UNIVERSIDAD TECNOLÓGICA DE LA MIXTECA

RECONOCIMIENTO DE OBJETOS INMERSOS EN IMÁGENES ESTÁTICAS MEDIANTE EL ALGORITMO HOG Y RNA-MLP

T E S I S

QUE PARA OBTENER EL GRADO DE

MAESTRO EN ELECTRÓNICA Y COMPUTACIÓN

PRESENTA

ING. AYAX ALDEMAR GARCÍA LÓPEZ

DIRECTOR: DR. ENRIQUE GUZMÁN RAMÍREZ

HUAJUAPAN DE LEÓN, OAXACA. AGOSTO, 2015

.

Resumen

La visión artificial o por computadora es una disciplina que, mediante la capacidad de percepción visual, busca la construcción de descripciones explícitas y significativas de los objetos físicos a partir de imágenes. Esta capacidad permite extraer y analizar información *espectral, espacial y temporal* de los distintos objetos contenidos en una imagen.

Gran variedad de tareas pueden ser realizadas por un sistema de visión artificial debido a esta capacidad; una de gran importancia es el reconocimiento de objetos, la cual se refiere a la identificación de un objeto con base en descriptores asociados con el objeto. Los procesos de extracción de características y clasificación, son de relevante importancia para que el reconocimiento de objetos cumpla su objetivo. Además, para obtener un buen desempeño en esta tarea es necesario contar con descriptores robustos y un clasificador de buena calidad.

En este sentido, el algoritmo de extracción de características propuesto por Navneet Dalal y Bill Triggs [1], denominado histogramas de gradientes orientados (*Histogram of Oriented Gradients*, HOG), ha demostrado que en representaciones normalizadas del objeto, sus descriptores ofrecen información discriminativa de los objetos presentes en una imagen, siendo además robusta gracias a su invariancia ante cambios en la iluminación, en el fondo o en la posición del objeto.

Por otro lado, es bien conocida la característica de las redes neuronales artificiales (RNA o ANN del inglés *Artificial Neural Network*) tipo perceptrón multicapa (MLP, *Multi-Layer Perceptron*) para solventar el problema que se presenta cuando los pocos datos disponibles durante el entrenamiento generalmente no son suficientes para cubrir la variabilidad en apariencia, lo que representa una buena opción para el proceso de clasificación.

Buscando ofrecer una alternativa competitiva con las existentes actualmente, y considerando la importancia que tiene la tarea de reconocimiento de objetos en diversas áreas de la ciencia y su gran potencial de aplicaciones, el presente trabajo de tesis presenta un sistema de reconocimiento de objetos que utiliza en el proceso de extracción de características al algoritmo HOG y a una RNA tipo MLP en el proceso de clasificación.

Abstract

Computer vision is a discipline that, through the ability of visual perception, search the construction of explicit and significant descriptions of physical objects from images. This ability allows to extract and analyse spectral, spatial and temporal information of the different object in an image.

Variety of task can be performed by a computer vision system due this ability; object recognition is one of the most important, which refers to the identification of an object based descriptors associated with the object. The process of feature extraction and classification, are of outstanding importance for the object recognition task in achieving its objective. Also, for a good performance in this task robust descriptors and a classifier of good quality are necessary.

At this sense, the feature extraction algorithm proposed by Navneet Dalal and Bill Triggs [1], called histogram of oriented gradients (HOG), demonstrated that under standard representations of objects, descriptors provide discriminative information from the objects in an image, and is also robust thanks to its invariance to changes in lighting, background or object position.

On the other hand, is highly known the characteristic of the artificial neural networks (ANN) multilayer perceptron (MLP) to solve the problem that occurs when the data available in training generally are not sufficient to cover the variability in appearance, which is a good option for the classification process.

Seeking to offer a competitive alternative to existing today, and considering the importance of the object recognition task in the different sciences areas and their potential in applications, the present work of thesis shows an object recognition system integrated by a HOG algorithm in the characteristics extraction process and ANN type MLP in the classification process.

Agradecimientos

Mi más profundo y sincero agradecimiento a todas aquellas personas que con su ayuda han colaborado en la realización del presente trabajo, en especial al Dr. Enrique Guzmán Ramírez, director de esta investigación, por la orientación, el seguimiento y la supervisión continúa de la misma, pero sobre todo por la motivación y el apoyo recibido a lo largo de estos años.

Especial reconocimiento merece el interés mostrado por mi trabajo y las sugerencias recibidas de los sinodales: Dr. Rosebet Miranda, Dr. Aníbal Arias, Dr. Agustín Santiago y Dr. Omar Caballero.

Un agradecimiento muy especial merece la comprensión, paciencia y el ánimo recibidos de todas y cada una de las personas que integran mi familia y amigos.

A todos ellos, muchas gracias.

Dedicatoria

A Dios.

Por permitirme llegar hasta este punto y haberme dado salud para lograr mis objetivos, además de su infinita bondad y amor.

A mi madre Guillermina.

Por darme todo su apoyo en todo momento, por sus consejos, sus valores, su bondad, por la motivación constante que me ha permitido ser una persona de bien, pero más que nada, por su amor y alegría.

A mi padre Vicente.

Por ser un pilar fundamental de mi formación académica y personal, por los ejemplos de perseverancia y constancia que lo caracterizan y que me ha infundado siempre, por el valor mostrado para salir adelante, por su esfuerzo, su voluntad y por su amor.

A mis familiares.

Merit, Eneas, Diomedes, Violeta y Leona, por estar conmigo y apoyarme siempre. Carlos, Delfino y Margarita por sus consejos. Y a todos aquellos que participaron directa o indirectamente en la elaboración de esta tesis.

Todo este trabajo ha sido posible gracias a ellos.

Índice

RESUMEN III
ABSTRACT V
AGRADECIMIENTOS VII
DEDICATORIA IX
ÍNDICE XI
ÍNDICE DE FIGURAS XV
ÍNDICE DE TABLAS XIX
CAPÍTULO 1 1
INTRODUCCIÓN 1
1.1 CONTEXTO 1
1.2 PROBLEMA A RESOLVER 3
1.3 JUSTIFICACIÓN 3
1.4 HIPÓTESIS 4
1.5 OBJETIVOS DEL TRABAJO 4
1.6 METAS 5
1.7 CONTRIBUCIONES 5

1.8 DESCRIPCIÓN DE LA SOLUCIÓN PROPUESTA 5
1.9 ORGANIZACIÓN DEL DOCUMENTO 7

CAPÍTULO 2 9

MARCO TEÓRICO 9

2.1 RECONOCIMIENTO DE OBJETOS 9
2.1.1 Métodos basados en apariencia 10
2.1.2 Métodos basados en características 11
2.1.3 Arquitectura de un sistema de reconocimiento de objetos 11
2.2 PROCESAMIENTO DIGITAL DE IMÁGENES 14
2.2.1 Representación de una imagen digital 14
2.2.2 Algoritmos usados en reconocimiento de objetos 15
2.3 CLASIFICACIÓN 23
2.3.1 Clasificación supervisada 25
2.3.2 Clasificación no supervisada 28
2.4 REDES NEURONALES ARTIFICIALES 29
2.4.1 Modelo biológico 30
2.4.2 Modelo artificial, conceptos generales 31
2.5 ESTADO DEL ARTE 34

