Universidad de Oviedo

ESCUELA POLITÉCNICA DE INGENIERÍA DE GIJÓN

GRADO EN INGENIERÍA ELECTRÓNICA INDUSTRIAL Y AUTOMÁTICA

ÁREA DE INGENIERÍA DE SISTEMAS Y AUTOMÁTICA

TRABAJO FIN DE GRADO Nº 17010427

CLASIFICACIÓN DE EXPRESIONES FACIALES A TRAVÉS DE TÉCNICAS DE VISIÓN POR COMPUTADOR

D. FERNÁNDEZ GARCÍA, Álvaro
TUTOR: D. GONZÁLEZ DE LOS REYES, Rafael Corsino

FECHA: Julio de 2017

Índice

MEMORIA

Índice de Memoria

Índice de Figuras

Índice de Tablas

1. Introducción

1.1 OBJETIVO GENERAL

El objetivo del presente proyecto es el estudio de la posibilidad de clasificar la expresión facial de un individuo a través de técnicas de visión artificial. De esta manera, se podría conseguir que ordenadores o sistemas informáticos adquieran un nivel de entendimiento básico de la comunicación no verbal de los seres humanos. Para llevar a cabo el trabajo, se analizará el estudio del arte y se escogerá, entre las diversas técnicas, un método que permita analizar la viabilidad del proyecto.

El paso del mundo real al lenguaje informático se lleva a cabo tomando imágenes y analizando los píxeles que las componen, es decir, mediante la visión por computador. La información extraída de las fotografías será analizada mediante técnicas de *machine learning* o aprendizaje automático para determinar qué tipo de emoción presenta el sujeto.

Para detectar la cara, y los ojos dentro de la misma, se usan clasificadores en cascada que hacen uso de los filtros de Haar. El resto de puntos de la cara se detectan mediante máquinas de soporte vectorial y los datos aportados por descriptores de *HOG* (*Histograms of Oriented Gradients*). La determinación de la expresión facial del individuo se lleva a cabo también con máquinas de soporte vectorial que analizan, entre dos imágenes, el desplazamiento de los puntos detectados.

1.2 ESTADO DEL ARTE

La tecnología y nuestra forma de interactuar con ella han sufrido un cambio colosal durante los últimos cien años. Tecnologías antaño punteras y reservadas a los usuarios más especializados son usadas, en la actualidad, por millones de personas en su día a día.

El intercambio de información entre el hombre y la máquina ha ido evolucionando radicalmente hacia una comunicación más natural. Basta con recordar la información visual que ofrecían los ordenadores hace 50 años y compararlo con las actuales interfaces gráficas, entornos de programación y pantallas táctiles o sistemas de reconocimiento de voz. Sin embargo, aunque sí se han dedicado grandes esfuerzos a que las máquinas sean capaces de entender distintos comandos de voz, aún no se ha conseguido un sistema capaz de

comprender la ingente cantidad de información que revelamos con nuestra comunicación no verbal.

La comunicación no verbal abarca todo aquel intercambio de información en el que no se usan palabras. Esto incluye los gestos que se realizan con las manos, el semblante del rostro, la postura que se tiene, cómo y con qué frecuencia varía, hacia dónde se dirige la mirada y la forma en que se hace etc. Este trabajo se centrará exclusivamente en la compleja tarea de analizar la expresión facial del individuo con el fin de poder intuir el estado emocional de la persona. Cabe mencionar que en esta memoria se hará uso de la palabra "emoción" como sinónimo de expresión facial. Si bien ambos términos no significan exactamente lo mismo, pudiendo una expresión facial ser falsa y no expresar la verdadera emoción del sujeto, se tomará esta licencia por comodidad lingüística.

Conseguir que las máquinas comprendan este lenguaje es complejo, pero permite explorar una forma de comunicación más similar a la que llevamos a cabo entre humanos. Las aplicaciones de estas técnicas prometen ser de gran utilidad en el futuro. Podrían implementarse en sistemas de comunicación con robot industriales o, principalmente, con robots de servicio, los cuales se prevé que tendrán un trato muy habitual con los humanos. Las técnicas que se analizarán podrán traer también posibles mejoras en los asistentes virtuales que ya existen en algunas páginas web o servicios de atención bancaria. Así mismo, entre otras muchas posibilidades, podrían también implementarse en aplicaciones de ocio e, incluso, en sofisticados sistemas de ayuda a invidentes.

El método que se propondrá en este proyecto se apoyará en las bases teóricas de Paul Ekman y Wallace V. Friesen, pioneros en el análisis por visión artificial de las expresiones faciales. Estos desarrollaron en 1977 el sistema de codificación facial o *Facial Action Coding System* [1], una teoría que define la expresión facial como una combinación de diferentes acciones llevadas a cabo por los distintos músculos de la cara. Cada una de estas acciones se denominan unidades de acción o *action units* (AUs) y pueden presentarse con mayor o menor fuerza. Puede consultarse en la siguiente referencia la lista completa de unidades de acción propuestas por Ekman y Friesen [2]. En la Figura 1.1 se pueden ver tres imágenes de rostros en los que aparecen varias unidades de acción.

		
***AU4**: Eyebrowns drawn medially and down* ***AU17**: Skin of chin elevated* ***AU23**: Lips tightened* ***AU24**: Lips pressed together*	***AU25**: Lips parted* ***AU26**: Jaw dropped*	***AU6**: Cheeks raised* ***AU12**: Lip cornes pulled up and laterally*

Figura 1.1- Imágenes de caras junto a las *AUs* que presentan y su significado.

En base a esto, se han desarrollado distintos clasificadores de la emoción que presenta un usuario. Por ello, siguiendo este método de análisis, cuando se plantea descubrir qué expresión facial tiene un individuo la pregunta que debemos hacernos es qué unidades de acción se encuentran presentes en el rostro de esa persona.

Existe también otro importante sistema de estudio de los parámetros faciales de creación más reciente. Los *Facial Animation Parameters* (FAPs) [3] nacieron en 1998, promovidos por el *Moving Pictures Experts Group (MPEG)*, con el objetivo de crear un estándar unificado en la animación por ordenador. Este define 84 puntos característicos, *feature points (FPs)*, cuyo movimiento se usa para construir expresiones en las caras animadas. Se denomina *FAP* a cada uno de esos movimientos, definiéndose 68 en el estándar MPEG-4. Las similitudes con el sistema *FAC* son amplias, por lo que existen varios trabajos que intentan relacionar directamente ambos estándares [4], [5].

Centrándose de nuevo en el sistema de Ekman y Friesen, detectar las diferentes *AUs* y la intensidad con la que estas aparecen no es una tarea trivial. Se debe tener en cuenta que no se partirá de una perfecta imagen de la cara, por lo que, para facilitar la localización de las diferentes unidades acción, es aconsejable desarrollar primero un método de detección facial.

Una vez reducido nuestro espacio de búsqueda al rostro del individuo, la tarea resulta mucho más simple y eficiente que rastrear toda la imagen.

El *machine learning* o aprendizaje automatizado engloba el conjunto de algoritmos por las cuales un ordenador es capaz de "aprender" a tomar decisiones a través de una serie de ejemplos dados. Estas técnicas son las bases de nuestro método de clasificación, usándose distintas variantes en cada etapa del proceso.

La detección facial se lleva a cabo en base al estudio de Viola-Jones [6], haciendo un uso combinado de filtros de Haar y clasificadores en cascada, en el que se profundizará más adelante. Si bien el método de Viola-Jones es uno de los más extendidos por su eficacia y rapidez, existen otros métodos de detección facial al ser este uno de los campos que más interés ha generado en el mundo de la visión artificial. Entre ellos, merece la pena nombrar el sistema de detección y seguimiento de Kanade, Lucas y Tomasi [7], [8]. Takeo Kanade, una eminencia en el campo, también desarrolló junto a Henry W. Schneiderman un algoritmo de detección basado en métodos estadísticos [9].

Podrían incluirse en un grupo aparte las aproximaciones basadas en el color de la piel del individuo. Por ejemplo, en el artículo de Esau et al. [10] se prefija una zona de interés basada en la mayor región de la imagen color carne para, a continuación, estudiar la respuesta de distintos modelos en ellos. El método *Camshift* [11], una mejora de *Meanshift* [12] desarrollada por Bradski et al., se basa también en gran medida en el color de piel de la persona. A través del algoritmo sólo se mantienen los píxeles con valores próximos a los de la piel. A continuación, mediante métodos iterativos, se localiza la cara como la zona de mayor densidad de píxeles. La gran ventaja frente a *Meanshift* es la capacidad de rotar y escalar su región de interés. Con el fin de que el funcionamiento sea más estable ante cambios de iluminación y para elaborar un método común para la mayor cantidad de personas posible se descartan este tipo de algoritmos.

Como se ha comentado, cada expresión facial puede definirse por combinaciones de unidades de acción. De esta manera, podemos asociar cada emoción a una serie de movimientos en el rostro. Para extraer esa información se analizará la variación de posición de 22 puntos característicos de la cara, Figura 1.2. Los puntos elegidos se basan en los utilizados en el artículo de Michel y El Kaliouby [13].

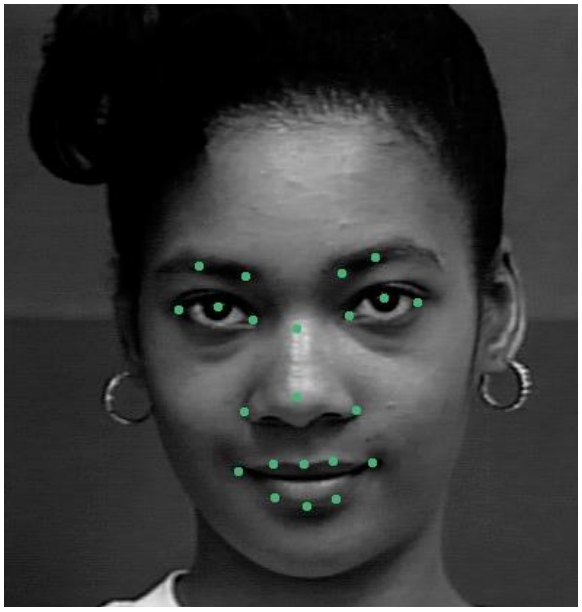


Figura 1.2.- Imagen en la que aparecen indicados los puntos que se pretenden detectar.

Para facilitar la detección de los puntos característicos de la cara detectaremos en primer lugar los ojos. Una vez obtenidas estas coordenadas es posible llevar a cabo la rotación de la imagen de modo que ambos puntos queden a la misma altura. Con este enderezado se homogenizan las imágenes facilitando la detección de los puntos y la posterior clasificación de emociones.

Para detectar las pupilas se seguirá un método análogo al de detección de la cara cambiando, en este caso, el propósito para el que se entrenan los clasificadores. La rotación que permitirá que los ojos se encuentren alineados se lleva a cabo mediante una transformación afín de la imagen.

La detección del resto de puntos se lleva a cabo con el uso combinado de descriptores *HOG* (*Histograms of Oriented Grandients*) y, de nuevo, métodos de aprendizaje automático. Los histogramas de gradientes orientados fueron introducidos por primera vez por Robert K. McConnell en 1982 [14], pero no fue hasta el año 2005 cuando Dalal y Triggs desarrollaron el algoritmo [15] en el que se basa nuestro método. Los descriptores extraen información de una determinada región de la cara y el clasificador determina la semejanza con el modelo de dicha región. De esta manera, los 22 puntos serán en realidad los centros de 22 regiones de la cara. Los histogramas de gradientes orientados se utilizan habitualmente en la detección

de peatones, pero, por sus características que se analizarán con detenimiento más adelante, es posible su uso para detección de puntos de la cara. Un ejemplo de esta última aplicación puede verse en el detector de puntos faciales de la librería Dlib [16], [17], aunque, como se verá más adelante, se desarrollarán detectores propios.

Como en todos los métodos de detección de objetos, existe gran variedad de procedimientos para determinar la posición de puntos del rostro, a parte de las dos comentadas. Una de las estrategias más utilizadas es la aplicación de los filtros de Gabor con algoritmos de aprendizaje. Ejemplos de ello pueden encontrarse en los trabajos de D.Vukadinovic y M.Pantic [18], y el propio Pantic y M. Valstar [19]. Otros sistemas parten de unas posiciones fijas y van realizando métodos de *tracking* o seguimiento, por ejemplo, mediante flujo óptico como en el artículo de A. Aghagolzadeh, H. Seyedarabi, y S. Khanmohammadi [20]. Otra opción más simple es el estudio de los cambios de intensidad en la imagen. Es el caso de la detección de los ojos y la boca en el proyecto de Vukadinovic y Pantic [18].

Para obtener información de las *AUs* que se producen se analizará la variación de las posiciones de los puntos con respecto a los que se detectan en una imagen en la que la emoción del sujeto es neutra, es decir, aparentemente, el individuo no presenta ninguna emoción en concreto.

Existen otros métodos de análisis de las unidades de acción. Son habituales los que llevan a cabo un análisis de la evolución temporal de los puntos [18], [19], [21] y de esta manera pueden diferenciar cuando la unidad de acción está apareciendo, cuando está en su punto álgido y cuando está desapareciendo. El análisis de los *tempos* de estos sucesos permite dirimir, en sistemas de gran complejidad, cuando se trata de una emoción real y cuando es una emoción impostada. Este y otros temas relacionados con el análisis temporal se tratan en profundidad en el libro de M.F.Valstar [22].

Otras técnicas utilizan modelos 3D de la cara que informan de manera conjunta de la posición de la misma cómo de las deformaciones que se producen en esta. Uno de los métodos más usados de este tipo es el *Candide grid* [23], [24], que utiliza una malla de triángulos, que define la cara a través de una malla de triángulos. Otro modelo importante es el *Piecewise Bezier Volume Deformation (PBVD)*, desarrollado por Tao y Huang [25]. Este último usa un modelo 3D compuesto por 16 volúmenes de Bézier. También cabe mencionar

el uso de los *Active Shape Models* o Modelos de Forma Activa [26], que se sirven de modelos estadísticos para averiguar las muecas y posición del rostro. Aunque cabe la posibilidad de que estos métodos arrojen mejores resultados que el escogido en este proyecto, la estrategia seguida permite analizar en mayor profundidad los problemas que supone la clasificación de emociones.

Una vez conocidas las coordenadas de la nube de puntos, se lleva a cabo una normalización de las mismas. Dividiendo los valores de cada punto por la distancia entre pupilas nos aseguramos un sistema más robusto al estar menos afectado por cambios de escala de la cara. De esta manera, el método es aplicable para diferentes tamaños de imagen.

Estas coordenadas normalizadas se comparan con las obtenidas previamente en una imagen en la que el usuario tiene una emoción neutra, es decir, el rostro serio sin mostrar ninguna emoción de manera clara. Se ha pasado de tener dos conjuntos de puntos a unos vectores de diferencias.

Existen otras formas de medir el desplazamiento de los puntos característicos. A modo de ejemplo, en ocasiones se estudian las posiciones relativas entre varios puntos [21] y en otras el ángulo formado por segmentos formados por los puntos [10].

Los vectores obtenidos se introducen en un clasificador que estima la emoción del usuario. Ekman y Friesen, tras sus viajes alrededor del mundo, teorizaron en [27] sobre la existencia de 7 emociones básicas y universales (felicidad, tristeza, enfado, sorpresa, asco, miedo y la emoción neutra o ausencia de emoción). En la base de imágenes de Cohn-Kanade [28], [29] se introduce, sin embargo, una octava, el desprecio. Siendo esta la fuente de muestras empleada en el proyecto, se intentará llevar a cabo la clasificación entre 8 expresiones distintas.

Una de las principales técnicas del *machine learning* es la clasificación, recurrentemente usada en el proyecto. En estos métodos, al sistema se le proporcionan muestras de datos especificando a qué conjunto pertenecen. En nuestro caso, para cada emoción se aportarían varios ejemplos de los vectores obtenidos. A partir de estos datos, los clasificadores son capaces establecer modelos y determinar a cuál de ellos se aproxima más unos nuevos datos que se introduzcan al sistema. Entre las distintas alternativas, los más utilizados en la

clasificación de emociones son las redes neuronales[20], los sistemas de lógica difusa [10] y las máquinas de soporte vectorial [13], [24]. Este último algoritmo será el sistema implementado en este proyecto, tanto para la clasificación de la emoción como para la detección mediante descriptores *HOG*.

Destacar, por último, el artículo de Vinay Bettadapura [30] en caso de que se busque profundizar más en el Estado del Arte del reconocimiento de expresiones faciales.

1.3 RESUMEN DEL MÉTODO

Un resumen general del método, en forma de diagrama de bloques, puede observarse en la Figura 1.3. El sistema queda dividido, por tanto, en tres módulos teniendo como entrada dos imágenes y como resultado la predicción de la expresión facial.

Figura 1.3- Diagrama de bloques, resumen general del proyecto

Para facilitar el análisis, el primer paso consiste en la detección de la cara mediante el método Viola-Jones [6]. Este método destaca por su rapidez y fiabilidad haciendo uso de clasificadores de cascada que toman decisiones en base a los conocidos como filtros de Haar.

