Real Time Face Detection

Author: Nafi Rushdi Nafi Hamdan (88267)
Advisor: Jeffrey Duffany, Ph.D
Department: Computer Engineering & Computer Science , MCS (ITMIA) (SP-18)

Abstract

This poster presents details of a prototype for research project on face detection systems. The prototype uses EmguCV cross platform .Net wrapper with the Intel OpenCV image processing library and C# .Net. The library allows capture and processing of image from a camera in real time. Principal Component Analysis (PCA) with Eigen face is used on the technology. Details of the project such as code samples, architecture diagram, system design ER diagrams, use cases, activity diagrams, data flow diagrams, class diagram, sequence diagram, testing and feasibility are presented.

Introduction

Face recognition is a computer based system to recognize faces by scanning the face in real-time using a camera, and matching the image to stored images from the database [1]. Data validation, matching facial attributes is the key to this technology, and the objective is to find the identity of the person, unknown or unknown, under the scanner [2]. The technology of face recognition forms an important part of bio-metric face recognition systems, used in offices, sensitive installations, to counter and detect terror activities, in crime fighting, and others [3].

The system proposed in this poster uses basic .Net API's to interact and acquire the output of a local camera used for scanning, such as a webcam or other attached cameras. The output from the camera is used to recognize faces in real time. The proposed system offers many advantages over existing systems [4]. Some of the existing systems such as human guards, security, locks and keys, finger print scanning, surveillance cameras, magnetic swipe cards, etc., are not very reliable. They can be hacked or misused and allow identity theft. This poster, presents findings from a research project on face recognition system [2]. Fig 1 and 2 present the system design and architecture.

Background

Woody Bldesoe and others researched methods for face recognition in the 1960s. The efforts were very crude since computer science and applications were very rudimentary. Problems such as variability of head rotation, facial expression lighting intensity, aging, and other factors, limited the scope of these efforts. Later, experiments were conducted by the University of Bochum, University of Southern California, Massachusetts Institute of Technology, and others were more promising. They were used by some banks for identification of customers. In the next development phases, high-resolution face images were developed along with 3D and iris scans [5].

Over the decades, the rate of error in detecting face reduced. Systems are now available where face recognition algorithms are used to extract features of the face. Apple X phone uses advanced face recognition system [6].

Problem

Existing systems have low speed of cross-referencing and matching thousands of records in the database, with the least number of errors. Available systems use proprietary software and hardware, and they are expensive. The objectives of this research are:

- Use Open Source software and low-cost hardware to develop a reliable system.
- Develop a prototype with code, hardware configuration and test.

