Sung and Poggio proposed a face detection method based on modeling distribution of human face pattern from face and non-faces image examples [37]. Difference of the given input image and distribution-based model is computed and faces are verified according to a multilayer perceptron classifier. In this method, faces and non-faces images are normalized and resized to 19X19 pixels. Then, this face patterns are grouped by six as follows;

In 2008, Wang et al. proposed a method for face detection based on combination of template-based histogram and Fisher's Linear Discriminant (FLD) which is needed to project samples from the high dimensional image space to a lower dimensional feature space [38]. Histograms are calculated with a new approach similar to the Viola and Jones' Integral Image [5] and it is named as "Integral Histogram Image". Then, Adaboost algorithm is used to select best features to describe face pattern.

- Neural Networks

There are many methods have been proposed for neural network based face detection. Face detection is assumed as a two class pattern recognition problem here.

In 1992, Agui et al. proposed a hierarchical neural network for face detection [39]. The inputs of the first stage are the intensity values of the image itself and a 3x3 filtered of the image. In the next stage facial features (geometric moments, std deviation of pixels etc.) are taken as input and the output of this stage determines if the region has a face or not.

In 1996 and following years, Rowley et al. built a neural network based face detector [39]. The faces are detected with the multilayer neural network and post-process method to eliminate multiple detections for the same face. The multilayer neural network is trained to classify a 20x20 image and it is applied every location of the given image. The given image is sub-sampled and detections are done on the new image again. The sub-sample process is performed repetitively to find the faces with different scales and positions. The detector is very slow because of sub-samplings of the given image but which it uses the most significant work in neural network approach according to the previous ones.

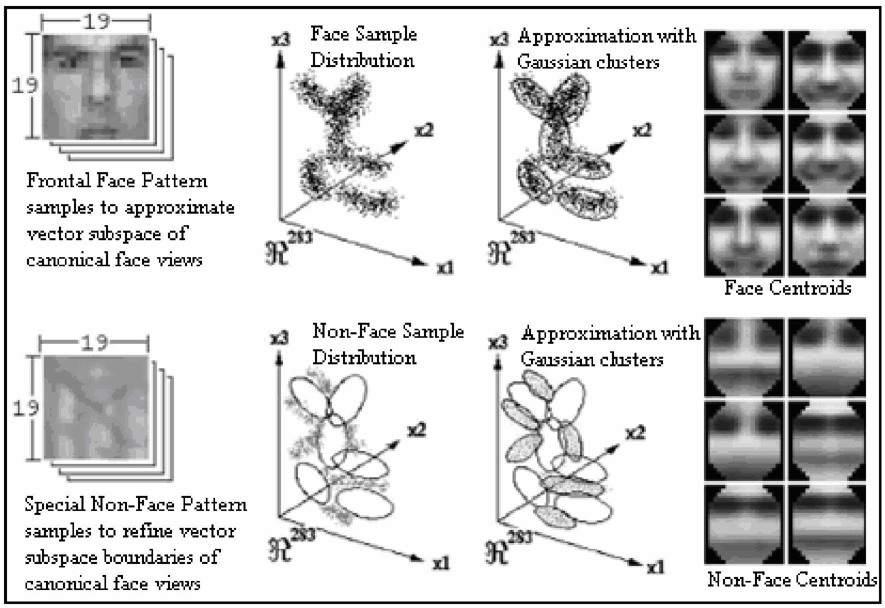

Figure 16: Face and non-face clusters used by Sung and Poggio [37]. Their method estimates density functions for face and non-face patterns using a set of Gaussians [1].

In 2008, Zuo et al. proposed fast face detector using an efficient architecture based on a hierarchical cascade of neural network ensembles. Some neural networks which are specialized on some regions of the face pattern space are ensembled. The accuracy and the fastness of face detection is improved with those ensembles.

- Support Vector Machines (SVM)

In 1997, Osuna et al. proposed the first SVM based face detection system [41]. SVM models are closely related to neural networks and they perform classification and optimally separate the given data into two categories. The Osuna's system uses 19x19 pixel images of two test sets for face detection. The system runs faster and has low error rates with respect to distribution-based methods.

In 2007, Chen et al. proposed a new method for face detection in which the false detections are re-trained in SVM [42]. In this method, a representative training set is resampled from a previously collected database which is used to train a face detector. Geodesic distances between pairs of face samples inside the face database are computed by isometric feature mapping. Then, the face set is converted to a low-dimensional manifold space to obtain low dimensional embedding. After this operation, the face set is created according to locally linear embedding (LLE). The same operation is done for non-faces, too. The face and non-face set is trained by Adaboost algorithm and it is examined on a large image database. The false alarms which occur during these trials are then re-trained in a one-class SVM to obtain a robust face detector.

- Sparse Network of Winnows

The SNoW (Sparse Network of Winnows) learning architecture is a sparse network of linear functions over a pre-defined or incrementally learned feature space. The main update rule utilized within SNoW is the Winnow algorithm developed by Littlestone [44]. This method is prepared for learning in the presence of a very large number of features.

In 2000, Yang et al. proposed the SNoW learning approach for human face detection [43]. It is aimed to detect faces under varied rotations, features, expressions and illumination conditions. Face patterns which are used for training are taken in different databases such as Olivetti, UMIST, Harvard, Yale, FERET, etc.

In 2005, Gundimada and Asari proposed an improved SNoW based face detection system [45]. In this method, significance of overlapping features is reduced for an efficient classification. With this enhancement, the technique showed better speed and accuracy according to other multi-class classification approaches for face detection.

- Adaboost

The Adaboost algorithm is relatively a new algorithm proposed by Freund and Schapire [46]. It constructs an ensemble of classifiers and uses a voting mechanism for the classification. Although the Adaboost is a quite new algorithm it has become very popular and many variants of the original algorithm have been proposed.

In 2001, Viola and Jones proposed a face detection system based on Adaboost algorithm on frontal faces [47]. The system is capable of processing images extremely rapidly and achieving high detection rates. Face detector is composed of three main part; a new image representation method called "integral image", Adaboost learning algorithm which select some critical visual features for training, and the third part is cascading the classifiers to speed up the overall system.

In 2003, Viola and Jones proposed another face detection system for profile views and rotated faces [48]. It extends the Viola' and Jones' previous face detector [47] for other poses of the humans face. Different detectors for different views of the face are built for multi-view detections.

In 2004, Huang et al. proposed a rotation invariant multi-view face detection method based on Real Adaboost algorithm [49]. Weak classifiers are produced according to Haar-like features of the train data and different views are trained for multi-view face detection. The whole 360° degree range was divided into 12 sub-ranges and their corresponding view based detectors are constructed separately. Pose estimation was done to select appropriate detector and to speed up detection time.

In 2006, Ichikawa et al. proposed a robust frontal face detection method based on Adaboost algorithm [50]. The detection system is enhanced for occluded faces by a novel method based on a decision tree. The face position is determined with the combination of a low-resolution of the face and face parts classifiers.

In 2008, Wu et al. presents a new approach to design node classifiers in the cascade detector [51]. In previous approaches the visual features are selected simultaneously. They have designed classifiers to addresses the difficulties caused by the asymmetric learning goal. Adaboost was used to train the features to reduce the training time in this research.

3. PROPOSED METHOD FOR FACE DETECTION AND TRACKING

As mentioned in the first, Viola and Jones introduced an impressive face detection system capable of detecting faces in real-time with both high detection rate and high speed [47]. Facial features are extracted from a set of training images and these features are trained by the Adaboost learning algorithm. A cascade of classifier technique is used for detection of the faces in a given input image or in a frame. This approach was used in our face detection system.

The proposed face detection system is developed for still images, for video frames and for frames taken from Web Cam's for different five poses. Different classifiers of the poses are run in parallel on the given image/frame for pose invariant face detection.

The detected faces are lately recorded in a buffer mechanism and they are detected according to the skin color histogram. CAMSHIFT algorithm is used for detected faces in tracking part of the system. CAMSHIFT algorithm is an enhanced version of Mean Shift Algorithm [52] and it was developed by Bradski in 1998.

3.1. Adaboost Algorithm

The Adaboost algorithm is relatively new algorithm proposed by Freund and Schapire [46]. The algorithm constructs an ensemble of classifiers and uses a voting mechanism for the classification. Its accuracy depends on the error of all classifiers in the ensemble which is often lower than the worst case. Although the Adaboost is a quite new algorithm it has become very popular and many variants of the original algorithm have been proposed. The Adaboost can be used together with other learning algorithms to improve their performance.

The idea of boosting occurred based on creating a single strong classifier from a set of weak classifiers. A weak learner is such a classifier which is only slightly correlated with the true classification.

Table 1: Pseudocode for Adaboost Algorithm

Given: $(x_1, y_1), \ldots, (x_m, y_m);\; x_i \in X, y_i \in \{-1, 1\}$

Initialize weights: $D_1(i) = 1/m$

For $t = 1, \ldots, T$:

1. **Find**

   $$h_t = \arg\min \varepsilon_j;\; \varepsilon_j = \sum_{i=1}^{m} D_t(i) I[y_i \neq h_j(x_i)] \quad ,(where\ h_j \in H)$$

2. **If** $\varepsilon_t \geq 1/2$ *then stop*

3. **Set** $\alpha_t = \frac{1}{2}\log\left(\frac{1+\varepsilon_t}{\varepsilon_t}\right)$

4. **Update**

   $$D_{t+1}(i) = \frac{D_t(i)\exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

   Output the final classifier:

   $$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

The initial weight of the classifiers is set by the first step as assigning $D_1(i)$. Then a simple loop is done for learning process. Weak classifiers are given to Adaboost and they are called in a series of rounds ($t = 1, \ldots, T$). In each rounds, a weak classifier h_t minimizing a weighted error on the training set (Step 1) is selected. The loop is terminated if this error exceeds 1/2 (Step 2). The value of α_t is computed next (Step 3) and the weights are updated according to the exponential rule (Step 4). In Step 4, Z_t is a normalization factor which assures D_{t+1} remains a distribution. The final decision rule is a linear combination of the selected weak classifiers weighted by their coefficients. The classifier decision is given by the sign of the linear combination.

As mentioned in [53], Adaboost minimizes an upper bound on the classification error $\varepsilon_{tr}(H)$ on the training set as follows;

$$\varepsilon_{tr}(H) \leq \prod_{t=1}^{T} Z_t = \frac{1}{2}\prod_{t+1}^{T}\sqrt{\varepsilon_t(1-\varepsilon_t)} \qquad \textbf{(3.1)}$$

Weak classifiers with the smallest weighted error on the training set are selected (Step 1) and their coefficients are set (Step 3) to minimize the upper bound of classification error.

Also, the re-weighting scheme assures that the updated distribution satisfies

$$\sum_{i=1}^{m} D_{t+1}(i)u_{t,i} = 0 \qquad \textbf{(3.2)}$$

Here, $\mathrm{u}_{\mathrm{t,i}} = \mathrm{h}_{\mathrm{t}}(\mathrm{x}_{\mathrm{i}})\mathrm{y}_{\mathrm{i}}$. After re-writing a weighted error in Step 1 of the Adaboost algorithm to the equivalent form

$$h_{t+1} = \arg\max \sum_{i=1}^{m} D_{t+1}(i)u_{q,i} \quad ,(where\ h_j \in H) \qquad \textbf{(3.3)}$$

It is evident that the selected h_{t+1} is “maximally independent” of the mistakes made by h_t enveloped in $\mathrm{u}_{\mathrm{t,i}}$.