Ya en el segundo bloque, Figura 1.4, los mismos métodos son utilizados para localizar los ojos del individuo. Llevar a cabo esta detección en primer lugar permite realizar una rotación de la imagen original que enderece la imagen en caso de que el sujeto se encuentre con la cabeza ladeada. A continuación, pueden detectarse el resto de puntos característicos que sirven para para definir la expresión del individuo. Para determinar la posición de los puntos se utiliza el método de Dalal y Triggs [15], adaptándose en este caso a las posiciones del rostro que nos interesan. Este algoritmo utiliza histogramas de gradientes orientados para extraer información de la imagen y máquinas de soporte vectorial para determinar si el área en cuestión se corresponde con un punto o no.

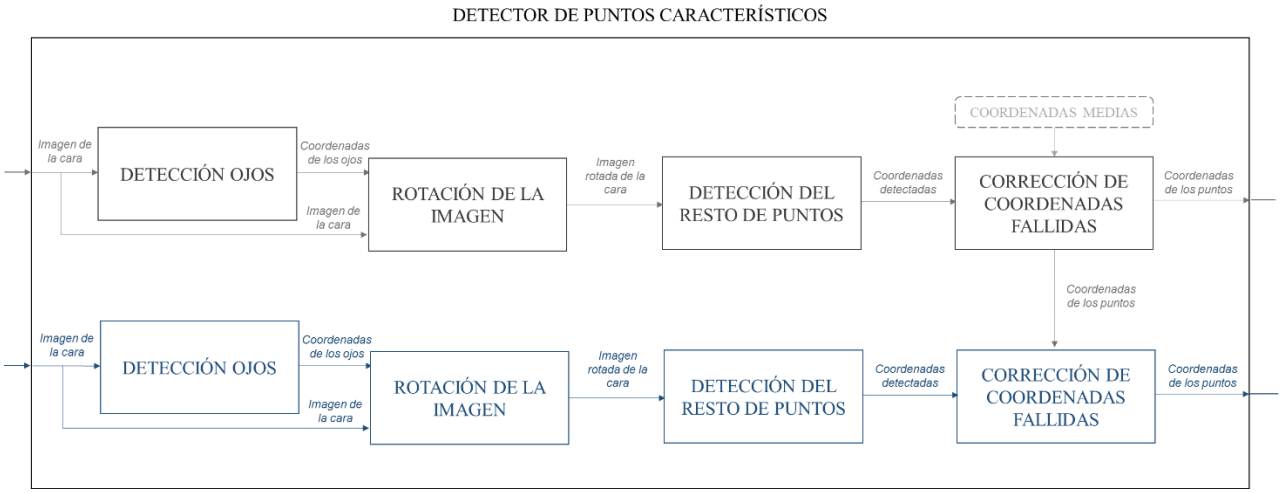


Figura 1.4- Diagrama de bloques que resume la detección de puntos característicos

Las coordenadas de los distintos puntos constituyen la entrada al tercer y último bloque, Figura 1.5. Las posiciones se normalizan utilizando la distancia entre ojos. De esta manera, se consigue que el método sea invariante ante el tamaño de la imagen. La diferencia entre ambas coordenadas proporciona la distancia que se han desplazado los puntos característicos. El desplazamiento de la red de puntos sirve como entrada a un conjunto de, de nuevo, máquinas de soporte vectorial que determinan entre las distintas opciones la emoción que presenta el sujeto.

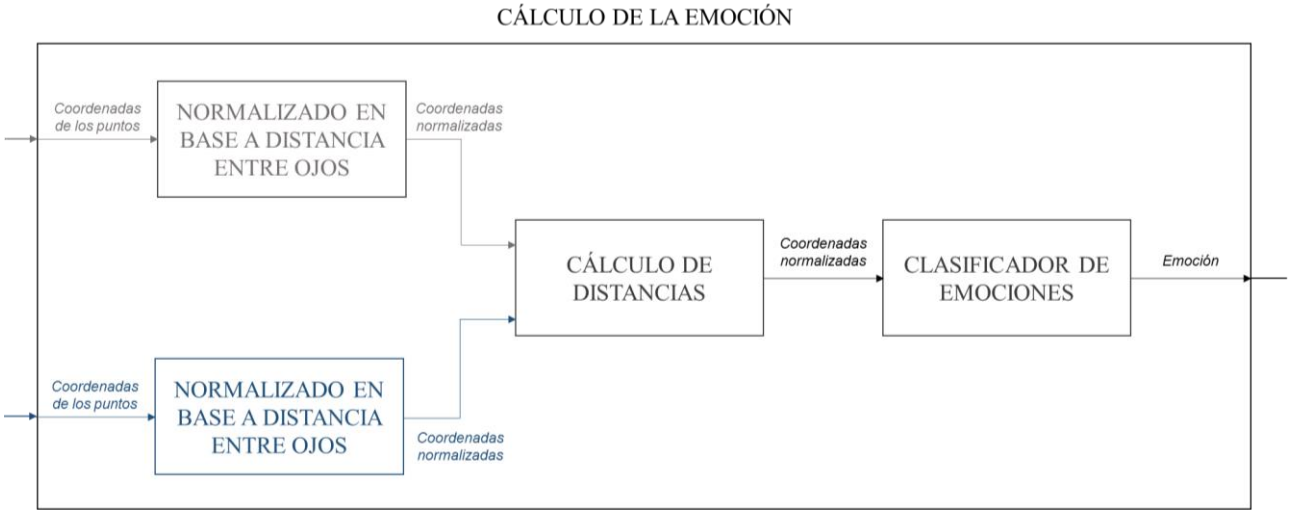


Figura 1.5- Diagrama de bloques que resume el cálculo de la emoción

2. Marco teórico

2.1 CLASIFICADORES EN CASCADA

Como ya se ha comentado, el método de este trabajo para detectar la cara se basa en el propuesto por Paul Viola y Michael J. Jones en 2004 [6]. Este algoritmo tuvo un gran impacto en la detección facial, siendo a día de hoy uno de la más extendidos. Destaca por su rapidez y bajo costo computacional lo cual resulta óptimo teniendo en cuenta que, en este tipo de estudios, interesa aproximarse a un funcionamiento en tiempo real. De nada sirve un robot que sea capaz de entender tu expresión facial horas después de haber interactuado contigo.

Una de las principales limitaciones del método es la incapacidad para detectar rostros de perfil. Al contrario que algunos de los modelos 3D comentados, Viola-Jones debe programarse para detectar caras frontales o perfiles, no permitiendo por su naturaleza detectar simultáneamente ambos tipos de rostros. De cualquier manera, el método escogido para calcular la emoción no sería posible con imágenes de perfil siendo, además, un campo menos avanzado al ofrecer estas imágenes mucha menos información.

La idea general del algoritmo de Viola-Jones consiste el uso de imágenes integrales para analizar sectores de la imagen de distintos tamaños, filtros de Haar para extraer las características de estas regiones y una cascada de clasificadores para determinar cuál de esas regiones tiene mayor posibilidad de ser una cara.

El concepto de imágenes integrales fue introducido también por Viola y Jones y se encuentra descrito en el artículo anteriormente citado. Las imágenes integrales son una representación matricial de la imagen real. Es importante, tanto para este concepto como para el resto del trabajo, entender la imagen como una matriz de valores. Cada posición corresponderá con un píxel y cada valor, al tratarse de imágenes en escala de grises, con lo cercano que este se encuentre al blanco o al negro. Generalmente, se trabaja con 256 valores (2^8), asociándose el 0 al negro y 255 al blanco.

Sin embargo, en la imagen integral, el valor asociado a la posición contiene la suma de los píxeles de la parte superior y a la izquierda de esas coordenadas, como se aprecia en la ecuación 2.1 y cuya utilidad puede verse en la Figura 2.1. Estas estructuras son unas de las

principales responsables de la alta velocidad del método pues permiten calcular eficientemente características de sectores de distintas escalas de la imagen.

$$ii(x,y) = \sum_{x' \leq x, y' \leq y} i(x',y') \quad (2.1)$$

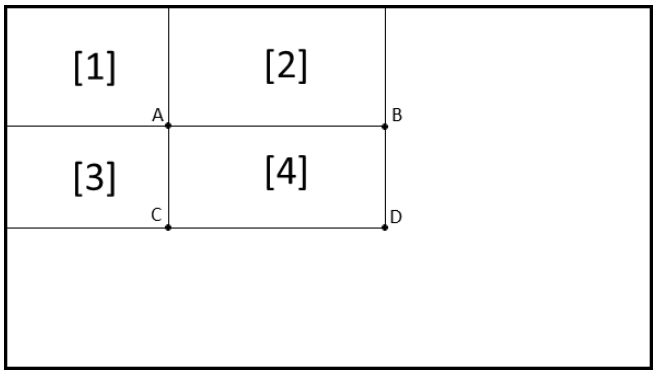


Figura 2.1-En una imagen integral, el conjunto de valores del rectángulo *4* puede calcularse con solo cuatro referencias. Σ (4) = *D + A - (B + C).*

En el caso del método Viola-Jones, estas características se extraen mediante los filtros de Haar, empleados por primera vez en la detección de objetos por Papageorgiou et al [31]. En la Figura 2.2 se pueden apreciar los principales filtros de Haar que se utilizan en este trabajo. El funcionamiento de estos es muy sencillo. Al aplicarse sobre una región de la imagen, la suma de los valores contenidos en la zona negra debe restarse a la suma de los valores contenidos en la zona blanca siendo el resultado lo que se conoce como una característica de Haar. El uso de imágenes integrales agiliza enormemente este proceso. Como se aprecia en la Figura 2.1 la suma de los valores de cualquier área rectangular puede obtenerse con un máximo de 4 operaciones. De lo contrario, con la imagen real, habría sido necesario llevar a cabo la suma de todos los valores.

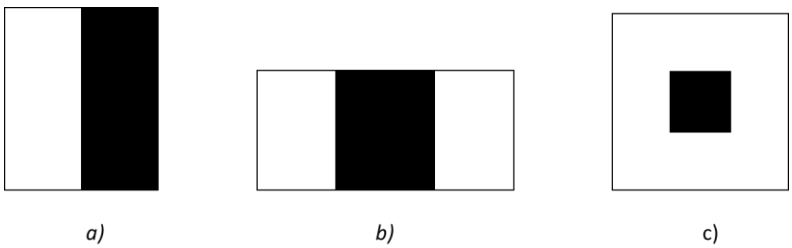


Figura 2.2- Filtros Haar principales para extraer características a) de contorno, b) de líneas, c) en torno al centro

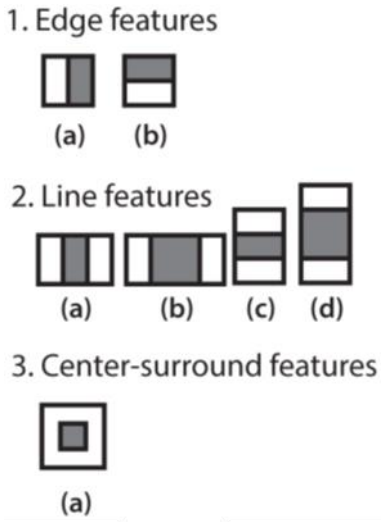


Figura 2.3- Todas las variantes de filtros Haar utilizados en este trabajo. Imagen extraída de [32].

Conocido el método por el que se extraen las características y la herramienta que ayuda agilizar el proceso falta concluir cómo se puede emplear para determinar si en una imagen aparece una cara o no.

Como ya se ha explicado, el proyecto gira en torno a métodos de *machine learning*. En este caso en concreto, se trata de una cascada de clasificadores entrenados mediante un método de *boosting*. Se denomina "entrenamiento" del clasificador al hecho de introducir en el sistema muestras positivas, es decir imágenes que sean una cara, y muestras negativas, imágenes al azar que no contengan una cara. De esta manera, el algoritmo puede determinar qué valores de las características estudiadas van asociadas a los rostros.

La idea principal que introducen las técnicas de *boosting* es la de, a partir de varios clasificadores débiles, construir un clasificador robusto. En concreto el algoritmo empleado es *Adaboost*, introducido por Freund y Shapire [33]. Para explicar este proceso nos apoyaremos en la Figura 2.4. Imagínese que usamos dos de los filtros de Haar y, por tanto, extraemos 2 valores de cada imagen. Representando uno de los valores en el eje de ordenadas y otro en el de abscisas podemos obtener una gráfica similar a la presentada en la figura. Las imágenes que corresponden a rostros se representan con un emoticono rojo mientras que las que no contienen una cara se corresponden con un signo de interrogación azul. En la primera iteración el algoritmo calcula el primer clasificador débil (*w1*, en la Figura 2.4) siendo aquel

que divide los datos cometiendo el mínimo error. Una de las principales características del *boosting* es el cambio de pesos de las muestras. Aquellas que han sido incorrectamente clasificadas en la anterior etapa se vuelven más importantes en la actual, siendo prioritaria su correcta clasificación frente a aquellas que mantienen el peso original. Este proceso se repite hasta que la combinación de los clasificadores débiles es capaz de crear un clasificador robusto (*C*, en la Figura 2.4). El clasificador final debe ser capaz de clasificar correctamente todas las muestras entrenadas. En este caso, se han empleado dos valores para poder ofrecer un ejemplo más gráfico pero los clasificadores pueden entrenarse para un solo valor o, por el contrario, y aunque no en este proyecto, para multitud de ellos.

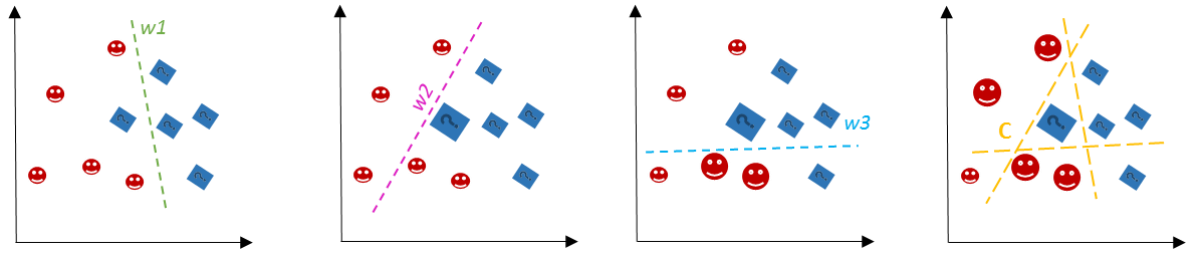


Figura 2.4- Ejemplo gráfico del método de *boosting*

Sin embargo, el método de Viola-Jones no se compone de un único clasificador y utiliza una estructura de clasificadores encadenados, reflejada en la Figura 2.5. En esta cascada cada etapa o nodo se corresponde con un clasificador. Aunque algunas etapas pueden llegar a estudiar hasta 3 valores, generalmente, cada uno de ellos ha sido entrenado para evaluar una única característica. Es decir, la mayoría de clasificadores consisten en saber si el valor devuelto tras aplicar el filtro de Haar se encuentra por encima o por debajo de una cifra. Cuanto se estudian dos características los clasificadores débiles serán rectas como las de la Figura 2.4. Por último, cuando sean tres los valores a analizar, los clasificadores serán planos que separen las muestras.

La cascada funciona de manera que la imagen candidata entra en ella y es evaluada una a una por los clasificadores. Para determinar que una imagen es una cara, esta debe haber superado con éxito todas las etapas. Por el contrario, en cuanto un clasificador determina que una imagen no corresponde a un rostro la imagen abandona la cascada y se considera que no pertenece a una cara. Los nodos de la cascada son entrenados de manera que son

altamente permisivos. De esta manera, imágenes que no contienen rostros habitualmente pasan la etapa como si lo tuvieran, o lo que es lo mismo, dan lugar a muchos falsos positivos. La posibilidad de que una imagen cualquiera sea calificada como una cara en este tipo de sistemas es de aproximadamente un 50%, mientras que una cara es correctamente detectada el 99,9% de las veces [34]. Aunque esto podría parecer negativo, el uso de varios clasificadores lo vuelve un hecho a nuestro favor. Con solo 20 etapas, se consigue reducir los falsos positivos a un 0.0001% manteniéndose la correcta detección de caras en un 98%. El sistema permite que imágenes falsas apenas consuman gasto computacional, al abandonar rápidamente el sistema en los primeros nodos. Solo las imágenes de caras llegan al final de la cascada, habiendo superado tantos clasificadores que la probabilidad de que sean correctas es muy alta.

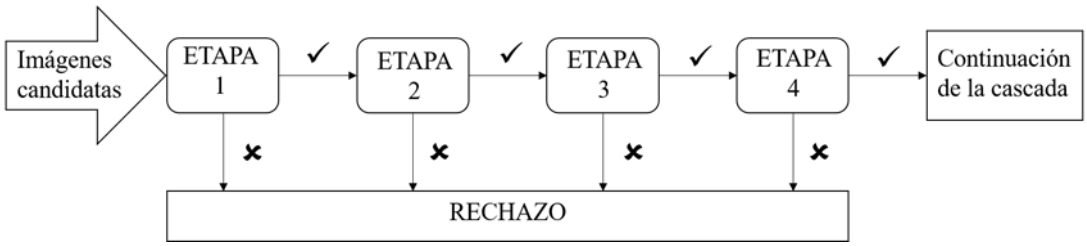


Figura 2.5- Esquema de la estructura de una cascada de clasificadores

Para seleccionar las imágenes candidatas se determina un tamaño mínimo y máximo del objeto a detectar y un factor de escala. Las imágenes se obtienen de recorrer la imagen extrayendo regiones cuadradas del tamaño mínimo. Una vez hecho esto se aplicará el factor de escala, recorriendo la imagen las veces necesarias hasta alcanzar el tamaño máximo, Figura 2.6 (a, b, c). Es habitual que la cara se detecte a diferentes escalas y en varias posiciones cercanas a la real por lo que puede establecerse un valor mínimo de detecciones superpuestas para considerar una zona como detectada. Se determinará la posición de la cara como la región de la imagen en la que más detecciones ha habido Figura 2.6 d.

