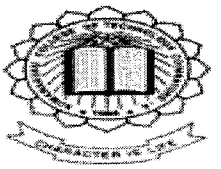


REAL TIME FACE DETECTION AND FACIAL EXPRESSION RECOGNITION

A PROJECT REPORT

Submitted by

ISWARYA.S	**71203104013**
RAVISHANKAR.R	**71203104035**
VIDHYA.S.V	**71203104054**

In partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE & ENGINEERING

KUMARAGURU COLLEGE OF TECHNOLOGY

COIMBATORE-641006

ANNA UNIVERSITY :CHENNAI 600 025

MAY 2007

ANNA UNIVERSITY : CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report **"REAL TIME FACE DETECTION AND FACIAL EXPRESSION RECOGNITION"** is the bonafide work of **"ISWARYA.S, RAVISHANKAR.R** and **VIDHYA.S.V"** who carried out the project work under my supervision.

SIGNATURE

Dr.S.THANGASAMY

HEAD OF THE DEPARTMENT

Computer Science and Engineering
Kumaraguru college of Technology
Coimbatore-641006.

SIGNATURE 20.4.07

Ms.S.RAJINI

SUPERVISOR

SENIOR LECTURER

Computer Science and Engineering
Kumaraguru college of Technology
Coimbatore-641006

Submitted for viva-voce examination held on 24.4.2007

24/4
Internal Examiner

24/4/07
External Examiner

DECLARATION

We hereby declare that the project entitled "Real time face detection and facial expression recognition", is a record of original work done by us and to the best of our knowledge, a similar work has not been submitted to Anna University or any other institution for fulfillment of the requirement of the course study.

This report is submitted in partial fulfillment of the requirements for the award of the degree of Bachelor of Computer Science and Engineering of Anna University, Chennai.

(ISWARYA.S)

Place: Coimbatore

Date: 20 - 4 - 2007

(RAVISHANKAR.R)

(VIDHYA.S.V)

ACKNOWLEDGEMENT

We are extremely grateful to **Dr.Joseph Thanikal,** Principal, Kumaraguru College of Technology, for having given us a golden opportunity to embark on this project.

We are deeply obliged to **Dr.S.Thangasamy,** Head of the Department, Department of Computer Science and Engineering, Kumaraguru College of Technology for his valuable guidance and useful suggestions.

We are grateful to **Mrs.P.Devaki,** Assistant professor, Department of Computer Science and Engineering, Kumaraguru College of Technology for her encouragement and support at various levels of this project work.

We also express our sincere thanks to **Ms.S.Rajini,** Senior lecturer, Project guide, Department of Computer Science and Engineering, Kumaraguru College of Technology for her valuable guidance and encouragement at every stage of this project.

Most of all, we thank our parents and friends for their blessings, help and support without which we would not be able to do anything.

ABSTRACT

Detecting facial features and identifying features is an important part of computer vision applications such as robotics, forensics, psychiatrics, application interfaces, etc.

While the human visual system can recognize various different kinds of objects very easily, visual recognition is generally a difficult task for computers. A general and comprehensive solution to the problem should be the ability to cope with the variability of object appearance in the scene, for example changing facial features.

The first step in emotion recognition is to segregate the facial and non facial features. Often this is done by using a multi-layered perceptron trained with the backpropagation of error algorithm on a subset of all facial expressions and subsequently tested on unseen face images.

Such approaches have three major disadvantages:

1. They are rather complex and therefore computationally expensive, if the test images are reasonably detailed.
2. The generalization performance on new images varied from 40% to 80% depending on the choice of the test images.
3. The introduction of the preprocessing stage to improve the generalization performance slowed down the learning by a factor of ten.

In the case of emotion recognition analysis a straight forward way to find facial expression is template matching. This is usually done by analyzing the variability of feature points. Once the facial features are identified expressions can be recognized easily.

TABLE OF CONTENTS

INTRODUCTION

1. INTRODUCTION

Real time face detection is becoming mandatory for many applications, which attracted much research work in this area. On this basis, many opportunities for new applications, and many others could be fully exploited. Certain systems that were infeasible with traditional techniques can now be implemented. Several examples are observed, such as in the areas of robotics, forensics, medical diagnosis and hospitality industry.

During the last few years, numerous architectures and algorithms for face recognition and expression recognition from static facial images have been proposed. A general distinction between feature and template-based approaches has been described but psychological experiments indicate that the human visual system process faces at least to some extend holistically, favoring template-based approaches over feature-based techniques for their biological validity (Feature Points).

Inter-personal human communication includes not only spoken language but also non-verbal cues such as hand gestures, facial expressions and tone of the voice, which are used to express feeling and give feedback to improve the accuracy and robustness of the emotion recognition system. If computers could recognize these emotional inputs, they could give specific and appropriate help to users in ways that are more in tune with the user's needs and preferences.

Facial expressions give important clues about emotions. Therefore, several approaches have been proposed to classify human effective states. The features used are typically based on local spatial position or displacement of specific points and regions of the face, unlike the approaches based on audio, which use global statistics of the acoustic features.

We can demonstrate that average generalization rates of 86% can be obtained for emotion recognition on novel individuals using techniques similar to work done in face recognition. Work on emotion recognition relies on image sequences and obtained recognition rates of nearly the same generalization. The model we develop here is potentially of more interest for emotion that makes use of static images of novel individuals in conducting their tests.

1.1 EXISTING SYSTEM

The study of the existing system helps for the new system to be developed. Currently there are many systems available for emotion and face detection. To mention, a few of them are

- The multi layered perceptron method
- Bi modal analysis
- Radial basis function approach

1.2 DISADVANTAGES OF THE EXISTING SYSTEM

All the above mentioned systems are specific only for face detection systems and emotion detection with low level of accuracy. Moreover, these systems also have some disadvantages.

- In multi layered perceptron method, the generalization performance on new images varied from 40% to 80% correct recognition, depending on the choice of the test images.
- In bimodal analysis method, the emotions are recognized with lower performance because none of the modalities considered can accurately separate these classes.
- Radial basis function (RBF) method requires the picture precision to be extremely high which can be achieved only in high resolution photography.

1.3 ADVANTAGES OF PROPOSED SYSTEM:

The main advantage of the proposed system is making use of low resolution bitmap images. It serves as a basic tool for various operations which were based on face detection.

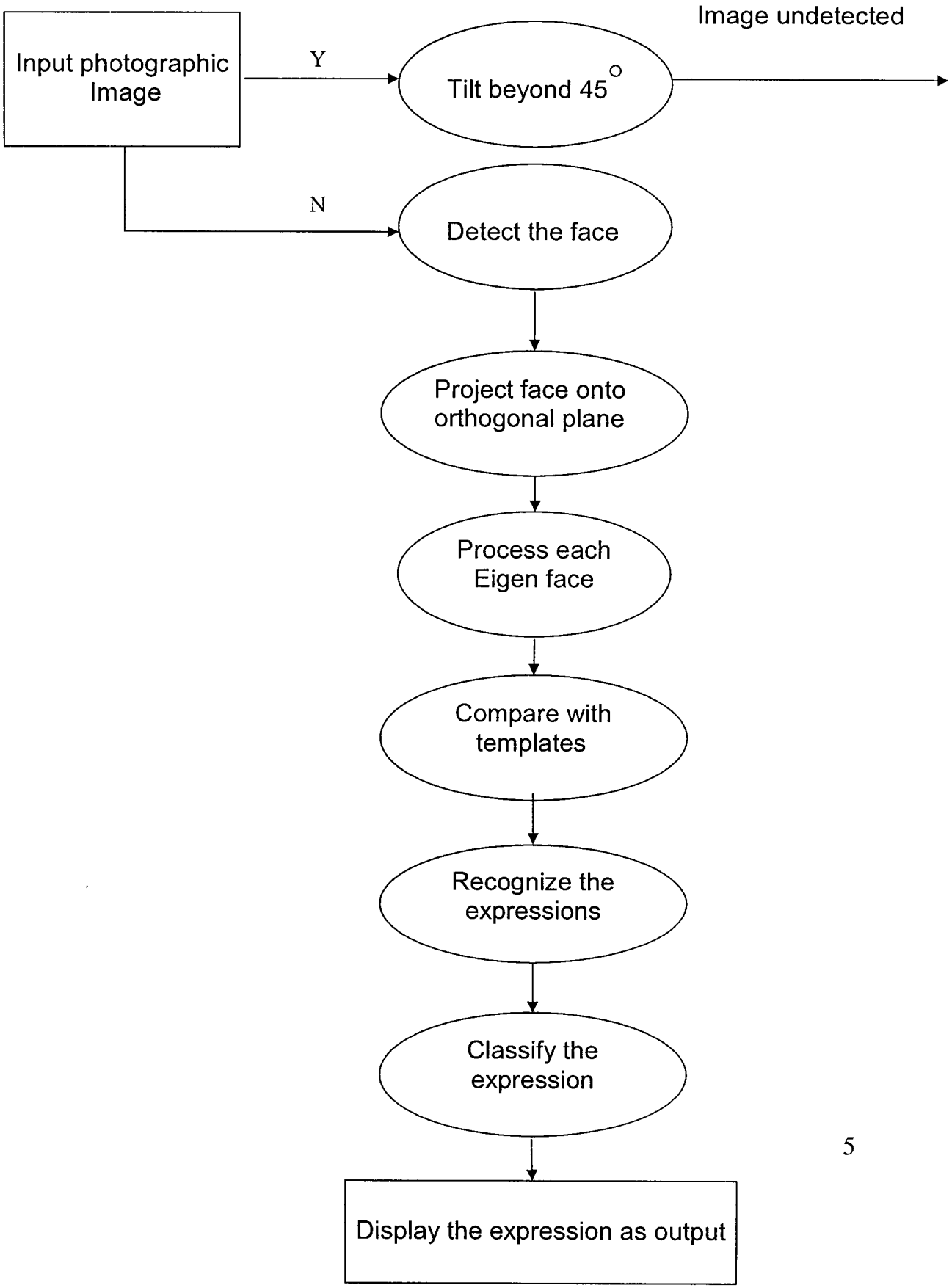
The general idea behind this approach known as "**PRINCIPAL COMPONENT ANALYSIS**" is to extract the main information in the

training set as represented by some template images that capture most of the variability in data. This uses a simpler technique to detect and recognize facial expressions by comparison of various template images. This reduces the complexity and difficulties encountered in other approaches, which are rather complex and therefore computationally expensive.

Many opportunities arise for this approach in this world where all activities are being recognized with the help of systems.

SYSTEM OVERVIEW

2.SYSTEM OVERVIEW



2.1 REQUIREMENTS:

2.1.1 HARDWARE REQUIREMENTS

Processor : Pentium IV 3.0 GHz

RAM : 512 MB

Hard Disk : 40GB

Monitor : SVGA color.

