ISSN: 1692-7257 - Volumen X – Número XX - 20XX

Revista Colombiana de Tecnologías de Avanzada

Recibido: 18 de enero de 2015
Aceptado: 17 de abril de 2015

RECONOCIMIENTO FACIAL BASADO EN EIGENFACES, LBHP Y FISHERFACES EN LA BEAGLEBOARD-xM

FACIAL RECOGNITION BASED ON EIGENFACES, LBPH AND FISHERFACES IN THE BEAGLEBOARD-xM

MSc. Carlos H. Esparza Franco, Ing. Christian Tarazona Ospina.
Ing. Esdras E. Sanabria Cuevas, MSc. Daniel A. Velazco Capacho.

Unidades Tecnológicas de Santander - UTS, Facultad de Ciencias Naturales e Ingenierías, Grupo de Investigación en Control Avanzado GICAV.
Calle de los Estudiantes, No. 9-82, Bucaramanga. Tel.: (+577) 6917700, Ext. 2008.
E-mail: {gicav, dvelazco}@correo.uts.edu.co,{christianxab,esanabriac}@gmail.com

Resumen: El presente trabajo consiste en la implementación de un sistema de detección de rostro aplicando procesamiento de imágenes, aplicando características basados en eigenfaces, Histogramas de patrones locales binarios LBPH y discriminantes Fisher faciales, sobre el dispositivo embebido BeagleBoard-xM. Para desarrollar este trabajo se empelaron librerías de OpenCV, Java, CMake y sistemas operativos como Debian y Ubuntu sobre la *BeagleBoard*. La validación del sistema determinó que el mejor resultado se obtiene con la técnica de *EigenFaces* ya que presento menor error de clasificación y 0% de errores de falsos positivos.

Palabras clave: BeagleBoard-xM, Eigenfaces, Fisherfaces, LBPH, OpenCV, Reconocimiento Facial.

Abstract: The main objective of this work is to implement a face detection system by using image processing, applying features based in eigenfaces, local binary pattern Histograms LBPH and FisherFaces, in the BeagleBoard-xM embedded hardware. To develop this work it was used OpenCV libraries, Java, CMake and Operation systems like Debian and Ubuntu in the BeagleBoard. The system validation shows that the best result was obtained with the EigenFaces technique giving the minor error and 0% false positive error.

Keywords: BeagleBoard-xM, EigenFaces, FisherFaces, LBPH, OpenCV, Facial recognition.

1. INTRODUCCIÓN

Los sistemas de reconocimiento de rostros son un problema que aún es tema de investigación, debido a que gran parte del problema se debe a los factores que pueden afectar el sistema al momento de efectuar el reconocimiento. Algunos de estos factores que dificultan su proceso, como son las oclusiones que evita un buen reconocimiento, ya sean por gestos que realice las personas, elementos que cubran la cara, iluminación, distancia, envejecimiento, entre otros (Ding, Xu, & Tao, 2015; Hassner, Harel, Paz, & Enbar, 2015; Pizarro Jara, 2011). El problema principal de este proyecto es implementar un sistema de reconocimiento facial, sobre rostros frontales, sobre un dispositivo embebido.

Este proyecto se desarrolló en una BeagleBoard-xM (BeagleBoard Group, 2014), la cual es una tarjeta de bajo costo y tamaño reducido.

Para resolver el problema se instaló el sistema operativo *Debian*, las librerías de OpenCV (*Open source Computer Vision library*), el lenguaje Java usando el JDK (*Java Development Kit*), el escritorio LXDE (*Lightweight X11 Desktop Environment*), y el IDE (*Integrated Development Environment*) de Eclipse en el cual se desarrolló el algoritmo del reconocimiento de rostros.

