UNIVERSIDAD POLITÉCNICA SALESIANA

UNIDAD DE POSGRADOS

MAESTRÍA EN CONTROL Y AUTOMATIZACIÓN INDUSTRIALES

Tesis previa a la obtención
del Grado de Magister
en Control y
Automatización Industriales

"IMPLEMENTACIÓN DE TÉCNICAS DE IDENTIFICACIÓN DE OBJETOS APLICADAS AL RECONOCIMIENTO DE ROSTROS EN VÍDEO VIGILANCIA DEL SIS-ECU-911"

Autor:
Juan Carlos Jiménez Encalada.

Director:
Vladimir Espartaco Robles Bykbaev.

UNIVERSIDAD POLITÉCNICA SALESIANA
UNIDAD DE POSGRADOS

MAESTRÍA EN CONTROL Y AUTOMATIZACIÓN INDUSTRIALES

Autor:
Juan Carlos Jiménez Encalada.

Director:
Vladimir Espartaco Robles Bykbaev.

"IMPLEMENTACIÓN DE TÉCNICAS DE IDENTIFICACIÓN DE OBJETOS APLICADAS AL RECONOCIMIENTO DE ROSTROS EN VÍDEO VIGILANCIA DEL SIS-ECU-911"

Este trabajo de tesis presenta una revisión de las principales metodologías de reconocimiento de rostros, a más realiza una aproximación para el reconocimiento de rostros a través de métodos por descriptores, normalmente utilizados para el reconocimiento de objetos

En el presente trabajo se ha utilizado software libre y es aprovechada la librería de código OpenCV para el análisis y tratamiento de las imágenes. Los resultados que son presentados podrán servir a posterior para el desarrollo de nuevas técnicas híbridas de reconocimiento de rostros.

“IMPLEMENTACIÓN DE TÉCNICAS DE IDENTIFICACIÓN DE OBJETOS APLICADAS AL RECONOCIMIENTO DE ROSTROS EN VÍDEO VIGILANCIA DEL SIS-ECU-911”

‘‘IMPLEMENTACIÓN DE TÉCNICAS DE IDENTIFICACIÓN DE OBJETOS APLICADAS AL RECONOCIMIENTO DE ROSTROS EN VÍDEO VIGILANCIA DEL SIS-ECU-911”

AUTOR:

Juan Carlos Jiménez Encalada
Ingeniero Electrónico
Licenciado en Docencia Técnica.
Egresado de la Maestría en Control y Automatización Industriales

DIRIGIDO POR:

Vladimir Espartaco Robles Bykbaev
Ingeniero en Sistemas
Máster en Inteligencia Artificial, Reconocimiento de Formas e Imagen Digital.
Estudiante de doctorado en Tecnología de la Información y Comunicación de la Universidad de Vigo

CUENCA ECUADOR

Datos de catalogación bibliográfica

JIMENEZ ENCALADA JUAN CARLOS

"IMPLEMENTACIÓN DE TÉCNICAS DE IDENTIFICACIÓN DE OBJETOS APLICADAS AL RECONOCIMIENTO DE ROSTROS EN VÍDEO VIGILANCIA DEL SIS-ECU-911"

Universidad Politécnica Salesiana, Cuenca – Ecuador, 2015

MAESTRÍA EN CONTROL Y AUTOMATIZACION INDUSTRIALES

Formato170 x 240 mm Páginas: 120

Breve reseña de los autores e información de contacto

Autor:

JUAN CARLOS JIMÉNEZ ENCALADA
Ingeniero Electrónico
Licenciado en Docencia Técnica.
Egresado de la Maestría en Control y Automatización Industriales.
jotace11@yahoo.com.mx

Dirigido por:

VLADIMIR ESPARTACO ROBLES VYKBAEV
Ingeniero en Sistemas
Máster en Inteligencia Artificial, Reconocimiento de Formas e Imagen Digital.
Estudiante de doctorado en Tecnología de la Información y Comunicación de la Universidad de Vigo

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE IMÁGENES

ÍNDICE DE ECUACIONES

Dedicatoria

A la China y al Pippo por la paciencia, a la Lechuza por su amor y comprensión, y por lógica al Chachi y la Lucy porque sin ellos no existiría este proyecto.

A la Tía (†) por la creencia siempre firme en las capacidades de su familia.

Juan Carlos Jiménez Encalada

PREFACIO

Este trabajo de tesis presenta una revisión de las principales metodologías de reconocimiento de rostros, a más realiza una aproximación para el reconocimiento de rostros a través de métodos por descriptores, normalmente utilizados para el reconocimiento de objetos

En el presente trabajo se ha utilizado software libre y es aprovechada la librería de código OpenCV para el análisis y tratamiento de las imágenes.

Los resultados que son presentados podrán servir a posterior para el desarrollo de nuevas técnicas híbridas de reconocimiento de rostros.

PRÓLOGO

En el presente trabajo se presentarán los resultados obtenidos en la experimentación de utilización de técnicas de reconocimiento de rostros y de patrones u objetos en stream de vídeo. Las técnicas utilizadas son: Eigenfaces(PCA, principal component analysis), Fisherfaces (Linear discriminant analysis), SIFT (Scale-invariant feature transform) y SURF (Speeded Up Robust Features). Estas técnicas permitiran identificar rostros ya sea por el acercamiento que tengan en referencia a procedimientos estadísticos de vecinos más cercanos, o incluso por la similitud que mantengan con algún o algunos vectores que serán identificados.

Para el desarrollo de la experimentación se trabajó de la siguiente manera:

- Obtención de una base de datos de varias fotografías de algunos individuos para el entrenamiento de los métodos holísticos.
- Obtención de una base de datos de una fotografía de los individuos para la experimentación de los métodos por descriptores.
- Programación de los algoritmos.
- Presentación de los resultados y de los valores porcentuales de reconocimiento.

AGRADECIMIENTOS

Al Ing. Vladimir Robles por su dirección y consejos para que el presente trabajo llegue a buen final.

A mi equipo de trabajo dentro del departamento de operaciones del ECU-911 por su incondicional apoyo y prestación para las experimentaciones.

A todos los docentes de la Maestría de Control y Automatización Industriales.

Al Ing. Eduardo Calle por su guía a través de la dirección de la Maestría.

Juan Carlos Jiménez Encalada

CAPÍTULO 1

ESTUDIO DEL ARTE EN LO REFERENTE AL RECONOCIMIENTO DE ROSTROS POR MEDIO DE TÉCNICAS DE VISIÓN ARTIFICIAL.

A la visión artificial no se la puede catalogar como una ciencia con un referente histórico que se remonta a un punto remoto en el tiempo, dado que las herramientas que han permitido su aplicación y sobre todo su desarrollo científico, no se las poseía anteriormente. Por ello, se está hablando de las computadoras y su popularización, y es solo en los últimos 30 años que esta situación ha cambiado, llevando a la práctica y al perfeccionamiento de algoritmos que permiten realizar el reconocimiento de objetos y las posteriores interpretaciones que se pueda realizar según lo que una cámara conectada a una computadora visualice.

A manera de conocimiento general, es importante mencionar que uno de los primeros experimentos de los que se tiene registro sobre visión artificial, se llevó a cabo en el año 1961 por Larry Roberts, científico norteamericano, quien en conjunto con otros investigadores desarrolla el primer programa computacional para el procesamiento de imágenes captadas por una cámara. La aplicación específica consistía en que un robot llegase a percibir y diferenciar un conjunto de bloques, moverlos y apilarlos sobre una mesa. (Vázquez Rodriguez & García Tovar, 2013).

Desde estos primeros pasos con los cuales se daba inicio a la visión por computador se ha progresado mucho y sobre todo se ha ido ramificando la visión artificial de acuerdo

a las pertinencias y especializaciones que el mundo moderno impone, entre algunas de las más importantes áreas de aplicación se puede citar a las siguientes:

- *Geología*: En esta rama científica la visión artificial aporta con reconstrucciones en tres dimensiones por medio de visión estereoscópica o por el análisis y procesamiento de imágenes, provenientes de los satélites que orbitan nuestro planeta con la finalidad de conocer los componentes de terrenos, determinar las erosiones presentadas y su posible causa entre otras aplicaciones (Pajares & De la Cruz, 2001).
- *Biología*: La visión artificial ha aportado a esta ciencia desarrollando programas para la caracterización de colores y texturas con las cuales se identifica la existencia de organismos ya sean micro o macro en diversos espacios físicos (Pajares & De la Cruz, 2001).
- *Medicina*: Rama sumamente amplia y en la cual la visión artificial también ha desempeñado un importante rol en los últimos tiempos, especialmente en lo referente a la coloración y categorización de nervios, tumores y tejidos. entre otros elementos que coadyuvan a que las imágenes médicas puedan ser valoradas de mejor manera por un profesional de esta ciencia (Pajares & De la Cruz, 2001). Entr algunas aplicaciones desarrolladas en esta área se pueden citar las siguientes: "Creación de una Herramienta que permita mover el cursor de un computador a partir del Movimiento Ocular" trabajo publicado en 2009 por Justo y Aguirre (Justo & Aguirre, 2009) , "Evaluación cuantitativa de la influencia de los espacios de color para la Detección Automática de Células" publicado en 2009 por Crespo y Ochoa (Crespo & Ochoa, 2009).
- Industria: En esta área también se han realizado aportes de interés. Algunos de los campos que se pueden mencionar son el control de calidad de los productos basándose en las imágenes para detectar si el tamaño, la textura o en general, las dimensiones de los mismos sean correctas y así evitar la producción en serie de mercancías defectuosas. De

igual forma, también existen trabajos desarrollados en ramas como la agroindustria (Pajares & De la Cruz, 2001) y algunas de las aplicaciones que se puede nombrar son: Reconocimiento Automático de Frutas ("*Automatic fruit recognition*"), presentado en 1999 por Jiménez, Jain, Ceres y Pons (Jiménez, Jain, Ceres, & Pons, 1999), técnicas de calibración de cámaras para alta precisión en visión 3D ("*A versatile camera calibration technique for High-Accuracy 3D machine visión metrology using off-the-shelf TV cameras and lenses*") trabajo publicado en 1987 por Tsai (Tsai, 1987).

- Seguridad: Con la mejora de las prestaciones que ofrecen en la actualidad las computadoras se ha llegado incluso a poseer una gran especificidad en el tratamiento de las imágenes, lo cual genera la posibilidad de aprovechar el reconocimiento de objetos a niveles de detección de determinado elemento en una escena o incluso en conjunto con la biometría realizar identificación porcentual de rostros, ambos temas muy necesarios en temas de seguridad (Pajares & De la Cruz, 2001). Entre algunas aplicaciones que se han desarrollado en ámbitos de seguridad están: Video Vigilancia Robusta para la detección de caídas basada en la deformación de la figura humana ("*Robust Video Surveillance for Fall Detection Based on Human Shape Deformation*") que fue publicado en 2011 y desarrollado por Rougier, Meunier, St-Arnaud y Rousseau (Rougier, Meunier, St-Arnaud, & Rousseau, 2011). Una aplicación más reciente es Vídeo Vigilancia Multicámara Inteligente ("*Intelligent multi-camera video surveillance*") desarrollado por Wang con publicación en 2013 (Wang, 2013)

El desarrollo de la visión artificial en el campo de la seguridad será el objeto de este trabajo, por lo cual es necesario iniciar la descripción del estado del arte en algunas de

las más importantes metodologías que se han implementado y que pueden ser aprovechadas por esta rama de la investigación científico-tecnológica.

En el siguiente gráfico se podrá observar la clasificación de los métodos de identificación y reconocimiento de objetos.

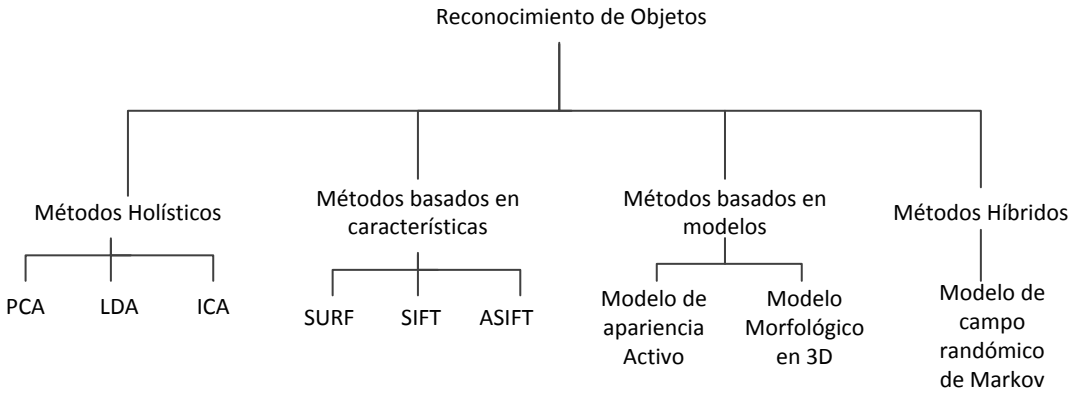

Figura 1 Clasificación de los métodos de reconocimiento de objetos (Pandya, Rathod, & Jadav, 2013)

Como se ha visto hasta el momento, existen varias técnicas mediante las cuales se puede desarrollar el propósito del reconocimiento de objetos, sin embargo, este estudio se centrará en las dos primeras metodologías, por ser las de mayor desarrollo y difusión en la rama de la discriminación de objetos aplicada al reconocimiento de rostros.

1.1. Técnicas holísticas de reconocimiento de rostros.

Dentro de la clasificación de los métodos holísticos de reconocimiento de objetos que pueden ser aplicados al reconocimiento de rostros, se encuentran los conocidos Eigenfaces y Fisherfaces, que son dos técnicas en las cuales el

algoritmo desarrollado necesita poseer un conjunto de rostros de entrenamiento, a fin de que en base a la descomposición de dichos rostros a través de técnicas estadísticas y matemáticas obtener un rostro estándar contra el cual comparar un nuevo rostro que se busque identificar.

Dicho de otro modo, la imagen capturada es representada como una matriz de 2 dimensiones mediante la cual se realiza el reconocimiento de los otros rostros, examinando la correlación existente entre el rostro patrón y los que sean ingresados para la comparación.

1.2.1 PCA y Eigenfaces.

A pesar de ser un algoritmo que trabaja con una base de rostros de entrenamiento, fue concebido inicialmente como una mejora substancial a la problemática de los años 80, década en la cual los esfuerzos científicos se centraban en el diseño de arquitecturas tecnológicas de aprendizaje y medición de características. La visión en aquel entonces del científico Matthew Turk, divergió de lo planteado hasta el momento y se propuso por su parte obtener un algoritmo en el cual se pudiera tener una estrategia total de reconocimiento facial basada en una representación del rostro humano. Asimismo, esta propuesta buscaba alejarse de la visión de las redes neuronales muy presentes por esos tiempos y más bien lograr un método que pudiese ser depurado y con una mayor comprensión matemática (Turk, 2013).

El método de Eigenfaces se basa en la posibilidad de reconstruir un determinado rostro usando una pequeña colección de rostros procesados estadística y matemáticamente. Se debe aclarar que la idea de esta aplicación de la matemática avanzada no fue directamente pensada por Turk, sino que más bien fue la aplicación de las investigaciones previamente realizadas por los científicos Sirovich y Kirby en 1986, quienes plantearon la necesidad de la representación de la fotografía a ser

estudiada en formato de grises y luego realizar las modificaciones matemáticas necesarias a fin de reducir la dimensión de tratamiento de dicha imagen a valores más tratables en términos computacionales (Sirovich & Kirby, 1987).

El método descrito por Sirovich y Kirby y aplicado por Turk obtuvo tasas de reconocimiento del 96% en situaciones en las cuales la iluminación no era similar entre el objeto a estudiar y las fotografías de entrenamiento, 85% cuando la orientación del rostro a estudiar era distinto de las fotografías de entrenamiento y 65% cuando la escala entre entrenamiento y estudio eran distintas (Jafri & Arabnia, 2009).

Para conseguir el mejoramiento dimensional posterior a los primeros estudios, Sirovich y Kirby optaron y verificaron que era mejor si se aplicaba la transformada de Karhunen-Loeve, trabajo que fue publicado en 1990 y del cual obtuvieron como conclusión que la representación de los patrones a estudiar se podían incluso mejorar y trabajar en base a la cantidad de *eigenpictures* (en aquel entonces aún no se utilizaba el nombre de eigenfaces) con las cuales se entrenaría el algoritmo, siendo sorprendente que los resultados con números impares de *eigenpictures* diferían de los resultados obtenidos con números pares (Kirby & Sirovich, 1990).

El método busca que a través de unas pocas combinaciones lineales no correlacionadas de las variables originales se pueda representar la mayor parte de información original.

Se asume que $X = \{x_1, x_2, \dots, x_n\}$ es la matriz en la cual están representados todos los elementos a estudiar siendo x_i cada uno de los vectores que representan a una imagen.

El cálculo de la media μ

$$\mu = \frac{1}{n} \sum_{i=1}^{n} x_i$$

Ecuación 1 Cálculo Media (PCA)

servirá para proceder luego con el cálculo de la matriz de covarianzas S

$$S = \frac{1}{2} \sum_{i=1}^{n} (x_i - \mu)\,(x_i - \mu)^T$$

Ecuación 2 Cálculo Covarianza (PCA)

De la matriz de covarianzas se calcularán los eigenvalores λ_i y sus respectivos eigenvectores v_i

La cantidad de eigenvectores a calcular dependerá de qué valor se determine que los eigenvalores son representativos en la muestra dada, este valor límite en cantidad es conocido como k.

Con estos cálculos es más simple pasar al nuevo espacio de trabajo que será representado como:

$$y = W^T(x - \mu)$$

Ecuación 3 Cambio de espacio (PCA)

Donde $W = (v_1, v_2, \dots, v_k)$

Los desarrollos posteriores sobre este algoritmo han ido demostrando la debilidad que el mismo mantiene respecto a la iluminación y la dirección de la luz sobre el rostro a estudiar, así como también en la escalabilidad de los objetos.

Algunos intentos por mejorar el método basado en PCA han sido desarrollados tomando como eje principal la discriminación de los rostros segmentando a los mismos en, por así llamarlo, sub-muestras, de un mismo universo para en base a esto obtener mejores resultados.

El desarrollo que se ha visto bajo esta óptica ha permitido llegar la técnica que actualmente se la conoce como MIMPCA (*Modular Image Principal Component Analysis*) que fue planteada por José Pereira, George Cavalcanti y Tsang Ren en 2009 y en la cual se propone aprovechar mejoras anteriores de la técnica como son IMPCA (*Image Principal Component Analysis*) también conocida como 2DPCA (PCA en dos dimensiones), en conjunto con otra vertiente de trabajo que proyectó la técnica MPCA (*Modular Principal Component Analysis*).

IMPCA o 2DPCA fueron planteados inicialmente por Yang en 2004 con la visión de tratar las imágenes como una matriz de dos dimensiones a diferencia del vector con el que se trabaja en PCA. Con esta particularidad se podía por tanto obviar el procedimiento en el cual la matriz era convertida a vector, lo que conlleva a un método más sencillo para la extracción de las características principales. De hecho, la matriz de covarianza puede ser construida usando directamente la matriz original de la imagen, disminuyendo de esta forma el costo computacional y al mismo tiempo incrementando la representación estadística de las imágenes de entrenamiento.

Por tanto, lo que ahora se tiene es un enfoque en el cual se trabaja con la matriz entera de la imagen para procesarla y obtener los nuevos vectores de los espacios característicos. Así la matriz de covarianza de la imagen a tratar es obtenida por:

$$Gt = \sum_{j=1}^{M} \left(A_j - \hat{A}\right)^T \left(A_j - \hat{A}\right)$$

Ecuación 4 Matriz de covarianza de una imagen (IMPCA)

Donde Gt es la matriz de covarianza total, $\hat{A}$ es la imagen promedio de todas las imágenes de entrenamiento, M denota el número total de muestras de todas las clases y A_j es la j-ésima imagen de la base de datos de entrenamiento.

Obtenida la matriz de covarianza se aplica el criterio de maximización de la dispersión para obtener el mejor vector unitario de proyección de la matriz que en este caso es el *eigenvector* correspondiente al mayor *eigenvalor* de Gt. Generalmente no es suficiente trabajar con la proyección obtenida con el valor de maximización descrito anteriormente, por lo cual es preferible trabajar con un set de proyecciones desde un $V_1, V_2 \dots V_d$ correspondiendo hasta el V_d d-ésimo mayor valor de la matriz Gt, cada Vn corresponde a un eigenvector de la matriz de covarianzas.

Con estas transformaciones la matriz final de proyección queda definida como:

$$P = [V_1, V_2, V_3, \dots, V_d]$$

Ecuación 5 Matriz de proyección de la imagen a estudiar (IMPCA)

Donde cada columna corresponde a un eigenvector de la matriz de covarianza.

Con lo desarrollado se obtiene un nuevo espacio de característico y la imagen de test puede ser proyectada bajo la siguiente expresión:

$$Y_t = P.I_t \qquad \forall t = 1 \dots T$$

Ecuación 6 Proyección de la imagen de test (IMPCA)

Siendo Y_t la representación de la T-ésima imagen de test mientras que T es el número total de imágenes de test de la base de datos aplicada.
Es notorio entonces que la técnica IMPCA implica un fácil cálculo de la matriz de covarianza debido a la baja dimensionalidad para los cálculos y al mismo tiempo involucra un mejoramiento en el tiempo para la determinación de los eigenvectores. (Pereira, Cavalcanti, & Ren, 2009) (Shah, Sharif, Raza, & Azeem, 2011).

La técnica del MPCA fue concebida con la finalidad de mejorar otro tema en el que PCA demostraba carencias, siendo este la falta de robustez para el reconocimiento ante variaciones propias de la imagen como orientación del objeto a reconocer y, en el caso de rostros, la expresión del semblante o incluso la misma iluminación dado que la información es manejada en su totalidad como un solo conjunto de datos. Como el mismo nombre sugiere, la técnica trabaja por módulos o en subregiones del rostro a reconocer extrayendo, las características de cada subregión, a fin de ser posteriormente analizadas y comparadas con las subregiones del rostro a identificar. Este enfoque fue desarrollado por Gottumukkal y Asari y publicado en 2004. La principal característica de mejora en este método está centrada en que al dividir la matriz representativa del rostro a buscar en varias subregiones o submatrices y los pesos de los vectores característicos de cada una de ellas representarán de mejor manera las particularidades del rostro analizado. Esta situación se fundamenta en que mientras existan variaciones debido a la pose o iluminación entre otros, solo ciertos segmentos del patrón serán afectados quedando inalterados algunas otras subregiones.
Hay que recordar que el método clásico del PCA realiza la representación de la matriz del rostro como un vector de dimensión L^2 tomando en cuenta

que la matriz original debería ser de tamaño L por L. Esta situación se verá alterada en la propuesta de MPCA puesto que cada imagen de la base de entrenamiento será dividida en N subregiones para su análisis individual, por tanto el tamaño de cada sub-imagen será L^2/N. Así las sub-imágenes podrán ser representadas como:

$$I_{ij}(m,n) = I_i\left(\frac{L}{\sqrt{N}}(j-1)+m\,,\frac{L}{\sqrt{N}}(j-1)+n\right)\ \ \forall i,j$$

Ecuación 7 Representación matemática de sub-imágenes (MPCA)

Donde i va de 1 a M, al ser M el número de imágenes de entrenamiento, j va de 1 a N y N es el número de sub-imágenes con el cual se trabajará y finalmente m y n irán de 1 a $L/\sqrt{N}$

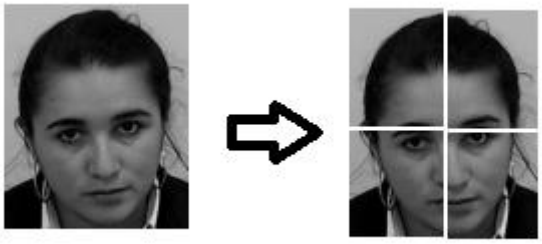

Imagen 1 División en sub-imágenes con N=4 (Autor)

Con esta división de la imagen la forma de expresar la imagen promedio matemáticamente es:

$$A = \frac{1}{M\,N}\sum_{i=1}^{M}\sum_{j=1}^{N} I_{ij}$$

Ecuación 8 Representación de Imagen Promedio en método (MPCA).

