UNIVERSIDAD NACIONAL DE INGENIERÍA

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

“UN ALGORITMO DE DETECCIÓN Y VALIDACIÓN DE ROSTROS EN ENTORNOS NO CONTROLADOS ORIENTADO AL MONITOREO BIOMÉTRICO DE CONDUCTORES VEHICULARES”

TESIS

PARA OBTENER EL GRADO ACADÉMICO DE MAESTRO EN CIENCIAS EN

INGENIERÍA ELECTRÓNICA CON MENCIÓN EN PROCESAMIENTO DE SEÑALES E IMÁGENES

ELABORADO POR

ALEJANDRO MOHAMARK RAMIREZ PEREZ GAO

ASESOR

Dr. GUILLERMO LEOPOLDO KEMPER VÁSQUEZ

LIMA – PERÚ

2019

DEDICATORIA

A dios, a mis padres don Javier Ramirez y Doña Isabel Perez Gao, a mi hermana Indhira Ramirez; por todo el apoyo que brindaron en esta etapa de mi vida, por los valores que me inculcaron y la admiración que les tengo.

AGRADECIMIENTOS

El presente trabajo de Tesis es el resultado de un conjunto de esfuerzos y sueños. En las siguientes líneas intentaré mencionar a las instituciones y personas que apoyaron de manera sustancial a que esto salga adelante.

A mi familia, en especial a mis padres y mi hermana, quienes con su compañía, consejos, regaños y risas me brindaron las fuerzas y motivación necesarias para poder salir de las complicaciones que se presentaban. A Alejandra Salazar, quien a lo largo de mi investigación estuvo ahí para ánimo y apoyo. Estas personas me enseñaron el valor de la palabra sacrificio, y les estaré eternamente agradecido.

A mi asesor de Tesis, el Dr. Guillermo Leopoldo Kemper Vásquez, por todo el apoyo recibido dentro y fuera de las aulas, tanto científico como personal. Sin su orientación este trabajo no hubiera sido posible. A los docentes especialistas Dr. Avid Román Gonzáles y M.Sc. Christian Carlos Del Carpio Damián, por los consejos brindados para terminar de pulir mi trabajo. También a la plana docente y administrativa de la escuela de Posgrado de la Facultad de Ingeniería Eléctrica y Electrónica, en especial al M.Sc. Segundo Gamarra y al M.Sc. Domingo Lazo.

Al Ingeniero Daniel Rodriguez, por su apoyo tanto en la planificación como en la ejecución del proyecto. Sobre todo, por su apoyo en los momentos más críticos de la investigación.

Al Instituto Nacional de Investigación y Capacitación de Telecomunicaciones del Perú (INICTEL – UNI), por brindarme acceso a sus instalaciones, dejarme usar sus equipos y dejar que conviva por más de 2 años con un excelente grupo humano.

Al convenio del Ministerio de Educación (MINEDU) y el Consejo Nacional de Ciencia, Tecnología e Innovación Tecnológica (CONCYTEC), por el apoyo económico brindado, el cual permitió que pueda tener una dedicación exclusiva hacia mis estudios.

ÍNDICE DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE ILUSTRACIONES

RESUMEN

En el presente trabajo de investigación se busca dar solución a un problema de biometría a través de las áreas de procesamiento digital de imágenes y visión computacional con la finalidad de realizar el monitoreo de conductores de transporte público de personas mediante el reconocimiento de rostros. El objetivo es el de proponer un algoritmo de detección e identificación de rostros que sea robusto ante las diversas condiciones de iluminación a las que los conductores puedan estar sometidos en entornos no controlados. Se utilizó el algoritmo de especificación rápida local de histograma para atenuar el ruido debido a la iluminación y se estudió cómo este algoritmo repercute en el rendimiento de la detección e identificación de rostros, las cuales se basan en los descriptores como los histogramas de gradientes orientados y/o el resultado de una red neuronal convolucional con arquitectura *FaceNet* respectivamente. Finalmente, se utilizaron máquinas de soporte vectorial para realizar la clasificación final en la detección e identificación de rostros, obteniéndose una sensibilidad del 71.98% y una especificidad del 97.65% en la detección de rostros y una asertividad del 99.2% en la identificación de rostros. Para obtener estas métricas se utilizaron 6’094 imágenes para el entrenamiento del algoritmo y 1’523 imágenes de prueba, las cuales corresponden a 30 personas diferentes.

ABSTRACT

This research work seeks to solve a biometrics problem in the areas of digital image processing and computer vision in order monitor public transport drivers through face identification. The aim of this work is to propose a face detection and identification algorithm that is robust to the various lighting conditions to which drivers may be subjected in wild environments. The fast local histogram specification algorithm was implemented to reduce the noise due lighting conditions and a posterior study was made to analyze how this algorithm affects the performance in the tasks of face detection and identification, which are based on descriptors generated by histograms of oriented gradients and a convolutional neural network with FaceNet architecture respectively. Finally, support vector machine was used to carry out the classification task in face detection and identification stages, achieving a 71.98% sensibility and 97.65% specificity in face detection and a 99.2% accuracy in face identification. Those metrics were obtained using 6’094 images to train the algorithm and 1’523 images for testing, corresponding to 30 different people.

INTRODUCCIÓN

Según la Policía Nacional del Perú [21], los accidentes de tránsito en carreteras poseen la mayor cantidad de muertos y heridos por incidente. Las causas más comunes son los choques (44.20%) y los despistes (42.84%). Si bien no hay una estadística concreta en cuanto a la causa de estos accidentes, la probabilidad de que sea por el adormecimiento de los conductores es lo suficientemente alta para inspirar al artículo 30 del Decreto Supremo Nº 017-2009-MTC: "Los conductores de vehículos destinados a la prestación del servicio de transporte público de personas, de ámbito nacional y regional, no deberán realizar jornadas de conducción continuas de más de cinco (5) horas en el servicio diurno o más de cuatro (4) horas en el servicio nocturno" (Ministerio de Justicia del Perú [20]).

Para poder cumplir con la norma mencionada, las empresas de transporte interprovincial en el Perú contratan el servicio de dos (2) conductores para viajes cuya duración estimada supera el tiempo estipulado. Esto se realiza con la finalidad de que, para un tramo lo suficientemente largo, los conductores puedan realizar relevos y así el tiempo de manejo continuo no supere al de la norma. Sin embargo, se han registrado casos en los que los conductores hacen caso omiso a estos requerimientos impuestos por las empresas de transporte, manejando cada uno un tramo completo. Esto representa un riesgo económico para la empresa, tanto por la posibilidad de recibir una multa por incumplir la norma como por el peligro que representa un conductor adormecido para sus clientes (pasajeros). Por esta razón, la importancia de que las empresas (así como el MTC) puedan realizar un monitoreo del tiempo de manejo de sus conductores durante su jornada laboral sale a relucir.

El presente trabajo de Tesis propone desarrollar un algoritmo que permita realizar la detección y validación del rostro de un conductor vehicular basado en las características faciales presentes en una imagen adquirida en la cabina del vehículo a través de una cámara digital, para que las empresas de transporte público de personas puedan realizar un monitoreo automático de sus conductores. Si bien existen algoritmos en la literatura que permitan realizar la detección e identificación automática de rostros, el rendimiento de estos modelos depende en gran medida a las condiciones de iluminación del entorno en el cual

se adquieren las imágenes. Por otro lado, las condiciones de iluminación a las cuales es sometida la cabina de un conductor son bastante inestables debido a la gran cantidad de agentes externos que no se pueden controlar. En base a la divergencia de las soluciones presentes en la literatura y la problemática de las condiciones de iluminación, el presente tiene como objetivo alcanzar una taza de identificación de rostros mayor al 90% en entornos no controlados.

Con esta finalidad, el presente documento de Tesis contiene un total de cuatro capítulos. En el Capítulo I se expone un panorama general del contexto en el cual se realiza el presente trabajo de investigación puntualizando la descripción de la realidad problemática, los antecedentes bibliográficos y la unidad de análisis; en este capítulo también se hace mención al planteamiento de la investigación mediante la formulación del problema, el objetivo, hipótesis y las variables dependientes e independientes consideradas en el presente trabajo. El Capítulo II abarca el marco teórico y el marco conceptual, los cuales contienen la descripción de las técnicas, algoritmos y conceptos relevantes para una comprensión de la metodología propuesta en el presente documento de Tesis. En el Capítulo III se describe el desarrollo del algoritmo propuesto, partiendo por la definición del tipo, nivel y diseño de investigación; las fuentes de información e instrumentos a utilizar, para luego proponer una metodología y describir su implementación. El Capítulo IV consta del análisis y resultados del algoritmo implementado, describiéndose el proceso de validación, el *gold standar* y las métricas utilizadas. Finalmente, para complementar los capítulos anteriormente mencionados, se finaliza con una sección de conclusiones y recomendaciones, los cuales constan de discusiones basadas en las métricas y en la experiencia al momento de implementar el algoritmo.

CAPÍTULO I
ANTECEDENTES Y DESCRIPCIÓN DEL PROBLEMA

En el presente capitulo se mencionaran los trabajos y/o investigaciones presentes en la literatura así como una descripción de la situación problemática de la cual se inspiró la presente tesis.

1.1. Antecedentes bibliográficos

El tópico de la detección y reconocimiento de rostros tiene una larga tradición en el campo de la visión computacional, en el estado del arte actual se pueden encontrar trabajos basados en visión computacional que han alcanzado una asertividad del 95% a más en ciertas bases de datos, algunos ejemplos son Chen et al. en [4] y, Lu y Tang en [18].

Existen trabajos que han abordado este tema utilizando un conjunto de técnicas que ha tenido mucha acogida en los últimos años: *Deep Learning*. Este último concepto hace referencia a un conjunto de técnicas que pertenecen al campo del aprendizaje automatizado, distinguiéndose de las técnicas convencionales por la cantidad de datos (notablemente superior) que utilizan al momento del entrenamiento. Esto permite alcanzar un nivel más profundo en la representación de los datos de entrenamiento y por ende permite un mejor aprovechamiento de los mismos. En los trabajos de Sun et al. en [26] y de Sun et al. en [27] se puede apreciar como la asertividad aumenta notablemente con respecto a las técnicas convencionales de modelamiento al utilizar técnicas de *Deep Learning*, llegando a obtener un 99% en algunas bases de datos.

Sin embargo, estos trabajos utilizan en su mayoría bases de datos adquiridas en ambientes en donde se controla la iluminación.

En general, los trabajos que abordan problemas de visión computacional en entornos reales (no controlados) son más escasos. Tanto con un enfoque clásico de visión computacional (Feng et al. en [7]) o utilizando *Deep Learning* (Taigman et al. en [28]), la

asertividad de estos trabajos queda estancada en un 87%, alejado de lo alcanzado por otras técnicas que utilizan bases de datos controladas.

Este problema también ha sido abordado por el área de procesamiento de imágenes, en el cual se hace énfasis en la atenuación de ruido mediante el realce de contraste mediante especificación de histogramas (Liu et al. en [15] y, Lui y Yang en [17]), sin embargo, aunque se alcanzan resultados visuales prometedores, estos son aplicados en bases de datos que se alejan mucho a las imágenes de conductores, tanto en tamaño con en condiciones de iluminación.

1.2. Descripción de la realidad problemática

Como se pudo apreciar en la sección de antecedentes bibliográficos, existe una brecha entre los algoritmos de detección e identificación de rostros de alto rendimiento y los algoritmos diseñados para realizar procesamiento en imágenes en entornos reales (no controlados). SI bien los algoritmos de alto rendimiento poseen porcentajes de asertividad superiores al 95%, estos porcentajes están ligados a las bases de datos con los cuales fueron entrenados.

En el contexto del Perú, no hay herramientas o sistemas tecnológicos que permitan realizar el monitoreo de conductores en general. Los sistemas disponibles están sujetos a una serie de restricciones que los hacen inviables de utilizar en aplicaciones reales. Los sistemas más robustos se encuentran en el extranjero, pero representan un mayor costo económico al promedio y no aseguran que puedan cumplir con el monitoreo de conductores.

1.3. Formulación del problema

El problema a nivel técnico viene a ser: ¿es posible de desarrollar un algoritmo de procesamiento de imágenes e inteligencia artificial que permita la detección y reconocimiento de rostros en ambientes con iluminación no controlada con una asertividad del 95%?

1.4. Justificación e importancia de la investigación

El desarrollo del presente trabajo de Tesis brindará una herramienta de monitoreo de conductores a las empresas de transporte que no se puede conseguir en el mercado local, y que en general es escasa ya que la aplicación de técnicas de procesamiento de imágenes y visión computacional en entornos reales (no controlados) es limitada.

A nivel científico-ingenieril, el trabajo se justifica porque busca proponer técnicas de inteligencia artificial (en el área de visión computacional) que permitan resolver problemas de detección e identificación de rostros en imágenes con condiciones de iluminación variables. Además, el presente trabajo de investigación se justifica porque se busca resolver una de las problemáticas más relevantes del procesamiento de imágenes: la aplicación de algoritmos en ambientes reales (específicamente en este trabajo, con iluminación no controlada). La importancia de proponer soluciones ante esta problemática recae en que se expande el área de impacto de esta rama de la ciencia, pudiendo asistir a necesidades como la expuesta en la presente Tesis, en donde para aminorar el riesgo de fatiga del chofer (y, por consiguiente, el riesgo de accidentes en carretera) son necesarios aplicar este tipo de algoritmos.

1.4.1. Objetivos

En base a la formulación del problema, en el presente trabajo de Tesis se plantea el siguiente objetivo general:

- Desarrollar un algoritmo de procesamiento de imágenes e inteligencia artificial orientado a la detección y reconocimiento de rostros en entornos de iluminación no controlada con grado de asertividad del 95%.

Y los siguientes objetivos específicos:

- Proponer distintos tipos de estrategia de pre-procesamientos ante diferentes escenarios de condiciones de iluminación con respecto a distintos periodos del día (mañana, tarde y noche).
- Desarrollar estrategias de pre-procesamiento basadas en inspección visual para resaltar las características de un rostro, evaluando su rendimiento mediante un análisis de sensibilidad y especificidad.

- Integrar las estrategias de pre-procesamiento en la identificación de rostros, evaluando su rendimiento a través de un análisis de asertividad y AUC (área bajo la curva ROC - del inglés Receiver Operating Characteristics) basado en las etiquetas de las imágenes.

1.5. Hipótesis

En base a los antecedentes bibliográficos y los objetivos trazados anteriormente, el presente trabajo de Tesis plantea la siguiente hipótesis principal:

- Una imagen obtenida a través de una cámara web, colocada en la cabina de un conductor bajo condiciones de iluminación no controlada, mediante el uso de técnicas de procesamiento digital de imágenes y visión computacional, genera información suficiente para realizar la detección e identificación de un rostro.

Y las siguientes hipótesis específicas:

- Mediante la atenuación del efecto de iluminación en las imágenes capturadas en la cabina del conductor utilizando estrategias de pre-procesamiento para distintos escenarios de iluminación, es posible obtener nuevas imágenes en donde el contraste de las regiones que contienen un rostro no dependa de las condiciones de iluminación inicial.
- Las imágenes pre-procesadas pueden resaltar las características generales de un rostro, permitiendo realizar una detección de rostro de manera automática, con una sensibilidad superior al 70% y una especificidad superior al 90%.
- Las imágenes pre-procesadas pueden resaltar los detalles de un rostro, permitiendo realizar una identificación de rostro de manera automática, con una asertividad mayor al 90% y un valor de AUC (área bajo la curva ROC - del inglés Receiver Operating Characteristics) superior a 0.85.

1.6. Variables e indicadores

En la tabla 1.1 se muestran las variables e indicadores que los cuantifican, tomados en cuenta para el presente trabajo de investigación. Esta Tabla muestra además los objetivos específicos a los cuales están relacionados y, por ende, pueden ser trazados hacia sus respectivas hipótesis específicas.

Tabla 1.1. Variables e indicadores

Objetivo específico	Variables independientes	Variables dependientes	Variables Intervinientes	Indicadores
1. Estrategias de pre-procesamiento	Imagen digital	Imagen pre-procesada	Hora de adquisición Resolución de la imagen	Inspección visual
2. Detección de rostro	Imagen pre-procesada	Imagen rostro	Histogramas de gradientes	Sensibilidad Especificidad
3. Identificación de rostro	Imagen rostro	Etiqueta de rostro estimada	Rostro codificado	Asertividad AUC

1.7. Unidad de análisis

El presente trabajo, al desarrollar un algoritmo de detección y reconocimiento de rostros, requiere de imágenes correspondientes a la cabina de un conductor. Por consiguiente, las imágenes obtenidas en la cabina del conductor son consideradas como la unidad de análisis de la presente Tesis.

Para generar la base de datos de imágenes, se utilizó un vehículo particular y la participación de 30 personas. Cabe resaltar que, al momento de recolectar la base de datos, se buscó contar tanto con la mayor cantidad de variaciones de las condiciones de iluminación posibles como con la mayor cantidad de posiciones del rostro y/o posturas del conductor.

La locación de las imágenes adquiridas para la base de datos se restringe únicamente a algunos distritos de la provincia de Lima, por lo que se cuenta únicamente con imágenes de personas conduciendo dentro de la ciudad.

1.8. Tipo y nivel de Investigación

La presente tesis, al tener como fin una aplicación práctica para el reconocimiento de conductores en entornos no controlados, propone una investigación aplicada. Esta investigación es de tipo mixta ya que si bien los resultados se evaluaran en base a medidas y/o estadísticas del rostro (cuantitativo), los resultados parciales de los algoritmos de detección de rostros se harán de manera visual (cualitativa).

El nivel de investigación de la presente tesis es de tipo correlativa debido a que se busca encontrar la relación de un conjunto de imágenes con la identidad de un conductor.

1.9. Periodo de análisis

El periodo de análisis es de tipo puntual, ya que los datos (imágenes) se recolectaron pos sesiones de aproximadamente 30 minutos por cada conductor. La recolección de datos se realizó desde Febrero Julio de 2017 hasta Octubre del 2018.

1.10. Fuentes de información e instrumentos utilizados

La información utilizada en el presente trabajo de investigación proviene de distintas fuentes, tales como publicaciones en revistas indexadas y libros concernientes técnicas de aumento de contraste en imágenes, extracción de características, *Machine Learning* y *Deep Learning*. Además, se utilizaron bases de datos de rostros libres en interenet.

1.11. Técnicas de recolección y procesamiento de datos

Para la recolección de imágenes se utilizó una interfaz gráfica de usuario en MatLab®, donde se registró la hora en la que se capturaron las imágenes y se le asigna un identificador único a cada conductor.

CAPÍTULO II
MARCO TEÓRICO Y MARCO CONCEPTUAL

En este capítulo se presentan las técnicas y conceptos a utilizar para lograr los objetivos de la presente tesis.

2.1 El marco teórico

A continuación, se procede a explicar el fundamento teórico de las técnicas utilizadas en la metodología propuesta por el presente trabajo. Dichas técnicas corresponden a las áreas de procesamiento digital de imágenes (corrección gamma, especificación local de histograma), visión computacional (histograma de gradientes orientados, regresión en cascada basado en arboles de decisiones), aprendizaje automatizado (máquinas de soporte vectorial) y aprendizaje profundo (redes neuronales convolucionales).

2.1.1 Fundamentos de procesamiento digital de imágenes

El procesamiento digital de imágenes es la rama del procesamiento digital de señales que tiene como objetivo extraer y/o resaltar la información contenida en una imagen digital. En esta sección se procederá a definir los fundamentos básicos del procesamiento digital de señales que se utilizarán en el presente trabajo de investigación: la imagen digital y los modelos de color.

2.1.1.1 Imagen digital

De manera general, una imagen es la representación visual de uno o más objetos, la cual se puede plasmar en una fotografía, un dibujo o un video. Partiendo de este concepto,

una imagen digital viene a ser la representación de una imagen mediante un matriz numérica. Gonzales y Woods [10] definen a la imagen digital como el resultado de muestrear y cuantizar una imagen en una función bidimensional de la forma $f_{im}(c_1, c_2)$, en donde el valor numérico de f_im refleja una característica física determinada a una coordenada espacial (c_1, c_2). En la Ecuación 2.1 se muestra de manera gráfica esta definición, para una imagen de M filas y N columnas.

$$f_{im}(c_1, c_2) = \begin{bmatrix} f_{im}(0,0) & f_{im}(0,1) & \dots & f_{im}(0, N-1) \\ f_{im}(1,0) & f_{im}(1,1) & \dots & f_{im}(1, N-1) \\ \vdots & \vdots & \ddots & \vdots \\ f_{im}(M-1,0) & f_{im}(M-1,1) & \dots & f_{im}(M-1, N-1) \end{bmatrix} \tag{2.1}$$

Cabe resaltar que, como se muestra en la Ecuación 2.1, las coordenadas espaciales tienen como origen las coordenadas $(0,0)$, y este se ubica en la esquina superior izquierda. Además, cada elemento de este arreglo matricial recibe el nombre de pixel (acrónimo del concepto “elemento de una imagen” en inglés).

Sin embargo, [10] recomienda utilizar una notación de matriz más tradicional debido a su simplicidad. Esta notación se muestra en la Ecuación 2.2:

$$A = \begin{bmatrix} a_{0,0} & a_{0,1} & \dots & a_{0,N-1} \\ a_{1,0} & a_{1,1} & \dots & a_{1,N-1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{M-1,0} & a_{M-1,1} & \dots & a_{M-1,N-1} \end{bmatrix} \tag{2.2}$$

Siendo $a_{i,j} = f_{im}(c_1 = i, c_2 = j) = f_{im}(i, j)$, de manera que las Ecuaciones 2.1 y 2.2 hacen referencia a una misma imagen digital.

Además de M y N, el proceso de digitalización de una imagen requiere que se defina el número de niveles discretos o valores digitales que puede adoptar un pixel. Para representar este número de niveles discretos se utiliza la variable L y, debido a que el almacenamiento se realiza en una computadora, su valor numérico es una potencia de 2 (Ecuación 2.3):

$$L = 2^{k_b} \tag{2.3}$$

Donde k_b es el número de bits con los que cuantiza un pixel.

2.1.1.2 Modelos de color

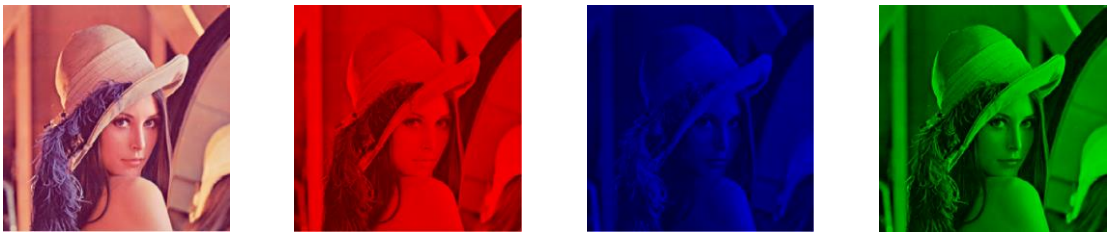
Los modelos o espacios de color son modelos numéricos que representan de forma abstracta los colores, asignándole un valor numérico único que puede tener varias componentes. Gonzales y Woods [10] define los modelos de color como una especificación de un sistema coordenado y un sub-espacio de este sistema, en el cual cada color es representado por un único punto.

En el presente trabajo se utilizarán el modelo de color RGB (de las siglas en inglés de rojo, verde y azul) y el espacio de color de escala de grises, por lo que se procede a realizar una descripción de estos conceptos:

- **RGB**: Este modelo de color se caracteriza por representar a los colores en base a los componentes espectrales primarios rojo, verde y azul. Además, esta representación la realiza sobre un sistema cartesiano de coordenadas, siendo un cubo el sub-espacio de interés de este modelo de color [10]. Las imágenes digitales que son representadas en este color presentan 3 componentes (cada una correspondiente a cada componente espectral primaria), por lo que se puede extender la definición de la Ecuación 2.1 y 2.2 a una función tridimensional (Ecuación 2.4):

$$A_{\vartheta} = \begin{bmatrix} a_{0,0,\vartheta} & a_{0,1,\vartheta} & \cdots & a_{0,N-1,\vartheta} \\ a_{1,0,\vartheta} & a_{1,1,\vartheta} & \cdots & a_{1,N-1,\vartheta} \\ \vdots & \vdots & \ddots & \vdots \\ a_{M-1,0,\vartheta} & a_{M-1,1,\vartheta} & \cdots & a_{M-1,N-1,\vartheta} \end{bmatrix} \tag{2.4}$$

Además, a partir de la Ecuación 2.3 se definen las matrices R, G y B para los valores de 0, 1 y 2 de la variable ϑ respectivamente. En la Figura 2.1 se muestra de manera visual una imagen RGB y sus componentes.



(a) (b) (c) (d)

Figura 2.1. Descomposición de (a) una imagen digital en su componentes (b) R, (c) G y (d) B.

- **Escala de grises**: Es un espacio de color que forma parte del modelo de color HSI [10]. Este se caracteriza por representar únicamente la intensidad de la información (generalmente representa la cantidad de luz) captada en un pixel, basándose únicamente en una sola componente. En la escala de grises, los pixeles con menor valor numérico son los correspondientes a las regiones oscuras (negro) y los pixeles con mayor valor numérico, a las regiones claras (blancas). Además, las imágenes representadas en este modelo de color entran en la definición de la Ecuación 2.2. En la Figura 2.2 se muestra un ejemplo de una imagen en escala de grises.

(a) (b)

Figura 2.2.(a) Imagen original y (b) su representación en escala de grises.

Cabe resaltar que la función de transformación de una imagen RGB a escala de grises es de naturaleza lineal, y está definida por le Ecuación 2.5 [10]:

$$GRAY_{i,j} = k_1 \cdot R_{i,j} + k_2 \cdot G_{i,j} + k_3 \cdot B_{i,j} \tag{2.5}$$

Siendo $GRAY$ la imagen en escala de grises; R, G y B las componentes de la imagen RGB; i, j las coordenadas de un pixel; y k_1, k_2 y k_3 constantes.

2.1.2 Realce de contraste en una imagen digital

El realce de contraste en una imagen digital consiste en realizar una transformación a los valores de sus pixeles con la finalidad de generar un mayor contraste en las zonas de interés. A continuación, se procederá a explicar las técnicas más utilizadas para mitigar los efectos de la iluminación: la Corrección Gamma y la Especificación Rápida Local de Histograma. Estas técnicas se utilizan para la etapa de pre-procesamiento.

2.1.2.1 Corrección Gamma

La corrección Gamma hace referencia a una transformación exponencial de los valores de los pixeles en una imagen digital [10]. Esta transformación tiene la forma básica que se muestra en la Ecuación (2.6):

$$o_{i,j} = c.{r_g}^{\gamma} \tag{2.6}$$

donde r_g es el valor original del pixel, o_g el valor corregido del pixel, γ la variable que controla la corrección y c una constante que escala los valores de o_g de 0 a $L-1$.

Este tipo de corrección se utiliza para aumentar el contraste en las regiones oscuras o claras de la imagen, dependiendo del valor de γ, a costa de sacrificar contraste en las regiones complementarias. En la Figura 2.3 se puede apreciar la naturaleza de esta transformación.

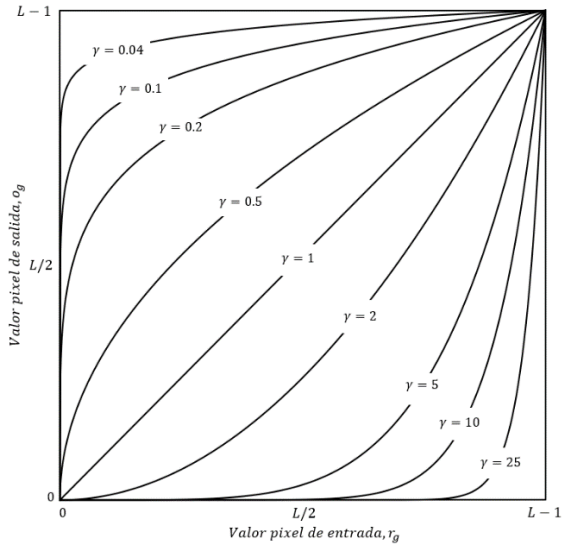


Figura 2.3**.** Gráfica de o_g vs r_g para diferentes valores de γ.

Finalmente, se interpreta de la Figura 2.3 que para valores de γ > 1 se aumenta el contraste de las regiones claras en una imagen mientras que para valores de γ < 1 se aumenta el contraste de las regiones oscuras. Cuando γ toma el valor de 1, no existe ninguna modificación en el valor del pixel y en su defecto, en la imagen.

2.1.2.2 Ecualización de histograma

El histograma de una imagen hace referencia a la frecuencia relativa de los niveles de grises presentes en la imagen, esto se puede interpretar además como la función de probabilidad de los valores del pixel en la imagen [10]. Analíticamente, el histograma registra el número de pixeles que pertenecen a cierta tonalidad (nivel) de gris.

En la Figura 2.4 se puede apreciar una imagen en escala de grises acompañada de su histograma.

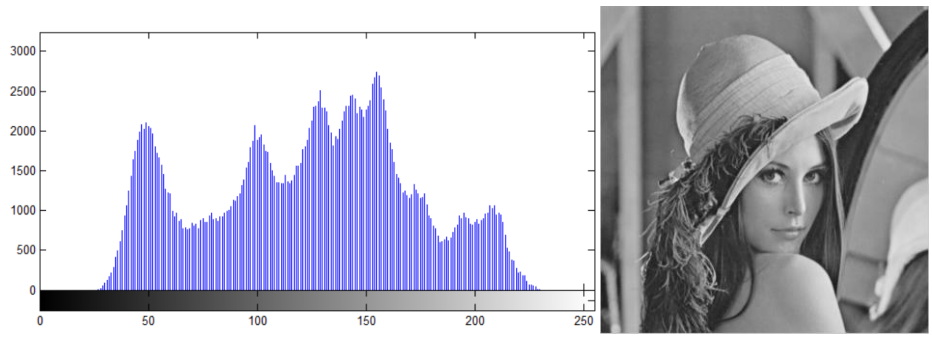


Figura 2.4. Histograma e imagen a la cual corresponde.

Según lo explicado, se puede concluir que al realizar una transformación al valor numérico de los pixeles de una imagen, el histograma de esta nueva imagen varía con respecto al de la original. La ecualización del histograma de una imagen consiste en el proceso inverso, modificar el histograma original de una imagen para generar una de mayor contraste. El objetivo de la ecualización de histograma es el de repartir de manera uniforme los valores numéricos de los pixeles de la imagen, lo cual se logra mediante una función que transforma al histograma en una función de probabilidad uniforme (o lo más parecido posible).

En la Figura 2.5 se puede apreciar el efecto de la ecualización de histograma en una imagen.

