# Clasificación de Género y Edad Facial en Video - Final

Proyecto de Grado

**Integrantes: Guzmán Olivera, Fabrizio Albertoni, Matías Tudurí**
**30/07/2016**
**Tutor: Álvaro Pardo**

# Índice

## Abstract

En el presente documento se plantea la problemática de reconocimiento, seguimiento y clasificación facial en tiempo real, según género y rangos etarios. Se estudian los problemas típicos y las técnicas más utilizadas, analizando el estado del arte en procesamiento de imágenes, aprendizaje automático y visión artificial. Se diseña un posible sistema para resolver este problema y se presenta un prototipo para evaluar el rendimiento del conjunto de técnicas elegidas.

# Introducción

## Planteo del problema

El desarrollo de esta investigación surge con el propósito de estudiar de forma automática las características de un público determinado en tiempo real, en escenarios tanto controlados como no controlados. En este caso, estas características se limitan al género y edad de la persona. Por ejemplo, dado una campaña publicitaria resultaría útil saber sobre qué tipo de público tuvo efecto la misma. Para esto, debe ser posible detectar los rostros en video, poder seguirlos a medida que se desplazan y diseñar un modelo que pueda decidir a qué clase pertenece cada uno.

## Objetivos y alcance

El objetivo principal de este proyecto es incursionar en el estado del arte de las áreas que conciernen al problema a resolver, estudiar las diversas técnicas y decidir cuáles son las más adecuadas y eficientes, para posteriormente implementar un sistema de visión artificial que cumpla los requisitos planteados. Para esto se probaron diferentes alternativas para cada subproblema que implica el problema a resolver hasta obtener soluciones satisfactorias.

Una vez que esté determinado el diseño de cada componente del sistema, se construirá un prototipo en C++ para demostrar la validez del sistema y verificar el rendimiento del mismo en las distintas tareas que debe realizar. Se busca que el sistema pueda funcionar en situaciones reales y en tiempo real, limitándose a aquellos escenarios controlados o semi-controlados donde se tenga algún conocimiento previo de cómo se dispondrán los rostros en la escena (por ejemplo, una puerta de entrada donde se tengan tomas de frente de las personas). Se pretende que el sistema sea robusto y saque provecho de tener varias tomas de los rostros al realizar un seguimiento de los mismos en video.

## Estructura del documento

### Estado del arte

En este capítulo se analizan las técnicas encontradas en el estado del arte para cada etapa del sistema, se analiza su aplicabilidad y eventualmente los resultados obtenidos. Además se describen algunas de las aplicaciones de la industria.

### Herramientas para la creación de Software de Visión Artificial

Se describieron algunas de las librerías y herramientas más populares para el procesamiento de imágenes y video que permiten desarrollar aplicaciones de visión artificial.

### Bases de datos

Se estudiaron diversas bases de datos de rostros humanos. Se estudia la información tal como la edad, género, raza entre otros de dichos rostros, así como el escenario en que fueron tomadas.

### Resultados y pruebas preliminares

En este capítulo se implementaron diversos métodos en las distintas etapas necesarias del sistema de visión propuesto, y se realizó una comparación de performance (precisión de cada solución) sobre distintos escenarios realizando varios experimentos.

### *Tracking*

El *tracking* de rostros de personas, en este caso, requiere un algoritmo lo suficientemente robusto como para soportar oclusiones, variación del tamaño del objeto *trackeado* y contar con capacidades como seguir a un objeto que se mueve a velocidad no constante. En este capítulo se estudian varios enfoques y se realizan experimentos para determinar la mejor solución al problema.

### Diseño

En este capítulo se detalla el diseño de la solución propuesta construida en base a la información obtenida en el estudio de las diferentes técnicas y en los experimentos.

## Acercamiento al problema

La clasificación de rostros en clases de distintos tipos es un área muy activa dentro de la visión artificial y el aprendizaje automático, tanto a nivel académico como industrial. Existen muchas investigaciones sobre diversos problemas de esta clase, y en los últimos años se han lanzado diversos productos que buscan solucionar un problema en este ámbito, desde estimar el género y la edad hasta el reconocimiento de emociones, gestos, etnia, entre otros.

La principal dificultad radica en decidir qué información extraer de los rostros, de forma que caracterice eficazmente las distintas clases a la cual pertenece y cómo

combinarlas con un clasificador que sea capaz de realizar la distinción de la mejor forma posible. El problema es aún mayor en sistemas en tiempo real, pues el tiempo que se tiene para procesar la imagen, extraer información y realizar la clasificación es limitado, lo cual hace que no cualquier técnica sea adecuada.

Existen varios problemas típicos con los que se debe lidiar en este tipo de sistemas que interactúan con el mundo físico, que pueden afectar considerablemente el rendimiento del mismo. A continuación se listan los principales factores a considerar:

- Cambios de iluminación: las técnicas de extracción de características que no sean invariantes a la iluminación (por ejemplo aquellas basadas en la distribución del color). Esto requiere que se tenga una base de datos de entrenamiento con distintos tipos de iluminación.

- Cambios de pose: estos cambios pueden perjudicar métodos que toman en cuenta la posición relativa de las facciones del rostro o que extraen características dependiendo de la sección del rostro. Por esto muchas bases de datos de entrenamiento brindan tomas de varios ángulos de la misma persona.

- Ruido: las condiciones en que se obtiene la imagen del rostro también juega un rol determinante. Se debe lidiar con el ruido inherente al proceso de obtención de la imagen y de pasaje de la señal analógica a la digital debido al muestreo y cuantización de la misma. Además, existe la pérdida de información que implica tratar un objeto tridimensional como es un rostro en una representación bidimensional del mismo. Existen diversas técnicas utilizadas para minimizar el ruido, teniendo especial cuidado de no afectar la información más relevante de la imagen.

Estos problemas son típicos y muchos se acentúan a medida que el escenario se hace menos controlado, por lo que es necesario establecer en qué tipo de escenarios se utilizará el sistema para saber a priori con qué tipo de dificultades se deberá tratar.

Por otro lado, el proceso de adquisición de las imágenes de los rostros implica detectarlos y seguirlos a lo largo del video para obtener varias tomas del mismo, ya que el proceso de detección suele ser computacionalmente costoso e inviable si se realiza cuadro a cuadro, por esto suele ser necesario valerse de un *tracker* para realizar el seguimiento de este una vez detectado, sin embargo el principal problema que presenta esta metodología es como realizar dicho proceso de forma robusta (ante momentos de oclusión del rostro y aparición posterior, cruzamiento de rostros, movimientos bruscos o cambio en el tamaño de los rostros al acercarse a la cámara

son las principales dificultades). Si bien existen numerosas soluciones al problema del seguimiento, muchas de estas están diseñadas para casos simples muy controlados y fallan cuando el escenario se vuelve de mayor complejidad. Algoritmos que asumen un fondo constante no pueden utilizarse cuando se tiene un alto flujo de personas desplazándose, por ejemplo.

## Estado del Arte

En este capítulo se presenta un estudio del estado del arte actual en cuanto al reconocimiento de género y edad, presentando métodos que fueron usados para atacar las distintas etapas del problema junto con sus correspondientes resultados. Estos métodos buscan solucionar el problema específico de clasificación de género y edad en imágenes o video, y no buscan ser aplicables al sistema particular que queremos diseñar, que cuenta con ciertas restricciones.

Los métodos propuestos para predecir el género de una persona dada una imagen de su rostro pueden alcanzar una precisión muy alta (mayor al 95%) si se posee una toma buena del rostro, en cambio el reconocimiento de edad depende mucho de los rangos etarios usados si se enfoca como un problema de clasificación, pero si se enfoca como un problema de regresión, el problema de definir los rangos etarios desaparece. Este último es un problema mucho más difícil que el primero, por lo cual la precisión de los métodos propuestos en el estado del arte es sensiblemente menor.

La mayoría de estas técnicas implican tres etapas: pre-procesamiento, extracción de características y clasificación. Por lo general el preprocesamiento depende mucho de la extracción de características que se realizará, por lo que estas etapas por lo general se presentarán juntas. A continuación se mencionan los métodos más usados para cada etapa y como se han combinado para resolver cada problema.

## Técnicas para la extracción de características y preprocesamiento

### Transformada *Wavelet*

La transformada wavelet ha sido usada como parte de la extracción de características del rostro en varias investigaciones. Por ejemplo, en (Hussain et al., 2013) se utiliza en particular la Dyadic Wavelet Transform junto con LBP (presentado más adelante) para construir un descriptor del rostro, el cual reporta una presición del 99.25% sobre la base de datos FERET. Dicha transformada se define como sigue:

$$\psi(t) = \sum_k g[k]\sqrt{2}\varphi(2t - k)$$

Donde se define como:

$$\varphi(t) = \sum_k h[k]\sqrt{2}\varphi(2t - k)$$

La idea de esta transformada es descomponer la imagen a diferentes escalas y frecuencias en varias sub-bandas que hace el análisis más simple. A diferencia de, por ejemplo, la transformada de Fourier, la cual brinda información sobre el espectro de una señal (en este caso una imagen) pero no resulta útil para extraer información en el dominio temporal, la transformada Wavelet utiliza una base de señales trasladadas y dilatadas, lo cual ayuda a analizar la señal en estos términos.

Por otro lado, muchas funciones pueden ser representadas de un modo mucho más compacto mediante la transformada Wavelet que con la transformada de Fourier, especialmente aquellas con discontinuidades (muy presentes al tratar con imágenes), cuya representación mediante una base de senos y cosenos sería más extensa. Dado esto, la Transformada Wavelet reduce la cantidad de bits requeridos para representar una imagen, por lo cual resulta muy útil en la extracción de características de un rostro. (Nithyashri & Kulanthaivel, 2012)

**LBP (*Local Binary Pattern*)**

Es una técnica muy frecuentemente utilizada en este ámbito (Ylioinas, Hadid, & Pietikainen, 2012), (Shan, 2012), (Sai, Wang, & Teoh, 2015), (Moeini, Faez, & Moeini, 2015). El procedimiento para crear el vector de características LBP, es básicamente el siguiente:

1) Utilizar una ventana de algún tamaño (3x3, por ejemplo) para dividir la imagen.
2) Para cada pixel de la celda compararlo con el valor de iluminación de los 8 vecinos.
3) Si el valor del vecino es menor al del centro entonces se escribe un 0 en éste, de lo contrario se escribe un 1.

4) El paso anterior resulta en un número binario de 8 dígitos que puede convertirse a decimal si fuese conveniente

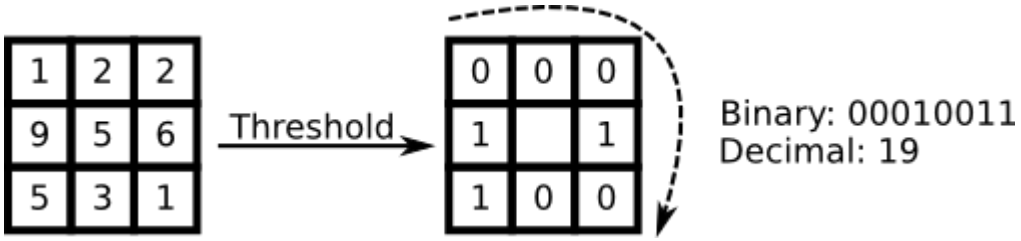



Fig. 1

5) Construir un histograma a partir de la cantidad de ocurrencias de cada número obtenido luego del paso 4) para cada celda.
6) Opcionalmente se normaliza este histograma
7) Se concatenan los histogramas de todas las celdas resultando en un vector de características para la ventana entera.
8) El vector de características puede ser procesado usando algún clasificador o algoritmo de aprendizaje automático para clasificar imágenes.

**LGBP *(Local Gabor Binary Pattern)***

LGBP (Moeini et al., 2015) es una variante de LBP y se propone como método para la extracción de características de un modelo 3D de un rostro construido mediante GEM (*Generic Elastic Model*) a partir de una imagen 2D frontal. De esta forma se considera información de textura y profundidad en el rostro.

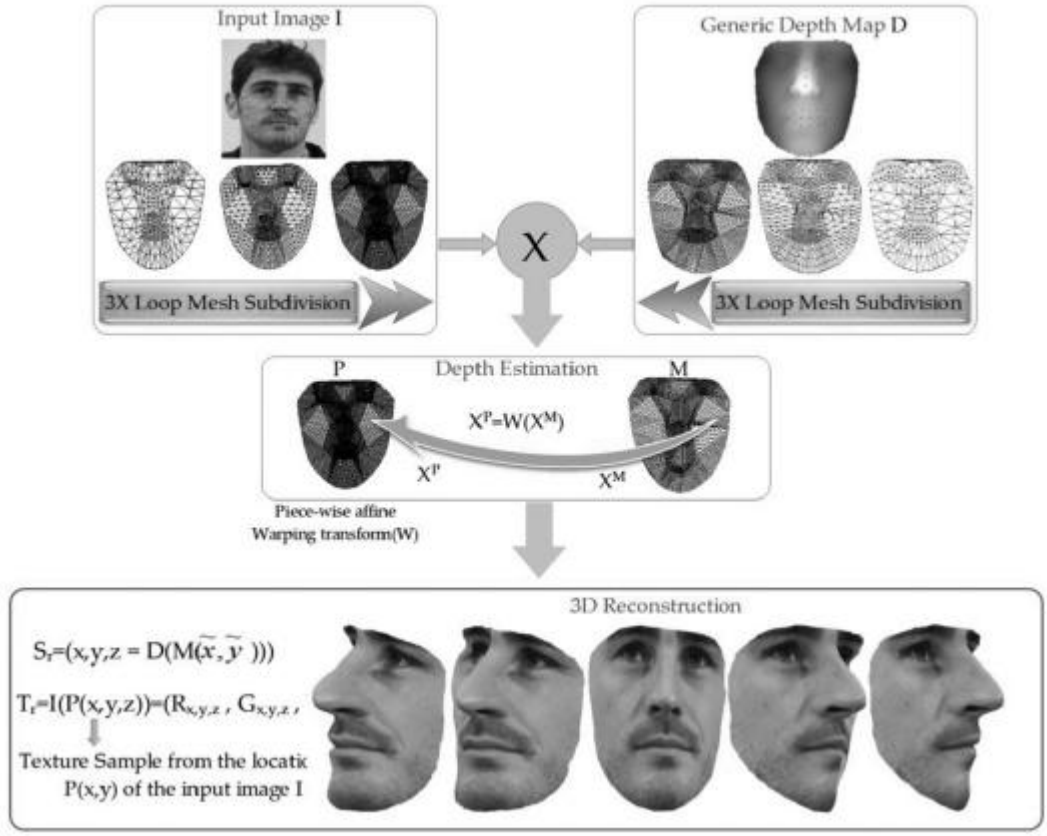



Fig. 2

En este trabajo, se combinaron los vectores de características para elaborar un histograma LGBP y luego clasificar los datos mediante SVM. Los experimentos se realizaron con las bases de datos LFW y FERET. El resultado fue de una tasa de acierto de 98.55 - 99.8% detectando los rostros con Viola-Jones, y entrenando y testeando con 5-fold *cross-validation*.