CAPÍTULO 3 39

ALGORITMO HOG, RNA-MLP Y ALGORITMO EBP 39

3.1 ALGORITMO HOG 39
3.1.1 Cálculo del gradiente 40
3.1.2 Agrupación de orientaciones 42
3.1.3 Bloques de características 43
3.1.4 Normalización de bloques 44
3.2 RNA MLP Y ALGORITMO EBP 45
3.2.1 Perceptron Simple 45
3.2.2 Arquitectura del MLP 47
3.2.3 Algoritmo BackPropagation 50

CAPÍTULO 4 55

S-ROHM, SISTEMA DE RECONOCIMIENTO DE OBJETOS BASADO EN HISTOGRAMAS DE GRADIENTES ORIENTADOS Y PERCEPTRÓN MULTICAPA 55

4.1 EXTRACCIÓN DE CARACTERÍSTICAS MEDIANTE HOG 56
4.1.1 Gradiente de la imagen 56
4.1.2 Agrupación de orientaciones 59

4.1.3 Bloques de características y normalización 62
4.2 CLASIFICADOR RNA-MLP Y ALGORITMO EBP 64
4.2.1 Fase de Entrenamiento 65
4.2.2 Fase de Operación 69
4.3 INTEGRACIÓN DEL SISTEMA 70
4.3.1 Parámetros de configuración de S-ROHM 71
4.3.2 Interfaz Gráfica de Usuario de S-ROHM 72

CAPÍTULO 5 83

RESULTADOS EXPERIMENTALES Y DISCUSIÓN 83

5.1 ASPECTOS GENERALES DE LA EXPERIMENTACIÓN 83
5.1.1 Base de datos del sistema S-ROHM 84
5.1.2 Entrenamiento del sistema S-ROHM 86
5.1.3 Evaluación del sistema S-ROHM 87
5.2 EXPERIMENTO 1. DESEMPEÑO DEL S-ROHM PARA CLASIFICACIÓN BINARIA 88
5.3 EXPERIMENTO 2. DESEMPEÑO DEL S-ROHM PARA RECONOCIMIENTO DE PERSONAS CON LA BASE DE DATOS INRIA 94
5.4 EXPERIMENTO 3. DESEMPEÑO DEL S-ROHM PARA CLASIFICACIÓN MULTICLASE 95

CAPÍTULO 6 99

CONCLUSIONES Y TRABAJO FUTURO 99

6.1 CONCLUSIONES 99
6.2 TRABAJO A FUTURO 102

REFERENCIAS 105

Índice de figuras

Figura 1.1 Diagrama de bloques del sistema propuesto. 6
Figura 2.1 Diagrama de bloques de la arquitectura de un sistema de reconocimiento de objetos. 11
Figura 2.2 Ventana de pixeles vecinos de tamaño 3×3.....17
Figura 2.3 Ejemplo de proceso de filtrado en el dominio de la frecuencia [52].....19
Figura 2.4 Sección de imagen de tamaño 2×2. 20
Figura 2.5 Sección de imagen de tamaño 3×3. 21
Figura 2.6 Imagen de un clasificador simple de dos clases. 24
Figura 2.7 Esquema de un modelo Bayesiano [68]. 26
Figura 2.8 Representación de la frontera de decisión (g). 27
Figura 2.9 Clasificación de espacios de características en dos y tres dimensiones [68]. 27
Figura 2.10 Componentes de una neurona [93]. 31
Figura 2.11 Esquema de una neurona artificial. 32
Figura 2.12 Esquema de una RNA totalmente conectada. 33
Figura 3.1 Esquema del proceso de extracción de características HOG. 40
Figura 3.2 Máscara de operador derivativo para cálculo de gradiente. 41
Figura 3.3 Representación de la agrupación de orientaciones de una celda. a) Celda de tamaño 8×8 con orientaciones y sus contenedores. b) Representación del vector de votos ponderados. 42
Figura 3.4 Representación de un bloque rectangular (R-HOG) [1].....43
Figura 3.5 Representación de un bloque circular (C-HOG) [1]. 44

Figura 3.6 Funciones de activación a) Función signo. b) Función escalón. 46
Figura 3.7 Unidad de proceso (Perceptron simple)..... 47
Figura 3.8 Arquitectura del Perceptrón Multicapa..... 48
Figura 3.9 Funciones de activación del MLP. 50
Figura 4.1 Imagen de rostro humano a) Imagen en escala de grises. b) Magnitud del gradiente. c) Ángulos de orientación del gradiente (valor del ángulo proporcional a la intensidad de la imagen)..... 57
Figura 4.2 Imagen de rostro humano con mejora del gradiente a) Imagen en escala de grises. b) Magnitud del gradiente. c) Ángulos de orientación del gradiente (con mejora)..... 58
Figura 4.3 Representación de contenedores. a) Espaciado o separación de 9 *bins* uniformemente (20°). b) Histograma con base a los *bins* centrales. 61
Figura 4.4 Imagen de la clase girafa. a) Imagen original en escala de grises. b) Magnitud del gradiente. c) Ángulos del gradiente. d) Celdas de histogramas R-HOG. 61
Figura 4.5 Obtención de bloques de caracteristicas (descripción gráfica)..... 64
Figura 4.6 Esquema del Clasificador en fase de entrenamiento. 65
Figura 4.7 Diagrama de flujo del módulo Entrenamiento. 69
Figura 4.8 Esquema del Clasificador en fase de operación. 70
Figura 4.9 Panel o barra de parámetros de configuración..... 73
Figura 4.10 Ventana del módulo de extracción de características HOG. 75
Figura 4.11 Ventana del módulo de extracción de características HOG procesando imágenes. 76
Figura 4.12 Paneles de procesamiento individual de imágenes positivas (superior) y negativas (inferior)..... 76
Figura 4.13 Ventana de procesamiento del sistema de reconocimiento de objetos. 77
Figura 4.14 Sección Base de datos de la pestaña Reconocimiento..... 78
Figura 4.15 Sección Ventana de reconocimiento de la pestaña Reconocimiento..... 78
Figura 4.16 Sección de paneles de ejecución de la pestaña Reconocimiento..... 79
Figura 4.17 Sección de paneles procesamiento general y almacenamiento de archivos de la pestaña Reconocimiento. 80
Figura 4.18 Ventana de la pestaña Reconocimiento despues de procesar información. 81
Figura 4.19 Ventana de la pestaña Visor despues de procesar información..... 82
Figura 5.1 Imágenes positivas de algunas categorías de la base de datos Caltech 101. 84
Figura 5.2 Imágenes negativas de la base de datos Caltech 101..... 85
Figura 5.3 Detecciónes para proceso de clasificación binaria en imágenes con entornos completos (se detectan las clases de izquierda a derecha y de arriba abajo: *Airplane, Butterfly, Chair, Motorbikes*)..... 90
Figura 5.4 Falsos positivos en el proceso de clasificación binaria (se detectan las clases de izquierda a derecha y de arriba a abajo: *SoccerBall, Motorbikes, Airplane, Butterfly*)..... 91