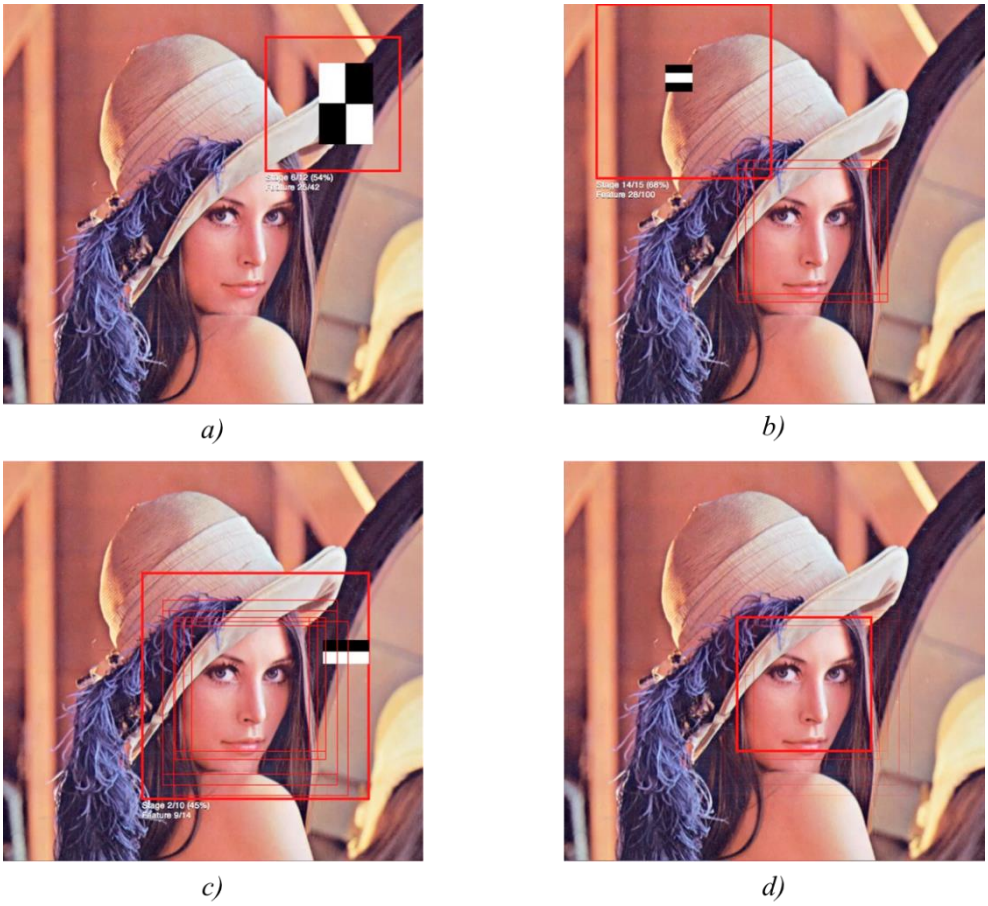


Figura 2.6-Visualización del proceso de detección facial por Viola-Jones. Imágenes extraídas de [35].

A lo largo de toda la explicación se ha hecho continua referencia a la detección facial, pues era el tema central del artículo de Viola y Jones, pero es posible entrenar cascadas de clasificadores para detectar múltiples objetos. Como ya se ha explicado, en este proyecto se emplean las mismas técnicas para la detección de ojos.

2.2 TRANSFORMACIÓN AFÍN

Explicadas ya las cascadas de clasificadores, utilizadas para detectar rostros en las imágenes y ojos en la imagen extraída de la cara, si se acude a las Figura 1.2 y Figura 1.3, se puede comprobar que el siguiente paso es la rotación de la imagen para conseguir que los ojos se encuentren alineados a la misma altura.

El enderezado de las imágenes permite que las posteriores detecciones de puntos y cálculo de la emoción sea más simple. Gracias a esta rotación las imágenes que entrenan al algoritmo

y las que luego serán clasificadas son más homogéneas facilitando la labor de los clasificadores.

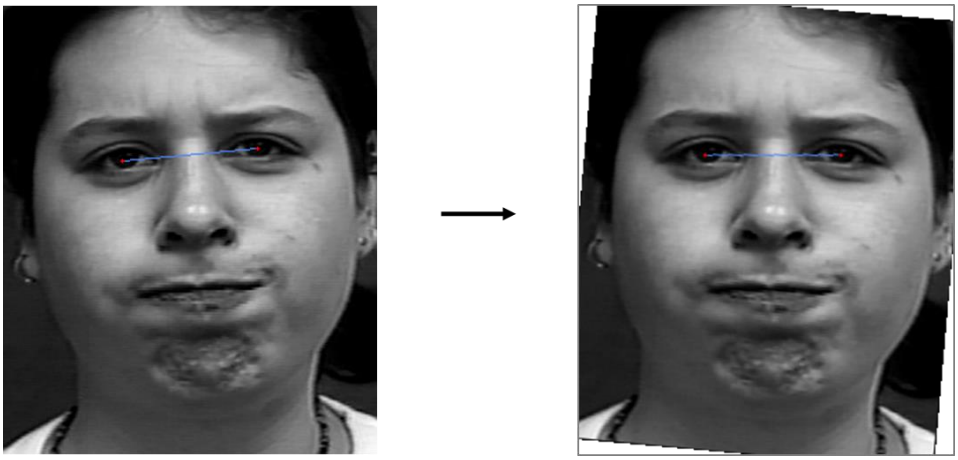


Figura 2.7- Enderezado de la imagen

La rotación se lleva a cabo mediante una matriz de transformación que sitúa el píxel, de coordenadas (x, y), de la imagen original en una nueva posición (x',y'). Las nuevas coordenadas no suelen ser valores enteros por lo que se debe llevar a cabo una interpolación entre los valores vecinos para conservar adecuadamente la información de la imagen.

En este caso la rotación se lleva a cabo mediante una transformación afín que queda definida partir de un punto y ángulo de giro. El punto escogido será el ojo derecho del sujeto y el ángulo el formado entre el segmento que une las dos pupilas y una línea horizontal que atraviese el ojo derecho.

$$r = \begin{bmatrix} cos\theta & sen\theta & Xe(1-cos\theta) - Ye(sen\theta) \\ -sen\theta & cos\theta & Xe(sen\theta) + Ye(1-cos\theta) \\ 0 & 0 & 1 \end{bmatrix} \quad (2.2)$$

El enderezamiento se lleva a cabo con la matriz de rotación r de la ecuación 2.2. Las variables Xe e Ye hacen referencia a las coordenadas del ojo y θ al ángulo entre pupilas.

Ignorando la última fila al encontrarnos en un plano bidimensional, las dos primeras columnas representan la rotación que se realiza en torno a la esquina inferior izquierda de la imagen. La última columna corresponde a la traslación llevada a cabo para mantener el contenido de la imagen en una posición similar a la anterior y, así, poder visualizarla correctamente.

Una vez realizada la operación muchas de las coordenadas no tienen un valor entero. Por tanto, para obtener el valor de ciertas posiciones de la nueva imagen habrá que realizar una interpolación con los valores cercanos. En este caso se realiza una interpolación lineal. Para cada posición, se exploran los cuatro puntos vecinos, de los que conocemos su valor y la posición, con decimales, recientemente calculada, Figura 2.8. Conociendo esto, se puede estimar el valor que le corresponde a la posición entera asumiendo que el brillo de la imagen sigue una función linear.

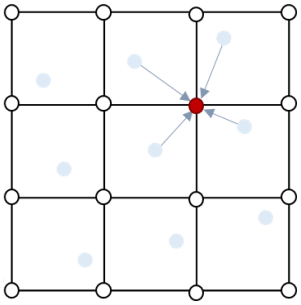


Figura 2.8- El valor del píxel rojo se calcula como la interpolación lineal entre sus 4 píxeles vecinos.

Esta interpolación provoca una ligera caída en la resolución y cierto desenfoque, sin embargo, con las imágenes utilizadas es una pérdida despreciable. No obstante, si se desea obtener más información sobre otros tipos de interpolaciones puede consultarse el libro de Sonka, Hlavac y Boyle [36].

2.3 DESCRIPTORES HOG

Una vez detectada y enderezada la cara, todo está listo para la detección de los 20 puntos restantes. Para ello se seguirá un método basado en el artículo de Dalal y Triggs [15]. En él describen un algoritmo de detección de peatones basado en histogramas de gradientes orientados que, en este caso, se adaptará para ubicar los puntos deseados de la cara.

Tanto en nuestro caso como en el del artículo se usan imágenes de 64x128 píxeles. Para calcular los histogramas de gradientes orientados, o *Histograms of Oriented Gradient (HOG)*, la imagen se divide en bloques cuadrados de 8 píxeles de lado. Esto dará lugar a un total de 128 (8 por fila y 16 por columna) regiones por imagen, Figura 2.9.

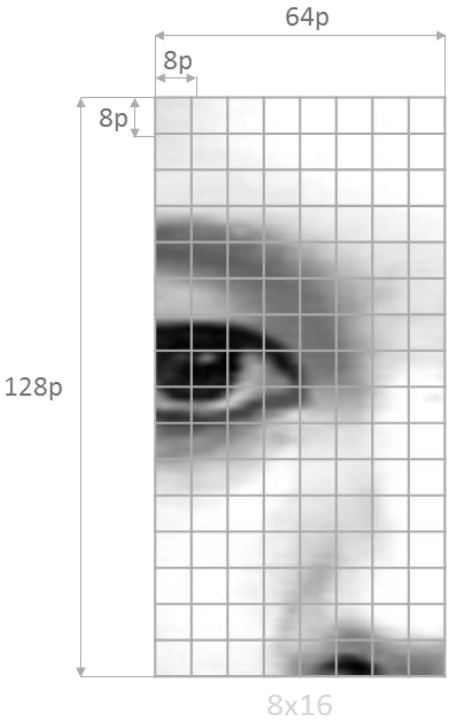


Figura 2.9- División de la imagen en celdas de 8 píxeles de lado

Será en estos cuadrados de 8x8 donde se calculen los gradientes horizontales y verticales para cada píxel. El gradiente de una imagen indica en qué dirección y con qué velocidad varía el nivel de gris de la imagen. Para calcularlo, simplemente se hará uso de los filtros detectores de gradiente de la Figura 2.10. De esta manera, para obtener el gradiente horizontal de un punto calcularemos la diferencia entre su pixel a la izquierda y su pixel a la derecha. Para el vertical haremos lo propio con el inferior y el superior. Conocidos esos dos valores se puede calcular la magnitud y orientación del vector, como se muestra en el ejemplo de la Figura 2.11.

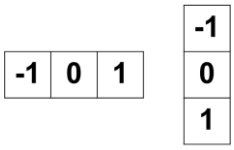
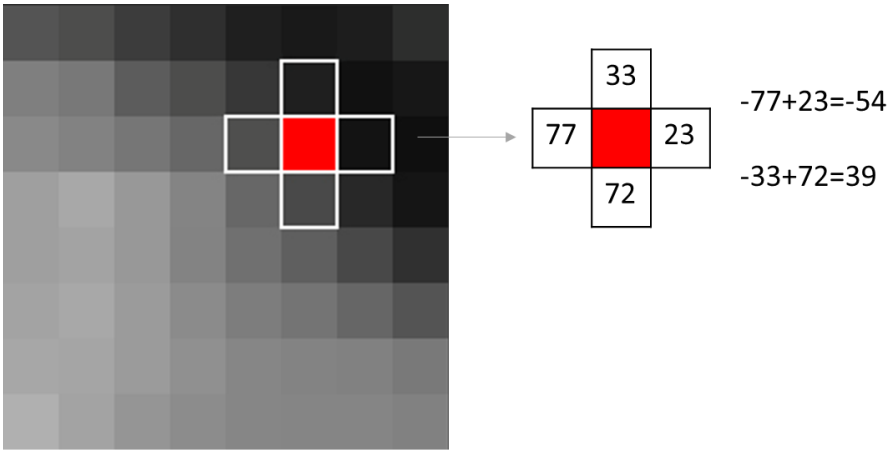


Figura 2.10- Filtros utilizados en el cálculo del gradiente

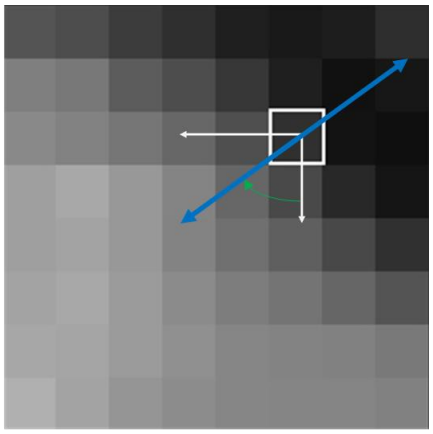


$$\nabla f = \begin{bmatrix} -54 \\ 39 \end{bmatrix}$$

$$|\nabla f| = \sqrt{(-54)^2+(39)^2} = 66{,}61$$

$$\varphi = \arctan\left(\frac{-54}{39}\right) = -0{,}95\ rads = -54{,}43°$$

Figura 2.11- Ejemplo de cálculo del gradiente en el píxel de una de las celdas



[66,61, 54,43°]

Figura 2.12- Información guardada del vector de cada píxel

Esta operación se realiza para todos los píxeles, correspondiéndole a cada celda un total de 64 vectores, cada uno con su magnitud y orientación, Figura 2.12. Esto supondría un total de 16384 valores para toda la imagen. Por este motivo, el método de Dalal y Triggs almacena la información de los vectores en un histograma en el que se representan valores de

orientación separados por 20º. Al tener en cuenta este método únicamente la dirección y no el sentido de los vectores, el histograma constará de 9 valores de orientación (de 0 a 180). Un ejemplo de cuál sería el aspecto de uno de estos histogramas puede verse en la Figura 2.13.

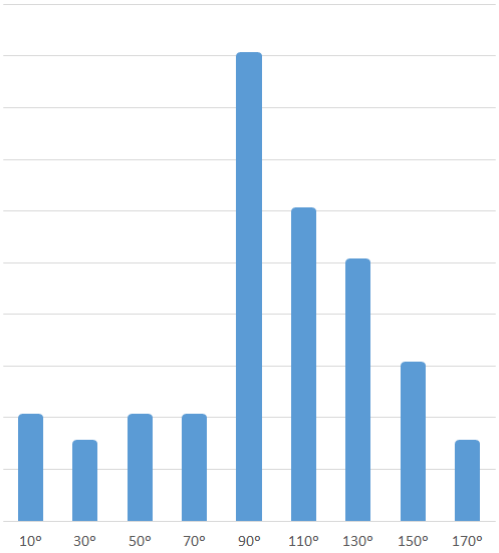


Figura 2.13- Representación del histograma que guarda la información de los vectores de cada celda.

La información de la magnitud de los vectores no se ha perdido en la construcción del histograma, como podría parecer, ya que cada vector contribuye al histograma en función de su magnitud. La orientación de un vector con una alta magnitud se verá más representada que si este tuviese un módulo menor.

También puede intuirse que se produce una pérdida de precisión al solo existir 9 posibles orientaciones ya que, generalmente, la del vector no coincidirá exactamente con ninguna, pudiendo quedar en medio de dos opciones. Para atenuar este efecto, la contribución de ese vector se dividirá proporcionalmente entre esas dos opciones en función de su cercanía a las mismas. Por ejemplo, un vector con orientación de 95º contribuirá con ¼ de su magnitud a la orientación de 110º y con los restantes ¾ a la de 90º, más cercana.

Para paliar los efectos de cambios de iluminación, diferencias entre los posibles fondos tras el peatón o diferencias en el color de la piel del individuo, en nuestro caso, se lleva a cabo la normalización de los histogramas anteriormente calculados. Esta normalización no se hace

bloque a bloque, sino que estos se agrupan de 4 en 4 en celdas, que se construyen superponiéndose un 50% con la anterior como se aprecia en la Figura 2.14.

