ANÁLISIS DE DESEMPEÑO DE ALGORITMOS DE RECONOCIMIENTO DE PLACAS VEHICULARES MEDIANTE LA APLICACIÓN DE LA TRANSFORMADA WAVELET DISCRETA Y LA CORRELACIÓN DIGITAL DE IMÁGENES

ANGIE MAYLIN NUÑEZ BEDOYA
NATALIA ISABEL MAYA PERFETTI

UNIVERSIDAD DEL CAUCA

FACULTAD DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES

DEPARTAMENTO DE TELECOMUNICACIONES

2018

ANÁLISIS DE DESEMPEÑO DE ALGORITMOS DE RECONOCIMIENTO DE PLACAS VEHICULARES MEDIANTE LA APLICACIÓN DE LA TRANSFORMADA WAVELET DISCRETA Y LA CORRELACIÓN DIGITAL DE IMÁGENES

ANGIE MAYLIN NUÑEZ BEDOYA
NATALIA ISABEL MAYA PERFETTI

Director:
Msc. Ing. HAROLD ARMANDO ROMO ROMERO

UNIVERSIDAD DEL CAUCA

FACULTAD DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES

DEPARTAMENTO DE TELECOMUNICACIONES

2018

TABLA DE CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

GLOSARIO DE ACRÓNIMOS

ABM *Area Based Matching,* Emparejamiento Basado en Área.

AI *Artifical Intelligence*, Inteligencia Artificial.

ASC *Asymmetric Correlation,* Correlación Asimétrica.

CC *Cross-Correlation,* Correlación Cruzada.

CV *Computer Vision*, Visión de Máquina.

CWT *Continuos Wavelet Transform,* Transformada Wavelet Continua.

DIC *Digital Image Correlation,* Correlación Digital de Imágenes.

DIP *Digital Image Processing*, Procesamiento Digital de Imágenes.

DWT *Discrete Wavelet Transform*, Transformada Wavelet Discreta.

EHD E*dge Histogram Descriptor*, Descriptor de Histograma de Bordes.

FBM *Feature Based Matching,* Emparejamiento Basado en Características.

GLOH *Gradient Location and Orientation Histogram*, Ubicación del Gradiente e Histograma de Orientación.

HOG *Histogram of Oriented Gradients*, Histograma de Gradientes Orientados.

HTD *Homogeneous Texture Descriptor*, Descriptor Homogéneo de Textura.

ML *Machine Learning*, Aprendizaje de Máquina.

MRA *Multiresolution Analysis*, Análisis Multiresolución.

NCC *Normalized Cross Correlation,* Correlación Cruzada Normalizada.

OSAD *Optimized Sum of Absolute Difference*, Suma de Diferencia Absoluta Optimizada.

OSSD *Optimized Sum of Squared Difference*, Suma de Diferencias al Cuadrado Optimizada.

QMF *Quadrature Mirror Filter,* Filtro Espejo en Cuadratura.

RANSAC *RANdom Sample and Consensus*, Algoritmo de Consenso de Muestra Aleatoria.

ROI *Region Of Interest*, Región De Interés.

SAD *Sum of Absolute difference*, Suma de Diferencia Absoluta.

SIFT *Scale Invariant Feature Transform*, Transformada de Característica Invariante a Escala.

SNR *Signal-to-Noise Ratio,* Relación Señal a Ruido.

SSD *Sum of Squared Difference*, Suma de Diferencias al Cuadrado.

SURF *Speeded Up Robust Feature*, Características Robustas de Alto Rendimiento.

USM *Unsharp Masking,* Filtro de Desenfoque.

WT *Wavelet Transform*, Transformada Wavelet.

ZNCC *Zero mean Normalized Cross Correlation,* Correlación Cruzada Normalizada de Media Cero.

INTRODUCCIÓN

En la actualidad, el flujo vehicular en las ciudades ha aumentado exponencialmente, haciendo cada vez más difícil ejercer control sobre los vehículos que se movilizan por las carreteras, lo que conlleva no solo al incremento en el número de infracciones de normas de tránsito cometidas y con esto, el número de accidentes de tránsito atribuidos al no cumplimiento de estas, sino también ocasionando que vehículos que se encuentran sin documentos o sean robados circulen libremente alrededor de la ciudad. Además, la falta de control sobre los vehículos que ingresan a parqueaderos interiores y/o exteriores repercute en la seguridad, principalmente en situaciones en las que los vehículos son dejados sin supervisión o el personal de vigilancia no da abasto con la cantidad de vehículos dentro del parqueadero.

Por lo anterior, y dada la deseada migración hacia las *Smart Cities,* se han empezado a implementar técnicas como el Procesamiento Digital de Imágenes (DIP, *Digital Image Processing*), que puedan ser empleadas bajo este tipo de contextos, mediante aplicaciones como el reconocimiento de placas vehiculares, objetivo del presente trabajo de grado.

El DIP es un conjunto de métodos y técnicas que permiten, tanto mejorar la calidad de una imagen, como extraer sus características para identificar o reconocer patrones de interés. Generalmente consta de cinco etapas (ver figura 1) que se describirán a continuación.

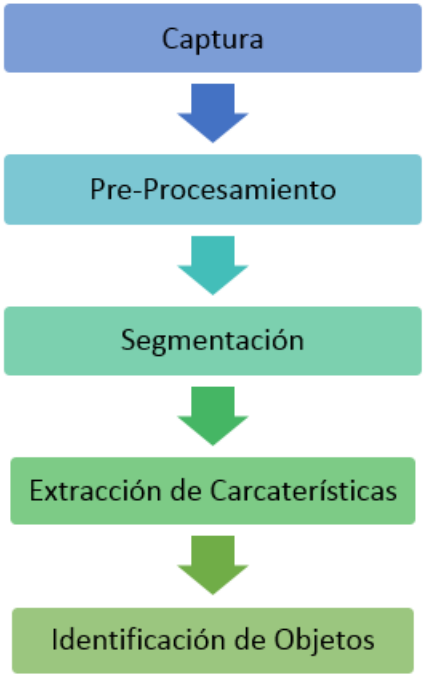

Figura 1 Etapas del Procesamiento Digital de Imágenes.

La etapa de *captura* consiste en la adquisición de una fotografía digital la cual es almacenada en un computador, luego se realiza la segunda etapa que corresponde al *pre-procesamiento,* el cual permite mejorar su calidad mediante operaciones comunes como: ajuste de tamaño y orientación, corrección de color y perfeccionamiento de bordes, en conjunto con operaciones más complejas usadas para reducir el ruido o corregir aspectos consecuentes al método de su captura. Una vez mejorada la calidad de la imagen se realiza la tercera etapa conocida como *segmentación*, en esta se divide la imagen en regiones más pequeñas u objetos de interés. En la cuarta etapa se seleccionan y se extraen las características apropiadas de los objetos segmentados y finalmente, en la quinta y última etapa se realiza la identificación y reconocimiento de los objetos. Este proceso y las etapas involucradas se describen con más detalle en el capítulo 1 del presente documento.

Dentro de los métodos de procesamiento de imágenes existentes se encuentra la Transformada Wavelet (WT, *Wavelet Transform*), cuya principal característica es la capacidad que tiene para descomponer una señal en

diferentes niveles, permitiendo revelar aspectos de dicha señal como cambios o discontinuidades. Además, el análisis de señales con la WT puede comprimir o reducir el ruido sin mayor degradación de la imagen, siendo un método viable para minimizar los posibles efectos relacionados a la adquisición de esta, adicionalmente, gracias a la posibilidad de descomponer una imagen en sub-imágenes de aproximación y detalles, es posible realizar una mejor identificación de los objetos dentro de la misma, permitiendo ser reconocidos y clasificados con mayor precisión.

Otro método empleado en el DIP es la Correlación Digital de Imágenes (DIC, *Digital Image Correlation*), la cual establece el grado de similitud entre dos imágenes mediante la comparación de estas. Esta técnica es frecuentemente utilizada para el reconocimiento de patrones en áreas como la robótica, la medicina, la seguridad, el transporte, entre otros.

Estos dos métodos son aplicados al análisis de desempeño de algoritmos de reconocimiento de placas vehiculares. Como primer paso se realiza el estudio de la Transformada Wavelet Discreta (DWT, *Discrete Wavelet Transform*) con el fin de extraer las principales características de los caracteres de la placa vehicular y realizar su reconstrucción a partir de los detalles horizontales y verticales. Luego se estudian los diferentes métodos de emparejamiento de patrones basados en área y métodos basados en características correspondientes al campo de la DIC, que permiten establecer la medida de similitud entre los caracteres de la placa que se quiere reconocer y los caracteres previamente almacenados en una base de datos, de esta manera se logra el reconocimiento del número de la placa vehicular. El estudio de estas dos técnicas se aborda dentro del capítulo 2.

En el capítulo 3, y a partir de los métodos y técnicas estudiadas dentro de los capítulos anteriores, se realiza el diseño y desarrollo de dos algoritmos: uno utilizando la DWT y un método de correlación de imágenes basado en área (Correlación Cruzada Normalizada) y otro en el que se utiliza la DWT en

conjunto con un método de correlación de imágenes basado en características (Histograma de Gradientes Orientados).

En el capítulo 4, se realiza un plan de pruebas para evaluar el desempeño de los algoritmos de reconocimiento de placas vehiculares desarrollados y así determinar cuál de estos permite un mejor reconocimiento de las placas vehiculares.

Finalmente, en el capítulo 5 se comentan las conclusiones basadas en los alcances y limitaciones encontradas en el desarrollo del proyecto, y se presentan algunos trabajos futuros que permitan dar continuidad al desarrollo de algoritmos aplicados al reconocimiento de placas vehiculares.

CAPÍTULO 1

VISIÓN DE MÁQUINA Y PROCESAMIENTO DIGITAL DE IMÁGENES

El aprendizaje de máquina (ML, *Machine Learning*) es un campo de la Inteligencia Artificial (AI, *Artificial Intelligence*) que estudia diversas técnicas matemáticas para la clasificación de patrones; estas técnicas son ampliamente utilizadas en el campo conocido como Visión de Máquina (CV, *Computer Vision*), el cual desarrolla modelos y métodos para adquirir, procesar, entender y analizar imágenes [1]; mediante las etapas de adquisición, procesamiento y análisis de la misma.

Cuando se habla del procesamiento de una imagen dentro del campo de CV, se hace referencia a la mejora de la imagen para su interpretación; es decir, todos los pasos necesarios que permiten la extracción de información de la imagen, para obtener una descripción explícita de los objetos que esta contiene y con esto realizar su análisis e interpretación de forma automática, mediante algoritmos de mayor nivel que involucran técnicas para el reconocimiento o emparejamiento de patrones (*pattern matching*) o el seguimiento de objetos (*object tracking).*

Para entender mejor el procesamiento digital de imágenes en la CV, es necesario hacer la distinción entre los términos *procesamiento* y *análisis*: en el primero, el principal objetivo consiste en la restauración y/o transformación de una imagen, bien sea para mejorar su calidad o realizar modificaciones que permitan la extracción de información y su posterior análisis (*procesos de bajo nivel*: realce de zonas de interés, mejora en la orientación, reducción de ruido, etc.). En el segundo, se emplean técnicas para la extracción y análisis de las características de la imagen (*procesos de nivel medio*), permitiendo su aplicación en la detección e identificación de patrones y finalmente su

reconocimiento e interpretación (*procesos de alto nivel*) [2] según se muestra en la figura 1.1.

Esta diferenciación brinda una visión más global sobre este campo y su aplicación en el reconocimiento de huellas digitales, análisis de muestras de sangre y radiografías, análisis de cultivos, bacteriología, diagnóstico médico o el reconocimiento de placas vehiculares que se desarrollará en el presente trabajo de grado.

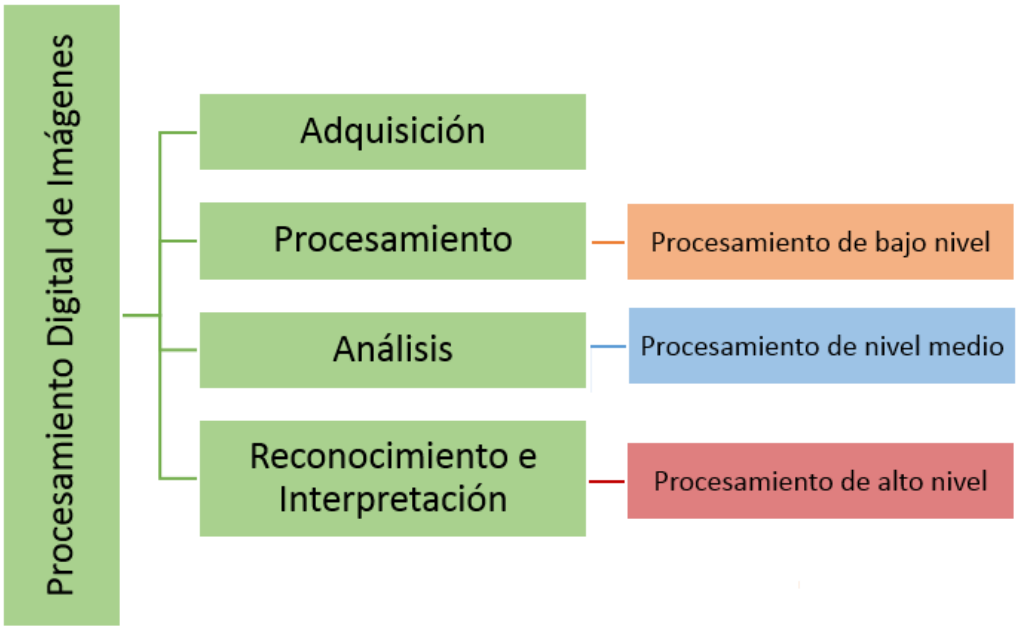

Figura 1.1 Pasos dentro del Procesamiento Digital de Imágenes y la Visión de Máquina.

1.1 ADQUISICIÓN DE LA IMAGEN

La adquisición de las imágenes[1] a tratar se realiza comúnmente mediante cámaras digitales obteniéndose fotografías a color o también conocidas como imágenes en formato RGB (*Red, Green, Blue*). Una de las principales

[1] Todas las imágenes capturadas deben ser almacenadas en una base de datos para realizar las posteriores etapas de procesamiento.

características del uso de cámaras digitales es que permiten una alta compresión de las imágenes pero brindan una resolución limitada[2].

1.1.1 Imagen RGB

Una imagen RGB está compuesta por tres matrices, cada matriz representa la intensidad de uno de los tres colores rojo, verde y azul; la intensidad de cada uno de ellos se establece por una gama que va del 0 al 255 y la mezcla de estas matrices da como resultado la tonalidad que se puede apreciar en las imágenes o fotografías.

1.2 PROCESAMIENTO DE BAJO NIVEL

En esta etapa se aplican diversas técnicas u operaciones dentro del dominio del espacio o el dominio de la frecuencia, que permiten mejorar la calidad de la imagen mediante: la reducción de ruido, el realce de contraste y bordes, la eliminación del fondo de la imagen, etc; permitiendo facilitar las etapas de segmentación y extracción de características para el reconocimiento de patrones.

1.2.1 Reducción de Ruido

El ruido dentro de una imagen se produce como consecuencia del proceso de adquisición y puede definirse como la variación no deseada del brillo o de la información de color de esta, sin embargo, puede ser mitigado o eliminado mediante el uso de filtros[3]. El proceso de filtrado dentro del dominio espacial se realiza mediante filtros lineales como el filtro de la media y el filtro gaussiano; y no lineales como el filtro de la mediana, filtro de máximo, filtro de mínimo, etc. A su vez, dado que el espectro del ruido corresponde a componentes de alta frecuencia, en el dominio de la frecuencia la aplicación de un filtro paso bajo dará como resultado la reducción de ruido.

[2] Debido a la compresión que presentan las imágenes puede existir pérdida de información, por lo tanto, la resolución que se requiera en una imagen depende del tipo de aplicación que se desee desarrollar.
[3] El uso de filtros es normalmente necesario antes de la aplicación de un detector de bordes.

Los filtros en el dominio del espacio reciben este nombre dado que son aplicados directamente a la imagen, es decir el nivel de gris de un pixel se obtiene directamente en función del valor de sus vecinos [3].

Por otra parte, los filtros en el dominio de la frecuencia aplican la Transformada de Fourier a una imagen, obteniendo una imagen en el dominio de la frecuencia que posteriormente es convolucionada con la función del filtro. Finalmente, al resultado obtenido, se le aplica nuevamente la antitransformada para obtener así la imagen filtrada. El filtro de suavizado en el dominio de la frecuencia corresponde a un filtro pasa bajo, el cual atenúa las frecuencias altas que corresponden a los cambios fuertes de intensidad [4].

Realizar el filtrado de una imagen en el dominio espacial es mejor dado que no requieren mucho consumo computacional, sin embargo los filtros en el dominio de la frecuencia son utilizados cuando no es posible obtener una máscara de convolución apropiada para el filtraje en el dominio espacial.

A los filtros encargados de reducir el ruido en las imágenes se les conoce como filtros de suavizado porque producen un efecto de desenfoque al difuminar bordes, esto conlleva a la reducción en la definición de la imagen ya que se eliminan diferencias entre píxeles vecinos.

1.2.2 Realce de Bordes

El realce de bordes está directamente asociado con los operadores de detección de bordes y el proceso de segmentación en una imagen. Esta operación consiste en realzar o mejorar los detalles de la imagen al identificar los píxeles en los cuales se producen cambios bruscos en la intensidad, de modo que mientras más brusco sea el cambio, más fácil es detectar el borde. Entre las técnicas de realce de bordes se encuentra el uso de filtros pasa altas, el Laplaciano Digital, los operadores de gradiente como el Operador Cruzado de Roberts y el Operador Sobel, entre otros [5].

1.3 PROCESAMIENTO DE NIVEL MEDIO

El objetivo principal de esta etapa es aplicar técnicas de segmentación y extracción de características.

1.3.1 Segmentación

La segmentación consiste en separar una imagen en partes más pequeñas o Regiones De Interés (ROI, *Region of Interest*) para ser analizadas y reconocidas de forma independiente de acuerdo con las discontinuidades o similitudes existentes entre los niveles de gris de los píxeles vecinos.

Existen dos alternativas para realizar el proceso de segmentación, una de ellas es la *segmentación basada en discontinuidades,* la cual se realiza teniendo en cuenta los cambios bruscos en el nivel de gris de los píxeles y el método más común es el método basado en la detección de bordes [6]. La otra consiste en la *segmentación basada en similitudes*, esta se realiza considerando las zonas de la imagen que presentan valores similares, dentro de los métodos se encuentran la segmentación basada en píxeles (segmentación basada en histograma y la segmentación basada en proyecciones) y la segmentación basada en regiones (crecimiento de una región, *split and merge*, etc.). A continuación, se describirá la segmentación basada en regiones utilizando *procesamiento morfológico* seguido del *etiquetado de componentes conectados*.

1.3.1.1 Método de Segmentación basado en Crecimiento de Regiones

I. Procesamiento Morfológico

El *procesamiento morfológico* es una técnica basada en la teoría de conjuntos, que permite realzar la geometría de los objetos en imágenes binarias o en escala de grises para extraer características que representan o describen objetos y/o regiones mediante la aplicación de *operaciones morfológicas*.

Las principales operaciones morfológicas son la dilatación, la erosión y la transformación *top-hat*, a partir de ellas se pueden formar otras como la apertura y cierre. Estas operaciones hacen uso de un elemento estructurante cuya forma y tamaño debe ser seleccionado antes de aplicar la operación y se elige en función de la forma de la región que se desea obtener, dado que es este el que determina la forma exacta en la cual los objetos van a ser dilatados o erosionados. El elemento estructurante puede ser un diamante, cuadrado, círculo, octágono, entre otros (ver figura 1.2), y funciona haciendo un desplazamiento sobre la imagen como si se tratara de una convolución.

Figura 1.2 Elementos estructurantes típicos.

- **Dilatación:**

  La dilatación aumenta el tamaño de un objeto en función de la forma y el tamaño elegido para el elemento estructural; es utilizado cuando se desea rellenar huecos y unir píxeles relacionados y su funcionamiento es el siguiente: Si el origen del elemento estructurante coincide con un píxel blanco en la imagen, no se realizan cambios y se continúa analizando el siguiente; si por el contrario, el origen del elemento estructurante coincide con un píxel negro en la imagen, todos los píxeles cubiertos por el elemento estructurante se hacen negro. Un ejemplo típico de aplicar dilatación se observa en la figura 1.3.

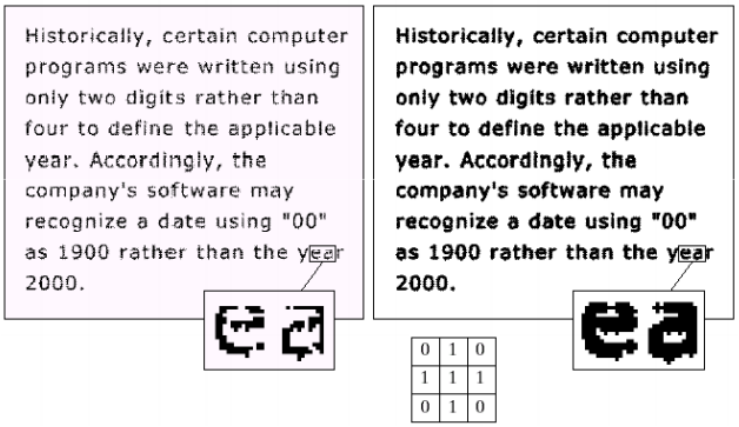

Figura 1.3 Dilatación de los caracteres de un texto.

- **Erosión:**

La erosión es el proceso inverso a la dilatación; es decir, si el origen del elemento estructurante coincide con un píxel blanco, no se realiza ningún cambio y se continúa con el siguiente píxel; sin embargo, si el origen de dicho elemento coincide con un píxel negro en la imagen y alguno de los píxeles negros del elemento estructurante cae sobre un píxel blanco en la imagen, entonces se cambia el píxel negro en la imagen por uno blanco. Esta operación generalmente cambia el tamaño de los objetos en función al tamaño del elemento estructurante empleado y se utiliza para eliminar objetos indeseados en las imágenes binarias (todos los objetos menores al tamaño del elemento estructurante elegido desaparecen como se muestra en el ejemplo de aplicación de la figura 1.4).

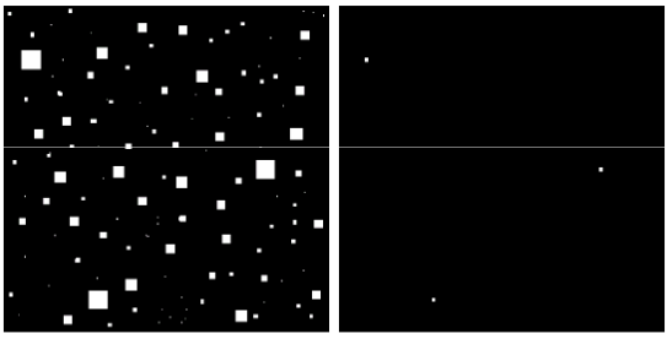

Figura 1.4 Ejemplo de aplicación de erosión.

El uso en conjunto de operaciones de erosión y dilatación da como resultado nuevas operaciones que permiten dar realce a los objetos; estas son: apertura y cierre.

- **Apertura:**

La apertura consiste en la aplicación en cascada de erosión y dilatación utilizando el mismo elemento estructurante. La apertura se utiliza para suavizar contornos y eliminar píxeles en regiones que son demasiado pequeñas para contener el elemento estructurante.

- **Cierre:**

El cierre, por su parte, consiste en una dilatación seguida de una erosión y consiste en marcar con un 1 aquellos píxeles aislados entre ceros [7], permitiendo rellenar huecos en la imagen y unir objetos similares cercanos.

- **Transformación Top-Hat:**

Otra de las transformaciones dentro del procesamiento morfológico corresponde a la transformación *Top-Hat*; esta transformación aumenta el contraste para la extracción de objetos pequeños y características claras u oscuras presentes en la imagen, y se utiliza cuando la iluminación en el fondo de la imagen no es uniforme y no es posible aplicar umbralización.

Existen dos tipos de transformaciones *Top-Hat*: *Open top-hat (white top-hat*) y *close top-hat (black top-hat)*; la primera actúa como un filtro pasa alto, resaltando todas las zonas claras de la imagen que son menores que el elemento estructurante (generalmente se utiliza para destacar los objetos claros sobre fondo negro). La segunda, conocida también como *Bottom-Hat,* remueve el fondo de la imagen dejando únicamente las zonas oscuras y de menor tamaño que el elemento estructurante, destacando así dichos objetos de interés (objetos oscuros) sobre un fondo blanco.

II. Etiquetado de componentes conectados

Para seleccionar y segmentar las ROI se utiliza el *etiquetado de componentes conectados*. Para esto, se deben determinar los límites de los objetos o las regiones contenidas en una imagen evaluando la conectividad entre los pixeles; para hacerlo se tiene en cuenta los valores de intensidad de los pixeles vecinos a un pixel determinado.

Se considera que dos píxeles son vecinos si cumplen con la definición de adyacencia, la cual establece que existen dos tipos de vecindad: Si los dos píxeles tienen en común una de sus fronteras se denominan *vecinos directos* y si comparten al menos una de sus esquinas se denominan *vecinos indirectos*, de modo que un pixel tiene cuatro vecinos directos y cuatro indirectos [7] (ver figura 1.5).

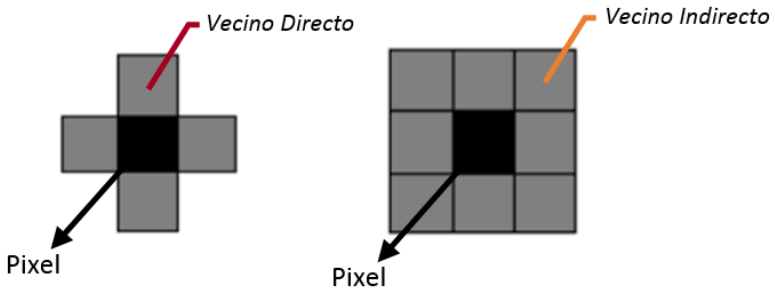

Figura 1.5 Vecindades de un píxel.

Establecer la vecindad de un píxel permite evaluar la conectividad del pixel con sus vecinos; se dice que un píxel está conectado a sus vecinos si estos satisfacen un nivel de intensidad específico; por ejemplo, para una imagen binaria, un píxel de valor 1 sólo estaría conectado a sus pixeles vecinos si tienen el mismo valor. Además, se habla de conectividad 4 cuando se evalúa la conectividad entre el pixel y sus vecinos directos, y de conectividad 8 cuando se evalúa la conectividad de todos los vecinos del píxel (ver figura 1.6).

Figura 1.6 Conectividad 8 entre píxeles.

El proceso de etiquetado se lleva a cabo una vez que se ha establecido la conectividad entre píxeles y se inicia haciendo un barrido de la imagen de derecha a izquierda y de arriba a abajo.