2.1.2 SOFTWARE SPECIFICATION

Operating system : Windows 98 and above

Microsoft Visual Sudio.Net (Visual Basic.Net)

2.2 SYSTEM ANALYSIS

A complete understanding of requirements is essential to the success of project development effort. No matter how well-designed or well coded, a program may be, a poorly analyzed and specified program will disappoint the user. So analysis is the first technical step in this process.
All analysis methods are related by asset of operational principles:
1.The information domain of a problem must be represented and understood.
2. The functions that software is to perform must be defined.
3. The behaviour of the software to perform must be defined.
4. The model that depicts information function and behaviour must be partitioned.
5. The analysis process should move from essential information toward implementation details.

By applying these principles we approach a problem systematically. The information domain is examined so that function maybe understood more completely. Partitioning is applied to reduce the complexity.

System design is a process of developing specifications for the proposed system that meet the criteria established in the system analysis. A major step in system design is the preparation of input and design of output reports in the form acceptable to the system design, which involves first logical design and then physical construction of the system. The logical design describes the structure and characteristic features such as output, input and procedures.

2.3 FEASIBILITY ANALYSIS

All projects are feasible-given unlimited resources and infinite time. But the development of computer based system is likely to be plagued by scarcity of resources and difficulty in completion dates A system which is ill conceived, if recognized early will avert months or years of effort, thousands of dollars and professional embarrassment.

Estimation of resource cost and schedule for software development effort requires experience, access to good historical information and the courage to commit to quantitative measures when qualitative data are all that exists. Estimation carries inherent risks and leads to uncertainty.

While discussing about feasibility we need to concentrate on the points given below

- Economic Feasibility
- Technical Feasibility
- Information and Data Feasibility
- Control and security Feasibility
- Behavioral Feasibility

2.4. SYSTEM DESIGN

2.4.1 INPUT DESIGN

Once the analysis of the system has been done, it would be necessary to identify the data that is required to produce the outputs. Input design features can ensure reliability of the system and generate correct reports from the accurate data. The input design also determines whether the user can interact efficiently with the system.

Since, VB.Net has been chosen for the system the user should easily understand screen design. The validations are carried out easily and the user will have no difficulty in adding and accessing the system.

2.4.2 OUTPUT DESIGN

Computer output is the most important and direct source of information to the user. Efficient, intelligible output design should improve the system relationship with the user and help in decision making. A major form of output is viewing the frames in the destination place.

2.4.3 CODE DESIGN

When large volumes of data are being handled, it is important that the items be stored for easy and quick selection. To accomplish this, each data item must have a unique selection and must be related to other forms or items of data of the same type.

The purpose of code is to facilitate the identification and retrieval of items of information. The system analysis will help to find code structures that will not always be the most suitable for efficient computer processing principles of code design. To accomplish this,each data item must have unique identification and must be related to other items.

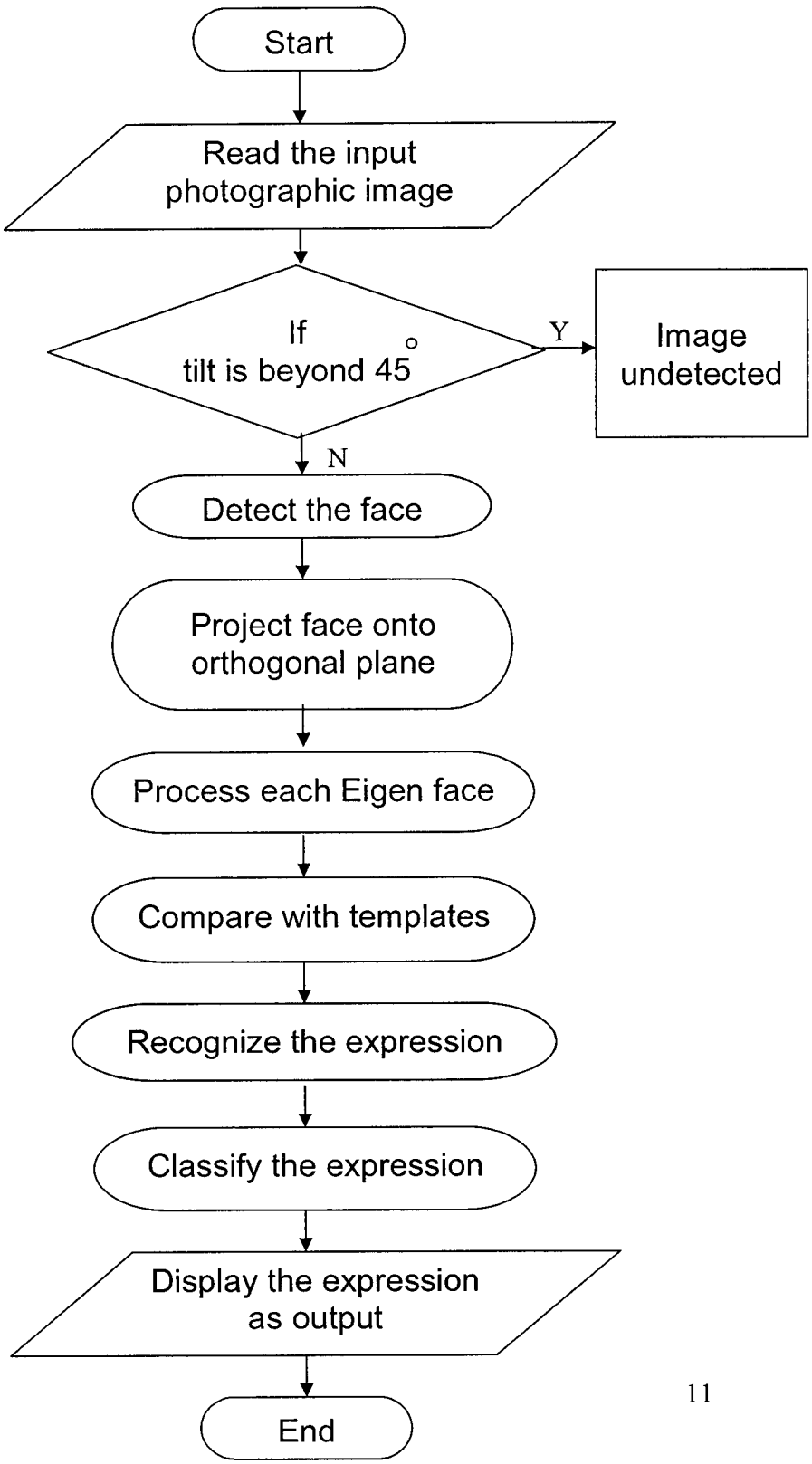
2.4.4 PROCESS DESIGN

The purpose of process design is to establish the communication between the code and data. The process takes the data from the database and checks these data with the user-input data or any other data. Once the process has been finished, the output or the report will be generated based on this process result.

DESCRIPTION OF THE PROPOSED SYSTEM

3.DESCRIPTION OF THE PROPOSED SYSTEM

3.1 FLOWCHART



3.2 MODULE DESCRIPTION

The various modules in the project are:

1. Face detection
2. Emotion recognition

FACE DETECTION:

This is based on information theory approach that decomposes a face into a set of characteristic feature images called "Eigen faces". Recognition is performed by projecting a new image into the subspace spanned by eigen faces and then classifying the face by comparing its position in the face space with the positions of known individuals.

The approach to face detection involves the following initialization steps:

1. Acquire an initial set of images (Training set).
2. Calculate the eigen faces from the training set, keeping only M images that correspond to highest eigen values. The M images define the face space. As new faces are experienced, eigen values can be updated or recalculated.
3. Calculate the corresponding distribution in the M-dimensional weight space for each known individual, by projecting their n face images onto a 'face space'.

These operations can be performed time to time whenever there is free excess computational capacity. Having initialized the system, the following steps are used for face recognition :

1.Calculate a set of weights based on the input image and the M eigen faces by projecting the input image onto each of the eigen faces.
2. Determine if the image is a face at all (whether known or unknown) by checking to see if the image is sufficiently close to the "face space".
3. If it is, a face classifies the weight pattern as either a known person or as unknown.
4. (Optional) Update the eigen faces and/or weight patterns.
5. (Optional) If the same unknown face is seen several times, calculate its characteristic using its weight pattern and incorporate it into known faces.

2.EMOTION RECOGNITION

Facial expression classification was based on Principal Component Analysis.The **Principal Component Analysis (PCA)** is one of the most successful techniques that have been used in image recognition and compression. PCA is a statistical method under the broad title of *factor analysis*. The purpose of PCA is to reduce the large dimensionality of the data space (observed variables) to the smaller intrinsic dimensionality of feature space (independent variables), which are needed to describe the data economically. This is the case when there is a strong correlation between observed variables. The jobs which PCA can do are prediction, redundancy removal, feature extraction, data compression, etc.

Because PCA is a classical technique which can do something in the linear domain, applications having linear models are suitable, such as signal processing, image processing, system and control theory, communications, etc.The main idea of using PCA for face recognition is to express the large 1-D vector of pixels constructed from 2-D facial image into the compact principal components of the feature space. This can be called eigen space projection. Eigen space is calculated by identifying the eigen vectors of the covariance matrix derived from a set of facial images (vectors). The emotion categories are : Happiness, sadness, surprise and neutral.

Various classes used in the above modules:

Image class

The class includes the following steps:

1. Create a new image.
2. Update the integral image.
3. Get the total pixel value for the given area.
4. Detect different types of features.
5. Use random number generator to select the input image.
6. Get the image from the bitmap and save it.
7. Sample the given image within the given bounding box.
8. Create a lookup table.
9. Update values of image object.
10. Recalculate the integral image.

Eigen image class

This class includes the following steps:

1. Initialise the class and add a new image.
2. Calculate the average image templates.
3. Update the eigen faces.
4. Identify the given image.
5. Calculate the eigen face and compare it with other eigen faces.
6. If the faces are different, load and save to the file.

Object detector class

This class includes the following steps:

1. Initialize and determine the number of classifiers for each object type.
2. Feed through hierarchical classifiers of increasing detail.
3. Return the confidence with which the classification was made as a percentage.
4. Scan the image for objects.
5. Learn, reset and return the number of positive and negative samples.

Training set class

This class includes the following steps:

1. Clear false positives.
2. Initialize and add an example bitmap.
3. Get a positive and negative example.
4. Find average positive and negative feature response.
5. Train the system with those samples.

Classifier class

This class includes the following steps:

1. Initialize and load the images from the file.
2. Classify the given image based on facial and non facial features.
3. Randomize the given feature and create a random set of features.
4. Produce the next generation - individuals with below average salience are replaced.
5. Modify thresholds and salient values.
6. Look for false positives and negatives.
7. Learn the set of maximally discriminating features.
8. Set some default values.
9. Determine the expression based on values.

