

121

# AUTOMATIC VIDEO INDEXING AND RETRIEVAL WITH SKIN COLOR BASED FACE DETECTION METHOD

A Thesis Submitted to the  
Graduate School of Natural and Applied Sciences of  
Dokuz Eylül University  
In Partial Fulfillment of the Requirements for the Degree of Master of Science  
in Computer Engineering, Computer Engineering Program

T.C. YÜKSEK ÖĞRETİM KURULU  
DOKÜMANTASYON MERKEZİ

by  
Taner DANIŞMAN

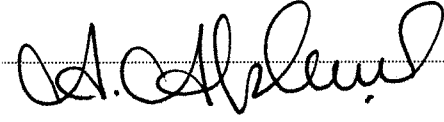
119578

July, 2002  
İZMİR

119578

## Ms.Sc. THESIS EXAMINATION RESULT FORM

We certify that we have read the thesis, entitled “**AUTOMATIC VIDEO INDEXING AND RETRIEVAL WITH SKIN COLOR BASED FACE DETECTION METHOD**” completed by **TANER DANIŞMAN** under supervision of **ASSIST.PROF.DR ADIL ALPKOÇAK** and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.



Assist. Prof.Dr. Adil ALPKOÇAK  
Supervisor

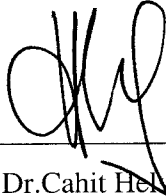


Prof.Dr. Cüneyt GÜZELİŞ  
Committee Member



Prof.Dr. İrem ÖZKARAHAN  
Committee Member

Approved by the  
Graduate School of Natural and Applied Sciences



Prof.Dr.Cahit Helvacı  
Director

---

## ACKNOWLEDGMENTS

---

I would like to thank all those people for giving me support and help in everything I do for accomplishing this project. My dearest mother Asiye DANIŞMAN, my father Ahmet DANIŞMAN and my brother Ertan DANIŞMAN, for all my life you will be my inspiration.

I am grateful to my thesis advisor, Assist. Prof. Dr. Adil ALPKOÇAK for his guidance, helpful suggestions and comments during my study, and supporting me by providing his multimedia equipments for the thesis.

Taner DANIŞMAN

---

## ABSTRACT

---

This thesis covers face detection, face tracking, facial expressions, video indexing and retrieval subjects and presents a system for automatically detecting, extracting, indexing and retrieval of human facial information from raw video.

In addition, it also introduces a new object motion based shot detection method. Objects are tracked and their locations are saved to detect video shots boundaries. If the object changes its previous location then a shot is detected.

The system extracts human face information from video frames and then locates facial features of the face. Time required to extract human facial information is about 0.33 seconds (3 fps) in average. Each of the face information in each frame is stored in a database of video frames. The user is then either uses the pre-extracted information in retrieval of faces or defines trajectories of human face objects in order to retrieve specific types of actions performed by the face.

The system developed using Borland C++ Builder 5.0 and can operate on AVI and MPEG files as well as real-time live video feeds from video camera.

**Keywords:** Video indexing, video retrieval, computer vision, face detection, face tracking, facial expressions

---

## ÖZET

---

Bu tez, yüz bulma, yüzsel ifadeler, video dizinleme ve sorgulama konularını içerir ve ham bilgidan otomatik olarak insan yüzüne ait bilgilerin bulunması, çıkarılması, dizinlenmesi ve sorgulanması üzerine bir sistem sunar.

Ek olarak nesne hareketi tabanlı yeni bir geçiş tanıma yöntemini tanıtır. Geçiş sınırlarını tanımak için nesnelere izlenir ve yerleri kaydedilir. Eğer nesne bir önceki yerini değiştirirse geçiş noktası bulunur.

Sistem, insan yüz bilgilerini video karelerinden çıkarır ve yüze ait olan yüzsel özellikleri belirler. İnsan yüzüne ait bilgilerin çıkarılması ortalama olarak 0.33 saniye sürmektedir (3 fps). Her video karesindeki her yüz bilgileri, video karelerini tutan bir veri tabanı içerisinde saklanır. Daha sonra, kullanıcı ya daha önceden çıkarılmış olan bilgiyi yüzlerin sorgulanmasında kullanır, ya da insan yüzüne ait yörüngeler tanımlayarak, yüzün yapmış olduğu özel hareketleri sorgulayabilir.

Bu sistem Borland C++ Builder 5.0 ile geliştirilmiştir. AVI ve MPEG dosyaları ile çalışabildiği gibi video kameradan gelen gerçek zamanlı görüntülerle de çalışabilmektedir.

**Anahtar Sözcükler:** Video dizinleme, video sorgulama, bilgisayar görüşü, yüz tanımlama, yüz izleme, yüzsel ifadeler

---

# CONTENTS

---

	Page
Contents.....	VII
List of Tables .....	XI
List of Figures .....	XII

## Chapter One INTRODUCTION

1.1 Aim of Thesis .....	2
1.2 Thesis Organization.....	3

## Chapter Two FACE DETECTION

2.1 Introduction .....	4
2.2 Object Detection.....	5
2.3 Face Detection with Static Background .....	6
2.4 Face Detection Using Skin Color .....	6
2.5 Face Detection by Motion Analysis .....	8
2.6 Feature Based Methods .....	10
2.7 Neural Network-Based Face Detection .....	11
2.8 Hidden Markov Models (HMM).....	13

2.9 Face Detection Study.....	15
-------------------------------	----

### Chapter Three

#### FACIAL EXPRESSIONS

3.1 Introduction .....	31
3.2 Background .....	32
3.3 Facial Expression using Neural Network-Based Models.....	40
3.4 Facial Expression using Hidden Markov Models .....	41
3.5 Facial Expression Recognition Study.....	42

### Chapter Four

#### VIDEO INDEXING and RETRIEVAL

4.1 Introduction .....	44
4.2 Video Analysis Techniques.....	46
4.2.1 Shot Boundary Detection .....	48
4.2.1.1 Pair-wise Pixel Difference.....	49
4.2.1.2 Histogram Comparison.....	50
4.2.1.3 Motion Continuity .....	51
4.2.1.4 Three frames Approach .....	52
4.2.1.5 Twin-Comparison Method .....	52
4.2.1.6 Compression Differences .....	53
4.2.1.7 Edge Tracking .....	54
4.2.2 Gradual Transition Detection .....	54
4.2.2.1 Plateau Detection.....	55
4.2.2.2 Feature Based Detection.....	55
4.2.3 Key Frame Selection .....	56

4.2.3.1 Average Pixel Method.....	56
4.3 Critical Video Formats .....	57
4.3.1 Audio Video Interleave (AVI).....	57
4.3.2 Moving Pictures Expert Group (MPEG).....	58
4.4 The Application Metadata Standards for Video Indexing.....	59
4.4.1 Dublin Core .....	59
4.4.2 MPEG-7.....	61
4.4.2.1 The Scope of MPEG-7 .....	61
4.4.2.2 Mpeg-7 Capabilities .....	62
4.5 Video Indexing Applications.....	62
4.5.1 Name-It (CMU and NACSIS).....	63
4.5.2 Skim Generation (CMU).....	63
4.6 Shot Detection Study.....	63
4.7 Video Indexing and Retrieval Study .....	64
4.7.1 Introduction .....	65
4.7.2 Video Indexing & Retrieval Results.....	73

## Chapter Five

### IMPLEMENTATION and EVOLUATION

5.1 Face Detection Study.....	75
5.1.1 Coding .....	75
5.1.2 Face Detection Study Results.....	76
5.2 Shot Detection Study.....	83
5.2.1 Coding .....	83
5.2.2 Interface.....	83
5.2.3 Shot Detection Study Results .....	83
5.3 Video Indexing and Retrieval.....	86
5.3.1 Coding .....	86
5.3.2 Interface.....	87



5.3.2.1 Real-Time Video Indexing ..... 88

5.3.2.2 Offline Video Indexing..... 93

5.3.3 Video Indexing and Retrieval Study Results..... 101

**CONCLUSION.....103**

**References.....106**



---

## LIST OF TABLES

---

	<b>Page</b>
Table 3.1 List of Action Units.....	32
Table 3.2 Details of facial points in Figure 3.8.....	39
Table 5.1 Single Face Detection results (Recall & Precision).....	77
Table 5.2 Multiple Face Detection results (Recall & Precision).....	77



---

## LIST OF FIGURES

---

	Page
Figure 2.1 Faces with different effects (rotation, brightness, background, etc.).....	5
Figure 2.2 Static Background in blue color.....	6
Figure 2.3 Background Removing & Segmentation .....	7
Figure 2.4 Typical motion image (Hjelmås et al, 1998).....	8
Figure 2.5 Amount of pixels on each line in the motion image (Hjelmås et al, 1998)	9
Figure 2.6 The Olivetti Research Laboratory (ORL) face database.....	10
Figure 2.7 Example image from the work (Rowley, 1999).....	12
Figure 2.8 Five state HMM .....	13
Figure 2.9 Sliding windows for generating observation vector .....	14
Figure 2.10 Horizontal Hidden Markov Model (Samaria, 1994).....	15
Figure 2.11 Step by step Face Detection .....	16
Figure 2.12 Removing static background.....	17
Figure 2.13 Removing dynamic background .....	18
Figure 2.14 Light effects on face.....	19
Figure 2.15 Eliminating small sized non-face regions .....	21
Figure 2.16 Eliminating both thin and thick geometric regions.....	23
Figure 2.17 Facial feature detection.....	25
Figure 2.18 Face template .....	26
Figure 2.19 Facial Feature localization .....	27
Figure 2.20 Detection of multiple faces .....	29
Figure 2.21 False Drops in detecting multiple faces. ....	30
Figure 3.1 Sample six basic facial expression data set from (Cohen, 2000, pp. 8-30) .....	34
Figure 3.2 Two different types of smile. (Lien, 1998).....	35
Figure 3.3 Example images from the video sequences. (Cohen, 2000, pp. 8-30).....	36

Figure 3.4 Result of PFA method. Arrows shows the principal features chosen.....	37
Figure 3.5 Algorithmic representation of ISFER Workbrench .....	38
Figure 3.6 Chain code eye brow (Pantic, & Rothkrantz, 2000, pp.881-905).....	38
Figure 3.7 Snake eye (Pantic, & Rothkrantz, 2000, pp.881-905) .....	38
Figure 3.8 Facial points of the frontal-view (Pantic, & Rothkrantz, 2000, pp.881-905 ).....	39
Figure 3.9 Morphs from happiness to disgust. ....	40
Figure 3.10 Systematic structure of the Neural Network Architecture (Franco, & Treves).....	41
Figure 3.11 Rectangular area at the rightmost figure used as input to NN (Franco, & Treves).....	41
Figure 3.12 Maximum Likelihood Classifier for emotion specific HMM case (Cohen, 2000, pp. 8-30) .....	42
Figure 3.13 Detection of surprise emotion .....	43
Figure 4.1 Conceptual Model for Content Based Video Indexing and Retrieval (Zhong et al, 1996) .....	46
Figure 4.2 Video hierarchy .....	47
Figure 4.3 Basic camera operations.....	48
Figure 4.4 Flow fields during camera motion .....	54
Figure 4.5 Average Pixel Method .....	57
Figure 4.6 Hierarchy in Mpeg Standards .....	59
Figure 4.7 An abstract representation of possible applications using MPEG-7.....	62
Figure 4.8 Real-Time Video Indexing & Retrieval Basic System Architecture .....	65
Figure 4.9 Real –Time Video Indexing Conceptual Schema .....	66
Figure 4.10 Real-Time Processing .....	67
Figure 4.11 Real-Time Processing (Detailed).....	67
Figure 4.12 Object Motion-based Shot detection algorithm .....	69
Figure 4.13 Location names .....	70
Figure 4.14 Estimating Face Locations .....	71
Figure 4.15 Trajectory-based query generation.....	72
Figure 4.16 Result of finding trajectory locations.....	73
Figure 5.1 Experimental Results of Face Detection on single face.....	79

Figure 5.2 Results of Face Detection on single face (in Percentages) .....	80
Figure 5.3 Experimental Results of Face Detection on multiple faces .....	81
Figure 5.4 Results of Face Detection on multiple faces (in Percentages) .....	82
Figure 5.5 Initial Screen of Histogram-based shot detection preliminary work .....	84
Figure 5.6 Result of Histogram Based Shot Detection Study. ....	85
Figure 5.7 Example shot detection screen .....	86
Figure 5.9 Design Time Components.....	87
Figure 5.10 Initial Screen of the Application .....	88
Figure 5.11 Starting video capture process .....	88
Figure 5.12 Selecting Video Format .....	89
Figure 5.13 Selecting Video Source.....	90
Figure 5.14 Indexing of Live Video feeds.....	90
Figure 5.15 Selecting a video index (vix) file .....	91
Figure 5.16 Indexing of live video feeds (screenshot) .....	91
Figure 5.17 Enabling boundary regions .....	92
Figure 5.18 Setting up frame rate .....	92
Figure 5.19 Setting up frame rate of video source .....	93
Figure 5.20 Offline Video Indexing (Open File).....	93
Figure 5.21 File open dialog box.....	94
Figure 5.22 Video retrieval typical structures .....	95
Figure 5.23 Video Retrieval form in design time.....	97
Figure 5.24 Execute a Query .....	97
Figure 5.25 Initial Screen of Video Retrieval form.....	98
Figure 5.26 Selecting a video index (vix) file .....	98
Figure 5.27 Text-based video retrieval.....	99
Figure 5.28 Giving mouth state.....	99
Figure 5.29 Executing a text-based query .....	100
Figure 5.30 Result of trajectory query.....	101

---

## CHAPTER ONE

# INTRODUCTION

---

Today, there is enormous growth in size of the multimedia databases, especially in video databases. Nevertheless, with increasing availability of multimedia database, difficulties of indexing are also emerging. In order to overcome this problem, new metadata standards Dublin Core and MPEG-7 are defined. Because of increasing availability of such data, many users want to query these multimedia data in an efficient way. However, before querying multimedia data, it must be defined in a structured way and must be ready for querying.

When we look at the previous work on video indexing, best candidates for the queries are actors, actions and events. By considering this, we decided to study on face detection, facial feature detection, summarization, indexing and retrieval of video.

Face detection from still and video images are an active research area with a number of commercial applications. It is the first step for face recognition systems. When it is used with face recognition systems, it can be used for security purposes including authorization for access, determining persons in criminal databases, or even changing keyboard-given instructions to more natural models of interaction. In order to develop a robust system, the system must have ability to detect and recognize human faces with varying;

- Backgrounds
- Rotation
- Races
- Lighting conditions

- Sex types

For the point of Human Computer Interaction (HCI), emotional computing is building systems that can recognize the emotions of humans. If the computer is able to detect the human's emotional state, which can be happy, angry, surprised, disgusted, feared or sad, then it can make interaction with the human according to the detected emotional expression or the system can be improved to integrate itself with current pulse based lie detectors.

### **1.1 Aim of Thesis**

In this thesis, a system developed in Borland C++ Builder 5.0 for automatically detecting, extracting, indexing and retrieval of human facial information with skin color based face detection method. The system can operate with AVI and MPEG files as well as real time live video feeds from a video camera. The system runs on a Windows XP Professional machine having Intel Pentium III 733Mhz cpu, 384MB of RAM and 20GB of harddisk. It uses skin based face detection method in order to extract facial information from still images extracted from either video frames or a single image. The software also provides querying the indexed database with either summarized text or by user defined trajectories.

User can define a trajectory in order to get a time-sequenced list of images satisfying the trajectory. Human face has to satisfy all of the trajectory constraints to get a result. These constraints are defined by the user and include all sequential paths. Start location of the trajectory have to be in boundaries of the face.

A preliminary work for face detection has been done by using C++ on a Red Hat Linux machine. It simply operates with JPEG formatted files and detects possible face candidates according to the skin color then, finds facial features for each of the face candidate. At last redraws the resulting processed JPEG formatted image. This work can be accessed via <http://pamir.cs.deu.edu.tr/~taner/face/Upload.exe>

## 1.2 Thesis Organization

Each chapter first gives a literature survey on chapter's topic then concludes with a simple system developed on that subject. The thesis is organized as follows:

Chapter Two describes different face detection methods including Neural Networks and Hidden Markov Models (HMM), their pros and cons. Chapter Three is about facial expressions and it gives examples for universal emotional states and Action Units (AUs) defined in FACS system (Ekman, Friesen, 1978). It also gives examples from previous works done on facial expression recognition. Chapter Four explains basics of Video Indexing and current state of art in video indexing. It also describes Dublin Core, DC types, MPEG-7 standard. Chapter Five explains all implementations, their explanations, developed system, its architecture, interface pros and cons of the system in detail and experimental results obtained from the developed systems are mentioned. Finally, Chapter Six concludes the thesis with interpretations about the work done and presents possible future works for this topic.



---

## CHAPTER TWO

# FACE DETECTION

---

This chapter describes a literature survey on detecting faces in images with various backgrounds. Neural Network and Hidden Markov Model based approaches are explained and finally, information about a skin color based face detection study is given.

### 2.1 Introduction

One of the main problems in the content of computer vision is object detection. Detection of human faces is the critical part on the road of image indexing and retrieval as well as video indexing and retrieval applications. Most of the applications are based on multi or single-level Neural Network based applications requiring supervised or unsupervised learning from a set of predetermined grey scale faces and non-faces. Current researches focus on finding a fast and robust detection algorithm to decrease the computational time cost and false results.

Human face is a dynamic object having many action capabilities thus many different views available for the same person. Many conditions affect the detection rate of human faces within a group of images including different lighting conditions, poses, facial expressions and also glasses or different race and peoples having different beards. A comprehensive face detection algorithm should consider and take care of all the requirements in an efficient way.

## 2.2 Object Detection

Object detection has an important place in the area of Computer Vision researches. Object detection is the process of searching a class of object within an image. As it is base of most of the other detection and recognition systems, object detection is an important process. Earlier errors in this step make difficulties on further processes.

Main differences of objects in images can be classified as follows;

- **Variation of Effects:** Face candidates can be located in images with effects such as rotation, scale, translation or mirroring or even different degree of brightness or contrast. Figure 2.1 shows variation effects. The upper left side of the image we can see that light effect completely change the half of the persons face.



Figure 2.1 Faces with different effects (rotation, brightness, background, etc.)

- **Variation Poses:** People usually look at the camera during taking the photo but in many cases, human face is not seen with frontal upright position in image. Rotation and translation effects make variant poses, i.e., with variant poses, some of the facial features disappear from the image. Different methods should be considered to detect this type of objects.

### 2.3 Face Detection with Static Background

It is the easiest way to find face candidates within images. In this technique, background has always a static color. Only thing to do is removing this disused background information from raw image using simple image processing techniques mostly known as “blue screening” and then performing other types of detection methods in order to find face candidates within this image.



Figure 2.2 Static Background in blue color

When the background is removed from the source image, the processed image has only two type of information, namely called face and non-face area.

### 2.4 Face Detection Using Skin Color

In past, skin color is mostly used for tracking faces. It is clear that, input images must be in color not gray scale as most of the other methods to perform skin color based face detection. For the gray scale images, the human face color information is usually found in background information. Therefore, use of gray scale images is not adequate for detecting human skin color. Human skin color can be used to detect face regions within the source image. It is easy to use but is not robust under different lighting conditions and human skin color naturally differs from each other even

under the same lighting conditions. This is the main problem called color constancy in color vision.

Skin color, itself usually is not adequate for face detection. It must be combined with other face detection techniques to increase the robustness. For example, it can be used to reduce the search space for neural network based approach. In this case, neural network filters can search only those parts of the image. This reduces the total computational time necessary to find human faces.

The most important advantage of using skin color model is its orientation invariant property and being one of the faster methods to detect human faces. According to Yang & Waibel, because of the fast processing, this method can be used in real-time systems (Yang & Waibel, 1996, pp.142-147).

It has some disadvantages. If other part of human body is appearing in the input image like arms, hands where these part can easily makes confusion and thus decreases the robustness.

Figure 2.3 shows the automatic background removing of an image with static background.



**Figure 2.3 Background Removing & Segmentation**

## 2.5 Face Detection by Motion Analysis

By using face and non-face classification, it is possible to detect human faces via principal motion analysis techniques. Hjelmås et al has used a motion detection system. In their work, they performed four steps (Hjelmås et al, 1998). These are as follows;

- Frame Differencing
- Thresholding
- Noise removal
- Computing total number of pixels found in each line of the motion image.

They have compared the current frame with the previous frame in video sequence.

If the difference between the pixel values is greater than value of  $\frac{\text{colors used}}{10}$ , then

pixels having this property are painted with black color. If the difference in pixel values is less than the threshold value, then those pixels are painted with white color.

Figure 2.4 show an example motion image

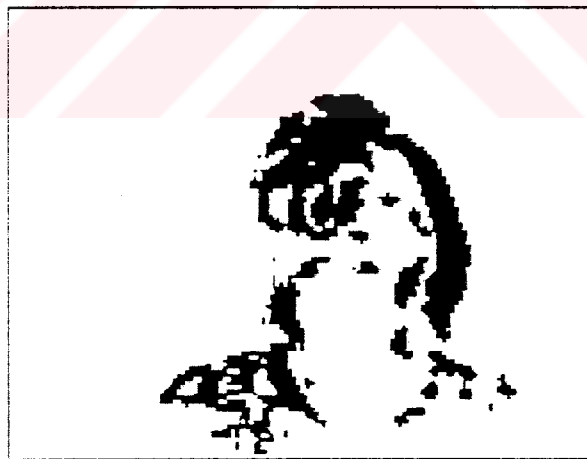
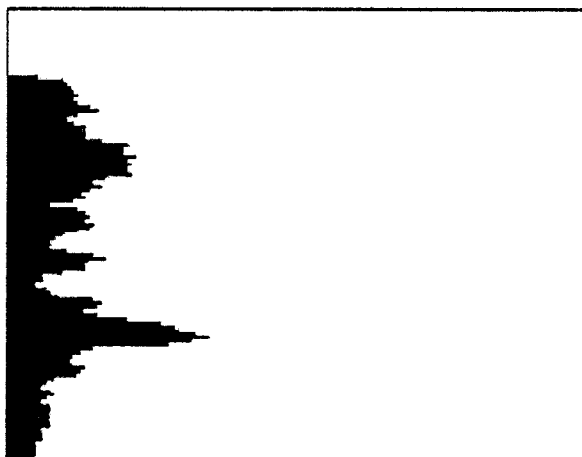


Figure 2.4 Typical motion image (Hjelmås et al, 1998)

They have used a 3×3 frame to remove unnecessarily information. If there are black pixels isolated by white pixels then these pixels are eliminated by using the 3×3 frame. After that, they have calculated total number of black pixels in each line

of the processed image and made some assumptions such as minimum size of objects, etc. Figure 2.5 shows the resulting image after the pixel count operation.



**Figure 2.5 Amount of pixels on each line in the motion image (Hjelmås et al, 1998)**

They have used the following heuristics in order to find human face.

- If there is a large moving object in the image, there may be a human present
- If the movement in the upper part of the moving object is larger than a threshold, this may be the top of the human, thus it is a face

After that, face is computed by using principal components analysis of the Olivetti Research Laboratory (ORL) database having 400 human faces. Figure 2.6 shows pictures from this database.

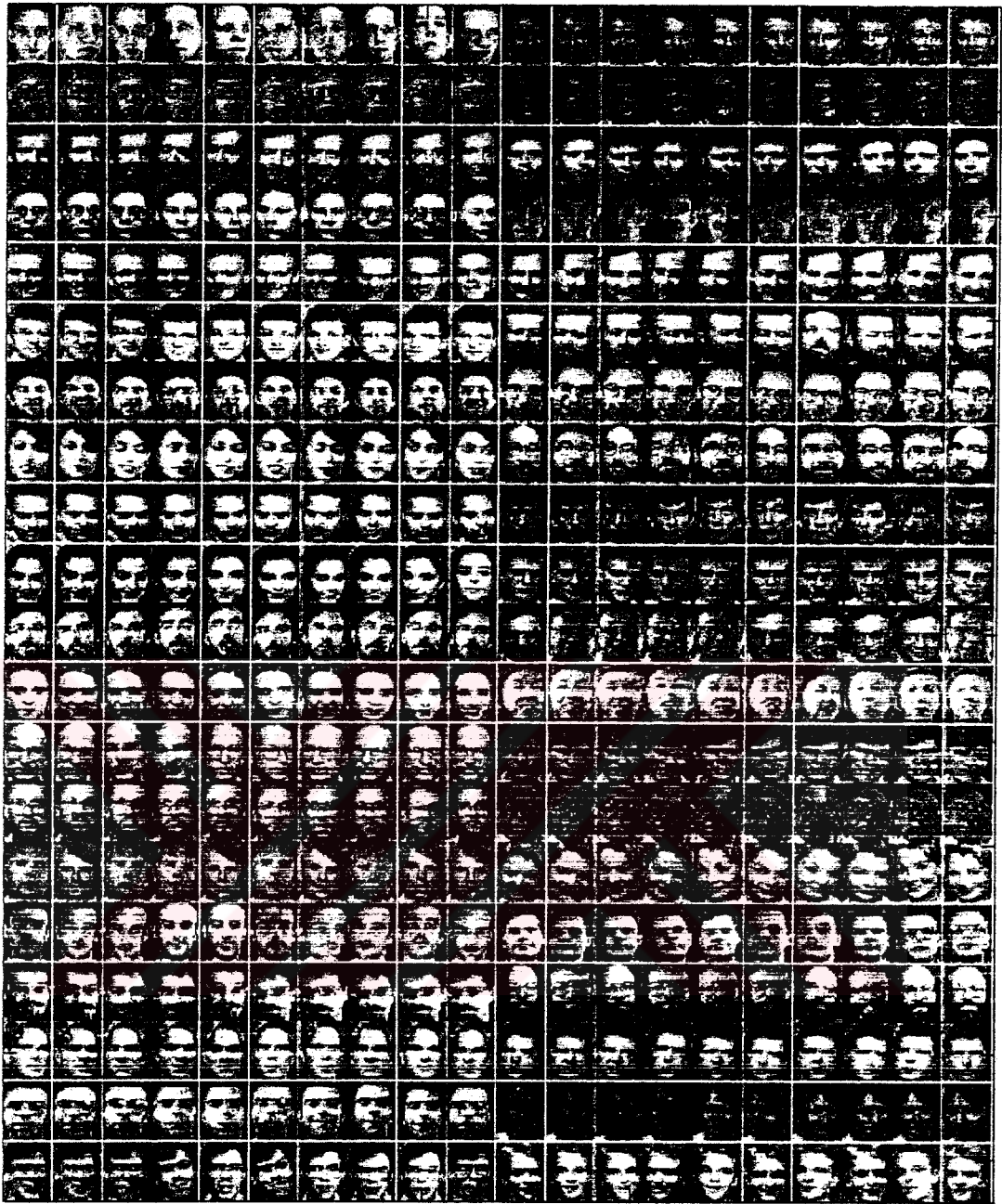


Figure 2.6 The Olivetti Research Laboratory (ORL) face database

## 2.6 Feature Based Methods

The first step in this method is to detect all facial features and collect or create representative features. These features may include relative distances of their centroids, their geometric structure, etc. Euclidian distance is usually used to

compute minimum distance of these features to recognize a face. As a result face candidates with the minimum total distance value is selected as a face.

## 2.7 Neural Network-Based Face Detection

Today, neural network-based face detection model is the most popular face detection technique. Many researchers in the area of face detection have been studying on single, multi level or artificial neural networks in their researches. However, it is difficult to compare the efficiency of each of the technique with each other because of different training conditions for both face and non-face learning. A general neural network-based face detection steps are as follows;

- Initially, a set of faces and non-faces for training is picked up.
- Each face is scaled to  $N \times N$  size of sub patterns.
- In order to reduce background effects, a mask is applied to each of the patterns.
- All these classes are learned using a training method.
- The system is ready to classify face and non-face regions.