Una vez realizados estos pasos se debe normalizar cada sub-imagen de entrenamiento:

$$Y_{ij} = I_{ij} - A \qquad \forall i, j$$

Ecuación 9 Normalización de sub-imágenes (MPCA)

Posteriormente se obtiene la matriz de covarianza

$$C = \frac{1}{M\,N} \sum_{i=1}^{M} \sum_{j=1}^{N} Y_{ij}\, Y_{ij}^{T}$$

Ecuación 10 Matriz de Covarianza (MPCA).

Para encontrar los eigenvectores de esta matriz de covarianza se debe buscar primero los eigenvalores de mayor representación en una cantidad M'. La representación de estos valores se da por $E_1, E_2, \dots, E_M$. Los pesos serán obtenidos por tanto de la siguiente expresión:

$$W_{pnjK} = E_K^T \left(I_{pnj} - A\right) \qquad \forall\, p, n, j, K$$

Ecuación 11 Cálculo de los pesos de los principales Eigenvalores (MPCA).

Donde K toma valores de 1 a M', n varía de 1 a G, siendo G el número de imágenes de entrenamiento por cada sujeto y finalmente p varía de 1 a P que representa la cantidad de sujetos con los cuales se realizará el entrenamiento del método.

Los eigenvalores de cada sub-imagen matemáticamente son obtenidos de:

$$W_{test\ jK} = E_K^T \left(I_{test} - A\right) \qquad \forall\ j, K$$

Ecuación 12 Cálculo de los eigenvectores de las sub-imágenes (MPCA)

La siguiente expresión simplemente permite obtener los eigenvectores de cada sub-imagen mediante la siguiente expresión:

$$T_{pjK} = \frac{1}{\Gamma}\sum_{K=1}^{M'}\sum_{n=1}^{\Gamma} W_{pnjK} \qquad \forall\ p, j$$

Ecuación 13 Cálculo eigenvectores (MPCA)

Finalmente, se usa la distancia euclidea para conocer el grado de similitud entre la imagen a estudiar y la imagen promedio:

$$D_{pj} = \frac{1}{M'}\sum_{K=1}^{M'} \left| W_{test\ jK} - T_{pjK} \right|$$

$$D_p = \frac{1}{\mathrm{N}}\sum_{j=1}^{N} D_{pj}$$

Ecuación 14 Calculo de distancias entre imagen promedio e imagen a estudiar (MPCA)

Por tanto, la imagen de test es reconocida y se establece su mayor similitud con respecto a la *p*-ésima imagen de entrenamiento (Gottumukkal & Asari, 2004).

Este método posee la desventaja que de realizar demasiadas sub-imágenes se corre el riesgo de perder información global del objeto estudiado y la precisión del mismo decaería. Otra peculiaridad que fue puesta en evidencia luego de algunos experimentos, es que incrementa el costo computacional. La parte alentadora es que se consigue mejorar el reconocimiento cuando las imágenes poseen diferencias de iluminación y expresión como se había puntualizado anteriormente (Pereira, Cavalcanti, & Ren, 2009).

MIMPCA es una técnica que fue desarrollada, como se había mencionado, en el año 2009, planteándose la posibilidad de unir dos métodos (IMPCA, MPCA) para alcanzar un proceso de identificación de rostros más robusto aprovechando las cualidades de ambas técnicas.

Por tanto y como se tiene ya detallado el proceso y las principales características de los métodos base, se procederá a revisar las variantes que se ha realizado en la técnica MIMPCA:

Se debe tener en primera instancia, y como para todos los métodos detallados hasta el momento; una serie de imágenes que servirán como imágenes de entrenamiento, a estas se les designará como $I1, I2, \dots I_M$, siendo cada una de estas matrices la representación de una imagen de dimensión KxL. La representación de estas imágenes pertenece a Q diferentes clases.

En este método, al igual que en MPCA, las imágenes son divididas en A particiones horizontales y B particiones verticales, en términos matemáticos se tendría que la cantidad de sub-regiones a tratar está representada por $N = K * L$ y dichas subregiones tendrían dimensiones de $(K * L)/N$ pixeles.

La representación de cada subregión quedaría expresada como:

$$I_{mij}(x, y) = I_m\left(\frac{K}{A}(j-1) + x, \qquad \frac{L}{B}(i-1) + y\right) \qquad \forall i, j$$

Ecuación 15 Representación de sub-regiones en método MIMPCA

Teniendo la variación de i desde 1 hasta A y j desde 1 hasta B, a más que I_{mij} representan a la sub-imagen localizada en las coordenadas i, j de la m-ésima imagen de entrenamiento.

El cálculo de la imagen promedio por tanto se lo realiza como se indica a continuación:

$$\overline{A} = \frac{1}{Q}\sum_{q=1}^{Q}\overline{A_q}$$

Ecuación 16 Imagen promedio en método MIMPCA

Donde $\overline{A_q}$ corresponde a la imagen promedio de la Q-ésima clase que estemos tratando y es calculada como sigue:

$$\overline{A_q} = \frac{1}{(A * B * |C_q|} \sum_{m=1}^{M}\sum_{i=1}^{A}\sum_{j=1}^{B} I_{mij} \qquad I_m \in C_q$$

Ecuación 17 Imagen promedio de cada clase en MIMPCA

Un paso adicional es la normalización de las sub-imágenes restando a estas la imagen promedio global:

$$Z_{mij} = I_{mij} - \overline{A}$$

Ecuación 18 Normalización de las sub-imágenes en el método MIMPCA

Obviamente Z_{mij} representa la región normalizada del vector de coordenadas ij de la m-esima imagen del set de entrenamiento.
La matriz de covarianza requerida se obtiene de la siguiente forma:

$$S = \frac{1}{Q}\sum_{q=1}^{Q} S_q$$

Ecuación 19 Matriz de covarianza Método MIMPCA

Como es natural pensar, S_q es la matriz de covarianza correspondiente a la q-ésima clase de las imágenes de entrenamiento. La misma es calculada como:

$$S_q = \frac{1}{(A * B * |C_q|} \sum_{m=1}^{M} \sum_{i=1}^{A} \sum_{j=1}^{B} (Z_{mij}\, Z_{mij}^T) \qquad I_m \in C_q$$

Ecuación 20 Matriz de covarianza de cada clase Método MIMPCA

Como en todos los métodos que se basan en PCA, el cálculo de los eigenvectores asociados a los eigenvalores de mayor representatividad es necesario y debe ser expresado como $E_1, E_2, \dots\dots, E_v$. El peso de cada imagen es calculado por la multiplicación de las imágenes normalizadas (Z_{mij}) por los eigenvectores de mayor representatividad, generando de esta manera los componentes principales:

$$W_{mij} = \left(I_{mij} - \overline{A}\right) P \qquad \forall\, m, i, j, v$$

Ecuación 21 Componentes principales Método MIMPCA

Como siempre, no es recomendable poseer solamente un eigenvector con el cual realizar el proceso de reconocimiento, por lo cual se sugiere obtener algunos V eigenvectores, lo que nos lleva a poseer N matrices conteniendo LxV coeficientes. Siendo esta una desventaja del método planteado dado que el espacio de almacenamiento para estas matrices es bastante superior al requerido en el método tradicional de PCA.

El cálculo final de la distancia entre la imagen de test y la imagen promedio debe ser realizada mediante la distancia Euclídea como:

$$d(I_m, I_t) = \sum_{i=1}^{A} \sum_{j=1}^{B} (W_{mij} - W_{tij})$$

Ecuación 22 Distancia entre imagen de test y promedio Método MIMPCA

Es importante mencionar que el método actualmente está en desarrollo, dada la principal falencia que el mismo tiene en el sentido de almacenaje de los vectores de representación de los componentes principales que dejan de ser un mero escalar. La propuesta de este método toma ventaja de las regiones de un rostro que no se ven afectadas por variaciones de iluminación, expresión facial u orientación del rostro (MPCA) y al mismo tiempo realiza el enfoque bidimensional para la extracción de mejores puntos representativos de los datos originales (IMPCA) reduciendo de esta forma los costes computacionales (Pereira, Cavalcanti, & Ren, 2009).

1.2.2 LDA y Fisherfaces.

El método de Fisherfaces es también muy conocido y utilizado desde hace un poco más de una década, siendo presentado por Belhumeur en 1997 (Delac, Grgic, & Bartlett, 2008). Esta técnica trabaja con Análisis Discriminante Lineal (LDA) para extraer las características de discriminación de mayor relevancia y así reducir la dimensionalidad de las matrices de representación de una imagen. En otras palabras, LDA busca por los vectores subyacentes en el espacio de trabajo que mejor discriminan entre los datos presentados. Más formalmente, dado un número de características independientes relativas a un set de datos, LDA

crea una combinación lineal que produce una gran diferenciación entre las clases estudiadas. En expresión matemática, para todos los muestreos de todas las clases, se definen dos medidas:

1) Matriz de dispersión dentro de las clases.

$$S_w = \sum_{j=1}^{c} \sum_{i=1}^{N_j} (x_i^j - \mu_j)(x_i^j - \mu_j)^T$$

Ecuación 23 Dispersión dentro de clases (LDI)

Donde x_i^j es la i-ésima muestra de una clase j, μ_j es la media de la clase j, c es el número de clases, y N_j es el número de imágenes de la clase j.

2) Matriz de dispersión entre las clases.

$$S_b = \sum_{j=1}^{c} (\mu_j - \mu)(\mu_j - \mu)^T$$

Ecuación 24 Dispersión entre clases (LDI)

Donde μ representa la media de todas las clases.

El propósito es maximizar la distancia entre las clases y minimizar la distancia dentro de las clases.

Como se puede observar, esta técnica trata la dispersión dentro y entre clases como los valores relevantes para el reconocimiento de rostros. La debilidad presentada es que se requiere una gran cantidad de imágenes de

muestra para una buena generalización de la imagen media (Yongzhong, Jingli, & Shengsheng, 2003).

De esta técnica también han existido varios intentos para mejorarla y poder obtener un sistema más robusto con el cual lograr identificar de mejor manera los objetos, entre los ensayos de mejora de mayor representatividad están: *Direct LDA, Direct-Weighted LDA, Nullspace LDA, Dual-space LDA, Pair-wise LDA, Regularized Discriminant Analysis, Generalized Singular Value Decomposition, Direct Fractional-Step LDA, Boosting LDA, Discriminant Local Feature Analysis*, entre otras. De ellas, la de más reciente desarrollo y en la cual todos los otros intentos han confluido para su mejor desempeño es conocida como GSVD-ILDA (*Generalized Singular Value Decomposition, Incremental LDA*) (Jafri & Arabnia, 2009) (Zhao & Yuen, February 2008).

El principal problema de los métodos que se basan en Análisis Discriminante Lineal, es la escalabilidad de las imágenes con las que se debe desarrollar su funcionalidad. Por ello, para superar esta limitación un enfoque incremental es la solución obvia, esto fue visto y estudiado por numerosos científicos que han dedicado varias alternativas y estudios, dado que se presenta bastante complicado realizar la inversa de la matriz de dispersión dentro de las clases.

A pesar de las dificultades matemáticas evidenciadas se han desarrollado ciertos procesos matemáticos para salvar estas particularidades.

El primer paso (según lo planteado por Kim, Stenger, Keittler y Cipolla en 2011), es realizar una maximización de separación de las clases de un set de datos dentro de una matriz U, esto se lo consigue mediante:

$$\arg max_U \frac{|U^T S_B U|}{|U^T S_W U|} = \arg max_U \frac{|U^T S_T U|}{|U^T S_W U|} = \arg max_U \frac{|U^T S_B U|}{|U^T S_T U|}$$

Ecuación 25 Maximización matriz U de datos (ILDA).

Donde S_B es la Matriz de dispersión entre clases y está representada por:

$$S_B = \sum_{i=1}^{C} n_i (m_i - \mu)(m_i - \mu)^T$$

Ecuación 26 Matriz de dispersión entre clases (ILDA).

S_W es la Matriz de dispersión al interior de una clase y está representada por:

$$S_W = \sum_{i=1}^{C} \sum_{x \in C_i} (x - m_i)(x - m_i)^T$$

Ecuación 27 Matriz de dispersión dentro de una clase (ILDA).

Y S_T es la matriz de dispersión total representada por:

$$S_T = \sum_{\forall x} (x - \mu)(x - \mu)^T = S_B + S_W$$

Ecuación 28 Matriz de dispersión total (ILDA)

C es el número total de clases, n_i el número de muestras de la clase i, m_i es la media de la clase i, y μ la media global de los datos.

La matriz S_T es utilizada como la matriz incremental en lugar de S_W para mantener una mejor discriminación de datos durante la actualización en el caso de que nuevas clases se adicionen al set de datos. Por ejemplo, si el trabajo se lo realizase exclusivamente extrayendo los componentes principales de las matrices S_B y S_W, cualquier información discriminatoria

contenida en el espacio nulo de S_W se perderá (hay que notar que cualquier componente en el espacio nulo maximiza el criterio de LDA). Esto último se explica dado que teóricamente los componentes desechados por ser menos significativos dentro los principales eigenvalores, podrían reaparecer en las actualizaciones de la base de datos dentro de la matriz S_T, aportando con una desconocida oportunidad hasta el momento y que es una importante información dentro del criterio LDA.
Los tres aspectos relevantes en las propuestas de un LDA incremental llegan a ser por tanto:

- Dados dos sets de datos, cada uno representado en un modelo de eigenespacio, los componentes principales de la matriz de dispersión S_T (resultante de la unión de los set de datos) son utilizados como la fusión de los modelos de los eigenespacios.
- Similar al proceso normal de componentes principales, la matriz de dispersión entre clases S_B es actualizada fusionando los respectivos modelos de eigenespacios.
- El paso final es trabajar con la discriminación de los componentes U usando la actualización de los componentes principales de los dos pasos ya descritos.

El propósito de estos tres pasos es general y puede ser aplicado a cualquier problema de aprendizaje incremental que persiga el encontrar componentes discriminativos por la maximización del radio que envuelve a dos diferentes matrices de covarianzas o correlaciones (Kim, Stenger, Kittler, & Cipolla, 2011).

La técnica GSVD-ILDA mencionada anteriormente como una de las de más reciente desarrollo y planteada en 2008 por Zhao y Yuen, fundamenta su funcionamiento en un proceso incremental que aporta dos principales ventajas: 1) La técnica puede ser usada para un procesamiento en partes o

en secuencia, lo cual es siempre conveniente dependiendo del tamaño de la base de datos y 2) Con la adición dinámica de las muestras, la técnica puede limitar el coste computacional.

En base el desarrollo de una Descomposición de Valor Singular Generalizada (GSVD) no es sino una alternativa al conocido PCA siendo exclusivamente un modelo de reducción del espacio dimensional con el que se deberá tratar la discriminación de los elementos a buscar entre dos set de datos, uno de entrenamiento y uno de identificación.

Fundamentalmente GSVD realiza la descomposición de una matriz dada A de IxJ dimensiones utilizando dos matrices cuadradas definidas positivas de IxI y JxJ dimensiones respectivamente. Estas dos matrices expresan las restricciones impuestas correspondientes a las filas y columnas de la matriz A.

Formalmente, si la matriz M es de IxI dimensiones y expresa las restricciones para las columnas de A y la matriz W es de es de JxJ dimensiones y al tiempo expresa las restricciones de las columnas de A, se puede descomponer a la matriz A como:

$$A = \tilde{A}\tilde{\Delta}\tilde{V}^T \text{ con: } \tilde{U}^T M\tilde{U} = \tilde{V}^T W\tilde{V} = I.$$

En otras palabras, el vector singular generalizado es ortogonal bajo las restricciones impuestas por M y W.

Estos resultados son obtenidos de la descomposición de valor singular estándar. Se debe iniciar definiendo la matriz $\tilde{A}$ como:

$$\tilde{A} = M^{\frac{1}{2}} A W^{\frac{1}{2}} \iff A = M^{-\frac{1}{2}} \tilde{A} W^{-\frac{1}{2}}$$

Entonces, se debe realizar el cálculo de la descomposición de valor singular estándar como $\tilde{A}$ de la siguiente manera:

$$\tilde{A} = P\Delta Q^T \ \ con \ \ P^T P = \ Q^T Q = I$$

Donde P es la matriz (normalizada) que contiene a los eigenvectores de la matriz AA^T. Las columnas de P son llamadas los vectores singulares izquierdos de A.

Q es la matriz (normalizada) que contiene a los eigenvectores de la matriz $A^T A$. Las columnas de Q son llamadas los vectores singulares derechos de A.

Δ es la matriz diagonal que contendrá a los valores singulares, $\Delta = \ \Lambda^{\frac{1}{2}}$ al ser Λ una matriz diagonal que contiene a los eigenvalores de la matriz AA^T o de la matriz $A^T A$ dado que son iguales.

Las matrices de los eigenvectores generalizados son obtenidas como:

$$\widetilde{U} = M^{-\frac{1}{2}} P \ \ y \ \ \tilde{V} = \ W^{-\frac{1}{2}} Q$$

La matriz diagonal de valores singulares es simplemente igual a la matriz de valores singulares de $\tilde{A}$, en otras palabras $\tilde{\Delta} = \ \Delta$, por lo cual se verifica que:

$$\tilde{A}\tilde{\Delta}\tilde{V}^T$$

y por sustitución:

$$A = \ M^{-\frac{1}{2}} \tilde{A} W^{-\frac{1}{2}} = \ M^{-\frac{1}{2}} P \Delta Q^T W^{-\frac{1}{2}} = \ \widetilde{U}\tilde{\Delta}\tilde{V}^T$$

Por tanto este método en conjunto se presenta como una alternativa en la reducción de la dimensionalidad mediante una metodología distinta al

clásico PCA y por otra parte, la discriminación de los elementos significativos lo realiza de forma incremental, lo que aporta mayores detalles para el reconocimiento de los patrones (Abdi, 2007).

A continuación se presenta una muestra de cómo se realiza la reducción de la dimensionalidad de una matriz representativa de una fotografía por el método de GSVD.

Imagen 2 Fotografía original (GSVD-ILDA) (Abdi, 2007)

Imagen 3 Fotografía luego del proceso de GSVD (GSVD-ILDA) (Abdi, 2007)

1.2. Técnicas por descriptores aplicadas al reconocimiento de rostros.

Otra visión de métodos para el reconocimiento de objetos está influenciada por el enfoque del manejo exclusivo de características predominantes dentro de un conjunto de datos. En los métodos basados en características, características

locales como la nariz y los ojos son segmentados y entonces se da al sistema de detección una tarea más simple de reconocimiento de estos sectores. (Pandya, Rathod, & Jadav, 2013)

1.2.1 SURF.

La tarea de encontrar correspondencia entre imágenes de objetos y/o escenas similares es esencial en las aplicaciones de visión por computador. Esto puede ser alcanzado utilizando tres pasos conocidos como: detección, descripción y emparejamiento de características (Schiele & Crowley, 2000). En la instancia de detección, los puntos de interés son seleccionados desde distintas ubicaciones dentro de la imagen como pueden ser esquinas, bordes, manchas, entre otros. Estos puntos deberían ser distintivos y repetibles, es decir han de ser detectables bajo diferentes e inclusive severas condiciones de vista como inclinación, variación de iluminación u otras peculiaridades dentro del campo visual. El paso de descripción, el vecindario de cada punto de interés o los vecinos más cercanos de cada punto de interés son representados por un vector de características. Finalmente, en la etapa de emparejamiento de características, los vectores característicos de diferentes imágenes son pareados, esto último comúnmente se lo realiza con la medición de las distancias entre los vectores, como ejemplo y uno de los métodos más utilizados es la distancia Euclidea.

Speeded Up Robust Features (SURF) es un detector robusto de características locales que fue estudiado y presentado por Herbert Bay en 2006, la cual introduce el concepto de detección de puntos de interés locales e invariantes. SURF por tanto es invariante a las transformaciones comunes de las imágenes como lo es la rotación, cambio en las escalas, cambios de iluminación y pequeños cambios en la perspectiva de visión. (Herbert, Tuytelaars, & Van Gool, 2006)

Dentro de esta técnica también es utilizado el concepto de imagen integral (suma de áreas), las cuales son una representación intermedia de la imagen y

contienen la suma de valores de los pixeles de una imagen en una escala de grises, de esta manera se reduce también el tiempo de cómputo. La detección está basada en la matriz Hessiana que garantiza un buen uso y rendimiento en costo computacional y precisión. Esto último es realizado gracias a la popularización del trabajo de imagen integral desarrollado por Viola y Jones (Viola & Jones, 2001), el cual reduce los tiempos de cómputo drásticamente.

Esta técnica está basada en el uso de los filtros de dos dimensiones de Haar o también conocidos como wavelet. Se usa una aproximación entera para determinar el Hessiano de una matriz, con este desarrollo se logra obtener velocidades de cómputo bastante rápidas con el uso de imágenes integrales.

Para la extracción de las características se utiliza la suma de las respuestas de los wavelets alrededor del punto de interés (Sonker & Behera, 2012).

Dado un punto $x = (x, y)$ en una imagen I, la matriz Hessiana del punto x a una escala σ es definido como $H(x, \sigma)$:

$$H(x,\sigma) = \begin{bmatrix} I_{xx}(x,\sigma) & I_{xy}(x,\sigma) \\ I_{xy}(x,\sigma) & I_{yy}(x,\sigma) \end{bmatrix}$$

Ecuación 29 Matriz Hessiana (SURF)

Donde $I_{xx}(x, \sigma)$, $I_{xy}(x, \sigma)$ y $I_{yy}(x, \sigma)$ representan la convolución entre la derivada Gausiana de segundo orden $\frac{\partial^2}{\partial x^2} g(\sigma)$ con la imagen I en el punto x.

El descriptor hace uso de las respuestas al filtro de Haar entre vecindario del punto de interés, por tanto el descriptor del punto de interés trabaja en primera instancia identificando una orientación reproducible basándose en la información obtenida en la región circular alrededor del punto de interés. Luego, este construye una región cuadrada alineada a la orientación seleccionada y extrae los datos del descriptor.