Equation 3.2 has implications on the weighted error ε_t^{t+1} of h_{t} where the upper index indicates that the error is measured on weights for the step t + 1. Since the error can be expresses as;

$$\varepsilon_t^{t+1} = \frac{1}{2}\left(1 - \sum_{i=1}^{m} D_{t+1}(i)u_{t,i}\right) \qquad \textbf{(3.4)}$$

As a summary, it can be said that the Adaboost algorithm minimizes the upper bound on the classification error, selects weak classifiers with the smallest weighted error, and the weak classifier selected at round t is maximally independent of the weak classifier selected at round t - 1.

3.2. Haar-like Features and Integral Image

Facial features encode information about the face model so they are used for face detections. In our face detection system, reminiscent of Haar Basis is used for features so they are called as Haar-like features. These features encode the existence of oriented contrasts between regions in the training images. They are computed similar to the coefficients in Haar wavelet transforms. Actually, each Haar-like feature consists of two, three or four jointed "black" and "white" rectangles as below;

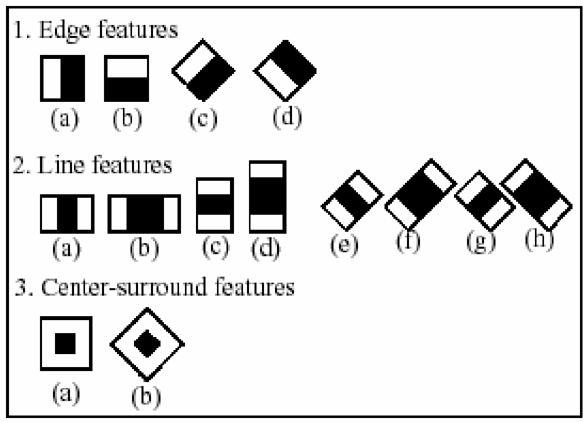

Figure 17: A set of extended Haar-like features

The value of a Haar-like feature is the difference between the sums of the pixels within black and white colored rectangular regions. Some of the possible features used in our face detection system are shown in Figure 17.

The Haar-like features are then used for face classifiers. The calculations of these features are done with a novel method presented by Viola and Jones [5]. The new representation is called as *Integral Image* and facial features computer very fast in this method. The value of the *Integral Image* at the coordinates (x; y) is the sum of all the pixels above and to the left of (x; y) as shown in Figure 18;

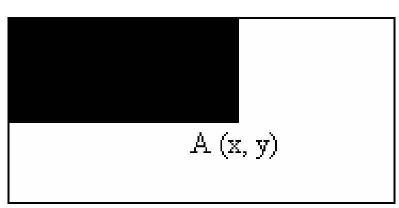

Figure 18: Representation of the Integral Image

Let *ii* be the integral image of the initial image "*i*" and *ii*(*x*, *y*) as the value of the integral image at the point (*x*, *y*). The equation of Integral Image will be;

$$ii(x, y) = \sum_{x' \le x, y' \le y} i(x', y') \quad \textbf{(3.5)}$$

Integral image of a rectangular can be computed with some points of neighbor rectangulars as below;

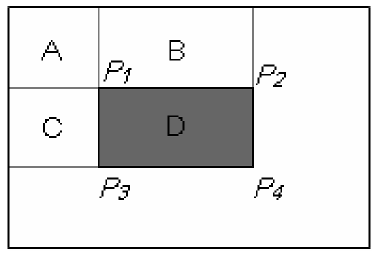

Figure 19: Integral Image of internal rectangular

$$\begin{aligned} &P_1 = A, \quad P_2 = A + B, \quad P_3 = A + C, \quad P_4 = A + B + C + D \\ &D = P_1 + P_4 - P_2 - P_3 = A + A + B + C + D - A - B - A - C = D \end{aligned} \quad \textbf{(3.6)}$$

3.3. Cascading Structure

The main structure of our face detection system is cascaded classifiers which make the detection faster and demonstrate high accuracy. The selected window is traversed on the given input image and cascaded classifiers are applied stage by stage. If the selected window of given image is not recognized as a face, it is rejected quickly and not passed to the next stage.

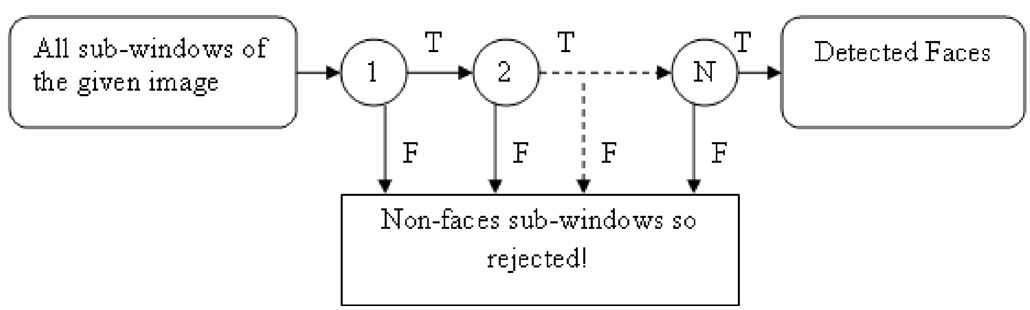

Figure 20: Cascaded classifiers with N stage

The first level cascades are mainly used to reject non-face regions of the given image to prevent high computation time for non-faces to go next stages for face verification. The upper stages requires high computation to verify that the given image window has a face. The sub-window is considered to be a face if it passes the all stages.

Every classifier is trained to achieve a detection rate d and a false negative f. The algorithm of cascade building is shown in Table 2.

Table 2: Algorithm for cascade building

Inputs: $f_{required}$, $d_{required}$ and f_{final} ; f is the allowed max false positive rate and d is the required detection rate $f_0 = 1$ and $d_0 = 1$, Repeat until $f_i > f_{final}$ Step 1: Train a classifier until $d_{current} > d_{requried}$ and $f_{current} > f_{requried}$ Step 2: $f_{i+1} = f_i * f_{current}$ Step 3: $d_{i+1} = d_i * d_{current}$ Step 4: Reject misclassified faces and use non-face data images to generate non-face data

3.4. Details of Proposed Face Detection System

3.4.1. Face Databases of Proposed Face Detection System

The face detection system is first started obtaining positive and negative images from popular image databases (The GTAV [54], CVL [55], The IMM [56] etc.) and internet sites. A detailed description of face sources/databases listed in Table 3. In this case, positive images are the face images itself. Negative images do not contain any face (backgrounds, office objects, forests etc..) images which are taken from arbitrary images.

Positive images are taken for five different upright poses which are right profile, right + 45°, frontal pose, left + 45° and left profile.

Right profile human faces are cropped from their chin to the forehead as shown in Figure 21. Different illuminated faces are cropped manually as described in Appendix B. Faces are selected from different ages, skin colors, facial expressions and accessorizes to simulate many types of faces in real world.

Table 3: Source of training images

#	Image Source	Database	Type of image
1	GTAV Face Database [54]	Total of 44 persons with 27 pictures per person. Pose views (0º, ±30º, ±45º, ±60º and v90º). Different illuminations, occlusion, facial expression and etc.	Positive
2	CVL Face Database [55]	Totally 114 persons with 7 different pose images for each person. (0º, ±45º, and ±90º)	Positive
3	The IMM Database [56]	Totally 240 still images of 40 different human faces.	
3	Web site: www.flickr.com and many other web sites	Lost of human faces with varied illumination, occlusion, skin color and pose (0º, ±30º,).	Positive
4	Web site: www.imageafter.com	Lots of non-faces images	Negative
5	Web site: www.bigfoto.com	Lots of non-faces images	Negative

Figure 21: Right profiles pictures from GTAV image database [54]

Right +45 profiles human faces are cropped from their chin to the forehead as shown in Figure 22. Different illuminated faces are cropped manually as described in

Appendix B. Faces are selected from different ages, skin colors, facial expressions and accessorizes to simulate many types of faces in real world.

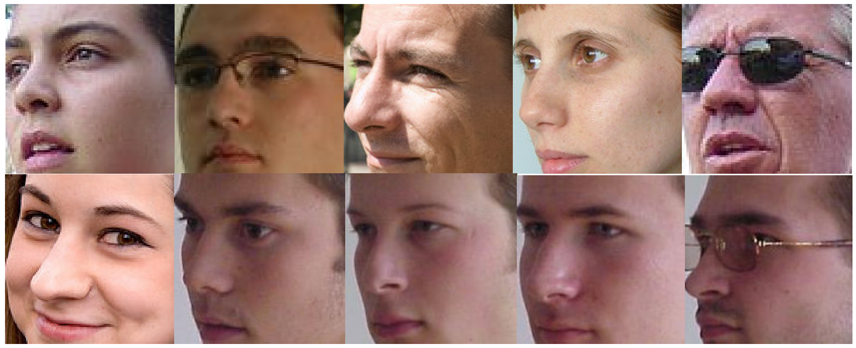

Figure 22: Right + 45° profiles pictures from image databases [54-56] and internet sites

Frontal human faces are cropped from bottom of their lips to the forehead as shown in Figure 23. The cropped images are mirrored to enhance the detection rates. The frontal images have varied illumination, partially occlusion, absence/presence of structural components (beard, mustaches, glass, etc.) and varied human face color.

Figure 23: Frontal poses taken from internet and different image databases [54-56]

For the left profiles and left + 45° profiles, the images which are obtained for right profiles and right + 45° are simply rotated horizontally but some time vice versa. A small subset of the right and right + 45° manually cropped images can be found in Figure 24 and Figure 25 respectively.

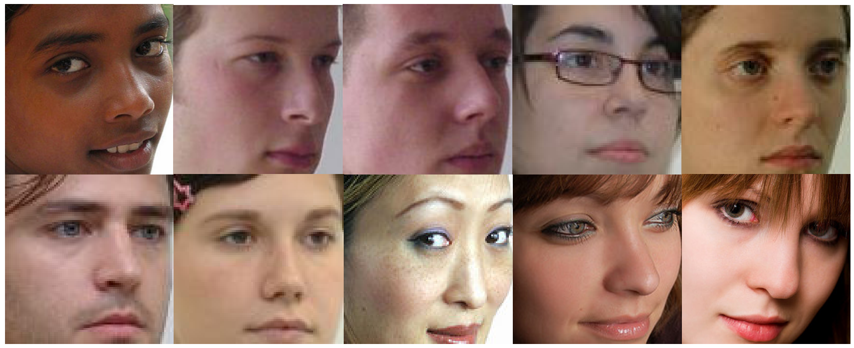

Figure 24: Left + 45 profiles taken from internet and different image databases [54-56]

Figure 25: Left profiles taken from internet and different image databases [54-56]

Numbers of faces for all poses are listed in Table 4 as below;

Table 4: Database of Our System

Pose type	**Number of total faces**
Left	1146
Left + 45°	1062
Front	3311
Right	1146
Right + 45°	1062

3.4.2. Training Phase of Cropped Faces

In the previous section, manually cropped faces are saved in different folders to be used in training according to their pose type. The faces are re-sized to 24x24 pixels and they are given to an OpenCV (Open Source Computer Vision Library, developed by Intel) based program to be normalized. Createsample.exe of OpenCV platform is used for this purpose. Details are described in Appendix C.