Como conjunto de imágenes de prueba se empleó la base de datos Cohn-Kanade (Kanade, Cohn, & Tian, 2000) desarrollada en *Carnegie Mellon University* con el propósito de tener un conjunto universal de datos referente para investigaciones a nivel mundial. Aunque esta base de datos se diseñó con la finalidad de realizar análisis de las expresiones faciales, ya que consiste en conjuntos de datos con personas realizando las expresiones faciales determinadas por Ekman (Ekman, 1993), la gran cantidad de información permite en este trabajo evaluar a las mismas personas con diferentes expresiones para validar el sistema.

El sistema de reconocimiento se realizó sobre el sistema embebido *BeagleBoard-xM* para comprobar su capacidad en procesamiento de imágenes para el reconocimiento de rostros. La validación empleó tres técnicas que son: Eigenfaces, LBPH (*Local Binary Patterns Histograms*) y *Fisherfaces*.

Para ejecutar correctamente los algoritmos de *Eigenfaces* y *Fisherfaces* se requirió implementar una técnica de reducción de dimensionalidad de los datos generados por las imágenes, análisis de componentes principales PCA (*Principal Component Analysis*) para el caso de *eigenfaces* y fisherfaces (Shah, Sharif, Raza, & Azeem, 2013) y análisis de discriminantes lineales LDA (*Linear Discriminat Analysis*) para el caso de *Fisherfaces*.

Posteriormente, mediante la distancia euclidiana se realiza la comparación de las características de los vectores de las imágenes de la base de datos con los vectores del rostro a reconocer.

En LBPH se realiza un procedimiento similar haciendo la comparación de datos pero estos son pixeles que se crean a partir de una matriz generada de 8x8, tomando un *pixel* central que se compara con sus vecinos, obteniendo un numero decimal que es tomado por el histograma para formar la descripción y obtener el reconocimiento facial.

2. MÉTODOS DE RECONOCIMIENTO DE ROSTROS

2.1 *EigenFaces*

Es una técnica que permite determinar, mediante la ortogonalidad dimensional, qué vectores ofrecen más información a un conjunto de datos de dimensión N. No obstante, la información N-dimensional obtenida con *Eigenfaces* contiene datos redundantes que solo ocasionan que un sistema de clasificación tenga un alto costo computacional. Para minimizar esto, se aplica análisis de componentes principales (PCA) (Gottumukkal & Asari, 2003), el cual toma una cantidad menor de los vectores entregados por las imágenes de la base de datos pero con información necesaria para la reconstrucción de los rostros de las imágenes ingresadas. Gracias a esto, se logra disminuir el costo computacional del procesamiento de datos.

El reconocimiento se realiza mediante la proyección de una imagen de un rostro en el subespacio formado por los *eigenfaces* y comparando su posición con la de los otros rostros conocidos. Los *eigenfaces* son un conjunto de vectores representados gráficamente, convirtiéndose en una especie de mapa de las variaciones entre imágenes. Estos vectores son el resultado de la aplicación de PCA a la matriz de covarianza de un conjunto de imágenes de rostros los cuales son denominados *eigenvectores*, siendo una imagen tratada como un vector en un espacio multidimensional. A su vez, cada cara individual puede ser representada exactamente en términos de una combinación lineal de las *eigenfaces*, la cual puede ser aproximada usando solamente los "mejores" *eigenfaces* que son las que tienen mayores eigenvalores. Los M mejores *eigenfaces* conforman un subespacio M-dimensional de las caras de todas las posibles imágenes.

Este método realiza las siguientes operaciones y pasos:

- Adquisición de una serie de imágenes de caras iniciales.
- Cálculo de los *eigenfaces* del conjunto de entrenamiento, almacenando únicamente las M dimensiones que correspondan con los eigenvalores mayores.
- Cálculo de un conjunto de pesos basados en la imagen de entrada y las M-*eigenfaces* mediante la proyección de la imagen de entrada sobre cada una de las *eigenfaces*.

- Determinar si la imagen de la cara pertenece o no al conjunto de entrenamiento, por medio de la distancia euclídea.