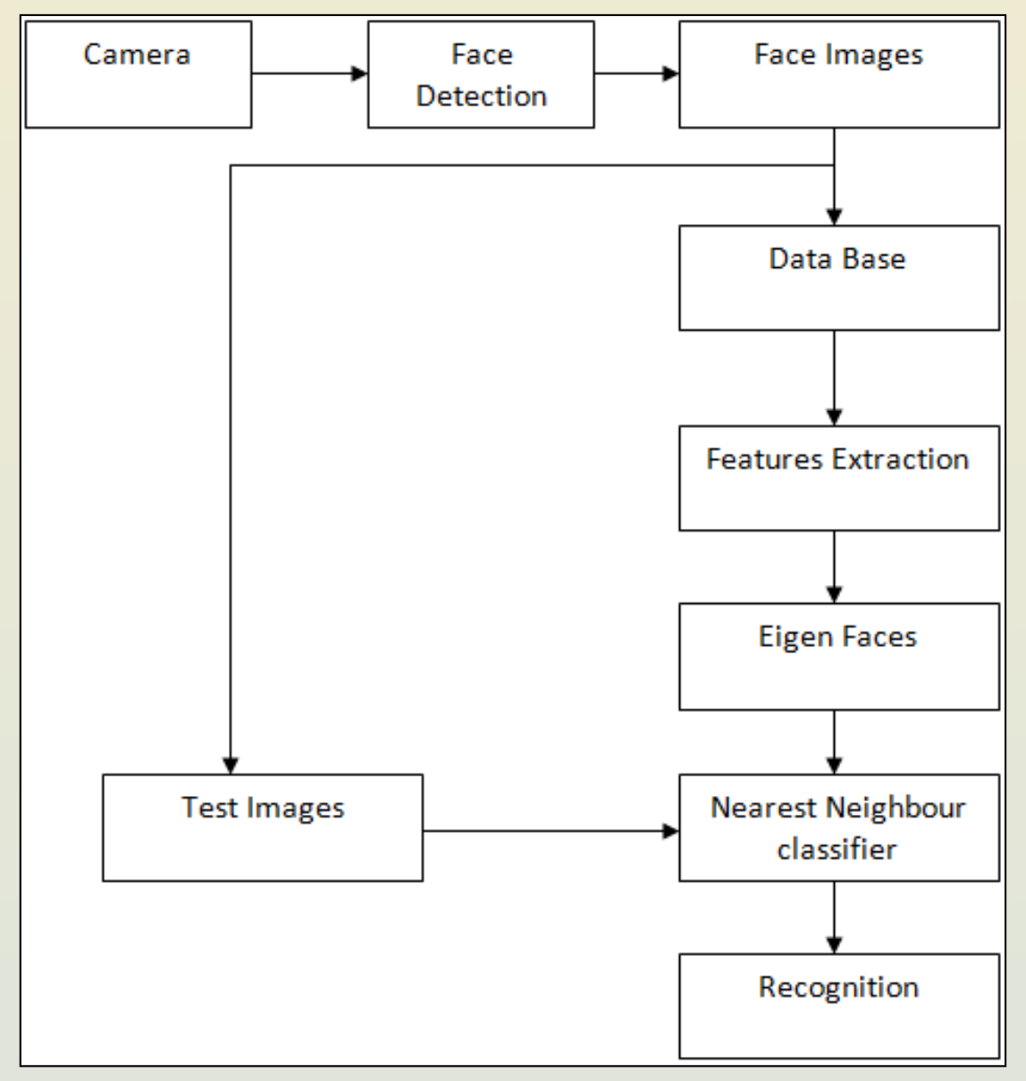


Figure 1. System Design

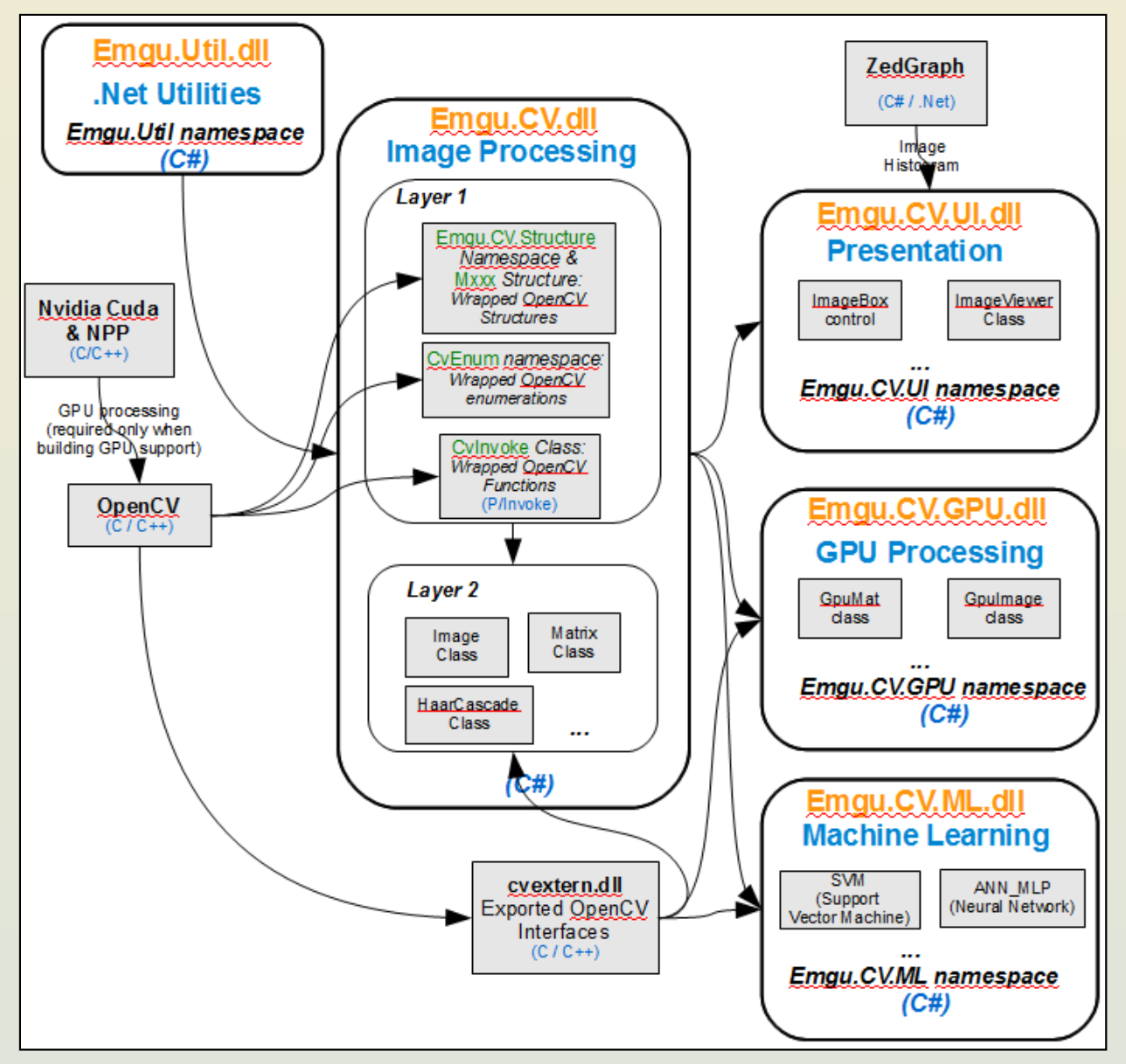


Figure 2. System Architecture

Methodology

The prototype is based on the Emgu CV cross platform. Code for the app was obtained from open source repository. Emgu CV has two layers, layer 1 or the basic layer and layer 2 or the second layer. Layer 1 carries the structure, enumeration,, functions mappings that show the ones on OpenCV. Layer 2 has the classes that use features of .Net.Image is specified by the generic parameters such as color and depth, and an 8bit unsigned Grayscale image is done in EmguCVby calling a function. Image pixels are stored in a 3D array. When a new image is stored, an identification number, and the pixel values are stored as numbers. When a face is scanned, the image is converted into pixel values and matched with the stored records. If there is a match, then the person is accepted, if there is a mismatch with the stored values, then the request is denied [7]. Fig 3 shows the welcome screen and Fig 4, a sample code.