**HOG**

El descriptor HOG (Histogram of Oriented Gradients) desarrollado en (Dalal & Triggs, 2005) con el propósito de realizar detección de objetos, ha sido ampliamente utilizado en otras áreas de la visión artificial y procesamiento de imágenes. Esta técnica en su versión más simple cuenta ocurrencias de determinadas orientaciones de gradientes en ciertas porciones de la imagen, pero se pueden encontrar muchas variantes del mismo. La idea detrás de HOG es capturar la forma del objeto, ya que esta se puede describir mediante la distribución de los gradientes de la intensidad de la imagen, o lo que es equivalente, la dirección de los bordes. Usualmente se divide la imagen en varios sectores y se calcula el vector HOG en cada sector, para luego concatenarlos todos en un solo vector de características.

Es evidente que para calcular HOG es necesario obtener el gradiente de la imagen

en cada punto, lo cual implica poder calcular las derivadas parciales de la misma. Existen varias formas de estimarlas, el método más común es utilizar el operador de Sobel (Sobel & Feldman, 1968) para estimar las derivadas parciales, que se basa en aplicar la convolución de la imagen con un filtro pequeño y separable (ver figura 3), lo cual hace muy eficiente su cómputo. Una vez obtenidas las derivadas parciales en cada punto, el cálculo del gradiente es trivial.

$$\mathbf{G}_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} * \mathbf{A}$$

Fig. 3 (Filtro de Sobel para calcular la derivada parcial en x, mediante la convolución en 2 dimensiones de un kernel de 3x3 con una imagen A)

**PHOG**

PHOG (Pyramid Histogram of Oriented Gradients) es una variación al método original, introducida por (Arigbabu, Syed Ahmad, Wan Adnan, Yussof, & Mahmood, 2015) atacando el problema de la clasificación de género en imágenes sin restricciones, siendo una técnica robusta ante variaciones fotométricas. Esta técnica introduce varios cambios a su versión original. En primer lugar, implementa una etapa de pre-procesamiento que involucra el suavizado de la imagen, con el fin de reducir el ruido, y la detección de los bordes de la misma. Esto se realiza aplicando un filtro Gaussiano para el suavizado y el posterior cálculo del Laplaciano de la imagen resultante. Este proceso queda determinado mediante la siguiente ecuación:

$$O(x,y) = \nabla^2(I(x,y) * g(x,y))$$

Siendo $\nabla^2$ el laplaciano de la imagen I, * el operador de convolución y g una gaussiana. En la figura 4 se muestra un ejemplo del resultado de este proceso.

*(Fig. 4)*

Esto puede implementarse mediante la convolución de la imagen con un solo filtro, conocido LoG (Laplacian of Gaussian). El kernel para aplicar este filtro se obtiene

utilizando el generador de kernels de Matlab. Según su documentación, el filtro LoG se crea utilizando las siguientes fórmulas.

$$h(n_1, n_2) = \frac{(n_1^2 + n_2^2 - 2\sigma^2)h_g(n_1, n_2)}{2\pi\sigma^6 \sum_{n_1} \sum_{n_2} h_g}$$

$$h_g(n_1{}^2, n_2{}^2) = e^{\frac{-\left(n_1^2+n_2^2\right)}{2\sigma^2}}$$

Lo más importante que se introduce es la estructura piramidal del descriptor, al dividir la imagen en una grilla 2x2, calcular HOG sobre cada sector, y subdividir cada grilla en otra de 2x2 N veces. En la figura 5 se muestra un ejemplo del resultado de una grilla de 8x8. Esta técnica, combinada con un clasificador SVM, alcanzó una precisión en la clasificación del 88.5% en el dataset LFW, que contiene imágenes en escenarios poco controlados.

(Fig. 5)

**Filtro de Gabor (2D Principal Component Analysis)**

En (Rai & Khanna, 2015) se propone un método para clasificación de género que resulta invariante a la iluminación, expresión del rostro y ruido, lo cual lo hace apropiado para aplicaciones del mundo real. Se utiliza el filtro de Gabor en dos dimensiones junto con 2DPCA (2D Principal Component Analysis). Esto brinda el espacio de Gabor real. El 2DPCA se utiliza en dirección horizontal y vertical para seleccionar las características más discriminativas del espacio de Gabor. Para la clasificación se utiliza SVM.

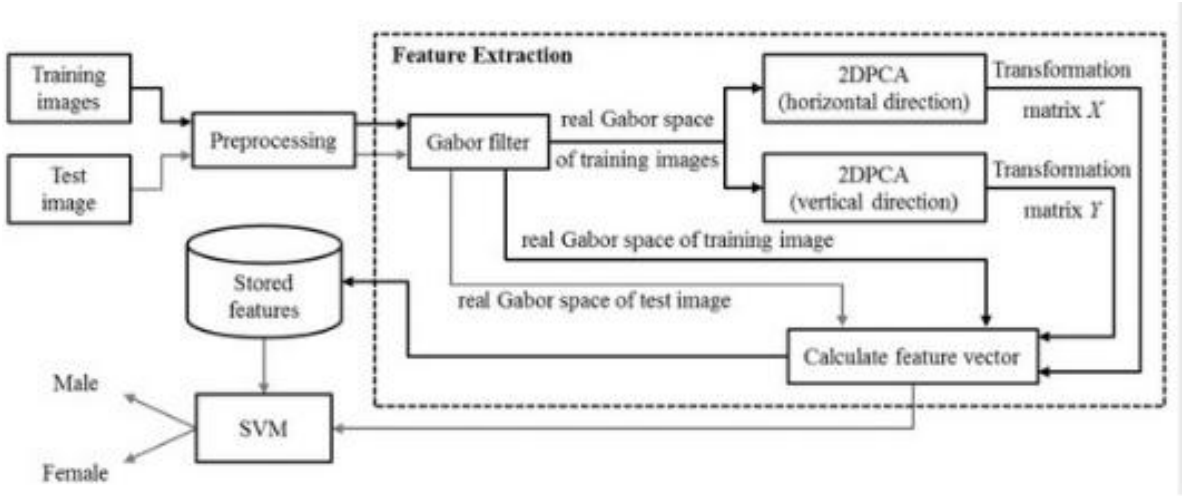



Fig.6

El sistema logra un ratio de acierto de 98.18%, 96.61%, 96.15%, 93.33%, y 88.34% para las bases de datos FERET, FIE A, AR, Indian Face, y LFW, respectivamente.

**Aprendizaje Manifold**

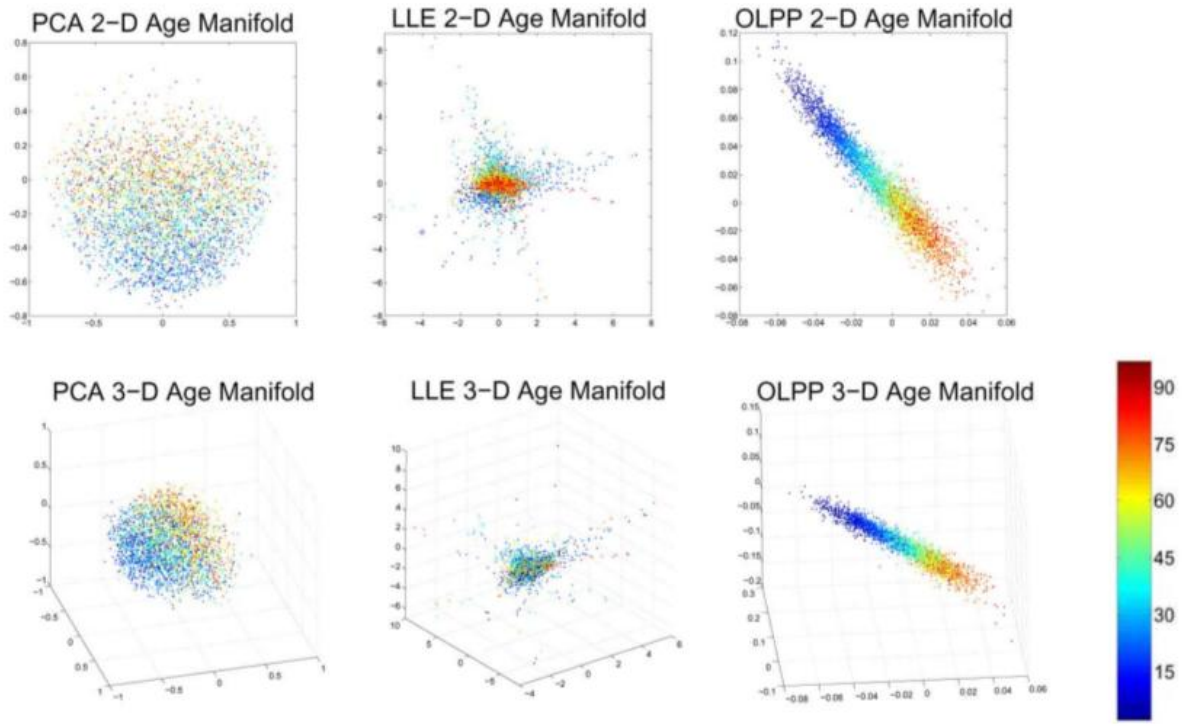
Manifold se ha utilizado como método para extraer características de edad de imágenes de rostros humanos (Guo, Fu, Dyer, & Huang, 2008). No analiza patrones de envejecimiento específicos para cada individuo permitiendo utilizar rostros de diferentes personas para la misma edad.

La edad se determina con una proyección en un subespacio que es capaz de reconstruir la imagen del mismo, esto se utiliza para aprender la edad en una topología de baja dimensión. Combinando éste con el método AAMs («Active Appearance Models», s. f.) se puede incorporar además información sobre la forma y la apariencia para estimar coeficientes de una función de regresión.

Básicamente esta técnica usa una función de proyección P con la que se realiza un mapeo 1-1 de los elementos del espacio de imágenes X al espacio de Manifold. En este último espacio las imágenes quedan etiquetadas con la edad del mundo real.

Se trata de un aprendizaje supervisado, considerando la información de la etiqueta del modelo.

Uno de los tipos de encaje para dicho método se conoce como OLPP (Orthogonal Locality Preserving Projections) (Cai, He, Han, & Zhang, 2006) y cuenta con el mayor poder de discriminación de los datos entre los métodos que se analizaron en este trabajo. El mismo produce funciones de base ortogonal basadas en LPP.




Fig, 7

Luego de aprender la edad manifold usando OLPP, cada imagen se proyecta a ésta para extraer un vector característico.

En cuanto a performance, Manifold es usado para predecir edad junto con un método de regresión LARR nombrado así por los autores, usando la base de datos UIUC-IFP-Y, alcanza un error de 5.25 años para el género femenino (usando 64 clases para particionar las edades de 0 a 93); y 5.3 años para el masculino cuando el ajuste es de 32 clases.

## Clasificación

En la fase de clasificación se separan los vectores de características en clases de acuerdo a sus valores. A continuación se describen los métodos más utilizados en la literatura.

### SVM

Support Vector Machines (SVM) es uno de los clasificadores más usados que fueron encontrados en la literatura y las distintas investigaciones. Se trata de un método propuesto en (Vapnik & Vapnik, 1998) cuyo objetivo consiste en separar, a través de un hiperplano, dos conjuntos de datos con la máxima distancia posible entre ellos. Garantizar esto permitirá conseguir una clasificación más acertada. Los

límites entre los conjuntos de datos y el hiperplano se conocen como Vectores de Soporte (*Support Vectors*). (Gumus, Kilic, Sertbas, & Ucan, 2010)

Cada punto del conjunto total de datos se define como $x_i \in \Re^n$ con i = 1,2, …, N y forma parte de una clase $y_i \in \{-1, 1\}$. Para una clasificación lineal se pueden identificar dos clases y los siguientes hiperplanos:

$$w \cdot x_i + b \geq 1, para \ y_i = 1$$
$$w \cdot x_i + b \leq -1, para \ y_i = -1$$

La distancia entre los vectores de soporte para este caso es:

$$d = \frac{2}{||w||}$$

Cuanto mayor sea $d$ , mejor será la separación entre las dos clases. Maximizar $d$ implica minimizar $w$. Este problema de optimización se puede resolver mediante una función de Lagrange:

$$L(w, b, \alpha) = \frac{||w||^2}{2} - \sum_{i=1}^{m} \alpha_i \cdot \{y_i \cdot [(w \cdot x_i) + b] - 1\}$$

En esta ecuación $\alpha_i$representa a los multiplicadores de Lagrange y se resuelve minimizando según $w$ y $b$ y maximizando según los valores $\alpha_i \geq 0$. La mayoría de valores adecuados para el parámetro $w$ pueden obtenerse con la siguiente condición:

$$w = \sum_{i=1}^{m} \alpha_i \cdot y_i \cdot x_i \cdot \alpha_i \geq 0$$

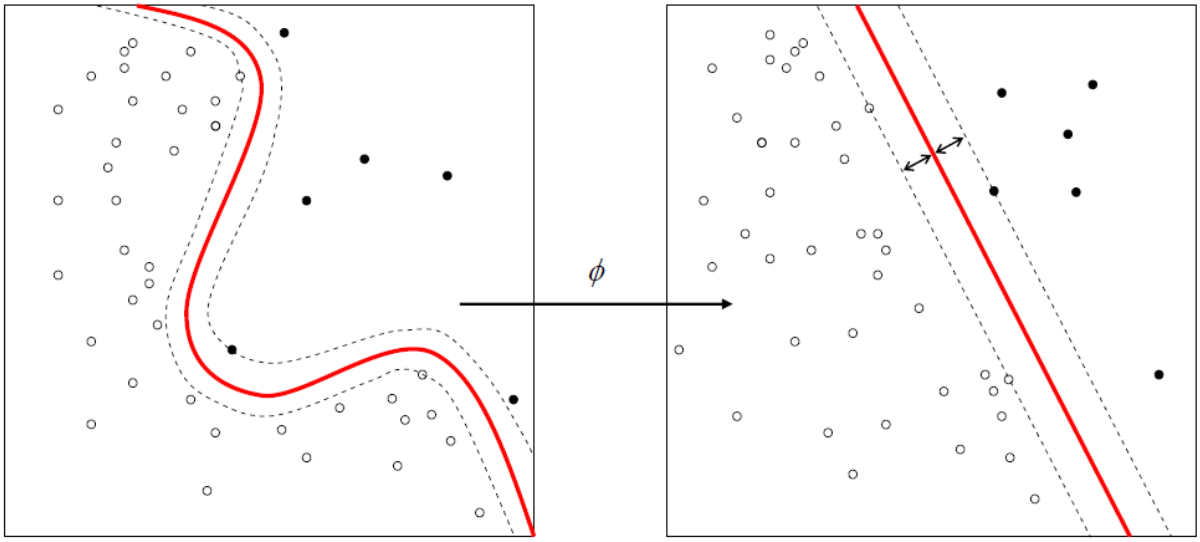
Luego, la distancia de cualquier punto $x$ al hiperplano se define como:

$$d(w, b, x) = \frac{|w \cdot x + b|}{||w||}$$

Combinando las dos últimas ecuaciones se puede expresar la ecuación de distancia de un modo más generalizado:

$$d(x) = \frac{(\sum_{i=1}^{m} \alpha_i \cdot y_i \cdot x_i) \cdot x + b}{\left|\left|\sum_{i=1}^{m} \alpha_i \cdot y_i \cdot x_i\right|\right|}$$

Mediante el signo de $d$ se obtiene a cuál clase pertenece cada punto $x$ y $|d(x)|$ es la distancia del punto $x$ al hiperplano. Cuando $|d|$ aumenta entonces se tiene un mejor resultado en la clasificación.