Createsample.exe normalizes the images to a given size and provides a vectorial representation of the images to be used in training. This phase is performed for all pose types and for negative images also.

Vectorial representations of the faces and non-faces were used in another program (Haartraining.exe) of the OpenCV platform to extract facial features which are described in Section 3.2. All the poses trained for 20 stages with maximum false alarm 0.5 and with a minimum hit rate of 0.995. Extracted facial features were given to Adaboost algorithm automatically and training phase of the poses were started to produce cascaded classifiers as mentioned in Section 3.3. After this phase finished, we were given a file with XML extension for all pose types. These XML files include cascaded weak classifiers for face detection.

The training phase lasts a bit long so we should be patient and wait for end of training. It can take longer or shorter according to the selected stages and number of faces in the training set.

Training phase of the poses lasted for approximately five days with the following computer skills;

- Intel Core2 Duo CPU: 2.00 GHz, T7300 micro processors
- 2 GB RAM
- Microsoft Windows XP Professional SP2 Operating System

3.4.3. Implemented Face Detection System

A user GUI (Graphical User Interface) was designed for our face detection system. Our face detection and tracking system is called as FFTS (Face Finder and Tracker System). FFTS is implemented in Microsoft Visual C++.NET 2005 platform and OpenCV platform is used as base for face processing utilities. Details of setups and implementation can be seen in Appendix C.

In this study, face detection system is designed for still images, for videos and for Web Cams. Detection of face on still images is a bit differing from other ones because of Real-time impact of image processing so it will be mentioned as a different section.

3.4.3.1. Face detection on still Images

A sub-window is traversed all over the image and the sub-windows are processed according to the XML files of each pose. The detection sub-window is set to 24x24 pixels as default according to the images used in training face. It can be manually changed in the configuration pane (Figure 27) of the FFTS to speed detection time. Actually, Face detection speed is affected by the size of used sub-window, scale factor, dimensions of input image and the accuracy of the training set.

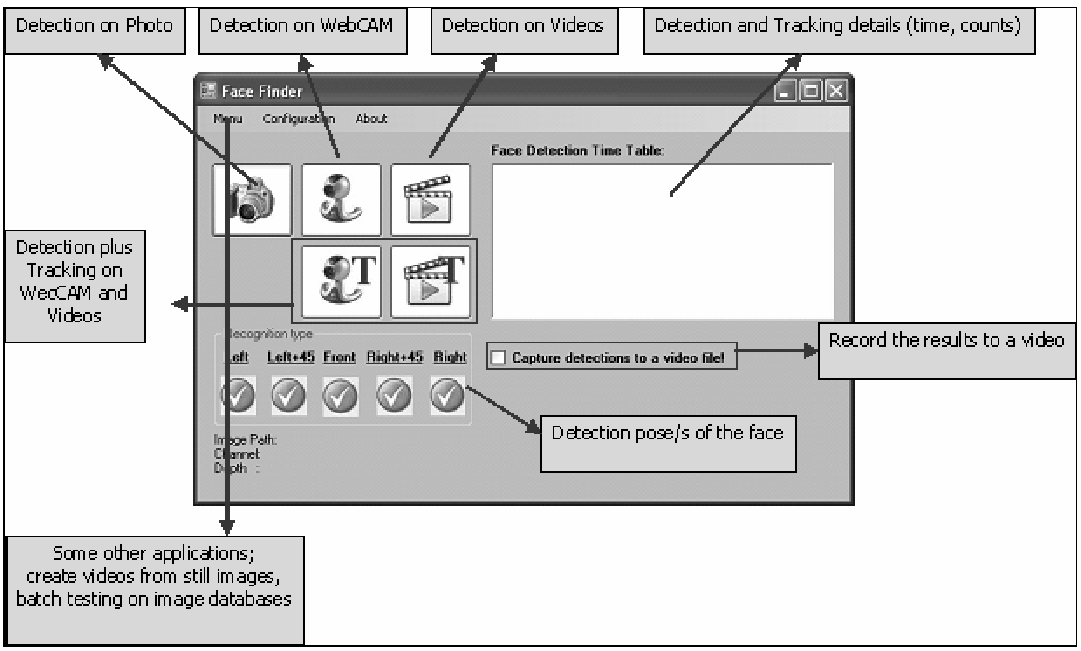

Figure 26: User Interface of FFTS

In FFTS, user selects an image from a directory and the image is read with OpenCV platform specific functions. Memory allocations are done for detections and OpenCV based object detection methods are called. Input image, scale of sub-window, minimum detected neighbor of face candidate, XML training file are given as parameter to the detection method.

Object detection method is called for each pose one by one and XML files of each pose are used for detection here. We are given a 2D point for each detected face and these points are used to draw rectangles or polygons with respect to its pose.

Frontal, Right + 45° and Left + 45° faces are drawn with colored rectangles as blue, turquoise blue and yellow respectively.

Right and Left profiles are drawn with colored polygons as red and green respectively. The polygons show the direction of the poses.

Finally, all the faces are displayed pose by pose and total detection time is displayed in time table of FFTS as milliseconds.

The configuration pane can be used to change the detection parameter of the OpenCV's object detection methods. In the pane refresh rate it not used for still images, it is for frames of video and Web Cam.

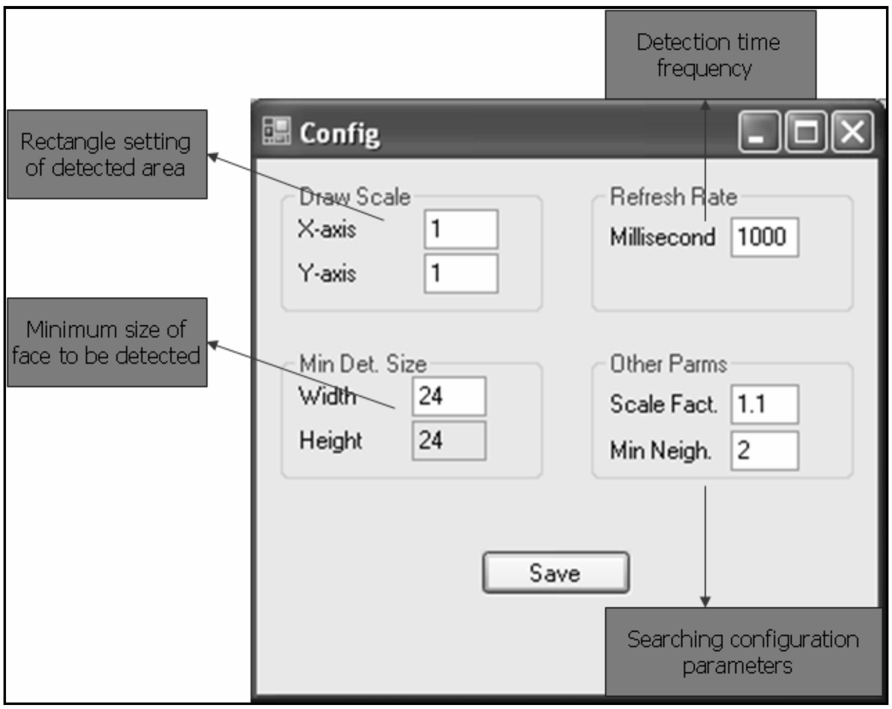

Figure 27: Configuration Pane of FFTS

All the poses are enabled to detect faces as default. They are seen as green in the FFTS user interface. Detected pose type can be selected or diselected according to the user wish so the detection can be stopped partially or fully. User simply click on the recognition type showed in Figure 26 and can enable or disable them. When a pose type is disabled it turns on as red to notice the user.

The overall face detection flow chart can be seen in Figure 28.

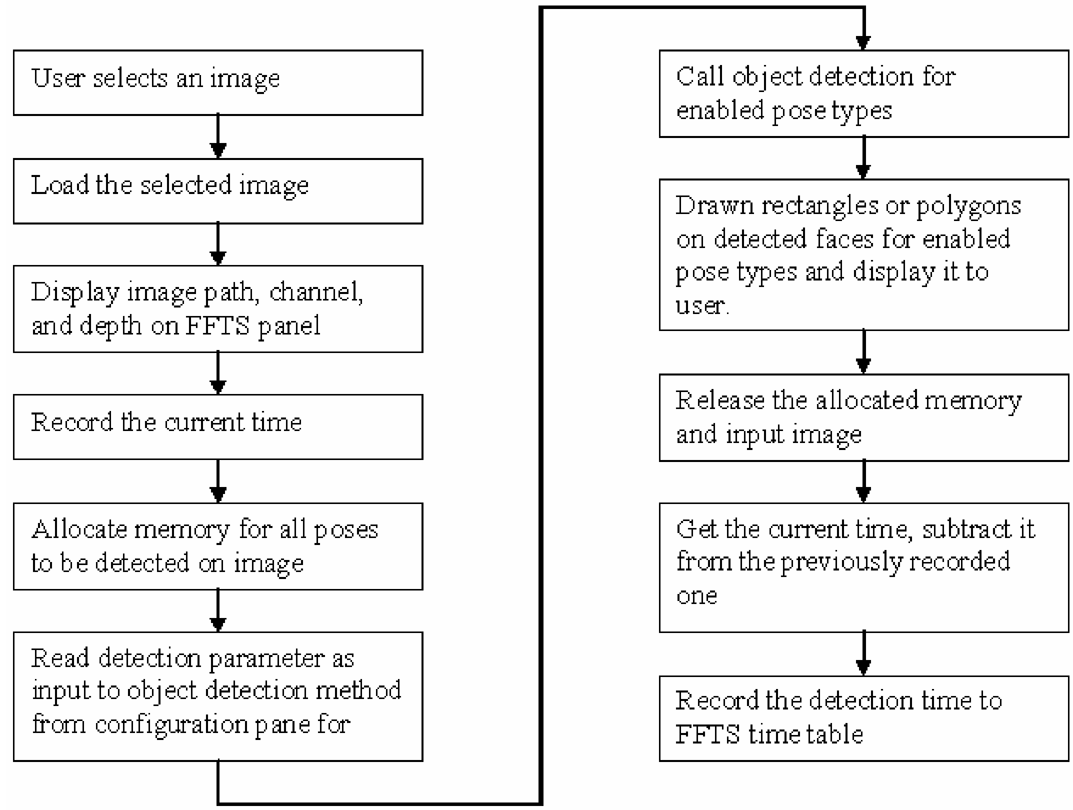

Figure 28: Flow Chart of Face Detection on Still Images

3.4.3.2. Face detection on Video and Web Cam Frames

The face detection method of still images is used here again but a different approach is used to prevent Real-time impact of image processing on the retrieved frames. Normally, every processed frame can be shown to the user with the detected faces but image processing can cause high computation time and this affect the user display screen. The system will loose many frames of Web Cams and user will see only the processed images with the detected faces. The worst case will be doing this on videos since frame loose will not be seen here and every frame of the video will be processed. Instead of a seeing a frame stream, user will see slow frame sequence. To avoid this inconvenience parallel processing is done to display the result of detected areas finer. There are two processes used here as main and child. In this approach, a child process is created and faces are detected in parallel according to a refresh rate frequency. The refresh rate frequency is selected one second as default but can be manually changed in the configuration pane of FFTS. Detection process is called every one second and current frame is read from a global variable and face detection results are recorded in another global variable. The current frame is obtained by the main process for use of child process. Detection time is also recorded to the time table of the FFTS.

