Sistema integrado para reconocimiento de rostros

Olga Lucia Echeverry.

Asesor: Camilo Rey, Rafael Garcia.

8 de agosto de 2014

0.1. Resumen

El reconocimiento facial, hoy por hoy, es uno de los campos de investigación más amplios que existen, con diferentes ramas como la biometría, reconocimiento de patrónes, reconocimiento por medio del mapa de las venas del rostro con luz infraroja..etc. Con aplicación en áreas como la medicina, telefónica celular, vigilancia y control de acceso, procesos de investigación y criminalística, software comercial, entre otros.

En este trabajo se propone un sistema que permita, por medio de la utilización de la técnica de reconocimiento PCA y una serie de imágenes contenidas en una base de datos, realizar detección y reconocimiento de rostros, capturando desde una cámara web, el rostro de la persona a reconocer y luego de aplicar un proceso de normalización a la imagen capturada, se realiza match contra las imágenes almacenadas en base de datos de cotejo, retornando la identificación del usuario que reconoce.

Posteriormente, se exponen los principales tipos de técnicas lineales que existen para el reconocimiento de rostros y que son utilizadas en diferentes trabajos de reconocimiento facial. Así mismo se exponen los problemas que se tendrán que enfrentar, como lo son los cambios de iluminación, los cambios en el ángulo de enfoque e la cámara e inclinación vertical de la cabeza de la persona pasando a exponer el diseño propuesto para la solución.

Luego de exponer las técnicas lineales más utilizadas para reconocimiento facial, se realiza el planteamiento general de la solución al problema expuesto. Por último, se realiza entrega de resultados con un análisis entre lo que se planteaba inicialmente y lo que se obtiene de la solución final.

Índice general

Capítulo 1

Principales técnicas de reconocimiento

En este capítulo se presenta una introducción a las principales técnicas de reconocimiento facial, basadas en imágenes fijas y que actualmente son las más utilizadas; dando lugar a la especificación de la técnica a utilizar en este trabajo, y posteriormente en el capítulo II, detallando la misma.

1.1. Imagen digital

En esta sección se realiza una breve introducción a lo que son las imágenes digitales y el mundo del procesamiento de imágenes, esto con el fin de dar paso a las secciones posteriores y permitir que se asocie de forma correcta el procesamiento de imágenes, con las técnicas de reconocimiento, ya que son temas que deben estar correctamente engranados.

1.1.1. Que es una imagen

Definición de una imagen digital: *Una imagen es una función de dos dimensiones f(x,y), donde x e y son coordenadas espaciales, y la amplitud de f en cualquier par de coordenadas, se llama intensidad o nivel de gris de la imagen en ese punto. Cuando x, y, y los valores de amplitud de f, son todos finitos, cantidades discretas, se le llama a la imagen, imagen digital.[7]*

La función que representa una imagen, esta compuesta por dos partes principales que son, la iluminación y la reflectancia, por lo cual se puede decir que:

$$f(x,y) = i(x,y)r(x,y)$$
$$\text{Donde,}$$
$$0<i(x,y)<\infty$$
$$\text{y}$$
$$0<r(x,y)<1$$

(1.1.0)

figura (a) muestra la representación digital de una imagen.

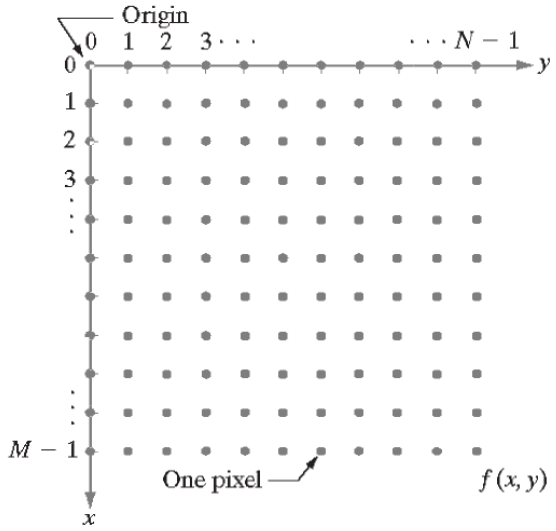


Figura 1.1: (a) Representación digital de una imagen, tomada de [7]

Por tanto un imagen se puede representar como la siguiente matriz:

$$A = \begin{pmatrix} a_{0,0} & a_{0,1} & \cdots & a_{0,N-1} \\ a_{1,0} & a_{1,1} & \cdots & a_{1,N-1} \\ \vdots & \vdots & \vdots & \vdots \\ a_{M-1,0} & a_{M-1,1} & & a_{M-1,N-1} \end{pmatrix} \tag{1.1.1}$$

En el mundo del procesamiento de imágenes, existen muchos temas interesantes a tratar, sin embargo por las limitaciones y el enfoque del presente trabajo, pasaremos directamente a hablar sobre como hallar contornos de una imagen, de algunas de las herramientas matemáticas que existen detrás de dicho proceso.

1.1.2. Detección de bordes en imágenes digitales

Dentro de la segmentación de una imágen se pueden contemplar diferentes tipos de segmentación depeniendo de lo que se requiera para el trabajo en curso, por ejemplo: filtros y suavisado, umbralización, crecimiento de region, Detección de discontinuidades...etc, en los cuales no se profundizará por la limitaciones del proyecto.

- **Segmentación:** Es un proceso en el cual es posible separar o delimitar zonas de la imagen que contienen caracteristicas similares, por ejemplo: niveles de gris. Existen diversos tipos de segmentación dependiendo de lo que se requiera de la imagen. Para este trabajo se utiliza umbralización y detección de bordes, sin embargo se mencionan algunas de las técnicas de segmentación más utilizadas.
- **Filtros de suavizado:** su objetivo consiste en disminuir el foco de una imagen.

- **Umbralización:** es un método de segmentación cuyo objetivo es tomar una imagen que tiene varios niveles de gris, y lograr que la misma tenga solo dos niveles de gris, permitiendo que los niveles de umbralización se den de acuerdo a las necesidades de la operación, ejecutando dicha operación pixel a pixel donde su resultado es una imagen binaria. El umbral determina el valor de cada pixel (0 ó 1).
  Imagen de Lenna

Figura 1.2: Imagen de Lenna donde (a) 256 niveles de gris; (b) Umbral bajo; (c) Umbral alto; (d) Umbral intermedio, tomada de [17]

- **Crecimiento de región:** Los métodos tradicionales de Crecimiento de Regiones se basan en la comparación de niveles de gris de píxeles vecinos, y suelen fallar cuando la región a segmentar contiene intensidades similares a regiones adyacentes. Si se indica una amplia tolerancia, los límites detectados superarán la región a identificar; por el contrario, si la tolerancia se disminuye demasiado, la región identificada será menor que la deseada [20], [19].

- **Detección de discontinuidades:** la cual se puede decir que es la aproximación más común a la detección de bordes.

Calculo de Bordes

A continuación se mencionan las precondiciones matemáticas que se deben cumplir para iniciar el cálculo de bordes de una imagen sobre la función de la misma, estos puntos son tomados de [7], tal como lo indica Gonzales R,C.

- **Primera derivada de la función de una imágen:**

  1. Debe ser cero en segmentos planos, es decir en áreas de valores constantes de nivel de gris.
  2. Debe ser diferentes de cero cuando existe un cambio de nivel de gris.
  3. Debe ser diferente de cero en todos los saltos de nivel de gris.

- **Segunda derivada de la función de una imágen:**

  1. Debe ser cero en segmentos planos, es decir en áreas de valores constantes de nivel de gris.
  2. Debe ser diferentes de cero cuando en el inicio y final de un salto de nivel de gris.
  3. Debe ser diferente de cero en todos los saltos de pendiente constante.

  Dado que se trata de cantidades digitales cuyos valores son finitos, el cambio de nivel de gris máximo posible, también es finito, y la distancia más corta a través de la cual puede ocurrir que el cambio sea entre los píxeles adyacentes.[7]

Definición primera derivada de una función *f(x).*

$$\frac{\partial f}{\partial x} = f(x+1) - f(x) \tag{1.1.2}$$

Definición segunda derivada de una función *f(x).*

$$\frac{\partial^2 f}{\partial x^2} = f(x+1) + f(x-1) - 2f(x) \tag{1.1.3}$$

Las derivadas de primer orden producen un borde grueso, mientras que las derivadas de segundo orden producen un nivel de presición superior sobre la busqueda del borde. las derivadas de primer orden tienen un mejor respuesta sobre los cambios en niveles de gris.

Operador Laplaciano El operador laplaciano es igual a la suma de todas las segundas derivadas parciales dependientes de una variable, cuya ventaja es que puede expresar problemas de $R^2 \rightarrow R$.

En este aparte se presenta la aplicación del Laplaciano en una imagen bi-dimensional (dos dimensiones), a partir de la especificación de la derivada de segundo orden, para permitir que la respuesta sea indiferente de la dirección de las discontinuidades en la imagen, en el sentido de que no afectaría si primero se aplica el filtro y luego se rota el resultado, o al contrario.

Definición segunda derivada de una funcion *f(x,y).*

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \tag{1.1.4}$$

la ecuación anterior debe ser definida en forma discreta, de manera que permita una mayor utilidad para el procesamiento digital de la imagen, a continuacion se describe la evolución

de la misma, tomada de [7].

Definición segunda derivada en dirección *x* *f(x,y)*.

$$\frac{\partial^2 f}{\partial^2 x^2} = f(x+1,y) + f(x-1,y) - 2f(x,y) \tag{1.1.5}$$

Definición segunda derivada en direccion *y* *f(x,y)*.

$$\frac{\partial^2 f}{\partial^2 y^2} = f(x,y+1) + f(x,y-1) - 2f(x,y) \tag{1.1.6}$$

Por lo anterior la implementación del Laplaciano se obtiene de suma de ambas componentes así: [7]

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \tag{1.1.7}$$

Se toma del calculo de:

$$\nabla^2 f = [f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1)] - 4f(x,y) \tag{1.1.8}$$

A partir de lo anterior, la forma básica en la que se utiliza el Laplaciano es la siguiente:

$$g(x,y) = \begin{cases} f(x,y) - \nabla^2 f(x,y) & \text{si el coeficiente central de la mascara Laplaciano es negativo} \\ f(x,y) + \nabla^2 f(x,y) & \text{si el coeficiente central de la mascara Laplaciano es positivo} \end{cases} \tag{1.1.9}$$

Como se puede observar en la imagen posterior, como se aplica el Laplaciano a una imagen cuando su coeficiente central es positivo y negativo, tomada de [7]:

1.2. Tipos de técnicas

A continuación se presentan las principales técnicas lineales, de reconocimiento facial, mostrando la forma como opera cada una, con una breve diferenciación entre los tipos de técnicas.

Actualmente se adelantan trabajos de investigación en reconocimiento facial con la aplicación de diferentes técnicas de reconocimiento, como los métodos basados en imágenes $3D$, basados en video, combinación de técnicas, o técnicas basadas en imágenes fijas que nuestro caso.

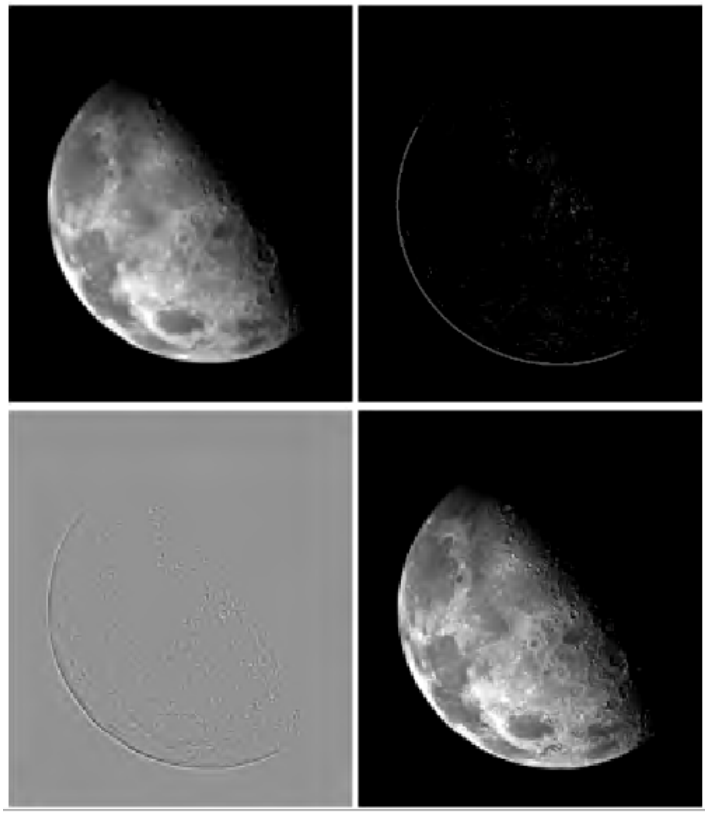


Figura 1.3: (a) Imagen del polo norte de la luna, (b) Se filtró la imagen con el Laplaciano,(c) Imagen a escala con Laplaciano para la visualización,(d) La imagen mejorada mediante el uso de la ecuación. anterior (1.1.9), Tomada de [7]).