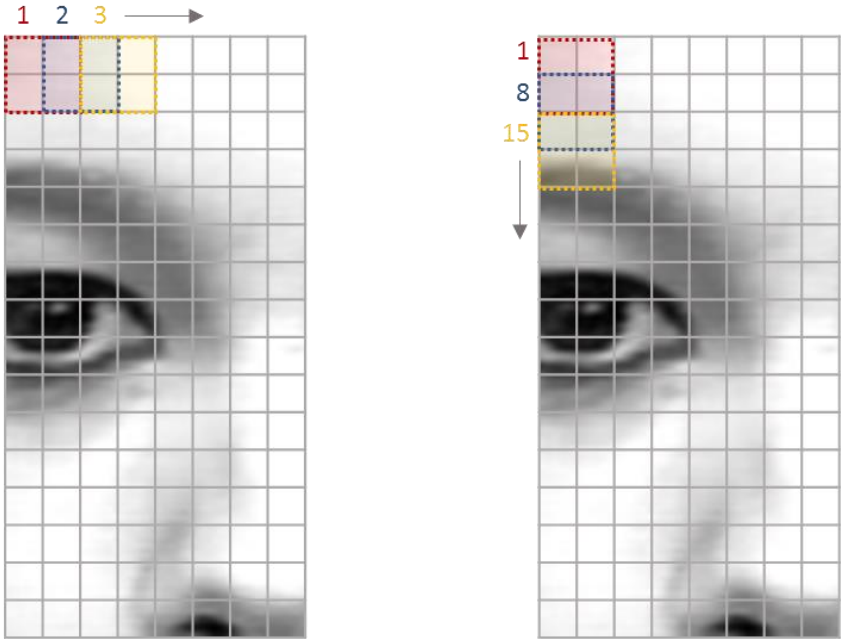


Figura 2.14- Construcción de celdas a partir de la combinación de varios bloques de 8x8.

Por tanto, concatenando los 4 histogramas de cada celda tendremos vectores de 36 componentes que se normalizarán al ser divididos por sus propias magnitudes. Apréciese que, a excepción de las celdas de las esquinas que solo se computan una vez y las de los bordes que solo en dos ocasiones, cada una de las celdas aporta su información 4 veces en el descriptor. Sin embargo, cada vez los valores serán distintos pues se aportan desde distintas normalizaciones. Normalizar de esta forma, por pequeñas regiones, asegura reducir más el impacto de los cambios en el contraste que hacerlo en toda la imagen.

Las 105 celdas, cada una con su vector de 36 componentes, dan lugar a un total de 3780 valores que son los que componen el vector conocido como descriptor de *HOG*.

El procedimiento para la detección es similar al descrito para la detección facial. Muestras positivas y negativas de los descriptores del punto en cuestión entrenan el clasificador. En este caso, se trata de una máquina de soporte vectorial cuyo funcionamiento se explica en detalle en el apartado 2.4.

La imagen se recorre variando el tamaño de las muestras recogidas en cada una de las pasadas. La única diferencia se encuentra en la relación de aspecto de las muestras que, en lugar de ser cuadrada, es de 1:2. En el proceso de clasificación, el tamaño de la muestra ya extraída se escala para que sea de 64x128, el que permite el cálculo del descriptor.

2.4 MÁQUINAS DE SOPORTE VECTORIAL

Las máquinas de soporte vectorial o *support vector machines* (SVM) son el método de *machine learning* empleado en este proyecto tanto en la detección de puntos a través de descriptores HOG como en el cálculo final de la emoción a partir de los desplazamientos de los 22 puntos característicos.

En el primer caso, el vector está compuesto por 3780 componentes. Una vez entrenado el clasificador con descriptores asociados a un punto en cuestión y otros tomados aleatoriamente por toda la cara, el algoritmo puede determinar cuándo una imagen corresponde a dicho punto a partir de su descriptor de histogramas de gradiente orientado. También es el método de clasificación escogido por Dalal y Triggs en la detección de peatones [15].

Aunque pudiera parecer que el cálculo de la emoción es completamente distinto al de la detección de puntos, en realidad, es bastante similar. Una vez se tienen las coordenadas de los puntos en la imagen con emoción neutra y en la imagen con emoción por determinar, estas se normalizan en base a la distancia entre pupilas del sujeto. Esta división permite una homogenización de los datos favoreciendo que el método sea viable con varios tamaños de imagen, ya sea por una mayor resolución o porque el sujeto se encuentra más cercano o alejado de la cámara. Las coordenadas normalizadas de una y otra imagen se restan para calcular el desplazamiento de los puntos. De esta manera, se obtienen vectores de 44 componentes, 2 por cada punto, que sirven para entrenar la *SVM*. Una vez entrenada con suficientes muestras será capaz de determinar si un vector se corresponde con la emoción asociada o si, por el contrario, se trata de otro tipo de expresión. Las *Support Vector Machines* se utilizan en la predicción de emociones en los trabajos de Michel y El Kaliouby [13] y en el de Kotsia y Pitas[24], entre otros.

Para facilitar la explicación del funcionamiento interno de la máquina de soporte vectorial, tomaremos un vector de dos dimensiones (o componentes). Aunque el número de

componentes es mucho menor que el usado en nuestros métodos, se puede apreciar en la Figura 2.15 que el problema es el mismo. El entrenamiento proporciona muestras de dos clases y el objetivo de la máquina de soporte vectorial es establecer una frontera que permita clasificar futuras muestras.

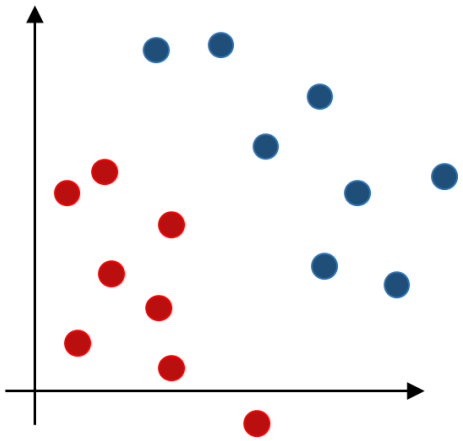


Figura 2.15- Ejemplo clásico de problema de clasificación.

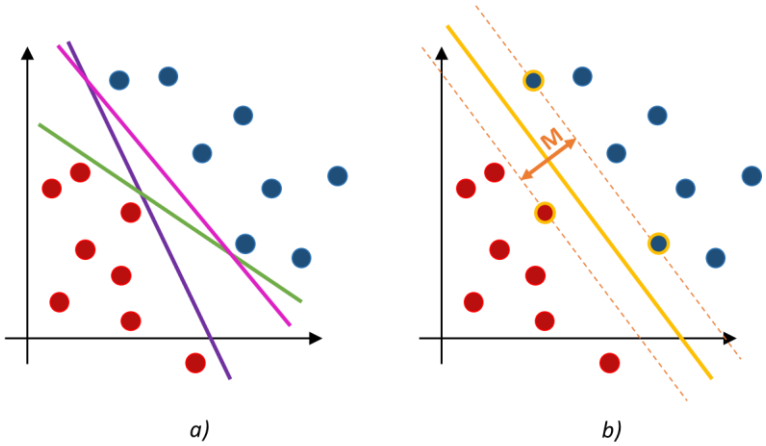


Figura 2.16- A la izquierda, posibles funciones lineales que separan los datos. A la derecha, función que los divide de manera más eficiente.

Como se ve en la Figura 2.16a se pueden definir varias funciones lineales que separen dichos datos. Se considerará como correcta aquella que maximice el ancho del área vacía entre las dos clases o, lo que es lo mismo, la distancia que hay entre la función que ejerce de frontera y la muestra o muestras límite de cada una de las clases, Figura 2.16b. A esta distancia se le denomina margen, acotado por la letra *M* en la Figura 2.16b. Por su parte, a las muestras

límite, que se representan con un contorno dorado en las figuras de la memoria, se les denomina vectores de soporte o *support vectors*.

Para entender el proceso de maximización del margen es necesario hacerlo desde un enfoque matemático. En este ejemplo, la función de discriminación no deja de ser una recta que divide ambas clases. Como tal, puede expresarse en función de las dos variables x e y, y de tres constantes p, q y b:

$$px + qy + b = 0 \tag{2.3}$$

La recta también puede expresarse como una multiplicación de vectores. Sea:

$$\vec{w} = \begin{bmatrix} p \\ q \end{bmatrix} \tag{2.4}$$

$$\vec{X} = \begin{bmatrix} x \\ y \end{bmatrix} \tag{2.5}$$

Podemos establecer que:

$$px + qy + b = 0 = \vec{w} \cdot \vec{X} + b \tag{2.6}$$

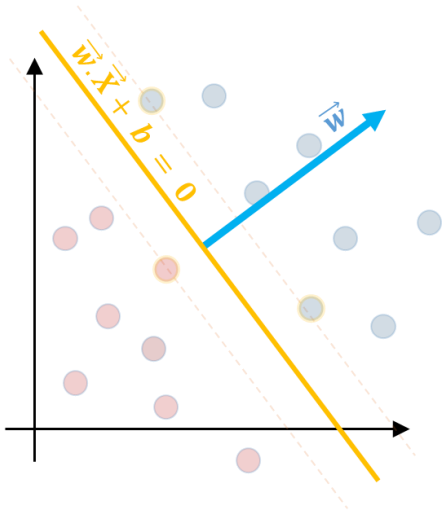


Figura 2.17- Representación de la función de discriminación y de la normal $\vec{w}$ que la define.

Como se puede apreciar en la Figura 2.17, el vector $\vec{w}$ representa la normal de la recta. Los coeficientes se ajustan para que el resultado de la ecuación en los *support vectors* sea igual a la unidad, Figura 2.18. De esta manera, a las muestras de una clase le corresponderán

valores positivos y a los de la otra, valores negativos. Siguiendo esto, a partir de ahora se denominará a las muestras azules, muestras positivas, y a las rojas, muestras negativas.

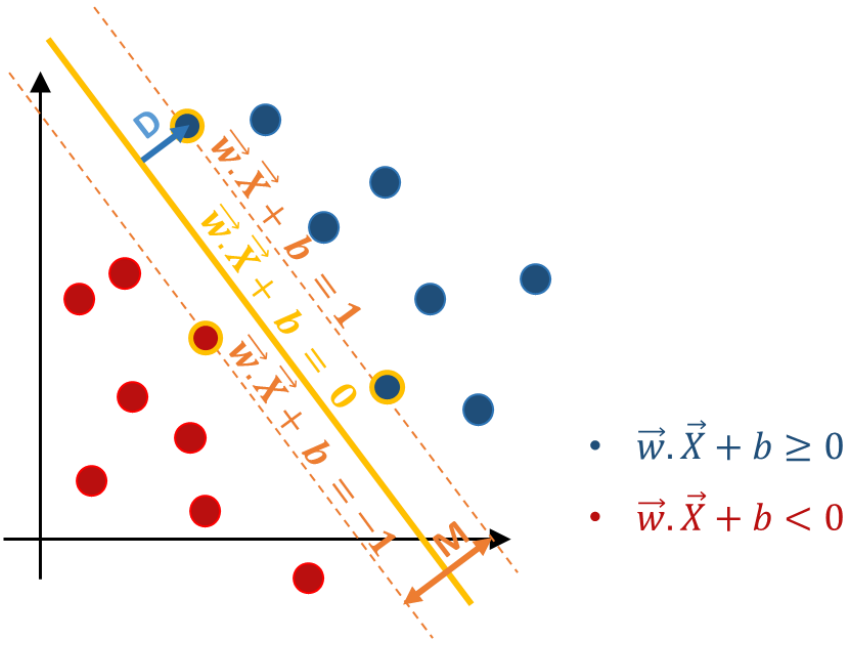


Figura 2.18- Representación de las funciones en los vectores de soporte.

Para entender cómo se encuentra la función maximiza el margen conviene recordar que la distancia entre un punto y una recta se expresa, con nuestra nomenclatura, como:

$$D = \frac{|qx_0+qy_0+b|}{\sqrt{p^2+q^2}} = \frac{|\vec{w}^T\vec{X}+b|}{\|\vec{w}\|} \quad (2.7)$$

Por tanto, la distancia a uno de los vectores de soporte, acotada en la Figura 2.18, será:

$$D = \frac{\pm 1}{\|\vec{w}\|} \quad (2.8)$$

Y por tanto el margen podrá expresarse como:

$$M = \left|\frac{1}{\|\vec{w}\|} - \frac{-1}{\|\vec{w}\|}\right| = \frac{2}{\|\vec{w}\|} \quad (2.9)$$

Con estos datos, se construye el enunciado clásico que define el problema de las máquinas de soporte vectorial. El objetivo es encontrar el $\vec{w}$ y el b que minimice la función 2.10.

$$\frac{1}{2}\vec{w}^T\vec{w} \tag{2.10}$$

Cumpliéndose que para cada $(\vec{X}_i, y_i)$ debe cumplirse que:

$$y_i\left(\vec{X}_i \cdot \vec{w} + b\right) \geq 1 \tag{2.11}$$

Donde $\vec{X}_i$ representa a cada una de las muestras de entrenamiento e y_i es una variable auxiliar creada para que sirva de etiqueta. Para las muestras positivas se estipula que $y = 1$ mientras que para las negativas tendrá el signo opuesto, $y = -1$.

Nótese por tanto que, al ir multiplicado por y_i, se cumplen las condiciones para cualquier muestra de entrenamiento, $\vec{X}_i$, en base a lo expuesto en la Figura 2.18. Los resultados positivos mantendrán su signo y los negativos los cambiarán, al ir multiplicados por su etiqueta y.

En resumen, se trata de llevar a cabo la optimización cuadrática de una función sujeta a condiciones lineales. Este tipo de problemas han sido profundamente estudiados y hay desarrollados muchos algoritmos para su resolución, incluyendo algunos de aplicación específica para *SVM*. En este caso la solución se resuelve haciendo uso de la función Lagrangiana. Para más detalles sobre el método de resolución puede consultarse el artículo de Enrique Carmona [37].

El resultado de la optimización es:

$$\vec{w} = \sum \alpha_i y_i \vec{X}_i \tag{2.12}$$

Los valores que componen el vector α_i son los conocidos como multiplicadores de Lagrange. Estas constantes son nulas excepto en el caso de los vectores de soporte, recibiendo en este caso un valor positivo.

Conocido $\vec{w}$, puede calcularse fácilmente el valor de la constante b. Como se representó en la Figura 2.18, la función en los *support vectors* proporcionará un resultado unitario. Aplicando esto para un vector de soporte de la clase, por ejemplo, positiva se puede despejar el valor de b.

$$\vec{w} \cdot \vec{X} + b = +1 \rightarrow b = 1 - \vec{w} \cdot \vec{X} \tag{2.13}$$

Por tanto, puede establecerse que la función que determina la decisión, 2.14, puede expresarse ahora como se ve en 2.15 . Recuérdese que $\vec{X}$ representa a un vector cualquiera, el que se pretende clasificar, y X_i representa a las muestras que provienen del entrenamiento. Al ir X_i multiplicado por α_i solo tienen relevancia los vectores de soporte,

$$\vec{w} \cdot \vec{X} + b \tag{2.14}$$

$$\sum \alpha_i y_i \overrightarrow{X_i} \cdot \vec{X} + b \tag{2.15}$$

Así que, finalmente, y conociendo que el nuevo vector arroja resultados positivos o negativos en base a la clase a la que pertenezca, se puede definir la función de clasificación estudiando el signo de la anterior ecuación.

$$f(x) = sign\left(\vec{w} \cdot \vec{X} + b\right) = sign\left(\sum \alpha_i \overrightarrow{X_i} \cdot \vec{X} + b\right) \tag{2.16}$$

Se concluye que el elemento crítico de la función de decisión es el producto escalar entre el vector a clasificar y los vectores de soporte multiplicados cada uno por su respectiva constante. De esta manera, el resto de muestras de entrenamiento influyen en la construcción de la función de decisión, pero no afectarán a las futuras decisiones que se tomen. Esto supone un ahorro en el costo computacional del método, disminuyendo cuantitativamente el número de operaciones por clasificación.

El problema se ha explicado para dos dimensiones, pero, como se comentó al comienzo del apartado, el algoritmo es aplicable a tantas dimensiones como sea necesario. Los vectores $\vec{w}$ y $\vec{X}$ pueden estar compuestos por más componentes y el procedimiento matemático descrito sería idéntico. Al incrementarse el número de variables, la función de discriminación, que en 2D se define como una línea recta y en 3D un plano, pasa a ser un hiperplano. Siguiendo la lógica anterior, para separar vectores de *N* componentes será necesario un hiperplano de *N-1* dimensiones.

El caso presentado hasta ahora solo permite clasificar vectores mediante una función de discriminación lineal. Sin embargo, esto no es posible en muchos casos. Por ejemplo, en la Figura 2.19 no existe ningún límite que pueda separar sin error ambos conjuntos en el espacio 1D.

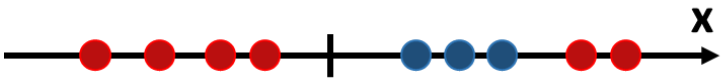


Figura 2.19- Ejemplo de problema de clasificación en 1D sin solución lineal.

Para resolver este conflicto, los vectores se llevan en un espacio dimensional superior. La nueva componente se define, en este caso, como el cuadrado de la original. En este nuevo espacio sí existe un clasificador lineal capaz de separar correctamente los vectores, Figura 2.20.