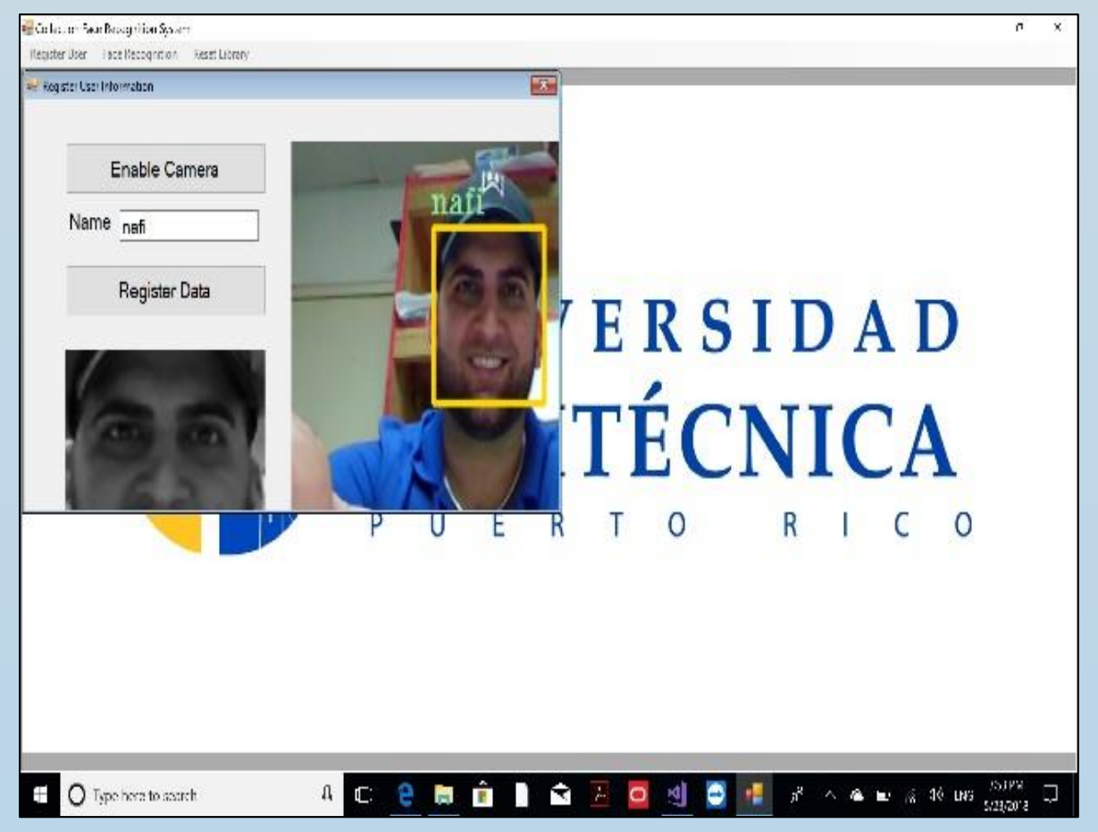
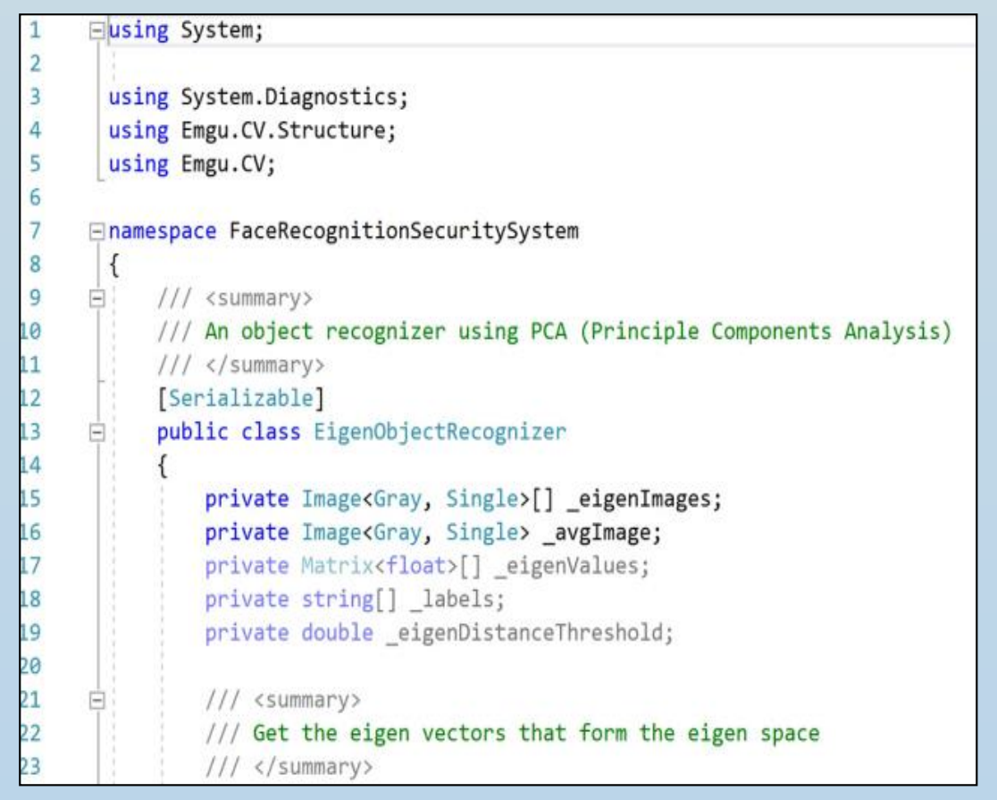