- **Técnicas basadas en video.** Las técnicas basadas en video proponen la asociación de cada uno de los recuadros del video, e implica que el video analizado sea de baja resolución, esto, dado su alto coste computacional, esta técnica actualmente es objeto de investigación de intel.

  Los resultados de las investigaciones y trabajos propuestos como el de Gorodnichy [21], donde propone una combinación entre la forma de pensar del ser humano y la técnica implementada, tal que el sistema reconozca un rostro que tenga minimo 12 pixeles de separación entre los ojos del individuo, no evolucionaron ya que estudios posteriores a este, demuestran el alto coste computacional versus los resultados obtenidos, demostrando las ventajas de las técnicas basadas en imágenes fijas.[11]

1.3. Técnicas basadas en imágenes fijas.

Las técnicas basadas en imágenes fijas, en comparación a las técnicas basadas en video tienen un coste computacional mas bajo, pero aun así su coste computacional ha sido objeto de grandes investigaciones para intentar la reducción del mismo.

1.3.1. Linear Discriminant Analysis (LDA)

Es un método supervisado, que utiliza la información obtenida de un conjunto de imágenes de la misma persona, es decir, imágenes de la misma clase [8], desarrollando un conjunto de vectores y permitiendo maximizar la varianza entre clases diferentes, garantizando la discriminación entre las mismas.

LDA convierte un problema de alta dimensionalidad en uno de baja dimensionalidad [15], por medio de una matriz de proyección, cuyas columnas son llamadas *FisherFaces*.

Una de las limitaciones de esta técnica es que se genera singularidad o *singularity Problem*, entre el número de imágenes y la dimensión de las mismas, lo cual se puede controlar la pseudo-inversa de la matriz de covarianza, con el fin de realizar previamente la reducción necesaria de los datos[15].

- **Definición matemática de LDA**

Su objetivo es maximizar la siguiente expresión:

$$J(w) = \frac{(w^T.S_B.w)}{w_T.S_w.w} \tag{1.3.1}$$

Donde:

- S_B es la matriz de dispersión entre clases
- S_w es la matriz de dispersión intra clase
- N_C es el numero de casos dentro de la clase

y se definen como[12]:

$$S_B = \sum_c N_c(\mu - \vec{x})(\mu_c - \vec{x})^T \tag{1.3.2}$$

$$S_w = \sum_c \sum_{i \in c} (\vec{x}_i \mu_c)(\vec{x}_i \mu_c^T) \tag{1.3.3}$$

$$\mu_c = \frac{1}{N_c}. \sum_{i \in c} x_i \tag{1.3.4}$$

$$\vec{x} = \frac{1}{N}. \sum_1 x_i = \frac{1}{N}. \sum_c N_c.\mu_c \tag{1.3.5}$$

Técnica desarrollada por *Belhumeaur, Hespanja, y Kriegman en 1997.* Llamada *Fisherface.*

Dado que los ojos nariz y mejillas, son zonas del rostro que tiene menor variación cuando

ocurre un cambio de expresión, esta técnica no se enfoca en la boca, ya que esta representa un cambio drástico en los cambios de expresión. El enfoque en ojos nariz y mejillas logra un mayor nivel de detección, cuando ocurren cambios en la expresión facial.

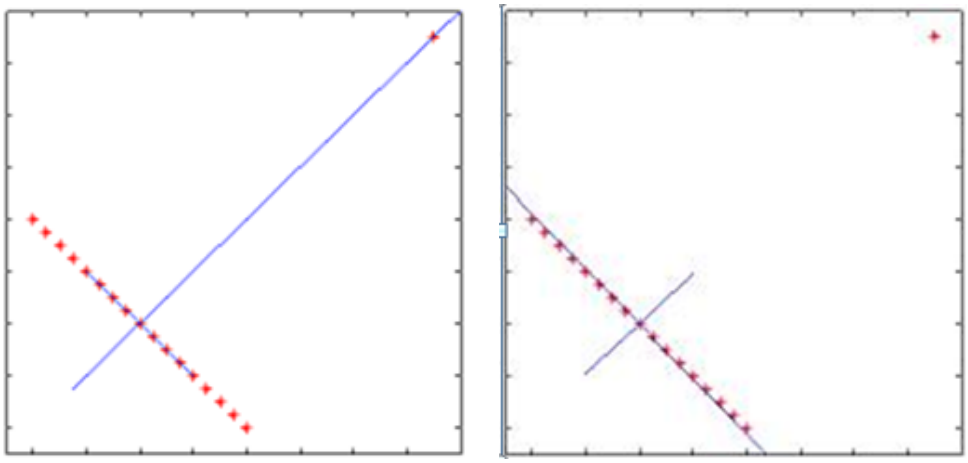
Esta técnica realiza una reducción dimensional, genera una matriz de proyección, y maximiza la distancia entre clases, para garantizar la discriminación entre las mismas [5], adicionalmente se enfoca en el reconocimiento incluyendo las variaciones de luz y de posición del individuo, por lo cual requiere de varias imágenes de muestra del mismo individuo, con diferentes condiciones de luz y de posición, haciendo que su coste computacional sea mayor que el de LDA.

LDA, toma las imágenes de muestra con las condiciones anteriormente descritas, y a partir de la descomposición en *Fisherfaces* sintetiza las variables en la función discriminante para que se realice la clasificación correspondiente, interpola o extrapola las demás condiciones de posición e iluminación que puedan presentarse.

Esta técnica utiliza dos matrices de covarianza, aquella que obedece a las muestras del individuo [?]7) y la de individuos diferentes.

1.3.2. Locality Preserving Projections o Laplancianface (LPP)

Esta técnica, proyecta los datos en la dirección de la máxima varianza, motivo por el cual se pude ver como una variación de PCA. Sin embargo LPP, es menos sensible a los datos atípicos como se puede ver de la Figura 2 y 3 [31], donde a diferencia de PCA, LPP es insensible al dato atípico.



(a) (b)

Figura 1.4: (a) Figura 2 PCA (b) Figura 3 LPP [31]

Los *Laplacianface*, son aproximaciones lineales óptimas a las funciones del operador de *Laplace-Beltrami* [14], que normalmente está relacionado con problemas de minimización

[32],[30].

Teniendo en cuenta que la estructura local de los datos es conservada (Ver figura 4), FLD permite una discriminación entre clases y dado que es un método lineal proporciona rapidéz, permitiendo trabajar con una matriz de transformación.

En la siguiente imagen, se puede observar, como LPP proyectó sobre un espacio dimensional bajo, lo cual, proporciona rapidéz [32].

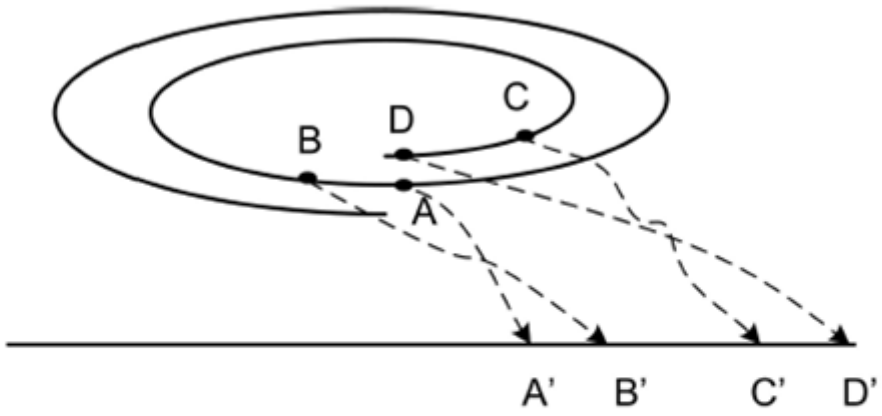


Illustration of dimensionality reduction for two-dimensional data embedded in a nonlinear manifold space with relative position information reserved.

Figura 1.5: Se puede observar que LPP conserva la estructura local de los datos, tomado de: [32]

1.3.3. Discrete Cosine Transform (DCT)

Con relación a PCA, esta técnica tiene un menor coste computacional por lo tanto suele ser muy utilizada en la extracción de características de las imágenes, ya que a diferencia de otras técnicas, no requiere un entrenamiento previo del sistema, además si se incluye una nueva imagen no requiere la ejecución y recalculo de las demás matrices ya que trabaja una matriz de proyección por cada imagen.

Esta técnica es muy utilizada para la compresión de imágenes [30], fué desarrollada por *n. Ahmed, t. Natarajan, and k. R. Rao* en 1974 [1]. La DCT sirve para múltiples aplicaciones en procesamiento de imágenes y conversión de señales, tal como se expresó anteriormente.

Para la conversión de señales, se utiliza una versión de DCT que es aplicable a señales unidimensionales [29], en cambio para el tratamiento de imágenes es necesario utilizar una versión de DCT aplicable para señales bidimensionales.

Para calcular la DCT a una imagen, es decir la 2D-DCT [1], se debe aplicar:

$$X(i,j) = \frac{2}{N} k_i k_j . \sum_{x=0}^{N-1} \sum_{y=0}^{M-1} I(x,y).cos(\frac{\pi(2x+1)i}{2N}).cos(\frac{\pi(2y+1)j}{2M}) \quad (1.3.6)$$

$$k_i, k_j = \begin{cases} \frac{1}{\sqrt{2}} coni, j = 0, 1 otherwise (1.3.7) \end{cases}$$

Donde *I(i,j)* es una imagen de N,M elementos , *X(i,j)* es su transformada [25], [27]. De esta forma se obtiene los coeficientes más importantes de la imagen, reduciendo su dimensionalidad sin afectar de forma notable la imagen misma.

Luego de aplicar la técnica $1D - DCT$, es necesario validar que la matriz resultado sea una matriz ortogonal

1.3.4. Principal Component Analysis (PCA).

Técnica impulsada por Kyrby Sirivich en 1998. Enfocada en el análisis de los principales rasgos de la imágen, por medio de una reducción dimensional de la imagen, para lo cual previamente, se debe preparar la imagen realizando una normalización de acuerdo a los requerimientos de técnica misma.

Con la reducción dimensional, se elimina gran parte de la información que no se requiere, descomponiendo la imagen en componentes ortogonales, llamados *Eigenfases*. Esta técnica requiere que el rostro esté ubicado de frente para lograr resultados óptimos.

PCA se basa en la Transformada de *Karhunen-Loeve (KLT)*, lo cual permite representar la imagen de un rostro utilizando como base, el análisis de muchos rostros, tratando siempre de reducir el número de componentes de la imagen, a lo cual se le llama reducción dimensional, proporcionando un bajo coste computacional.

PCA es una de las técnicas más utilizadas para sistemas de reconocimiento facial, representando las imágenes en un plano $2D$, y clasificandolas con las del mismo individuo, en clases separadas de otros individuos, como lo muestra la figura.

A pesar de ser una técnica antigua, hoy se sigue utilizando la técnica original, como técnicas derivadas a partir de esta, tales como CPCA (PCA por partes), KPCA (PCA basado en kernels específicos) [23], obteniendo resultados mayores en rendimiento computacional en comparación con la técnica original PCA.

- **Transformada de Karhunen-Loeve (KLT).**

Principalmetne basada en las propiedades estadísticas de la imagen, introducida por Karthunen y Loeve, esta transformada representa la imagen en componentes, donde cada componente contiene cierta información de la imagen, de forma que estos son ordenados de mayor a menor contenido de información, descartando aquellos que contienen muy pequeña información y catalogándolo como ruido y descartándolo.

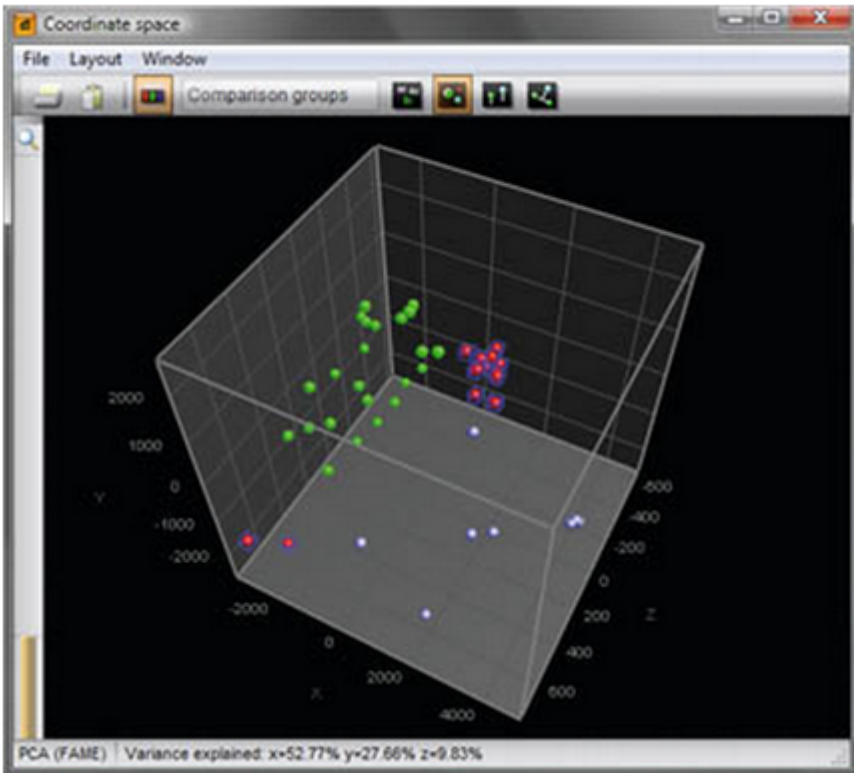


Figura 1.6: clases que caracterizan 3 individuos diferentes, tomado de: [3]

El 95 % de la información más importante se encuentra alojada en los dos primeros componentes principales. La KLT, permite eliminar la correlación entre variables, a su ves permitiendo eliminar información, con fines propios del trabajo que se esté realizando, por ejemplo almacenamiento de las imágenes.