Rowley et al presented a Neural Network-based face detection system (Rowley et al, 1996). They have used bootstrap algorithm for training. However, this training is used to add false detections of face, thus decreases time required to manually select non-face candidates from a large set of images and reduces size of the training set. Their system has two steps: The first step consists of a neural-network based filter. The system receives a small region of the input image and produces output ranging from 1 to -1. These values represent the presence or absence of the face within the input image. They have used face databases of CMU and Harvard Universities. The second step combines the results comes from networks in first stage. They have tried both ANDing and ORing of results, meaning that if the result appears in both of the networks then it has precedence in case of ANDing. Figure 2.7 shows an example image from their work.



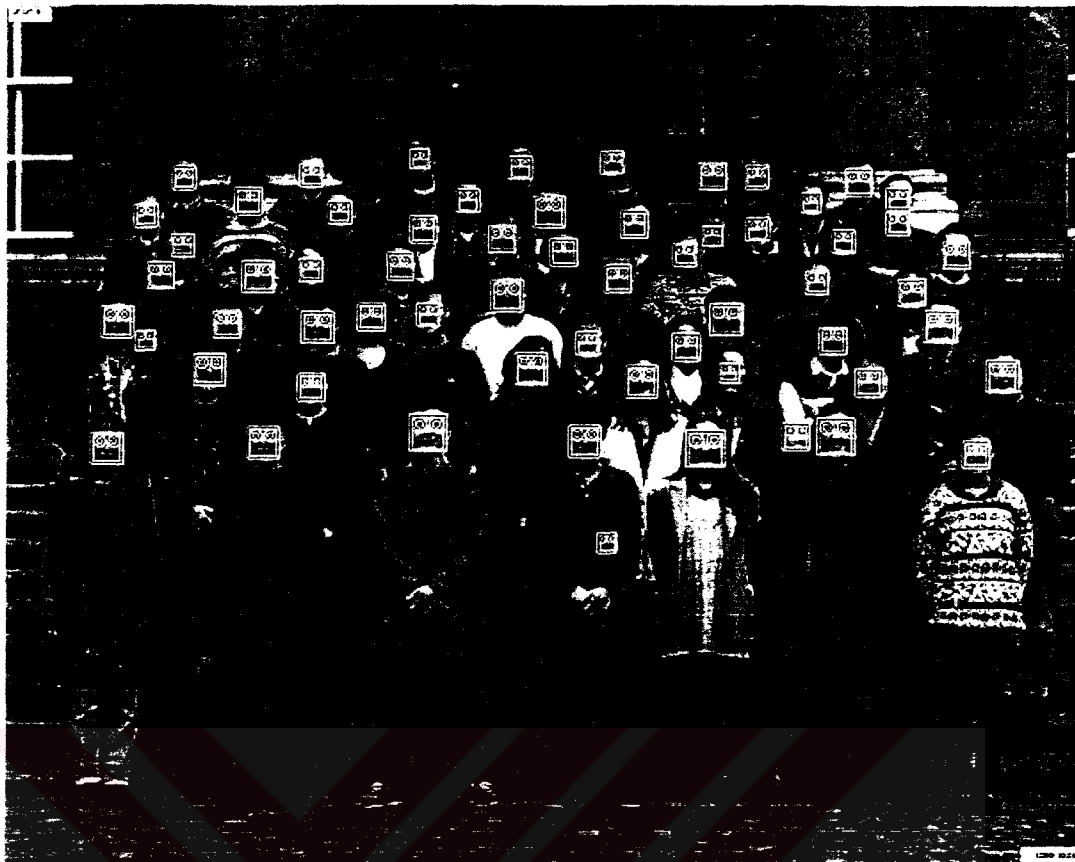


Figure 2.7 Example image from the work (Rowley, 1999)

Their algorithm detect up to 92.9% of faces in a set of test images.

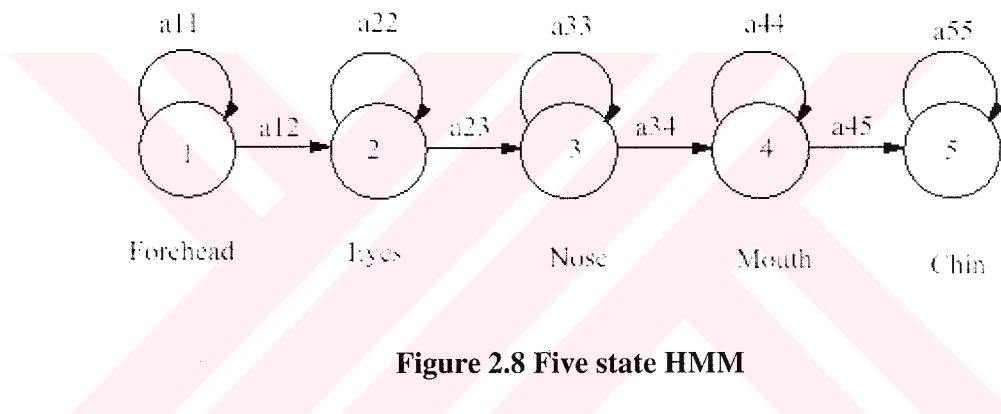
Han et al presented a technique for fast face detection via morphology-based pre processing (Han et al, 1997). Their work consists of three main steps;

- In the first step, a morphology-based technique is used to perform eye-analog segmentation. In this step, morphological operators applied to the original image to get eye-analog segmentation.
- In the second step, previously determined eye-analog segments are used to search for a potential face region.
- The last step is used for verification, each image is normalized in size and then each of the images is given to a backpropagation neural network as an input for identification.

## 2.8 Hidden Markov Models (HMM)

Hidden Markov Models are a set of statistical flexible model-based approach to detect frontal upright faces. This model is used in face recognition, speech recognition and video indexing systems where the data is usually one-dimensional. Each frontal upright face has facial features such as two eyes, one nose and one mouth and there is a natural order of locations of these features on the human face even if the head is rotated in a significant degree. Unlike other methods, in this method, feature vectors are used to recognize face not to detect face.

A Hidden Markov model has a Markov Chain having a number of hidden states, an initial state distribution and a state transition probability matrix. Each states generates observations according to some probability distribution.



**Figure 2.8 Five state HMM**

Hidden Markov Models has a set of  $N$  states,  $S = \{S_1, S_2, \dots, S_N\}$ , with the state at time is represented as  $q_t \in S$ . Initial state distribution  $\Pi = \{\pi_i\}$  is represented by

$$\pi_i = P[q_1 = S_i], 1 \leq i \leq N$$

The state transition probability matrix is  $A = \{a_{ij}\}$  and it satisfies;

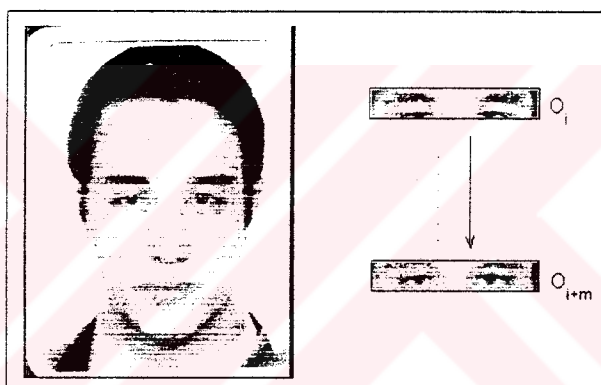
$$a_{ij} = P[q_t = S_j | q_{t-1} = S_i], 1 \leq i, j \leq N, 0 \leq a_{ij} \leq 1, \sum_{j=1}^N a_{ij} = 1, 1 \leq i \leq N$$

The probability distribution matrix for the observations is  $B = \{b_j(O_t)\}$  where

$$b_j(O_t) = P(O_t | q_t = S_j)$$

Samaria, & Fallside used one dimensional HMM to detect frontal upright human faces (Samaria, & Fallside, 1993). They assumed that faces are in frontal upright position and facial features occur in a sequential order thus they have used top-to-bottom model.

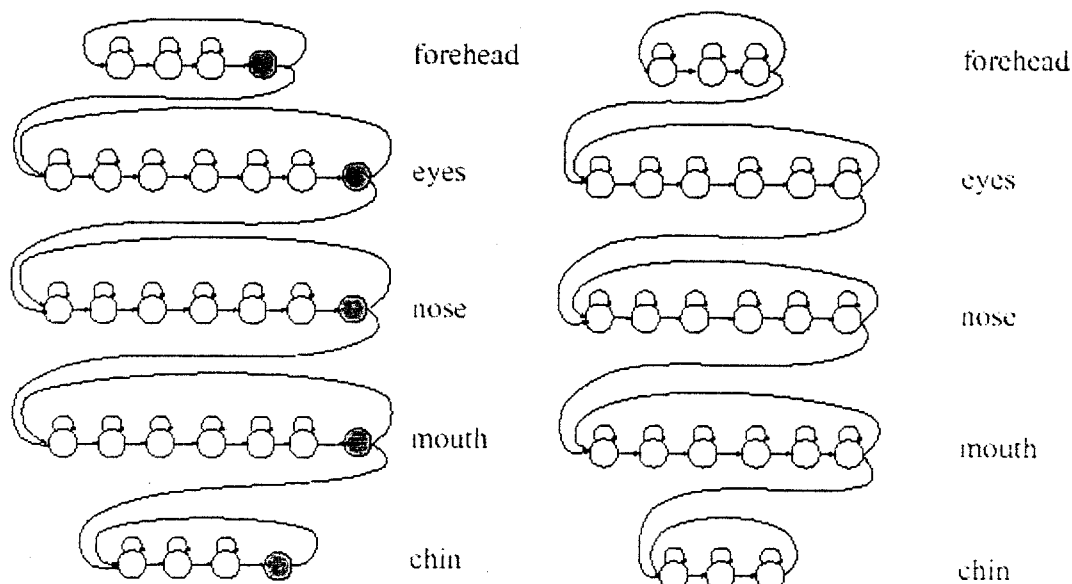
Samaria, & Harter generates an observation sequence  $O$  from  $X \times Y$  image using an  $X \times L$  sampling window with  $X \times M$  pixels overlap area (Samaria, & Harter, 1994, pp.138-142). In their method, each observation vector has  $L$  lines and there is an  $M$  line overlap between successive observations. Figure 2.9 shows how observation vectors are created.



**Figure 2.9 Sliding windows for generating observation vector**

To detect each facial feature, Samaria has increased the number of states in each level and used a sliding window that moves from left to right and top to bottom to get pixel intensities as observation vectors (Samaria, 1994). Figure 2.10 shows the horizontal Hidden Markov Model.

In Figure 2.10, process starts at forehead layer and continues executing in horizontal line after reaching the end it jumps to the next (bottom) level (eyes). Process continues until reaching the bottom level (chin).



**Figure 2.10 Horizontal Hidden Markov Model (Samaria, 1994)**

Face detection rate of the above work is about 95%. However, when moving the sliding window from left to right and top-to-bottom there exists many observation vectors. The system operates very slowly because of this huge number of observation vectors. (The system requires four minutes to recognize a face on a sparc 20 machine)

As a result, Hidden Markov Model-based approach represents a systematic method for both face detection and face recognition. However, these methods cannot be used in real time systems because of the high computational time required to process images.

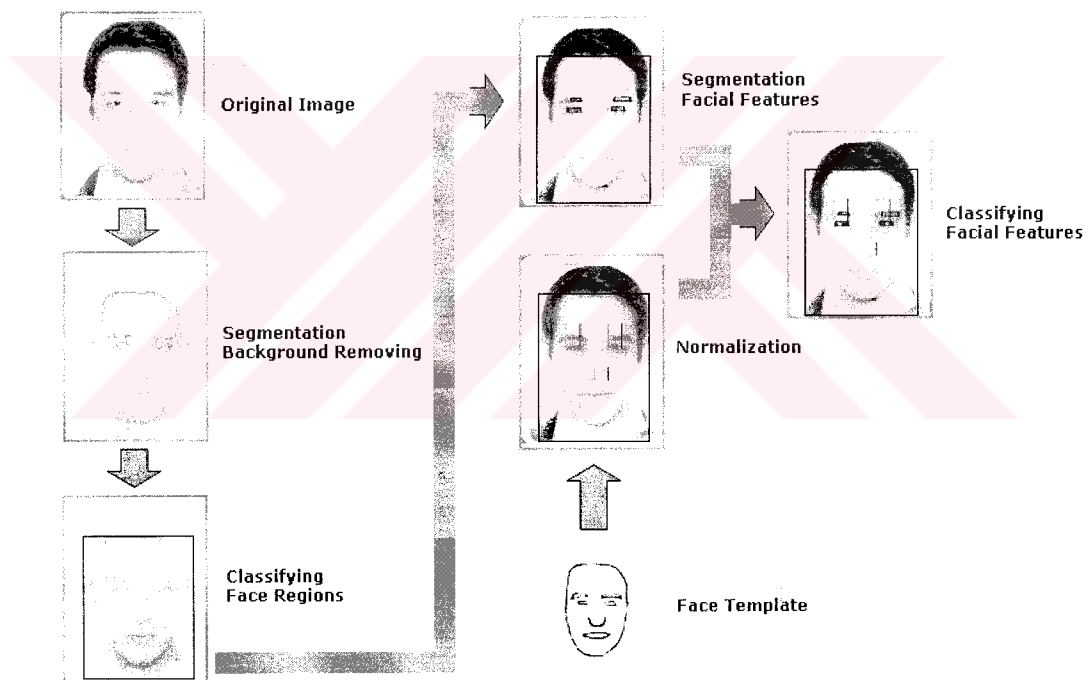
## 2.9 Face Detection Study

In this study, skin color-based frontal upright face detection method is used to find human face candidates within an image. The most significant facial feature for a frontal upright face is eyes, eyebrows and mouth. This technique is one of the fastest methods in the content of face detection but is not as robust as neural network based techniques and it should be used with other models to increase performance. Skin color based face detection has the advantage that color information is invariant from

variation effects such as rotation, mirroring, translation, etc. However, it can easily change under different lighting conditions. Another point is that different peoples from different races have different average face color.

After manually testing human faces, we realized that for European and Asian race, there is a relation between the R, G and B color channels. Average R-values are greater than G and all G values are greater than B values. Thus, our background removal algorithm just looks at these values and classifies the color page into two classes. Face regions and non-face regions.

Figure 2.11 shows step by step face detection algorithm used in this work. It has four steps. Almost all of the steps have their own heuristics.



**Figure 2.11 Step by step Face Detection**

### **Step 1: Background Removing (Segmentation)**

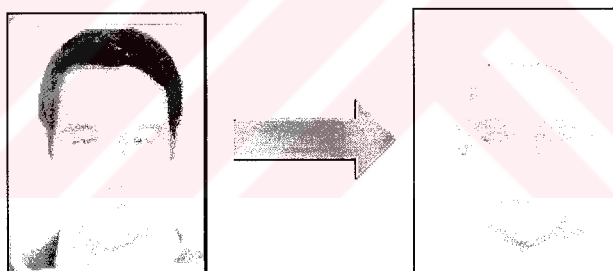
In the first step, input image pixels are segmented as face and non-face pixels. This is done by removing the background information from the image by applying the

following fast code segment; When the R, G, B value of the current pixel does not satisfies the condition then, all color channels (R, G, B) is set to value 255 (white). Figure 2.12 and Figure 2.13 shows visual examples of the algorithm result.

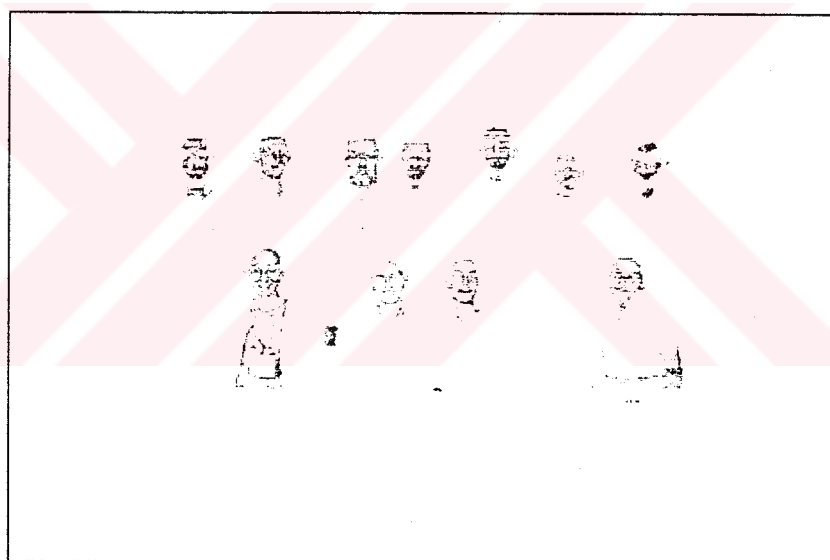
*For each pixel in  $N \times M$  size of image do*  
*If (not( $R > G > B$ ))) then mark this point with white color*  
*Continue processing of image until reaching to the end*

For static background, it is easy to remove background information from the source image and it does not need to apply morphological operators. Nevertheless, if the image has a dynamic background then number of colors increases enormously. This situation makes removing process difficult.

When we look at Figure 2.12, we see that in the processed image some facial features are also removed. This is what we want because the average color of human face eyebrow, eye or mouth differs from the average human face color. Figure 2.13 shows dynamic background removing operation.



**Figure 2.12 Removing static background**



**Figure 2.13 Removing dynamic background**

This step does not deal with either classifying face candidates or geometry such as minimum size of face candidates. To simplify source image, it just removes unnecessary background information.

Result of this step is a JPEG formatted image that has only white and skin color regions. Background removing process feeds further steps; therefore, accuracy in removing of background information is very important.

During the development phase of this step, following difficulties encountered;

- Face detection rate can easily be affected by various lighting conditions. Figure 2.14 shows an example. A modified region growing algorithms can be used to reduce the lighting effect. For example, if the eight neighboring pixel values do not exceed a predetermined or adaptable threshold value then these parts are not removed from the face. Of course, this work should be done in the first step.



**Figure 2.14 Light effects on face**

- RGB color space is not linear therefore restricting the R, G and B values with a range of values is usually difficult. One pass over image is usually not enough to remove background information.
- According to our test results, it is difficult to remove closed lip region because of its color similarity to human face. However, an open mouth is the easiest facial feature to extract.



## Step 2: Finding Face Candidates & Classifying

This step aims to search and classify the possible face candidates. A face is a skin colored closed region that is isolated by white pixels or the boundaries of the image. This definition makes it easy to find boundaries of face regions.

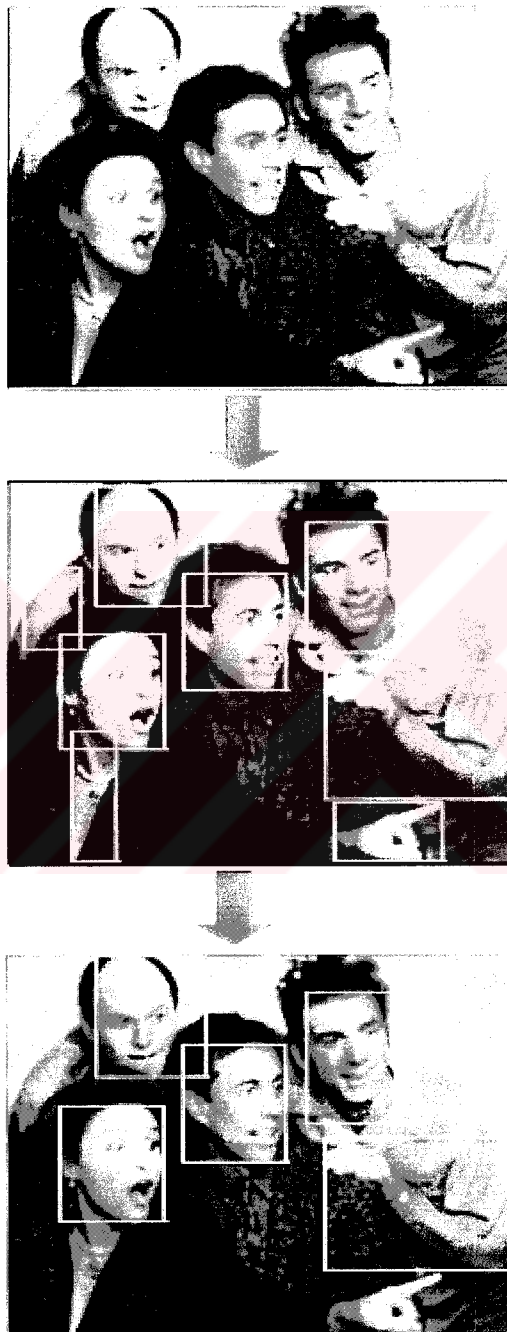
Classification process is simple and a modified version of a recursive Boundary fill algorithm is used to find closed regions. Closed region is considered as a specific colored region that is isolated with either white or skin colored pixels. During the execution of boundary fill algorithm, a second buffer is needed to keep track of visited pixels. If the current pixel is not visited before and it is a skin color then boundary fill function starts to make recursive calls with looking up its eight neighboring pixels in counter clockwise direction. It continues to make recursive calls until it reaches to;

- Image boundary
- White colored pixel (marked at Step 1)

For each of recursive calls if the current pixel satisfies skin color range then, size of the area of the current face region is increased by one. The size of area is then used in further steps. In addition, each face candidate has its minimum  $X$  and  $Y$ , values representing the upper left corner of the face region (according to our view angle) and maximum  $X$  and  $Y$  value representing maximum and minimum values in  $XY$  coordinate system. If it is necessary then these minimum and maximum values are updated.

Each main call of the boundary fill function either returns a face region or not. A face is returned if any of the initial minimum and maximum values have been changed. Before storing each of the face result in an array of faces, some clean-up heuristics applied.

The first clean-up heuristic eliminates relatively small face regions. If the size of the area is less than 900 pixels or any dimension of the face region is less than 30 pixels then this area is ignored. Figure 2.15 shows the effect of using this heuristic.

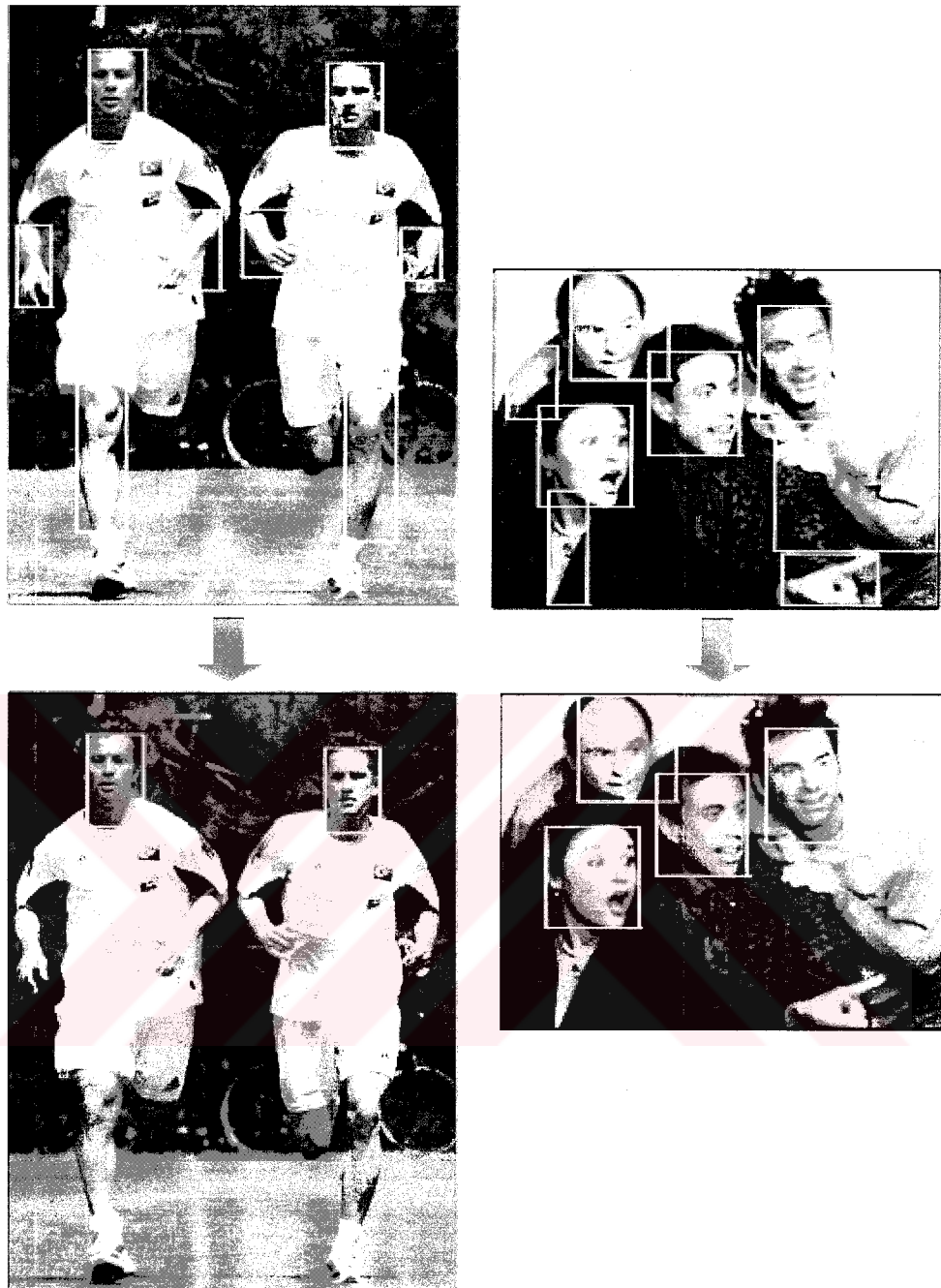


**Figure 2.15** Eliminating small sized non-face regions

Using the first heuristic allows us to remove false drops from the face index. When we look at to the Figure 2.15, two hands and one arm is removed by using this heuristic. However, there are still false drops in the Figure 2.15, thus we need some other heuristics.

The second clean-up heuristic is about the geometry of the face region. If the width or height of the face region exceeds 1.25 times of the height or width of the face region then this area is corrected and either width or height of the face area is recomputed or the whole face candidate is discarded. Figure 2.16 shows the effects of combining this heuristic (without re-computing, using discarding) with the first one. Combining area-based and geometry-based clean-up heuristics slightly improves the performance of the whole system.

For the Figure 2.16, without using any heuristic there are 80 candidate faces exist in rightmost image. Most of them are so small and cannot be seen in the figure. After applying the heuristics, it is reduced to number four, which is the exact number of faces exist in image..



**Figure 2.16** Eliminating both thin and thick geometric regions

Another heuristic is that if the currently detected face region entirely overlaps with another face region then small region can be discarded. Because, in general

there are no more than two or more human face entirely overlaps each other in images.

The last one depends on the face detection strategy. If the user wants to find only one face within the image, then the face candidate that has the maximum area can be selected for face detection.

Result of the second step is an array of faces candidates each having its minimum and maximum upper left corner and down right corner values. These values will then used to search for facial features. Of course, at the end of this step a jpeg-formatted image is created. In general, classifying is a difficult task because of the complex features of the human face. However using skin color based approach makes it easy to classify face and non-face regions from the source image. Since, skin color just gives global information about the region and does not support verification step. Neural networks are able to detect and verify the correctness of the results.