Para la utilización de esta técnica se define una imagen I(*x*,*y*) como una matriz bidimensional de N filas y N columnas, cuyos valores de intensidad varíen entre 0 y 255 (8 bits), correspondientes a una imagen en escala de grises. Estos vectores definen el sub-espacio de imágenes de caras. Cada vector es de longitud N_2, que describe a una imagen de tamaño *N*x*N*, y es una combinación lineal de la imagen de una cara original (Kshirsagar, Baviskar, & Gaikwad, 2011).

El conjunto de imágenes de caras para el entrenamiento es $\Gamma_1, \Gamma_2, \Gamma_3, \cdots \Gamma_M$. Después de haber obtenido su conjunto se tiene la cara promedio dada por la ecuación 1:

$$\Psi = \frac{1}{M}\sum_{n=1}^{M}\Gamma_n \qquad (1)$$

Luego cada cara difiere entre el rostro de entrada y la media del mismo según el vector de la ecu. 2.

$$\Phi_i = \Gamma_i - \Psi \qquad (2)$$

Este conjunto de grandes vectores está sujeto al análisis de componentes principales, el cual busca una serie de M vectores ortonormales u_n que mejor describan la distribución de los datos. El *k*-ésimo vector u_k, es elegido tal como se presenta en la ecuación 3:

$$l_k = \frac{1}{M}\sum_{n=1}^{M}\left(u_k^T\Phi_n\right)^2 \qquad (3)$$

Siendo el máximo sujeto como se representa en la ecuación 4, para valores entre 1 y 0.

$$u_l^T u_k = d_k \begin{cases} 1, & \text{para } l = k \\ 0, & \text{otro valor} \end{cases} \qquad (4)$$

Los vectores u_k y los vectores l_k, son los eigenvectores y los eigenvalores respectivamente de la matriz de covarianza en la ecuación 5.

$$C = \frac{1}{M}\sum_{n=1}^{M}\Phi_n\Phi_n^T = AA^T \qquad (5)$$

Donde A es la matriz $A = [\Phi_1, \Phi_2, \cdots, \Phi_n]$, y la matriz C es de dimensión de N_2xN_2.

Si el número de puntos de datos en el espacio de imágenes es menor que la dimensión del espacio ($M<N_2$), habrá solamente *M*-1 valores menores que N_2 eigenvectores significativos (los restantes eigenvectores estarán asociados a los eigenvalores de cero). El sistema puede resolverse para los eigenvectores N2 dimensionales primero para los eigenvectores de una matriz *M*x*M* y luego realizando las apropiadas combinaciones lineales de las imágenes de las caras Φ_i. Considerando los eigenvectores V_i de AA^T, tales como la ecuación 6.

$$AA^T V_i = m_i V_i \qquad (6)$$

Pre multiplicando ambos miembros por A, se tiene en la ecuación 7.

$$AA^T AV_i = m_i AV_i \qquad (7)$$

Donde puede observarse que AV_i son los eigenvectores de $C = AA^T$. Con esto se construye la matriz $L=AA^T$ donde $L_{mn} = \Phi_m\Phi_m$ y se encuentran los *M* eigenvectores, V_k de *L*. Estos vectores determinan la combinación lineal de las *M* imágenes del conjunto de caras de entrenamiento para formar las *eigenfaces* u_l, como en la ecu. 8.

$$u_l = \sum_{k=1}^{M} V_{lk}\Phi_k \qquad i = 1, \cdots, M \qquad (8)$$

Primero se compara la imagen de entrada (*r*), (proyectada dentro del espacio de caras) con la imagen media y se multiplica su diferencia con cada vector propio de la matriz L como la ecu. 9.

$$w_k = u_k^T\left(\Gamma - \Psi\right) \qquad \text{para } k = 1, \cdots, M \qquad (9)$$