Figura 5.5 Gráfica de evaluación de rendimiento ROC usando escalas logarítmicas (menos es mejor), diferencia entre dos configuraciones de números de celdas por ventana para la clase *Airplane*. 92
Figura 5.6 Gráfica de evaluación de rendimiento ROC usando escalas logarítmicas (menos es mejor), diferencia entre dos configuraciones de números de celdas por ventana para la clase *Faces*. 92
Figura 5.7 Gráfica de evaluación de rendimiento ROC usando escalas logarítmicas (menos es mejor), diferencia entre dos configuraciones de números de celdas por ventana para la clase *Motorbikes*. 93
Figura 5.8 Gráfica de evaluación de rendimiento ROC usando escalas logarítmicas (menos es mejor), diferencia entre dos configuraciones de números de celdas por ventana para la clase *Butterfly*. 93
Figura 5.9 Resultados del proceso de reconocimiento de personas. (a) S-ROHM para la base de datos INRIA. (b) Diferentes sistemas comparados con HOG para la base de datos INRIA [1]. 95

Índice de tablas

Tabla 4.1 Pseudocódigo del algoritmo que obtiene el gradiente de la imagen. 56
Tabla 4.2 Pseudocódigo del algoritmo que obtiene los histogramas de orientaciones. 60
Tabla 4.3 Pseudocódigo del algoritmo que obtiene los votos ponderados. 60
Tabla 4.4 Pseudocódigo del algoritmo que obtiene los bloques de características. 63
Tabla 4.5 Pseudocódigo del algoritmo que calcula la normalización de las características. 63
Tabla 4.6 Pseudocódigo del algoritmo que inicializa los pesos sinápticos de la RNA-MLP. 67
Tabla 4.7 Pseudocódigo del algoritmo de propagación de señales de la RNA-MLP. 68
Tabla 4.8 Pseudocódigo del algoritmo de decisión. 70
Tabla 4.9 Parámetros de configuración del sistema. 72
Tabla 4.10 Códigos de los parámetros de configuración del clasificador. 74
Tabla 4.11 Archivos generados por el sistema de reconocimiento de objetos. 80
Tabla 5.1 Detalles de las bases de datos utilizadas. 85
Tabla 5.2 Resultados del experimento 1 (clasificación binaria). 89
Tabla 5.3 Resultados del experimento 3 (clasificación multiclase). 96

Capítulo 1

Introducción

1.1 Contexto

La visión es uno de los mecanismos sensoriales de percepción más importantes que tiene el ser humano y muchos de los organismos biológicos; se utiliza para percibir el entorno que nos rodea y poder interactuar eficientemente con él. En este sentido es de gran importancia el poder detectar los objetos que son de nuestro interés por medio de su forma, color, relieve, dimensiones, distancia a la que se encuentra, etcétera.

Por su parte, la visión artificial o por computadora es una disciplina mediante la cual se dota a una máquina de la capacidad de percibir el mundo que le rodea, para deducir la estructura y las propiedades del mundo tridimensional a partir de una o más imágenes bidimensionales. De acuerdo con Ballard y Brown, la visión por computadora se refiere a la construcción de descripciones explícitas y significativas de los objetos físicos a partir de imágenes [2]. Es decir, la capacidad de percibir permite a una máquina extraer y analizar información *espectral, espacial y temporal* de los distintos objetos contenidos en una imagen. Mientras que la *información espectral* incluye frecuencia (color) e intensidad (tonos de gris), la *información espacial* se refiere a aspectos como forma y posición (una, dos y tres dimensiones) y la *información temporal*

comprende aspectos estacionarios (presencia y/o ausencia) y dependientes del tiempo (eventos, movimientos, procesos).

Debido a esta capacidad, la visión artificial es una disciplina en creciente auge y con una gran cantidad de aplicaciones que pueden ser clasificadas de acuerdo al tipo de tarea que realizan en [3]:

- **Detección de fallas:** se basa en un análisis cualitativo que involucra la detección de defectos o artefactos no deseados, con forma desconocida en una posición desconocida. Por ejemplo, encontrar defectos en la pintura de un auto nuevo, o agujeros en hojas de papel. [4], [5], [6].
- **Verificación:** chequeo cualitativo de que una operación de ensamblaje ha sido llevada a cabo correctamente. Por ejemplo, que no falte ninguna tecla en un teclado, o que no falten componentes en un circuito impreso. [7], [8], [9].
- **Reconocimiento:** involucra la identificación de un objeto con base en descriptores asociados con el objeto. Por ejemplo, la clasificación de cítricos (limones, naranjas, mandarinas, etc.) por color y tamaño. [10], [1], [11], [12], [13].
- **Identificación:** proceso de identificar un objeto por el uso de símbolos en el mismo. Por ejemplo, el código de barras, o códigos de perforaciones empleados para distinguir hule de espuma de asientos automotrices. [14], [15].
- **Análisis de localización:** evaluación de la posición de un objeto. Por ejemplo, determinar la posición donde debe insertarse un circuito integrado. [16], [17], [18], [19].
- **Guía:** significa proporcionar adaptativamente información posicional de retroalimentación para dirigir una actividad. El ejemplo típico es el uso de un sistema de visión para guiar un brazo robótico mientras suelda o manipula partes, otro ejemplo sería la navegación en vehículos autónomos [20], [21], [22], [23], [24].

La tarea de reconocimiento de objetos es de particular interés para esta investigación, sabiendo que esta tarea involucra los procesos de extracción de características y clasificación. Considerando lo mencionado, resulta evidente, y así lo expresan Pietikäinen *et al*. [25], que para obtener un buen desempeño en esta tarea es necesario contar con descriptores robustos y un clasificador de buena calidad.

Ahora se puede definir formalmente al reconocimiento de objetos como la tarea, dentro de un sistema de visión artificial, de encontrar e identificar objetos en una imagen o una secuencia de video.

Este tipo de tarea es tan importante que incluso se requiere para completar algunas de las otras tareas o aplicaciones mencionadas. Existe una gran cantidad de áreas y/o aplicaciones que hacen uso del reconocimiento de objetos, por ejemplo: clasificación de frutos [26], detección de rostros [27], detección de personas [28], reconocimiento de rostros [29], detección de placas [15],

seguimiento automático de objetos [30], reconocimiento automático de señales de tránsito [31], entre muchas más.