La asignación de la orientación: En primera instancia las respuestas del filtro de Haar-wavelet en las direcciones de x y y son calculadas en un vecindario

circular de radio 6s con centro en el punto de interés, s es la escala en la que el punto de interés fue detectado. Las respuestas del filtro de Haar son presentadas como vectores. Posteriormente, todas las respuestas dentro de una ventana de orientación deslizante que cubre un ángulo de 60 grados son sumadas. Las dos respuestas horizontales y verticales dentro de la ventana son sumadas dando como resultado un nuevo vector. Dentro de estos, el vector de mayor magnitud será el vector dominante.

Descripción: El paso de la descripción incluye la construcción de una región cuyo centro es el punto de interés y con la orientación prevaleciente del punto anterior. La región de interés en este caso es dividida en subregiones de 4x4 cuadrados en los que contiene 5x5 puntos de muestra regularmente espaciadas al interior de cada cuadrado.

La respuesta al filtro de Haar para dx y dy son calculadas donde dx, dy son las direcciones horizontal y vertical, respectivamente. Estas respuestas se ponderan con un kernel gaussiano centrado en el punto de interés para aumentar la robustez a las deformaciones y errores de localización. Las respuestas a dx y dy sobre cada subregión se suman formando por separado un primer conjunto de entradas para el vector de características. Para obtener información sobre la polaridad de los cambios de intensidad, la suma los valores absolutos de las respuestas | dx | y | dy | es extraída.

La intensidad de la estructura para cada subregión es descrita por:

$$V = \left(\sum d_x , \sum d_y , \sum |d_x| , \sum |d_y|\right)$$

Ecuación 30 Intensidad de subregiones (SURF)

Finalmente, el vector es normalizado en una unidad de longitud para lograr invariancia de contraste. (Kandil, H; Atwan, A;, 2012)

Para proseguir y con la finalidad de realizar que el presente documento sea entendible por sí mismo, se realizará la explicación de lo que es una Imagen Integral.

El concepto de imagen integral se centra en permitir un rápido procesamiento de los filtros de convolución. La entrada de una imagen integral $I_{\Sigma}(x)$ en un punto en particular $x = (x, y)^T$ es representado por la suma de todos los pixeles de la imagen de entrada I dentro de una región rectangular formada por el origen y x.

Una vez que la imagen integral ha sido procesada, se deberá realizar tres adiciones para calcular la suma de las intensidades que están actuando sobre cualquier área rectangular. Con este tratamiento se consigue que el tiempo de cálculo sea independiente del tamaño de la muestra. Esto último es sumamente importante cuando son usados filtros de tamaños grandes.

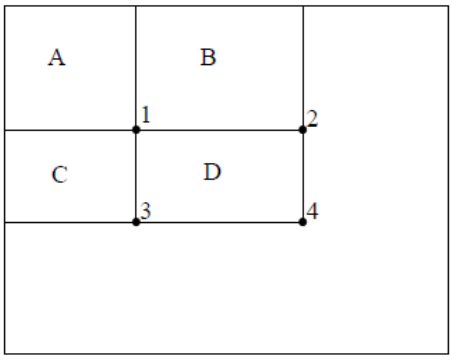

Figura 2 Representación de una imagen integral. (SURF) (Viola & Jones, 2001)

En la figura 2 se muestra un ejemplo de un segmento de una imagen cualquiera. La suma de los pixeles dentro del rectángulo D puede ser calculada con 4 arreglos de referencia. El valor de la imagen integral en la posición 1 es la suma de los pixeles en el rectángulo A. El valor en la posición

2 es A+B, en la ubicación 3 es A+C y en la posición 4 es A+B+C+D. La suma dentro de D puede ser calculada como 4+1-(2+3) (Viola & Jones, 2001).
Evidentemente al usar la imagen integral, solamente toma tres sumas y cuatro memorias el acceder al cálculo de la suma de las intensidades dentro de una región rectangular de cualquier tamaño.
Para la mejora del desempeño del método SURF han sido propuestas algunas alternativas, pero el progreso introducido no ha llegado a ser del todo alentador siendo el esfuerzo computacional involucrado demasiado afanoso como para desechar el uso mucho más dinámico y simple del SURF original (Pang, Li, Yuan, & Pan, 2012).

Por propósitos investigativos es menester presentar una breve descripción de los intentos por mejorar el desempeño del método SURF.
MU-SURF. Modificación Vertical de SURF (Modified Upright SURF), de acuerdo a sus autores (Agrawal, Konolige, Blas) es una técnica en la cual se pretende superar los métodos de detectores en el reconocimiento de patrones, a más de tener mejores características computacionales y al mismo tiempo poder ser implementada en tiempo real. Este trabajo fue publicado en 2008.
El enfoque con el que se trabaja en este método se refiere a la consecución de características de gran escala para lo cual se analizan todas las características de todas las escalas y se selecciona los extremos tanto en las escalas como en las posiciones de la matriz a estudiar. Evidentemente esto provoca un procesamiento excesivo lo cual desemboca en la necesidad de computadores de mayores prestaciones. La principal preocupación será encontrar kernels que sean invariantes a la rotación pero de fácil procesamiento computacional.
El meollo para poder realizar este método de manera eficiente dependerá, al igual que en los métodos anteriores, de la capacidad que se tenga para disminuir la dimensionalidad del problema, por lo cual es conveniente tratar nuevamente con imágenes integrales. Es menester dentro de las imágenes

(matrices) que estén siendo estudiadas realizar una minimización de los efectos que podrían tener los bordes o contornos en los cuales los descriptores abruptamente cambian de orientación o de un histograma a otro. A estos efectos en el método de SURF tradicional se los intenta anular con la ponderación de los filtros de Haar, usando los conceptos de Gauss. Este simple esquema de ponderación otorga resultados no muy alentadores (según los autores del método MU-SURF) dado que no es posible concebir descriptores resultantes sin los efectos de los contornos de las imágenes tratadas. Por su parte en el método MU-SURF se enfrenta esta problemática agrandando la escala de s a 2s y el área de trabajo de 20s a 24s donde s es la escala de la característica de mayor relevancia. En otras palabras las respuestas a los filtros de Haar tanto en la dirección horizontal (dx) como vertical (dy) son procesadas por cada 24x24 puntos en la región de escala 2s, se crea por tanto una imagen promedio en la cual cada pixel es la suma de una región de tamaño 2s. La salida de este proceso entregará como resultado 4 imágenes dx, dy, $|dx|$, $|dy|$ de tamaño fijo en 24 pixeles independientemente de la escala dx, dy, $|dx|$, $|dy|$.

Cada una de estas 4 imágenes son luego divididas en 4x4 subregiones cuadradas sobrepuestas de 9x9 pixeles con una superposición de 2 pixeles con cada una de las regiones vecinas. Para cada subregión los valores son ponderados y pre procesados por un filtro Gaussiano ($\sigma_1 = 2.5$) con centro en el punto central de la subregión y resumido en un vector con el usual procedimiento de descriptores de SURF. Cada vector de subregión es nuevamente ponderado usando un nuevo filtro Gaussiano ($\sigma_1 = 1.5$) definiendo una máscara de 4x4 y centrada en el punto de la característica. Finalmente este nuevo vector es normalizado.

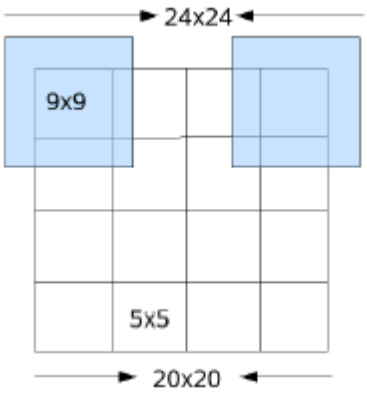

Figura 3 Regiones y subregiones (MU-SURF) (Agrawal, Konolige, & Blas, 2008)

La superposición consiente a cada subregión trabajar en áreas de mayor tamaño que las muestreadas normalmente mejorando la probabilidad de que el vector de la subregión no conlleve un efecto de contorno. De igual manera los filtros Gaussianos que sean aplicados a las señales cercanas a los contornos tendrán un menor impacto en el vector descriptor de la subregión (Agrawal, Konolige, & Blas, 2008).

Otra alternativa de mejora al método SURF es la técnica de SUSurE (Speeded Up Surround Extrema Features) propuesta por Ebrahimi y Mayol-Cuevas en 2009. Básicamente este nuevo enfoque plantea una reducción en los cálculos de la extracción de los descriptores realizando que si determinado valor de $|dx_{\{i,j\}}|$ de una región de escala S_i es menor a un umbral δ_2, el componente $|dx_{\{i+1,j\}}|$ de la muestra adyacente derecha no será procesado. Así entonces el valor de $|dx_{\{i,j\}}|$ es usado en lugar de $|dx_{\{i+1,j\}}|$ en el cálculo del vector de la subregión. De igual manera si $|dy_{\{i,j\}}|$ de la región de escala S_i es menor al umbral δ_2, el componente $|dy_{\{i+1,j\}}|$ de la muestra adyacente inferior no será procesado utilizando en su lugar $|dy_{\{i,j\}}|$ en el cálculo del vector de la

subregión. El valor recomendado por los autores del método es de $\delta_2 = 20$. (Ebrahimi & Mayol-Cuevas, 2009)

Como se puede observar el cambio reside únicamente en una mejora en la velocidad de los cálculos al dejar de lado ciertos valores que no puedan aportar mayores detalles en la búsqueda de los descriptores.

Finalmente se tiene el método Fully Affine Invariant SURF (FAIR SURF) planteado en 2012 por Yanwei Pang. Este método proyecta la alternativa de realizar un pre-procesamiento de las matrices a trabajar en el cual se simule las posibles alteraciones que la imagen contenida en dicha matriz sufriría por cambios en la orientación del eje óptico de la cámara desde una posición frontal al objeto a representar en la matriz.

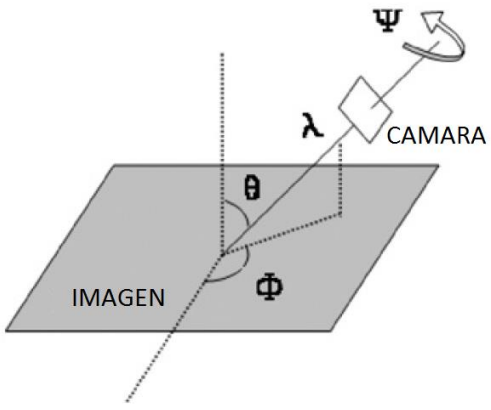

Figura 4 Interpretación geométrica (FAIR-SURF) (Pang, Li, Yuan, & Pan, 2012)

En la figura 4 se puede apreciar los distintos ángulos que se forman el momento de tomar una fotografía, con esto se pretende expresar lo complicado que puede llegar a ser la simulación de las distorsiones que se provocarían por cambios en estos ángulos. ϕ y θ son ángulos de longitud y

latitud referentes al eje óptico de la cámara, ψ es el ángulo de spin de la cámara y λ representa el parámetro de zoom. Así entonces las rotaciones e inclinaciones están representadas por un número finito de ángulos de longitud y latitud. Luego las imágenes son sometidas a rotaciones e inclinaciones, los ángulos con los cuales se realizará este proceso son seleccionados teniendo en cuenta que deben representar todos los posibles enfoques que a posterior la figura podría tener dentro de un campo visual, sin descuidar la necesidad de una extracción eficiente de las características.

Las representaciones de las distorsiones provocadas son procesadas buscando sobreponer las máscaras de Haar dado que es un hecho que dos máscaras adyacentes tiene una superposición de la mitad de longitud que una máscara normal. Esta propiedad de este método coadyuva a la velocidad de procesamiento (Pang, Li, Yuan, & Pan, 2012).

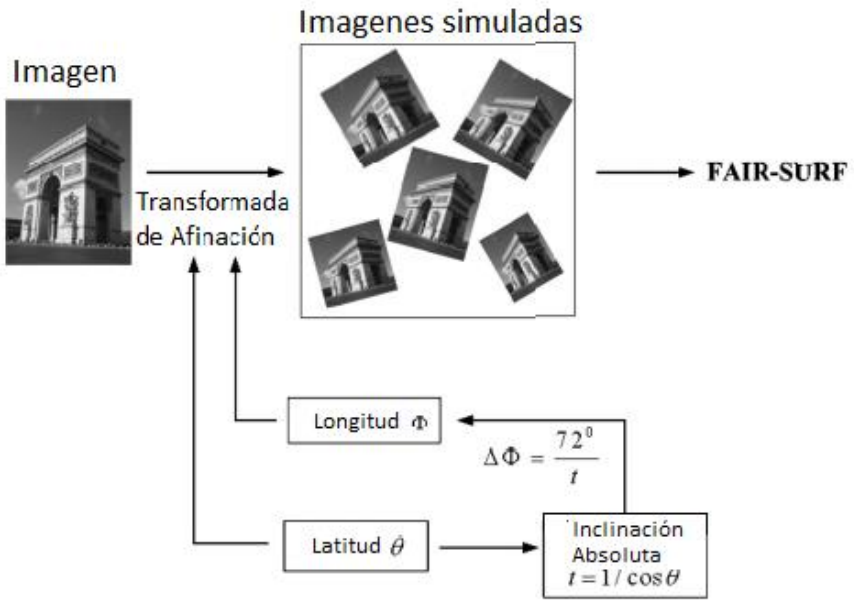

*Figura 5 Simulación de las distorsiones (FAIR-SURF***)** (Pang, Li, Yuan, & Pan, 2012)

En la figura 5 se puede apreciar el proceso en el cual se realizan las simulaciones sobre la figura inicial y luego las representaciones resultantes pasan a ser procesadas por el método SURF tradicional.

1.2.2 SIFT.

Scale Invariant Feature Transform es un descriptor de imágenes basado en el desarrollo realizado por David Lowe entre 1999 y 2004 (Lowe, Object recognition from local scale-invariant features, 1999). Este procedimiento tiene la ventaja de ser invariante a las posibles distorsiones debidas a traslación, rotación y escala de la imagen a ser estudiada y posee cierto grado de robustez ante transformaciones debidas a las variaciones de iluminación o del ángulo de perspectiva con el cual se capte al objeto de estudio.

En base lo que este método plantea es la detección de puntos de interés dentro de una matriz de representación de un objeto en la cual y a través de la sumatoria de las intensidades de los distintos gradientes se obtengan al aplicar el proceso de diferencias Gaussianas a una pirámide también conocida como pirámide Gaussiana, proceso que fue propuesto por Burt and Adelson en 1983 (Burt & Adelson, 1983)

La pirámide Gaussiana es construida a partir de una imagen de muestra de entrada y mediante muestras suavizadas constantes de esta imagen se van realizando submuestreos de las mismas, mientras que las diferencias Gaussianas corresponden a las diferencias entre los niveles adyacentes de la pirámide. De esta manera los puntos de interés se obtienen a partir de los puntos en los que los valores de la diferencia Gaussiana asumen extremos con respecto tanto a las coordenadas espaciales en el dominio de la imagen y el nivel de escala en la pirámide.

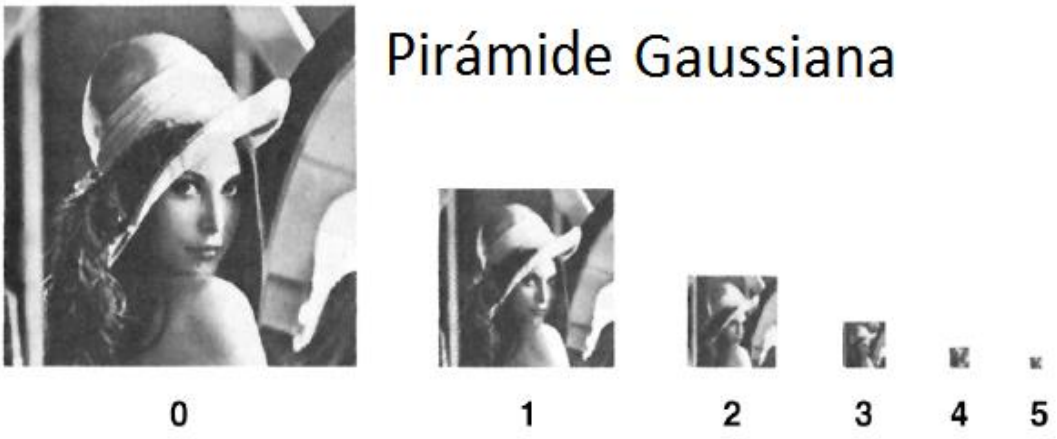

Imagen 4 Pirámide Gaussiana (SIFT) (Burt & Adelson, 1983)

En la figura 4 se puede apreciar lo propuesto bajo el nombre de pirámide Gaussiana, obviamente en cada uno de estos espacios escalas se deberán realizar los suavizados con los cuales se obtendrá las diferencias Gaussianas.
La primera situación a ser satisfecha para poder obtener una imagen libre de ruido es pasar a la imagen por algún filtro con el cual eliminar posibles distorsiones que se pudiesen detectar como puntos característicos sin serlo. Este método en particular utiliza filtros de Gauss:

$$G_{(x,y;s)} = \frac{1}{2\pi s}\, e^{-(x^2+y^2)/(2s)}$$

Ecuación 31 Filtro de Gauss (SIFT)

Donde $s = \sigma^2$ siendo σ la desviación estándar y s la varianza de los núcleos (kernel) Gaussianos.
Esta nueva matriz filtrada representa a la misma imagen original pero el resultado obtenido es una imagen suavizada y con eliminación de ruido.

La matriz original y la matriz filtrada son convolucionadas para obtener una mejor muestra sobre la cual descubrir los puntos característicos. Esta matriz se conoce como Laplaciana, esta operación es realizada de la siguiente manera:

$$L_{(x,y,s)} = G_{(x,y,s)} * I_{(x,y)}$$

Ecuación 32 Calculo Matriz Laplaciana (SIFT)

Donde $L_{(x,y,s)}$ es la matriz resultante de la convolución, $G_{(x,y,s)}$es la matriz suavizada e $I_{(x,y)}$ es la matriz original. El factor s es una magnitud de escala de la matriz.

La matriz Laplaciana así encontrada es sometida a un nuevo proceso, la ya mencionada diferencia de Gaussianas siendo necesario introducir un factor de escalamiento:

$$DOG_{(x,y,s)} = L_{(x,y,s+\Delta s)} - L_{(x,y,s)}$$

Ecuación 33 Cálculo de la matriz de diferencia de Gaussiana (SIFT)

Es evidente que mediante el procedimiento previo los puntos de interés contenidos en la matriz DOG, serán invariantes a la escala en el sentido que estos estarán preservados ante las transformaciones de escala y las transformaciones de niveles de la escala de la imagen están en concordancia con sus propias escalas. Al estar todo este proceso basado en un operador Laplaciano se aprovecha la condición del mismo de ser rotacionalmente invariante y por tanto los puntos de interés heredarán esta condición.

Esta matriz tiene todavía algunos puntos que podrían ser suprimidos con la finalidad de no realizar u obtener puntos característicos en elementos que no contengan mayor información de la imagen a tratar, parte de estos puntos

suprimibles son eliminados por la coincidencia o superación con cierto valor de umbral. Otros puntos que igualmente deben ser estudiados y en caso necesario suprimidos son los puntos que estén sobre bordes o límites de una figura dada, esto conlleva un tratamiento de mayor minuciosidad en el cual se identifica a los mismos a través del criterio de un radio entre los eigenvalores de la matriz Hessiana adquirida de DOG.

$$HL = \begin{bmatrix} DOGxx & DOGxy \\ DOGxy & DOGyy \end{bmatrix}$$

Ecuación 34 Matriz Laplaciana de diferencia Gaussiana (SIFT)

Una vez procesada esta matriz en la posición y escala del punto de interés, la misma puede ser reformulada in términos del determinante de la matriz Hessiana para permitir cálculos de mayor eficiencia:

$$\frac{L_{xx}L_{yy} - L_{xy}^2}{(L_{xx} + L_{yy})^2} \geq \frac{r}{(r+1)^2}$$

Ecuación 35 Calculo del radio de acción de los puntos de interés (SIFT)

Donde $r \geq 1$ denota el límite que discrimina entre eigenvalores relevantes y los que pueden ser despreciados. (Lowe, Distinctive image features from scale-invariant keypoints, 2004) (Lindeberg)

El siguiente paso ahora se centra en la determinación de la dirección dominante de los gradientes, para conseguir este objetivo se debe realizar la ubicación de los picos dentro de un vecindario de gradientes. Puede existir más de una dirección principal si es que alguna de ellas llega a representar un valor del 80% o más del gradiente de mayor magnitud, en este caso cada pico es procesado como un nuevo descriptor de la imagen para la correspondiente estimación de la orientación.

$$m(x,y) = \sqrt{((L(x+1,y) - L(x-1,y))^2 + ((L(x,y+1) - L(x,y-1))^2}$$

$$\theta(x,y) = \arctan\frac{L(x,y+1) - L(x,y-1)}{L(x+1,y) - L(x-1,y)}$$

Ecuación 36 magnitud y orientación de gradientes (SIFT)

Una vez obtenidas las escalas y orientaciones de los puntos de interés, una grilla rectangular es tendida en el dominio de la imagen siendo su centro el mismo punto de interés a estudiar.

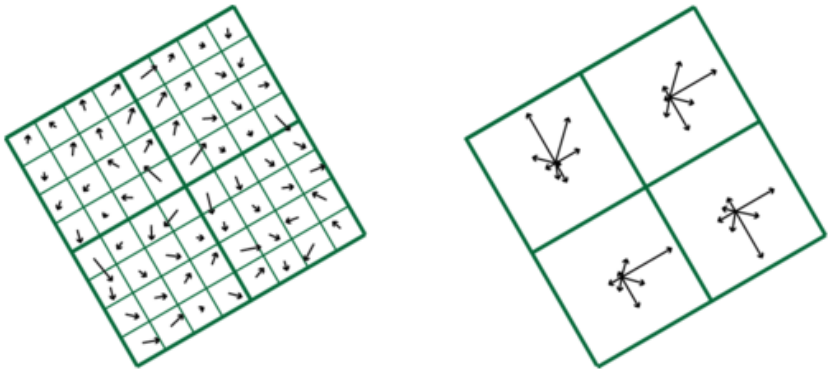

Figura 6 Orientación de gradientes. (SIFT) (Flores & Braun, 2011)

En la Figura 6 se pude apreciar como SIFT procesa desde los valores muestreados de las orientaciones y magnitudes de los gradientes para la obtención de un punto de interés con orientación y magnitud determinada por el pico dominante dentro de un histograma que cubre mallas de 4x4 puntos.

Algunas variaciones con las cuales se ha pretendido realizar mejoras al proceso de SIFT han sido ensayadas siendo una de ellas el método PCA-SIFT

planteado por Ke y Sukthankar en 2004 y el cual aplica el análisis de componentes principales (PCA) a la matriz de los gradiente normalizado extraída en el proceso de SIFT (Tao, Skubic, Han, Xia, & Chi, 2010). Con esto se logra obtener vectores característicos significantemente menores a los del estándar SIFT. (Ke, Yan; Sukthankar, Rahul;, 2004).