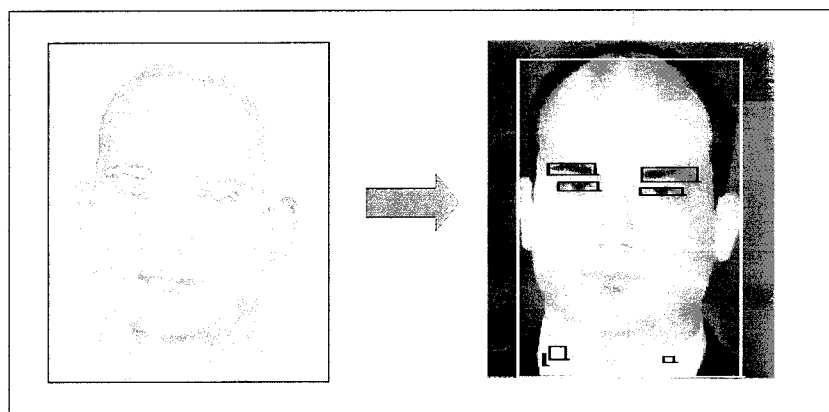
During the development phase of this step, following problems are encountered;

- Recursive boundary fill algorithm requires huge amount of stack size available to the application.
- In some cases, used heuristics does not work and rarely do they remove actual face regions. Therefore, they can just remove the face candidates that are slightly different from the acceptable face template.
- Some false drops can be prevented by adapting the algorithm in the step one to the domain. However, this is a supervised solution.

### **Step 3: Finding Facial Features**

Input for the third step is the segmented image file from the first step and array of face candidates obtained form the second step. In this step, the same algorithm with the previous step is used to detect closed regions. In this case, facial features are closed regions filling up with all white color and is isolated by skin color. A parameter called closed region type is added to detect either face candidate or facial

feature candidate to use the same function with the previous step. For the point of closed region detection algorithm, the only difference between the two steps is the value of return value. Figure 2.17 shows the result of facial feature detection.



**Figure 2.17 Facial feature detection**

Example facial features are two eyes, two eyebrows, noise and mouth. Average RGB colors of these regions are different from the skin face therefore; they are previously segmented in the first step.

The algorithm looks for closed regions and stores the minimum and maximum point values of upper left and down right corners of the closed region. In each iteration boundaries of a new face candidate is searched to find closed regions. When the search process ends, possible facial features are computed according to the size of the current face candidates. This method allows us to label closed regions.

After a number of tests, open mouth found to be the easiest facial feature to detect. During this step following problems occurred;

- In some cases, there are many small region detected. These are part of background and not actual facial features. However, some of the small regions are really close to the original facial locations in the template and if they are not removed then they make problems in facial feature classifying step. To correct this problem, size of total white pixel area of the facial

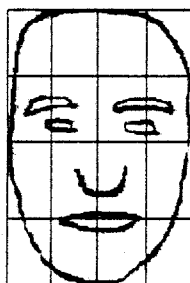
feature is considered. If any dimension of the region is less than 5 pixels or its area size is less than 25 pixels then they are ignored.

- Sometimes the first step is not able to remove close facial features such as eyes and eyebrows. It is usually difficult to consider them as part of background and remove them in the first step. Therefore in some cases both eyes and eyebrows are grouped into one facial region either eye or eyebrow. Because of this, there are false drops in determining close facial regions in the classification step.

#### **Step 4: Combining results**

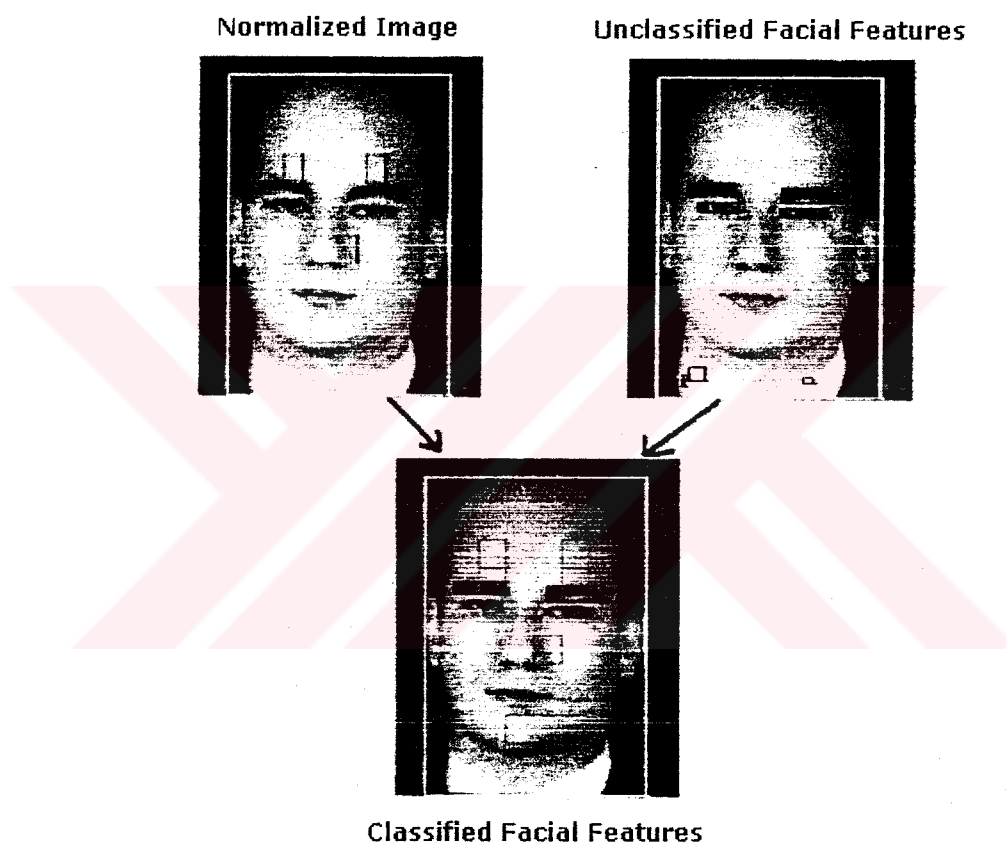
The previous step finds facial regions. However, without classifying them, we are not able to determine which regions represent eye or mouth, etc. This step classifies facial regions by using a facial template.

Figure 2.18 represent the face template used in this application. The template is divided into 4×4 regions and minimum  $x$  and  $y$  positions of the facial features are computed. By using the face template, we are able match our facial features with the template's facial features. In order to match these different object similarities face region is normalized with the template.



**Figure 2.18 Face template**

Figure 2.19 shows how the two images are used to combine a result image. On the upper left corner, the image is just normalized and approximate locations of the facial feature computed. Green colored rectangles in this image represent the computed approximate locations of facial features. On the upper right corner, there is an image having actual facial features, however in this image there is no classification performed. It means that we do not know which facial feature is left eye, or which one is right eyebrow.



**Figure 2.19 Facial Feature localization**

Each facial feature in face candidates are sorted in descending order according to size of their areas and then the first seven regions are selected to classify facial regions. Other regions are just ignored. The size of area is not the area of the rectangular facial feature. It is actual number of white pixels in this facial region.

Before giving the algorithm let define some functions and objects.



Let  $A$  is the source face and  $T$  is the template face region,  $F_A$  and  $F_T$  are set of facial regions belonging to face region  $A$  and  $T$  respectively.

$F_A(i)$  and  $F_T(j)$  are individual facial regions of the two face regions and their upper left corner coordinates is represented by  $\min(F_A(i)_x)$ ,  $\min(F_A(i)_y)$  and  $\min(F_T(j)_x)$ ,  $\min(F_T(j)_y)$

$d(F_A(i), F_T(j))$  represents the Euclidian distance of the facial feature  $F_A(i)$  to facial feature  $F_T(j)$  and represented by

$$d(F_A(i), F_T(j)) = \sqrt{(\min(F_A(i)_x) - \min(F_T(j)_x))^2 + (\min(F_A(i)_y) - \min(F_T(j)_y))^2}$$

The algorithm is as follows;

Select a facial feature  $F_T(j)$  from the template face  $T$ .

Compare the distance of every facial feature  $F_A(1..n)$  found in the current face  $A$  to this facial feature  $F_T(j)$ .

Classify or match the facial feature  $F_A(i)$  to  $F_T(j)$  that has the minimum distance  $d(F_A(i), F_T(j))$  value.

Continue until all facial regions in the face region  $A$  are classified.

Result of the third step is a set of face candidates and corresponding facial features attached to left eyebrow, right eyebrow, left eye, right eye, and nose and at last mouth. Now we are ready to do everything because we had summarized the content of the raw image.

Dimensions of the facial features considered as a part of distance metric, but other features could also be added in some way to improve efficiency. Classification is difficult task and there are a number of ways to classify facial regions. Most of the solutions are domain specific.

This work can also be done with multiple faces in a single image. The system is developed in a parametric way that the user is able to select maximum number of

persons can be detected by the system. Figure 2.20 shows detection of multiple faces with all eleven faces is detected. Because of using heuristics, in Figure 2.20, at the lower right most location of the image, a girl's arm is discarded.



**Figure 2.20 Detection of multiple faces**

In Figure 2.21, there are wrong face candidates. This is because; the first step in the algorithm did not effectively segment the face and background regions. Variety of lighting conditions affects the segmentation process. To overcome this problem skin color based face detection method should be combined with other approaches like Neural Networks or Hidden Markov Models.



**Figure 2.21 False Drops in detecting multiple faces.**

During the development and phase of this step, following problems encountered;

- Even if the first step identically segments each facial feature, classification process makes the final assignments. If there is a small error in the positions of facial features then the system will lead to a wrong state that causing false drops.
- Classification step can be improved such that, every facial feature can be selected only once. This type of feature assignment allows others a chance to be selected by different facial features.

---

## CHAPTER THREE

# FACIAL EXPRESSIONS

---

This chapter presents a literature survey on facial expression recognition and more focuses on the Facial Action Coding System (FACS) found by Ekman & Friesen (Ekman & Friesen, 1978). Neural Network and Hidden Markov Model Based approaches are also explained. Finally, a small application to detect surprise emotion by checking mouth state is described.

### 3.1 Introduction

Facial expressions have a great role in face-to-face non-verbal human communication. Because, humans do not only communicate with words, they also use body language to support the focus of the subject. According to the Mehrabian, 55% of communicative message is transferred by facial expressions (Mehrabian, 1968, pp. 53-56).

Humans express their feeling by facial emotions. Face-to-face communication has two main aspects namely called auditory (verbal & hearing) and visual (non-verbal). Smallest units of verbal interaction are words in verbal communication while body movements and facial expressions in non-verbal communication.

Until now, touch key-based simple devices like computer keyboard and mouse has been used to satisfy interaction between the human and computer. There are also devices developed for specific purposes like joysticks, etc. However, in all cases, human will adapted to use those devices. With the growing enhancements in human computer interaction styles, as in speech recognition systems, these styles are going

into more natural forms. Now it is time for computers to adapt human behaviors. Understanding emotional state of a person will enhance the abilities of computers.

### 3.2 Background

Duchenne du Boulogne first expresses facial expressions in 1862. He was a pioneering neurophysiologist and photographer. Most researchers acknowledge their debt to Duchenne and his book "The Mechanisms of Human Facial Expression".

Ekman & Friesen presented the most important comprehensive study in the content of facial expression recognition, called Facial Action Coding System (FACS) (Ekman & Friesen, 1978). They have defined a method for describing and measuring facial behaviors and facial movements based on anatomical analysis of facial action.

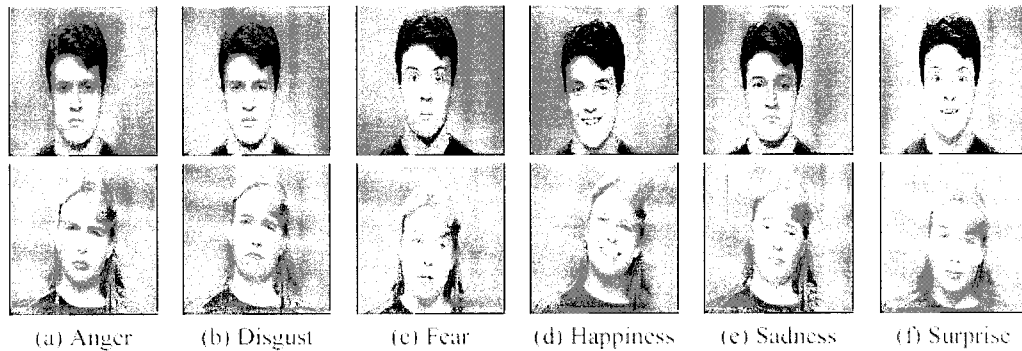
Measurement unit of the FACS system is Action Units (AUs). They have defined a set of 44 Action Units (AUs) in original work that having a unique numeric code, which represents all possible distinguishable facial movements because of change in muscular actions. 30 of them are related to a specific contraction of muscles and 14 of them are unspecified. Table 3.1 shows the set of AUs.

**Table 3.1 List of Action Units**

AU	NAME	AU	NAME
AU1	Inner Brow Raiser	AU31	Jaw Clencher
AU2	Outer Brow Raiser	AU32	Lip Bite
AU4	Brow Lowerer	AU33	Cheek Blow
AU5	Upper Lid Raiser	AU34	Cheek Puff
AU6	Cheek Raiser	AU35	Cheek Suck
AU7	Lid Tightener	AU36	Tongue Bulge
AU8	Lips Toward Each Other	AU37	Lip Wipe
AU9	Nose Wrinkler	AU38	Nostril Dilator
AU10	Upper Lip Raiser	AU39	Nostril Compressor

AU11	Nasolabial Furrow Deepener	AU41	Lip Droop
AU12	Lip Corner Puller	AU42	Slit
AU13	Cheek Puffer	AU43	Eyes Closed
AU14	Dimpler	AU44	Squint
AU15	Lip Corner Depressor	AU45	Blink
AU16	Lower Lip Depressor	AU46	Wink
AU17	Chin Raiser	AU51	Head Turn Left
AU18	Lip Puckerer	AU52	Head Turn Right
AU19	Tongue Show	AU53	Head Up
AU20	Lip Stretcher	AU54	Head Down
AU21	Neck Tightener	AU55	Head Tilt Left
AU22	Lip Funneler	AU56	Head Tilt Right
AU23	Lip Tightener	AU57	Head Forward
AU24	Lip Presser	AU58	Head Back
AU25	Lips Part	AU61	Eyes Turn Left
AU26	Jaw Drop	AU62	Eyes Turn Right
AU27	Mouth Stretch	AU63	Eyes Up
AU28	Lip Suck	AU64	Eyes Down
AU29	Jaw Thrust	AU65	Walleye
AU30	Jaw Side to Side	AU66	Crosseye

Most of the researchers use six basic “universal facial expressions” corresponding to happiness, surprise, sadness, fear, anger and last disgust. Figure 3.1 shows sample set from the (Cohen, 2000, pp. 8-30).



**Figure 3.1 Sample six basic facial expression data set from (Cohen, 2000, pp. 8-30)**

Ekman studied on video tapes in order to find changes in human face when there is an emotion exists. According to the work, a smile exists if the corners of the mouth lift up through movement of a muscle called zygomaticus major, and the eyes crinkle, causing "crow's feet," through contraction of the orbicularis oculi muscle.

Changes in location and shape of the facial features are observed. Score of a facial expression consists of a set of Action Units. Duration and intensity of the facial expression are also used. Observed raw FACS scores should be analyzed in order to produce behavior that is more meaningful. FACS has four main steps;

- Observe movements and then match the AUs with the observed movements.
- An intensity score is given for each one of the actions
- Determine the action units type as asymmetry or joint
- Determining the face and facial feature positions during the movement of the face in the sequence.

Interpreting AU is a difficult task. For example, there are six main emotional states exists but each of them has many variations. Figure 3.2 shows two different type of smile of the same person. Therefore, usually each emotional state is represented by a set of action units. Thus most of the action units are additive.



AU12



AU6+12+25

**Figure 3.2 Two different types of smile. (Lien, 1998)**

One of the limitations of the FACS system is nonexistence of a time element for the action units. Electro Myo Graphy (EMG) studies, which are based on the measurement of electrical activity of muscles, showed that facial expressions occur in a time-aligned sequence beginning with application, continuing with release and finally relaxation.

Face tracking is needed to compute the movements of each facial feature. Terzopoulos, & Water used making up facial features (Terzopoulos, & Water, 1993, pp. 569-579) and Cohen used different face tracking algorithms including 3D-based models (Cohen, 2000, pp. 8-30)

Chen modified the Piecewise Bezier Volume Deformation (PBVD) tracker of Tao & Huang to extract facial feature information (Chen, 2000), (Tao, & Huang, 1998). In this work, the first frame of the image sequence is selected and processed to find facial features such as eye corners and mouth corners. Then generic face model is warped to fit the selected features. Their face model consists of 16 surface patches embedded in Bezier volumes. By using this way, the surface is guaranteed to be continuous and smooth. The shape of the mesh is changed by changing control points in the Bezier volume.



Terzopoulos, & Water developed a model that tracked facial features in order to observe required parameters for a three dimensional wire frame face model (Terzopoulos, & Water, 1993, pp. 569-579). However, their work has a limitation in which the humans facial features should be marked up to robustly track these facial features.

The most difficult task in facial expression detection is tracking and extracting facial features from a set of image sequence. A huge number of parameters and features should be considered (Cohen, 2000, pp. 8-30). Therefore, it is necessary to decrease the number of points that are required to track facial features thus decreasing computational time. Principal Feature Analysis makes the this task and finds the most important feature points needed to be tracked.

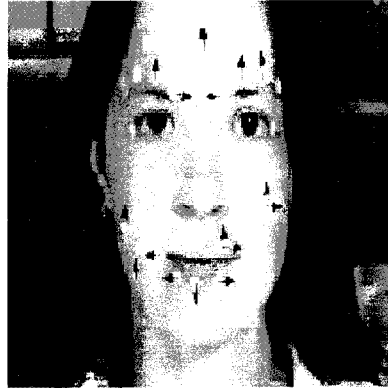
Cohen used principal feature analysis method to find the best facial feature points (Cohen, 2000, pp. 8-30). To do this, Cohen initially marked up the face to be tracked to get robust results and then tracked a video of 60 seconds at 30fps. Figure 3.3 shows example images from the video sequences.



**Figure 3.3 Example images from the video sequences. (Cohen, 2000, pp. 8-30)**

Cohen has used 40 facial points each having two directions, horizontal and vertical to be tracked. For the principal feature analysis, these points divided into two groups, namely upper face (eyes and above) and lower face. And then the correlation matrix is computed. After applying principle feature analysis method, the resulting image showed in Figure 3.4.

In the Figure 3.4, selected feature points are marked by arrows. According to Cohen's work, principle feature analysis method is able to model complex face motions and reduces the complexity of existing algorithms.

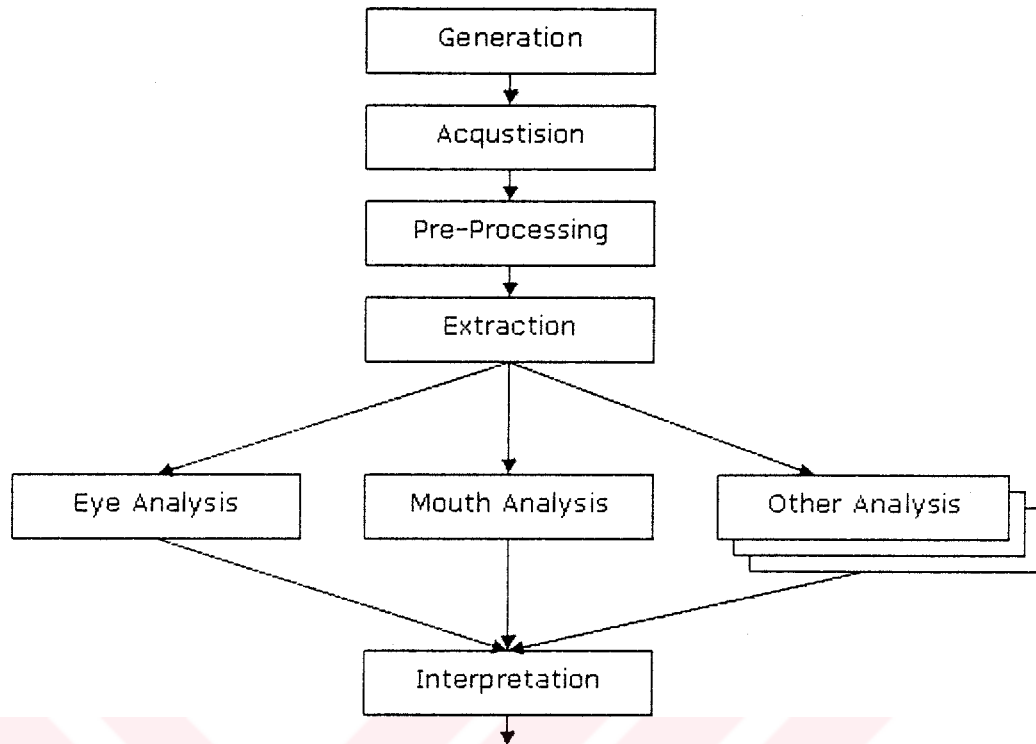


**Figure 3.4 Result of PFA method. Arrows shows the principal features chosen**

In model based recognition systems, a feature vector should be defined for each expression and a similarity metric should be used to compute the difference between these expressions.

Integrated System for Facial Expression Recognition (ISFER) is an expert system for emotional classification of human facial expressions from still full-face images (Pantic, & Rothkrantz, 2000, pp.881-905). The system has two main parts. The first part is called ISFER Workbench, used for feature detection and the latter is an inference engine called HERCULES.

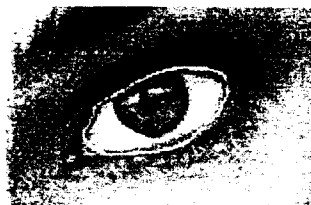
First part of the system, ISFER Workbench presents a system for hybrid facial feature detection. In this part, multiple feature detection techniques are applied in parallel. Therefore, it gives a chance to use redundant parts with eliminating uncertain or missing data. It has several modules, each doing different types of pre-processing, detection and extraction. Figure 3.5 shows algorithmic representation of ISFER Workbench. Figure 3.6 and Figure 3.7 shows example modules.



**Figure 3.5 Algorithmic representation of ISFER Workbench**



**Figure 3.6 Chain code eye brow (Pantic, & Rothkrantz, 2000, pp.881-905)**



**Figure 3.7 Snake eye (Pantic, & Rothkrantz, 2000, pp.881-905)**

The second part of the system, HERCULES, converts low-level face geometry in high-level facial actions.

ISFER is complete automated system that is able to;

- Automatically extract facial features from digitized still images. It does not deal with image sequences.
- Automatic encoding of facial Action Units (Ekman & Friesen, 1978)
- Automatically classifies six basic universal emotional expressions, happiness, anger, surprise, fear, sadness and disgust.

They have used both frontal view and side view of human faces. Figure 3.8 shows the frontal-view template from their work. Details of these points are described in Table 3.2.

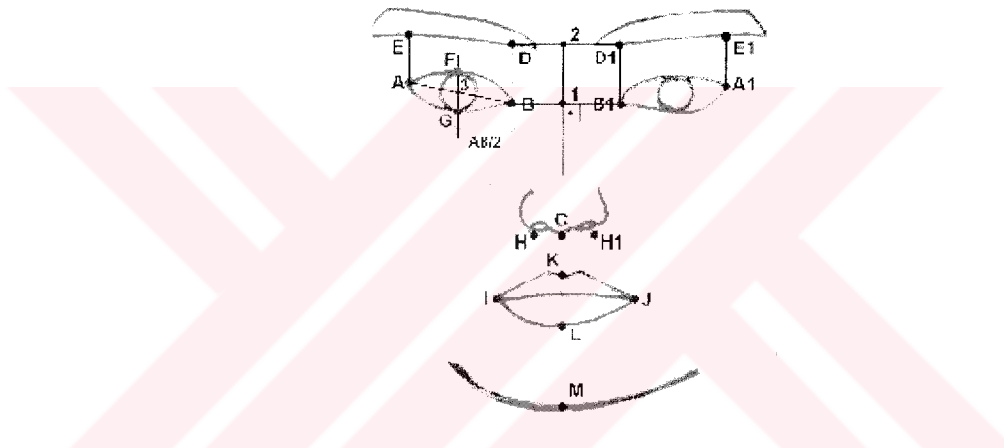


Figure 3.8 Facial points of the frontal-view (Pantic, & Rothkrantz, 2000, pp.881-905 )

Table 3.2 Details of facial points in Figure 3.8

B	Left eye inner corner, stable point
B1	Right eye inner corner, stable point
A	Left eye outer corner, stable point
A1	Right eye outer corner, stable point
H	Left nostril centre, non-stable
H1	Right nostril centre, non-stable
D	Left eyebrow inner corner, non-stable
D1	Right eyebrow inner corner, non-stable
E	Left eyebrow outer corner, non-stable
E1	Right eyebrow outer corner, non-stable
F	Top of the left eye, non-stable

F1	Top of the right eye, non-stable
G	Bottom of the left eye, non-stable
G1	Bottom of the right eye, non-stable
K	Top of the upper lip, non-stable
L	Bottom of the lower lip, non-stable
I	Left corner of the mouth, non-stable
J	Right corner of the mouth, non-stable
M	Tip of the chin, non-stable

### 3.3 Facial Expression using Neural Network-Based Models

Dailey et al showed that a simple biologically neural network model, trained to classify facial expressions matches a variety of psychological data into six universal basic emotions (Dailey et al, 2002). They have considered categorization, similarity, reaction times, discrimination, and recognition difficulty, in both qualitatively and quantitatively. Figure 3.9 shows morphing from happiness to disgust. They have used Morphs software version 2.5.



**Figure 3.9 Morphs from happiness to disgust.**

Franco & Treves inserted a local unsupervised processing stage within a neural network to recognize facial expressions (Franco, & Treves, 1997). They worked with Yale Faces database and their neural net architecture has four layers of neurons. They have success at rate of 84.5% on unseen faces and 83.2% when principal component analysis processing applied at the initial stage.

Figure 3.10 shows systematic structure of the neural network system and Figure 3.11 shows sample images showing four full-face images from their work.