Dada la importancia que tiene la tarea de reconocimiento de objetos en diversas áreas de la ciencia y su gran potencial de aplicaciones, y con la finalidad de ofrecer una alternativa competitiva con las existentes actualmente, en este trabajo de tesis se presenta un sistema de reconocimiento de objetos basado en el algoritmo HOG, propuesto por Navneet Dalal y Bill Triggs en el año de 2005 [1], y redes neuronales artificiales (RNAs).

1.2 Problema a resolver

Se ha comentado que el objetivo de un sistema de visión artificial es dotar a una máquina con habilidades para percibir su entorno y poder interactuar con él. Una de estas habilidades es el reconocimiento de objetos, que involucra el poder detectar y determinar la identidad de los objetos de interés, siendo esta tarea uno de los retos más importantes para investigadores de esta área.

Aunque el reconocimiento de objetos es un campo de investigación muy activo, todavía es considerada una tarea en demasía compleja debido a las siguientes dificultades:

- La alta variabilidad de la apariencia de objetos del mismo tipo. Pueden existir objetos del mismo tipo con gran diversidad de forma, color y textura, además múltiples factores como la posición, la iluminación, las oclusiones, entre otras, pueden aumentar estas diferencias.
- La carencia de imágenes tomadas como referencia durante la fase de entrenamiento. Los pocos datos disponibles generalmente no son suficientes para cubrir la variabilidad en apariencia. Aunado a esto, pueden existir diferencias significativas en las condiciones del entrenamiento y de la operación del sistema.

El problema de visión por computadora sigue siendo un reto abierto para el cuál no existe algún algoritmo eficaz que reconozca todo tipo de objetos en cualquier ambiente y en el tiempo en que el sentido de la vista del ser humano lo realiza.

De acuerdo a lo mencionado en este apartado y a las definiciones, referentes al reconocimiento de objetos, se puede definir al reconocimiento de objetos como un problema de etiquetado que se basa en modelos de objetos conocidos. Dicho de otra manera, este problema consiste en, dada una imagen que contiene uno o más objetos de interés (y el fondo de la imagen) y un conjunto de etiquetas, una para cada modelo conocido por el sistema, el sistema debería asignar etiquetas correctas a regiones o conjunto de regiones en la imagen.

1.3 Justificación

Debido a que la utilización de dispositivos de visión artificial se ha vuelto de uso común en diferentes tareas, desde el hogar hasta la industria, y a que para la realización de sus tareas, como

la identificación de objetos, requiere un equipo de cómputo, aunque no siempre con la exactitud deseada, resulta evidente la necesidad de generar procesos y algoritmos para el reconocimiento de objetos específicos que mejoren el desempeño de los existentes.

Además, tratar de dotar a un sistema de habilidades que le permitan detectar y determinar la identidad de los objetos continúa siendo uno de los retos más importantes para el ser humano, ya que sistemas con esta habilidad no solo liberarían al hombre de tareas tediosas o peligrosas, sino que también podrían realizar algunas que son imposibles para él.

En esta investigación se ha puesto especial énfasis en el estudio de los HOGs para la etapa de extracción de características. Esto se debe a que en representaciones normalizadas del objeto, estos descriptores ofrecen información discriminativa de los objetos presentes en una imagen, siendo además robusta gracias a su invariancia ante cambios en la iluminación, en el fondo o en la posición del objeto.

En el estudio bibliográfico realizado, se observa que es común encontrar en un sistema de reconocimiento de objetos basado en el algoritmo HOG a un clasificador SVM (*Support Vector Machine*) como su complemento. Además, no fue posible encontrar reporte alguno donde implementen un sistema de reconocimiento de objetos que integre al algoritmo HOG con un clasificador que use un enfoque neuronal. En este sentido, la propuesta de este trabajo tiene por finalidad determinar si una RNA tipo MLP puede ser un mejor complemento que algoritmos como el SVM para un extractor de características basado en HOG.

También se busca explotar las características bien conocidas de la RNA tipo perceptrón multicapa (MLP) para solventar el problema que se presenta cuando los pocos datos disponibles durante el entrenamiento generalmente no son suficientes para cubrir la variabilidad en apariencia.

1.4 Hipótesis

El incluir un clasificador con un enfoque neuronal, como una RNA tipo MLP, en un sistema de reconocimiento de objetos que usa en la etapa de extracción de características al algoritmo HOG, puede mejorar su rendimiento.

1.5 Objetivos del trabajo

El objetivo principal del presente trabajo de tesis es diseñar e implementar un sistema de reconocimiento de objetos inmersos en imágenes estáticas utilizando en la etapa de extracción de características al algoritmo HOG y en la de clasificación a una RNA tipo MLP entrenada con el algoritmo *backpropagation*.

Para cumplir con el objetivo planteado, los siguientes objetivos secundarios son necesarios:

- Implementar el algoritmo HOG en un lenguaje de alto nivel.
- Implementar una RNA-MLP y el algoritmo *backpropagation* en un lenguaje de alto nivel.

- Implementar un sistema de reconocimiento de objetos integrado por el algoritmo HOG, en la etapa de extracción de características, y como clasificador una RNA-MLP.
- Implementar una interfaz de usuario que permita acceder a las diferentes fases que integran al sistema final.

1.6 Metas

1. Generar un esquema modular del algoritmo HoG, lo cual permitirá evaluar variantes en cada una de las etapas que lo integran.
2. Generar un sistema de reconocimiento de objetos integrado por el esquema generado en la Meta 1, en la etapa de extracción de características, y como clasificador una RNA-MLP.
3. Generar una interfaz gráfica de usuario (GUI, Graphical, User Interface) que permita evaluar el desempeño del sistema propuesto.

1.7 Contribuciones

El presente trabajo de tesis tiene como principales contribuciones las siguientes:

- Conformar un sistema de estructura modular enfocado al reconocimiento de objetos; la estructura con la que fue diseñado representa una plataforma de desarrollo que permite integrar y evaluar variantes en cada una de las etapas que lo integran.
- Integrar un extractor de características basado en HOG con un clasificador que opera con un enfoque neuronal.
- Generar un sistema capaz de llevar a cabo el reconocimiento de múltiples clases (objetos) dentro de una misma escena.

1.8 Descripción de la solución propuesta

Sistema de Reconocimiento de Objetos basado en Histogramas de gradientes orientados y perceptrón Multicapa (S-ROHM), es el sistema propuesto en este trabajo de tesis y producto de la misma. Se trata de un sistema que está orientado a la identificación automática de objetos y está integrado por los procesos de extracción de características y clasificación. La Figura 1.1 muestra los elementos que integran a S-ROHM y la relación existente entre ellos. S-ROHM fue desarrollo de este trabajo realizado mediante el lenguaje de programación JAVA.