Otra alternativa planteada por Mikolajczyk y Schmid en 2005 es la técnica conocida como *Gradient Location-Orientation Histogram* (GLOH) la cual fue diseñada para incrementar la robustez y la varianza en los puntos de interés del descriptor, esta extensión cambia la localización de la grilla y usa PCA para reducir el tamaño de la misma mediante el procesamiento logarítmico de las posiciones polares de las direcciones radiales de la matriz de puntos característicos. (Mikolajczyk & Schmid, 2005)

CAPÍTULO 2

ALGORITMOS DE VISIÓN ARTIFICIAL Y SUS APLICACIONES EN RECONOCIMIENTO DE ROSTROS.

2.1. PCA utilización y formas de aplicación.

Principal Componenet Analysis (PCA) en particular es una técnica muy difundida para la parametrización de: formas, apariencia y movimiento, la misma ha sido desarrollada como una especie de paradigma en la visión artificial. Las representaciones que realiza *PCA* han demostrado ser útiles para la resolución de problemas como reconocimiento de objetos y rostros, seguimiento, detección y modelado de fondos.

Típicamente, los datos de entrenamiento para PCA son pre-procesados de alguna manera o son generados por algún otro algoritmo de visión (De la Torre & Black, 2001).

Esta técnica fue desarrollada en primera instancia en la comunidad de ingeniería donde la noción de un filtro es común mientras que el concepto de maximización de probabilidad no es muy conocido. Es comúnmente aplicada a datos reales o medidos.

La comunidad científica se ha dado cuenta que existen enfoques relativamente similares de aplicaciones en las cuales se utiliza los conceptos desarrollados por el método de PCA, entre algunos que se conocen están: *Grado de membresía (Woodburry & Manton, 1982)* utilizado en modelos de ciencias sociales, demográficos e informes médicos, *Inferencia de genotipo utilizando aditivos*

(Pritchard, 2000), Indexación semántica de probabilidad latente (Hofmann, 1999), Asignación latente de Dirichlet (Blei, 2003) y lógicamente el conocido y descrito ya en este trabajo *Eigenfaces* . Todos los métodos mencionados son equivalentes si se ignora la notación y la metodología estadística (Buntine & Jakulin, 2004).

A continuación se realiza un esquema de la aplicación del algoritmo independientemente del campo de investigación o desarrollo que se esté estudiando:
Se realiza la reducción de la dimensionalidad del espacio del problema a fin de llevar a cabo la reducción de la dimensionalidad del espacio del problema.

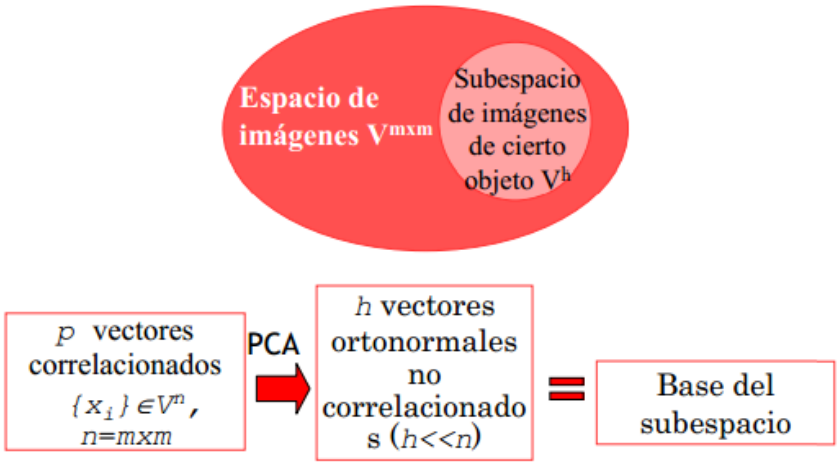

Figura 7 Esquema PCA (ESCET, 2014)

Como se puede leer en la figura 7, es necesario que los vectores descriptores sean ortogonales y no correlacionados para que de esta manera sea posible conseguir la mayor cantidad de información selectiva que no altere al resto de vectores, por tanto se maximizan las variaciones entre las muestras de

entrenamiento y entre ellas se encuentran los vectores de la nueva base (componentes principales).

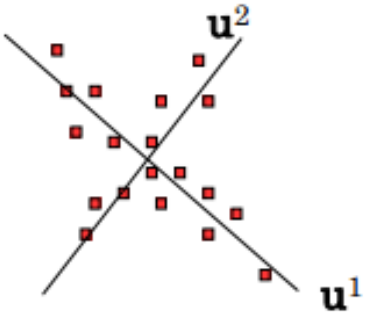

Figura 8 Representación de nuevos vectores para representación de una matriz. (ESCET, 2014)

Estos nuevos vectores deben estar ordenados según su varianza: los h primeros aportan un gran porcentaje de la varianza total.

Es recomendable trabajar con un máximo de vectores igual en cantidad al número de muestras originales o de preferencia menor.

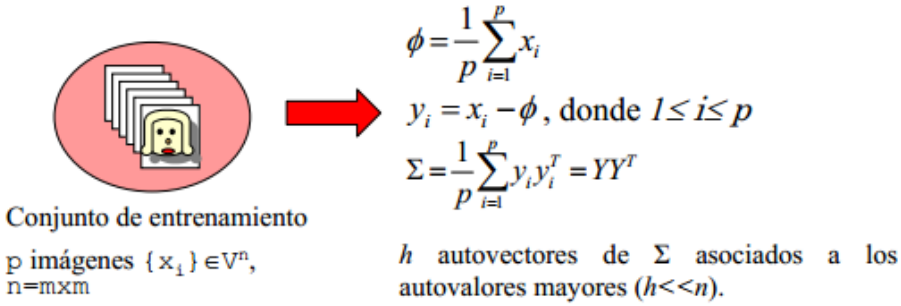

Figura 9 Modo de entrenamiento con las matrices de datos. (ESCET, 2014)

Una vez conseguida la matriz sobre la cual buscar la similitud con un nuevo elemento dentro del conjunto de datos se trabaja como se ejemplifica en la figura 10.

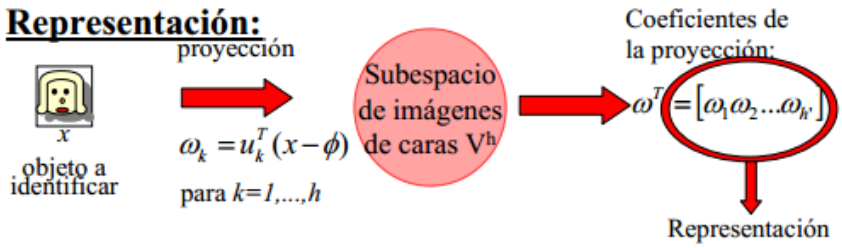

Figura 10 Representación de un nuevo ingreso sobre la base de datos. (ESCET, 2014)

A continuación se incluye el algoritmo programado para el reconocimiento de rostros:

```
void MainWindow::eigen()
{
//Se crean dos parametros para la afinación del algoritmo, el número de componentes y el umbral de reconocimiento:

    int num_components = componentesEigen;
    double threshold = umbralEigen;

//Obtener la ruta de acceso al archivo xls, archivo para la detección de los rostros, ojos, etc.
        string fn_haar = string("C:/opencv/data/haarcascades/haarcascade_frontalface_alt_tree.xml");

    //Obtener la ruta de acceso al archivo csv, archivo base donde están nombres y equivalencias de los sujetos a entrenar.
        string fn_csv = string("D:/Projects/Qt/eigenfaces_qt_V_0.2/csv.ext");

        int deviceId = atoi("0");

    //Vectores que contienen las imágenes y las etiquetas.
        vector<Mat> images;
        vector<int> labels;
    //Verificar los datos (Si el archivo no contiene direcciones útiles, da un error en la pantalla
        try {
            read_csv(fn_csv, images, labels);
        } catch (cv::Exception& e) {
```

```
            cerr << "Error al abrir el archivo \ " << fn_csv << "\". Razón: " << e.msg <<
endl;
    //cerrar la aplicación
            exit(1);
        }
        int im_width = images[0].cols;
        int im_height = images[0].rows;
    //Entrenar al FaceRecognizer con las imagenes del csv.
        Ptr<FaceRecognizer> model = createEigenFaceRecognizer(4, 50.0);
        model->train(images, labels);

    //Iniciar el trabajo con con el archivo xls que a la final es el clasificador
        CascadeClassifier haar_cascade;
        haar_cascade.load(fn_haar);
        VideoCapture cap; //iniciar variable para captar video de ipcamera
        cap.open("http://204.248.124.202/mjpg/video.mjpg"); //captar video de ipcamera

        if(!cap.isOpened()) {
            cerr << "Capture Device ID " << deviceId << "no puede ser abierto." << endl;
    //        return -1;
        }
    // Vector para los rectángulos de las caras
        vector < Rect_<int> > faces;
    // Mantener el frame del dispositivo de vídeo:
        Mat frame;
        for(;;){
            cap >> frame;
    // Clonar el frame para procesarlo
            Mat original = frame.clone();
    //Pasa a grises.
            Mat gray;
            cvtColor(original, gray, CV_BGR2GRAY);
    //Encuentra los rostros
            haar_cascade.detectMultiScale(gray, faces);

    //Se inicia el trabajo en el vídeo
            for(int i = 0; i < faces.size(); i++) {
    //busca los rostros
                Rect face_i = faces[i];
    //Recortar solo la cara
                Mat face = gray(face_i);
    //Cambiar el tamaño del rostro para homologar el trabajo.
                Mat face_resized;
```

```
            cv::resize(face, face_resized, Size(im_width, im_height), 1.0, 1.0,
INTER_CUBIC);
   //Se busca los matches
            int prediction = model->predict(face_resized);
   //Dibujar un rectangulo para ir escribiendo los nombres de los sujetos reconocidos
            rectangle(original, face_i, CV_RGB(255,0,0),1);
   //Linea donde escribir el resultado
            string box_text = format("Sujeto = %d", prediction);
            int pos_x = std::max(face_i.tl().x - 10, 0);
            int pos_y = std::max(face_i.tl().y - 10, 0);
            putText(original, box_text, Point(pos_x, pos_y),
FONT_HERSHEY_PLAIN, 1.0, CV_RGB(0,255,0), 2.0);
          }
          imshow("face_recognizer", original);
          if(flag_destroy ==1){
             key = 27;
             }
          else key = (char) waitKey(20);


  //Salir del loop presionando ESC:
          if(key==27){
             destroyAllWindows();
             break;
//             key = 2;
             key = (char) waitKey(20);
          }
       }
          }
```

2.2. LDA utilización y formas de aplicación.

Al igual que *PCA* el objetivo del *Linear Discriminant Analysis (LDA)* es la reducción de la dimensionalidad de un set de datos. El objetivo anterior puede ser realizado de varias maneras, la manera en que lo realiza *PCA* es en base a la reducción de redundancias y minimización de ruido sin que esto se detenga a analizar si los datos eliminados representan características de una o más clases. Esta pérdida de perspectiva es corregida en *LDA* a través de una clara diferenciación entre las distintas clases por el aumento de la dispersión entre las

mismas al tiempo que reduce la dispersión al interior de una clase concentrando de esta manera las características de mayor relevancia de cada clase.

La metodología LDA es utilizada en varias ciencias con distintas finalidades sin embargo, siempre tomando en cuenta que lo que se busca es identificar patrones, entre las principales aplicaciones están:

Medicina: Detección precoz del cáncer.

Ingeniería: Reconocimiento de Voz.

Informática: Clasificación de correo spam.

Economía: Adjudicación de créditos bancarios.

Literatura: Autoría de manuscritos de autores desconocidos.

Básicamente el algoritmo busca una proyección en un nuevo espacio dimensional en el que se pueda apreciar de mejor manera cada una de las clases, a fin de ubicar posteriormente si un dato cualquiera pertenece o no a determinada clase.

Como primer paso se asume la existencia de varias clases (colecciones de datos que representan algo en particular).

El segundo paso es la búsqueda de un escalar por la proyección de las muestras X en una línea dentro del nuevo espacio dimensional.

El segundo paso debe ser realizado con plena conciencia y reiteradamente para obtener una línea que maximice la separabilidad entre los nuevos escalares de representación de las clases.

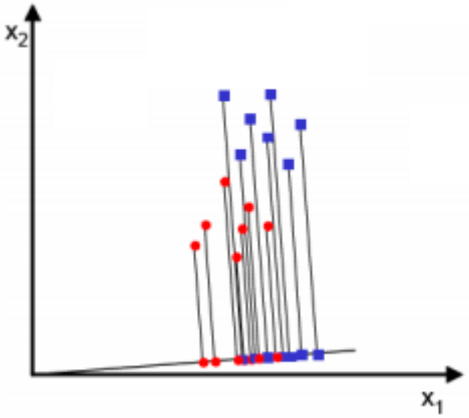

Figura 11 Representación de dos clases en las cuales no existe una clara separación (ESCET, 2014)

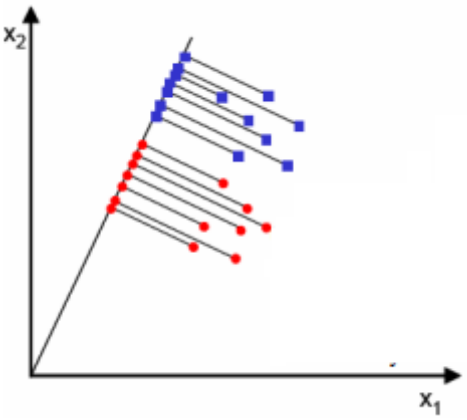

Figura 12 Representación de dos clases en una proyección a posterior de un tratamiento LDA (ESCET, 2014)

El concepto de la búsqueda del escalar Y en el cual las distintas clases puedan ser representadas de manera separada está ilustrado en las figuras 11 y 12. En la figura 11 se puede apreciar que las dos clases en discusión no están bien separadas cuando son proyectadas en la línea propuesta. Por otra parte la proyección

realizada en la figura 12, es exitosa en separar las dos clases y al mismo tiempo, reduce la dimensionalidad del problema de dos características (x1, x2) a solamente un escalar de valor Y.

En definitiva el clasificador de Fisher trata de encontrar una buena función discriminante que sea una combinación lineal de las variables originales. Cuando aplicamos la función a un dato nuevo nos dice a qué grupo pertenece.

Geométricamente lo que se busca es una dirección apropiada sobre la que proyectar los datos de los grupos conocidos y delos que queremos clasificar. En si la clasificación se la realiza en función de qué grupo está más cerca en esa dirección elegida.

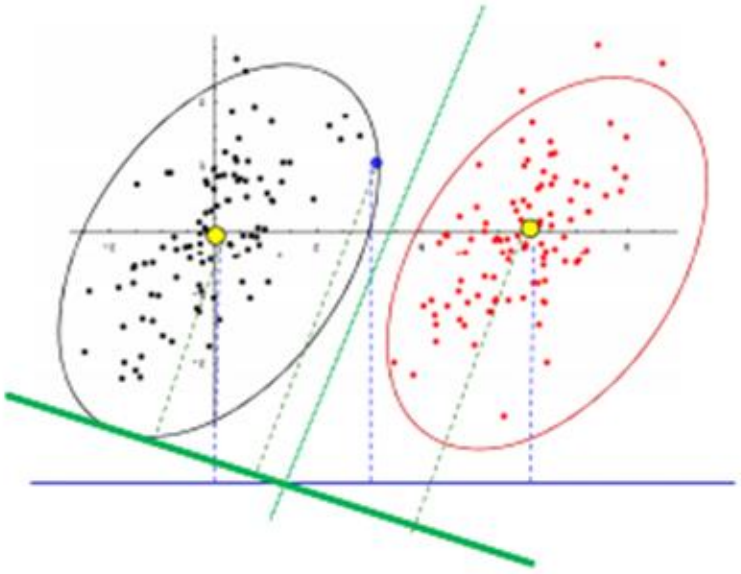

Figura 13 Representación de dos conjuntos de datos (ESCET, 2014)

Es importante, como se ha mencionado ya en el capítulo 1, que el momento de realizar la proyección de los datos se consiga una máxima variabilidad entre los grupos y una mínima variabilidad dentro de los mismos.

La discriminación sobre a qué grupo pertenece un dato nuevo dentro del universo de entrenamiento se la realiza en base a la mayor o menor distancia que el nuevo elemento posea en referencia a cada uno de los grupos, un ejemplo gráfico puede ser apreciado en la figura 14.

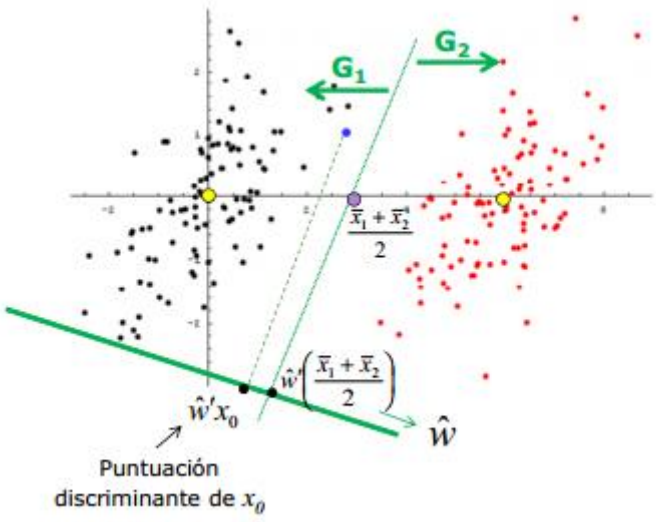

Figura 14 La regla de Fisher gráficamente. (ESCET, 2014)

En la figura 14 se puede apreciar cómo se realizaría la clasificación de un elemento en base a la distancia Euclídea que existe entre el dato a clasificar y los grupos de datos de entrenamiento, estas distancias son representadas como G1 y G2 respectivamente y en las direcciones de cada uno de los dos grupos de entrenamiento.

A continuación se encuentra el algoritmo programado para el reconocimiento de rostros.

```
void MainWindow::fisher()
{
   // Ruta a los archivos CSV:
   string fn_csv = string("D:/Projects/Qt/eigenfaces_qt_V_0.2/csv.ext");
```

```
    string fn_haar =
string("C:/opencv/data/haarcascades/haarcascade_frontalface_alt_tree.xml");
    int deviceId = atoi("0");
    // Estos vectores contendrán las imagenes y las etiquetas correspondientes:
    vector<Mat> images;
    vector<int> labels;
    // Leer los datos (se entregará un mensaje de error si el archivo es no válido)
    try {
        read_csv(fn_csv, images, labels);
    } catch (cv::Exception& e) {
        cerr << "Error al abrir el archivo \"" << fn_csv << "\". Razón: " << e.msg <<
endl;
        // no se puede hacer nada más y se sale de la rutina
        exit(1);
    }
    int im_width = images[0].cols;
    int im_height = images[0].rows;
    // Crear el reconocedor y entrenarlo:
    Ptr<FaceRecognizer> model = createFisherFaceRecognizer();
    model->train(images, labels);

    CascadeClassifier haar_cascade;
    haar_cascade.load(fn_haar);
    VideoCapture cap(deviceId);
    if(!cap.isOpened()) {
        cerr << "El Dispositivo ID " << deviceId << "no puede ser abierto." << endl;
//        return -1;
    }

    Mat frame;
    for(;;) {
        cap >> frame;
        Mat original = frame.clone();
        Mat gray;
        cvtColor(original, gray, CV_BGR2GRAY);
        vector< Rect_<int> > faces;
        haar_cascade.detectMultiScale(gray, faces);

        for(int i = 0; i < faces.size(); i++) {
            Rect face_i = faces[i];

            Mat face = gray(face_i);
            Mat face_resized;
```

```
        cv::resize(face, face_resized, Size(im_width, im_height), 1.0, 1.0,
INTER_CUBIC);

        int prediction = model->predict(face_resized);

        rectangle(original, face_i, CV_RGB(0,0,0), 1);
        string box_text = format("Sujeto Identificado = %d", prediction);
        int pos_x = std::max(face_i.tl().x - 10, 0);
        int pos_y = std::max(face_i.tl().y - 10, 0);
        // And now put it into the image:
        putText(original, box_text, Point(pos_x, pos_y), FONT_HERSHEY_PLAIN,
1.0, CV_RGB(0,0,0), 2.0);
      }
      imshow("face_recognizer", original);

      if(flag_destroy ==1){
         key = 27;
      }
      if(key == 27){
         destroyAllWindows();
         key = 2;
         break;
      }
      else key = (char) waitKey(20);
   }
    }
```

2.3. SURF utilización y formas de aplicación.

En *Speeded Up Robust Features* se maneja un concepto diferente, ya que se propone buscar las características relevantes de determinado elemento sin que en este proceso se involucre una reducción de la dimensionalidad del problema.

Esta técnica ha sido utilizada para realizar operaciones como el reconocimiento de objetos, comparación de características y reconocimiento de rostros.

Un vector de características describe a un conjunto de puntos que consiste específicamente en la localización de los mismos dentro de la imagen, la orientación local dentro de la detección del punto y la escala a la cual el punto de

interés fue detectado. Un vector descriptor puede ser luego usado para compararlo con otros descriptores de característicos.

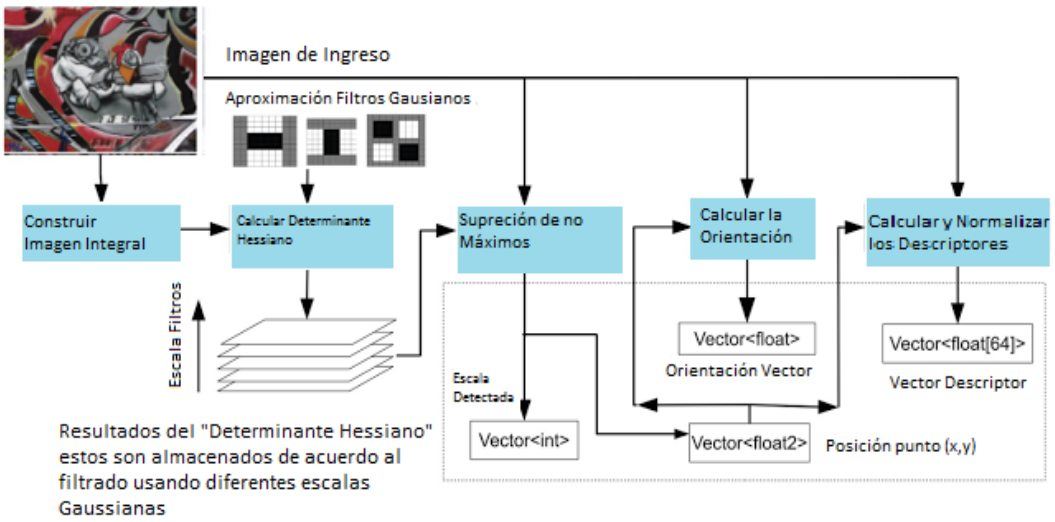

Figura 15 Proceso de algoritmo SURF (Computer Robot Vision).

En la figura 15 se puede apreciar un diagrama de la aplicación del algoritmo SURF para la identificación de características dentro de una matriz que para el caso de este trabajo deberá ser la representación de un rostro.

Para encontrar los puntos de interés SURF aplica una detección rápida por el hessiano de la matriz que utiliza una aproximación por filtros Gaussianos a diferentes escalas para generar un arreglo de matrices Hessianas que representará a la matriz original motivo del estudio.