Figure 3. Welcome Screen



```
using System;

using System.Diagnostics;
using Emgu.CV.Structure;
using Emgu.CV;

namespace FaceRecognitionSecuritySystem
{
    /// <summary>
    /// An object recognizer using PCA (Principle Components Analysis)
    /// </summary>
    [Serializable]
    public class EigenObjectRecognizer
    {
        private Image<Gray, Single>[] _eigenImages;
        private Image<Gray, Single> _avgImage;
        private Matrix<float>[] _eigenValues;
        private string[] _labels;
        private double _eigenDistanceThreshold;

        /// <summary>
        /// Get the eigen vectors that form the eigen space
        /// </summary>
```

Figure 4 Sample Code

Sample codes, obtained from Emgu CV, compiled and run as sub-routines. On the Project Executable folder called "Faces" was created and images were stored as BMP formats. Names of the images were face1.bmp, face2.bmp, etc, and the names were stored in a file "TrainedLabels.txt", in the same folder. It is a flat file with CSV structure and data can be obtained from the folder "bin/x86/Debug/Faces/". A Euclidean algorithm was used to compare the images and to select the image that are match the face on the camera [7]. Figure 5 shows the Data Flow Diagram.

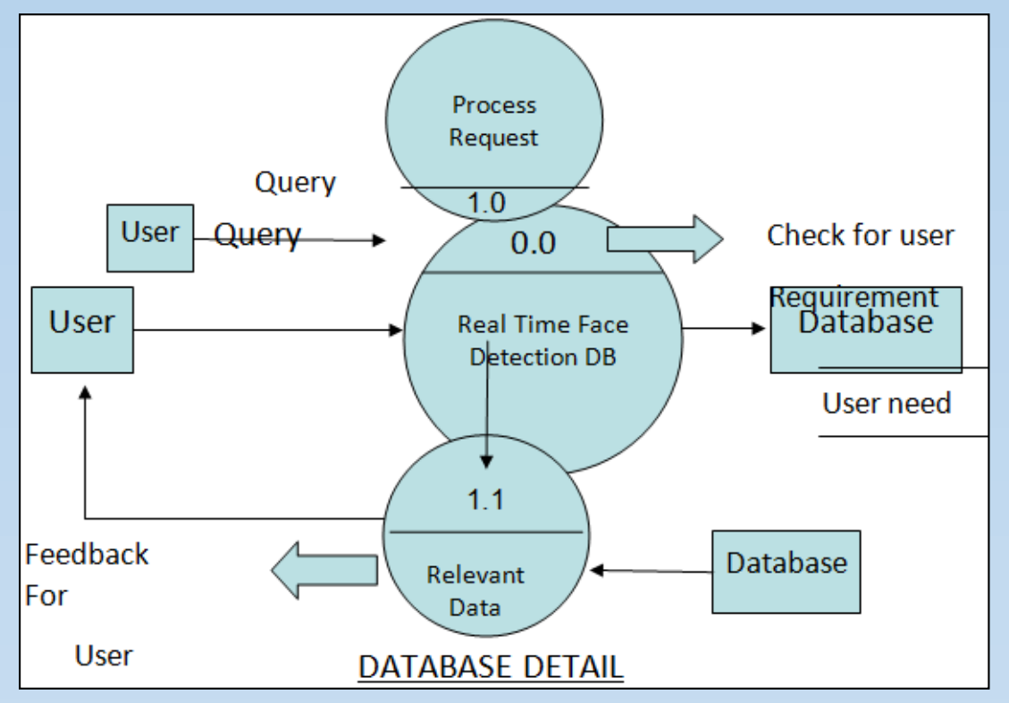


Figure 5 DFD Level 1

User sends the query and this is taken as a process request, which is passed to the real time face detection database. If the user records exist or if the scan is matched, then the request is processes, a match found and feedback given to the user, with the relevant data.