Como lo muestra la figura, el proceso de extracción de características será realizado mediante el algoritmo HOG mientras que el proceso de clasificación se realizara mediante una RNA tipo MLP la cual será adaptada a una aplicación específica mediante el algoritmo *backpropagation*.

Siguiendo el esquema presentado en la Figura 1.1, se observa que el algoritmo HOG guarda una estructura modular compuesta por los siguientes elementos:

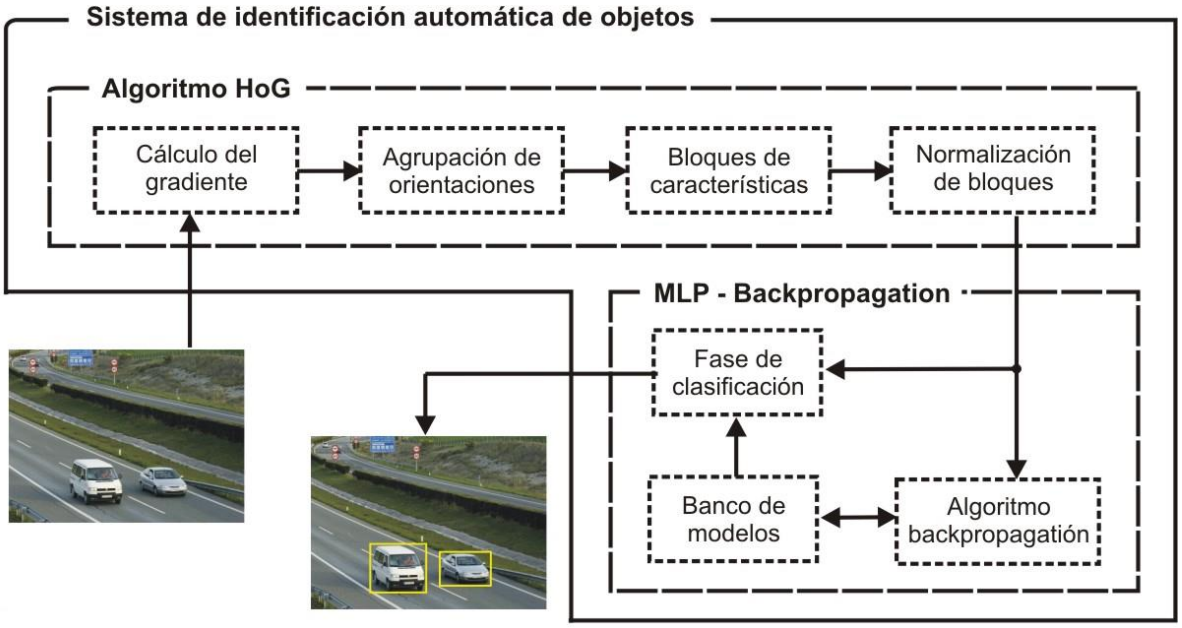

Figura 1.1 Diagrama de bloques del sistema propuesto.

Cálculo de gradiente. En este módulo se calcula el gradiente de la imagen utilizando filtros Gaussianos seguido por operadores derivativos de primer orden. Si se tiene una imagen de tamaño $N \times M$, el gradiente de la imagen se calcula para todos y cada uno de los pixeles de la imagen, obteniendo de esta forma dos vectores de tamaño $N \times M$, el primero contiene al gradiente y el segundo al ángulo de éste.

Agrupación de orientaciones. En este módulo, cada pixel procesa un "voto", que representa una función de la magnitud del gradiente, ponderado para el histograma de orientación de borde basado en la orientación del gradiente de cada elemento, cada voto se acumula en los contenedores (*bins*) de la región espacial local donde fue calculado, a esto se le llama "*celda*".

Bloque de características. Este módulo consiste en ordenar la información obtenida en los pasos anteriores; la forma en que se ordenan los histogramas de gradientes consiste en agrupar celdas para formar un "bloque". Cada ventana de detección, por lo tanto, está divida en una cantidad específica de bloques. Para el algoritmo HOG se consideran dos clases de geometrías: cuadradas o rectangulares, que particionan en cuadriculas las celdas, y las circulares, que particionan las celdas en forma logarítmica-polar. A estas clases se le denominan R-HOG y C-HOG para características HOG rectangulares y circulares respectivamente.

Normalización de bloques. Buscando obtener un mejor rendimiento, este módulo realiza una normalización local a los bloques generados por el módulo anterior.

Por su parte, como cualquier RNA, la empleada en este trabajo consta de 2 fases:

Fase 1. Aprendizaje. El algoritmo de aprendizaje *backpropagation* tiene por función adaptar el MLP a una aplicación específica. Para conseguirlo, hace uso de los resultados generados por el algoritmo HOG en la construcción el banco de modelos.

Fase 2. Clasificación u operación. Una vez formado el banco de modelos, el MLP entra en su fase de clasificación y puede, utilizando los resultados entregados por el algoritmo HOG, identificar los objetos indicados o contenidos en el banco de modelos.

1.9 Organización del documento

La estructura de este documento de tesis está dividida en 6 capítulos, detallados a continuación.

Capítulo 1. Introducción. Capítulo presente, donde se explica en forma sucinta el tema a desarrollar.

Capítulo 2. Marco teórico. Este capítulo contiene una breve descripción de los conceptos fundamentales necesarios para la comprensión del trabajo desarrollado. Iniciando con una introducción al tema de reconocimiento de objetos, seguida de la conceptualización del procesamiento digital de imágenes, haciendo énfasis en algoritmos empleados en el reconocimiento de objetos, para posteriormente abordar el tema de clasificación poniendo especial atención en el enfoque neuronal; el capítulo finaliza con el obligado tema del estado del arte.

Capítulo 3. Algoritmo HOG, RNA-MLP y algoritmo EBP. En este capítulo se describen en forma detallada el algoritmo HOG, la RNA tipo MLP y el algoritmo de aprendizaje *backpropagation*, todos ellos conforman los métodos utilizados en este trabajo de tesis.

Capítulo 4. S-ROHM, Sistema de Reconocimiento de Objetos basado en Histogramas de gradientes orientados y perceptrón Multicapa. Este capítulo describe explícitamente el desarrollo del sistema propuesto en este trabajo de tesis.

Capítulo 5. Resultados experimentales y discusión. Este capítulo consta de una serie de experimentos que tienen por finalidad mostrar el desempeño del sistema resultante de esta tesis y compararlo con esquemas similares documentados.

Capítulo 6. Conclusiones y trabajo futuro. Este capítulo presenta las conclusiones obtenidas de este trabajo de tesis y las perspectivas o trabajos futuros que se plantean para la continuación de esta investigación.