El uso de la imagen integral, explicada ya anteriormente, radica en que la misma otorga las facilidades necesarias para escalar los filtros en lugar de escalar la imagen.

La ubicación de los puntos de interés es calculada por la búsqueda de los máximos y mínimos dentro de la imagen a diferentes escalas usando el arreglo de matrices hessianas generado anteriormente.

La orientación de un punto de interés siempre será invariante respecto a la rotación que pudiera tener la imagen. La cuarta etapa expuesta en la figura 15; es calculada usando la respuesta que se obtenga a los filtros wavelet en las direcciones X e Y en los vecinos más cercanos al punto detectado. La orientación dominante del vecindario estudiado es seleccionada por la rotación de un segmento de círculo de 60º alrededor del origen (Computer Robot Vision). En cada posición, las respuestas al interior de este segmento en los ejes X e Y son asumidas y usadas para formar un vector de respuesta. La orientación del vector de mayor dimensión se transforma en la orientación del punto característico (The Magazine of Record for the Embedded computing industry).

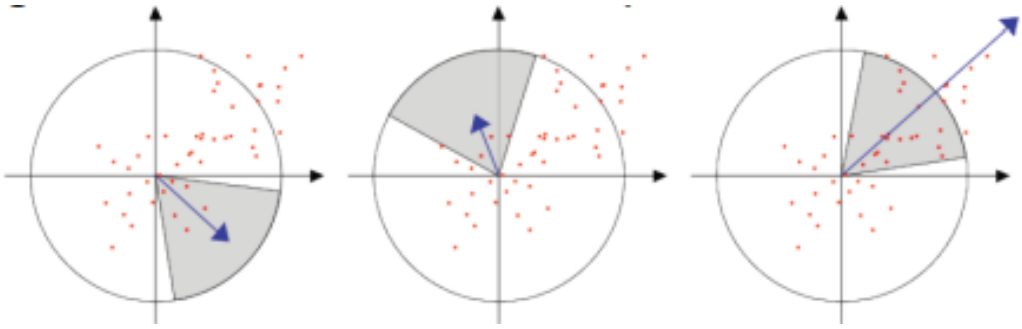

Figura 16 Selección del vector de orientación predominante (Computer Robot Vision)

A continuación el algoritmo con el cual se desarrolló el presente trabajo:

```
void MainWindow::surf()
{
      CascadeClassifier cascade;
      if
(!cascade.load("D:/Projects/Qt/FaceTrackingSurf_V_0.2/FaceTracking_plus_Surf/haa
rcascade_frontalface_alt.xml")){
         return;
      }
      QString nombre= QFileDialog::getOpenFileName();
```

```
        Mat object = imread(nombre.toLatin1().data());
        Mat gray_1;
        if (object.empty()){
           std::cout<< "Error al leer el archivo fotográfico base " << std::endl;
           return;
        }
        cvtColor(object, gray, CV_BGR2GRAY);
        equalizeHist(gray, gray);
        QImage photo = QImage((const unsigned char*)(object.data),
        object.cols,object.rows,QImage::Format_RGB888);
        vector<Rect> faces;
        cascade.detectMultiScale(gray, faces, 1.2, 3);
        for (int i = 0; i < faces.size(); i++)
        {
           Rect r = faces[i];
           rectangle(gray, Point(r.x, r.y), Point(r.x + r.width, r.y + r.height),
CV_RGB(0,255,0));
        }
        double minHessian = Hessianominimo;
        SurfFeatureDetector detector(minHessian);
        std::vector<KeyPoint> kp_object;
        detector.detect(gray, kp_object);
        SurfDescriptorExtractor extractor;
        Mat des_object;
        extractor.compute(gray, kp_object,des_object);
        FlannBasedMatcher matcher;
        namedWindow("PUNTOS ENCONTRADOS");
              CascadeClassifier                                    face_cascade;
             face_cascade.load("D:/Projects/Qt/FaceTrackingSurf_V_0.2/FaceTracking_
             plus_Surf/haarcascade_frontalface_alt.xml");
        VideoCapture cap(0);
        Mat captureFrame;
        Mat grayscaleFrame;
        std::vector<Point2f> obj_corners(4);
        //Obtener las esquinas del objeto identificado
        obj_corners[0] = cvPoint(0,0);
        obj_corners[1] = cvPoint( object.cols, 0 );
        obj_corners[2] = cvPoint( object.cols, object.rows );
        obj_corners[3] = cvPoint( 0, object.rows );
        int framecount = 0;
        while (key != 27){
           Mat frame;
           cap >> frame;
           ///*
```

```
        cvtColor(frame, grayscaleFrame, CV_BGR2GRAY);
        equalizeHist(grayscaleFrame, grayscaleFrame);
        std::vector<Rect> faces;
        face_cascade.detectMultiScale(grayscaleFrame, faces, 1.1, 3,
CV_HAAR_FIND_BIGGEST_OBJECT|CV_HAAR_SCALE_IMAGE, Size(30,30));
        for(int i = 0; i < faces.size(); i++){
           Point pt1(faces[i].x + faces[i].width, faces[i].y + faces[i].height);
           Point pt2(faces[i].x, faces[i].y);
           rectangle(frame, pt1, pt2, CV_RGB(0,1,0));
        }
        if (framecount < 5)
        {
           framecount++;
           continue;
        }

        Mat des_image, img_matches;
        std::vector<KeyPoint> kp_image;
        std::vector<vector<DMatch > > matches;
        std::vector<DMatch > good_matches;
        std::vector<Point2f> obj;
        std::vector<Point2f> scene;
        std::vector<Point2f> scene_corners(4);
        Mat H;
        Mat image;
        cvtColor(frame, image, CV_RGB2GRAY);
        detector.detect( image, kp_image );
        extractor.compute( image, kp_image, des_image );
        matcher.knnMatch(des_object, des_image, matches, 2);
        for(int i = 0; i < min(des_image.rows-1,(int) matches.size()); i++)
        {
           if((matches[i][0].distance < 0.6*(matches[i][1].distance)) && ((int)
matches[i].size()<=2 && (int) matches[i].size()>0))
           {
              good_matches.push_back(matches[i][0]);
           }
        }
        drawMatches( gray, kp_object, image, kp_image, good_matches,
img_matches, Scalar::all(-1), Scalar::all(-1), vector<char>(),
DrawMatchesFlags::NOT_DRAW_SINGLE_POINTS );
        if (good_matches.size() >= 4)
        {
           for( int i = 0; i < good_matches.size(); i++ )
           {              obj.push_back(kp_object[good_matches[i].queryIdx].pt);
```

```
            scene.push_back(kp_image[good_matches[i].trainIdx].pt);
        }
    //Mostrar los resultados
    imshow("PUNTOS ENCONTRADOS",img_matches);
    QImage img= QImage((const unsigned char*)(frame.data),
    frame.cols,frame.rows,QImage::Format_RGB888);
    if(flag_destroy ==1){
        key = 27;
    }
    else key = (char) waitKey(1);
 }
 if(key==27){
    destroyAllWindows();
    key = 2;
 }
}
```

2.4. SIFT utilización y formas de aplicación.

Para la aplicación del algoritmo SIFT son llevados a cabo 4 pasos principales:

a) Detección de extremos en escala-espacio:

   Con este primer paso lo que se pretende conseguir es la identificación de puntos invariantes a la traslación, el escalado y la rotación. Adicionalmente los puntos que se seleccionan en este paso deberán ser afectados mínimamente por el ruido o pequeñas distorsiones. Los puntos que serán seleccionados pertenecerán a los máximos y mínimos obtenidos de las diferencias Gaussianas aplicadas en el espacio-escala de la imagen (Flores & Braun, 2011) (Aracena-Pizarro & Daneri-Alvarado, 2013).

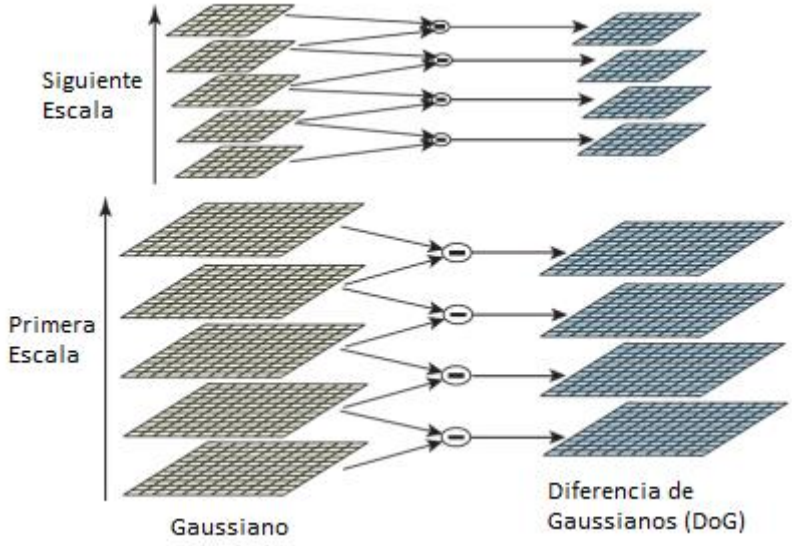

Figura 17 Escalas y diferencia de Gaussianas (Lowe, Distinctive image features from scale-invariant keypoints, 2004)

b) Localización del "punto clave": En cada localización candidata se adecua un modelo cuadrático detallado para determinar la localización y la escala. Los puntos clave se seleccionan basándose en su estabilidad.

   Además de retirar los puntos con poco contraste, se deben retirar también los puntos candidatos que tienen procedencia sobre una línea y no de una esquina (Flores & Braun, 2011) (Aracena-Pizarro & Daneri-Alvarado, 2013).

c) Asignación de orientación: A cada punto clave debe asignarse una orientación que está basada en la dirección del gradiente de la imagen. Al realizar futuras operaciones con los datos de estos puntos clave y sus orientaciones, se garantizará una invariancia a las transformaciones de orientación, escala y localización (Flores & Braun, 2011) (Aracena-Pizarro & Daneri-Alvarado, 2013).

d) Descriptor de “puntos clave”: Hasta este punto lo que se ha realizado es la asignación de escala, localización y orientación de los puntos clave. Por ello ahora se debe determinar un descriptor para cada punto, de modo que dicho descriptor posea cierta invariabilidad a cambios de iluminación y otras peculiaridades, todo esto deberá ser realizado en base al entorno del mismo (Flores & Braun, 2011) (Aracena-Pizarro & Daneri-Alvarado, 2013).

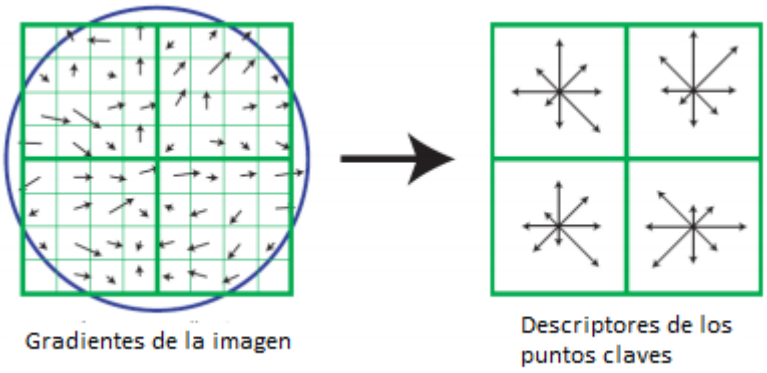

Figura 18 Paso de gradientes en regiones a descriptores de cada subregión (Lowe, Distinctive image features from scale-invariant keypoints, 2004).

A continuación el algoritmo con el cual se desarrolló el presente trabajo:

```
void MainWindow::sift()
{
//
**************************************************************************
*********************************************
   CascadeClassifier cascade_sift;
   if
(!cascade_sift.load("D:/Projects/Qt/FaceTrackingSurf_V_0.2/FaceTracking_plus_Surf
/haarcascade_frontalface_alt.xml")){
```

```
        return;
    }
//
*****************************************************************************
*********************************************

//
*****************************************************************************
*********************************************
    QString nombre_sift= QFileDialog::getOpenFileName();
    Mat object_sift = imread(nombre_sift.toLatin1().data());
    if (object_sift.empty()){
        std::cout<< "Error al leer el archivo fotográfico base " << std::endl;
        return;
    }
    cvtColor(object_sift, gray_sift, CV_BGR2GRAY);
    equalizeHist(gray_sift, gray_sift);
//
*****************************************************************************
*********************************************

//
*****************************************************************************
*********************************************
    vector<Rect> faces_sift;
    Mat just_face_sift;
    cascade_sift.detectMultiScale(gray_sift, faces_sift, 1.2, 3);
    for (int i = 0; i < faces_sift.size(); i++)
    {
        Rect r_sift = faces_sift[i];
//        Recortar solo la cara
        just_face_sift = gray_sift(r_sift);
//                rectangle(gray, Point(r.x, r.y), Point(r.x + r.width, r.y + r.height),
CV_RGB(0,255,0));
        rectangle(gray_sift, faces_sift[i], CV_RGB(255,0,0),1);
    }
//
*****************************************************************************
*********************************************

//
*****************************************************************************
*********************************************
//    SiftFeatureDetector detector_sift(Laplacianominimo);
```

```
    SiftFeatureDetector detector_sift(Hessianominimo);
//    SiftFeatureDetector detector_sift();
    std::vector<KeyPoint> kp_object_sift;
    detector_sift.detect(just_face_sift,kp_object_sift);
//
*****************************************************************************
*******************************************

//
*****************************************************************************
*******************************************
    SiftDescriptorExtractor extractor_sift;
    Mat des_object_sift;
    extractor_sift.compute(just_face_sift, kp_object_sift, des_object_sift);
//
*****************************************************************************
*********************************************

//
*****************************************************************************
*********************************************
    FlannBasedMatcher matcher_sift;
    namedWindow("DESCRIPTORES SIFT");
//
*****************************************************************************
*********************************************

//
*****************************************************************************
*********************************************
    CascadeClassifier face_cascade_sift;

face_cascade_sift.load("D:/Projects/Qt/FaceTrackingSurf_V_0.2/FaceTracking_plus_
Surf/haarcascade_frontalface_alt.xml");
    VideoCapture cap(0);
    Mat grayscaleFrame_sift;
    std::vector<Point2f> obj_corners(4);
    obj_corners[0] = cvPoint(0,0);
    obj_corners[1] = cvPoint( object.cols, 0 );
    obj_corners[2] = cvPoint( object.cols, object.rows );
    obj_corners[3] = cvPoint( 0, object.rows );
    int framecount = 0;
    while (key != 27){
        Mat frame;
```

```
        cap >> frame;
        cvtColor(frame, grayscaleFrame_sift, CV_BGR2GRAY);
        equalizeHist(grayscaleFrame_sift, grayscaleFrame_sift);
        std::vector<Rect> faces_sift;
        //find faces and store them in the vector array
        face_cascade_sift.detectMultiScale(grayscaleFrame_sift, faces_sift, 1.1, 3,
CV_HAAR_FIND_BIGGEST_OBJECT|CV_HAAR_SCALE_IMAGE, Size(30,30));

        //draw a rectangle for all found faces in the vector array on the original image
        for(int i = 0; i < faces_sift.size(); i++){
           Point pt1(faces_sift[i].x + faces_sift[i].width, faces_sift[i].y +
faces_sift[i].height);
           Point pt2(faces_sift[i].x, faces_sift[i].y);
           rectangle(frame, pt1, pt2, CV_RGB(0,1,0));
        }
        if (framecount < 5)
        {
           framecount++;
           continue;
        }
        Mat des_image_sift, img_matches_sift;
        std::vector<KeyPoint> kp_image_sift;
        std::vector<vector<DMatch > > matches_sift;
        std::vector<DMatch> good_matches_sift;
        std::vector<Point2f> obj_sift;
        std::vector<Point2f> scene_sift;
//        std::vector<Point2f> scene_corners(4);
//        Mat H;
        Mat image_sift;
        cvtColor(frame, image_sift, CV_RGB2GRAY);
        detector_sift.detect( image_sift, kp_image_sift );
        extractor_sift.compute( image_sift, kp_image_sift, des_image_sift );
        matcher_sift.knnMatch(des_object_sift, des_image_sift, matches_sift, 2);
        for(int i = 0; i < min(des_image_sift.rows-1,(int) matches_sift.size()); i++)
//THIS LOOP IS SENSITIVE TO SEGFAULTS
        {
           if((matches_sift[i][0].distance < 0.6*(matches_sift[i][1].distance)) && ((int)
matches_sift[i].size()<=2 && (int) matches_sift[i].size()>0))
           {
              good_matches_sift.push_back(matches_sift[i][0]);
           }
        }
//        Dibujar las lineas entre los puntos de coincidencia
```

```
//        drawMatches(gray, kp_object, image, kp_image, good_matches, img_matches, Scalar::all(-1), Scalar::all(-1), vector<char>(), DrawMatchesFlags::NOT_DRAW_SINGLE_POINTS );
        drawMatches(just_face_sift, kp_object_sift, image_sift, kp_image_sift, good_matches_sift, img_matches_sift, Scalar::all(-1), Scalar::all(-1), vector<char>(), DrawMatchesFlags::NOT_DRAW_SINGLE_POINTS );
        if (good_matches_sift.size() >= 4)
        {
            for( int i = 0; i < good_matches_sift.size(); i++ )
            {
                //Obtener los puntos claves obtenidos en goodmatches
                obj_sift.push_back(kp_object_sift[good_matches_sift[i].queryIdx].pt);
                scene_sift.push_back(kp_image_sift[good_matches_sift[i].trainIdx].pt);
            }
//*************************************************************************************************

//*************************************************************************************************
        }

        //Mostrar los resultados

//          if (good_matches.size()>15) {
        if (good_matches_sift.size()>0) {
            string box_text = format("Cantidad de descriptores encontrados = %d", good_matches_sift.size());
            // And now put it into the image:
            putText(img_matches_sift, box_text, Point(10,10), FONT_HERSHEY_PLAIN, 1.0, CV_RGB(255,0,0), 2.0);
            imshow("DESCRIPTORES SIFT",img_matches_sift);
        }else {
            string box_text = format("Sujeto no reconocido %d", good_matches_sift.size());
            // And now put it into the image:
            putText(object_sift, box_text, Point(10,10), FONT_HERSHEY_PLAIN, 1.0, CV_RGB(255,0,0), 2.0);
//                putText(just_face, box_text, Point(10,10), FONT_HERSHEY_PLAIN, 1.0, CV_RGB(255,0,0), 2.0);
            imshow("DESCRIPTORES SIFT",object_sift);
//              imshow("DESCRIPTORES SIFT",just_face);
        }
//                                  QImage img = QImage((const unsigned char*)(frame.data),frame.cols,frame.rows,QImage::Format_RGB888);
```

```
        if(flag_destroy ==1){
            key = 27;
        }
        else key = (char) waitKey(1);

    }
    if(key==27){
        destroyAllWindows();
        key = 2;
    }

}
```

CAPÍTULO 3

ARQUITECTURA Y DISEÑO PARA LA IMPLEMENTACIÓN DE LOS ALGORITMOS DE IDENTIFICACIÓN DE ROSTROS DENTRO DE LA PLATAFORMA DEL ECU-9-1-1.

En las últimas décadas, el campo de los sistemas de vigilancia ha capturado la atención de la industria, la investigación y de la academia. El interés recientemente ha crecido en virtud de la necesidad de garantizar la seguridad pública, por lo tanto, gobiernos, empresas e incluso ciudadanos invierten más dinero en equipamiento, aspecto que hace que se sientan seguros (Gascueña & Fernández-Caballero, 2011).

El reciente interés en la vigilancia pública, militar y comercial ha incrementado la necesidad de crear y desplegar sistemas de vigilancia automáticos e inteligentes. En escenarios como por ejemplo el transporte público, estos sistemas pueden ayudar en el monitoreo y almacenamiento de situaciones de interés en las cuales está envuelto el público, analizado desde los dos puntos de vista: como individuos independientes o como muchedumbre (Valera & Velastin, 2005).

La llave de funcionalidad de estas aplicaciones es la detección de eventos anormales usando sistemas multicámara. Sin embargo, los sistemas inteligentes de vídeo encaran muchos retos. Primero se debe considerar el vasto volumen de datos que debe ser manejado. Segundo la libertad de las acciones humanas significa que los sistemas se verán confrontados con muchas e impredecibles tareas y esto lleva a requerimientos cumputacionales ilimitados. En tercer lugar, la provisión de un apropiado servicio requiere que el sistema conozca el escenario contextual, esto significa que incluso tareas más complejas serán necesarias de procesamiento. (Brut, Mihaela; et al., 2011).

Bajo las premisas anteriores es evidente que se debe poseer una arquitectura con la cual se pueda manejar eficientemente los sistemas de vídeo vigilancia y al mismo tiempo permitir una escalabilidad y expansión futura.

La funcionalidad ofertada por un sistema de video vigilancia depende siempre de las mejoras tecnológicas que se han ido introduciendo en los mismos, por esta razón se puede hablar de tres generaciones:

3.1. Primera Generación:

En la primera generación (1960-1980), circuitos cerrados de televisión (CCTV) análogos fueron usados, los mismos consistían de varias cámaras conectadas a monitores en serie. Estos sistemas no procesaban información y requerían de un operador humano permanentemente concentrado en analizar la situación observada en un monitor. Esta práctica es tediosa, costosa e ineficiente, dado que la atención del operador visual hacia los monitores decrece a inaceptables niveles luego de un corto período de tiempo (aproximadamente 20 minutos), de esta manera la probabilidad de que una situación anómala no sea detectada se ve incrementada. Adicionalmente las señales analógicas también tienen desventajas por ser muy propensas al ruido, requieren grandes anchos de banda para su transmisión y más espacio para ser almacenadas que una señal digital (Gascueña & Fernández-Caballero, 2011).

La arquitectura básica de estos sistemas está representada en la Figura 19, es relevante observar en la misma la forma de grabación que era, por aquel entonces, en cintas de vídeo.