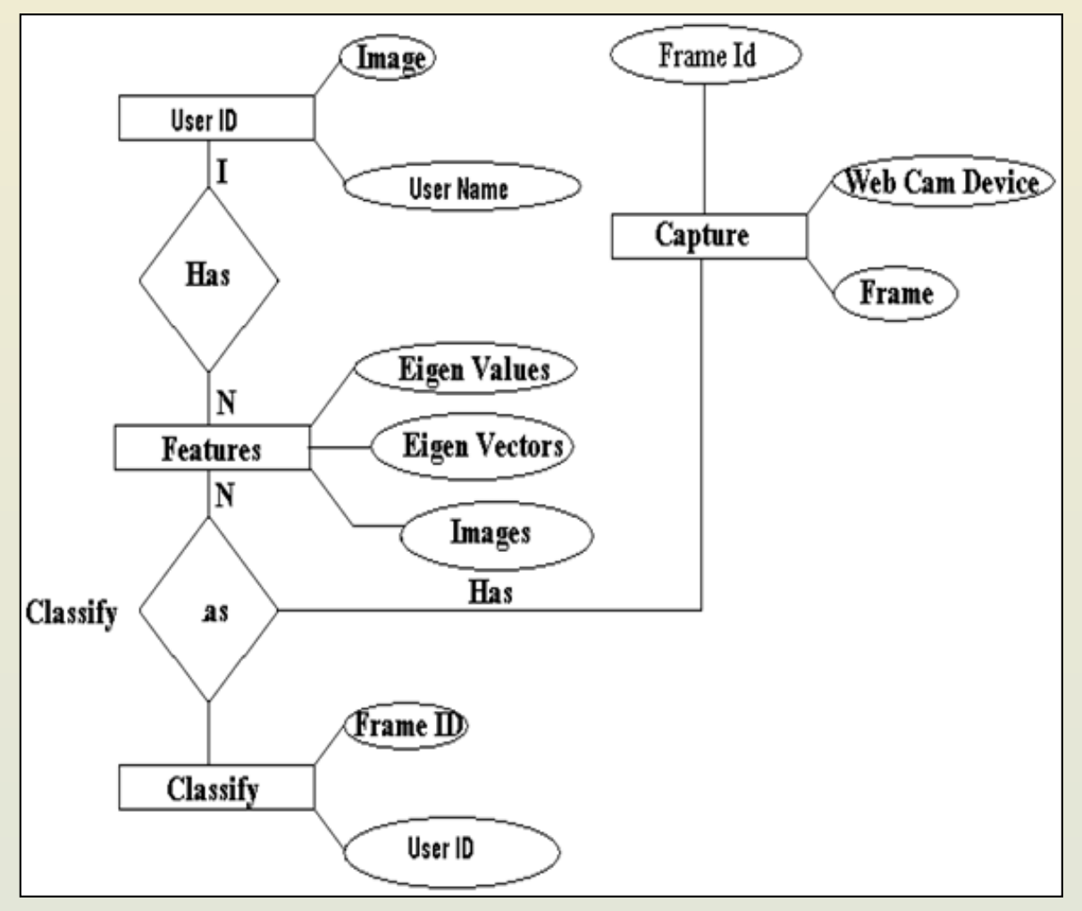


Figure 5 ER Diagram

As seen in Fig 4, the entity User Id has attributes like Image and User Name. The entity User Id is connected to another Entity Features. Features have attributes like Eigen Values, Eigen Vectors and Images. Thus, a User Id can have 1 or many features like Eigen Values, Eigen Vectors and Images.

The features of User Id are connected to another entity Classify which has entities like Frame Id and User Id. Thus, features are classified into Frame Id and User Id. The entity Features is connected to another entity Capture which has attributes Frame Id, Webcam Device and Frame. Thus, Features of the User Id captured has Frame Id, Webcam Device and Frame associated to it.