(Fig. 8)

Hasta el momento se ha explicado la teoría en casos donde los conjuntos de datos son linealmente separables. Pero en casos donde no se puede obtener una separación lineal de los datos (las clases se superponen espacialmente) tal y como sucede en nuestro escenario, se realiza una conversión del espacio de características proyectándolo a una dimensión mayor donde se puede efectuar nuevamente una separación lineal (ver figura 8). Esta expansión se lleva a cabo con el operador $\phi(\cdot)$ y la función del hiperplano se modifica ligeramente:

$$f(x) = w \cdot \phi(x) \; + \; b$$

Considerando la sustitución de la variable w, la ecuación resultaría en:

$$f(x) = \sum_{i=1}^{m} \alpha_i \cdot y_i \cdot (\phi(x_i) \cdot \phi(x)) \; + \; b$$

Dado que el producto en espacios de dimensiones muy altas en intratable se utilizan funciones de Kernel $K(x_i, x) = \phi(x_i) \cdot \phi(x)$. Para estos propósitos se utilizan fundamentalmente dos funciones de Kernel:

1) Función de Kernel Polinómica (*Polynomial Kernel Function*)

$$K(x_i, x) = (x_i \cdot x + 1)^p$$

2) Función de Kernel de Base Radial (*RBF Kernel Function*)

$$K(x_i, x) = exp\left[-\gamma\left||x - x_i|\right|^2\right]$$

(Gumus et al., 2010)

Cabe mencionar que esta formulación asume un problema de clasificación en dos clases, sin embargo permanece igual para problemas de clasificación en N clases mediante un enfoque de uno contra el resto, esto es, tener un clasificador que indique si determinado vector pertenece a una de las N clases o no, necesitando N modelos.

Otro enfoque es tener un clasificador para cada par de clases, es decir (N-1)N/2 modelos, e implementar un sistema de votación entre todos los modelos, siendo esto más lento que el primer enfoque debido a la cantidad de clasificadores utilizados, pero produce mejores resultados.

**Distancia mínima**

Este clasificador calcula la distancia entre los vectores almacenados en fase de entrenamiento y el nuevo vector en fase de pruebas y se seleccionan los k ejemplos más próximos. El nuevo ejemplo se clasifica con la clase que más se repite en los vectores seleccionados. Entre los métodos matemáticos para calcular la distancia mínima se han utilizado: L1,L2 y CS. Se describen brevemente cada una de estas distancias:

**L1 *(city block distance)***

*City Block Distance*, también conocida como distancia de *Manhattan*, calcula la distancia en base a las diferencias en valor absoluto de los componentes de los objetos, un caso genérico para un espacio n-dimensional sería:

$$d_1(p, q) = \left||p - q|\right|_1 = \sum_{i=1}^{n} |p_i - q_i|$$

**L2 *(euclidean distance)***

La distancia euclídea entre dos puntos p y q pertenecientes a un espacio n-dimensional es la distancia en línea recta entre ellos y se calcula como:

$$d(p,q) = d(q,p) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$$

**CS *(Chi Square)***

El criterio de *chi-square* se basa en la suma, sobre grupos predefinidos, de la diferencia cuadrática entre los valores observados y los esperados, sobre los valores esperados. Su fórmula se define como:

$$\chi^2 = \sum \frac{(f_o - f_t)^2}{f_t}$$

.

De acuerdo con (Hussain et al., 2013) se obtuvo el mejor resultado con el clasificador L1 alcanzando 99.25% de precisión en la base datos FERET.

**Redes Neuronales Artificiales**

Las redes neuronales son una técnica de aprendizaje automático inspirada en el sistema nervioso biológico. Los primeros modelos propuestos datan del año 1943, por los neurólogos Warren McCulloch y Walter Pitts. No fue hasta 1958 que se desarrolló el primer perceptrón simple (red con una sola capa oculta) por Rosenblatt. El siguiente avance importante fue el desarrollo del método de aprendizaje (Witaszek, 1995) siendo hasta hoy uno de los más utilizados, entre otros.
Una red neuronal se compone de unidades llamadas neuronas, que reciben varias entradas y tienen una única salida. La relación entre entradas y salida está dada por la siguiente ecuación:

$$g(in_i) = g\left(\sum_{j=0}^{n} W_{j,i}a_j\right)$$

Donde $a_j$ son las entradas, $W_j$ los pesos a aprender por la red, y $g$ la función de activación, usualmente elegida entre la función sigmoidea, la tangente hiperbólica o la función escalón.

**Teoría de Resonancia Adaptativa**

Esta teoría se desarrolla en respuesta al dilema de la estabilidad y elasticidad del aprendizaje:

- **Plasticidad del aprendizaje**: permite a una red neuronal poder aprender nuevos patrones.
- **Estabilidad del aprendizaje**: permite a una red neuronal poder retener los patrones aprendidos.

Resulta sencillo cumplir con una, pero no con ambas. Por ejemplo, para el caso del perceptrón multicapa aprender nuevos patrones supone el olvido de lo anterior.
Se propone un aprendizaje por realimentación creado por la competencia entre neuronas de salida y de entrada. Esto se logra con una arquitectura como la mostrada en la figura 9. Consiste de 3 tipos de neuronas:

1. Unidades de entrada (nivel F1)
2. Unidades de clúster (nivel F2)
3. Mecanismo de reseteo

El nivel F1 toma las entradas y transfiere su salida a la neurona en F2 cuyo vector de pesos se acerca más al vector de entrada. Cada neurona en F2 emite una señal negativa (proporcional a la calidad en la coincidencia con el vector de entrada) a todas las otras neuronas del nivel, inhibiendo sus salidas. Esto se conoce como inhibición lateral, permitiendo que cada neurona de F2 represente una categoría de clasificación.

Luego de la clasificación, se compara el nivel de confianza de la misma contra cierto parámetro llamado de vigilancia. Si es superado, se comienza el entrenamiento, haciendo que los pesos de la neurona ganadora se ajusten conforme al vector de entrada. Sino, la neurona ganadora es inhibida y se comienza una búsqueda donde las neuronas de F2 con mayor activación son desactivadas una a una hasta que el parámetro de vigilancia es superado. La vigilancia tiene una influencia muy importante en la memoria de la red.

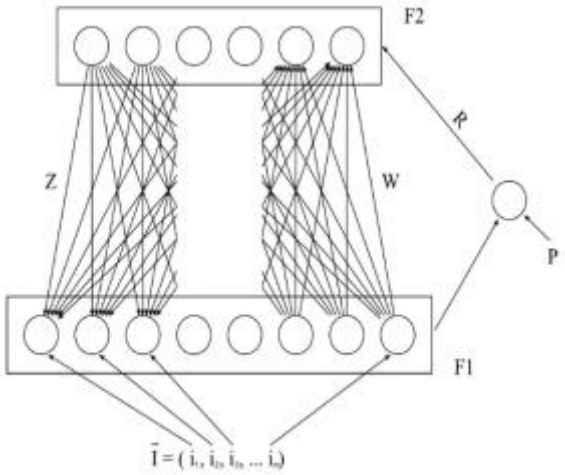



Fig. 9

Este tipo de redes fueron usadas en (Nithyashri & Kulanthaivel, 2012) para clasificación de edad, junto con un proceso de extracción de características basado en Wavelets y distancia euclídea entre las características. Los resultados obtenidos rondaron el 94 % para la base de datos FG-NET, con rangos de 0-12, 13-18, 19-59 y mayor a 60 años.

**Adaboost**

Adaboost (abreviatura de Adaptive Boosting) es un meta algoritmo de *machine learning* introducido por (Freund & Schapire, 1995). Es muy utilizado en problemas de clasificación para dos clases. Básicamente su funcionamiento, iterativo, se resume de la siguiente manera de acuerdo a (Wang, Yau, & Wang, 2009):

1- Para cada iteración t, se proporciona un conjunto de entrenamiento con una nueva distribución D, la cual depende de la performance de los clasificadores de base de la iteración inmediatamente anterior a t.
2- El error de cada clasificador se mide con respecto a D y es usado para calcular el peso de cada ejemplo en D.
3- En la ronda t, si el ejemplo se clasifica correctamente por el clasificador de base se le reduce su peso.
4- Se utilizan estos pesos en el remuestreo (resampling) o en la nueva ponderación (reweighting).
5- En consecuencia, aquellos patrones mal clasificados varias veces aumentan sus pesos y los clasificadores generados secuencialmente son forzados a concentrarse más en estos patrones difíciles.

***Extreme Learning Machines***

Extreme Learning Machines (ELM) son algoritmos simples y eficientes de aprendizaje para redes neuronales hacia adelante con una sola capa oculta. Resulta más rápido para este tipo de redes y provee mejor performance que el aprendizaje basado en un descenso de gradiente. A su vez, supera problemas como óptimos locales y sobreajustes, para los cuales el aprendizaje del gradiente necesitaría métodos adicionales para superarlos.

Esta técnica fue utilizada en (Sai et al., 2015) para clasificación de edad junto con LGBP (Local Gabor Binary Patterns), arrojando resultados de entre 60% y 80% (dependiendo de los rangos utilizados) en la base de datos FG-NET.

## Sistemas similares en la industria

Existen diversos programas de aplicación y librerías tanto gratuitas como pagas relacionadas con la idea de nuestro trabajo. Algunas de ellas son productos, a veces multiplataforma, que trabajan exclusivamente con imágenes y no con video y obtienen resultados de edad precisa (no rangos) y género (How old I look - de tipo web - y Gender Recognition System - desarrollada en Matlab -). Hay librerías que ofrecen además otros datos como expresiones faciales, humor, color de vestimentas, posición de la cabeza, etnias, gestos y poses de manos y dedos, entre otros (Sky Biometry, CrowdSight SDK, Kairos' human analytics platform) y la mayoría pueden utilizarse tanto para procesamiento de videos como de imágenes.

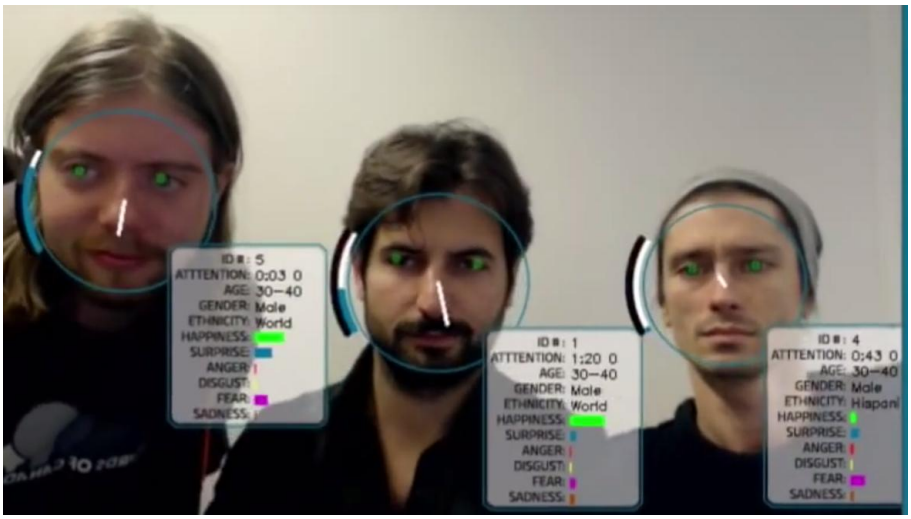



Fig. 10: *CrowdSight SDK*

Empresas como Google, Microsoft y IBM cuentan con APIs propias (Google CLOUD VISION API, Microsoft Face API y IBM AlchemyVision respectivamente) que efectúan reconocimiento de género y edad de personas y en algunos caso clasifican otros patrones como las emociones. En particular, la API de Microsoft además puede procesar audio para identificar a las personas o reconocer órdenes.

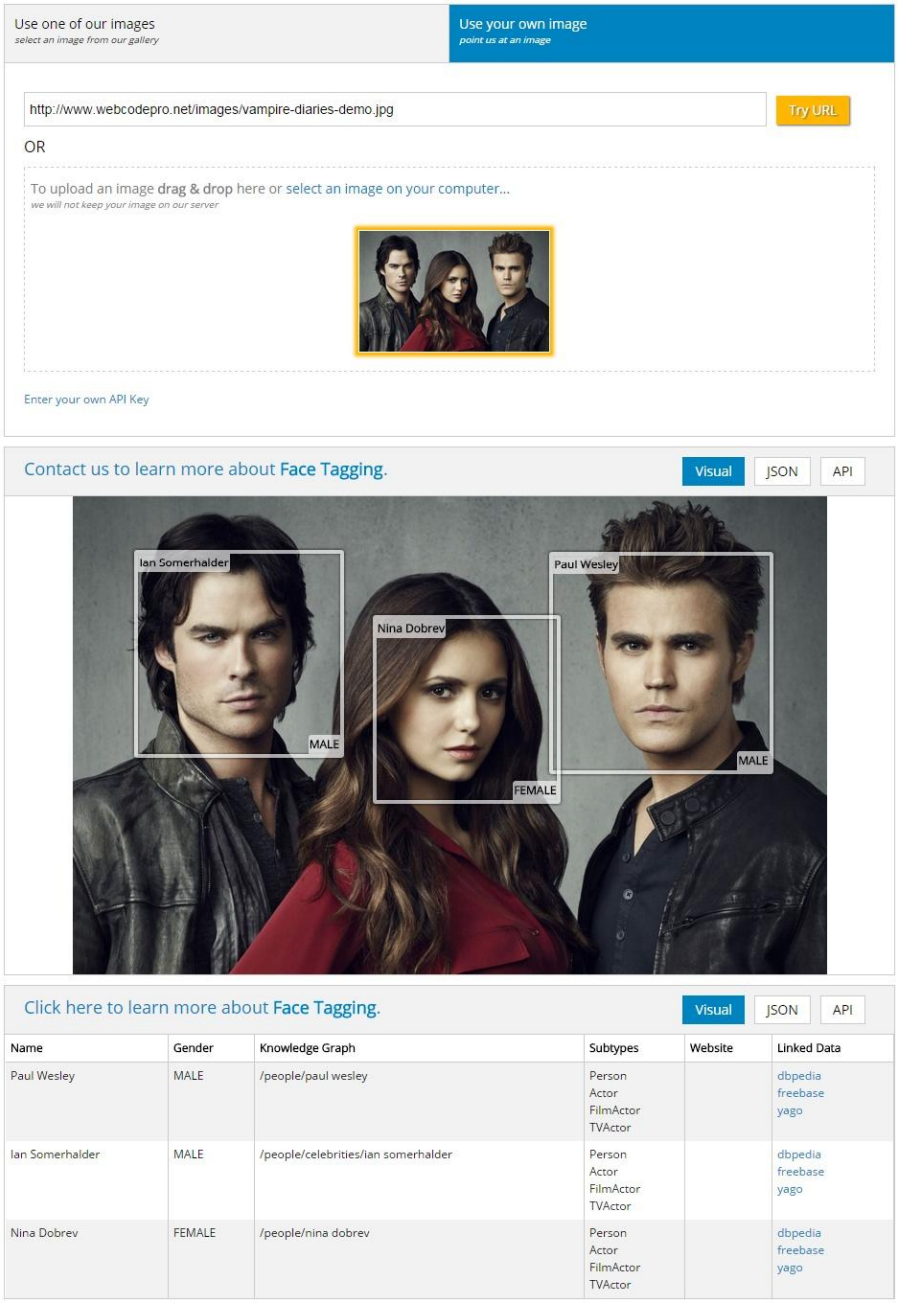



*Fig 11: AlchemyVision*

Por otro lado, Panasonic cuenta con un software llamado FacePro que trabaja con cámaras i-PRO de Panasonic para verificar la identidad de personas y mostrar reportes con los datos de esas personas incluyendo género y edad. Permite

búsquedas de personas detectadas en su base de datos y puede funcionar en tiempo real con hasta 20 cámaras por servidor.