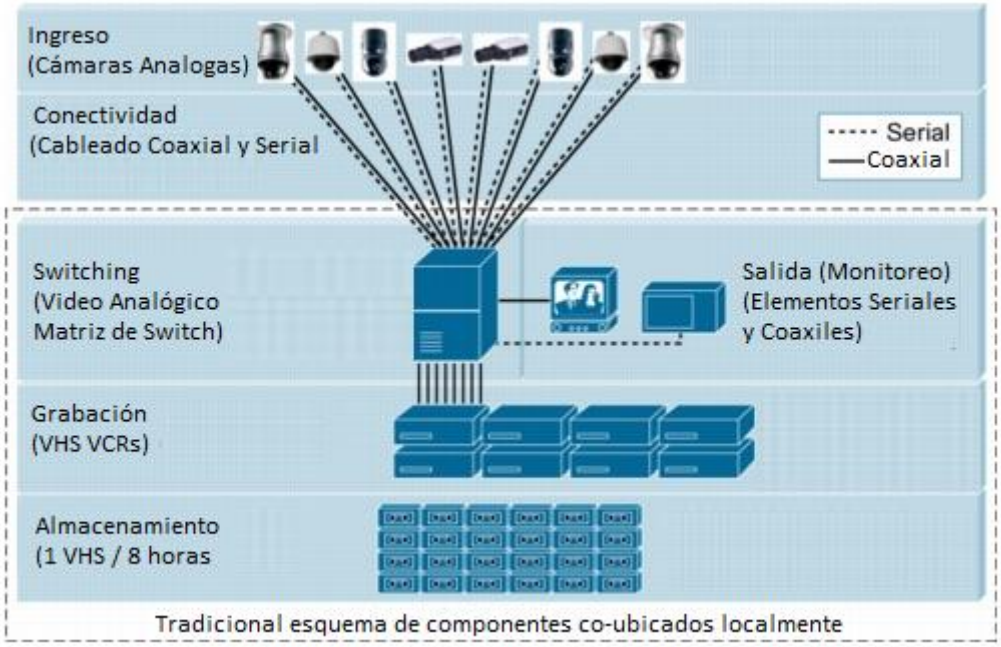

Figura 19 Arquitectura de sistema de Video de primera generación (Valera & Velastin, 2005)

3.2. Segunda Generación:

En la segunda generación (1990-2000), los avances alcanzados en la comunicación de vídeos digitales, como por ejemplo: compresión digital, reducción del ancho de banda y transmisión robusta; fueron usados para incrementar la eficiencia de los sistemas de vídeo vigilancia. Los sistemas de CCTV son usados entonces con el acompañamiento de tecnología de visión por computadora para la automatización de procesos en las imágenes con la intención de detectar eventos de alarma en tiempo real mientras son grabados los vídeos, en la Figura 20 se representa la arquitectura de estos sistemas en la cual existe la Unidad de Procesamiento de Imágenes. Estos sistemas semiautomáticos requieren de algoritmos de detección y seguimiento de imágenes robustos para poder

analizar el comportamiento en una escena. Es obvio que esos sistemas representan una clara mejora en referencia a los sistemas de la primera generación puesto que reducen la dependencia de los operadores visuales para la detección de situaciones anómalas. La dificultad que presentan es que los algoritmos utilizados no incorporan aún un grado de madurez suficiente dado que generan un alto porcentaje de falsas alarmas. La mayoría de estos sistemas utilizan una arquitectura centralizada para el procesamiento de datos (Gascueña & Fernández-Caballero, 2011). La Figura 21 esquematiza la manera en la que trabaja la Unidad de Procesamiento de Imágenes y la generación de las alarmas que deberán ser interpretadas por los operadores.

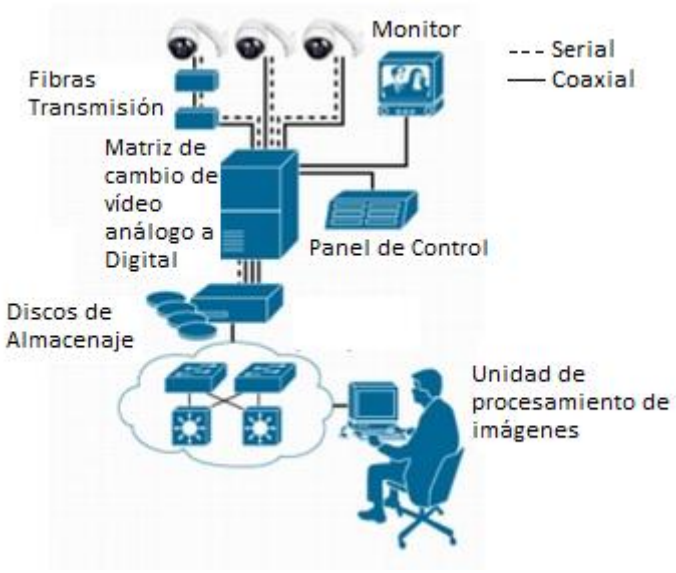

Figura 20 Arquitectura de sistema de Vídeo de segunda Generación (Valera & Velastin, 2005)

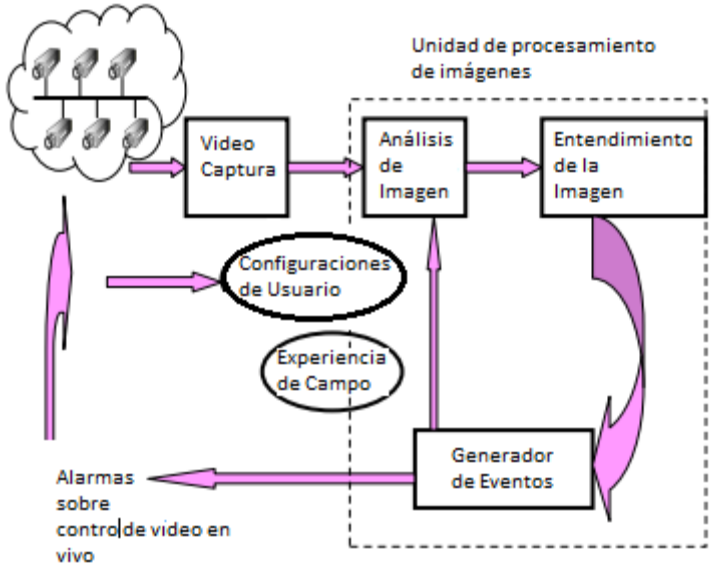

Figura 21 Esquema de funcionamiento de sistemas de segunda Generación (Valera & Velastin, 2005)

3.3. Tercera Generación

En la actual generación de sistemas de vigilancia (2000 hasta la fecha) se han acoplado muchas herramientas mediante las cuales se puede realizar una inspección más profunda del escenario en el que se desenvuelve la plataforma de vigilancia, para el caso se puede mencionar: cámaras fijas, cámaras PTZ (*pan-tilt-zoom*), sensores de audio e incluso en algunos casos sensores RFID (*Radio-Frequency Identification*) que son distribuidos geográficamente a lo largo del área a vigilar. Desde el punto de vista del procesamiento de imágenes, estos nuevos sistemas están basados en arquitecturas distribuidas con la finalidad de aprovechar la capacidad de inteligencia embebida en dispositivos ofreciendo al mismo tiempo escalabilidad y robustez. El principal problema que estos nuevos paradigmas involucran subyace en la necesidad de resolver la integración de las señales emitidas por sensores de diversas funcionalidades, siendo necesario también

establecer una correcta correspondencia de las señales en tiempo y espacio al coordinar y distribuir las tareas de procesamiento y comunicación entre los elementos (Gascueña & Fernández-Caballero, 2011). Un esquema de la arquitectura con la que estos sistemas son armados se la observa en la Figura 22.

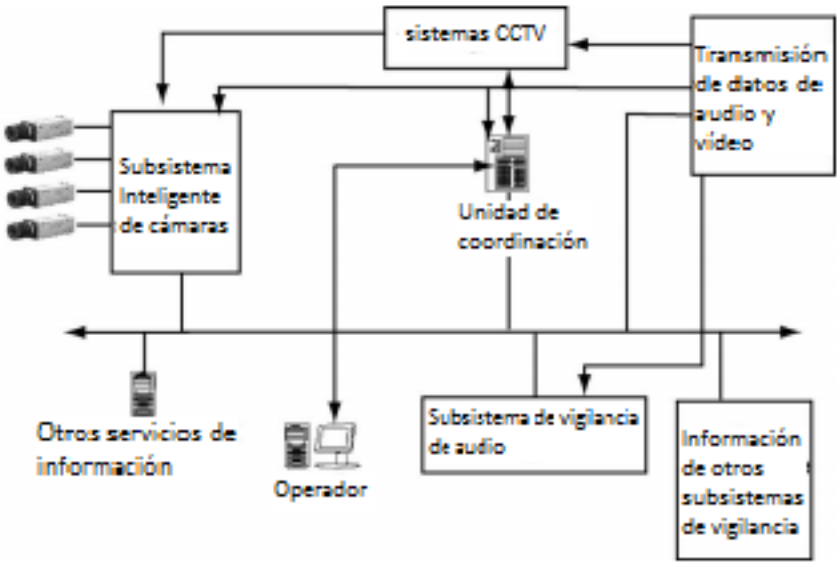

Figura 22 Arquitectura de sistema de Vídeo de Tercera Generación (Valera & Velastin, 2005)

Con lo descrito hasta aquí, se evidencia que una arquitectura centralizada no es posible de ser implementada y mucho menos mantenida adecuadamente en la actualidad, puesto que sería imposible la concentración de todas las tareas que pueden ser realizadas sobre un sistema de vídeo en un solo procesador o unidad de procesamiento, siendo lógico realizar subsistemas inteligentes designados exclusivamente al procesamiento y especialización de una tarea en concreto.

Con la base de las premisas anteriores y dado que actualmente en el Ecuador existe una plataforma ya instalada de aproximadamente 2000 cámaras a la que se pretende

expandir, al menos al doble en los próximos años, se debe tener en consideración estrategias tecnológicas para el manejo de esta gran cantidad de información. Por lo tanto, cuestiones como escalabilidad y usabilidad se presentan como aspectos sumamente importantes de ser atendidos y solucionados.

Para hacer frente a la creciente demanda de sistemas de vídeo vigilancia y seguridad, investigadores y desarrolladores han estado llevando a cabo mejoras a los sistemas tanto bajo la visión comercial como académica. Las aristas principales en las cuales se han concentrado, por ser las de mayor repercusión, han sido procesamiento de señales, comunicaciones, ingeniería de sistemas y visión por computadora (Valera & Velastin, 2005).

Los sistemas de vigilancia creados desde la parte comercial difieren de cierta manera de los desarrollados desde la parte académica, puesto que en el mundo comercial prevalece la tendencia a usar hardware de propósito específico y al incremento del uso de redes (*network*) y de cámaras digitales inteligentes. Entre las principales tareas que estos sistemas realizan están la detección de movimiento e intrusión en alguna área determinada y también la detección de paquetes (Valera & Velastin, 2005).

De su lado, la academia se ha concentrado en la tendencia de mejora de las tareas de procesamiento de imágenes al generar algoritmos más precisos y robustos para la detección, reconocimiento, seguimiento de objetos o personas y al mismo tiempo, el desarrollo de algoritmos de reconocimiento de actividades humanas (Valera & Velastin, 2005).

La aplicación de los algoritmos de reconocimiento de rostros en la plataforma del ECU-9-1-1 se deberá realizar de acuerdo a las tecnologías y arquitecturas de más reciente aplicación, a pesar de que la plataforma que se tiene instalada es un híbrido entre la primera y la segunda generación, dado que si bien todo el sistema es digital, no existen aplicativos con los cuales se pueda dar el soporte adecuado a los vídeo operadores del servicio. La pretensión de realizar modificaciones que actualicen la

plataforma a un sistema de 3ra generación se plantea como trabajo posterior al presente estudio, puesto que esto involucra el diseño de sistemas embebidos que realicen las tareas de vigilancia que sean necesarias, algunas ya se han mencionado anteriormente, como por ejemplo seguimiento de personas, o detección de movimiento. Asimismo, es necesario dotar al sistema de sensores complementarios como los RFID o micrófonos ambientales para conocer y tratar las señales de sonido que se pueden obtener en el espacio público.

CAPÍTULO 4

EVALUACIÓN DE LAS APLICACIONES INFORMÁTICAS UTILIZADAS.

4.1. Diseño del plan de experimentación.

La experimentación a realizar tiene la intención de establecer la técnica óptima con la cual se deberá realizar un trabajo de reconocimiento de rostros en aspectos de seguridad, principalmente en una plataforma pública como lo es el ECU-9-1-1. No se debe dejar de mencionar que lo que se pretende obtener es una herramienta de apoyo en la gestión de seguridad y más no un aplicativo exclusivo y con el cual basar la toma de decisiones, sino solamente aportar factores y criterios para el discernimiento.

Las pruebas deberán mantener el siguiente orden:

a) Recopilación de las bases de rostros a aplicar en los algoritmos holísticos.
b) Recopilación de las bases de rostros a aplicar en los algoritmos por descriptores
c) Ejecución de pruebas en función de las bases de entrenamiento que se hayan proyectado para algoritmos Holísticos.
d) Ejecución de pruebas de los algoritmos por descriptores.
e) Evaluación de las pruebas.

4.2. Ejecución de las pruebas.

El primer punto a desarrollar como se comentó anteriormente, es la recopilación de bases de rostros para la aplicación de los algoritmos. Para realizar este proceso se cumplió con la captura de fotografías del personal operativo del ECU-9-1-1 Zonal Austro, llegando a construir una base de datos de 58 individuos con 10 muestras de cada uno. Las fotografías están disponibles en formato PGM (*Portable Gray Map*) con dimensiones de 92x112 Bits, esto con la finalidad de mantener una uniformidad con los formatos que para similares propósitos proyectó la compañía AT&T entre los años 1992 y 1994. Una muestra de entrenamiento se la observa en la Imagen 5, en ella se presenta varias fotografías con las cuales se pretende obtener una imagen promedio de una persona en particular, por esta razón las imágenes son tomadas de diferentes ángulos y alturas.

Imagen 5 Ejemplo de cuatro muestras de un mismo individuo en formato PGM. (Autor)

En el caso de los métodos holísticos, es importante el poder trabajar con un formato ligero por dos principales temáticas: la necesidad de almacenar una gran base de datos y la posibilidad de realizar las tareas de entrenamiento y procesamiento de forma ágil. Por ello, es importante destacar que el formato PGM presenta un nivel aceptable en los dos campos descritos.

Por otra parte, para el trabajo de los métodos por descriptores se recogió una base de datos que no posee las restricciones requeridas por los métodos holísticos, dado que no es necesario el mantener una uniformidad entre las muestras, ya que los métodos por descriptores no necesitan entrenamiento y por lo tanto las imágenes pueden ser heterogéneas tanto entre la muestra a buscar y la muestra a discriminar. Una muestra de un sujeto puede ser cualquier fotografía del mismo que posea niveles mínimos de iluminación directa como se la que se aprecia en la Imagen 6.

Imagen 6 Ejemplo de una fotografía base para el renocimiento por métodos de descriptores. (Autor)

En la ejecución de las pruebas de los métodos por descriptores se orientó el entrenamiento con toda la base de los 58 individuos.

4.2.1. Holísticos

Eigenfaces: En estos algoritmos se tratará de reconocer a los individuos cambiando los dos parámetros fundamentales dentro de la estructura matemática que rige a este método, es decir el número de eigenvectores (eigenfaces) y el umbral bajo el cual los puntos similares serán reconocidos.

El número de eigenfaces con el cual se trabajará será ingresado primero y el valor de umbral bajo el cual se podrá decir que los puntos son similares se lo colocará en segundo lugar, así la simbología a utilizarse será: 17/2500 lo que representa 17 eigenfaces con un umbral de reconocimiento igual o menor a 2500

4.2.1.1. Con el algoritmo de Eigenfaces y con parámetros de (17/5000.);

SUJETO	RECONOCE AL SUJETO
66	60
80	60
82	77
86	83
87	85
88	85
120	87
83	88
115	88
94	93
104	94
107	94
105	96
99	116
112	119
124	123
116	124
127	124
108	127
60	SI
61	SI
62	SI
63	SI
64	SI
65	SI
67	SI
68	SI
69	SI
70	SI
71	SI
72	SI
73	SI
74	SI
75	SI
76	SI
77	SI
78	SI
79	SI
81	SI
84	SI
85	SI
89	SI
90	SI
91	SI
92	SI
93	SI
95	SI
96	SI
97	SI
98	SI
100	SI
101	SI
102	SI
103	SI
106	SI
109	SI
110	SI
111	SI
113	SI
114	SI
117	SI
118	SI
119	SI
121	SI
122	SI
123	SI
125	SI
126	SI

Tabla 1 Resultados experimentación Eigenfaces (17/5000).

4.2.1.2. Con el algoritmo de Eigenfaces y con parámetros de 17/2500.

SUJETO	RECONOCE AL SUJETO
66	NO
68	NO
69	NO
82	NO
83	NO
99	NO
101	NO
102	NO
104	NO
120	NO
80	60
86	83
87	85
88	85
115	88
94	93
107	94
105	96
112	119
124	123
116	124
127	124
108	127
60	SI
61	SI
62	SI
63	SI
64	SI
65	SI
67	SI
70	SI
71	SI
72	SI
73	SI
74	SI
75	SI
76	SI
77	SI
78	SI
79	SI
81	SI
84	SI
85	SI
89	SI
90	SI
91	SI
92	SI
93	SI
95	SI
96	SI
97	SI
98	SI
100	SI
103	SI
106	SI
109	SI
110	SI
111	SI
113	SI
114	SI
117	SI
118	SI
119	SI
121	SI
122	SI
123	SI
125	SI
126	SI

Tabla 2 Resultados experimentación Eigenfaces (17/2500)

4.2.1.3. Con el algoritmo de Eigenfaces y con parámetros de 17/1250

SUJETO	RECONOCE AL SUJETO
60	NO
63	NO
64	NO
66	NO
67	NO
68	NO
69	NO
70	NO
74	NO
76	NO
77	NO
78	NO
79	NO
80	NO
81	NO
82	NO
83	NO
84	NO
85	NO
86	NO
87	NO
88	NO
90	NO
91	NO
92	NO
93	NO
94	NO
96	NO
97	NO
98	NO
99	NO
100	NO
101	NO
102	NO
103	NO
104	NO
105	NO
106	NO
107	NO
108	NO
109	NO
111	NO
112	NO
113	NO
115	NO
116	NO
117	NO
119	NO
120	NO
121	NO
122	NO
123	NO
124	NO
127	NO
61	SI
62	SI
65	SI
71	SI
72	SI
73	SI
75	SI
89	SI
95	SI
110	SI
114	SI
118	SI
125	SI
126	SI

Tabla 3 Resultados experimentación Eigenfaces (17/1250)

4.2.1.4. Con el algoritmo de Eigenfaces y con parámetros de 80/5000

SUJETO	RECONOCE AL SUJETO
66	63
80	60
83	93
86	83
87	85
88	85
94	93
103	112
104	94
105	96
107	94
108	92
112	119
115	126
116	124
120	87
124	126
127	90
60	SI
61	SI
62	SI
63	SI
64	SI
65	SI
67	SI
68	SI
69	SI
70	SI
71	SI
72	SI
73	SI
74	SI
75	SI
76	SI
77	SI
78	SI
79	SI
81	SI
82	SI
84	SI
85	SI
89	SI
90	SI
91	SI
92	SI
93	SI
95	SI
96	SI
97	SI
98	SI
99	SI
100	SI
101	SI
102	SI
106	SI
109	SI
110	SI
111	SI
113	SI
114	SI
117	SI
118	SI
119	SI
121	SI
122	SI
123	SI
125	SI
126	SI

Tabla 4 Resultados experimentación Eigenfaces (80/5000)

4.2.1.5. Con el algoritmo de Eigenfaces y con parámetros de 80/2500

SUJETO	RECONOCE AL SUJETO
60	NO
63	NO
66	NO
68	NO
69	NO
70	NO
76	NO
77	NO
78	NO
79	NO
80	NO
81	NO
82	NO
83	NO
84	NO
86	NO
87	NO
89	NO
98	NO
99	NO
100	NO
101	NO
102	NO
103	NO
104	NO
105	NO
107	NO
108	NO
109	NO
112	NO
115	NO
116	NO
120	NO
123	NO
124	NO
127	NO
94	93
61	SI
62	SI
64	SI
65	SI
67	SI
71	SI
72	SI
73	SI
74	SI
75	SI
85	SI
88	SI
90	SI
91	SI
92	SI
93	SI
95	SI
96	SI
97	SI
106	SI
110	SI
111	SI
113	SI
114	SI
117	SI
118	SI
119	SI
121	SI
122	SI
125	SI
126	SI

Tabla 5 Resultados experimentación Eigenfaces (80/2500)

4.2.1.6. Con el algoritmo de Eigenfaces y con parámetros de 80/1250

SUJETO	RECONOCE AL SUJETO
60	NO
61	NO
62	NO
63	NO
64	NO
65	NO
66	NO
67	NO
68	NO
69	NO
70	NO
72	NO
74	NO
75	NO
76	NO
77	NO
78	NO
79	NO
80	NO
81	NO
82	NO
83	NO
84	NO
85	NO
86	NO
87	NO
88	NO
89	NO
90	NO
91	NO
92	NO
93	NO
94	NO
96	NO
97	NO
98	NO
99	NO
100	NO
101	NO
102	NO
103	NO
104	NO
105	NO
106	NO
107	NO
108	NO
109	NO
110	NO
111	NO
112	NO
113	NO
114	NO
115	NO
116	NO
117	NO
118	NO
119	NO
120	NO
121	NO
122	NO
123	NO
124	NO
126	NO
127	NO
71	SI
73	SI
95	SI
125	SI

Tabla 6 Resultados experimentación Eigenfaces (80/1250)

Fisherfaces: Con el algoritmo de Fisherfaces se realizará una experimentación similar a la ya trabajada con Eigenfaces, la única diferencia es que ahora el primer parámetro a definir es el número de clases con la que se trabajará, a manera de breve recordatorio se menciona que una clase es el agrupamiento de puntos relevantes resultante de variables estadísticas que maximizan la dispersión entre puntos similares y minimiza la dispersión dentro del grupo de puntos similares, y como en el caso anterior, el segundo parámetro es el valor de umbral bajo el cual se podrá decir que un grupo de puntos, o para este caso, clase, son similares a otro grupo. Así la simbología queda expresada de igual manera que en el método anterior 17/2500 significa que se buscará entre 17 clases con un umbral para determinar similitud de 2500.