Al final de la tesis, se presenta la bibliografía utilizada como base para el desarrollo de la presente investigación.

Capítulo 2

Marco Teórico

Este capítulo contiene los fundamentos teóricos de los temas involucrados en el desarrollo de este trabajo de tesis. En la primera parte se incluyen los aspectos relevantes del tema de reconocimiento de objetos, describiendo la arquitectura de un sistema de reconocimiento y detallando brevemente algunos conceptos generales referentes a este tema. Posteriormente, se exponen los conceptos y definiciones básicas del procesamiento digital de imágenes, necesarias para el entendimiento y desarrollo de sistemas de reconocimiento de objetos. Finalmente, se aborda el tema de clasificación, que representa a la última de las fases de un sistema de reconocimiento de objetos.

2.1 Reconocimiento de objetos

En los sistemas inteligentes o de inteligencia artificial, el sistema de reconocimiento de objetos tiene por finalidad permitirle interactuar eficientemente con su entorno, dotándolo con habilidades que le permitan detectar la presencia de objetos de su interés.

El tratar de dotar a un sistema de habilidades que le permitan detectar y determinar la identidad de los objetos continúa siendo uno de los retos más importantes para el ser humano. Sistemas con esta habilidad no solo liberarían al hombre de tareas tediosas o peligrosas, sino que también podrían realizar algunas que son imposibles para el ser humano. Disciplinas como procesamiento

de imágenes, reconocimiento de patrones y visión artificial son de imperiosa necesidad en el diseño y construcción de un sistema de reconocimiento de objetos.

Según Forsyth D. A. y Ponce Jean, el reconocimiento de objetos o reconocimiento de patrones consiste en comparar una imagen o parte de ésta con alguna información almacenada en su propia base de datos con la finalidad de identificarla [32].

Por su parte William R. Uttal y Hillsdale N. J. [33], definen al reconocimiento de objetos, desde un punto de vista psicológico, como la acción de conceptuar, categorizar o clasificar un determinado estimulo como miembro de una clase de estímulos.

Más acorde al área de estudio de este trabajo, está la definición vertida por Tou J. T. y Gonzalez R. C., donde la establecen como la categorización de datos de entrada en clases identificadas, por medio de la extracción de características significativas o atributos de los datos extraídos de un medio ambiente que contiene detalles irrelevantes [34].

De acuerdo a estas definiciones y en forma general, el problema del reconocimiento de objetos se puede definir como un problema de etiquetado que se basa en modelos de objetos conocidos. De manera formal, este problema consiste en, dada una imagen que contiene uno o más objetos de interés (y el fondo de la imagen) y un conjunto de etiquetas, una para cada modelo conocido por el sistema, el sistema debería asignar etiquetas correctas a regiones o conjunto de regiones en la imagen.

A lo largo de los años han sido desarrollados e implementados múltiples métodos para el reconocimiento de objetos en imágenes digitales, principalmente se pueden clasificar en dos categorías [10], [35], [36], [37]: aquellos basados en apariencia y aquellos basados en características.

2.1.1 Métodos basados en apariencia

El reconocimiento visual inicialmente centro sus investigaciones en el reconocimiento de patrones simples, principalmente caracteres alfanuméricos. Estos patrones eran comparados con prototipos almacenados en memoria. La metodología consistía en que para cada letra o número debería existir un prototipo, o mejor conocido como plantilla. Indiscutiblemente, esta metodología a simple vista puede ser reconocida como una de las más costosas en cuanto a recursos computacionales se refiere y a robustez, ya que, un mismo objeto se ve diferente bajo gran cantidad de condiciones, entre ellas: cambio en tamaño o forma del objeto, cambio en dirección o sentido de observación y/o cambio en el color o iluminación, haciéndolo un método poco confiable y de gran demanda computacional para subsanar estas carencias. Por lo tanto, este método sería inútil para reconocer objetos naturales o patrones complejos [38]. Un proceso de normalización, el cual ajustara en tamaño, orientación, color, etc. los objetos antes de la comparación con las plantillas almacenadas, es una posible solución a este problema [37].

2.1.2 Métodos basados en características

Teóricamente esta postura contempla que el sistema de reconocimiento disponga de detectores específicos de ciertas características. Por lo tanto, para un patrón determinado su reconocimiento se obtendría por medio de la detección de las características definitorias de éste [37]. Un paso, ejercicio o método utilizado para poder definir un objeto dentro de una imagen digital consiste en la extracción de puntos característicos que puedan ser usados para encontrar emparejamientos factibles entre patrones de objeto e imagen.

2.1.3 Arquitectura de un sistema de reconocimiento de objetos

Los bloques que constituyen un sistema de reconocimiento de objetos en imágenes estáticas son los siguientes:

- Base datos o banco de modelos
- Captura o digitalizador de la imagen.
- Preprocesamiento o acondicionamiento.
- Segmentación.
- Extracción de características.
- Clasificador.

Un diagrama de bloques de un sistema de reconocimiento se puede apreciar en la Figura 2.1.

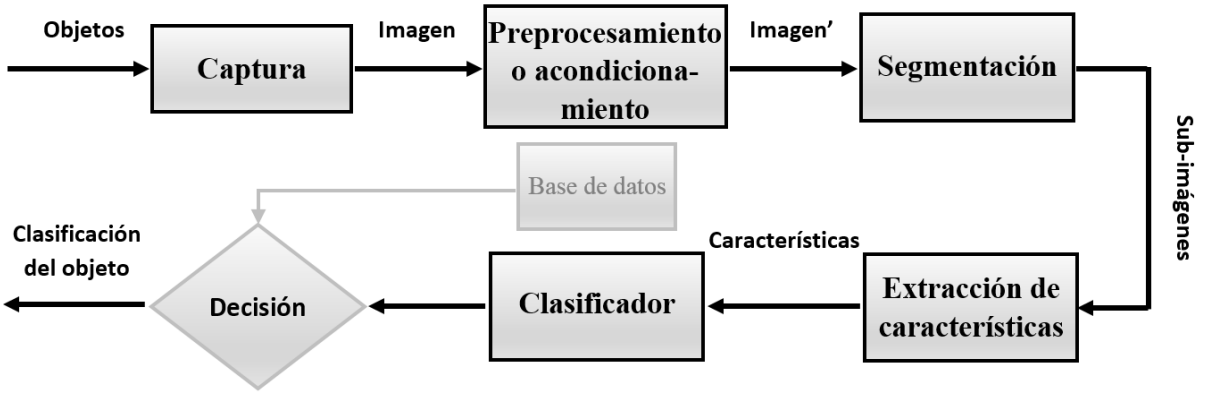

Figura 2.1 Diagrama de bloques de la arquitectura de un sistema de reconocimiento de objetos.