Con esto se obtiene un conjunto de pesos que conforman el vector $\Omega T = [w_1, w_2, \cdots, w_M]$ que describe la contribución de cada *eigenface* en la representación de la imagen de entrada. Por tanto, este vector puede ser usado como modelo estándar dentro de un algoritmo de reconocimiento sin más que evaluar una distancia entre vectores y mediante umbralización, determinar si una imagen de una cara puede considerarse perteneciente al espacio de caras de entrenamiento o no. Este valor será el que minimiza la distancia euclidiana en la ecuación 10.

$$e_k^2 = \left\|\Omega - \Omega_k\right\| \qquad (10)$$

Donde Ω_k es el vector que describe la *k*-ésima cara (Reyes López, 2005; Turk & Pentland, 1991). El resultado final después de analizar una imagen aplicando eigenfaces es el que se muestra en la figura 1.

Fig. 1. Análisis de Eigenfaces

2.2 LBPH

El método de patrones binarios locales fue diseñado para la descripción de texturas (Silva, Esparza, & Mejía, 2012). El uso de descripciones locales en algunas regiones del rostro aportan más información que otras, por lo que los descriptores de texturas tienden a promediar la información que describen, lo cual no es conveniente al describir rostros puesto que mantener la información de las relaciones espaciales es importante (Alvarado & Fernández, 2012). Para formar la descripción global, la imagen del rostro es dividida en diferentes regiones, a las que se les aplica un histograma con el que se obtiene el operador LBPH que describe información independiente por región. Estas descripciones locales son entonces concatenadas para construir una descripción global del rostro (Álvarez, 2013).

El método de LBPH asigna etiquetas a cada uno de los pixeles de la imagen tomando en cuenta la distribución de los vecinos. Estos son los pasos que el LBPH realiza para su respectivo reconocimiento de imágenes.

- Una máscara de tamaño determinado (8x8), recorre la imagen de manera iterativa seleccionando cada vez un *pixel* central y sus vecinos.
- Este *pixel* central se compara con cada uno de sus vecinos de forma ordenada. Se asigna un 1 cada vez que el *pixel* central sea menor que el *pixel* comparado y un 0 en el caso contrario, como se muestra en la ecuación 11 y se visualiza en la figura 2.

$$LBP = \sum_{p=0}^{7} s\left(g_p - g_c\right)2^p \qquad (11)$$

$$s\left(x\right) = \begin{cases} 1, & \text{si } x \geq 0 \\ 0, & \text{otro valor.} \end{cases}$$

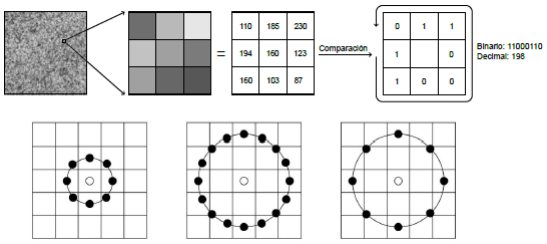

Fig. 2. Obtención de los parámetros LBP

- El número binario resultante se convierte en un número decimal que es contado en el histograma para formar la descripción. El histograma de las etiquetas de todos los pixeles es posteriormente utilizado como una descripción de la textura de la imagen, que se indica en la ecuación 12.

$$H_i = \sum_{xy} I\left[LBP\left(x,y\right) = i\right], \quad i = 0, \cdots, n-1 \qquad (12)$$

$$I\left(x\right)\begin{cases} 1, & \text{si } x \text{ es verdadero.} \\ 0, & \text{otro valor.} \end{cases}$$