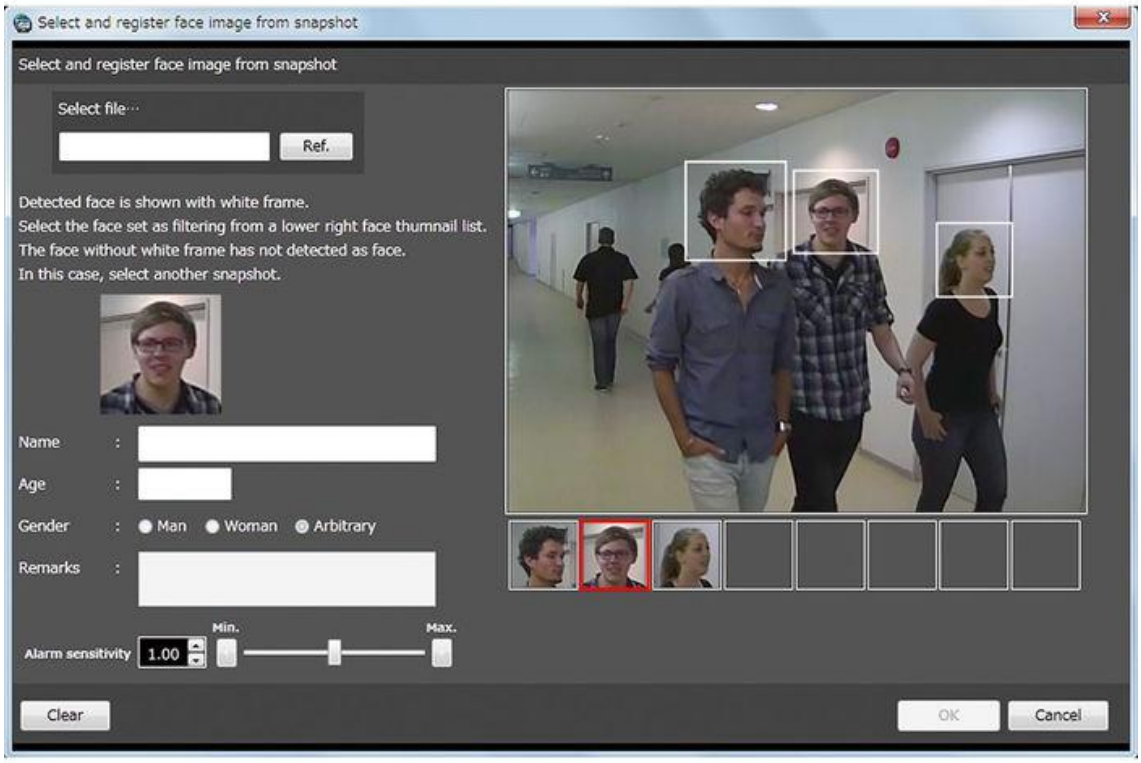



Fig. 12: *Panasonic FacePro*

## Herramientas para la creación de software de visión artificial

Existe una gran cantidad de bibliotecas disponibles para la creación de software de visión artificial, a continuación detallamos aquellas que han sido las más utilizadas durante estos últimos años:

### OpenCV

Librería libre de visión artificial y machine learning, diseñada para proveer una infraestructura común para aplicaciones de visión artificial y acelerar el uso comercial en productos.

Contiene más de 2500 algoritmos optimizados para el reconocimiento de rostros, identificación de objetos, clasificación de acciones humanas en videos, seguimiento de los movimientos de cámara y objetos, etc.

Así como también contiene una interfaz C++, C, Python y Java y soporta Windows, Linux, Android y Mac OS.

Esta librería posee más de 7 millones de descargas y es la más utilizada por compañías, grupos de investigación y entes gubernamentales.

**Matlab**

Matlab es un IDLE matemático que a nuestro interés proporciona un conjunto de funciones para el procesamiento y manipulación de vídeos e imágenes, conocidas como *Toolboxes*, por ejemplo: *Computer Vision System Toolbox* e *Image Processing Toolbox*. («Features - Computer Vision System Toolbox - Matlab», s. f.)

Como gran ventaja se destaca lo fácil y rápido que resulta desarrollar con esta herramienta, además de que es posible integrar proyectos de OpenCV en Matlab(«MATLAB and OpenCV - MATLAB», s. f.).

Sin embargo como desventaja frente a OpenCV es una herramienta paga que utiliza un lenguaje de alto nivel que permite generar código C de algunos objetos y funciones con ciertas limitaciones («Functions and Objects Supported for C and C++ Code Generation -- Alphabetical List», s. f.). OpenCV con C++ es más eficiente debido a que trabaja a bajo nivel. Sin embargo Matlab resulta muy útil para el prototipado rápido, lo cual es conveniente al probar algoritmos antes de su implementación en C++, además de contar con una documentación muy completa.

**SimpleCV**

Es un framework de código abierto para construir fácilmente aplicaciones de visión artificial en Python. Con SimpleCV también se pueden acceder a librerías de OpenCV, entre otras librerías de visión artificial existentes.

Luego de una revisión de las herramientas disponibles decidimos trabajar con OpenCV al ser una biblioteca libre de visión artificial y machine learning altamente utilizada, principalmente debido a que su publicación se da bajo licencia BSD permitiéndonos usarla libremente para propósitos comerciales y de investigación, además al estar diseñada en C++ y proporcionar dicha interfaz de desarrollo nos es útil en términos de optimización para la implementación del prototipo. Por otro lado, utilizaremos Matlab como entorno de pruebas de las diferentes técnicas que se estudien.

## Bases de datos

Luego de una búsqueda exhaustiva se logró confeccionar un posible listado de base de datos de entrenamiento a ser utilizadas, que cumplen con las características necesarias para la confección del prototipo («FACE RECOGNITION HOMEPAGE», s. f.):

**MORPH**

Se trata de una base de datos con 55,134 imágenes tomadas de un conjunto de más de 13,000 individuos, con un rango de edad desde 16 a 77 y una edad media de 33, posee un promedio de 4 imágenes por individuo y un tiempo promedio entre las fotos de 164 días. Proporciona metadata sobre la raza, género, peso, coordenadas de los ojos y fecha de nacimiento.

Para disponer de acceso al conjunto, se ha establecido un costo mínimo de US$ 99 para apoyar su desarrollo y distribución, sin embargo existen cupos limitados disponibles para estudiantes.(«MORPH | I3S», s. f.) («MORPH Non-Commercial Release Whitepaper», s. f.) («MORPH Database (non-commercial)», s. f.)

**Adience**

Adience es una base de datos de 26,580 imágenes de un conjunto de 2,284 individuos, esta colección proviene de álbumes de Flickr e intenta reflejar todos los posibles escenarios y condiciones de las imágenes, tales como variaciones de la iluminación, ruido y distintas poses.(«Face Image Project - Data - Adience», s. f.)

Proporciona metadata sobre el género, identificación de sujetos y edad. Los autores de este contenido permiten su utilización no comercial al público bajo la licencia CC (Creative Commons), en donde se debe añadir la siguiente cita en caso de que se utilice: (Eidinger, Enbar, & Hassner, 2014).

Su contenido es accesible para uso no comercial vía el enlace citado: («Sitio de descarga de adiencedb», s. f.). Mediante las siguientes credenciales:

**User**: adiencedb
**Password**: adience

En el directorio podemos observar los siguientes archivos:

- faces.tar.gz - Imágenes faciales recortadas.
- aligned.tar.gz - Imágenes faciales recortadas y alineadas.

- fold_0_data.txt - fold_4_data.txt -archivos de texto con la metadata de todas las imágenes
- fold_frontal_0_data.txt - fold_frontal_4_data.txt - archivos de texto con metadata solo de las imágenes de pose frontal.
- README.txt. provee detalles de los archivos de texto.
- LICENSE.txt provee información sobre los términos de utilización de la base de datos.

**Indian Movie Face database (IMFDB)**

Colección de 34,512 imágenes de 100 actores Indios y extraídas de más de 100 videos, estas imágenes fueron seleccionadas y cortadas manualmente de distintos cuadros de cada video, lo que resulta en un alto grado de variabilidad en términos de escala, pose, expresiones, iluminación, edad, resolución, oclusión y maquillaje. IMFDB contiene para cada imagen anotaciones detalladas acerca de la edad, pose, género, expresión y tipo de oclusión, los cuales detallamos a continuación («Indian Movie Face database (IMFDB)», s. f.):

1. Expressions : *Anger, Happiness, Sadness, Surprise, Fear, Disgust*
2. Illumination : *Bad, Medium, High*
3. Pose : *Frontal, Left, Right, Up, Down*
4. Occlusion : *Glasses, Beard, Ornaments, Hair, Hand, None, Others*
5. Age : *Child, Young, Middle and Old*
6. Makeup : *Partial makeup, Over-makeup*
7. Gender : *Male, Female*

Su contenido es accesible para uso no comercial en el enlace citado («Enlace de descarga de IMFDB», s. f.)

**10k US Adult Faces Database**

Conjunto de 10,168 imágenes generada a partir de una muestra aleatoria extraída de Google Imágenes y nombres generados aleatoriamente en base a la distribuciones de nombre en los EEUU según el censo de 1990. Debido a esta metodología utilizada, la distribución de las caras coincide con la distribución demográfica de los EEUU (por ejemplo, edad, raza y género).
Esta base de datos contiene además una amplia gama de caras, cada una es rodeada por un óvalo para eliminar los efectos de fondo, además se tiene para un conjunto aleatorio de 2.222 de las caras información demográfica y se atribuyen coordenadas para que investigadores puedan crear sus propios conjuntos de expresiones faciales.

Su contenido es accesible únicamente con motivos de investigación mediante la solicitud de acceso mediante correo electrónico, donde posteriormente se suministran las credenciales para poder disponer acceso a la misma. («10k US Adult Faces Database», s. f.).

**Face Recognition Data, University of Essex**

Conjunto de 7900 imágenes de 395 individuos, con 20 fotografías por cada uno de ellos. Este contiene imágenes de ambos géneros, variedad de etnias y la mayoría de las fotografías son de personas entre 18 y 20 años de edad. Sin embargo no posee metadatos generados sobre estas características, sino que únicamente se tiene separado el género por carpetas. Su contenido es accesible únicamente con motivos de investigación en la cita.(«Face Recognition Data, University of Essex», s. f.)

**Color FERET**

Conjunto de 14,126 imágenes de 1199 individuos recolectadas de un ambiente semi controlado, en donde se tomaron fotografías en lapsos de 2 años para algunas personas a fines de poder estudiar los cambios en la apariencia de los sujetos a través de los años.

Su contenido es accesible únicamente con motivos de investigación mediante la solicitud de acceso por correo electrónico, donde posteriormente se suministran las credenciales para poder disponer acceso a la misma. («The Color FERET Database», s. f.)

**LFW (Labeled Faces in the Wild)**

Conjunto de 13,000 imágenes recolectadas de la web, cada cara fue etiquetada con el nombre de la persona. 1680 personas poseen 2 o más fotografías en el conjunto, como desventaja las caras fueron detectadas por el detector de rostros Viola Jones. Su contenido es accesible en el enlace citado («Labeled Faces in the Wild Home», s. f.)

**FG NET (Face and Gesture Recognition Working Group)**

Base de datos de 1,002 imágenes de Flickr de 82 sujetos para la estimación de la edad en personas, para evaluar la performance. Cada imagen está etiquetada dentro de un rango de 7 categorías de edad: 0-2,3-7,8-12, 13-19, 29-36,37-65 y 66+. («FG-NET Human Age Estimation», s. f.) («Suggested databases for age estimation - answers.opencv.org», s. f.).

**The Images of Groups Dataset**

Colección de 5,080 imágenes extraídas de Flickr, etiquetadas con género y edad para cada fotografía. Se utilizaron las siguientes categorías de clasificación etaria: 0-2, 3-7, 8-12, 13-19, 20-36, 37-65, and 66+.

Su contenido es accesible para uso no comercial en el link citado. («The Images of Groups Dataset», s. f.).

**Multi PIE**

Conjunto de 750,000 imágenes pertenecientes a 337 individuos, tomadas desde 15 distintos puntos de vista y bajo 19 tipos de iluminación, su contenido es accesible por medio de pago y bajo la licencia Carnegie Mellon University para propósitos de investigación interna. Su tamaño es de 308 GB, por lo cual no está disponible su descarga, sino que se envía una memoria USB al solicitante.

**AR Face Database**

Conjunto de 4,000 imágenes tomadas de 126 individuos en un ambiente controlado, cada imagen posee metadata sobre el género, su contenido es accesible de forma gratuita únicamente para propósitos académicos en el enlace citado. («AR Face Database Webpage», s. f.)

Luego de una revisión de los distintos conjuntos de datos disponibles decidimos trabajar con el conjunto MORPH, ya que además de cumplir con los metadatos requeridos como género, etnia y edad, posee una gran cantidad de imágenes e individuos con respecto a las otros, así como también ha sido de las más referenciadas en las investigaciones que hemos estudiado y nos posibilita la opción de adquirir su versión comercial.

Decidimos utilizar la base de datos MORPH para realizar experimentos preliminares con los algoritmos que implementamos en Matlab a fin de comparar nuestros resultados con otras técnicas que hayan usado la misma base de datos. La elección se debió principalmente al gran número de imágenes que brinda, la calidad de las mismas y a los metadatos que posee, en este caso estamos interesados en el género y la edad. En el caso del género, utilizamos una base de datos con igual cantidad de hombres como de mujeres para equilibrar las clases. No se realizó así para el caso de la edad debido a que algunas clases tenían muy pocas muestras, por lo cual la confianza de la clasificación de ciertas clases con pocas muestras de entrenamiento serán menos confiables. Este desbalance de clases en cuanto a la edad también implica que la clasificación de género en personas cuya edad es pobremente representada en la base de datos sea menos confiable.

Para el prototipo final se combinó MORPH, FERET y 10K para aumentar la cantidad de datos de entrenamiento de los clasificadores para maximizar la precisión, filtrando las imágenes por raza caucásica, teniendo en cuenta que este tipo de población es amplia mayoría en el escenario donde se va a aplicar, y el hecho de restringir la raza disminuye la variabilidad entre los rostros, lo cual simplifica el problema de clasificación. El sistema puede adecuarse a otro tipo de población simplemente modificando las imágenes de entrenamiento.

## Pruebas y resultados preliminares

A continuación se describen los experimentos realizados en las distintas etapas de la solución.

### Extracción de características

En esta etapa se estudiaron varias técnicas y finalmente se seleccionaron y probaron dos: LBP (Local Binary Patterns) y PHOG (Pyramid Histogram of Oriented Gradients).

Éstas fueron implementadas en Matlab, extrayendo vectores de características de imágenes de la base de datos MORPH y utilizando éstos como entrada de distintos tipos de clasificadores utilizando el software Weka («Weka 3 - Data Mining with Open Source Machine Learning Software in Java», s. f.).