Los pasos generales para la implementación de KLT, son:

1. Calculo de la matriz de covarianza de la secuencia de datos.
2. Diagonalización de la matriz de covarianza, (autovalores).
3. Se calculan los autovectores y autovalores de la matriz de covarianza.

Posteriomente, la información es analizada en el sistema definido por los autovectores y autovalores de la matriz de covarianza, donde los autovalores se encuentran organizados en la diagonal principal de la matriz y los autovectores son ordenados de mayor a menor de acuerdo a los autovalores encontrados.

- **Definición matemática de KLT:**

$$K(k,l) = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} F(m,n) A(m,n;k,l) \tag{1.3.8}$$

La función base A(m,n;k,l), satisface la ecuación:

$$\lambda(k,l)A(m,n;k,l) = \sum_{m'=0}^{N-1}\sum_{n'=0}^{N-1} C_F(m,n;m',n')A(m',n';k,l) \tag{1.3.9}$$

Donde $C_F(m,n;m',n')$ denota la función de covarianza del conjunto de datos de la imagen y $\lambda(k,l)$ es una constante fija para cada punto de la funciona de KLT.[18] Las funciones base dependen de la matriz de covarianza y por tanto no se pueden predeterminar; por lo que existen unicamente, unos pocos casos en los que las soluciones analíticas, estan disponibles. Sin embargo no se profundizará en dichos casos ya que no hacen parte del alcance de este trabajo.

- **Limitaciones del algoritmo PCA:**

1. Variaciones en la iluminación (iluminación de exteriores vs interior fluorescente).
2. Postura (frontal vs inclinada)
3. Expresión (sonrisa vs ceño fruncido)

1.4. Elementos matemáticos

En esta sección se propone la descripción de los elementos matemáticos necesarios para el desarrollo del proyecto, como complementación al capítulo para proporcionar información no solo sobre lo que matemáticamente se requiere para la implemetación de una técnica de reconocimiento facial, sino también para los procesos previos como los es el procesamiento de imágenes.

Se deja al lector libre autonomía sobre la iniciativa de profundizar en la investigación de los elementos aquí mencionados.

1. **Elementos para el procesamiento de las imágenes:**

   Para trabajos en reconocimiento de rostros, es necesario proporcionar un procesamiento de las imágenes ya que estas deben llegar de forma adecuada a la solución. Ejemplo: el ángulo de enfoque, la iluminación, profundidadetc. Por lo cual es necesario retirar aquella información de la imagen que no nos interesa y entregarla en forma precisa.

   Si bien existen muchas herramientas y lenguajes de programación que facilitan el procesamiento de las imágenes, todos estos contienen las implementaciones matemáticas correspondientes para realizar su trabajo. Aquí se pretende exponer con un enfoque técnico revisando cómo sucede el procesamiento de imágenes, claro está de forma general.

   A continuación, algunas áreas en las que existen no solo trabajos finalizados sino una amplia gama de investigaciones en procesamiento de imágenes [23]:

- Procesamiento de vídeo.
- Creación de herramientas para la post-producción de cine digital
- Análisis de imágenes médicas.
- Fotografía digital.
- Visión estéreo.
- Reconstrucción tridimensional a partir de secuencias de vídeo.
- Restauración e interpretación de las imágenes tomadas por satélites.
- Reconocimiento de formas.
- Búsqueda de imágenes en la web.
- Compresión de imágenes.
- Procesamiento de superficies.
- Síntesis de imágenes.
- Simulación para videojuegos.

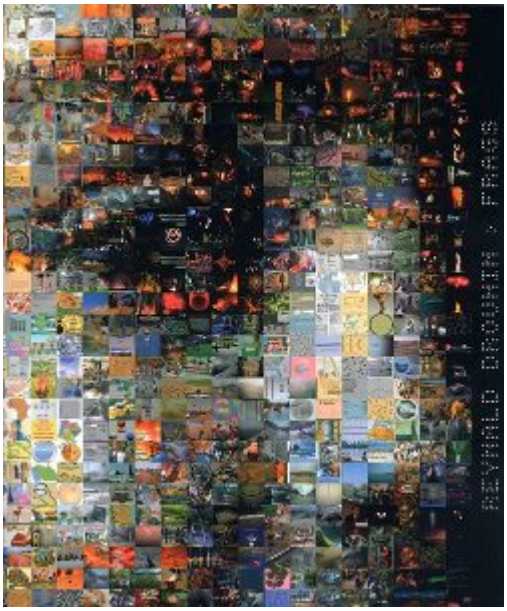


Figura 1.7: Resultado procesamiento de imágenes, tomada de: [6]

El procesamiento de imágenes se puede extender en muchos tipos de técnicas sin embargo las técnicas más generales y mencionadas son:

- Modificación de Color
- Modificación de Imagen
- Generación de efectos.

La técnica de codificación de las imágenes que permite mostrar la imágenes hasta con 16.8 millones de diferentes colores, es llamada RGB.

Modificación del color y detección de orillas:
La modificación del color consiste en modificar los pixeles de la imagen, permitiendo que estos conserven su posición, mientras que la detección de orillas se encarga de recorrer la imagen pixel por pixel y preguntar a su vecino de la derecha y de abajo si tiene su mismo color, de no ser así, entonces se asume que el pixel hace parte del borde de la imagen y lo pone en blanco.

La formula matemática que permite calcular la distancia entre dos vectores, donde cada vector representa un color es:

$$D(C1, C2) = \sqrt{(R1 - R2)^2 + (G1 - G2)^2 + (B1 - B2)^2} \tag{1.4.1}$$

Donde C1 y C2 son los vectores:

C1 = (R1, G1, B1) y C2 = (R2, G2, B2)

La anterior es una de las muchas técnicas que existen para calcular las orillas o bordes de una imagen y realizar el cambio del color.

Procesamiento de imágenes con C# y EmguCv: EmguCV, es un Wrapper de OpenCV, el cual expone interfases de OpenCV permitiendo desde C y otros lenguajes, acceder a las funciones de OpenCV.

OpenCV es una librería desarrolada por Intel, con su versión inicial desde 1999, publicada para trabajos con vision artificial y procesamiento de imágenes.

En la siguente figura se puede observar la arquitectura de EmguCV.

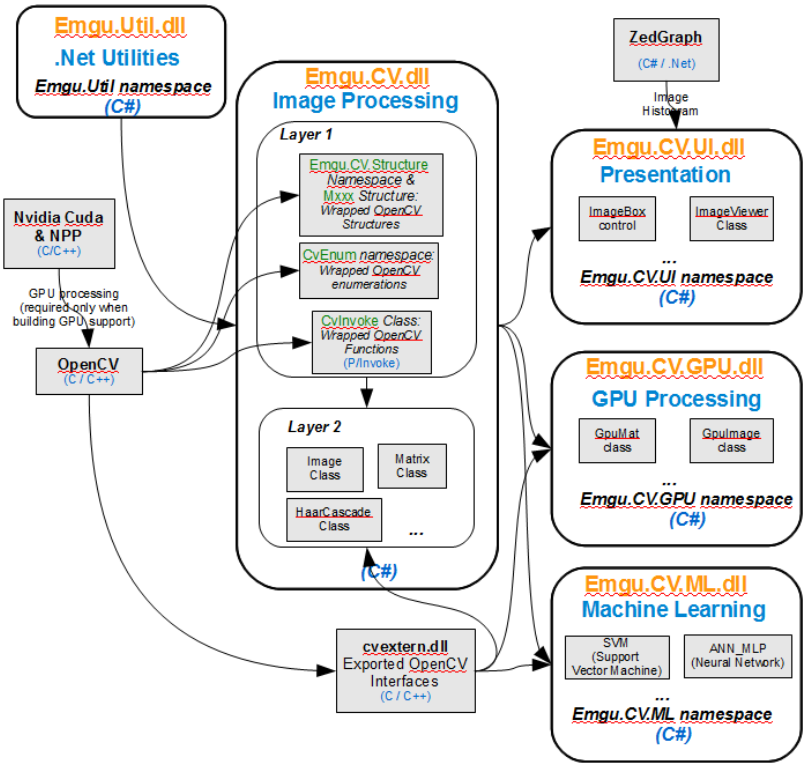


Figura 1.8: Arquitectura EmguCV, imagen tomada de [13]

```
***Convierte a escala de grises***\\
Convert<Gray, byte>()
\\\\
***Cambia el tamaño de la imagen***
Resize(100, 100, Emgu.CV.CvEnum.INTER.CV_INTER_CUBIC)\\
```

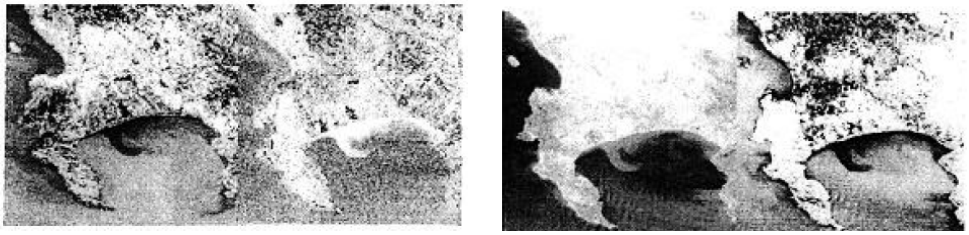
2. **Elementos para reconocimiento facial:**

   En este aparte se relacionan elementos matemáticos, utilizados para el reconocimiento facial, como lo es la *Transformada de Karhunen Loéve*, en la cual se basa PCA.

   Transformada de Karhunen-Loéve (K-L):

   En esta transformada se basa la técnica de reconocimiento facial PCA, para lo cual se expone su correspondiente especificación amtemática.

   Si la imagen original tiene m bandas, la transformada KL de la imagen también tendrá m bandas. Sin embargo, las bandas en la imágen transformada multi-espectral están ordenadas de acuerdo con la información que contienen.[16]



(a) (b)

Figura 1.9: (a) Muestra las componentes con mayor valor (b) Muestra las componentes con menor valor [31]

Matríz de covarianza:

Es una matríz cuadrada de orden *pxp* simétrica.

Definición matemática de la matriz de covarianza:

$$S = \begin{pmatrix} S_1^2 & S_{12} & \dots & S_{1p} \\ S_{21} & S_2^2 & \dots & S_{2p} \\ \vdots & \vdots & \ddots & \\ S_{p1} & S_{p2} & \dots & S_p^2 \end{pmatrix} \tag{1.4.2}$$

Ésta matríz S, se puede calcular directamente de la matriz de datos X. Se define la matriz de datos centrada, como la matríz de datos a la cual, a cada columna, se le resta la media respectiva, así: [21]

$$\vec{X} = X - 1\vec{X}^t \tag{1.4.3}$$

De forma más detallada:

$$\vec{X} = \begin{pmatrix} X_{11} & X_{12} & \dots & X_{1p} \\ X_{21} & X_{22} & \dots & X_{2p} \\ \vdots & \vdots & \dots & \vdots \\ X_{n1} & X_{n2} & \dots & X_{np} \end{pmatrix} - \begin{pmatrix} \vec{X}_1 & \vec{X}_2 & \dots & \vec{X}_p \\ \vec{X}_1 & \vec{X}_2 & \dots & \vec{X}_p \\ \vdots & \vdots & \dots & \vdots \\ \vec{X}_1 & \vec{X}_2 & \dots & \vec{X}_p \end{pmatrix} \tag{1.4.4}$$

Sustituyendo el vector de medias:

$$X = X - \frac{1}{n}11^t X \tag{1.4.5}$$

$$= 1 - (\frac{1}{n}11^t).X \tag{1.4.6}$$

$$= P.X \tag{1.4.7}$$

Donde la matríz P, está definida por:

$$P = (I - \frac{1}{n}11^t) \tag{1.4.8}$$

Entónces la matríz de varianza y covarianza se puede escribir como:

$$S = (\frac{1}{n-1}X^t\vec{X}) \tag{1.4.9}$$

1.5. Elementos de sistemas necesarios:

Dado que se requiere la implementación de una solución que permita el reconocimiento de los rostros, en este esta sección, se propone investigación sobre la librería a utilizar, el lenguaje de programación, ide de desarrollo y elementos necesarios para lograr el objetivo de este proyecto.

1. **Librerías**:

   Se propone OpenCv ya que esta proporciona herramientas útiles y con las cuales se han publicado numerosos trabajos sobre la materia, lo cual facilita el trabajo de investigación.

   - OpenCV:
     Esta librería tiene sus orígenes en los laboratorios de Intel para aplicaciones particulares que se estaban investigando [25], luego fue adaptada para chips intel.