2.3 *Fisherfaces*

Fisherface es una técnica de reconocimiento de rostros, que tiene en cuenta la luz y expresiones faciales. Este se encarga de clasificar y reducir las dimensiones de las caras usando el método FLD (*Discriminant Lineal Fisher*). En este sentido, el análisis discriminante de Fisher intenta proyectar los datos de manera que su nueva dispersión sea óptima para la clasificación. Mientras PCA busca los vectores que mejor describen los datos, LDA (*Discriminant Lineal Analysis*) busca los vectores que proporcionan mejor discriminación entre clases después de la proyección (Martínez & Kak, 2001). *Fisherfaces* realiza un LDA, donde se busca aprovechar la información disponible, sobre la clasificación de las imágenes de entrenamiento, para buscar una proyección que maximice la separación entre imágenes de diferentes personas (o clases) y minimice la distancia entre imágenes de una misma clase. Así logra concentrar las

imágenes mejorando, en forma importante, la tasa de reconocimiento (Belhumeur, Hespanha, & Kriegman, 1997).

Se define la matriz de varianza entre clases (imágenes de personas distintas) como en la ecuación 13 y la varianza dentro de la clase (imagines de una misma persona) como en la ecuación 14. Donde m_i corresponde a la imagen promedio de la clase X_i, $|X_i|$ al número de puntos dentro de la clase X_i y m es el promedio de todas las imágenes.

$$S_B = \sum_{i=1}^{c} |X_i| (m_i - m)(m_i - m)^T \quad (13)$$

$$S_W = \sum_{i=1}^{c} \sum_{X_k \in X_i} (X_k - m_i)(X_k - m_i)^T \quad (14)$$

Luego, en forma similar a PCA, se debe encontrar la matriz de proyección $W \in R^{(mxn)}$. Resolviendo el problema de optimización se obtiene la ecuación 15, donde se debe minimizar S_W y maximizar S_B de las ecuaciones 13 y 14. El resultado de dicho problema es una matriz con los vectores propios de $S_W S_B^{-1}$.

$$W_{opt} = \arg\max_W \left| \frac{W^T S_B W}{W^T S_W W} \right| \quad (15)$$

Para esto se debe reducir la dimensionalidad de las imágenes a no más de número de imágenes – número de clases. Luego se puede aplicar LDA para reducir a número de clases -1 elementos y agrupar las imágenes según la clase a la que pertenecen. Si la primera reducción se realiza utilizando *Eigenfaces*, al método se le llama *Fisherfaces*.

Así el método de *Fisherfaces* se puede separar en dos partes. Primero resolver las ecuaciones dadas en PCA, donde con S_B y S_W creadas a partir de las imágenes proyectadas sobre el subespacio de *eigenfaces*. En forma equivalente se puede resolver la ecuación 16 con W_{PCA} (Seo, 1998). El resultado final de aplicar *fisherfaces* a una imagen es el que se muestra en la figura 3.

Estos métodos han mostrado gran eficiencia y popularidad debido a la relativa simpleza. Sin embargo, la cantidad de cálculos y requerimientos de memoria son muy altos debido a las grandes dimensionalidades de las imágenes.

$$W_{opt} = \arg\max_W \left| \frac{W^T \cdot W_{PCA}^T \cdot S_B \cdot W_{PCA} \cdot W}{W^T \cdot W_{PCA}^T \cdot S_W \cdot W_{PCA} \cdot W} \right| \quad (16)$$

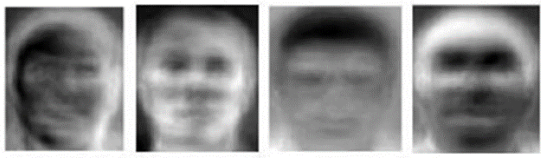

Fig. 3. Resultado de aplicar Fisherfaces

3. RESULTADOS

En esta sección se describen los resultados de las pruebas realizadas para evaluar la eficiencia y el tiempo del algoritmo tanto en un portátil como en la BeagleBoard-xM. En el portátil se utilizó el sistema operativo Windows 7 de 64 bits con procesador Intel Dual Core a 2.4GHz, en donde se instaló una máquina virtual para poner la imagen de Ubuntu 14.04, de 64 bits, RAM de 512 MB y un disco de 200 GB. Por otra parte, la BeagleBoard-xM cuenta con un procesador Cortex A8 1 GHz y una memoria SDRAM de 512 MB. La memoria ROM es de 4 GB MDDR a una velocidad de bus de 200 MHz.