DESCRIPTION OF LANGUAGE USED

4. DESCRIPTION OF LANGUAGE USED

TECHNOLOGY OVERVIEW

What is Microsoft .NET?

Microsoft .NET is a software that connects information, people, system, and devices. It spans clients, servers and developer tools and consists of :

- The .NET framework 1.1 used for building and running all kind of software, including web-based applications, smart client applications and XML web services-components that facilitate integration by sharing data and functionality over a network through standard, platform- independent protocols such as XML, SOAP and HTTP.
- Developer tool, such as Microsoft Visual Studio.NET 2003 which provides an integrated development environment for maximizing productivity with .NET framework.
- A set of servers, including Microsoft windows server, Microsoft SQL server and Microsoft BizTalk Server, that integrates , runs , operates, and manages web services and web based application.
- Client software, such as Windows CE, and Microsoft office XP, that helps developers deliver a deep and compelling user experience across a family of devices and existing products.

- The .Net Framework is an integral Windows component for building and running the next generation of software application and Web services.
- Microsoft .Net support over 20 different programming languages.
- Manages much of the plumbing involved in developing software, enabling developers to focus on the business logic code.
- Makes it easier than ever before to build, deploy and administer secure, robust and high-performing application.

COMMON LANGUAGE RUNTIME

The common language runtime is responsible for runtime services such as language integration, security enforcement, and memory, process and thread management. In addition, the CLR has a role at development time when features such as lifecycle management, storage type naming, cross-language exception handling, and dynamic binding reduce the amount of code that a developer must write to turn business logic into a reusable component.

CLASS LIBRARIES

Base classes provide standard functionality such as input/output, string manipulation, security management, network communications, thread management, text management and user interface design features.

The ADO.NET classes enable developers to interact with data accessed in the form of XML through the OLEDB, ODBC, Oracle and

SQL Server interfaces. XML classes enable XML manipulation, searching, and translation. The ASP.NET classes support the development of Web services. The Windows Forms classes support the development of desktop based smart client application.

Together, the class libraries provide a common and consistent development interface across all languages supported by the .Net framework.

RAPID DEVELOPMENT

The multiple language capability of the .Net Framework enables developers to use the programming language that is most appropriate for a given task and to combine languages within a single application. Components written in different languages can consume functionality from each other transparently, without any extra work required from the developer. Support for the .NET Framework has been announced for over 20 commercial and academic programming languages.

The component based plumbing-free design of the .NET Framework minimizes the amount of code developers have to rewrite and maximizes potential for code reuse.

Visual Basic .Net

.NET is going to change the way we design our applications as much as the introduction of classes to VB changed the best way to build our

VB5 or VB6 applications. With its release for the .NET platform, the Visual Basic language has undergone dramatic changes.

- The language itself is now fully object-oriented.
- Applications and components written in Visual Basic .NET have full access to the .NET Framework which has an extensive class library that provides system and application services.
- All applications developed using Visual Basic .NET run within a managed runtime environment, the .NET common language runtime.

Advantages of shifting from Vb6 to Vb.Net

- **The Common Language Runtime**

Visual Basic has always used a runtime environment. So it may seem strange to say that the biggest change to VB that comes with .NET is the change to a Common Language Runtime (CLR) shared by *all* .NET languages. The reason is that while on the surface the CLR is a runtime library just like the C Runtime library, MSVCRTXX.DLL, or the VB Runtime library, MSVBVMXX.DLL is much larger and has greater functionality. Because of its richness, writing programs that take full advantage of the CLR often seems like you are writing for a whole new operating system API.5 Since all languages that are .NET-compliant use the *same* CLR, there is no need for a language-specific runtime. Code that is CLR can be written in *any* language and still be used equally well by *all* .NET CLR-compliant languages. Our VB code can be used by C# programmers and vice versa with no extra work. Next, there is a common

file format for .NET executable code, called *Microsoft Intermediate Language* (MSIL, or just IL). MSIL is a semi compiled language that gets compiled into native code by the .NET runtime at execution time. This is a vast extension of what existed in all versions of VB prior to version 5. VB applications used to be compiled to p-code (or pseudo code, a machine language for a hypothetical machine), which was an intermediate representation of the final executable code. The various VB runtime engines, interpreted the p-code when a user ran the program. People always complained that VB was too slow because of this. This happened starting in version 5, when you had a choice of p-code (small) or native code (bigger but presumably faster). The key point is that .NET languages combine the best features of a p-code language with the best features of compiled languages. By having all languages write to MSIL, a kind of p-code, and then compile the resulting MSIL to native code, it makes it relatively easy to have cross-language compatibility. But by ultimately generating native code you still get good performance.

- **Completely Object Oriented**

The object-oriented features in VB5 and VB6 were somewhat limited. One key issue was that these versions of VB could not automatically initialize the data inside a class when creating an instance of a class. This led to classes being created in an indeterminate (potentially buggy) state and required the programmer to exercise extra care when using objects. To resolve this, VB .NET adds an important feature called *parameterized constructors.* Another problem was the lack of true *inheritance.* Inheritance is a form of code reuse where you use certain objects that are really more

specialized versions of existing objects. Inheritance is thus the perfect tool when building something like a better textbox based on an existing textbox. In VB5 and 6 you did not have inheritance, so you had to rely on a fairly cumbersome wizard to help make the process of building a better textbox tolerable.

- **Automatic Garbage Collection: Fewer Memory Leaks**

Programmers who used Visual Basic always had a problem with memory leaks from what are called *circular references.* (A circular reference is when you have object A referring to object B and object B referring to object A.) Assuming this kind of code was not there for a reason, there was no way for the VB compiler to realize that this circularity was not significant. This meant that the memory for these two objects was never reclaimed. The *garbage collection* feature built into the .NET CLR eliminates this problem of circular references using much smarter algorithms to determine when circular references can be "cut" and the memory reclaimed. Of course, this extra power comes at a cost, and Chapter 4 will explain the advantages and disadvantages of automatic garbage collection.

- **Structured Exception Handling**

All versions of Visual Basic use a form of error handling that dates back to the first Basic written almost 40 years ago. To be charitable, it had problems. To be uncharitable, it is absurd to use On Error Go To with all the spaghetti code problems that ensue in a modern programming language. Visual Basic has *structured exception handling* which is the most modern and most powerful means of handling errors.

- **True Multithreading**

Multithreaded programs seem to do two things at once. E-mail programs that let you read old e-mail while downloading new e-mail are good examples. Users expect such apps, but you could not write them very easily in earlier versions of VB. In Chapter 10 we introduce you to the pleasures and pitfalls of this incredibly powerful feature of VB .NET. Besides, there are some definite pluses to VB .NET over C#. Here is our top five countdowns:

5. **Inclusion of many of the familiar VB/VBScript functions** such as Mid, Sin(x) instead of Math.Sin(x), or Format Number instead of the more cryptic and often harder to use functions in the .NET Framework.

4. **Readability**. VB .NET uses human-readable words for everything. For example, C# uses a ":", and VB .NET uses "inherits" or "implements." C# uses words like abstract, sealed, and virtual, while VB .NET uses Must Inherit, Not Inheritable, Overridable, Overrides and Shadows.

3. We still have **background compilation** of our code. This means we get immediate feedback from the compiler. (This is much better than simply parsing your code, as is done in C#.)

2. VB .NET is **case insensitive and has a smart editor** that changes the case to reflect your declarations. C#, like all languages in the C family, is case sensitive.

MICROSOFT FOUNDATION CLASSES:

MFC is the C++ class library Microsoft provides to place an object-oriented wrapper around the Windows API.MFC version 4 contains nearly 200 classes, some of which can be used directly and others will serve primarily as base classes for classes of own. MFC is an object-oriented interface to Windows. Object linking and embedding (OLE) and other components of API are merely impossible to stay on the top of the latest developments. That's why MFC acts as an insulating layer with its class libraries between the user and API.

The use of Graphical user interface and Graphical design interface applications remain with growing power with the help of MFC applications which in turn calls API functions. Each of the three components namely user, GDI and kernel is provided as a dynamic linked library (DLL). An application can call functions in the DLL as though they were a part of the application. The API DLLs are normally found in the Windows directory, where files required by the system are usually stored. MFC library is created with the goal of facilitating and simplifying the process of programming to Microsoft Windows due to event driven functions. MFCs are generally referred to as "Framework Application" since it gives the user a framework for an application.

Thus, Visual C++ is a visual development tool that makes use of the MFC library to make the development under Windows environment faster and reliable.

SYSTEM TESTING

5. SYSTEM TESTING

Testing of the debugged programs is one of the most critical aspects of the computer programming triggers. Without programs that work, the system would never produce the output for which it is being designed. Testing is best performed when developers are asked to assist in identifying all errors and bugs. The sample data are used for testing. It is not the quantity but the quality of the data that is used for testing. Testing is aimed at ensuring that the system works accurately and efficiently.

5.1 UNIT TESTING

In this testing we test each module individually and integrate with the overall system. Unit testing focuses on verification effort on the smallest unit of the software design in the module. This testing is carried out during programming stage itself. In this testing step, each module is found to work satisfactorily as regard to expected output from the module. There are some validation checks for fields also. It is very easy to find errors in the system.

5.2 INTEGRATED TESTING

Data can be lost across an interface; one module can have an adverse effort on the other sub functions which when combined may not produce the desired major functions. Integrated testing is systematic testing

for finding the uncovered errors with in the interface. The testing was done with sample data. The need for integrated test is to find the overall system performance.

5.3 VALIDATION TESTING

At the culmination of the black box testing, software is completely assembled as a package, interfacing errors have been uncovered and corrected and final series of the software tests were seen. Validation succeeds when the software functions in the manner that can be reasonably accepted by the customer. The functions or performance characteristics confirming to specification and are accepted.

5.4 OUTPUT TESTING

After performance of the validation testing, the next step is output testing of the proposed system since no system would be useful if it does not produce the required output in the specific format. Asking the user about the format required by system tests the output displayed or generated by the system under consideration.

Here the output format is considered as the screen display, the output format on the screen is found to be correct as the format was designed

in the system phase according to the user need. For the hard copy also the output comes out as specified by the user. Hence the output testing does not result in any correction in the system.

5.5 EXECUTION TESTING

Execution testing is designed to determine whether the system achieves the desired level of proficiency in the production status. Execution testing can verify response times, turn around times as well as design performance. The execution of the system can be tested in whole or part using actual system or a simulated model of a system. It is done to determine whether the system can meet the specified performance criteria.

Thus, various testing are performed and they are checked for its correctness.

SYSTEM IMPLEMENTATION

6. SYSTEM IMPLEMENTATION

After proper testing and validation the question arises whether the system can be implemented or not. Implementation includes all the activities that take place to convert system from old to new.