Para el caso de conectividad 4, se hace el barrido en grupos de tres píxeles. Si p corresponde al píxel central y, $v1$ y $v2$ son sus vecinos (ubicados arriba y a la izquierda respectivamente), la forma como se realiza el barrido implica que antes de llegar a p, los pixeles $v1$ y $v2$ ya han sido encontrados y etiquetados en caso de ser 1.

En p se verifica el valor; si $p = 0$, se corre a la siguiente posición del barrido; por el contrario, si $p = 1$ se verifica el valor de $v1$ y $v2$: en el caso de que ambos pixeles sean cero, se le asigna una etiqueta a p; por otro lado, si alguno de los dos es 1, se le asigna una etiqueta diferente a p; y por último, si ambos pixeles tienen el valor de 1 y además, comparten la misma etiqueta, se le asigna esa etiqueta a p y se pone un identificador que especifique que ambas etiquetas son equivalentes. Realizar esto implica que al terminar el barrido todos los pixeles con valor de 1 han sido marcados, pero sólo algunos de éstos son equivalentes.

El siguiente paso consiste en asignar todos los pares equivalentes de etiquetas a pares equivalentes de clase, para esto, a cada clase se le asigna una etiqueta diferente y se realiza un barrido por segunda vez reemplazando cada etiqueta por la asignada a su clase de equivalencia, en la figura 1.7 se observan las regiones etiquetadas utilizando conectividad 4.

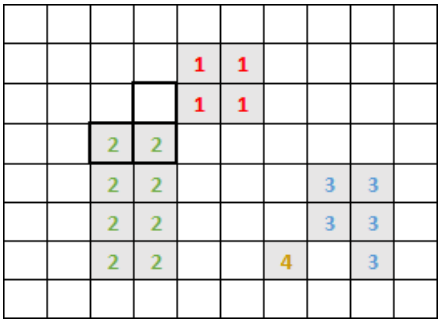

Figura 1.7 Regiones obtenidas a partir de la conectividad 4.

Este proceso se aplica de manera similar para la conectividad 8, es decir, se realiza el proceso descrito anteriormente teniendo en cuenta además los pixeles diagonales superiores a p [7] (ver figura 1.8).

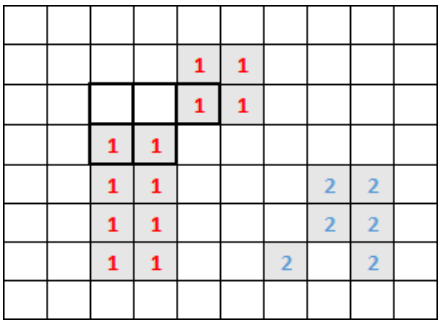

Figura 1.8 Regiones obtenidas a partir de la conectividad 8.

1.3.2 Extracción de Características

Para la extracción de características de una imagen se puede emplear la DWT, esta técnica es utilizada en el reconocimiento de patrones dada su capacidad de otorgar información detallada de una imagen. Existen diferentes tipos de *familias wavelet* que se pueden utilizar bajo diferentes aplicaciones, para la detección de características en particular, las *familias wavelet haar*, *daubechies2* y *symlets2* son las más recomendadas [8]. En el capítulo 2 se aborda con más detalles la DWT y su aplicación en imágenes.

1.4 PROCESAMIENTO DE ALTO NIVEL

La etapa final en el campo de visión de máquina corresponde al reconocimiento de patrones, para lo cual pueden emplearse dos técnicas basadas en la correlación digital de imágenes: una para establecer la Correspondencia Basada en Área (ABM, *Area Based Matching*) mediante la Correlación Cruzada Normalizada (NCC, *Normalized Cross-Correlation*), y la otra para establecer la Correspondencia Basada en Características (FBM, *Feature Based Matching*) mediante el uso del Histograma de Gradientes Orientados (HOG, *Histogram of Oriented Gradients*). El uso de estas técnicas para establecer la correlación o correspondencia entre patrones se describen detalladamente en el capítulo 2.

CAPÍTULO 2

TRANSFORMADA WAVELET Y CORRELACIÓN DIGITAL DE IMÁGENES

2.1 TRANSFORMADA WAVELET

La transformada *wavelet* (WT, *Wavelet Transform*) otorga un análisis de señales transitorias y variantes en el tiempo, dando información simultánea de tiempo y frecuencia, permitiendo identificar tendencias, cambios bruscos, inicio o finalización de eventos que ocurren en una determinada señal. Para esto, la WT descompone la señal en términos de *funciones wavelets* a partir de una función de transformación llamada base o *wavelet madre* [9]. La *wavelet madre* $\psi_{a,b}(t)$ es una función prototipo que se escala y se traslada mediante los parámetros a y b respectivamente, para analizar la señal en diferentes resoluciones [10]. Actualmente, existen una gran variedad de *wavelets madre* cuya efectividad depende de la aplicación y requerimiento, las más conocidas son: *Haar, Daubechies, Symlets, Coiflets, Biortogonales, Meyer, Shannon* y *Morlet*.

2.1.1 Transformada Wavelet Continua (CWT)

En la CWT (CWT, *Continuos Wavelet Transform*) los parámetros de escala a y desplazamiento b varían de forma continua y la señal a analizar es de naturaleza analógica y de energía finita. La CWT resulta de un proceso de comparación que se hace mediante el producto interno entre la señal a analizar $x(t)$ y todas las posibles versiones escaladas y trasladadas de la *wavelet* madre [11], dichas versiones son generadas mediante la variación de los parámetros continuos a y b, a este proceso se le conoce como análisis y se define en la ecuación (2.1):

$$CWT_{(a,b)} = \int_{-\infty}^{+\infty} \frac{1}{\sqrt{a}}\, \psi^*_{a,b}(t)\, x(t)\, dt, \tag{2.1}$$

con:

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}} \psi\left(\frac{t-b}{a}\right); \quad a, b \in \mathcal{R}, \ a \neq 0. \tag{2.2}$$

La ecuación (2.2) corresponde a la función *wavelet madre*, la cual debe pertenecer a una familia de funciones que satisfacen los siguientes tres criterios matemáticos [12]:

1. La función debe tener energía finita, esto hace referencia a que la función debe ser concentrada en un intervalo de tiempo, por lo cual debe satisfacer:

$$\boldsymbol{E} = \int_{-\infty}^{+\infty} |\psi(t)|^2 dt \ < \infty. \tag{2.3}$$

2. La función debe cumplir con el criterio de admisibilidad, el cual hace referencia a que la señal debe tener media nula, por tanto debe satisfacer:

$$\boldsymbol{C_\psi} = \int_0^{\infty} \frac{\left|\breve{\psi}(f)\right|^2}{f}\, df \ < \infty, \tag{2.4}$$

   donde $\breve{\varphi}(f)$ es la tranformada de Fourier de la *wavelet madre* y f la frecuencia.

3. Si la *wavelet madre* es compleja, la transformada de Fourier debe ser real y desvanecida para frecuencias negativas.

Además, en la ecuación (2.2) la constante $\frac{1}{\sqrt{a}}$ actúa como una normalización energética, lo que implica que la trasformada de la señal tendrá la misma energía en cada escala [9]. Por otro lado, la translación b está relacionada con la localización de la wavelet madre con respecto a la señal, esta se puede encontrar adelantada o retrasada con referencia a dicha señal, lo que permite recorrerla. La escala a hace referencia a la contracción o dilatación a la que es sometida la *wavelet madre*, esto permite analizar diferentes bandas de frecuencia. La variación de los parámetros a y b en la ecuación (2.2) genera una serie de coeficientes *wavelet,* los cuales indican qué tan parecida es una determinada señal a la *wavelet madre* y a sus versiones escaladas según la magnitud de los *coeficientes wavelet*. Esta comparación permite conocer las componentes espectrales contenidas en la señal. Lo anterior puede verse ejemplificado en la figura 2.1, donde la señal que se quiere analizar se encuentra en azul y la *wavelet madre* en rojo.

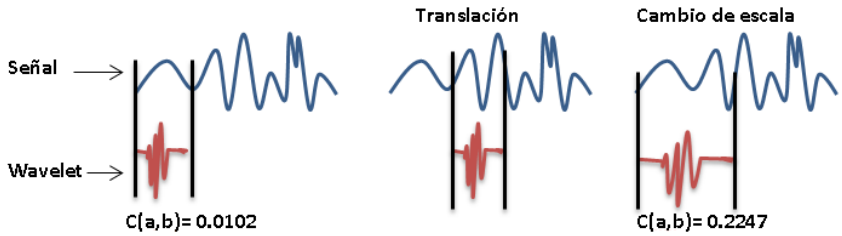

Figura 2.1 Translación, cambio de escala y coeficientes *wavelet.*

En esta imagen se pueden evidenciar tres hechos: primero, la *wavelet madre* se puede trasladar a lo largo de la señal, segundo la *wavelet madre* puede cambiar de escala, es decir estar contraída o dilatada y tercero, entre más se parezca una porción de la señal a la *wavelet madre* se obtienen unos coeficientes más altos.

Otra propiedad importante de la CWT es que permite a partir de los coeficientes reconstruir la señal original. A este proceso inverso se le llama síntesis o reconstrucción y se realiza mediante la ecuación (2.5).

$$x(t) = c_{\psi}^{-1} \int \int CWT\,(a,b) \frac{1}{\sqrt{|a|}} \psi_{a,b}^{*} \left(\frac{t-b}{a}\right) \frac{dadb}{a^2}\,. \qquad (2.5)$$

En general, la CWT permite localizar las componentes de frecuencia de interés; en el caso de una imagen como la de una placa vehicular, esta podría ser reconocida a partir de sus componentes frecuenciales, por ejemplo, un conjunto de altas frecuencias encontradas en la imagen podría indicar los contornos de la placa, sin embargo no es ideal trabajar con la CWT a nivel computacional debido a que la información entregada es altamente redundante para reconstruir la señal y por tanto aumenta significativamente el tiempo de procesamiento [9]. Considerando lo anterior, se hace necesario utilizar la DWT la cual además de disminuir el tiempo de procesamiento, es más fácil de implementar.

2.1.2 Transformada Wavelet Discreta (DWT)

La DWT cumple el mismo propósito que la CWT, solo que en la primera se utilizan técnicas de filtrado digital [9], donde se hace un filtrado de la señal original mediante filtros pasa bajo y pasa alto que permiten eliminar ciertos componentes de alta o baja frecuencia.

Para realizar la DWT se hace necesaria la discretización de los parámetros de *escala* a y *translación* b mediante el uso de una escala diádica, haciendo $a = 2^{-j}$ y $b = k2^{-j}$, donde la variable j es un entero positivo que hace referencia al espacio *wavelet* donde está trabajando la *wavelet madre*, lo que permite la variación de la escala y por tanto el nivel de resolución; el parámetro k es un entero positivo que representa la posición, de modo que la función de la ecuación (2.2) pasa a convertirse en la ecuación (2.6) y la DWT queda definida mediante la ecuación (2.7):

$$\psi_{j,k}(t) = 2^{\frac{j}{2}} \psi\left(2^{j}t - k\right); \quad j,k \,\epsilon\, \mathcal{Z}. \qquad (2.6)$$

$$DWT_{j,k} = 2^{\frac{j}{2}} \int_{-\infty}^{+\infty} x(t)\, \psi\left(2^j t - k\right) d(t). \qquad (2.7)$$

Ahora, en la ecuación (2.6) la constante $2^{\frac{j}{2}}$ se encarga de la normalización de la energía necesaria para cumplir la condición de ortonormalidad, la cual permite calcular los coeficientes de manera más simple y rápida a nivel computacional gracias a que brinda independencia entre los diferentes niveles de descomposición, evitando que se genere información redundante de la señal [13].

Por tanto, la condición de ortonormalidad establece que la función *wavelet madre* debe satisfacer las siguientes ecuaciones [14].

$$\int \psi_k(t)\psi_m^*(t)dt = 0; \qquad k \neq m, \qquad (2.8)$$

y

$$\int |\psi_k(t)|^2 dt = 1. \qquad (2.9)$$

Por otra parte, luego de calcular los coeficientes mediante la DWT es posible reconstruir la señal original $x(t)$ a partir de estos mediante dos funciones: la función *wavelet madre* $\psi(t)$ ($ecuación\ (2.6)$) y la *función escala* $\phi(t)$.

La *función escala* $\phi(t)$ permite obtener una versión menos detallada de $x(t)$ y al igual que la función *wavelet madre* $\psi(t)$, satisface la condición de ortonormalidad, la cual es dilatada por un factor de escala 2^j y es desplazada por un factor discreto de translación k [13]. La ecuación (2.10) representa la función escala $\phi(t)$ y la ecuación (2.11) corresponde a la síntesis de la señal original, donde $a_{j,k}$ corresponde a los coeficientes de escala y $d_{j,k}$ a los *coeficientes wavelet*. Para entender como son generados estos coeficientes es

necesario introducir un nuevo concepto: el Análisis Multiresolución (MRA, *Multi Resolution Analysis*).

$$\phi_{j,k}(t) = 2^{\frac{j}{2}}\phi\left(2^{j}t - k\right); \quad j,k \in \mathcal{Z}, \tag{2.10}$$

$$x(t) = \sum_{k}\sum_{j} a_{j,k}\ \phi(t) + \sum_{k}\sum_{j} d_{j,k}\ \psi(t); \quad j,k \in \mathcal{Z}. \tag{2.11}$$

2.1.2.1 Análisis Multiresolución (MRA)

El análisis multiresolución proporciona un método rápido para calcular los coeficientes de *escala* y *wavelet* de la señal para realizar su reconstrucción a partir de estos [12]. El análisis de la señal se realiza en el dominio discreto cuando se descompone en aproximaciones (*coeficientes de escala*) y detalles (*coeficientes wavelet*), donde las aproximaciones son las componentes de gran escala y baja frecuencia y los detalles son los componentes de baja escala y alta frecuencia, de este modo se pueden analizar las diferentes componentes de frecuencia mediante filtros paso alto y pasa bajo [9].

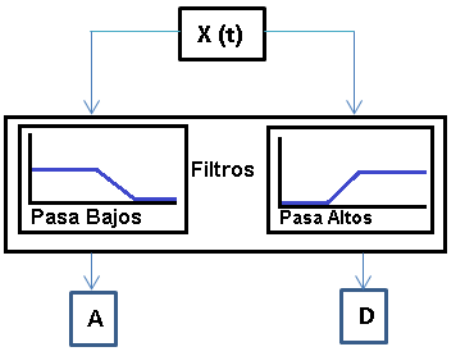

Figura 2.2 Primer nivel de descomposición *wavelet.*

La figura 2.2 muestra un primer nivel de descomposición de una señal, en donde se obtienen dos señales denominadas aproximaciones (A) y detalles (D). Las aproximaciones (A) son el resultado de pasar la señal por el filtro pasa bajo y por tanto brindan un análisis de las bajas frecuencias, las cuales aportan

mayor parte de la información; los detalles (D), son el resultado de pasar la señal por el filtro pasa alto, permitiendo analizar las altas frecuencias las cuales otorgan información de eventos particulares. Este proceso de filtrado puede ser repetido cuantas veces sea necesario dando origen a una descomposición multinivel o árbol de descomposición *wavelet* como se observa en la figura 2.3. En este caso, se ha descompuesto la señal hasta el nivel 4, obteniendo detalles en cada nivel y aproximaciones de último nivel mediante el uso de filtros pasa alto (h), pasa bajo (g) y un proceso de decimado $(2\downarrow)$ que se explicará más adelante. Este proceso de filtrado permite caracterizar la información contenida en la señal.

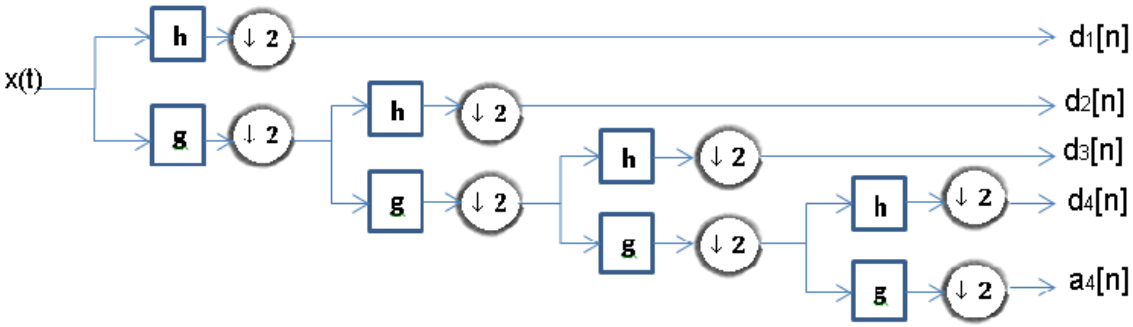

Figura 2.3 Árbol de descomposición wavelet de 4 niveles.

Una vez obtenidas las aproximaciones y los detalles mediante las ecuaciones (2.12) y (2.13), es posible realizar la síntesis de la señal $x(t)$ a partir de cualquier nivel de descomposición mediante la ecuación (2.11).

$$d_{j,k} = 2^{\frac{-j}{2}} \int_{-\infty}^{+\infty} x(t)\, \psi\left(2^{-j}t - k\right)dt, \tag{2.12}$$

$$a_{j,k} = 2^{\frac{-j}{2}} \int_{-\infty}^{+\infty} x(t)\, \phi\left(2^{-j}t - k\right)dt. \tag{2.13}$$

Además, la ventaja que se obtiene con el MRA es que en altas frecuencias, la DWT tiene una buena resolución en el dominio del tiempo y una baja resolución

frecuencial, mientras que en bajas frecuencias tiene una buena resolución frecuencial y una baja resolución en el tiempo. Esto implica que las altas frecuencias tienen una mejor localización en el tiempo, caso contrario en las bajas frecuencias donde su localización en el tiempo no es muy precisa, como se observa en la figura 2.4.

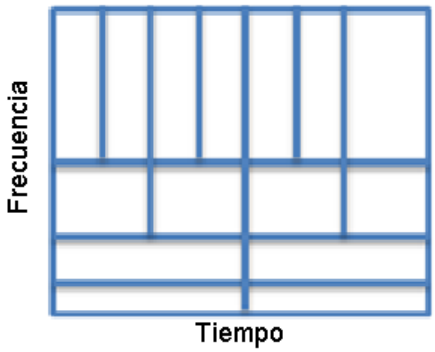

Figura 2.4 Resolución tiempo-frecuencia.

En resumen, el MRA permite analizar señales en múltiples bandas de frecuencias, el proceso para hacerlo se conoce como codificación de sub-bandas, tema que será tratado a continuación.

2.1.2.2 Codificación de Sub-Bandas

En la codificación de sub-bandas el filtrado es realizado mediante un proceso matemático donde se hace la convolución de la señal con la respuesta al impulso del filtro, de modo que la salida de los filtros pasa bajo y pasa alto queda definida por las ecuaciones descritas a continuación:

$$Y_{baja}(n) = \sum_{k=-\infty}^{\infty} x(k) \,.\, g(2n-k), \tag{2.14}$$

$$Y_{alta}(n) = \sum_{k=-\infty}^{\infty} x(k) \,.\, h(2n-k). \tag{2.15}$$

Después del filtrado se debe hacer un proceso de decimado (ver figura 2.3), el cual consiste en submuestrear la señal en un factor de dos, es decir, la señal tendrá la mitad de puntos o muestras dado que esta es muestreada a la frecuencia de *Nyqui*st, la cual debe ser el doble de la frecuencia máxima; por consiguiente, la mitad de las muestras se deben eliminar siguiendo este criterio dado que ahora la señal tiene su frecuencia máxima reducida a la mitad. Este submuestreo cambia la escala pero no afecta la resolución, dado que la mitad de las muestras serán redundantes y por ende se pueden eliminar.

Si bien en la descomposición se eliminan ciertas muestras, en la reconstrucción deben añadirse. Para realizar este proceso, se emplea los filtros de reconstrucción perfecta o Filtros Espejo en Cuadratura (QMF, *Quadrature Mirror Filter*) los cuales deben satisfacer la siguiente condición de ortogonalidad:

$$HG^T = GH^T = 0, \tag{2.16}$$

$$HH^T = GG^T = I. \tag{2.17}$$

Los operadores G y H hacen referencia a los filtros, las notaciones G^T y H^T indican la transpuesta de G y H respectivamente e I hace referencia a la matriz identidad [12].

Los filtros QMFs están relacionados con la función escala y la *función wavelet* mediante las siguientes ecuaciones:

$$g[n] = \langle \phi(t), \sqrt{2}\,\phi(2t-n) \rangle, \tag{2.18}$$

$$h[n] = \langle \psi(t), \sqrt{2}\,\psi(2t-n) \rangle, \tag{2.19}$$

donde g y h hacen referencia a los filtros pasa bajo y alto respectivamente, estos permiten una reconstrucción exacta de la señal original desde los coeficientes de la DWT.

2.2 DWT EN IMÁGENES (2D-DWT)

La DWT aplicada a imágenes (señales en dos dimensiones (2D)), descompone la imagen en filas y luego en columnas realizando el mismo proceso descrito anteriormente, considerando cada fila y cada columna como señales uni-dimensionales. Este proceso genera cuatro sub-matrices de la imagen original: la primera corresponde a una sub-matriz llamada matriz de aproximación (LL), la segunda (LH) corresponde a los detalles horizontales de la imagen, la tercera (HL) corresponde a los detalles verticales y finalmente, la cuarta sub-matriz (HH) corresponde a los detalles diagonales contenidos en la imagen original, como se observa en la figura 2.5.

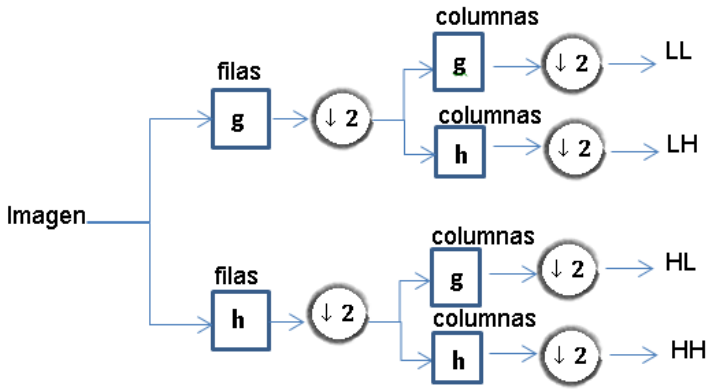

Figura 2.5 Descomposición *wavelet* de una imagen.

Las sub-matrices generadas caracterizan la información contenida en la imagen: la matriz de aproximaciones indica la tendencia de la imagen, mientras que las matrices de detalles indican los cambios que ocurren en las distintas direcciones (horizontal, vertical y diagonal) por separado [9], este proceso aplicado a una imagen real se observa en la figura 2.6.

Figura 2.6 Descomposición wavelet en un nivel utilizando la *wavelet Haar*
Imagen original (izquierda), descomposición *wavelet* en aproximaciones y detalles (derecha).

La matriz de aproximaciones resultante del primer nivel de descomposición se puede seguir descomponiendo de forma similar y recursiva hasta llegar a la máxima escala j.

Además, en la DWT bidimensional también es posible hacer la reconstrucción o síntesis de la imagen a partir de los coeficientes de aproximación o detalles mediante la combinación de las cuatro sub-matrices y haciendo un proceso de undecimado o interpolación, el cual consiste en añadirle muestras a la señal (ver figura 2.7) [12].

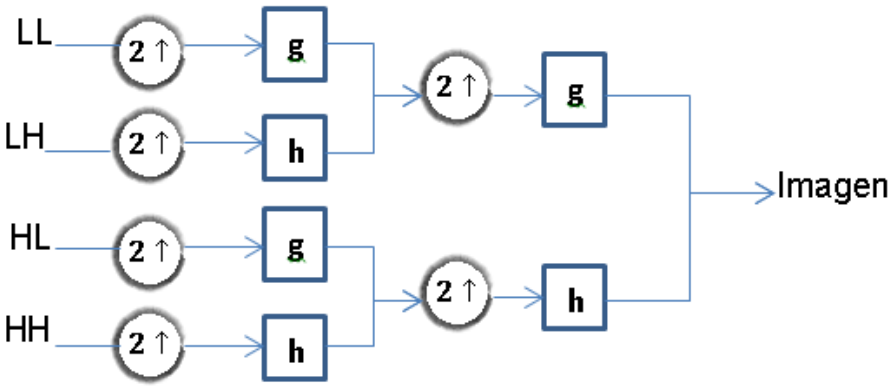

Figura 2.7 Reconstrucción *wavelet* de una imagen.

La calidad de la información obtenida al aplicar la DWT a una imagen y en general a una señal, dependerá de la similitud existente entre la *wavelet madre* y la señal de interés, por esta razón, la correcta selección de una *wavelet*

madre varía de acuerdo al caso de estudio en particular, dado que cada *familia wavelet* presenta características propias cuyo resultado cambia según sea la señal analizada [15].

Las familias *wavelet* que pueden ser empleadas en el análisis de señales en 2D son: *haar, daubechies, symlet, coiflet, biorthogonal* y *reverse biorthogonal*.

- **Wavelet Haar:** Es la más sencilla de las *familias wavelet*, esta es de tipo discontinua y se asemeja a la *función escala* [8]. Su forma de onda se observa en la figura 2.8.

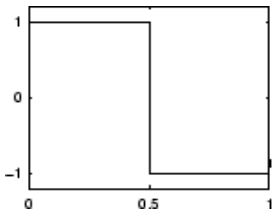

Figura 2.8 Wavelet Haar.

- **Wavelet Daubechies:** Nombrada bajo el nombre de su inventora *'Ingrid Daubechies',* quien creó *wavelets* ortogonales de soporte compacto facilitando el *análisis wavelet discreto*. Las *wavelet* d*aubechies (*ver figura 2.9*)* se representan por dbN, donde N es el orden de la *wavelet*, es decir, el número de momentos de desvanecimiento que presenta la señal. La *wavelet* db1 representa la *wavelet haar* [8].

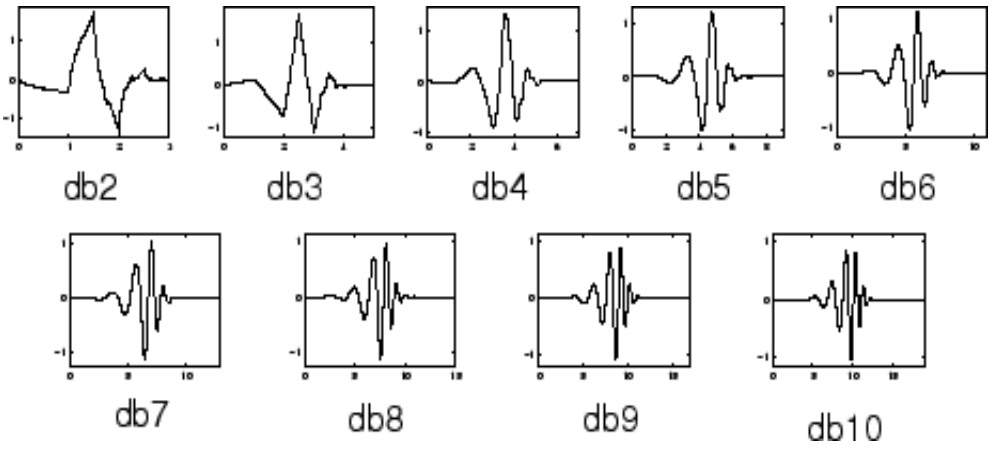

Figura 2.9 Familia Wavelet Daubechies.