La implementación de las técnicas de reconocimiento en la BeagleBoard-xM se realizó en dos etapas, una de entrenamiento y otra de validación, las cuales se describen en las figuras 4 y 5 respectivamente.

Para la etapa de entrenamiento se seleccionaron de forma aleatoria a 20 personas de la base de datos Cohn-Kanade. Para cada una de estas personas se seleccionó un total de siete fotos, construyendo así una base de datos de 140 imágenes. De cada una de la personas se seleccionaron cinco imágenes en pose neutral y otras dos realizando una expresión facial. El conjunto de entrenamiento para cada uno de los tres casos fue de 100 imágenes, cinco para cada individuo, y 40 para validación, 2 por cada individuo. Estas imágenes se seleccionaron aleatoriamente y se empleó correlación cruzada *ten-folded-validation* (Refaeilzadeh, Tang, & Liu, 2009) con el fin de obtener mayor número de resultados que corroboraran la información estadística.

Antes de iniciar los procesos de entrenamiento y validación se empleó el clasificador Haar para la la detección de rostros diseñado por Viola-Jones (Viola & Jones, 2004), lo cual genera un rectángulo sobre el rostro detectado. Se usa el rostro recortado y se normaliza al mismo tamaño usado en el entrenamiento, para luego aplicar cualquiera de las técnicas de reconocimiento, sea *Eigenfaces*, LBPH o *Fisherface*.

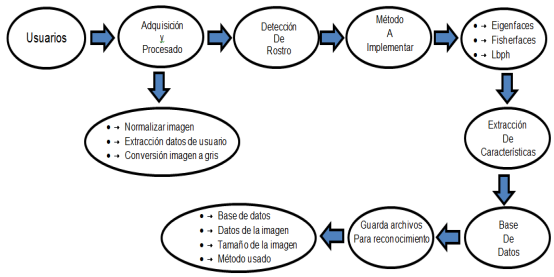

Fig. 4. Etapa de entrenamiento.

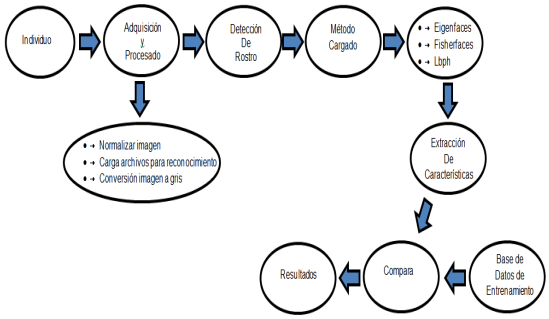

Fig. 5. Etapa de validación.

Con esta detección de rostro se asegura la normalización de los datos para que cada rostro tenga el mismo tamaño y por ende la longitud de todos los vectores de información requerida por cada método sea la misma.

Para la etapa de validación se realiza el mismo procedimiento de detección de rostro para normalizar el resultado con cada uno de los sistemas. Las pruebas de validación se realizan aplicando el procedimiento mostrado en el diagrama de bloques de la figura 5. Es claro anotar que al seleccionar de forma aleatoria cinco de las imágenes de cada individuo para entrenamiento y 2 para validación, es probable que en la validación quede una de las dos imágenes seleccionadas con alguna expresión facial. Esto se hizo para ingresar información adicional al sistema y analizar cómo reacciona ante variaciones en la pose.

Una segunda prueba se hizo añadiendo imágenes de personas que no existían en la base de datos de entrenamiento. Esto con el fin de validar los errores de falsos positivos, ya que en un sistema de seguridad basado en reconocimiento de rostro no se puede permitir que a una persona desconocida la detecte como una que se encuentra en la base de datos. Para esta prueba se empleó un total de 15 imágenes.