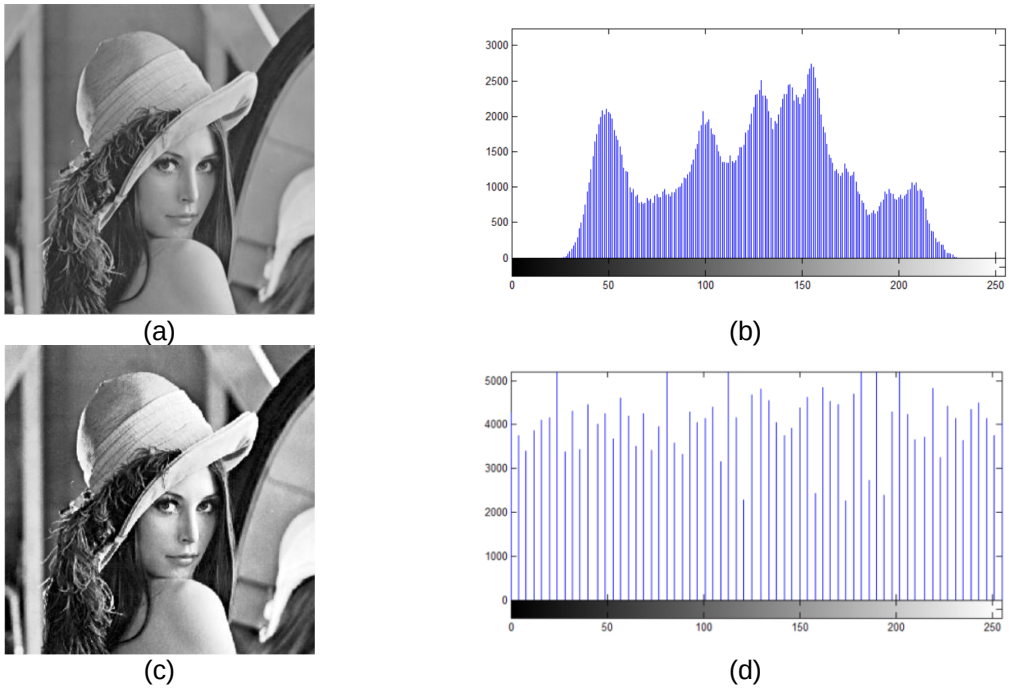


Figura 2.5. (a) Imagen original, (b) histograma de la imagen original, (c) imagen con histograma ecualizado e (d) histograma de la imagen “(c)”.

2.1.2.3 Especificación de histograma

La Especificación de Histograma o *Histogram Matching* es una transformación en el histograma que busca generar contraste en la imagen. La especificación de histograma busca que el histograma adopte una distribución específica (no necesariamente una distribución uniforme como en la ecualización). Este enfoque, dependiendo de la aplicación puede resultar más útil que la ecualización de histograma [10].

Liu et al. [16] señala que la especificación de histograma, de manera resumida, se obtiene de la Ecuación (2.7):

$$y_f = \arg\min_{t} \left| H(t) - P(x_f) \right|, \tag{2.7}$$

donde $\arg\min$ es una función que retorna el argumento que minimiza la expresión $\left| H(t) - P(x_f) \right|$, x_f representa el valor del pixel de la imagen original, P la función que acumula el histograma (CDF) de x_f, H el histograma acumulado deseado y y_f el valor del pixel de la imagen resultante.

A continuación se procederá a explicar una el concepto de la especiación local de histograma y una técnica que la optimiza.

2.1.2.4 Especificación rápida local de histograma

El método de especificación local de histograma consiste en definir una vecindad de pixeles rectangular y/o cuadrada, la cual tendrá el nombre de ventana, y desplazar el centro de pixel a pixel. Para cada ventana se realiza el método de especificación de histograma, teniendo en cuenta a la ventana como imagen original y especificando todas a un mismo histograma deseado [10]. Sin embargo, por cada ventana solo se computa la modificación realizada en el pixel central por lo que si la imagen original es de dimensión MxN, se realizarán M.N especificaciones, lo cual se traduce a un alto costo computacional.

Liu et al. [16] propone un método para reducir el costo computacional de la especificación local de histograma y lo llaman Especificación Rápida Local de Histograma (FLHS). A continuación se procede a indicar conceptos básicos y definiciones utilizados en este método.

Para un pixel central de una ventana perteneciente a una imagen I en escala de grises con L niveles digitales, se define la CDF de un nivel de gris l en la Ecuación (2.8). Se puede apreciar que el objetivo de esta ecuación es el de procesar la menor cantidad de datos para obtener la CDF.

$$P(l) = \begin{cases} \sum_{t=0}^{l} h(t), & l < \frac{L}{2} \\ 1 - \sum_{t=0}^{l} h(t), & l \geq \frac{L}{2} \end{cases} \tag{2.8}$$

donde $h(t)$ es el histograma normalizado la ventana.

Como se mencionó anteriormente, el histograma de cada ventana se actualiza para realizar la especificación. En cada actualización de la ventana se procederá a dividir al intervalo [0 - (L -1)] de los histogramas en p sub-intervalos sin solapamiento. Al histograma formado por la unión de estos sub-intervalos se le llamará sub-histograma.

Para el caso en el que $L/2$ sea par, el número de sub-intervalos óptimos para dividir el histograma de cada ventana viene dado por $p^* = \left\lceil \sqrt{L/2 + 5/4} - 1/2 \right\rceil$, siendo $\lceil \cdot \rceil$ la representación de la aproximación al entero superior de un número. Sin embargo, estos sub-intervalos no son de longitud constante. La longitud optima de cada sub-intervalo entre [0-$L/2$-1] se muestra en la Ecuación (2.9).

$$z_i^* = 2(p^* - e_i - i + 1), \qquad i = 1, \dots p^* \tag{2.9}$$

Además, se define z' como el reflejo de como z^* y a $z = [z^*, z']$.

Una vez definidos la longitud de sub-intervalos, se genera una tabla de mapeo T entre el histograma y el sub-histograma de cada ventana, como se aprecia en la Figura 2.6.

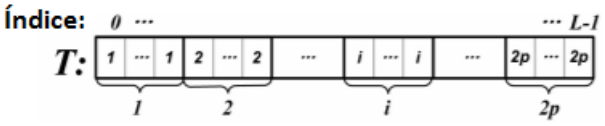


Figura 2.6. Tabla de mapeo entre el histograma y sub-histograma de una ventana [16].

Previo al cálculo de la CDF (Ecuación 2.10) de un nivel de gris l, se define Z al sub-histograma formado a partir del histograma de la ventana y la tabla T.

$$P(l) = \begin{cases} \sum_{j=1}^{i-1} Z(j) + \sum_{j \in I\mathcal{T}_i \wedge j \le l} h(j), & l < \frac{L}{2} \wedge l \le \lfloor \mathcal{T}_i^m \rfloor \\ \sum_{j=1}^{i} Z(j) - \sum_{j \in I\mathcal{T}_i \wedge j > l} h(j), & l < \frac{L}{2} \wedge l > \lfloor \mathcal{T}_i^m \rfloor \\ 1 - \left(\sum_{j=1}^{i-1} Z(j) + \sum_{j \in I\mathcal{T}_i \wedge j \le l} h(j) \right), & l \ge \frac{L}{2} \wedge l \ge \lfloor \mathcal{T}_i^m \rfloor \\ 1 - \left(\sum_{j=1}^{i} Z(j) - \sum_{j \in I\mathcal{T}_i \wedge j > l} h(j) \right), & l \ge \frac{L}{2} \wedge l < \lfloor \mathcal{T}_i^m \rfloor \end{cases} \tag{2.10}$$

donde i es el sub-intervalo que contiene al nivel de gris l, $\mathcal{T}_i$ son los índices contenidos en el sub-intervalo i, $\mathcal{T}_i^m$ el valor promedio de grises del sub-intervalo i y $\lfloor \cdot \rfloor$ el máximo entero de un número.

Una vez definido los conceptos básicos se procede a mostrar el algoritmo de FLHS en la Figura 2.7, del cual se procederá a dar una breve explicación en los puntos más importantes.

Entrada: Imagen $I \in \mathcal{L}^{HxW}$, Histograma con la función de densidad acumulada deseada $\{H_{ij}(l)\}_{(i,j,l)\in AxL}$
Parámetro: Radio de ventana r_f
Salida: Imagen Procesada Yf
1: Agregar bordes a la imagen I para obtener la imagen Xf, donde $Xf \in \mathcal{L}^{(H+2r)x(W+2r)}$
2: **Definir** $Yf = Xf$, $h(l) = 0 \quad \forall(l) \in \mathcal{L}$
3: **Definir** $(i,j) = (0,0)$; $d = 1/(2r_f + 1)^2$
4: **Para todo** $(i,j) \in A$:
5: Si $(i,j) = (0,0)$
6: $\forall(u,v) \in Sf_{i,j}^r$: $h(xf_{u,v}) = h(xf_{u,v}) + d,\ Z(T(xf_{u,v})) = Z(T(xf_{u,v})) + d$;
7: **De otro modo**
8: $\forall(u,v) \in Sf_{i',j'}^r \backslash Sf_{i,j}^r$: $h(xf_{u,v}) = h(xf_{u,v}) - d,\ Z(T(xf_{u,v})) = Z(T(xf_{u,v})) - d$;
9: $\forall(u,v) \in Sf_{i,j}^r \backslash Sf_{i',j'}^r$: $h(xf_{u,v}) = h(xf_{u,v}) + d,\ Z(T(xf_{u,v})) = Z(T(xf_{u,v})) + d$;
10: **Fin del Si**
11: Calcular la CDF $P_{i,j}(xf_{i,j})$ para $xf_{i,j}$ utilizando h, T y Z en la Ecuación (2.7)
12: Utilizar el algoritmo de búsqueda binaria para resolver:

$$yf_{ij} = \arg\min_{t} |H_{i,j}(t) - P_{i,j}(xf_{i,j})|$$

13: $Yf_{i,j} = yf_{i,j}$;
14: $(i',j') = (i,j)$; $(i,j) = siguiente(i',j')$ // El centro de la ventana se traslada a un pixel adyacente
15: **Fin del Para todo**
16: $Yf = Yf(1:H, 1:W)$;

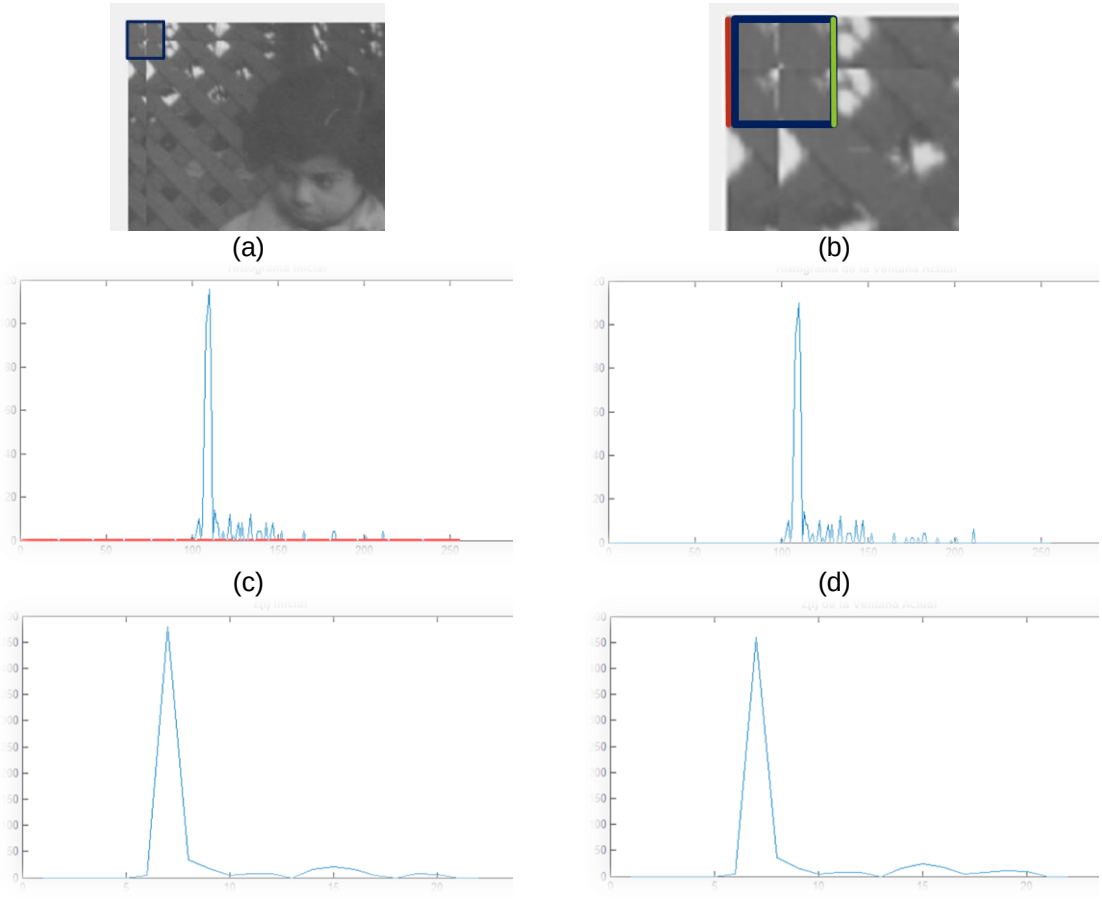
Figura 2.7. Pseudocódigo del algoritmo FLHS [16].

Para este método se considera una ventana cuadrada de lado "2r_f+1", siendo r el único parámetro a definir en este método. El paso 1 del algoritmo consiste en agregar bordes de ancho r a la imagen original tal como se muestra en la Figura 2.8. Tal como se aprecia en el paso 5 y 6, el histograma $h()$ y sub-histograma $Z()$ se computan en su totalidad en el primer pixel de la imagen y a medida que la ventana se va desplazando se agregan nuevos pixeles $Sf_{i,j}^r$ (paso 8) y se descartan los pixeles que se dejan atrás $Sf_{i',j'}^r$ (paso 9), evitando re calcular una cantidad considerable de pixeles. Este proceso se puede apreciar en la Figura 2.9 (a) donde se muestra la ventana correspondiente al primer pixel de la imagen y la Figura 2.9 (b) donde se representa de color verde al conjunto de pixeles $Sf_{i,j}^r$ y del rojo al conjunto de pixeles $Sf_{i',j'}^r$.

El paso 12 hace referencia a la Ecuación (2.7), con la sugerencia de que se puede realizar la búsqueda más óptima utilizando el algoritmo de búsqueda binaria. El paso 13 indica que solo se computa el valor del pixel central. En la Figura 2.10 se puede apreciar el resultado del método FLHS utilizando como histograma deseado un histograma logarítmico. Finalmente, vale mencionar que [17] sugiere que se utilice histogramas orientados a rostros para la especificación local.

(a) (b)

Figura 2.8. (a) Imagen original e (b) Imagen con bordes duplicados.



(a) (b)

(c) (d)

(e) (f)

Figura 2.9. (a) Primera ventana en la imagen con bordes duplicados, (b) Ventana adyacente a la ventana mostrada en (a), (c) histograma de la ventana mostrada en (a), (d) histograma de la ventana mostrada en (b), (e) sub-histograma de la ventana mostrada en (a) y (f) sub-histograma de la ventana mostrada en (b).

(a) (b)

Figura 2.10. (a) Imagen original e (b) Imagen resultante al aplicar FLHS.

2.1.3 Extracción de características

La extracción de características, en procesamiento de imágenes, consiste en obtener valores numéricos en función de una imagen con la finalidad de eliminar información innecesaria y facilitar el reconocimiento de objetos. A continuación se procede a explicar las técnicas utilizadas para la extracción de características: Histograma de Gradientes Orientadas (utilizado para detectar rostros), Redes Neuronales Convolucionales (utilizado para extraer características de un rostro) y Regresión en cascada basado en árboles de decisión (utilizado para estimar la orientación y/o postura de un rostro).

2.1.3.1 Histograma de gradientes orientados

El histograma de Gradientes Orientados (HOG, del inglés *Histogram of Oriented Gradients*) es un descriptor utilizado en procesamiento de imágenes y visión computacional para la detección de objetos (Dalal y Triggs [5]). Para obtener este descriptor es necesario realizar una serie de pasos, los cuales se nombrarán a continuación:

Se comienza con el cálculo de la gradiente de una imagen, para lo cual se utiliza una máscara de derivación discreta tanto en las direcciones horizontal y vertical de la imagen. Específicamente, el cálculo de la gradiente se realiza mediante el filtrado de la imagen por los siguientes *kernels*: $D_X = [-1 \quad 0 \quad 1], D_Y = [-1 \quad 0 \quad 1]^{\mathrm{T}}$.

Por consiguiente, para una imagen I, las derivadas se obtienen mediante la convolución de I con D_X y D_Y (Ecuación 2.11):

$$I_{X_{i,j}} = I_{i,j} * D_{X_{i,j}}; I_{Y_{i,j}} = I_{i,j} * D_{Y_{i,j}} \tag{2.11}$$

Donde I_X e I_Y son las derivadas de la imagen I con respecto a la horizontal y vertical respectivamente. Por último, se calcula la magnitud de la gradiente y su orientación (Ecuaciones 2.12 y 2.13 respectivamente).

$$|G|_{\mathrm{i,j}} = \sqrt{I_{X_{i,j}}^2 + I_{Y_{i,j}}^2} \tag{2.12}$$

$$\theta_{i,j} = \tan^{-1}\frac{I_{Y_{i,j}}}{I_{X_{i,j}}} \tag{2.13}$$

A continuación, en la Figura 2.11, se procede a mostrar unos resultados a manera de ejemplo.

(a) (b) (c) (d)

Figura 2.11. (a) Imagen original, (b) derivada vertical de la Imagen “(a)”, (c) derivada horizontal de la imagen “(a)” y (d) magnitud de la gradiente (Brehar [2]).

Una vez realizado el cálculo de la gradiente, se procede a la ponderación de la orientación la cual se realiza en sub-muestras de la imagen correspondiente a la magnitud de la gradiente (celdas), cuyo tamaño se define a conveniencia. En la Figura 2.12 se puede apreciar un arreglo de celdas.

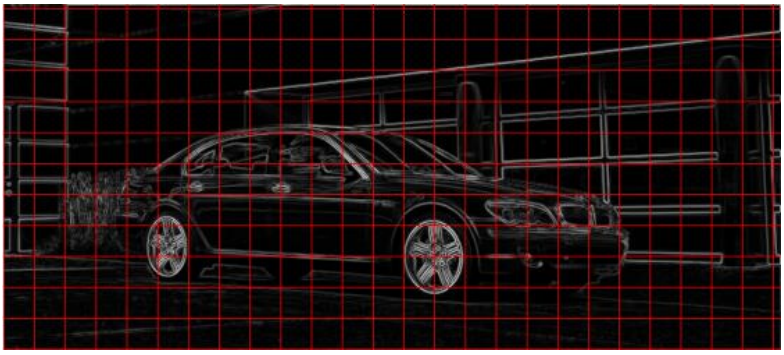


Figura 2.12. División por celdas de la magnitud de la gradiente [2].

Luego, la ponderación de la orientación se realiza por cada celda, promediando las orientaciones de cada pixel.

Posteriormente, se realiza una descripción por bloques. Se denomina un bloque como un conjunto de celdas vecinas. Para añadir un grado de robustez, existe un grado de solapamiento entre bloques. Las configuraciones más usadas de bloques son la rectangular (R-HOG) y circular (C-HOG). Finalmente, el descriptor HOG es un vector que contiene los histogramas de los bloques conformado por celdas normalizadas (Figura 2.13).

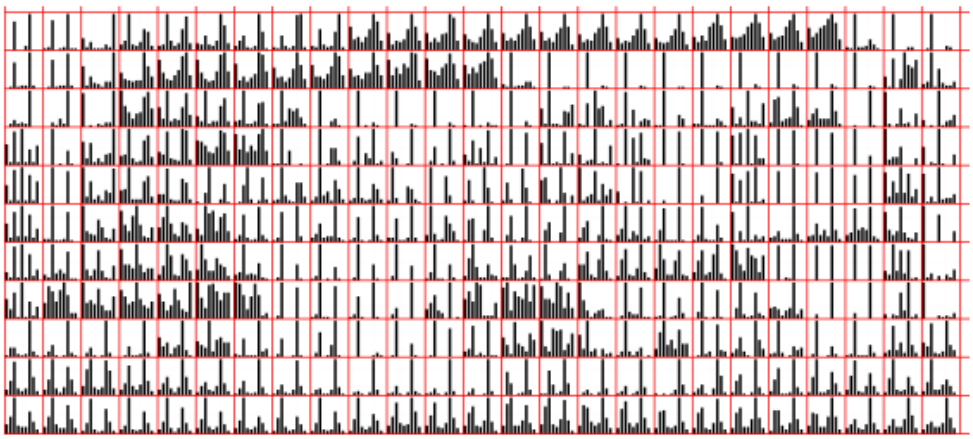


Figura 2.13. Histogramas de gradientes orientadas por cada celda [2].

Finalmente, se realiza una normalización de dichos bloques. Para esto se considera v como el vector no normalizado de los histogramas de las gradientes y $\|\mathbf{v}\|_k$ la norma de grado $k = 1,2,\dots,n$ del vector v. En la Tabla 2.1 se pueden apreciar los tipos de normas más utilizados.

Tabla 2.1. Tipos de normalización para un vector "**v**".

NORMA	FÓRMULA
L_2norm	$f(\mathbf{v}) = \frac{\mathbf{v}}{\sqrt{\|\mathbf{v}\|_2{}^2 + e_{norm}^2}}$
L_1norm	$f(\mathbf{v}) = \frac{\mathbf{v}}{\|\mathbf{v}\|_1 + e_{norm}}$
L_2sqrt	$f(\mathbf{v}) = \sqrt{\frac{v}{\|\mathbf{v}\|_1 + e_{norm}}}$

siendo e_{norm} una constante pequeña que evita la división por cero.

En la Figura 2.14 se puede apreciar de manera gráfica la síntesis de lo anteriormente explicado.

Figura 2.14. Cálculo de los Histogramas de Gradientes Orientados [2].

2.1.3.2 Redes neuronales convolucionales

Las redes neuronales convolucionales (CNN, del inglés *Convolutional Neural Networks*) son un tipo de red neuronal artificial (ANN, del inglés *Artificial Neural Network*) con propagación hacia adelante (*feedforward*) propuesto por LeCun et al. en [14]. Este tipo de redes se utiliza ampliamente en el campo del aprendizaje profundo (*deep learning*) y se encuentra actualmente presente en el estado del arte de diversas áreas de la visión computacional.

Las entradas de una CNN son de topología tipo grilla con dimensiones definidas. La dimensión de las entradas puede variar, tomando como ejemplo las series temporales (grilla 1D), una imagen en escala de grises (grilla 2D) o una imagen a color (grilla 3D).

Los conceptos básicos de una ANN, desde la propagación de la data hasta el proceso de entrenamiento son explicados por Bishop en [1]. En la presente sección se expondrán los puntos en los que la CNN difiere de una ANN convencional (perceptrón multicapa) y se explicará el fundamento teórico de una CNN orientada al procesamiento de imágenes.

La CNN recibe su nombre debido a que utiliza la operación matemática de convolución. A diferencia de las ANN convencionales, en donde las entradas se relacionan entre sí mediante multiplicaciones con pesos, las entradas de una CNN están sometidas a capas de convolución por filtros.

Sin embargo, existen ciertas diferencias entre la definición matemática de la convolución y el operador utilizado en las CNN (Goodfellow et al. [11]). Propiamente dicho, la convolución entre dos funciones discretas de dos dimensiones es un operador matemático que se define de la siguiente manera (Ecuación 2.14):

$$Scnn_{i,j} = (I * Kcnn)_{i,j} = \sum_{m}\sum_{n} I_{m,n} Kcnn_{i-m,j-n} \tag{2.14}$$

En general, y en posteriores referencias del presente trabajo, el primer argumento de la función de convolución (I) recibirá el nombre de entrada (o imagen de entrada) mientras que el segundo argumento ($Kcnn$) recibirá el nombre de *kernel*. Así mismo, la salida de la función de convolución ($Scnn$) será llamada mapa de características (*feature map*).

La convolución es una operación matemática que posee la propiedad conmutativa, por lo que un equivalente a la expresión mostrada en la Ecuación 2.14 viene a ser (Ecuación 2.15):

$$Scnn_{i,j} = (Kcnn * I)_{i,j} = \sum_{m}\sum_{n} I_{i-m,j-n} Kcnn_{i,j} \tag{2.15}$$

Esta última ecuación es más directa al momento de la implementación, ya que generalmente la entrada posee un tamaño significativamente más grande que el del *kernel*. Por esta razón, esta ecuación es la más utilizada al momento de implementarse la convolución en librerías de programación.

De las Ecuaciones 2.14 y 2.15 se puede observar que la razón por la cual la convolución posee la propiedad conmutativa es por el “giro” que se realiza en el kernel, es decir, se invierte el orden de los elementos del *kernel* en sus 2 dimensiones. Sin embargo,

se puede plantear también que el “giro” en el *kernel* es el responsable de que la convolución tenga propiedad conmutativa.

Si bien la propiedad conmutativa en la convolución es bastante útil en algunas demostraciones matemáticas, en una CNN no tiene mayor impacto. Por esto, la mayoría de implementaciones de CNN utilizan la función de correlación cruzada (*cross-correlation*), similar a la convolución sin realizar el “giro” del *kernel* [11]. En la Ecuación 2.16 se muestra la definición de la correlación cruzada:

$$Scnn_{i,j} = (I * Kcnn)_{i,j} = \sum_{m}\sum_{n} I_{i+m,j+n} Kcnn_{i,j} \tag{2.16}$$

Debido a que la mayoría de fuentes de información y librerías de programación referentes a la CNN utilizan el nombre de convolución para hacer referencia a la correlación cruzada, el presente trabajo también utilizará dicha convención.

Por otro lado, en el campo del procesamiento de imágenes, la Ecuación 2.16 es utilizada también para describir el proceso de filtrado lineal de un mascara o *kernel* K sobre una imagen I (Szeliski [25]). En la Figura 2.15 se muestra el resultado de una la convolución de un filtro laplaciano (explicado en [10]) sobre una imagen con la finalidad de resaltar sus bordes.

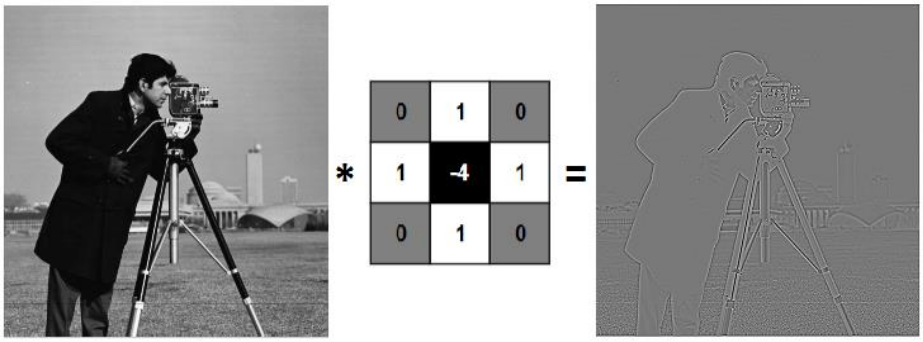


Figura 2.15. Ejemplo de filtrado de imagen

En base a esto, se puede considerar a las capas de convolución como una capa de filtrado y a la etapa de entrenamiento de una CNN como la búsqueda de filtros que permitan extraer la mayor cantidad de características de una imagen. Es por esto que en el campo del aprendizaje profundo (*deep learning*), no se acostumbra realizar un pre-procesamiento a las imágenes de entrada a una CNN, sino que se plantea que la red “aprenda” la manera más adecuada de pre-procesar una imagen en base al problema que quiere resolver.

El flujo de procesamiento de una CNN comienza con la capa de entrada o capa visible, la cual recibe los datos de entrada a la red. Después, las capas de procesamiento manipulan y extraen características de la data de entrada. Cabe resaltar que las capas de procesamiento son secuenciales, es decir, tienen una secuencia definida en donde la salida de la primera es la entrada de la segunda, la salida de la segunda es la entrada de la tercera y así sucesivamente. Además, a mayor número de capas ocultas mayor es la complejidad de las características extraídas o más "profundo" es el aprendizaje de la red. Finalmente, a la salida de la red se encuentran las capas de conectividad total (*fully connected layers*), en donde se interconectan todos los elementos de los mapas de características resultantes mediante operaciones lineales.

En la Figura 2.16 se muestra la arquitectura genérica de una CNN, tomando como ejemplo una imagen en escala de grises (grilla 2D).

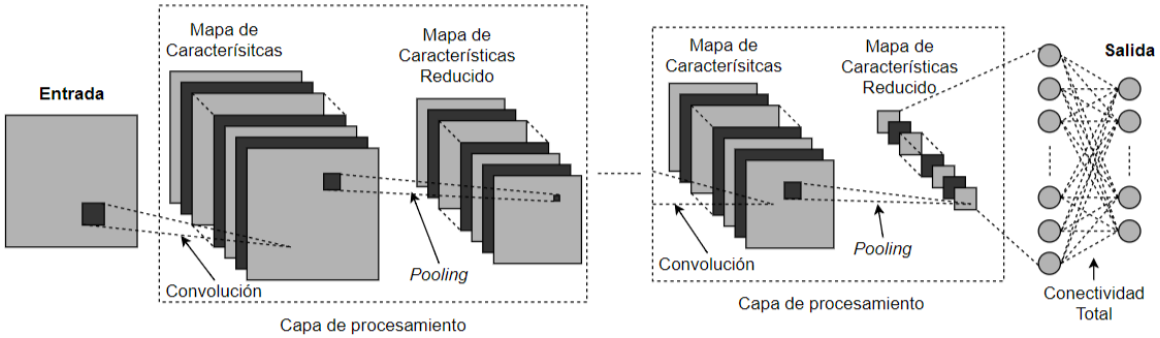


Figura 2.16. Arquitectura general de una Red Neuronal Convolucional (CNN).

De manera general, las capas de procesamiento de una CNN pueden presentar 3 tipos de etapas de procesamiento [11].

En la primera etapa se realiza una serie de convoluciones en paralelo para generar nuevos patrones mediante operaciones lineales. Las convoluciones en paralelo filtran las entradas con distintos *kernels* y los resultados se concatenan para formar el espacio de características, es decir, la información de cada proceso de filtrado no interfiere con los demás. En la Figura 2.16 se representa la concatenación de resultados como los volúmenes de 3 dimensiones conformado por matrices de diferentes colores.

En la segunda etapa se ejecuta una función de activación sobre los mapas de características obtenidos en las capas de convolución. Las funciones de activaciones son de tipo no lineal y se ejecutan sobre cada uno de los elementos que conforman el mapa de características.

En la tercera etapa se emplea una función de *pooling* para modificar la salida de la capa oculta. Las funciones de *pooling* transforman y reducen el tamaño de los mapas de características teniendo en cuenta la ubicación de los elementos y la estadística de sus vecinos. Esta etapa brinda cierto grado de robustez ante la variación de la posición de las características relevantes en la data de entrada a la red [11].

Las capas de procesamiento, por lo general, están formadas por dos tipos de capas que engloban las etapas de procesamiento previamente explicadas: las capas de convolución y las capas de *pooling*.

Las capas de convolución realizan una sucesión de convoluciones en paralelo e inmediatamente después aplican una función de activación no lineal (primera y segunda etapa). En las capas de convolución de una CNN, los *kernels* vienen a estar conformados por un conjunto de parámetros que realizan una activación lineal sobre las entradas de la capa. Estos parámetros reciben el nombre de "pesos" y al proceso que busca optimizar dichos parámetros se le conoce como entrenamiento.

Basándose en la Ecuación 2.14, la propagación hacia delante por una capa de convolución en una CNN se expresa en las Ecuaciones 2.17 y 2.18:

$$xcnn_{i,j}^{l} = \sum_{a=0}^{m-1} \sum_{b=0}^{n-1} ycnn_{i+a,j+b}^{l-1} wcnn_{a,j}^{l} \tag{2.17}$$

$$ycnn_{i,j}^{l} = \sigma\left(xcnn_{i,j}^{l}\right) \tag{2.18}$$

siendo $xcnn^{l}$ el mapa de características calculado sin activación lineal, $ycnn^{l-1}$ el mapa de características resultante de la capa anterior, $wcnn^{l}$ los pesos del *kernel* de convolución de dimensión MxN , $\sigma(\cdot)$ la función de activación y $ycnn^{l}$ el mapa de características resultante de la capa de convolución l.

En la tabla 2.2 se muestran algunas de las funciones de activación más populares utilizadas en una CNN [11].