- **Wavelet Biorthogonal:** Esta familia *wavelet* al igual que la *wavelet haar* presenta fase lineal necesaria para la reconstrucción de señales e imágenes. Estas tienen dos funciones escala: Una para *análisis* y otra para *síntesis*, es decir, una para descomposición y otra para reconstrucción. Tanto la *wavelet* de *análisis* como la de *síntesis* pueden tener diferentes momentos de desvanecimiento, por tanto se puede utilizar la *wavelet* con mayor número de momentos de desvanecimiento para el *análisis* y la más suave (con menos momentos de desvanecimiento) para la reconstrucción.

  La *wavelet biorthogonal* (ver figura 2.10) se representa como $BioNd.Nr$, donde Nr y Nd corresponde al número de momentos de desvanecimiento de los filtros de reconstrucción y descomposición respectivamente [8].

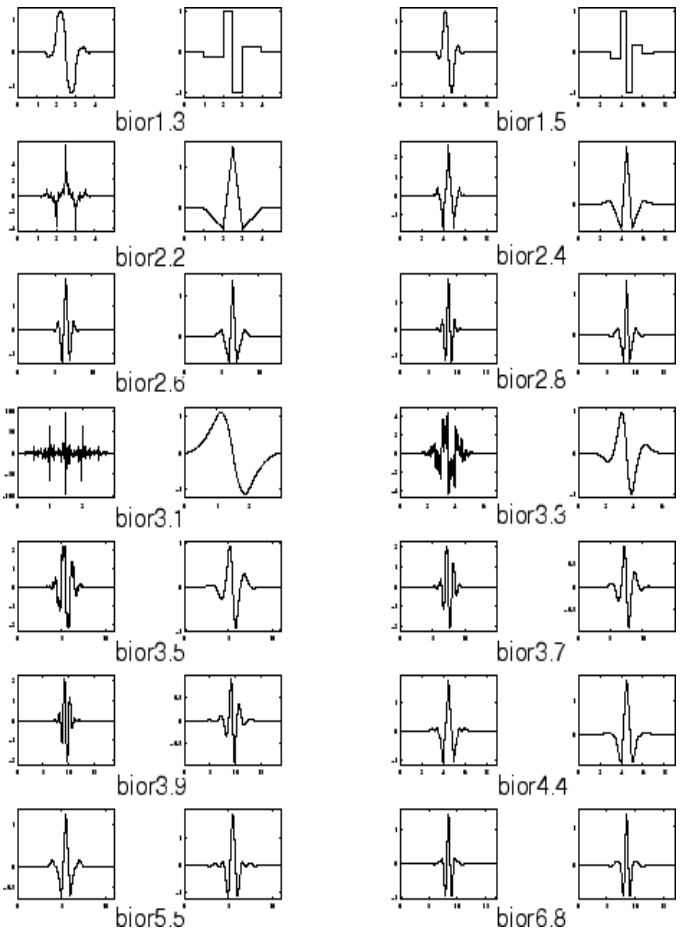

Figura 2.10 Familia Wavelet Biorthogonal.

- **Wavelet Reverse Biorthogonal:** Esta familia se obtiene a partir de un par de *wavelets biorthogonales* las cuáles se representan como $RbioNd.Nr$, donde Nr y Nd corresponde al número de momentos de desvanecimiento de los filtros de reconstrucción y descomposición respectivamente [8].

- **Wavelet Symlet:** Las *wavelet symlet* (ver figura 2.11) nacen como modificaciones realizadas por I. Daubechies a la *familia wavelet* que lleva su mismo nombre. Estas son casi simétricas y presentan propiedades similares a la familia *db* [8].

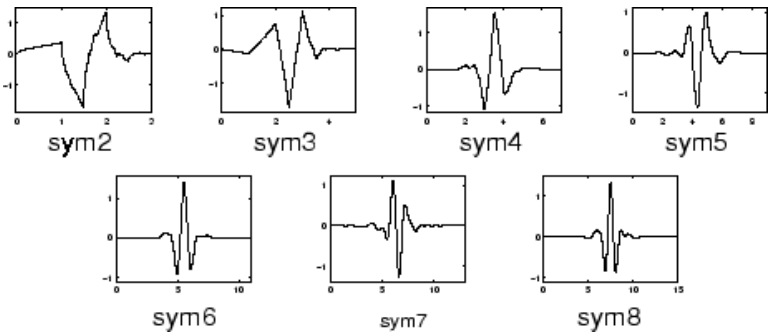

Figura 2.11 Familia Wavelet Symlets.

- **Wavelet Coiflet:** Estas funciones *wavelet* fueron construidas también por I. Daubechies (ver figura 2.12); se representan como $coifN$, donde N es el número de momentos de desvanecimiento para las funciones *wavelet* y *escala* [8].

Figura 2.12 Familia Wavelet Coiflet.

Todas estas familias difieren entre sí de acuerdo a propiedades como: soporte de la *wavelet* en tiempo y frecuencia, simetría o asimetría de la *wavelet*, número de momentos de desvanecimiento, regularidad de la *wavelet* y

existencia de una función escala (ϕ). Sin embargo, las propiedades más importantes se definen a continuación [8]:

- **Ortogonalidad:** Si una *wavelet* es ortogonal, al aplicar la DWT a una señal se preserva la energía [8].

- **Momentos de Desvanecimiento:** El número de momentos de desvanecimiento está ligado con la oscilación de la *wavelet* (entre mayor sean los momentos de desvanecimiento, mayor será la oscilación). Esta propiedad se relaciona con la compresión de información y la eliminación de ruido [8].

- **Regularidad:** La regularidad puede ser considerada como una medida de la suavidad de la *wavelet* y corresponde a la capacidad de reconstruir fielmente una señal a partir de los coeficientes calculados en el proceso de transformación [15]. Si una señal presenta transiciones suaves, una *wavelet* regular es la opción más adecuada [8].

- **Simetría:** La simetría hace referencia a la linealidad de los filtros de reconstrucción de la transformada *wavelet*, esta es útil para evitar desfase en el procesamiento de las señales. "*Si la wavelet es simétrica, al verla como un filtro se puede decir que tiene fase lineal, si no es simétrica se introduce distorsión en la fase. Esto es de especial interés en aplicaciones de procesamiento de sonido e imágenes*" [16].

En la Tabla 1 se presenta a manera de resumen las diversas *familias wavelet* y sus respectivas propiedades [15].

Tabla 1. Propiedades de algunas familias *wavelet*

	Haar	Mexican Hat	Morlet	Dabeuchies	Symmlets	Coiflets	Gaussiana	Biorthogonal	Reverse Biortogonal
Propiedades	(haar)	(mexh)	(morl)	(dbN)	(symN)	(coifN)	(gausN)	(biorNr.Nd)	(rbiorNr.Nd)
Orden	...	...	...	1,2,....,452	1,2,...,412	1,2,....,52	1,2,...,442	1.1,1.3,...,6.82	1.1,1.3,...,6.82
Regularidad	No	Si	Si	Relativa	Relativa	Relativa	Si	Relativa	Relativa
Tamaño del Soporte	1	[-5,5]	[-4,4]	2N-1	2N-1	6N-1	[5,-5]	2Nd+1	2Nr+1
Longitud del filtro	2	...	...	2N	2N	6N	...	...	...
Simetría	Si	Si	Si	No	Aproximada	Aproximada	Si	Si	Si
Momentos de Desvanecimiento	1	...	...	N	N	2N	...	Nr	Nd
Función de Escala	Si	No	No	Si	Si	Si	No	Si	Si
CWT	Posible	Posible	Posible	Posible	Posible	Posible	Posible	Posible	Posible
DWT	Posible	No permite	No permite	Posible	Posible	Posible	No permite	Posible	Posible

Por otra parte, dado que no existe un método específico que permita realizar la selección apropiada de una *wavelet madre* a partir de sus propiedades [15], es necesario elegir la familia *wavelet* de manera visual, teniendo en cuenta la semejanza existente entre la forma de onda de la *wavelet madre* y la señal de análisis. Otra forma consiste en seleccionar mediante el método de *ensayo y error* la familia *wavelet* que mejor se adapte a las características de la imagen y elegir aquella con la que se obtienen mejores resultados teniendo en cuenta que cada una de las familias presentan propiedades específicas que se pueden adecuar según sea la aplicación en particular.

A continuación, se describirán las posibles aplicaciones de las familias *wavelet* de acuerdo al proceso que se desee realizar sobre la señal.

- **Preservación de Energía:** Si se desea preservar energía en la etapa de análisis, es importante el uso de una *wavelet* ortogonal y de soporte compacto. Sin embargo, es importante tener en cuenta que las *wavelet* ortogonales de soporte compacto son asimétricas (excepto la *wavelet haar*) [8].

- **Detección de Características:** Las *wavelet* de soporte compacto como *haar*, *daubechies2*, o *symlet2* son utilizadas en los casos en los que se desean encontrar características poco espaciadas en las imágenes, el soporte debe ser compacto dado que permite separar las zonas de

interés, porque si se utilizan familias *wavelet* con alto soporte se pueden obtener coeficientes incapaces de distinguir características individuales [8].

- **Eliminación de Ruido:** Tanto las *wavelets symlet* como las *daubechies* son buenas elecciones para eliminar ruido de señales o imágenes debido a su ortogonalidad, sin embargo también es una buena opción el uso de *wavelets biortogonales* [8].

- **Compresión:** Si lo que se desea es comprimir señales o imágenes, es recomendable la elección de las *wavelet biortogonales* ya que presentan un par de funciones *wavelet* y *escala* para el análisis y la síntesis. Además, el utilizar una *wavelet* con muchos momentos de desvanecimiento resultará en menos coeficientes *wavelet* significativos, lo que mejora la compresión de una imagen [8].

2.3 CORRELACIÓN DIGITAL DE IMÁGENES

La DIC es una técnica empleada en el reconocimiento de patrones debido a que permite comparar y determinar la similitud entre dos imágenes, se utiliza en áreas de investigación en torno a la deformación de materiales y la construcción dado que permite determinar cambios en la forma de un material sin entrar en contacto directo con él o con ninguna estructura en específico, pero también es ampliamente utilizada en campos como la robótica, imágenes médicas para el diagnóstico de enfermedades, análisis de imágenes satelitales, identificación de huellas dactilares, reconocimiento de rostros, aplicaciones biométricas, reconocimiento de caracteres, etc.

Lo anterior es posible porque las imágenes contienen información relevante que puede ser extraída en forma de características al aplicar sobre ellas diversas técnicas de procesamiento. A partir de las características principales obtenidas, se puede comparar una imagen (*imagen referencia*) de un objeto con otra (*imagen patrón*) y validar la similitud existente entre ellas. Sin embargo, como se trabaja con imágenes tomadas mediante cámaras digitales existen múltiples inconvenientes que se ven reflejados al hacer el proceso de localización de dichos patrones, puesto que el proceso de adquisición de la *imagen patrón* conlleva a que esta se vea expuesta a diferentes ángulos de inclinación, altitud, *SNR*, distorsión, cambios de luz por efecto del clima, etc., resultando en variaciones de color, orientación y tamaño, en comparación con la *imagen referencia*, por este motivo es de gran importancia el *pre procesamiento* de dichas imágenes para lograr una calidad tal que no se presente pérdida de información y que el uso de las técnicas de correlación sea eficiente.

Entre los métodos de correlación de imágenes aplicados al reconocimiento de patrones se destacan: métodos de correspondencia basada en área y métodos de correspondencia basada en características [17].

- **Métodos de correspondencia basada en área:** Establecen la similitud de dos imágenes en torno a los niveles de intensidad de los píxeles, haciendo una comparación píxel a píxel.

- **Métodos de correspondencia basada en características:** La similitud entre las imágenes se encuentra con base en un conjunto de características de interés como: bordes, áreas específicas de la fotografía, etc.

2.3.1 Métodos de correspondencia basada en área (ABM)

Los métodos ABM o también conocidos como métodos basados en intensidades (*Intensity based methods*) comparan la distribución de los niveles de gris de una imagen patrón con la distribución de los niveles de gris en otra imagen (imagen de referencia). La imagen de referencia es aquella de mayor tamaño y que permanece en una posición fija, la imagen patrón es aquella que se desea localizar dentro de la imagen de referencia; la comparación se realiza mediante técnicas que permiten calcular el nivel de semejanza de las dos imágenes, entre las más destacadas se encuentran la correlación cruzada y la correspondencia por mínimos cuadrados.

El fundamento principal de la correlación de imágenes mediante el método de intensidades es la técnica de convolución, siendo el cómputo del coeficiente de correlación el factor que permite establecer la medida de similitud entre dos imágenes.

La convolución discreta consiste en calcular el valor de un píxel en función de su propio valor y el de los píxeles que se encuentran a su alrededor (*vecindad*), mediante la aplicación de una operación matemática a partir de la que se obtiene un valor resultante para cada píxel en específico. La convolución se representa mediante la ecuación (2.20).

$$g(x,y) = f(x,y) * h(x,y), \tag{2.20}$$

donde $h(x,y)$ representa una máscara de convolución, $f(x,y)$ es la imagen de entrada y $g(x,y)$ es la imagen resultante. La expresión matemática para el caso bidimensional discreto, es decir, para la aplicación en imágenes digitales está dada por la expresión (2.21).

$$g(x,y) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(x_m, y_n) h(x - x_m, y - y_n). \tag{2.21}$$

En general, la convolución es una suma ponderada de todos los píxeles que se encuentran dentro del vecindario del píxel a analizar (*píxel central*). La máscara de convolución es una matriz que determina los coeficientes a aplicar sobre los píxeles de dicha área, esta debe ser de tamaño tal que sea posible determinar el centro de la matriz. El píxel que se encuentra en el centro corresponde a la posición del píxel que se está calculando, denominado también *píxel de salida*. La ventana de convolución, conocida también como *ventana deslizante*, se centra en cada píxel de una imagen de entrada generando un nuevo valor para el *píxel de salida*.

Para aplicar la máscara a una zona se multiplican los valores de los puntos que rodean al píxel sobre el que se está actuando (*píxel central*) por su correspondiente entrada o coeficiente en la máscara, luego se suman los resultados de dichos productos [18], dando origen al nuevo valor del *píxel central*.

Dado que la ventana de convolución empieza a deslizarse por la imagen, en cada deslizamiento se tendrá un nuevo *píxel central* sobre el que se aplica el proceso de convolución, por lo cual, cada valor resultante debe ubicarse en una nueva imagen o matriz ya que de no ser así, el valor obtenido reemplazaría al valor original *(pixel central)* y este sería utilizado para calcular el siguiente píxel (dado que quedaría como un pixel de la vecindad), resultando en un error en el

proceso. En la figura 2.13 se observa el proceso de convolución que ha sido descrito y en la figura 2.14 la matriz resultante de aplicar este proceso.

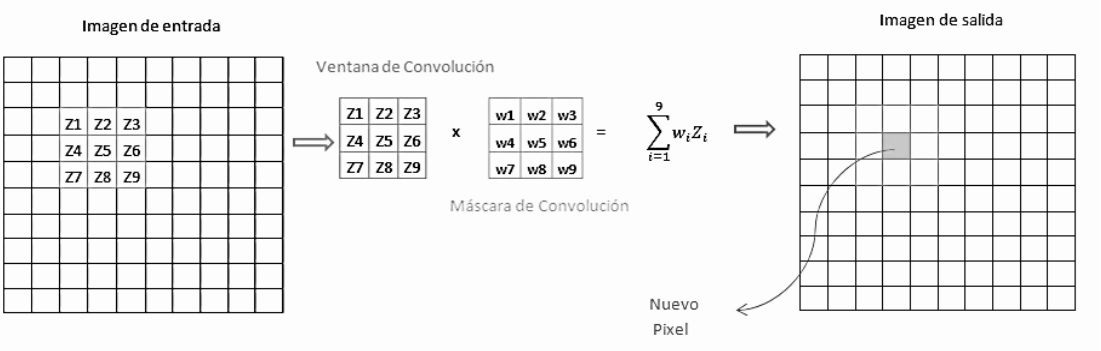

Figura 2.13 Proceso de convolución.

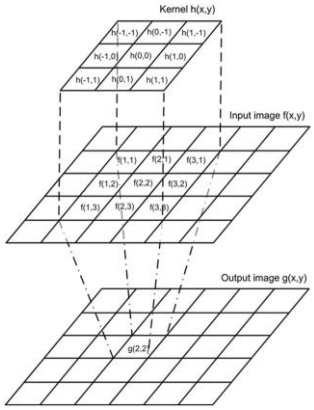

Figura 2.14 Obtención de una nueva matriz después de realizar el proceso de convolución.

2.3.1.1 Técnica de Correlación

La correlación de dos imágenes se utiliza principalmente en aplicaciones de reconocimiento para encontrar la mayor correspondencia entre una imagen adquirida y las imágenes de una base de datos previamente establecida, de forma que al calcular el coeficiente de correlación de una pareja de píxeles, el valor más alto que se obtenga corresponderá a la imagen buscada [19].

La técnica de correlación sigue los mismos principios de la convolución y está dada por la ecuación (2.22).

$$C(i,j) = \sum_{x=0}^{L-1} \sum_{y=0}^{K-1} w(x,y) f(x+i, y+j), \tag{2.22}$$

donde $w(x,y)$ corresponde a la *imagen patrón* representada por una matriz de tamaño $K \times L$, $f(x,y)$ corresponde a la *imagen de referencia* dentro de la cual se identificará el patrón y se representa como una matriz de tamaño $M \times N$ (ver figura 2.15). Se debe tener en cuenta que $K \leq M$y $L \leq N$.

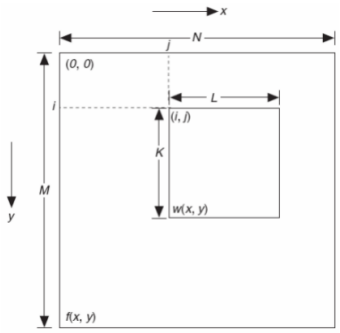

Figura 2.15 Representación en matrices de la Imagen de Referencia y la Imagen Patrón.

La *imagen patrón* hace un recorrido dentro de la *imagen referencia*, esto implica realizar una multiplicación entre los pixeles del patrón y los de la imagen referencia, la suma del resultado de dichas multiplicaciones corresponde al *coeficiente de correlación*. El valor máximo que se obtiene en $C(i,j)$ indica la posición que mejor se ajusta a la imagen patrón [20].

En general, el procedimiento que se lleva a cabo al aplicar la técnica de correlación se realiza de la siguiente manera:

1. Se selecciona la *imagen referencia* y la *imagen patrón* teniendo en cuenta que cumplan con la condición: $K \leq M$y $L \leq N$, es decir, la imagen de mayor tamaño corresponderá a la imagen referencia y la de menor tamaño será el patrón a localizar.

2. Una vez seleccionada la imagen patrón, se determina su centro. Este es el punto que se utiliza para hacer el desplazamiento alrededor de la *imagen referencia* y obtener los coeficientes de correlación.

3. Se determina dentro de la imagen de referencia, la región de interés donde posiblemente se encuentra el patrón.

4. Se calculan los coeficientes de correlación para todas las posiciones dentro de la matriz de la imagen de referencia.

5. Se analizan los valores obtenidos de los coeficientes de correlación para encontrar el máximo valor y validar la correspondencia de las dos imágenes.

Si la imagen es de gran tamaño, la aplicación de esta técnica implica un alto costo computacional debido a las múltiples operaciones matemáticas que deben ser realizadas, sin embargo este costo puede ser reducido bien sea, trabajando con patrones más pequeños, reduciendo la región de interés de la imagen de referencia en donde el patrón va a ser localizado, utilizando imágenes a escala de grises o mediante una combinación de lo previamente mencionado.

Existen diferentes alternativas para el reconocimiento de patrones mediante la técnica de correlación, entre las que se encuentra la Correlación Cruzada Normalizada (NCC*, Normalized Cross Correlation*), Correlación Cruzada Normalizada de Media Cero (ZNCC, *Zero mean Normalized Cross Correlation*) y Correlación Asimétrica (ASC, *Asymmetric Correlation*). La ZNCC es más robusta que la NCC, pero su costo computacional es mayor; la ASC por su parte es invariante a cambios de iluminación, es robusta al ruido extremo, pero su tiempo de procesamiento es muy lento [21].

Por tanto, una de las técnicas de correlación más simples pero más empleada gracias a su efectividad en el reconocimiento de patrones e imágenes es la

NCC ya que no es susceptible a variaciones de brillo [22], por lo que su aplicación resulta de suma utilidad en este contexto en donde se trabaja con imágenes que son adquiridas bajo diferentes condiciones de luz como consecuencia del entorno de adquisición. Sin embargo, esta técnica presenta algunos problemas cuando hay cambios bruscos en la rotación o en la escala de las imágenes, por lo cual debe considerarse situaciones en las que los cambios de rotación en las dos imágenes a comparar sean leves para que el reconocimiento de patrones se realice con mayor exactitud. A pesar de esto, se han obtenido resultados de exactitud en el reconocimiento de patrones como en el caso de Nadir Nourain Dawoud et al. (2012) [23] de hasta el 80% y del 73% en imágenes con fondo congestionado (*cluttered background*).

2.3.1.2 Correlación Cruzada

La correlación cruzada (CC, *Cross-Correlation*), conocida dentro del campo de procesamiento de imágenes como correlación, es una técnica matemática que permite cuantificar la similitud entre dos señales o datos que han sido obtenidos con variaciones temporales o variaciones espaciales [24]. Esta técnica es robusta al ruido y puede ser normalizada para su aplicación en la comparación o emparejamiento de patrones (*pattern matching*) [25].

2.3.1.3 Correlación Cruzada Normalizada

La NCC se define según la ecuación (2.23), donde $f(x,y)$ representa la imagen original y $h(x,y)$ es la imagen patrón.

$$NCC = \frac{\sum_x \sum_y f(x,y) h(x-x_m, y-y_m)}{\sqrt{\left[\sum_x \sum_y f(x,y)^2\right]\left[\sum_x \sum_y h(x-x_m, y-y_{m)}{}^2\right]}}. \qquad (2.23)$$

La NCC resulta de gran utilidad en escenas en las que se presentan variaciones de iluminación; la principal ventaja que tiene el uso de los

coeficientes de correlación es que la comparación de las imágenes se ve reducida a un valor escalar. Para el caso de la NCC, este valor puede darse dentro de un rango $[-1, 1]$; es decir, el valor máximo a obtener es 1 y sucede cuando la imagen objetivo se encuentra dentro de la imagen de referencia siguiendo exactamente el mismo patrón, entre más grande sea la similitud entre las dos imágenes, más alto será el coeficiente de correlación.

Dado que las imágenes objetivo muchas veces no son adquiridas de forma que su patrón coincida exactamente con el patrón dentro de la imagen de referencia, se tendrá un valor de correlación diferente a la unidad, sin embargo el valor más cercano a 1 será el punto a considerar en donde se ha presentado la mayor correlación e indicará que allí se encuentra el patrón que se ha deseado localizar.

En general, se puede afirmar que si se obtiene un valor de correlación superior a 0.7 o 0.8, el reconocimiento del patrón es satisfactorio. Estos valores de la matriz de correlación pueden ser visualizados por medio de un mapa de superficie (ver figura 2.16), en donde los picos más altos representan los puntos donde se encuentra la máxima correlación.

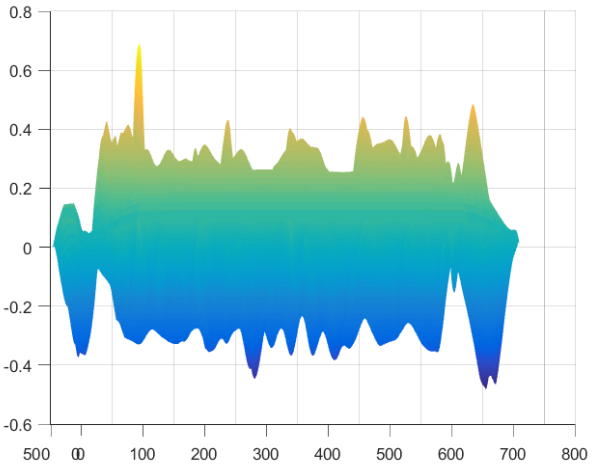

Figura 2.16 Representación del máximo coeficiente de correlación.

Existen otras alternativas para la comparación de patrones que pueden ser utilizadas en un sinnúmero de aplicaciones, sin embargo, su eficacia en el reconocimiento depende de la aplicación en la cual se utilice cada técnica. Entre estas se encuentran: Suma de Diferencia Absoluta (SAD, *Sum of Absolute difference*), Suma de Diferencias al Cuadrado (SSD, *Sum of Squared Difference*), Suma de Diferencia Absoluta Optimizada (OSAD, *Optimized Sum of Absolute Difference*), Suma de Diferencias al Cuadrado Optimizada (OSSD, *Optimized Sum of Squared Difference*). Sin embargo, en este trabajo se utilizará la NCC como el método de correspondencia de imágenes basado en área para el desarrollo de uno de los algoritmos de reconocimiento de placas vehiculares.

2.3.2 Métodos de correspondencia basada en características (FBM)

A diferencia de los métodos basados en área, los métodos de correspondencia basada en características no necesitan trabajar con los valores de intensidad de los píxeles, sino que involucran etapas de detección, descripción y comparación de características para establecer una similitud, motivo por el cual este tipo de métodos es útil cuando la imagen presenta cambios de iluminación [26].

Existen dos formas de representar una imagen a partir de sus características locales o globales. Las características globales representan una imagen en un único *vector de características* que describe toda su información, donde los valores del vector puede representar aspectos de la imagen como color, textura o forma, siendo posible comparar dos imágenes a partir de la comparación del *vector de características* global extraído para cada una. Por su parte, la representación de una imagen mediante características locales se realiza utilizando descriptores locales que se extraen a partir de regiones de interés de la imagen.

Para lograr establecer la similitud entre dos imágenes, el primer paso consiste en la detección de las características; en esta etapa se localizan las partes de la imagen en donde se encuentra la información más relevante, una vez se han detectado las zonas de interés en la imagen se procede a utilizar descriptores los cuales se clasifican según la característica de interés, es decir, pueden ser descriptores de color, textura o forma.

- **Descriptores de Color:** El color es una de las características más utilizadas puesto que es invariante a los cambios en rotación, traslación o escala, sin embargo se ve afectado por cambios de iluminación.