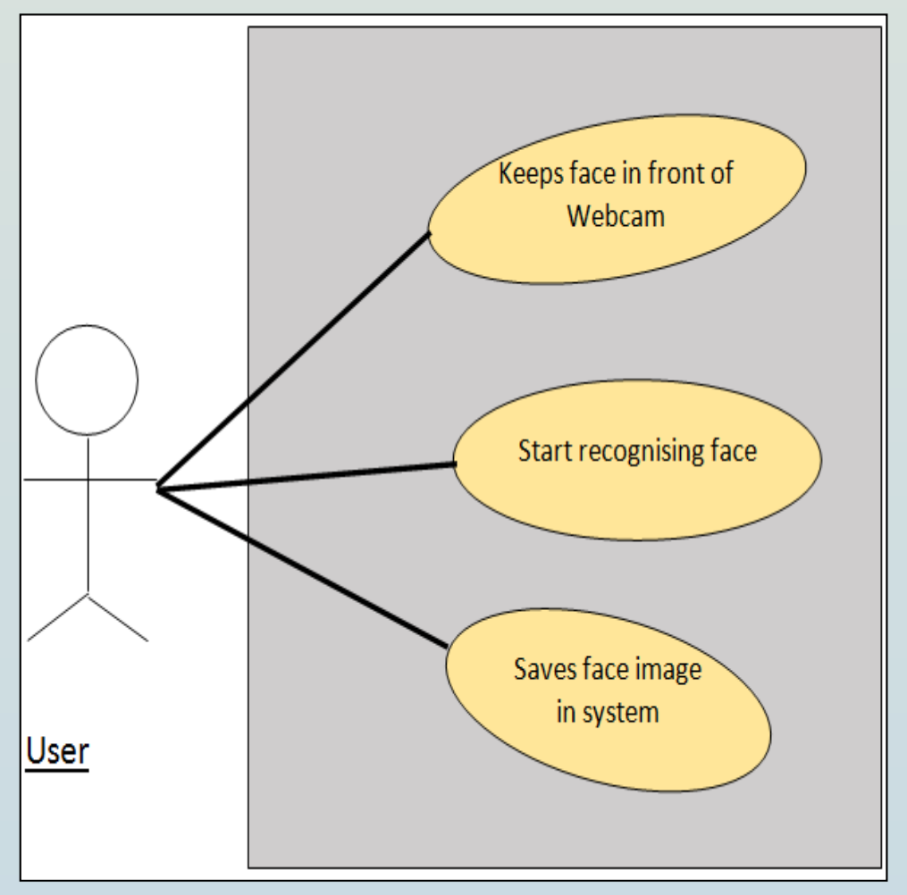


Figure 6 Use Case Diagram

Figure 6 presents the use case diagram. The use case is to click a photo using webcam; user has to perform certain activities. User has to initially keep his face in front of the Webcam. The webcam will use face recognition technology to detect the face in front of it and when the face is recognized it will process and save the image.

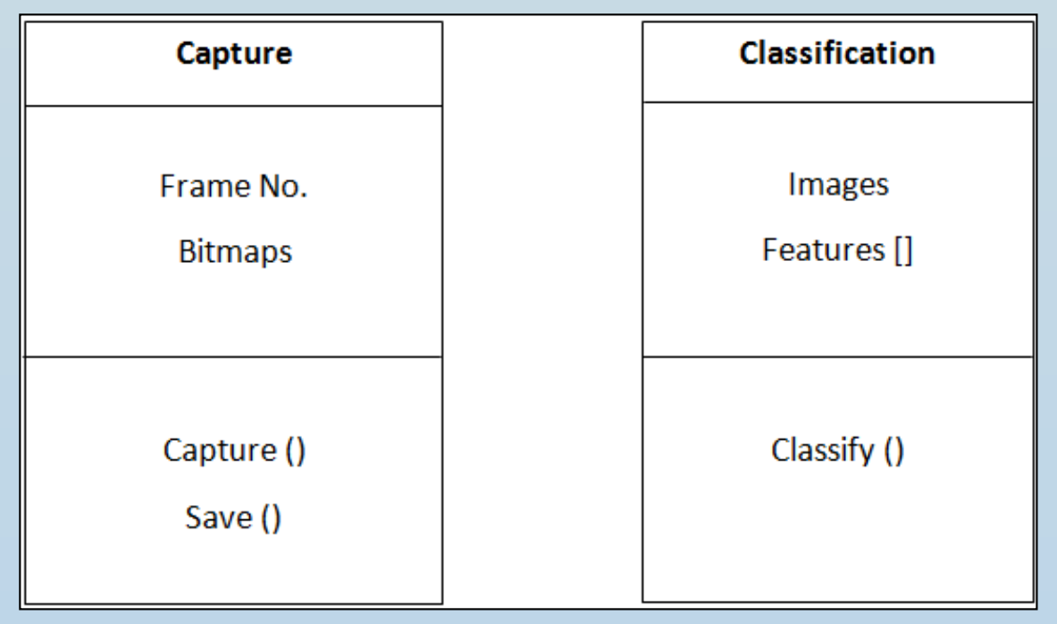


Figure 7 Class Diagram

Figure 7 illustrates the class diagram of the system. There are two classes present here namely, Capture and Classification. Class Capture has attributes like Frame No and Bitmaps. It means every Image that is captured is classified by Frame No and their Bitmaps.