Tabla 2.2. Tipos de función de activación $\boldsymbol{\sigma}(\cdot)$ [11].

FUNCIÓN DE ACTIVACIÓN	FÓRMULA
sigomoid	$\sigma(\tau) = \frac{1}{1+e^{-\tau}}$
ReLu	$\sigma(\tau) = \max(0, \tau)$
tanh	$\sigma(\tau) = \frac{e^{\tau} - e^{-\tau}}{e^{\tau} + e^{-\tau}}$

Así mismo, se denomina capas de *pooling* (o de sub-muestreo) a aquellas que ejecutan dicha función sobre las salidas de las capas de convolución (tercera etapa). El *pooling*, por lo general, es una operación que reemplaza al mapa de características de entrada por uno de menor tamaño, sintetizando la información del primero. En la Figura 2.17 se puede apreciar cómo funciona el *pooling* en una matriz, siendo la ventana de *pooling* de un tamaño de 2x2.

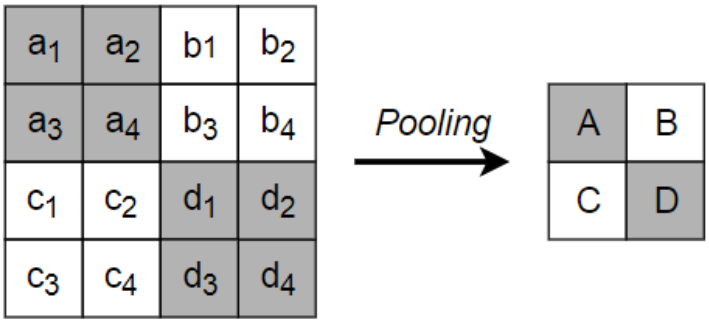


Figura 2.17. Ejemplo de *pooling* con una ventana de 2x2 en una matriz de 8x8.

siendo $\mathrm{A} = f_{pool}(\mathrm{a}_1, \mathrm{a}_2, \mathrm{a}_3, \mathrm{a}_4)$, $\mathrm{B} = f_{pool}(\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4)$, $\mathrm{C} = f_{pool}(\mathrm{c}_1, \mathrm{c}_2, \mathrm{c}_3, \mathrm{c}_4)$ y $\mathrm{D} = f_{pool}(\mathrm{d}_1, \mathrm{d}_2, \mathrm{d}_3, \mathrm{d}_4)$.

Esta capa es importante por dos razones: reduce el costo de procesamiento de la red y brinda tolerancia a pequeñas traslaciones de características en la data de entrada. Lo primero se produce debido a que se merma de manera significativa el tamaño de los mapas de características a lo largo de la red, agilizando tanto el procesamiento de datos como el proceso de aprendizaje. Por otro lado, la segunda ventaja se refiere a que la salida de una CNN no varía de manera drástica ante pequeños desplazamientos espaciales de las características en la data de entrada, protegiendo a la CNN del sobre entrenamiento.

En la Tabla 2.3 se muestra algunas de las funciones de *pooling* utilizadas en CNNs [11].

Tabla 2.3. Tipos de función de *pooling* ($\boldsymbol{f_{pool}}$) para un vector "$\mathbf{ve}$" de "$\boldsymbol{n}$" elementos

POOLING	FÓRMULA
$average - pooling$	$f_{pool}(\mathbf{ve}) = \frac{1}{n}\sum_{i}^{n} \mathbf{ve}_i^2$
$L_2 norm - pooling$	$f_{pool}(\mathbf{ve}) = \frac{1}{n}\sqrt{\sum_{i}^{n} \mathbf{ve}_i^2}$
$max - pooling$	$f_{pool}(\mathbf{ve}) = \max_{i=1,..,n} \mathbf{ve}_i$

El flujo de procesamiento de una CNN termina con las capas de conectividad total (*fully connected layer*), las cuales analizan toda la información que ingresa a estas. Para esto, re-estructuran el mapa de características de entrada en un vector. Una vez re-estructurada la entrada, se combinan todos sus elementos de dicho vector mediante una serie de de activaciones polinomiales de primer grado, siendo los coeficientes de dicha activación polinomial definidos como pesos $wcnn$ y al termino independiente de los mismos como bias.

El procesamiento en esta capa se realiza de la misma forma que en la capa oculta de una ANN perceptron multi-capa [1], describiéndose en las ecuaciones 2.19 y 2.20:

$$xcnn_j^l = \sum_{a=0}^{i-1} ycnn_a^{l-1} wcnn_{a,j}^l \tag{2.19}$$

$$ycnn_j^l = \sigma\left(xcnn_j^l\right) + bcnn_j \tag{2.20}$$

siendo $xcnn^l$ la activación lineal de la capa de conectividad total l, $ycnn^{l-1}$ el vector de entrada, $wcnn^l$ los pesos de activación de la capa de conectividad total, $bcnn$ la variable de compensación (bias), $\sigma(\cdot)$ la función de activación (Tabla 2.2) y $ycnn^l$ la salida de la capa de conectividad total l (que representa la salida de la CNN).

La capa de conectividad total es importante debido a que comunica y/o interrelaciona el resultado de los diferentes procesos de filtrado en las capas de convolución, que se encontraban aisladas entre sí, sintetizando la información relevante. Esto le permite a la

CNN discernir en qué magnitud impactan las características obtenidas en los procesos de convolución. Al igual que en las capas de convolución, el entrenamiento de la CNN tiene como finalidad el encontrar un conjunto de pesos $wcnn$ que optimice a la red.

Por último, el entrenamiento de una CNN se basa en los mismos principios que una ANN [1]. Consiste en minimizar una función de error E que refleje y/o penalice las discrepancias de la salida de la red $\{ycnn_n\}$ con un valor esperado $\{t_n\}$ para una base de datos formada por las entradas $\{xcnn_n\}$, donde $n = 1, \dots, Nbd$.

La función de error E se define en la Ecuación 2.21:

$$E(wcnn) = \frac{1}{2}\sum_{n=1}^{Nbd} \|ycnn_n(wcnn) - t_n\|^2 \tag{2.21}$$

siendo $ycnn_n$ la salida de la CNN dada la entrada $xcnn_n$.

Este problema de minimización se aborda utilizando el enfoque de descenso por la gradiente (*gradient descent*) y el algoritmo de retro-propagación (*back propagation*), los cuales utilizan la derivada parcial del error con respecto a los pesos ($\frac{\partial E}{\partial wcnn}$) para encontrar los valores óptimos de los pesos $wcnn$.

El fundamento teórico del descenso por la gradiente y la retro-propagación son conceptos básicos en el área de redes neuronales convencionales [1], por lo que no se abordaran en esta sección.

Sin embargo, la gradiente del error ($\frac{\partial E}{\partial wcnn}$) se define de manera distinta a lo expuesto en el descenso por la gradiente por Bishop [1] para las capas de convolución, ya que en las ANN convencionales no utilizan este operador. En la Ecuación 2.22 se define la gradiente del error teniendo en cuenta la capa de convolución l utilizando la regla de la cadena (Gibiansky [8]):

$$\frac{\partial E}{\partial wcnn_{a,b}^{l}} = \sum_{i=0}^{M-m}\sum_{j=0}^{N-n} \frac{\partial E}{\partial xcnn_{i,j}^{l}} \frac{\partial xcnn_{i,j}^{l}}{\partial wcnn_{a,b}^{l}} \tag{2.22}$$

Reemplazando la derivada parcial de la Ecuación 2.17 con respecto a los pesos w se obtiene lo siguiente (Ecuación 2.23):

$$\frac{\partial E}{\partial wcnn_{a,b}^{l}} = \sum_{i=0}^{M-m}\sum_{j=0}^{N-n}\frac{\partial E}{\partial xcnn_{i,j}^{l}} ycnn_{i+a,j+b}^{l-1} \tag{2.23}$$

En la Ecuación 2.23 se puede observar que aparece el término $ycnn^{l-1}$, haciendo referencia a la salida de la capa anterior (retro-propagación). Para continuar con el cálculo de la gradiente del error, se debe despejar el término $\frac{\partial E}{\partial xcnn_{i,j}^{l}}$. En la Ecuación 2.24 se aplica la regla de la cadena con este propósito:

$$\frac{\partial E}{\partial xcnn_{i,j}^{l}} = \frac{\partial E}{\partial ycnn_{i,j}^{l}}\frac{\partial ycnn_{i,j}^{l}}{\partial x_{i,j}^{l}} \tag{2.24}$$

Reemplazando la Ecuación 2.18 se obtiene (Ecuación 2.25):

$$\frac{\partial E}{\partial xcnn_{i,j}^{l}} = \frac{\partial E}{\partial ycnn_{i,j}^{l}}\sigma'\left(xcnn_{i,j}^{l}\right) \tag{2.25}$$

siendo $\sigma'(\cdot)$ la derivada de la función de activación $\sigma(\cdot)$.

En cuanto a la retro-propagación, [8] expresa la gradiente del error con respecto a la salida de la capa anterior como se muestra en la Ecuación 2.26:

$$\frac{\partial E}{\partial ycnn_{i,j}^{l-1}} = \sum_{a=0}^{m-1}\sum_{b=0}^{m-1}\frac{\partial E}{\partial xcnn_{i-a,j-b}^{l}} wcnn_{a,b} \tag{2.26}$$

2.1.3.3 Regresión en cascada basado en árboles de decisión

La regresión en cascada basados en árboles de decisión (ERT, del inglés *Ensemble of Regresion Trees*) es un algoritmo propuesto por Kazemi y Sullivan en [13] y tiene como finalidad la estimación de la ubicación de los puntos de referencia propios de un rostro en una imagen.

La importancia de este algoritmo radica en que la ubicación de los puntos de referencia brinda información referente a la pose e inclinación de un rostro, para luego poder realizar correcciones geométricas sobre el mismo.

A continuación se procede a explicar el fundamento teórico del algoritmo de regresión en cascada basado en árboles de decisión propuesto por [13]. Para abreviar dicha explicación, de aquí en adelante se utilizará el término “regresor” para hacer referencia a un algoritmo de regresión.

El término de regresión en cascada hace referencia a un conjunto regresores dispuestos de manera secuencial, es decir, la salida del primero se convierte en la entrada del segundo y así sucesivamente. Esta estrategia permite realizar regresiones de alta complejidad con un conjunto de regresores de complejidad relativamente baja, lo cual hace que el algoritmo consuma menos recursos computacionales.

Los regresores van a operar sobre la variable $\mathrm{xr_i} \in \mathbb{R}^2$ que contiene las coordenadas x, y del i-ésimo punto de referencia de un rostro en una imagen I. En la Ecuación 2.27 se define el vector de forma:

$$\mathrm{S} = \left(\mathrm{xr}_0^{\mathrm{T}}, \mathrm{xr}_1^{\mathrm{T}}, \dots, \mathrm{xr}_{\mathrm{p}}^{\mathrm{T}}\right)^{\mathrm{T}} \in \mathbb{R}^{2\mathrm{p}} \tag{2.27}$$

El vector de forma contiene todas las coordenadas de los p puntos de referencia de un rostro en la imagen I. Se define a $\hat{\mathrm{S}}^{(t)}$ como la estimación del vector de forma en la posición t de la cascada. Cabe resaltar que la imagen I contiene solo un rostro.

Cada regresor $\mathrm{r}_t(\cdot,\cdot)$ predice un vector de actualización de la estimación del vector de forma tomando como referencia la imagen I y el vector $\hat{\mathrm{S}}^{(t)}$. En la Ecuación 2.28 se muestra la actualización de la estimación del vector de forma en la cascada:

$$\hat{\mathrm{S}}^{(t+1)} = \hat{\mathrm{S}}^{(t)} + \mathrm{r}_t(I, \hat{\mathrm{S}}^{(t)}) \tag{2.28}$$

Previo al entrenamiento de los regresores, se define la data de entrenamiento $(I_1, \mathrm{S}_1), (I_2, \mathrm{S}_2), \dots, (I_n, \mathrm{S}_n)$, donde I_i es una imagen correspondiente a un rostro y S_i su respectivo vector de forma.

Para entrenar el primer regresor de la cascada (r_0), se forman ternas compuestas por la imagen de un rostro, una estimación inicial del vector de forma y un vector de actualización: (I_{π_i}, $\hat{\mathrm{S}}_i^{(0)}$, $\Delta S_i^{(0)}$). Donde:

$$\pi_i \in \{1,2, \dots, n\} \tag{2.29}$$

$$\hat{\mathrm{S}}_i^{(0)} \in \{\mathrm{S}_1, \mathrm{S}_2, \dots, \mathrm{S}_\mathrm{n}\} \backslash \mathrm{S}_{\pi_i} \tag{2.30}$$

$$\Delta \mathrm{S}_i^{(0)} = \mathrm{S}_{\pi_i} - \hat{\mathrm{S}}_i^{(0)} \tag{2.31}$$

para un $i = 1,2, \dots, N$. El número de total de ternas utilizadas es $N = nR$, donde R es el número de inicializaciones por imagen.

Una vez definidas las ternas, se entrena el regresor r_0 utilizando el algoritmo basado en árboles de decisión *gradient tree boosting*, con una optimización basada en la suma de errores cuadráticos (Hastie et al. [12]). Posterior a esto, se actualizan las ternas de entrenamiento a $(I_{\pi_i}, \hat{\mathrm{S}}_i^{(1)}, \Delta \mathrm{S}_i^{(1)})$, para poder entrenar el siguiente regresor (r_1). Generalizando, las ternas se actualizan según las Ecuaciones 2.32 y 2.33:

$$\hat{\mathrm{S}}_i^{(t+1)} = \hat{\mathrm{S}}_i^{(t)} + \mathrm{r}_t(I_{\pi_i}, \hat{\mathrm{S}}_i^{(t)}) \tag{2.32}$$

$$\Delta \mathrm{S}_i^{(t+1)} = \mathrm{S}_{\pi_i} - \hat{\mathrm{S}}_i^{(t+1)} \tag{2.33}$$

para un $t = 0,1, , \dots, T_c - 1$, siendo T_c el número total de regresores en la cascada.

El algoritmo *gradient tree boosting* utiliza un factor de aprendizaje ν para evitar el sobre entrenamiento. Es recomendable que este parámetro, conocido también como factor de merma, se encuentre en el rango de $0 < \nu_{ert} < 1$ [12].

En la Figura 2.18 se muestra el pseudocódigo del algoritmo de aprendizaje de cada regresor r_t. Como se mencionó anteriormente, cada capa de la cascada está compuesta por uno de estos regresores.

Entrada: Ternas de entrenamiento $(I_{\pi_i}, \hat{\mathrm{S}}_i^{(t)}, \Delta\mathrm{S}_i^{(t)})$
Parámetro: Factor de aprendizaje ν_{ert}
Salida: Regresor $r_t(I, \hat{\mathrm{S}}_i^{(t)})$

1: Inicializar: $f_0\left(I, \hat{\mathrm{S}}_i^{(t)}\right) = \underset{\gamma_{ert}\in\mathbb{R}^{2\mathrm{p}}}{\arg\min} \sum_{i=1}^{N} \left\| \Delta\mathrm{S}_i^{(t)} - \gamma_{ert} \right\|^2$
2: **Para** $k = 1, \ldots., K$:
3: **Definir para** $i = 1, \ldots., N$:
4: $r_{ik} = \Delta\hat{\mathrm{S}}_i^{(t)} - f_{k-1}\left(I, \hat{\mathrm{S}}_i^{(t)}\right)$
5: **Fin del para (*i*)**
6: Entrenar una función de regresión débil $g_k(I, \hat{\mathrm{S}}_i^{(t)})$ utilizando r_{ik}
7: Actualizar: $f_k\left(I, \hat{\mathrm{S}}_i^{(t)}\right) = f_{k-1}\left(I, \hat{\mathrm{S}}_i^{(t)}\right) + \nu g_{k-1}\left(I, \hat{\mathrm{S}}_i^{(t)}\right)$
8: **De del Para (*k*)**
9: $r_t\left(I, \hat{\mathrm{S}}_i^{(t)}\right) = f_K\left(I, \hat{\mathrm{S}}_i^{(t)}\right)$

Figura 2.18. Pseudocódigo del algoritmo de entrenamiento del regresor r_t [12].

donde γ_{ert} es la estimación de $\Delta\mathrm{S}_i^{(t)}$ por parte del regresor r_t.

2.1.4 Corrección geométrica

Las correcciones geométricas, en procesamiento de imágenes, consisten en una serie de transformadas que se realiza con respecto a las coordenadas de un pixel. En el realce de contraste, se pudo apreciar que las modificaciones que todos los cambios que se realizaron fueron con respecto al valor numérico de los pixeles. Sin embargo, en las trasformaciones geométricas se busca modifcar la posición de un pixel en la imagen. A continuación se procederá a explicar las técnicas que se utilizarán para realizar las correcciones geométricas: la transformación afín. En el presente trabajo de investigación, estas correcciones se utilizan para estandarizar la orientación y/o pose de los rostros antes de entrar a la etapa de codificación, con la finalidad de tener un resultado más constante.

2.1.4.1 Transformaciones afín

La transformación afín, en el área de procesamiento digital de imágenes, es aquella transformada geométrica que permite realizar un mapeo lineal de las coordenadas espaciales de un pixel a un nuevo sistema de coordenadas (Solomon y Breckon [24]).

La transformación afín, en su forma más general, puede incluir a las transformaciones geométricas (lineales) de traslación, rotación, escalamiento, estiramiento y cizallamiento.

En la Ecuación 2.34 se expresa de modo general una transformación afín (en el plano 2D) de las coordenadas cartesianas (i_a,j_a):

$$\begin{bmatrix} i'_a \\ j'_a \end{bmatrix} = \begin{bmatrix} a_{af} & b_{af} \\ d_{af} & e_{af} \end{bmatrix} \begin{bmatrix} i_a \\ j_a \end{bmatrix} + \begin{bmatrix} c_{af} \\ f_{af} \end{bmatrix} \tag{2.34}$$

Que también puede ser expresada de manera matricial (Ecuación 2.35):

$$\mathrm{x'_{af}} = \mathrm{T'_{af}} \cdot \mathrm{x} + \mathrm{d_{af}} \tag{2.35}$$

Sin embargo, esta expresión puede continuar simplificándose al representar a la transformada afín en función a coordenadas homogéneas (Ecuación 2.36):

$$\mathrm{T_{af}} = \begin{bmatrix} \alpha_{11} & \alpha_{12} & \alpha_{13} \\ \alpha_{21} & \alpha_{22} & \alpha_{23} \\ 0 & 0 & 1 \end{bmatrix} \tag{2.36}$$

En donde los parámetros α_{11}, α_{12}, α_{21}, α_{22}, α_{13} y α_{23} corresponden a a_{af}, b_{af}, d_{af}, e_{af}, c_{af} y f_{af} de la Ecuación 2.34 respectivamente.

La ventaja de expresar a la transformación afín en coordenadas homogéneas es que el efecto de 2 transformaciones afines consecutivas (con matrices de transformación $\mathrm{T_{af1}}$ y $\mathrm{T_{af2}}$ respectivamente) se puede simplificar por una transformada afín (de matriz de transformación $\mathrm{T_{af3}}$), cumpliéndose que $\mathrm{T_{af3}} = \mathrm{T_{af1}} \cdot \mathrm{T_{af2}}$ y $\mathrm{T_{af3}}$ conserva la misma forma que la matriz definida en la Ecuación 2.36, es decir, que solo depende de 6 parámetros.

En la Tabla 2.4 se resumen las características de los parámetros de la transformación afín $\mathrm{T_{af}}$ para generar una transformación geométrica.

Tabla 2.4. Valor de los coeficientes de la transformación afín para generar una transformación geométrica lineal [24].

Transformación	α_{11}	α_{12}	α_{13}	α_{21}	α_{22}	α_{23}
Traslación por (x_{af}, y_{af})	1	0	x_{af}	0	1	y_{af}
Rotación θ_{rot}	$\cos\theta_{\mathrm{rot}}$	$\sin\theta_{\mathrm{rot}}$	0	$\sin\theta_{\mathrm{rot}}$	$\cos\theta_{\mathrm{rot}}$	0
Escalamiento por s_{af}	s_{af}	0	0	0	s_{af}	0
Cizallamiento X por s_{af}	1	s_{af}	0	0	1	0
Cizallamiento Y por s_{af}	s_{af}	0	0	s_{af}	1	0

Además, en la Figura 2.19 se muestra de manera gráfica los efectos de estas transformaciones geométricas lineales en coordenadas cartesianas, donde las coordenadas de la imagen se trasladan de tal manera que las coordenadas (0,0) coincidan con su centro y no con la esquina superior izquierda como se ha definido en la Ecuación 2.2.

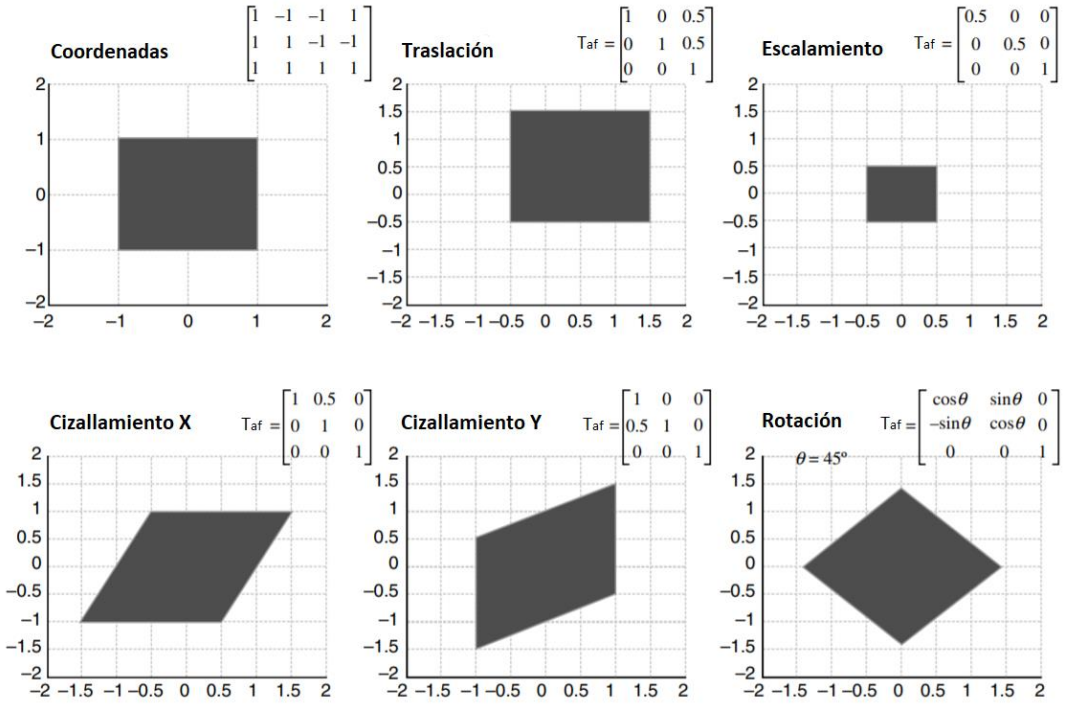


Figura 2.19. Transformaciones geométricas lineales básicas. Traducido de [24].

2.1.5 Algoritmos de clasificación

Los algoritmos de clasificación pertenecen al área del aprendizaje automatizado y se utilizan para agrupar patrones y/o características (por lo general vectores) en subconjuntos que reciben el nombre de clases. A continuación se procederá a explicar el algoritmo de clasificación de Máquinas de Soporte Vectorial.

2.1.5.1 Máquinas de soporte vectorial

Las Máquinas de Soporte Vectorial (SVM, del inglés *Support Vector Machine*) es un algoritmo supervisado del campo de aprendizaje automático (*machine learning*) utilizado para resolver problemas de clasificación y regresión. Sin embargo, para el presente trabajo, se utilizará el acrónimo SVM para referirse únicamente a las máquinas de soporte vectorial aplicadas a la clasificación.

La principal fortaleza de un SVM (y lo que la hizo tan popular en el área de detección de objetos) radica en que los parámetros del modelo se calculan mediante la solución de un problema de optimización convexa, por lo que su solución local corresponde a un óptimo global [1]. En otras palabras, el entrenamiento de una SVM encuentra la solución óptima para un conjunto de datos de entrenamiento determinado.

Un SVM es un algoritmo de clasificación que busca separar diferentes clases en un espacio de características por medio de hiperplanos, siendo un hiperplano la extensión del concepto de un plano: una función lineal de "n" dimensiones. Por esta razón, la SVM puede resolver problemas de clasificación siempre y cuando las características o variables que definen cada clase sean linealmente separables. En la Figura 2.20 se muestra un ejemplo de un conjunto de datos que contiene dos clases separadas por un hiperplano.

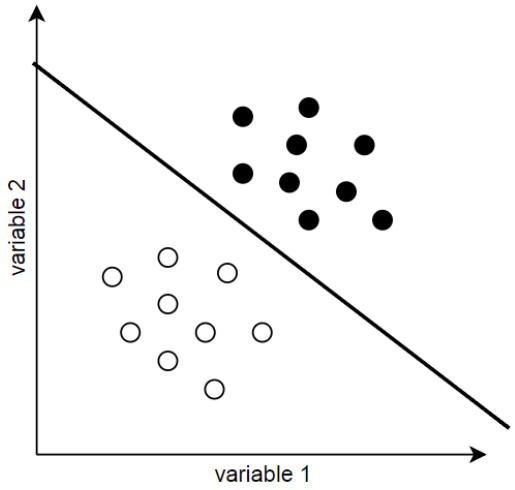


Figura 2.20. Ejemplo de un hiperplano que separa 2 clases.

Los hiperplanos que separan a dos clases distintas reciben el nombre de frontera de decisión. En un SVM, la frontera de decisión se determina utilizando la información de un sub-conjunto de elementos del conjunto de datos denominados vectores de soporte. Los vectores de soporte son elegidos con la finalidad de maximizar el margen entre clases

distintas, siendo el margen definido como la distancia entre los hiperplanos que contienen a los vectores de soporte de una clase determinada [1].

En la Figura 2.21 se resaltan los vectores de soporte del conjunto de datos mostrado en la Figura 2.20, dibujándose además los hiperplanos que contienen dichos vectores de soporte con líneas punteadas.

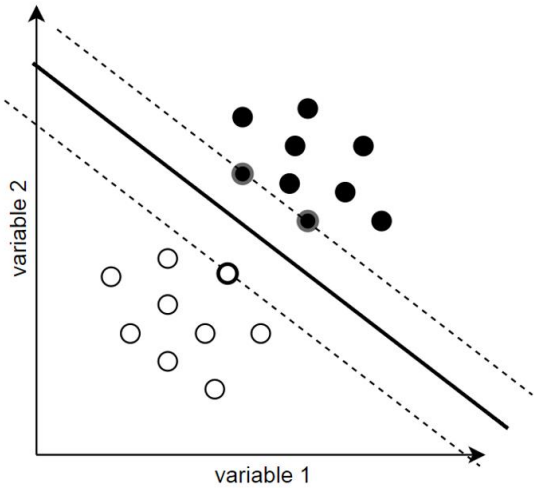


Figura 2.21. Vectores de soporte resaltados e hiperplanos que los contienen del conjunto de datos de la Figura 2.20

En la Figura 2.21, el margen entre ambas clases vendría a ser la distancia entre las líneas punteadas (hiperplanos de soporte). Para realizar el cálculo de dicho margen es necesario elegir qué elementos del conjunto de datos son los más apropiados para cumplir el rol de vectores de soporte.

Para explicar el fundamento teórico de un SVM y establecer los criterios de selección para los vectores de soporte, Burges [3] comienza definiendo el conjunto de datos a utilizar de la siguiente manera (Ecuación 2.37):

$$(x_1, y_1), \ldots, (x_n, y_n) \tag{2.37}$$

donde x_i son las características del i-ésimo punto y $y_i \in \{-1,1\}$ su correspondiente etiqueta.

En un inicio, [3] parte del supuesto que las dos clases son separables (como los datos mostrados en la Figura 2.20). Por lo tanto, existe un hiperplano que separa ambas clases y los puntos que lo conforman satisfacen la siguiente relación (Ecuación 2.38):

$$\mathbf{w}^{\mathrm{T}}\mathbf{x} + \mathrm{b} = 0 \tag{2.38}$$

siendo $\mathbf{x}$ las características de dichos puntos, $\mathbf{w}$ un vector de coeficientes y b una variable de compensación (bias).

Además, los elementos del conjunto de datos separados por este hiperplano (Ecuación 2.38) deberán cumplir con las siguientes condiciones (Ecuaciones 2.39 y 2.40):

$$\mathbf{w}^{\mathrm{T}}\mathbf{x}_{\mathrm{i}} + \mathrm{b} \geq 1, \qquad \text{para } \mathrm{y}_{\mathrm{i}} = +1 \tag{2.39}$$

$$\mathbf{w}^{\mathrm{T}}\mathbf{x}_{\mathrm{i}} + \mathrm{b} \leq -1, \qquad \text{para } \mathrm{y}_{\mathrm{i}} = -1 \tag{2.40}$$

O, de manera general, cumplir con (Ecuación 2.41):

$$\mathrm{y}_{\mathrm{i}}(\mathbf{w}^{\mathrm{T}}\mathbf{x}_{\mathrm{i}} + \mathrm{b}) - 1 \geq 0 \tag{2.41}$$

Por otro lado, los hiperplanos de soporte anteriormente mencionados son aquellos que cumplen con la condición de frontera mostrada en la Ecuación 2.41. Es decir, los puntos que contienen estos hiperplanos satisfacen una de las siguientes ecuaciones: $\mathbf{w}^{\mathrm{T}}\mathbf{x} + \mathrm{b} = 1$ o $\mathbf{w}^{\mathrm{T}}\mathbf{x} + \mathrm{b} = -1$. En la Figura 2.22 se muestra cómo estos conceptos se aplican al ejemplo mostrado en la Figura 2.21.

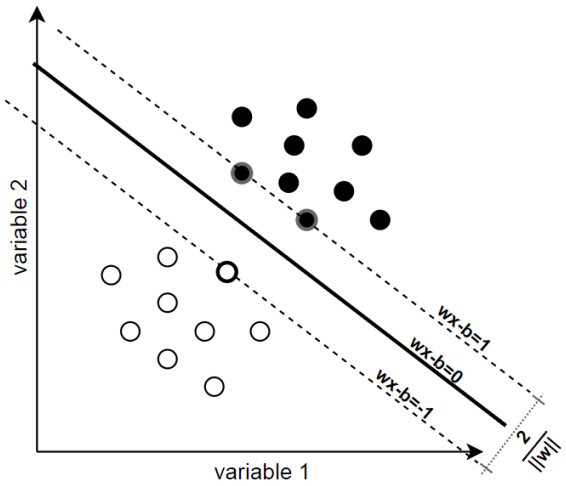


Figura 2.22. Vectores de soporte resaltados e hiperplanos que los contienen del conjunto de datos de la Figura 2.20

Cabe resaltar que debido a que el conjunto de datos es separable, no existen elementos que se encuentren entre los hiperplanos de soporte. Además, en la Figura 2.22 se puede apreciar que la distancia entre los hiperplanos de soporte (margen) tiene un valor constante de $2/\|\mathbf{w}\|$ (siendo $\|\mathbf{w}\|$ el modulo o norma L_2 de $\mathbf{w}$). El valor del margen se calcula utilizando conceptos básicos de geometría analítica (distancia de un hiperplano al origen de coordenadas) y es explicado por [3].

Una vez definido el margen, y teniendo en cuenta que el SVM busca maximizar dicho valor, el problema original de un SVM se puede re-interpretar como un problema de minimización en función de $\mathbf{w}$. El nuevo problema de minimización se plantea de la siguiente manera (Ecuación 2.42):

$$\min_{w,b} \frac{1}{2}\|\mathbf{w}\|^2 \tag{2.42}$$

sujeto a la condición mostrada en la Ecuación 2.41.