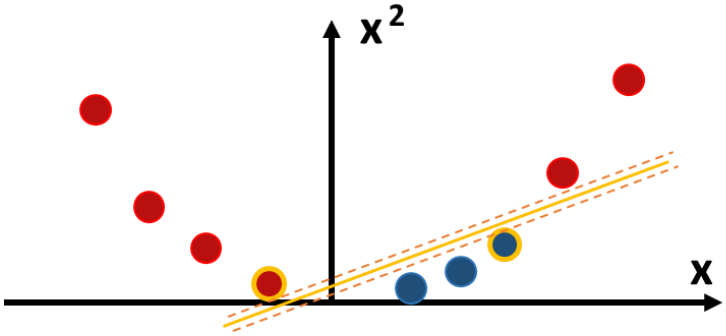


Figura 2.20- Uso de un espacio dimensional superior (2D) para resolver el problema.

Este proceso también puede generalizarse para cualquier número de dimensiones. En la Figura 2.21 puede verse de nuevo un conjunto de muestras que no permiten llevar a cabo una clasificación lineal.

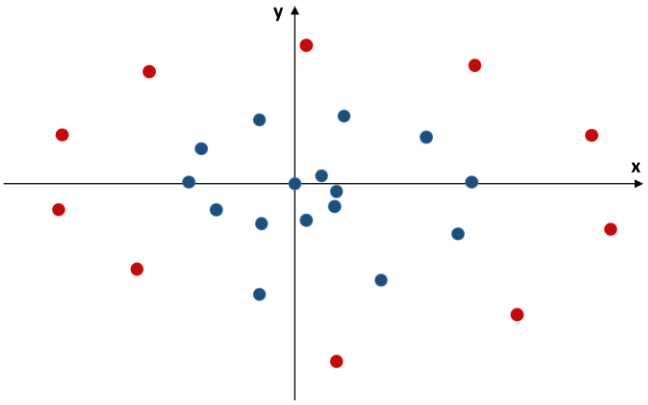


Figura 2.21- Ejemplo de problema de clasificación en 2D sin solución lineal

Mediante una operación como la que se ve en 2.17, los vectores pasan a ser la entrada una función que devuelve un nuevo vector en un espacio dimensional superior.

$$\Phi: \vec{X} \rightarrow \varphi(\vec{X}) \tag{2.17}$$

El resultado puede verse en la Figura 2.22 y 2.23. En la nueva dimensión, *z*, se representa la distancia al origen permitiendo ahora que los vectores sí sean linealmente separables, en este caso mediante un plano.

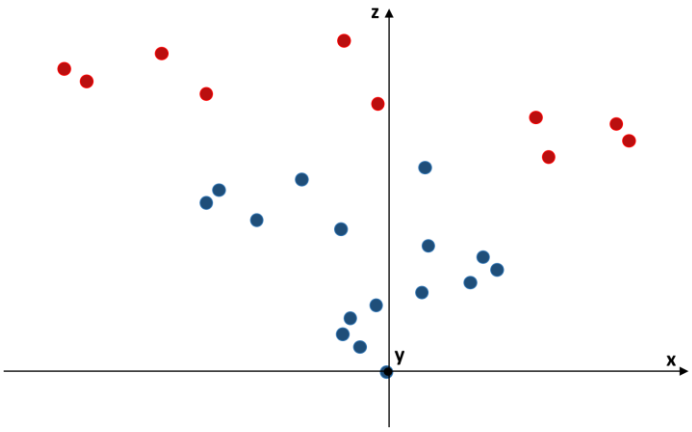


Figura 2.22- Uso de un espacio dimensional superior (3D) para resolver el problema

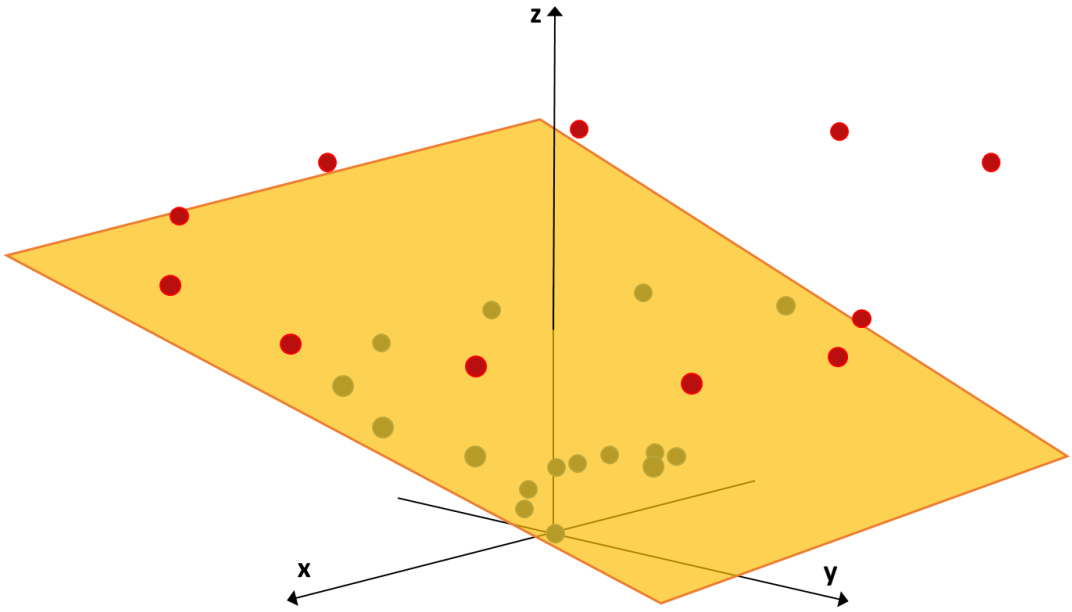


Figura 2.23- Representación del plano que permite dividir mejor las muestras.

Como ya se ha comentado, la base de las máquinas de soporte reside en el producto entre el vector de entrada y los vectores de soporte. Tan vital es esta operación en las *SVM* que recibe el nombre de *kernel*, de manera que un producto escalar se representa como:

$$K(\vec{X}_i, \vec{X}_j) = \vec{X}_i \cdot \vec{X}_j = \vec{X}_i^T \vec{X}_j \quad (2.18)$$

Y, por tanto, cuando se aplica una función como en 2.17:

$$K(\vec{X}_i, \vec{X}_j) \rightarrow \varphi(\vec{X}_i)^T \varphi(\vec{X}_j) \quad (2.19)$$

Los *kernel* no lineales no dejan de ser una operación que en un espacio dimensional superior es un producto vectorial y, por tanto, puede actuar como un separador lineal de ambas clases.

En este proyecto se consideraron diferentes *kernels* que se pueden ver en la Tabla 2.1. Los resultados obtenidos en el cálculo de emociones por cada variante se analizan en el capítulo de "Experimentación y resultados".

Lineal	$K(\overrightarrow{X_i}, \overrightarrow{X_j}) = \overrightarrow{X_i} \cdot \overrightarrow{X_j}$
Función de base radial	$K(\vec{X}_i, \overrightarrow{X_j}) = e^{-\gamma \lVert \overrightarrow{X_i} \cdot \overrightarrow{X_j} \rVert^2}$
Sigmoide	$K(\vec{X}_i, \vec{X}_j) = \tanh(\gamma \overrightarrow{X_i} \cdot \vec{X}_j + k)$
Distribución chi cuadrado	$K(\vec{X}_i, \overrightarrow{X_j}) = e^{-\gamma \lVert \overrightarrow{X_i} \cdot \overrightarrow{X_j} \rVert^2}$
Intersección de histogramas	$K(\vec{X}_i, \vec{X}_j) = \sum \min(\overrightarrow{X_i}, \overrightarrow{X_j})$

Tabla 2.1- Conjunto de *kernels* evaluados en el proyecto

En la detección de puntos no se trabaja en un espacio dimensional superior al original (3780 dimensiones) y se trabaja con el *kernel* lineal. Para los clasificadores del cálculo de la emoción sin embargo se hace uso del *kernel* de intersección de histogramas. Con esta función se comparan uno a uno los valores de ambos vectores y se toma el de menor valor. El resultado del *kernel* es la suma conjunta de todos los valores mínimos.

Para concluir la explicación teórica de las *SVM* nos centraremos en los sistemas multiclase, es decir, que permiten clasificar en más de dos categorías. Aunque los ejemplos de clasificación binaria se adaptan perfectamente a la detección de puntos (la imagen se

corresponde con la una zona de la cara o no) podría parecer que no son adecuados para el cálculo de la emoción, al pretenderse clasificar entre 6 emociones.

Sin embargo, los algoritmos empleados para la clasificación de la expresión facial no hacen más que simular esta elección entre múltiples clases con el uso de varios clasificadores binarios. Existen dos principales variantes entre este tipo de estrategias. Por un lado, los sistemas *"One vs. Many"* y por otro los *"One vs. One"*.

El primer sistema construye una máquina por cada una de las clases que se quieran diferenciar. Las muestras de cada clase se usan como muestras positivas para su clasificador y negativas para el resto. El método funciona de manera que cuando se introduce un nuevo modelo este es evaluado por todos los clasificadores. Mediante métodos de aproximación estadística cada clasificador devuelve un valor indicando la confianza de la clasificación. Se determina que el modelo entrante pertenece a aquella clase cuyo clasificador haya devuelto una mayor confianza.

El método *"One vs. One"*, el empleado en este proyecto, no hace uso de estas estrategias de aproximación. Para diferenciar n clases se construyen $n(n-1)/2$ clasificadores. De esta forma, cada clasificador se hará cargo de uno de los posibles enfrentamientos binarios, quedando cubiertas todas las combinaciones de clases. El sistema determina la clase a la que pertenece una muestra como aquella que ha sido más veces elegida por los clasificadores.

3. Experimentos y resultados

Para poner a prueba el método escogido se ha desarrollo un código que permita conocer más en profundidad los métodos a través de las complicaciones en el diseño del algoritmo y los resultados obtenidos. El programa se ha realizado en el lenguaje de programación C++ haciendo uso de la librería OpenCV [38]en su versión 3.2.0 y en el sistema operativo Ubuntu 16.04 LTS.

Para llevar a cabo las pruebas se han utilizado imágenes de la base de datos de Cohn-Kanade, [28], [29], que alberga secuencias de imágenes de 123 sujetos. Todas las secuencias contienen imágenes frontales del rostro de una persona que van desde una primera, que contiene la emoción neutra, hasta una última imagen en la que las *AUs* asignadas a la secuencia aparecen con mayor intensidad. El conjunto de las imágenes garantiza suficiente variedad en género y etnia para crear un método que funcione de manera universal en estos aspectos. Dispone además de imágenes de sujetos entre 18 y 50 años que cubren al grueso de la población adulta.

La base de datos proporciona además información sobre las secuencias indicándose que unidades acción aparecen en cada una de ellas, incluyendo algunas directamente la emoción presente.

Por motivos legales solo es posible que aparezcan en la memoria imágenes de 11 sujetos concretos, lo que explica la falta de variedad en este aspecto.

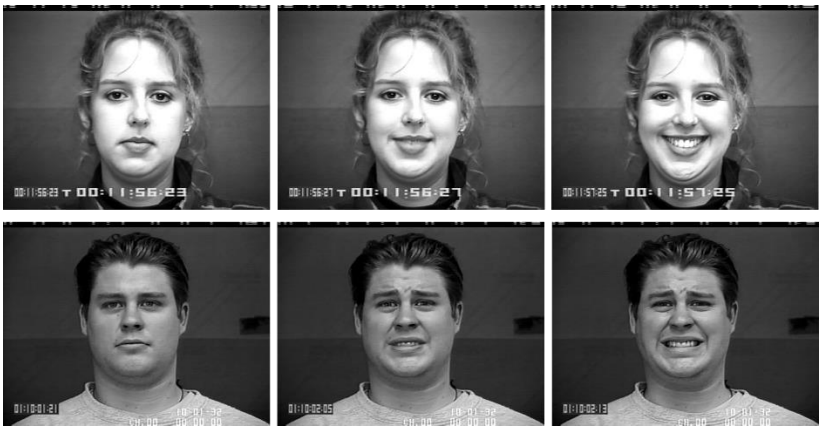


Figura 3.1-Imagen inicial, intermedia y final de dos secuencias de la base de datos

3.1 CLASIFICADORES EN CASCADA

3.1.1 Detección facial

Para estudiar los resultados de la detección facial se tomará un conjunto de 20 imágenes, distintas a las usadas para calibrar los métodos, de las cuales 10 corresponderán a sujetos que muestran una emoción neutra y las otras 10 a imágenes en las que sí aparecen una o varias unidades de acción.

Se emplea el clasificador entrenado *haarcascade_frontalface_alt.xml*, incluido en la librería OpenCV. Los resultados son excelentes obteniéndose un 100% de aciertos en las 20 imágenes. Además, cabe destacar que, a lo largo del desarrollo del código en ningún momento, a pesar de la inmensa cantidad de imágenes procesadas, la detección facial supuso un problema. Se puede considerar, por tanto, que la detección es prácticamente perfecta para esta base de datos.

Estos resultados son coherentes dado que el método Viola-Jones está altamente contrastado. También debe tenerse en cuenta que, como se aprecia en la Figura 3.1, la base de datos está compuesta por imágenes frontales de la cara y tomadas en un estudio de fotografía.

Cabe destacar que se realiza una modificación a la imagen de la cara. A las imágenes se les aumenta, siempre que sea posible y no exceda los límites de la imagen original, un 20% su altura. Esto se realiza porque las imágenes cuadradas que devuelve el clasificador, en ocasiones, cuando presentan expresiones exageradas, no contienen todos los puntos característicos que se pretenden estudiar, Figura 3.2.

Figura 3.2- Expresión exagerada en la que se representa la imagen original (blanco) y la imagen que se toma para evitar la pérdida de información (azul).

3.1.2 Detección ocular

Para la detección ocular se probó, junto a los clasificadores en cascada finalmente elegidos, métodos de detección basados en el estudio de las proyecciones de la imagen y en la transformada de Hough. Antes de analizar los resultados obtenidos se realizará una breve explicación de los métodos alternativos para poder entender mejor las causas de su fracaso.

3.1.2.1 Análisis de proyección vertical y horizontal de la imagen

Esta estrategia de detección ocular se basa en el artículo de Vukadinovic y Pantic[18]. En él se propone la división de la cara en cuatro cuadrantes iguales y se supone que los ojos recaerán en los dos superiores. En las pruebas realizadas en este trabajo se comprobó que esta condición se cumplía en todas las ocasiones.

El concepto de la proyección vertical y horizontal de la imagen es aplicado para determinar la posición del ojo dentro del cuadrante. Se estima que la presencia de la pupila y el iris en contraste con el blanco del ojo supone los mayores cambios en los valores de los píxeles. La proyección vertical calcula la suma de las intensidades en cada fila. La coordenada 'y' del ojo se situará, por tanto, en la fila en la que se encuentra la mayor diferencia de intensidad con respecto a la siguiente. Para la coordenada x se realiza un proceso análogo con la proyección horizontal, atendiendo, en este caso, a la diferencia entre la suma de columnas.

$$p_h(i) = \sum_j f(i,j), \qquad p_v(j) = \sum_i f(i,j) \tag{2.20}$$

3.1.2.2 Transformada de Hough para círculos

El otro método analizado también hace uso de los dos cuadrantes superiores de la imagen. Se puede emplear la transformada de Hough para detectar la forma circular del iris.

Para ello es necesario realizar con antelación una detección de contornos. La mayoría de detectores siguen una filosofía similar al cálculo de gradientes en los descriptores de *HOG*. Se aplican operadores a todos los píxeles calculando sumas y diferencias de los píxeles aledaños. Un repaso a los principales métodos de detección de contornos puede encontrarse

en el ya citado libro de Sonka, Hlavac y Boyle[42]. El detector empleado en este proyecto es el de Sobel, siendo sus operadores vertical y horizontal los siguientes:

$$G_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} \qquad G_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -2 \end{bmatrix} \tag{2.21}$$

Una vez detectados los contornos, la imagen se umbraliza, es decir, solo aquellos contornos detectados con mayor intensidad permanecerán en la imagen.

La transformada de Hough crea una matriz de datos, inicializada a cero, en la que guardará la información extraída de la imagen. El detector analizará la imagen buscando curvas que cumplan la función *f(x, a),* donde *a* es un vector de *n* dimensiones con los parámetros de la curva. En el caso de la detección de circunferencias la ecuación sería la siguiente, donde, (a, b) es el centro del círculo y r el radio.

$$(x - a)^2 + (y - b)^2 = r^2 \tag{2.22}$$

El valor de r varía entre un valor máximo y uno mínimo para evitar detecciones que no correspondan a las de un ojo.

Se recorrerá toda la imagen y, para cada píxel(x, y) que cumpla la función, se aumentan el valor del acumulador A(a, b, r). Finalizado el proceso, dentro de la matriz de datos una combinación de a, b y r será la más repetida. En otras palabras, un centro y un radio serán los que más se repiten en el acumulador. La hipótesis testeada en este trabajo es que, en una imagen de uno de los cuadrantes superiores de la cara, esta circunferencia corresponderá con el iris del individuo.

3.1.2.3 Resultados

Con el fin de analizar los resultados obtenidos en la detección de las pupilas se definirán 3 grados de acierto que se describen en la Figura 3.3. La comprobación se realiza de manera manual.