Se demostró que tanto LBP como PHOG (y otras variantes de HOG) funcionan muy bien para la clasificación de género. Por otro lado, encontramos que métodos similares de extracción de características se utilizaron para clasificación de edad (por ejemplo (Ylioinas et al., 2012), e incluso en simultáneo para edad y género utilizando las mismas características (ejemplo (Eidinger et al., 2014), por lo que encontramos muy conveniente probar el mismo vector para estimar la edad, reduciendo costos de cómputo y produciendo resultados satisfactorios. El método PHOG requiere definir ciertos parámetros que se detallan en la sección Diseño, estos fueron seleccionados probando distintas combinaciones y midiendo los resultados que producía cada clasificador.

### Clasificación

En cuanto a la clasificación, encontramos al revisar el estado del arte que tanto para género como para edad, el problema se resuelve utilizando clasificadores como SVM (Support Vector Machines) o Redes Neuronales, además de que los métodos de extracción de características que seleccionamos fueron diseñados para ser utilizados junto con estos clasificadores, con los cuales produjeron muy buenos

resultados. Sin embargo, decidimos añadir otro tipo de clasificadores a las pruebas, principalmente como forma de validación pues si el problema de clasificación está bien formulado, al probar un conjunto de clasificadores distintos, estos deben producir resultados similares, teniendo unos pocos que sobresalgan. Si se ven resultados muy dispares, es muy probable que el problema esté mal formulado.

Se comparó la precisión al extraer características con LBP utilizando la base de datos MORPH con varias imágenes por persona, la cual cuenta con un total de 5000 imágenes, alcanzando 94% con parámetros C=8 y Gamma=16. Este fue menor al obtenido por PHOG (97%) utilizando la misma base de datos y parámetros C=200 y Gamma=64, por lo que decidimos utilizar este último. Vale aclarar que, al tener varias tomas por persona, se cuenta con más información para cada persona y cuando se separan los datos para entrenamiento y pruebas es muy probable que la misma persona que se usó en el conjunto de entrenamiento aparezca en el de pruebas, por esto las pruebas se repitieron manteniendo una sola toma por persona. Sin embargo, consideramos que estas pruebas fueron suficientes para determinar el mejor rendimiento de PHOG por sobre LBP para este problema.

La base de datos que se usó para clasificar género se basa en la base de datos Morph original pero dejando un solo rostro por persona y balanceando la cantidad de mujeres y hombres en la misma lo cual hace que disminuya su tamaño a un total de 1264 imágenes. Otra aclaración es que se usaron los siguientes parámetros de PHOG: L = 3; K = 16. Esto significa que se utilizó una pirámide de 3 niveles, y que cada histograma de gradientes tiene un largo de 16, siendo estos los parámetros que mejores resultados produjeron. Por otro lado, el filtro LoG se estima con un kernel de 7x7 y un sigma de 3. Los resultados óptimos que se obtuvieron al entrenar con los distintos clasificadores para PHOG por género se indican en la siguiente gráfica:

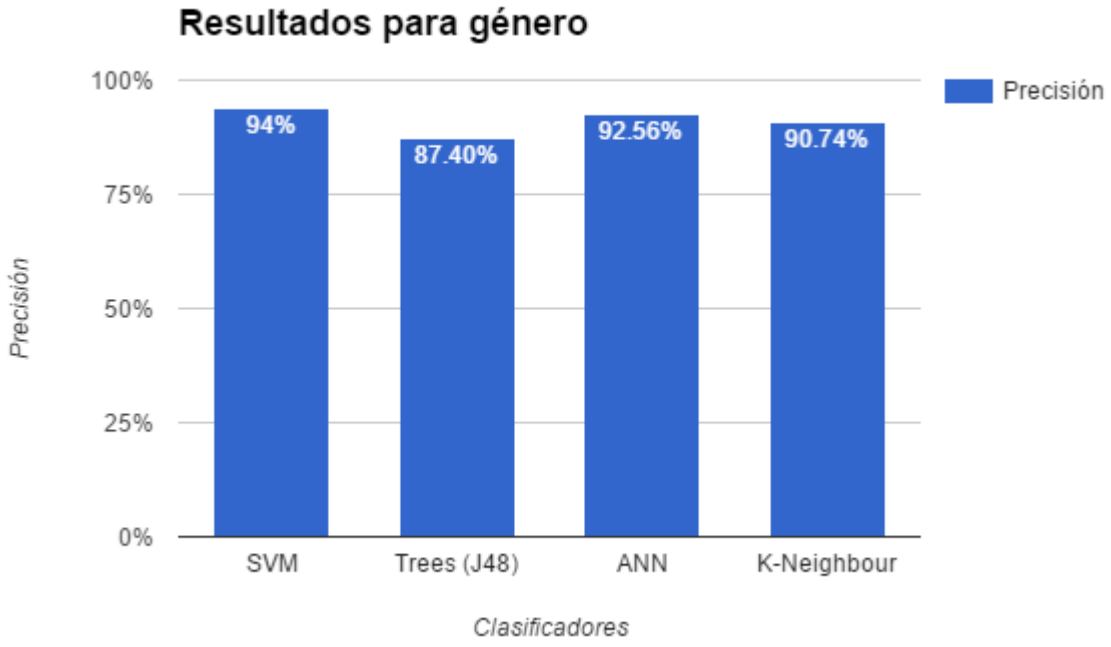



Fig. 13

A continuación se presenta en detalle la precisión para cada clase para el mejor clasificador:

| Clase | Verdadero Positivo | Falso Positivo | Precisión |
|---|---|---|---|
| **Masculino** | 94% | 6% | 94% |
| **Femenino** | 94% | 6% | 94% |
| **Promedio ponderado** | 94% | 6% | 94% |

Como puede verse, la confianza de la clasificación es la misma tanto si el resultado es masculino como femenino.

Los detalles de los parámetros utilizados para cada clasificador fueron los siguientes:

| Algoritmo | Parámetros | | | | | | | Precisión |
|---|---|---|---|---|---|---|---|---|
| | C | Gamma | | | | | | |
| SVM | 132 | 2 | | | | | | 94% |
| | C | M | | | | | | |
| Trees (J48) | 0.1 | 2 | | | | | | 87,84% |
| | L | M | N | V | S | E | H | |
| ANN | 0.3 | 0.2 | 500 | 0 | 0 | 20 | 50 | 92,56% |
| | K | | | | | | | |
| K-Neighbour | 5 | | | | | | | 90,74% |

La base de datos que se usó para clasificar edad también se basa en la base de datos Morph original pero dejando un solo rostro por persona y sin balancear la cantidad de mujeres y hombres en la misma lo cual hace que su tamaño sea un poco mayor a la de género, en concreto con un total de 2686 imágenes.

Los resultados óptimos que se obtuvieron al entrenar con los distintos clasificadores para PHOG para el caso de edad se aprecia en el siguiente gráfico:

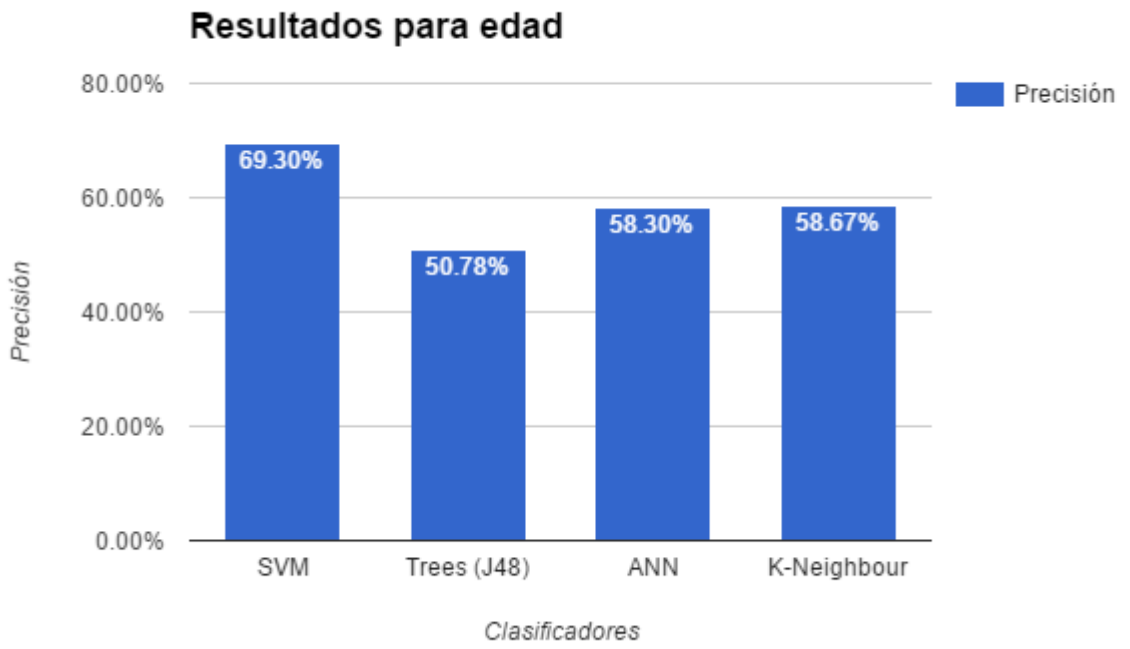



Fig. 14

A continuación se presentan los resultados de la clasificación desglosado por cada clase:

| Clase | Verdadero Positivo | Falso Positivo | Precisión | Recall |
|---|---|---|---|---|
| **Mayor a 57** | 6.5% | 3% | 30% | 6.5% |
| **44 a 56** | 41.4% | 6.5% | 57.1% | 41.1% |
| **33 a 43** | 65.6% | 24.7% | 59.3% | 65.6% |
| **16 a 32** | 84.8% | 17% | 80.7% | 84.8% |
| **Promedio ponderado** | 69.2% | 17.6% | 68.2% | 69.2% |

A diferencia de la clasificación por género, esta tabla ofrece información más interesante. En primer lugar notar que los resultados para las edades mayores son muy pobres porque no están bien representadas en las bases de dates usadas (ni en ninguna de las estudiadas) por lo que la clasificación en dichos rangos etarios será menos precisa.

Sin embargo, dada una clasificación emitida por el SVM, la precisión es mayor al 50% en todos los rangos etarios salvo el mayor, un resultado que se equipara a los encontrados en el estado del arte.

Por otro lado, la probabilidad de acertar dada una cierta clase (conocido como *Recall*) es mayor al 65% para los tres rangos etarios más jóvenes, lo cual es una precisión muy buena teniendo en cuenta que el sistema real tomará varias tomas de cada persona. Dicho esto, entendemos que el rendimiento será muy superior en escenarios donde predominen personas de edades menores a 44 años.

Los rangos no incluyen edades menores a 16 años por no disponer de imágenes para tales edades.

El rendimiento general puede mejorarse tanto al combinar varias bases de datos y obtener las muestras faltantes en los rangos de peor rendimiento, como al combinar los dos rangos de mayor edad en uno solo para obtener más muestras para la misma clase, además de facilitar el problema de clasificación. Esto se discute en la sección Clasificación del capítulo Diseño.

Los detalles de los parámetros utilizados para cada clasificador fueron los siguientes:

| **Algoritmo** | **Parámetros** | | | | | | | **Precisión** |
|---|---|---|---|---|---|---|---|---|
| | **C** | **Gamma** | | | | | | |
| **SVM** | 113 | 6 | | | | | | **69,3%** |
| | **C** | **M** | | | | | | |
| **Trees (J48)** | 0.1 | 2 | | | | | | **50,78%** |
| | **L** | **M** | **N** | **V** | **S** | **E** | **H** | |
| **ANN** | 0.3 | 0.2 | 500 | 0 | 0 | 20 | 50 | **58,30%** |
| | K | | | | | | | |
| **K-Neighbour** | 5 | | | | | | | **58,67%** |

Para cada algoritmo se configuraron distintos parámetros que se describen a continuación:

- Para SVM:
  - C: costo.
  - G: gamma.
- Para Trees (J48)
  - C: umbral de confianza para la poda.
  - M: cantidad mínima de instancias por hoja.
- ANN
  - L: Tasa de aprendizaje del algoritmo *backpropagation*
  - M: Momentum
  - N: Cantidad de épocas
  - V: porcentaje del conjunto de validación, se usa para terminar el entrenamiento
  - S: semilla para el generador de números aleatorios.
  - E: cantidad de errores consecutivos que se permiten durante las pruebas de validación antes de que la red termine.
  - H: cantidad de capas ocultas.
- K-Neighbor
  - K: cantidad de vecinos más cercanos.

A continuación se explica cómo se obtuvieron los valores de los parámetros para cada uno de los clasificadores:

  - *ANN*
  - *Trees*
  - *K-Neighbour*

- Para cada método se obtuvieron los parámetros óptimos utilizando distintas técnicas para cada caso y por último ejecutando *10-Fold Cross Validation* con los mejores parámetros para conseguir la precisión en cada caso:

    - En *ANN* se varió la cantidad de capas ocultas y se determinó que el resultado óptimo fue con 50 capas.
    - En *Trees* se realizaron varias pruebas en un intervalo de confianza de 0.1 a 0.5 en pasos de tamaño 0.1 (5 etapas) determinando el mejor intervalo de confianza.
    - En *K-Neighbour* se realizaron varias pruebas con K=1, K=5 y K=10 determinando el parámetro K más apropiado.
- Para estos 3 clasificadores se utilizó el meta clasificador de weka CVParameterSelection, mientras que para SVM se realizó una búsqueda telescópica en el espacio de los parámetros gamma y costo mediante el meta clasificador GridSearch de la siguiente forma:
    - GridSearch trabaja sobre una grilla en donde se busca la combinación gamma-costo que ofrece la mejor precisión para la clasificación.
    - La idea de la búsqueda telescópica es iterar sobre una ventana dentro de la grilla lo suficientemente grande y con saltos largos en las iteraciones para reducir la duración de las pruebas encontrando hacia qué valores tiende la combinación gamma-costo más óptima.
    - Una vez que se encuentra esta combinación se realiza una nueva búsqueda centrando una ventana más pequeña en ese valor y se realizan saltos más cortos en las iteraciones. Esto se repite hasta que la mejora en la precisión se hace despreciable.

Por otro lado, también investigamos en el área de Deep Learning. La principal ventaja de este modo de aprendizaje es que no requiere extracción de características, ya que la red “aprende” por si sola las características necesarias, siendo que las neuronas de las primeras capas se especializan en detectar elementos de más bajo nivel como bordes, y las neuronas de capas más profundas se activan cuando la entrada corresponde a cierta clase. La motivación para investigar esta área fue el auge que tuvo este método de aprendizaje en los últimos años y el éxito que tuvieron compañías como Google utilizando Deep Learning, además de la amplia disposición de herramientas para entrenar redes neuronales convolutivas. Sin embargo, al realizar pruebas nos encontramos con problemas de tiempo de cómputo de la salida de la red neuronal, lo cual lo hacía inviable para una aplicación en tiempo real. Por otro lado, se probó la clasificación de género obteniendo resultados similares a los que se consiguen mediante métodos convencionales. Finalmente se pudo concluir que para que Deep Learning supere el rendimiento de los métodos clásicos, el tamaño del conjunto de datos de entrenamiento debe ser de órdenes de magnitud más grande que los que se cuentan.