El argumento de la función de minimización ($\|\mathbf{w}\|^2$) puede ser reemplazado por $\mathbf{w}^{\mathrm{T}}\mathbf{w}$. Con esto, el problema planteado adquiere la forma (Ecuación 2.43):

$$\min_{w,b} \frac{\mathbf{w}^{\mathrm{T}}\mathbf{w}}{2} \tag{2.43}$$

sujeto a la condición mostrada en la Ecuación 2.41.

La razón de realizar este cambio de forma en el problema de optimización es que ahora corresponde a un problema de minimización convexa debido a la función cuadrática. Para encontrar la solución a este problema se utilizará el procedimiento de los multiplicadores de Lagrange, obteniéndose así la función de Lagrange que se muestra en la Ecuación 2.44 [3]:

$$\mathrm{L}(\mathbf{w}, \mathrm{b}, \alpha,) = \frac{\mathbf{w}^{\mathrm{T}}\mathbf{w}}{2} - \sum_i \alpha_{\mathrm{i}} \mathrm{y}_{\mathrm{i}} (\boldsymbol{w}^T \mathbf{x}_{\mathrm{i}} + b) + \sum_i \alpha_{\mathrm{i}} \tag{2.44}$$

siendo α_{i} los multiplicadores de Lagrange.

Ahora, se busca la minimización de la función de Lagrange con respecto a los parámetros w y b. Sin embargo, [3] explica que también se debe realizar una maximización de la función $L(w, b, \alpha,)$ sujeto a que los multiplicadores de lagrange sean números no negativos ($\alpha_i \geq 0$).

Para esto, primero se minimizará la función $L(\mathbf{w}, b, \alpha,)$ con respecto a las variables w y b. Esto se logra calculando las derivadas parciales de la función de Lagrange con respecto a las variables mencionadas (Ecuaciones 2.45 y 2.46):

$$\frac{\delta L}{\delta \mathbf{w}} = \mathbf{w} - \sum_i \alpha_i y_i \mathbf{x}_i \tag{2.46}$$

$$\frac{\delta L}{\delta b} = -\sum_i \alpha_i y_i \tag{2.46}$$

E Igualando estas derivadas parciales a cero (Ecuaciones 2.47 y 2.48):

$$\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i \tag{2.47}$$

$$\sum_i \alpha_i y_i = 0 \tag{2.48}$$

sujeto a $\alpha_i \geq 0$.

Es importante mencionar que, como se aprecia en la Ecuación 2.39, el vector w es el resultado de la combinación de información de las muestras del conjunto de datos que posean un multiplicador de Lagrange mayor a cero ($\alpha_i > 0$). Justamente, son estas muestras las que reciben el nombre de vectores de soporte, mientras que a las demás muestras del conjunto de datos les corresponde un multiplicador de Lagrange igual a cero ($\alpha_i = 0$).

Reemplazando la expresión mostrada en la Ecuación 2.47 en la función de Lagrange (Ecuación 2.44) se obtiene una nueva formulación para $L(\mathbf{w}, b, \alpha)$ (Ecuación 2.49):

$$L(\mathbf{w}, b, \alpha) = \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j \mathbf{x}_i \mathbf{x}_j \tag{2.49}$$

Como se mencionó anteriormente, la solución que propone la SVM se genera maximizando la nueva función de Lagrange con respecto a los multiplicadores de Lagrange. Lo antes mencionado se muestra en la Ecuación 2.50:

$$\max_{\alpha} \sum_{i} \alpha_i - \frac{1}{2} \sum_{i} \sum_{j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \mathbf{x}_j \tag{2.50}$$

sujeto a la Ecuación 2.48 y a $\alpha_i \geq 0$

A esta expresión se le conoce como formulación dual del SVM, ya que corresponde a una minimización de la función de Lagrange con respecto a w y b, pero también a una maximización de la misma con respecto a α.

Esta formulación dual del SVM se plantea partiendo del supuesto que el conjunto de datos a clasificar es linealmente separable. Sin embargo, en problemas reales, generalmente los conjuntos de datos no cumplen ni con la condición de que exista una función lineal que pueda diferenciar las clases ni con la condición de que estas sean completamente separables. Por estas razones, el SVM plantea las siguientes estrategias: el truco del *kernel* (*kernel trick*) y el criterio del margen suave (*soft margin*). El truco del *kernel* aborda el problema de la no linealidad mientras que el criterio del margen suave aborda el problema de la no-separabilidad del conjunto de datos [3].

Burges [3] explica que el truco del *kernel*, o formulación del *kernel*, es un recurso que se utiliza cuando las clases en el conjunto de datos no son linealmente separables. Esto quiere decir que no existe hiperplano que cumpla con la condición en la Ecuación 2.41. En la Figura 2.23 se muestra un ejemplo de clases que no son linealmente separables y, por lo tanto, no cumplen con los requisitos para poder ser clasificadas mediante un SVM.

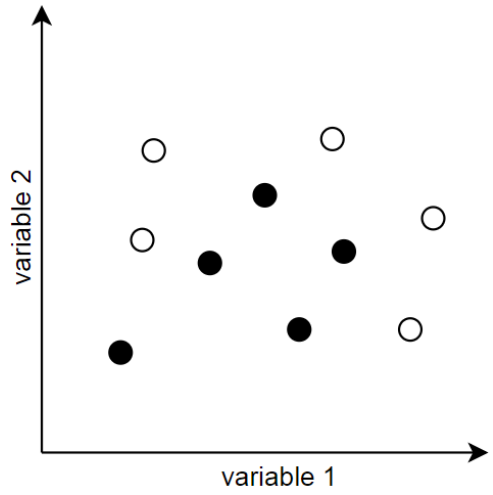


Figura 2.23. Ejemplo de un conjunto de datos linealmente no separables.

Para aplicar el truco del *kernel* primero se debe mapear a los elementos del conjunto de datos a un nuevo espacio de características de mayor dimensionalidad (con más características) que tal vez pueda hacer a las clases linealmente separables. Burges [3] define a la función de mapeo como $\varphi(\cdot)\colon \mathbf{x} \in \mathrm{R}^{\mathrm{a}} \mapsto \varphi(\mathrm{x}) \in \mathrm{R}^{\mathrm{b}}$, siendo $\mathrm{a} < \mathrm{b}$.

Reemplazando la función de mapeo en la formulación dual del SVM (Ecuación 2.50) se obtiene una nueva formulación dual del SVM (Ecuación 2.51):

$$\max_{\alpha} \sum_{i} \alpha_{\mathrm{i}} - \frac{1}{2} \sum_{i} \sum_{j} \alpha_{\mathrm{i}} \alpha_{\mathrm{j}} \mathrm{y}_{\mathrm{i}} \mathrm{y}_{\mathrm{j}} \varphi(\mathbf{x}_{\mathrm{i}})^{\mathrm{T}} \varphi(\mathbf{x}_{\mathrm{j}}) \tag{2.51}$$

Sin embargo, la dimensionalidad del nuevo espacio de características puede tener valores muy altos (hasta infinitos) y realizar el cálculo de $\varphi(\mathbf{x}_{\mathrm{i}})^{\mathrm{T}} \varphi(\mathbf{x}_{\mathrm{j}})$ en el entrenamiento del SVM podría ser inviable. En respuesta a esto, [3] plantea que se utilice un tipo de función especial que sintetice y/o aproxime el cálculo de $\varphi(\mathbf{x}_{\mathrm{i}})^{\mathrm{T}} \varphi(\mathbf{x}_{\mathrm{j}})$ en el proceso de entrenamiento de una SVM. A este tipo de función especial se le conoce como función *kernel* $\mathrm{K}(\cdot,\cdot)$y cumple con la condición de Mercer (Vapnik [29]) que se muestran en la Ecuación 2.52:

$$\mathrm{K}(\mathbf{x}, \mathbf{z}) = \varphi(\mathbf{x})^{\mathrm{T}} \varphi(\mathbf{z}) \tag{2.52}$$

La función *kernel* permite entrenar el SVM sin definir de manera explícita la función de mapeo $\varphi(\cdot)$ [3].

El truco del *kernel* consiste en reemplazar la Ecuación 2.52 en la nueva formulación del problema dual del SVM (Ecuación 2.51) para poder hacer que el entrenamiento del SVM sea viable. La nueva formulación con *kernel* del SVM se muestra en la Ecuación 2.45, tras reemplazar la Ecuación 2.51 en la formulación del problema dual descrita en la Ecuación 2.42

Burges [3], al reemplazar el mapeo (φ) de los elementos x_i y x_j en la formulación dual del SVM (Ecuación 2.50) y teniendo en cuenta la condición de Mercer (Ecuación 2.51), genera una nueva formulación del problema de optimización del SVM (Ecuación 2.53):

$$\max_{\alpha} \sum_{i} \alpha_i - \frac{1}{2} \sum_{i} \sum_{j} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i \mathbf{x}_j) \tag{2.53}$$

La Figura 2.24 muestra de manera gráfica cómo la función *kernel* volvería a los datos de la Figura 2.23 linealmente separables.

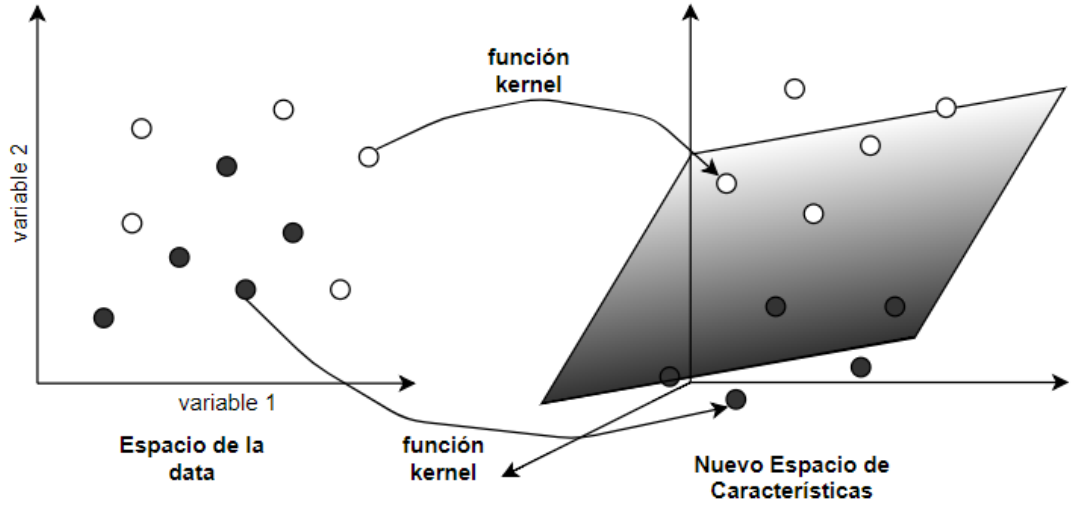


Figura 2.24. Ejemplo gráfico de cómo funciona el truco del *kernel*.

En la Tabla 2.5 se muestran algunas de las funciones *kernel* más utilizadas en el área de detección de patrones:

Tabla 2.5. Tipos de función kernel [3].

TIPO	FÓRMULA
Polinomial	$\mathrm{K}(\mathbf{x},\mathbf{z}) = \langle \mathbf{x},\mathbf{z} \rangle^{\mathrm{d}}$
Base Radial	$\mathrm{K}(\mathbf{x},\mathbf{z}) = \mathrm{e}^{-\|\mathbf{x}-\mathbf{z}\|^2/(2\sigma_{\mathrm{k}})}$
Sigmoide	$\mathrm{K}(\mathbf{x},\mathbf{z}) = \tanh(\kappa \mathbf{x} \cdot \mathbf{z} - \delta)$

Cabe resaltar que, según se muestra en la Tabla 2.5, la función kernel añade nuevos parámetros en la formulación del problema de optimización. Al igual que con los multiplicadores de Lagrange, estos parámetros se ajustan durante el proceso de entrenamiento del SVM.

Por otro lado, [3] explica que el criterio del margen suave es un recurso que se utiliza cuando las clases del conjunto de datos no son totalmente separables. Esto quiere decir que no existe un hiperplano que separe sin errores a los elementos de distintas clases (condición 2.41). En la Figura 2.25 se muestra una variación del conjunto de datos mostrados en la Figura 2.21, donde se puede apreciar que ahora aparecen elementos entre los hiperplanos de soporte.

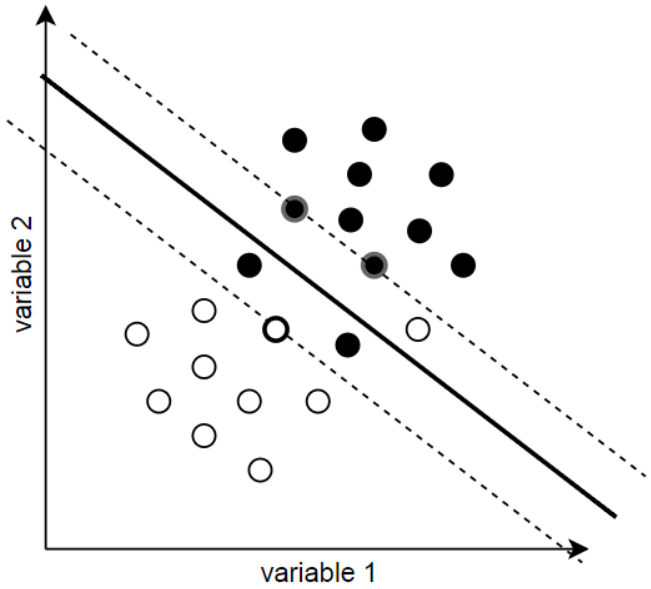


Figura 2.25. Ejemplo de un conjunto de datos no separables.

El criterio del margen suave se basa en utilizar variables de holgura (*slack variables*) que permitan que la condición de la Ecuación 2.33 sea más flexible en aceptar cierto

margen de error en la clasificación [3]. Las variables de holgura se definen en la Ecuación 2.54:

$$\xi_i = \max(0,1 - y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_{\mathrm{i}} + b)) \tag{2.54}$$

siendo ξ_i la variable de holgura correspondiente al i-ésimo elemento

En la Ecuación 2.54 se puede apreciar que las variables de holgura siempre tienen un valor no negativo ($\xi_i \geq 0$). Burges [3] introduce estas variables de holgura en el planteamiento inicial del SVM (Ecuación 2.35), siendo su influencia regulada por un parámetro C que adquiere el nombre de coeficiente de margen suave (*soft margin*). En la Ecuación 2.55 se muestra la función de optimización descrita en la Ecuación 2.43 modificada para aceptar variables de holgura.

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi_i \tag{2.55}$$

sujeto a $\mathrm{y_i}(\mathbf{w}^{\mathrm{T}}\mathbf{x}_{\mathrm{i}} + \mathrm{b}) \geq 1 - \xi_i$

Cabe resaltar que la aparición de las variables de holgura no modifica la expresión del problema dual del SVM (Ecuación 2.50). Sin embargo, restringe los valores de los multiplicadores de Lagrange.

Además, juntando las estrategias del truco del *kernel* y el criterio del margen suave en la formulación del problema dual del SVM se obtiene (Ecuación 2.56):

$$\max_{\alpha} -\frac{1}{2}\sum_i\sum_j \alpha_{\mathrm{i}}\alpha_{\mathrm{j}}\mathrm{y_i}\mathrm{y_j}\mathrm{K}(\mathbf{x}_{\mathrm{i}}\mathbf{x}_{\mathrm{j}}) + \sum_i \alpha_{\mathrm{i}} \tag{2.56}$$

sujeto a $\sum_i \alpha_{\mathrm{i}}\mathrm{y_i} = 0$ y $0 \leq \alpha_{\mathrm{i}} \leq C$

Con esto, se obtiene un algoritmo de clasificación capaz de resolver problemas de clasificación no lineales con datos reales (no necesariamente separables a la perfección). El rendimiento de dicho algoritmo, para un problema determinado, depende de la función kernel escogida y del coeficiente de margen suave.

Finalmente, el resultado de la clasificación utilizando SVM se calcula utilizando la Ecuación 2.57:

$$f_{SVM}(\mathbf{x}) = \mathrm{sgn}\left(\sum_i \alpha_i y_i K(\mathbf{x}_i \mathbf{x}_j) + b\right) \tag{2.57}$$

2.2 El marco conceptual

A continuación se procederán a definir los conceptos importantes para asegurar una mejor comprensión del presente trabajo. Los conceptos definidos a continuación pertenecen a las áreas de procesamiento de imágenes, aprendizaje automatizado y aprendizaje profundo.

2.2.1 Entorno no controlado

En el área de procesamiento de imágenes se define como entorno no controlado a aquel ambiente en el cual no se interviene, ya sea de manera directa o indirecta, sobre las potenciales fuentes de ruido externas a la cámara (en el caso particular del presente trabajo, iluminación).

En el presente trabajo, el término de entorno no controlado se utiliza para hacer hincapié en que las condiciones de iluminación de la cabina del conductor añaden distintos tipos y niveles de ruido en las imágenes adquiridas.

2.2.2 Generalización de un algoritmo

En el área de aprendizaje automatizado, el término generalización de un algoritmo se utiliza cuando un modelo ha sincronizado correctamente sus parámetros para resolver un problema determinado. Dicho de otra manera, un modelo de aprendizaje automatizado que generaliza es aquel cuyo rendimiento se mantiene (ya sea en clasificación o regresión) aun cuando es evaluado con datos distintos a los que se utilizaron para entrenar dicho algoritmo. Por esta razón, al momento de entrenar un algoritmo de aprendizaje automatizado se divide la base de datos en data de entrenamiento y data de prueba, con la finalidad de evaluar la generalización del algoritmo.

2.2.3 Sobreajuste

El sobreajuste (en inglés, *overfitting*) es un fenómeno en el rendimiento de los algoritmos de aprendizaje automatizado. Este fenómeno consiste en que el modelo implementado, con un error de entrenamiento bajo, presenta altas tasas de error cuando se evalúan nuevos datos.

Generalmente, este fenómeno ocurre cuando la complejidad del modelo matemático utilizado es mucho mayor a la complejidad requerida para resolver un problema. Sin embargo, cabe resaltar que mientras mayor sea la complejidad del modelo matemático, este puede resolver problemas más complejos. Por esta razón, cuando se entrena un algoritmo de aprendizaje automático, el grado de complejidad asignado al modelo matemático es un parámetro importante que debe ser sintonizado de manera que sea lo suficientemente complejo para resolver un problema pero que además evite el sobreajuste.

En la Figura 2.26 se muestra un ejemplo de un modelo matemático que posee sobreajuste y un modelo matemático que generaliza de manera satisfactoria un problema de regresión.

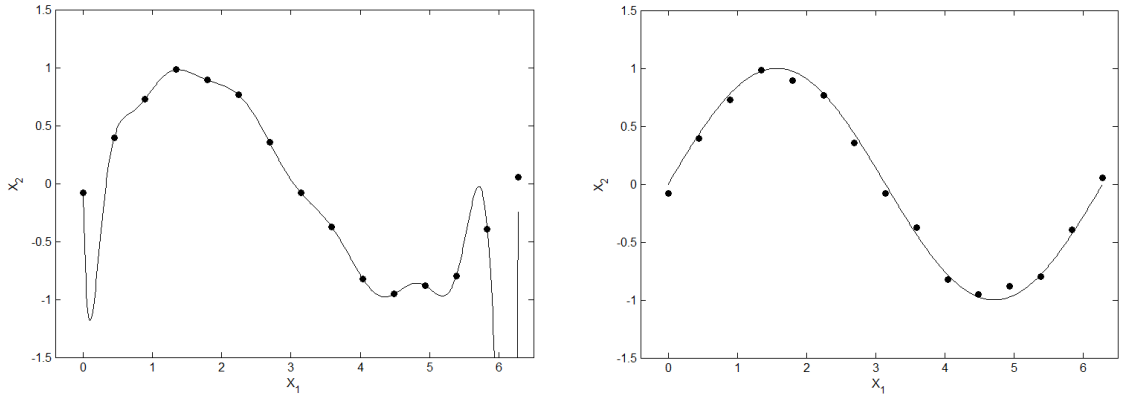


Figura 2.26. Algoritmo de regresión (a) con sobreajuste y (b) que alcanzó la generalización.

2.2.4 Transferencia de conocimiento

La transferencia de conocimiento (en inglés, *transfer learning*) es un recurso utilizado en el área del aprendizaje profundo y aplicado principalmente a las redes neuronales convolucionales. Este concepto consiste en entrenar un modelo matemático que ya ha sido entrenado previamente. Sin embargo, este recurso se utiliza únicamente cuando el problema que se busca resolver y el problema para el cual fue entrenado el modelo matemático son similares.

El proceso de entrenamiento al aplicar transferencia de conocimiento difiere del modo convencional de entrenamiento en un punto: no se sintonizan todos los pesos de las capas de procesamiento. La finalidad de no modificar los pesos de las capas de procesamiento durante el entrenamiento es la de conservar las representaciones de bajo nivel (como

formas, siluetas o texturas) que han sido aprendidas por el modelo. Esto permite aligerar el proceso de entrenamiento, ya que la mayoría de la carga computacional de éste consiste en sintonizar los parámetros que permitan que el modelo aprenda las representaciones de bajo nivel.

Generalmente, para una red neuronal convolucional, el proceso de entrenamiento utilizando transferencia de conocimiento consiste en entrenar modificando únicamente las capas de conectividad total y conservando los pesos de las capas de convolución. En la Figura 2.27 se muestra este tipo de entrenamiento de manera gráfica.

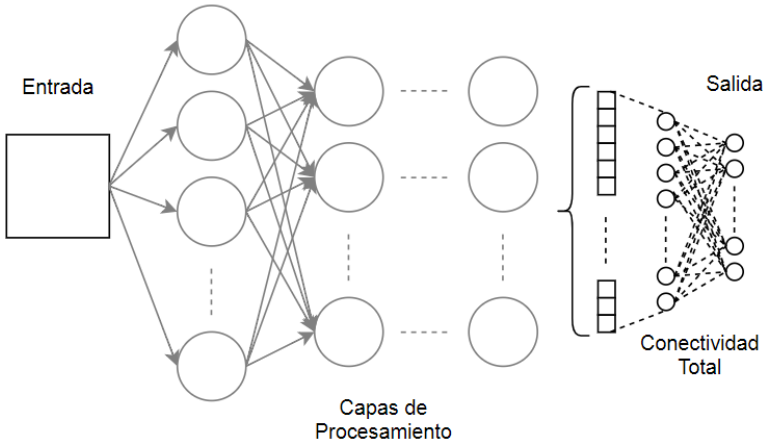


Figura 2.27. Representación gráfica de las capas a entrenarse (en color negro) en una red neuronal convolucional utilizando transferencia de conocimiento.

2.3 Comentarios finales del capítulo

En el presente capítulo se expuso el fundamento teórico de todas las técnicas que darán soporte a la metodología y al algoritmo propuesto por el presente trabajo de Investigación. Además, se realizó las definiciones de los términos clave para una completa comprensión de la metodología. Cabe resaltar que, en la sección de implementación del algoritmo propuesto del siguiente capítulo, se estará realizando una evocación constante a las ecuaciones descritas en este capítulo, asignando además valores numéricos a los parámetros de las técnicas utilizadas en la metodología propuesta.

CAPÍTULO III
DESARROLLO DEL ALGORITMO PROPUESTO

En el presente capítulo, se explican los aspectos relevantes en cuanto al desarrollo de la propuesta de la tesis.

3.1 Diseño de investigación

El flujo de trabajo propuesto se muestra de manera resumida en el diagrama de bloques de la Figura 3.1.

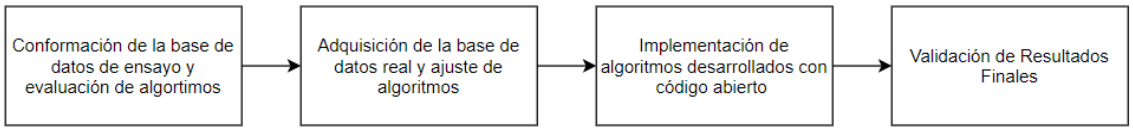


Figura 3.1. Diagrama de flujo de trabajo.

A continuación se procede a explicar de manera más detallada en qué consisten cada uno de los bloques del diagrama del flujo de trabajo.

3.1.1. Conformación de la base de datos de ensayo y evaluación de algoritmos

Se establece las siguientes sub-etapas:

- *Conformación de la base de datos de ensayo*: Se considera a este tipo de base de datos como el conjunto de imágenes en los cuales se realizarán las pruebas de detección y reconocimiento de rostros en general.

Estas imágenes no corresponden necesariamente a conductores de vehículos terrestres ni poseen necesariamente el ruido que podría captar una cámara en la cabina de conducción.

- *Evaluación de algoritmos*: Esta etapa consiste en implementar diversos tipos de algoritmos de pre-procesamiento, segmentación, extracción de características y reconocimiento de rostro; y aplicarlos en la base de datos de ensayo.

La implementación de estos algoritmos se realizará en el software MatLab®. Además se procederá a sintonizar los parámetros más adecuados de cada algoritmo, para lo cual se les aplicaran métodos experimentales de evaluación parcial.

3.1.2. Adquisición de la base de datos reales y ajuste de algoritmos

Se establece las siguientes sub-etapas:

- Generación de la base de datos real: se considera a esta base de datos como el conjunto de imágenes reales de conductores, las cuales serán adquiridas por medio de un sensor óptico ubicado en la cabina del conductor de un automóvil y una laptop para realizar la adquisición.

En esta etapa se busca definir la posición más adecuada para la instalación de la cámara en la cabina del conductor. A fin de prever los distintos tipos de escenarios, estas imágenes serán adquiridas a diferentes horas del día y con diferentes condiciones de tráfico e iluminación.

- *Ajuste de algoritmos*: En esta etapa se evaluará el desempeño que poseen los algoritmos (previamente evaluados en la etapa de ensayo) en la base de datos real con la finalidad de evaluar, según su comportamiento, si estos requieren un ajuste de parámetros o si requieren ser cambiados por otro tipo de algoritmo. Del mismo modo, se procederá a evaluar las limitaciones que poseen las imágenes de la base de datos real. A estos algoritmos también se les aplicará métodos experimentales de evaluación parcial.

3.1.3. Implementación de algoritmos desarrollados con código abierto

Esta etapa consiste en migrar el código de los algoritmos implementados en MatLab® a un lenguaje de programación de más bajo nivel para mejorar el tiempo de procesamiento. Además, dicho lenguaje será de uso libre (sin licencia) para que no exista restricción en cuanto a la aplicación comercial del algoritmo.

Se eligió que el lenguaje de programación al cual se migrará el código de los algoritmos desarrollados sea Python. Este último, además de poseer una licencia de código abierto, es un lenguaje de programación multiparadigma, es decir, puede ser utilizado como un lenguaje orientado a objetos, a programación imperativa y/o programación funcional.

Además, se utilizará la librerías de acceso libre: OpenCv para procesamiento de imágenes/visión computacional; sklearn para implementación de algoritmos de aprendizaje automatizado; dlib y TensorFlow para algoritmos de aprendizaje profundo. Cabe resaltar que las librerías anteriormente mencionadas son compatibles con Python.

3.1.4. Validación de resultados finales

La validación se llevará a cabo con imágenes que no han sido utilizadas en el entrenamiento del algoritmo de clasificación, pero el resultado de esta tendrá repercusiones en el conjunto de algoritmos que conforman el método propuesto en la tesis.

Los resultados de la detección de rostro se evaluarán mediante un análisis de sensibilidad y especificidad, siendo los niveles mínimos aceptados del 70% y 90% respectivamente. Cabe aclarar que el algoritmo propuesto será aplicado para realizar un monitoreo constante de un conductor, por lo cual se define como prioridad que éste no detecte rostros donde no los hay (alta especificidad). Por otro lado, al realizar un monitoreo constante del conductor, con un 70% de sensibilidad aseguras la detección de un rostro en 3 imágenes con 97.3% de éxito.

Por otro lado, los resultados de la identificación de rostro se evaluarán mediante un análisis de asertividad y AUC, siendo los niveles mínimos aceptados del 90% y un valor de 0.85 respectivamente.

3.2 Fuentes de información e instrumentos utilizados

A continuación se procederá a mencionar las fuentes de información e instrumentos que se utilizarán en la presente tesis.

3.2.1. Fuentes de información

El presente trabajo utiliza como fuente de información externa la base de datos de libre acceso *Labeled Face Parts in the Wild* (LFPW). Esta base de datos fue publicada por Sagonas et al. [22] y contiene 994 imágenes distribuidas en 778 imágenes de entrenamiento y 216 imágenes de prueba.

Las imágenes de esta base de datos, como su nombre sugieren, contiene rostros con distintas condiciones de iluminación, postura, resolución, oclusión y tamaño. Sin embargo, su principal utilidad es que estas imágenes poseen 68 puntos de referencia de un rostro (*face landmarks*) etiquetados siguiendo la distribución de la Figura 3.2. Además, estos puntos de referencia fueron etiquetados de manera semi-automatica.

En la Figura 3.3 se pueden apreciar algunos ejemplos de los rostros que contiene la base de datos LFPW con sus respectivos puntos de referencia resaltados. Estos ejemplos solo corresponden a la parte de los rostros, siendo las imágenes originales de mayor tamaño.

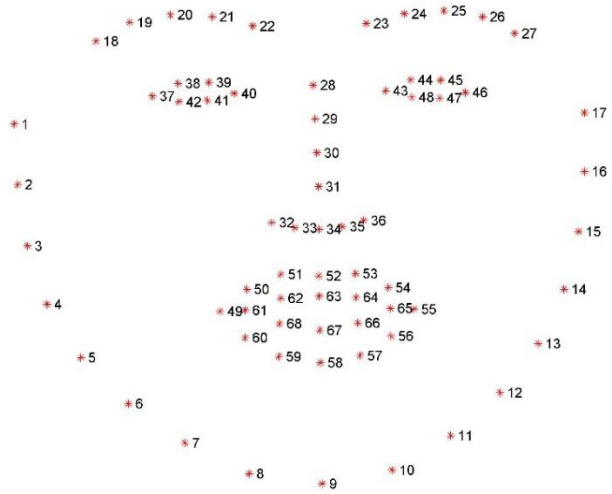


Figura 3.2. Distribución de los puntos de referencia de un rostro en las imágenes de la base de datos LFPW.

Figura 3.3. Muestras de la base de datos LFPW [22].

También se utiliza como fuente de información externa a la base de datos libre *Extended Face Yale Database B*, publicada por Georghiades et al. [9]. Esta base de datos está compuesta por 16’128 imágenes de 28 personas distintas, cada una con el rostro posicionado de 9 formas diferentes y con 64 tipos de condiciones de iluminación. Todas las imágenes de esta base de datos poseen una resolución de 480x640 pixeles y están disponibles únicamente en escala de grises. En la Figura 3.4 se puede apreciar algunos ejemplos de las imágenes de esta base de datos.

Figura 3.4. Muestras de la base de datos *Extended Face Yale Database B* [9].

3.2.2. Instrumentos utilizados

La principal entrada del algoritmo que se propone en el presente trabajo es una imagen digital, ya que esta es la que contiene información del rostro. Por esta razón, es necesario contar con una cámara digital para la toma y/o recolección de muestras (base de datos).

Debido a que no se plantea ningún tipo de restricción en las condiciones de iluminación con las que se capturan las imágenes, la capacidad de tomar fotos en ambientes con condiciones de escasa iluminación es una característica que se considera prioritaria del sensor óptico, ya que la mayoría de estos ha sido diseñado para tomar fotos en ambientes iluminados. Existen distintos tipos de cámaras digitales en el mercado que presentan cierto grado de robustez ante la ausencia de iluminación (como las infrarrojas), sin embargo, tienden a ser costosas.