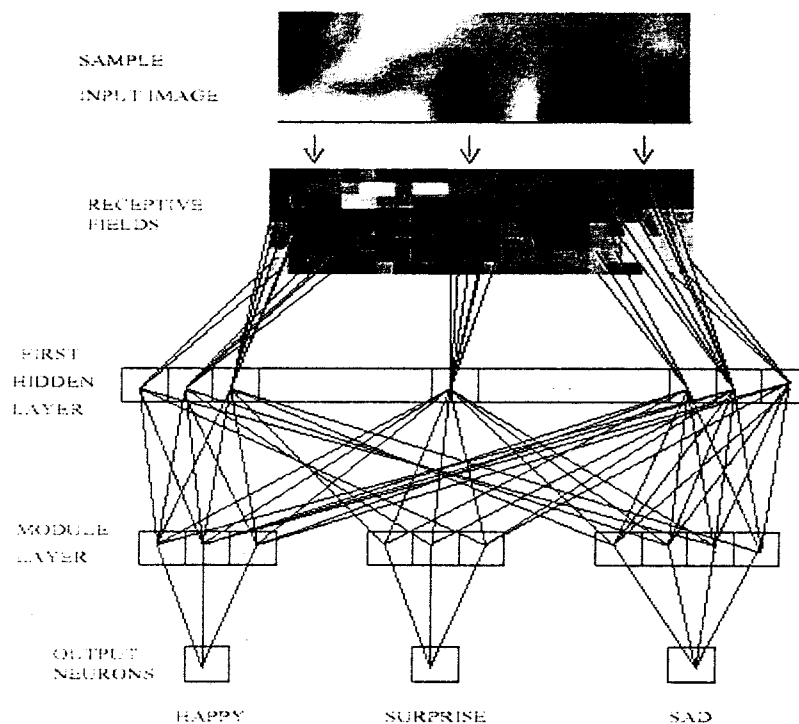


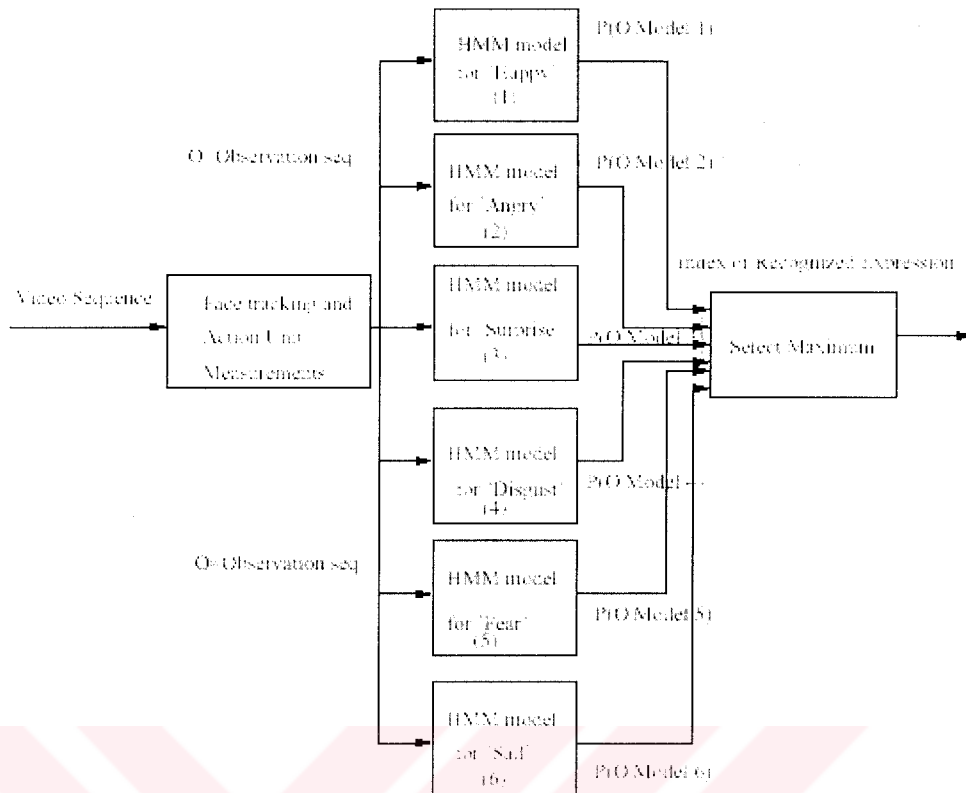
Figure 3.10 Systematic structure of the Neural Network Architecture (Franco, & Treves)



Figure 3.11 Rectangular area at the rightmost figure used as input to NN (Franco, & Treves)

### 3.4 Facial Expression using Hidden Markov Models

Hidden Markov Models usually used to solve classification problems especially for speech recognition systems because of its ability to model or classify non-static events. However, compared with other models, time required to solve the problem is significantly higher. Figure 3.12 shows a maximum likelihood classifier for emotion specific HMM.



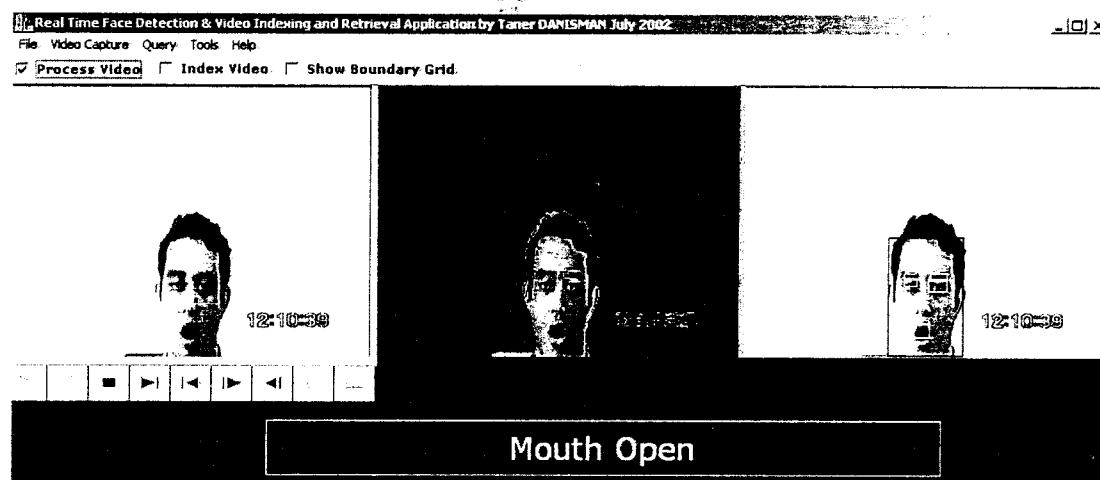
**Figure 3.12 Maximum Likelihood Classifier for emotion specific HMM case (Cohen, 2000, pp. 8-30)**

According to this figure, after making face tracking and Action Unit measurements, each one of the six Hidden Markov Model representing six universal emotions produces a result showing the probability of belonging to a specific type of emotion. At the end of the system, the emotion having the maximum probability is chose as observed emotion.

### 3.5 Facial Expression Recognition Study

There are many ways developed to detect facial expressions. In this study, geometry of facial features is considered to recognize surprise emotional expression. It is a similar approach to distance based methods. However, the study is simple, and assumes that people usually open their mouth when they are surprised. In addition face must be in frontal upright position. This is performed by checking for the height

property of the mouth facial feature. If it exceeds 10% of face height value then mouth is said to be opened thus surprise emotion is detected.



**Figure 3.13** Detection of surprise emotion

It works well under good lighting conditions where mouth state is detected. Width of human mouth facial feature is larger than height value in frontal upright position. When the face region is rotated then width of the mouth becomes height and gives false drops.



---

## CHAPTER FOUR

# VIDEO INDEXING and RETRIEVAL

---

In the recent years, size of raw video data increases enormously and efficient accessing to these data is getting more and more important day by day. Because of increasing availability of visual information to the user, there is a corresponding need for tools that used to make well organized indexing, searching and querying of this information.

Compared with other multimedia data, size of video has the biggest cost. Video is the most complex multimedia object; it includes still images as frames, motion, audio, etc. As a result, without summarizing such big sized data, efficient access is almost impossible.

### **4.1 Introduction**

Automatic detection, extraction, summarization, indexing and querying of human facial information from raw video source are an important research problem in the context of content-based video indexing and retrieval. It allows fast indexing and summarization of binary information with no human intervention. Traditional indexing techniques that applied to the textual data are not applicable to digital audiovisual information. Textual information in video based indexing represents only a small portion of the volume of data available. Most of the current state of art for extracting and detecting the above information uses Multi-Level Neural networks with Hidden Markov Models (HMM). Automatic extraction of video objects can be used to generate a video database for efficient querying. The problem is how to automatically identify and extract these features from the raw video source. System architecture for a video indexing system is application model dependent.

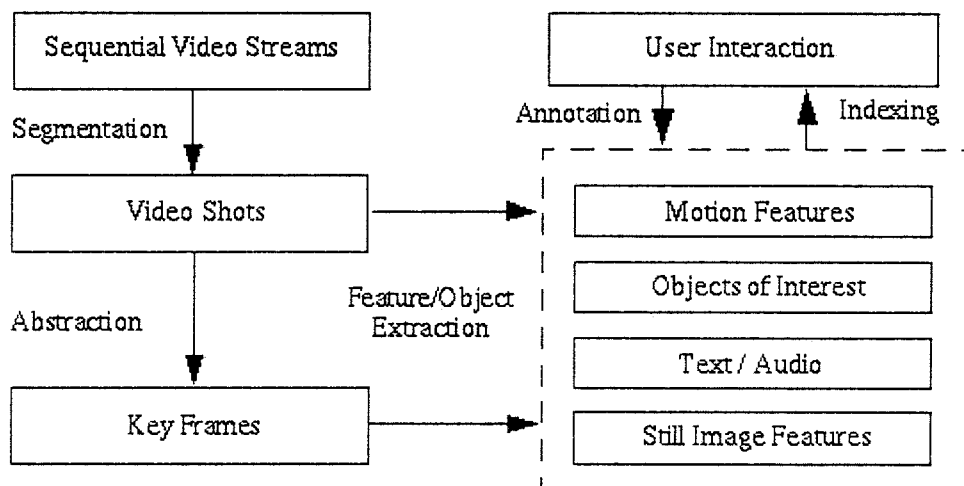
“Video indexing is attaching content-based labels to (searchable) video units.” (Ahanger & Little, 1995). If the data space is not too large then users can use a priori knowledge of the database contents to retrieve data efficiently. However, a large size of data space strains the user's ability to operate with the data content. Human visual system is limited to operate with a few numbers of visual digital data. In that case, users need to manually search on large data sets and these are very time-consuming tasks.

When we look at movies, we can easily see that the most important content of the movies are humans, means actors, actions and events if it is not an animal documentary film. In order to develop a good video indexing application these subject must be considered in a significant manner.

Conventionally, video data have to be manually annotated with keywords or in some cases with speaker's sentences. But in the area of Video Indexing and Retrieval, it is known that text based video annotation without automatically summarizing raw video is often insufficient because of time required to annotate video object movements, events is too much. However, it is also impossible to develop a fully qualifying system with current state of art that automatically extracts features from video and operates with a wide variety of video sources. Because of complex nature of the multimedia objects, it is difficult to extract human related information in an automatic way.

In most of the previous work, skin color information is used to speedup NN or HMM-based systems. Using skin color provides eliminating of unimportant image locations and helps segmentation process thus it decreases the search area for the NN based systems. However, there is no developed comprehensive system that which can robustly extract human facial information from large variety of video.

Figure 4.1 shows conceptual architecture or content-based video indexing and retrieval. (Zhong et al, 1996)



**Figure 4.1 Conceptual Model for Content Based Video Indexing and Retrieval  
(Zhong et al, 1996)**

There are a number of video analysis techniques developed in order to operate the model in Figure 4.1. Most of these models use video frame, motion, audio and text based feature extraction methods. Most research on video content involves automatically detecting the boundaries between camera shots. The first step in video indexing and retrieval is to parse a given video into its elemental units (shots).

## 4.2 Video Analysis Techniques

The increased availability and usage of on-line digital video has causing an emerging need for automated video content analysis techniques. The most common video analyze techniques are shot based method.

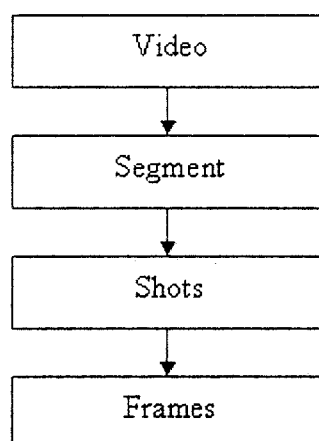
A typical video indexing application consists of the following steps:

- Segment the video hierarchically into sequences, scenes, and shots.
- Describe the complete video, bibliographic information (title, creator, dates, subjects, item numbers, publisher details, names, synopsis etc.) plus format, frame rate, duration etc

- Describe each sequence-id, start time/frame, end time/frame, brief textual summary
- Describe each scene-id, start time/frame, end time/frame, brief textual summary, transcript (ideally derived from a closed caption decoder)
- Describe each shot-id, start time/frame, end time/frame, keyframe (first frame of the shot, ideally derived from an automatic shot detection algorithm)

Figure 4.2 shows the hierarchical structure of video data. According to this figure;

- **Shot:** A shot is a continuous sequence of frames that presents continuous action captured from one camera. It is the basic unit of video. The process of video editing usually refers to the process of deciding the temporal ordering of the shots.
- **Scene:** A scene is composed of one or more shots, which present different views of the same event, related in time or space. There are problems in automatically defining scene regions. For example, a person walking on a street would be one scene, even though different camera angles might be shown. Two camera shots showing two different people walking on a street might also be one scene if the important object was the street and not the people.
- **Segment:** A segment is composed of one or more related scenes.



**Figure 4.2 Video hierarchy**

### 4.2.1 Shot Boundary Detection

In order to detect shot boundaries within a video it needs to find for some changes across the boundary. Most of the previous works focused on cut detection. The more recent works have focused on detecting gradual transitions. There are a number of different types of transitions or boundaries between shots.

- **Cut:** A cut is an abrupt shot change that occurs in a single frame.
- **Fade:** A fade is a slow change in brightness usually resulting in or starting with a solid black frame.
- **Dissolve:** A dissolve occurs when the images of the first shot get dimmer and the images of the second shot get brighter, with frames within the transition showing one image superimposed on the other.
- **Wipe:** A wipe occurs when pixels from the second shot replace those of the first shot in a regular pattern such as in a line from the left edge of the frames.

Of course, many other types of gradual transition are possible. A good shot boundary detection algorithm should correctly identify gradual shot transitions using some difference metrics. In addition, basic camera operations need to be determined. Figure 4.3 shows basic camera operations.

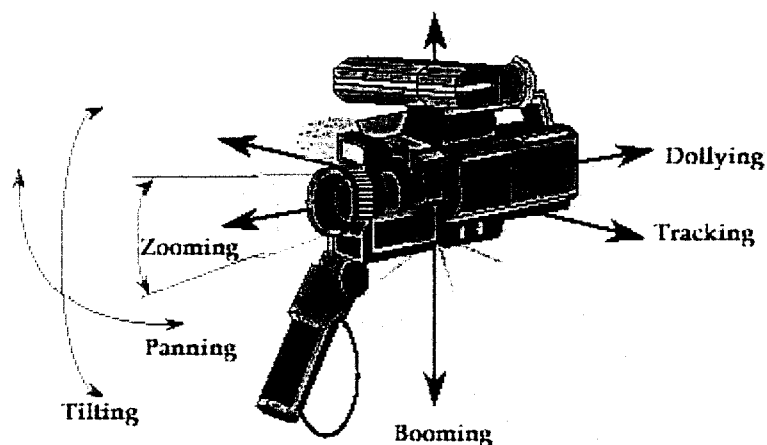


Figure 4.3 Basic camera operations

#### 4.2.1.1 Pair-wise Pixel Difference

Pair-wise pixel difference is the obvious metric to consider first. Considering two sequential frames;

Let  $P_i(k,l)$  represents  $(k,l)$  pixels of  $i^{th}$  frame and  $DP_i$  indicates that in  $i^{th}$  frame, whether pixels are changed or not.

$$DP_i = \begin{cases} 1 & \text{if } |P_i(k,l) - P_{i+1}(k,l)| > t \\ 0 & \text{otherwise} \end{cases} \text{ where } t \text{ is the threshold value.}$$

The for  $M \times N$  size frames a shot boundary exists if :

$$\sum_{k,l=1}^{M,N} \frac{DP_i(k,l)}{M * N} > T_b \text{ But this is very sensitive to both camera and object motion}$$

According to Boreczky & Rowe, the easiest way of detecting whether the two frames are significantly different or not is to count the number of pixels that change in value more than some threshold value  $t$ , this total is then compared against a second threshold to determine whether a shot boundary exists or not (Boreczky, & Rowe ,1994). This method is sensitive to camera motion. Zhang et al implemented this method (Zhang et al, 1993, pp. 10-28). In their work, they had used a  $3 \times 3$  averaging filter. According to their results, by selecting a threshold tailored to the input sequence good results were obtained. However, their methods are a bit slower because of the additional  $3 \times 3$  filter. Another point is that manually adjusting the threshold works better but unlikely to be practical.

Hampapur et al used another technique and they have computed chromatic images by dividing the change in gray level of each pixel between two images by the gray level of that pixel in the second image (Hampapur et al, 1994, pp. 357-364). During the dissolve and fade effect, this image assumes almost a constant value.

Shahraray has divided the images into 12 regions (Shahraray, 1995, pp. 2-13). In his work, he tried to find the closest match for each region in source image neighborhood around the region in the other image. This action is similar to motion

vector extraction methods. Weighted sum of the region differences provides the image difference measure.

#### 4.2.1.2 Histogram Comparison

Histograms are one of the most commonly used methods to detect shot boundaries within video data. In this method, histograms of colored or gray scale pixels in each frame are used to detect shot boundaries. It assumes that the background information does not change either so frequently or strongly among the boundaries of a shot region. If the bin-wise difference between the two histograms exceeds the threshold value then this state results finding of shot boundaries.

If there are  $n$  frames each of size  $M \times N$  and let  $H_i(j)$  be the histogram value of  $j^{\text{th}}$  bin of the  $i^{\text{th}}$  frame. Then the difference between the  $i^{\text{th}}$  and  $(i+1)^{\text{th}}$  frame can be defined as;

$$SD_i = \sum_{j=1}^m |H_i(j) - H_{i+1}(j)|$$

If the  $SD_i$  value is greater than the threshold  $T_b$  then a shot boundary is detected.

It is clear that for both approaches, selection of threshold value critically effects the number of shots can be detected by the system. If it has a lower value, then number of detected shot increases, otherwise decreases.

Adaptive threshold based methods depending on content of the video can be a solution to this problem, but it is quite hard to implement.

Ueda et al have used the color histogram change rate to find shot boundaries. (Ueda et al, 1997, pp. 343-350)

Nagasaka & Tanaka find that, the best result can be obtained from dividing the images into 16 regions, using  $\chi^2$  test on color histograms of those regions. In order

to reduce the effect of object motion and noise, they have eliminated 8 largest differences (Nagasaka & Tanaka, 1992, pp. 113-127).

Swanberg et al used gray level histogram differences in regions; on CNN Headline News (Swanberg et al, 1993, pp. 173-187). Their test video had a very regular structure. They did some simple shot categorization by comparing shots with predefined known types of shots such as anchorperson shot from a database. By using this method, they were also group shots into scenes and segments

Zhang et al compared different types of methods including pixel differences, statistical differences and histogram methods (Zhang et al, 1993, pp. 10-28). According to their results histogram based methods has advantages on accuracy and speed. In their work, they have used two thresholds to detect wipe and dissolve effect, if the histogram difference is between the two different threshold values then they mark this point as a beginning point of a gradual transition effect.

#### 4.2.1.3 Motion Continuity

One of the characteristics of the video is its motion capability. Motion is a good candidate for detecting shot boundaries. This method also uses a threshold value to detect shot boundaries. Most of the commonly used video compression algorithms and almost all of the video compression hardware use block-based motion vectors.

MPEG Motion vector information can also be used from compressed video sequences. Nevertheless, MPEG selects vectors based on compression efficiency. As a result, MPEG usually selects inappropriate vectors for image processing purposes.

If we use  $D = (x, y)$  as a motion vector and divide each frame of the video into a number of small blocks then;

For each block  $b_i$  define



$$\omega_{1_i}(b) = \begin{cases} 1 & \text{if } |D_{b_i}| > t_s \\ 0 & \text{otherwise} \end{cases} \quad \text{and} \quad \omega_{2_i}(b) = \begin{cases} 1 & \text{if } |D_{b_{i-1}} - D_{b_i}| > t_m \\ 0 & \text{otherwise} \end{cases}$$

$\sum_b \omega_{1_i}(b)$  Counts the number of significant motion vectors in frame  $i$ .

$\sum_b \omega_{2_i}(b)$  Counts the number of motion vectors in frame  $i$  which differ significantly from their corresponding vectors in frame  $i+1$ .

The smoothness of frame  $i$ ,  $W_i$ , can be defined as;

$$W_i = \frac{\sum_b \omega_{1_i}(b)}{\sum_b \omega_{2_i}(b)} \quad \text{As a result, if } W_i < T \text{ then a shot boundary exists.}$$

In order to detect shot type as a zoom or pan, (Ueda et al, 1997, pp. 343-350) and (Zhang et al, 1993, pp. 10-28) used motion vectors.

#### 4.2.1.4 Three frames Approach

Sehti & Patel considered three consecutive frames (Sehti & Patel, 1995). These are formally called  $r$ ,  $s$  and  $t$ .  $D_{rs}$  and  $D_{st}$  are the measure of the frame dissimilarities. According to these values Observer Motion Coherence OMC, defined by;

$$OMC(r, s, t) = \frac{|D_{rs} - D_{st}|}{D_{rs} + D_{st}}$$

If the  $OMC(r, s, t)$  value is, a number that close to 1, it means that there is no change between the consecutive frames  $r$ ,  $s$ , and  $t$ . A shot is detected when the value of the  $OMC(r, s, t)$  close to number zero.

#### 4.2.1.5 Twin-Comparison Method

This approach is widely using for detecting edit effects and gradual transitions within video sequence. The basic idea is that frames are marked differently before and after the gradual transitions.

The important problem of this method is that basic camera operations including pan and zoom can be misinterpreted as special effects. Threshold based solutions cannot be used because pan and zoom operations produce the same change effects as special effects. In addition, if a threshold based method is used then the value of the threshold must be lower value compared with standard shot detection threshold value.

Motion feature is a solution that can be used to detect this kind of camera operations. During the pan and zoom operation of the camera, motion vector fields have the same direction with almost a fixed angle value. The algorithm works as follows;

Let  $SD_i$  represents Standard Deviation of frame  $i$  and  $T_b$  is the threshold value.

Compute  $SD_i$  for all frames in the video.

Find camera breaks where  $SD_i > T_b$

Mark potential gradual transition subsequences defined by  $GT$

$GT = \{[F_s, F_e]\}$  of the video wherever  $SD_i > T_b$  for  $F_s \leq i \leq F_e$  where  $F_s$  is start frame and  $F_e$  is end frame of gradual transition.

For each gradual transition calculate frame to frame difference:

$$AC = \sum_{i=F_s}^{F_e} SD_i$$

If  $AC > T_b$ , then declare  $[F_s, F_e]$  as a gradual transition effect.

#### 4.2.1.6 Compression Differences

Little et al used differences in the size of JPEG compressed frames with same compression rate to detect shot boundaries as a supplement to a manual indexing system (Little et al, 1993, pp. 427-436).

Arman et al used differences in the discrete cosine transform (DCT) coefficients of JPEG compressed frames as a measure of frame similarity, as a result of this

challenging situation, they have decreased computation time by not making compression on frames (Arman et al, 1994, pp. 211-219). They have also considered the differences between color histogram values in order to detect potential shot boundaries.

#### 4.2.1.7 Edge Tracking

Zabih et al compared several methods for shot detection including their own algorithm, based on edge detection (Zabih et al, 1993, pp. 189-200). They have calculated the percentage of edges that enter and exit between the two frames. After that, they have look for a large percentage value on edge change. According to their results, their method was more accurate at detecting cuts than histogram-based methods.

#### 4.2.2 Gradual Transition Detection

This class includes a set of effects such as dissolving, fading in, fading out, mattes and translates.

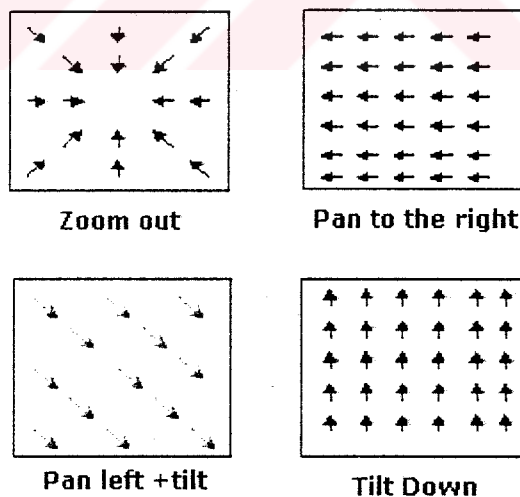


Figure 4.4 Flow fields during camera motion

**Fade in-out:** A fade in-out is an optical process of allowing the progressive darkening of a shot until the last frame becomes completely black (fade out). A

fade in is the opposite of fade out and in this case, the last frame of the shot boundary becomes completely to light.

**Dissolve:** It is a superimposition of a fade in and fades out. In this case, the first shot frame fades out while other sequential frames fade in to full light.

**Translate:** A translate occurs when the first shot frame translates out, uncovering the shot that follows the edit

#### 4.2.2.1 Plateau Detection

Because of the small size of changes in frames in gradual transitions, standard comparison based methods are not suitable for detection of gradual effects (Yeo, 1996). In this method, first  $i^{th}$  frame is compare to  $i^{th} + k^{th}$  frame in which  $k$  is a constant value for each shot region. Then they obtained a sequence of delayed inter-frame distances.

$$D_i^k = d(X_i, X_{i+k})$$

If the value of  $k$  as a frame is greater than the measured gradual transition then the sequence  $D_i^k$  is defined as plateau of maximum width. This kind of detection can also detect absorb changes in consecutive frame.

#### 4.2.2.2 Feature Based Detection

Zabih et al realize that during a cut or dissolve effect new intensity edges appear far from location of old edges (Zabih et al, 1995). Edge pixels that appear far from the existing pixels are called as entering pixels and, edge pixels that disappears far from existing edge pixels are called exiting pixels. The algorithm is as follows;

Frames  $F_i$  and  $F_{i+1}$  are aligned using a global motion compensation algorithm.

Edge detection is made by applying Canny algorithm

Binary edge maps are dilated by radius  $r$

Fraction of entering edge pixels  $\rho_{in}$  and exiting edge pixels  $\rho_{out}$  are computed.

Then breaks is detected by looking at the value of  $\rho = \max(\rho_{in}, \rho_{out})$

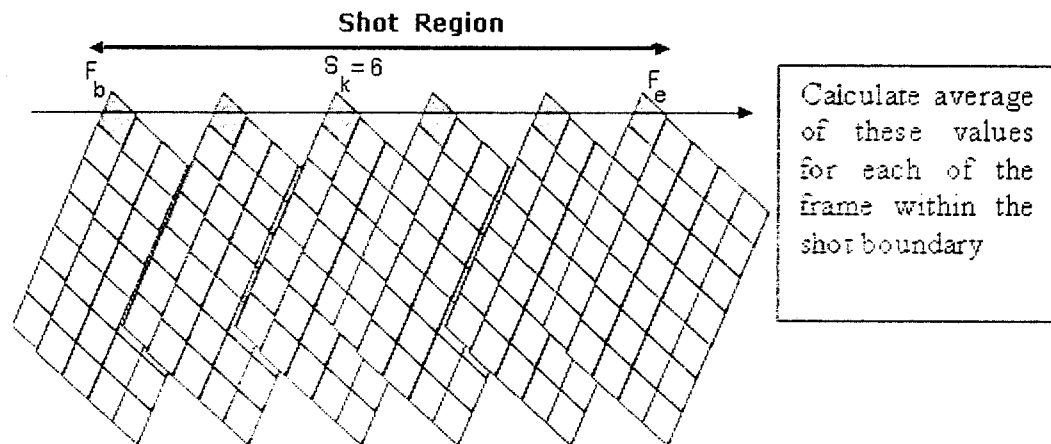
During a fade in, value of  $\rho_{in}$  is higher than value of  $\rho_{out}$  and  $\rho_{out}$  is much more than  $\rho_{in}$  during a fade out.

### 4.2.3 Key Frame Selection

Key frames have an important role in video indexing and retrieval. Shots are basic units of video but there is a necessity of having a handle that represents content of video. It is a simple method. If it is used with boundary detection methods then key frame selection can operate in better way. In that case, first frame of the current shot can be selected as a key frame. But in some circumstances the first frame of the shot can be meaningless and can not cover the content of the shot region therefore some other techniques can be used such as in a simple way, selecting the middle frame within a shot region as a key frame. These methods can be improved to select the best key frame for shot region.