Los resultados finales de cada una de las dos pruebas realizadas en el computador se presentan en las tablas 1 y 2, en las que se aprecia que el mejor sistema de clasificación es el de *Eigenface* el cual no presentó ningún error de falso positivo que es el más importante en esta clase de sistemas de reconocimiento de rostro. A su vez, el tiempo de ejecución de este clasificador es 83.86% menor al tiempo de LBPH que es el segundo mejor clasificador en resultados de error.

Tabla 1: Resultados de las pruebas de error en el portátil.

Eigenfaces		**LBPH**		**Fisherfaces**	
Efic. (%)	Tiempo (ms)	Efic. (%)	Tiemp. (ms)	Efic. (%)	Tiemp. (ms)
97	4.67	96	28.95	94	3.42

Tabla 2: Resultados de las pruebas de falsos positivos en el portátil.

Eigenfaces		**LBPH**		**Fisherfaces**	
Efic. (%)	Tiempo (ms)	Efic. (%)	Tiemp. (ms)	Efic. (%)	Tiemp. (ms)
0	5.6	0	26.9	6	3.4

De la misma manera que se realizaron las pruebas 1 y 2 en el computador se realizaron en los sistemas en la BeagleBoard-xM. El único sistema de clasificación que no funcionó de igual manera en el computador y en la BeagleBoard-xM fue el LBPH, el cual no reconoció a ningún usuario dentro de la base de datos, arrojando un 100% de error. Esto se debe que en la BeagleBoard-xM se instaló una versión de OpenCV anterior a la usada en el portátil, debido a que en sistemas ARM (*Acorn RISC Machine*) la versión usada en el portátil no estaba disponible.

Finalmente en las tablas 3 y 4 se presentan los errores de clasificación y errores de falsos positivos para cada clasificador en la BeagleBoard-xM.

Tabla 3: Resultados de las pruebas realizadas en la BeagleBoard-xM.

Eigenfaces		**LBPH**		**Fisherfaces**	
Efic. (%)	Tiemp. (ms)	Efic. (%)	Tiemp. (ms)	Efic. (%)	Tiemp. (ms)
97	174.78	0	562.72	94	95.4

Tabla 4: Resultados de la pruebas de falsos positivos realizada en la BeagleBoard-xM.

Eigenfaces		**LBPH**		**Fisherfaces**	
Efic. (%)	Tiemp. (ms)	Efic. (%)	Tiemp. (ms)	Efic. (%)	Tiemp. (ms)
0	154.4	0	567.3	6	98.5

4. CONCLUSIONES

A través de la realización de este proyecto se pudo determinar que el algoritmo de clasificación empleado para reconocimiento facial que presenta los mejores resultados entre los tres propuestos es el método de *Eigenfaces*, ya que es el que presenta menor porcentaje de error de clasificación y menor tiempo de ejecución. Así mismo, el que presenta el peor resultado es el clasificador *Fisherface* debido a que incurre en el error de falso positivo, lo cual es grave cuando se desea implementar este tipo de trabajo en un entorno real.

Al cargar los archivos generados en el entrenamiento para realizar el reconocimiento, sucedió que el primer usuario identificado, tuvo un tiempo mayor que los siguientes. Esto, debido que en el entrenamiento al cargar las imágenes se pone lento el sistema, lo cual el proceso se demora en el reconocimiento de la primera imagen.

Por medio de varias pruebas con diferentes sistemas operativos se comprobó que el mejor para el desarrollo del algoritmo de reconocimiento facial fue Debian en la BeagleBoard-xM, debido a que en los otros no se permitían la instalación de algunas librerías de OpenCV, necesarias para la ejecución del código.

Se presentaron inconvenientes para realizar pruebas en tiempo real por medio de cámara *web*, debido que la memoria y procesador de la BeagleBoard-xM no soporta el procesamiento de este algoritmo.