     Las siglas OpenCV, provienen de términos anglosajones Open Source Computer Vision Library [27], destinada para el tratamiento de imágenes de visión por computador. Esta librería permite su utilización en lenguajes como C/C++/JAVA, es open source, para sistemas operativos Windows, Linux, MacOS.

     Sin embargo por facilidad y conocimientos de programación, de decide trabajar con el lenguaje de programación C, desde IDE de desarrollo Visual Studio 2010, pero dado que desde c, no es posible acceder a OpenCV, de forma directa, entonces se toma la decisión de trabajar con EmguCV, que es un Wrapper de OpenCV, que

expone interfaces, para que desde C, sea posible acceder a funciones misma de OpenCV.

- EmguCV:
  Como se menciona en anteriormente, EmguCV, es un Wrapper de OpenCV, que expone interfaces que permiten el acceso a las funciones de OpenCV, su configuración en Visual Studio, consiste en realizar la inclusiones de librerías y referencias especificas de la misma, según el trabajo que se esté realizando.

  Esta librería es presentada en versiones para 32 y 64 bits, eligiendo la versión para 64x, ya que esta plataforma es hace parte del requerimiento inicial del proyecto.

  Haar-Cascade:
  EmguCV, proporciona la función Haar-Cascade, la cual se puede decir que es un clasificador o detector de rostros humanos, esta funciona ha sido probada en miles de rostros humanos, esta función utiliza el algoritmo de Viola and Jones.

  `HAAR_DETECTION_TYPE.DO_CANNY_PRUNING:\\`

  Es la función o detector de rostros, la cual recibiendo la parametrización de acuerdo a las necesidades de trabajo, puede aumentar o disminuir sus tiempos de respuesta. Por ejemplo: si se requiere detectar un solo rostro en una escena, o si se detectarán muchos rostros.

  Este es un ejemplo de la parametrización de la función, tomado de [4]

```
var faces = grayframe.DetectHaarCascade
(haar, 1.4, 4, HAAR_DETECTION_TYPE.DO_CANNY_PRUNING, new Size (25, 25)) [0];
```

  Donde el primer parámetro de la función es el xml o clasificador.

  Existen diferentes publicaciones de trabajos de investigaciones realizados con esta librería para el reconocimiento facial, entre estos se encuentra la publicación de libre uso y distribución de Sergio Andres Guitierrez [17], trabajo que a partir de la investigación realizada se propone para este proyecto que dicho trabajo se aun base y una guía para la implementación de la solución de este proyecto.

  Para más referencias y profundización sobre el manejo de EmguCv, diríjase a:
  http://www.emgu.com/wiki/files/2.4.2/document/Index.html.

Capítulo 2

Modelamiento

Este capítulo contempla el planteamiento general del proyecto incluyendo requerimientos, diseño general de la solución, diagramas UML, así como la implementación de la solución propuesta, con sus algoritmos principales, dando lugar a un análisis final de resultados.

2.1. Planteamiento del proyecto

En esta sección se presenta el problema, con todos los requerimientos iniciales detectados.

2.1.1. Problema a resolver.

La identificación del personal en una organización, se convierte en un problema ya que hoy por hoy, se utilizan muchos métodos intrusivos para esta tarea, por ejemplo las planillas que el personal debe llenar para soportar su asistencia, la toma de datos en las recepciones, verificaciones uno a unoetc. Lo cual conlleva a una inversión importante de esfuerzos y recursos versus unos resultados inconclusos y desorganizados dando pié a la materialización de in riesgo en la seguridad, tanto física como tecnológica.

El proceso de control de asistencia de los participantes en las maratones de programación es un vivo ejemplo del problema, ya que no existe una solución efectiva que proporcione la seguridad, y garantíce que el participante activo, en realidad si es quien previamente fue inscrito, lo cual puede ocasionar baches en la seguridad de las maratones y dar lugar a suplantaciones de identidad, ejemplo que es aplicable a cualquier entidad en cuyos procesos se encuentre el registro y control de asistencia de personal.

Por lo anterior surge la necesidad de un sistema de reconocimiento que permita por medio de la captura de la imagen del rostro de una persona, versus la imágen almacenada por medio de un sistema de logueo, la verificación constante y reconocimiento de la misma, permitiendo identificar si la imagen capturada corresponde a la imagen del participante previamente inscrito.

2.1.2. Alcance

El proyecto en desarrollo tiene como alcance la investigación e implementación de un sistema que permita por medio del control de la cámara del equipo de computo, realizar la captura de la imagen de un usuario, y posteriormente contra una base de datos de cotejo, se compare la imagen capturada, realizando match y luego se determine, si ésta, corresponde al mismo usuario previamente registrado.

2.1.3. Limitaciones al alcance

a) El reconocimiento facial se enfocará a un rostro por cada autenticación.
b) No se realizarán reconocimientos grupales.
c) No se contemplarán módulos de seguridad.

2.1.4. Objetivo General

Proponer, una solución de control de asistencia, no intrusiva, en un entorno empresarial donde se requiera validar la asistencia de personal.

2.1.5. Objetivos específicos

a) Estudiar las técnicas lineales de reconocimiento facial.
b) Proponer el diseño a una solución al problema de control de asistencia de personal.
c) Proponer una solución que permita la implementación de una técnica de reconocimiento facial.

2.1.6. Requerimientos

Se plantean los requerimientos funcionales y no funcionales para el trabajo.

Requerimientos Funcionales

a) La solución de software deberá permitir capturar la imagen del usuario que actualmente se encuentra frente a la cámara web.
b) El sistema debe realizar la normalización de las imágenes capturadas.
c) El sistema debe realizar la normalización de las imágenes de cotejo para poder realizar el match correspondiente.
d) El sistema debe realizar match entre la imagen capturada y las imágenes de cotejo, previamente almacenadas.
e) El sistema debe mostrar un resultado del match realizado, indicando si el usuario fue reconocido dentro de los usuarios previamente registrados.

Requerimientos no funcionales

a) El sistema debe ser compatible con el sistema operativo Windows de 64 bits.

b) El software debe tener una interfaz grafica de usuario, para búsqueda de usuario registrado y la captura de imágenes de cotejo que se almacenaran en base de datos.

c) El software debe tener menús y opciones de fácil acceso al usuario final que le permitan a este dirigirse a cualquier parte del programa de una forma sencilla

d) El sistema tendrá un modulo de registro de usuarios para captura de imágenes de cotejo.

e) El sistema debe permitir consultar la existencia del usuario en un servidor ftp.

f) El sistema debe permitir en envío de imágenes de cotejo a un servidor ftp que contendrá la base de datos de imágenes de cotejo.

2.2. Diseño de la solución

En esta sección se plasma el diseño general de la solución propuesta, incluyendo el Modelamiento UML correspondiente, aportando información relevante a l temática del capítulo, el cual propone el planteamiento del proyecto.

2.2.1. Objetos y relaciones

En este aparte se plasman los diagramas correspondientes al Modelamiento UML, del diseño propuesto.

a) **Diseño de datos:**

Se requiere una base de datos que contenga los códigos y datos generales de los usuarios registrados, para proporcionar información a la búsqueda del usuario cuando se pretenda realizar el proceso de captura de imágenes de cotejo.

b) **UML:**

Diagrama Entidad Relación:

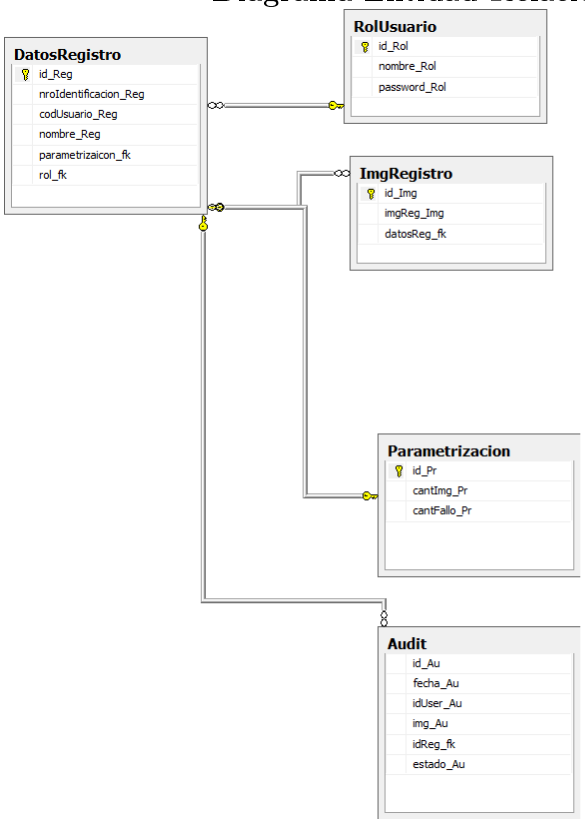


Figura 2.1: Diagrama Entidad Relación de la Base de datos que contiene las imágnes

Diagrama de clases:

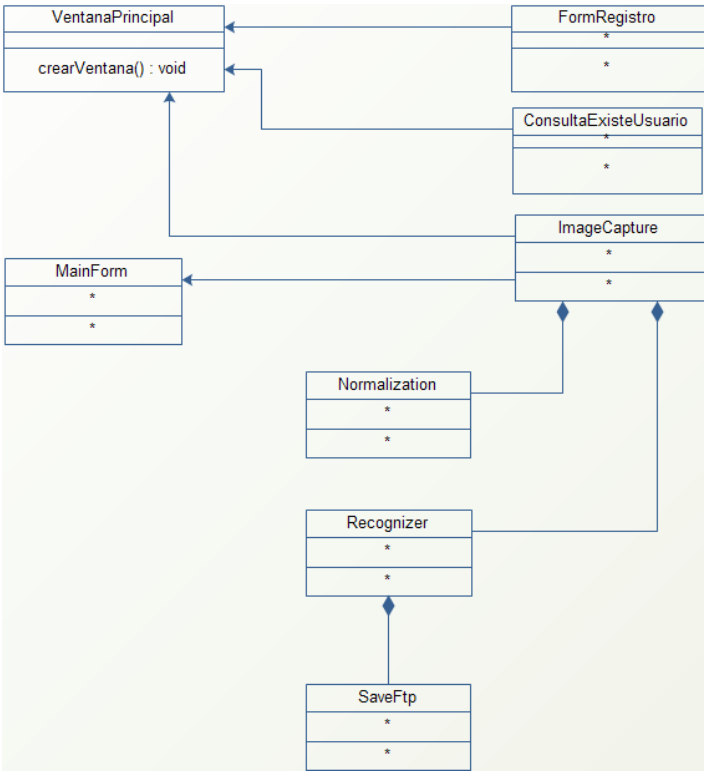


Figura 2.2: Diagrama de clases

Casos de uso:

Caso 01: Busqueda de usuario.
Actores: Usuario encargado de registro.
Pre condición: El usuario debe tener ser un usuario registrado
Post condición: El usuario ha sido encontrado.
Escenario principal, curso lógico de sucesos:

usuario
1. El actor pulsa sobre el botón buscar 2. El sistema muestra una caja de texto para introducir el código de usuario a buscar. 3. El actor introduce el código del usuario. 4. El sistema comprueba que el usuario exista en base de datos y muestra resultado de la búsqueda.

Caso 02: Captura imagen de cotejo.
Actores: Usuario encargado de captura.
Pre condición: El usuario debe tener ser un usuario registrado.
Post condición: Las imágenes de cotejo han sido capturadas y enviadas al servidor ftp.
Escenario principal, curso lógico de sucesos:

usuario
5. El actor pulsa sobre el botón buscar 6. El sistema muestra una caja de texto para introducir el código de usuario a buscar. 7. El actor introduce el código del usuario. 8. El sistema comprueba que el usuario exista en base de datos y muestra resultado de la búsqueda. 9. El sistema habilita el botón capturar. 10. El sistema activa la cámara web. 11. El actor da clic en botón de captura. 12. El sistema captura y envía la imagen al servidor ftp. 13. El usuario repite el paso 7, las veces establecidas por la política de la entidad.

Caso 03: Normalización y reconocimiento.
Actores: Sistema
Pre condición:
1. El usuario debe tener ser un usuario registrado.
2. El usuario debe tener previamente las imágenes de cotejo ya capturadas y almacenadas en el servidor ftp.
3. En la maquina del usuario debe estará instalada la solución de reconocimiento.
Post condición: El match ha sido exitoso.
Escenario principal, curso lógico de sucesos:

usuario
1. El sistema dispara la cámara web y captura la imagen del usuario. 2. El sistema realiza la normalización (Procesamiento de la imagen.) de la imagen capturada. 3. El sistema realiza la normalización (Procesamiento de la imagen.) de las imágenes de cotejo previamente capturadas y almacenadas en el servidor ftp. 4. El sistema realiza el match correspondiente e identifica si la imagen capturada, es del usuario registrado.