4.2.1.7. Con el algoritmo de Fisherfaces y con parámetros de 5/800

SUJETO	RECONOCE AL SUJETO
67	73
72	64
80	82
101	102
113	118
119	127
121	125
125	121
126	127
60	SI
61	SI
62	SI
63	SI
64	SI
65	SI
66	SI
69	SI
73	SI
74	SI
75	SI
76	SI
77	SI
79	SI
81	SI
82	SI
83	SI
84	SI
86	SI
87	SI
88	SI
89	SI
90	SI
91	SI
92	SI
93	SI
94	SI
95	SI
96	SI
97	SI
100	SI
102	SI
103	SI
104	SI
105	SI
106	SI
107	SI
108	SI
109	SI
110	SI
111	SI
112	SI
114	SI
115	SI
116	SI
117	SI
118	SI
120	SI
122	SI
123	SI
124	SI
127	SI

Tabla 7 Resultados experimentación Fisherfaces (5/800)

4.2.1.8. Con el algoritmo de Fisherfaces y con parámetros de 5/1250

SUJETO	RECONOCE AL SUJETO
67	73
72	64
80	82
101	102
113	118
119	127
121	125
125	121
126	127
60	SI
61	SI
62	SI
63	SI
64	SI
65	SI
66	SI
69	SI
73	SI
74	SI
75	SI
76	SI
77	SI
79	SI
81	SI
82	SI
83	SI
84	SI
86	SI
87	SI
88	SI
89	SI
90	SI
91	SI
92	SI
93	SI
94	SI
95	SI
96	SI
97	SI
100	SI
102	SI
103	SI
104	SI
105	SI
106	SI
107	SI
108	SI
109	SI
110	SI
111	SI
112	SI
114	SI
115	SI
116	SI
117	SI
118	SI
120	SI
122	SI
123	SI
124	SI
127	SI

Tabla 8 Resultados experimentación Fisherfaces (5/1250)

4.2.1.9. Con el algoritmo de Fisherfaces y con parámetros de 10/800

SUJETO	RECONOCE AL SUJETO
100	NO
108	NO
112	NO
88	93
91	83
127	123
60	SI
61	SI
62	SI
63	SI
64	SI
65	SI
66	SI
67	SI
69	SI
72	SI
73	SI
74	SI
75	SI
76	SI
77	SI
79	SI
80	SI
81	SI
82	SI
83	SI
84	SI
86	SI
87	SI
89	SI
90	SI
92	SI
93	SI
94	SI
95	SI
96	SI
97	SI
101	SI
102	SI
103	SI
104	SI
105	SI
106	SI
107	SI
109	SI
110	SI
111	SI
113	SI
114	SI
115	SI
116	SI
117	SI
118	SI
119	SI
120	SI
121	SI
122	SI
123	SI
124	SI
125	SI
126	SI

Tabla 9 Resultados experimentación Fisherfaces (10/800)

4.2.1.10. Con el algoritmo de Fisherfaces y con parámetros de 10/1250

SUJETO	RECONOCE AL SUJETO
88	93
91	83
108	113
127	123
60	SI
61	SI
62	SI
63	SI
64	SI
65	SI
66	SI
67	SI
69	SI
72	SI
73	SI
74	SI
75	SI
76	SI
77	SI
79	SI
80	SI
81	SI
82	SI
83	SI
84	SI
86	SI
87	SI
89	SI
90	SI
92	SI
93	SI
94	SI
95	SI
96	SI
97	SI
100	SI
101	SI
102	SI
103	SI
104	SI
105	SI
106	SI
107	SI
109	SI
110	SI
111	SI
112	SI
113	SI
114	SI
115	SI
116	SI
117	SI
118	SI
119	SI
120	SI
121	SI
122	SI
123	SI
124	SI
125	SI
126	SI

Tabla 10 Resultados experimentación Fisherfaces (10/1250)

4.2.1.11. Con el algoritmo de Fisherfaces y con parámetros de 17/800

SUJETO	RECONOCE AL SUJETO
72	NO
100	NO
104	NO
107	NO
108	NO
112	NO
114	NO
118	NO
127	NO
121	125
126	127
60	SI
61	SI
62	SI
63	SI
64	SI
65	SI
66	SI
67	SI
69	SI
73	SI
74	SI
75	SI
76	SI
77	SI
79	SI
80	SI
81	SI
82	SI
83	SI
84	SI
86	SI
87	SI
88	SI
89	SI
90	SI
91	SI
92	SI
93	SI
94	SI
95	SI
96	SI
97	SI
101	SI
102	SI
103	SI
105	SI
106	SI
109	SI
110	SI
111	SI
113	SI
115	SI
116	SI
117	SI
119	SI
120	SI
122	SI
123	SI
124	SI
125	SI

Tabla 11 Resultados experimentación Fisherfaces (17/800)

4.2.1.12. Con el algoritmo de Fisherfaces y con parámetros de 17/1250

SUJETO	RECONOCE AL SUJETO
108	113
121	125
126	127
60	SI
61	SI
62	SI
63	SI
64	SI
65	SI
66	SI
67	SI
69	SI
72	SI
73	SI
74	SI
75	SI
76	SI
77	SI
79	SI
80	SI
81	SI
82	SI
83	SI
84	SI
86	SI
87	SI
88	SI
89	SI
90	SI
91	SI
92	SI
93	SI
94	SI
95	SI
96	SI
97	SI
100	SI
101	SI
102	SI
103	SI
104	SI
105	SI
106	SI
107	SI
109	SI
110	SI
111	SI
112	SI
113	SI
114	SI
115	SI
116	SI
117	SI
118	SI
119	SI
120	SI
122	SI
123	SI
124	SI
125	SI
127	SI

Tabla 12 Resultados experimentación Fisherfaces (17/1250)

4.2.2. Descriptores

SURF: El método de SURF trabaja con la matriz Hessiana, como se explicó en el capítulo 1, el valor de umbral con el cual se distinguirá si los valores de los vectores que arroje todo el procedimiento matemático de obtener la segunda derivada de una matriz (fotografía) será el único parámetro que se irá ajustando para obtener mejores resultados. De esta manera la simbología en este método es única, se colocará solamente un valor siendo el mismo el umbral bajo el cual se podrá decir que un vector reconocido es similar al de la imagen de muestra.

4.2.2.1. Con el algoritmo SURF y el parámetro en 17

SUJETO	PARAMETROS	CANTIDAD DE DESCRIPTORES ENCONTRADOS	RECONOCE AL INDIVIDUO EN UN STREAM DE VIDEO
106	17	0	NO
145	17	X	FP
61	17	1	SI
62	17	47	SI
63	17	41	SI
64	17	62	SI
65	17	4	SI
66	17	15	SI
67	17	3	SI
72	17	4	SI
73	17	30	SI
74	17	34	SI
75	17	25	SI
76	17	13	SI
77	17	4	SI
79	17	25	SI
80	17	10	SI
81	17	2	SI
84	17	35	SI
86	17	18	SI
87	17	4	SI
88	17	3	SI
89	17	8	SI
90	17	11	SI
92	17	4	SI
93	17	2	SI
94	17	2	SI
97	17	2	SI
100	17	40	SI
101	17	4	SI
103	17	9	SI
104	17	42	SI
105	17	32	SI
107	17	44	SI
108	17	20	SI
109	17	2	SI
110	17	28	SI
112	17	5	SI
113	17	4	SI
114	17	6	SI
115	17	1	SI
116	17	11	SI
118	17	62	SI
119	17	7	SI
121	17	15	SI
122	17	5	SI
123	17	1	SI
125	17	10	SI
128	17	36	SI
129	17	40	SI
130	17	61	SI
131	17	24	SI
132	17	3	SI
133	17	3	SI
134	17	12	SI
135	17	9	SI
136	17	7	SI
137	17	10	SI
138	17	3	SI
139	17	3	SI
140	17	5	SI
141	17	4	SI
142	17	10	SI
143	17	5	SI
144	17	5	SI
146	17	2	SI
147	17	7	SI

Tabla 13 Resultados experimentación SURF (17)

4.2.2.2. Con el algoritmo SURF y el parámetro en 1

SUJETO	PARAMETROS	CANTIDAD DE DESCRIPTORES ENCONTRADOS	RECONOCE AL INDIVIDUO EN UN STREAM DE VIDEO
106	17	0	NO
61	17	1	SI
62	17	29	SI
63	17	37	SI
64	17	72	SI
65	17	24	SI
66	17	10	SI
67	17	2	SI
72	17	3	SI
73	17	53	SI
74	17	49	SI
75	17	21	SI
76	17	4	SI
77	17	8	SI
79	17	10	SI
80	17	11	SI
81	17	1	SI
84	17	2	SI
86	17	17	SI
87	17	3	SI
88	17	3	SI
89	17	5	SI
90	17	10	SI
92	17	1	SI
93	17	5	SI
94	17	2	SI
97	17	3	SI
100	17	30	SI
101	17	5	SI
103	17	9	SI
104	17	41	SI
105	17	30	SI
107	17	31	SI
108	17	9	SI
109	17	6	SI
110	17	34	SI
112	17	2	SI
113	17	2	SI
114	17	14	SI
115	17	3	SI
116	17	12	SI
118	17	58	SI
119	17	6	SI
121	17	17	SI
122	17	5	SI
123	17	3	SI
125	17	13	SI
128	17	6	SI
129	17	36	SI
130	17	54	SI
131	17	33	SI
132	17	21	SI
133	17	1	SI
134	17	4	SI
135	17	4	SI
136	17	5	SI
137	17	2	SI
138	17	2	SI
139	17	3	SI
140	17	2	SI
141	17	4	SI
142	17	4	SI
143	17	5	SI
144	17	2	SI
145	17	3	SI
146	17	4	SI
147	17	6	SI
148	17	1	SI

Tabla 14 Resultados experimentación SURF (1)

4.2.2.3. Con el algoritmo SURF y el parámetro en 1500

SUJETO	PARAMETROS	CANTIDAD DE DESCRIPTORES ENCONTRADOS	RECONOCE AL INDIVIDUO EN UN STREAM DE VIDEO
106	1500	0	NO
133	1500	0	NO
61	1500	1	SI
62	1500	3	SI
63	1500	7	SI
64	1500	5	SI
65	1500	3	SI
66	1500	10	SI
67	1500	2	SI
72	1500	3	SI
73	1500	12	SI
74	1500	6	SI
75	1500	3	SI
76	1500	5	SI
77	1500	2	SI
79	1500	5	SI
80	1500	7	SI
81	1500	1	SI
84	1500	2	SI
86	1500	4	SI
87	1500	2	SI
88	1500	1	SI
89	1500	3	SI
90	1500	13	SI
92	1500	1	SI
93	1500	2	SI
94	1500	2	SI
97	1500	1	SI
100	1500	11	SI
101	1500	5	SI
103	1500	4	SI
104	1500	12	SI
105	1500	10	SI
107	1500	10	SI
108	1500	2	SI
109	1500	2	SI
110	1500	6	SI
112	1500	2	SI
113	1500	2	SI
114	1500	3	SI
115	1500	4	SI
116	1500	2	SI
118	1500	17	SI
119	1500	4	SI
121	1500	10	SI
122	1500	3	SI
123	1500	2	SI
125	1500	7	SI
128	1500	6	SI
129	1500	13	SI
130	1500	16	SI
131	1500	11	SI
132	1500	10	SI
134	1500	1	SI
135	1500	4	SI
136	1500	6	SI
137	1500	3	SI
138	1500	1	SI
139	1500	6	SI
140	1500	1	SI
141	1500	4	SI
142	1500	1	SI
143	1500	2	SI
144	1500	2	SI
145	1500	2	SI
146	1500	2	SI
147	1500	3	SI
148	1500	1	SI

Tabla 15 Resultados experimentación SURF (1500)

SIFT: Al trabajar con el método de SIFT se deberá también tener un umbral bajo el cual reconocer los puntos que luego de ser realizada toda el procesamiento matemático de los vectores explicados en el capítulo 1. Esté será por tanto el único valor con el cual se representará en la simbología del método. De esta manera la simbología será solo un número representando este umbral.

4.2.2.4. Con el algoritmo SIFT y el parámetro en 17

SUJETO	PARAMETROS	CANTIDAD DE DESCRIPTORES ENCONTRADOS	RECONOCE AL INDIVIDUO EN UN STREAM DE VIDEO
67	17	0	NO
80	17	0	NO
94	17	0	NO
106	17	0	NO
109	17	0	NO
112	17	0	NO
116	17	0	NO
119	17	0	NO
139	17	0	NO
140	17	0	NO
144	17	0	NO
146	17	0	NO
147	17	0	NO
148	17	0	NO
72	17	X	FP
79	17	X	FP
87	17	X	FP
92	17	X	FP
93	17	X	FP
123	17	X	FP
125	17	X	FP
61	17	3	SI
62	17	21	SI
63	17	20	SI
64	17	26	SI
65	17	17	SI
66	17	20	SI
73	17	34	SI
74	17	27	SI
75	17	9	SI
76	17	6	SI
77	17	1	SI
81	17	1	SI
84	17	2	SI
86	17	12	SI
88	17	1	SI
89	17	6	SI
90	17	16	SI
97	17	1	SI
100	17	14	SI
101	17	2	SI
103	17	12	SI
104	17	1	SI
105	17	16	SI
107	17	20	SI
108	17	4	SI
110	17	2	SI
113	17	1	SI
114	17	3	SI
115	17	4	SI
118	17	32	SI
121	17	2	SI
122	17	8	SI
128	17	17	SI
129	17	26	SI
130	17	20	SI
131	17	17	SI
132	17	19	SI

Tabla 16 Resultados experimentación SIFT (17)

4.2.2.5. Con el algoritmo SIFT y el parámetro en 1

SUJETO	PARAMETROS	CANTIDAD DE DESCRIPTORES ENCONTRADOS	RECONOCE AL INDIVIDUO EN UN STREAM DE VIDEO
80	1	0	NO
87	1	0	NO
94	1	0	NO
106	1	0	NO
109	1	0	NO
116	1	0	NO
119	1	0	NO
123	1	0	NO
140	1	0	NO
144	1	0	NO
145	1	0	NO
146	1	0	NO
147	1	0	NO
148	1	0	NO
67	1	X	FP
72	1	X	FP
79	1	X	FP
92	1	X	FP
93	1	X	FP
141	1	X	FP
61	1	3	SI
62	1	18	SI
63	1	23	SI
64	1	22	SI
65	1	4	SI
66	1	14	SI
73	1	32	SI
74	1	23	SI
75	1	10	SI
76	1	5	SI
77	1	1	SI
81	1	1	SI
84	1	6	SI
86	1	16	SI
88	1	1	SI
89	1	4	SI
90	1	13	SI
97	1	1	SI
100	1	17	SI
101	1	3	SI
103	1	9	SI
104	1	1	SI
105	1	25	SI
107	1	24	SI
108	1	2	SI
110	1	1	SI
112	1	1	SI
113	1	1	SI
114	1	2	SI
115	1	7	SI
118	1	36	SI
121	1	1	SI
122	1	9	SI
125	1	5	SI
128	1	17	SI
129	1	20	SI
130	1	8	SI
131	1	4	SI

Tabla 17 Resultados experimentación SIFT (1)

4.2.2.6. Con el algoritmo SIFT y el parámetro en 1500

SUJETO	PARAMETROS	CANTIDAD DE DESCRIPTORES ENCONTRADOS	RECONOCE AL INDIVIDUO EN UN STREAM DE VIDEO
67	1500	0	NO
80	1500	0	NO
88	1500	0	NO
94	1500	0	NO
106	1500	0	NO
109	1500	0	NO
112	1500	0	NO
116	1500	0	NO
119	1500	0	NO
123	1500	0	NO
140	1500	0	NO
141	1500	0	NO
146	1500	0	NO
148	1500	0	NO
72	1500	X	FP
79	1500	X	FP
87	1500	X	FP
107	1500	X	FP
125	1500	X	FP
61	1500	3	3
62	1500	16	16
63	1500	18	18
64	1500	17	17
65	1500	19	19
66	1500	8	8
73	1500	7	7
74	1500	27	27
75	1500	9	9
76	1500	8	8
77	1500	1	1
81	1500	1	1
84	1500	2	2
86	1500	11	11
89	1500	5	5
90	1500	6	6
92	1500	1	1
93	1500	1	1
97	1500	1	1
100	1500	22	22
101	1500	4	4
103	1500	9	9
104	1500	1	1
105	1500	23	23
108	1500	2	2
110	1500	2	2
113	1500	1	1
114	1500	4	4
115	1500	16	16
118	1500	36	36
121	1500	2	2
122	1500	8	8
128	1500	17	17
129	1500	23	23
130	1500	12	12
131	1500	4	4
132	1500	4	4
133	1500	8	8
134	1500	5	5

Tabla 18 Resultados experimentación SIFT (1500)

4.3. Evaluación de las pruebas.

4.3.1. Evaluación métodos Holísticos

En las aplicaciones de métodos holísticos y de lo expuesto en las Tablas 1 a la 12; se puede apreciar que el porcentaje de reconocimiento llega a tener en el mejor de los casos un 95,08% de taza de reconocimiento en el método de Fisherfaces.

Resultado	*EIGENFACES*						*FISHERFACES*					
	80/5000	*80/2500*	*80/1250*	*17/5000*	*17/2500*	*17/1250*	*10/800*	*10/1250*	*17/800*	*17/1250*	*5/800*	*5/1250*
NO	0,00%	52,94%	94,12%	0,00%	14,71%	79,41%	4,92%	0,00%	14,75%	0,00%	0,00%	0,00%
FP	26,47%	1,47%	0,00%	27,94%	19,12%	0,00%	4,92%	6,56%	3,28%	4,92%	14,75%	14,75%
SI	73,53%	45,59%	5,88%	72,06%	66,18%	20,59%	90,16%	93,44%	81,97%	95,08%	85,25%	85,25%

Tabla 19 Evaluación de métodos holísticos en el reconocimiento de rostros

La Tabla 19 nos muestra que en apariencia el mejor método, para realizar el reconocimiento de rostros cuando se posee una base de datos de al menos 9 muestras de la misma persona, es el método de Fisherfaces con parámetros de 17 clases de segmentación de datos y un umbral de 1250 para identificar los datos que serán asumidos como similares dentro de esas clases. Ahora bien, en este punto resulta necesario realizar algunas aclaraciones:

- En primera instancia la experimentación fue realizada con la misma base de rostros tanto para los algoritmos de Eigenfaces como también para los algoritmos de Fisherfaces, obteniendo los datos que están presentados en la Tabla 20.

Resultado	EIGENFACES						FISHERFACES					
	80/5000	80/2500	80/1250	17/5000	17/2500	17/1250	17/5000	17/1250	10/5000	10/1250	5/5000	5/1250
NO	0,00%	52,94%	94,12%	0,00%	14,71%	79,41%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%
FP	26,47%	1,47%	0,00%	27,94%	19,12%	0,00%	64,71%	64,71%	63,24%	63,24%	76,47%	76,47%
SI	73,53%	45,59%	5,88%	72,06%	66,18%	20,59%	35,29%	35,29%	36,76%	36,76%	23,53%	23,53%

Tabla 20 Evaluación de primera experimentación con métodos holísticos

Los resultados mostrados en la Tabla 20 obviamente no son muy alentadores en lo referente al algoritmo de Fisherfaces. Ante estos inusuales datos se realizó varias experimentaciones más siempre en la búsqueda de que los porcentajes subieran, resultando toda prueba infructuosa si se trabajaba con una sola base de datos con la cual se entrenaba el algoritmo con todos los rostros a reconocer. El desempeño del algoritmo mejora ostensiblemente cuando la base de rostros con la cual se entrena el algoritmo es pequeña, de esta manera se optó por dividir la base de datos en 4 partes.

- Al tener una sub-bases de datos se logra realizar una mejor discriminación entre las clases que componen a cada rostro y, por supuesto, manejar con menores umbrales los valores bajo los cuales se reconocerían los puntos integrantes de esas clases.
- El método de Fisherfaces posee un mejor desempeño para el reconocimiento, siempre y cuando no existan demasiados rostros para reconocer, al tener varios rostros, las clases de cada uno de los individuos pueden empezar a ser similares lo cual dificulta el reconocimiento. A fin de evitar estos falsos positivos derivados de clases muy cercanas entre los rostros, es conveniente bajar el nivel del umbral, lógicamente al bajar este valor, lo que se obtiene es un

algoritmo mucho más restrictivo en lo concerniente a la clasificación y por tanto el mismo empieza a dejar de reconocer a los individuos.

- Al realizar la experimentación, en el método de Fisherfaces se pudo deducir que existe un matrimonio entre: la cantidad de rostros y sus respectivas fotografías para entrenamiento, el valor de clases con las cuales se entrenará al algoritmo y el valor de umbral para el reconocimiento de los valores pertenecientes a cada clase.

A continuación en las Imágenes 7, 8 y 9 se presentan algunos de los individuos y sus fotografías de entrenamiento

Imagen 7 Base de entrenamiento del sujeto 94 (Autor)

De la Imagen 7 se deduce que la base de entrenamiento no mantiene una homogeniedad en el ángulo de visión ni tampoco en la oclusión por elementos, en este caso anteojos.

Uno de los sujetos que no pudo ser identificado con todos los algoritmos holísticos fue el 88, cuya base de fotografías de entrenamiento se la expone en la Imagen 8.

Imagen 8 Base de entrenamiento del sujeto 84(Autor)

Un tercer sujeto que por el contrario si pudo ser identificado fue el 92, la base de fotos para el entrenamiento de los algoritmos holísticos está presentada en la Imagen 9, mientras que en la imagen 10 muestra el reconocimiento que se logró del individuo.

Imagen 9 Base de entrenamiento del sujeto 92 (Autor)

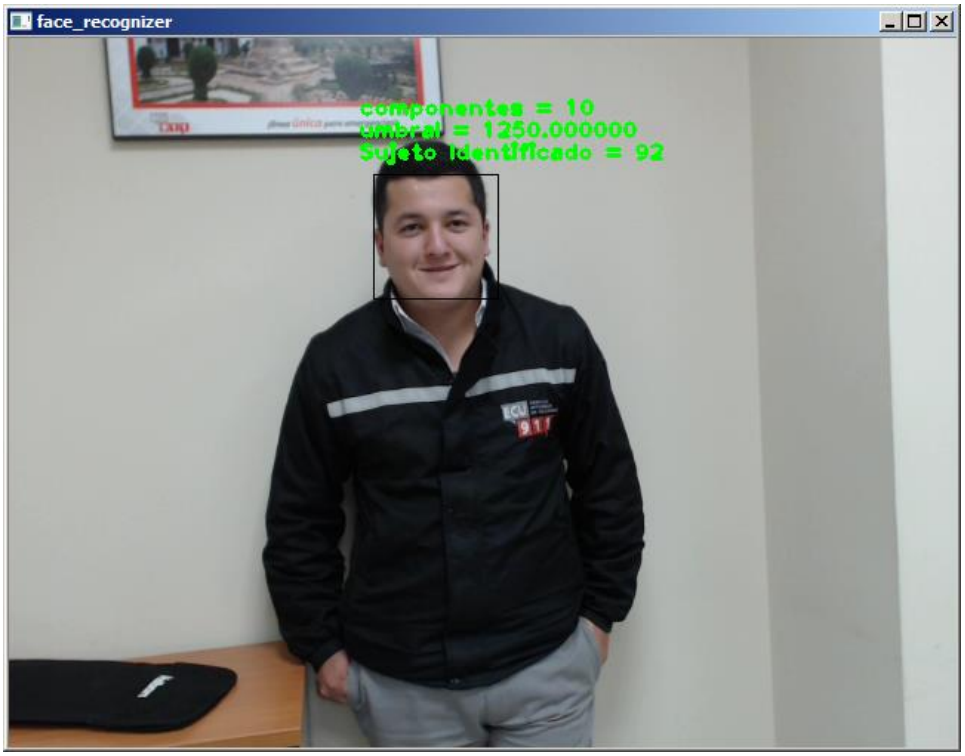

Imagen 10 Reconocimiento del sujeto 92 (Autor)

Entre las imágenes 7, 8 y 9 se aprecia claramente que no existe mayor diferencia en las bases de entrenamiento tanto en el ángulo con el que fueron captadas las imágenes de entrenamiento, como tampoco en las posibles oclusiones.

4.3.2. Evaluación métodos por Descriptores.

Los métodos por descriptores han demostrado poseer niveles mucho más altos de reconocimiento de un rostro, llegando a tener un 98.55% de efectividad al momento de reconocer a un individuo con el algoritmo de SURF.

	SURF			SIFT		
	17	*1*	*1500*	*17*	*1*	*1500*
NO	1,45%	1,45%	2,90%	20,29%	20,29%	20,29%
FP	1,45%	0,00%	0,00%	10,14%	8,70%	7,25%
SI	97,10%	98,55%	97,10%	69,57%	71,01%	72,46%
Promedio de descriptores encontrados	15,32	13,70	4,83	10,19	9,33	8,02
Superior al promedio	30,30%	29,85%	37,88%	40,43%	35,42%	30,61%
Inferior al promedio	69,70%	70,15%	62,12%	59,57%	64,58%	69,39%

Tabla 21 Evaluación de métodos por descriptores en el reconocimiento de rostros

Como se puede apreciar en la Tabla 21, el mayor porcentaje de reconocimiento fue obtenido con el algoritmo SURF con un valor del Hessiano 1.