La **base de datos o banco de modelos** (o descripciones) contiene la información de todos los objetos que el sistema conoce. Esta información es creada y utilizada por el clasificador para cumplir su tarea. Normalmente, la información contenida por este bloque es organizada en vectores de *rasgos* abstractos, donde un rasgo es un atributo propio de un objeto que es utilizado para su descripción y discriminación con respecto a otros objetos. Todos los vectores que

cumplen cierto criterio de certidumbre son agrupados en un elemento del banco de modelos denominado *clase*.

El **bloque de captura** hace referencia al proceso de adquisición de imágenes, los sensores transforman la intensidad de la luz que es reflejada por los objetos a cargas eléctricas, generando una señal. Esta señal posteriormente es digitalizada, que podrá de ser interpretada por los sistemas de cómputo para así generar una imagen digital. Una imagen digital es medida en pixeles, cada pixel representa un valor en código binario del tono de color de la imagen.

En el bloque de **preprocesamiento o acondicionamiento** de la imagen se realizan cálculos sobre ésta con el fin de mejorar, restaurar o normalizar sus propiedades. En ocasiones es necesario mejorar la imagen con el fin de eliminar aspectos no deseados, como por ejemplo ruido, mejorar contraste; así mismo, es útil resaltar ciertos aspectos o propiedades de la imagen. Cualquier operación o proceso que sea aplicado a una imagen suele llamarse transformación, así, es válido pensar que al transformar una imagen se obtiene una nueva imagen. Se puede concluir que el objetivo de este bloque es entregar una imagen transformada que permitirá que bloques posteriores tengan mejores posibilidades de éxito.

Aunque en muchos casos la segmentación es omitida del proceso de reconocimiento, ambos están íntimamente relacionados entre sí, esto se debe a que sin un reconocimiento al menos parcial de los objetos, la segmentación no es posible; de la misma manera, sin segmentación previa no se puede llevar acabo el reconocimiento de objetos. Es decir, aunque un diagrama de bloques de un sistema de reconocimiento omita el bloque de segmentación, éste está intrínsecamente incluido. Por otro lado, cada vez es más frecuente incluir un bloque de segmentación después del de preprocesamiento (ver por ejemplo [39] y [40]). El proceso de **segmentación** particiona o divide la imagen en un conjunto de sub-imágenes buscando que cada sub-imagen se aproxime en lo posible a la región de cada uno de los objetos de la imagen.

La **extracción de características** aplica operadores sobre una imagen segmentada o no segmentada, con la finalidad de transformar la información del objeto observado en valores simbólicos que definen rasgos propios del objeto y que ayudaran en la formación de hipótesis sobre la presencia de dicho objeto en la escena en análisis. En general, este bloque genera un conjunto de rasgos o características que son utilizadas por el clasificador de datos para generar el banco de modelos o para realizar el proceso de reconocimiento.

El bloque **clasificador** utiliza el conjunto de rasgos generados por el extractor de características para llevar a cabo las dos fases que lo integran, el aprendizaje, mediante el cual crea al banco de modelos, y el reconocimiento, que permite determinar cuál o cuáles de los objetos pertenecientes al banco de modelos está presente en la imagen de estudio. El proceso de reconocimiento de un objeto significa asociarlo a la clase con la que presenta mayor grado de certidumbre; es decir, este proceso le otorga al sistema la facultad de decidir a qué clase corresponde cada objeto procesado, [41], [38], [42].

En general, la operación de un sistema de reconocimiento de objetos empieza con el modelado o descripción del objeto, lo cual consiste en representarlo tomando en cuenta sus atributos, usualmente denominados rasgos o características, que permitan fácilmente diferenciarlo de otros objetos presentes en la escena en estudio. Es típico representar a un objeto a través de una tupla de *n* rasgos (*n*-tupla), de manera que formen un vector columna o un vector fila.

Una vez representados mediante *n*-tuplas todos los objetos que se desea el sistema sea capaz de reconocer, y considerando a cada tupla una clase, éstas son utilizadas para formar la base datos o banco de modelos del sistema. Completada esta tarea, el sistema de reconocimiento está completo y listo para empezar la fase de reconocimiento.

Para la fase de reconocimiento, la *n*-tupla que describe un objeto dado es comparada con las descripciones de todos los objetos del universo de trabajo aprendidas por el sistema y previamente almacenadas en el banco de modelos. El reconocimiento de un objeto dado, consiste en identificar de qué objeto se trata encontrando el mayor grado de similitud o pertenencia que tiene con cada posible objeto previamente definido.

2.1.3.1 Propiedades de un rasgo

Hasta ahora se ha mencionado constantemente el término rasgo, e incluso se ha vertido su definición, pero no se ha profundizado lo suficiente en este tema de gran importancia para el proceso de reconocimiento de un objeto. Por tal motivo, este sub-apartado toca los aspectos relacionados con este atributo de un objeto.

Para que un rasgo resulte de utilidad para el proceso de reconocimiento de un objeto, es deseable que posea un conjunto de propiedades, entre las que destacan las siguientes [43]:

- Discriminación. Los rasgos deben permitir discriminar objetos de diferentes clases y unir los de una misma clase; tienen que producir valores numéricos diferentes para objetos de clases distintas y valores similares para los de una misma clase.
- Fiabilidad. Las características obtenidas deben ser altamente confiables; es decir, los valores de las características de objetos de la misma clase deben tener cambios pequeños.
- Incorrelación. Las características deben guardar la menor relación entre ellas.
- Rapidez. Los tiempos de cálculo de los rasgos deben ser mínimos; estos tiempos normalmente están determinados por la aplicación.
- Economía. Los sensores, captores, transductores utilizados para obtener las características de un objeto deben ser económicos.

Además, un rasgo puede ser:

- Una parte del objeto con algunas propiedades especiales, por ejemplo, una línea, un conjunto de puntos, una parte de la superficie del objeto, bordes, vértices, una región de una textura determinada, etc.

- Una propiedad global extraída del objeto completo o de una parte del mismo, por ejemplo, el promedio del nivel de gris del objeto, el histograma, el área en píxeles contenidos por una región del objeto, etc.

Finalmente, es importante distinguir entre dos tipos de rasgos de un objeto:

- Índices visuales. Son aquellos rasgos que debido a su posición en el objeto resultan de particular interés, como esquinas o puntos de gran curvatura, o segmentos de una línea recta o de una curva.
- Rasgos objeto. Se refieren a mediciones geométricas o topológicas como el área obtenida a partir del contorno del objeto o la región de píxeles que forman al objeto.

2.2 Procesamiento digital de imágenes

En este apartado se exponen los conceptos y definiciones básicas, relacionadas con el procesamiento digital de imágenes, necesarias para el entendimiento y desarrollo de sistemas de reconocimiento de objetos.

2.2.1 Representación de una imagen digital

Una imagen analógica puede ser definida como una función real $i(x,y)$ con integral finita y soporte compacto S tal que, para todo punto $p \in S$, $x > 0$, $y > 0$.