- **Descriptores de Textura:** Describir una imagen mediante textura es uno de los mejores métodos para establecer la correlación o correspondencia de dos imágenes. Entre los descriptores de textura se encuentran: Descriptor Homogéneo de Textura (HTD, *Homogeneous Texture Descriptor*), Descriptor de Histograma de Bordes (EHD, E*dge Histogram Descriptor*), Transformada de Característica Invariante a Escala (SIFT, *Scale Invariant Feature Transform*), Características Robustas de Alto Rendimiento (SURF, *Speeded Up Robust Feature*) y Ubicación del Gradiente e Histograma de Orientación (GLOH, *Gradient Location and Orientation Histogram*) [27]. Estos últimos son denominados descriptores locales y se caracterizan por ser más robustos al ruido, a las obstrucciones o la congestión en la imagen.

- **Descriptores de Forma:** En muchas aplicaciones la forma de un objeto es la más adecuada para establecer la comparación o correspondencia entre dos imágenes, en estos casos es de suma importancia que los descriptores utilizados sean invariantes al escalamiento, rotación o traslación para hacer una buena identificación del patrón. Para esto, se han desarrollado métodos basados en frontera (*boundary based*) y basados en región (*region based*): los métodos basados en frontera solo requieren trabajar con los píxeles del contorno, sin embargo no son

adecuados para ser aplicados cuando se tienen objetos con formas complejas. Entre los descriptores para este tipo de método están los códigos de cadena y los descriptores de Fourier.

Por otra parte, los métodos basados en región son de utilidad cuando se tienen formas complejas en una imagen dado que su aplicación requiere de tanto los píxeles de contorno como los que se encuentran alrededor de la región de interés. Como descriptores se utilizan momentos geométricos como el de Hu o Zernike [27].

2.3.2.1 Histograma de Gradientes Orientados (HOG)

Dado que en la etapa de adquisición de la imagen se pueden presentar variaciones como iluminación, rotación y/o escala, el principal objetivo de los descriptores consiste en obtener las mínimas variaciones posibles en dichos parámetros. El Histograma de Gradientes Orientados (HOG, *Histogram of Oriented Gradients*) es un descriptor de la imagen que se basa en el cálculo de gradientes, los cuales representan los cambios de intensidad de una imagen.

I. Cálculo del Gradiente

El gradiente brinda información de los cambios de intensidad de la imagen, este se calcula para todos los píxeles y se define como un vector teniendo en cuenta la dirección donde el cambio de intensidad es máximo y la magnitud del cambio en la dirección de máxima variación (la longitud del vector es proporcional a la magnitud). Esto permite determinar las variaciones locales alrededor de cada píxel tales como su contraste y forma local [28].

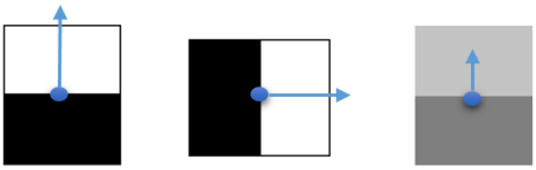

Figura 2.17 Cambio de Gradiente de un pixel
(a) Gradiente vertical, (b) Gradiente Horizontal, (c) Gradiente Vertical de menor magnitud.

En la figura 2.17 se aprecia el cambio de gradiente para un píxel; en (a) el gradiente tiene una dirección vertical debido a que el cambio de contraste se presenta en esta dirección, además la magnitud es elevada debido a que dicho cambio también los es. En (b), la dirección del gradiente es vertical considerando que en esta dirección ocurre el cambio de contraste y finalmente en (c), se puede evidenciar que a menor cambio de contraste, menor es la magnitud del gradiente.

El cálculo del gradiente se puede realizar de varias maneras, sin embargo en el contexto del *descriptor HOG* se calcula a partir de la diferencia de la intensidad de los pixeles adyacentes en la dirección horizontal y vertical.

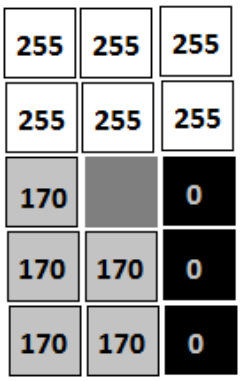

Figura 2.18 Niveles de gradiente de los píxeles de una imagen.

Si se considera una imagen como la de la figura 2.18, para el píxel central el cálculo del gradiente en la dirección dx será 170, mientras que en la dirección dy el gradiente será 85 siguiendo las ecuaciones (2.24) y (2.25) respectivamente.

$$dx = I(x+1, y) - I(x-1, y), \tag{2.24}$$

$$dy = I(x, y+1) - I(x, y-1). \tag{2.25}$$

Este proceso se realiza para todos los píxeles de la imagen obteniendo como resultado información local alrededor de cada uno de ellos.

A partir de las diferencias dx y dy obtenidas se pueden calcular las magnitudes y las orientaciones globales de los gradientes trasladando dichas diferencias al eje de coordenadas como se aprecia en la figura 2.19.

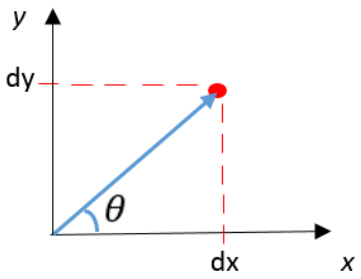

Figura 2.19 Magnitud y dirección de un vector.

En donde el vector de la imagen corresponde al *vector gradiente* para el píxel central a partir del cual se calcula la *orientación del gradiente* la cual se define como el ángulo que forma el vector con el eje horizontal y se calcula mediante la ecuación (2.26). La *magnitud del gradiente* por su parte, corresponde a la longitud del vector y se define según la ecuación (2.27).

$$\theta(x, y) = \arctan\left(\frac{dy}{dx}\right), \tag{2.26}$$

$$g(x, y) = \sqrt{dx^2 + dy^2}. \tag{2.27}$$

Comúnmente los cambios de intensidad más bruscos se concentran en el contorno de la imagen, por lo que se obtendrán valores más altos de *magnitud del gradiente*, mientras que la *orientación del gradiente* proporciona información sobre la forma del contorno. Esto permite caracterizar la imagen según su forma y distinguirla así de otros objetos.

El cálculo del gradiente proporciona información local de cada pixel, por lo cual se hace necesario convertir dicha información local en información global que caracterice toda la imagen y se represente mediante un *vector de características* [28].

II. Cálculo de los Histogramas de Orientación

Para realiza el cálculo de los histogramas de orientación, la imagen es dividida en un número de celdas con el fin de obtener el histograma de cada una de ellas, para esto se define el tamaño de las celdas en las que será dividida la imagen tanto en ancho como en alto, los valores típicos para este parámetro varían entre 6 y 8 píxeles. Enseguida, se divide el rango de orientaciones del gradiente, cabe resaltar que las orientaciones del gradiente pueden darse de 0° a 360° si se tiene en cuenta el signo, o de 0° a 180° en caso contrario; para este último, si dos gradientes tienen la misma dirección pero sentido diferente, se asignan al mismo intervalo; y finalmente se divide dicho rango de orientaciones en un número de intervalos predefinido (comúnmente son 9 intervalos). En la figura 2.20 se observa la orientación del gradiente desde 0° hasta 180° dividido en 9 intervalos en donde cada uno de estos tiene un rango de 20°.

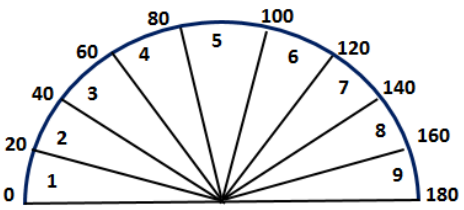

Figura 2.20 Rango de orientación del gradiente

A partir de esto, cada gradiente de la celda (compuesto por orientación y magnitud) queda asignado a un rango de orientación, y cada intervalo tendrá un conjunto de gradientes cuya orientación se encuentra en dicho rango. Luego, el valor de cada intervalo dentro del histograma de orientación estará

dado por la suma de las magnitudes de los gradientes que se encuentran en dichos intervalos.

Matemáticamente el cálculo del histograma se define según la ecuación (2.28),

$$h_{i,j}(k) = \sum_{(x,y)\in C_{i,j}} w_k(x,y)g(x,y), \tag{2.28}$$

donde $h(k)$ es el histograma para un intervalo k, $g(x,y)$ es la magnitud de cada pixel de la celda y $w_k(x,y)$ es la asociación del gradiente a dicho intervalo, es decir, w_k tomará el valor de 1 para todos los gradientes cuya orientación está dentro del rango del intervalo k, y 0 en los casos en donde las orientaciones están fuera de dicho rango como se denota en (2.29)

$$w_k = \begin{cases} 1, & si\ (k-1)\delta\theta \le \theta(x,y) < k\delta \\ 0, & en\ caso\ contrario. \end{cases} \tag{2.29}$$

Sin embargo, uno de los problemas que se presenta en esta asignación es que aunque existan gradientes con orientaciones similares es posible que queden asignados a intervalos diferentes. Este error de asignación puede conllevar a que pequeñas variaciones en la imagen de entrada repercutan significativamente en el vector de características que representa la imagen; por este motivo se debe hacer una interpolación en orientación, la cual consiste en asignar el pixel a los dos intervalos más cercanos, asignando un peso proporcional a la distancia desde la orientación del gradiente al centro de cada intervalo como se muestra en la figura 2.21. Dicha distancia se utiliza para calcular el valor $w(k)$ siguiendo la ecuación (2.30); en donde, $\theta(x,y)$ corresponde a la orientación del gradiente, $\theta(k)$ es el centro del intervalo y $\delta\theta$ el rango de los intervalos del histograma si la distancia es igual a cero.

$$w_k(x,y) = \max\left(0, 1 - \frac{\theta(x,y) - \theta_k}{\delta\theta}\right). \tag{2.30}$$

Si la distancia resultante es cercano a 0, el valor de w_k será cercano a 1; mientras que si el valor de la distancia es cercano a $\delta\theta$, w_k será cercano a 0. Este valor w_k será el que se utiliza en la ecuación descrita previamente.

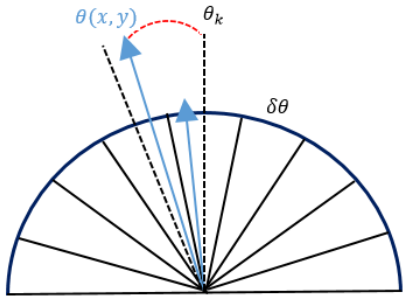

Figura 2.21 Distancia entre el gradiente y el centro del intervalo.

El cálculo del histograma se realiza para cada una de las celdas de la imagen y por tanto, cada celda tendrá su propio histograma como se muestra en la figura 2.22.

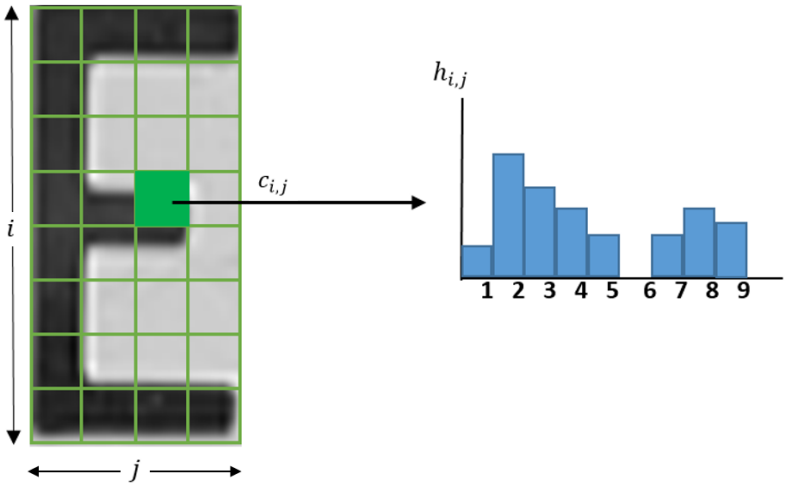

Figura 2.22 Histograma de cada celda de una imagen.

Sin embargo, si se tienen dos píxeles muy cercanos, es posible que estos sean asignados a celdas diferentes, presentándose el mismo problema anterior, en el que cambios muy pequeños en la descripción de la forma del objeto de la imagen pueden conllevar a un error en la descripción global de dicha imagen; este inconveniente también es fácil de solucionar mediante el uso de la interpolación espacial. En este caso, cada uno de los pixeles es asignado a las

cuatro celdas más cercanas con un peso proporcional a la distancia desde el pixel hasta el centro de la celda, calculando así la distancia desde el píxel hacia el centro de la celda en las direcciones x e y como se muestra en la figura 2.23.

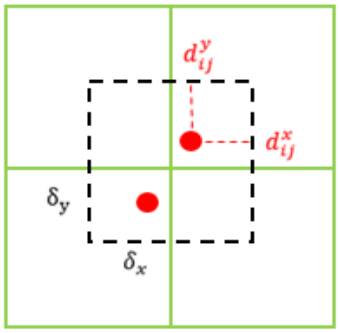

Figura 2.23 Calculo de la distancia desde el pixel hasta el centro de la celda.

Las distancias obtenidas se utilizan dentro de la ecuación (2.31) y (2.32) para calcular los factores w_{ij} que representan la asignación de los píxeles a la celda, así:

$$w_{ij}^{x}(x,y) = \max\left(0, 1 - \frac{d_{ij}^{x}}{\delta x}\right), \qquad (2.31)$$

$$w_{ij}^{y}(x,y) = \max\left(0, 1 - \frac{d_{ij}^{y}}{\delta y}\right), \qquad (2.32)$$

donde d_{ij}^{x}, corresponde a la distancia desde el píxel hacia el centro de la celda en dirección x; d_{ij}^{y} es la distancia desde el píxel hacia el centro de la celda en dirección y, δx y δy es la distancia entre los centros de dos de las celdas.

Finalmente, los resultados obtenidos en w_{ij}^{x}, w_{ij}^{y} y w_k son utilizados para obtener así el valor final del histograma de cada intervalo k y de cada celda $C_{i,j}$ de la imagen mediante la ecuación (2.33)

$$h_{i,j}(k) = \sum_{(x,y)\in C_{i,j}} w_k^{x}(x,y)\, w_k^{y}(x,y)\, w_k(x,y) g(x,y). \qquad (2.33)$$

III. Cálculo del Descriptor HOG

A partir de los histogramas de orientación obtenidos para cada una de las celdas de la imagen, se procede a conformar el vector de características [28].

Dado que las imágenes de entrada pueden presentar cambios de iluminación como consecuencia de las condiciones de su adquisición, estos cambios a su vez se verán reflejados en la intensidad del gradiente y por consiguiente, en los valores del histograma de orientación de las celdas (a mayor contraste en la imagen, mayores serán los valores obtenidos en el histograma); por tal motivo, es necesario normalizar los valores de los histogramas con el fin de que la magnitud de cada histograma sea igual para todas las imágenes, pero como la iluminación a lo largo de una imagen no es uniforme, no es conveniente realizar una única normalización a toda la imagen, sino que se agrupan varias celdas dentro de un bloque de tamaño bxb[4]; los histogramas de cada una de las celdas del bloque se concatenan para formar un vector que represente dicho bloque y con esto realizar la normalización de los histogramas a través de la normalización del vector correspondiente a cada bloque. Esta normalización se realiza siguiendo la ecuación (2.34) y su concatenación se representa mediante la figura 2.24

$$v' = \frac{v}{\sqrt{\|v\|_2^2 + \varepsilon}} \quad \text{donde} \quad v = (x_1, x_2, \dots, x_n), \qquad (2.34)$$

[4] En el descriptor HOG, se utilizan comúnmente bloques de dos celdas.

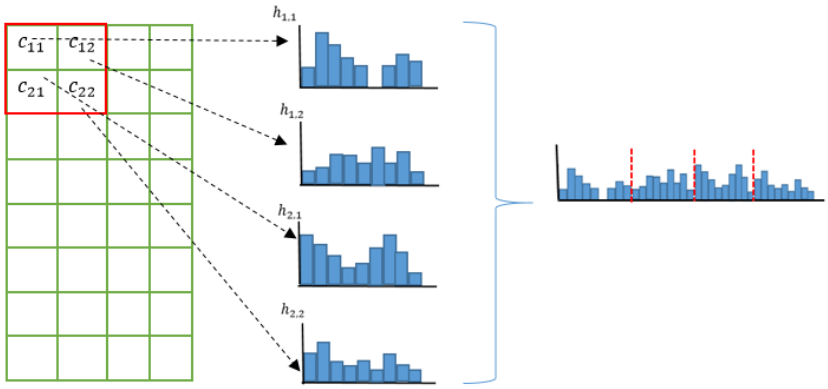

Figura 2.24 Histograma de un bloque de celdas.

en donde v representa cada elemento del vector, y $\|v\|$ es la norma de dicho vector. La norma se define según la ecuación (2.35)

$$\|v\| = \sqrt{\sum {x_i}^2}\ . \tag{2.35}$$

El valor ε por su parte, se utiliza para evitar divisiones por 0 (si la intensidad de todo el bloque es constante y por tanto, la magnitud de todo el gradiente es cero); sin embargo, es un valor muy pequeño, casi que despreciable.

Es de importancia resaltar que los bloques que se conforman deben solaparse entre sí para obtener un descriptor que sea robusto a cualquier cambio en la forma del objeto o patrón; y la separación de cada bloque es de una celda entre sí. Finalmente, se obtiene el *descriptor HOG* concatenando los histogramas normalizados de cada uno de los bloques solapados.

En general, los parámetros del descriptor HOG están dados por:

- **Tamaño de cada celda**: Según el tamaño de la imagen y el tamaño de la celda, se podrá obtener el número de celdas en la imagen. El valor típico del tamaño de la celda es de 8×8.

- **Signo del gradiente:** Si se considera un gradiente desde 0~180° o de 0~360°.

- **Número de intervalos del Histograma:** Comúnmente 9 intervalos.

- **Número de celdas dentro de un bloque:** Bloques de 2×2 celdas.

La dimensión final del *descriptor HOG* estará dada en función de los parámetros descritos anteriormente; sin embargo, estos parámetros pueden variar según la aplicación de reconocimiento de objetos que se desea realizar.

La dimensión del vector HOG está dada según la ecuación (2.36)

$$n = \frac{\#\ bloques\ \times \#\ celdas}{bloque\ \times \#\ intervalos}, \qquad (2.36)$$

donde, el número de bloques está definido según (2.37), y se calcula tanto en la dirección x como en la dirección y

$$\#\ bloques = \#\ celdas - {}^{\#\ celdas}/_{bloque} + 1. \qquad (2.37)$$

Finalmente, después de obtenidos los puntos de interés (figura 2.25) de las imágenes a comparar (imagen referencia e imagen patrón) se puede determinar la correspondencia existente entre ellas (ver figura 2.26). Algunos de los métodos que permiten realizar este proceso son: distancia euclidiana, la transformada de característica invariante a escala SIFT y el algoritmo de Consenso de Muestra Aleatoria (RANSAC, *RANdom Sample and Consensus*), el cual permite seleccionar únicamente los puntos de mayor correspondencia (*inliners*) y descartar aquellas características que no son similares en ambas imágenes.

Figura 2.25 Detección de diversas características en una imagen.

Figura 2.26 Correspondencia entre dos imágenes.

2.3.2.2 Distancia Euclidiana

La Distancia Euclidiana *(Euclidean Distance)*, se utiliza para medir la longitud de un segmento entre dos puntos en un plano o en un espacio de N dimensiones. En un plano, esta distancia está determinada según el teorema de Pitágoras, siguiendo la ecuación (2.38)

$$d(A,B) = \sqrt{(b_1 - a_1)^2 - (b_2 - a_2)^2}. \qquad (2.38)$$

Del mismo modo, para un espacio de N dimensiones la distancia euclidiana entre dos vectores se define mediante la ecuación (2.39)

$$d(A,B) = \sqrt{\sum_{i=1}^{N}(b_i - a_i)^2} = \sqrt{(b_1 - a_1)^2 + (b_2 - a_2)^2 + \cdots + (b_N - a_N)^2}\,. \qquad (2.39)$$

El cálculo de la distancia euclidiana, para el reconocimiento de patrones, se realiza a partir de dos vectores: el primero corresponde al vector de características extraídas de la imagen de referencia, y el segundo corresponde al vector de características extraídas de la imagen de prueba; una distancia de cero o cercana a cero indica que los vectores son semejantes permitiendo determinar si poseen las mismas características.

En conclusión, tanto los métodos ABM como los FBM presentan ventajas y desventajas. Los métodos basados en área son más fáciles de implementar que los métodos basados en características; sin embargo, si la imagen no presenta muchas texturas es posible que se obtengan falsas correspondencias; además, no funcionan bien bajo diferentes ángulos de visión debido a los cambios producidos por la iluminación. Los métodos basados en características, por su parte, son útiles cuando la imagen contiene muchos detalles relevantes a partir de los cuáles se pueden extraer características, estos son más rápidos que los métodos basados en convolución y son relativamente insensibles a cambios de iluminación [17]. Por lo anterior, la elección de utilizar un método u otro debe hacerse considerando la aplicación que se desea desarrollar de forma que sea posible aprovechar al máximo las ventajas que cada uno de estos presenta.

Con ánimo de establecer el mejor método de correspondencia de imágenes que se puede aplicar al reconocimiento de placas vehiculares, se desarrollaron dos algoritmos en los cuáles se utiliza en uno de ellos un método de ABM y en el otro, un método de FBM, permitiendo así evaluar el desempeño de estos dos y determinar cuál es el más apropiado en este campo de aplicación.

CAPÍTULO 3

ALGORITMOS PARA EL RECONOCIMIENTO DE PLACAS VEHICULARES

El diseño propuesto de los algoritmos implementados se realizó conforme a las etapas presentes en la figura 3.1.

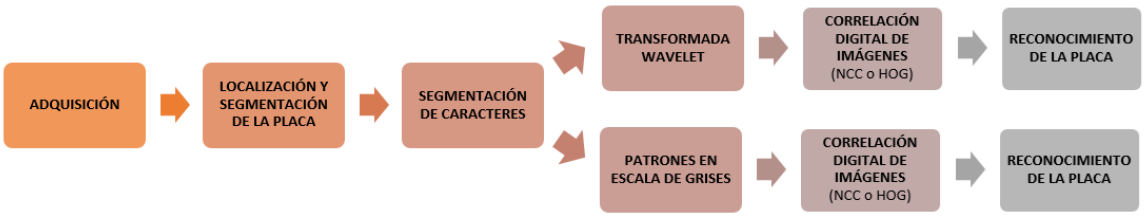

Figura 3.1 Diseño propuesto para los algoritmos de reconocimiento de placas vehiculares.

El diseño inicia con la *adquisición* de la fotografía del vehículo, seguido de la *localización* de la placa dentro de la imagen y su posterior *segmentación*. Una vez obtenida la placa se realiza la *segmentación de caracteres* de forma individual para luego aplicar la DWT a cada uno de ellos. Posteriormente se aplican técnicas de *correlación digital de imágenes* para establecer la medida de similitud entre cada carácter extraído de la placa vehicular y cada uno de los caracteres de referencia previamente almacenados en una base de datos. Finalmente, se hace el reconocimiento de la placa vehicular teniendo en cuanta los valores de similitud.

Paralelo a esto y a manera de comparación se realizó el emparejamiento de patrones a partir de caracteres en escala grises es decir, sin aplicar la DWT tanto a los caracteres segmentados de la placa vehicular, como a los caracteres de la base de datos.

Para el diseño propuesto se desarrollaron dos algoritmos, en los cuales, la etapa de correlación digital de imágenes para la comparación de caracteres es

diferente: en uno de ellos se hace uso de la correlación cruzada normalizada como método de correlación basado en área, mientras que en el otro se utiliza el descriptor *HOG* como método de correlación basado en características (ver figura 3.2).

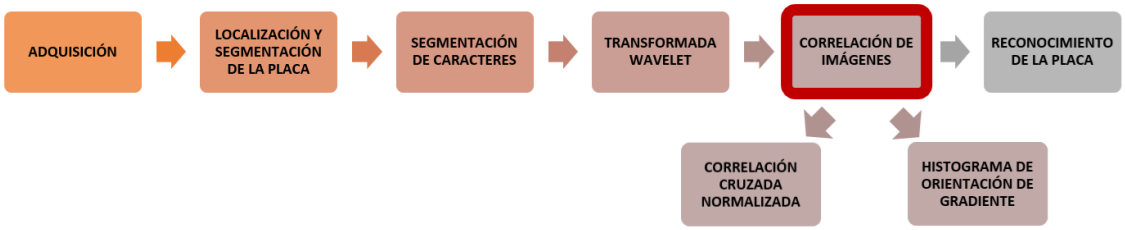

Figura 3.2 Esquema de los algoritmos propuestos.

A continuación se describirán cada una de las etapas del diseño propuesto.

3.1 ADQUISICIÓN

En la etapa de *adquisición* se obtuvieron 438 fotografías de placas de vehículos[5] mediante la cámara de un dispositivo móvil Samsung SM-G930F con 4.7MB de resolución[6]. Las tomas fueron realizadas durante el día y de manera frontal, bajo diferente luminosidad y con las mejores condiciones de captura posibles, con el fin de minimizar variaciones en su adquisición. La distancia de captura entre la placa y la cámara fue de aproximadamente 150 cm y 200 cm para garantizar tomas similares y que los caracteres de las placas conservaran aproximadamente el mismo tamaño. Las imágenes resultantes fueron de 2160×2160 pixeles de dimensión (ver figura 3.3), formato de compresión JPG y fueron almacenadas para la creación de la base de datos con la cual se desarrollaron los algoritmos.

[5] Se consideraron automóviles con placas en buen estado para la captura de las fotografías. Solo se tuvieron en cuenta vehículos privados de Colombia, cuya característica es fondo amarillo con caracteres negros.

[6] Se denomina resolución al número de pixeles por pulgada utilizados en el proceso de adquisición, esta se indica en dpi (*dots per inch* = puntos por pulgada). Mientras mayor es la resolución mayor es la fidelidad de la imagen, es decir se obtienen más detalles de ella.

Figura 3.3 Fotografía de un carro con su placa vehicular.

3.2 LOCALIZACION Y SEGMENTACION DE LA PLACA VEHICULAR

Dado que la etapa de reconocimiento depende directamente de la localización de la placa y la posterior segmentación de los caracteres, es imprescindible la selección de técnicas que garanticen la correcta segmentación de la placa vehicular dentro de la fotografía. Es por esto que se desarrollaron una serie de algoritmos para localizar y segmentar la placa con el fin de seleccionar aquel que presentara un mejor desempeño (ver apéndice I). Para hacerlo se involucraron técnicas de procesamiento de bajo nivel y nivel medio, utilizando la herramienta Matlab R2015a en conjunto con el *toolbox* de procesamiento de imágenes (*Image Processing Toolbox*).

Como parte del procesamiento de bajo nivel se realizaron las operaciones que se indican en la figura 3.4.

Figura 3.4 Operaciones de bajo nivel aplicadas.

- **Redimensionamiento**

Inicialmente se cuenta con una fotografía digital cuyo tamaño es 2160×2160 pixeles, dado que se trabajó con múltiples imágenes y de manera simultánea, fue imperativo reducir su tamaño para facilitar el tiempo de procesamiento, para esto y como primera medida, se utilizó la función '*imresize'* para cambiar las dimensiones de la imagen, en este caso la imagen resultante es 720×860 pixeles.