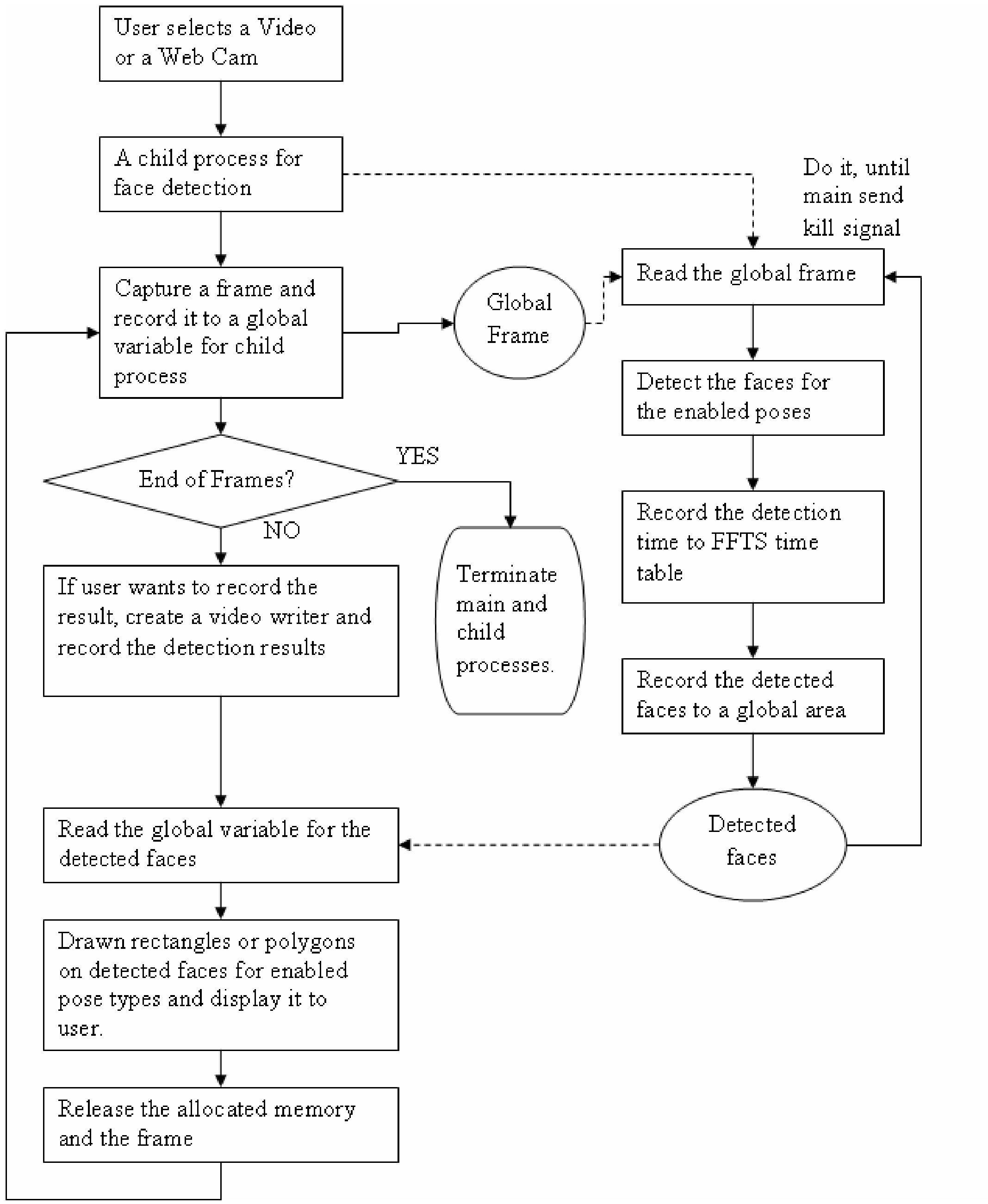

Figure 29: Flow Chart of Face Detection on Frames

The purpose of main process is retrieving every frame of video or Web Cam and display it to the user. Also, the retrieved frame is recorded to a global variable for use of the child process. Before displaying every frame to the user screen, the global variable which includes detected faces is read. If there are detected faces, rectangles or polygons are drawn to the detected areas.

In this approach, the detection is done periodically. Only one frame is processed every one second but all the frames are displayed to the user so neither frame lost nor real-time impact of frame processing is encountered.

3.5. Details of Proposed Face Tracking System

3.5.1. Introduction

Multi object tracking is performed and face tracking system of FFTS is based on skin color. There is a communication between face detection and face tracking systems. Not all skin color regions are detected but the skin color regions which is determined by face detection system is used here. Shortly, face detection gives the position of the face to the tracking system and after that tracking system works.

Our tracking system use CAMSHIFT algorithm which is very useful for Real-time processing because of it low computation time.

3.5.2. CAMSHIFT Algorithm

This algorithm is an adaptation of the Mean Shift Algorithm [52] for object tracking. The Mean Shift Algorithm is modified to deal with dynamically changing color probability distributions derived from frame sequences. It is a robust non-parametric iterative technique for finding the mode of probability distributions. The algorithm is designed to consume low computation time so a single channel (hue) is considered in the color model.

Video frames are normally in RGB color space but RGB color model is much more sensitive to lighting changes since saturation (which is influenced by lighting) is not separated out of that model hence, frames HSV color space conversion is required.

In CAMSHIFT, histogram of hue channel is quantized into bins. This required to reduce the computation time and to cluster similar color values together. The histogram bins then can be scaled between the minimum and maximum probability image intensities.

Table 5: Comparison of Mean Shift and CAMSHIFT Algorithms [6]

General Mean Shift Algorithm:	Derived CAMSHIFT algorithm:
Step 1: Choose a search window size. **Step 2.** Choose the initial location of the search window. **Step 3.** Compute the mean location in the search window. **Step 4.** Center the search window at the mean location computed in Step 3. **Step 5.** Repeat Steps 3 and 4 until convergence (or until the mean location moves less than a preset threshold).	**Step 1:** Choose the initial location of the search window. **Step 2:** Mean Shift as above (one or many iterations); store the zeroth moment. **Step 3:** Set the search window size equal to a function of the zeroth moment found in Step 2. **Step 4:** Repeat Steps 2 and 3 until convergence (mean location moves less than a preset threshold).

The mean location (the centroid) within the search window for a 2D image is calculated with Formulas 3.7 – 3.11;

First calculate the zeroth moment,

$$M_{00} = \sum_x \sum_y I(x, y) \quad \textbf{(3.7)}$$

Then, calculate the first moment of x and y respectively,

$$M_{10} = \sum_x \sum_y xI(x, y), \quad M_{01} = \sum_x \sum_y yI(x, y) \quad \textbf{(3.8)}$$

Finally, the location of mean search window which is the centroid;

$$x_c = \frac{M_{10}}{M_{00}}, y_c = \frac{M_{01}}{M_{00}} \quad \textbf{(3.9)}$$

3.5.3. Proposed Face Tracking System

The OpenCV platform contains an implementation of the CAMSHIFT algorithm which is used to tracks face movement using a one-dimensional histogram consisting of quantized channels from the HSV color space. This algorithm is implemented for multi face tracking. Tracking system is combined with the detection system as detected faces are given to the CAMSHIFT algorithm. Detection rate of system can be configured manually in the configuration pane of FFTS.

There are some difficulties for multi face tracking as listed below;

- There can be numerous faces in the given frame
- Trackers should be recorded in a robust way for further processing
- The faces in the frame can be too closer
- Size of the faces can be too smaller some times
- Zombie face trackers could exist because of disappeared
- Zombie tracker can track previously tracked faces
- Multiple tracking problem for the same face (In every detection cycle, the tracked faces could be detected again)

Our tracking system is robust for the aforementioned conditions.

Two new classes were written for tracking the faces. First class (Tracker Class) is aimed to record all information about a tracked face and the second class was written reach all the trackers easily. The second class (Tracker_LL) is used to keep all trackers in a linked list structure.

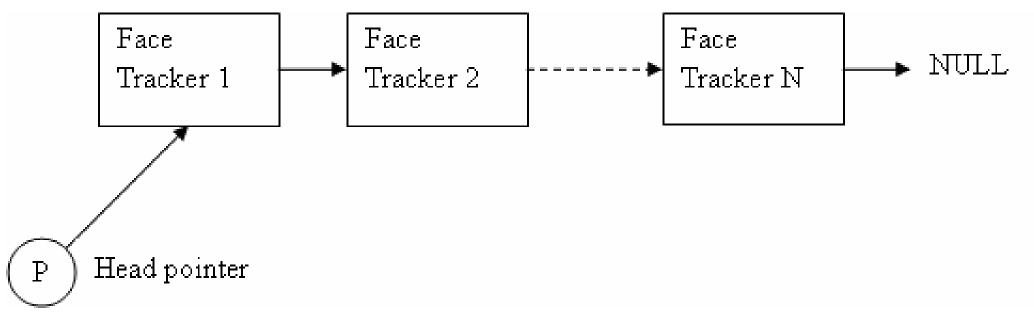

Figure 30: Tracker Linked List Structure

Tracked faces are drawn with green rectangles. FFTS track the faces until frames finished. The decision of tracker deletion is done under following conditions;

- If size (width or height) of tracked area is smaller than 10 pixels
- If corner of the trackers are inside each other
- If they are too near to another tracker (The difference between center of tracked regions are smaller than half of the total of minimum values of their width or height)

The same restrictions are done while new faces are detected by the child process and we are about to create new trackers for those faces. The detection time of the faces are written to the time table of FFTS by child process. Also, the total number of tracked faces is recorded to the time table to provide user number of current people in the camera view or video.