Diagrama de casos de uso:

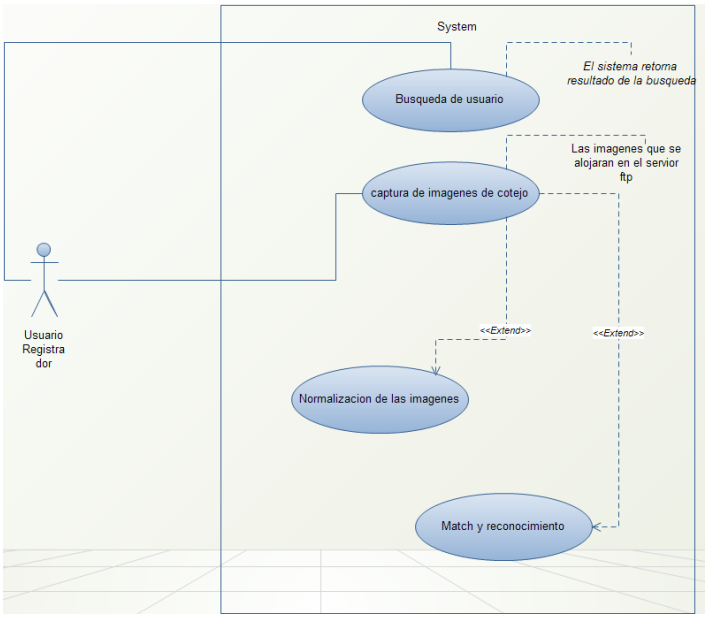


Figura 2.3: Diagrama de casos de uso

2.2.2. Arquitectura

Durante el desarrollo e investigación de este trabajo se han presentado varias propuestas de arquitectura, dentro de las cuales se pretendía aprovechar y por medio del este proyecto dar solución al problema de control de asistencia de los participantes de las

Diagrama de secuencia - caso de uso 01:

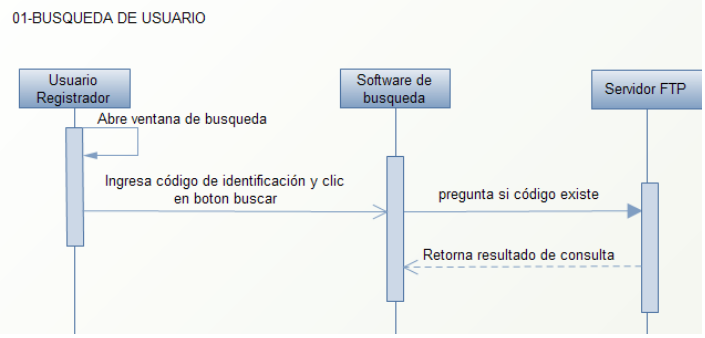


Figura 2.4: Caso de uso 01

Diagrama de secuencia - caso de uso 02:

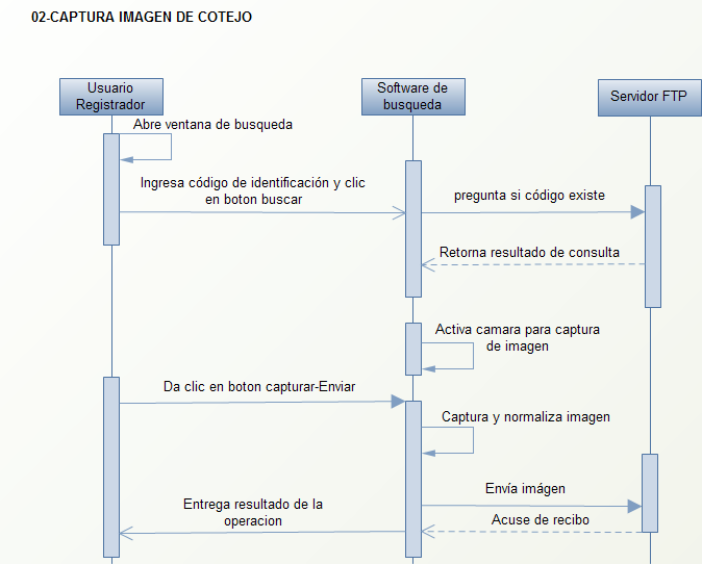


Figura 2.5: Caso de uso 02

maratones de programación.

A partir dichas propuestas se concluye una propuesta final, sin embargo es importante dejar plasmada la evolución que tuvo el trabajo durante la etapa de investigación:

Diagramas de secuencia de la arquitectura del sistema:

Se presentan diagramas de secuencia de propuestas iniciales y propuesta final de la arquitectura de la solución.

Secuencia propuesta 1:

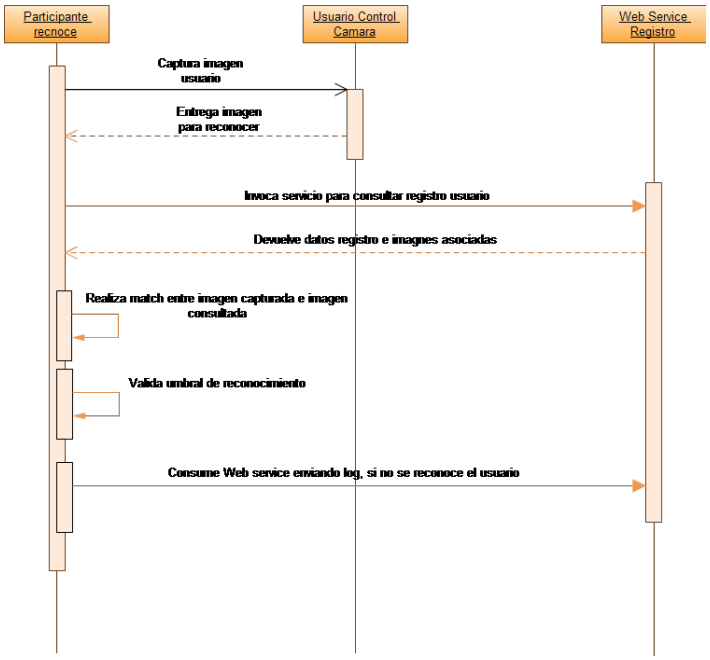


Figura 2.6: Secuencia arquitectura propuesta 1

Secuencia propuesta 2:

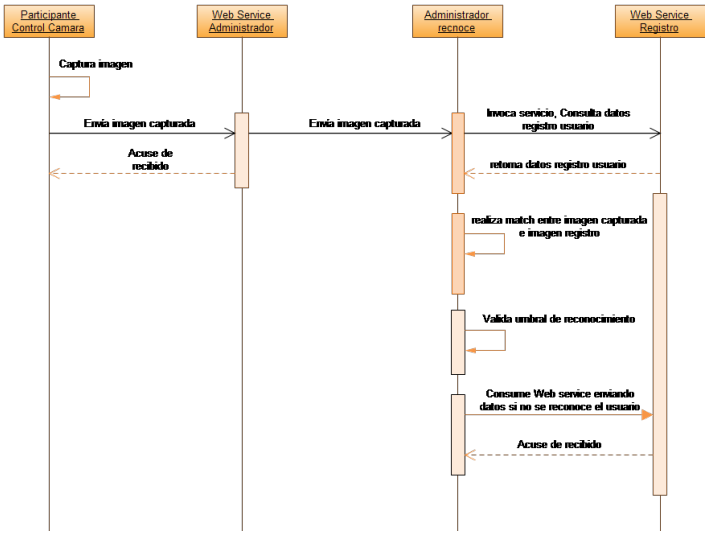


Figura 2.7: Secuencia arquitectura propuesta 2

Secuencia propuesta final:

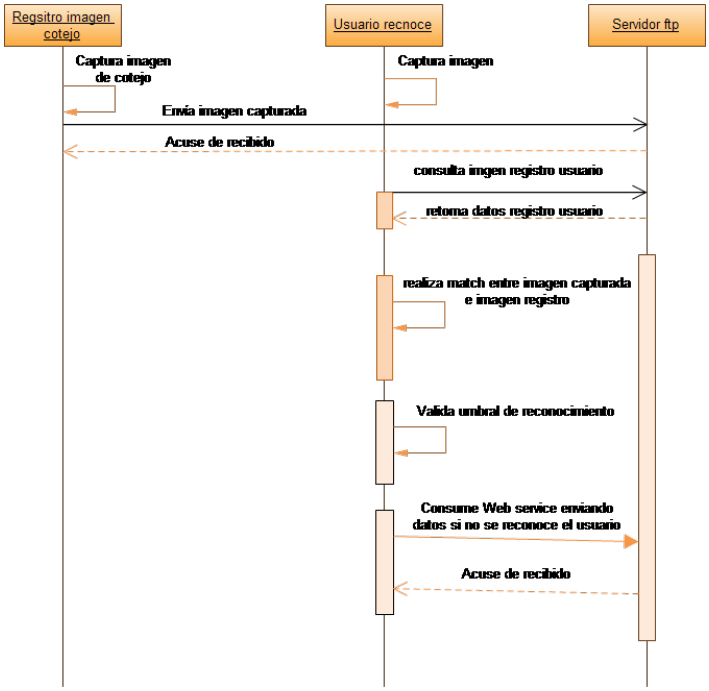


Figura 2.8: Secuencia arquitectura propuesta final

a) **Diagrama de arquitectura propuesta 1**

Figura 2.9: Aquitectura del sistema - Propuesta 1

b) **Diagrama arquitectura general propuesta 2:**

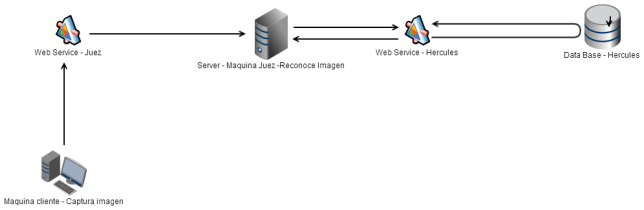


Figura 2.10: Aquitectura del sistema - Propuesta 2

c) **Diagrama arquitectura general propuesta final:**

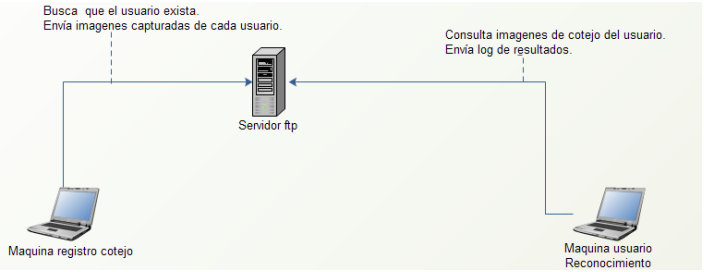


Figura 2.11: Aquitectura del sistema - Propuesta final

d) **Diseño interfaz de usuario:**

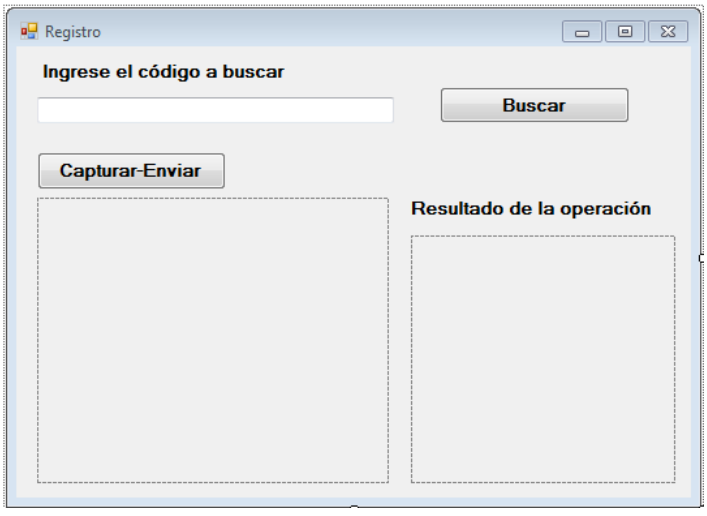


Figura 2.12: Interfaz gráfica de usuario

Capítulo 3

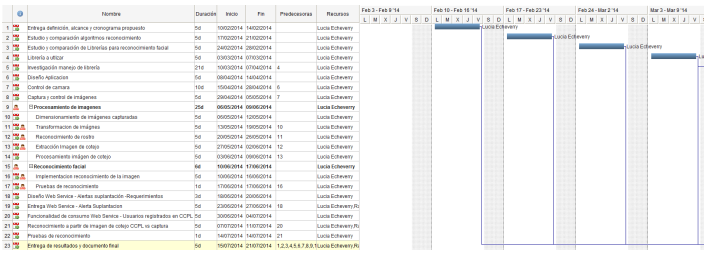
Resultados obtenidos

Este capítulo contiene los resultados obtenidos a partir del trabajo propuesto e investigación realizada en los capítulos 1 y 2, se incluye un análisis de los resultados, pruebas realizadas, y se detallan los cambios entre lo que se propuso inicialmente y los cambios que se presentaron durante el desarrollo de actividades.

3.1. Cronograma

Aquí se plasma el cronograma que se propone al iniciar las actividades de investigación.