- **Conversión a escala de grises**

Comúnmente, las fotografías que se capturan mediante una cámara digital corresponden a imágenes RGB, es decir imágenes en color, sin embargo para reducir aún más el tiempo de procesamiento se realiza la conversión de la imagen original a una imagen en escala de grises. El hacer este tipo de conversión implica que la cantidad de información representada es menor, pero aun así se considera que la información es suficiente para extraer las principales características de la imagen y lograr un adecuado procesamiento. Esta conversión se realizó mediante la función '*rgb2gray'* de Matlab.

- **Reducción de ruido**

Para mitigar el efecto del ruido en la imagen es necesario utilizar un filtro de suavizado para reducir las variaciones de intensidad entre píxeles vecinos. Como método de reducción de ruido se utilizó el filtro de media, con una matriz de filtrado de tamaño 5×5 pixeles, este filtro presenta la ventaja de conseguir

que las intensidades de los objetos pequeños se mezclen con el fondo de la imagen de forma que permita detectar los objetos de mayor tamaño, resultando en una imagen con menor ruido y mayor suavizado. En la figura 3.5 (izquierda) se observa la imagen con la aplicación del filtro de media.

- **Realce de bordes**

Dado que es en los bordes donde se concentra la mayor cantidad de información relevante de una imagen, se hizo uso de la técnica *Unsharp Masking* (USM, Máscara de desenfoque) mediante la función '*imsharpen'* para mejorar el enfoque de los bordes y facilitar su detección en las siguientes etapas (ver figura 3.5 (derecha)).

Figura 3.5 Imagen en escala de grises.
Filtro de media (Izquierda), Realce de bordes (Derecha).

En cuanto al procesamiento de nivel medio, se realizó un crecimiento de regiones mediante procesamiento morfológico y se seleccionaron las regiones de interés mediante el etiquetado de componentes conectados, dado que estas técnicas fueron empleadas en el algoritmo de localización de la placa que presentó el mejor desempeño (ver Apéndice I). Las técnicas utilizadas se indican en la figura 3.6.

Figura 3.6 Operaciones de nivel medio.

- **Procesamiento Morfológico**

Vale la pena resaltar que las operaciones morfológicas no se aplican mediante pasos definidos, sino por ensayo y error, teniendo los resultados que se desea obtener.

Las primeras operaciones morfológicas empleadas fueron la erosión y la apertura, como se muestra en la figura 3.7, estas transformaciones se realizaron mediante las funciones '*imerode'* e '*imdilate'* respectivamente y se utilizo como elemento estructurante[7] un cuadrado con 5 pixeles de ancho, definido mediante la función '*strel'*. Estas operaciones permiten rellenar huecos en la imagen al igual que unir líneas discontinuas, dando como resultado bordes definidos.

[7] El tamaño y forma del elemento estructurante se selecciona a priori, y en función de la forma que se desea extraer.

Figura 3.7 Operaciones Morfológicas. Erosión (derecha), apertura (izquierda).

Enseguida se realizó una operación adicional a partir de la diferencia de las dos imágenes resultantes. El resultado se muestra en la figura 3.8.

Figura 3.8 Resta de la erosión y la dilatación.

Sin embargo, la imagen resultante aún contenía muchas regiones que no eran de interés, por lo cual se continuó el procesamiento morfológico, iniciando con un filtro de *bottom-hat* mediante la función '*imbothat'* y utilizando como elemento estructurante un disco de 5 píxeles de radio. Este filtro permitió resaltar las características claras sobre un fondo oscuro. En la figura 3.9 se observa la imagen después de aplicar el filtro *bottom-hat.*

Figura 3.9 Imagen después del filtro *bottom-hat.*

Con el fin de binarizar la imagen y seleccionar los pixeles cuyas intensidades fueran mayores que un umbral definido, se realizó un proceso de umbralización como se muestra en la figura 3.10.

Figura 3.10 Umbralización.

A continuación, se realizaron nuevamente las operaciones morfológicas de cierre y apertura con un elemento estructurante rectangular ya que esta forma se adapta mejor a la placa vehicular, el tamaño seleccionado para el elemento estructurante rectangular es de 20×60 pixeles.

El cierre permitió rellenar huecos en la imagen y unir pixeles relacionados como se observa en la figura 3.11.

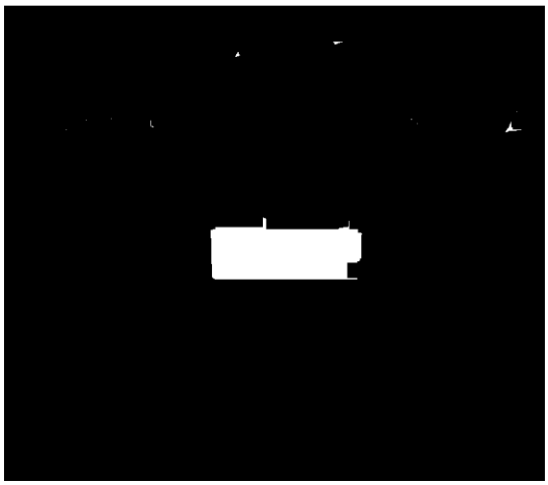

Figura 3.11 Resultado de aplicar cierre.

Por otra parte, la apertura suavizó los contornos de la región y eliminó los objetos más pequeños que el elemento estructurante, obteniendo la imagen de la figura 3.12.

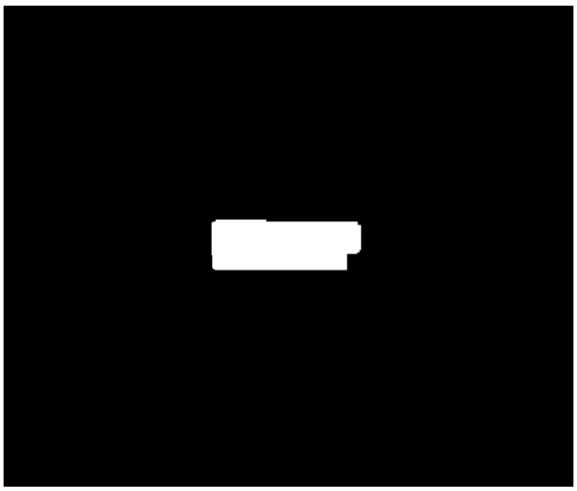

Figura 3.12 Resultado de aplicar cierre.

Finalmente, se aplica nuevamente la operación morfológica de dilatación con elemento estructurante en forma de disco con radio de 10 pixeles, para garantizar que la región resultante contenga la totalidad de la placa vehicular (ver figura 3.13).

Figura 3.13 Dilatación.

- **Etiquetado de componentes conectados**

En esta etapa se utilizó la función '*bwlabel'* la cual permite calcular los componentes conectados en caso de existir.

Dado que las operaciones morfológicas previamente aplicadas identifican una única región que corresponde a la región de interés, se procede a etiquetar dicha región. Por último, se recorta y se segmenta la placa vehicular como se muestra en la figura 3.14.

Figura 3.14 Extracción de la placa vehicular.

3.3 SEGMENTACIÓN DE LOS CARACTERES

Una vez obtenida la placa vehicular, se procede a realizar la segmentación de cada uno de los caracteres, para esto se aplica nuevamente la técnica de segmentación basada en crecimiento de regiones cuyos pasos son similares a los descritos anteriormente.

Para mejorar el corte de la placa extraída y eliminar zonas indeseadas, la imagen es binarizada utilizando un umbral de 120, permitiendo seleccionar los pixeles cuya intensidad sea mayor a dicho umbral. Luego se etiquetan las regiones y se selecciona aquella cuya área sea mayor a 4000 pixeles, obteniendo la imagen de la figura 3.15.

Figura 3.15 Placa vehicular con mejor corte.

A la placa recortada se le aplica el filtro *bottom-hat* con un disco de 12 píxeles como elemento estructurante, el resultado se observa en la figura 3.16.

Figura 3.16 Filtro b*ottom-hat* sobre la placa.

Nuevamente, se binarizó la imagen con un umbral de 100 para filtrar los pixeles de las regiones indeseadas, permitiendo la distinción de los caracteres como se muestra en la figura 3.17.

Figura 3.17 Placa Umbralizada.

Utilizando la función '*bwlabel'* se detectan los componentes conectados. Una vez hecho esto, se utiliza la función '*regionprops'* y se busca todas aquellas regiones cuyas áreas sean mayor al 1.5% de la imagen, este valor se define de manera aproximada y funciona como un filtro para detectar únicamente las zonas más grandes que corresponde a los seis caracteres de la placa vehicular, eliminando así las demás regiones que se tenían anteriormente (ver figura 3.18).

Figura 3.18 Regiones de interés de la placa vehicular.

Como paso adicional y en caso de que la placa resultante aún contenga regiones indeseadas, es necesario eliminarlas para garantizar un mejor proceso de etiquetado, por lo tanto se especificó un valor aproximado de 0.7%, de modo que todos los objetos que ocupen un área mayor a este valor sean mostrados, resultando entonces en una imagen que contiene únicamente dichas regiones.

Para finalizar, se etiquetan las regiones y se segmenta cada uno de los caracteres como se muestra en la figura 3.19.

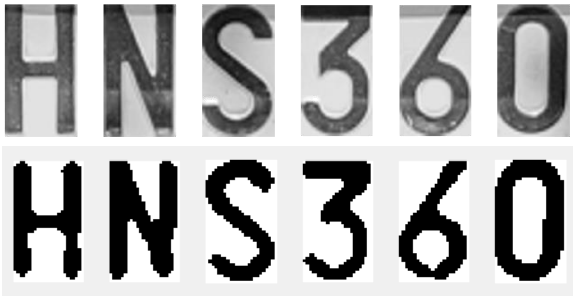

Figura 3.19 Caracteres segmentados
Caracteres en escala de grises (superior), caracteres binarizados (inferior).

3.4 TRANSFORMADA WAVELET DISCRETA

Para el desarrollo de esta etapa, se aplicó la DWT a un solo nivel de descomposición a los caracteres provenientes de la placa y a los caracteres de la base de datos utilizando las familias *wavelet haar*, *daubechies, biorthogonal, reverse biorthogonal* y *symlets*. El proceso de descomposición 2D se realizó mediante la función *wavedec2* del *'Wavelet Toolbox'* de Matlab.

Después de realizada la descomposición, se extraen los coeficientes de detalles y aproximaciones mediante las funciones '*detcoef2'* y '*appcoef2'*

respectivamente y se reconstruyen los caracteres utilizando únicamente los detalles horizontales y verticales, siendo estos suficientes para la correcta descripción de la forma de los caracteres de la placa como se muestra en la Figura 3.20.

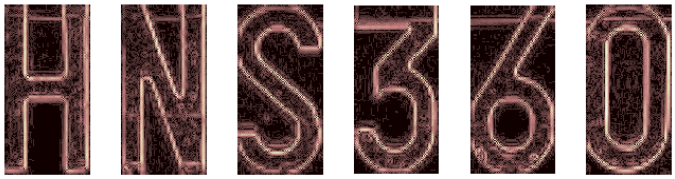

Figura 3.20 Reconstrucción de los caracteres de la placa mediante los detalles horizontales y verticales.

Lo anterior se hace también para los caracteres referencia de la base de datos con los cuales se realizará la comparación (ver figura 3.21).

Figura 3.21 Números y letras reconstruidas a partir de los detalles horizontales y verticales mediante la DWT.

3.5 CORRELACION DIGITAL DE IMÁGENES

Para establecer la medida de similitud entre los caracteres de referencia de la base de datos y los caracteres segmentados de la placa vehicular, se aplica un procesamiento de nivel alto. A continuación se define el funcionamiento de cada uno de los algoritmos.

3.5.1 Correlación Cruzada Normalizada (NCC)

Como método de correlación basada en área se empleó la NCC mediante la función '*corr2'*.

Este método calcula el coeficiente de correlación entre cada carácter segmentado de la placa vehicular y cada carácter de referencia, en donde un máximo coeficiente de correlación implica una buena similitud entre los caracteres comparados.

En la figura 3.22, se muestra un mapa de superficie que permite ver los coeficientes de correlación obtenidos al comparar las letras de la placa vehicular *HNS*-360 y las letras de la base de datos.

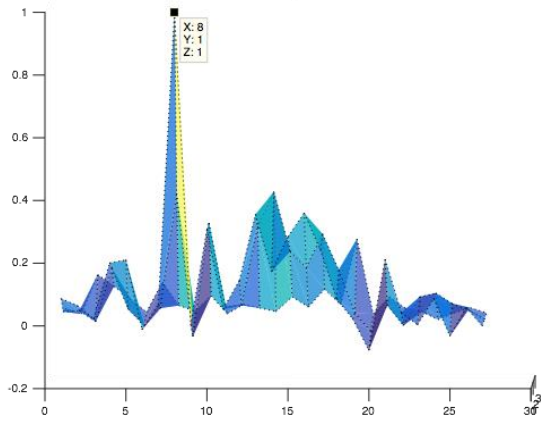

Figura 3.22 Coeficientes de correlación obtenidos para las letras HNS.

Los tres máximos coeficientes representan la correlación de las tres letras de la placa vehicular con las letras H, N y S de la base de datos. El máximo valor que se observa en la imagen fue obtenido porque la letra H de la base de datos fue tomada de esta placa vehicular, esto demuestra que cuando dos imágenes son exactamente la misma se obtiene el máximo coeficiente de correlación.

3.5.2 Histograma de Gradientes Orientados (HOG)

Con esta técnica se extraen las características *HOG* para cada uno de los caracteres de la placa vehicular en forma de *'vector de características'*, este vector es posteriormente comparado con el *vector de características HOG* de cada uno de los caracteres de referencia.

La extracción de las características *HOG* se realiza mediante la función '*extractHOGFeatures'* del *toolbox* '*Machine Vision*' de Matlab, definiendo un tamaño de celda de 8×8 dado que es el tamaño más común y aconsejado en el campo de visión de máquina.

Considerando que cada carácter fue de tamaño 64×32 pixeles, se agruparon en celdas de 8×8, resultando en una imagen de 8×4 celdas.

$$Celdas\ en\ la\ imagen\text{:}\ \frac{64}{8} \times \frac{32}{8} = 8 \times 4$$

Los demás parámetros se tomaron por defecto y están dados de la siguiente manera:

- Celdas x Bloque (*BlockSize*): 2×2
- Intervalos de Orientación del Histograma (*NumBins*): 9

Luego, se debe calcular el número de bloques tanto en x como en y:

$$No.Bloques = (No.Celdas - No.Celdas/Bloque) + 1$$

$$No.Bloques\ en\ x = (4 - 2) + 1 = 3$$
$$No.Bloques\ en\ y = (8 - 2) + 1 = 7$$

Por lo tanto, el número de bloques es:

$$7 \times 3 = 21\ bloques$$

Con esto se puede determinar el tamaño del vector de características *HOG* de las imágenes de la siguiente manera:

$$Tamaño\ del\ vector\ HOG = No.Bloques \times No.Celdas/bloque \times No.Intervalos$$
$$Tamaño\ del\ vector\ HOG = 21 \times 4 \times 9 = 756$$

Es decir, se tienen 756 características para cada caracter.

Para establecer la medida de similitud entre el vector de características de cada caracter de la placa y el vector de características de cada carácter de la base de datos, se utilizó el método de *Distancia Euclidiana*. La implementación de este método se realizó de la siguiente manera:

$$d2 = sum((hog1 - hog2).\text{^}2).\text{^}0.5$$

En donde $hog1$ representa el *vector de características* del caracter segmentado de la placa y $hog2$ representa el *vector de características* del caracter de la base de datos.

3.6 RECONOCIMIENTO DE LA PLACA VEHICULAR

A continuación, se describe el método de reconocimiento implementado para la etapa final. Se debe tener en cuenta que esta etapa es similar para ambos algoritmos: en el caso de la NCC se tienen en cuenta los coeficientes máximos de correlación y en el caso del descriptor HOG se tienen en cuanta los mínimos valores de distancia euclidiana.

1. Inicialmente se crea una matriz de 3×26 celdas para las letras y una matriz de 3×10 celdas para los números (ver figura 3.23), sobre las cuales se almacenan los valores de correlación (algoritmo 1) o los valores obtenidos del cálculo de la distancia euclidiana (algoritmo 2), según sea el caso.

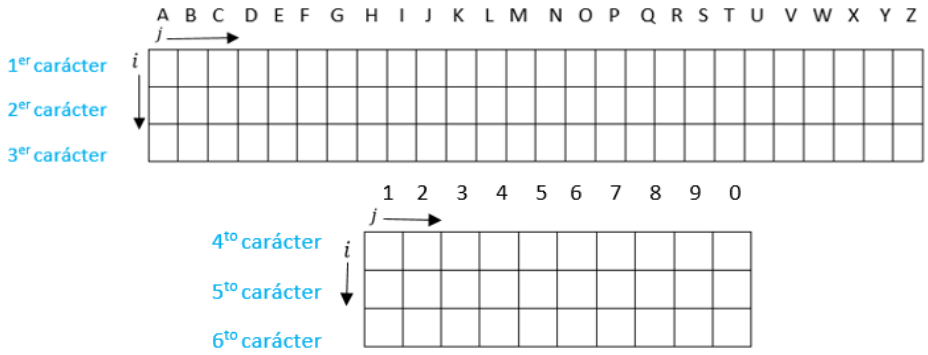

Figura 3.23 Matriz letra y número.

Tanto la *matriz letra* como la *matriz número* están compuestas de tres filas, en cada fila se almacenan los valores de correlación (para el primer algoritmo) o distancia euclidiana (para el segundo algoritmo) entre la primera letra (o número) segmentado de la placa y cada uno de los caracteres de la base de datos. Por ejemplo, la primera fila de la *matriz letra,* almacena el valor de correlación o distancia euclidiana entre la primera letra (primer carácter) de la placa y cada letra de la base de datos (26 caracteres), la segunda fila almacena el valor de correlación o distancia euclidiana entre la segunda letra (segundo carácter) de la placa y cada letra de la base de datos (26 caracteres) y la última fila almacena el valor de correlación o distancia euclidiana entre la tercera letra (tercer carácter) de la placa y cada letra de la base de datos (26 caracteres).

Este proceso es igual para la *matriz núm*ero, en donde cada fila de esta matriz almacena el valor de correlación o distancia euclidiana entre el primero, segundo y tercer número (cuarto, quinto y sexto carácter respectivamente) de la placa y cada número de la base de datos (10 caracteres).

2. Cada columna de la *matriz de letras* se relaciona con una letra en el orden del abecedario (de la A a la Z, excluyendo la Ñ), igualmente para

la *matriz de números*, en donde cada columna de esa matriz tendrá asignado un número como se muestra en la figura 3.23.

3. Una vez almacenados los valores de correlación y distancia euclidiana en las matrices *letra* y *número*, se procede a buscar el valor máximo (en el caso de correlación) y el valor mínimo (en el caso de la distancia euclidiana) en cada fila de las matrices.

4. Después de detectado el máximo de correlación o la mínima distancia euclidiana, se determina la posición (i, j) en donde se obtuvo alguno de estos valores.

5. Finalmente, a partir de la posición (i, j) y los pasos anteriormente descritos, se determina el carácter que dicha posición está representando.

Por último, para visualizar la placa vehicular extraída de la fotografía, se utilizó el comando '*horzcat'*.

CAPÍTULO 4

PRUEBAS Y ANÁLISIS DE DESEMPEÑO

Con ánimo de verificar el desempeño de los algoritmos desarrollados, se realizaron dos tipos de pruebas diferentes: la primera corresponde al reconocimiento de los caracteres de la placa vehicular, la segunda por su parte corresponde al tiempo que tarda cada algoritmo en ejecutarse (*tiempo de procesamiento*). Todo esto con el fin de determinar el mejor algoritmo a aplicar en el campo de reconocimiento de placas vehiculares.

Para esto, se utilizó una base de datos con 100 fotografías de automóviles seleccionadas a partir de las 438 fotografías obtenidas en la etapa de adquisición descrita en la sección 1.1 del Capítulo 3. Además, se garantizó la correcta segmentación de los caracteres de las placas para las 100 fotografías de prueba seleccionadas, para evitar que la etapa de segmentación influya negativamente en el reconocimiento final de la placa vehicular.

Los dos algoritmos desarrollados emplean la transformada wavelet discreta para realizar la descomposición de los caracteres y reconstruirlos a partir de sus detalles horizontales y verticales. Se seleccionaron cinco familias *wavelet*: *haar*, *daubechies2, symlets2*, *biorthogonal1.1 y reverse biorthogonal3.1*; las tres primeras se seleccionaron acorde con las recomendaciones de la literatura, mientras que las dos últimas fueron seleccionadas de acuerdo a un criterio visual, teniendo en cuenta que estas resaltan mejor los detalles de la imagen.

4.1 PRUEBAS DE RECONOCIMIENTO DE LA PLACA VEHICULAR

El objetivo principal de esta prueba consiste en determinar el porcentaje de placas reconocidas satisfactoriamente, con cada uno de los algoritmos y evaluar así el desempeño.

4.1.1 Método basado en área: Correlación Cruzada Normalizada (NCC)

Se inició realizando las pruebas con el algoritmo que utiliza la NCC empleando las familias *wavelet haar, daubechies2 (db2)* y *symlets2 (sym2)* a 1 nivel de descomposición, dado que estas son las más utilizadas para la extracción de características en imágenes [8]; además, se realizaron pruebas empleando las familias *wavelet biorthogonal1.1 (bior1.1)* y *reverse biorthogonal3.1 (rbio3.1)* a 1 nivel de descomposición. Adicionalmente, se aplicó la NCC para la comparación de patrones en escala de grises, es decir, sin descomposición alguna de la imagen mediante la transformada *wavelet*. Los resultados obtenidos para la base de datos de 100 imágenes se muestran en la figura 4.1

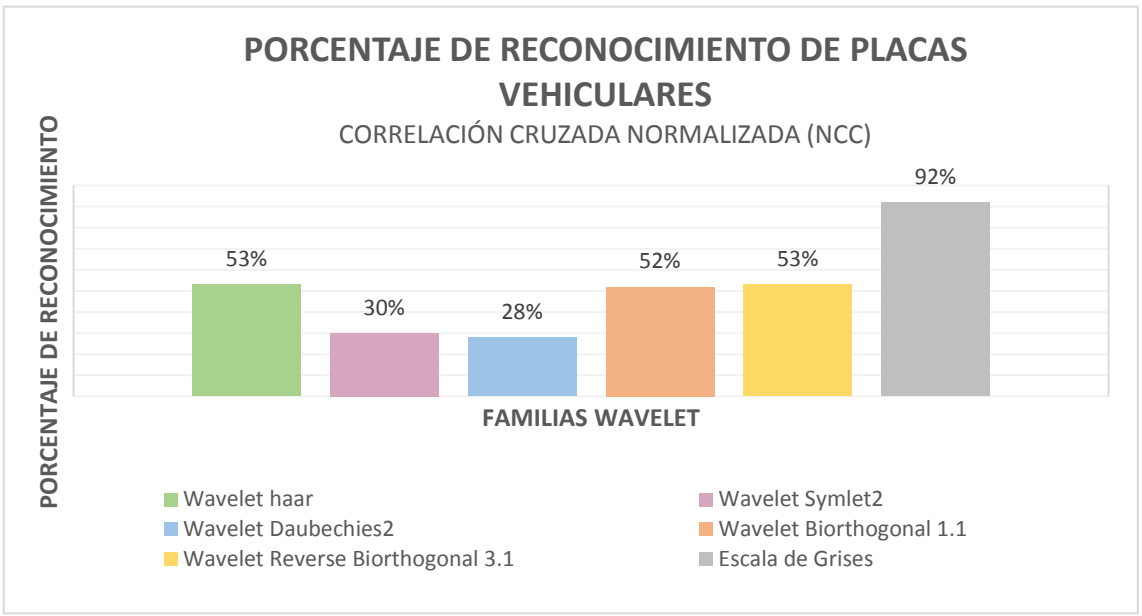

Figura 4.1 Número de placas reconocidas para cada tipo de familia wavelet y patrones en escala de grises.

En la figura 4.1 se observa un buen desempeño cuando se utiliza las *wavelets haar*, *biorthogonal* y *reverse biorthogonal*; sin embargo, el desempeño de este algoritmo cuando se utilizan patrones en escala de grises es notablemente superior.

A partir de los resultados obtenidos, y como primera medida, se debe analizar el rendimiento de las familias *wavelet* empleadas: teniendo en cuenta que la transformada *wavelet* permite, mediante el análisis multiresolución, extraer las componentes de detalles de una imagen en función de la forma de onda de la *wavelet* utilizada. Es posible afirmar que se obtuvo un mejor desempeño en el reconocimiento de los caracteres utilizando tanto la *wavelet haar* como la *wavelet bior1.1* y *rbio3.1,* debido a que la forma de onda de estas presenta cambios bruscos como lo hacen los cambios de intensidad en los bordes de cada caracter de la placa, resultando entonces mejores patrones con los cuales establecer la correlación, tal como se muestra en la figura 4.2.

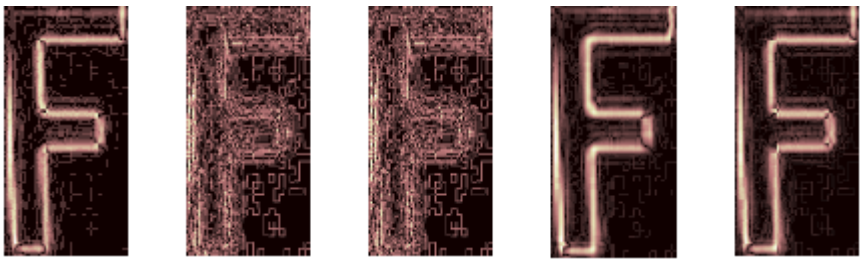

Figura 4.2 Letra F con diversas wavelet (de izquierda a derecha):
Haar, Sym2, Db2, Bior1.1, Rbio3.1.

Aunque se obtuvo unos buenos patrones con las wavelets haar, bior1.1 y rbio3.1, el resultado del algoritmo utilizando estas familias no fue tan alto como se esperaba, dado que la técnica de correlación cruzada es un método basado en área, lo que la hace sensible a los cambios en la intensidad de los píxeles de la imagen producidos por el ruido o por la iluminación en el momento de la adquisición de la misma, resultando en variaciones apreciables entre la imagen patrón y la imagen referencia.

Considerando que los bordes de la imagen son las zonas en las que hay mayores cambios de nivel de intensidad (cambios más bruscos), son estos los que brindan la información más relevante para efectuar el proceso de correlación y hacer posible la comparación entre dos imágenes. Sin embargo, al aplicar la descomposición *wavelet* a cada carácter de la placa y reconstruirla únicamente a partir de sus detalles horizontales y verticales, se están eliminando componentes importantes que también brindan información acerca de la imagen, tal como es el caso de las aproximaciones en las cuales se concentra la mayor cantidad de energía.