6.1 TRAINING

A well designed system, if not operated and used properly could fail. Training the users is important. If not done well, it could prevent the successful implementation of an information system.

The training should cover:

- Familiarization with processing the system itself i.e., the equipment used for data entry or processing.
- Training in using the application i.e., the software.
- Good documentation is essential, but this cannot replace training.

6.2 DETECTION

Detection is a process in which the input photographic image is converted to eigen values. The resulting image is compared with the averaged templates stored in the buffer. After detecting the face the emotions are recognized in a similar manner. Some of the steps followed are:

Pre-processing

First of all, the face has to be detected in a photographic image. The facial image is converted to gray scale and segmented into two sub images, the upper and the lower face, respectively. The eye region is determined by locating the eye positions within the upper face image and scaling the relevant image section to a size of 150X100 pixels. The lip region is determined by locating the mouth within the lower face image and scaling the relevant image section to a size of 75X75 pixels. Normalization is applied since the size of the face is not the same in all images.

Feature Extraction

The feature extraction step consists of multi-scale, multiorientation filtering of the region of interest images and a PCA for dimensionality reduction.

Principal Component Analysis

The PCA is applied to each region of interest for each orientation and each scale separately on a speaker-dependent basis.The number of basis images (Eigenfaces) used is determined by the reconstruction error. It can be observed that for both the eye region and the lip region, a different number of coefficients is necessary for different scales.For the eye region we chose 30, 20, and 10 coefficients for each orientation of the scales k = ¡/2, k = ¡/4, and k = ¡/8 respectively, to achieve a reconstruction error below 1.2%.

For the lip region we chose 20, 10, and 5 coefficients for each orientation of the scales k = ¡/2, k = ¡/4, and k = ¡/8 respectively, to achieve a reconstruction error below 1.5%.

Emotion Classification

An Artificial Neural Net (ANN) is used for the classification. A two-layer perceptron with one hidden layer in Feed-Forward topology is chosen. Adaption of the weights is achieved using the Back-Propagation algorithm.The classification is done for the lip region and the eye region separately, and with both taken together. It is done for the emotion categories as well as for the emotion space. At the output, seven neurons are used for the emotion categories. Due to the small number of images, testing was done based on leave one out (LOO) cross validation. The training set was used to train the net 10 times with an adaptation constant $\mu(i) = 0.7 \cdot (5/6)$ i- 1 in the ith iteration.

CONCLUSION

7. CONCLUSION

The system was successfully completed and tested. The system has been designed keeping user interactivity as the major commitment, implementing the project in VB.Net and VC++ with which both code and user control are maintained.

The system is user friendly rather than being expert friendly. We also gain knowledge in developing a .Net application. Enhanced controls like Windows forms are there in VC++.

Here we explored and compared the classification and generalization performance of various supervised and unsupervised techniques for emotion classification from static images.

The template-based approach for emotion classification from static images has good recognition and generalization capabilities. Its poor performance may be caused by the smoothing of important individual facial detail, e.g. the curling of the forehead, and by small misalignment of the faces. The PCA-representation showed excellent classification and reconstruction performance on the training-set but does not allow generalizing to novel faces. The MLP-network trained with back propagation on the other hand showed perfect classification and an acceptable generalization performance, if the test images were not too dissimilar from the training faces.

The generalization performance to novel images critically depends on a good alignment of the facial images, an expressive facial emotion display of the individuals, and a sufficient resolution of the face images, allowing subtle detail to be extracted.