Cronograma de actividades:



	Nombre	Duración	Inicio	Fin	Predecesoras	Recursos
1	Entrega definición, alcance y cronograma propuesto	5d	10/02/2014	14/02/2014		Lucia Echeverry
2	Estudio y comparación algoritmos reconocimiento	5d	17/02/2014	21/02/2014		Lucia Echeverry
3	Estudio y comparación de Libreríias para reconocimiento facial	5d	24/02/2014	28/02/2014		Lucia Echeverry
4	Libreríia a utilizar	5d	03/03/2014	07/03/2014		Lucia Echeverry
5	Investigación manejo de librería	21d	10/03/2014	07/04/2014	4	Lucia Echeverry
6	Diseño Aplicacion	5d	08/04/2014	14/04/2014		Lucia Echeverry
7	Control de camara	10d	15/04/2014	28/04/2014	6	Lucia Echeverry
8	Captura y control de imágenes	5d	29/04/2014	05/05/2014	7	Lucia Echeverry
9	Procesamiento de imagenes	25d	06/05/2014	09/06/2014		Lucia Echeverry
10	Dimensionamiento de imágenes capturadas	5d	06/05/2014	12/05/2014		Lucia Echeverry
11	Transformacion de imágenes	5d	13/05/2014	19/05/2014	10	Lucia Echeverry
12	Reconocimiento de rostro	5d	20/05/2014	26/05/2014	11	Lucia Echeverry
13	Extracción imagen de collejo	5d	27/05/2014	02/06/2014	12	Lucia Echeverry
14	Procesamiento imágen de collejo	5d	03/06/2014	09/06/2014	13	Lucia Echeverry
15	Reconocimiento facial	6d	10/06/2014	17/06/2014		Lucia Echeverry
16	Implementacion reconocimiento de la imagen	5d	10/06/2014	16/06/2014		Lucia Echeverry
17	Pruebas de reconocimiento	1d	17/06/2014	17/06/2014	16	Lucia Echeverry
18	Diseño Web Service - Alertas suplantación -Requerimientos	3d	18/06/2014	20/06/2014		Lucia Echeverry
19	Entrega Web Service - Alerta Suplantacion	5d	23/06/2014	27/06/2014	18	Lucia Echeverry,Ri
20	Funcionalidad de consume Web Service - Usuarios registrados en CCPL	5d	30/06/2014	04/07/2014		Lucia Echeverry
21	Reconocimiento a partir de imagen de collejo CCPL vs captura	5d	07/07/2014	11/07/2014	20	Lucia Echeverry,Ri
22	Pruebas de reconocimiento	1d	14/07/2014	14/07/2014	21	Lucia Echeverry
23	Entrega de resultados y documento final	5d	15/07/2014	21/07/2014	1,2,3,4,5,6,7,8,9,1	Lucia Echeverry,Ri

Figura 3.1: Cronograma

3.2. Resultados

En esta sección se muestran los resultados obtenidos a partir de las bases tomadas que fué el proyecto iniciado por *Gutierrez* [26], el cual se utiliza como base para la implementación del presente trabajo y teniendo en cuenta lo planteado en el capítulo 2.

3.2.1. Solución

Teniendo en cuenta lo planteado en el capítulo II, aquí se presentan los resultados obtenidos a nivel de diseño:

1. **Diagrama de clases solucion Reconocimiento:**

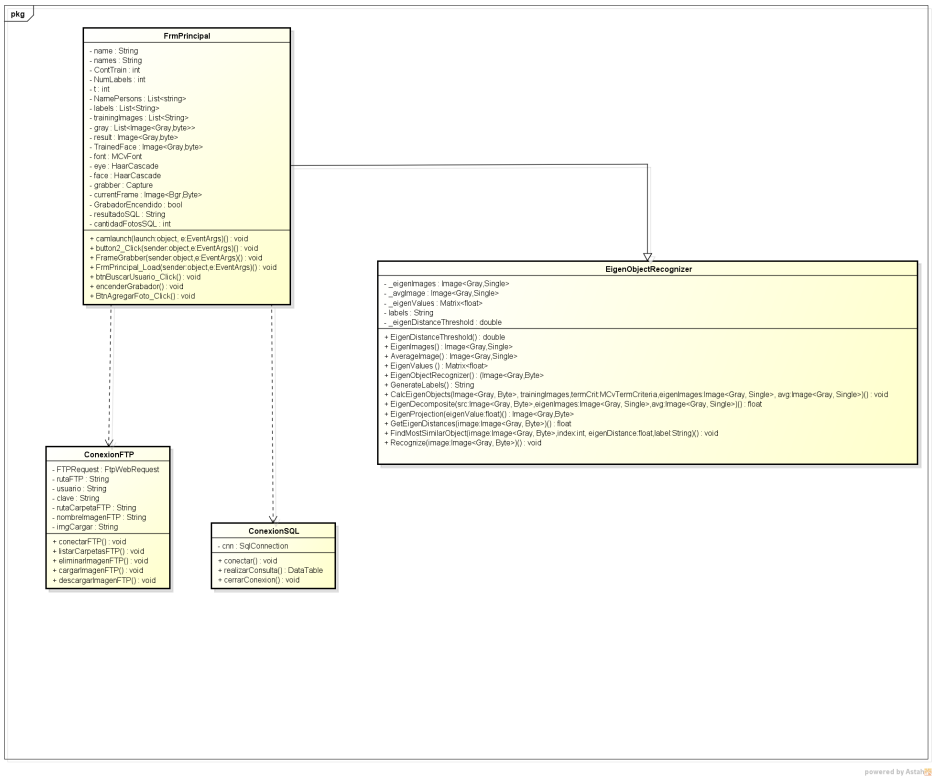


Figura 3.2: Diagrama de clases reconocimiento

3. **El resultado de la propuesta realizada es el siguiente:**

Se puede decir que existen dos módulos que componen la solución de reconocimiento facial los cuales son:

a) Captura de imágenes de cotejo.

b) Solución de detección y reconocimiento.

a) **C**aptura de imágenes de cotejo.

Este modulo se encarga de realizar la consulta hacia el servidor ftp, de la existencia del usuario del cual se pretende realizar la captura de las imágenes; luego de que el servidor ftp retorna el resultado de la consulta del registro del usuario, se habilita la captura de las imágenes del mismo, se capturan las imágenes y son enviadas al servidor ftp.

2. **Diagrama de clases solución registro:**

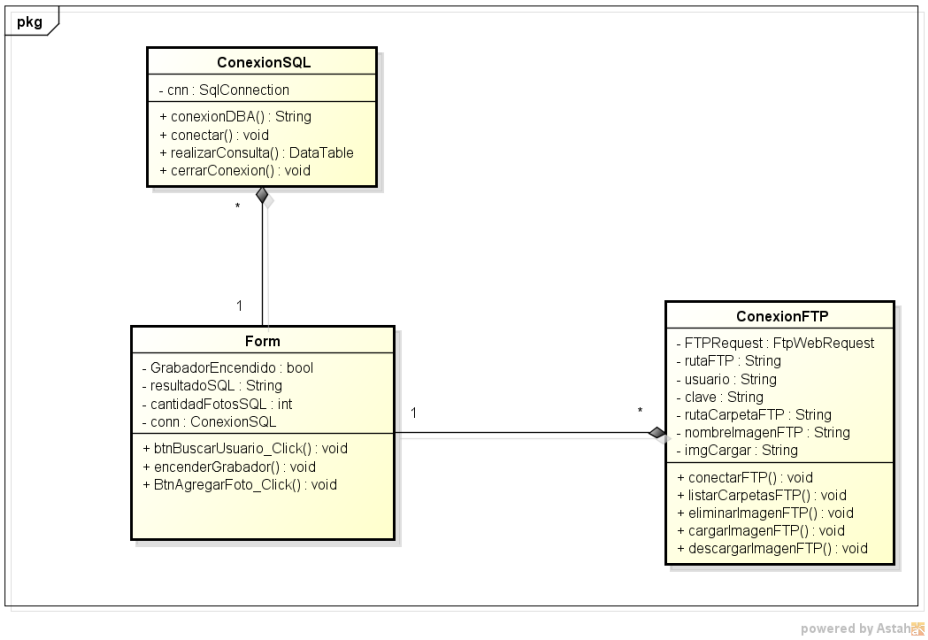


Figura 3.3: Diagrama de clases captura

Las imágenes capturadas se convierten en las imágenes de cotejo de la base de datos contra la que la aplicación de reconocimiento realiza el match para identificar al usuario.

El modulo de busqueda y captura, se relaciona directamente con el diagrama e secuencia de la propuesta en el capitulo II:

d) **Solución de detección y reconocimiento.**

El modulo es el más relevante de este proyecto ya que es el que se encarga de realizar la detección y reconocimiento del usuario. Es importante que el lector tenga presente que las actividades para el modulo 1 y 2 se realizan en momentos diferentes.

Los siguientes son los pre-requisitos para que se pueda obtener un resultado optimo de detección y reconocimiento:

- El usuario a reconocer debe estar registrado en la base de datos del servidor ftp. Para este punto se aclara que no se contempla ningún modulo de registro de usuarios, estos datos de registro deben cargarse a la base de datos como un archivo plano generado de la herramienta que la entidad utilice para el registro de usuarios.
- Se deben haber capturado las imágenes de cotejo del usuario a reconocer.

b) **Interfaz de usuario:**

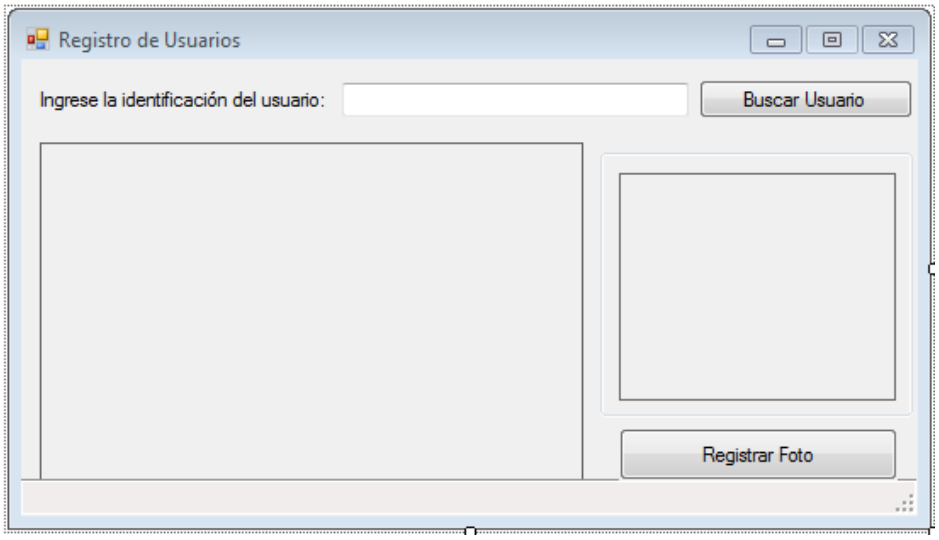


Figura 3.4: Interfaz de usuario

c) **Diagrama de secuencia:**

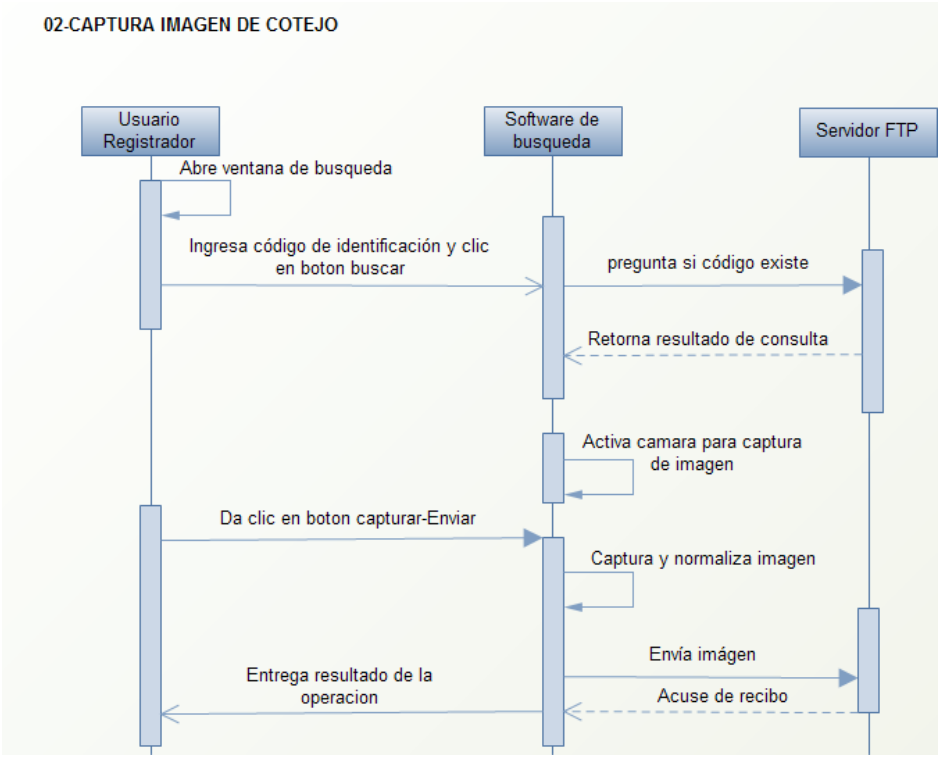


Figura 3.5: Secuencia

A partir de los puntos anteriores la aplicación de reconocimiento puede iniciar su trabajo, aqui se muestran los resultados de la ejecución de la herramienta de reconocimiento.