Debido a que cumple con la capacidad de adecuarse a ambientes con poca iluminación, el sensor óptico que se utilizará para la toma de imágenes es la cámara USB

“HD PRO WEBCAM C920” de la empresa Logitech. En la Tabla 3.1 se mencionan sus especificaciones técnicas.

Tabla 3.1. Especificaciones técnicas de la cámara USB HD PRO WEBCAM C920.

	Logitech: HD PRO WEBCAM C920
Resolución	Videos: Hasta Full HD 1080 (1920 x 1080), 30fps Fotos: 15 Megapíxel
Conexión	USB 3.0
Características Especiales	Tecnología Fluid Crystal, corrección automática de iluminación escasa

Sin embargo, en condiciones de baja iluminación, la cámara demora en actualizar la imagen si es que se adquiere a su máxima resolución (1920x1080). Por esta razón, las imágenes se van a adquirir en el formato de color verdadero (RGB) con una resolución de 960x720 pixeles.

3.2.3. Adquisición de imágenes

El algoritmo propuesto en el presente trabajo está orientado al monitoreo biométrico de conductores vehiculares de transporte público de personas. Por esta razón, las imágenes con las cuales se entrenará a la mayor parte de los algoritmos utilizados son adquiridas dentro de la cabina de un automóvil. Esto se realiza con el objetivo de obtener un conjunto de imágenes con las condiciones de iluminación lo más parecido a las que se podrían tomar en la cabina de un bus de transporte público de personas.

Con la finalidad de poder capturar el rostro del conductor de la manera más frontal posible sin obstaculizar su visión de la autopista, el sensor óptico se encuentra instalado a la altura del espejo retrovisor (como se ilustra en la Figura 3.5).

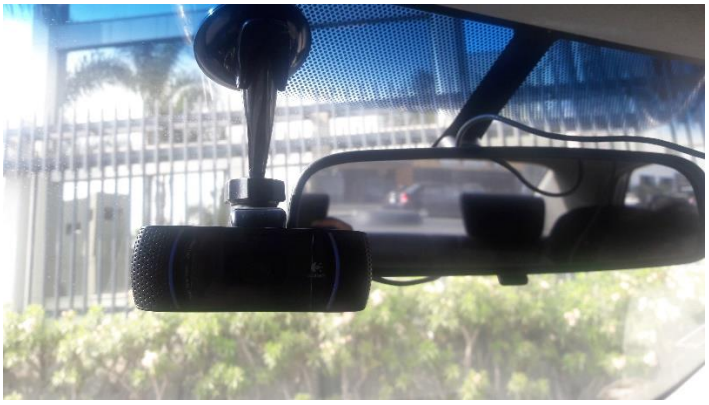


Figura 3.5. Cámara digital instalada en la cabina del conductor de un automóvil de uso particular.

Además, el proceso de adquisición se realizará de manera automática utilizando un ordenador portátil. Cabe resaltar que, con la finalidad de que la iluminación de las fotos obtenidas sea lo más variante posible (iluminación no controlada), el proceso de adquisición de imágenes se realizará en autopistas o lugares aledaños a estas (entorno real) en diferentes horas del día (mañana, tarde y noche).

3.3 Metodología propuesta

En la presente sección se procederá a explicar la metodología propuesta en presente trabajo de Tesis. Una vez expuesta la metodología, se procederá a explicar cada una de las etapas que la conforman de manera detallada, así como su implementación.

La presente metodología se desarrolló con la finalidad de realizar el monitoreo biométrico de conductores de vehículos terrestres, con énfasis en choferes de vehículos de transporte público de personas. Como se mencionó anteriormente, las condiciones de iluminación en dichos escenarios no son controladas. Por esta razón, en el desarrollo de la presente metodología, en un inicio se hace hincapié en la atenuación del ruido debido a las variaciones de iluminación.

La metodología está compuesta por diferentes etapas, cada una con sus propias métricas, alineadas al objetivo general de la presente tesis. Estas etapas son: pre-procesamiento, extracción de características, detección de rostros, corrección geométrica, codificación de rostro y validación de rostro.

La etapa de *pre-procesamiento* aborda la problemática del ruido causado por las variaciones de iluminación. En esta etapa se busca homogenizar y/o estandarizar el área

de las imágenes que contengan un rostro para que, independientemente de las condiciones de iluminación, las operaciones posteriores puedan brindar resultados similares para una misma persona. Para esta etapa se utilizarán las técnicas de corrección gamma y especificación rápida local de histograma (FLHS), teniendo como entrada una imagen en escala de grises y generando una imagen pre-procesada (también en escala de grises).

La etapa de *extracción de características* se centra en codificar la imagen por regiones en vectores de menor dimensión llamados descriptores. Estos descriptores no se enfocan en captar detalles sino más bien tendencias en regiones. Es decir, brindan la información suficiente para detectar rostros en general, pero no para realizar la identificación de personas. En esta etapa de extracción de características se utilizará al descriptor de histograma de gradientes orientados sobre las imágenes pre-procesadas, obteniéndose un conjunto de descriptores de una imagen.

La etapa de *detección de rostros* se enfoca en identificar patrones en los descriptores que permitan identificar un rostro en general. Esta etapa consiste en generar un modelo matemático (clasificador) que delimite las regiones de la imagen pre-procesada que contienen un rostro en base a los descriptores obtenidos en la etapa anterior. Para esta etapa se utilizará una máquina de soporte vectorial, la cual estimará la región de la imagen en la cual se encuentra el rostro. La salida de esta etapa corresponde a la sección de la imagen pre-procesada, la cual será almacenada en una nueva imagen que tendrá el nombre de “Imagen de rostro”.

La etapa de *corrección geométrica* busca compensar, en lo posible, las distorsiones que pueden generarse debido a la orientación del rostro del conductor. La finalidad de esta etapa es realizar una transformación que proyecte al rostro de manera perpendicular a la imagen y que además lo alinee a una posición estándar. Para esto, se utilizará la técnica de regresión en cascada basada en arboles de decisión para estimar la orientación del rostro, para posteriormente aplicar una transformación afín que realice la corrección geométrica. Esta etapa tiene como entrada a la imagen de rostro (obtenida en el proceso anterior) y tiene como salida la imagen de rostro corregido.

La etapa de *codificación de rostro* está orientada a cuantificar las características de un rostro en vectores de codificación de rostro (*face encodings*). A diferencia de la etapa de extracción de características, en esta etapa se busca resaltar y/o cuantificar las características de un rostro a detalle, es decir, encontrar unos nuevos descriptores que nos permitan identificar un rostro. Para esto, se utilizará una red neuronal convolucional, la cual

tendrá como entrada a la imagen de rostro corregido (con las modificaciones pertinentes) y generará a partir de esta un vector de codificación de rostro.

Finalmente, la etapa de *validación de rostro* se enfoca en generar un modelo matemático (clasificador) que permita identificar los rostros en las imágenes en base a los vectores de codificación de rostro obtenidos en la etapa anterior. Para esta tarea final, al igual que en la etapa de detección de rostros, se utilizará una máquina de soporte vectorial que tenga como entrada al vector de codificación de rostro. El resultado de esta etapa, y del algoritmo propuesto, son las etiquetas de rostro que identifican a cada conductor.

Cabe resaltar que el tiempo de manejo a monitorear está comprendido entre 4 a 5 horas (Ministerio de Justicia del Perú [20]), por lo que el procesamiento no necesariamente debe ser en tiempo real. En base a esto, se estima que realizar una validación de rostro de conductor cada 15 minutos es prudente.

En la Figura 3.6 se puede apreciar el diagrama de flujo que sintetiza el flujo de la imagen a través de las etapas explicadas en la metodología.

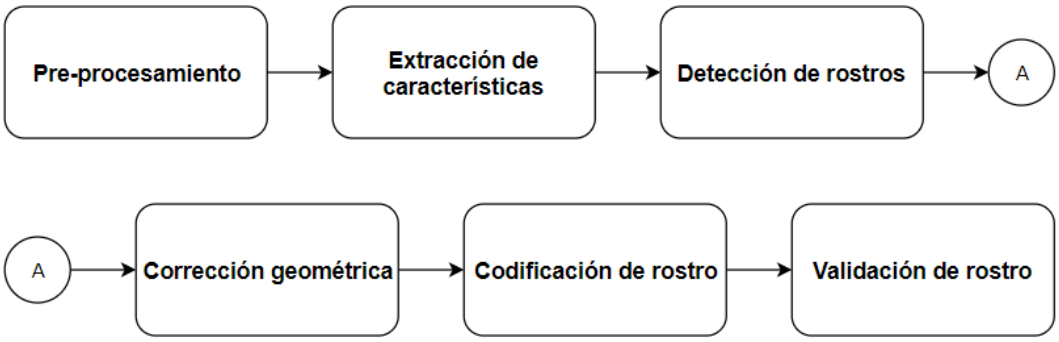


Figura 3.6. Diagrama de flujo de la metodología.

3.4 Implementación del algoritmo propuesto

En la siguiente sección se procede a describir la descripción de la implementación del algoritmo propuesto en la metodología, en donde se ajustan los parámetros de las diversas técnicas implementadas y explicadas en el marco teórico. Sin embargo, para comenzar con la implementación del algoritmo propuesto, se requiere generar una base de datos que cumplan con distintas condiciones de iluminación. Por esta razón, antes de proceder a la explicación de la implementación de las etapas de la metodología, se procede a explicar la etapa de adquisición de imágenes.

3.4.1. Adquisición de Imágenes

La etapa de adquisición de imágenes consiste en generar una base de datos con la cual realizar el entrenamiento de los algoritmos mencionados anteriormente. Las imágenes de esta base de datos fueron tomadas con la finalidad de capturar la mayor cantidad de condiciones de iluminación diferentes a la que son expuestas las cabinas de los buses de transporte público de personas mientras realizan un recorrido. Cabe resaltar que las imágenes adquiridas corresponden a la cabina de un automóvil y no a la de un autobús, pero se estima que las condiciones de iluminación deben ser similares.

La adquisición de imágenes se realizó en la cabina del conductor de un automóvil particular de carrocería sedán, modelo Cerato (de la marca Kia) del año 2009. Las imágenes de la base de datos fueron adquiridas en distintos distritos de Lima (San Borja, San Miguel y Surco) y en distintos horarios.

Estas imágenes se adquirieron utilizando la cámara web “HD PRO WEBCAM C920” (de la marca Logitech) mediante una interfaz gráfica en MatLab®. Las imágenes se adquirieron con una resolución de 960x720 pixeles, en color verdadero (RGB) y con un intervalo de 5 segundos entre cada imagen. Si bien la cámara web posee una resolución máxima de 1920x1080 pixeles, adquirir imágenes en esta resolución no es lo óptimo debido a que el sensor presenta un *delay* en la toma de imágenes y esto genera una distorsión en las mismas, principalmente aquellas con escasa iluminación. Por esta razón, la captura de las imágenes se realizó con una resolución de 960x720 pixels: los suficientemente pequeña para que el delay en la toma de imágenes no afecte a las características de un rostro y lo suficientemente amplias para que un rostro tenga el tamaño requerido para la realizar una correcta codificación de rostro (Tabla 3.3). Estas imágenes son tomadas con una resolución de 8 bits.

Finalmente, las imágenes obtenidas son guardadas en una laptop en un formato que no comprime información de la imagen: *bitmap*.

En la Figura 3.7 se puede apreciar el proceso de adquisición de las imágenes para la generación de la base de datos, estando la cámara en la posición que se señala en la Figura 3.5.

(a) (b)

Figura 3.7. Proceso de adquisición de imágenes: (a) conductor y (b) adquisición de la imagen mediante una laptop.

Al final se generó una base de datos que consta de 13’000 imágenes de 30 personas distintas, en donde las condiciones de iluminación del entorno y las fuentes de iluminación (interna y externa) varían de acorde a lo planificado. Esta base de datos tiene un tamaño aproximado de 30.1 GB y en la Tabla 3.2 se muestran sus características:

Tabla 3.2. Características de la base de datos de imágenes de conductores.

	Nro. Imágenes	**Nro. Personas**	**Horario**
TIPO 1	4’600	17	10:00 – 15:59
TIPO 2	3’300	11	15:59 – 17:59
TIPO 3	5’100	17	18:00 – 22:00
Total	**13’000**		

En la Figura 3.8 se muestran algunos ejemplos de las imágenes que conforman la base de datos, donde además se pueden apreciar distintas condiciones de iluminación.

Figura 3.8. Muestras de la base de datos de imágenes de conductores generada para el presente trabajo.

Sin embargo, con la finalidad de que la base de datos represente con mayor fidelidad a las imágenes que podrían adquirirse de un conductor de bus de transporte público de personas mientras conduce, se tomó en cuenta las distintas posiciones que el conductor puede adoptar en las cuales no se pueda realizar la identificación de rostro. Además, también se consideraron aquellos escenarios en los cuales el rostro del conductor está en movimiento (y las características obtenidas no son lo suficientemente nítidas para la identificación del rostro). En la Figura 3.9 se pueden apreciar algunos ejemplos de este tipo de imágenes.

Figura 3.9. Muestras de la base de datos de imágenes de conductores las cuales se consideran que no deberían ser identificables.

Cabe resaltar que el último grupo de imágenes mencionadas anteriormente no se utilizará en el entrenamiento de la etapa de validación de rostro pero sí como refuerzo negativo en el entrenamiento de la etapa de detección de rostros.

3.4.2. Pre-procesamiento

La etapa de pre-procesamiento tiene como objetivo la atenuación del ruido con respecto a la iluminación de las imágenes adquiridas en la cabina del conductor. Cabe resaltar que hay distintas fuentes de ruido por iluminación como la luz ambiental (y la posición del sol con respecto al rostro del conductor), el alumbrado público, las luces de otros vehículos, etc.

Las imágenes que conforman la base de datos han sido adquiridas en el modelo de color RGB, por lo que si a estas imágenes les asignamos la variable I_{RGB}, esta tiene una dimensionalidad de 960x720x3. Sin embargo, las técnicas de pre-procesamiento expuestas en el marco teórico admiten únicamente imágenes en escala de grises, por lo que se procede a realizar la conversión de I_{RGB} a I_{GRAY} utilizando la Ecuación 2.5 con los siguientes coeficientes (Ecuación 3.1):

$$I_{GRAY_{i,j}} = 0.299 \cdot I_{R_{i,j}} + 0.587 \cdot I_{G_{i,j}} + 0.114 \cdot I_{B_{i,j}} \tag{3.1}$$

Siendo $I_{GRAY_{i,j}}$, $I_{R_{i,j}}$, $I_{G_{i,j}}$ y $I_{B_{i,j}}$ los pixeles con coordenadas (i, j) de la imagen I_{GRAY} y los componentes rojo, verde y azul de la imagen I_{RGB} respectivamente.

Los valores de estos coeficientes utilizados en la Ecuación 3.1 son los recomendados por [10] en el modelo de color YUV, siendo estos los parámetros utilizados para estimar la componente de iluminación de dicho modelo de color.

En la etapa de pre-procesamiento, se utiliza el algoritmo especificación rápida local de histograma (FLHS) como método principal. La razón por la que se escogió dicho algoritmo es porque al ser una especificación local de histograma, se adapta a las regiones oscuras y claras de una imagen. Además, al ser una optimización de la especificación local de histograma en cuanto a tiempo, consume menos recursos computacionales y además es de menor complejidad (y por ende, su tiempo de ejecución es menor para una misma entrada).

Sin embargo, el algoritmo FLHS posee una complejidad proporcional al número de pixeles en la imagen (según [16]) por lo que utilizar la imagen con la resolución y formato adquiridos puede seguir siendo costoso computacionalmente a pesar de ser una optimización. Por esta razón, se decidió trabajar con una imagen re-escalada de la imagen adquirida y, además, en escala de grises. La imagen original I_{GRAY}, con una resolución de 960x720 pixeles, fue re-escalada a una imagen de 640x360 pixeles, que es representada por I_{2GRAY}. Este cambio, como se aprecia en la Tabla 3.3, permite que el tiempo de ejecución del algoritmo FLHS se reduzca 6.75 veces.

Tabla 3.3. Comparación de la cantidad de pixeles en diferentes formatos de imagen

	CANTIDAD DE PIXELES	**TAMAÑO EN ESCALA**
Imagen adquirida (RGB)	2’073’600 (960x720x3)	6.75x
Imagen re-escalada (escala de grises)	307200 (640x480x1)	1x

Como se mencionó en la etapa de adquisición de imágenes, la resolución del pixel se mantiene en 8 bits, por lo que las imágenes I_{2GRAY} presentan un total de 256 niveles digitales de resolución (L=256). Una vez definido el número de niveles digitales y dado que $L/2$ es par, se define el número de sub-intervalos óptimos p^* (Ecuación 3.2):

$$p^* = \left\lceil \sqrt{256/2 + 5/4} - 1/2 \right\rceil = 11 \tag{3.2}$$

Una vez definido p^*, se infiere que el nuevo histograma de sub-intervalos posee 22 ($2p^*$) intervalos o niveles. Además, por temas de practicidad y recomendación de [16], se define $e = [0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0]$.

Los valores de e y p^*, son reemplazados en la Ecuación 2.9 para definir la cantidad de elementos (niveles del histograma tradicional). La longitud de los sub-intervalos de la primera mitad z^* se define en la Ecuación 3.3:

$$z^* = [22, 20, 18, 16, 14, 12, 10, 8, 4, 2, 2] \tag{3.3}$$

Mientras que la longitud de sub-intervalos de la segunda mitad z' se define en la Ecuación 3.4, siendo además por definición el reflejo de z^*:

$$z' = [2, 2, 4, 8, 10, 12, 14, 16, 18, 20, 22] \tag{3.4}$$

Luego, el vector total de las longitudes de las particiones del histograma tradicional z se obtiene al concatenar los vectores z^* y z' (Ecuación 3.5):

$$z = [22, 20, 18, 16, 14, 12, 10, 8, 4, 2, 2,2, 2, 4, 8, 10, 12, 14, 16, 18, 20, 22] \tag{3.5}$$

Los autores de [16] sugieren que se utilice un histograma logarítmico como histograma base para realizar la especificación. Esto es debido a la calidad de los resultados que se obtuvieron utilizando dicho histograma en pruebas con distintos tipos de imágenes. Cabe resaltar que dichos resultados se midieron por medio de inspección visual. En la Figura 3.10 se puede apreciar un histograma logarítmico generado utilizando la ecuación $y = \ln(x + 1) + 1$, el cual ha sido normalizado utilizando la norma L_1 que se muestra en la Tabla 2.1.

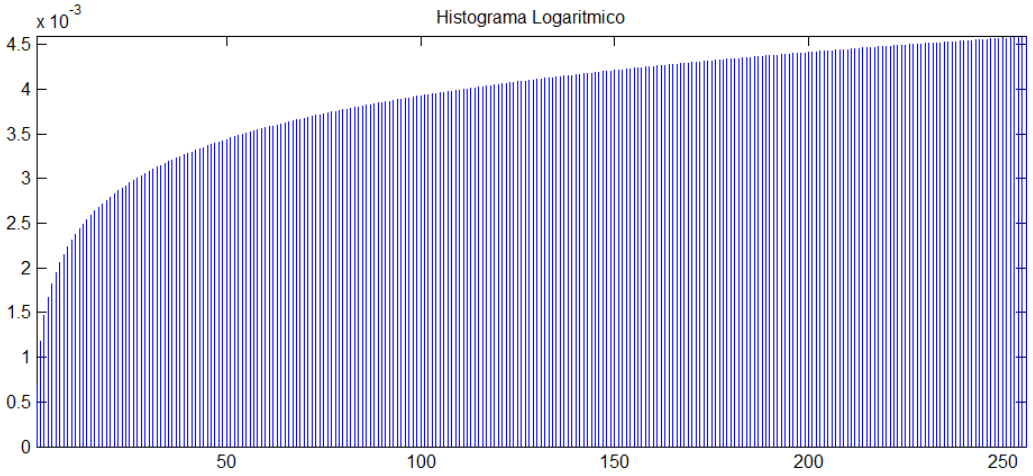
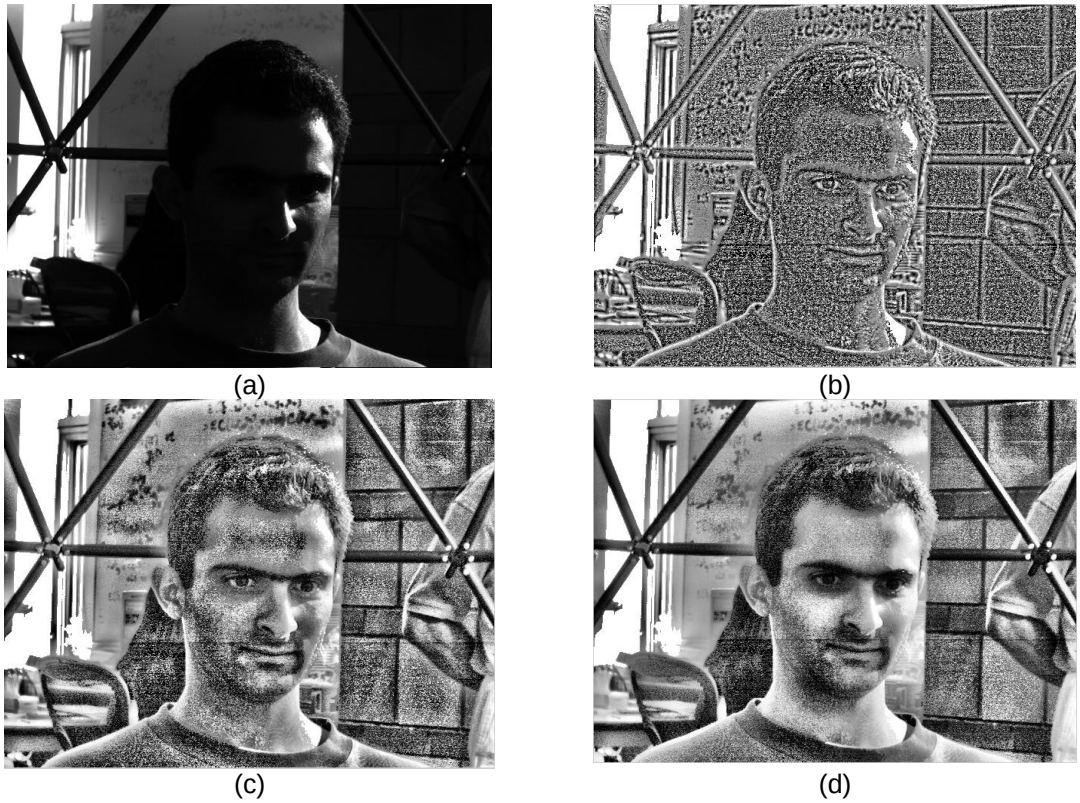


Figura 3.10. Histograma logarítmico.

Por último, el algoritmo FLHS, al ser una especificación local del histograma, requiere que se defina las dimensiones de la ventana de procesamiento que recorrerá pixel por a pixel toda la imagen. Para la presente aplicación, se seleccionó un tipo de ventana cuadrada, ya que no favorece a ninguna dirección en particular (horizontal o vertical). El tamaño de la ventana está definido por el parámetro r_f, siendo la longitud de cada lado de la ventana "$2r_f$+1" pixeles. La longitud de cada lado de la ventana de procesamiento está definida de esta manera porque siempre debe haber un pixel central sobre el cual se realizará el procesamiento.

La elección del parámetro r_f no repercute de manera tan significativa en el coste computacional como hacen el tamaño de la imagen o el número de niveles digitales L. Sin embargo, es el parámetro que más influye en el resultado del algoritmo. Esto ocurre porque mientras más grande es r_f, ingresan más pixeles vecinos al histograma y por ende, se puede obtener una mejor muestra estadística del vecindario del pixel central. Para esto, se realizaron pruebas en la base de datos *Extended Face Yale Database B*, la cual está conformada por imágenes de 640x480 pixeles en escala de grises. En la Figura 3.11 se puede apreciar el efecto de la elección del tamaño de r_f en el resultado del algoritmo FLHS sobre imágenes de 640x480, aplicando el pseudocódigo descrito en la Figura 2.7.



(a) (b)

(c) (d)

Figura 3.11. Influencia del tamaño de ventana en el resultado del algoritmo FLHS en imágenes de la base de datos *Extended Face Yale Database B*: (a) imagen original, (b) r_f=5, (c) r_f=25 y (d) r_f=50.

Debido a los resultados que se muestran en la Figura 3.11, se determinó que la ventana de procesamiento será un cuadrado de 51x51 pixeles (r_f=25) para imágenes con una resolución de 640x480 pixeles. Sin embargo, también se puede apreciar que al aplicar el algoritmo FLHS introduce ruido de alta frecuencia en la imagen.

Sin embargo, para las imágenes obtenidas para el presente trabajo en la cabina de un automóvil de uso particular (reales, de similar resolución y tamaño de rostro), este tipo de ruido de alta frecuencia se eleva a niveles alarmantes. Esto se puede apreciar en la Figura 3.12, donde los niveles de ruido afectan no solamente al entorno, sino también a zonas del rostro.

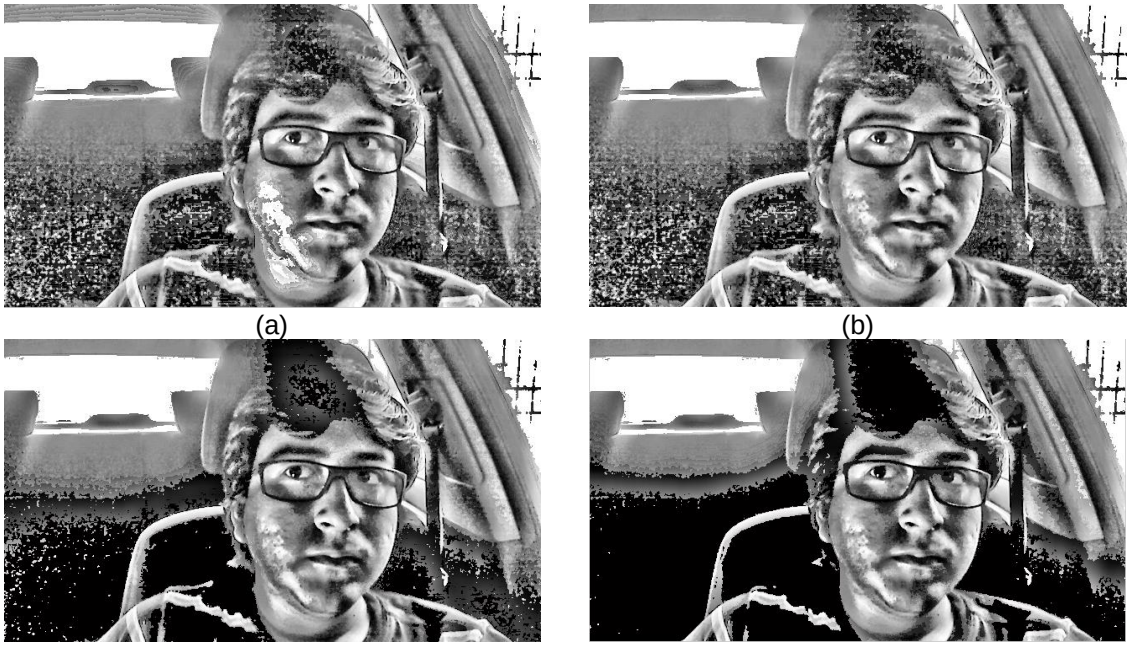
(a) (b)

Figura 3.12. Aplicación del algoritmo FLHS sobre una imagen adquirida para el presente trabajo de investigación.

Este tipo de ruido se presenta en zonas medianamente homogéneas en la imagen, y la razón es que en esas zonas existe tan poco contraste que la menor de las variaciones en la intensidad de los pixeles puede ser la diferencia en que dichos pixeles resulten ser blancos o negros después de aplicar el algoritmo FLHS. Por esta razón, y para aumentar el contraste en ciertas regiones, se aplicará una corrección gamma.

Sin embargo, la corrección gamma aumenta el contraste en regiones con cierto tipo de iluminación específica y disminuye el contraste en otras. Por esta razón, se tiene que tener en cuenta que si bien aumentar el contraste de las regiones claras de una imagen con iluminación alta puede mejorar el resultado del pre-procesamiento, realizar la misma corrección en una imagen oscura puede ser contraproducente. Por esta razón, como estrategia de pre-procesamiento, se plantean 3 tipos de imágenes (como se muestra en la Tabla 3.2): imágenes del TIPO 1, adquiridas durante la mañana; imágenes del TIPO 2, adquiridas durante la tarde e imágenes del TIPO 3, adquiridas de noche.

Una vez formado grupos de imágenes correspondientes a su tipo, se procede a hacer análisis de diferentes tipos de corrección gamma y de cómo esta corrección afecta los resultados de del algoritmos FLHS. En la Figura 3.13 se muestra el análisis realizado para imágenes del TIPO 1, aplicando la Ecuación 2.6 a los elementos de la imagen I_{2GRAY} para diferentes valores de γ y así obteniendo la imagen pre-procesada I_{pp}.



(a) (b) (c) (d)

Figura 3.13. Influencia del filtro gamma antes de utilizar el algoritmo FLHS (r=25) utilizando los parámetros: (a) γ = 0.5, (b) γ = 0.75, (c) γ = 1.5 y (d) γ = 2.

Teniendo en cuenta la inspección visual como punto de referencia, la corrección gamma que mejor se adecua a las imágenes de TIPO 1 es cuando γ = 2. De manera similar, se realizó el análisis para imágenes del TIPO 2 y TIPO 3, obteniéndose los parámetros de pre-procesamiento que se indican en la Tabla 3.4:

Tabla 3.4. Parámetros de pre-procesamiento según el tipo de imagen.

IMAGEN	$\boldsymbol{\gamma}$ **(Corrección gamma)**	$\boldsymbol{r_f}$ **(FLHS)**
TIPO 1	2	25
TIPO 2	2	25
TIPO 3	0.5	25

De la Tabla 3.4 se puede apreciar que las estrategias de pre-procesamiento de las imágenes del TIPO 1 y TIPO 2 son iguales. En la Figura 3.14 se muestran los resultados de las imágenes adquiridas en la sección anterior (I_{RGB}), agrupadas según el tipo de imagen al que corresponden, y los resultados de las estrategias de pre-procesamiento (I_{pp}) que se muestran en la Tabla 3.4.

Figura 3.14. Imágenes de la base de datos (a) del TIPO 1, (c) del TIPO 2 y (e) del TIPO 3 junto a sus respectivos resultados del pre-procesamiento (b), (d) y (f).

3.4.3. Extracción de características

La etapa de extracción de características tiene como objetivo calcular valores numéricos (descriptores), cuya combinación en un orden determinado pueda permitir la identificación de zonas en la imagen que contengan un rostro. La entrada a esta etapa de procesamiento es la imagen pre-procesada I_{pp}.

En la etapa de extracción de características se utiliza el descriptor de histograma de gradientes orientados (HOG), el cual es ampliamente usado en el área de visión computacional para realizar la detección de objetos en general. Se decidió trabajar con este descriptor debido al grado de robustez a las condiciones de iluminación que presenta al operar sobre las derivadas de una imagen (Ecuación 2.11). Además, este descriptor posee un costo computacional bajo, por lo que es posible implementarlo en tiempo real en lenguajes de programación de bajo nivel.

Para la detección de objetos, [5] sugiere que la imagen se divida en celdas con un tamaño de 8x8 pixeles y que los bloques contengan un arreglo de 2x2 celdas. Con este tamaño de celda y distribución de bloques, se puede conseguir una buena descripción de las formas contenidas en una imagen digital. Por otro lado, el principal objetivo de esta etapa es la de detectar de manera general la presencia de un rostro (no necesariamente captar todas sus características), razón por la cual la se realizará un re-escalamiento a la imagen I_{pp} para obtener una nueva imagen en escala de grises I_{2pp} con dimensión de 320x240 pixeles. En la Figura 3.15 y 3.16 se muestra de manera gráfica la división de una imagen en celdas y el agrupamiento de las celdas en bloques respectivamente.