En resumen, las pruebas indicaron que el mejor rendimiento se obtuvo utilizando PHOG para extraer características, con una función gaussiana de gamma 3, a cuyo resultado se le calcula el Laplaciano, ambas operaciones realizadas con una única máscara de 7x7, y utilizando un clasificador SVM que resulta ser el clasificador empleado en el paper original que propone el método de extracción de características PHOG. Cabe destacar que los parámetros SVM fueron estimados para medir su rendimiento óptimo mediante Weka a partir de los vectores PHOG construidos con Matlab, pero deberán ser buscados nuevamente en la implementación en C++, debido a que la extracción de características será levemente diferente (tanto por implementación de los algoritmos de OpenCV como por diferencias del lenguaje) y el conjunto de entrenamiento utilizado será mayor, al combinar varias bases de datos de imágenes.

**Tracking**

El seguimiento de rostros surge de la necesidad de poder determinar la cantidad de personas que concurren a un determinado lugar además de mejorar la performance de nuestro sistema al no tener que realizar detección de rostros cuadro por cuadro. También obtener una mejor precisión de la clasificación de una persona al correlacionar varias caras con ésta, ya que al disponer información de un sujeto a lo largo de varios cuadros, nos permitirá determinar con mayor aproximación su edad y género, al aplicar la mediana de cada conjunto.

Para poder llevar a cabo dicho cometido, el algoritmo a implementar en nuestra solución debía cumplir con las siguientes características:

- Funcionar en tiempo real.
- Estar enfocado a seguimiento de personas.
- Robusto ante problemas de oclusión y ambientes no controlados.
- Fácil de integrar a nuestra solución y flexible.

Luego de investigar cuales eran los algoritmos más utilizados para resolver esta problemática, seleccionamos los siguientes:

  - API OpenCV
  - CamShift
  - MeanShift
  - Kalman Filter
  - Optical Flow

En primera instancia a modo de simplificar la complejidad de la investigación, decidimos investigar si las herramientas que ofrece por defecto OpenCV contaba con algoritmos para resolver este problema, encontrándonos que efectivamente este dispone de una API para realizar *multitracking*, haciendo disponible diferentes tipos

de algoritmos para dicho cometido. Luego de varias pruebas variando los distintos algoritmos que este disponía y sometiéndolo al escenario planteado, observamos que la performance disminuía a medida que la cantidad de objetos a realizar seguimiento en escena aumentaba, lo cual no cumplía con parte de los requisitos planteados. Además los algoritmos utilizados estaban más enfocados al seguimiento de objetos en general y no particularmente a personas. Por último, una gran desventaja que nos enfrentamos al utilizar este tipo de implementación, fue que al no tener control ninguno de los algoritmos, nos imposibilitaba tener una estructura de personas que pudiéramos manejar para mejorar la precisión en la clasificación de cada una. («OpenCV: cv::MultiTrackerTLD Class Reference», s. f.)

Al no tener éxito con la metodología anterior procedimos a estudiar MeanShift (Cheng, 1995) y CamShift (Bradski, 1998) que consiste en convertir la imagen en un espacio HSV (Hue,Saturation,Value), con la idea de reducir los efectos del brillo y la luz. De esta forma podemos determinar por ejemplo qué tan separado está el color de la saturación, es decir cómo se concentra el color.

Luego, a partir de la región de interés a seguir se crea el histograma tomando el canal H del espacio HSV, para obtener la probabilidad de distribución del color de cada cara.

Durante cada iteración, se utiliza el histograma guardado anteriormente para cada cara como modelo, o *lookup table*, y se convierte el cuadro actual en una imagen de probabilidad de colores, que luego utilizando dicha probabilidad, se opera sobre la distribución de la imagen derivada del histograma de color, calculando el centroide y re centrando la ventana de seguimiento, para finalmente determina el área para el siguiente tamaño de ventana.

Fig. 15

La principal diferencia entre CamShift y MeanShift es que este último no adapta el tamaño de la ventana cuando el objeto se acerca hacia la cámara o se aleja, como se puede ver en la figura 15. Por tanto, CamShift es una variación de MeanShift que resuelve este problema y que no solo adapta el tamaño de la ventana sino que también tiene en cuenta la rotación del objeto.

Por otro lado, cuando el brillo es bajo, es decir V es cercano a 0 y la saturación también es baja, es decir S es cercano a 0, la matriz se vuelve bastante ruidosa, ya que los píxeles de tono discretos no pueden representar cambios ligeros en el espacio RGB. Para solucionar esto, lo que se hace es ignorar píxeles que tengan brillo bajo, lo que significa que para cada escena, la cámara se debe calibrar para tener más brillo. Sin embargo, la luz del sol o de la escena puede provocar que los colores blanco brillantes puedan adquirir un tono de piel humana, por lo que para ello se establece un umbral de manera de descartar falsos pixeles y pixeles con altos niveles de brillo.

Si bien la performance del algoritmo es muy buena y a simple vista parecía cumplir con todas nuestras expectativas, resultó no funcionar bien para escenarios con mucho público en movimiento, debido principalmente a la dificultad de calibrar los niveles de brillo para la escena, como podemos apreciar a continuación (Bradski, 1998).

Fig. 16: escenario controlado con fondo blanco detrás

Fig. 17: escenario no controlado, con mucho público y fondo con altos niveles de brillo

En vista de este problema se investigó Kalman filter. Este método realiza una predicción óptima de la próxima ubicación de un objeto minimizando el error de la covarianza estimada. Utiliza un conjunto de ecuaciones para la predicción y otro para la corrección con las que va actualizando los valores de las variables involucradas a lo largo de sucesivas iteraciones.

**Conjunto de ecuaciones para la predicción**:

1) $$x_t = F_t x_{t-1} + B_t u_t + w_t$$

Donde $x_t$ es el estado del vector que contiene los términos de interés del sistema (en nuestro caso, posición y velocidad), $u_t$ guarda entradas de control y $B_t$ es la matriz de control de la entrada la cual aplica el efecto de cada parámetro de control del vector $u_t$ (esto no aplica para nuestro problema). $F_t$ es la matriz de transición que aplica el efecto del vector que representa el estado en el tiempo t-1 para formar el estado del sistema en el tiempo t, básicamente consiste en aplicar un movimiento a velocidad constante durante una unidad de tiempo. $w_t$ es el vector que contiene valores de ruido que se suman al vector de estados resultante.

2) $$P_t = F_t P_{t-1} F_t^T + Q_t$$

Donde $Q_t$ es la matriz de covarianza asociada a las entradas de control con ruido. $P_t$ es otra matriz de covarianza que almacena en su diagonal las varianzas asociadas a los correspondientes términos del vector de Estados y en el resto de las celdas a las covarianzas entre los términos del mismo vector.

**Conjunto de ecuaciones para la corrección:**

Ganancia de Kalman:

1) $$K_t = P_t H_t^T \left(H_t P_t H_t^T + R_t\right)^{-1}$$

Donde $H_t$ es la matriz de transformación que mapea los parámetros del vector de estados con el dominio de medición. $R_t$ es otra matriz de covarianza referida al ruido blanco Gaussiano. Esto sirve para implementar la corrección del estado previamente predicho.

Actualizar la estimación mediante $z_t$:

2) $$x_t = x_{t-1} + K_t (z_t - H_t x_t)$$

Donde $z_t$ es el vector que almacena las mediciones de los parámetros del sistema.

Actualizar el error en la matriz de covarianza:

3) $P_t = P_t - K_t H_t P_t$

Para adaptarlo a nuestro problema, utilizamos un modelo de predicción de movimiento en el que los rostros se desplazan a velocidad constante, donde el estado son los vectores de posición y velocidad en un momento dado. Por otro lado, como corrección se tomó la posición del rostro que predijo el modelo y se aplicó un detector de rostros localmente para corregir la misma.

Kalman Filter no alcanzó los resultados esperados fallando ante cambios bruscos de velocidad u oclusiones del rostro. A pesar de ser un método robusto para el seguimiento de objetos (Faragher, 2012), no se mostró así para seguir rostros en los escenarios pretendidos.

El método que consiguió mejores resultados fue un Feature Tracker basado en Flujo Óptico. Este se basa en construir un campo vectorial sobre el video que brinde información sobre el aparente movimiento de los objetos de una escena. Luego se intenta aproximar el movimiento entre un rostro en tiempo $t$ y otro en tiempo $t + \Delta t$. Este problema se plantea como sigue:

Para una imagen I(x,y,t), suponemos una *restricción de constancia de intensidad*

$$I(x, y, t) = I(x + \Delta x, y + \Delta y, t + \Delta t)$$

Asumiendo que el movimiento en el video será pequeño para un $\Delta t$ suficientemente pequeño. Luego expandimos esta ecuación por series de Taylor:

$$I(x + \Delta x, y + \Delta y, t + \Delta t) = I(x, y, t) + \frac{\delta I}{\delta x}\Delta x + \frac{\delta I}{\delta y}\Delta y + \frac{\delta I}{\delta t}\Delta t$$

Donde se descartan los términos de mayor orden. De esto sigue que:

$$\frac{\delta I}{\delta x}\Delta x + \frac{\delta I}{\delta y}\Delta y + \frac{\delta I}{\delta t}\Delta t = 0$$

Dividiendo entre Δt:

$$\frac{\delta I}{\delta x}\frac{\Delta x}{\Delta t} + \frac{\delta I}{\delta y}\frac{\Delta y}{\Delta t} + \frac{\delta I}{\delta t}\frac{\Delta t}{\Delta t} = 0$$

Esto resulta en:

$$\frac{\delta I}{\delta x} V_x + \frac{\delta I}{\delta y} V_y + \frac{\delta I}{\delta t} = 0$$

Con Vx y Vy los componentes horizontal y vertical de la velocidad o flujo óptico respectivamente de I(x,y,t). Esta ecuación puede reescribirse como:

$$\nabla I^T \cdot V = -I_t$$

Siendo $I_t$ la derivada parcial de I respecto a t y V el vector correspondiente con componentes Vx y Vy . Esta ecuación tiene dos incógnitas, por tanto no tiene una solución única y no puede resolverse por si sola. Esto se conoce como el *problema de la apertura*. Para hallar el flujo óptico (léase $V_x$ y $V_y$), deben introducirse otro conjunto de ecuaciones al introducir más restricciones. Para esto utilizamos el método de Lucas-Kanade (Lucas & Kanade, 1981):

Para cada pixel que se quiera calcular el flujo óptico, se toma una vecindad de 3x3, asumiendo que todos se mueven de igual forma. Entonces, para cada punto se puede calcular $I_x$ , $I_y$ y $I_t$ . Como asumimos que $V_x$ y $V_y$ es igual para cada pixel del vecindario, el problema se transforma en resolver 9 ecuaciones con 2 incógnitas, el cual está determinado.

Se pueden obtener resultados más exactos para este mismo problema utilizando mínimos cuadrados, que resulta en el siguiente sistema de 2x2:

$$\begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} \sum_i I_{x_i}^2 & \sum_i I_{x_i} I_{y_i} \\ \sum_i I_{x_i} I_{y_i} & \sum_i I_{y_i}^2 \end{bmatrix}^{-1} \cdot \begin{bmatrix} -\sum_i I_{x_i} I_{t_i} \\ -\sum_i I_{y_i} I_{t_i} \end{bmatrix}$$

Para determinar sobre qué puntos calcular el flujo óptico y construir dicho campo vectorial se utilizaron características propias de las caras detectadas, en este caso se utiliza una función de OpenCV llamada goodFeaturesToTrack() la cual determina las esquinas más destacadas en una región particular de la imagen. Para conseguir esto lleva a cabo los siguientes pasos:

1 - Calcula la calidad de cada esquina en cada pixel de la imagen a través de alguna de las funciones: cornerMinEigenVal() ó cornerHarris(). La primera es la que se usa por defecto y básicamente para cada pixel p, esta función considera una ventana de cierto tamaño en la que calcula la matriz de covarianza de las derivadas de todos sus vecinos S(p):

$$M = \begin{bmatrix} \sum_{S(p)} (dI/dx)^2 & \sum_{S(p)} (dI/dx dI/dy) \\ \sum_{S(p)} (dI/dx dI/dy) & \sum_{S(p)} (dI/dy)^2 \end{bmatrix}$$

Aquí las derivadas se calculan mediante Sobel y luego se extraen los valores y vectores propios de la matriz M. Por último la calidad resulta del mínimo entre los valores propios hallados $\lambda_1$y $\lambda_2$. La función cornerHarris() consiste en realizar un procedimiento muy similar.

2- Se aplica una función que descarta máximos locales en vecinos dentro de un rango de 3x3.

3- Se rechazan las esquinas con valor propio mínimo menor al siguiente producto:

$$\mathtt{qualityLevel} \cdot \max_{x,y} \mathtt{qualityMeasureMap}(x,y)$$

Donde qualityLevel es un parámetro de tolerancia y $\max_{x,y}$ qualityMeasureMap(x,y) es la mayor calidad de entre todas las esquinas.

4- Las esquinas restantes son ordenadas en orden descendente en base a la medida de calidad.

5- Si se detectó una esquina más fuerte que otra en una distancia menor al parámetro *maxDistance,* se descarta la más débil. («Feature Detection — OpenCV 2.4.13.0 documentation», s. f.)

Fig. 18

Finalmente luego de las pruebas realizadas, se encontró que esta técnica cumplía con la mayoría de los requisitos planteados, la performance era óptima, funciona bien en ambientes no controlados y controlados, sin embargo no era robusto ante problemas de oclusión, así como también el cálculo de la ventana de seguimiento no era tenido en cuenta en este método. Pese a esto, prometía buenos resultados. Los detalles de su implementación y la solución a dichos problemas se tratarán más adelante. En cuanto a la detección facial, se utilizó el algoritmo de (Viola & Jones, s. f., p.)

# Diseño

El sistema, fue desarrollado en C++ utilizando OpenCV 3.0 en Microsoft Visual Studio. Cuenta con los siguientes módulos: detección facial y seguimiento, preprocesamiento y normalización, extracción de características y clasificación. Su diseño es modular para simplificar la sustitución de sus componentes en caso de ser necesario y para poder probar nuevas técnicas en busca de un mejor rendimiento.

### Normalización y preprocesamiento

En la etapa de preprocesamiento y normalización se reescala la imagen del rostro utilizando una interpolación bicúbica y se transforma a escala de grises tomando únicamente el canal verde de la imagen, lo cual ahorra tiempo de cómputo frente al método clásico de pasaje a escala de grises. El método clásico se calcula como sigue:

$$Y = 0.2126R + 0.7152G + 0.0722B$$

Siendo R, G y B los valores correspondientes a cada canal RGB. Esto implica calcular dos sumas y tres productos, y teniendo en cuenta que un video estándar tiene 30 cuadros por segundo y varios rostros por cuadro, optimizaciones de este tipo impactan notoriamente en el rendimiento del sistema.