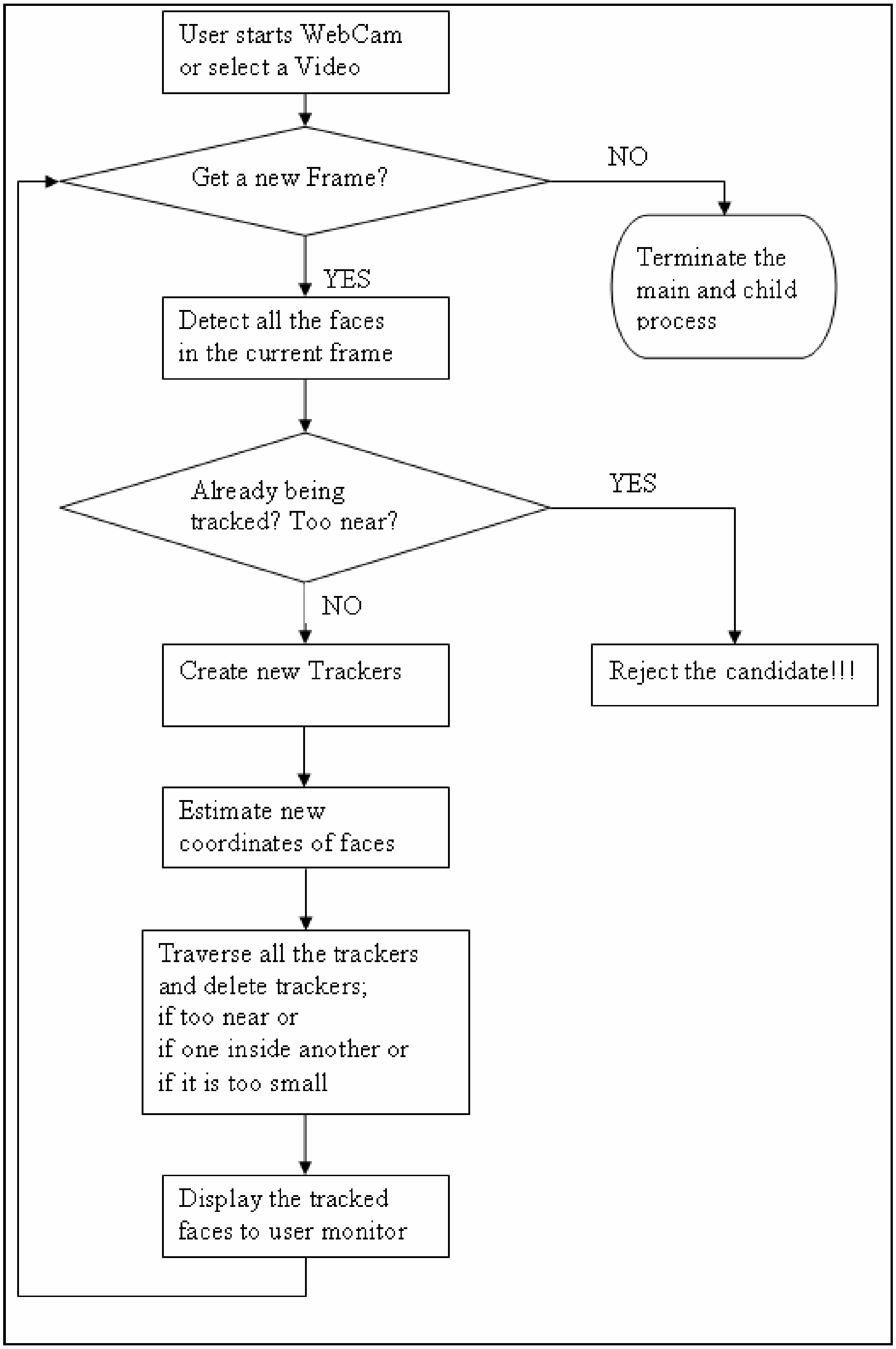

Figure 31: Flow Chart of Face Tracking

4. EXPERIMENTAL RESULTS

The study tested on three image databases (The UMIST Image Database [58], The IMM Image Database [56], MIT CBL Face Recognition Database [59]) and one video database (The VidTIMIT Audio-Video Dataset [57]). The results will be explained in detail one by one.

The UMIST Face Database consists of 1012 images of 20 distinct persons. However, some of the images were corrupted so they are deleted and tests were done on 933 images of 20 persons. Each person provides a wide range of poses from full left profile to full right profile. The image database is created with people of different race, race and an appearance. Face images of each person exist in different folders and the folders are labeled as 1a, 1b, …, 1t. The images all in PGM format with approximately 220 x 220 pixels and they are numbered consecutively as they were taken.

Figure 32: Some testing images from the UMIST Database

There are about 50 images per person in these database and the images were taken in a wide range between full left and full right profiles. The database was very useful to test our study for our five different pose detection. After that, the trained classifiers were tested on the UMIST database and very high detection rates was obtained.

Table 6: Test Results on the UMIST Image Database

Database Name	# of total Images	Correct Detections	False Positive	False Negative	Detection Rate
The UMIST Image DB	933	889	53	44	95%

For testing process a batch test application is implemented to the FFTS. This help us to accelerate the testing period. The batch test application can be reached in the FFTS panel over Menu->Batch Image Test. For this purpose all the test images renamed numerically and the number was increased by one. After the first test image was selected sequential testing was performed on that directory until all images tested. There is 2 second delay between every detection period and it enough to write down the detection results on a paper.

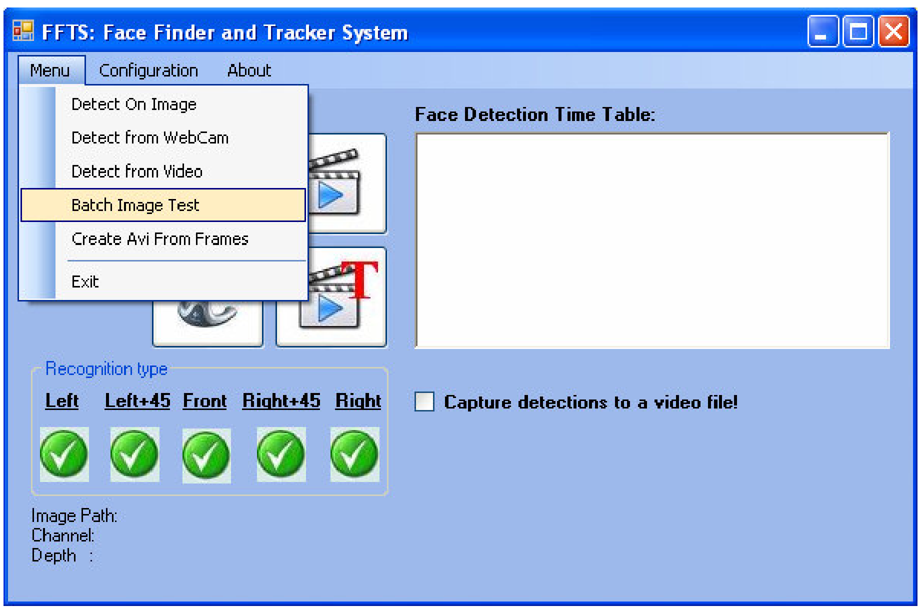

Figure 33: Batch Test Panel of FFTS to Accelarate Testing Process

The IMM image database consists of 240 annotated monocular images of 40 distinct human faces. All the faces are without glasses and there are 7 females with 33 males used for this database. Images are in 640x480 dimensions and they are in JPEG format. Frontal images are having neutral and happy facial expression and illuminated with diffuse light or spot light. Each person has poses as 30° turned to left side and right side. These poses are used for our right + 45° and left + 45° face detection type and results were pretty successful.

Table 7: Test Results on the IMM Image Database

Database Name	# of total Images	Correct Detections	False Positive	False Negative	Detection Rate
The IMM Image DB	240	234	13	6	97%

Figure 34: Some testing images from the IMM Image Database

MIT CBL Face Database is consists of 2000 images of 10 distinct person. The images were taken in a varied illumination, pose and background. The faces are all gray level images and all in PGM format. The sizes of the images are not fixed. Tests on this database present the same one with the previous two face database. High detection rates are seen in this database, too.

Table 8: Test Results on the MIT CBL Image Database

Database Name	# of total Images	Correct Detections	False Positive	False Negative	Detection Rate
MIT CBL Image DB	2000	1908	30	92	95%

FFTS also tested on The VidTIMIT Audio-Video Dataset for testing tracking faces on videos. The VidTIMIT dataset is consists of video and corresponding audio recordings of 43 people. There are 3 video for every subject in the database. This database provides the video frames instead of videos directly. Therefore another application is implemented under Menu panel of FFTS which "Create Avi From

Frames". For this purpose, the obtained frames are converted to JPG type images and they are combined together to present a video. This application provides a video based on the given frames and records it under frame's folder as "Sample.avi". Almost all of the faces are detected and tracked successfully.

Figure 35: Some testing images from the MIT CBL Image Database

Table 9: Test Results on VidTIMIT Video Dataset

Database Name	# of total Videos	Correct Tracking	False Positive	False Negative	Detection Rate
The VidTIMIT Audio-Video Dataset	129	129	3	0	97.6%

High detection rates obtained on the image databases. The list of tested image databases and test results are shown in Tables 6 - 8. Also the FFTS tested on The VidTIMIT Audi-Video Dataset [57] for Real-time face detection and tracking. High successful results obtained during face tracking on videos. The tracking results can be captured if the user ticked the recording option of the FFTS in the main pane. The video of the results recorded as "Results.avi".

Some of the test results on pictures taken from internet sites are listed below. In Figure 36, there is one face not detected by the frontal pose detector but it was later detected by the Right + 45° pose detector. In the last image of family photo, all of them pose detectors were run and all the faces were detected.

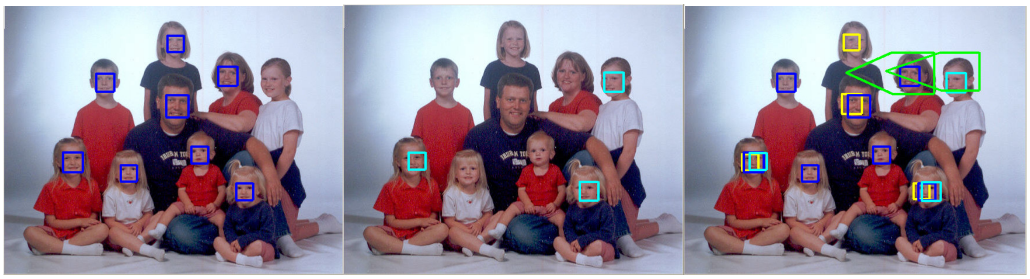

Figure 36: Test results of the FFTS on a family photo respectively for front pose detection, Right + 45° direction and for all directions.

Figure 37: All five pose detectors provide a very nice result on the photo of an amateur football team.

In Figure 38, all of the faces were detected and also it can be seen that most of the images were detected by more than one pose detector. This is a very useful since if one detector can not detect the face, other pose may detect them and this improves the detection rates of the faces.

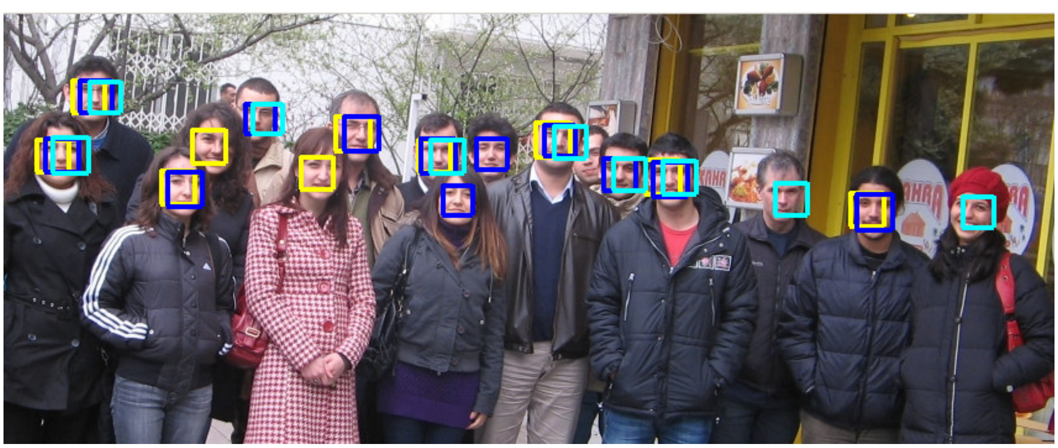

Figure 38: My colleagues in Nortel Networks – Netaş, İstanbul, March – 2007

Detection Results on some photos from the MIT+CMU Face Database [60];