Se realiza la captura de la imagen:

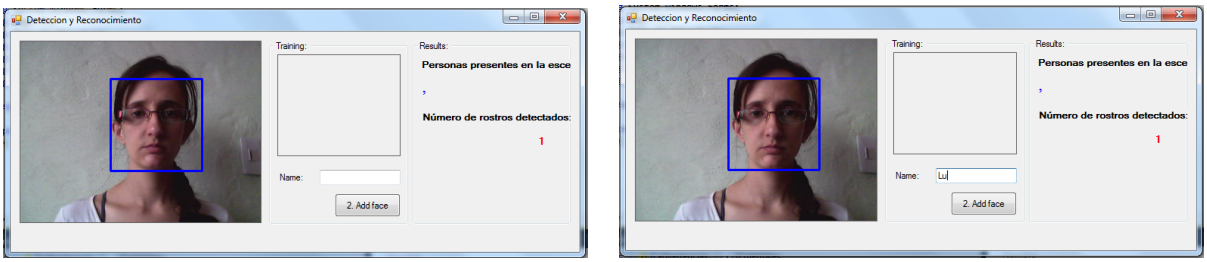


Figura 3.6: En estas imágenes se puede observar que la aplicación realiza la detección del rostro humano.

Imagen agregada:

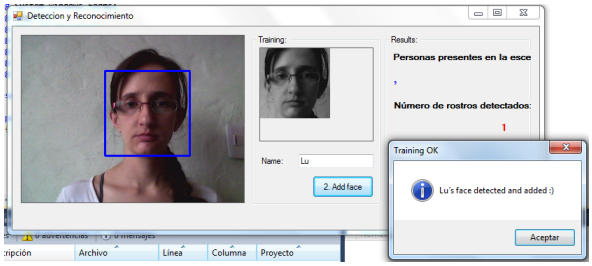


Figura 3.7: Captura de imagen

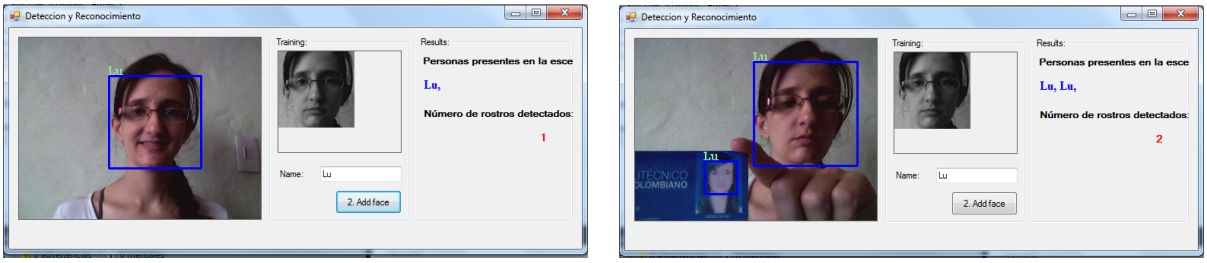
3.2.2. Herramientas usadas

Se utilizaron como herramientas informáticas, Visual Studio 2010, se utilizó equipo de computo de 64x, librería EmguCV, equipo de cómputo con procesador de 64x; elementos necesarios para la construcción de la aplicación.

3.2.3. Algoritmos implementados

Los algoritmos más relevantes para este proyecto son aquellos que se encargan de normalizar la imagen, detectar un rostro y reconocerlo. En esta sección se muestran los métodos

Reconocimiento de rostro:



(a) (b)

Figura 3.8: (a) Detecta rostro capturado, (b) Detecta dos rostros y reconoce la misma persona.

correspondientes.

1. **Método 1**

   `(button2Click()):`

\\\\
Permite detectar un rostro humano captura la imagen y la normaliza y realiza el entrenamiento con las imágenes capturadas.
\\\\
\begin{verbatim}
private void button2_Click(object sender System.EventArgs e){
try
{
//Trained face counter
ContTrain = ContTrain + 1;
//Get a gray frame from capture device
gray = grabber.QueryGrayFrame().Resize(
320 240 Emgu.CV.CvEnum.INTER.CV_INTER_CUBIC
);//320240
//Face Detector
MCvAvgComp[][] facesDetected = gray.DetectHaarCascade(
face
1.2
10
Emgu.CV.CvEnum.HAAR_DETECTION_TYPE.DO_CANNY_PRUNING
new Size(20 20));//2020
//Action for each element detected
foreach (MCvAvgComp f in facesDetected[0])
{

TrainedFace = currentFrame.Copy(f.rect).Convert<Gray byte>();
break;
}
//resize face detected image for force to compare the same size with the
//test image with cubic interpolation type method
TrainedFace = result.Resize(100 100 Emgu.CV.CvEnum.INTER.CV_INTER_CUBIC);
trainingImages.Add(TrainedFace);
labels.Add(textBox1.Text);
//Show face added in gray scale
imageBox1.Image = TrainedFace;
//Write the number of triained faces in a file text for further load
File.WriteAllText(Application.StartupPath + "/TrainedFaces/TrainedLabels.txt" trainingImages.ToArray() .Length.ToString() + "%");
//Write the labels of triained faces in a file text for further load
for (int i = 1; i < trainingImages.ToArray().Length + 1; i++)
{
trainingImages.ToArray()[i 1].Save(Application.StartupPath + "/TrainedFaces/face" + i + ".bmp");
File.AppendAllText(Application.StartupPath + "/TrainedFaces/TrainedLabels.txt"
labels.ToArray()[i 1] + "%");
}
MessageBox.Show(textBox1.Text + "´s Rostro detectado y agregado" "Training OK" MessageBoxButtons.OK MessageBoxIcon.Information);
}
catch
{
MessageBox.Show("Enable the face detection first" "Training Fail"
MessageBoxButtons.OK MessageBoxIcon.Exclamation);
}
}

Método 2 (FrameGrabber()):

Pasa la imagen a escala de grises, y la muestra en pantalla, contiene el criterio de terminación para el reconocimiento, donde se indica el máximo número de iteraciones:

```
void FrameGrabber(object sender, EventArgs e) {
label3.Text = "0";
//label4.Text = "";
NamePersons.Add("");
```

```
//Get the current frame form capture device
currentFrame =
grabber.QueryFrame().Resize(320, 240, Emgu.CV.CvEnum.INTER.CV_INTER_CUBIC);
//Convert it to Grayscale
gray = currentFrame.Convert<Gray, Byte>();

//Face Detector
MCvAvgComp[][] facesDetected = gray.DetectHaarCascade(
face,
1.2,
10,
Emgu.CV.CvEnum.HAAR_DETECTION_TYPE.DO_CANNY_PRUNING,
new Size(20, 20));

//Action for each element detected
foreach (MCvAvgComp f in facesDetected[0])
{
t = t + 1;
result = currentFrame.Copy(f.rect).Convert<Gray, byte>().Resize(
100, 100, Emgu.CV.CvEnum.INTER.CV_INTER_CUBIC);
//draw the face detected in the 0th (gray) channel with blue color
currentFrame.Draw(f.rect, new Bgr(Color.Blue), 2);


if (trainingImages.ToArray().Length != 0)
{
//TermCriteria for face recognition with numbers of trained images like maxIteration
MCvTermCriteria termCrit = new MCvTermCriteria(ContTrain, 0.001);

//Eigen face recognizer
EigenObjectRecognizer recognizer = new EigenObjectRecognizer(
trainingImages.ToArray(),
labels.ToArray(),
3000,
ref termCrit);

name = recognizer.Recognize(result);

//Draw the label for each face detected and recognized
currentFrame.Draw(name, ref font, new Point(f.rect.X - 2, f.rect.Y - 2),
new Bgr(Color.LightGreen));

}

NamePersons[t-1] = name;
NamePersons.Add("");
```

```
//Set the number of faces detected on the scene
label3.Text = facesDetected[0].Length.ToString();


//Set the region of interest on the faces

gray.ROI = f.rect;
MCvAvgComp[][] eyesDetected = gray.DetectHaarCascade(
eye,
1.1,
10,
Emgu.CV.CvEnum.HAAR_DETECTION_TYPE.DO_CANNY_PRUNING,
new Size(20, 20));
gray.ROI = Rectangle.Empty;
foreach (MCvAvgComp ey in eyesDetected[0])
{
Rectangle eyeRect = ey.rect;
eyeRect.Offset(f.rect.X, f.rect.Y);
currentFrame.Draw(eyeRect, new Bgr(Color.Blue), 2);
}

}
t = 0;

//Names concatenation of persons recognized
for (int nnn = 0; nnn < facesDetected[0].Length; nnn++)
{
names = names + NamePersons[nnn] + ", ";
}
//Show the faces procesed and recognized
imageBoxFrameGrabber.Image = currentFrame;
label4.Text = names;
names = "";
//Clear the list(vector) of names
NamePersons.Clear();

}
```

Método 3 (EigenObjectRecognizer()):

Crea el objeto de reconocimiento utilizando la información del entrenamiento, retornando el objeto más similar. Obtien los eigenValues, y eigenFaces, para realizar match.

```
public EigenObjectRecognizer(Image<Gray, Byte>[] images,
String[] labels, double eigenDistanceThreshold, ref MCvTermCriteria termCrit)
{
Debug.Assert(images.Length ==
labels.Length, "The number of images should equals the number of labels");
Debug.Assert(eigenDistanceThreshold >= 0.0,
"Eigen-distance threshold should always >= 0.0");

CalcEigenObjects(images, ref termCrit, out _eigenImages, out _avgImage);

_eigenValues = Array.ConvertAll<Image<Gray, Byte>,
Matrix<float>>(images, delegate(Image<Gray, Byte> img)
{
return new Matrix<float>(EigenDecomposite(img, _eigenImages, _avgImage));
});

_labels = labels;

_eigenDistanceThreshold = eigenDistanceThreshold;
}
```

Método 4:

EigenProjection() Obtiene los eigenValues creando la matriz de proyección de la imagen.

```
public Image<Gray, Byte> EigenProjection(float[] eigenValue)
{
Image<Gray, Byte> res = new Image<Gray, byte>(_avgImage.Width, _avgImage.Height);
CvInvoke.cvEigenProjection(
Array.ConvertAll<Image<Gray, Single>, IntPtr>(_eigenImages, delegate(Image<Gray, Single> img
{ return img.Ptr; }),
eigenValue,
_avgImage.Ptr,
res.Ptr);
return res;
}
```

Metdo 5:

GetEigenDistances ()

Obtiene las eigenDistancias, y con esto puede clasificar las clases para las imágenes, Ver[Cap1].

```
public float[] GetEigenDistances(Image<Gray, Byte> image)
{
using (Matrix<float> eigenValue = new Matrix<float>(EigenDecomposite(image, _eigenImages, _a
return Array.ConvertAll<Matrix<float>, float>(_eigenValues,
delegate(Matrix<float> eigenValueI)
{
return (float)CvInvoke.cvNorm(eigenValue.Ptr,
eigenValueI.Ptr, Emgu.CV.CvEnum.NORM_TYPE.CV_L2, IntPtr.Zero);
});
}
```

Metodo 6:

FindMostSimilarObject() Obtiene la imagen a ser examinada y compara contra la base de datos, encontrando el objeto con más similitudes.

```
public void FindMostSimilarObject(Image<Gray, Byte> image,
out int index, out float eigenDistance, out String label)
{
float[] dist = GetEigenDistances(image);

index = 0;
eigenDistance = dist[0];
for (int i = 1; i < dist.Length; i++)
{
if (dist[i] < eigenDistance)
{
index = i;
eigenDistance = dist[i];
}
}
label = Labels[index];
}
```

Metodo 7:

Recognize(). Realiza el reconocimiento y retorna el resultado.

```
public String Recognize(Image<Gray, Byte> image)
{
int index;
float eigenDistance;
String label;
FindMostSimilarObject(image, out index, out eigenDistance, out label);
```

```
return (_eigenDistanceThreshold <= 0 || eigenDistance
< _eigenDistanceThreshold )  ? _labels[index] : String.Empty;
}
```

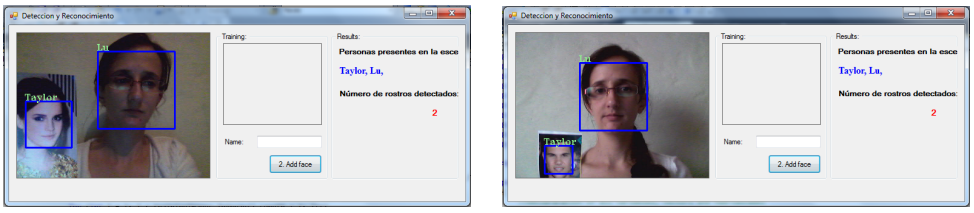
3.2.4. Etápa de pruebas

Aquí se mostrarán los diferentes casos de pruebas funcionales realizadas, de los cuales algunos son pruebas negativas realizadas con el fin de confirmar las restricciones de la técnica utilizada, como también mostrar los casos exitosos de la misma, mencionados en el capítulo 1.