Figura 3.15. Imagen dividida en celdas (rojo).

Figura 3.16. Bloque de 2x2 celdas (azul) resaltado en una imagen dividida en celdas.

Cabe resaltar que las Figuras 3.15 y 3.16 muestran la división espacial de la imagen, pero los histogramas que se utilizarán para describir los objetos se hallan a partir de las imágenes de gradientes ($|G|_{i,j}$) y la imagen de orientación (θ) calculadas a partir de las Ecuaciones 2.12 y 2.13 respectivamente. Para este cálculo se utilizan las imágenes I_{2pp_X} y I_{2pp_Y} que son las derivadas horizontal y vertical de la imagen I_{2pp} (Ecuaciones 2.10 y 2.11) respectivamente.

Además, para la presente Tesis, se toman en cuenta 9 niveles y/o rangos de orientaciones en los histogramas de gradiente orientado. Estos rangos del histograma tienen una amplitud de 20 grados sexagesimales y tienen como puntos centrales a ángulos sexagesimales múltiplos de 20 desde el 0 a 160. Para orientaciones que se encuentren fuera del rango total del histograma (desde -10 hasta 170 grados sexagesimales), estas se computaran en la zona de su ángulo reciproco que si pertenezca a este rango (por ejemplo, un ángulo de -40 grados sexagesimales se computará como 140 grados sexagesimales en el histograma). La Figura 3.17 ilustra los puntos centrales de los niveles considerados en los histogramas de gradientes orientados.

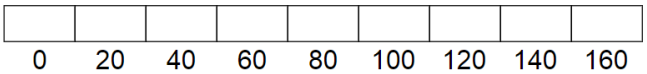


Figura 3.17. Rango de ángulos de orientación (en grados sexagesimales) que serán tomados en el histograma de gradientes orientados.

En la Ecuación 3.6 se muestra la definición de un histograma de orientaciones para la celda con coordenadas (i_{celda}, j_{celda}), la cual contiene los pixeles con coordenadas (i, j) tal que $i \in [8 \times i_{celda}, 8 \times i_{celda} + 7]$ y $j \in [8 \times j_{celda}, 8 \times j_{celda} + 7]$.

$$\text{hist}\theta_{i_{celda},j_{celda}} = [\text{hist}\theta^{1}_{i_{celda},j_{celda}}, \dots, \text{hist}\theta^{9}_{i_{celda},j_{celda}}] \tag{3.6}$$

Del mismo modo, en la Ecuación 3.7 se define el histograma de orientaciones de un bloque con coordenadas (i_{bloque}, j_{bloque}), el cual se calcula como suma de los histogramas de orientaciones de las celdas $\text{hist}\theta$ con las coordenadas (i_{celda}, j_{celda}) de tal manera que $i_{celda} \in [2 \times i_{bloque}, 2 \times i_{bloque} + 1]$ y $j_{celda} \in [2 \times j_{bloque}, 2 \times j_{bloque} + 1]$.

$$\mathbf{hog}\theta'_{i_{bloque},j_{bloque}} = [\text{hog}\theta'^{1}_{i_{bloque},j_{bloque}}, \dots, \text{hog}\theta'^{9}_{i_{bloque},j_{bloque}}] \tag{3.7}$$

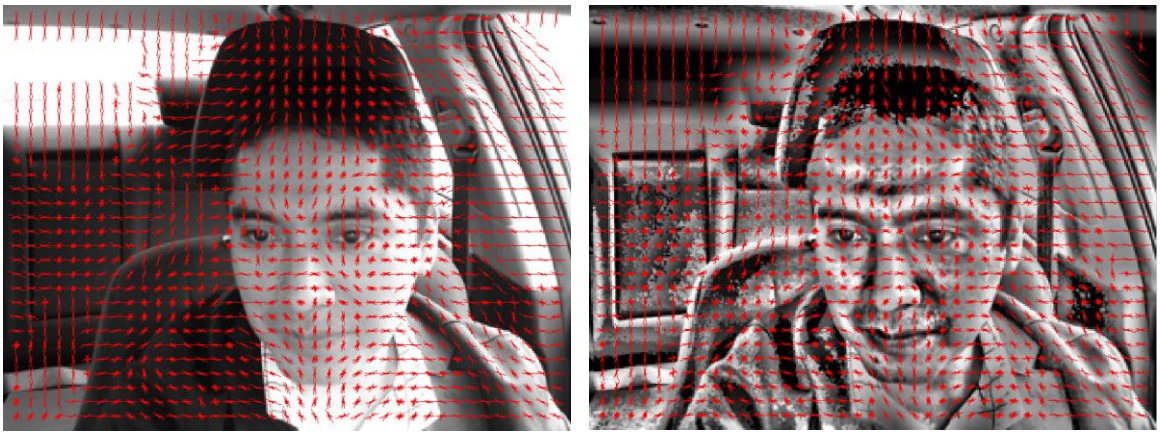
Donde $\mathbf{hog}\theta'^{,\text{k}_{\text{ang}}}_{i_{bloque},j_{bloque}} = \sum_{i_{celda}} \sum_{j_{celda}} \text{hist}\theta^{\text{k}_{\text{ang}}}_{i_{celda},j_{celda}}$, para k_{ang}=1,2,…,9

Finalmente, el histograma de gradientes orientados se calcula como muestra la Ecuación 3.8:

$$\mathbf{hog}\theta_{i_{bloque},j_{bloque}} = L_2 norm\left(\mathbf{hog}\theta'_{i_{bloque},j_{bloque}}\right) \tag{3.8}$$

Siendo $L_2 norm$ definido en la Tabla 2.1 con un valor de la constante e_{norm}=0.001.

En la Figura 3.18 se muestra de manera visual el descriptor de histograma de gradientes orientados dibujando un conjunto de líneas rojas al centro de cada bloque, en donde sus longitudes son proporcionales al valor $\mathbf{hog}\theta_{i_{bloque},j_{bloque}}$ y sus orientaciones son correspondientes al ángulo central de cada rango.



(a) (b)

Figura 3.18. Visualización del descriptor HOG sobre una imagen (a) en escala de grises y (b) pre-procesada.

3.4.4. Detección de rostros

La etapa de detección de rostros tiene como objetivo segmentar las regiones de la imagen en las cuales aparezca un rostro. Para esto, se entrena un algoritmo de clasificación que pueda sobrellevar el ruido presente en las imágenes a trabajar.

En la etapa de detección de rostros se utiliza una máquina de soporte vectorial (SVM) debido a que, en conjunto al descriptor HOG, marcan el estado del arte en la detección de rostros en el área de los métodos tradicionales de visión computacional (sin considerar *deep learning*). Además, la mayor ventaja de este tipo de clasificadores es que su resultado converge en el mínimo global de su función de costo, a diferencia de una red neuronal que puede converger en un mínimo local (solución no óptima).

Sin embargo, como se expuso en el marco teórico, para realizar el entrenamiento de un SVM se necesita contar con una base de datos debidamente etiquetada. Para esto, se define por medio de inspección visual el tamaño promedio de un rostro para una imagen de 320x240, con la finalidad de tener una estructura determinada para los datos de entrada al SVM. En la Figura 3.19 se muestra esta aproximación, en donde se puede apreciar que el rostro está contenido en un arreglo de 12x12 bloques, 13x13 celdas o de 104x104 pixeles.

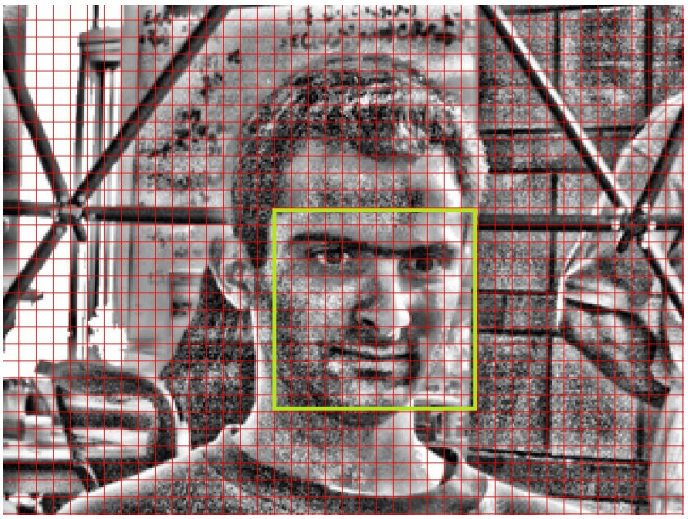


Figura 3.19. Zona de la imagen correspondiente a un rostro (verde).

Con el tamaño de rostro definido, se procede a formar el vector de entrada al SVM ubicando todos los elementos de los histogramas de gradiente orientado correspondientes a los bloques que contiene un arreglo de 12x12. En la Ecuación 3.9 se muestra lo anteriormente mencionado:

$$\mathbf{xHOG}_{\mathrm{i_{HOG}},j_{HOG}} = [\mathbf{hog}\theta_{\mathrm{i_{HOG}},j_{HOG}}, \dots, \mathbf{hog}\theta_{\mathrm{i_{HOG}}+11,j_{HOG}+11}] \tag{3.9}$$

siendo la dimensionalidad de $\mathbf{hog}\theta$ la misma que $\mathbf{hog}'\theta$ e igual a 9 (Ecuación 3.7), por lo que la dimensionalidad de xHOG viene a ser 12x12x9 = 1296.

Con el proceso de generación del vector xHOG definido, se procede a realizar el etiquetado de las imágenes. Para esto, se utilizó una interfaz gráfica de usuario desarrollada en MatLab® tal y como se muestra en la Figura 3.20.

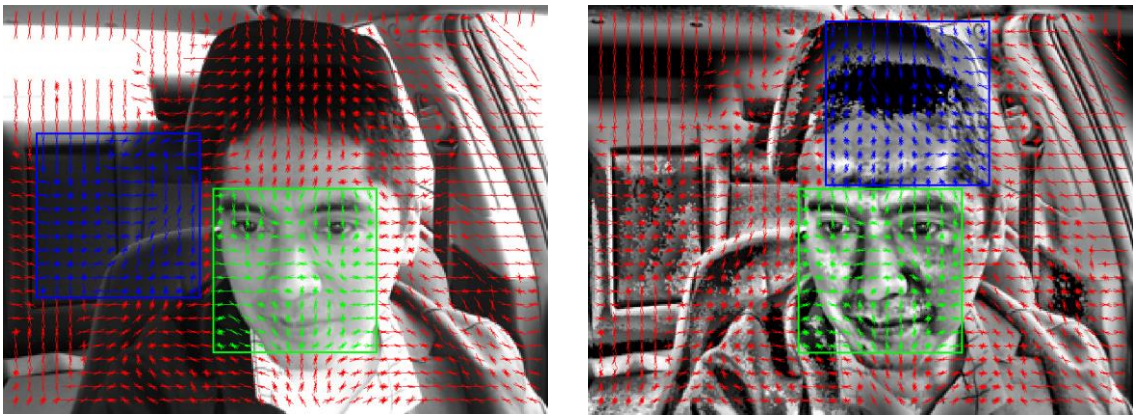
Figura 3.20. Interfaz gráfica de usuario para realizar la tarea de etiquetado de rostro.

En donde se puede apreciar que la interfaz requiere que se ingrese la dirección de la carpeta que contiene las imágenes. Esta dirección se agrega tras abrir el explorador del sistema utilizando el botón “Explorar”.

El proceso de etiquetado consiste, de manera manual, delimitar las regiones que contienen un rostro en la interfaz gráfica de usuario y presionar el botón “Siguiente” (Figura 3.20). Esta acción genera 2 vectores $\mathbf{xHOG}$, uno correspondiente a la región delimitada en la interfaz gráfica y otro completamente aleatorio, teniendo en cuenta las dimensiones anteriormente mencionadas. Adicionalmente a esto, se generan también las etiquetas yHOG en base a la siguiente lógica (Ecuación 3.10):

$$\mathrm{yHOG} = \begin{cases} 1, & \textbf{xHOG} \text{ corresponde a un rostro} \\ -1, & \text{cualquier otro caso} \end{cases} \tag{3.10}$$

En la Figura 3.21 se muestra de manera gráfica la generación de los 2 vectores xHOG por imagen:



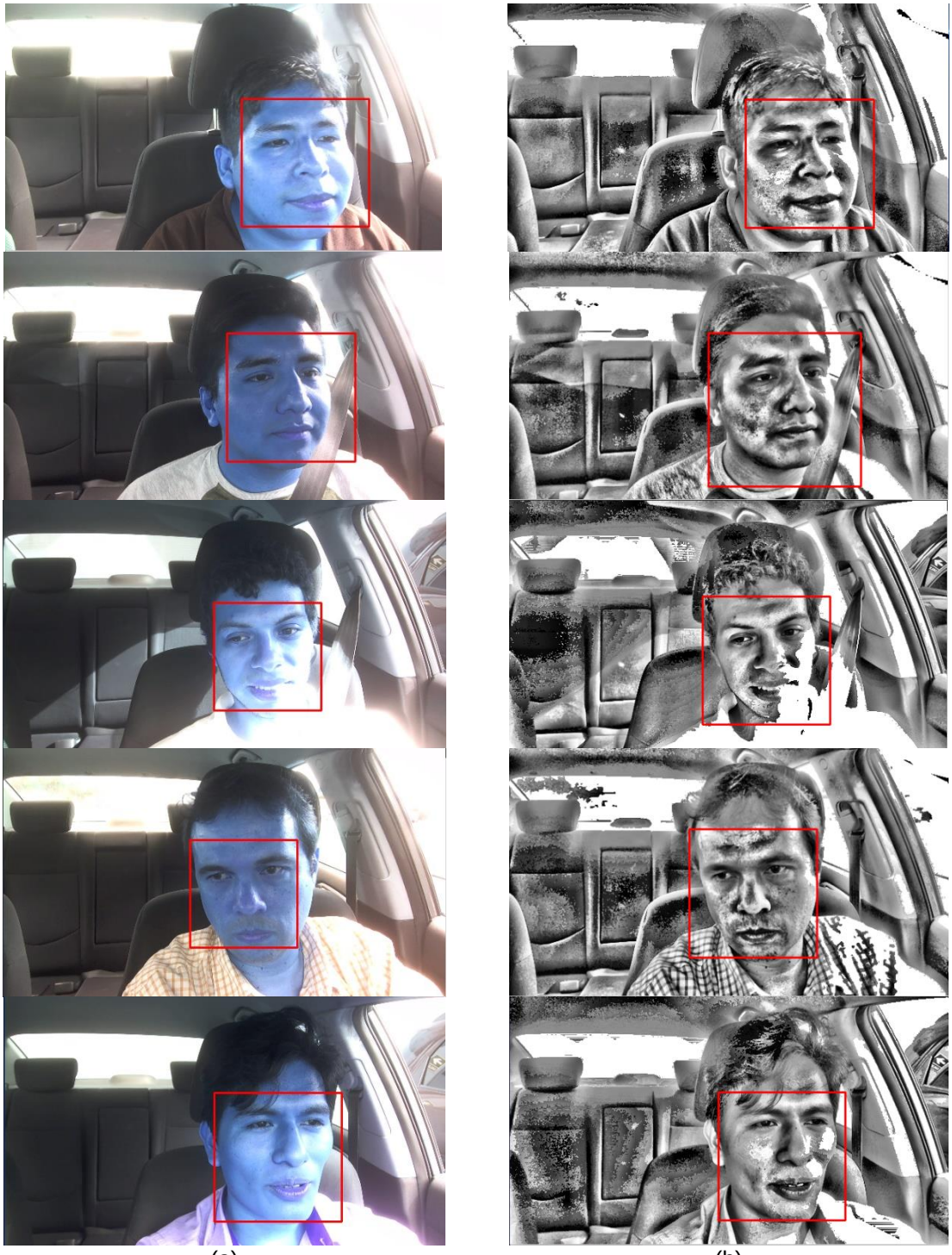
(a) (b)

Figura 3.21. Rostro etiquetado basado en el descriptor HOG en una imagen (a) en escala de grises y (b) pre-procesada. Las zonas verdes corresponden a los descriptores HOG de un rostro, mientras que las azules a regiones que no contienen uno.

Sin embargo, en la Figura 3.20 se puede apreciar un botón extra: “No hay rostro. Este botón se utiliza cuando se considera que el rostro está en una posición que no debería ser reconocida por el algoritmo (como los ejemplos mostrados en la Figura 3.9).

Tras realizar el etiquetado manual, se generó un total de 7617 vectores xHOG correspondientes a rostros (yHOG = 1) y 11751 vectores xHOG que no corresponden a un rostro (yHOG = -1).

Una vez generada la base de datos correspondiente a la detección de rostros, se entrenó un SVM con un coeficiente de margen suave (C) igual a 0.1 y utilizando un *kernel* lineal (Tabla 2.5, polinomial con el parámetro d=1). Los valores de los parámetros anteriormente mencionados son los recomendados por [5] para realizar la detección de objetos utilizando un SVM. Cabe resaltar que el entrenamiento de la SVM se realizó utilizando el 80% de la base de datos de detección de rostros, siendo el otro 20% reservado como data de prueba para obtener las métricas de sensibilidad y especificidad que se muestran en la Tabla 4.2. En la Figura 3.22 se muestra de manera gráfica los resultados obtenidos tras entrenar una SVM que detecte los rostros en imágenes I_{2GRAY} y I_{2pp}.



(a) (b)

Figura 3.22. Detección de rostros utilizando (a) imágenes RGB y (b) imágenes pre-procesadas.

Finalmente, se extrae la región del rostro a partir de las coordenadas $(\mathrm{i}_{\mathrm{HOG}}, j_{HOG})$ correspondientes a los vectores xHOG que el SVM detecta como rostro (les asigna una

etiqueta 1). Cabe resaltar que estas coordenadas se re-escalan para que las regiones del rostro encontradas en las imágenes de 320x240 coincidan con las regiones de las imágenes de mayor dimensión y/o resolución I_{2GRAY} e I_{pp}. En base a esto, se generan nuevas imágenes de rostro $I_{rostroGRAY}$ e $I_{rostropp}$.

3.4.5. Corrección geométrica

La etapa de corrección geométrica tiene como objetivo realizar transformaciones en las coordenadas de los pixeles de la imagen de un rostro con la finalidad de alinear y/o estandarizar la distribución de las regiones de interés (ojos, boca, nariz, etc.). Esta estandarización es necesaria debido a que los cambios de postura de una persona pueden afectar al rendimiento de los algoritmos utilizados en etapas posteriores (codificación y validación de rostro).

En la etapa de corrección geométrica se utiliza la transformación afín basada en la detección de puntos relevantes de un rostro mediante una regresión en cascada basado en árboles de decisiones (ERT). Se escogió la transformación afín porque es una transformada que permite proyectar puntos de una imagen a otra de manera precisa. Por otro lado, se escogió el algoritmo ERT debido a que una vez implementado, el algoritmo es ligero y puede ser implementado en tiempo real.

De manera intuitiva, lo que se busca con la trasformación afín es que la imagen de un rostro sea sometida a una rotación que alinee a los ojos de manera horizontal y que posteriormente se someta a un cizallamiento que alinee de manera vertical el centro de la boca con el punto medio entre ambos ojos. Esto se muestra de manera gráfica en la Figura 3.23, en donde se aprecia que las líneas azules representan a la orientación de los ojos en el rostro y las líneas rojas representan la ubicación de los centros de la boca y el punto medio entre los ojos.

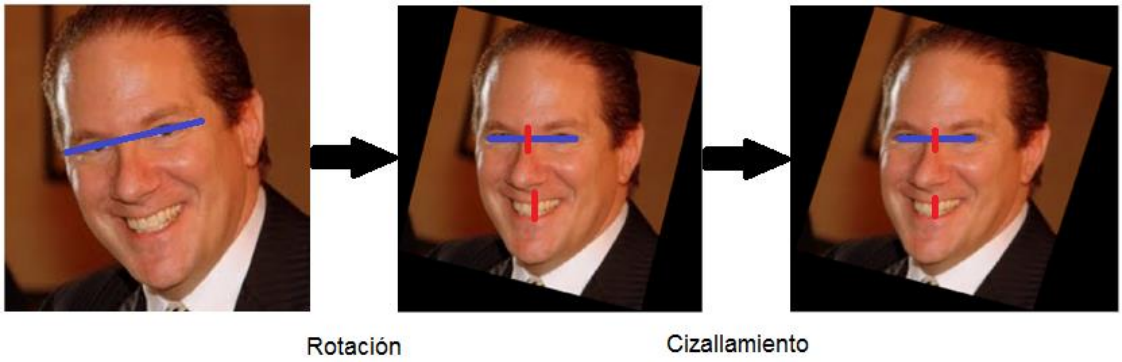


Figura 3.23. Flujo de procesamiento esperado en la etapa de corrección geométrica. Fuente de la imagen original: [22].

Sin embargo, para realizar dicha transformación afín es necesario identificar regiones del rostro tales como los ojos y la boca. Para realizar esta identificación, se entrenará el algoritmo de regresión en cascada basado en arboles de decisión utilizando la base de datos *Labeled Face Parts in the Wild* (LFPW), publicada por [22].

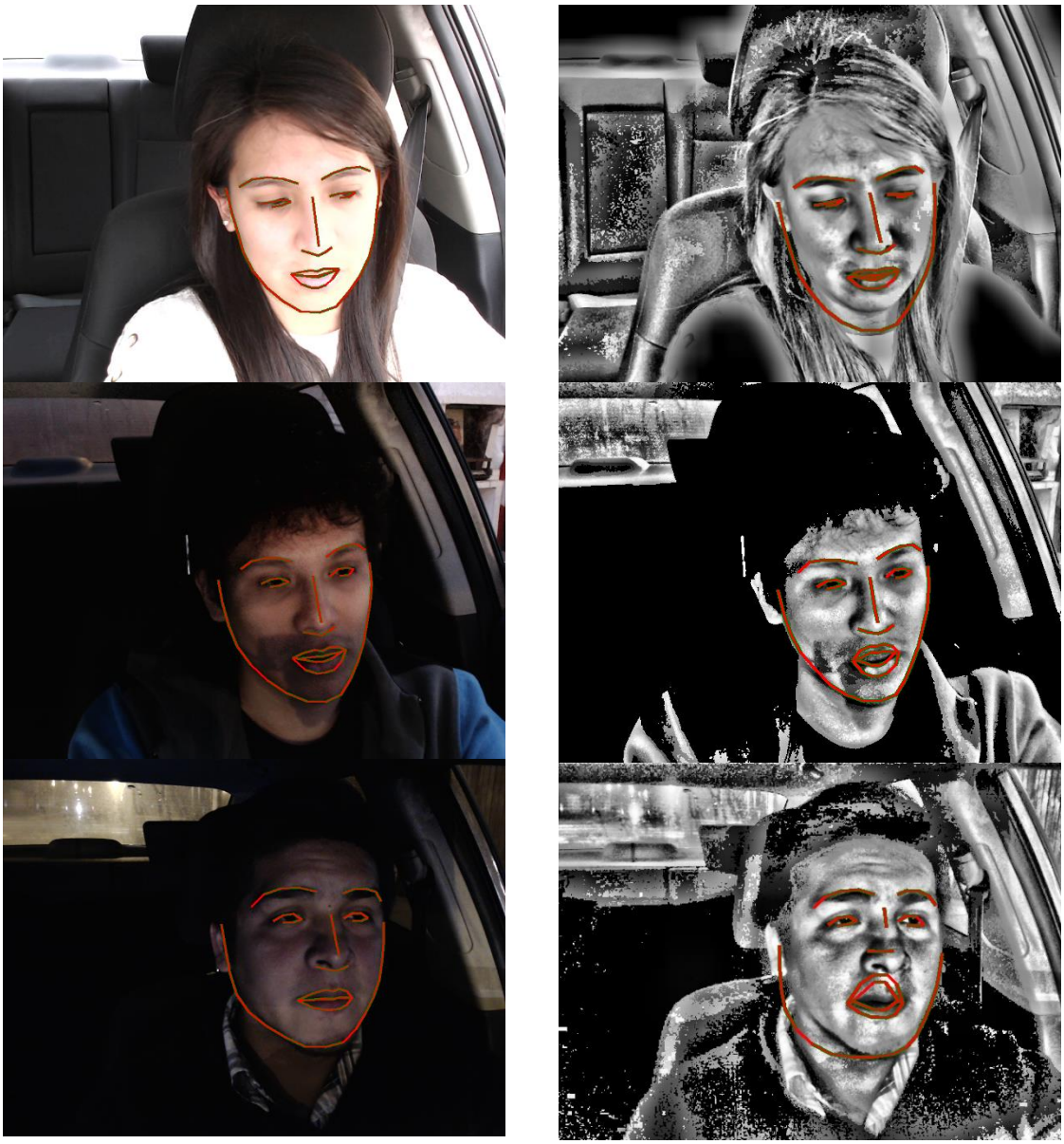
Como sugiere [13], la regresión en cascada basado en arboles de decisión se entrenó utilizando un factor de aprendizaje ν_{ert} = 1, 10 regresores (r_t) basados en arboles de decisión utilizando *gradient tree boosting*, donde cada regresor está conformado por K = 500 regresores débiles (g_k) con una profundidad de 5 niveles. Los datos de entrenamiento se generan realizando 20 inicializaciones diferentes de la estimación del vector de forma ($\hat{\mathrm{S}}_i^{(0)}$) por cada imagen, es decir, por cada imagen de la base de datos de entrenamiento se generan 20 ternas de enteramiento (I_{π_i}, $\hat{\mathrm{S}}_i^{(0)}$, $\Delta\hat{\mathrm{S}}_i^{(0)}$) que cumplen con las Ecuaciones 2.29, 2.30 y 2.31. Para el entrenamiento, como se expuso en la sección de Fuentes de información (sub-capítulo 3.2.1), se utilizaron 778 imágenes de la base de datos LFPW. Adicionalmente, el rendimiento del algoritmo ERT es evaluado utilizando el error normalizado propuesto por [13] y definido en la Ecuación 3.11:

$$\text{error normalizado} = \frac{\text{distancia}(\mathrm{S} - \hat{\mathrm{S}})}{\text{distancia}(\mathrm{p}_{40} - \mathrm{p}_{43})} \tag{3.11}$$

siendo S es el vector de forma esperado, $\hat{\mathrm{S}}$ el vector de forma estimado por el algoritmo ERT y los puntos p_{40}-p_{43} corresponden a los puntos internos de los ojos en el vector de forma esperado (la distribución de estos puntos se puede observar en la Figura 3.2).

El promedio de los errores normalizados calculados a partir de las 216 imágenes de prueba es de 0.057, valor similar al promedio de los errores normalizados (0.049) obtenido por [13].

En la Figura 3.24 se muestran los resultados de la estimación de los vectores de forma en la base de datos generada para el presente trabajo. Cabe resaltar que en esta Figura se muestran los vectores de forma calculados a partir de las imágenes $I_{rostroGRAY}$ e $I_{rostropp}$, pero son mostrados sobre las imágenes I_{RGB} e I_{pp}.



(a) (b)

Figura 3.24. Estimación del vector de forma en imágenes de la base de datos (a) sin pre-procesar y (b) con pre-procesamiento

En base a los resultados mostrados en la Figura 3.24, se puede apreciar que el rendimiento del algoritmo ERT sobre las imágenes de rostro pre-procesadas ($I_{rostropp}$) no es el adecuado. Por esta razón, la estimación del vector de forma de un rostro ($\hat{S}$) se realiza aplicando el algoritmo ERT sobre imágenes de rostro sin pre-procesar ($I_{rostroGRAY}$).

Una vez obtenida la estimación del vector de forma de un rostro ($\hat{S}$) se define que la estimación de pose de un rostro se realizará teniendo en cuenta la zona interna de los ojos con respecto al rostro y el punto medio del labio superior, las cuales serán alineadas como sugiere la Figura 3.22. De la Figura 3.2 se puede apreciar que estas regiones corresponden a los puntos del vector de forma de un rostro ($\hat{S}$) etiquetados con los números 40, 43 y 42; a los cuales se hará referencia en adelante como los puntos p_{40}, p_{43} y p_{52}. Estos puntos serán proyectados en nuevo sistema de coordenadas mediante una transformación afín, siendo p'_{40}, p'_{43} y p'_{52} sus recíprocos en el nuevo sistema. En la Figura 3.25 se muestra de forma gráfica la proyección deseada de los puntos p_{40}, p_{43} y p_{52} para realizar la corrección geométrica sobre una imagen.

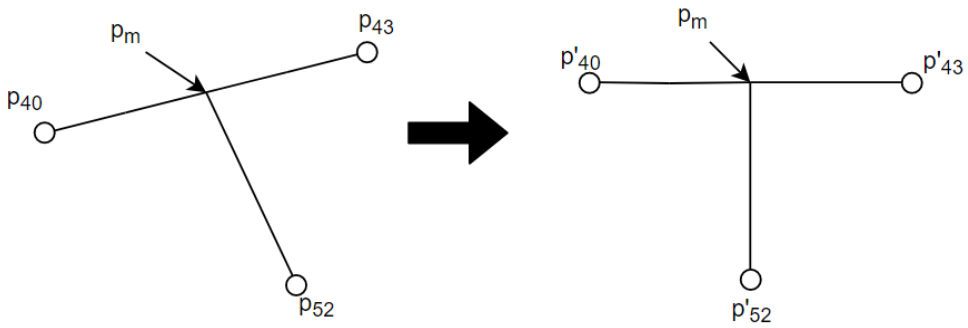


Figura 3.25. Proyección de los puntos p_{40}, p_{43} y p_{52} para realizar correcciones geométricas en las imágenes que contienen un rostro.

Donde p_m es el punto medio de los puntos p_{40} y p_{43}, siendo sus coordenadas (i_{p_m}, j_{p_m}) calculadas según la Ecuación 3.12:

$$(i_{p_m}, j_{p_m}) = \left(\frac{i_{p_{40}} + i_{p_{43}}}{2}, \frac{j_{p_{40}} + j_{p_{43}}}{2}\right) \tag{3.12}$$

Donde (i_{p_q}, j_{p_q}) son las coordenadas del punto p_q para $q \in \{40,43,52\}$.