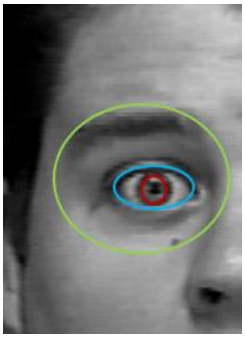
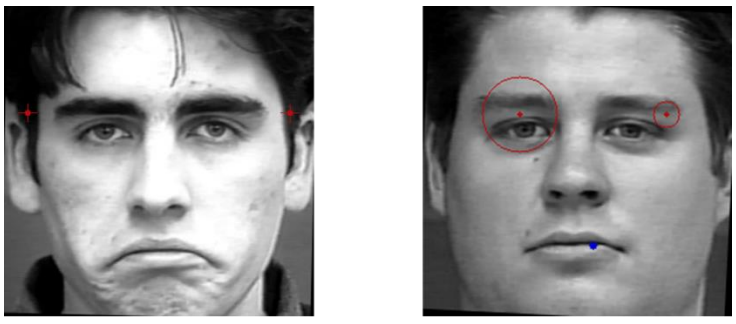


Figura 3.3- Se considerará un acierto de tipo I, aquellos que se localicen dentro del círculo rojo, de tipo II a los que lo hagan dentro del azul y de tipo III a los que lo hagan en el verde.

	Acierto tipo I	Acierto tipo II	Acierto tipo III
Estudio de la proyección horizontal y vertical	0,0%	20,0%	20,0%
Transformada de Hough	42,5%	62,5%	87,5%
Clasificadores en cascada	95,0%	95,0%	95,0%

Tabla 3.1- Resultados de los distintos métodos de detección ocular



a) *b)*

Figura 3.4- Errores más comunes en los siguientes métodos: a) Proyecciones de la imagen, b) Hough

Como se puede observar en la Tabla 3.1, los resultados del análisis de las proyecciones hacen dudar de si es un método digno de ser ni siquiera mencionado. Sin embargo, su inclusión se debe a que durante una primera fase del proyecto se utilizaban regiones de la cara recortadas manualmente. Estas eran significativamente más pequeñas que las proporcionadas por el detector finalmente implementado. Utilizando esas imágenes más concretas el algoritmo alcanzaba resultados medianamente aceptables, pero regiones más grandes incluyen gran parte del fondo y del pelo del usuario. Esto hace que las máximas diferencias suelen ubicarse en la zona de las patillas, Figura 3.4a.

La transformada de Hough ofrece, sin embargo, resultados moderadamente mejores. El gran problema de este método es que, a parte de los ojos, existen muchos posibles círculos en la cara, Figura 3.4b. En ocasiones, estos círculos se presentan más claros para el detector de Hough que los de los ojos, pues estos no tienen una forma completamente circular. Esto hace que el método sea muy poco estable y, de hecho, en los casos estudiados, ambos ojos en una persona sólo se detectan correctamente el 20% de las veces. Además, muchos de los aciertos del detector se deben a que detectan el brillo que en ocasiones aparece en la pupila cuando se realiza una fotografía. La eficacia del detector no puede depender de ese hecho que no siempre se produce.

Por último, los clasificadores en cascada ofrecen unos resultados muy superiores. Concretamente, en este trabajo se utiliza otro clasificador de OpenCV ya entrenado, el *haarcascade_eye_tree_eyeglasses.xml*. El 5% de error se debe a que en una de las 20 imágenes el sujeto aparece con los ojos cerrados.

El impedimento de los ojos cerrados no debe obviarse. Es muy común en ciertas expresiones faciales que los ojos se entrecierren o se cierren por completo, sin obviar que los parpadeos supondrían un gran problema. La solución adoptada en este proyecto es la de omitir la rotación en este caso y sustituir las coordenadas del ojo por las más comunes estadísticamente. Ese paso se realiza para todos los puntos característicos por lo que el proceso se explicará más en detalle en el siguiente sub-apartado.

3.2 DETECCIÓN DE PUNTOS CARACTERÍSTICOS

Se hace uso del código de Jan Hendricks [39] que no es más que una implementación directa del artículo de Valstar haciendo uso de la librería OpenCV para la extracción de los

descriptores. El método se sirve de la librería SVMlight [40] para el entrenamiento del clasificador. Para hacer más cómodo el trabajo se dota los 22 puntos de la Figura 1.1 de una nomenclatura, Figura 3.5.

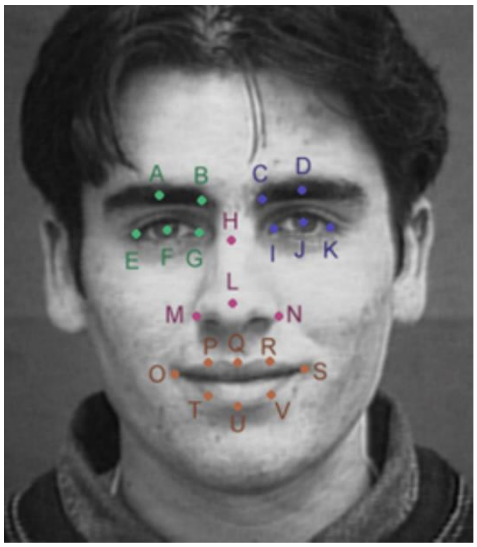


Figura 3.5-Nomenclatura empleada para los 22 puntos característicos.

Como ya se ha explicado, siendo estrictos no serán los puntos lo que se detecte si no el área de 64x128 que les rodea, siendo nuestro punto el centro de dicha región. Un ejemplo de esto puede verse en la Figura 3.6. El tamaño de todas las muestras será de 640x490, el más habitual en la base de datos, por lo que directamente se extraerán imágenes de 64x128 sin necesidad de re-escalados.

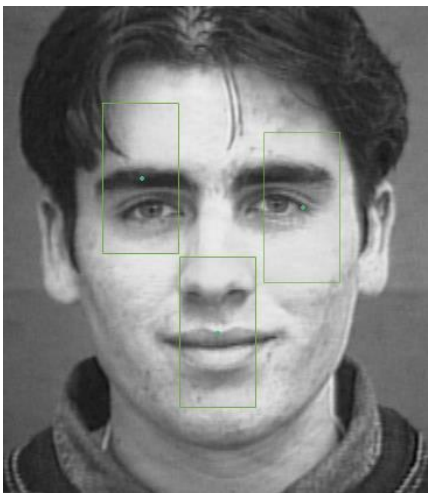


Figura 3.6-Áreas de 64x128 asociadas a los puntos *A*, *K* y *Q*.

Un factor que debe tenerse en cuenta es el de la simetría del rostro humano. Debido a esto, el descriptor de *HOG* de ciertas imágenes será muy similar al de otras si previamente se invierten en sentido horizontal, Figura 3.7. Esta particularidad puede aplicarse a los 9 puntos que se encuentran a la izquierda del eje imaginario formado por *H*, *L*, *Q* y *U*. De esta manera, en lugar de 20 detectores (recuérdese que los ojos ya se han detectado), solo serán necesarios 12.

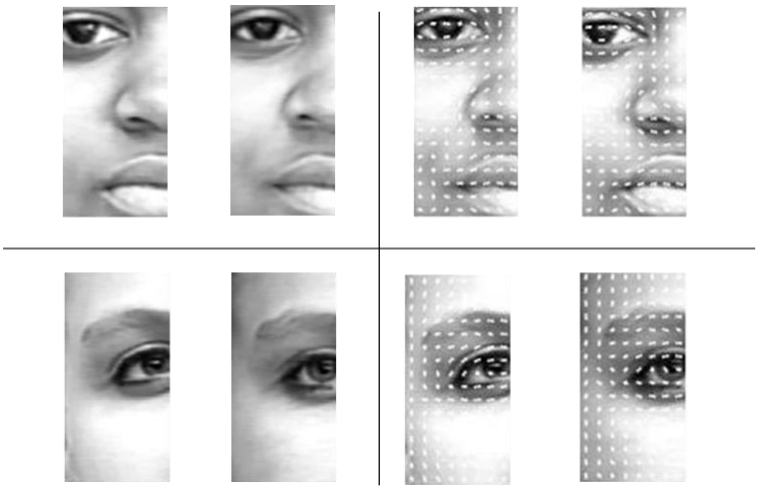


Figura 3.7- Ejemplo de la similitud entre el punto *M* y *N* invertido (arriba) y *E* y *K* invertido (abajo). A la derecha pueden apreciarse los parecidos en la representación del descriptor de *HOG*.

Para intentar mejorar la eficacia de los clasificadores las imágenes usadas para entrenar la máquina de soporte vectorial son lo más variadas posibles. Las muestras positivas de cada punto se toman con ejemplos de varias expresiones y tono de piel. Las negativas, al estar pensada la detección en imágenes de la cara, serán imágenes de otros puntos de la cara. En el proceso de toma de muestras negativas, una vez se ha adquirido una cantidad aceptable de imágenes aleatorias, los esfuerzos se centran en incluir puntos que puedan ser parecidos al que se desea detectar. De esta forma se intentarán definir los mejores vectores de soporte posibles y por tanto aumentar la exactitud del clasificador.

En la fase de entrenamiento el número de muestras positivas se mantiene igual que el de negativas para no construir un clasificador muy estricto ni uno muy permisivo con los falsos positivos. El número de muestras utilizado para cada clasificador se muestra en la Figura 3.8.

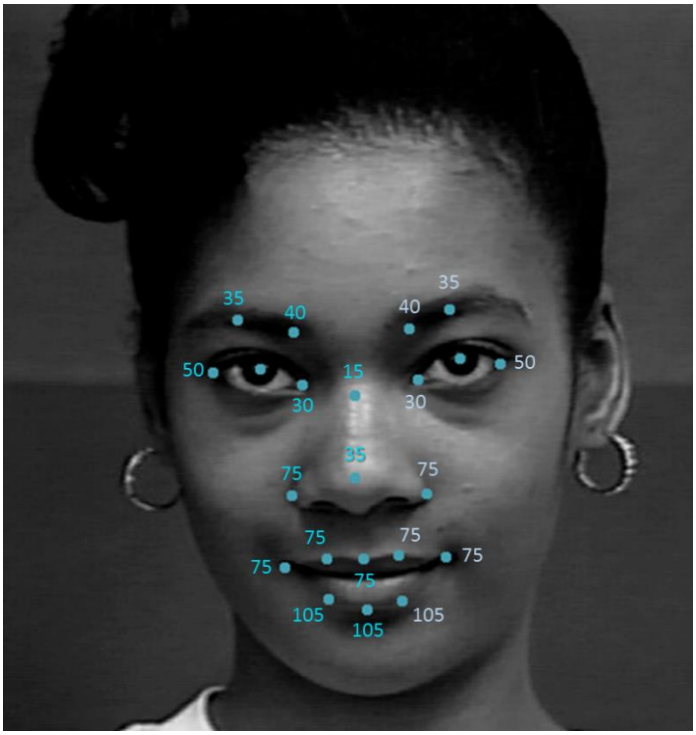


Figura 3.8- Imagen con el número de muestras positivas utilizadas para cada punto. En aquellos puntos que se detectan con el clasificador de su simétrico el número aparece más grisáceo.

La validación de los resultados se realiza en dos etapas. Una primera aproximación se realiza evaluando la calidad de la detección en tiempo real mediante el uso de la webcam del ordenador. Si la localización no augura buenos resultados, se vuelve de nuevo a la fase de entrenamiento intentando contrarrestar, en la toma de muestras, los errores expuestos. Si el detector, por el contrario, se muestra fiable se pasa a una segunda etapa en la que se valida el clasificador en 20 imágenes. Los resultados finales se obtienen evaluando los puntos en 35 imágenes, asegurándose de nuevo de que hay suficiente variedad en las muestras. Se definen, de nuevo, diferentes grados de acierto. El grado 3 se corresponderá con una detección perfecta, el 2 con una detección aceptable y el 1 con una mala detección. El grado 0 se reserva para aquellas ocasiones en las que no se detecta ningún punto.

Puntos	✓	3	2	1	0
A	**88,57%**	80,0	8,57	0,00	11,43
B	**85,71%**	74,29	11,43	2,86	11,43
C	**80,00%**	77,14	2,86	8,57	11,43
D	**74,29%**	60,00	14,29	11,43	14,29
E	**97,14%**	74,29	22,86	2,86	0,00
G	**100%**	94,29	5,71	0,00	0,00
H	**91,43%**	88,57	2,86	0,00	8,57
H'	**88,57%**	82,86	14,29	2,86	0,00
I	**97,14%**	82,86	14,29	2,86	0,00
K	**82,86%**	45,71	37,14	14,29	2,86
L	**88,57%**	88,57	0,00	0,00	14,43
L'	**85,71%**	85,71	0,00	0,00	14,29

Tabla 3.2-Resultados de los 10 puntos superiores de la cara

Puntos	✓	3	2	1	0
M	**97,14%**	62,86	34,29	2,86	0,00
N	**94,29%**	62,86	31,43	5,71	0,00
O	**80,00%**	48,57	31,43	20,00	0,00
P	**97,14%**	57,14	40,00	2,86	0,00
Q	**91,43%**	68,57	22,86	5,71	2,86
Q'	**94,29%**	82,86	11,43	2,86	2,86
R	**97,14%**	71,43	25,71	2,86	0,00
S	**71,43%**	42,86	28,57	25,71	2,86
T	**82,86%**	65,71	17,14	8,57	8,57
U	**68,57%**	48,57	20,00	20,00	11,43
U'	**71,43%**	54,29	17,14	14,29	14,29
V	**71,43**	57,14	14,29	5,71	22,86

Tabla 3.3- Resultados de los 10 puntos inferiores de la cara

Neutro	97,50%	Dientes	90,00%
Boca cerrada	79,55%	Boca abierta	83,00%

Tabla 3.4- Porcentaje de puntos detectados en función de la forma de la boca.

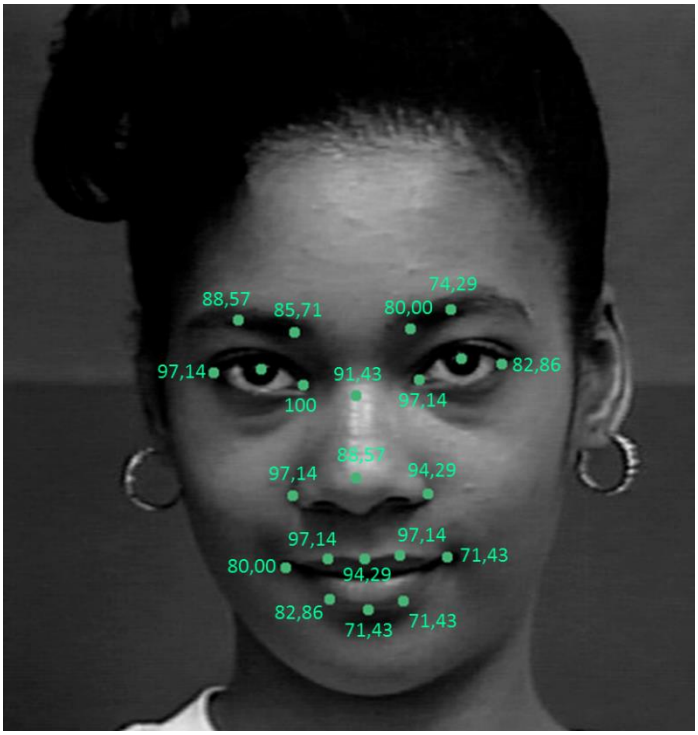


Figura 3.9- Porcentaje de acierto en la detección de cada uno de los 20 puntos detectados por *HOG*.

Los resultados arrojados por los clasificadores pueden verse en la Tabla 3.2 y 3.3. Como se puede comprobar, los puntos que se detectan invirtiendo la imagen no tienen peores resultados que los detectados con normalidad. De esta manera, la hipótesis de la simetría facial se da por válida.

En los puntos centrales de la cara se comprueba cómo funcionaría la detección si la imagen estuviese invertida. Al ser estos simétricos, idealmente no debería apreciarse diferencia entre ambos detectores, sin embargo, como la simetría no es exacta se comprueba que los puntos *U* y *Q* de la boca funcionan mejor de esta manera.

En la Tabla 3.4 se recogen de manera simplificada los distintos tipos de imágenes utilizadas en los *tests*. Como puede comprobarse, la detección es muy buena en imágenes con una

emoción neutra o en aquellas que se muestran los dientes. Los posicionamientos empeoran en aquellas en las que la boca aparece abierta y obtienen sus peores resultados en aquellas muecas se realizan con la boca cerrada.

Como se aprecia, los peores resultados se centran en la boca, en concreto en su parte baja. Esto se debe a que se trata de los puntos que presentan mayor variedad, como puede verse en las representaciones de los vectores de *HOG* en la Figura 3.10. Otro ejemplo de este problema se da cuando aparecen muecas con los labios cerrados. Este tipo de expresiones añaden un grado de dificultad pues, en ocasiones, los pliegues en el mentón se confunden con los labios que además suelen desdibujarse con estos gestos, Figura 3.11a,b. Los resultados contrastan con la eficacia en la detección de gestos en los que se muestran los dientes, Figura 3.11c. A pesar de lo comentado, se trata de unos resultados aceptables y que mejoran para el resto de puntos de la cara obteniendo buenos resultados en todas las expresiones, Figura 3.12.