Results and Discussion

The project was implemented using Microsoft Visual Studio 2008. All databases were created and stored in SQL Server 2005. Tables were created and query run to store data or record the project. The software and hardware used in the research are detailed as follows:

Hardware Requirement: i3 Processor Based Computer, 1GB-Ram, 5 GB Hard Disk.
Software Requirement: Windows XP, Windows 7 (ultimate & enterprise), Visual studio 2008, SQL Server 2005.
Operating System: Windows XP, 7(ultimate & enterprise)
Languages: Asp.Net with C# (.Net 2008)
Database System: MS-SQL Server 2005
Documentation Tool: MS - Word 2010
These are the basic hardware and software requirements and they can be enhanced if required.

Tests were run to identity the desired output for all types of inputs. This approach is essential for a successful project. System testing was done to find if user requirements were met. The code for the new system was written with ASP.NET and with C#, which was also used as the interface for front-end designing. The system was tested with users and all applications were tested for different use cases. Some defects and bugs were found in some of the components, and these were corrected before implementation. Form and data flow was as per the requirements. The testing done here was System Testing checking whether the user requirements were satisfied. The code for the new system has been written completely using ASP .NET with C# as the coding language, C# as the interface for front-end designing. The new system has been tested well with the help of the users and all the applications have been verified from every nook and corner of the user.

Conclusions

The poster detailed the research on face recognition system and developed a prototype. The prototype was run and used for sample face recognition. Important aspects of the project such as the details of the proposed system, various assets such as system design, ER diagrams, use case diagrams, activity diagrams, etc. were presented. Methods of testing and feasibility were also presented. The conclusion is that the project is feasible, useful, and it can be implemented.

Future Work

Future work will investigate large scale usability. The work also needs to consider connected databases and recognize faces under different conditions.

Acknowledgements

I would like thank my advisor Dr. Jeffrey Duffany for his excellent , exceptional guidance and support in this project .

References

[1] Chen, Yon-Ping, Chong-Hsien Liu, Kuan-Yu Chou, and Shun-Yi Wang. "Real-time and low-memory multi-face detection system design based on naive Bayes classifier using FPGA." *In Automatic Control Conference (CACS), 2016 International*, pp. 7-12. IEEE, 2016.
[2] Kumar, CN Ravi, and A. Bindu. "An efficient skin illumination compensation model for efficient face detection." *In IEEE Industrial Electronics, IECON 2006-32nd Annual Conference on*, pp. 3444-3449. IEEE, 2006, Paris, France,
[3] Mandal, Bappaditya. "Face recognition: Perspectives from the real world." *In Control, Automation, Robotics and Vision (ICARCV)*, 2016 14th International Conference on, pp. 1-5. IEEE, 2016.
[4] Manlises, Cyrel O., Jesus M. Martinez, Jackson L. Belenzo, Czarleine K. Perez, and Maria Khristina Theresa A. Postrero. "Real-time integrated CCTV using face and pedestrian detection image processing algorithm for automatic traffic light transitions." *In Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM)*, 2015 International Conference on, pp. 1-4. IEEE, 2015.
[5] Yang, Shuo, Ping Luo, Chen-Change Loy, and Xiaoou Tang. "From facial parts responses to face detection: A deep learning approach." *In Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3676-3684.
[6] Yang, Shuo, Ping Luo, Chen-Change Loy, and Xiaoou Tang. "Wider face: A face detection benchmark." In Proceedings of the IEEE Conference on Computer Vision and *Pattern Recognition*, 2016, pp. 5525-5533.
[7] Emgu CV (2018, April, 9). *Emgu CV Open CV 3x*. [Online]. Available: http://www.emgu.com/wiki/index.php/Main_Page