8.APPENDIX

SAMPLE CODES:

```
Private components As System.ComponentModel.IContainer
Public ToolTip1 As System.Windows.Forms.ToolTip
Public WithEvents cmdTest As System.Windows.Forms.Button
Public WithEvents Frame2 As System.Windows.Forms.GroupBox
Public WithEvents cmdStop As System.Windows.Forms.Button
Public WithEvents txtPerformance As System.Windows.Forms.TextBox
Public WithEvents cmdStart As System.Windows.Forms.Button
Public WithEvents picSource As System.Windows.Forms.PictureBox
Public WithEvents Frame1 As System.Windows.Forms.GroupBox
Public WithEvents _picObjectDetected_5 As
System.Windows.Forms.PictureBox
Public WithEvents _picObjectDetected_4 As
System.Windows.Forms.PictureBox
Public WithEvents _picObjectDetected_3 As
System.Windows.Forms.PictureBox
Public WithEvents _picObjectDetected_2 As
System.Windows.Forms.PictureBox
Public WithEvents _picObjectDetected_1 As
System.Windows.Forms.PictureBox
Public WithEvents _picObjectDetected_0 As
System.Windows.Forms.PictureBox
Public WithEvents picScene As System.Windows.Forms.PictureBox
Public WithEvents Label1 As System.Windows.Forms.Label
Public WithEvents _lblObjectType_5 As System.Windows.Forms.Label
Public WithEvents _lblObjectType_4 As System.Windows.Forms.Label
Public WithEvents _lblObjectType_3 As System.Windows.Forms.Label
Public WithEvents _lblObjectType_2 As System.Windows.Forms.Label
Public WithEvents _lblObjectType_1 As System.Windows.Forms.Label
Public WithEvents _lblObjectType_0 As System.Windows.Forms.Label
Public WithEvents lblObjectType As
Microsoft.VisualBasic.Compatibility.VB6.LabelArray
Public WithEvents picObjectDetected As
Microsoft.VisualBasic.Compatibility.VB6.PictureBoxArray
```

```
<System.Diagnostics.DebuggerStepThrough()> Private Sub
InitializeComponent()
Dim resources As System.Resources.ResourceManager = New
System.Resources.ResourceManager(GetType(Form1))
Me.components = New System.ComponentModel.Container
Me.ToolTip1 = New System.Windows.Forms.ToolTip(components)
Me.ToolTip1.Active = True
Me.Frame2 = New System.Windows.Forms.GroupBox
Me.cmdTest = New System.Windows.Forms.Button
Me.Frame1 = New System.Windows.Forms.GroupBox
Me.cmdStop = New System.Windows.Forms.Button
Me.txtPerformance = New System.Windows.Forms.TextBox
Me.cmdStart = New System.Windows.Forms.Button
Me.picSource = New System.Windows.Forms.PictureBox
Me._picObjectDetected_5 = New System.Windows.Forms.PictureBox
Me._picObjectDetected_4 = New System.Windows.Forms.PictureBox
Me._picObjectDetected_3 = New System.Windows.Forms.PictureBox
Me._picObjectDetected_2 = New System.Windows.Forms.PictureBox
Me._picObjectDetected_1 = New System.Windows.Forms.PictureBox
Me._picObjectDetected_0 = New System.Windows.Forms.PictureBox
Me.picScene = New System.Windows.Forms.PictureBox
Me.Label1 = New System.Windows.Forms.Label
Me._lblObjectType_5 = New System.Windows.Forms.Label
Me._lblObjectType_4 = New System.Windows.Forms.Label
Me._lblObjectType_3 = New System.Windows.Forms.Label
Me._lblObjectType_2 = New System.Windows.Forms.Label
Me._lblObjectType_1 = New System.Windows.Forms.Label
Me._lblObjectType_0 = New System.Windows.Forms.Label
Me.lblObjectType = New
Microsoft.VisualBasic.Compatibility.VB6.LabelArray(components)
Me.picObjectDetected = New
Microsoft.VisualBasic.Compatibility.VB6.PictureBoxArray(components)
CType(Me.lblObjectType,
System.ComponentModel.ISupportInitialize).BeginInit()
CType(Me.picObjectDetected,
System.ComponentModel.ISupportInitialize).BeginInit()
Me.BackColor = System.Drawing.Color.White
Me.Text = "HumanEmotionRecognition"
Me.ClientSize = New System.Drawing.Size(625, 494)
```

```
Me.Location = New System.Drawing.Point(4, 23)
Me.Icon = CType(resources.GetObject("Form1.Icon"),
System.Drawing.Icon)
Me.StartPosition =
System.Windows.Forms.FormStartPosition.WindowsDefaultLocation
Me.Font = New System.Drawing.Font("Arial", 8.0!,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
CType(0, Byte))
Me.AutoScaleBaseSize = New System.Drawing.Size(5, 13)
Me.FormBorderStyle = System.Windows.Forms.FormBorderStyle.Sizable
Me.ControlBox = True
Me.Enabled = True
Me.KeyPreview = False
Me.MaximizeBox = True
Me.MinimizeBox = True
Me.Cursor = System.Windows.Forms.Cursors.Default
Me.RightToLeft = System.Windows.Forms.RightToLeft.No
Me.ShowInTaskbar = True
Me.HelpButton = False
Me.WindowState = System.Windows.Forms.FormWindowState.Normal
Me.Name = "Form1"
Me.Frame2.BackColor = System.Drawing.Color.White
Me.Frame2.Text = "Testing"
Me.Frame2.Size = New System.Drawing.Size(170, 107)
Me.Frame2.Location = New System.Drawing.Point(3, 199)
Me.Frame2.TabIndex = 19
Me.Frame2.Font = New System.Drawing.Font("Arial", 8.0!,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
CType(0, Byte))
Me.Frame2.Enabled = True
Me.Frame2.ForeColor = System.Drawing.SystemColors.ControlText
Me.Frame2.RightToLeft = System.Windows.Forms.RightToLeft.No
Me.Frame2.Visible = True
Me.Frame2.Name = "Frame2"
Me.cmdTest.TextAlign = System.Drawing.ContentAlignment.MiddleCenter
Me.cmdTest.BackColor = System.Drawing.Color.White

Me.cmdTest.Text = "Test"
Me.cmdTest.Size = New System.Drawing.Size(113, 37)
```

```
Me.cmdTest.Location = New System.Drawing.Point(27, 35)
Me.cmdTest.TabIndex = 20
Me.cmdTest.Font = New System.Drawing.Font("Arial", 8.0!,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
CType(0, Byte))
Me.cmdTest.CausesValidation = True
Me.cmdTest.Enabled = True
Me.cmdTest.ForeColor = System.Drawing.SystemColors.ControlText
Me.cmdTest.Cursor = System.Windows.Forms.Cursors.Default
Me.cmdTest.RightToLeft = System.Windows.Forms.RightToLeft.No
Me.cmdTest.TabStop = True
Me.cmdTest.Name = "cmdTest"
Me.Frame1.BackColor = System.Drawing.Color.White
Me.Frame1.Text = "Training"
Me.Frame1.Size = New System.Drawing.Size(169, 175)
Me.Frame1.Location = New System.Drawing.Point(4, 16)
Me.Frame1.TabIndex = 14
Me.Frame1.Font = New System.Drawing.Font("Arial", 8.0!,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
CType(0, Byte))
Me.Frame1.Enabled = True
Me.Frame1.ForeColor = System.Drawing.SystemColors.ControlText
Me.Frame1.RightToLeft = System.Windows.Forms.RightToLeft.No
Me.Frame1.Visible = True
Me.Frame1.Name = "Frame1"
Me.cmdStop.TextAlign = System.Drawing.ContentAlignment.MiddleCenter
Me.cmdStop.BackColor = System.Drawing.Color.White
Me.cmdStop.Text = "Stop"
Me.cmdStop.Size = New System.Drawing.Size(113, 33)
Me.cmdStop.Location = New System.Drawing.Point(27, 121)
Me.cmdStop.TabIndex = 18
Me.cmdStop.Font = New System.Drawing.Font("Arial", 8.0!,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
CType(0, Byte))
Me.cmdStop.CausesValidation = True
Me.cmdStop.Enabled = True
Me.cmdStop.ForeColor = System.Drawing.SystemColors.ControlText
Me.cmdStop.Cursor = System.Windows.Forms.Cursors.Default
Me.cmdStop.RightToLeft = System.Windows.Forms.RightToLeft.No
```

```
Me.cmdStop.TabStop = True
Me.cmdStop.Name = "cmdStop"
Me.txtPerformance.AutoSize = False
Me.txtPerformance.Size = New System.Drawing.Size(59, 22)
Me.txtPerformance.Location = New System.Drawing.Point(72, 35)
Me.txtPerformance.TabIndex = 17
Me.txtPerformance.Text = "0"
Me.txtPerformance.Font = New System.Drawing.Font("Arial", 8.0!,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
CType(0, Byte))
Me.txtPerformance.AcceptsReturn = True
Me.txtPerformance.TextAlign =
System.Windows.Forms.HorizontalAlignment.Left
Me.txtPerformance.BackColor = System.Drawing.SystemColors.Window
Me.txtPerformance.CausesValidation = True
Me.txtPerformance.Enabled = True
Me.txtPerformance.ForeColor =
System.Drawing.SystemColors.WindowText
Me.txtPerformance.HideSelection = True
Me.txtPerformance.ReadOnly = False
Me.txtPerformance.MaxLength = 0
Me.txtPerformance.Cursor = System.Windows.Forms.Cursors.IBeam
Me.txtPerformance.Multiline = False
Me.txtPerformance.RightToLeft = System.Windows.Forms.RightToLeft.No
Me.txtPerformance.ScrollBars = System.Windows.Forms.ScrollBars.None
Me.txtPerformance.TabStop = True
Me.txtPerformance.Visible = True
Me.txtPerformance.BorderStyle =
System.Windows.Forms.BorderStyle.Fixed3D
Me.txtPerformance.Name = "txtPerformance"
Me.cmdStart.TextAlign = System.Drawing.ContentAlignment.MiddleCenter
Me.cmdStart.BackColor = System.Drawing.Color.White
Me.cmdStart.Text = "Start"
Me.cmdStart.Size = New System.Drawing.Size(113, 33)
Me.cmdStart.Location = New System.Drawing.Point(27, 88)
Me.cmdStart.TabIndex = 16
Me.cmdStart.Font = New System.Drawing.Font("Arial", 8.0!,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
CType(0, Byte))
```

```
Me.cmdStart.CausesValidation = True
Me.cmdStart.Enabled = True
Me.cmdStart.ForeColor = System.Drawing.SystemColors.ControlText
Me.cmdStart.Cursor = System.Windows.Forms.Cursors.Default
Me.cmdStart.RightToLeft = System.Windows.Forms.RightToLeft.No
Me.cmdStart.TabStop = True
Me.cmdStart.Name = "cmdStart"
Me.picSource.Size = New System.Drawing.Size(28, 29)
Me.picSource.Location = New System.Drawing.Point(24, 31)
Me.picSource.Image = CType(resources.GetObject("picSource.Image"),
System.Drawing.Image)
Me.picSource.TabIndex = 15
Me.picSource.Font = New System.Drawing.Font("Arial", 8.0!,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
CType(0, Byte))
Me.picSource.Dock = System.Windows.Forms.DockStyle.None
Me.picSource.BackColor = System.Drawing.SystemColors.Control
Me.picSource.CausesValidation = True
Me.picSource.Enabled = True
Me.picSource.ForeColor = System.Drawing.SystemColors.ControlText
Me.picSource.Cursor = System.Windows.Forms.Cursors.Default
Me.picSource.RightToLeft = System.Windows.Forms.RightToLeft.No
Me.picSource.TabStop = True
Me.picSource.Visible = True
Me.picSource.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.Normal
Me.picSource.BorderStyle = System.Windows.Forms.BorderStyle.Fixed3D
Me.picSource.Name = "picSource"
Me._picObjectDetected_5.BackColor = System.Drawing.Color.White
Me._picObjectDetected_5.Size = New System.Drawing.Size(73, 80)
Me._picObjectDetected_5.Location = New System.Drawing.Point(541, 371)
Me._picObjectDetected_5.TabIndex = 6
Me._picObjectDetected_5.Font = New System.Drawing.Font("Arial", 8.0!,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
CType(0, Byte))
Me._picObjectDetected_5.Dock = System.Windows.Forms.DockStyle.None
Me._picObjectDetected_5.CausesValidation = True
Me._picObjectDetected_5.Enabled = True
```

```
Me._picObjectDetected_5.ForeColor=
System.Drawing.SystemColors.ControlText
Me._picObjectDetected_5.Cursor= System.Windows.Forms.Cursors.Default
Me._picObjectDetected_5.RightToLeft=
System.Windows.Forms.RightToLeft.No
Me._picObjectDetected_5.TabStop = True
Me._picObjectDetected_5.Visible = True
Me._picObjectDetected_5.SizeMode=
System.Windows.Forms.PictureBoxSizeMode.Normal
Me._picObjectDetected_5.BorderStyle=
System.Windows.Forms.BorderStyle.Fixed3D
Me._picObjectDetected_5.Name = "_picObjectDetected_5"
Me._picObjectDetected_4.BackColor = System.Drawing.Color.White
Me._picObjectDetected_4.Size = New System.Drawing.Size(73, 80)
Me._picObjectDetected_4.Location = New System.Drawing.Point(466, 372)
Me._picObjectDetected_4.TabIndex = 5
Me._picObjectDetected_4.Font = New System.Drawing.Font("Arial", 8.0!,
System.Drawing.FontStyle.Regular,    System.Drawing.GraphicsUnit.Point,
CType(0, Byte))
Me._picObjectDetected_4.Dock = System.Windows.Forms.DockStyle.None
Me._picObjectDetected_4.CausesValidation = True
Me._picObjectDetected_4.Enabled = True
Me._picObjectDetected_4.ForeColor=
System.Drawing.SystemColors.ControlText
Me._picObjectDetected_4.Cursor= System.Windows.Forms.Cursors.Default
Me._picObjectDetected_4.RightToLeft=
System.Windows.Forms.RightToLeft.No
Me._picObjectDetected_4.TabStop = True
Me._picObjectDetected_4.Visible = True
Me._picObjectDetected_4.SizeMode=
System.Windows.Forms.PictureBoxSizeMode.Normal
Me._picObjectDetected_4.BorderStyle=
System.Windows.Forms.BorderStyle.Fixed3D
Me._picObjectDetected_4.Name = "_picObjectDetected_4"
Me._picObjectDetected_3.BackColor = System.Drawing.Color.White
Me._picObjectDetected_3.Size = New System.Drawing.Size(73, 80)
Me._picObjectDetected_3.Location = New System.Drawing.Point(391, 373)
Me._picObjectDetected_3.TabIndex = 4
```

```
Me._picObjectDetected_3.Font = New System.Drawing.Font("Arial", 8.0!,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
CType(0, Byte))
Me._picObjectDetected_3.Dock = System.Windows.Forms.DockStyle.None
Me._picObjectDetected_3.CausesValidation = True
Me._picObjectDetected_3.Enabled = True
Me._picObjectDetected_3.ForeColor=
System.Drawing.SystemColors.ControlText
Me._picObjectDetected_3.Cursor= System.Windows.Forms.Cursors.Default
Me._picObjectDetected_3.RightToLeft=
System.Windows.Forms.RightToLeft.No
Me._picObjectDetected_3.TabStop = True
Me._picObjectDetected_3.Visible = True
Me._picObjectDetected_3.SizeMode=
System.Windows.Forms.PictureBoxSizeMode.Normal
Me._picObjectDetected_3.BorderStyle=
System.Windows.Forms.BorderStyle.Fixed3D
Me._picObjectDetected_3.Name = "_picObjectDetected_3"
Me._picObjectDetected_2.BackColor = System.Drawing.Color.White
Me._picObjectDetected_2.Size = New System.Drawing.Size(73, 80)
Me._picObjectDetected_2.Location = New System.Drawing.Point(316, 372)
Me._picObjectDetected_2.TabIndex = 3
Me._picObjectDetected_2.Font = New System.Drawing.Font("Arial", 8.0!,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
CType(0, Byte))
Me._picObjectDetected_2.Dock = System.Windows.Forms.DockStyle.None
Me._picObjectDetected_2.CausesValidation = True
Me._picObjectDetected_2.Enabled = True
Me._picObjectDetected_2.ForeColor=
System.Drawing.SystemColors.ControlText
Me._picObjectDetected_2.Cursor= System.Windows.Forms.Cursors.Default
Me._picObjectDetected_2.RightToLeft=
System.Windows.Forms.RightToLeft.No
Me._picObjectDetected_2.TabStop = True
Me._picObjectDetected_2.Visible = True
Me._picObjectDetected_2.SizeMode=
System.Windows.Forms.PictureBoxSizeMode.Normal
Me._picObjectDetected_2.BorderStyle=
System.Windows.Forms.BorderStyle.Fixed3D
```

```
Me._picObjectDetected_2.Name = "_picObjectDetected_2"
Me._picObjectDetected_1.BackColor = System.Drawing.Color.White
Me._picObjectDetected_1.Size = New System.Drawing.Size(73, 80)
Me._picObjectDetected_1.Location = New System.Drawing.Point(241, 372)
Me._picObjectDetected_1.TabIndex = 2
Me._picObjectDetected_1.Font = New System.Drawing.Font("Arial", 8.0!,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
CType(0, Byte))
Me._picObjectDetected_1.Dock = System.Windows.Forms.DockStyle.None
Me._picObjectDetected_1.CausesValidation = True
Me._picObjectDetected_1.Enabled = True
Me._picObjectDetected_1.ForeColor=
System.Drawing.SystemColors.ControlText
Me._picObjectDetected_1.Cursor= System.Windows.Forms.Cursors.Default
Me._picObjectDetected_1.RightToLeft=
System.Windows.Forms.RightToLeft.No
Me._picObjectDetected_1.TabStop = True
Me._picObjectDetected_1.Visible = True
Me._picObjectDetected_1.SizeMode=
System.Windows.Forms.PictureBoxSizeMode.Normal
Me._picObjectDetected_1.BorderStyle=
System.Windows.Forms.BorderStyle.Fixed3D
Me._picObjectDetected_1.Name = "_picObjectDetected_1"
Me._picObjectDetected_0.BackColor = System.Drawing.Color.White
Me._picObjectDetected_0.Size = New System.Drawing.Size(73, 80)
Me._picObjectDetected_0.Location = New System.Drawing.Point(167, 372)
Me._picObjectDetected_0.TabIndex = 1
Me._picObjectDetected_0.Font = New System.Drawing.Font("Arial", 8.0!,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
CType(0, Byte))
Me._picObjectDetected_0.Dock = System.Windows.Forms.DockStyle.None
Me._picObjectDetected_0.CausesValidation = True
Me._picObjectDetected_0.Enabled = True
Me._picObjectDetected_0.ForeColor=
System.Drawing.SystemColors.ControlText
Me._picObjectDetected_0.Cursor= System.Windows.Forms.Cursors.Default
Me._picObjectDetected_0.RightToLeft=
System.Windows.Forms.RightToLeft.No
Me._picObjectDetected_0.TabStop = True
```

```
Me._picObjectDetected_0.Visible = True
Me._picObjectDetected_0.SizeMode=
System.Windows.Forms.PictureBoxSizeMode.Normal
Me._picObjectDetected_0.BorderStyle=
System.Windows.Forms.BorderStyle.Fixed3D
Me._picObjectDetected_0.Name = "_picObjectDetected_0"
Me.picScene.BackColor = System.Drawing.Color.White
Me.picScene.Size = New System.Drawing.Size(403, 338)
Me.picScene.Location = New System.Drawing.Point(185, 20)
Me.picScene.TabIndex = 0
Me.picScene.Font = New System.Drawing.Font("Arial", 8.0!,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