#### 4.2.3.1 Average Pixel Method

The first step for this method is shot boundary detection. If we assume that, each frame has size of  $M \times N$  and shot region  $k$  starts with frame  $F_b$ , ends at frame  $F_e$  and has  $S_k$  number of frames in shot  $k$  then average pixel values of shot  $k$  can be defined as average value of same pixels of each frame. As a result, the averages frame  $F_{avg}$  usually a blur image and the object motion can affect it. Figure 4.5 explains the algorithm.



**Figure 4.5 Average Pixel Method**

In another words 
$$\mu(i, j) = \frac{1}{S_k} \sum_{i=1}^{S_k} pixel(i, j)$$

By using the above formula, an average frame of shot  $k$  is calculated for each  $M \times N$  pixel. After calculating the average frame  $F_{avg}$ , compute the distance of every frame within the shot  $k$  to the average frame  $F_{avg}$ . Let  $FK_k$  will be the key frame and  $F_i$  is a frame within the shot region shot  $k$  then,

$$FK_k = \left\{ F_i \mid \exists i \left| (F_i - F_{avg_k}) \right| \leq \forall j \left| (F_j - F_{avg_k}) \right| \text{ where } 0 \leq i, j \leq S_k \right\}$$

### 4.3 Critical Video Formats

Most known file formats used in personal computers are files that have AVI or MPG extension.

#### 4.3.1 Audio Video Interleave (AVI)

It is one of the most common video types used in personal computers. AVI stands for Audio Video Interleave. Microsoft defined AVI file format.

An AVI file is a video format, comparable in nature to QuickTime. Microsoft developed this format for Microsoft Windows 3.1 in 1991. Since its release, the format has generally become cross-platform. However, because of licensing issues, not all AVI video will work anywhere. AVI files are generally smaller than MPEG files, but with lower quality. AVI files are slightly easier to edit than MPEG files.

Capturing 352×240 AVIs at 30.x fps can fill up 90MB of space per minute, depending on the settings of the video and audio quality. At the highest quality, a full hour of AVI recording requires over 5GB.

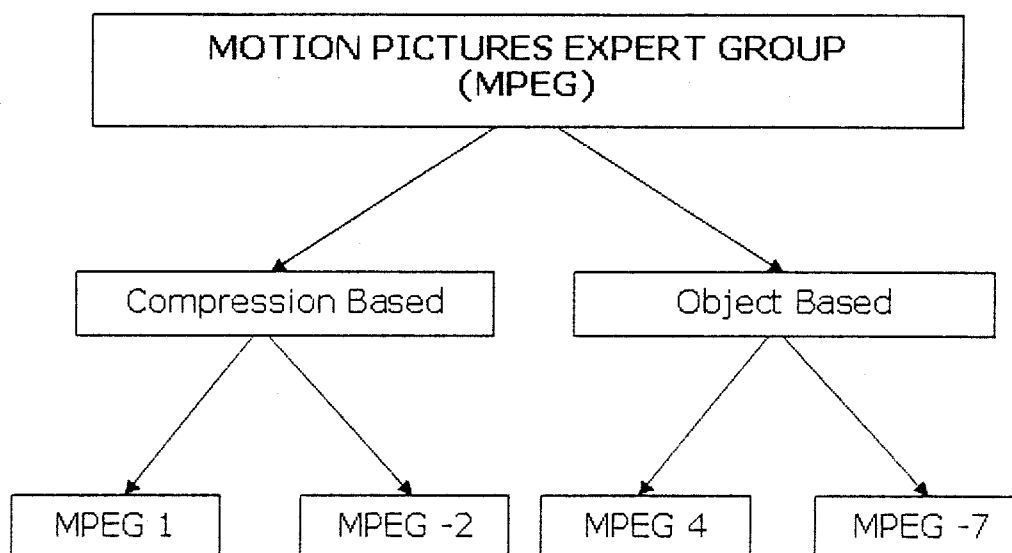
#### **4.3.2 Moving Pictures Expert Group (MPEG)**

MPEG, (pronounced as M-peg), stands for **Moving Picture Experts Group**. It is the name of family of standards used for coding audio-visual information (e.g., video, audio) in a digitally compressed format. MPEG provides CD quality audio at low data transfer rates. This enables video and audio compression with acceptable quality and retrieving speed.

MPEG is derived from the original work of the Joint Pictures Expert Group (JPEG). The JPEG standard is a lossy technique for compressing still images. It removes disused information from the image that humans eye do not see. MPEG video adopts the YCrCb color model.

Suppose you want to capture a 3-minute video at 15 frames per second having 24-bit color in a window that has size of 320×240 pixels. Your digitized source file would be approximately 622 MB

- $(320) \times (240) \times (24) \times (15) \times (60) \times (3) = 622 \text{ MB}$  with no compression.



**Figure 4.6 Hierarchy in Mpeg Standards**

In MPEG Systems, the video and audio streams coordinated as a single data stream in order to use it for suitable digital storage and transmission. The MPEG system consists of two layers:

- System Layer (timing information to synchronize video and audio)
- Compression Layer (includes audio and video streams)

#### **4.4 The Application Metadata Standards for Video Indexing**

There are two main standard developed, in to cover video indexing requirements. These are Dublin Core and MPEG-7 standard. Details are as follows;

##### **4.4.1 Dublin Core**

The Dublin Core Metadata Initiative (DCMI) is a META data standard whose development began in 1995 at an OCLC meeting in Dublin, OH. Its objective is to develop a META data standard to enhance core set of META data elements or



attributes to structure the description of networked resources. The DCMI assists the simple description of a networked resource, but is not accepted by all search engines.

The Dublin Core Metadata Element Set Version 1.1 consists of 15 descriptive data elements relating to content, intellectual property and instantiation. These elements includes title, creator, publisher, subject, description, source, language, relation, coverage, date, type, format, identifier, contributor and rights. Each Dublin Core element is defined using a set of ten attributes from the ISO/IEC 11179 [ISO11179] standard for the description of data elements. (Dublin Core Metadata Element Set, 1999) Details of data element information are as follows;

**Name:** The label assigned to the data element

**Identifier:** The unique identifier assigned to the data element

**Version:** The version of the data element

**Registration Authority:** The entity authorized to register the data element

**Language:** The language in which the data element is specified

**Definition:** A statement that clearly represents the concept and essential nature of the data element

**Obligation:** Indicates if the data element is required to always or sometimes be present (contain a value)

**Datatype:** Indicates the type of data that can be represented in the value of the data element.

**Maximum Occurrence:** Indicates any limit to the repeatability of the data element

**Comment:** A remark concerning the application of the data element

These 15 DC META data elements are grouped in three main classes. These are,

**Content:** Title, Subject, Description, Source, Language, Relation, Coverage

**Intellectual Property:** Creator, Publisher, Contributor, Rights

**Instantiation:** Date, Type, Format, Identifier

#### 4.4.2 MPEG-7

MPEG-7 is an ISO/IEC standard being developed by MPEG (Moving Picture Experts Group). MPEG-7 will be standard for describing the multimedia content data that will support operational necessities. MPEG will not regulate or evaluate applications or is not aimed at any one application in particular; somewhat, the elements that MPEG-7 standardizes shall support as wide range of applications as possible.

MPEG-1 and MPEG-2 standards make video on CD-ROM, DVD, and Digital Television possible. MPEG-4 provides the standardize distribution and content access paradigms of the fields of digital television, interactive graphics and interactive multimedia.

MPEG-7 aims at offering a comprehensive set of audiovisual description tools to create descriptions, which will form the basis for applications enabling the needed quality access to content, good storage solutions, accurate filtering, searching and retrieval.

##### 4.4.2.1 The Scope of MPEG-7

Figure 4.7 explains a hypothetical MPEG-7 sequence in way. The circular boxes illustrate tools that are doing things, such as encoding or decoding, whereas the square boxes denotes static elements, such as a description. The grayed boxes in the figure cover the normative elements of the MPEG-7 standard. The standard does not describe the development of (automatic) extraction of descriptions/features, nor does it specify the search engine, filter agent, or any other program that can make use of the descriptions. (Martinez,ISO N3545,2000)

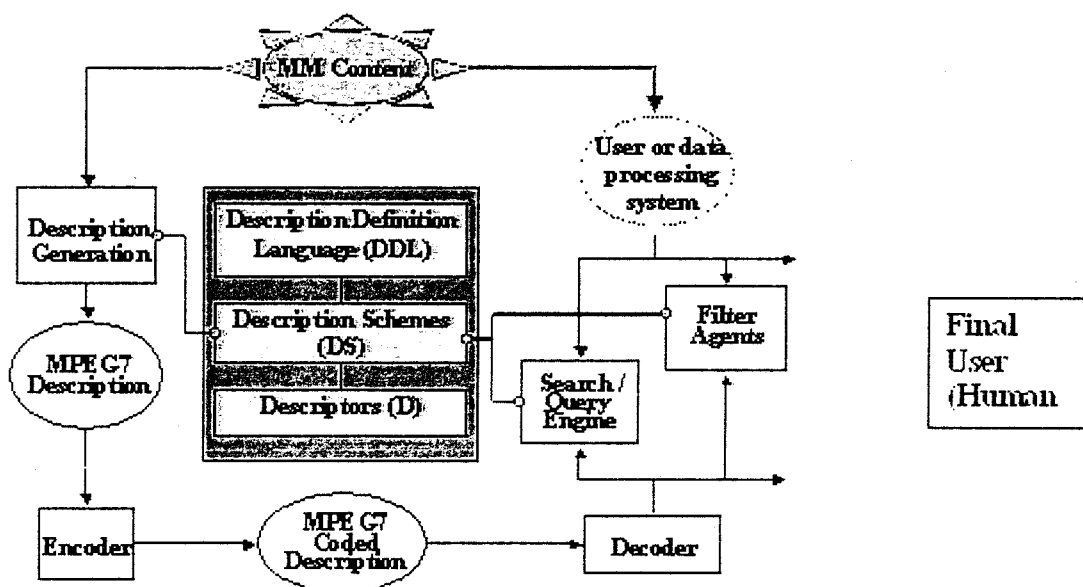


Figure 4.7 An abstract representation of possible applications using MPEG-7

#### 4.4.2.2 Mpeg-7 Capabilities

With MPEG-7 enabled technology. We will be able to:

- Play a few notes on a keyboard and get in return a list of musical pieces containing (or close to) the required tune in some way matching the notes,
- Draw a few lines on a screen and get in return a set of images containing similar graphics, logos, ...
- On a given set of objects, express actions and connections between objects and get in return a list of animations fulfilling the described temporal and spatial relations.
- Using an piece of Pavarotti's voice, and getting a list of Pavarotti's records, video clips where Pavarotti is singing or video clips where Pavarotti is present

#### 4.5 Video Indexing Applications

Automatic indexing and retrieval of video is an important subject. Most of the work in this area is based on image retrieval systems.

#### **4.5.1 Name-It (CMU and NACISIS)**

Satoh & Kanade developed Name-It system at CMU as a part of Informedia project. It automatically labels faces in video databases (Satoh & Kanade, 1996), (Satoh & Kanade, 1997, pp. 368-373). It uses a face detector to find human faces inside the video and when it finds the face, it records the time information.

The system also uses an automatic speech recognition that finds English words and produces with the previous information, textual transcripts of the video by considering time information.

The system also considers co-occurrence of the same faces within a sequence by comparing the similarity of the current face to all other detected faces in the database. The co-occurrence value is then used to answer user queries.

The user of the system could give a query like "Clinton", and get faces that are commonly associated with the name "Clinton".

#### **4.5.2 Skim Generation (CMU)**

Smith and Kanade developed another application from the Informedia project generating summaries from video sources (Smith & Kanade, 1996), (Smith & Kanade, 1997, pp. 775-781). It uses only important parts of the video in order to summarize the whole video. They have using some heuristics such as, important parts of the video immediately come after the scene cut, and it contains explanation text and contains faces. They have used upright face detector.

#### **4.6 Shot Detection Study**

In this thesis, as a preliminary work, a small Histogram Difference-based shot detection application has been developed. It computes frame-to-frame differences and then calculates a histogram consists of 27 bins calculated from each frame. After completing, histogram creation, mean variance and standard deviation values are

computed. These values represent frame-to-frame differences. Threshold value  $T_b$  is calculated as follows;

$$\text{The mean value } \bar{x} = \frac{\sum_{i=1}^n (H_{i+1} - H_i)}{n}$$

$$\text{Standard Deviation } SD = \frac{\sum_{i=1}^n \sqrt{(H_i - \bar{x})^2}}{n-1}$$

Threshold  $T_b = \bar{x} + (\alpha * SD)$  where  $\alpha$  value is given by the user and its default value is 0.25.

After that, if any frame-by-frame difference value between two sequential video frames exceeds the value of  $T_b$ , then a shot exists and the program shows the shot boundary frames as a JPEG formatted files on the browser.

If the MPEG file is not processed previously then, the program reads each frame and then calculates pair-wise differences among each frame. The threshold value is a parameter to the functions so the user gives it before the application is executed. When a frame that exceeds the threshold value is detected then this frame and its successor frame showed. During this step, extracted information added to the database.

#### 4.7 Video Indexing and Retrieval Study

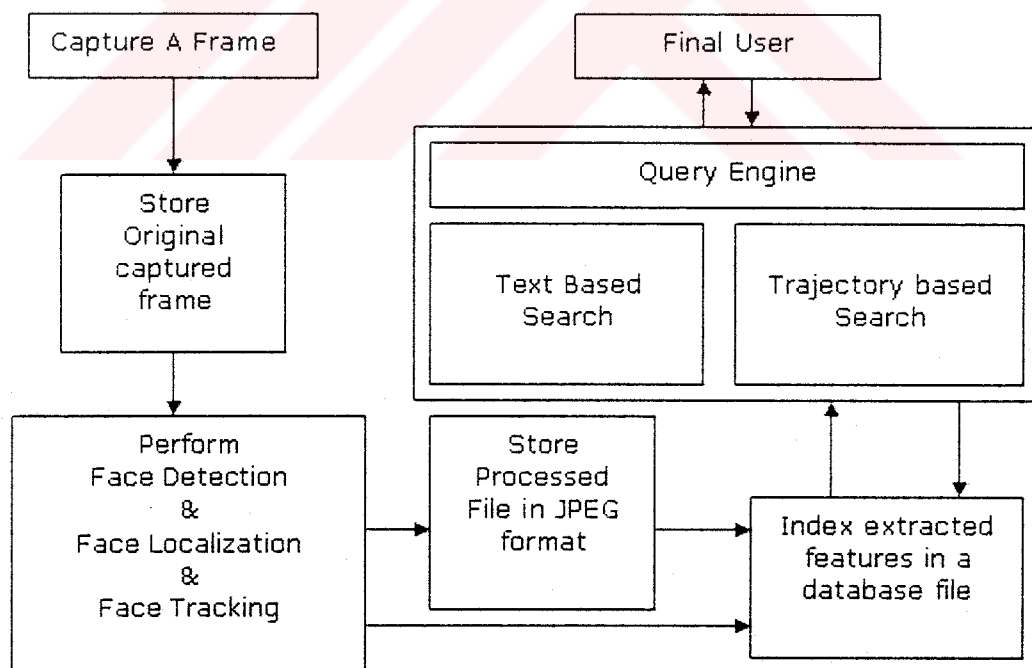
The system developed in the content of this thesis is a fully automatic video indexing and retrieval application that uses skin color based frontal upright face detection method in order to extract human facial information as well as classification of these objects. The system has full functionality on both uncompressed AVI files and real-time live video feeds.

### 4.7.1 Introduction

Automatic extraction of human facial information from raw video files is an active research area in content of Computer Vision. It allows fast indexing and summarization of binary information with no human intervention. In recent years, the need of summarization tools for video increased. They are most applicable to TV corporations. They have huge amount of video data and everyday they are searching large amount of video data in order to produce one or two minute news video.

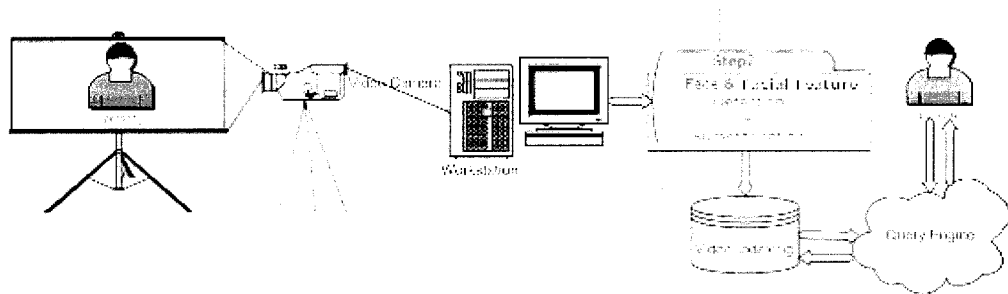
Nevertheless, in order to satisfy user needs programs should be work with wide range of videos each having different background styles, color information, peoples from different races, etc. A comprehensive system should support almost all of these requirements.

Figure 4.8 represents the basic system architecture for real-time video indexing and retrieval.



**Figure 4.8 Real-Time Video Indexing & Retrieval Basic System Architecture**

The first mode of the system is real-time processing mode. The conceptual schema for Real-Time video indexing is shown in Figure 4.9



**Figure 4.9 Real –Time Video Indexing Conceptual Schema**

According to the Figure 4.9 the algorithm works as follows;

- User initiates and starts the video capturing.
- Captured video frame is processed as BMP formatted file and then it is stored in a JPEG formatted file with 50% compression rate to reduce the space needed for captured frames. File names are stored in time format including milliseconds to prevent writing on existing filename.
- The process of face detection is the same as the preliminary work done for the thesis.
- Each captured video frame is then summarized and indexed in a video index file.

In Figure 4.10, image at the left hand side is the source image captured from the video device. Image in the middle of the figure is the segmented image file where the background information is removed and there is only skin colored region exists on this image. On the right side of Figure 4.10 Processed Image is shown. A border in black color restricts the boundaries of the human skin face and there are several white colored rectangles in it. Each of these rectangles shows one distinct human facial feature such as eyebrows, eyes, nose or mouth. Figure 4.11 show more detailed information about the processed image.

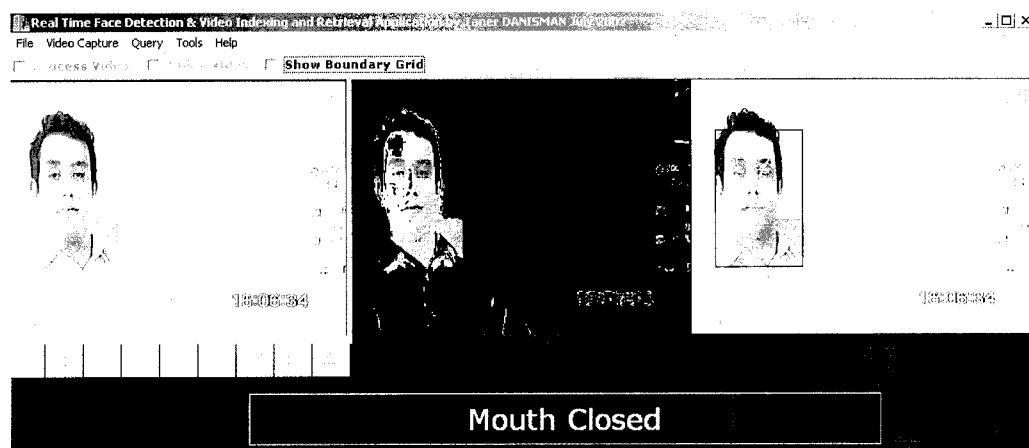


Figure 4.10 Real-Time Processing



Figure 4.11 Real-Time Processing (Detailed)

In Figure 4.11, when the background is easy to remove then location of eyes and mouth is easily detected. In Figure 4.11 there are white lines connected to each of the



facial features. Each of the lines is upper left minimum  $(x, y)$  coordinates of the template face's (Figure 2.18) facial features. Line length is used to match minimum distance of founded regions and real facial features.

In this thesis, a new object motion based shot detection method has been proposed. In order to detect video shots boundaries, objects are tracked and their locations are logged. I assume that these objects are usually human face. If the location of the object changes its previous location then a shot is detected. Compared with histogram based methods, this method did not considers global information. It has just considers movement of objects in different and previously determined locations. These locations are called as shot locations. If any object changes its shot location to another one then a shot exists. More formally;

- Let we have an image of size  $M \times N$  and the image is divided into  $w \times h$ -sized blocks called as  $I_b$ . where  $0 \leq b \leq i * j$ . These blocks represent shot boundary regions.
- Any object  $O_i$  of size  $P_i \times R_i$  is represented by a rectangular area covering boundaries of  $O_i$  and has coordinates of its upper left corner  $O_i^x$  and  $O_i^y$  respectively.
- Each object  $O_i$  of size  $P_i \times Q_i$  is also divided into  $r \times s$  size of blocks and each sub block  $O_{i,t}$  of  $O_i$  has a specific weight value  $W_{O_{i,t}}$ . In order to detect a shot boundary, weighted sum of all pixels that intersects with image region of  $O_i$  is computed.

- Select the  $b$  value that maximizes the following equation

$$b = \begin{cases} k & \max( \sum_{t=0}^{(P_i/r) * (R_i/s)} W_{O_{i,t}} * \text{Intersect}(Q_{i,t}, I_k) ) \text{ where } 0 \leq k \leq (M/w) * (N/h) \\ 0 & \text{otherwise} \end{cases}$$

where,

- $\text{Intersect}(Q_{i,t}, I_b) = \begin{cases} 1 & Q_{i,t}^{(x,y)} \in I_b^{(x,y)} \\ 0 & \text{otherwise} \end{cases}$

- For the two sequential frame  $I^i$  and  $I^{i+1}$ , a shot exist if and only if  $I_b^i \neq I_b^{i+1}$

visually shows the object motion based shot detection algorithm.

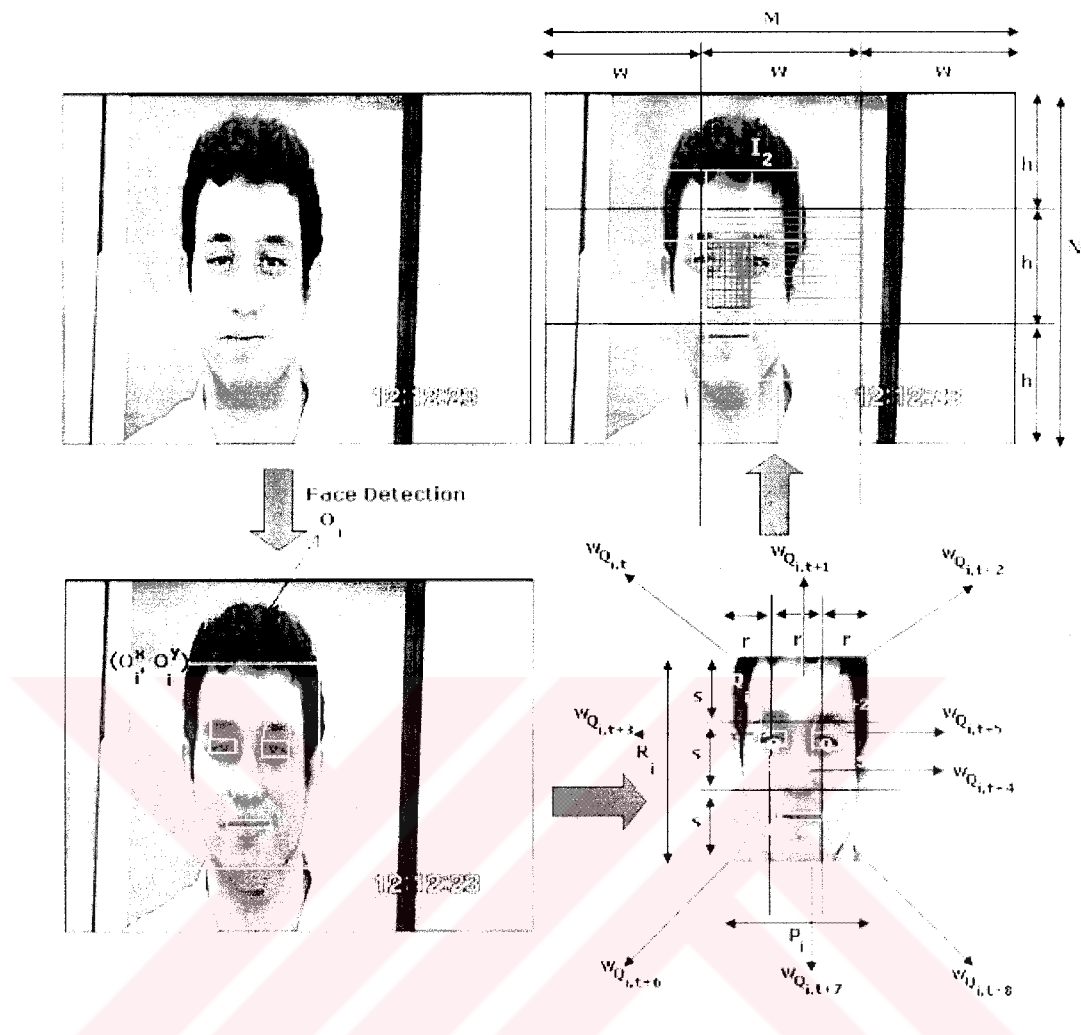


Figure 4.12 Object Motion-based Shot detection algorithm

In the above example, parameters are as follows;

- Image size  $M=320$ ,  $N=240$
- Size of Image sub bocks (shot regions )  $w=106.6$  ,  $h=80$
- Size of the object  $O_i$ ,  $P_i = 95$  and  $R_i = 140$
- Upper left corner coordinates of  $O_i$ ,  $O_i^x = 78$  and  $O_i^y = 56$
- Size of each sub block of  $O_i$ ,  $r = 31.6$  and  $s = 46.6$

- By giving higher weight values to the middle and upper half of the face, the origin moves to the upper parts and in this way we had considered nonexistence of hair feature. Weights of the  $O_i$  sub blocks;

$$W_{O_i,0} = 2, W_{O_i,1} = 2, W_{O_i,2} = 2,$$

$$W_{O_i,3} = 2, W_{O_i,4} = 5, W_{O_i,5} = 2,$$

$$W_{O_i,6} = 1, W_{O_i,7} = 1, W_{O_i,8} = 1.$$

- After computations the value of object location is  $b = 5$ . It means that the face or object  $O_i$  is located in the middle of the current frame and a shot exist if and only if the value is changed in one of the future frames. When the value is changed then it is updated with the new shot region to continue to detect other shots as well.

Figure 4.13 show all location names and their corresponding numbers respectively. It is in row order. If the currently detected face is located in the middle of frame then location value is set to number four.

UPPERLEFT 0	UP 1	UPPERRIGHT 2
LEFT 3	MIDDLE 4	RIGHT 5
DOWNLEFT 6	DOWN 7	DOWNRIGHT 8

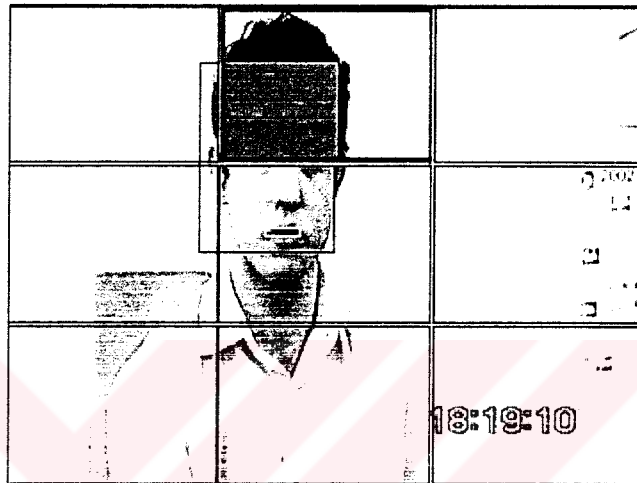
**Figure 4.13 Location names**

Determination of location of a face can be found as follows.;

- For each of the nine locations, calculate the weighted intersection area. In order to give more weight, upper half part of the face candidate is multiplied by the number two

- Select the location that has the highest weighted intersection value.