Si el sistema de reconocimiento se emplea para un sistema de seguridad el usuario pondrá una cara neutral para su ingreso, haciendo fácil el reconocimiento. En este proyecto se realizó el reconocimiento con diferentes gestos, lo cual lo hace más complejo y aumenta los errores.

A futuro este proyecto se puede mejorar para realizar diferentes trabajos en el reconocimiento de personas en tiempo real por medio de alguna cámara de video y sería de gran ayuda que los sistemas embebidos tuvieran una cámara propia para no consumir tantos recursos.

REFERENCIAS

Alvarado, J. D., & Fernández, J. (2012). Análisis de textura en imágenes a escala de grises, utilizando patrones locales binarios (LBP). *ENGI Revista Electrónica de La Facultad de Ingeniería*, *1*(1), 1–6.

Álvarez, P. A. (2013). *Prototipo de sistema piloto para control de acceso basado en reconocimiento de rostros*. Universidad Militar Nueva Granada, Facultad de Ingeniería, Ingeniería en Multimedia.

BeagleBoard Group. (2014). BeagleBoard-xM. Retrieved May 15, 2015, from http://beagleboard.org/beagleboard-xm

Belhumeur, P. N., Hespanha, J. P., & Kriegman, D. J. (1997). Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *Pattern Analysis and Machine Intell., IEEE Trans. on*, *17*(9), 711–720.

Ding, C., Xu, C., & Tao, D. (2015). Multi-task pose-invariant face recognition. *Image Processing, IEEE Transactions on*, *24*(3), 980–993.

Ekman, P. (1993). Facial expression and emotion. *American Psychologist*, *48*(4), 384.

Gottumukkal, R., & Asari, V. K. (2003). System level design of real time face recognition architecture based on composite PCA. In *Proceedings of the 13th ACM Great Lakes symposium on VLSI* (pp. 157–160). AMC.

Hassner, T., Harel, S., Paz, E., & Enbar, R. (2015). Effective face frontalization in unconstrained images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 4295–4304).

Kanade, T., Cohn, J. F., & Tian, Y. (2000). Comprehensive database for facial expression analysis. In *Automatic Face and Gesture Recognition, 2000. Proceedings. Fourth IEEE International Conference on* (pp. 46–53). IEEE.

Kshirsagar, V., Baviskar, M., & Gaikwad, M. (2011). Face recognition using Eigenfaces. In *Computer Research and Development (ICCRD), 2011 3rd International Conference on* (pp. 302–306). IEEE.

Martínez, A. M., & Kak, A. C. (2001). PCA versus LDA. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *23*(2), 228–233.

Pizarro Jara, P. A. (2011). *Implementación en hardware de algoritmo de reconocimiento de rostros BDPCA+LDA*. Universidad de Concepción. Facultad de Ingeniería. Departamento de Ingeniería Eléctrica.

Refaeilzadeh, P., Tang, L., & Liu, H. (2009). Cross-validation. In *Encyclopedia of database systems* (pp. 532–538). Springer.

Reyes López, C. (2005). *Reconocimiento facial mediante visión artificial*. Universidad de

Sevilla, Escuela Técnica Superior de Ingenieros.

Seo, N. (1998). Eigenfaces and Fisherfaces. University of Maryland.

Shah, J. H., Sharif, M., Raza, M., & Azeem, A. (2013). A Survey: Linear and Nonlinear PCA Based Face Recognition Techniques. *Int. Arab J. Inf. Technol.*, *10*(6), 536–545.

Silva, E., Esparza, C., & Mejía, Y. (2012). POEM-based facial expression recognition, a new approach. In *Image, Signal Processing, and Artificial Vision (STSIVA), 2012 XVII Symposium of* (pp. 162–167). IEEE.

Turk, M., & Pentland, A. (1991). Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, *3*(1), 71–86.

Viola, P., & Jones, M. J. (2004). Robust real-time face detection. *International Journal of Computer Vision*, *57*(2), 137–154.