Figure 39: The Adam's family, A photo from MIT+CMU Face Database

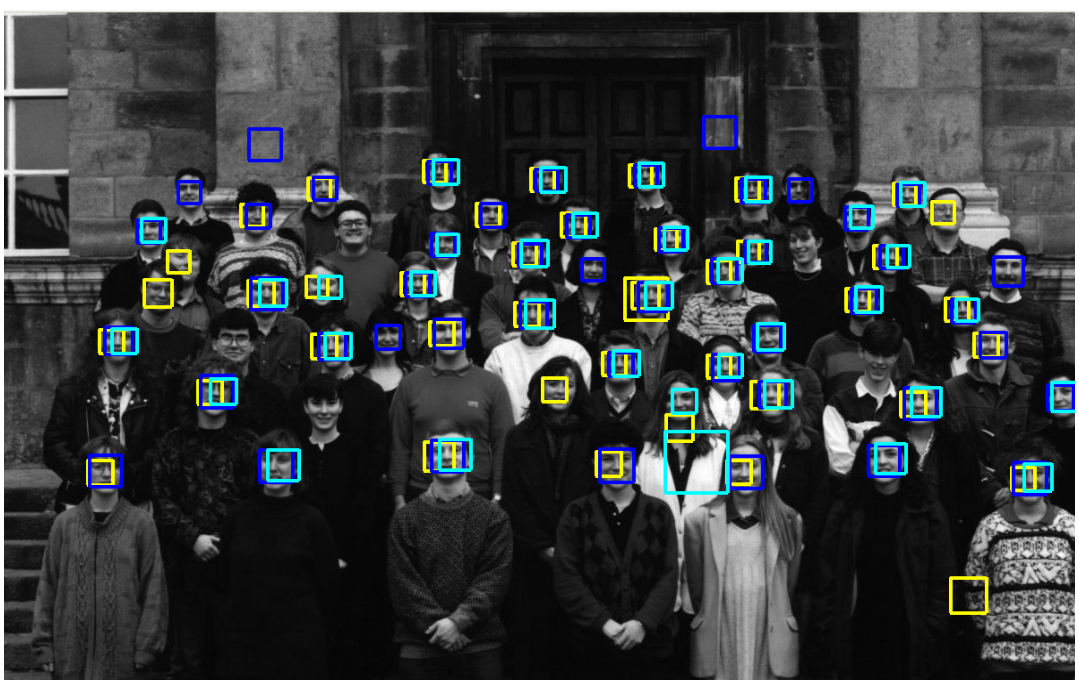

Figure 40: A class photo from MIT+CMU Face Database

In Figure 40, there are 57 faces and 52 of the faces were detected with different pose detectors. Also there are some false positives which means; actually there isn't any face but the region is detected as face.

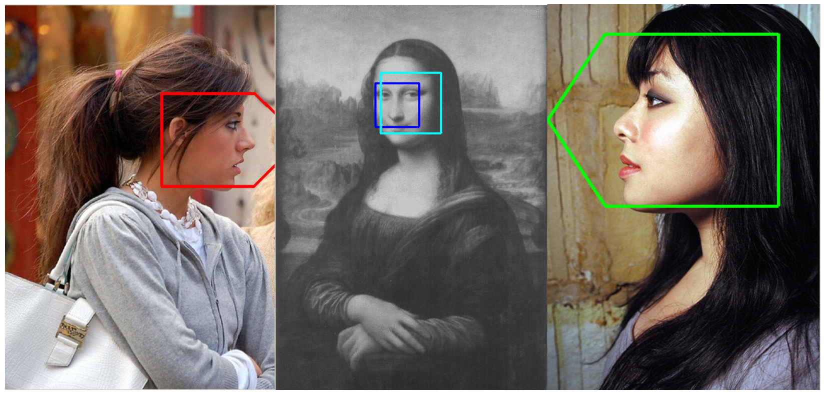

Figure 41: Some more examples from internet and MIT+CMU database

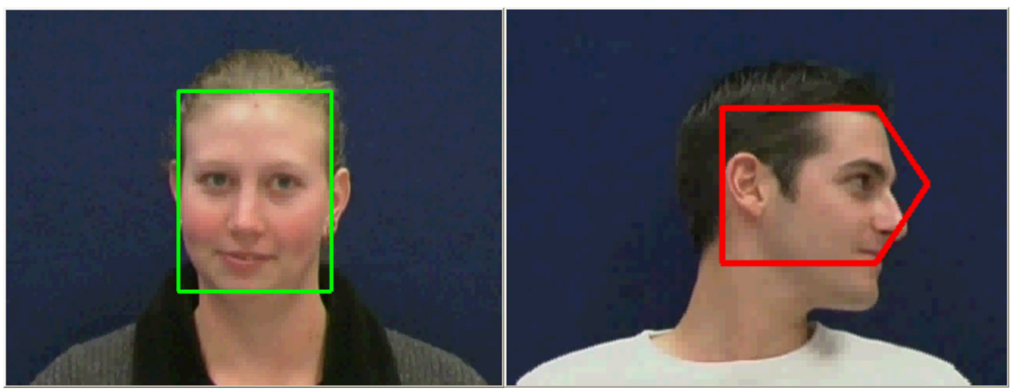

Figure 42: Face Tracking on the Left Frame and Face Detection on the Right Frame, Videos from VidTIMIT Audio-Video Dataset

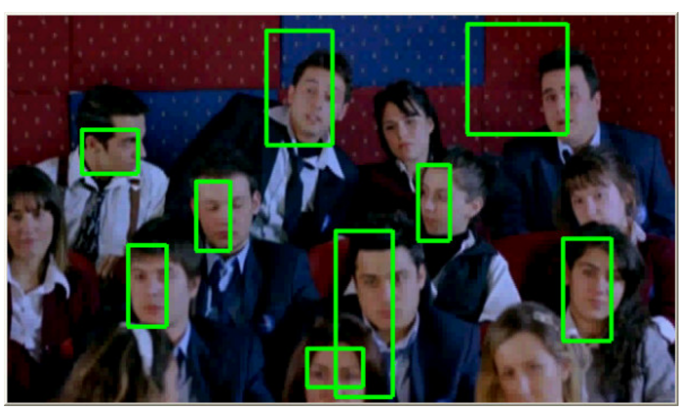

Figure 43: Multiple Face Tracking Results on a Cinema Film

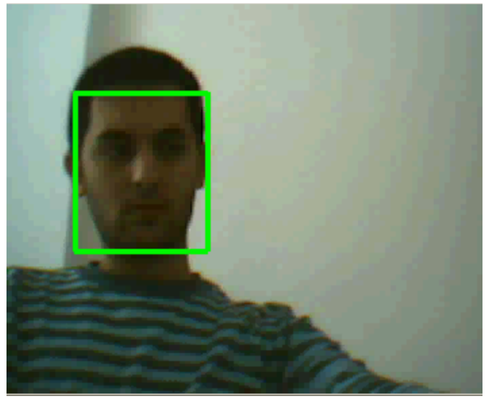

Figure 44: Single Face Tracking Result on My Own Video

5. CONCLUSION AND FUTURE WORK

The complexity and difficulties of the face detection problem has been shown and history of face detection methods is reviewed for comparison of proposed method. A combination of scale and pose invariant face detection and tracking system has been developed based on Adaboost algorithm and Haar-like facial features. The combination of Adaboost, Integral Image and Haar-like features has significant advantages with comparison to other methods so this approach is very popular among the Computer Vision community.

- FFTS has shown pretty good performance for all five pose detector based on the results of three image database and one video database
- Although the FFTS face detectors are running one by one for the enabled detection types, detection time is in reasonable range and very good for a Real-time application
- Parallel processing of face detection and displaying results by different two processes (main and child) increased the Real-time performance significantly
- Simple Haar-like features and cascading structure of Adaboost presented rapid detections with high accuracy
- The performance of the face detectors are directly interrelated with the prepared training set's accuracy and its largeness so more samples should be used for high detection rates.
- Tracking part of FFTS is working pretty well with the face detectors and the performance of CAMSHIFT algorithm is very nice in reasonable illuminations.
- The proposed method can be used as the first step of face applications such as face recognition, facial expression analysis etc.
- A pose detector can detect ~ ±15°-20° in plane rotations so that the proposed method can be applied for ~10-12 different poses to cover the full 360 degrees of possible rotations.

There are also some limitations of this method as other methods have;

- Proposed method can last for many days according to dataset, selected training stage and required accuracy
- Training needs for all poses to cover the full 360 degrees of possible rotations and the training process can take very long
- Tracking of the multiple faces in video is achieved but multiple tracking has Real-time impact

For future works;

- The proposed method can be used as the first step of face applications and new application (face recognition, facial expression recognitions, lip reading, human counting, video surveillance systems, etc.) can be build on FFTS
- The training set of FFTS can be improved for higher detection rates especially for profile poses
- The proposed method is works for enabled face detectors one by one and it can be improved. Instead of this, a pose estimator can be used to determine the pose and so only the related pose detector can run on that sub-window

In conclusion, based on the test of image and databases, our face detection and tracking method presents significantly high performance and it can be extended to different face applications.

6. REFERENCES

[1] **Yang, M.-H., Kriegman, D. and Ahuja, N.,** 2002. Detecting Faces in Images: A Survey, *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 24, no. 1, pp. 34-58.

[2] **Mohabbati, B., Shiri, M. and Kasaei, S.,** 2005. An efficient wavelet/neural network-based face detection algorithm*, Proceeding of the First International Conference on Modeling, Simulation and Applied Optimization, Sharjah, U.A.E. February 1-3.*

[3] **Huang, C., Ai, H., Li, Y. and Lao, S.,** 2005. Vector Boosting for Rotation Invariant Multi-View Face Detection, *Tenth IEEE International Conference on Computer Vision (ICCV'05),* vol. 1, pp. 446-453.

[4] **The University of Surrey Centre for Vision,** Speech and Signal Processing, Photometric Normalization, http://www.ee.surrey.ac.uk/CVSSP/SignalProcessing /Biometrics/Illumination.

[5] **Viola, P. and Jones, M.,** 2001. Rapid object detection using a boosted cascade of simple features, *Proc. Intl. Conf. on Computer Vision and Pattern Recognition (CVPR),* pp.511–518.

[6] **Bradski, G. R.,** 1998. Real time face and object tracking as a component of a perceptual user interface, *Proc. of the Fourth IEEE Workshop on Applications of Computer Vision*, pp.214-219.

[7] **Samal, A. and Iyengar, P.A.,** 1992. Automatic Recognition and Analysis of Human Faces and Facial Expressions: A Survey, *Pattern Recognition*, vol. 25, no. 1, pp. 65-77.

[8] **Chellappa, R., Wilson, C.L., and Sirohey, S.,** 1995. Human and Machine Recognition of Faces: A Survey, *Proc. IEEE*, vol. 83, no. 5, pp. 705-740.

[9] **Kanade, T.,** 1973. ``Computer Recognition of Human Faces,'' Basel and Stuttgart: Birkhauser.

[10] **Kelly, M.D.,** 1970. “Visual Identification of People by Computer,” Technical Report AI-130, Stanford AI Project, Stanford, CA.

[11] **Kotropoulos, C. and Pitas, I.,** 1997. “Rule-Based Face Detection in Frontal Views,” *Proc. Int’l Conf. Acoustics, Speech and Signal Processing*, vol. 4, pp. 2537-2540.

[12] **Yang, G. and Huang, T. S.,** 1994. "Human Face Detection in Complex Background," *Pattern Recognition*, vol. 27, no. 1, pp. 53-63.

[13] **Pigeon, S. and Vandendrope, L.,** 1997. The M2VTS Multimodal Face Database, *Proc. First Int'l Conf. Audio- and Video-Based Biometric Person Authentication.*

[14] **Chiang, C.C., Huang, Y. T. and Shi, Y. X.,** 2002. A Rule-based Real-Time Face Detector, *In Proc. of International Computer Symposium.*

[15] **Lin, C., Fan, K.C.,** 2000. Human Face Detection Using Geometric Triangle Relationship, *15th International Conference on Pattern Recognition (ICPR'00),* vol. 2*,* p. 2941.