### Extracción de características

Para la extracción de características se utilizó el método PHOG. A continuación explicamos su funcionamiento y detalles de implementación:

1. Se reescala la imagen a 250x250 píxels utilizando interpolación bicúbica si la imagen es de menor tamaño. Esta transformación funciona como un filtro pasobajo que filtra los componentes de alta frecuencia, logrando disminuir el ruido, suavizar la imagen y dejarla en una escala similar a la de las imágenes de entrenamiento.
2. Si la imagen es mayor a dicho tamaño, se reescala a 250x250 píxels, pero en lugar de interpolar, se re-muestrea usando la relación entre áreas de píxels como método de antialiasing.
3. Se aplica el filtro LoG (Laplacian of Gaussian) a la imagen en grises de entrada utilizando el siguiente kernel:

| | | | | | | |
|---|---|---|---|---|---|---|
| 0.0048 | 0.0088 | 0.0120 | 0.0128 | 0.0120 | 0.0088 | 0.0048 |
| 0.0088 | 0.0132 | 0.0082 | 0.0023 | 0.0082 | 0.0132 | 0.0088 |
| 0.0120 | 0.0082 | -0.0232 | -0.0471 | -0.0232 | 0.0082 | 0.0120 |
| 0.0128 | 0.0023 | -0.0471 | -0.0829 | -0.0471 | 0.0023 | 0.0128 |
| 0.0120 | 0.0082 | -0.0232 | -0.0471 | -0.0232 | 0.0082 | 0.0120 |
| 0.0088 | 0.0132 | 0.0082 | 0.0023 | 0.0082 | 0.0132 | 0.0088 |
| 0.0048 | 0.0088 | 0.0120 | 0.0128 | 0.0120 | 0.0088 | 0.0048 |

4. De la imagen resultante se calcula, para cada punto, el vector gradiente. Los vectores de cada punto se almacenan como dos matrices, donde cada entrada es el módulo y la orientación del gradiente respectivamente.
5. Ambas imágenes se dividen en una grilla de $2^l$ x $2^l$ celdas.
6. Se divide el rango $[0, \pi]$ en k orientaciones.
7. Para la celda actual, se crea un vector de largo k+1. Cada entrada representa una posible orientación de gradientes.
8. Para cada gradiente en dicha celda, se toma su orientación $\theta$ y se distribuye el valor de su módulo linealmente entre las dos entradas del vector que correspondan a las orientaciones más cercanas a $\theta$ , en función de la diferencia entre $\theta$y las orientaciones representadas en el vector.
9. Avanzar una celda y volver a 7 hasta cubrir toda la grilla actual.
10. Se debe repetir el proceso para una grilla de $2^{l-1}$ x $2^{l-1}$. En lugar de esto, aprovechamos la linealidad del problema y construimos el vector de cada una de las $2^{l-1}$ x $2^{l-1}$ celdas como la suma de los 4 vectores de las celdas que abarcan la misma región en la grilla de $2^l$ x $2^l$. El mismo concepto aplica para los sucesivos valores de *l*

11. Finalmente se concatenan todos los vectores en un único vector de histogramas y se normaliza el mismo dividiéndolo entre la norma L2.

**Clasificación**

Para la clasificación, se utilizarán clasificadores SVM por ser los que mejor resultados produjeron. Sin embargo, debido a que para aumentar la precisión decidimos combinar las bases de datos MORPH, FERET y 10K, los rangos etarios utilizados en los experimentos preliminares con MORPH debieron ser modificados,

debido a que las bases de datos añadidas no tienen información exacta sobre edades, sino que están indicadas en rangos, obligándonos a adaptarnos a ellos. Debido al problema detectado en la clasificación de los rangos etarios de mayor edad, causada por las pocas muestras disponibles para dichos rangos, decidimos unir los dos rangos de mayor edad y trabajar con 3 clases en lugar de 4:

- Entre 16 y 30 años
- Entre 30 y 45 años
- Más de 45 años

De esta forma, pretendemos mejorar la precisión del clasificador para las personas de mayor edad al combinar tres bases de datos, y tomar a las personas mayores a 45 años como una única clase, en lugar de separarlas en dos.

Para estimar el género y la edad de cada rostro, se utiliza el mismo vector de características extraído una única vez para cada toma de cada rostro. De esta forma se logra optimizar el tiempo de cómputo y lograr que el sistema funcione en tiempo real sin tener altos requerimientos de hardware.

Estos clasificadores fueron entrenados teniendo en cuenta el escenario en que serán aplicados y adaptándolos al mismo. Esto implica, principalmente, tomar provecho del tamaño promedio que tienen los rostros detectados en el escenario considerado, para reescalar las imágenes de entrenamiento a un tamaño similar (para la secuencia considerada en el presente trabajo, se tomó un tamaño de 250x250), de lo contrario la performance del mismo se vería disminuida. Por otro lado, el video de entrada no puede tener una resolución superior a HD (720 x 1280 píxels), de lo contrario el rendimiento cae y pierde la capacidad de correr en tiempo real. Esto ocasiona que los rostros capturados tengan una calidad notoriamente inferior a las imágenes de entrenamiento y reduce considerablemente la precisión de los clasificadores. Para mitigar este efecto, se extraen características de las imágenes de entrenamiento con PHOG utilizando un kernel LoG con un suavizado gaussiano más intenso que el aplicado a los rostros del video, lo cual hace que los rostros de entrenamiento pierdan definición y se asemejen más a los rostros detectados en el video.

Por último, debido a estos cambios, los parámetros C y  de los clasificadores SVM debieron ser buscados nuevamente. Para maximizar el rendimiento, se tomó como conjunto de test una secuencia de video, se extrajeron los rostros y se tomaron los parámetros que brindaron mayor precisión. Luego el sistema se probó en una secuencia distinta, pero filmada bajo idénticas condiciones, obteniendo un resultado satisfactorio.

Este proceso de adaptación implica que, si el escenario en el que el sistema funciona fuera cambiado a condiciones distintas, sería necesario repetirlo para adaptar las distintas variables del mismo a dicho escenario, aprovechando la información previa que se disponga del mismo.

**Detección y *tracking***

Una vez detectado cada rostro se procede a realizarles seguimiento a lo largo de la secuencia de cuadros, hasta el momento en que se realiza nuevamente la detección. La lógica de detección/seguimiento del sistema se formaliza con la siguiente máquina de estados:

Inicialmente se corre el detector de rostros en el instante 0, en donde todas las caras detectadas pasan al estado “Trackeando”, ya que estas conforman la población inicial a realizarle seguimiento, permaneciendo en este estado hasta que el detector de caras vuelva a correr y determine su próximo estado.

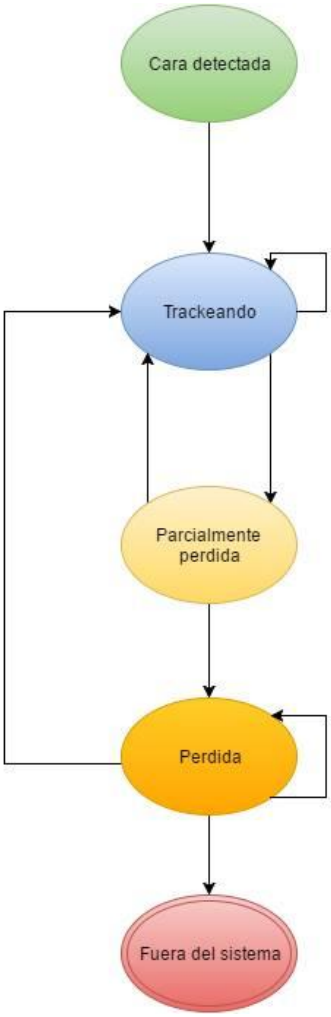



Fig. 19

Luego, el detector de rostros corre cada cierta cantidad fija de cuadros, de manera de encontrar nuevas personas en escena o simplemente actualizar la información de los sujetos en seguimiento. Para ello se determina en primer lugar si hay una relación entre las nuevas caras detectadas y los sujetos en seguimiento. Dos rostros están relacionados si la correlación entre sus histogramas de color supera cierto umbral. Esta búsqueda se hace únicamente con los rostros cuya última posición *trackeada* no esté a más de ε *píxels* de distancia de la posición del nuevo rostro detectado, suponiendo que el movimiento en un *frame* es menor a ε. De existir dicha relación, significa que la persona sigue en escena, por tanto se actualiza su información y permanece en el estado “Trackeando”.

De no existir esta correlación, se repite el procedimiento contra las personas en estado “Perdida” (en este caso sin tomar en cuenta la restricción de distancia ε), que en caso de poder relacionarse con alguna de ellas, esta será reactivada pasando del estado “Perdida” a ser seguida nuevamente (estado “Trackeando”). Si ambas pruebas fallan, se asume que esta constituye una nueva persona y se la ingresa con el estado “Trackeando” y un nuevo Id.

Por otro lado, para determinar si una persona se perdió de la escena, se mantiene una puntuación global, la cual cuenta la cantidad de veces que corrió el detector. Cada persona en seguimiento mantiene una puntuación local, que aumentará cada vez que sea detectada. Si eventualmente la persona se pierde, pasará del estado “Trackeando” a “Perdida parcialmente”, para luego volver a “Trackeando” si al volver a correr el detector, logra relacionarse con alguna de las nuevas caras detectadas. Si esto no es así, pasa al estado “Perdida”. En este estado, la puntuación local no aumenta cada vez que corre el detector. Si la diferencia entre puntuación local y global supera cierto umbral, la persona sale del sistema. De lo contrario, si es re-detectada antes de que esto suceda (midiendo correlaciones), vuelve al estado “Trackeando”. Esto quiere decir que si al momento de correr el detector de rostros, no se encontró una cara que se relacione con la persona, permanecerá en un estado de espera por una cantidad fija de detecciones.

**Feature Tracker con Flujo óptico**

El tracking por flujo óptico permite realizar seguimiento de ciertos puntos del rostro. Esto introduce el problema de cómo dibujar el cuadro que contiene la cara (conocido como *bounding box* o región de interés). Si bien es posible dibujar el cuadro de menor área tal que contenga todos los puntos que se están siguiendo, esto es muy poco robusto, debido a que cualquier punto que se aleje del resto (lo cual es bastante común en momentos de oclusión parcial del rostro) implicará que el cuadro sea demasiado grande.

Para solucionar esto y completar una solución de seguimiento robusta, se consideró estimar el movimiento del cuadro mediante un vector de movimiento, y determinarlo mediante el algoritmo Ransac (Random Sample Consensus, (Fischler & Bolles, s. f.))

Para la primera propuesta, se adaptó el mencionado algoritmo para nuestro problema. Ransac es un algoritmo utilizado para estimar parámetros de cualquier modelo matemático en base a un conjunto de datos que contiene *inliers* y *outliers* (muestras que, con cierta tolerancia, cumplen o no cumplen con el modelo matemático respectivamente). En nuestro caso, el modelo matemático es simplemente un vector de movimiento. A modo de resumen, Ransac funciona como sigue:

1. Tomar aleatoriamente la menor cantidad de muestras necesarias para estimar el modelo matemático. Para determinar un vector, solo necesitamos tomar uno de los puntos trackeados.
2. Resolver el modelo. Al tener el punto original y el trackeado, la resta de los mismos determinará un vector de movimiento.
3. Determinar la cantidad de muestras que cumplen con el modelo, con cierta tolerancia ε. Es decir, cuántos puntos cumplen con el modelo correspondiente a trasladar el recuadro del *frame* anterior según el vector estimado. Cumplir con el modelo significa que el punto queda dentro del recuadro, o a una distancia menor a ε del mismo.
4. Si la fracción de puntos que cumplen el modelo (*inliers*) sobre el total de puntos superan cierto umbral $\boldsymbol{\tau}$, se re-estima el modelo utilizando todos los *inliers* y se termina. La reestimación es simplemente calcular el mínimo recuadro que contenga a todos los *inliers.* Esto puede realizarse ahora debido a que los *outliers* ya fueron descartados.
5. Sino, repetir pasos 1 a 4 un máximo de N veces.

El número máximo de iteraciones N se puede estimar como sigue:

Sea $p$ la probabilidad de que al menos uno de los conjuntos de muestras aleatorias no incluya un *oulier* (generalmente fijada en 0.99), *u* la probabilidad de que una muestra sea un *inlier* y *v* = 1 - *u* la probabilidad de que sea un *outlier*. Si se itera N veces para obtener el número mínimo de muestras $m$ para determinar el modelo, tenemos que:

$$1 - p = (1 - u^m)^N$$

Entonces:

$$N = \frac{\log(1-p)}{\log(1-(1-v)^m)}$$

Determina la cantidad de iteraciones necesaria. Esto introduce también la estimación de *v* y *u*. Esta tarea es difícil debido a que depende mucho de cada secuencia de frames. Nos basamos en una estimación empírica utilizando un video en particular y registrando con qué frecuencia un punto se volvía outlier para determinar un valor aceptable de N.

Para medir objetivamente el rendimiento de esta técnica, calculamos el error de la misma respecto al tiempo. Para medir esto, corrimos el *tracker* y comparamos el error del recuadro generado contra el recuadro que calcula correr el detector de Viola-Jones en cada uno de los cuadros. El error se mide como la distancia entre el centroide del recuadro dibujado por el *tracker* y el dibujado por el detector. A continuación se muestra la gráfica de los errores de ambos *trackers*:

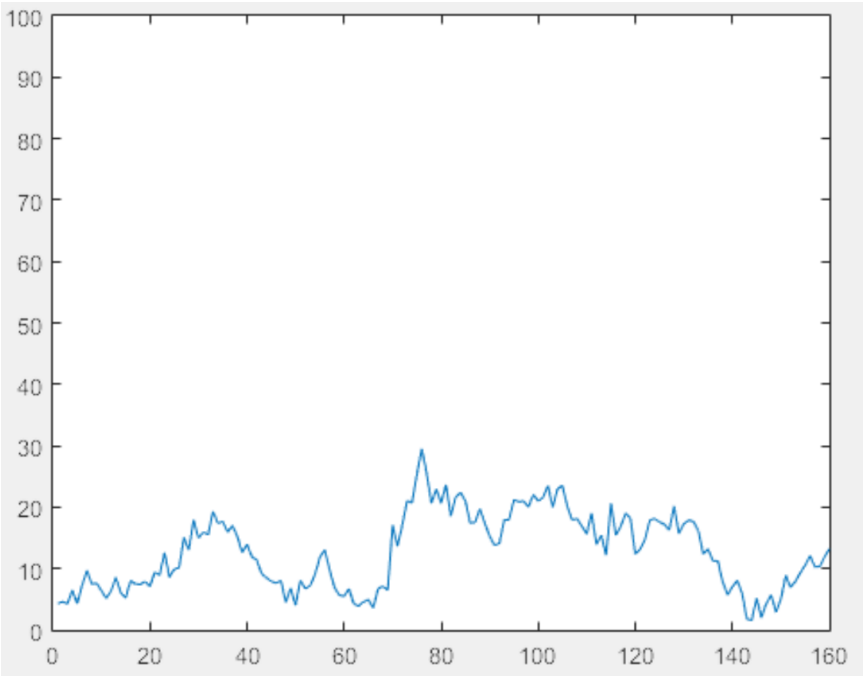



Fig. 20

Como vemos en la figura 20, el error se mantiene acotado a un máximo de 30 píxels a lo largo de 160 cuadros de video.

## Implementación

El sistema se divide en 5 módulos:

- Un detector
- Extractor de características
- Clasificador de características
- Tracker
- Visualización

El funcionamiento, en líneas generales, es como sigue:

El detector se ejecuta en el primer cuadro y cada N cuadros. Su función es detectar los rostros en escena. Existen varios modelos de detección facial (como cascadas de HAAR o de LBP), así como también existe la posibilidad de acelerar la detección de rostros mediante CUDA.