Si bien la transformación afín genera cierta distorsión en la geometría propia de un rostro (como se mostró en la Figura 3.22), para poder realizar una codificación adecuada se necesita conservar la mayor cantidad de información correspondiente a esta. Por esta

razón, la transformación afín a utilizar se realiza bajo la condición de que las distancias entre los puntos p_{40}-p_{43} y los puntos p_m-p_{52} debe mantenerse en sus recíprocos. Para esto, se definen las distancias d_1 y d_2 en las Ecuaciones 3.13 y 3.14:

$$\mathrm{d}_1 = \sqrt{\left(i_{p_{40}} - i_{p_{43}}\right)^2 + \left(j_{p_{40}} - j_{p_{43}}\right)^2} \tag{3.13}$$

$$\mathrm{d}_2 = \sqrt{\left(i_{p_{52}} - i_{p_m}\right)^2 + \left(j_{p_{52}} - j_{p_m}\right)^2} \tag{3.14}$$

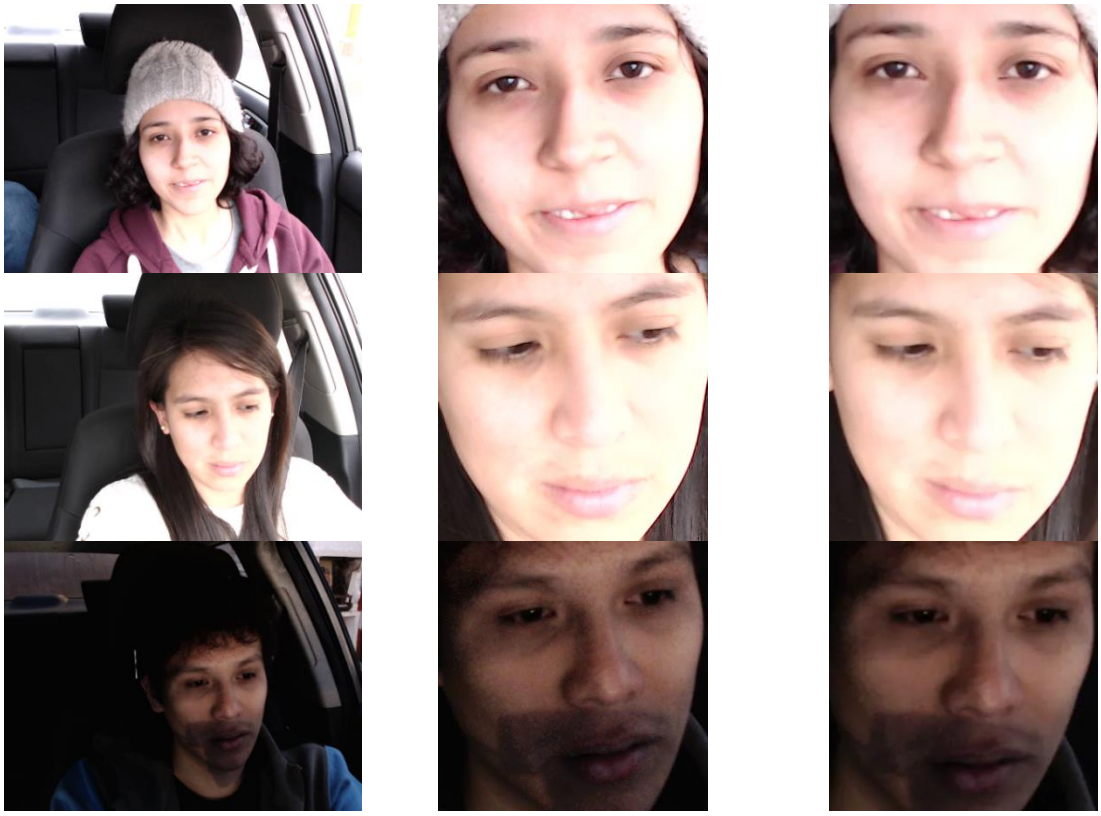
Donde, a partir de estas distancias, se definen las coordenadas de los puntos recíprocos de p_{40}, p_{43} y p_{52} en las Ecuaciones 3.15, 3.16 y 3.17:

$$\left(i_{p'_{40}}, j_{p'_{40}}\right) = \left(i_{p_{40}}, j_{p_{40}}\right) \tag{3.15}$$

$$\left(i_{p'_{43}}, j_{p'_{43}}\right) = \left(i_{p_{40}} + \mathrm{d}_1, j_{p_{40}}\right) \tag{3.16}$$

$$\left(i_{p'_{52}}, j_{p'_{52}}\right) = \left(i_{p_{40}} + \frac{\mathrm{d}_1}{2}, j_{p_{40}} + \mathrm{d}_2\right) \tag{3.17}$$

Una vez definidas las coordenadas de los puntos p_{40}, p_{43} y p_{52} y sus recíprocos p'_{40}, p'_{43} y p'_{52}, se calculan los coeficientes de la transformación afín $\mathrm{T_{af}}$ que genera la correspondencia entre sus sistemas de coordenadas. Como último paso de la etapa de la corrección geométrica, se aplica la transformada afín a todos los pixeles de las imágenes que contienen un rostro, obteniendo así las imágenes de rostro corregidas $I_{geocRGB}$ (para imágenes sin pre-procesar) e I_{geocpp} (para imágenes pre-procesadas). En la Figura 3.26 se muestran los resultados de la etapa de corrección geométrica.



(a) (b) (c)

Figura 3.26. Resultados de la corrección geométrica: (a) imágenes de la base de datos, (b) rostro detectado y (c) rostro tras aplicar la corrección geométrica.

3.4.6. Codificación de rostro

La etapa de codificación de rostro tiene como objetivo el de sintetizar las imágenes de rostro corregidas ($I_{geocRGB}$ y I_{geocpp}) en un vector de características representativo. Dicho vector permitirá cuantificar las características de un rostro.

Para esta etapa se utilizará una red neuronal convolucional (CNN), la técnica que actualmente imparte el estado del arte en el área de identificación de rostros. Sin embargo, para poder entrenar una CNN se requiere de una cantidad considerable de imágenes (en el rango de los cientos de miles). Por esta razón, en el presente trabajo se ha optado por utilizar el recurso de transferencia de conocimiento (*transfer learning*). Con esta finalidad, se realizó la descarga de un modelo de CNN ya entrenado basado en la arquitectura de FaceNet (Schroff et al. [23]). Este modelo fue entrenado utilizando la base de datos CASIA-WebFace (Yi et al. [30]), la cual consta de 453'453 imágenes de un total de 10'575 personas distintas. En la Tabla 3.5 se muestra de manera detallada la arquitectura la de CNN anteriormente mencionada, siendo *conv#* utilizado para hacer referencia a una capa de convolución, *pool#* a una capa de *pooling*, *fc#* a una capa de conectividad total (*fully*

connected) y *L2* la normalización aplicada al vector de características. Además, # hace mención al número o identificado de cada capa en la nomenclatura explicada.

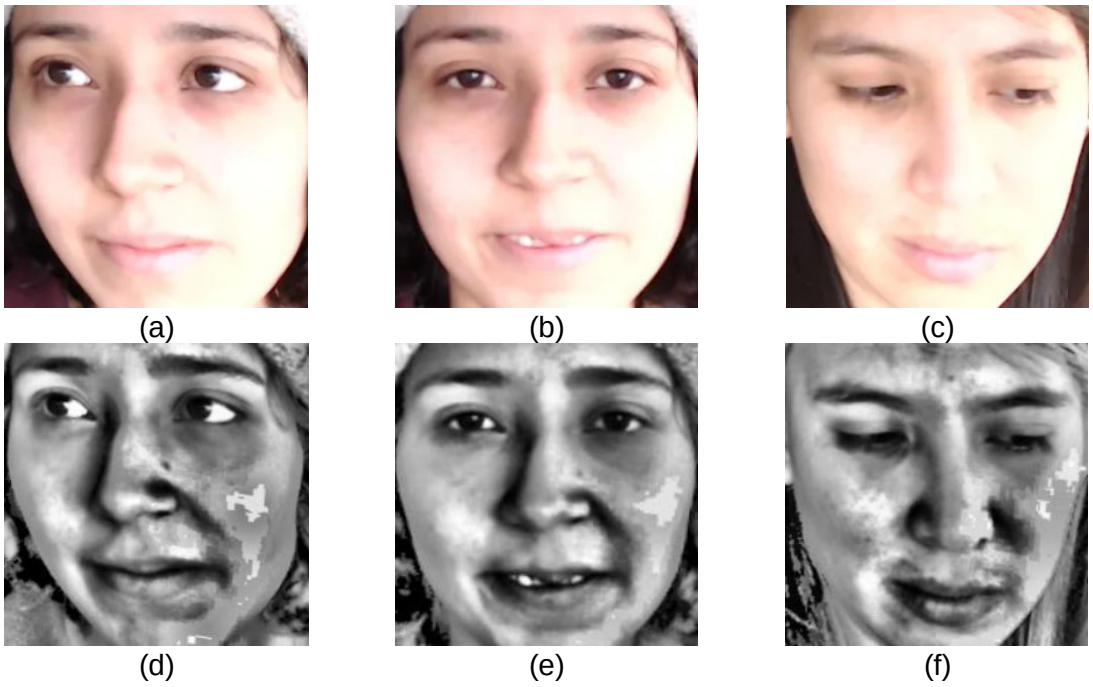
Tabla 3.5. Arquitectura de la CNN *FaceNet* [23].

CAPA	Dim. Entrada	Dim. Salida	*kernel*	param	FLPS
conv1	220x220x3	110x110x64	7x7x3, 2	9K	115M
pool1	110x110x64	55x55x64	3x3x64, 2	0	
rnorm1	55x55x64	55x55x64		0	
conv2a	55x55x64	55x55x64	1x1x64, 1	4K	13M
conv2	55x55x64	55x55x192	3x3x64, 1	111K	335M
rnorm2	55x55x192	55x55x192		0	
pool2	55x55x192	28x28x192	3x3x192, 2	0	
conv3a	28x28x192	28x28x192	1x1x192, 1	37K	29M
conv3	28x28x192	28x28x384	3x3x192, 1	664K	521M
pool3	28x28x384	14x14x384	3x3x384, 2	0	
conv4a	14x14x384	14x14x384	1x1x384, 1	148K	29M
conv4	14x14x384	14x14x256	3x3x384, 1	885K	173M
conv5a	14x14x256	14x14x256	1x1x256, 1	66K	13M
conv5	14x14x256	14x14x256	3x3x256, 1	590K	116M
conv6a	14x14x256	14x14x256	1x1x256, 1	66K	13M
conv6	14x14x256	14x14x256	3x3x256, 1	590K	116M
pool4	14x14x256	7x7x256	3x3x256, 2	0	
concat	7x7x256	7x7x256		0	
fc1	7x7x256	1x32x128		103M	103M
fc2	1x32x128	1x32x128		34M	54M
fc7128	1x32x128	1x1x128		542K	0.5M
L2	1x1x128	1x1x128		0	
Total				140M	1.6B

Siendo *Dim. Entrada,* la dimensión del volumen de entrada en cada capa, Dim. Salida, la dimensión del volumen de salida en cada capa, *kernel* las dimensiones del *kernel* de convolución utilizado (alto×ancho×profundidad, paso; siendo este último el paso de desplazamiento del operador de convolución), *param,* el número de parámetros o pesos en cada capa y *FLPS,* el número de FLOPS teóricos de cada capa (cantidad de operaciones). Después de cada capa de convolución la CNN utiliza una función de rectificación lineal (ReLU, Tabla 2.2) como función de activación y en cada capa de *pooling* utiliza la función de *maxpooling* (Tabla 2.3).

Para continuar entrenando esta red se definen “ternas de entrenamiento” a partir de las imágenes de rostro corregidas ($I_{geocRGB}$ y I_{geocpp}). Las ternas de entrenamiento consisten en 3 imágenes de rostros: una imagen de rostro, una imagen de refuerzo positivo (que pertenece a la misma clase que el rostro) y una imagen de refuerzo negativo (que pertenece a una clase distinta). Sin embargo, en la Tabla 3.5 se puede apreciar que la imagen de entrada a la CNN es una imagen de dimensión 220x220x3 (en RGB), por lo que se debe realizar un re-escalamiento de la imagen $I_{geocRGB}$ y, por otro lado, generar una

imagen en RGB a partir de I_{geocpp} para después re-escalarla. En la Figura 3.27 se ilustran ternas de entrenamiento generadas con las imágenes $I_{geocRGB}$ y I_{geocpp}.



(a) (b) (c)

(d) (e) (f)

Figura 3.27. Ejemplo de terna de entrenamiento para FaceNet sin pre-procesar y pre-procesadas: [(a) y (d)] imágenes de rostro, [(b) y (e)] imágenes de refuerzo positivo e [(c) y (f)] imágenes de refuerzo negativo

Para el entrenamiento de una CNN se utilizan 4’500 imágenes de rostros corregidas y re-escaladas de 30 personas, generándose un total de 12’000 ternas de entrenamiento. Como se comentó anteriormente, el entrenamiento se realiza únicamente sobre las capas de conectividad total (fc1 y fc2 de la Tabla 3.5) debido a que se está utilizando el recurso de transferencia de conocimiento.

Por último, la Tabla 3.5 indica que la salida de una CNN con arquitectura FaceNet es un vector de 128 dimensiones. En paralelo, se entrenan 2 modelos de CNN con la arquitectura y estrategia anteriormente mencionadas, uno a partir de imágenes sin pre-procesar y otro a partir de imágenes pre-procesadas, siendo las salidas los vectores de rostro codificado $\mathbf{v}_{faceRGB}$ y $\mathbf{v}_{facepp}$ respectivamente.

3.4.7. Validación de rostro

La etapa de validación de rostro tiene como objetivo la identificación de conductores en base a la codificación de rostros abordados en la etapa anterior. Esta es la última etapa del algoritmo propuesto en el presente trabajo, y es en donde se evaluará el rendimiento de todas las etapas combinadas.

Para realizar la identificación de los rostros se utilizará una máquina de soporte vectorial (SVM) debido a que, como se mencionó anteriormente, este modelo encuentra una solución óptima (convergencia global) en los problemas de clasificación para un conjunto de datos determinado.

La entrada al SVM que realizará la identificación de rostros es el vector de rostro codificado ($\mathbf{v}_{faceRGB}$, $\mathbf{v}_{facepp}$) obtenido en la etapa anterior. Por otro lado, la salida de este modelo será la etiqueta y/o código asignado a cada persona en la base de datos adquirida para el presente trabajo. Estas etiquetas o códigos son asignados a las personas mas no al tipo de imagen (referente a sus condiciones de iluminación) de la cual proceden los vectores de rostro codificados.

Para el entrenamiento del SVM se generó una base de datos de 7617 vectores de rostro codificado, ligados a sus correspondientes etiquetas $y_{\text{face}} \in \{1,2,\dots,30\}$. Esta base de datos se separó en 6’094 vectores de entrenamiento y 1’525 vectores de prueba.

A diferencia de la etapa de detección de rostros, en esta etapa no existe un consenso y/o claro estándar en cuanto a los parámetros de la SVM. Además, se considera que la complejidad del modelo de clasificación para la identificación de rostros es mayor al de la detección. Por esta razón, para encontrar los parámetros adecuados del modelo de clasificación se entrenan múltiples SVM, con diferentes valores del coeficiente de margen suave (C) y *kernels* (K). En el caso del *kernel* de base radial, se utilizan distintos tipos de desviación estándar (σ_{k}). En las Tablas 3.6 y 3.7 se muestra el rendimiento de los modelos entrenados utilizando los vectores de rostro codificado $\mathbf{v}_{faceRGB}$ y $\mathbf{v}_{facepp}$ respectivamente.

Tabla 3.6. Asertividad [%] del clasificador de rostros en base a distintos parámetros de entrenamiento utilizando imágenes sin pre-procesar

Kernel	**Base Radial**					**Lineal**
C\ σ_k	0.001	0.01	0.1	1	10	-
0.001	7.70	7.70	7.70	7.70	7.70	7.70
0.01	7.70	7.70	7.70	40.27	24.18	16.58
0.1	7.70	7.70	54.22	97.70	94.71	96.68
1	7.70	55.34	97.86	98.72	98.45	98.45
10	55.34	97.86	98.72	**99.20**	98.56	99.09

De la Tabla 3.6 se aprecia que la mejor opción es la máquina de soporte vectorial entrenada con un parámetro de margen suave (C) igual a 10, un kernel del tipo base radial con una desviación estándar (σ_k) igual a 1.

Tabla 3.7. Asertividad [%] del clasificador de rostros en base a distintos parámetros de entrenamiento utilizando imágenes pre-procesadas

Kernel	**Base Radial**					**Lineal**
C\ σ_k	0.001	0.01	0.1	1	10	-
0.001	8.32	8.32	8.32	8.32	8.32	8.32
0.01	8.32	8.32	8.32	18.51	17.33	16.69
0.1	8.32	8.32	26.72	93.60	91.15	89.92
1	8.32	28.00	94.19	96.48	96.75	96.05
10	28.05	94.29	96.32	**97.23**	96.85	96.85

De la Tabla 3.7 se aprecia que la mejor opción es la máquina de soporte vectorial entrenada con un parámetro de margen suave (C) igual a 10, un *kernel* del tipo base radial con una desviación estándar (σ_k) igual a 1.

3.5. Comentarios finales del capitulo

En el presente capitulo se expuso el desarrollo del trabajo de investigación, desde el tipo, nivel y diseño de investigación hasta la propuesta de la metodología y su implementación. Cabe resaltar que en este capítulo se realizó la sintonización de los parámetros de los métodos explicados en el fundamento teórico y se presentó el flujo que siguen las imágenes.

CAPÍTULO IV
PRUEBAS DE VALIDACIÓN Y RESULTADOS

En el presente capítulo se procederá a exponer el análisis y resultados del presente trabajo de Tesis. Para esto, se comenzará por describir los escenarios en los cuales se realizaron las pruebas y continuará con la descripción de las medidas de validación. En esta última, cabe resaltar que las métricas contrastaran los valores obtenidos por los algoritmos con los resultados estimados por una persona (*gold estandard*). Después, se cuantificará el desempeño del algoritmo en las tareas de detección e identificación de rostros, basado en los escenarios de prueba y métricas pertinetnes. Además, se realizará un análisis del efecto de las estrategias de pre-procesamiento en las salidas de la etapa de codificación de rostro. Finalmente, el capítulo terminara con la contratación de la hipótesis general y las hipótesis especificas planteadas en el Capítulo I.

4.1 Descripción de los escenarios de pruebas

El presente trabajo de Tesis tiene como entrada imágenes digitales adquiridas en la cabina del conductor de un automóvil utilizando una cámara web cuyas características se mencionan en la Tabla 3.1. Estas imágenes digitales forman una base de datos, la cual tiene como objetivo captar la mayor cantidad de condiciones de iluminación y la mayor cantidad de poses y/u orientaciones de rostros (como se muestra en las Figuras 3.8 y 3.9).

Como se muestra en la Tabla 3.2, las imágenes adquiridas para el presente trabajo de investigación son categorizadas en 3 clases basadas en la hora de adquisición: imágenes TIPO 1, TIPO2 y TIPO 3. En general, estas clases presentan las siguientes características:

- **Imagen TIPO 1**: este grupo de imágenes abarca aquellos escenarios que presentan un nivel elevado de iluminación natural externa y que, para fines prácticos, es considerada constante en toda la imagen. Además, idealmente, en este tipo de imágenes no existe

la presencia de sombras en el rostro. Sin embargo, en ciertos casos puede presentarse la saturación del sensor para niveles de iluminación externa elevados.

- **Imagen TIPO 2**: este grupo de imágenes abarca aquellos escenarios que presentan un nivel moderado de iluminación natural externa y que, en consecuencia, pueden generar regiones con distintos niveles de iluminación en un rostro (aparición de sombras). En este tipo de imágenes, las fuentes de iluminación artificial (luz de la cabina, luces de otros vehículos, alumbrado exterior, etc.) comienzan a tener impacto en la iluminación de las regiones del rostro.
- **Imagen TIPO 3**: este grupo de imágenes abarca aquellos escenarios que poseen un nivel bajo de iluminación natural externa y que, por consiguiente, las fuentes de iluminación artificial (luz de la cabina, luces de otros vehículos, alumbrado exterior, etc.) tienen un mayor impacto en los niveles de iluminación de las regiones de un rostro. Las imágenes que pertenecen a esta clase pueden presentar iluminación total, parcial o nula del rostro.

En general, en la metodología del presente trabajó de Tesis se plantea que cada tipo de imagen sea pre-procesada utilizando una estrategia distinta (Tabla 3.4). En la Figura 4.1 se muestra de forma gráfica lo anteriormente mencionado:

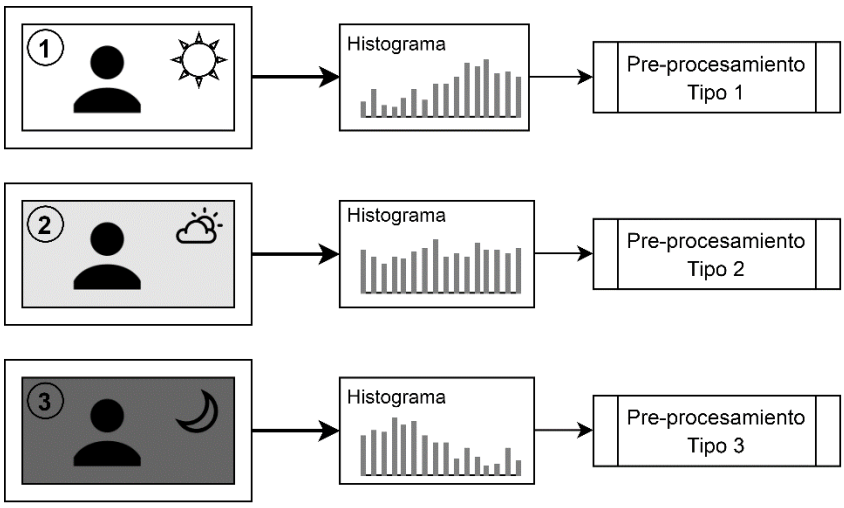


Figura 4.1. Estrategia de pre-procesamiento en la metodología propuesta.

Además de captar distintos escenarios de iluminación, la base de datos generada en el presente trabajo de investigación se adquirió teniendo en cuenta lo siguiente:

- Distintos tipos de orientación y/o pases del rostro de los conductores.
- El uso de anteojos con resina transparente para algunos conductores.
- Para imágenes de TIPO 3, distintos tipos de iluminación artificial.

- Casos en los cuales no es posible realizar la detección o identificación de rostros.

En la Figura 4.2 se muestran imágenes de la base de datos generada para el presente trabajo, agrupadas en las clases TIPO 1, TIPO 2 y TIPO; que presentan las características mencionadas anteriormente.

Figura 4.2. Imágenes de la base de datos generada para el trabajo de Tesis, agrupadas en imágenes (a, d) TIPO 1, (b, e) TIPO 2 y (c, f) TIPO 3, siendo las imágenes (a), (b) y (c) ejemplos de la Figura 3.8 y (d), (e) y (f) ejemplos de la Figura 3.9.

Cabe resaltar que las imágenes ubicadas en la parte derecha de la Figura 4.2 corresponden a imágenes en las cuales se considera inviable la detección e identificación de un rostro. Sin embargo, algunas de estas imágenes ingresan al entrenamiento y validación de resultados de la presente Tesis.

4.2 Medidas de validación

En la presente sección se procederá a describir las métricas utilizadas para cuantificar el rendimiento del algoritmo propuesto en la presente Tesis y poder contrastar las hipótesis propuestas por el mismo. Además, se procederá a dar una descripción breve de la interpretación de cada métrica utilizada.

Las métricas explicadas a continuación están relacionadas a los resultados de modelos de clasificación. Para poder continuar con la descripción de estas métricas, Fawcett [6] define las siguientes variables:

- VP: Verdadero positivo, esta variable hace mención al número de casos en los que el modelo de clasificación realiza una estimación acerada de un caso positivo (i.e.: el número de rostros detectados correctamente).
- VN: Verdadero negativo, esta variable hace mención al número de casos en los que modelo de clasificación rechaza de manera acertada un caso negativo (i.e.: el número de cuadros que no tienen un rostro y son clasificados como tal).
- FP: Falso positivo o tipo de error I, esta variable hace mención al número de casos en los cuales el modelo de clasificación estima de manera incorrecta un elemento como caso positivo (i.e.: el número de recuadros clasificados como rostro pero que no los contienen).
- FN: Falso negativo o tipo de error II, esta variable hace menciona al número de casos en los cuales el modelo de clasificación rechaza elementos que fueron etiquetados como positivos (i.e.: el número de rostros que no fueron detectados).

4.2.1 Sensibilidad

La sensibilidad es una métrica que puntúa únicamente la capacidad del modelo de clasificación de estimar correctamente los casos positivos (verdaderos positivos) del total de casos positivos [6]. En la Ecuación 4.1 se muestra el cálculo de dicha métrica:

$$\text{sensibilidad}[\%] = \frac{\text{VP}}{\text{VP} + \text{FN}} \times 100\% \tag{4.1}$$

Fawcett [6] remarca que un alto grado de sensibilidad indica que los elementos clasificados como positivos por el modelo de clasificación tienen una alta posibilidad de

que realmente lo sean. Dicho de otra manera, es poco probable que el modelo cometa errores del tipo I (falso positivo).

4.2.2 Especificidad

La especificidad es una métrica que puntúa únicamente la capacidad del modelo de clasificación de estimar correctamente los casos negativos (verdaderos negativos) del total de casos negativos [6]. En la Ecuación 4.1 se muestra el cálculo de dicha métrica:

$$\text{especificidad}[\%] = \frac{\text{VN}}{\text{FP} + \text{VN}} \times 100\% \tag{4.2}$$

Fawcett [6] remarca, a diferencia de la sensibilidad, un alto grado de especificidad indica que los elementos clasificados como negativos por el modelo de clasificación tienen una alta posibilidad de que realmente lo sean. Dicho de otra manera, es poco probable que el modelo cometa errores del tipo II (falso negativo).

4.2.3 Asertividad

La asertividad es una métrica que puntúa de manera general la capacidad del modelo de clasificación a estimar clases correctamente y/o a no equivocarse [6]. Esta métrica, como indica la Ecuación 4.3, se calcula teniendo en cuenta todos los casos correctos (VP + VN):

$$\text{asertividad}[\%] = \frac{\text{TP} + \text{TN}}{\text{VP} + \text{FP} + \text{VN} + \text{FN}} \times 100\% \tag{4.3}$$

Fawcett [6] remarca que un alto grado de asertividad indica que la clase estimada por el modelo de clasificación tiene una posibilidad de ser la correcta. Esto quiere decir que es poco probable que el modelo cometa errores del tipo I y II.

4.2.4 AUC – Área bajo la curva ROC

La curva ROC (del inglés *Receiver Operating Characteristics*) es una técnica que permite visualizar, organizar y comparar modelos de clasificación basados en su rendimiento. En esta curva representa el equilibrio entre la razón de verdaderos positivos (representado por la sensibilidad) frente a la razón de falsos positivos (representado por el complemento de la especificidad) al cambiar el umbral de discriminación del modelo. Esta curva empezó utilizándose para el análisis médico, pero en los últimos años su uso se ha ido incrementando en la evaluación de modelos de *Machine Learning* [6].

En la Figura 4.3 se muestra un ejemplo del grafico ROC que contiene la información de cinco clasificadores distintos con un umbral determinado.

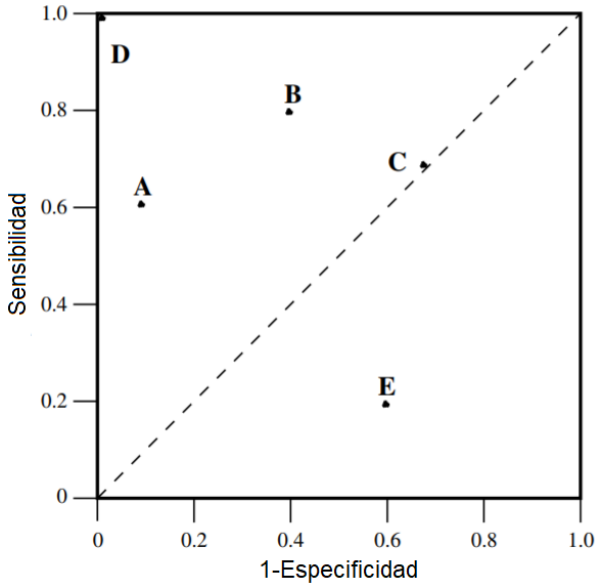


Figura 4.3. Gráfico ROC que muestra clasificadores discretos. Traducido de [6].

Fawcett [6] realiza la siguiente interpretación de la Figura 4.3:

- El punto A: corresponde a un clasificador presenta tasas bajas de falsos positivos y negativos en su clasificación.
- El punto B: corresponde a un clasificador con tasa bajas de falsos positivos y negativos en su clasificación, pero con mayores tendencias a dejar pasar falsos negativos que A.
- El punto C: corresponde a un clasificador trivial, los clasificadores pertenecientes a la línea punteada tienen una asertividad de 0.5 (si fuese clasificación binaria, tendría el mismo rendimiento que un clasificador aleatorio)
- El punto D: corresponde a un clasificador ideal, no comete errores de clasificación.

- El punto E: corresponde a un clasificador ineficiente, no recomendable bajo ningún parámetro.

Sin embargo, el ejemplo mostrado en la Figura 4.3 puede prestarse a interpretaciones subjetivas. Por esta razón, se introduce la métrica del área bajo la curva AUC (del inglés *Area Under the Curve*), que permite realizar una comparación numérica entre clasificadores. Para realizar esto, comienza a variar el nivel de umbralizacion (que generalmente tiene un valor de 0.5). Al alterar el nivel de umbralización, el clasificador presenta distintos niveles de sensibilidad y especificidad, generando así una curva como la que se muestra en la Figura 4.4.

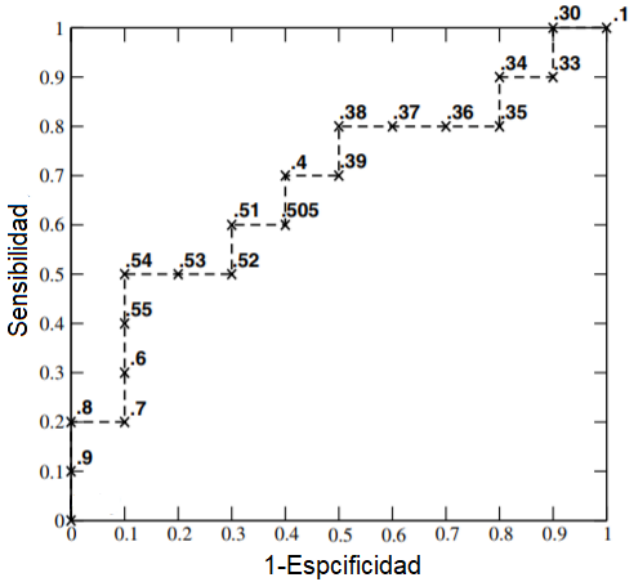


Figura 4.4. Curva ROC para un clasificador. Traducido de [6].

Finalmente, como su nombre indica, AUC viene a ser el área bajo la curva ROC, interpretándose como un clasificador más exacto a aquel que posea un valor de AUC mayor. De la Figura 4.4 se puede apreciar también que el valor máximo que puede la métrica AUC es de 1.

4.3 Resultados de la detección e identificación de rostros

Como se mencionó en el sub-capitulo 3.4, para la detección de rostro se entrenaron 2 máquinas de soporte vectorial para que puedan detectar rostros en imágenes pre-procesadas (I_{2pp}) y en imágenes en escala de grises (I_{2GRAY}).

El objetivo de realizar el entrenamiento de estos 2 modelos de clasificación fue el de evaluar en qué casos conviene aplicar estrategias de pre-procesamiento, ya que en el estado del arte se presentan resultados con bastante exactitud. En la Tabla 4.1 se muestran las métricas de sensibilidad y especificidad obtenidas al realizar el entrenamiento de las máquinas de soporte vectorial a partir de información de imágenes sin pre-procesar (I_{2GRAY}) e imágenes pre-procesadas (I_{2pp}), en ambos casos utilizando 15'494 vectores xHOG de entrenamiento y 3'874 vectores $\mathbf{xHOG}$ de prueba.

Tabla 4.1. Resultados de la detección de rostro en imágenes.

	sin pre-procesamiento		**con pre-procesamiento**	
IMAGEN	sensibilidad [%]	especificidad [%]	sensibilidad [%]	especificidad [%]
TIPO 1	**86.37**	**98.92**	76.52	97.61
TIPO 2	**79.24**	**99.69**	66.07	94.54
TIPO 3	54.12	99.69	**71.98**	**97.65**

En donde se puede apreciar que, en las imágenes de TIPO 1 y TIPO 2, las estrategias de pre-procesamiento disminuyen la sensibilidad y especificidad de los resultados obtenidos aplicando técnicas del estado del arte. Sin embargo, si bien en las imágenes TIPO 3 se aprecia una leve disminución en la especificidad, la sensibilidad de la detección de rostros es bastante superior en el caso del algoritmo entrenado con imágenes pre-procesadas.