CType(0, Byte))
Me.picScene.Dock = System.Windows.Forms.DockStyle.None
Me.picScene.CausesValidation = True
Me.picScene.Enabled = True
Me.picScene.ForeColor = System.Drawing.SystemColors.ControlText
Me.picScene.Cursor = System.Windows.Forms.Cursors.Default
Me.picScene.RightToLeft = System.Windows.Forms.RightToLeft.No
Me.picScene.TabStop = True
Me.picScene.Visible = True
Me.picScene.SizeMode=
System.Windows.Forms.PictureBoxSizeMode.Normal
Me.picScene.BorderStyle = System.Windows.Forms.BorderStyle.Fixed3D
Me.picScene.Name = "picScene"
Me.Label1.BackColor = System.Drawing.Color.White
Me.Label1.Text = "Expression"
Me.Label1.Size = New System.Drawing.Size(62, 15)
Me.Label1.Location = New System.Drawing.Point(99, 457)
Me.Label1.TabIndex = 13
Me.Label1.Font = New System.Drawing.Font("Arial", 8.0!,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
CType(0, Byte))
Me.Label1.TextAlign = System.Drawing.ContentAlignment.TopLeft
Me.Label1.Enabled = True
Me.Label1.ForeColor = System.Drawing.SystemColors.ControlText
Me.Label1.Cursor = System.Windows.Forms.Cursors.Default
Me.Label1.RightToLeft = System.Windows.Forms.RightToLeft.No
Me.Label1.UseMnemonic = True
```

```
Me.Label1.Visible = True
Me.Label1.AutoSize = False
Me.Label1.BorderStyle = System.Windows.Forms.BorderStyle.None
Me.Label1.Name = "Label1"
Me._lblObjectType_5.TextAlign=
System.Drawing.ContentAlignment.TopCenter
Me._lblObjectType_5.BackColor = System.Drawing.Color.White
Me._lblObjectType_5.Text = " "
Me._lblObjectType_5.Size = New System.Drawing.Size(75, 20)
Me._lblObjectType_5.Location = New System.Drawing.Point(541, 455)
Me._lblObjectType_5.TabIndex = 12
Me._lblObjectType_5.Font = New System.Drawing.Font("Arial", 8.0!,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
CType(0, Byte))
Me._lblObjectType_5.Enabled = True
Me._lblObjectType_5.ForeColor=
System.Drawing.SystemColors.ControlText
Me._lblObjectType_5.Cursor = System.Windows.Forms.Cursors.Default
Me._lblObjectType_5.RightToLeft=
System.Windows.Forms.RightToLeft.No
Me._lblObjectType_5.UseMnemonic = True
Me._lblObjectType_5.Visible = True
Me._lblObjectType_5.AutoSize = False
Me._lblObjectType_5.BorderStyle=
System.Windows.Forms.BorderStyle.None
Me._lblObjectType_5.Name = "_lblObjectType_5"
Me._lblObjectType_4.TextAlign=
System.Drawing.ContentAlignment.TopCenter
Me._lblObjectType_4.BackColor = System.Drawing.Color.White
Me._lblObjectType_4.Text = " "
Me._lblObjectType_4.Size = New System.Drawing.Size(75, 19)
Me._lblObjectType_4.Location = New System.Drawing.Point(466, 455)
Me._lblObjectType_4.TabIndex = 11
Me._lblObjectType_4.Font = New System.Drawing.Font("Arial", 8.0!,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
CType(0, Byte))
Me._lblObjectType_4.Enabled = True
Me._lblObjectType_4.ForeColor=
System.Drawing.SystemColors.ControlText
```

```
Me._lblObjectType_4.Cursor = System.Windows.Forms.Cursors.Default
Me._lblObjectType_4.RightToLeft=
System.Windows.Forms.RightToLeft.No
Me._lblObjectType_4.UseMnemonic = True
Me._lblObjectType_4.Visible = True
Me._lblObjectType_4.AutoSize = False
Me._lblObjectType_4.BorderStyle=
System.Windows.Forms.BorderStyle.None
Me._lblObjectType_4.Name = "_lblObjectType_4"
Me._lblObjectType_3.TextAlign=
System.Drawing.ContentAlignment.TopCenter
Me._lblObjectType_3.BackColor = System.Drawing.Color.White
Me._lblObjectType_3.Text = " "
Me._lblObjectType_3.Size = New System.Drawing.Size(75, 20)
Me._lblObjectType_3.Location = New System.Drawing.Point(391, 455)
Me._lblObjectType_3.TabIndex = 10
Me._lblObjectType_3.Font = New System.Drawing.Font("Arial", 8.0!,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
CType(0, Byte))
Me._lblObjectType_3.Enabled = True
Me._lblObjectType_3.ForeColor=
System.Drawing.SystemColors.ControlText
Me._lblObjectType_3.Cursor = System.Windows.Forms.Cursors.Default
Me._lblObjectType_3.RightToLeft=
System.Windows.Forms.RightToLeft.No
Me._lblObjectType_3.UseMnemonic = True
Me._lblObjectType_3.Visible = True
Me._lblObjectType_3.AutoSize = False
Me._lblObjectType_3.BorderStyle=
System.Windows.Forms.BorderStyle.None
Me._lblObjectType_3.Name = "_lblObjectType_3"
Me._lblObjectType_2.TextAlign=
System.Drawing.ContentAlignment.TopCenter
Me._lblObjectType_2.BackColor = System.Drawing.Color.White
Me._lblObjectType_2.Text = " "
Me._lblObjectType_2.Size = New System.Drawing.Size(75, 19)
Me._lblObjectType_2.Location = New System.Drawing.Point(317, 455)
Me._lblObjectType_2.TabIndex = 9
```

```
Me._lblObjectType_2.Font = New System.Drawing.Font("Arial", 8.0!,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
CType(0, Byte))
Me._lblObjectType_2.Enabled = True
Me._lblObjectType_2.ForeColor=
System.Drawing.SystemColors.ControlText
Me._lblObjectType_2.Cursor = System.Windows.Forms.Cursors.Default
Me._lblObjectType_2.RightToLeft=
System.Windows.Forms.RightToLeft.No
Me._lblObjectType_2.UseMnemonic = True
Me._lblObjectType_2.Visible = True
Me._lblObjectType_2.AutoSize = False
Me._lblObjectType_2.BorderStyle=
System.Windows.Forms.BorderStyle.None
Me._lblObjectType_2.Name = "_lblObjectType_2"
Me._lblObjectType_1.TextAlign=
System.Drawing.ContentAlignment.TopCenter
Me._lblObjectType_1.BackColor = System.Drawing.Color.White
Me._lblObjectType_1.Text = " "
Me._lblObjectType_1.Size = New System.Drawing.Size(75, 16)
Me._lblObjectType_1.Location = New System.Drawing.Point(240, 457)
Me._lblObjectType_1.TabIndex = 8
Me._lblObjectType_1.Font = New System.Drawing.Font("Arial", 8.0!,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
CType(0, Byte))
Me._lblObjectType_1.Enabled = True
Me._lblObjectType_1.ForeColor=
System.Drawing.SystemColors.ControlText
Me._lblObjectType_1.Cursor = System.Windows.Forms.Cursors.Default
Me._lblObjectType_1.RightToLeft=
System.Windows.Forms.RightToLeft.No
Me._lblObjectType_1.UseMnemonic = True
Me._lblObjectType_1.Visible = True
Me._lblObjectType_1.AutoSize = False
Me._lblObjectType_1.BorderStyle=
System.Windows.Forms.BorderStyle.None
Me._lblObjectType_1.Name = "_lblObjectType_1"
Me._lblObjectType_0.TextAlign=
System.Drawing.ContentAlignment.TopCenter
```

```
Me._lblObjectType_0.BackColor = System.Drawing.Color.White
Me._lblObjectType_0.Text = " "
Me._lblObjectType_0.Size = New System.Drawing.Size(75, 16)
Me._lblObjectType_0.Location = New System.Drawing.Point(166, 457)
Me._lblObjectType_0.TabIndex = 7
Me._lblObjectType_0.Font = New System.Drawing.Font("Arial", 8.0!,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
CType(0, Byte))
Me._lblObjectType_0.Enabled = True
Me._lblObjectType_0.ForeColor=
System.Drawing.SystemColors.ControlText
Me._lblObjectType_0.Cursor = System.Windows.Forms.Cursors.Default
Me._lblObjectType_0.RightToLeft=
System.Windows.Forms.RightToLeft.No
Me._lblObjectType_0.UseMnemonic = True
Me._lblObjectType_0.Visible = True
Me._lblObjectType_0.AutoSize = False
Me._lblObjectType_0.BorderStyle=
System.Windows.Forms.BorderStyle.None
Me._lblObjectType_0.Name = "_lblObjectType_0"
Me.Controls.Add(Frame2)
Me.Controls.Add(Frame1)
Me.Controls.Add(_picObjectDetected_5)
Me.Controls.Add(_picObjectDetected_4)
Me.Controls.Add(_picObjectDetected_3)
Me.Controls.Add(_picObjectDetected_2)
Me.Controls.Add(_picObjectDetected_1)
Me.Controls.Add(_picObjectDetected_0)
Me.Controls.Add(picScene)
Me.Controls.Add(Label1)
Me.Controls.Add(_lblObjectType_5)
Me.Controls.Add(_lblObjectType_4)
Me.Controls.Add(_lblObjectType_3)
Me.Controls.Add(_lblObjectType_2)
Me.Controls.Add(_lblObjectType_1)
Me.Controls.Add(_lblObjectType_0)
Me.Frame2.Controls.Add(cmdTest)
Me.Frame1.Controls.Add(cmdStop)
Me.Frame1.Controls.Add(txtPerformance)
```

```
Me.Frame1.Controls.Add(cmdStart)
Me.Frame1.Controls.Add(picSource)
Me.lblObjectType.SetIndex(_lblObjectType_5, CType(5, Short))
Me.lblObjectType.SetIndex(_lblObjectType_4, CType(4, Short))
Me.lblObjectType.SetIndex(_lblObjectType_3, CType(3, Short))
Me.lblObjectType.SetIndex(_lblObjectType_2, CType(2, Short))
Me.lblObjectType.SetIndex(_lblObjectType_1, CType(1, Short))
Me.lblObjectType.SetIndex(_lblObjectType_0, CType(0, Short))
Me.picObjectDetected.SetIndex(_picObjectDetected_5, CType(5, Short))
Me.picObjectDetected.SetIndex(_picObjectDetected_4, CType(4, Short))
Me.picObjectDetected.SetIndex(_picObjectDetected_3, CType(3, Short))
Me.picObjectDetected.SetIndex(_picObjectDetected_2, CType(2, Short))
Me.picObjectDetected.SetIndex(_picObjectDetected_1, CType(1, Short))
Me.picObjectDetected.SetIndex(_picObjectDetected_0, CType(0, Short))
CType(Me.picObjectDetected,
System.ComponentModel.ISupportInitialize).EndInit()
CType(Me.lblObjectType,
System.ComponentModel.ISupportInitialize).EndInit()
End Sub
Const OBJECT_FACE As Short = 0
Const OBJECT_EXPRESSION_HAPPY As Short = 1
Const OBJECT_EXPRESSION_SAD As Short = 2
Const OBJECT_EXPRESSION_NORMAL As Short = 3
Const OBJECT_EXPRESSION_SURPRISED As Short = 4

Const IMAGE_WIDTH As Short = 160
Const IMAGE_HEIGHT As Short = 160

Const DETECTED_OBJECT_WIDTH As Short = 25
Const DETECTED_OBJECT_HEIGHT As Short = 25

Const PostitiveImages_faces As Short = 400
Const NegativeImages_faces As Short = 220
Const PostitiveImages_expressions As Short = 15
Const NegativeImages_expressions As Short =
(PostitiveImages_expressions * 3)

Private Sub Form1_Load(ByVal eventSender As System.Object, ByVal
eventArgs As System.EventArgs) Handles MyBase.Load
```

```
Dim aspectRatio As Single
Dim NoOfRotations As Short

NoOfRotations = 0
aspectRatio = VB6.PixelsToTwipsX(picScene.Width) /
VB6.PixelsToTwipsY(picScene.Height) * 1.2

Call addDetector(OBJECT_FACE, 0, NoOfRotations, aspectRatio, -1)
Call training_Load()

Call createAttentionMap(0, 0, IMAGE_WIDTH, IMAGE_HEIGHT)

Randomize()
End Sub
Private Sub RandomScene()
Dim i As Short

i = CShort(Rnd() * 9) + 1

picScene.Image = System.Drawing.Image.FromFile(VB6.GetPath &
"\scenes\test" & i & ".jpg")
End Sub
Private Sub LoadImages_faces()
Dim i As Short

For i = 1 To PostitiveImages_faces
picSource.Image = System.Drawing.Image.FromFile(VB6.GetPath &
"\faces\" & i & ".bmp")
picSource.Refresh()
Call picToBitmap(picSource, 24, 24)
Call training_addPositiveExample(0, bmp)
Next

For i = 1 To NegativeImages_faces
picSource.Image = System.Drawing.Image.FromFile(VB6.GetPath &
"\nonfaces\" & i & ".bmp")
picSource.Refresh()
Call picToBitmap(picSource, 24, 24)
```

```
Call training_addNegativeExample(0, bmp)
Next

End Sub



Private Sub LoadImages_expressions()
Dim i As Short

For i = 1 To PostitiveImages_expressions
picSource.Image = System.Drawing.Image.FromFile(VB6.GetPath & 
"\faces\expressions\happy" & i & ".bmp")
picSource.Refresh()
Call picToBitmap(picSource, 24, 24)
Call addIndividual(bmp, 1)

picSource.Image = System.Drawing.Image.FromFile(VB6.GetPath & 
"\faces\expressions\sad" & i & ".bmp")
picSource.Refresh()
Call picToBitmap(picSource, 24, 24)
Call addIndividual(bmp, 2)

picSource.Image = System.Drawing.Image.FromFile(VB6.GetPath & 
"\faces\expressions\normal" & i & ".bmp")
picSource.Refresh()
Call picToBitmap(picSource, 24, 24)
Call addIndividual(bmp, 3)

picSource.Image = System.Drawing.Image.FromFile(VB6.GetPath & 
"\faces\expressions\surprised" & i & ".bmp")
picSource.Refresh()
Call picToBitmap(picSource, 24, 24)
Call addIndividual(bmp, 4)
Next

End Sub
```

```
long classimage::detectFeature(int x, int y, int wdth, int hght, int
featureType)
{
 long area[4];
 long v=0;

 switch(featureType)
 {
  case 0:    {
    area[0] = getIntegral(x, y, x + wdth, y + hght + hght, 0);
    area[1] = getIntegral(x + wdth, y, x + wdth + wdth, y + hght + hght, 0);
    v = abs(area[0] - area[1]);
        break;
       }

  case 1:
       {
    area[0] = getIntegral(x, y, x + wdth, y + hght, 0);
    area[1] = getIntegral(x, y + hght, x + wdth, y + hght + hght, 0);
    v = abs(area[0] - area[1]);
        break;
       }

  case 2:
       {
    area[0] = getIntegral(x, y, x + wdth, y + hght, 0);
    area[1] = getIntegral(x + wdth, y, x + wdth + wdth, y + hght, 0);
    area[2] = getIntegral(x + wdth + wdth, y, x + wdth + wdth + wdth, y +
hght, 0);
    v = abs((area[1] * 2) - (area[0] + area[2]));
        break;
       }
```

```
 case 3:      {
    area[0] = getIntegral(x, y, x + wdth, y + hght, 0);
    area[1] = getIntegral(x + wdth, y, x + wdth + wdth, y + hght, 0);
    area[2] = getIntegral(x, y + hght, x + wdth, y + hght + hght, 0);
    area[3] = getIntegral(x + wdth, y + hght, x + wdth + wdth, y + hght +
hght, 0);

v = abs((area[1] + area[2]) - (area[0] + area[3]));
         break;
        }
           case 4:
        {
    area[0] = getIntegral(x, y, x + wdth, y + hght + hght, 1);
    area[1] = getIntegral(x + wdth, y, x + wdth + wdth, y + hght + hght, 1);
    v = abs(area[0] - area[1]);
         break;
        }

   case 5:
        {
    area[0] = getIntegral(x, y, x + wdth, y + hght, 1);
    area[1] = getIntegral(x, y + hght, x + wdth, y + hght + hght, 1);
    v = abs(area[0] - area[1]);
         break;
        }
   case 6:    {
    area[0] = getIntegral(x, y, x + wdth, y + hght, 1);
    area[1] = getIntegral(x + wdth, y, x + wdth + wdth, y + hght, 1);
    area[2] = getIntegral(x + wdth + wdth, y, x + wdth + wdth + wdth, y +
hght, 1);
    v = abs((area[1] * 2) - (area[0] + area[2]));
         break;
        }
   case 7:    {
    area[0] = getIntegral(x, y, x + wdth, y + hght, 1);
    area[1] = getIntegral(x + wdth, y, x + wdth + wdth, y + hght, 1);
    area[2] = getIntegral(x, y + hght, x + wdth, y + hght + hght, 1);
    area[3] = getIntegral(x + wdth, y + hght, x + wdth + wdth, y + hght +
hght, 1);
```

```
   v = abs((area[1] + area[2]) - (area[0] + area[3]));
         break;
        }

   case 8:
        {
    v = getIntegral(x, y, x + wdth + wdth, y + hght + hght, 0);

 break;

        }
   case 9:
        {
    v = getIntegral(x, y, x + wdth + wdth, y + hght + hght, 1);
         break;
        }
 }

 return(v);
}

void classeigenimage::updateImageTemplate()
{
 int i,x,y;

 for (i=0;i<NoOfImages;i++)
 {
  for (x=0;x<imageWidth;x++)
       {
    for (y=0;y<imageHeight;y++)
        {
     if (i > 0)
       imageTemplate[x][y]    =    imageTemplate[x][y]    +    image[i]-
>image[x][y][0];
       else
       imageTemplate[x][y] = image[i]->image[x][y][0];
    }
  }
 }
```

```
 for (x=0;x<imageWidth;x++)
 {
  for (y=0;y<imageHeight;y++)
   imageTemplate[x][y] = (long)(imageTemplate[x][y] / NoOfImages);
 }
}


void classeigenimage::updateEigenImages()
{
 int i,x,y,df;

 for (i=0;i<NoOfImages;i++)
 {
  for (x=0;x<imageWidth;x++)
      {
   for (y=0;y<imageHeight;y++)
       {
    df = image[i]->image[x][y][0] - imageTemplate[x][y];
    if (df < 0) df = 0;
    EigenImage[i]->image[x][y][0] = (unsigned char)(df);
   }
  }
 }
}

int classeigenimage::Identify(classimage *img)
{
 int i,x,y,df;
 long Distance,minDistance;
 int a,b;
  IdentityID=-1;
 testImage->updateFromImage(img);
   for (x=0;x<imageWidth;x++)
 {
  for (y=0;y<imageHeight;y++)
  {
   df = testImage->image[x][y][0] - imageTemplate[x][y];
   if (df < 0) df = 0;
```

```
  testEigenImage->image[x][y][0] = (unsigned char)df;
   }
  }
   minDistance = 99999999;
  for (i=0;i<NoOfImages;i++)
  {
   Distance = 0;


for (x=0;x<imageWidth;x++)
   {
    for (y=0;y<imageHeight;y++)
        {
     a = EigenImage[i]->image[x][y][0];
     b = testEigenImage->image[x][y][0];
     df = abs(a - b);
     Distance += df;
    }
   }

   if (Distance < minDistance)
       {
    minDistance = Distance;
    Identity = image[i];
        IdentityID = imageID[i];
   }
  }

  return(IdentityID);
}

int classobjectdetector::totalPositives()
{
 return(TrainingSet->NoOfPositives);
}

int classobjectdetector::totalNegatives()
{
 return(TrainingSet->NoOfNegatives);
}
```

```
long classtrainingset::averagePositiveFeature(int tx, int ty, int width, int
height, int featureType)
{
 int i;
 long v=0;

 if (NoOfPositives>0)
 {

for (i=0;i<NoOfPositives;i++)
        {
   v += Positive[i]->detectFeature(tx, ty, width, height, featureType);
        }
  v /= NoOfPositives;
 }
 return(v);
}
long classtrainingset::averageNegativeFeature(int tx, int ty, int width, int
height, int featureType)
{
 int i,cts;
 long v=0,v2;

 if (NoOfNegatives>0)
 {
  cts = 1;
  for (i=0;i<NoOfNegatives;i++)
       {
    if (FalsePositive[i])
         {
     v2 = Negative[i]->detectFeature(tx, ty, width, height, featureType);
     v += v2;
     cts++;
        }
       }
  v /= cts;
 }
 return(v);
}
```