En el primer cuadro se ejecuta el detector para comenzar el trackeo, el cual será responsable de seguir a los rostros en los cuadros que no se ejecute el detector. Cada cara detectada queda asociada con un modelo persona, en donde a su vez se calculan sus puntos característicos para inicializar el modelo del tracking. En los siguientes cuadros se utilizará para detectar nuevas personas y actualizar la información de las personas activas para el seguimiento.

Luego, en el cuadro N+1 se activa un hilo responsable de realizar la extracción de características (PHOG) y la clasificación (SVM). Esto es necesario para poder tener una primera aproximación de la edad y género de la persona en el momento en el que entra en escena.

Este hilo recorre los rostros almacenados de las personas trackeadas activas, es decir aquellas que están en escena y para cada uno se detectan los ojos y la nariz. Para aquellos que la detección sea exitosa, se determina su pose calculando el vector resultante entre los ojos detectados, y tomando el ángulo entre dicho vector y el eje horizontal. De igual forma se realiza entre el punto medio de los ojos detectados y la nariz. Con esto se establecen 2 clases con diferentes ponderaciones a ser tenidas en cuenta en la clasificación:

- Aquellas cuyo ángulo del vector de ojos sea a lo sumo 15º y el de nariz entre 75º y 105º. Se le asigna un peso de 1. Se muestra un ejemplo en la figura 21

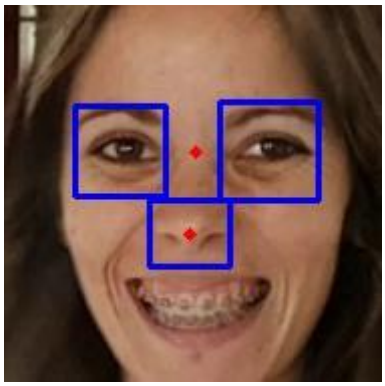


Fig. 21

- Aquellas cuyo ángulo del vector de ojos este entre 15º y 30º, o el de nariz entre 60º y 75º o entre 105º y 120º. Se le asigna un peso de 0.5. Se muestra otro ejemplo en la figura 22.

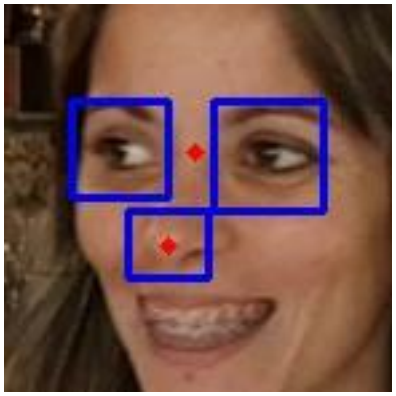


Fig. 22

Aquellos rostros que no cumplan esto se descartan.

A las caras que cumplen con el filtro anterior se les realizará la extracción de características que luego serán utilizadas por el clasificador correspondiente, de acuerdo al tamaño de la imagen del rostro, para determinar a qué clase corresponde. En concreto, si el tamaño está comprendido entre 120 píxeles y 250 píxeles la imagen se escala a 250 píxeles y si está entre 250 píxeles y 450 píxeles se redimensiona a 400 píxeles.

Luego, considerando los resultados de las clasificaciones ya almacenadas se determina el género y la edad como la suma ponderada por los pesos de la pose del rostro y la clasificación obtenida en cada caso.

Finalmente el módulo de visualización desplegará la información de clasificación obtenida para cada persona y mostrará la ventana de trackeo que es obtenida mediante RANSAC.

## Resultados finales

Los resultados obtenidos en C++ con la librería libSVM de OpenCV 3.0 fueron muy similares a los que se alcanzaron con Matlab utilizando la misma base de datos (Morph con una toma única por persona). Sin embargo la precisión cae cuando el sistema corre con un video real, alcanzando niveles satisfactorios (sobretodo al añadir una ponderación de las clasificaciones en función de la pose del rostro) pero inferiores a los obtenidos en un marco totalmente controlado como lo es una base de datos de rostros.

Por otro lado, se logró construir un tracker muy robusto ante movimientos bruscos y oclusiones que logra seguir los rostros en escena para una gran variedad de escenarios al combinar un tracker basado en seguimiento de características por flujo óptico con Ransac para la estimación del movimiento del cuadro de los rostros.

Como primer prototipo de solución, tenemos un sistema que funciona de forma satisfactoria para cierto tipo de escenarios, teniendo la posibilidad de adaptarse a otros si se realizan ajustes como los mencionados en el capítulo de Diseño.

Por último, destacamos la innovación de utilizar el mismo vector de características para la clasificación de género y edad simultáneamente, obteniendo resultados a la altura de los del estado del arte. Si bien esto ya había sido realizado en algunas investigaciones, no fue utilizando el vector PHOG como se realizó en este trabajo.

## Conclusión y futuros pasos

La conclusión del presente trabajo resulta en que se lograron los objetivos primarios, se realizó una extensa investigación en el área en general y en las técnicas para resolver nuestro problema en particular. Se realizaron experimentos que aportaron información valiosa sobre los rendimientos de distintas combinaciones de extractores de características y clasificadores.

Cabe destacar que la arquitectura del sistema permite a futuro probar distintas técnicas de extracción de características y clasificadores para mejorar el rendimiento del mismo en escenarios reales.

Por otro lado, encontramos muy interesante la posibilidad de añadir un módulo al sistema que realice la misma clasificación de género y edad tomando como entrada señales de audio, lo cual contribuiría a construir una solución más completa.

Además, consideramos que utilizar imágenes de resoluciones diferentes para construir varios modelos de clasificación y varios clasificadores, diferenciando los rostros del video según su resolución y utilizando el clasificador que mejor se adecúe al mismo. Con esto se mejorará la precisión global del sistema en casos

donde los rostros se detecten a distancias variables, además de ampliar la cantidad de escenarios en los que el sistema es aplicable sin la necesidad de un proceso de adaptación tan costoso.

## Bibliografía


10k US Adult Faces Database. (s. f.). Recuperado a partir de http://www.wilmabainbridge.com/facememorability2.html

Active Appearance Models. (s. f.). Recuperado 6 de julio de 2016, a partir de http://www.imm.dtu.dk/~aam/

Arigbabu, O. A., Syed Ahmad, S. M., Wan Adnan, W. A., Yussof, S., & Mahmood, S. (2015). Soft Biometrics: Gender Recognition from Unconstrained Face Images Using Local Feature Descriptor. *Journal of Information & Communication Technology*, *14*, 111-122.

Bradski, G. R. (1998). *Computer Vision Face Tracking For Use in a Perceptual User Interface*.

Cai, D., He, X., Han, J., & Zhang, H. J. (2006). Orthogonal Laplacianfaces for Face Recognition. *IEEE Transactions on Image Processing*, *15*(11), 3608-3614. http://doi.org/10.1109/TIP.2006.881945

Cheng, Y. (1995). Mean shift, mode seeking, and clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *17*(8), 790-799. http://doi.org/10.1109/34.400568

Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. En *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)* (Vol. 1, pp. 886-893 vol. 1). http://doi.org/10.1109/CVPR.2005.177

Eidinger, E., Enbar, R., & Hassner, T. (2014). Age and Gender Estimation of Unfiltered Faces. *IEEE Transactions on Information Forensics and Security*, *9*(12), 2170-2179. http://doi.org/10.1109/TIFS.2014.2359646

Enlace de descarga de IMFDB. (s. f.). Recuperado a partir de http://cvit.iiit.ac.in/projects/IMFDB/pages/downloadDB.jsp

Face Image Project - Data - Adience. (s. f.). Recuperado a partir de http://www.openu.ac.il/home/hassner/Adience/data.html

Face Recognition Data, University of Essex. (s. f.). Recuperado a partir de http://cswww.essex.ac.uk/mv/allfaces/index.html

FACE RECOGNITION HOMEPAGE. (s. f.). Recuperado a partir de http://www.face-rec.org/databases/

Faragher, R. (2012). Understanding the Basis of the Kalman Filter Via a Simple and Intuitive Derivation [Lecture Notes]. *IEEE Signal Processing Magazine*, *29*(5), 128-132. http://doi.org/10.1109/MSP.2012.2203621

Feature Detection — OpenCV 2.4.13.0 documentation. (s. f.). Recuperado 27 de julio de 2016, a partir de http://docs.opencv.org/2.4/modules/imgproc/doc/feature_detection.html

Features - Computer Vision System Toolbox - Matlab. (s. f.). Recuperado a partir de http://www.mathworks.com/products/computer-vision/features.html

FG-NET Human Age Estimation. (s. f.). Recuperado a partir de http://www.cse.msu.edu/~hhan/download.htm

Fischler, R., & Bolles, M. (s. f.). Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *ResearchGate*, *24*, 619-638.

Freund, Y., & Schapire, R. E. (1995). A Decision-theoretic Generalization of On-line Learning and an Application to Boosting. En *Proceedings of the Second European Conference on Computational Learning Theory* (pp. 23–37). London, UK, UK: Springer-Verlag. Recuperado a partir de http://dl.acm.org/citation.cfm?id=646943.712093

Functions and Objects Supported for C and C++ Code Generation -- Alphabetical List. (s. f.). Recuperado a partir de http://www.mathworks.com/help/simulink/ug/functions-supported-for-code-generation--alphabetical-list.html

Gumus, E., Kilic, N., Sertbas, A., & Ucan, O. N. (2010). Evaluation of face recognition techniques using PCA, wavelets and SVM. *Expert Systems with Applications*, *37*(9), 6404-6408. http://doi.org/10.1016/j.eswa.2010.02.079

Guo, G., Fu, Y., Dyer, C. R., & Huang, T. S. (2008). Image-Based Human Age Estimation by Manifold Learning and Locally Adjusted Robust Regression. *IEEE Transactions on Image Processing*, *17*(7), 1178-1188. http://doi.org/10.1109/TIP.2008.924280

Hussain, M., Ullah, I., Aboalsamh, H. A., Muhammad, G., Bebis, G., & Mirza, A. M. (2013). Gender Recognition from Face Images with Dyadic Wavelet Transform and Local Binary Pattern. *International Journal on Artificial Intelligence Tools*, *22*(6), 1. http://doi.org/10.1142/S021821301360018X

Indian Movie Face database (IMFDB). (s. f.). Recuperado a partir de http://cvit.iiit.ac.in/projects/IMFDB/

Labeled Faces in the Wild Home. (s. f.). Recuperado a partir de http://vis-www.cs.umass.edu/lfw/

Lucas, B. D., & Kanade, T. (1981). An Iterative Image Registration Technique with an Application to Stereo Vision.” 7th IJCAI, 674-679. En *ResearchGate* (pp. 674-679). Recuperado a partir de https://www.researchgate.net/publication/220814700_An_Iterative_Image_Registration_Technique_with_an_Application_to_Stereo_Vision_7th_IJCAI_674-679

MATLAB and OpenCV - MATLAB. (s. f.). Recuperado a partir de http://www.mathworks.com/discovery/matlab-opencv.html

Moeini, A., Faez, K., & Moeini, H. (2015). Real-world gender classification via local Gabor binary pattern and three-dimensional face reconstruction by generic elastic model. *IET Image Processing*, *9*(8), 690-698. http://doi.org/10.1049/iet-ipr.2014.0733

MORPH | I3S. (s. f.). Recuperado a partir de http://www.faceaginggroup.com/morph/

MORPH Database (non-commercial). (s. f.). Recuperado a partir de https://ebill.uncw.edu/C20231_ustores/web/product_detail.jsp?PRODUCTID=8

MORPH Non-Commercial Release Whitepaper. (s. f.). Recuperado a partir de http://people.uncw.edu/vetterr/MORPH-NonCommercial-Stats.pdf

Nithyashri, J., & Kulanthaivel, G. (2012). Classification of human age based on Neural Network using FG-NET Aging database and Wavelets. En *2012 Fourth International Conference on Advanced Computing (ICoAC)* (pp. 1-5). http://doi.org/10.1109/ICoAC.2012.6416855

OpenCV: cv::MultiTrackerTLD Class Reference. (s. f.). Recuperado 27 de julio de 2016, a partir de

http://docs.opencv.org/master/d2/d33/classcv_1_1MultiTrackerTLD.html#gsc.tab=0

Rai, P., & Khanna, P. (2015). An illumination, expression, and noise invariant gender classifier using two-directional 2DPCA on real Gabor space. *Journal of Visual Languages & Computing*, *26*, 15-28. http://doi.org/10.1016/j.jvlc.2014.10.016

Sai, P.-K., Wang, J.-G., & Teoh, E.-K. (2015). Facial age range estimation with extreme learning machines. *Neurocomputing*, *149*, *Part A*, 364-372. http://doi.org/10.1016/j.neucom.2014.03.074

Shan, C. (2012). Learning local binary patterns for gender classification on real-world face images. *Pattern Recognition Letters*, *33*(4), 431-437. http://doi.org/10.1016/j.patrec.2011.05.016

Sitio de descarga de adiencedb. (s. f.). Recuperado a partir de http://www.cslab.openu.ac.il/download/

Sobel, I., & Feldman, G. (1968). Feldman, G.: A 3x3 isotropic gradient operator for image processing. *ResearchGate*. Recuperado a partir de https://www.researchgate.net/publication/210198558_Feldman_G_A_3x3_isotropic_gradient_operator_for_image_processing

Suggested databases for age estimation - answers.opencv.org. (s. f.). Recuperado a partir de http://answers.opencv.org/question/58882/aging-database-for-age-estimation/

The Color FERET Database. (s. f.). Recuperado a partir de http://www.nist.gov/itl/iad/ig/colorferet.cfm

The Images of Groups Dataset. (s. f.). Recuperado a partir de http://chenlab.ece.cornell.edu/people/Andy/ImagesOfGroups.html

Vapnik, V. N., & Vapnik, V. (1998). *Statistical learning theory* (Vol. 1). Wiley New York. Recuperado a partir de http://www.dsi.unive.it/~pelillo/Didattica/Artificial%20Intelligence/Old%20Stuff/Slides/SLT.pdf

Viola, P., & Jones, M. J. (s. f.). Robust Real-Time Face Detection. *International Journal of Computer Vision*, *57*(2), 137-154. http://doi.org/10.1023/B:VISI.0000013087.49260.fb

Wang, J.-G., Yau, W.-Y., & Wang, H. L. (2009). Age categorization via ECOC with fused gabor and LBP features. En *2009 Workshop on Applications of Computer Vision (WACV)* (pp. 1-6). http://doi.org/10.1109/WACV.2009.5403085

Weka 3 - Data Mining with Open Source Machine Learning Software in Java. (s. f.). Recuperado 27 de julio de 2016, a partir de http://www.cs.waikato.ac.nz/ml/weka/

Witaszek, J. (1995). Book review: Backpropagation: Theory, architectures, applications. By Yves Chauvin and David E. Rumelhart (eds). Lawrence Erlbaum, Hillsdale, NJ, Hove, UK, 1995. ISBN 0-8058-1258-X, pp. 561. *Neurocomputing*, *9*, 358-359. http://doi.org/10.1016/0925-2312(95)90002-0

Ylioinas, J., Hadid, A., & Pietikainen, M. (2012). Age Classification in Unconstrained Conditions Using LBP Variants. En *2012 21st International Conference on Pattern Recognition (ICPR)* (pp. 1257-1260).