[16] **Ho, K., Chen, T.S., Chang, M.Y. and Chen, J.,** 2008. "New feature based human face detection method", The Imaging Science Journal, vol. 56, no. 1, pp. 56-61.

[17] **Sirohey, S.A.,** 1993. Human Face Segmentation and Identification, *Technical Report CS-TR-3176,* Univ. of Maryland.

[18] **Chetverikov, D. and Lerch, A.,** 1993. "Multiresolution Face Detection," Theoretical Foundations of Computer Vision, vol. 69, pp. 131-140.

[19] **Graf, H.P., Chen, T., Petajan, E. and Cosatto, E.,** 1995. Locating Faces and Facial Parts, *Proc. First Int'l Workshop Automatic Face and Gesture Recognition,* pp. 41-46.

[20] **Leung, T.K., Burl, M.C. and Perona, P.,** 1995. Finding Faces in Cluttered Scenes Using Random Labeled Graph Matching, *Proc. Fifth IEEE Int'l Conf. Computer Vision,* pp. 637-644.

[21] **Yow, K.C. and Cipolla, R.,** 1997. Feature-Based Human Face Detection, Image and Vision Computing, vol. 15, no. 9, pp. 713-735.

[22] **Han, C.-C., Liao, H.-Y.M., Yu, K.-C. and Chen, L.-H.,** 1998. Fast Face Detection via Morphology-Based Pre-Processing, *Proc. Ninth Int'l Conf. Image Analysis and Processing,* pp. 469-476.

[23] **Dai, Y. and Nakano, Y.,** 1996. Face-Texture Model Based on SGLD and Its Application in Face Detection in a Color Scene, Pattern Recognition, vol. 29, no. 6, pp. 1007-1017.

[24] **Augusteijn, M.F. and Skujca, T.L.,** 1993. Identification of Human Faces through Texture-Based Feature Recognition and Neural Network Technology, Proc. IEEE Conf. Neural Networks, pp. 392-398.

[25] **Lai, W.-H., Li, C.-T.,** 2006. Skin Colour-Based Face Detection in Colour Images, *Proc. IEEE Conf. Video and Signal Based Surveillance,* pp. 56.

[26] **Zhang, Q. and Izquierdo,** 2006. E., MULTI-FEATURE BASED FACE DETECTION, *Proc. IET Int'l Conf. Visual Information Engineering,* pp. 572-576.

[27] **Sakai, T., Nagao, M. and Fujibayashi, S.,** 1969. Line Extraction and Pattern Detection in a Photograph, Pattern Recognition, vol. 1, pp. 233-248.

[28] **Craw, I., Ellis, H. and Lishman, J.,** 1987. Automatic Extraction of Face Features, Pattern Recognition Letters, vol. 5, pp. 183-187.

[29] **Govindaraju, V., Sher, D.B., Srihari, R.K. and Srihari, S.N**., 1989. Locating Human Faces in Newspaper Photographs, *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 549-554.

[30] **Tsukamoto, A., Lee, C.-W. and Tsuji, S.,** 1994. Detection and Pose Estimation of Human Face with Synthesized Image Models, *Proc. Int'l Conf. Pattern Recognition,* pp. 754-757, 1994.

[31] **Samal, A. and Iyengar, P.A.**, 1995. "Human Face Detection Using Silhouettes," Int'l J. Pattern Recognition and Artificial Intelligence, vol. 9, no. 6, pp. 845-867.

[32] **Miao, J., Yin, B., Wang, K., Shen, L. and Chen, X.,** 1999. A Hierarchical Multiscale and Multiangle System for Human Face Detection in a Complex Background Using Gravity-Center Template, Pattern Recognition, vol. 32, no. 7, pp. 1237-1248.

[33] **Yuille, A., Hallinan, P. and Cohen, D.,** 1992. Feature Extraction from Faces Using Deformable Templates, *Int'l J. Computer Vision*, vol. 8, no. 2, pp. 99-111.

[34] **Jin, Z., Lou, Z., Yang, J. and Sun, Q.,** 2007. Face detection using template matching and skin-color information, Neurocomputing, v.70 n.4-6, p.794-800.

[35] **Kohonen T.,** 1989. *Self-Organization and Associative Memory.*

[36] **Jie, Y., Xufeng, L., Yitan, Z. and Zhonglong, Z,** 2008. "A face detection and recognition system in color image series," Mathematics and Computers in Simulation, vol. 77, no. 5-6, pp. 531-539.

[37] **Sung, K.-K. and Poggio, T.,** 1998. "Example-Based Learning for View- Based Human Face Detection," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 20, no. 1, pp. 39-51.

[38] **Wang, H., Li, P. and Zhang, T.,** 2008. Histogram feature-based Fisher linear discriminant for face detection, Neural Computing & Applications, vol. 17, no. 1, pp. 49-58.

[39] **Agui, T., Kokubo, Y., Nagashashi, H. and Nagao, T.,** 1992. Extraction of FaceRecognition from Monochromatic Photographs Using Neural Networks, *Proc. Second Int'l Conf. Automation, Robotics, and Computer Vision,* vol. 1, pp. 18.8.1-18.8.5.

[40] **Rowley, H., Baluja, S. and Kanade, T.,** 1996. Human Face Detection in Visual Scenes, Advances in Neural Information Processing Systems 8, D.S. Touretzky, M.C. Mozer, and M.E. Hasselmo, eds., pp. 875- 881.

[41] **Osuna, E., Freund, R. and Girosi, F.,** 1997. Training Support Vector Machines: An Application to Face Detection, *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 130-136.

[42] **Chen, J., Wang, R., Yan, S., Shan, S., Chen, X. and Gao, W.,** 2007. Enhancing Human Face Detection by Resampling Examples Through Manifolds, *Proc. IEEE Transactions on Systems,* Man & Cybernetics, vol. 37, no. 6, pp.1017-1028.

[43] **Yang, M.-H., Roth, D. and Ahuja, N.,** 2000. A SNoW-Based Face Detector, Advances in Neural Information Processing Systems 12, pp. 855-861.

[44] **Littlestone, N.,** 1988. Learning Quickly when Irrelevant Attributes Abound: A New Linear-Threshold Algorithm, Machine Learning, vol. 2, pp. 285-318, 1988.

[45] **Gundimada, S. and Asari, V.,** 2005. An Improved SNoW Based Classification Technique for Head-pose Estimation and Face Detection, Applied Imagery and Pattern Recognition Workshop, pp. 94-99.

[46] **Freund, Y. and Schapire, R. E.,** 1995. A decision-theoretic generalization of on-line learning and an application to boosting, In *European Conference on Computational Learning Theory*, pages 23–37.

[47] **Jones, M. and Viola, P.,** 2001. Rapid Object Detection using a Boosted Cascade of Simple Features, In Proc. IEEE Conf. on Computer Vision and Pattern Recognition, Kauai, Hawaii, USA.

[48] **Jones, M. and Viola, P.,** 2003. Fast Multi-view Face Detection, *In IEEE Conference on Computer Vision and Pattern Recognition.*

[49] **Wu, B., Ai, H., Huang, C. and Lao, S.,** 2004. "Fast Rotation Invariant Multi-View Face Detection Based on Real AdaBoost," Proc. Sixth Int'l Conf. Automatic Face and Gesture Recognition, pp. 79-84.

[50] **Ichikawa, K., Mita, T. and Hori, O.,** 2006. **"**Component-based robust face detection using AdaBoost and decision tree," In Proc. Int'l Conf. on Automatic Face and Gesture Recognition, pp. 413-420.

[51] **Wu, J., Brubaker, S.C., Mullin, M.D., and Rehg, J.M.,** 2008. Fast Asymmetric Learning for Cascade Face Detection, *Proc. IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 30, no. 3, pp. 369-382.

[52] **Fukunaga, K. and Hostetler, L. D.,** 1975. "The estimation of the gradient of a density function, with applications in pattern recognition," *IEEE Trans. Information Theory*, vol. 21, pp. 32 – 40.

[53] **Sochman J.,** 2004. AdaBoost for Fast Face Detection, *PhD Thesis*, Czech Technical University, Prague.

[54] **Tarrés, F. and Rama, A.,** GTAV Face Database: http://gps-tsc.upc.es/GTAV/ResearchAreas/UPCFaceDatabase/GTAVFaceDatabase.htm

[55] **Solina, F., Peer, P., Batagelj, B., Juvan, S., Kovac, J., 2003.** Color-based face detection in the 15 seconds of fame art installation, Conference on Computer Vision / Computer Graphics Collaboration for Model-based Imaging in Mirage.

[56] **Stegmann, M.B., Ersboll, B.K. and Larsen, R.,** 2003. FAME - a flexible appearance modeling environment. *IEEE Trans. on Medical Imaging*, 22(10):1319-1331.

[57] **Sanderson, C. and Paliwal, K.K.,** 2004. Identity Verification Using Speech and Face Information, Digital Signal Processing, vol. 14, no. 5, pp. 449-480.

[58] **Graham, D. B. and Allinson, N. M.,** 1998. Characterizing virtual eigensignatures for general purpose face recognition, in: Face Recognition: From Theory to Applications, H. Wechsler, P. J. Phillips, V. Bruce, F. Fogelman-Soulie, and T. S. Huang (Eds.), NATO ASI Series F, Computer and Systems Sciences, pp. 446-456.

[59] **Weyrauch, B., Huang, J., Heisele, B. and Blanz, V.,** 2004. Component-based Face Recognition with 3D Morphable Models, *First IEEE Workshop on Face Processing in Video, Washington*, D.C..

[60] **CMU**, Vision and autonomous systems center's (VASC) image database: http://vasc.ri.cmu.edu/idb/

APPENDIX A: C++.NET AND OPENCV PLATFORM SETUPS

Step 1: Download C++.NET 2005 Express IDE or upper version from;
http://msdn2.microsoft.com/en-us/express/aa975050.aspx

Step 2: In the Express version of C++ you should install SDK and apply the required changes mentioned from;
http://msdn.microsoft.com/en-us/express/aa700755.aspx

Step 3: Download OpenCV platform from;
http://sourceforge.net/projects/opencv/

Step 4: Run setup programs for OpenCV and C++.NET 2005 and install them to their default directory. E.g.: Install OpenCV to following folder “C:\Program Files\OpenCV\”

Step 5: Open the Visual C++ .Net Application. In the menu bar, select **Tools-> Options**

- In the listing, choose Projects->VC++ Directories.
- First, select Library files from the "Show Directories for" List Box.
- Click the Insert New icon, and locate the folder where you have installed OpenCV.
- Consider that it is installed in "C:/Program Files/OpenCV".
- In the Library files list, locate and add: "C:\Program Files\OpenCV\lib"

Step 6: Choose Include files in the list box, and locate and add the following directories:

- "C:\Program Files\OpenCV\cv\include"
- "C:\Program Files\OpenCV\cxcore\include"
- "C:\Program Files\OpenCV\otherlibs\highgui"
- "C:\Program Files\OpenCV\cvaux\include"
- "C:\Program Files\OpenCV\otherlibs\cvcam\include"

Step 7: Choose source files in the list box, and locate and add the following directories;

- "C:\Program Files\OpenCV\cv\src"
- "C:\Program Files\OpenCV\cxcore\src"
- "C:\Program Files\OpenCV\cvaux\src"
- "C:\Program Files\OpenCV\otherlibs\highgui"
- "C:\Program Files\OpenCV\otherlibs\cvcam\src\windows"

Step 8: Now click OK in the Options dialog. To configure the global settings successfully.