1. Caso 1:

   Cuando no se tienen al menos tres imágenes del usuario con variación de la posición y los gestos, este puede ser confundido con otro usuario y los resultados son imprecisos, sin embargo esto no es una sorpresa ya que PCA utiliza el entrenamiento del sistema el cual se debe realizar con varias imágenes del sujeto:

   En este caso Taylor solo tiene una imágen capturada en base de datos de cotejo y la imágen de Emma no ha sido capturada,pero aún así se reconoce a Emma como Taylor, lo cual es incorrecto, esto sucede porque Taylor no tiene la cantidad suficiente de imágenes de cotejo captuadas:



(a) (b)

Figura 3.9: (a) Resultado correcto (b) Resultado incorrecto

2. Caso 2:

   Detección de ambos rostros que pertenecen a un mismo individuo y su resutado es correcto:

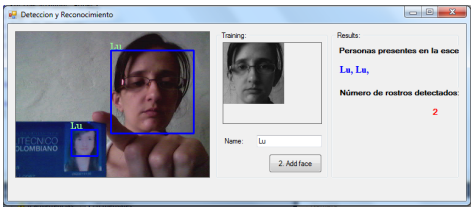
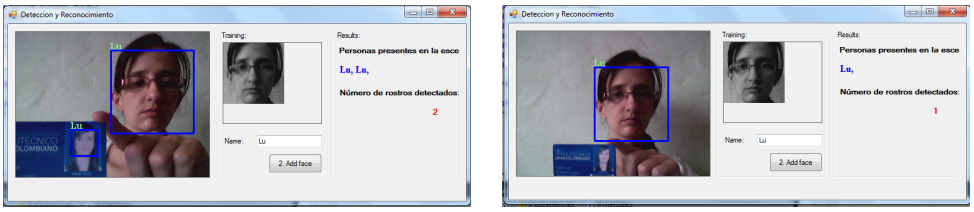


Figura 3.10: Reconocimiento correcto

3. Caso 3:

   Se comprueba una de las restricciones de la técnica y es la distancia, como se puede observar en la figura (a) se detecta y reconoce, pero en la figura (b) solo se detecta ya que la distancia de la cámara es mayor.

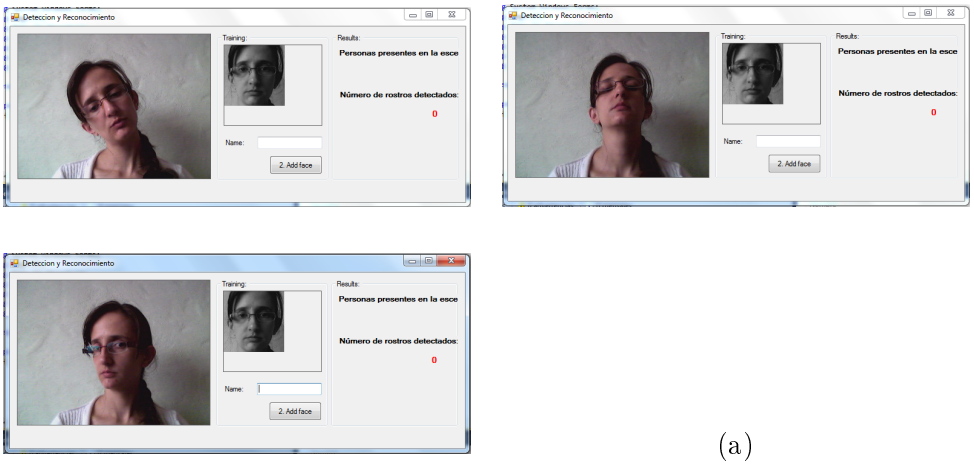


(a) (b)

Figura 3.11: (a) Distancia corta reconoce (b) Imágen muy alejada no detecta

4. Caso 4:

   Se puede apreciar que cuando hay una inclinación importante, no se reconoce que existe un rostro humano, con lo que se confirma una de las restricciones del capítulo 1 (inclinación):



(a)

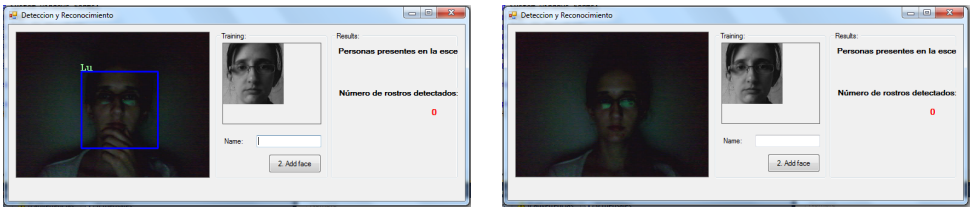
(b)

(c)

Figura 3.12: (a) No detecta rostro (b) No detecta rostro (c) No detecta rostro

5. Caso 5:

   A pesar de las restricciones de iluminación que contiene la técnica, en condiciones de iluminación no óptimas, en algunos casos se reconoce el rostro, en otros no.



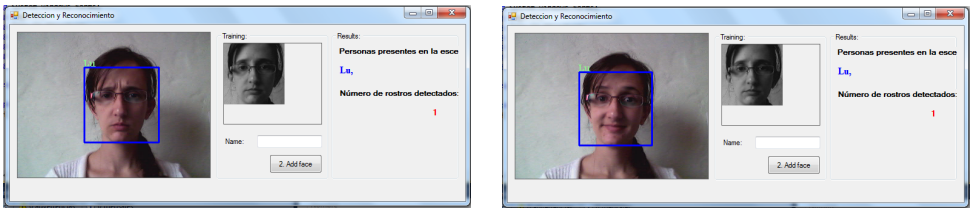
(a)

(b)

Figura 3.13: (a) Reconoce a pesar de la mala iluminación, (b) No detecta rostro

6. Caso 6:

   Cuando se producen cambios de expresión, el resultado es positivo.

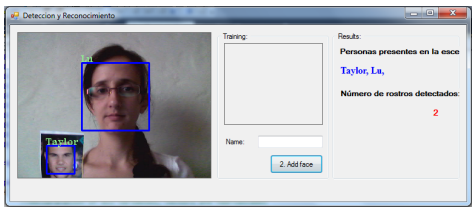


(a) (b)

Figura 3.14: (a) expresión 1 y reconoce, (b) Expresión 2 y reconoce

7. Caso 7:

   Luego de capturar la imagen del rostro de Taylor, se pone a prueba el reconocimiento y la discriminación entre individuos:



(a)

Figura 3.15: Imagen de Taylor con reconocimiento

8. Caso 9:

   Se realiza reconocimiento del usuario con gafas y sin gafas, incluyendo la mano en la boca, con el fin de validar si esto causa un alto impacto.

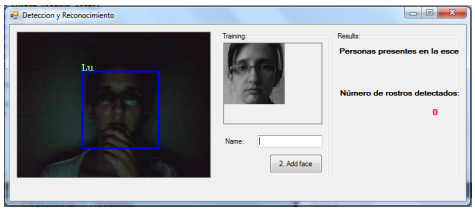


Figura 3.16: (a) Cubriendo la boca y ahora sin gafas

3.3. Conclusiones.

1. La técnica de Análisis de Componentes Principales, es una técnica robusta que puede ser utilizada en proyectos de una mayor extensión, como por ejemplo una combinación entre técnicas de inteligencia artificial y PCA, lo cual daría resultados aún mejores.

2. Las variaciones de iluminación, distancia e inclinación del rostro, deben ser en lo posible mínimas, para proporcionar al sistema condiciones de éxito en la detección y reconocimiento del rostro.
3. Dadas las exigencias a nivel computacional, cuando se ejecuta la aplicación, se confirma que fué una decisión acertada, que se ejecute de forma independiente en la máquina donde estará el usuario a monitorear.
4. El pre-procesamiento de las imágenes influye de forma significativa, el desempeño de los sistemas de reconocimiento, este pre-procesamiento puede estar enfocado a mejorar las condiciones de iluminación, o la correcta alineación de las imágenes, sin embargo dependiendo de las exigencias en el procesamiento puede aumentar o disminuir el coste computacional.
5. Existe un riesgo que es innegable y es el porcentaje de rechazo y falsa aceptación, sin embargo es un riesgo que se corre en la mayoría de trabajos de esta índole, ya que hasta el momento no se ha logrado implementar una tecnica de reconocimiento que sea 100 % efectiva. Estos porcentajes de rechazo están dados dependiendo del tipo de imágnes que se utilicen, la combinación de técnicas...etc.
6. El uso de una cantidad muy alta de imágenes para un usuario aumenta el porcentaje de reconocimiento pero disminuye el rendimiento de la aplicación.

3.4. Trabajo futuro.

1. Trabajar en la escalabilidad de la aplicación, para proporcionar que corra en diferentes plataformas y arquitecturas, recomendable que se trabaje para que corra en plataformas cruzadas.
2. Ampliar la investigación, e implementar un algoritmo que permita tener una tasa más alta de aciertos, esto con el fin de reducir el riesgo de los falsos positivos o incapacidad de reconocimiento en conediciones no optimas de iluminación, inclinación o distancia, proporcionando tolerancia a fallos.
3. Mejorar el pre-procesamiento de las imágenes, con el fin proporcionar una mejor tasa de rendimiento al sistema.
4. Retomar el mismo trabajo con un enfoque más especifico, para intentar que sea implementado en las maratones de programación, en conjunto con las áreas y entidades interesadas y que administran dicho proceso.
5. Trabajar en la implementación de un modulo de seguridad.
6. Proporcionar integración entre la base de datos y el sitema de registro de usuarios, de manera que el regsitro de cada usuario sea poblado de forma automática.

Bibliografía

[1] T. Natarajan Ahmed, N. and K. R. Rao. *On image processing and a discrete cosine transform. IEEE Transactions on Computers C-23.*

[2] Carlos Perez German Ojeda ICP Calle Andres, Zayra Perez. *Análisis e implementación de la Transformad e Karhunen-Loéve (K-L), en datos sísmicos de reflexión.*

[3] Manage. Analyze. Discover. *Applied Maths Software.*

[4] Roberto Javier Soto Entrena. *FACE DETECTION IN C.* Kraków 2012.

[5] R. A. Fisher. *The use of multiple measures in taxonomic problems.*

[6] Jorge Rafael Valvert Gamboa. *Métodos y técnicas de reconocimiento de rostros en imágenes digitales bidimencionales.* 2006.

[7] E Gonzales R, C; Woods R. *Digital Image Processing.* 1992.

[8] Ricardo Gutierrez-Osuna. *Introduction to Pattern Analysis.*

[9] Eliseo Martínez H. *Tratamiento matricial de los datos multivariantes.*

[10] Dr. J.B. Hayet. *Introduccion al uso de OpenCV.* 2007.

[11] Ph. D Henry Arguello Fuentes. Recognition systems based in the facial image. 2011.

[12] Roger Gimeno Hernández. *Estudio de técnicas de reconocimiento facial.*

[13] Intel. Emgucv. http://www.emgu.com/wiki/index.php/Main$_{P}age$.

[14] David Lambraño Jaramillo. *Aproximación del operador Laplace-Beltrami por mallas y nubes de puntos.*

[15] Qi Li Jieping Ye, Ravi Janardan. *Two-Dimensional Linear Discriminant Analysis.*

[16] Karhunen-Lóeve. *Transformada de Karhunen-Loeve.*

[17] Lenna. Imagen de lenna. 1972.

[18] Alfonso Marín. Transformada de karhunen-loéve. http://alfonsomartin.es/tdi/pdf/karhunen.pdf.

[19] Moler Emilce Meschino Gustavo. *Algoritmo de Crecimiento de Regiones con característicos de texturas.*

[20] O. Monga. *An Optimal Region Growing Algorithm for Image Segmentation.* 1987.

[21] Gorodnichy D. O. *Video-based framework for face recognition in video.* 2005.

[22] M. Turk A. Pentland. *Eigenfaces for Recognition.* 1991.

[23] Jr. P.C. Prospero C. Narval. *Recognizing Faces using Kernel Eigenfaces and Support Vector Machines.*

[24] Carlos Medrano Raúl Igual. *Algoritmos de seguimiento de objetos en tiempo real, Tutorial de OpenCV.* 2008.

[25] Iain Matthews Ralph Gross and Simon Baker. *Appearance-Based Face Recognition and Light-Fields. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2002.*

[26] Colombia Sergio Andrés Guitérrez Rojas Bucaramanga. Multiple face detection and recognition in real time. http://www.codeproject.com/Articles/239849/Multiple-face-detection-and-recognition-in-real-ti.

[27] Borja Peláez Fernández. U.P.C. *Reconocimiento de caras en entornos no controlados.* 2012.

[28] Universitat Pompeu Fabra Vincent Caselles Costa. Las matemáticas y el procesamiento de imágenes. http://www.madrimasd.org/blogs/matematicas/2008/06/06/93964.html, 2013.

[29] Andrew B. Watson. *Image Compression using the Discrete Cosine Transform. NASA Ames Research. Center, Florida, United States of America. Mathematica Journal.* 1994.

[30] Partha Niyogi Xiaofei He. *Locality Preserving Projections.*

[31] Yuxiao Hu Xiaofei He*, Shuicheng Yan and Hong-Jiang Zhang. *Learning a Locality Preserving Subspace for Visual Recognition. Proceedings of the Ninth IEEE International Conference on Computer Vision.* 2003.

[32] Richard Rose Yun Tang. *A study of using locality preserving projections for feature extraction in speech recognition. Mcgill university.*