Figure 4.14 shows that location one namely called as UP is selected for the current face and the most of the upper part of the face is located in location one so, boundaries of location one is painted with red color. Intersection area is shown in blue color.



**Figure 4.14 Estimating Face Locations**

Until now, we were focused on face detection and video indexing. In the previous sections we captured, segmented, classified, extracted and summarized video.

Video Retrieval is the process of retrieving previously well-organized video data. There are two main standard has been developed for video indexing and retrieval purposes, Dublin Core and MPEG-7.

Video retrieval can be accomplished by either text based or trajectory based. For moving objects, trajectory based queries gives more efficient results. In text based retrieval queries consists of name of events, duration or static information such as actor names, year of movie etc. However, trajectory based queries does not use static

information. Queries consists of one ore more objects and their trajectories. Of course, combination of both models can be used to retrieve more accurate results.

In this study both of text and of trajectory-based queries are implemented. Text based queries are based on the outputs of the new object motion based shot detection algorithm. Thus, text based queries can include location information such as “Give me the sequence of frames where human face is located in the middle”.

The second query generation system is trajectory-based queries. In this case, the user is able to draw his/her object trajectory on a trajectory board. Then the system perceives and then interprets the users query by processing the trajectory board. It first checks the logical things such that whether the start location of the trajectory is in the defined face or not. If it is not then the query is rejected. The rest of the system works like the text-based query system. Figure 4.15 shows how the user expresses the query.

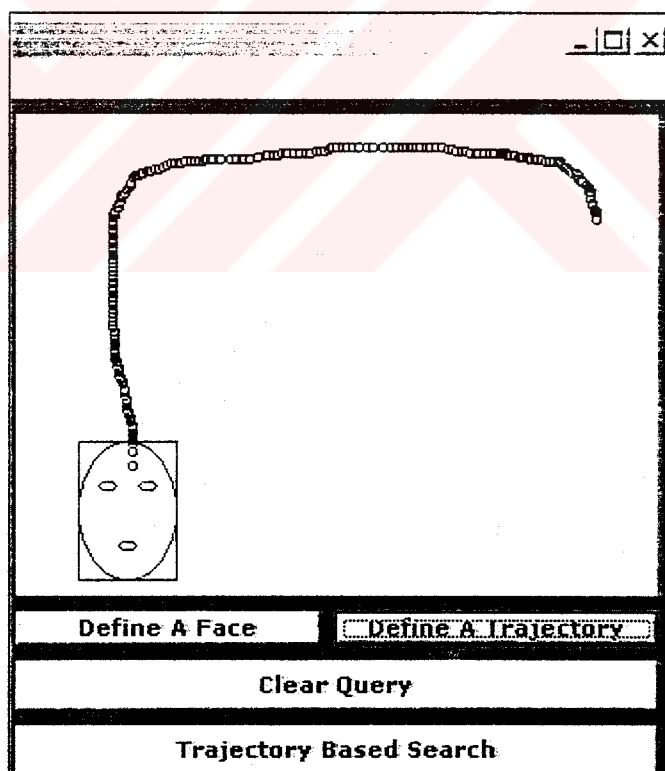
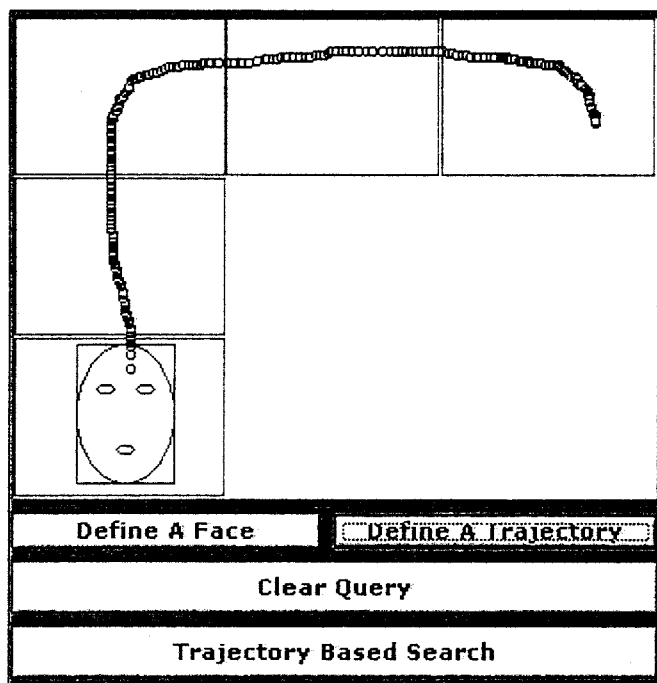


Figure 4.15 Trajectory-based query generation



**Figure 4.16 Result of finding trajectory locations**

#### 4.7.2 Video Indexing & Retrieval Results

Applications can index a frame in approximately 0.33 seconds. Another word its processing power is 3fps. During this process, it creates four different result images in JPEG format of size 7KB each for future use. The first image is the original file; the second image is segmented image, the third image is classified image and the last image is resulting image with detected face and facial features.

Query generation time for trajectory based queries is about 3 seconds. In some cases, it is difficult to represent the object movement when the object crosses at the corner points from one region to another. To prevent such misunderstandings, the source image can be divided more sub blocks in order to increase the accuracy of trajectory-based search.

Retrieval time depends on the size of the index file and the constraints on the query sentence. For 90 seconds of video clip it takes on average one second to retrieve a set of video frames.

According to the observed results, application gives 96% correct results on determining the location of the face without considering mouth state. 4% of the result set is false because of the error in earlier stages. When the face detection fails then retrieval segment also fails.

In addition, mouth detection rate is same as single-face mouth detection rate, which is approximately 51.6%. All the results show that, face detection step has a great importance on location based video indexing and retrieval application.



---

## CHAPTER FIVE

# IMPLEMENTATION and EVOLUATION

---

This chapter first describes implementation details of skin color based face detection, shot detection and video indexing and retrieval application respectively. For each implementation, brief information about coding, interface and experimentation results are given.

### 5.1 Face Detection Study

#### 5.1.1 Coding

Face detection application is developed for both of Linux and Windows platforms in C++. It uses two external library called libmpeg3 to extract RGB frames from MPEG formatted files and IJG JPEG library to read and write these type of files. The application is first developed on Linux environment and then it is ported to Windows platform.

Under Linux environment, **g++ 2.96** has a stack overflow bug on the process of recursive boundary fill algorithm. Therefore, **g++ 3.0.4** has been used.

Earlier we have just work with single JPEG formatted color image files. After being successful on single images then the code is improved in order to operate with a sequence of video frames.



### 5.1.2 Face Detection Study Results

We have used a 250-passport photo of graduate students at Dokuz Eylul University Graduate School of Natural and Applied Sciences. For each image, we observed the number of;

- Total faces detected by the system
- True/False face detections
- True/False left eye detections
- True/False right eye detections
- True/False mouth detections

We have also computed the recall and precision values.

Recall is the ratio of the number of faces detected correctly over the actual number of faces. For example if you knew that, there were 3000 faces in an image database and 300 of these faces are detected then your recall would be 10%. Another word, recall is how much of the available faces actually detected. In case of multiple face detection 20 images used taken from Advances In Information Systems 2001 (ADVIS 2001) conference.

Precision: The ratio of the number of faces detected correctly over the total number of faces detected. If 100 faces detected and 20 of these are really represents a face region, then precision will be 20%. Another word precision is how much of the detected face regions are really a face region. For single face detection, 301 regions are detected to be a faces region. 239 of them are real faces. Table 5.1 and Table 5.2 shows recall and precision values of individual facial features for single and multiple face detection respectively.

**Table 5.1 Single Face Detection results (Recall & Precision)**

Facial Feature Type	Actual Number of Objects (A)	Total Number of Objects Detected (B)	Number of Objects Detected Correctly (C)	Recall =(C/A)	Precision =(C/B)
Face	250	301	239	95.6%	79.4%
Left Eyebrow	250	107	95	38%	88.78%
Right Eyebrow	250	102	90	36%	88.23%
Left Eye	250	183	159	63.6%	86.88%
Right Eye	250	184	159	63.6%	86.41%
Mouth	250	129	107	42.8%	82.94%

**Table 5.2 Multiple Face Detection results (Recall & Precision)**

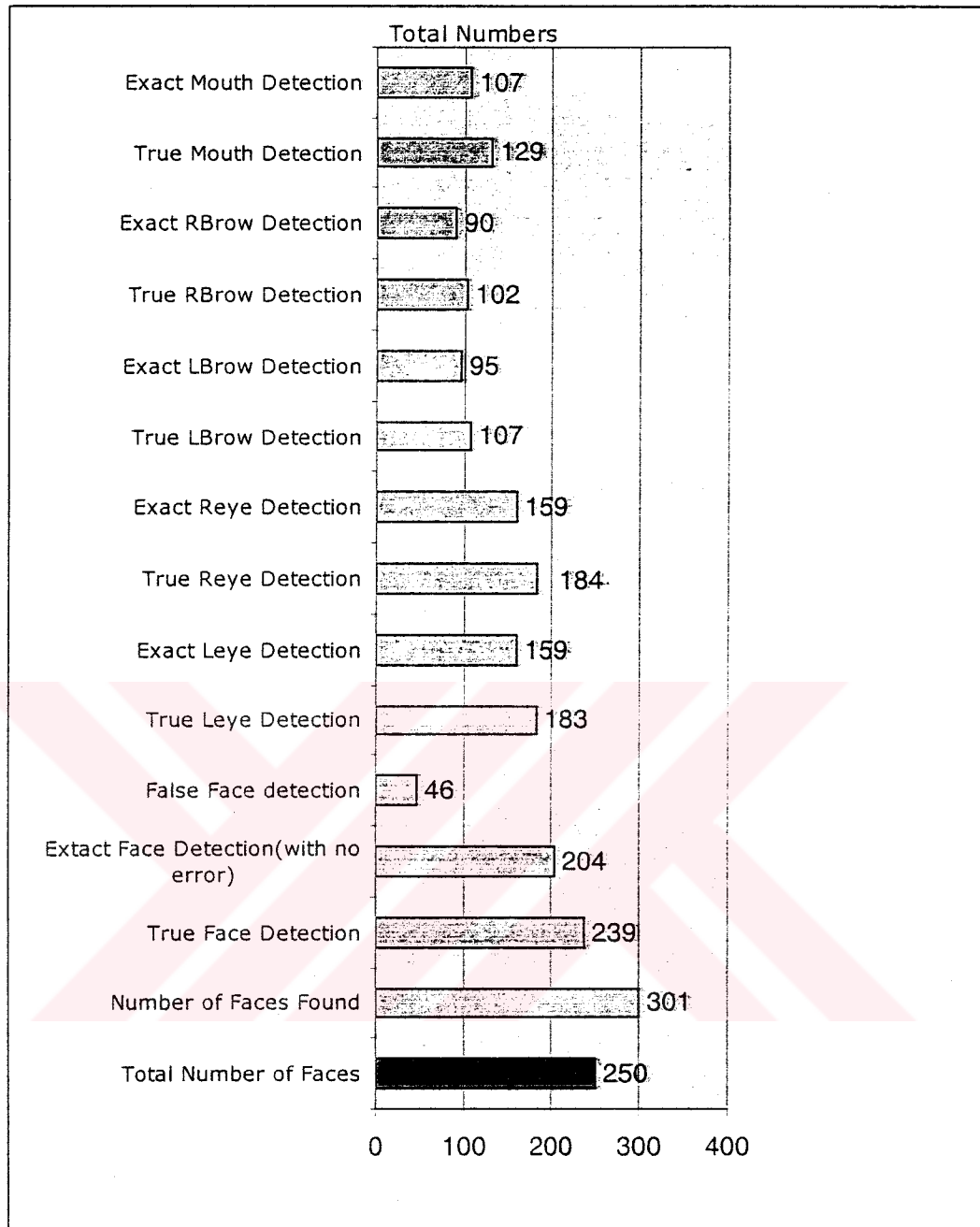
Actual Number of Faces (A)	Total Number of Faces Detected (B)	Number of Faces Detected Correctly (C)	Recall =(C/A)	Precision =(C/B)
12	16	10	83.33%	62.5%
7	7	4	57.14%	57.14%
7	9	5	71.42%	55.55%
11	11	11	100%	100%
2	2	2	100%	100%
5	11	3	60%	27.27%
2	4	2	100%	50%
3	3	2	66.66%	66.66%
3	4	2	66.66%	50%
2	6	2	100%	33.33%
12	15	10	83.33%	66.66%
3	10	3	100%	30%
6	11	3	50%	27.27%
4	5	4	100%	80%

12	12	7	58.33%	58.33%
3	9	3	100%	33.33%
<b>Total : 94</b>	<b>Total : 135</b>	<b>Total : 73</b>	<b>Average : 85.22%</b>	<b>Average : 56.12%</b>

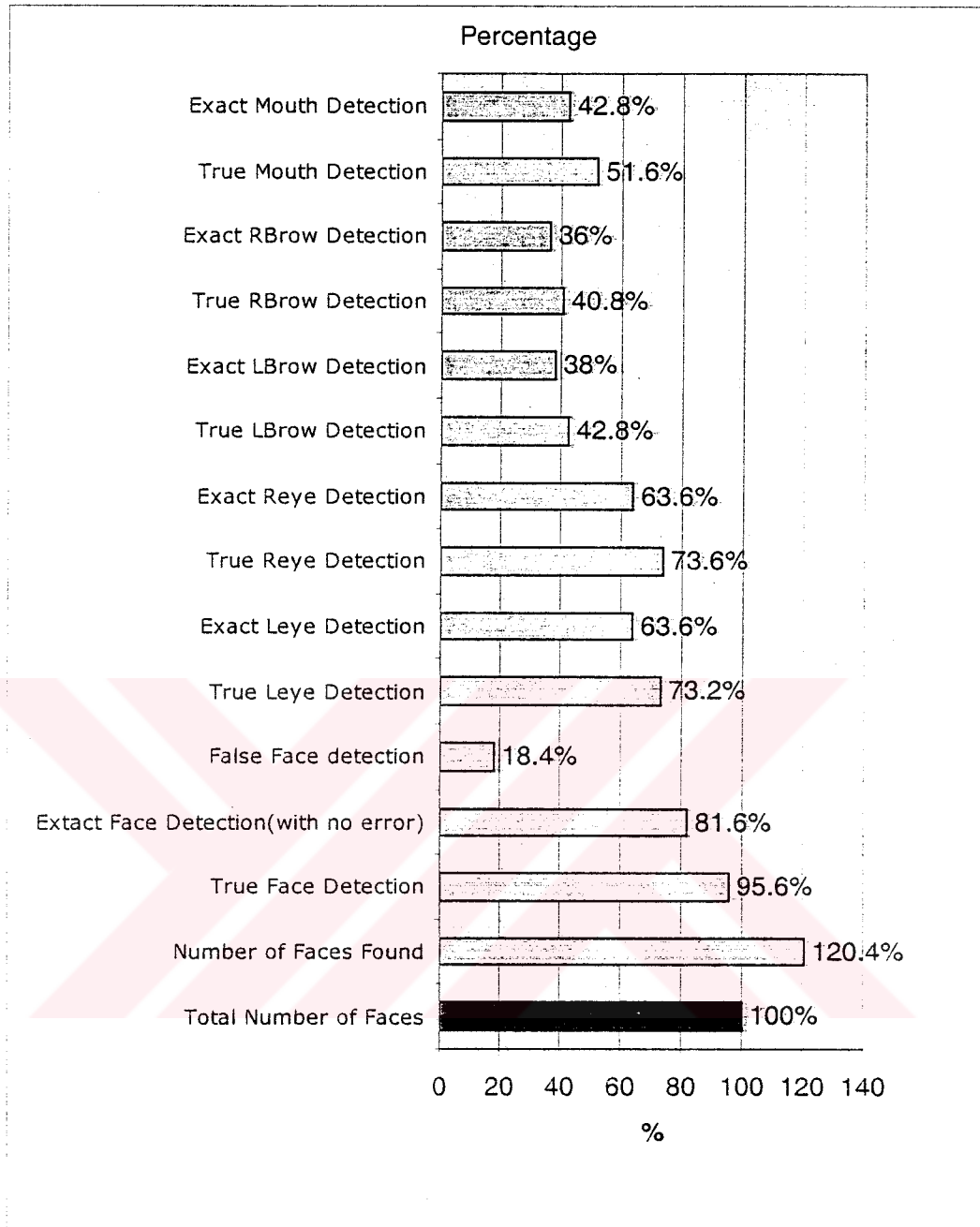
At last, we compute the exact number of features detected by system with no error. The exact detection rate is the rate where there is exactly one face found in the passport image. True detection rate is the rate where system finds more than one face instead of finding only one and at least one of the faces is a true face. Figure 5.1 and Figure 5.2 shows the result of single face image test.

After processing 250 images each of size 197×276 in 85 seconds, (340 milliseconds for each of them) we reached up to 81.6% exact face detection rate. Without considering false detections, this number increases to 95.6%.

For both of the eyes exact detection rate is 63.6% and true detection rate is 73.6%. For both of the eyebrows exact detection rate is 36% and true detection rate is 42.8%. Exact mouth detection rate is 42.8% and true detection rate is 51.6%.



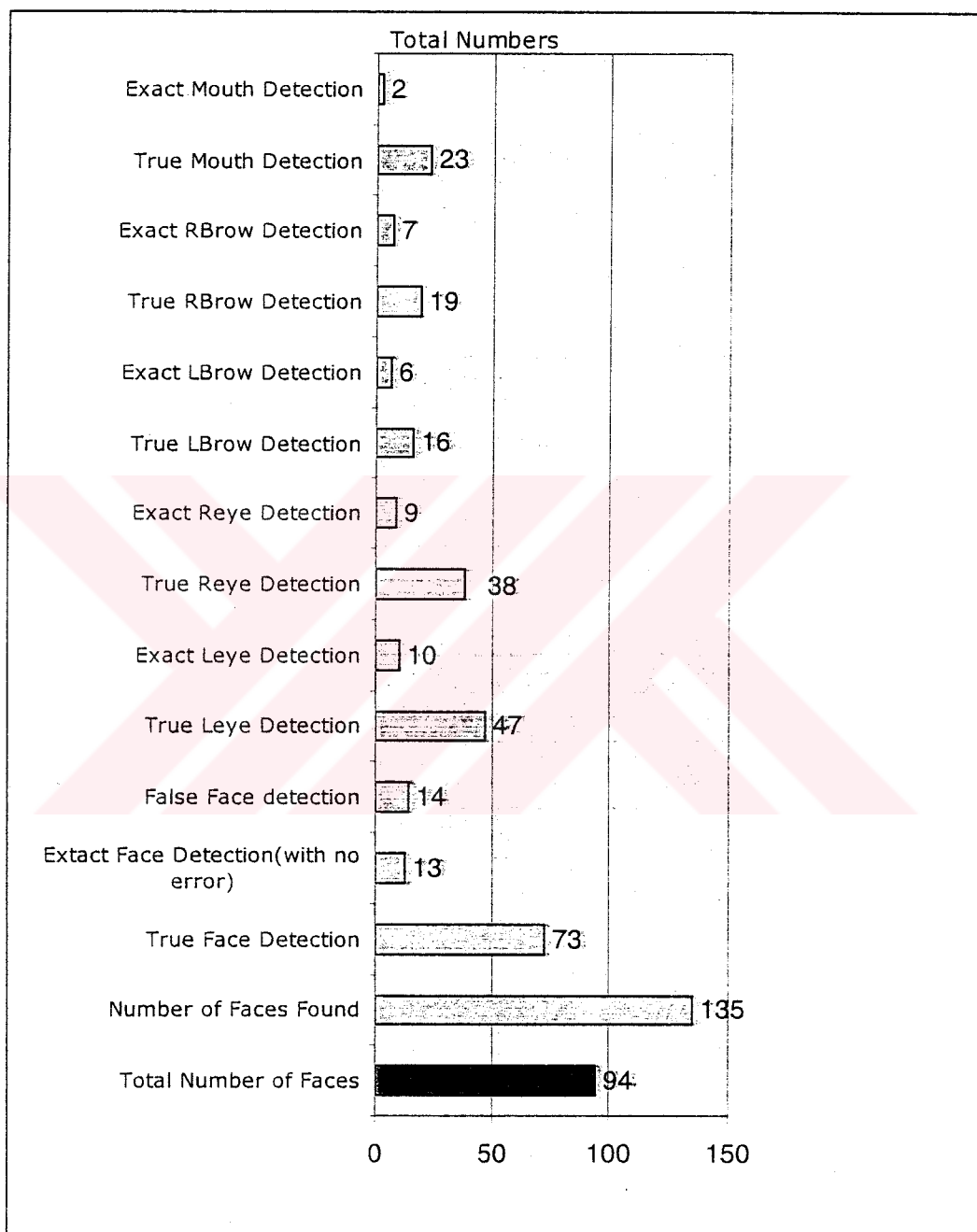
**Figure 5.1 Experimental Results of Face Detection on single face**



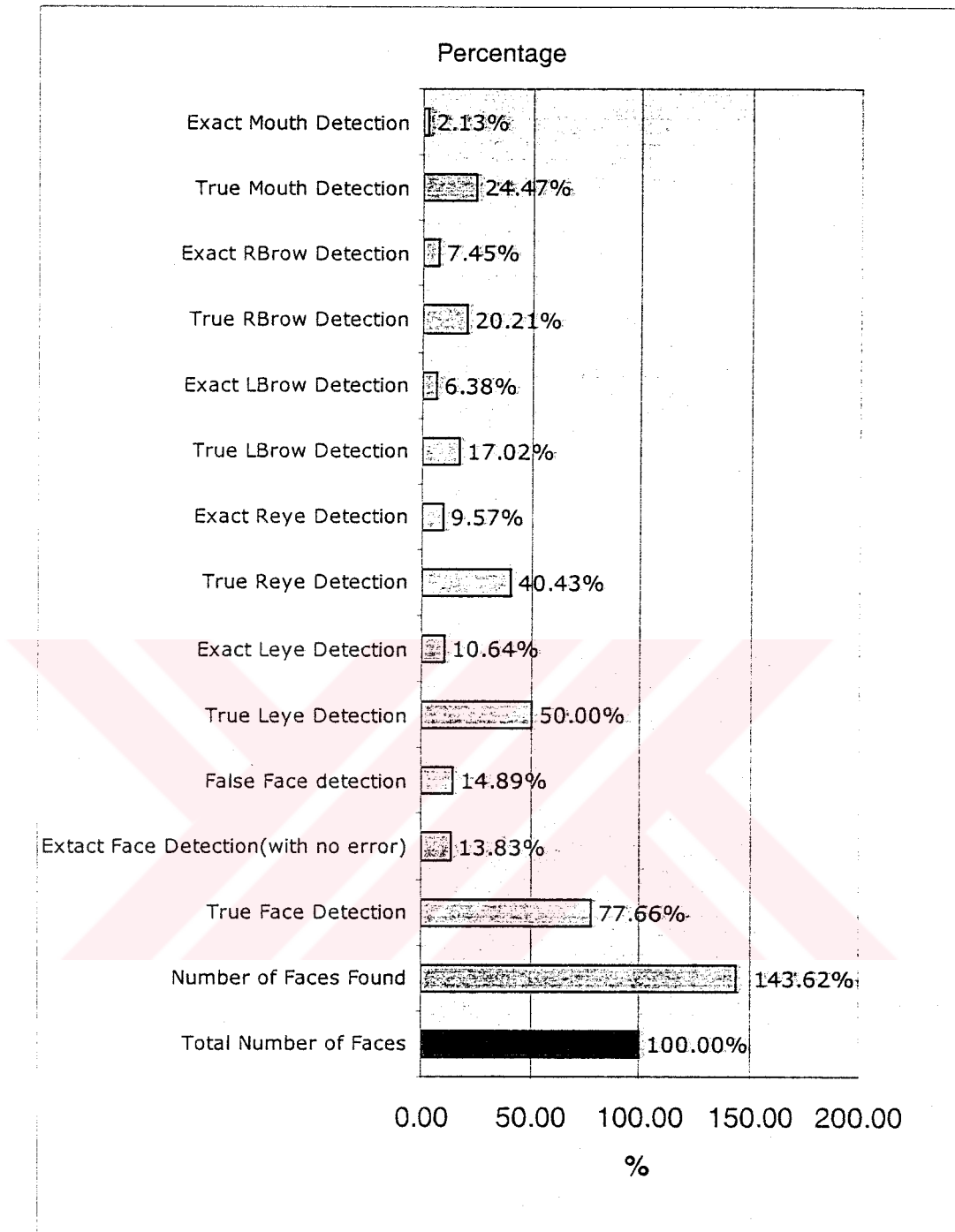
**Figure 5.2 Results of Face Detection on single face (in Percentages)**

For multiple-face images, 20 different sized randomly selected image with complex background from ADVIS 2001 conference is selected and following results. Observed results are shown in Figure 5.3 and Figure 5.4.

It is clear that exact detection rate decreases to 13%. This is because even if there are 20 exact detections exist and there is only one false detection occurs then; the complete processing result is set to false.



**Figure 5.3 Experimental Results of Face Detection on multiple faces**



**Figure 5.4 Results of Face Detection on multiple faces (in Percentages)**

According to the both of the single and multiple-face detection results, the algorithm usually found in average 20% more than the exact number of faces exist in images. In addition, number of true face detections that is eliminated because of the heuristic functions is almost 2% percent. This result shows that more heuristics

should be used to get results that are more robust. However, for both single and multiple-face detection results showed in Figure 5.2 the most robustly detected facial feature is eyes and the second is mouth.

## **5.2 Shot Detection Study**

### **5.2.1 Coding**

The software developed by using C++ language (g++ Version 2.96), built with libmpeg3 library and currently runs on a Red Hat Linux 7.0 machine via CGI interface. MPEG uses YCrCb instead of RGB color space, but libmpeg3 library functions provides a set of different color conversions so MPEG3\_RGB888 color model is used.

Then the program opens the selected mpeg file and checks its database status. If it is previously extracted then it just shows the extracted information including number of frames, frame rate, number of audio channels, audio frequency, etc and stops for further processing of the raw video file.

### **5.2.2 Interface**

The user gives the following information to the program via interface;

- Name of the MPEG formatted file
- Compression rate for JPEG images extracted from selected MPEG formatted video clip.
- $\alpha$  value as a threshold value (Default value is 0.25 )

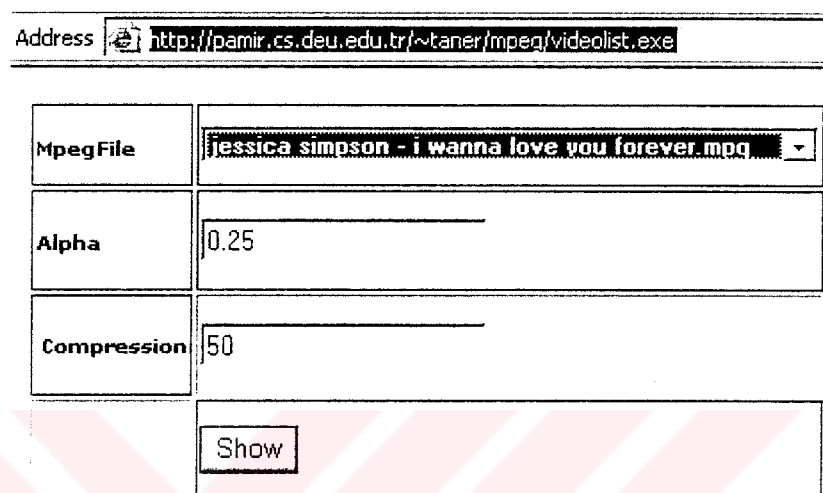
### **5.2.3 Shot Detection Study Results**

According to the published literature, comparing the shot boundary detection based methods is a difficult task. The first reason is that few studies report



quantitative results and the latter is that people are usually working on different set of video databases.