Step 9: Create new form application within C++.NET Studio;

- Select from menu "File"->"New"->"Projects" tab
- Choose "Windows Form Application" as in figure
- Type the project name and click OK to open the new form application

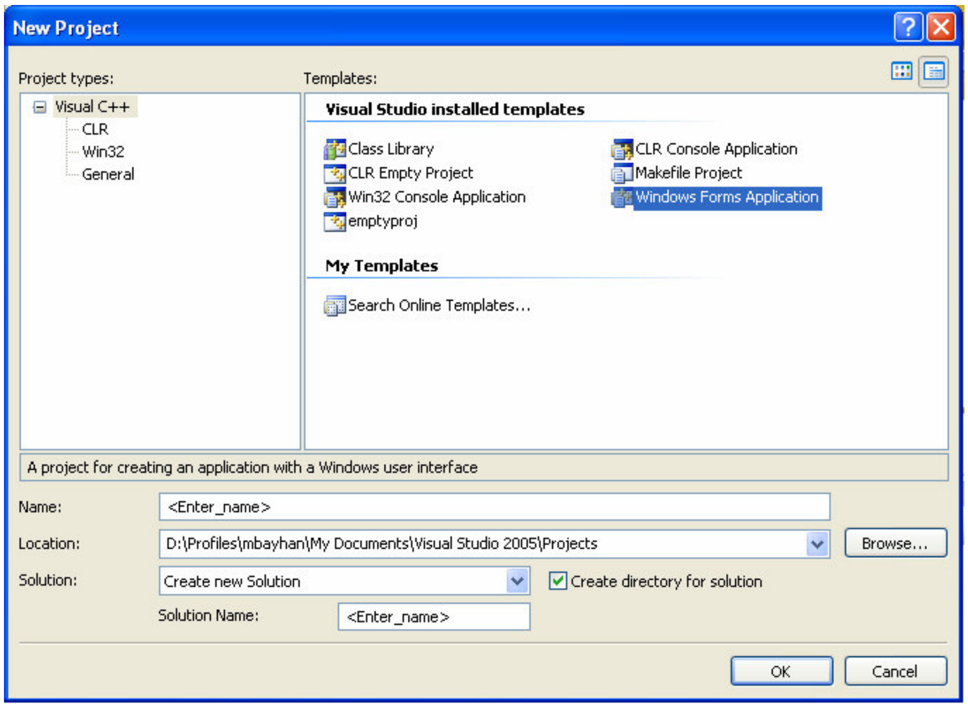

Figure 45: Open a New Form in C++.NET IDE

Step 10: For example, consider that we have created a new "Test_For_Thesis" Project;

- **Open the Form1.h and add the required headers as below;**
  - #include <cv.h>
  - #include <cxcore.h>
  - #include <highgui.h>
- **Add dependency projects into workspace**
  - Choose from menu: "Project" -> " Test_For_Thesis Properties".
  - Choose "Linker" tab -> "Input" category -> "Additional Dependencies:"
  - Add the paths to all necessary import libraries as "cv.lib cxcore.lib highgui.lib cvaux.lib cvcam.lib" one space between them.

Step 11: Thats it!.. Now Build and Run the application using F5 key. If the build process complains about a missing 'windows.h' header file, then you'll need to install the latest version of the Microsoft Windows SDK mentioned in Step 2.

APPENDIX B: METHODS TO PREPARE DATASETS FOR TRAINING

There is no need to crop negative images. They can be downloaded and saved to a different folder. The following steps are required to crop face regions and obtain positive images.

Step 1: Crop the faces from the collected database

- Only face regions should be cropped
- The cropped regions should be similar for the same training set

Step 2: A very useful commercial photo editor is used in this study. It can be downloaded from the following site: http://www.click2crop.com/

Step 3: Install the program and run it

Step 4: Select an image folder from the File->Open folder..

Step 5: Select and image and modify the cropping rectangle manually to adjust required region to be cropped. The cropped region displayed in the right menu as in Figure 43.

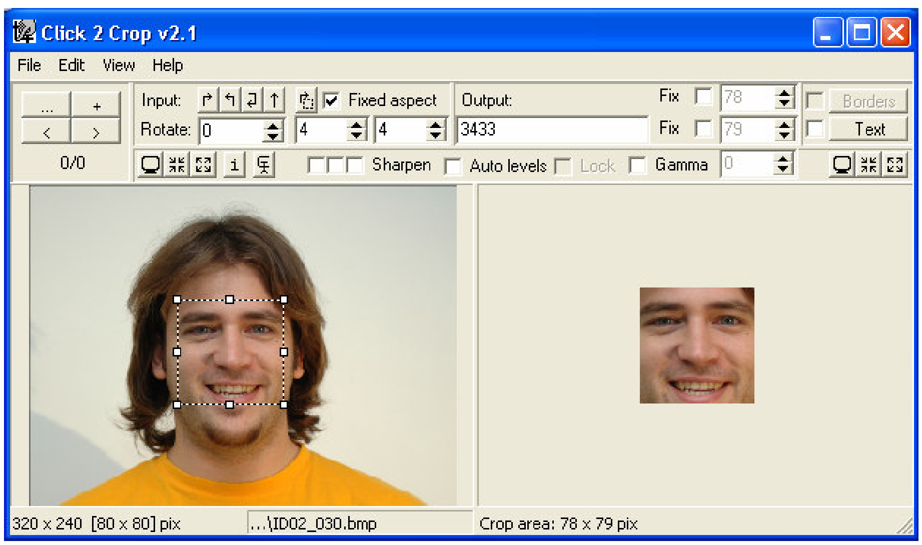

Figure 46: Cropping Face Area with Click2Crop Program

Step 6: Enter a numeric number to Output text field to give a name the cropped region

Step 7: Double click the cropping rectangle or press the "+" button to save it to a user specified folder.

Step 8: Press the ">" button to go to the next image and crop another face in the same directory

Step 9: Do Step 7 and Step 8 until all the faces of that folder cropped. That's it…

APPENDIX C: FACE DATA TRAINING IN OPENCV PLATFORM

In the previous appendix it has been explained how obtain negative and positive image samples. In this appendix the training phase of the positive and negative images will be presented.

Step 1: Produce a text file including the path of positive samples for each poses. A sample is shown in Figure 44. This data is created as follows;

Table 10: Preparing input for Createsample.exe Training of Positives

Image path	# of faces	x coord	y coord	width	length	x2 coord.	y2 coord.	width2	length2
1.bmp	1	0	0	24	24				
2.bmp	2	0	0	24	24	0	40	24	24

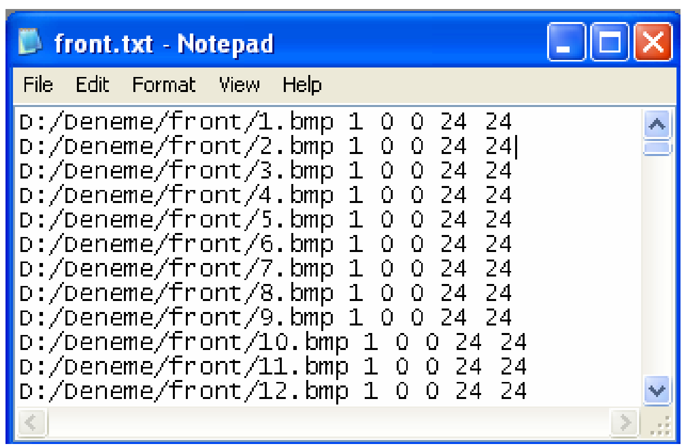

Figure 47: Sample Positive Image File for Training

Step 2: Create vector files with "createsamples.exe" which is distributed by OpenCV. It can be reached in folder "C:\Program Files\OpenCV\bin"

Step 3: Copy the prepared file for positive samples under the same directory of "createsample.exe". Open a windows terminal ("Start"-> "Run"-> "cmd" in Windows Operating System) and change the current directory to createsample.exe's directory ("cd C:\Program Files\OpenCV\bin").

Step 4: Modify the following command to represent your data and run the createsample.exe in the window's terminal with the below command;

createsamples.exe -info front.txt -vec front.vec -num 3311 -w 24 –h 24

– info parameter tells the program to read image details from front.txt file.
– vec parameter tells the program to save vectorial file as front.vec.
– num parameter is used to inform # of positive sample in front.txt.
– w and – h parameters are used to normalize the width and height of the positive samples as 24 pixels.

Step 5: The previous step will create a vector file for the positive samples under the directory of createsample.exe.

Step 6: Create another –info file for Negative images. It will just contain the path and name of the images as shown in Figure 45. All the files were renamed numerically in the beginning but it is not needed.

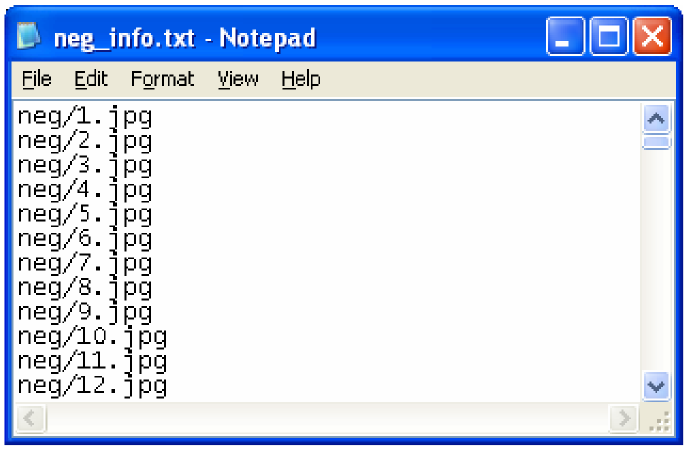

Figure 48: Sample Negative Image File for Training

Step 6: To train the positive samples, "haartraining.exe" is used and it is distributed by OpenCV. It can be reached in folder "C:\Program Files\OpenCV\bin".

Step 7: Open a windows terminal as described in Step 3 and modify the following command according to your data and run it.

haartraining.exe -data data/frontcascade -vec front.vec -bg neg_info.txt -npos 3311 -nneg 4973 -nstages 20 -mem 900 -mode ALL -w 24 -h 24

- data <directory name>	: Cascades of classifier are saved in this folder
- vec <vectorial file>	: Contains the positive sample's information
- bg <bkg_filename>	: Contains the negatives sample's info
- npos <#_of_positives>	: Number of positive samples
- nneg <#_of_negatives>	: Number of negative samples
- nstages <#_of_stages>	: Number of stage in cascade during training
- mem	: Memory assigned foe processing
- mode <mode_type>	: Specify the type of Haar-like Features to be used
-w and -h	: To determine the minimal size of positives

BIOGRAPHY

Mehmet Şerif Bayhan was born in Diyarbakır, Turkey, in 1983. He graduated from Diyarbakır Science High School in 2000 with a high degree and demonsrated his sucess in the university entrance examination. He has received his B.Sc. in Computer Enginering from İstanbul Technical University in 2005 with honours.

He has been working in Nortel Networks – Netaş as a Global Product Support and Design engineer since 2005.