A pesar de obtener el mayor porcentaje de efectividad, al momento de evidenciar la cantidad de descriptores encontrados, se distingue que la mayor cantidad de ellos es inferida cuando el Hessiano mínimo es 17 con 15 descriptores como promedio en el reconocimiento de un rostro, el valor de un Hessiano de 1500 provoca el más bajo desempeño del algoritmo con 5 descriptores en promedio al momento de reconocer un rostro.

Por su parte el algoritmo SIFT que posee 40,43% de reconocimiento con una cantidad de descriptores sobre el promedio de descriptores encontrados en

todas las pruebas, sin embargo, el porcentaje de reconocimiento llega a 69,57%; siendo el mismo inferior al 98,55% del mejor desempeño de SURF.

Los sujetos que en cualquiera de los métodos y con cualquier parámetro de reconocimiento fueron detectados con un número superior al número promedio de descriptores son los siguientes: 63, 64, 74, 100, 105, 118 y 129. En las imágenes 11, 12, 13 y 14 están mostrados los cuatro primeros individuos listados.

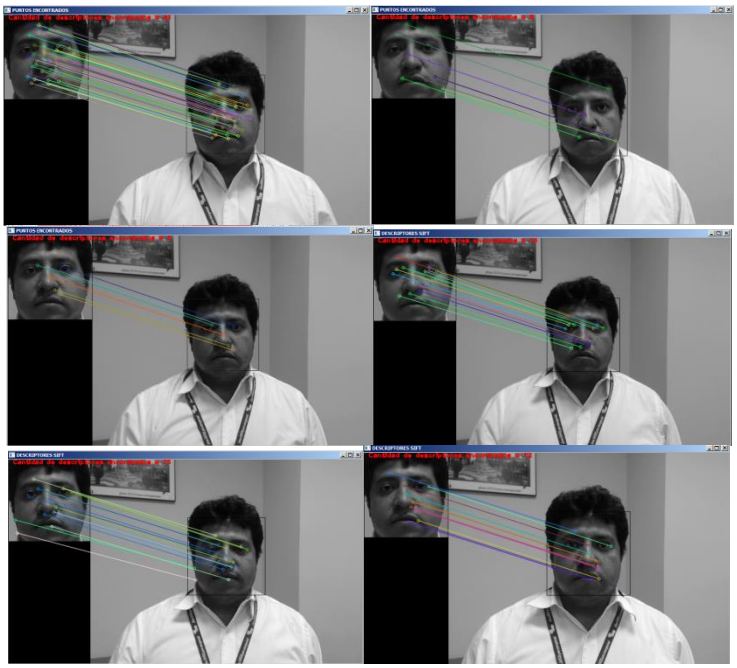

Imagen 11 Resultados de reconocimiento del sujeto 63 con métodos por descriptores. (Autor)

Imagen 12 Resultados de reconocimiento del sujeto 64 con métodos por descriptores (Autor)

Imagen 13 Resultados de reconocimiento del sujeto 74 con métodos por descriptores(Autor)

Imagen 14 Resultados de reconocimiento del sujeto 100 con métodos por descriptores (Autor)

Los sujetos que fueron reconocidos por todos los algoritmos y con cualquier valor del Hessiano mínimo, pero con un número de descriptores siempre por debajo del promedio fueron: 77, 81, 89, 97, 113, 122, 134, 135, 137, 138, 142 y 143. De estos, en las Imágenes 15, 16 y 17 se presentan las experimentaciones de los sujetos 77, 81 y 134.

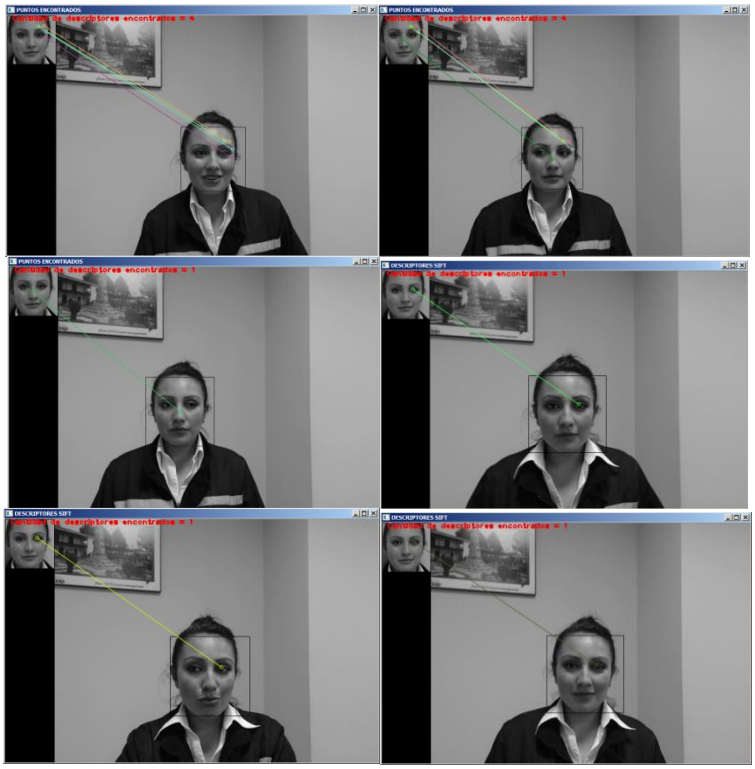

Imagen 15 Resultados de reconocimiento del sujeto 77 con métodos por descriptores(Autor)

Imagen 16 Resultados de reconocimiento del sujeto 81 con métodos por descriptores(Autor)

Imagen 17 Resultados de reconocimiento del sujeto 134 con métodos por descriptores(Autor)

Como se puede apreciar los sujetos que son reconocidos con un mayor número de descriptores son personas con la tez obscura, mientras que los sujetos con valores siempre debajo del promedio en lo que refiere a cantidad de descriptores encontrados son personas con la tez clara. Con la intensión de mejorar este desempeño con los rostros claros se intentó realizar un ecualización previa de las imágenes pero los resultados solamente empeoraron la calidad del reconocimiento.

CAPÍTULO 5

CONCLUSIONES Y RECOMENDACIONES

CONCLUSIONES

Una vez realizada la experimentación y basándose en la teoría de cada uno de los métodos utilizados en el presente trabajo, se plantean las siguientes apreciaciones a manera de conclusiones:

- Los métodos holísticos representan una alternativa positiva en lo referente al reconocimiento facial siempre y cuando se puedan obtener varias muestras de una misma persona. Quizá por su básica forma de realizar un rostro promedio y posterior a ello comparar el mismo contra un sujeto que no necesariamente podrá ser captado en la misma posición ni circunstancias similares, puede tener colocadas una gafas, haberse dejado crecer la barba, en el caso de los varones, estos algoritmos no podrán ser 100% confiables al momento de deducir si se trata de una misma persona.
- El método de Fisherfaces, puede mejorar sustancialmente su desempeño si trabaja con bases de rostros de pocos individuos a diferencia de lo que ocurre con el método de Eigenfaces que no interesa si trabaja con bases de rostros de pocos individuos o de varios puesto que su desempeño no varía en referencia a este particular.
- Los métodos holísticos necesitan para ir mejorando progresivamente en su desempeño, cada vez una base de información mayor, cuestión complicada por la cantidad de almacenamiento que se deberá tener para satisfacer esta necesidad.

- Los métodos por descriptores se han mostrado mucho más precisos al momento de obtener resultados de reconocimiento de un sujeto determinado siendo incluso invariantes a ciertas condiciones particulares pero bien definidas como por ejemplo la angulación con la cual es captada la imagen de muestra y a posterior la comparación dentro del stream de vídeo.
- Los métodos por descriptores pueden ser mejorados sustantivamente en tareas de identificación de rostros si se logra poseer una iluminación constante y directa con la cual sean capturados tanto la imagen de muestra como también el stream de vídeo que servirá para realizar el reconocimiento de los sujetos.
- Los métodos por descriptores poseen debilidad ante el manejo de la iluminación, esto está obviamente expuesto en que los rostros de tez clara fueron de mayor dificultad al momento del reconocimiento, mientras que rostros de piel más obscura fueron reconocidos con mayor facilidad. Esto se debe al principio que colores obscuros absorben más la luz mientras que los colores claros la refractan ocasionando de esta forma la pérdida de características al rechazar la luz provocando un efecto espejo.

RECOMENDACIONES

En el desarrollo del presente trabajo y luego de plantear las conclusiones del estudio se realizan las siguientes recomendaciones:

- Es indispensable que las cámaras de seguridad que vayan a ser utilizadas para el reconocimiento de rostros sean ubicadas a una altura no superior a los 2 metros desde el piso siendo una buena práctica que la altura sea siempre igual a la estatura promedio de la población a ser discriminada, esto evitará que las angulaciones en la captación del stream de video representen un problema y al mismo tiempo se tendrá la mayor cantidad

de elementos discriminatorios al captar el rostro directamente. Ejemplos de lo redactado pueden ser apreciados en la Imagen 18 y 19, en la primera se muestra la robustez a pesar de la angulación y la oclusión parcial, mientras que en la figura 21 se puede observar como si la imagen mantiene casi igual altura y angulación, la cantidad de descriptores es elevada.

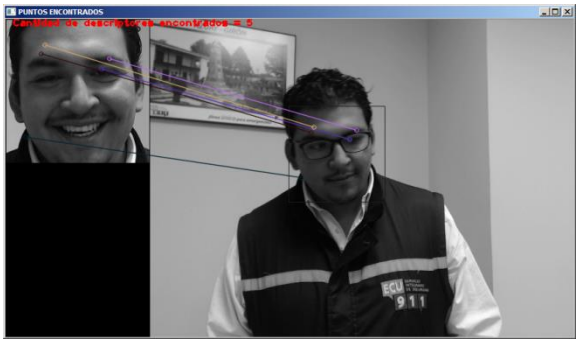

Imagen 18 Reconocimiento a pesar de oclusión parcial y distinta angulación del rostro (Autor)

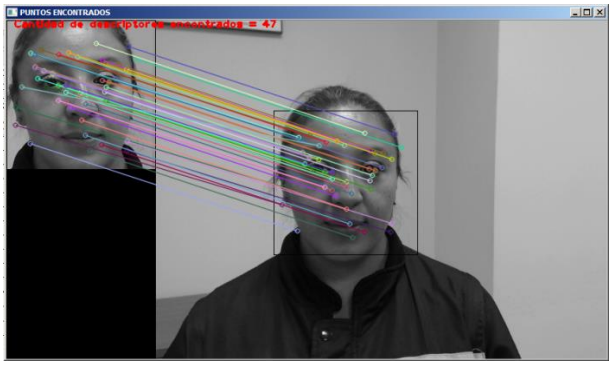

Imagen 19 Mejora en el reconocimiento cuando no existe oclusión y la muestra y el vídeo poseen similares ángulos de captación. (Autor)

- Para un mejor desempeño de los algoritmos se deberá acompañar a la cámara de vigilancia con una fuente emisora de luz. Esto es necesario para garantizar de alguna manera, condiciones mínimas de estabilidad en la iluminación de los rostros. Esta pequeña ayuda puede ser de gran valía al

momento de obtener resultados satisfactorios en el reconocimiento de rostros.

BIBLIOGRAFÍA

Abdi, H. (2007). *The University of Texas at Dallas.* Retrieved 06 02, 2014, from http://www.utd.edu/~Herve/Abdi-SVD2007-pretty.pdf

Agrawal, M., Konolige, K., & Blas, M. R. (2008). Center surround extremas for realtime feature detection and matching. *Computer Vision-ECCV*, pp. 102-115.

Alonso, S. K. (2009). *eMathTeacher*. Retrieved 02 21, 2011, from eMathTeacher: http://www.dma.fi.upm.es/research/FundMatSoftComputing/fuzzyinf/mamdani1.htm

Amazon. (2011). *www.amazon.com*. Retrieved 03 12, 2011, from www.amazon.com: http://www.amazon.com/gp/aw/d.html?is=200&a=B000VTQ3LU&in=3

Aracena-Pizarro, D., & Daneri-Alvarado, N. (2013). Detección de puntos claves mediante SIFT paralelizado en GPU. *Ingeniare. Revista chilena de ingeniería, 21*(3), 438-447.

Brut, Mihaela; et al. (2011, January). A distributed architecture for flexible multimedia management and retrieval. *Database and Expert Systems Applications. Springer Berlin Heidelberg*, pp. 249-263.

Buntine, W., & Jakulin, A. (2004). Applying discrete PCA in data analysis. *Proceeding of the 20th conference on Uncertainty in artificial intelligence*, 59-66.

Burt, P. J., & Adelson, E. H. (1983). The Laplacian pyramid as a compact image code. *Communications, IEEE Transactions on, 31*(4), 532-540.

Cataluña, U. P. (2008). *www.tdr.cesca.es*. Retrieved 03 04, 2011, from www.tdr.cesca.es: http://www.tdr.cesca.es/TESIS_UPC/AVAILABLE/TDX-0207105-105056//04Rpp04de11.pdf

Computer Robot Vision. (n.d.). Retrieved 06 13, 2014, from http://www.computerrobotvision.org/2010/tutorial_day/tam_surf_rev3.pdf

Crespo, C., & Ochoa, D. (2009). *Evaluación Cuantitativa de la Influencia de los Espacios de Color para la Detección Automática de Células.* Guayaquil, Ecuador: Escuela Superior Politécnica del Litoral.

De la Torre, F., & Black, M. J. (2001). Robust principal component analysis forcomputer vision. *Computer Vision, 1*(Eighth IEEE International Conference), 362-369.

Delac, K., Grgic, M., & Bartlett, M. S. (2008). *Recent Advances in Face Recognition.* Croatia: In-teh.

Ebrahimi, M., & Mayol-Cuevas, W. W. (2009). SUSurE: Speeded Up Surround Extrema Feature Detector and Descriptor for Realtime Applications. *Computer Vision and Pattern Recognition Workshops*. Bristol, United Kingdom.

ESCET. (2014, 08 04). Retrieved 08 04, 2014, from http://www.escet.urjc.es/~visiona/tema5.pdf

Flores, P., & Braun, J. (2011). Algoritmo SIFT: fundamento teórico.

Gascueña, J. M., & Fernández-Caballero, A. (2011). On the use of agent techonology in intelligent, multisensory and distributed surveillance. *The Knowledge Engineering Review, 26*(02), 192-208.

Gottumukkal, R., & Asari, V. K. (2004). An improved face recognition technique based on modular PCA approach. In *Pattern Recognition Letters 25* (pp. 429-436). ELSEVIER.

Herbert, B., Tuytelaars, T., & Van Gool, L. (2006). SURF: Speeded Up Robust Features. In *Computer Vision-ECCV* (pp. 404-417). Berlin: Springer.

Hernán González, M. S. (2006). *www.catic.unab.edu.co*. Retrieved 02 28, 2011, from www.catic.unab.edu.co: http://catic.unab.edu.co/2congresomecatronica/images/docum/14.pdf

IFR. (2011). *www.cfievalladolid2.net.* Retrieved Marzo 01, 2011, from www.cfievalladolid2.net: http://cfievalladolid2.net/tecno/cyr_01/robotica/industrial.htm

Isidro Zabalza, J. R. (2007, 10 23). *www.imem.unavarra.es.* Retrieved 02 28, 2011, from www.imem.unavarra.es: http://www.imem.unavarra.es/isidro/articles/Zabalza-Cuzco.pdf

Jafri, R., & Arabnia, H. R. (2009, Junio). A Survey of Face Recognition Techniques. *Journal of Information Processing Systems*, pp. Vol 5, No. 2.

Jiménez, A. R., Jain, A. K., Ceres, R., & Pons, J. L. (1999). Automatic fruit recognition: a survey and new results using Range/Attenuation images. *Pattern recognition, 32*(10), 1719-1736.

Justo, F. A., & Aguirre, I. (2009). Creación de una Herramienta que permita mover el cursor de un computador a partir del movimiento ocular, utilizando técnicas de visión artificial. *Seventh LACCEI Latin American and Caribbean Conference for Engineering and Technology.* San Cristobal, Venezuela.

Kandil, H; Atwan, A;. (2012). A Comparative Study between SIFT-Particle and SURF-Particle Video Tracking Algorithms. *International Journal of Signal Processing, Image Processing & Pattern Recognition., 5*(3), 111-122.

Ke, Yan; Sukthankar, Rahul;. (2004). PCA-SIFT: A more distinctive representation for local image descriptors. *Computer Vision and Pattern Recognition, 2*, II-506-II-513.

Kim, T.-K., Stenger, B., Kittler, J., & Cipolla, R. (2011). Incremental linear discriminant analysis using sufficient spanning sets and its applications. *International Journal of Computer Vision, 91*(2), 216-232.

Kirby, M., & Sirovich, L. (1990, Enero). Application of the Karhunen-Loève Procedure for the Characterization of Human Faces. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, p. Vol.12 No. 1.

León, J. A. (2009, 05 28). Diseño, análisis y construcción de un robot paralelo traslacional. *Diseño, análisis y construcción de un robot paralelo traslacional*. Querétaro, México.

Lindeberg, T. (n.d.). *Scholarpedia.org*. Retrieved 06 03, 2014, from http://www.scholarpedia.org/article/Scale_Invariant_Feature_Transform#Scale-invariant_interest_points_from_scale-space_extrema

Lowe, D. G. (1999). Object recognition from local scale-invariant features. *International Conference on Computer Vision , 2*, 1150-1157.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision, 60*(2), 91-110.

Madrid, U. P. (2007). *www.sadesisam.etsii.upm.es*. Retrieved 02 26, 2011, from www.sadesisam.etsii.upm.es: http://sadesisam.etsii.upm.es/trepa/es/project.asp

Mikolajczyk, K., & Schmid, C. (2005). A performance evaluation of local descriptors. *Pattern Analysis and Machine Intelligence, IEEE Transactions, 27*(10), 1615-1630.

Ogata, K. (1997). *Ingeniería de Control Moderno.* Minnesota: Prentice-Hall.

Pajares, G., & De la Cruz, J. M. (2001). *Visión por computador. Imágenes Digitales y Aplicaciones.* Ra-Ma.

Palitroquez, P. (2008, 12 27). *www.picmania.garcia-cuervo.net*. Retrieved 03 12, 2011, from www.picmania.garcia-cuervo.net: http://picmania.garcia-cuervo.net/invitados_primer18f4550.php

Pandya, J. M., Rathod, D., & Jadav, J. J. (2013, Enero-Febrero). A Survey of Face Recognition approach. *IJERA*, pp. 632-635.

Pang, Y., Li, W., Yuan, Y., & Pan, J. (2012). Fully affine invariant SURF for image matching. *Neurocomputing, 85*, 6-10.

Pereira, J. F., Cavalcanti, G., & Ren, T. I. (2009). Modular Image Principal Component Analysis for Face Recognition. *Conference on Neural Networks.* Atlanta-Georgia.

Rafael Aracil, R. S. (2006, 01). *arvc.umh.es.* Retrieved 03 03, 2011, from arvc.umh.es: http://arvc.umh.es/documentos/articulos/RIAI%202006.pdf

Reznik, L. (1997). *Fuzzy Controllers.* Melbourne, Australia: NEWNES.

Robotmania. (2010, 11 18). *www.robotionary.com*. Retrieved 02 26, 2011, from www.robotionary.com: http://robotionary.com/robotics/world%E2%80%99s-fastest-robot-quattro.php

Rougier, C., Meunier, J., St-Arnaud, A., & Rousseau, J. (2011). Robust Video Surveillance for Fall Detection Based on Human Shape Deformation. *Circuits and Systems for Video Technology, IEEE Transactions, 21*(5), 611-622.

Schiele, B., & Crowley, J. L. (2000). Recognition without correspondence using multidimensional receptive field histograms. *International Journal of Computer Vision, 36*(1), 31-50.

Shah, J., Sharif, M., Raza, M., & Azeem, A. (2011). A survey: Linear and Nonlinear PCA Based Face Recognition Techniques. *COMSATS Institute of Information Technology.* Paquistan.

Silva, L. A. (2005). *www.oa.upm.es*. Retrieved 02 29, 2011, from www.oa.upm.es: http://oa.upm.es/378/1/luis_angel_silva.pdf

Sirovich, L., & Kirby, M. (1987). Low-dimensional procedure for the characterization of human faces. *Division of Applied Mathematics, Brown University*. Rhode Island.

Sonker, V. P., & Behera, M. (2012). *Biometric Face Recognition System using SURF Based Approach.* Rourkela: Computer Science and Engineering National Institute of Technology.

Tao, Y., Skubic, M., Han, T., Xia, Y., & Chi, X. (2010). Performance Evaluation of SIFT-Based Descriptors for Object Recognition. *Proceedings of the International MultiConference of Enginners and Computer Scientists, 2*.

The Magazine of Record for the Embedded computing industry. (n.d.). Retrieved 05 21, 2014, from http://rtcmagazine.com/articles/view/102184

Tsai, R. Y. (1987). A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses. *Robotics and Automation, IEEE Journal, 3*(4), 323-344.

Turk, M. (2013, Octubre). Over Twenty Years of Eigenfaces. *ACM Trnas. Multimedia Comput.*(9), p. Article 45.

Valera, M., & Velastin, S. (2005). Intelligent distributed surveillance systems: a review. *Vision, Image and Signal Processing, IEE Proceedings, 152*(2), 192-204.

Varbuchta, R. (2004). *www.capek.misto.cz.* Retrieved 02 28, 2011, from www.capek.misto.cz: http://capek.misto.cz/english/robot.html

Vázquez Rodriguez, C. A., & García Tovar, L. (2013). Artificial Vision in 3D perspective. For object detection on planes, using points clouds. *International Journal of Combinatiorial Optimization Problems and Informatics.* México.

Velasco, I. H. (2006). *www.docentes.unal.edu.co*. Retrieved 02 21, 2011, from www.docentes.unal.edu.co: http://www.docentes.unal.edu.co/hfvelascop/docs/OTROS/LOGICADIFUSA/5%20An%E1lisis%20de%20Controladores%20Difusos.pdf

Viola, P., & Jones, M. (2001, February). Robust Real-time Object Detection. *Journal of Computer Vision, 4*, 34-47.

Wang, X. (2013). Intelligent multi-camera video surveillance: A review. *Pattern recognition letters*, *34*(1), 3-19.

Wikipedia. (2011, 02 10). *es.wikipedia.org.* Retrieved 03 03, 2011, from es.wikipedia.org: http://es.wikipedia.org/wiki/Karel_%C4%8Capek

Yongzhong, L., Jingli, Z., & Shengsheng, Y. (2003). A survey of Face Detection, Extraction and Recognition. *Computing and Informatics*, *22*.

Zhao, H., & Yuen, P. C. (February 2008). Incremental Linear Discriminant Analysis for Face Recognition. *IEEE Transactions on Systems, Man and Cybernetics*, *38*(1), 210-221.