Figure 5.5, Figure 5.6 and Figure 5.7 shows example screenshots from the application.




Address  <a href="http://pamir.cs.deu.edu.tr/~taner/mpeg/videlist.exe">http://pamir.cs.deu.edu.tr/~taner/mpeg/videlist.exe</a>	
MpegFile	jessica simpson - i wanna love you forever.mpg
Alpha	0.25
Compression	50
<input type="button" value="Show"/>	

Figure 5.5 Initial Screen of Histogram-based shot detection preliminary work



jessica simpson - i wanna love you forever.mpg		
A U D I O	Has Audio	yes
	Total Audio Streams	1
	Audio Channels	2
	Sample Rate	44100
	Audio Samples	11287295
V I D E O	Has Video	Yes
	Total Video Streams	1
	Video Width	352
	Video Height	240
	Frame Rate	29.9700
	Total Video Frames	7670
Step 1: Calculating Frame to Frame Differences		Already Processed 
Step 2: Calculating Mean and Standard Deviation		Processed 
Mean		3897.792
Standard Deviation		142836.562
Alpha		0.800
Tb=mean*(alpha^standard deviation)		118167.047
Step 3: Detecting Video Shots		Processing

Figure 5.6 Result of Histogram Based Shot Detection Study.

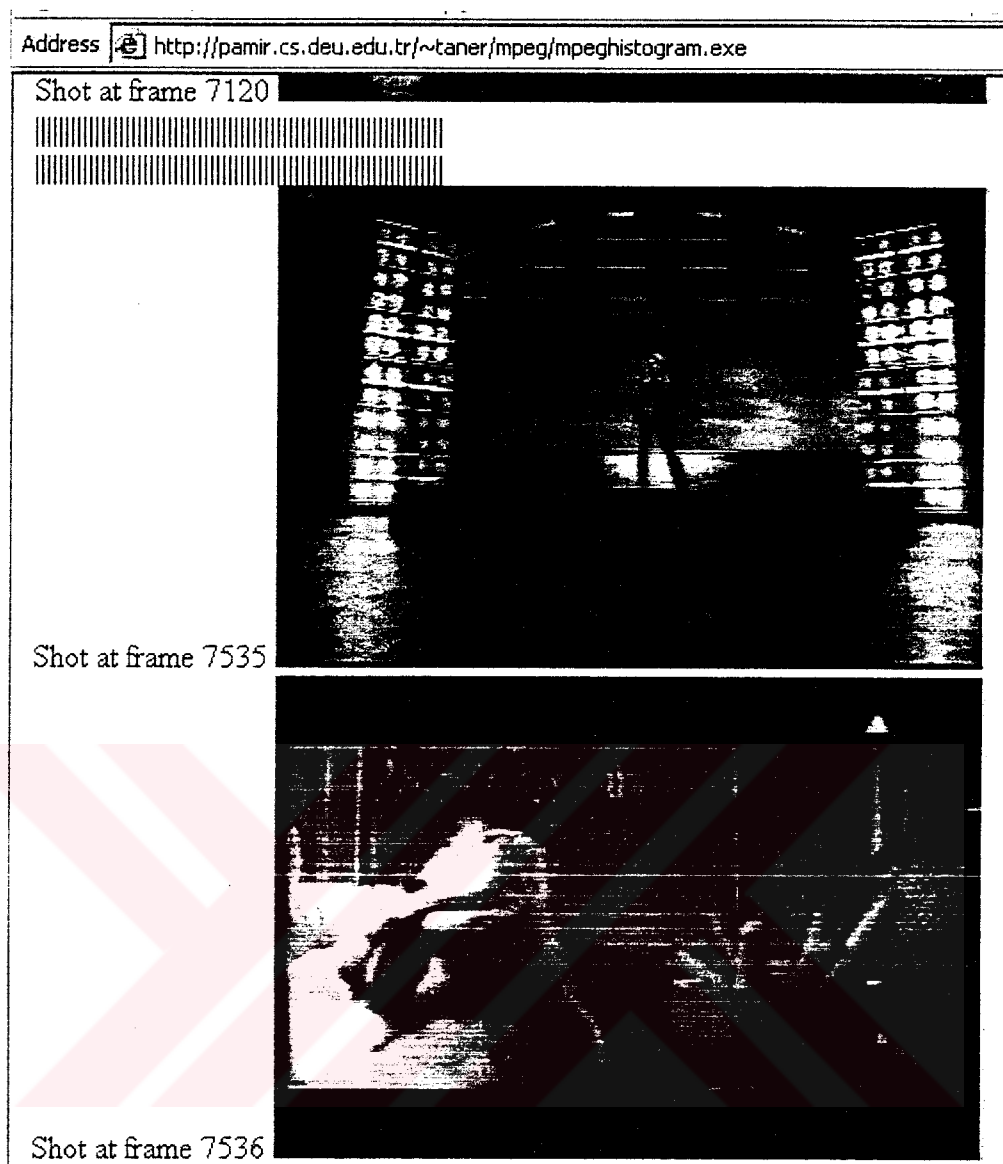


Figure 5.7 Example shot detection screen

### 5.3 Video Indexing and Retrieval

#### 5.3.1 Coding

In the content of this thesis, a video indexing and retrieval application is developed for only Windows platforms by using Borland C++ 5.0. Experimentations are collected on a standard workstation running on Windows XP Professional, operating system having Intel Pentium III 733 MHz Processor with 384MB Ram and

20GB harddisk drive. LifeView TV card is used to get real time video frames from a Sony Handycam TR425E video device. During the video capture process, 320×240 pixels PAL\_B video standard is used.

RGB color space is selected to operate with different video formats. It uses MediaPlayer component in order to extract RGB color information frames from uncompressed AVI formatted files and uses TJPEG component to read and write these types of files. TVideo component is used to capture video frames from the video devices.

### 5.3.2 Interface

Figure 5.8 shows design time appearance of the developed system and Figure 5.9 shows initial screen of the application. The system has three main functionalities. The first one is Real-Time Video Indexing, the second one is Offline Video Indexing and the last one is Video Retrieval.

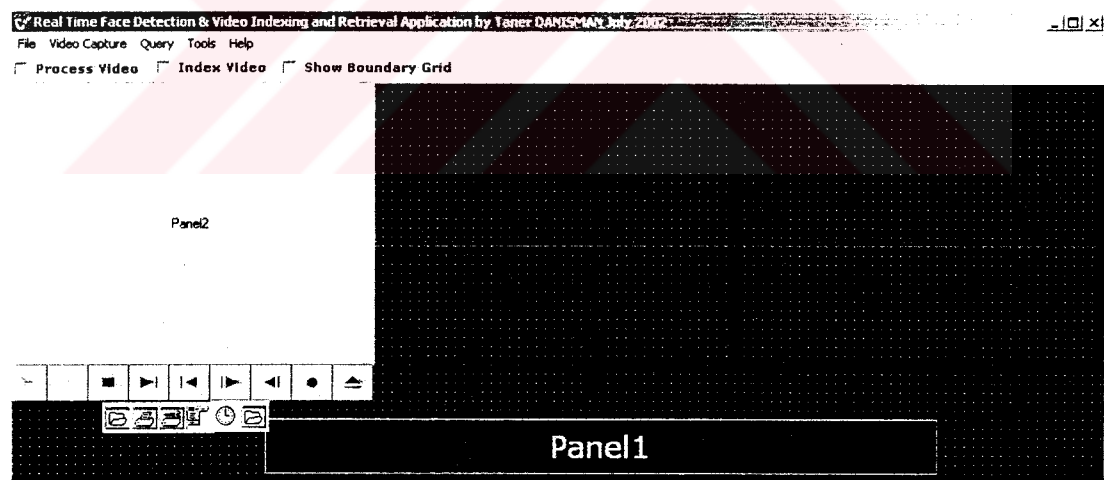
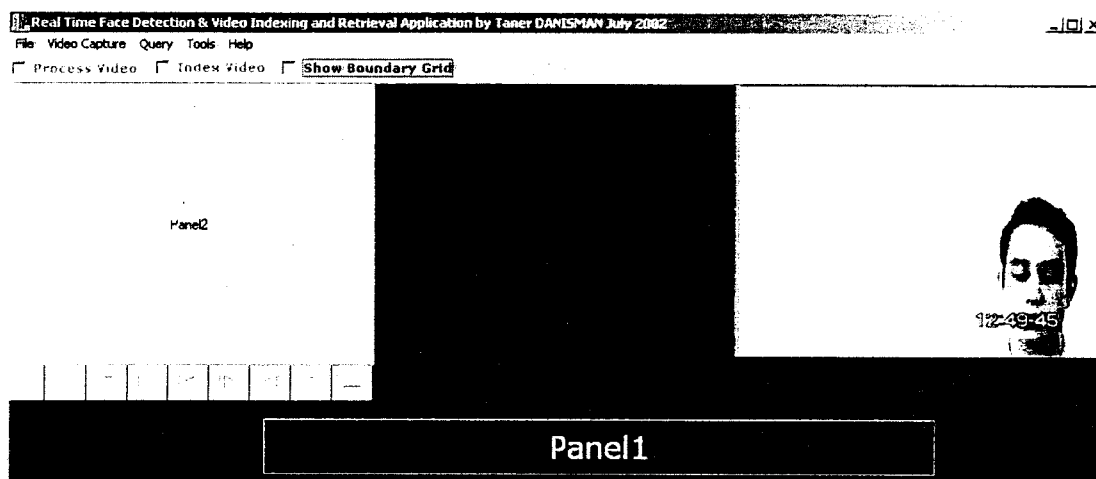


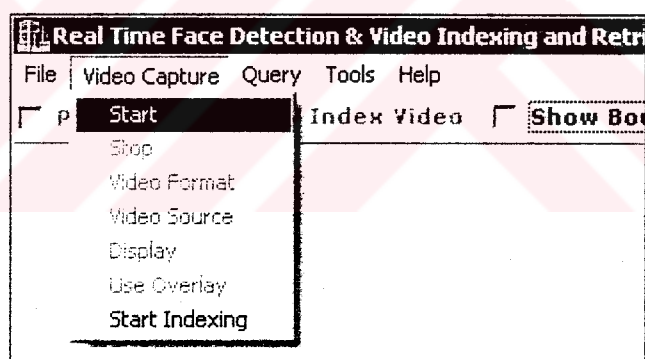
Figure 5.8 Design Time Components



**Figure 5.9 Initial Screen of the Application**

### 5.3.2.1 Real-Time Video Indexing

Real-Time video indexing mode can be activated by selecting **Video Capture->Start** from the upper menu bar Figure 5.10 shows the related actions.



**Figure 5.10 Starting video capture process**

Selecting this menu item enables the video device and opens the connection to this device. After a few seconds, frames from the video device are shown on the main form. Figure 4.10 shows the result of selecting this menu item.

In this work, 24-bit RGB color space is used to get more accurate results. To select video format go to menu item **Video Capture->Video Format**. Figure 5.11

shows the actions and video format selection screen. After selecting the video format following screen appears. The default value is 320x240 pixels with 24-bit RGB color. This selection makes the size of each frame to 230400 bytes.

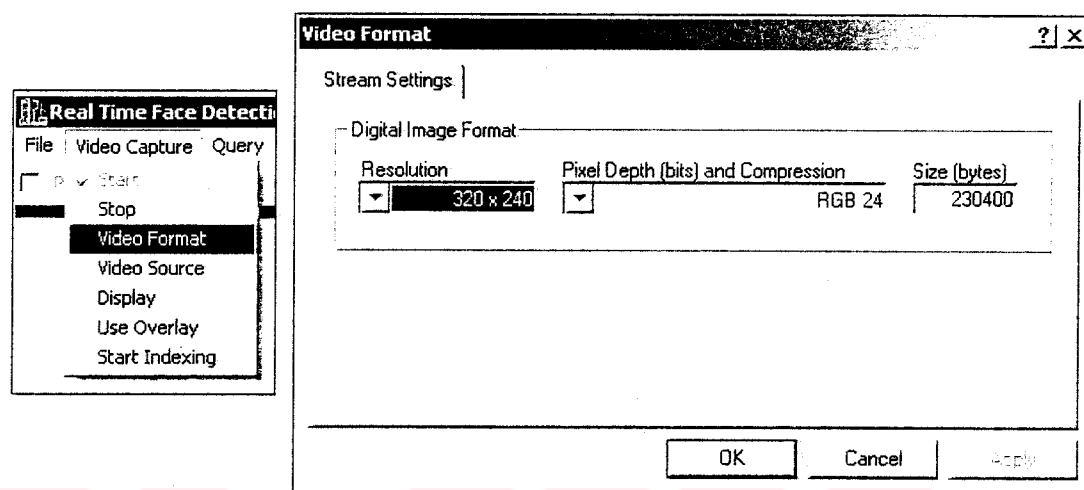


Figure 5.11 Selecting Video Format

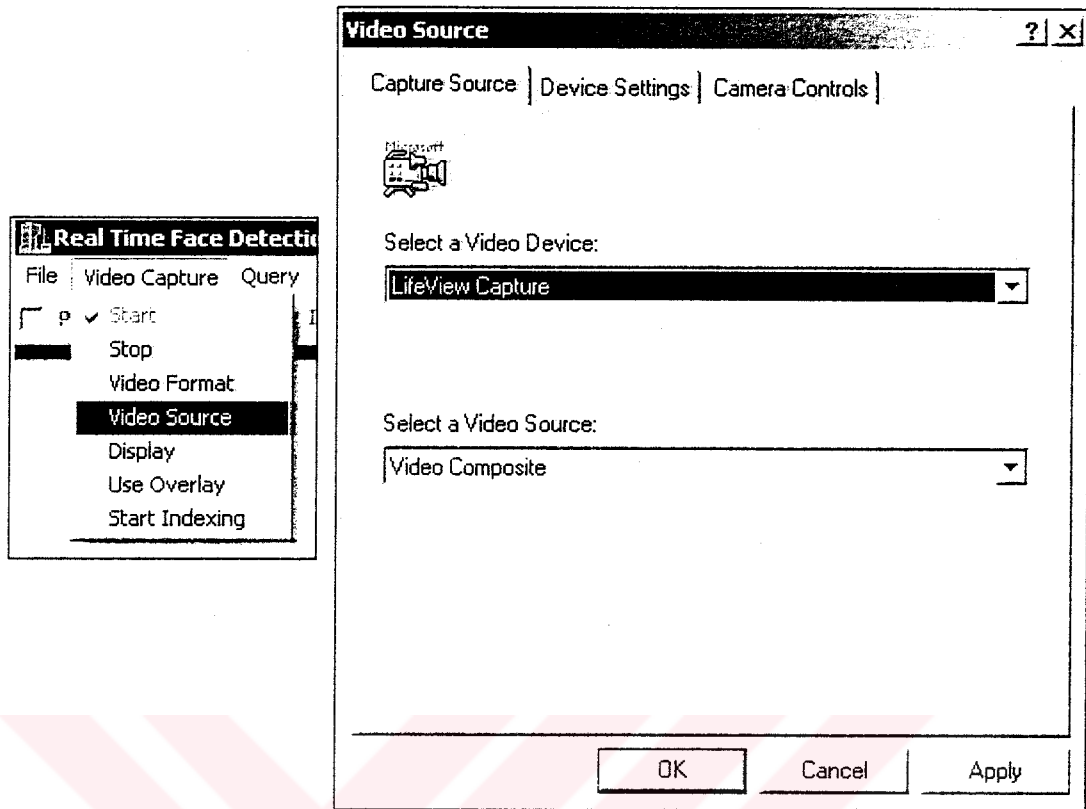


Figure 5.12 Selecting Video Source

In Figure 5.12 video source selection box is shown. To capture live video feeds from video camera **Video Composite** should be selected.

Another menu item is **Use Overlay**. It is used to get hardware accelerated video frames. Its frame rate is higher than the standard mode but it only changes the size and performance of video source. It does not affect the overall system performance.

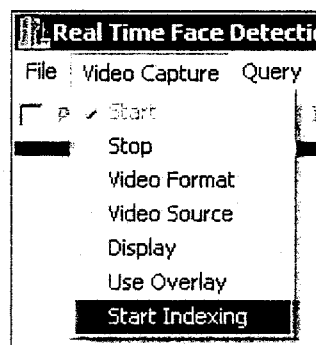
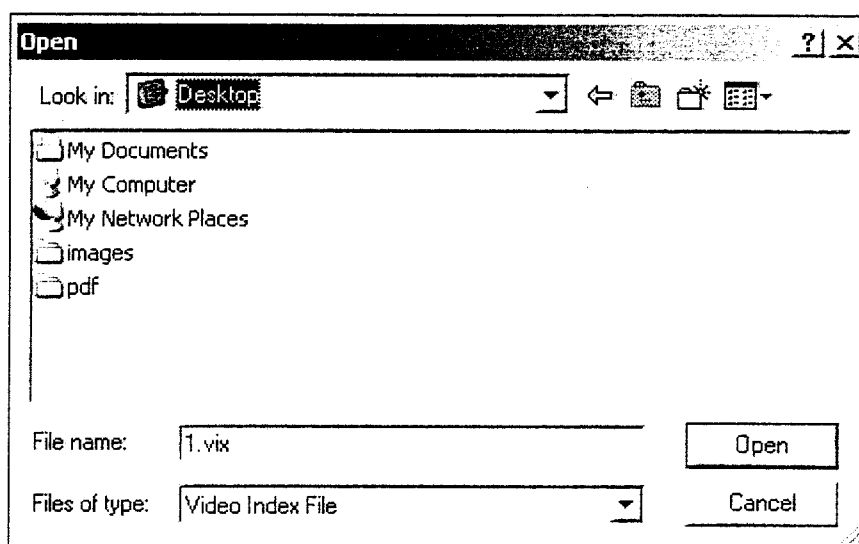
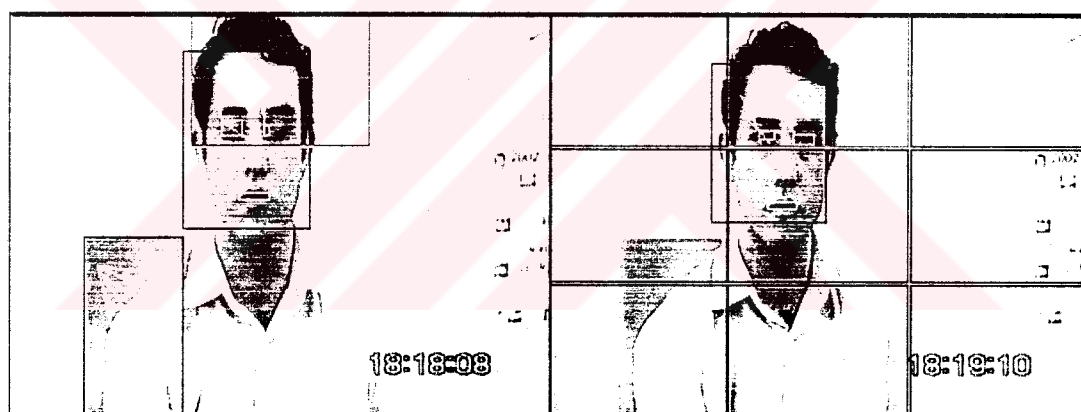


Figure 5.13 Indexing of Live Video feeds



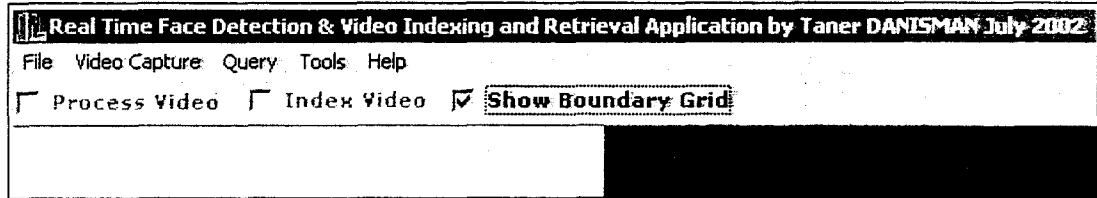
**Figure 5.14** Selecting a video index (vix) file



**Figure 5.15** Indexing of live video feeds (screenshot)

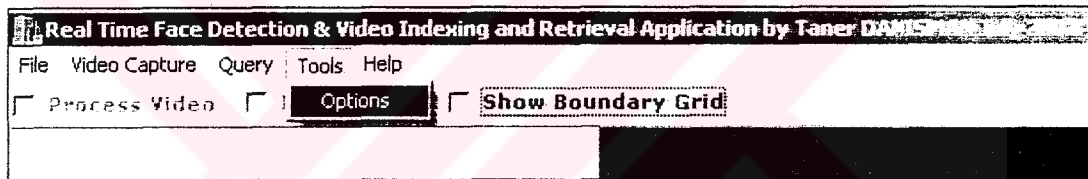
In order to index real time video frames, **Video Capture->Start Indexing** menu item should be selected as in Figure 5.13. After that, the program asks a file name that will be used to store summarized video information as in Figure 5.14. Then the main window is shown as in Figure 5.15 indicating that video index process has been started. If you select **Show Boundary Grid** checkbox as in Figure 5.16 then, we get an image like the right hand side image of Figure 5.15.





**Figure 5.16 Enabling boundary regions**

During the process of show boundary, both  $x$  and  $y$  dimensions of each video frame is divided into three equal parts. Boundaries of nine parts are painted with black color. As shown in Figure 5.15, during the indexing process, founded face location is painted in red color. Change in location of the red rectangular area also indicates that a shot boundary is detected. These parts are then used for face tracking process.



**Figure 5.17 Setting up frame rate**

In order to set up frame rate of the video device, **Tools->Options** menu item should be select as in Figure 5.17. Then the following screen appears. Changing and setting the frame rate value immediately effects the video source

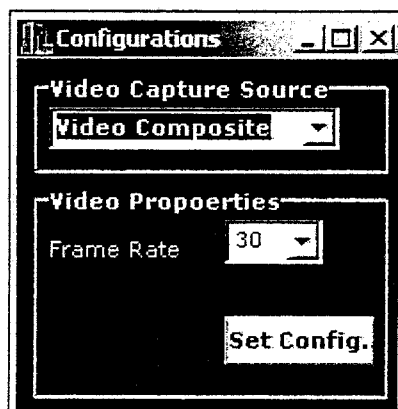


Figure 5.18 Setting up frame rate of video source

### 5.3.2.2 Offline Video Indexing

The second mode of the system is offline video indexing mode. This mode can be activated by selecting **File->Open** from the upper menu bar or just pressing F3 button. Figure 5.19 shows the related actions.

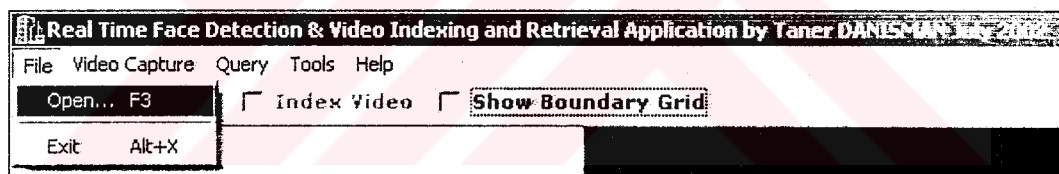
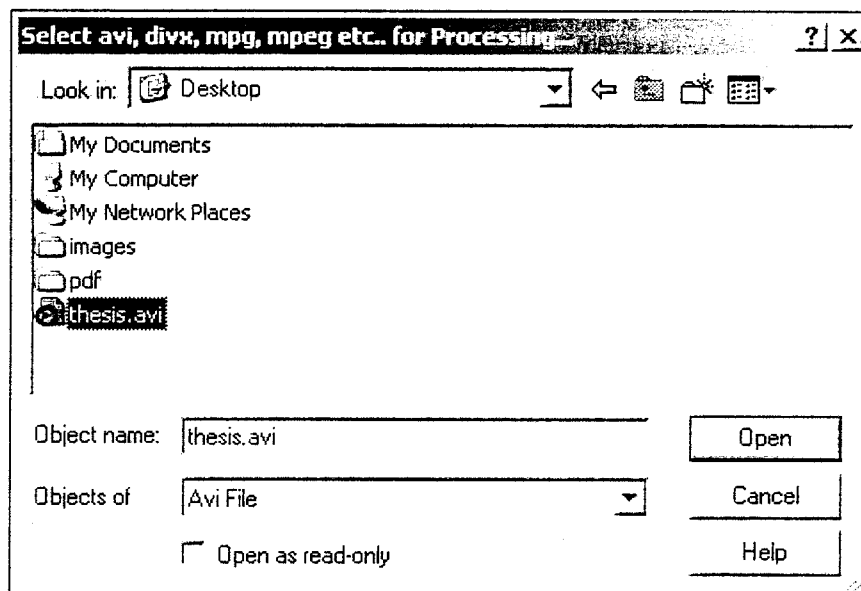


Figure 5.19 Offline Video Indexing (Open File)

Then the following screen appears.



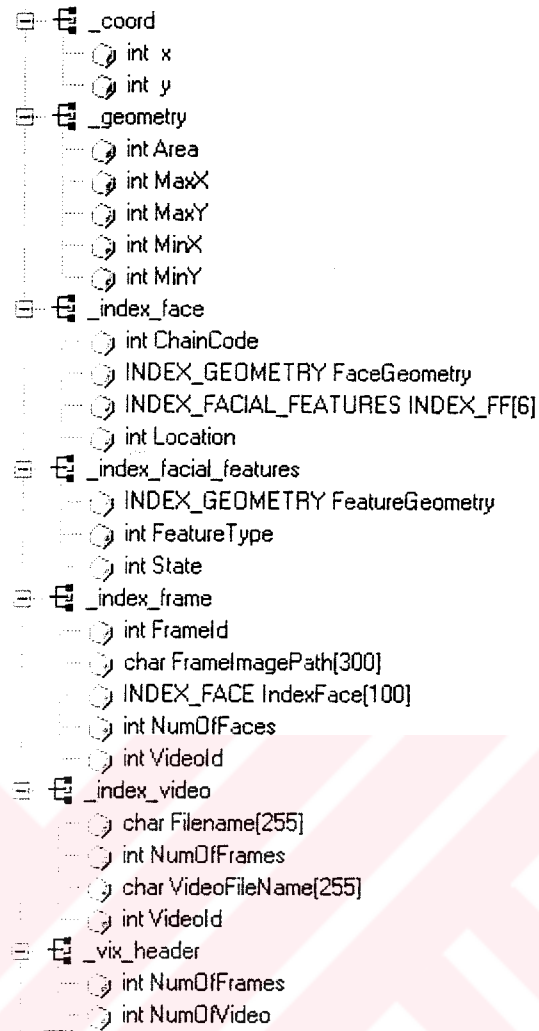
**Figure 5.20 File open dialog box**

After selecting an AVI file the system is ready to operate with the file. Next step can be just playing the selected AVI file or both playing, processing and indexing the file. The options to do the next step are placed on the top of the main form, under the menu bar. Here;

When the user checks the **Process Video** checkbox then, during the video playing each captured frame is processed against finding human faces and facial features.