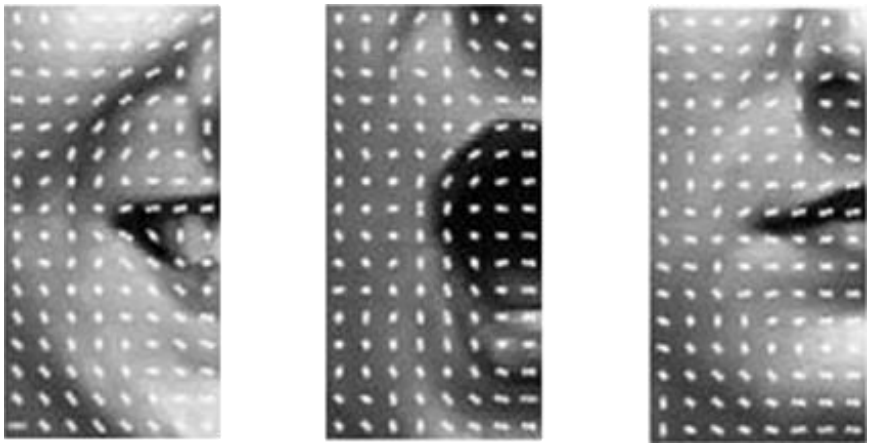


Figura 3.10- Representación de la variedad de los descriptores *HOG* para el punto O de la boca.

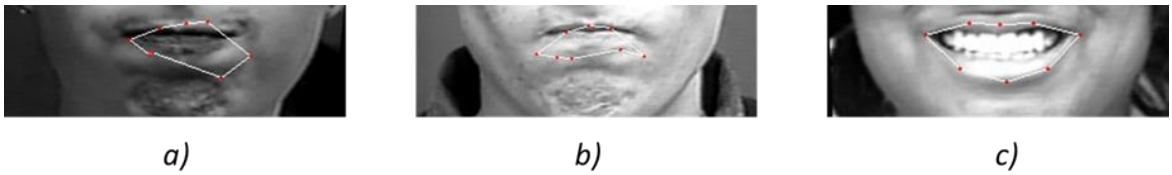


Figura 3.11- Ejemplos de detección de puntos en la boca.

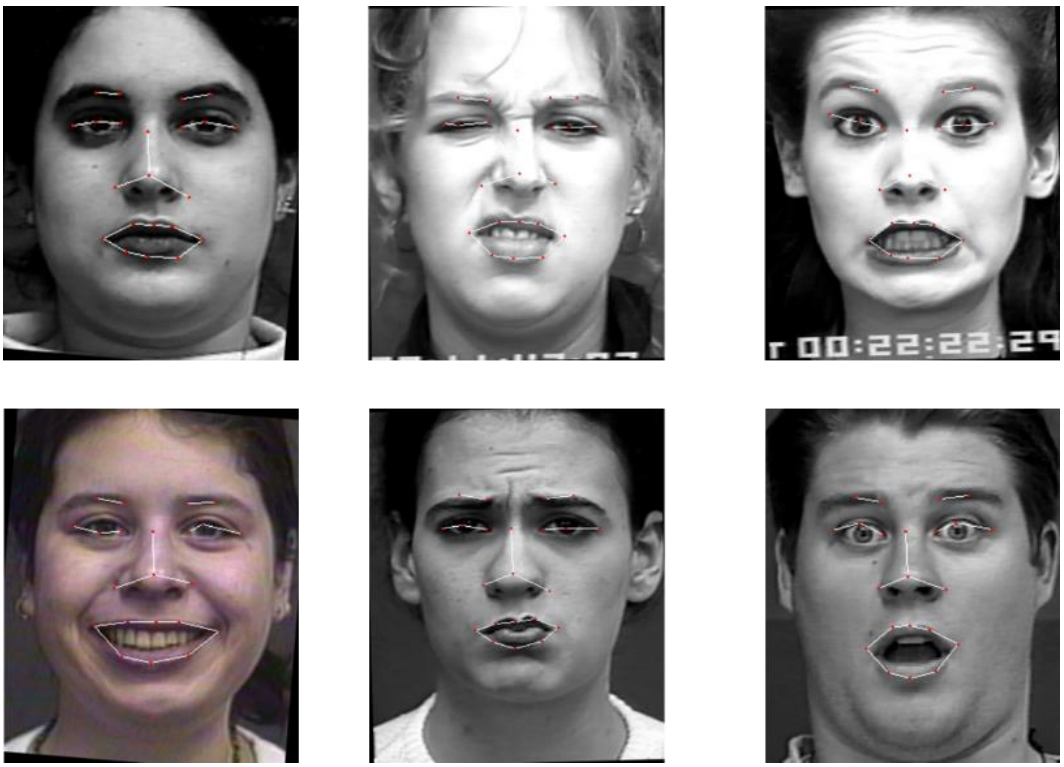


Figura 3.12- Ejemplo de una correcta detección de los 22 puntos en imágenes con expresiones variadas.

A pesar de que la detección permite continuar con el proyecto, para minimizar la posibilidad de que se cometan grandes errores en la detección de un punto, se realizan dos posibles correcciones a las coordenadas de los puntos. Para evitar grandes errores solo se guardan aquellas coordenadas cuya posición sea coherente, Tabla 3.5. De esta manera, no se aceptan coordenadas del lado derecho de la cara que aparezcan en la mitad izquierda o puntos de la parte superior que se ubiquen en la inferior. Estas limitaciones se han determinado de manera experimental sobre las imágenes de validación y su funcionamiento general se ha comprobado a la hora de extraer los resultados.

	$X_{mín}$	$X_{máx}$	$Y_{mín}$	$Y_{máx}$
A, D, E, K	0	0,4	0	0,4
B, C	0	0,5	0	0,5
G, I	0	0,6	0	0,6
H	0	0,6	0	1
L	0,6	0,2	0,2	1
M, N	0,3	0,6	0,4	1
O, S	0,2	0,5	0,5	1
R	0,3	0,5	0,5	1
Q	0,4	0,55	0,5	0,7
P, T, V	0,3	0,5	0,6	1
U	0,4	0,55	0,6	1

Tabla 3.5- Limitaciones a las coordenadas de los puntos expresadas de manera porcentual.

Por motivo de esta corrección o porque el detector no encuentra un candidato, se plantea un segundo método de mejora. A partir de 10 imágenes se extrae la posición media de los 22 puntos característicos (en realidad 12, ya que de nuevo se aplica simetría) en un sujeto con expresión facial neutra. Las coordenadas se expresan en función del ancho y el alto de la imagen para atenuar el efecto de la diferencia de tamaños. Aunque pudiera parecer que 10 muestras no son suficientes, la desviación típica de las mediciones realizadas hacía indicar que aumentar el número de imágenes no mejoraría en exceso los resultados. En la Figura 3.13 se puede apreciar la disposición de esos puntos en un rostro neutro.

Figura 3.13- Representación de los puntos de posición estándar en dos sujetos.

Sin embargo, en el caso de la imagen cuya expresión se desconoce, cuando un punto no tiene coordenadas a este se le asocian las de ese mismo punto en la imagen neutra. Esto se realiza porque se considera preferible omitir un movimiento que tener en cuenta una variación en la posición que no se da en las imágenes. En el caso de que se llevase a cabo una implementación en la que, en lugar de dos imágenes, se analizará la secuencia completa la posición sustituta pasaría a ser la del *frame* inmediatamente anterior.

Por último, es conveniente obtener, al menos, una estimación del funcionamiento conjunto de la malla de puntos. Para llevar a cabo el análisis se hace uso de las imágenes de la base de datos cuya expresión facial se encuentra etiquetada. Como para algunas expresiones el número de imágenes no es tan elevado como se desearía es conveniente aumentar el número de muestras. Para ello se incorporan a las imágenes etiquetadas por la base de datos imágenes etiquetadas manualmente. Aunque solo algunas secuencias están acompañadas de la emoción que representan, todas ellas tienen asociado un archivo que indica qué unidades de acción aparecen en ellas. De esta forma, gracias a las equivalencias propuestas en el artículo incluido en la propia base de datos[29], que traducen la presencia o ausencia de *AUs* en expresiones faciales, se puede aumentar el número de muestras.

Los resultados por tanto para emoción serán los que se reflejan en la Tabla 3.6.

Emoción	Nº de muestras	% de acierto	Emoción	Nº de muestras	% de acierto
Neutral	104	88,46%	**Miedo**	27	88,88%
Enfado	44	43,18%	**Felicidad**	80	73,75%
Desprecio	16	56,25%	**Tristeza**	33	75,75%
Asco	58	58,62%	**Sorpresa**	73	38,35%

Tabla 3.6- Estimación de la precisión de la malla en las distintas emociones

Debe tenerse en cuenta, que cada porcentaje representa que la malla de puntos se adapta bien en las dos imágenes, la inicial neutra y la segunda en la que aparece una emoción. Por tanto, aunque en las imágenes sin expresión facial la detección es buena, algunas secuencias no se consideran como acertadas por errores en la primera imagen.

La bondad de una detección se determina de manera cualitativa llegándose a aceptar en ocasiones secuencias como la que se ve en la Figura 3.14. Solo 2 de los 22 puntos se detectan mal en la segunda imagen y lo hacen de manera que no deforman totalmente la expresión ni incurren en errores ilógicos (por ejemplo, situar las cejas por debajo de los ojos). Cuando los errores afectan a más puntos o puede intuirse que la expresión queda sustancialmente alterada es cuando se considera errónea.

Figura 3.14- Ejemplo de secuencia con errores menores que se considera correcta.

El efecto de las imágenes neutras y el carácter altamente subjetivo de la evaluación hacen que los resultados solo puedan tomarse como una estimación del efecto de cada expresión al estudio. Para obtener una idea de la efectividad resulta más conveniente consultar las Tablas 3.2 y 3.3.

Las expresiones de sorpresa son las que peor se detectan debido a que, como se ve en la Figura 3.10 los descriptores de *HOG* son los más diferentes. La peor detección de enfado, asco y desprecio se debe principalmente a errores como los de la Figura 3.11. El posicionamiento en el resto de expresiones puede considerarse más que aceptable dado al alto número de puntos detectados.

3.3 CÁLCULO DE LA EMOCIÓN

Para evaluar la eficiencia de la clasificación en función de la expresión se utilizarán solo imágenes correctamente detectadas. Con el fin de intentar conseguir el mejor entrenamiento posible se realiza una nueva evaluación de las secuencias. En esta segunda pasada se hace un juicio más exigente, sobre todo en el caso en el que el número de muestras es abundante. La cantidad de muestras disponibles para entrenar se puede ver en la Tabla 3.7.

Neutral	61	Miedo	27
Enfado	14	Felicidad	59
Desprecio	8	Tristeza	29
Asco	31	Sorpresa	28

Tabla 3.7- Numero de muestras disponibles por emoción para el entrenamiento.

Con apenas unas pruebas rápidas se puede comprobar que el número de muestras para el enfado y el desprecio será demasiado bajo. En el primer caso se debe principalmente a los problemas en la detección de puntos que se expusieron en la Figura 3.11. En el caso de las expresiones relacionadas con el desprecio el principal problema es el bajo número de muestras en la propia base. Por tanto, se tratará de construir un sistema que clasifique entre las siguientes emociones: neutral, asco, miedo, felicidad, tristeza y sorpresa.

Para no favorecer a ninguna expresión, todas contarán con el mismo número de muestras positivas, 27. Como ya se explicó en el marco teórico, no será necesario introducir muestras negativas en el sistema *One vs. One* porque las positivas de otras clases actúan como tal. Con vectores de un número tan elevado de dimensiones es imposible predecir qué kernel es el más apropiado por lo que se escogerá mediante métodos empíricos. De las 27 muestras por expresión, 15 se dedicarán al entrenamiento de los clasificadores, 5 para validar el método y determinar con qué *kernel* funciona mejor, y, finalmente, 7 para los resultados finales.

Kernel	Emoción	% aciertos	*Kernel*	Emoción	% aciertos
Lineal	Neutral	80	**Sigmoide**	Neutral	0
	Asco	100		Asco	0
	Miedo	0		Miedo	0
	Felicidad	40		Felicidad	0
	Tristeza	60		Tristeza	0
	Sorpresa	80		Sorpresa	0
Función de base radial	Neutral	80	**Exponencial chi cuadrado**	Neutral	20
	Asco	100		Asco	0
	Miedo	0		Miedo	0
	Felicidad	40		Felicidad	0
	Tristeza	60		Tristeza	20
	Sorpresa	100		Sorpresa	0

Tabla 3.8- Validación de los *kernels*.

Kernel	Emoción	% aciertos
Intersección de histogramas	Neutral	80
	Asco	100
	Miedo	20
	Felicidad	60
	Tristeza	60
	Sorpresa	80

Tabla 3.9- Resultados de la validación de la intersección de histogramas

En las Tabla 3.8 pueden verse los *kernels* cotejados y los porcentajes de acierto proporcionados. Se comprueba que algunos de los kernels ofrecen un comportamiento

inaceptable, destacando el basado en la función sigmoide, incapaz de predecir correctamente ninguna expresión. En la Tabla 3.9 aparece la intersección de histogramas kernel que ofrece los mejores resultados siendo, además, el único de los evaluados capaz de obtener aciertos en todas las emociones.

Una vez escogido el *kernel* se pueden estimar, con las muestras restantes, la eficiencia de los clasificadores. Los resultados del test se pueden ver en la Tabla 3.10 que representa una matriz de confusión. En ella puede comprobarse la probabilidad de que se calcule cada una de las clases para una muestra de una emoción dada. De esta manera en la diagonal aparecerá la eficacia de cada clasificador.

Emoción calculada

Emoción real	Neutral	Asco	Miedo	Felicidad	Tristeza	Sorpresa
Neutral	**100**	0,00	0,00	14,29	42,86	0,00
Asco	0,00	**100**	14,29	14,29	0,00	0,00
Miedo	0,00	0,00	**28,57**	14,29	0,00	0,00
Felicidad	0,00	0,00	14,29	**57,14**	0,00	0,00
Tristeza	0,00	0,00	42,86	0,00	**57,14**	0,00
Sorpresa	0,00	0,00	0,00	0,00	0,00	**100**

Tabla 3.10- Matriz de confusión con las expresiones estudiadas.

Las emociones neutrales y de felicidad son las únicas que permiten hacer un estudio más detallado. En la siguiente tabla se aprecian los resultados para 25 nuevas muestras.

Neutral	92%	Felicidad	52%

Tabla 3.11- Resultados de test con más muestras para la expresión neutral y la de felicidad.

En la Tablas 3.9, 3.10 y 3.11 puede apreciarse como los resultados se mantienen más o menos estables a pesar de variar las muestras estudiadas. Los resultados son muy buenos para la expresión neutral, de asco y de sorpresa. La felicidad y la tristeza obtienen unos resultados moderados mientras que las expresiones de miedo se detectan de manera pobre.

Debe tenerse en cuenta en estos resultados analizan el funcionamiento de los clasificadores sobre muestras en las que la malla de puntos de posiciona correctamente. Para hacerse una idea del funcionamiento conjunto del proyecto sobre la base de datos pueden combinarse los valores de la Tabla 3.10 y la Tabla 3.6.

En la matriz de confusión, Tabla 3.10, destacan claramente dos errores. Por un lado, las expresiones de miedo tienden a clasificarse más como tristeza. También es remarcable el alto número de ocasiones en la que sujetos que muestran tristeza son clasificados como sujetos con expresión neutra. En ambos casos, puede deberse a que, en ocasiones, la posición de los puntos en esas emociones es muy similar, Figura 3.15. Esta situación es más evidente entre el miedo y la tristeza debido a que incluso comparten la presencia de la *AU1 (Inner Brow Raiser)* y la *AU4 (Brow Lowerer)*. Una detección más precisa de los puntos y un mayor número de muestras que permita definir mejor los vectores de soporte podría ser la solución a este problema.

Figura 3.15-A la izquierda, similitud entre expresión neutral y de tristeza. A la derecha, similitud entre expresión de miedo y, de nuevo, tristeza.

3.4 TIEMPO DE EJECUCIÓN

Aunque el objetivo de este trabajo es el de llevar a cabo una investigación, y no el desarrollo de un producto o aplicación funcional, no está de más analizar, aunque sea superficialmente, el tiempo de ejecución del código implementado. Para estudiar esto utilizaremos el código para 8 imágenes de sujetos y expresiones diferentes. De esta forma podrá verse si existe algún hecho que afecte significativamente al rendimiento. Se dividen las mediciones en:

detección facial, detección de pupilas y rotación, detección del resto de puntos y, finalmente, cálculo de la emoción.

t_{total} =	t_{cara}	$+t_{ojos}$	$+t_{puntos}$	$+t_{emoción}$
1,1951±0,18 s	0,2466±0,017	0,1088±0,0207	0,8245±0,1622	0,01966±0,00184

Tabla 3.12- Media de tiempo empleado (en segundos) en ejecutar el código

Los resultados utilizando un Lenovo B50-80 con 11,6GB de memoria RAM y procesador de 4 núcleos (2,2GHz) pueden verse en la Tabla 3.12. El funcionamiento no resulta muy alejado de una predicción por segundo. No aparecen grandes variaciones entre las distintas imágenes, sean o no sean clasificadas correctamente. Sin duda, la mayor parte del tiempo del proceso se emplea en la detección de los 20 puntos. A pesar de ello, si se tiene en cuenta el número de puntos, puede determinarse que los detectores en base a los histogramas de gradientes orientados son más veloces que los que utilizan filtros de Haar. La diferencia entre la duración de la detección de caras y de ojos se debe, probablemente, a que en la primera se inspecciona una imagen mucho mayor.