Además, si se comparan los patrones obtenidos para una misma letra aplicando diferentes tipos de familias *wavelet* (ver figura 4.2), se observa que los patrones cuyos bordes no están completamente definidos (como en el caso de las *wavelets db2* y *sym2*) son aquellos en donde se presenta un menor nivel de correlación, dado que la reconstrucción de los patrones mediante estas *wavelets* no representa adecuadamente la información contenida en los bordes de la imagen, lo que repercute negativamente en el reconocimiento de la placa vehicular.

Por otra parte, aunque se obtuvo buenos patrones con las *wavelets haar*, *bior1.1* y *rbio3.1*, la información contenida en los bordes de la imagen al aplicar estas transformaciones no es suficiente, lo que conlleva a un menor porcentaje de reconocimiento, si se compara con el obtenido al utilizar imágenes en escala de grises, en el cual, el desempeño es mucho mayor (92%), ya que se tienen bordes más definidos y con mucha más información para ser extraída y comparada.

4.1.2 Método basado en características: Histograma de Gradientes Orientados (HOG)

Para el algoritmo que utiliza el descriptor HOG y la distancia euclidiana, se realizó la prueba con las diferentes familias *wavelet* ya mencionadas; los resultados se presentan en la figura 4.3.

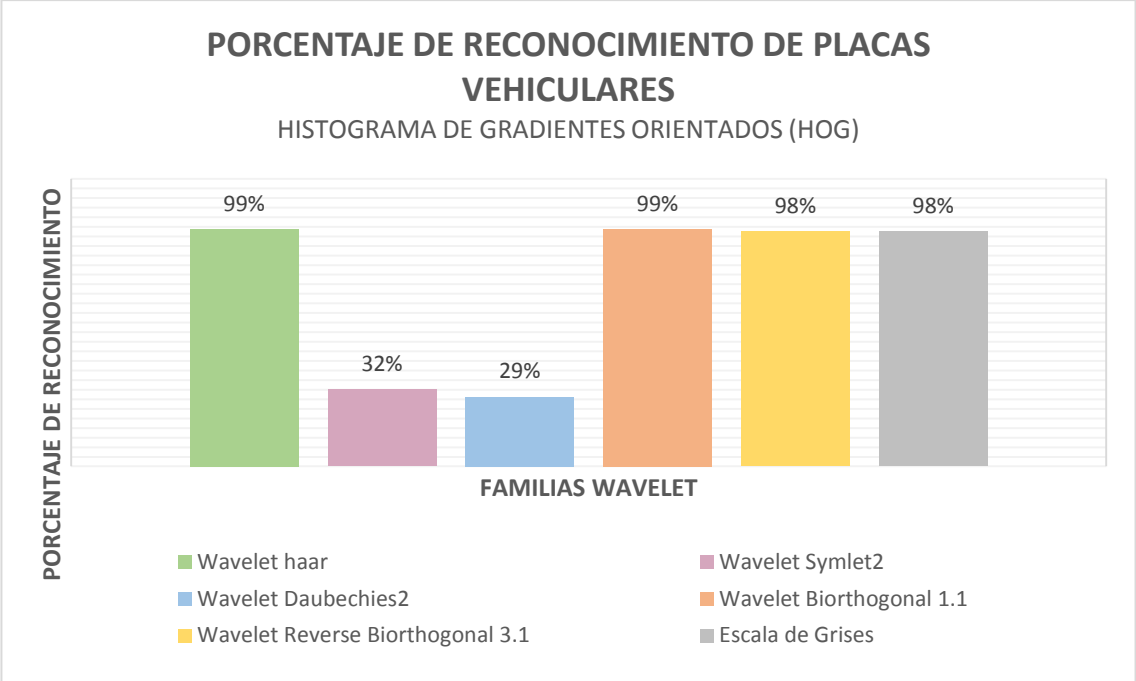

Figura 4.3 Porcentaje de reconocimiento para todas las wavelet empleadas y escala de grises.

Como se observa en la figura 4.3, las wavelet *sym2* y *db2* continúan teniendo un bajo desempeño, debido a que, estas no representan correctamente la forma del caracter. Sin embargo, el desempeño de las *wavelets haar*, *bior1.1* y *rbio3.1* mejoró notablemente, al igual que cuando se utilizan patrones en escala de grises, alcanzando porcentajes de desempeño cercanos al 100%.

El alto porcentaje de reconocimiento obtenido, es debido a que el histograma de gradientes orientados se basa en la extracción de las características de la imagen para crear un *vector HOG* a partir de ellas. Por lo tanto, si se parte del hecho que una imagen contiene un grupo de características específicas y únicas que hacen posible su identificación, dos imágenes semejantes tendrán entonces características semejantes, por lo cual, un *vector de características*

HOG de una imagen será similar a un *vector de características HOG* de su imagen semejante, de modo que, entre mayor semejanza exista entre dos vectores, mayor será la similitud entre las dos imágenes. Por consiguiente se determina que *HOG* es una técnica robusta a las variaciones en la iluminación, producto de la adquisición de las fotografías, lo que conlleva a que el porcentaje de reconocimiento sea mayor.

A continuación se presenta una comparación en el desempeño de los dos algoritmos implementados (ver figura 4.4):

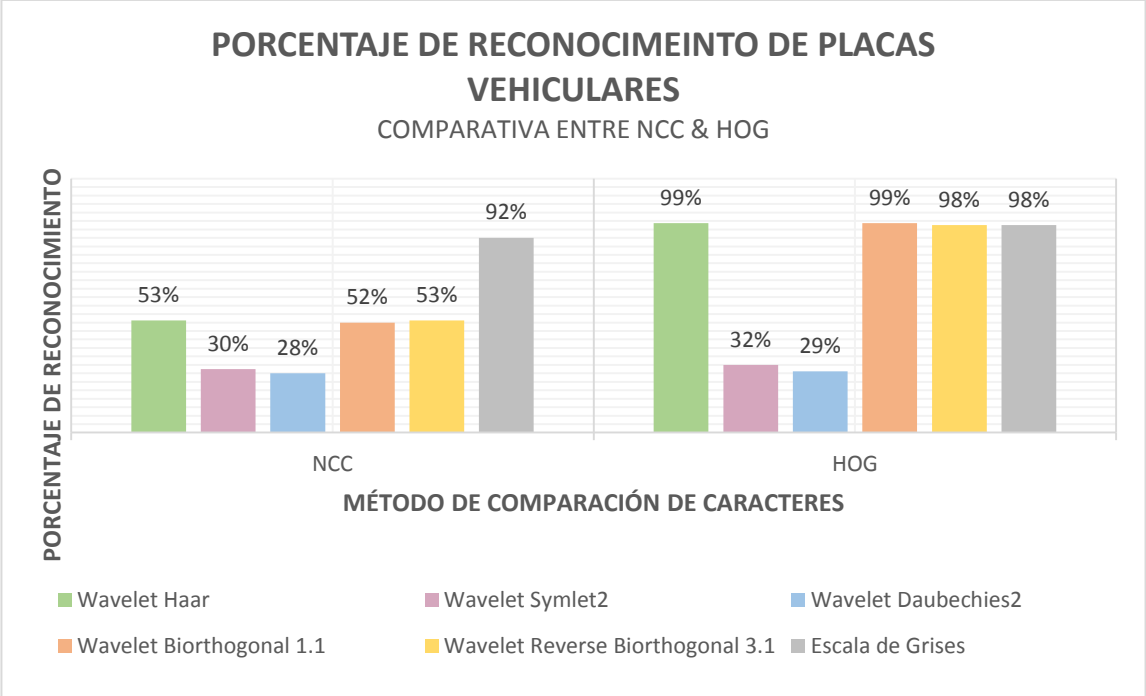

Figura 4.4 Comparación de desempeño en el reconocimiento de placas vehiculares de los algoritmos desarrollados.

En la figura 4.4 se observa que el porcentaje de reconocimiento obtenido con el algoritmo que emplea la NCC es mucho menor, comparado con el algoritmo que utiliza el *descriptor HOG.*

En la NCC, la comparación entre los caracteres de la placa y cada caracter de referencia se realiza teniendo en cuenta los valores de intensidad de los píxeles, por lo tanto, se ve afectada por cambios en la iluminación y presencia

de ruido. No obstante, vale la pena resaltar que cuando se utilizan patrones en escala de grises, el desempeño del algoritmo está muy por encima del desempeño obtenido cuando se emplea la transformada *wavelet,* dado que se está conservando la totalidad de la información de la imagen. En conclusión, la NCC presenta mejor desempeño cuando se realiza la comparación de imágenes con todas sus características (caracteres en escala de grises), por consiguiente, una imagen representada únicamente por sus bordes no es adecuada para realizar la comparación entre imágenes cuando se emplea esta técnica.

El *descriptor HOG*, por su parte, al ser una técnica que no tiene en cuenta los valores absolutos de intensidad, permite una mayor invarianza a brillos, sombras y demás cambios de iluminación, al mismo tiempo que permite cambios moderados de posición. Es por esto que su desempeño es similar cuando se utilizan patrones en escala de grises o patrones reconstruidos con la transformada *wavelet*. Además, es de destacar que el porcentaje de reconocimiento obtenido con este algoritmo cuando se emplean patrones a los que se les ha aplicado las *wavelet haar* y *bior1.1* es levemente mayor (1%), si se compara con el porcentaje de reconocimiento obtenido cuando se emplean patrones en escala de grises, puesto que la magnitud y la orientación de los gradientes en los patrones en escala de grises son más suaves que en un imagen representada únicamente por sus bordes, por lo tanto, una magnitud y una dirección de gradiente más fuerte, permite un mejor reconocimiento. Si bien, este porcentaje no es muy determinante bajo un contexto académico, puede llegar a serlo en aplicaciones en las que se requiera que el porcentaje de error admisible sea casi nulo.

En general, las familias *wavelet* con las que se obtuvo mejor desempeño en el reconocimiento de los caracteres de la placa vehicular son *haar, bior1.1* y *rbio3.1,* por tanto, son estas con las cuáles se realizó la prueba para el análisis de desempeño de los algoritmos tomando como factor el tiempo de procesamiento.

4.2 TIEMPO DE PROCESAMIENTO

Una forma de determinar el tiempo que tardan los algoritmos en realizar el reconocimiento de la placa vehicular consiste en el cálculo de las operaciones matemáticas que realizan los algoritmos, sin embargo esta prueba no fue realizada debido a la complejidad de la misma. En su lugar, se utilizó el comando *'profile viewer'* de Matlab para establecer el tiempo que tarda cada algoritmo en ejecutar las funciones principales, las cuales son: la *transformada wavelet discreta*, la *correlación cruzada normalizada* y el *descriptor HOG*.

Las pruebas fueron realizadas en un equipo DELL con las siguientes características:

- Sistema operativo de 64 bits.
- Procesador Intel Core i5 de 3GHz.
- Memoria RAM de 4GB.

Se inició realizando la prueba con el algoritmo que implementa la NCC, obteniéndose los datos que se muestran en la figura 4.5.

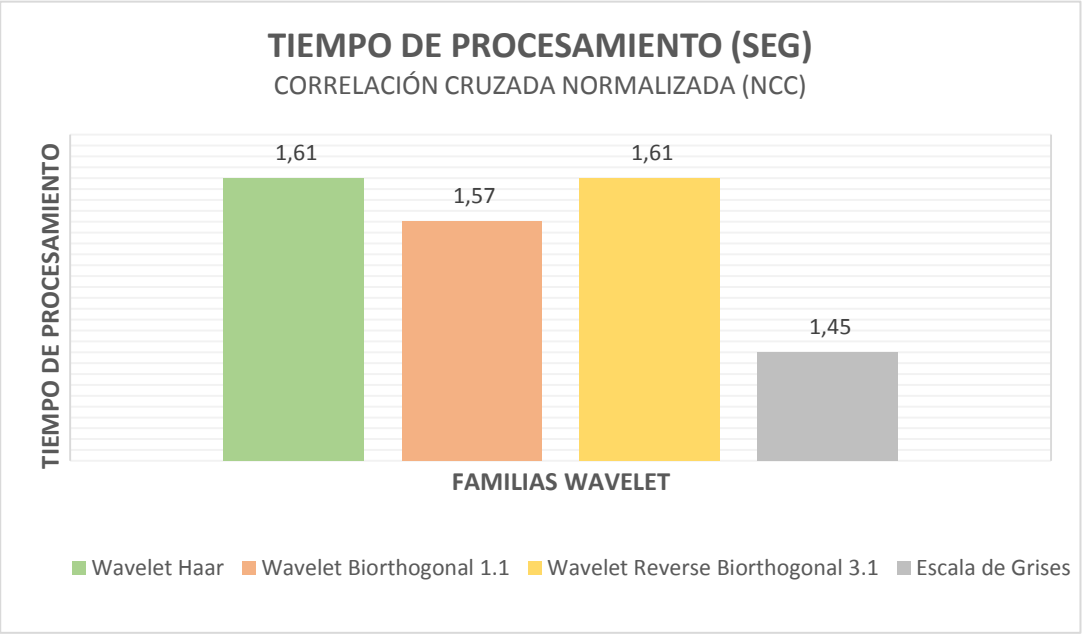

Figura 4.5 Tiempo de procesamiento del algoritmo que emplea el método basado en área (Correlación Cruzada Normalizada).

De la figura 4.5 se tiene que el tiempo total que tarda el algoritmo de reconocimiento de placas vehiculares utilizando la NCC es bastante bajo, además, es similar sin importar la familia *wavelet* empleada, sin embargo, es válido afirmar que el tiempo total de procesamiento cuando se utiliza la imagen en escala de grises es menor dado que no se realiza el proceso adicional de aplicar la DWT.

En este mismo orden de ideas se procede a determinar el tiempo total que tarda el algoritmo de reconocimiento de placas vehiculares utilizando el descriptor HOG. En la figura 4.6 se presentan los resultados obtenidos para esta prueba.

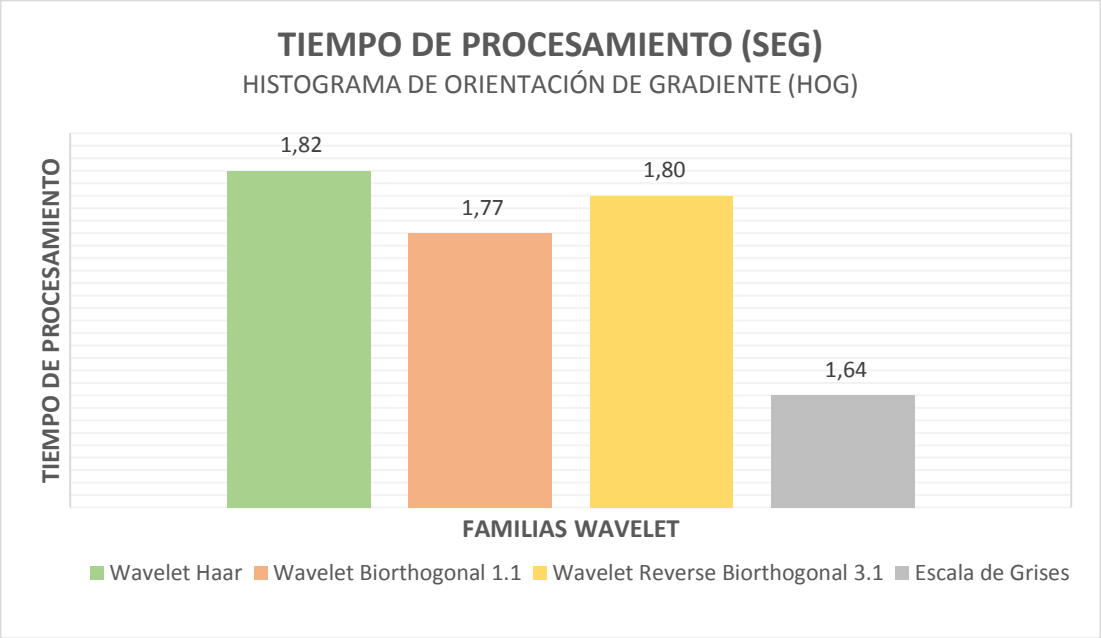

Figura 4.6 Tiempo de procesamiento del algoritmo que emplea el método basado en características (Histograma de Gradientes Orientados).

Se observa que el tiempo total de procesamiento que tarda el algoritmo que utiliza el descriptor *HOG* es mayor cuando se emplea las familias *wavelet* y menor cuando se emplean caracteres en escala de grises, análogamente a lo que sucede con el algoritmo que implementa la NCC, además se puede apreciar que los tiempos de procesamiento son relativamente bajos.

Por último, si se compara el tiempo total que tardan los dos algoritmos en realizar el reconocimiento de la placa vehicular (ver figura 4.7) se observa que el algoritmo que utiliza el *detector HOG* toma más tiempo en realizar dicho reconocimiento, esto debido a que requiere dividir la imagen en celdas y calcular un histograma de gradiente para cada una de ellas, por tanto, realizar este proceso aumenta el tiempo de procesamiento tal y como se evidencia en los resultados obtenidos.

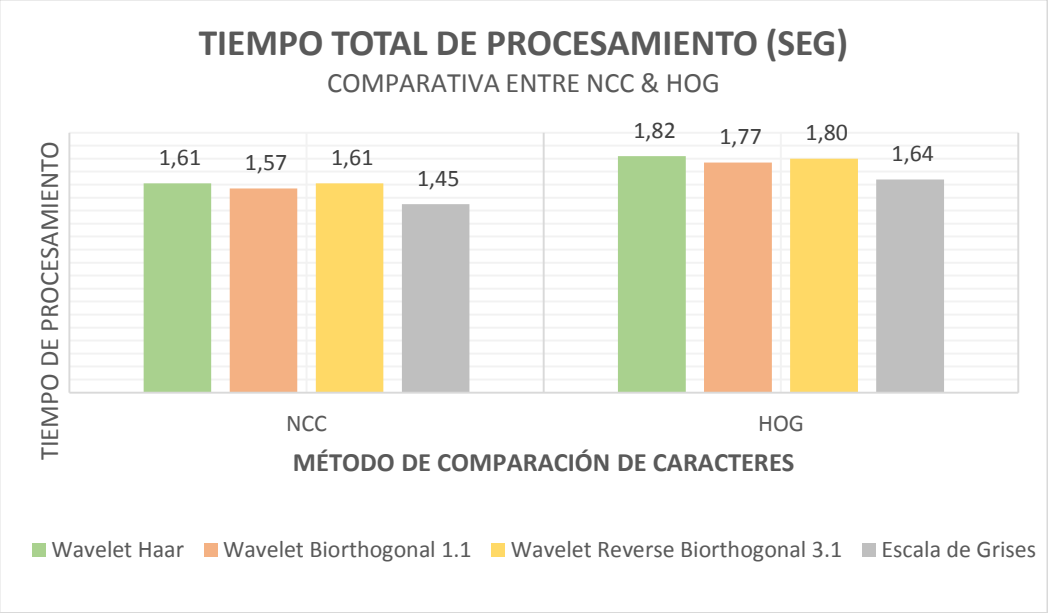

Figura 4.7 Comparación del tiempo de procesamiento de los algoritmos desarrollados.

En conclusión, al analizar los resultados de desempeño obtenidos para los dos algoritmos se observa que uno supera al otro en función del tipo de prueba realizada. Sin embargo, es imperativo deducir que el algoritmo más óptimo para el reconocimiento de placas vehiculares empleando técnicas de comparación o emparejamiento de patrones es aquel basado en características, es decir, aquel que implementa el *descriptor HOG*, ya que si bien, el algoritmo que utiliza la NCC presenta un mejor desempeño en cuanto a costo computacional, el porcentaje de reconocimiento es muy bajo usando la transformada *wavelet* (entre el 51% y 53%) a diferencia del que utiliza el descriptor *HOG* que presenta porcentajes de reconocimiento entre el 98% y 99% (aunque el tiempo de procesamiento es levemente mayor).

CAPÍTULO 5

CONCLUSIONES Y TRABAJOS FUTUROS

5.1 CONCLUSIONES

- La correlación cruzada normalizada es ampliamente utilizada para el reconocimiento de patrones dado que permite realizar la comparación de dos imágenes para establecer su similitud; sin embargo, esta presenta algunas dificultades con respecto al entorno de adquisición de las fotografías debido a que los cambios de luminosidad en el momento de adquisición afectan directamente los niveles de intensidad de los píxeles de la imagen, lo que influye negativamente en el correcto reconocimiento de las placas vehiculares.

- Dado que la correlación cruzada normalizada es una técnica de comparación de patrones basada en área, es indispensable la presencia de abundante información para garantizar la correcta comparación entre dos imágenes. Debido a lo anterior, esta técnica no es tan efectiva cuando se aplica sobre imágenes que resultan de la reconstrucción a partir de detalles horizontales y verticales de la *trasformada wavelet*, resultando mejor realizar la comparación de patrones directamente sobre las imágenes en escala de grises.

- El uso del *descriptor HOG* como método de comparación de los patrones reconstruidos a partir de los detalles horizontales y verticales de la imagen, supone un leve aumento del desempeño del algoritmo de reconocimiento de placas vehiculares en comparación al uso de patrones en escala de grises. Esto se debe a que el *descriptor HOG* aprovecha la información de las características más pequeñas y representativas presentes en los bordes del patrón. Sin embargo, el tiempo de procesamiento es mayor en

comparación a la correlación cruzada normalizada aplicada sobre patrones en escala de grises.

- En comparación con la técnica de correlación cruzada, el *descriptor HOG* presenta un mejor desempeño en el algoritmo de reconocimiento de placas vehiculares tanto con patrones reconstruidos a partir de la *transformada wavelet* (desempeño del 99%) como con patrones en escala de grises (desempeño del 98%). No obstante, el tiempo de procesamiento es levemente mayor al obtenido con la correlación cruzada normalizada.

- La transformada *wavelet discreta* aplicada a fotografías de placas vehiculares en mal estado resulta en la obtención de patrones de detalles horizontales y verticales cuyos bordes no se encuentran totalmente definidos y/o en patrones con cambios en la morfología del caracter, esto reduce el porcentaje de reconocimiento de la placa vehicular especialmente en el algoritmo de la correlación cruzada normalizada, dado que esta es susceptible a los cambios de intensidad de los píxeles en los bordes de la imagen. Este problema no se presenta en la comparación de patrones mediante el *descriptor HOG,* debido a que se hace mediante la extracción de un vector de características.

- Teniendo en cuenta que la correlación cruzada normalizada, y la técnica basada en el *descriptor HOG* son fáciles de implementar y presentan un bajo costo computacional, la elección entre una técnica u otra debe hacerse en función de la aplicación que se desea desarrollar y del margen de error permisible en dicha aplicación. Por tanto, a partir de los resultados obtenidos se puede afirmar que la técnica que mejor se adapta al reconocimiento de placas vehiculares en particular corresponde al *descriptor HOG* debido al alto desempeño que esta técnica presenta bien sea con patrones en escala de grises, o con patrones reconstruidos mediante la transformada *wavelet* discreta.

- Aunque las familias *wavelet symlet2* y *daubechies2* son recomendadas para la detección de características, estas no presentan un buen desempeño en la representación de los caracteres, dado que su forma de onda no se asemeja a los cambios bruscos producidos en los bordes de estos. En su lugar las familias *wavelet biorthogonal 1.1* y *reverse biorthogonal 3.1* presentan un buen desempeño, dado que sus transiciones se adaptan mejor a los cambios producidos en los bordes del caracter.

- Es de suma importancia que la técnica empleada en la etapa de localización de la placa dentro de la imagen sea adaptable a cualquier tipo de entorno de adquisición, dado que a partir de esta se procede a realizar el proceso de segmentación de los caracteres sobre los cuáles se aplica, bien sea, la correlación cruzada normalizada o el *descriptor HOG* para realizar la comparación y el reconocimiento de las placas vehiculares.

5.2 TRABAJOS FUTUROS

A partir de los resultados y conclusiones obtenidas mediante el desarrollo del trabajo de grado denominado ‘Reconocimiento de placas vehiculares utilizando la transformada wavelet discreta y la correlación digital de imágenes’, se plantean los siguientes trabajos futuros:

- Se propone implementar un sistema de reconocimiento en tiempo real que permita realizar el reconocimiento de placas vehiculares, incluso a partir de video.

- Dado que los algoritmos de reconocimiento desarrollados en este trabajo se centran en imágenes adquiridas en ambientes favorables, se espera que en líneas futuras se implementen técnicas de umbralización, corrección de rotación e iluminación, etc., que sean adaptables a las características particulares de cada fotografía sin importar el entorno de adquisición (sea diurno, nocturno, con mucha o poca luz).

- Se propone el desarrollo de algoritmos que permitan el reconocimiento de placas vehiculares, independientemente del tipo de vehículo o de las características de color de su placa.

- Se sugiere implementar este tipo de algoritmos en sistemas para la localización de automóviles dentro de un parqueadero a partir del reconocimiento de la placa vehicular mediante el dispositivo móvil de un usuario. Además, sistemas que permitan el ingreso automático a edificios a partir del reconocimiento de la placa del vehículo, y la implementación de sistemas que faciliten el control de la seguridad en establecimientos tanto públicos como privados.

REFERENCIAS

[1] Computer vision lab DRESDEN, “Computer Vision”. [En línea] Available: http://cvlab-dresden.de

[2] L. E. Sucar y G. Gomez, “Visión Computacional”, Inst. Nac. Astrofísica, Óptica y Electrónica, pp. 185, 2011.

[3] P. D. P. Nuñez, “Procesamiento Digital de Imágenes”, 2006.

[4] E. Halberg, “Transformada de Fourier”, no. 2, pp. 1–3, 2012.

[5] R. Medina y J. Bellera, “Bases del Procesamiento de Imágenes Médicas”, pp. 1–34, 2014.

[6] Grupo de Topología Computacional y Matemática Aplicada, “Tema 4: Segmentación de imágenes”, 2017.

[7] A. Carcedo y Franco, “Programa de segmentación de regiones en imágenes médicas en MATLAB, Capítulo 1. Teoría de procesamiento de imágenes”, pp. 7–29, 2004.

[8] Mathworks, MATLAB, “Choose a Wavelet” [En Línea] Available: https://www.mathworks.com/help/wavelet/gs/choose-a-wavelet.html

[9] F. Ramirez y R. Valdiviezo, “Reconocimiento de Patrones en Imágenes Fijas Usando Wavelets”. Trabajo de grado, Facultad de Ingeniería Electrónica y Telecomunicaciones, Universidad del Cauca, 2007.

[10] O. Pérez Ramírez, “Algoritmos de Comprensión de Imágenes sin Movimiento para Comunicaciones Móviles (3G)”. Tesis doctoral, Dpt. De Ingeniería Electrónica. Escuela de Ingeniería, Universidad de las Américas Puebla, 2004.

[11] J. Ramirez y J. Moreno. “Análisis del Desempeño de la Modulación Wavelet”. Trabajo de grado, Facultad de Ingeniería Electrónica y Telecomunicaciones, Universidad del Cauca, 2009.

[12] D. Gasca y L. Rojas, “Extracción de Características Descriptoras de Movimientos de la Mano a partir de Señales Electromiografías (EMG)

Aplicando Técnicas Wavelet". Trabajo de grado, Facultad de Ingeniería Electrónica y Telecomunicaciones, Universidad del Cauca, 2006.