When the user checks the **Index Video** checkbox then, the user is wanted to select a video index file to store the summarized information. The file extension is “.vix”. This process is same as real-time video indexing process described in the previous section.

A simple indexing structure is used to store and retrieve video information. Figure 5.21 shows the typical structures defined in application;



**Figure 5.21 Video retrieval typical structures**

According to this structure, each video has its own:

- **VideoId:** It is a unique number that represents the video itself.
- **FileName:** Name of the video index (vix) file associated to the video file.
- **VideoFileName:** It is the name of the video file. It can have extensions like .AVI, .MPG, etc...
- **NumberOfFrames:** It is the number of frame the video file *VideoFileName* has.

Each Video Frame has the following properties:

- **FrameId:** It is a unique original video frame Id number that, represents the frame

- **FrameImagePath:** It is a path to the binary frame image.
- **NumberOfFaces:** It is the number of faces found by the face detection algorithm within this frame represented by **FrameId**
- **VideoId:** It is the unique id of video file, where this frames belongs.
- **IndexFace:** It is a set of faces found in the frame image.

Each Face has the following information:

- **FaceGeometry:** It is a kind of struct that holds the minimum and maximum  $(x, y)$  coordinates of the face with respect to the frame image.
- **Index\_FF:** A number of facial features that is detected by system and their corresponding values.
- **Location:** It is the location information of the face and can be an integer value ranging from zero to eight.

Last, each facial feature has its own:

- **FeatureGeometry:** It is same as **FaceGeometry** value
- **FeatureType:** It is an integer value ranging from zero to five. It represents facial information type. Defined as macro types are MOUTH, LEYE, REYE, LBROW, RBROW; NOISE.
- **State:** It is the state of the facial feature and can be integer values such as OPENED, CLOSED, UNKNOWN. These macros are defined in corresponding header files. For example for the mouth, the state can be OPENED or CLOSED. The same is true for eyes.

Figure 5.22 shows the design time controls of the video retrieval form of the application.

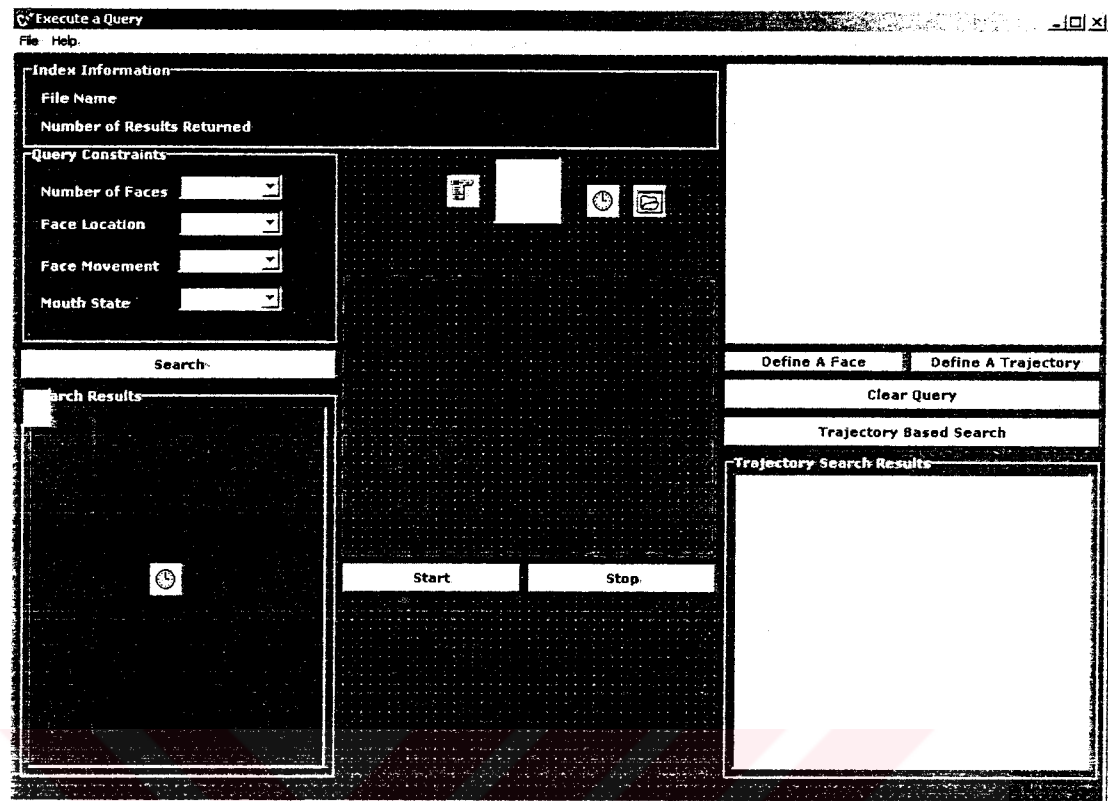


Figure 5.22 Video Retrieval form in design time

The system allows the user to select one of the two video querying solutions. The first solution is to use almost text-based query generating system. Figure 5.23 represents how to start video retrieval process. The system works as follows;

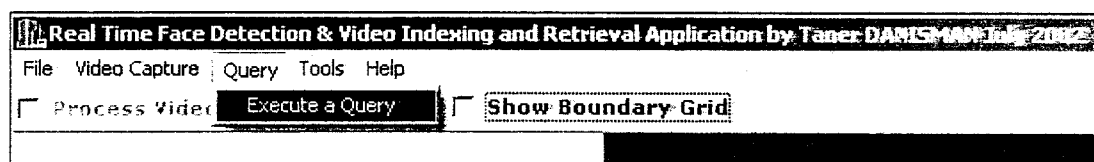


Figure 5.23 Execute a Query

In the first step user selects **Query->Execute a Query** menu item from the menu bar as in Figure 5.23 and then following screen appears.

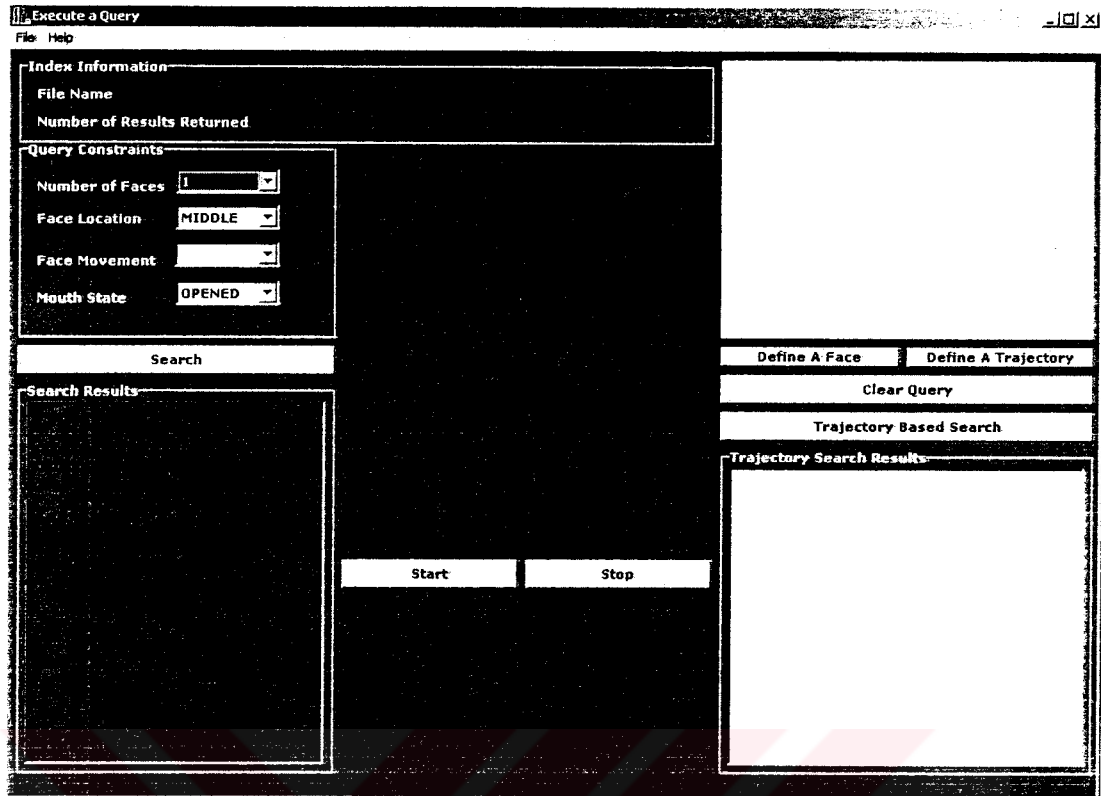


Figure 5.24 Initial Screen of Video Retrieval form

In order to make a query first a video index (vix) file should be opened via. **File->Open** menu item. Figure 5.25 shows the related action.

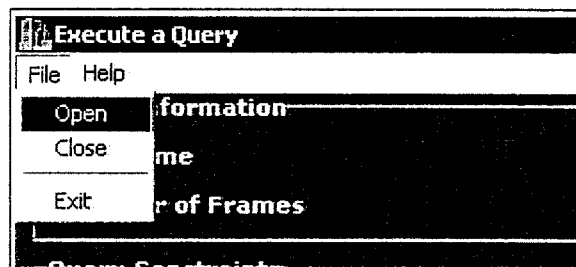


Figure 5.25 Selecting a video index (vix) file

By using the first textual method, the user is able to ask the system to;

- Retrieve human faces that are located in a specific place in the frame. The user is able to select the location of the face via a drop-down list as seen in Figure 5.26. The user has also a chance to ignore the location of face in each

frame by selecting IGNORE value from the corresponding drop-down list. The default value is MIDDLE.

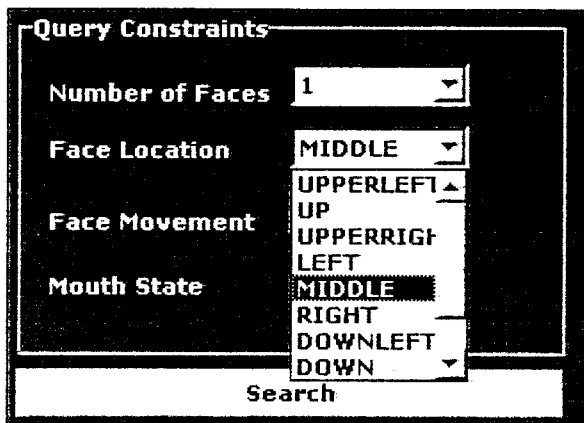


Figure 5.26 Text-based video retrieval

- The user has also a chance to give the state of the mouth facial feature. It can take three parameter, UNKNOWN, OPENED, CLOSED and IGNORE again. It is shown in Figure 5.27.

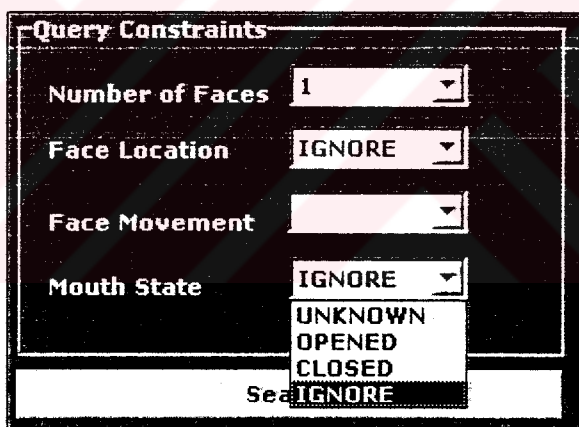


Figure 5.27 Giving mouth state

If the user chooses the both constraint to the IGNORE value then the whole database is returned in case of a query execution. After determining the query constraints then we are ready to process our query by simply pressing the Search button. Figure 5.28 shows the result screen of executing the query having constraints such that Face Location is MIDDLE and mouth state is CLOSED.



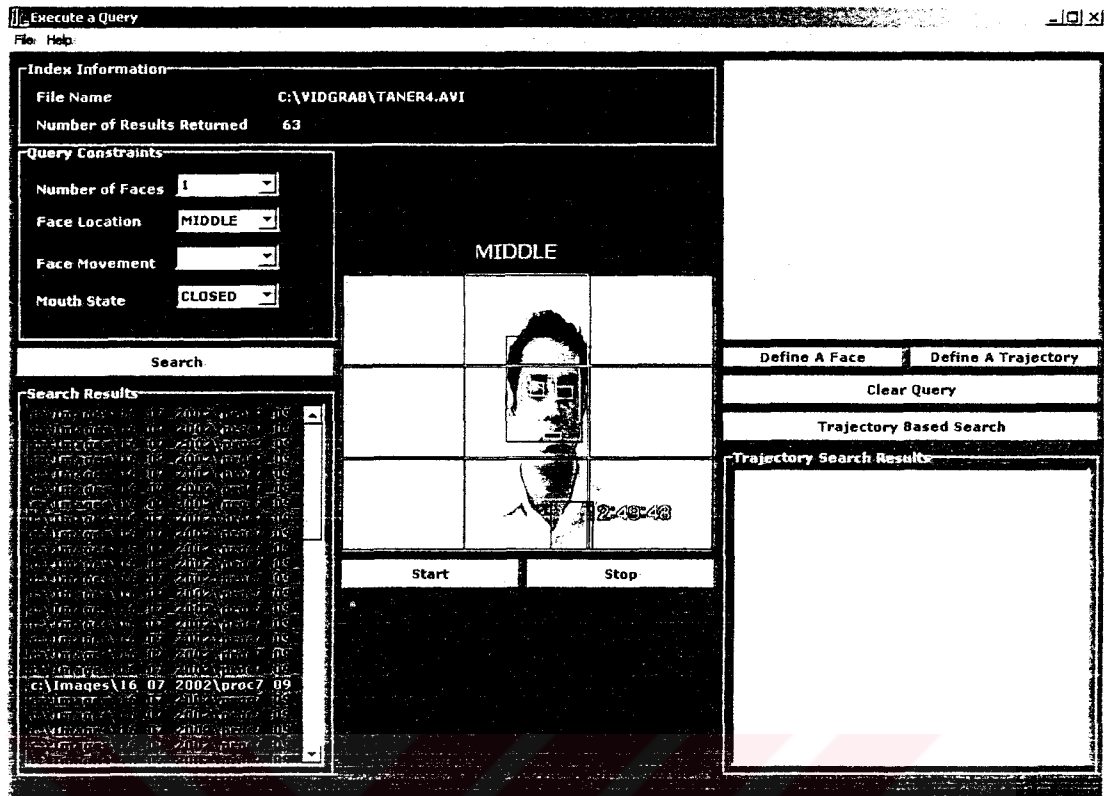


Figure 5.28 Executing a text-based query

On the top of the screen, number of results returned value is stored. After completing the query, the system automatically shows the result set in time order. There are two additional button placed at the bottom of the result image captioned with **Start** and **Stop** respectively. These buttons are used to start and stop showing the result set in sequential order. User can select an individual result by clicking the associated list box item.

In case of trajectory based query, first, the user presses define a face button and then clicks on the trajectory board. After that, clicks to the Define a Trajectory button and clicks to the face. After that, the user moves the mouse on the trajectory board until he/she re-clicks to the board. The mouse movements are logged and locations where mouse is moved are computed. Figure 4.16 shows the result of finding locations. Boundaries of the locations are painted with red colored rectangles, while the user-defined trajectory is in black color.

When the user presses the trajectory based search button, the system looks for faces that satisfy the given trajectory. The rest of the system works like the text-based method and retrieves a set of images satisfying the constraint. Show the result of executing the trajectory query.

In fact, the results returned from the trajectory-based search can be defined as user defined shot regions where the user initiates the start point of the shot boundary and ends wherever he/she wants.

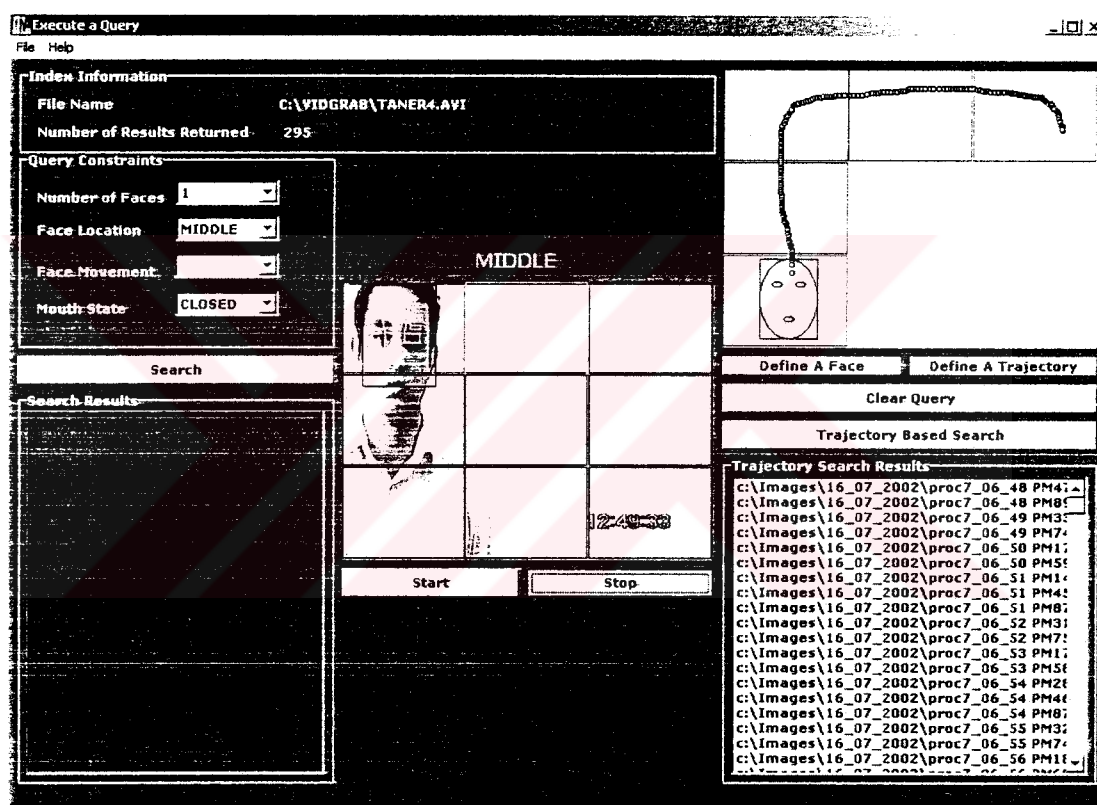


Figure 5.29 Result of trajectory query

### 5.3.3 Video Indexing and Retrieval Study Results

Text based indexing and retrieval part of the study is based on both face detection and shot detection studies. Therefore, result of textual indexing and retrieval is same as those studies. In addition, trajectory based retrieval differs from textual retrieval. Trajectory based queries allows us to give a trajectory thus enabling object motion

tracking. In this study, each frame is divided into nine equal rectangles. This approach is enough to track objects via trajectories when the object size is close to the shot region size. However, number of shot regions should be increased to improve the accuracy of object tracking when the size of the object is smaller than the shot region. The more shot regions give the higher error-free object tracking.



---

## CHAPTER SIX

# CONCLUSION

---

Recently there is a serious increase in digital video data. This is because, VCD, and DVD prices are getting lower day by day. However, without making a well-organized structure of these data, they will just be garbage of serial access binary data. As a result, automatic video indexing and retrieval is an active research area because of increasing need of tools that performs automatic indexing and retrieval of video data.

This thesis is about automatic extraction, summarization and retrieval of human facial information from raw video files. The thesis presents a system developed in Borland C++ Builder 5.0 and can operate on AVI and MPEG files as well as real-time live video feeds from video camera for automatically detecting, extracting, indexing and retrieval of human facial information from raw video. It also covered video indexing standards namely called Dublin Core and MPEG-7.

The system captures from real-time video frames at 25fps on an Intel Pentium III 733Mhz machine having 384 MB of Ram. Time required extracting human facial information and indexing operation takes in average about 0.33 seconds (3 fps). This information is stored in a database of video frames. Variety of lighting conditions affects the face detection rate therefore, our system did not achieve to the performance of Neural Network or Hidden Markov Model based systems but can be used to speedup these models. Several extensions can be added in order to increase the performance of the current architecture. The system can also be used with a Neural Network based or Hidden Markov Model based application. This will lead us to more robust and accurate results.

There are a number of keyframe selection strategies such as getting the first frame, getting the  $k^{\text{th}}$  frame or even computing an average frame and selecting the closest frame to the average frame. Furthermore, object movement-based shot detection method can be used to select candidate keyframes. This will lead us to decrease the amount of space used for indexing the video data.



---

## REFERENCES

---

- Ahanger, G., & Little, T.D.C., (1995), A Survey of Technologies for Parsing and Indexing Digital Video, Journal of Visual Communication and Image Representation
- Arman, F., Hsu, A., & Chiu, M-Y., (1994), Image Processing on Encoded Video Sequences, Multimedia Systems Vol. 1, No. 5, pp. 211-219
- Boreczky, John S., & Rowe , Lawrence A., (1994), Comparison of video shot boundary detection techniques, Berkeley Multimedia Research Center
- Chen, L. S., (2000) Joint processing of audio-visual information for the recognition of emotional expressions in human-computer interaction, PhD dissertation, University of Illinois at Urbana-Champaign, Dept. of Electrical Engineering
- Cohen, Ira, (2000), Automatic Expression Recognition from Video Sequences Using Temporal Information, Msc Thesis, University of Illions, pp. 8-30
- Cohen, Ira, Garg, Ashutosh, & Huang, Thomas S., (2000), Emotion Recognition from Facial Expressions using Multilevel HMM, NIPS Workshop on Affective Computing, Colorado
- Dublin Core Metadata Element Set,( 1999), Version 1.1: Reference Description
- Ekman, P. & Friesen, W. V., (1978), Facial Action Coding System. Consulting Psychologists Press Inc., 577 College Avenue, Palo Alto, California 94306

- Franco, Leonardo, & Treves, Alessandro, (1997), A Neural Network Facial Expression Recognition System using Unsupervised Local Processing Cognitive Neuroscience Sector, SISSA
- Hampapur, A., Jain, R., & Weymouth, T., (1994), Digital Video Segmentation, Proc. ACM Multimedia 94, San Francisco, CA, October, pp. 357-364
- Han, Chin-Chuan, Liao, Hong-Yuan Mark, Yu, Kuo-Chung, & Chen, Liang-Hua, (1997), Fast Face Detection via Morphology-based Pre-processing, Academia Sinica Institute of Information Science
- Hjelmås, Erik, Lerøy, Christel Beate, & Johansen, Henry, (1998), Detection and Localization of Human Faces in the ICI System: A First Attempt, Department of Electrical Engineering and Science, Gjøvik College
- Lien, Jen-Jier James, (1998), Automatic Recognition of Facial Expressions Using Hidden Markov Models and Estimation of Expression Intensity, The Robotics Institute, Carnegie Mellon University
- Little, T.D.C, Ahanger, G., Folz, R.J., Gibbon, J.F., Reeve, F.W., Schelleng, D.H., & Venkatesh, D., (1993), A Digital On- Demand Video Service Supporting Content-Based Queries, Proc. ACM Multimedia 93, Anaheim, CA, August, pp. 427-436.
- Martinez, Jose, M., Introduction to Mpeg-7, ISO/IEC JTC1/SC29/WG11 N3545
- Mehrabian, A., (1968), Communication without words, Psychology Today 2, pp. 53-56
- Nagasaka, A. & Tanaka, Y., (1992), Automatic Video Indexing and Full-Video Search for Object Appearances, Visual Database Systems II, Knuth, E., Wegner, L., Editors, Elsevier Science Publishers, pp. 113-127

- Samaria, F., & Fallside, F., (1993), Face segmentation for identification using Hidden Markov Models, British Machine Vision Conference
- Smith, M., & Kanade, Takeo, (1996), Skimming for quick browsing based on audio and image characterization, CMU-CS-96-186R, Carnegie Mellon University
- Smith, M. A., & Kanade, Takeo, (1997), Video skimming and characterization through the combination of image and language understanding techniques, Computer Vision and Pattern Recognition, pp. 775-781, San Juan, Puerto Rico
- Swanberg, D., & Shu C.F., & Jain, R., (1993), Knowledge Guided Parsing and Retrieval in Video Databases, in Storage and Retrieval for Image and Video Databases, Wayne Niblack, Editor, Proc. SPIE 1993, pp. 173-187.
- Tao, H., & Huang, T. S., (1998.), Connected vibrations: A modal analysis approach to non-rigid motion tracking, Proc. IEEE Conference on Computer Vision and Pattern Recognition,(CVPR'98), (Santa Barbara, CA, USA), June 23-25
- Terzopoulos , D., & Waters, K., (1993), Analysis and Synthesis of Facial Image Sequences Using Physical and Anatomical Models. IEEE Trans. Pattern Analysis and Machine Intelligence, pp. 569-579
- Ueda, H., Miyatake, T., & Yoshizawa, S., (1991), IMPACT: An Interactive Natural-motion-picture Dedicated Multimedia Authoring System, Proceedings of CHI, (New Orleans, Louisiana, Apr.-May, 1991) ACM, New York, 1991, pp. 343-350
- Yang, Jie, & Waibel, Alex, (1996), A real-time face tracker. Third IEEE Workshop on Applications of Computer Vision, pp. 142-147, Sarasota, Florida, USA
- Yeo, B.L, (1996), Efficient Processing of Compressed Images and Video, PhD Thesis, Dept. of Electrical Engineering, Princeton University



- Pantic, M., & Rothkrantz, L.J.M., (2000), Expert system for automatic analysis of facial expressions, Image and Vision Computing, pp. 881-905
- Rowley, Henry A., Baluja, Shumeet, & Kanade ,Takeo, (1996), Neural Network-Based Face Detection, School of Computer Science, Carnegie Mellon University
- Rowley, Henry A., (1999), Neural Network-Based Face Detection, CMU-CS-99-117, PhD Thesis, School of Computer Science, Carnegie Mellon University
- Satoh, Shin'ichi, & Kanade, Takeo, (1996), Name-it: Association of face and name in video. CMU-CS-96-205, Carnegie Mellon University
- Satoh, Shin'ichi, & Kanade, Takeo, (1997), Name-it: Association of face and name in video. Computer Vision and Pattern Recognition, pp. 368-373, San Juan, Puerto Rico, June
- Sehti, I.K., & Patel, N.,A, (1995), Statistical Approach to Scene Change Detection, Proceedings of SPIE Storage and Retrieval for Image and Video Databases III, number 2420 in SPIE, San Jose, CA, USA
- Shahraray, B., (1995), Scene Change Detection and Content-Based Sampling of Video Sequences, Digital Video Compression: Algorithms and Technologies, Arturo Rodriguez, Robert Safranek, Edward Delp, Editors, Proc. SPIE 2419, February, pp. 2-13
- Samaria, F., (1994), Face Recognition Using Hidden Markov Models, PhD thesis, University of Cambridge
- Samaria, F., & Harter, A., (1994), Parameterization of stochastic model for human face identification", Proceedings of the Second IEEE Workshop on Application of Computer Vision, pp.138-142

Zabih, R., Miller, J., & Mai, K., (1993), A Feature-Based Algorithm for Detecting and Classifying Scene Breaks, Proc. ACM Multimedia 95, San Francisco, CA, November, pp. 189-200

Zhang, H.J., Kankanhalli, (1993), A., and Smoliar, S.W., Automatic Partitioning of Full-motion Video, Multimedia Systems (1993) Vol. 1, No. 1, pp. 10-28.

Zhong, D. & Zhang, H.J. & Chang, Shih-Fu, (1996), Clustering methods for video browsing and annotation, Storage and Retrieval for Still Image and Video Databases IV, IS&T/SPIE's Electronic Imaging: Science & Technology 96 [2670-38]