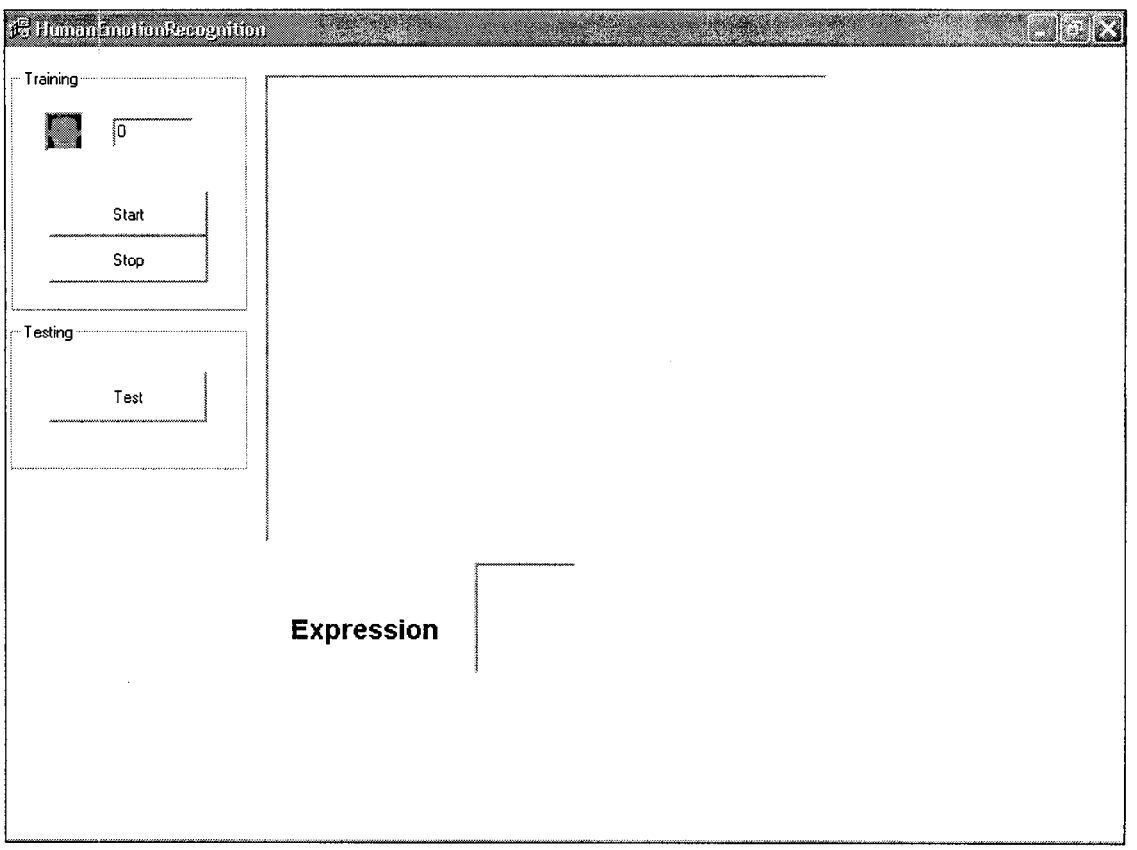
```
bool classclassifier::ClassifyImage(classimage *img, bool applyThreshold)
{
 int f,tx,ty,width,height,featureType,hits,misses,p;
 long v,pos,neg;
   pos = 0;
 neg = 0;
 for (f=0;f<NoOfFeatures;f++)


{
  if (Feature[f][5] == 1)
      {
    tx = Feature[f][0];
    ty = Feature[f][1];
    width = Feature[f][2];
       height = Feature[f][3];
    featureType = Feature[f][4];

    v = img->detectFeature(tx, ty, width, height, featureType);

    if (v > FeatureThreshold[f])
       {
     if (FeatureThresholdDirection[f])
            {
      pos += v;
      FeatureResult[f] = true;
      hits++;
            }
      else
            {
      neg += v;
      FeatureResult[f] = false;
      misses++;
     }
       }
    else
       {
     if (FeatureThresholdDirection[f])
     {
```

```
neg += v;
        FeatureResult[f] = false;
        misses++;
                }
       else
                {
        pos += v;
        FeatureResult[f] = true;


hits++;
      }
     }

   }
  }

  if (applyThreshold)
  {
   p = hits / (hits + misses + 1) * 100;
   if ((p > Threshold) && (pos > neg))
        {
         classificationProbability = p;
    return(true);
        }
   else
        {
         classificationProbability = 0;
    return(false);
        }
  }
  else
  {
   if (pos > neg)
        {
         classificationProbability = 0;
    return(true);
        }
```