[13] M. Laster, "Introducción a la Transformada Wavelet, Descomposición de Señales". Agencia Nacional de Promoción Científica y Tecnológica, Universidad Nacional del Centro de la Provincia de Buenas Aires, 2006.

[14] J. Martínez y R. de Castro, "Análisis de las Teorías de las Ondílicuas orientada a las Aplicaciones en la Ingeniería Eléctrica". Tesis doctoral, Dpt. De Ingeniería eléctrica, E.T.S.I. Industriales, 2002.

[15] Gómez-Luna, E.; Silva, D. y Aponte, G. "Selección de una wavelet madre para el análisis frecuencial de señales eléctricas transitorias usando WPD Ingeniare". *Revista Chilena de Ingeniería*, vol. 21, no. 2, agosto, 2013, pp. 262-270. Universidad de Tarapacá Arica, Chile.

[16] Ruiz Salazar, J, y Alexander Orejuela Caicedo, D. "Implementación de la transformada wavelet sobre un sistema embebido para el pre-procesamiento de señales unidimensionales no estacionarias". Universidad de San Buenaventura, Cali, Colombia, 2016.

[17] Á. Suárez Bravo y G. Pajares Martinsanz, "Análisis De Métodos De Procesamiento De Imágenes Estereoscópicas Forestales", 2009.

[18] C. Alberto, R. Martínez, y M. S. Física, "Conceptos Básicos del Procesamiento Digital de Imágenes Usando OrquideaJAI Calculadora Digital", pp. 1–61, 2006.

[19] J. Melorose, R. Perroy, y S. Careas, "Implementación De Un Correlador De Imágenes Usando La Transformada De Fourier Fraccional Mediante Un Procesador Digital De Señales", *Statew. Agric. L. Use Baseline 2015*, vol. 1, pp. 22–26, 2015.

[20] J. A. Ramírez, "Reconocimiento de imágenes", pp. 41–54, 2010.

[21] B. K. Desai, M. Pandya, y M. B. Potdar, "Comparison of Various Template Matching Techniques for Face Recognition", *Int. J. Eng. Res. Dev.*, vol. 8, no. 10, pp. 16–18, 2013.

[22] F. Zhao, Q. Huang, y W. Gao, "Image matching by normalized cross correlation", pp. 729–732, Beijing, China, 2006.

[23] N. N. Dawoud, B.B. Samir, y J. Janier, “Fast Template Matching Method Based on Optimized Metrics for Face Localization”, *Proc. Int. MultiConference Eng. Comput. Sci.*, vol. I, pp. 14–17, 2012.

[24] D. Taylor, “Journal of the Mechanical Behavior of Biomedical Materials”, *Journal of the Mechanical Behavior of Biomedical Materials*, vol. 14.

[25] M. F. Shinwari, N. Ahmed, H. Humayun, I. U. Haq, S. Haider, y A. U. Anam, “Classification Algorithm for Feature Extraction using Linear Discriminant Analysis and Cross-correlation on ECG Signals”, *Int. J. Advanved Sci. Technol.*, vol. 48, pp. 149–162, 2012.

[26] B. Zitová y J. Flusser, “Image registration methods: A survey”, *Image Vis. Comput.*, vol. 21, no. 11, pp. 977–1000, 2003.

[27] V. Fachbereich, F. Alhwarin, S. Referent, A. Gr, y D. Silber, “Fast and robust image feature matching methods for computer vision applications”, no. April, 2011.

[28] A. López Peña, (2017). “L4.2. HOG – Cálculo del gradiente”, *Universitat Autónoma de Barcelona* (En Línea)
Available: http://www.coursera.org/learn/deteccion-objetos/lecture/uAEKB/l4-4-hog-calculo-del-descriptor

APÉNDICE I

TÉCNICAS PARA LA LOCALIZACIÓN DE LA PLACA VEHICULAR

Dado que el reconocimiento de patrones parte de la correcta localización de la placa dentro de la fotografía, se aplicaron diferentes técnicas para realizar este proceso y seleccionar aquella que brinda mejores resultados. Estos algoritmos para la localización de la placa y la segmentación de los caracteres fueron probados utilizando una muestra de 438 fotografías de vehículos con el fin de garantizar la selección del algoritmo con mejor desempeño. A continuación se presentan las técnicas empleadas con las cuáles se obtuvieron los resultados más significativos.

Proyección del Histograma

La primera técnica implementada para realizar la localización de la placa dentro de la fotografía y con esto, la segmentación de los caracteres es conocida como 'Proyección del Histograma' la cual se basa en las proyecciones de los niveles de intensidad de la imagen sobre los ejes de coordenadas.

En el eje de coordenadas y, se proyecta la sumatoria de los niveles de intensidad de los píxeles contenidos en cada una de las filas de la imagen; y en el eje de coordenadas x, se proyecta la sumatoria de los niveles de intensidad de los píxeles contenidos en cada una de las columnas. Dada una imagen de tamaño $n \times m$, se emplean las funciones (1.1) y (1.2) para realizar su proyección horizontal y vertical respectivamente.

$$H(m) = \sum_{j=0}^{N-1} f(m,j), \qquad m = 0,1,\dots,M-1 \tag{1.1}$$

$$V(n) = \sum_{i=0}^{M-1} f(i,n), \qquad n = 0,1,\dots,N-1 \tag{1.2}$$

A partir de los valores máximos arrojados por estas funciones se pueden determinar subintervalos; si $[h1, h2]$ es un subintervalo horizontal y $[v1, v2]$ es un subintervalo vertical, entonces el objeto que se desea segmentar puede estar en el rectángulo $[h1, h2] \times [v1, v2]$ según se muestra en la figura I.

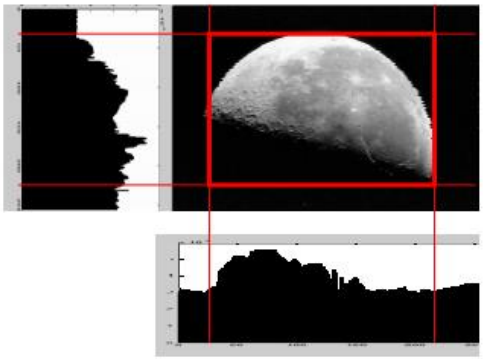

Figura I Proyección del Histograma.

Con esta técnica se lograron porcentajes de localización y segmentación del 66.67%, lo que se considera un porcentaje muy bajo para este tipo de aplicación. Una de las razones principales por las cuáles se obtiene este valor tan pequeño se debe a que esta técnica depende directamente de los niveles de intensidad de la imagen; por tanto, el factor de iluminación en el momento de la adquisición es determinante en el proceso de segmentación cuando se emplea la *proyección del histograma*.

Hibrido: Proyección del Histograma y Crecimiento de Regiones

La siguiente técnica con la cual se realizó la segmentación de la placa corresponde a un híbrido entre la proyección vertical del histograma y el crecimiento de regiones.

La proyección vertical del histograma se realizó con el fin de obtener una aproximación horizontal o un segmento horizontal de la imagen donde posiblemente se encuentre la región de interés; es decir, la placa vehicular. Seguido de esto, se aplicó la técnica conocida como crecimiento de regiones para obtener de forma un poco más precisa, la región que corresponde a la placa vehicular. Sin embargo, el algoritmo de localización desarrollado utilizando este tipo de segmentación obtuvo resultados mucho menores a los obtenidos con la técnica anterior (46.8%), motivo por el cual fue descartada.

Crecimiento de Regiones

El tercer método de localización desarrollado corresponde a la segmentación de las placas vehiculares implementando únicamente la técnica de crecimiento de regiones mediante la aplicación de operaciones morfológicas de dilatación, apertura y cierre, utilizando como elemento estructurante un cuadrado de 5 píxeles de lado; además, se utilizó un filtro *bottom-hat* con elemento estructurante rectangular de tamaño 20×60. Con esta técnica se obtuvo un 67.81% de placas localizadas y segmentadas correctamente; sin embargo, solo se logró superar la técnica de 'Proyección del Histograma' en un 1.14%.

Finalmente, se probó la técnica de crecimiento de regiones utilizando como elemento estructurante un cuadrado de 5 píxeles de lado para realizar las operaciones morfológicas anteriormente mencionadas, seguido de un filtro *bottom-hat* con un elemento estructurante en forma de disco. A la imagen obtenida de este proceso se le aplicó nuevamente operaciones morfológicas de cierre y apertura utilizando como elemento estructurante un rectángulo de 20×60 píxeles y un filtro bot*tom-hat* con elemento estructurante en forma de disco de 10 píxeles de lado.

Con esta última técnica, se obtuvieron los mejores porcentajes de reconocimiento (72.60%) con lo que se garantiza que un mayor número de placas vehiculares van a ser correctamente segmentadas para posteriormente

ser reconocidas mediante los algoritmos desarrollados como parte de este trabajo de grado.

A continuación, se muestra de manera gráfica una comparativa del porcentaje de reconocimiento obtenidos utilizando las técnicas anteriormente mencionadas (ver figura II).

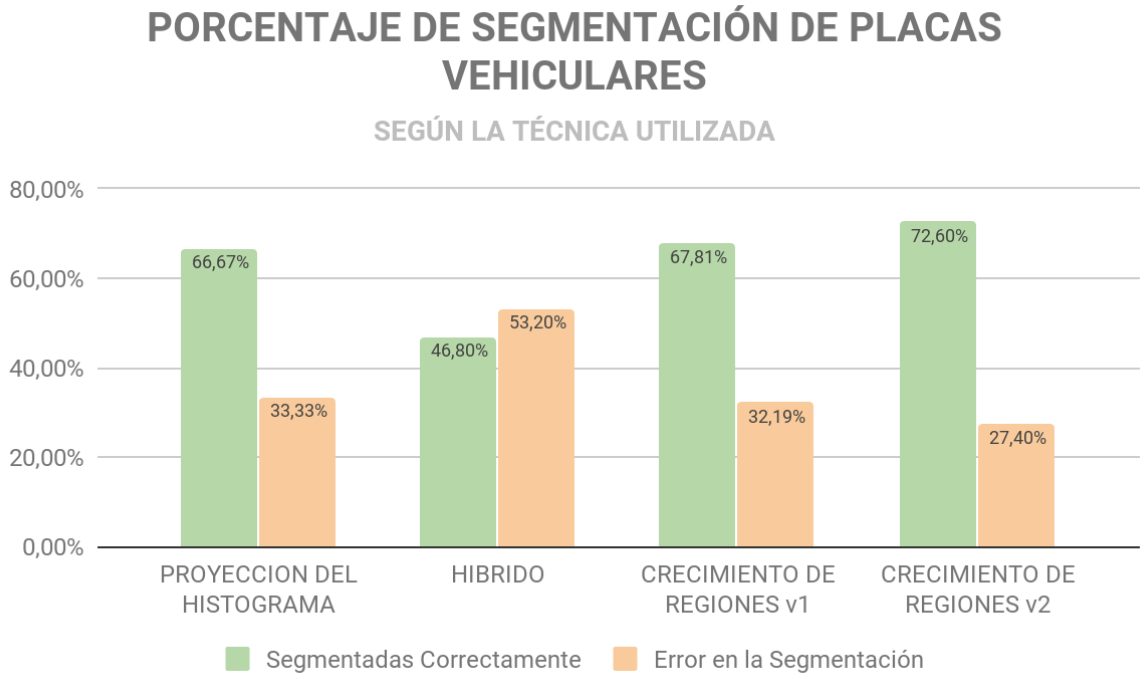

Figura II Comparación del desempeño de los algoritmos de localización.

Adicionalmente, se implementaron otras técnicas como correlación cruzada y el método *SURF*, etc. Para la localización de la placa vehicular; sin embargo, con estas no se obtuvo buenos resultados.

APÉNDICE II

GUI Y MANUAL DE USUARIO

Se desarrolló la interfaz gráfica de usuario (GUI, *Graphic User Interface*) utilizando la herramienta GUIDE de Matlab 2015a. A continuación se muestran los pasos a seguir para realizar el reconocimiento de una placa vehicular en esta interfaz.

Abrir la Interfaz

Inicialmente, se debe ejecutar el archivo *reconocimientodeplacas.fig* para abrir la interfaz de usuario que se presenta en la figura III.

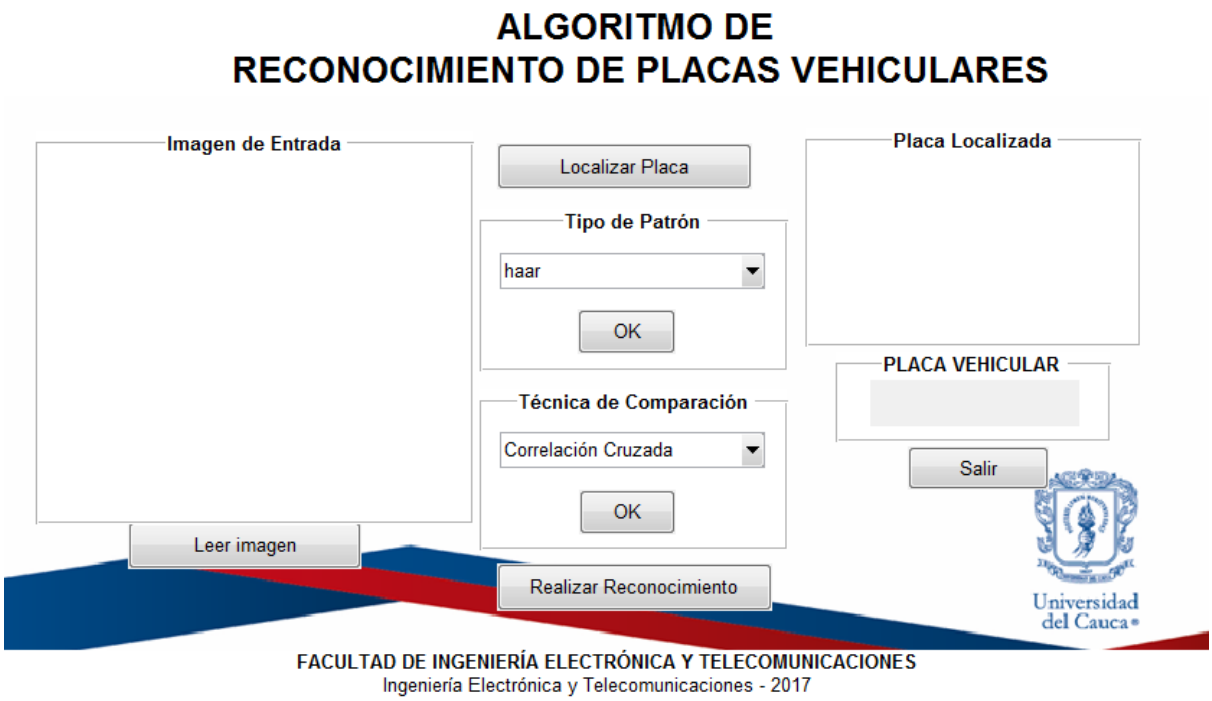

Figura III Interfaz de Usuario.

Leer una imagen

Para cargar la imagen con la placa vehicular que se desea reconocer se debe pulsar en el botón *'leer imagen'* tal y como se muestra en la figura IV. Las imágenes pueden ser importadas desde cualquier directorio, y los formatos soportados son *.bmp*, *.jpg*, *.tif* y *.png*

Figura IV Imagen de entrada

Localización de la placa

Posteriormente se debe realizar la localización de la placa dentro de la fotografía. Para esto, se debe presionar el botón *'localizar placa'*. Como se observa en la figura V, la imagen de la placa localizada deberá desplegarse en el campo '*placa localizada*' de la interfaz.

Figura V Placa Localizada

Selección del patrón y la técnica de Comparación

Una vez localizada la placa, se debe seleccionar el tipo de patrones con los cuales se va realizar la comparación de los caracteres; esto se hace en la sección de la interfaz titulada '*tipo de patrón*' en donde se presenta un menú desplegable (ver figura VI) para seleccionar entre las opciones de patrones con wavelet *haar*, *bior1.1*, *rbio3.1* o los patrones en escala de grises. Una vez seleccionado el tipo de patrón se debe presionar el botón *OK*.

Similar al paso anterior, se debe seleccionar la técnica con la cual se desea realizar la comparación de los patrones. Las técnicas de comparación implementadas en este trabajo de grado son: Correlación Cruzada e Histograma de Gradientes Orientados; su elección se realiza desde el menú desplegable que se encuentra en el campo de la interfaz denominado '*Tipo de técnica*' (ver figura VII). Una vez seleccionada se debe dar *click* en el botón *OK*.

Figura VI Selección del tipo de patrón

Figura VII Selección de la técnica de comparación

Finalmente, al presionar el botón '*realizar reconocimiento*', los caracteres de la placa vehicular serán mostrados en el campo de la interfaz denominado '*Placa vehicular*' tal y como se observa en la figura VIII.

Figura VIII Placa Reconocida

Si se desea realizar el reconocimiento de una nueva placa, se puede seleccionar *'Leer imagen'* para cargar una nueva fotografía y ejecutar los pasos descritos anteriormente. Sin embargo, si desea salir de la interfaz, basta con presionar el botón *'salir'* y confirmar esta acción.

Figura IX Salir de la interfaz

APÉNDICE III

CÓDIGO EN MATLAB

Reconocimientodeplacas.m

Código principal de los algoritmos de reconocimiento de placas vehiculares desarrollados en Matlab 2015a. Posteriormente se explicará en que consiste cada una de las funciones aquí empleadas y sus respectivos parámetros de entrada y salida.

```
%% ADQUISICIÓN %%
[img, map]=imread('imagen.jpg');

%% PREPROCESAMIENTO %%
[media, media2, grises] = preprocesamiento(img);

%% LOCALIZACION %%
[placa1, C22]=localizacion(media, media2, grises);

%% SEGMENTACION %%
[L1b, L2b, L3b, L4b, L5b, L6b]=segmentacion(placa1, C22);
```

%% TRANFORMADA WAVELET DISCRETA %%

La DWT es aplicada a los caracteres segmentados en la etapa anterior. Esta reconstrucción de caracteres se realiza a partir de los detalles horizontales y verticales utilizando las familias wavelet: haar, biorthogonal1.1 y reversebiorthogonal3.1 dado que con estas familias wavelet se obtuvo un alto desempeño de reconocimiento.

```
% TRANSFORMADA WAVELET DE LOS CARACTERES DE LA PLACA VEHICULAR:
```

De acuerdo a la familia wavelet seleccionada, se tienen las diversas funciones que aplican la DWT a los caracteres segmentados de la placa vehicular

```
[matricula, plac]=waveletHaar(L1b, L2b, L3b, L4b, L5b, L6b);
[matricula, plac]=waveletbio(L1b, L2b, L3b, L4b, L5b, L6b);
[matricula, plac]=waveletrbior(L1b, L2b, L3b, L4b, L5b, L6b);

%% PATRONES RECONSTRUIDOS MEDIANTE LA DWT %%
```

De manera similar, estas funciones permiten cargar los patrones de la base de datos de referencia reconstruidos mediante las familias wavelet haar, biorthogonal1.1 y reversebiorthogonal3.1

```
[abc, numeros]=patroneshaar();
[abc, numeros]=patronesbio();
```

```
[abc, numeros]=patronesrbio();
```

%% TÉCNICAS DE COMPARACIÓN DE PATRONES %%

En esta sección del código se puede realizar el reconocimiento de patrones empleando bien sea la Correlación Cruzada Normalizada o el descriptor HOG; sin embargo los parámetros de entrada de estas funciones son similares independientemente del caso

```
%% CORRELACION CRUZADA NORMALIZADA - NCC %%
[matrizl, matrizn]=miCorrelacion(plac, abc, numeros);

%% HISTOGRAMA DE ORIENTACIÓN DE GRADIETE - HOG%%
[matrizl, matrizn]=miHOG(plac, abc, numeros);


%% RECONOCIMIENTO %%
[OCR]=reconocimiento(matrizl, matrizn);
```

1. ADQUISICÓN

Las fotografías obtenidas del proceso de adquisición se almacenaron en una base de datos de pruebas, y fueron utilizadas en el código haciendo uso de la función `imread` de Matlab.

En este caso, imagen.jpg consiste en la imagen del vehículo cuya placa se desea reconocer. Esta será almacenada en una variable llamada *'imagen'*

```
[imagen, map]=imread('imagen.jpg');
```

2. LOCALIZACIÓN DE LA PLACA VEHICULAR

Para la etapa de localización se inició realizando procesamiento de nivel bajo en donde se aplicaron técnicas de transformación de la imagen.

PREPROCESAMIENTO	
Parámetros de entrada	**Parámetros de salida**
imagen: Imagen de entrada	**grises:** Imagen resultante al convertir la imagen a una en escala de grises.

	media: Imagen resultante después de aplicar el filtro de media al canal rojo de la imagen. **media2:** Imagen resultante después de aplicar el filtro de media a la imagen en escala de grises.
Función implementada: function [media, media2, grises] = preprocesamiento(imagen)	

preprocesamiento.m

```
%% PROCESAMIENTO DE BAJO NIVEL %%

function [media, media2, grises] = preprocesamiento(imagen)

imagen=imresize(imagen,[720 860]);
grises=rgb2gray(imagen);
im_r =(imagen(:,:,1)) ;
sharp = imsharpen(im_r,'threshold', 0.5);
sharp2 = imsharpen(grises,'threshold', 0.5);
media = medfilt2(sharp,[5 5]);
media2 = medfilt2(sharp2,[5 5]);
end
```

localización	
Parámetros de entrada	**Parámetros de salida**
media, media2	**placa1:** Placa extraída y recortada **C2:** Corte definido para la placa
Función implementada: [placa1, C2]=114osición114ión(media, media2, grises)	

localizacion.m

```
%% PROCESAMIENTO DE NIVEL MEDIO PARA LA LOCALIZACIÓN Y SEGMENTACIÓN DE
LA PLACA VEHICULAR %%


function [placa1, C22]=localizacion(media, media2, grises)


%% OPERACIONES MORFOLÓGICAS %%

se =strel('square',5);
st=strel('disk',5);

erosion = imerode(media,se);
```

```
apertura = imdilate(erosion,se);
H = apertura-erosion;
img1=imbothat(H,st);
%% UMBRALIZACIÓN %%
umb=img1>80;


 %% OPERACIONES MORFOLÓGICAS PARA ELIMINAR REGIONES PEQUEÑAS %%

st1=strel('rectangle', [20,60]);
st2=strel('disk',10);

im2=imclose(umb,st1);
im3=imopen(im2,st1);
im4=imdilate(im3,st2);

%% ETIQUETADO DE REGIONES %%

L=bwlabel(im4);
stats=regionprops(L,'all');
Idx=find([stats.Area]>(7000));
im4=ismember(L,Idx);
L=bwlabel(im4);

stats = regionprops(L,'all');
E=stats(1).BoundingBox;
X=E.*[[1] [0] [0] [0]]; X=max(X);
Y=E.*[[0] [1] [0] [0]]; Y=max(Y);
W=E.*[[0] [0] [1] [0]]; W=max(W)
H=E.*[[0] [0] [0] [1]]; H=max(H);
Corte=[X Y W H];

PLACA1=imcrop(media,Corte);
PLACA2=imcrop(grises,Corte);



    %% MEJORA DEL CORTE DE LA PLACA DADO QUE AÚN SE TIENEN BORDES NO
                    DESEADOS EN LA IMAGEN RESULTANTE %%


%% UMBRALIZACIÓN %%
umbral=120;
placa=PLACA1>umbral;


%% ETIQUETA REGIONES DE LA PLACA SEGMENTADA %%
L=bwlabel(placa);
stats=regionprops(L,'all');
placadx=find([stats.Area]>(4000));
placa=ismember(L, placadx);
E2=stats(1).BoundingBox;

L=bwlabel(placa);
stats=regionprops(L,'all');
E2=stats(1).BoundingBox;
X2=E2.*[1 0 0 0]; X2=max(X2);
Y2=E2.*[0 1 0 0]; Y2=max(Y2);
```

```
W2=E2.*[0 0 1 0]; W2=max(W2);
H2=E2.*[0 0 0 1]; H2=max(H2);
Corte2=[X2 Y2 W2 H2];

C1=imcrop(PLACA1,Corte2);
C2=imcrop(PLACA2,Corte2);
Wx=round(W2*0.94);
Hx=round(H2*0.756);
Corte3=[4 12 Wx Hx];

%Cortes sobre la imagen de la placa original
C1=imcrop(C1,Corte3);
C2=imcrop(C2,Corte3);


%% OPERACIONES MORFOLOGICAS %%

st3=strel('disk', 12);
C3=imbothat(C1,st3);

%% UMBRALIZACIÓN PARA ELIMINAR LAS REGIONES PEQUEÑAS %%

umbral2=100;
C5=C3>umbral2;

                        %% PLACA RECORTADA %%

L=bwlabel(C4);
stats=regionprops(L,'all');
placadx=find([stats.Area]>((W2*H2)*0.015));
placa1=ismember(L,placadx);

end
```

3. SEGMENTACIÓN DE LOS CARACTERES DE LA PLACA VEHICULAR

SEGMENTACIÓN	
Parámetros de entrada	**Parámetros de salida**
placa1, C2	**L1b,L2b, L3b, L4b, L5b, L6b:** Caracteres de la placa recortados individualmente
Función implementada: function [L1b,L2b, L3b, L4b, L5b, L6b] = segmentacion(placa1, C2)	

segmentacion.m

```
function [L1b, L2b, L3b, L4b, L5b, L6b]=segmentacion(placa1, C22)
```

%% SEGMENTACION DE LOS CARACTERES %%

```
L=bwlabel(placa1);
stats=regionprops(L,'all');

%%% PRIMER CARACTER %%%%
E3=stats(1).BoundingBox;
X3=E3.*[[1] [0] [0] [0]]; X3=max(X3);
Y3=E3.*[[0] [1] [0] [0]]; Y3=max(Y3);
W3=E3.*[[0] [0] [1] [0]]; W3=max(W3);
H3=E3.*[[0] [0] [0] [1]]; H3=max(H3);
CorteC=[X3 Y3 W3 H3];

%Recorte primer caracter:
L1=imcrop(C2,CorteC);
L1b=imresize(L1,[512 256]);

%%% SEGUNDO CARACTER %%%%
E3=stats(2).BoundingBox;        %Tama?o 117osici 2
X3=E3.*[[1] [0] [0] [0]]; X3=max(X3);
Y3=E3.*[[0] [1] [0] [0]]; Y3=max(Y3);
W3=E3.*[[0] [0] [1] [0]]; W3=max(W3);
H3=E3.*[[0] [0] [0] [1]]; H3=max(H3);
CorteC=[X3 Y3 W3 H3];

%Recorte segundo caracter:
L2=imcrop(C2,CorteC);
L2b=imresize(L2,[512 256]);

%%% TERCER CARACTER %%%%
E3=stats(3).BoundingBox;
X3=E3.*[[1] [0] [0] [0]]; X3=max(X3);
Y3=E3.*[[0] [1] [0] [0]]; Y3=max(Y3);
W3=E3.*[[0] [0] [1] [0]]; W3=max(W3);
H3=E3.*[[0] [0] [0] [1]]; H3=max(H3);
CorteC=[X3 Y3 W3 H3];

%Recorte tercer caracter
L3=imcrop(C2,CorteC);
L3b=imresize(L3,[512 256]);


%%% CUARTO CARACTER %%%%
E3=stats(4).BoundingBox;
X3=E3.*[[1] [0] [0] [0]]; X3=max(X3);
Y3=E3.*[[0] [1] [0] [0]]; Y3=max(Y3);
W3=E3.*[[0] [0] [1] [0]]; W3=max(W3);
H3=E3.*[[0] [0] [0] [1]]; H3=max(H3);
CorteC=[X3 Y3 W3 H3];

%Recorte Cuarto caracter
L4=imcrop(C2,CorteC);
L4b=imresize(L4,[512 256]);


%%% QUINTO CARACTER %%%%
E3=stats(5).BoundingBox;
X3=E3.*[[1] [0] [0] [0]]; X3=max(X3);
```

```
Y3=E3.*[[0] [1] [0] [0]]; Y3=max(Y3);
W3=E3.*[[0] [0] [1] [0]]; W3=max(W3);
H3=E3.*[[0] [0] [0] [1]]; H3=max(H3);
CorteC=[X3 Y3 W3 H3];

%Recorte Quinto Caracter
L5=imcrop(C2,CorteC);
L5b=imresize(L5,[512 256]);


%%% SEXTO CARACTER %%%%
E3=stats(6).BoundingBox;
X3=E3.*[[1] [0] [0] [0]]; X3=max(X3);
Y3=E3.*[[0] [1] [0] [0]]; Y3=max(Y3);
W3=E3.*[[0] [0] [1] [0]]; W3=max(W3);
H3=E3.*[[0] [0] [0] [1]]; H3=max(H3);
CorteC=[X3 Y3 W3 H3];

%Recorte Sexto Caracter
L6=imcrop(C2,CorteC);
L6b=imresize(L6,[512 256]); %42 24
end
```

4. TRANSFORMADA WAVELET DISCRETA

Este código permite aplicar la DWT a los caracteres segmentados de la placa vehicular para reconstruirlos a partir de sus detalles horizontales y verticales. Esto fue realizado para las *wavelet*: *haar, biorthogonal1.1 y reversebiorthogonal3.1* dado que fueron estas las que presentaron los mejores desempeños.