4. Discusión

Si se buscase mejorar los resultados se deben acometer los problemas en el número de muestras y en la precisión a la hora de detectar los puntos característicos. A pesar de que la base de datos Cohn-Kanade parece la mejor entre las disponibles, podría considerarse, llegado el caso, la combinación con otras bases de datos. Debido a que el sistema puede funcionar con varios tamaños de imagen esto no debería suponer ningún problema, aunque queda por probar su eficacia con imágenes de resolución notablemente superior. Aumentando el número de muestras podría mejorarse el entrenamiento de las máquinas de soporte vectorial. La consecuente mejora en la detección de puntos supondría también la posibilidad de utilizar más muestras que actualmente se analizan erróneamente por lo que la mejora sería doble.

Aunque la detección de la pupila obtiene buenos resultados podría mejorarse para que detecte el punto central del ojo con mayor precisión. Para abordar este proceso podría tomarse la detección actual como una primera aproximación y estudiar los niveles de gris para determinar el punto central.

El uso de detectores en base *HOG* es satisfactorio y mejorados hasta el nivel de precisión de los clasificadores en cascada podrían sustituirse para la detección de cara y ojos ya que es más veloz. Solo en los puntos inferiores de la boca parece necesaria la búsqueda de una alternativa. El problema de la variedad podría superarse con el entrenamiento de varios clasificadores para los puntos de la boca. De esta manera existirían varias opciones correspondientes a varias muecas y podría así seleccionarse aquella que se detectase mejor.

La precisión también podría mejorarse basando las restricciones de la Tabla 3.5 en un estudio estadístico más profundo que permita limitaciones más estrictas y precisas.

El resultado del trabajo puede considerarse satisfactorio logrando clasificar imágenes en función de la expresión facial del individuo a través del desplazamiento de puntos característicos de la cara. Como se ha mostrado, los resultados y la velocidad de procesamiento pueden considerarse aceptables para las ambiciones de este proyecto, aunque su idoneidad dependerá totalmente de la aplicación en la que pretende integrarse.

El desarrollo futuro del trabajo dependería de la orientación que se le quisiera dar. Si se desea continuar el proceso de investigación se deberían probar distintos métodos y bases de datos para cotejar los resultados obtenidos. Los clasificadores desarrollados en este proyecto no están diseñados para el funcionamiento en niños, personas de avanzada edad o, simplemente, personas con notable vello facial. Intentar abarcar a la mayor cantidad de gente posible podría ser una posible vía de avance.

En el que caso de que se buscase una aplicación directamente práctica se debería lograr un aumento en la precisión y en la velocidad de ejecución, centrándose en uno de estos dos aspectos en función del fin ideado. Llegado el punto de llevarse a cabo una implementación práctica sería conveniente estudiar la posibilidad de permitir que el sistema se re-entrene periódicamente. Durante su propio funcionamiento, podría tomar aquellas predicciones correctas como muestras positivas que permitan al algoritmo no dejar nunca de “aprender”.

5. Conclusiones

A pesar de la complejidad que supone una clasificación de expresiones faciales se puede concluir que se ha logrado cumplir con el objetivo planteado al comienzo del proyecto. Recuperando el diagrama resumen del proyecto podemos ver que se ha conseguido un resultado aceptable en los tres módulos, Figura 5.1.

Figura 5.1- Diagrama de bloques del proyecto.

El primer bloque ofrece unos resultados excelentes gracias al método Viola-Jones, con un 100% de aciertos detectando la cara dentro de la imagen. En la segunda etapa, la eficacia de los clasificadores en cascada permite también detectar con precisión los ojos en la imagen. Se consigue entrenar máquinas de soporte vectorial que detecten los puntos deseados gracias a los datos aportados por los histogramas de gradientes orientados. De nuevo son las máquinas de soporte vectorial las que, en la tercera etapa, dirimen qué emoción presenta el sujeto. A pesar de que el estudio no ha permitido cubrir las 8 emociones existentes en la base de datos, los resultados son excelentes para tres (emoción neutra, asco y sorpresa) de las 6 emociones analizadas. Solo las expresiones relacionadas con el miedo obtienen resultados claramente negativos.

Dejando a un lado aspectos relacionados con la eficiencia, dado que este proyecto se enfoca como un proyecto de investigación, el estudio realizado ha permitido obtener, desde cero, una visión general del estado del arte de la clasificación de expresiones mediante visión artificial y sus retos futuros.

En un plano más personal, el desarrollo de este proyecto ha potenciado mi capacidad para el aprendizaje autónomo. A pesar de que las nociones generales de visión artificial sí lo hacen, las técnicas de *machine learning* no entran dentro de los contenidos aprendidos en el grado.

Además, entre las competencias adquiridas, se encuentra la introducción a la programación en C++ y la utilización de la librería OpenCV. El trabajo también me ha introducido en el uso de un sistema operativo Linux, sistema en el que se ha llevado a cabo la elaboración de la totalidad del código.

6. Referencias

[1] P. Ekman y W. V. Friesen, "Manual for the Facial Action Coding System", presentado en Consulting Psychologists Press, 1977.
[2] P. Ekman, W. V. Friesen, y J. C. Hager, "Facial Action Coding System Investigator's Guide", presentado en A Human Face, Salt Lake City, 2002.
[3] MPEG Video and SNHC, "Text of ISO/IEC FDIS 14 496-3: Audio", Atlantic City MPEG Mtg, 1998.
[4] A. Raouzaiou, N. Tsapatsoulis, K. Karpouzis, y S. Kollias, "Parameterized facial expression synthesis based on MPEG-4", *EURASIP J. Adv. Signal Process.*, vol. 2002, nº 10, p. 521048, 2002.
[5] L. Malatesta, A. Raouzaiou, K. Karpouzis, y S. Kollias, "Towards modelling embodied conversational agent character profiles using appraisal theory predictions in expression synthesis", *Appl Intell This Issue*, 2007.
[6] P. Viola y M. J. Jones, "Robust real-time face detection", *Int. J. Comput. Vis.*, vol. 57, nº 2, pp. 137–154, 2004.
[7] B. D. Lucas y T. Kanade, "An Iterative Image Registration Technique with an Application to Stereo Vision", 1981, pp. 121–130.
[8] C. Tomasi y T. Kanade, "Detection and tracking of point features", 1991.
[9] H. Schneiderman y T. Kanade, "Probabilistic modeling of local appearance and spatial relationships for object recognition", en *Computer Vision and Pattern Recognition, 1998. Proceedings. 1998 IEEE Computer Society Conference on*, 1998, pp. 45–51.
[10] N. Esau, E. Wetzel, L. Kleinjohann, y B. Kleinjohann, "Real-time facial expression recognition using a fuzzy emotion model", en *2007 IEEE International Fuzzy Systems Conference*, 2007, pp. 1–6.
[11] G. R. Bradski, "Computer Vision Face Tracking For Use in a Perceptural User Interface", presentado en Intel Technology Journal Q2(Q2):214-219, 1998.
[12] "OpenCV: Meanshift and Camshift". [En línea]. Disponible en: http://docs.opencv.org/3.1.0/db/df8/tutorial_py_meanshift.html. [Accedido: 10-dic-2016].
[13] P. Michel y R. El Kaliouby, "Real time facial expression recognition in video using support vector machines", en *Proceedings of the 5th international conference on Multimodal interfaces*, 2003, pp. 258–264.
[14] R. K. McConnell, "Method of and apparatus for pattern recognition", ene. 1986.
[15] N. Dalal y B. Triggs, "Histograms of oriented gradients for human detection", en *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, 2005, vol. 1, pp. 886–893.
[16] [En línea]. Disponible en: http://dlib.net/face_landmark_detection.py.html. [Accedido: 16-jun-2017].
[17] "dlib C++ Library". [En línea]. Disponible en: http://dlib.net/. [Accedido: 16-jun-2017].
[18] D. Vukadinovic y M. Pantic, "Fully automatic facial feature point detection using Gabor feature based boosted classifiers", en *2005 IEEE International Conference on Systems, Man and Cybernetics*, 2005, vol. 2, pp. 1692–1698.

[19] M. Valstar y M. Pantic, "Fully automatic facial action unit detection and temporal analysis", en *2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'06)*, 2006, pp. 149–149.
[20] A. Aghagolzadeh, H. Seyedarabi, y S. Khanmohammadi, "Single and Composite Action Units Classification in Facial Expressions by Feature-Points Tracking and RBF Neural Networks", en *Ukrainian Int. conf. signal/Image processing, UkrObraz 2004*, 2004, pp. 181–184.
[21] E. Cerezo *et al.*, "Real-time facial expression recognition for natural interaction", en *Iberian Conference on Pattern Recognition and Image Analysis*, 2007, pp. 40–47.
[22] M. F. Valstar, *Timing is everything: A spatio-temporal approach to the analysis of facial actions*. Imperial College London, 2008.
[23] M. Rydfalk, "CANDIDE, a parameterized face ", Dept. of Electrical Engineering ,Linköping University, Sweden, LiTH-ISY-I-866, 1987.
[24] I. Kotsia y I. Pitas, "Real time facial expression recognition from image sequences using support vector machines", en *Visual Communications and Image Processing 2005*, 2005, p. 59602E–59602E.
[25] A. C. Bovik, C. W. Chen, D. B. Goldgof, y T. S. Huang, Eds., *Advances in image processing and understanding: a festschrift for Thomas S. Huang*. Singapore ; River Edge, N.J: World Scientific, 2002.
[26] T. Cootes, "Model-based methods in analysis of biomedical images", *Image Process. Anal.*, pp. 223–248, 2000.
[27] P. Ekman y W. V. Friesen, "Constants across cultures in the dace and emotion", vol. 17.
[28] T. Kanade, J. F. Cohn, y Y. Tian, "Comprehensive database for facial expression analysis", en *Automatic Face and Gesture Recognition, 2000. Proceedings. Fourth IEEE International Conference on*, 2000, pp. 46–53.
[29] P. Lucey, J. F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, y I. Matthews, "The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression", en *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, 2010, pp. 94–101.
[30] V. Bettadapura, "Face expression recognition and analysis: the state of the art", *ArXiv Prepr. ArXiv12036722*, 2012.
[31] C. P. Papageorgiou, M. Oren, y T. Poggio, "A general framework for object detection", 1998, pp. 555–562.
[32] A. Kaehler y G. Bradski, *Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library*. O'Reilly Media, Inc., 2016.
[33] Y. Freund y R. E. Schapire, "A desicion-theoretic generalization of on-line learning and an application to boosting", en *European conference on computational learning theory*, 1995, pp. 23–37.
[34] A. Kaehler y G. Bradski, "Supervised Learning and Boosting Theory", en *Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library*, O'Reilly Media, Inc., 2016, pp. 879–888.
[35] A. Harvey, "OpenCV Face Detection: Visualized", *Vimeo*. [En línea]. Disponible en: https://vimeo.com/12774628. [Accedido: 01-jun-2017].
[36] M. Sonka, V. Hlavac, y R. Boyle, "Geometric transformatios", en *Image Processing, Analysis, and Machine Vision*, 4th ed., Cengage Learning, 2015, pp. 120–124.

[37] E. J. C. Suárez, "Tutorial sobre máquinas de vectores soporte (sVM)", *Tutor. Sobre Máquinas Vectores Soporte SVM*, 2014.
[38] "OpenCV | OpenCV". [En línea]. Disponible en: http://opencv.org/. [Accedido: 11-dic-2016].
[39] Jan, *trainHOG: Example program showing how to train your custom HOG detector using openCV*. 2017.
[40] "SVM-Light Support Vector Machine". [En línea]. Disponible en: http://svmlight.joachims.org/. [Accedido: 17-jun-2017].
[41] Marcel M., *Of Monsters And Men - Dirty Paws Instrumental Cover*. 2014.
[42] M. Sonka, V. Hlavac, y R. Boyle, "Edge Detectors", en *Image Processing, Analysis and Machine Vision*, 2015, pp. 133–147.

ANEXO I:

MATERIAL COMPLEMENTARIO

Para complementar el Trabajo Fin de Grado se adjuntan junto a la memoria una serie de archivos que dan una idea más cercana del trabajo desarrollado. El material adicional se organiza de la siguiente manera:

- Código
  - Principal
  - Recortes
  - SVM
- Imágenes
  - 0-Neutral
  - 3-Asco
  - 4-Miedo
  - 5-Felicidad
  - 6-Tristeza
  - 7-Sorpresa
- Vídeo

En la carpeta *Principal* se encuentra el código que implementa el método al completo preparado para ser utilizado con la base de datos Cohn-Kanade. Este aparece tanto en los formatos originales como en una versión en *pdf* que puede facilitar la lectura. También se incluyen en una subcarpeta los clasificadores empleados. En las carpetas *Recortes* y *SVM* se encuentran otras herramientas empleadas durante el desarrollo del proyecto. La primera alberga el programa empleado para recortar las imágenes de 64x128 utilizadas para entrenar los detectores de puntos. En *SVM* puede encontrarse la programación del entrenamiento, testeo y validación de las Máquinas de Soporte Vectorial empleadas para el cálculo de la emoción.

La carpeta *Imágenes* pretende ser una muestra de las imágenes utilizadas para entrenar la clasificación de cada una de las emociones abarcadas. Como ya se ha comentado, debido a cuestiones legales solo es posible difundir imágenes de 11 sujetos[1] de la base de datos. Para paliar esta limitación se incluye por cada expresión facial: un GIF animado que muestra la

[1] S52, S55, S74, S106, S111, S113, S121, S124, S125, S130 y S132

detección de puntos a lo largo de una secuencia completa y las subcarpetas *Distancias* y *Sujetos*. En la primera de ellas se encuentran imágenes que representan el desplazamiento de los puntos característicos, apareciendo en rojo los de la emoción a detectar y en azul los de la emoción neutra. Por otra parte, en *Sujetos* se guardan las imágenes libres de derechos junto al vector, utilizado para el entrenamiento, que las representa

Por último, en la carpeta *Vídeo* puede encontrarse un vídeo que resume el método implementado en 5 minutos. La música de fondo ha sido extraída del siguiente enlace[41].

ANEXO II:

PRESUPUESTO

1. Costes de ejecución material

El coste de ejecución material incluye tres categorías, *Coste de equipos*, *Coste de software* y *Coste de mano de obra*. Para los equipos y para el software se considera una amortización de 3 años. La duración del proyecto se estima en 59 días y puede verse desglosada en el *Coste de mano de obra*.

1.1 Costes de equipos

CONCEPTO	*PRECIO UNITARIO*	*USO*	*SUBTOTAL*
Ordenador portátil Lenovo B50-80	527 €	59 días	28,48€

Subtotal: ····· **28,48 €**

1.2 Costes de software

CONCEPTO	PRECIO UNITARIO	USO	SUBTOTAL
Sistema operativo Linux: Ubuntu 16.04	0 €	44 días	0 €
Librería OpenCV 3.2	0 €	44 días	0 €
Base de datos Cohn-Kanade	0 €	44 días	0 €
Microsoft Office 365	10,70 €/mes	17 días	6,06 €

Subtotal: ····· **6,06 €**

1.3 Costes de mano de obra

El coste unitario de la mano de obra será de 30 €/h. La dedicación será de 8 horas al día, 5 días a la semana.

CONCEPTO	TIEMPO EMPLEADO	SUBTOTAL
Estudio del problema	10 días	2400 €
Estudio OpenCV	5 días	1200 €
Desarrollo algoritmo detección facial y ocular	5 día	1200 €
Toma de muestras para descriptores HOG	5 días	1200 €
Entrenamiento descriptores HOG	1 día	240 €
Toma de muestras para el cálculo de la emoción	5 días	1200 €
Entrenamiento SVMs para el cálculo de la emoción	1 día	240 €
Desarrollo del código	10 días	2400 €
Pruebas de funcionamiento y extracción de resultados	2 día	1200 €
Realización de la documentación	15 días	3600 €

Subtotal: ..... **14880,00 €**

1.4 Coste total del presupuesto de ejecución material

CONCEPTO	SUBTOTAL
Coste de equipos	28,48 €
Coste de software	6,06 €
Coste de mano de obra	14880,00 €

Subtotal: ..... **14914,54 €**

2. Gastos generales y beneficio industrial

Los gastos generales y beneficio industrial son los gastos obligados que se derivan de la utilización de las instalaciones de trabajo más el beneficio industrial. Se estima un porcentaje del 10 % para los gastos generales y un 6% de beneficio.

CONCEPTO	SUBTOTAL
Gastos generales y beneficio industrial	2386,33 €

3. Importe total

CONCEPTO	SUBTOTAL
Coste total del presupuesto de ejecución material	14914,54 €
Gastos generales y beneficio industrial	2386,33 €

TOTAL: ..... **17300,87€**

IVA 21%: ..... **3633,18 €**

TOTAL, IVA INCLUIDO: ..... **20934,05€**

El Importe Total del proyecto suma la cantidad de:

Veinte Mil Novecientos Treinta y Cuatro, con Cinco Céntimos