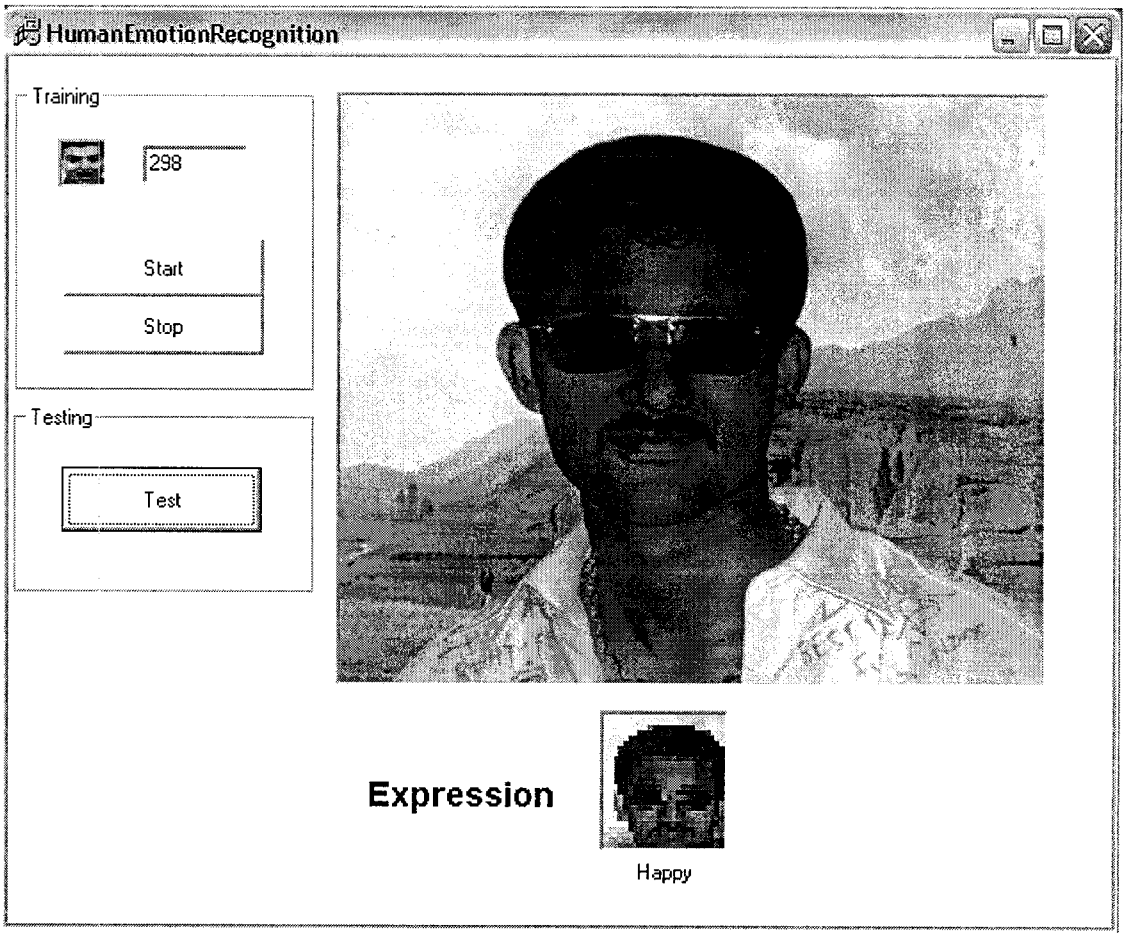
```
else
        {
         classificationProbability = 0;
    return(false);
        }
  }

}
```

SNAP SHOTS

STAGE 1 :


HumanEmotionRecognition
Training
0
Start
Stop
Testing
Test
Expression

STAGE 3 :


HumanEmotionRecognition
Training
298
Start
Stop
Testing
Test
Expression
Happy

9. BIBLIOGRAPHY

1. R. Chellappa, C.L. Wilson, and Sirohey, "Human and Machine recognition of faces: A survey," *Proceedings of IEEE*, Vol. 83, No. 5, 1995, pp. 705-740.

2. R. Brunelli and T. Poggio, "Face recognition: Features versus templates," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 15, No. 10, 1993.

3. M. Kirby and L. Sirovich, "Application of the Karhunen-Loeve procedure for the characterization of human faces," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.12, No. 1, 1990, pp. 103-108.

4. M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of Cognitive Neuroscience*, Vol. 3, 1991, pp.71-86.

5. A.J.O'Toole, H.Abdi, K.A.Deffenbacher and D.Valentin, "A low-dimensional represen-tation of faces in the higher dimensions of the space," Journal of the Optical Society of America A, Vol. 10, 1993, 405-411.

WEBSITES:

1. http://en.wikipedia.org
2. http://www.microsoft.com
3. http://www.ieee.org