Por facilidad de presentación del algoritmo, solo se mostrará la función implementada para la *wavelet haar*.

TRANSFORMADA WAVELET(HAAR)	
Parámetros de entrada	**Parámetros de salida**
L1b, L2b, L3b, L4b, L5b, L6b	**plac, plachog:** vector 1x6 que contiene los caracteres reconstruidos a partir de la transformada wavelet discreta.
Función implementada: function [matricula, plac, matriculahog, plachog]= wavelethaar(tipow, L1b, L2b, L3b, L4b, L5b, L6b)	

waveletthaar.m

```
%% ANÁLISIS Y SINTESIS DE LOS CARACTERES DE LA PLACA VEHICULAR
                    MEDIANTE LA DWT %%

%% FAMILIA WAVELET %%
 En caso de querer aplicar otra familia wavelet, es necesario cambiar
    el parámetro familia de la siguiente manera: bior1.1 (wavelet
     biorthogonal 1.1), rbio3.1(wavelet reverse biorthogonal3.1)

familia='haar';
nivel=1;

%% PRIMER CARACTER %%
%% DESCOMPOSICIÓN WAVELET %%
[c,s] = wavedec2(L1b,nivel,familia);
[Hc,Vc,Dc] = detcoef2('all',c,s,nivel);
Ac = appcoef2(c,s,familia,nivel);
%% RECONSTRUCCIÓN %%
X1 = idwt2([], Hc, Vc, [],familia);
VL1b = wcodemat(X1,255,'mat',1);

%% SEGUNDO CARACTER %%
%% DESCOMPOSICIÓN WAVELET %%
[c,s] = wavedec2(L2b,nivel,familia);
[Hc,Vc,Dc] = detcoef2('all',c,s,nivel);
Ac = appcoef2(c,s,familia,nivel);
%% RECONSTRUCCIÓN %%
X2 = idwt2([], Hc, Vc, [],familia);
VL2b = wcodemat(X2,255,'mat',1);

%% TERCER CARACTER %%
%% DESCOMPOSICIÓN WAVELET %%
[c,s] = wavedec2(L3b,nivel,familia);
[Hc,Vc,Dc] = detcoef2('all',c,s,nivel);
Ac = appcoef2(c,s,familia,nivel);
%% RECONSTRUCCIÓN %%
X3 = idwt2([], Hc, Vc, [],familia);
VL3b = wcodemat(X3,255,'mat',1);

%% CUARTO CARACTER %%
%% DESCOMPOSICIÓN WAVELET %%
[c,s] = wavedec2(L4b,nivel,familia);
[Hc,Vc,Dc] = detcoef2('all',c,s,nivel);
Ac = appcoef2(c,s,familia,nivel);
%% RECONSTRUCCIÓN %%
X4 = idwt2([], Hc, Vc, [],familia);
VL4b = wcodemat(X4,255,'mat',1);

%% QUINTO CARACTER %%
%% DESCOMPOSICIÓN WAVELET %%
[c,s] = wavedec2(L5b,nivel,familia);
[Hc,Vc,Dc] = detcoef2('all',c,s,nivel);
Ac = appcoef2(c,s,familia,nivel);
%% RECONSTRUCCIÓN %%
X5 = idwt2([], Hc, Vc, [],familia);
VL5b = wcodemat(X5,255,'mat',1);
```

```
%% SEXTO CARACTER %%
%% DESCOMPOSICIÓN WAVELET %%
[c,s] = wavedec2(L6b,nivel,familia);
[Hc,Vc,Dc] = detcoef2('all',c,s,nivel);
Ac = appcoef2(c,s,familia,nivel);
%% RECONSTRUCCIÓN %%
X6 = idwt2([], Hc, Vc, [],familia);
VL6b = wcodemat(X6,255,'mat',1);

newmap=pink(255);


   %% VECTORES QUE ALMACENAN LOS CARACTERES RECONSTRUIDOS MEDIANTE LA
                        TRANSFORMADA WAVELET %%


%% CORRELACIÓN
matricula=[[VL1b] [VL2b] [VL3b] [VL4b] [VL5b] [VL6b]];
plac=mat2cell(matricula,[512],[256 256 256 256 256 256]);


  %% DADO QUE CUANDO SE UTILIZA EL DESCRIPTOR HOG SE EMPLEAN CARACTERES
     DE TAMAÑO 64X32, ES NECESARIO REDIMENSIOANRLOS Y ALMACENARLOS EN
                        DIFERENTES VECTORES %%

% Redimensionamiento de los caracteres para el algoritmo que usa HOG
HL1b =imresize(VL1b,[64 32]);
HL2b =imresize(VL2b,[64 32]);
HL3b =imresize(VL3b,[64 32]);
HL4b =imresize(VL4b,[64 32]);
HL5b =imresize(VL5b,[64 32]);
HL6b =imresize(VL6b,[64 32]);

%% HOG
matriculahog=[[HL1b] [HL2b] [HL3b] [HL4b] [HL5b] [HL6b]];
plachog=mat2cell(matriculahog,[64],[32 32 32 32 32 32]);

end
```

5. CORRELACIÓN DIGITAL DE IMÁGENES

Para realizar el proceso de correlación, es necesario cargar los caracteres de referencia de la base de datos que han sido previamente reconstruidos a partir de detalles horizontales y verticales mediante la aplicación de la DWT.

A manera de ejemplo, se muestra este proceso para los caracteres de referencia reconstruidos mediante la *wavelet haar*.

PATRONES WAVELET HAAR	
Parámetros de entrada	**Parámetros de salida**
-	**abc, abchog:** Vector de tamaño 1x26 donde se guardan los patrones wavelet de las letras de la base de datos de referencia **numeros, numeroshog:** Vector de tamaño 1x10 donde se guardan los patrones wavelet de los números de la base de datos de referencia
Función implementada: function [abc, abchog, numeros, numeroshog]=patroneshaar()	

patroneshaar.m

```
function [abc, abchog, abecedario, abecedariohog, numero, numerohog,
numeros, numeroshog]=patroneshaar()
```

```
%% CARGA LOS CARACTERES DE REFERENCIA DE LA BASE DE DATOS,
RECONSTRUIDOS MEDIANTE LA WAVELET HAAR %%

%% NUMEROS DE LA BASE DE DATOS DE REFERENCIA%%
uno=imread('caracteres/haar/1.bmp');
dos=imread('caracteres/haar/2.bmp');
tres=imread('caracteres/haar/3.bmp');
cuatro=imread('caracteres/haar/4.bmp');
cinco=imread('caracteres/haar/5.bmp');
seis=imread('caracteres/haar/6.bmp');
siete=imread('caracteres/haar/7.bmp');
ocho=imread('caracteres/haar/8.bmp');
nueve=imread('caracteres/haar/9.bmp');
cero=imread('caracteres/haar/0.bmp');

%% MATRIZ DE NUMEROS %%
numero=[[uno] [dos] [tres] [cuatro] [cinco] [seis] [siete] [ocho]
[nueve] [cero]];
numeros=mat2cell(numero,[512],[256 256 256 256 256 256 256 256 256
256]);

%% LETRAS DE LA BASE DE DATOS DE REFERENCIA %%
aa=imread('caracteres/haar/A.bmp');
bb=imread('caracteres/haar/B.bmp');
cc=imread('caracteres/haar/C.bmp');
dd=imread('caracteres/haar/D.bmp');
ee=imread('caracteres/haar/E.bmp');
ff=imread('caracteres/haar/F.bmp');
gg=imread('caracteres/haar/G.bmp');
hh=imread('caracteres/haar/H.bmp');
ii=imread('caracteres/haar/I.bmp');
jj=imread('caracteres/haar/J.bmp');
kk=imread('caracteres/haar/K.bmp');
```

```
ll=imread('caracteres/haar/L.bmp');
mm=imread('caracteres/haar/M.bmp');
nn=imread('caracteres/haar/N.bmp');
oo=imread('caracteres/haar/O.bmp');
pp=imread('caracteres/haar/P.bmp');
qq=imread('caracteres/haar/Q.bmp');
rr=imread('caracteres/haar/R.bmp');
ss=imread('caracteres/haar/S.bmp');
tt=imread('caracteres/haar/T.bmp');
uu=imread('caracteres/haar/U.bmp');
vv=imread('caracteres/haar/V.bmp');
ww=imread('caracteres/haar/W.bmp');
xx=imread('caracteres/haar/X.bmp');
yy=imread('caracteres/haar/Y.bmp');
zz=imread('caracteres/haar/Z.bmp');
```

%% MATRIZ DE LETRAS %%

```
abecedario=[[aa] [bb] [cc] [dd] [ee] [ff] [gg] [hh] [ii] [jj] [kk]
[ll] [mm] [nn] [oo] [pp] [qq] [rr] [ss] [tt] [uu] [vv] [ww] [xx] [yy]
[zz]];
abc=mat2cell(abecedario,[512],[256 256 256 256 256 256 256 256 256 256
256 256 256 256 256 256 256 256 256 256 256 256 256 256 256 256]);
```

%% DADO QUE CUANDO SE UTILIZA EL DESCRIPTOR HOG SE EMPLEAN CARACTERES DE TAMAÑO 64X32, ES NECESARIO REDIMENSIOANRLOS Y ALMACENARLOS EN DIFERENTES VECTORES %%

%% REDIEMNSION DE LOS NUMEROS DE LA BASE DE DATOS DE REFERENCIA %%

```
one =imresize(uno,[64 32]);
two =imresize(dos,[64 32]);
three =imresize(tres,[64 32]);
four =imresize(cuatro,[64 32]);
five =imresize(cinco,[64 32]);
six =imresize(seis,[64 32]);
seven =imresize(siete,[64 32]);
eight =imresize(ocho,[64 32]);
nine =imresize(nueve,[64 32]);
zero =imresize(cero,[64 32]);
```

%% MATRIZ DE NUMEROS REDIMENSIONADOS %%

```
numerohog=[[one] [two] [three] [four] [five] [six] [seven] [eight]
[nine] [zero]];
numeroshog=mat2cell(numerohog,[64],[32 32 32 32 32 32 32 32 32 32]);
```

%% REDIMENSION DE LAS LETRAS DE LA BASE DE DATOS DE REFERENCIA %%

```
aaa =imresize(aa,[64 32]);
bbb =imresize(bb,[64 32]);
ccc =imresize(cc,[64 32]);
ddd =imresize(dd,[64 32]);
eee =imresize(ee,[64 32]);
fff =imresize(ff,[64 32]);
ggg =imresize(gg,[64 32]);
hhh =imresize(hh,[64 32]);
iii =imresize(ii,[64 32]);
```

```
jjj =imresize(jj,[64 32]);
kkk =imresize(kk,[64 32]);
lll =imresize(ll,[64 32]);
mmm =imresize(mm,[64 32]);
nnn =imresize(nn,[64 32]);
ooo =imresize(oo,[64 32]);
ppp =imresize(pp,[64 32]);
qqq =imresize(qq,[64 32]);
rrr =imresize(rr,[64 32]);
sss =imresize(ss,[64 32]);
ttt =imresize(tt,[64 32]);
uuu =imresize(uu,[64 32]);
vvv =imresize(vv,[64 32]);
www =imresize(ww,[64 32]);
xxx =imresize(xx,[64 32]);
yyy =imresize(yy,[64 32]);
zzz =imresize(zz,[64 32]);


%% MATRIZ DE LETRAS REDIMENSIONADAS %%
abecedariohog=[[aaa] [bbb] [ccc] [ddd] [eee] [fff] [ggg] [hhh] [iii]
[jjj] [kkk] [lll] [mmm] [nnn] [ooo] [ppp] [qqq] [rrr] [sss] [ttt]
[uuu] [vvv] [www] [xxx] [yyy] [zzz]];
abchog=mat2cell(abecedariohog,[64],[32 32 32 32 32 32 32 32 32 32 32
32 32 32 32 32 32 32 32 32 32 32 32 32 32 32]);

end
```

A. CORRELACIÓN CRUZADA NORMALIZADA

CORRELACIÓN CRUZADA NORMALIZADA	
Parámetros de entrada	**Parámetros de salida**
plac, abc, numeros	**matrizl:** Matriz de tamaño 3x26 en donde se almacenan los valores de correlación entre cada letra de la placa y los 26 patrones de caracteres. **Matrizn:** Matriz de tamaño 3x10 en donde se almacenan los valores de correlación entre cada número de la placa y los 10 números patrón.
Función implementada: function[matrizl, matrizn]=micorr(plac, abc, numeros)	

micorr.m

```
function[matrizl, matrizn]=micorr(plac, abc, numeros)

% Se crean dos matrices para almacenar los valores de
correlación de los caracteres de la matrícula con los caracteres
de la base de datos de cada una de las letras y números

letra=zeros(3,26);
num=zeros(3,10);

%La matriz de letras se compone de 3 filas y 26 columnas. La
Fila 1 almacena todos los valores de correlación de la primera
letra con cada uno de los cacateres de la BDD, en fila 2 se
almacenan los valores de correlación de la segunda letra, y en
la fila tres, los de la tercera letra.

Fila=1;
ind=1;
```

%% SE EVALUAN LAS LETRAS %%

```
while fila < 4
posicion=1;

%Para cada letra de la placa….
for posicion=1:3
    plc=plac{1,posicion};
    pos=1;
    labc=0;

%Hace la correlacion
    while pos<27
        labc=abc{1,pos};

         %hace la corelacion de una letra de la matriz abc
        co=corr2(labc,plc);

        %con la letra de la placa, y almacena el valor de la
        correlaci?n en la matriz letra
        letra(fila,pos)=co;
        pos=pos+1;
    end

    %va a la otra letra de la matricula
    fila=fila+1;
    posicion=posicion+1;
end
end

for ind = 1:3
    % Encuentra el máximo de correlación de cada fila
```

```
    maxs=max(letra,[],2);
    % Encuentra la posición de los máximos de correlación
    [posx posy]=find(letra==maxs(ind,1));

   %Se obtiene la posici?n de la columna para cada uno de los
   tres valores m?ximos
     letras(ind)=posy;
    ind=ind+1;
end

fila=1;
ind=1;
```

%% SE EVALUAN LOS NUMEROS %%

```
while fila < 4
posicion=4;

%% Para cada número….
for posicion=4:6
    plc=plac{1,posicion};
    pos=1;
    nnumeros=0;

    % Hace la correlación….
    while pos<11
        nnumeros=posicion{1,pos};
        co=corr2(nnumeros,plc);
        num(fila,pos)=co;
        pos=pos+1;
    end
fila=fila+1;
posicion=posicion+1;
end
end

ind=1;

for ind = 1:3
    maxs=max(num,[],2);
    [posx posy]=find(num==maxs(ind,1));
    nums(ind)=posy;
    ind=ind+1;
end

matrizl='a';
ltr=1;
lt=1;
```

%% ASIGNO EL VALOR DE LA POSICIÓN A CADA LETRA DE LA MATRIZ %%

```
while ltr < 4
    while lt < 4
        if letras(ltr) == 1
            matrizl(lt)='A';

        elseif letras(ltr) == 2
            matrizl(lt)='B';

        elseif letras(ltr) == 3
            matrizl(lt)='C';

        elseif letras(ltr) == 4
            matrizl(lt)='D';

        elseif letras(ltr) == 5
            matrizl(lt)='E';

        elseif letras(ltr) == 6
            matrizl(lt)='F';

        elseif letras(ltr) == 7
            matrizl(lt)='G';

        elseif letras(ltr) == 8
            matrizl(lt)='H';

        elseif letras(ltr) == 9
            matrizl(lt)='I';

        elseif letras(ltr) == 10
            matrizl(lt)='J';

        elseif letras(ltr)  ==  11
            matrizl(lt)='K';

        elseif letras(ltr)  == 12
            matrizl(lt)='L';

        elseif letras(ltr) == 13
            matrizl(lt)='M';

        elseif letras(ltr) == 14
            matrizl(lt)='N';

        elseif letras(ltr) == 15
            matrizl(lt)='O';

        elseif letras(ltr) == 16
            matrizl(lt)='P';

        elseif letras(ltr) == 17
```

```
            matrizl(lt)='Q';

        elseif letras(ltr) == 18
            matrizl(lt)='R';

        elseif letras(ltr) == 19
            matrizl(lt)='S';

        elseif letras(ltr) == 20
            matrizl(lt)='T';

        elseif letras(ltr) == 21
            matrizl(lt)='U';

        elseif letras(ltr) == 22
            matrizl(lt)='V'

        elseif letras(ltr) == 23
            matrizl(lt)='W';

        elseif letras(ltr) == 24
            matrizl(lt)='X';

        elseif letras(ltr) == 25
            matrizl(lt)='Y';

        elseif letras(ltr) == 26
            matrizl(lt)='Z';

        else
            matrizl(lt)='Error';
        end

        lt=lt+1;
        ltr=ltr+1;
    end
end

nmro=1;
matrizn='1';
nm=1;


    %% SE INDICA EL NÚMERO QUE CORRESPONDE A CADA POSICIÓN %%
while nmro < 4
    while nm < 4

        if nums(nmro)== 1
            matrizn(nm)='1';

        elseif nums(nmro) == 2
            matrizn(nm)='2';
```

```
        elseif nums(nmro) == 3
            matrizn(nm)='3';

        elseif nums(nmro) == 4
            matrizn(nm)='4';

        elseif nums(nmro) == 5
            matrizn(nm)='5';

        elseif nums(nmro) == 6
            matrizn(nm)='6';

        elseif nums(nmro) == 7
            matrizn(nm)='7';

        elseif nums(nmro) == 8
            matrizn(nm)='8';

        elseif nums(nmro) == 9
            matrizn(nm)='9';

        elseif nums(nmro) == 10
            matrizn(nm)='0';

        else
            disp('Error')
        end

        nm=nm+1;
        nmro=nmro+1;
    end
end
```

B. HISTOGRAMA DE GRADIENTES ORIENTADOS

HOG	
Parámetros de entrada	**Parámetros de salida**
plachog, abchog, numeroshog	**matrizl:** Matriz de tamaño 3x26 en donde se almacenan las características HOG entre cada letra de la placa y los 26 patrones de caracteres. **matrizn:** Matriz de tamaño 3x10 en donde se almacenan las características HOG entre cada número de la placa y los 10 números patrón.

Función implementada:
function [matrizl, matrizn]=miHOG(plachog, abchog, numeroshog)

miHOG.m

```
% Se crean dos matrices para almacenar los valores de distancia
euclidiana calculados entre los vectores de características
extraídos de los caracteres de la placa vehicular y los vectores
de los caracteres de la base de datos de cada una de las letras
y números:

letra=zeros(3,26);
num=zeros(3,10);

% La matriz de letras se compone de 3 filas y 24 columnas.
% La Fila 1 almacena todos los valores de distancia euclidiana
de la primera letra con cada uno de los caracteres de la BDD, en
fila 2 se almacenan los valores de distancia euclidiana de la
segunda letra, y en la fila tres, los de la tercera letra.
Fila=1;
ind=1;

while fila < 4
posicion=1;

%Para cada letra de la placa….
for posicion=1:3
    plc=plachog{1,posicion};

   %Extrae características HOG
   [hog1, vis1] = extractHOGFeatures(plc,'CellSize',[8 8]);

    pos=1;
    labc=0;
    while pos<27
       labc=abchog{1,pos};
       [hog2, vis2] = extractHOGFeatures(labc,'CellSize',[8 8]);

       % Calcula la distancia euclidiana
       d1=sum((hog1 - hog2).^2).^0.5;

       % Almacena el valor de distancia euclidiana en la matriz
       letra
        letra(fila,pos)=d1;
         pos=pos+1;
    end

  % Va a la otra letra de la matricula…
    fila=fila+1;
    posicion=posicion+1;
end
```

```
end

for ind = 1:3

% Encuentra el mínimo de distancia euclidiana de cada fila
    minimo=min(letra,[],2);

% Encuentra la posición de los minimos de distancia euclidiana
    [posx posy]=find(letra==minimo(ind,1));

%Se obtiene la posición de la columna para cada uno de los tres
valores mínimos
    letras(ind)=posy;
    ind=ind+1;
end

fila=1;
ind=1;


                    %% SE EVALUAN LOS NUMEROS %%
while fila < 4
posicion=4;

% Para cada número….
for posicion=4:6
    plc=plachog{1,posicion};
    [hog3, vis3] = extractHOGFeatures(plc,'CellSize',[8 8]);

    pos=1;
    nnumeros=0;

    while pos<11
        nnumeros=numeroshog{1,pos};
% Extrae las características HOG
   [hog4, vis4] = extractHOGFeatures(nnumeros,'CellSize',[8 8]);

         % Calcula la distancia euclidiana
        d2=sum((hog3 - hog4).^2).^0.5;
        num(fila,pos)=d2;
        pos=pos+1;
    end

fila=fila+1;
posicion=posicion+1;
end
end

ind=1;

for ind = 1:3
    minimo2=min(num,[],2);
    [posx posy]=find(num==minimo2(ind,1));
```

```
    nums(ind)=posy;
    ind=ind+1;
end

matrizl='a';
ltr=1;
lt=1;
```

%% SE ASIGNA EL VALOR DE LA POSICION A CADA LETRA DE LA MATRIZ

```
while ltr < 4
    while lt < 4
        if letras(ltr) == 1
            matrizl(lt)='A';

        elseif letras(ltr) == 2
            matrizl(lt)='B';

        elseif letras(ltr) == 3
            matrizl(lt)='C';

        elseif letras(ltr) == 4
            matrizl(lt)='D';

        elseif letras(ltr) == 5
            matrizl(lt)='E';

        elseif letras(ltr) == 6
            matrizl(lt)='F';

        elseif letras(ltr) == 7
            matrizl(lt)='G';

        elseif letras(ltr) == 8
            matrizl(lt)='H';

        elseif letras(ltr) == 9
            matrizl(lt)='I';

        elseif letras(ltr) == 10
            matrizl(lt)='J';

        elseif letras(ltr)  ==  11
            matrizl(lt)='K';

        elseif letras(ltr)  == 12
            matrizl(lt)='L';

        elseif letras(ltr) == 13
            matrizl(lt)='M';

        elseif letras(ltr) == 14
```

```
            matrizl(lt)='N';

        elseif letras(ltr) == 15
            matrizl(lt)='O';

        elseif letras(ltr) == 16
            matrizl(lt)='P';

        elseif letras(ltr) == 17
            matrizl(lt)='Q';

        elseif letras(ltr) == 18
            matrizl(lt)='R';

        elseif letras(ltr) == 19
            matrizl(lt)='S';

        elseif letras(ltr) == 20
            matrizl(lt)='T';

        elseif letras(ltr) == 21
            matrizl(lt)='U';

        elseif letras(ltr) == 22
            matrizl(lt)='V'

        elseif letras(ltr) == 23
            matrizl(lt)='W';

        elseif letras(ltr) == 24
            matrizl(lt)='X';

        elseif letras(ltr) == 25
            matrizl(lt)='Y';
        elseif letras(ltr) == 26
            matrizl(lt)='Z';

        else
            matrizl(lt)='Error';
        end

        lt=lt+1;
        ltr=ltr+1;
    end
end


nmro=1;
matrizn='1';
nm=1;
```

% A CADA POSICION, SE ASIGNA EL NÚMERO QUE EL CORRESPONDE %%

```
while nmro < 4
    while nm < 4

        if nums(nmro)== 1
            matrizn(nm)='1';

        elseif nums(nmro) == 2
            matrizn(nm)='2';

        elseif nums(nmro) == 3
            matrizn(nm)='3';

        elseif nums(nmro) == 4
            matrizn(nm)='4';

        elseif nums(nmro) == 5
            matrizn(nm)='5';

        elseif nums(nmro) == 6
            matrizn(nm)='6';

        elseif nums(nmro) == 7
            matrizn(nm)='7';

        elseif nums(nmro) == 8
            matrizn(nm)='8';

        elseif nums(nmro) == 9
            matrizn(nm)='9';

        elseif nums(nmro) == 10
            matrizn(nm)='0';
        else
            disp('Error')
        end

        nm=nm+1;
        nmro=nmro+1;
    end
end
end
```

6. RECONOCIMIENTO DE LA PLACA VEHICULAR

Finalmente, el reconocimiento de la placa vehicular se realiza utilizando el comando 'horzcat' en Matlab.

Horzcat Concatena arrays horizontalmente **sintaxis:** C = horzcat(A1,…,AN)	
Parámetros de entrada	**Parámetros de salida**
matrizl, espacio, matrizn	**placavehicular:** Placa resultante del proceso de reconocimiento

```
espacio='-';

placavehicular=horzcat(matrizl,espacio,matrizn);
fprintf('La placa del vehiculo es: ')
```