En cuanto al modelo de identificación de rostro, también se realizó el entrenamiento de 2 máquinas de soporte vectorial que evalúen la información obtenida a partir de redes neuronales convolucionales (los vectores de rostro codificado $\mathbf{v}_{facepp}$ y $\mathbf{v}_{faceRGB}$) entrenadas con imágenes de rostro pre-procesadas (I_{geocpp} genera el vector de rostro codificado $\mathbf{v}_{facepp}$) e imágenes de rostro sin pre-procesar ($I_{geocRGB}$ genera el vector de rostro codificado $\mathbf{v}_{faceRGB}$). Estos algoritmos de clasificación se entrenaron utilizando vectores de rostro 6'094 vectores de entrenamiento y 1'525 vectores de rostro codificado de prueba, siendo los parámetros de entrenamiento obtenidos a partir de las Tablas 3.6 y 3.7.

Si bien se evalúa la asertividad de los modelos de clasificación, esta evaluación se realiza de manera general, sin tener en cuenta los tipos de imagen. Por esta razón, en la

Tabla 4.2 se presenta la asertividad de ambos modelos evaluada con respecto a los distintos tipos de imágenes (TIPO 1, TIPO 2 y TIPO 3).

Tabla 4.2. Resultados de la identificación de rostros por tipo de imágenes.

IMAGEN	**sin pre-procesamiento** asertividad [%]	**con pre-procesamiento** asertividad [%]
TIPO 1	**99.75**	99.15
TIPO 2	**99.24**	97.21
TIPO 3	**98.38**	95.39

En la Tabla 4.2 se puede apreciar que los niveles de asertividad obtenidos procesando vectores de rostro codificado de imágenes sin pre-procesar ($\mathbf{v}_{faceRGB}$) es mayor al de los obtenidos a partir de imágenes pre-procesadas ($\mathbf{v}_{facepp}$).

En la Figura 4.5 se muestran las curvas ROC de ambos modelos de clasificación entrenados.

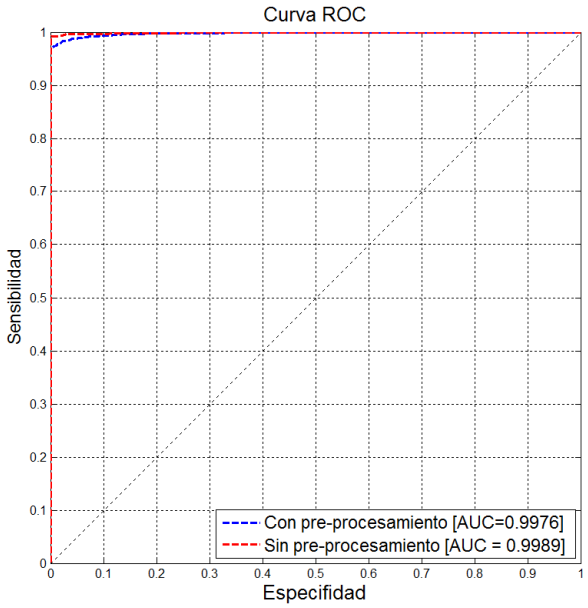


Figura 4.5. Curva ROC de la validación de rostros.

En la Figura 4.5 se puede apreciar que las métricas AUC para los modelos utilizando los vectores de rostro pre-procesados y sin pre-procesar son 0.9976 y 0.9989 respectivamente. En base a lo planteado por [6], el mejor clasificador es aquel que fue entrenado con vecotores de codificación de rostro sin pre-procesar ($\mathbf{v}_{faceRGB}$).

Finalmente, cabe resaltar que las métricas obtenidas en la identificación de rostro se obtuvieron a partir de las imágenes de rostro obtenidas en el proceso de detección de rostro, por lo que la asertividad se podría considerar como la probabilidad condicional de identificar un rostro dado que se detectó un rostro.

4.4 Análisis del impacto del pre-procesamiento en la codificación de rostros

En la presente sección, se procederá a hacer un análisis del impacto de las estrategias de pre-procesamiento en la codificación de rostro mediante una red neuronal convolucional con arquitectura *FaceNet*.

Como se mencionó en la sección de implementación del pre-procesamiento (sub-capítulo 3.3.2), los parámetros de pre-procesamiento (γ y r_f, que se muestran en la Tabla 3.4) se obtuvieron a través de la inspección visual, tras un proceso de ensayo y error como se muestra en las Figuras 3.11, 3.12, 3.13 y 3.14. Sin embargo, si bien este proceso de sintonización de parámetros tiene sentido para el juicio experto de un humano, no necesariamente impacta de forma positiva en los resultados de la codificación de rostro.

Por esta razón, y a modo de contribuir con la interpretabilidad del impacto de las estrategias de pre-procesamiento en la codificación de rostro (y en el algoritmo propuesto), se realizó un contraste de los vectores de rostro codificado pre-procesados ($\mathbf{v}_{facepp}$) y sin pre-procesar ($\mathbf{v}_{faceRGB}$) , el cual incluye además los tipos de imágenes de la base de datos (TIPO 1, TIPO 2 y TIPO 3).

Para esto, se parte de la premisa de la estrategia de pre-procesamiento: estandarizar la distribución de valores de pixeles para un rostro, independientemente de las condiciones de iluminación. Esta estandarización consiste en que, para un mismo rostro, las condiciones de iluminación no afecten a la salida de la red neuronal convolucional (vector de rostro codificado).

La red neuronal convolucional utilizada en el presente trabajo de Tesis genera un vector de 128 dimensiones, el cual caracteriza a un rostro. Si bien la evaluación del entrenamiento de esta red se evaluará en la etapa de validacion de rostro, se utilizó la tecnica t-SNE (Maaten y Hinton [19]) para realizar una inspección visual de los resultados de la red neuronal convolucional. Las Figuras 4.6 y 4.7 se muestran las proyecciones de las salidas de las redes neuronales convolucionales entrenadas utilizando las imágenes

sin pre-procesar y pre-procesadas respectivamente, de un espacio de dimensionalidad 128 a un espacio de dimensionalidad 2.

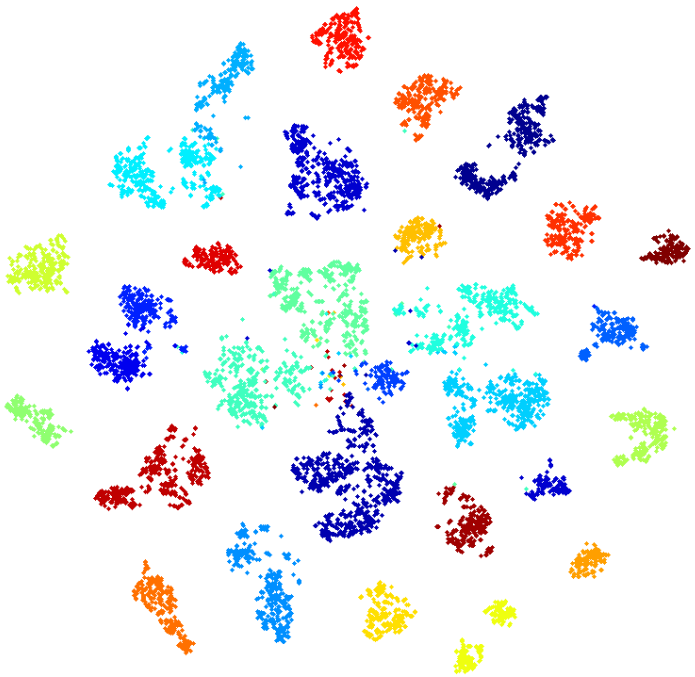


Figura 4.6. Visualización de la codificación de rostro en imágenes $I_{geocRGB}$ utilizando t-SNE.

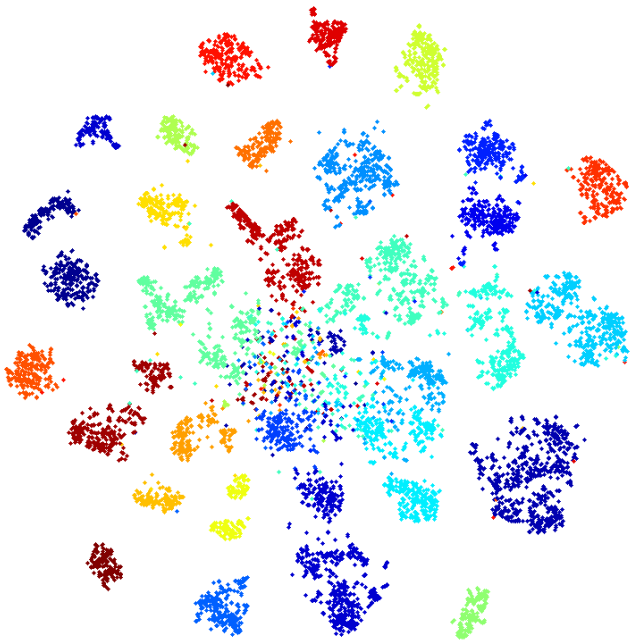


Figura 4.7. Visualización de la codificación de rostro en imágenes I_{geocpp} utilizando t-SNE.

En donde cada color de los puntos de las Figuras 4.6 y 4.7 representa la codificación del rostro de una misma persona. Además, se puede apreciar en las Figuras los puntos de cierto color están en zonas bastante específicas y diferenciables, mostrando que pueden ser separables por un algoritmo de clasificación. Por otro lado, para ser más específicos en el impacto del pre-procesamiento, en la Figura 4.8 se muestran las proyecciones de algunas etiquetas basados en el tipo de imagen (TIPO 1, TIPO 2 y TIPO 3).

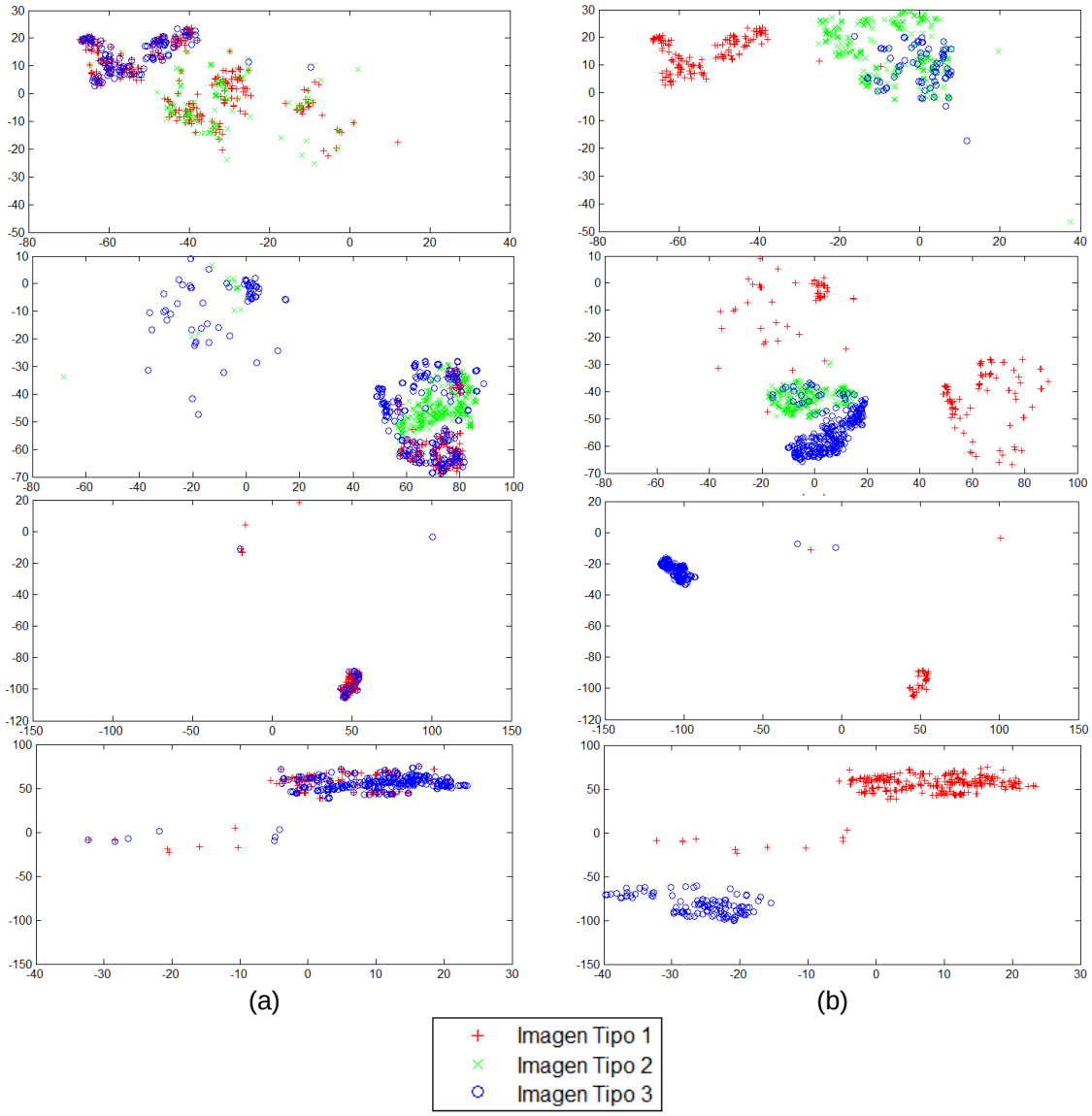


Figura 4.8. Comparación de la distribución de los tipos de imágenes en la visualización t-SNE para una misma etiqueta: (a) imágenes pre-procesadas, (b) imágenes sin pre-procesar.

En la Figura 4.8 se puede apreciar que, para una misma etiqueta/correspondientes a un mismo conductor (graficas en la misma fila) el nivel de dispersión de los puntos es mayor en los vectores de codificación de rostro obtenidos a partir de imágenes sin pre-procesar.

4.5 Contrastación de hipótesis

A continuación, en función a las métricas y resultados obtenidos en el presente trabajo de investigación, se procederá a realizar la contrastación de las hipótesis formuladas por la presente Tesis.

4.5.1 Contraste de hipótesis específica 1

La primera hipótesis específica propone que “mediante la atenuación del efecto de iluminación en las imágenes capturadas en la cabina del conductor utilizando estrategias de pre-procesamiento para distintos escenarios de iluminación, es posible obtener nuevas imágenes en donde el contraste de las regiones que contienen un rostro no dependa de las condiciones de iluminación inicial”. En la Figura 3.14 se puede apreciar que visualmente, las regiones de los rostros se encuentran en condiciones similares. Además, en la Figura 4.8 (b) se muestra que la distribución de los vectores de rostro codificado se reduce en las imágenes pre-procesadas.

Por estas razones se concluye que se comprueba la hipótesis especifica 1.

4.5.2 Contraste de hipótesis específica 2

Ante la hipótesis específica 2: “las imágenes pre-procesadas pueden resaltar las características generales de un rostro, permitiendo realizar una detección de rostro de manera automática, con una sensibilidad superior al 70% y una especificidad superior al 90%”. Se puede apreciar de la Tabla 4.1 que la sensibilidad de las imágenes pre-procesadas del TIPO 1 (86.37%), TIPO 2 (79.24%) y TIPO 3 (79.24%) superan el 70%, por otro lado, la sensibilidad de todos los tipos de imágenes (TIPO 1: 98.92%, TIPO 2: 99.69%, TIPO 3: 97.65%) superan el 90%.

Por estas razones se concluye que se comprueba la hipótesis especifica 2.

4.5.3 Contraste de hipótesis específica 3

Ante la hipótesis específica 3: “las imágenes pre-procesadas pueden resaltar los detalles de un rostro, permitiendo realizar una identificación de rostro de manera automática, con una asertividad mayor al 90% y un valor de AUC superior a 0.85”. En la

Tabla 4.2 se puede observar que la asertividad de todos los tipos de imágenes (TIPO 1: 99.75%, TIPO 2: 99.24%, TIPO 3: 98.38%) superan el 90%. Además, en la Figura 4.5 se puede observar que el valor de AUC de la curva ROC es de 0.9989, superando el 0.85.

Por estas razones se concluye que se comprueba la hipótesis específica 3.

4.5.4 Contraste de hipótesis general

La hipótesis general de la presente Tesis es: “Una imagen obtenida a través de una cámara web, colocada en la cabina de un conductor bajo condiciones de iluminación no controlada, mediante el uso de técnicas de procesamiento digital de imágenes y visión computacional, genera información suficiente para realizar la detección e identificación de un rostro”. Habiéndose contrastado las hipótesis especificas concernientes a las estrategias de pre-procesamiento, detección de imágenes e identificación de rostros en base a lo señalado en la Tabla 1.1, se comprueba la hipótesis general.

4.6 Comentarios finales del capitulo

En el presente capítulo se realizó un análisis de los escenarios de validación y los resultados obtenidos por el algoritmo evaluados según las métricas expuesta.

Cabe resaltar que, como era de esperarse, los resultados al entrenar los algoritmos del estado del arte obtuvieron una mayor puntuación que los entrenados con las imágenes pre-procesadas. Sin embargo, en las imágenes TIPO 3 se puede apreciar una mejora en la combinación de sensibilidad y especificidad en la detección de rostros. Por esta razón, se sugiere utilizar el pre-procesamiento únicamente para la detección de rostros en las imágenes TIPO 3, las cuales son tomadas en el horario señalado en la Tabla 3.2.

CONCLUSIONES

Se desarrolló un algoritmo de detección y validación de rostros de conductores de vehículos terrestres, utilizando técnicas de procesamiento de imágenes, visión computacional y *deep learning*. Este algoritmo se puede desglosar en 2 componentes fundamentales: etapa de detección de rostro y etapa de validación de rostro.

En la etapa de detección de rostro se implementaron las técnicas de visión computacional y aprendizaje automatizado que marcan el estado del arte en cuanto al problema de la detección de rostro. Además, se implementaron estrategias de pre-procesamiento como se sugiere en la Figura 4.1, analizando y contrastando sus repercusiones en la detección de rostros con respecto al estado del arte. Además, en la Tabla 3.4 se muestra que los parámetros de las 2 primeras estrategias de pre-procesamiento son iguales, de lo cual se puede concluir que para la base de datos generada, son necesarias solamente 2 estrategias de pre-procesamiento.

Como se puede apreciar en la Tabla 4.1, las técnicas de pre-procesamiento mejoran únicamente la detección de rostros en las imágenes de noche. Sin embargo, no utilizar técnicas de pre-procesamiento sería una restricción bastante grave para trabajar con imágenes de noche. Para acercarse a las métricas establecidas como satisfactorias (sensibilidad del 80% y especificidad del 90%) la mejor alternativa para la detección de rostros es utilizar únicamente el pre-procesamiento en las imágenes de TIPO 3. Si bien no hay problema en la especificada, la sensibilidad solo llega al umbral establecido para las imágenes de TIPO 1 sin ser pre-procesadas. Sin embargo, con la finalidad de monitorear a un conductor se busca priorizar la especificidad que la sensibilidad.

En la componente de validación de rostros se implementaron técnicas de visión computacional, aprendizaje automatizado y aprendizaje profundo que marcan el estado del arte en la identificación de rostros. Comenzando con las correcciones geométricas, en la Figura 4.9 se observa de manera visual que el pre-procesamiento genera un efecto

negativo en la estimación de los vectores de forma de un rostro, por lo que se decidió trabajar en esa etapa utilizando imágenes sin pre-procesar.

Se entrenaron 2 redes neuronales convolucionales mediante el recurso de la transferencia de conocimiento, utilizando imágenes de la base de datos sin pre-procesar y pre-procesadas. Las Figuras 4.6 y 4.7 dan indicios de que las clases (personas) pueden ser separadas mediante un algoritmo de clasificación, lo cual es avalado por los porcentajes de asertividad mostrados en la Tabla de 4.2. Los niveles de asertividad alcanzados en la etapa de identificación de rostros, mediante una máquina de soporte vectorial con *kernel* de tipo base radial, superan el 90% establecido para ser considerado que un algoritmo adecuado para la validación de rostros. También se superó la barrera de 0.9 correspondiente al área bajo la curva ROC (AUC).

A pesar de que las dispersiones de los datos (rostros codificados) concernientes a las imágenes pre-procesadas es menor al de sus análogos sin pre-procesar, la asertividad de la red entrenada con las imágenes sin pre-procesar es ligeramente mayor. Una posible explicación a que esto ocurra es que se está evaluando el rendimiento de una red con 30 personas mientras que esta fue entrenada originalmente con más de 10’000. Puede que esta menor dispersión juegue un papel más significativo si se contemplaran más personas en la base de datos.

En base a las métricas obtenidas en la detección de rostros se puede concluir que se han desarrollado distintos tipos de estrategia de pre-procesamiento que han permitido menguar las perturbaciones de iluminación, por lo que la hipótesis específica se comprueba.

RECOMENDACIONES

La cámara digital utilizada en el presente trabajo (Logitech HD PRO C920) presenta una variación automática en el enfoque de las imágenes que adquiere dependiendo de las condiciones de iluminación, por lo que se recomienda para trabajos posteriores se busque la manera de definir estos parámetros como constantes y/o poder acceder a ellos mediante software.

Para evitar el uso de la inspección visual para la validación de las etapas de pre-procesamiento, se puede realizar un análisis de correlación en los histogramas correspondientes a los rostros. Para esto se requiere un nivel de etiquetado con mayor detalle al realizado en el presente trabajo, pero con esto se podría realizar una sintonización de parámetros del algoritmo FLHS más exacta.

Para mejorar los resultados en la etapa de detección de rostros para imágenes pre-procesadas se podrían utilizar filtros pasa-baja o pasa-banda que disminuyan el ruido de alta frecuencia en las regiones del rostro, cuyos parámetros deben de sintonizarse en función a cada estrategia de pre-procesamiento.

Las condiciones de iluminación pueden variar independientemente de la hora en la que las imágenes fueron adquiridas (como se muestra en la Tabla 3.2). Por esta razón, para trabajos futuros, es recomendable desarrollar un modelo matemático que pueda distinguir el tipo de imagen si se desea implementar un sistema de monitoreo.

Los algoritmos implementados (sobre todo los de pre-procesamiento) no cumplen con la característica de que puedan ejecutarse en tiempo real. Por ello, se recomienda en futuros trabajos hacer la implementación de los algoritmos de pre-procesamiento en lenguajes de programación de más bajo nivel, para que puedan ser ejecutados en tiempo real. Adicionalmente, para la implementación de un sistema de monitoreo, se recomienda implementar el algoritmo optimizado en un sistema embebido con GPU (con características similares a la placa Apalis TK1 de la marca Toradex®).

GLOSARIO DE VARIABLES Y FUNCIONES

A continuación, se presenta un resumen de las variables y funciones más relevantes para el presente trabajo de Tesis. Cabe resaltar que en el presente glosario no están consideradas todas las variables utilizadas debido a su relevancia, sin embargo, se hace hincapié a que dichas variables no se encuentran repetidas a lo largo del presente trabajo.

$\arg\min_{(\cdot)}(\cdot)$: función argumento del valor mínimo

AUC: área bajo la curva ROC

α: multiplicador de Lagrange

b_cnn: variable de compensación de una capa de convolución

C: coeficiente de margen suave (SVM)

$f_{pool}(\cdot)$: función de *pooling*

$\mathrm{f}_{\mathrm{SVM}}(\cdot)$: clasificador entrenado mediante máquinas de soporte vectorial

γ: parámetro de la corrección gamma

γ_{ert}: salida del regresor $\mathrm{r}_t(\cdot,\cdot)$ en el algoritmo *gradient tree boosting*

$g_k(\cdot,\cdot)$: regresor débil

$H(\cdot)$: histograma con la función de densidad acumulada deseada (FLHS)

$\mathrm{Hog}\theta$: histograma de gradiente orientado

$h(\cdot)$: histograma con la función de densidad deseada (FLHS)

I: imagen digital
Variantes: I_{RGB} imagen de la base de datos
I_{GRAY}, I_{2GRAY}, I_{pp}, I_{2pp}, $I_{rostroGRAY}$, $I_{rostropp}$, $I_{geocRGB}$, I_{geocpp} imágenes procesadas

i, j: coordenadas de un pixel en una imagen digital

$\mathrm{K}(\cdot,\cdot)$: función de transformación *kernel* (SVM)

K: número de regresores débiles utilizados en el entramiento de r_t

$\mathrm{L}(\cdot,\cdot,\cdot)$: función de Lagrange

L: número de niveles de gris de un pixel

$\max_{(\cdot)}(\cdot)$: función valor máximo

$\min_{(\cdot)}(\cdot)$: función valor mínimo

r_f: radio de la ventana de especificación (FLHS)

$\mathrm{r}_t(\cdot,\cdot)$: regresor

R: número de inicializaciones por imagen en el entrenamiento de un regresor

S: vector de forma de un rostro
variantes: $\hat{\mathrm{S}}$ (estimación del vector S) y $\Delta\hat{\mathrm{S}}$ (vector de actualización de S)

$\mathrm{sgn}(\cdot)$: función signo

$\sigma(\cdot)$: función de activación

σ_k: desviación estándar del Kernel de base radial

T_c: número total de regresores en cascada

$\mathrm{T_{af}}$: matriz de transformación afín

v_{ert}: factor de aprendizaje del algoritmo *gradient tree boosting*

$\|v\|_k$: vector de histogramas de gradientes orientados normalizado (HOG)

$\mathrm{v}_{faceRGB}$ y v_{facepp}: vectores de rostro codificados

xHOG: Histograma de gradientes orientados

Xf: imagen de entrada con bordes (FLHS) variantes: xf_{ij} (valor de un pixel con posición i, j)

$xcnn$: resultado de la convolución en una capa de convolución

x: vector de entrada en el entrenamiento de una máquina de soporte vectorial

$wcnn$: pesos de una capa de convolución

xHOG: Etiqueta de un histograma de gradientes orientados

Yf: imagen de salida (FLHS) variantes: yf_{ij} (valor de un pixel con posición i, j)

$ycnn$: salida activada de una capa de convolución

y: etiqueta en el entrenamiento de una máquina de soporte vectorial

$\mathrm{y_{face}}$: vector de entrada en el entrenamiento de una máquina de soporte vectorial

ξ_i: variable de holgura (SVM)

BIBLIOGRAFÍA

[1] **Bishop, C.** (2006). Pattern Recognition and Machine Learning. New York, NY: Springer.

[2] **Brehar, R.** (2016). Histogram of Oriented Gradients [Material de clase]. Pattern Recognition Systems, Technical University of Cluj-Napoca, Rumania. Disponible en: http://users.utcluj.ro/~raluca/prs/prs_lab_05e.pdf

[3] **Burges, C. J.** (1998). A tutorial on support vector machines for pattern recognition. Data mining and knowledge discovery, 2(2), 121-167.

[4] **Chen, D., Ren, S., Wei, Y., Cao, X., & Sun, J.** (2014). Joint cascade face detection and alignment. In European Conference on Computer Vision (pp. 109-122). Springer International Publishing.

[5] **Dalal, N., & Triggs, B.** (2005). Histograms of oriented gradients for human detection. In Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on (Vol. 1, pp. 886-893). IEEE.

[6] **Fawcett, T.** (2006). An introduction to ROC analysis. Pattern recognition letters, 27(8), 861-874.

[7] **Feng, Z. X., Yuan, Y., & Lai, J. H.** (2015). Learning Blur Invariant Face Descriptors for Face Verification Under Realistic Environment. In CCF Chinese Conference on Computer Vision (pp. 355-365). Springer Berlin Heidelberg.

[8] **Gibiansky, A.** (2014). Convolutional Neural Networks [post en blog]. Disponible en: http://andrew.gibiansky.com/blog/machine-learning/convolutional-neural-networks/

[9] **Georghiades, A. S., Belhumeur, P. N., & Kriegman, D. J.** (2001). From few to many: Illumination cone models for face recognition under variable lighting and pose. IEEE transactions on pattern analysis and machine intelligence, 23(6), 643-660.

[10] **Gonzales, R. & Woods, R.** (2008). Digital Image Processing – 2nd Edition. New Jersey: Pretience Hall.

[11] **Goodfellow, I., Bengio, Y., & Courville, A.** (2016). Deep learning (Vol. 1). Cambridge: MIT press.

[12] **Hastie, T., Tibshirani, R., & Friedman, J.** (2001). The elements of statistical learning (Vol. 1, No. 10). New York, NY, USA: Springer series in statistics.

[13] **Kazemi, V., & Sullivan, J.** (2014). One millisecond face alignment with an ensemble of regression trees. In 27th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, United States (pp. 1867-1874). IEEE Computer Society.

[14] **LeCun, Y., Jackel, L. D., Boser, B., Denker, J. S., Graf, H. P., Guyon, I., Henderson, D., Howard, R. E., and Hubbard, W.** (1989). Handwritten digit recognition: Applications of neural network chips and automatic learning. IEEE Communications Magazine, 27(11), 41–46.

[15] **Liu, H. D., Yang, M., Gao, Y., & Cui, C.** (2014). Local histogram specification for face recognition under varying lighting conditions. Image and Vision Computing, 32(5), 335-347.

[16] **Liu, H. D., Yang, M., Gao, Y., & Cao, L.** (2014). Fast Local Histogram Specification. IEEE Transactions on Circuits and Systems for Video Technology, 24(11), 1833-1843.

[17] **Liu, H. D., & Yang, M.** (2012). Local histogram specification using learned histograms for face recognition. In Image Processing (ICIP), 2012 19th IEEE International Conference on (pp. 597-600). IEEE.

[18] **Lu, C., & Tang, X.** (2014). Surpassing human-level face verification performance on LFW with GaussianFace. In *AAAI* (pp. 3811-3819).

[19] **Maaten, L. V. D., & Hinton, G.** (2008). Visualizing data using t-SNE. Journal of machine learning research, 9(Nov), 2579-2605.

[20] **Ministerio de Justicia del Perú** (2009). DECRETO SUPREMO Nº 017-2009-MTC. Disponible en: http://transparencia.mtc.gob.pe/. Revisado el 12 nov., 2016.

[21] **Policía Nacional del Perú** (2015). Anuario Estadístico. Disponible en: https://www.pnp.gob.pe/anuario_estadistico/documentos/ANUARIO%20PNP%202015%20DIREST%20PUBLICACION.pdf

[22] **Sagonas, C., Tzimiropoulos, G., Zafeiriou, S., & Pantic, M.** (2013). A semi-automatic methodology for facial landmark annotation. En las Memorias de IEEE conference on computer vision and pattern recognition workshops (pp. 896-903).

[23] **Schroff, F., Kalenichenko, D., & Philbin, J.** (2015). Facenet: A unified embedding for face recognition and clustering. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 815-823).

[24] **Solomon, C., & Breckon, T.** (2011). Fundamentals of Digital Image Processing: A practical approach with examples in Matlab. John Wiley & Sons.

[25] **Szeliski, R.** (2010). Computer vision: algorithms and applications. Springer Science & Business Media.

[26] **Sun, Y., Chen, Y., Wang, X., & Tang, X.** (2014). Deep learning face representation by joint identification-verification. In Advances in neural information processing systems (pp. 1988-1996).

[27] **Sun, Y., Wang, X., & Tang, X.** (2015). Deeply learned face representations are sparse, selective, and robust. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 2892-2900).

[28] **Taigman, Y., Yang, M., Ranzato, M. A., & Wolf, L.** (2014). Deepface: Closing the gap to human-level performance in face verification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 1701-1708).

[29] **Vapnik, V.** (2013). The nature of statistical learning theory. Springer science & business media.

[30] **Yi, D., Lei, Z., Liao, S., & Li, S. Z.** (2014). Learning face representation from scratch. arXiv preprint arXiv:1411.7923.