La representación discreta de una imagen analógica, llamada imagen digital, es una matriz de bits que definen tanto la intensidad como el color de cada pixel en una imagen. Un pixel es el elemento mínimo de una imagen digital y se define como cada una de las celdas o casillas en las que se puede descomponer ésta. Dependiendo de cómo se codifica el color de la imagen se determina la cantidad de bits que necesita cada pixel para guardar toda información de la imagen. Considerando a $\Im$ el conjunto de los enteros, una imagen digital I, denotada como $f(x, y)$, es un arreglo bidimensional $\in \Im \times \Im$ si (x, y) son enteros de $\Im \times \Im$ y es una función f que asigna a cada par (x, y) un número de $\Im$ [43]. La ecuación 2.1 representa la definición matemática de la intensidad de un pixel.

$$I_{pixel} = f(x, y) \tag{2.1}$$

donde el valor de f representa la intensidad de color en las coordenadas espaciales (x, y) en ese punto de la imagen.

Actualmente las computadoras poseen tarjetas gráficas con capacidades avanzadas de procesamiento, debido a esto, la mayoría de las imágenes son del tipo "color verdadero" (del inglés "*true color*"), mejor conocido como espacio RGB (*Red*, *Green*, *Blue*) el cual maneja tres canales de colores, uno para el color rojo, color verde y un último para color azul, estos son conocidos como colores primarios y de la combinación de éstos resulta una amplia gama de

tonalidades que representa la imagen digital. En este tipo de codificaciones son necesarios 24 bits (3 bytes) para cada pixel de la imagen, cada color primario, por lo tanto, resulta en una codificación de 8 bits (1 byte) que recae en un rango de 0 a 255, tomando así 256 valores posibles. Debido a esta característica, los pixeles RGB son considerados tridimensionales. La representación de una imagen RGB es definida en la ecuación 2.2, en donde el valor de f es la intensidad del color rojo, verde y azul respectivamente, en las coordenadas espaciales (x, y) del $Pixel_{RGB}$.

$$Pixel_{RGB} = f(R, G, B) \tag{2.2}$$

Una imagen en escala de grises (*grayscale*) es una imagen que está únicamente representada en tonalidades de gris. En este tipo de imágenes los valores de sus componentes rojos, verdes y azules son del mismo valor, por lo tanto, esta clase de imágenes puede ser representada con menor información especificando únicamente un solo valor por cada pixel, que representa su intensidad (o nivel de gris); debido a esta causa, es común denominar a este tipo de imagen como "imagen intensidad". Por lo mencionado, resulta evidente que en esta codificación solo se requieren 8 bits por pixel, lo que permite representar 256 tonalidades de gris que van desde el tono negro al blanco, y economizar en recursos (memoria y potencia de cálculo) necesarios para su almacenamiento y procesamiento. La definición expuesta sobre imagen digital, y denotada por la ecuación 2.1, se puede adaptar para una imagen en escala de grises cuando el valor de $f(x, y) \in [0{,}255]$. La mayor parte del procesamiento desarrollado en este trabajo de tesis será realizado sobre imágenes representadas en escala de grises.

Una representación más de una imagen digital es conocida comúnmente como imagen en blanco y negro o imagen binaria. Se trata de una imagen digital donde $f(x, y)$ ha sido cuantificada a dos niveles de intensidad, 0 y 1 (0 representa el color negro y 1 al blanco).

Finalmente, una imagen digital simbolizada por una matriz de $M \times N$ elementos, es la representación de una imagen analógica en forma aproximada por una serie de muestras (cantidades discretas) espaciadas equitativamente [44], [45], [40]:

$$f(x, y) \approx \begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0, M-1) \\ f(1,0) & f(1,1) & \cdots & f(1, M-1) \\ \vdots & \vdots & & \vdots \\ f(N-1,0) & f(N-1,1) & \cdots & f(N-1, M-1) \end{bmatrix} \tag{2.3}$$

2.2.2 Algoritmos usados en reconocimiento de objetos

Según Petrou y Bosdogianni, el desarrollo de algoritmos de procesamiento de imágenes se ha dado en respuesta a tres grandes problemas concernientes a imágenes [46]:

- La digitalización y codificación de imágenes que facilite la transmisión, impresión y almacenamiento de las mismas.
- Mejora y restauración de una imagen para interpretar más fácilmente su contenido sobre una superficie.
- Descripción y segmentación de imágenes para una etapa inicial de visión robótica.

La gran cantidad de algoritmos de procesamiento de imágenes existentes, hace necesaria su clasificación, de acuerdo a [47], [48] y [49], las técnicas de procesamiento de imágenes entran en una de las siguientes categorías:

- Realce y mejora de la imagen.
- Restauración de la imagen.
- Compresión de imágenes.
- Segmentación.

Para este trabajo de tesis, son de interés los algoritmos de procesamiento de imágenes relacionados con el proceso de reconocimiento de objetos. Por tal motivo, este apartado hace referencia a tales algoritmos poniendo énfasis en aquellos que son utilizados en esta investigación, los pertenecientes al realce y mejora de la imagen y la segmentación.

2.2.2.1 Realce y mejora de la imagen

Todos aquellos algoritmos de procesamiento de imágenes destinados a resaltar, agudizar y/o contrastar determinados aspectos de la imagen, y también aquellos que ayudan a eliminar efectos no deseados sobre ellas, como toda clase de ruido (aditivo, sustractivo, multiplicativo, etc.), son técnicas de realce o mejora de la imagen [50].

Para completar esta definición, se debe mencionar que estas mejoras pueden ser, además de reducir el ruido, en cuanto a contraste, escala de grises, distorsiones, falta de nitidez, luminosidad, brillo, etc., o bien convertir la imagen a una mejor forma para su análisis.

Jahne B. argumenta que el objetivo principal de este tipo de procesamiento es obtener información de las propiedades físicas que están en la imagen mediante transformaciones aplicadas directamente sobre ella, logrando que éstas sean de mayor importancia y utilidad en las etapas posteriores [51].

Actualmente existe una enorme cantidad de operaciones o transformaciones que pueden ser realizadas sobre las imágenes digitales con el fin de realzarlas y mejorarlas, por lo que éstas son agrupadas en cuatro categorías:

El conjunto de algoritmos de realce y mejora de imagen comúnmente es dividido en dos grandes grupos:

- Algoritmos en el dominio espacial.
- Algoritmos en el dominio de la frecuencia.

2.2.2.1.1 Algoritmos en el dominio espacial

En esta categoría se utilizan métodos que procesan la imagen a nivel de pixeles, es decir, se realiza un recorrido para todos y cada uno de los pixeles, en los cuales se llevan a cabo operaciones de mejora o transformación tomando en cuenta la vecindad del pixel o simplemente el mismo pixel según el método. Matemáticamente tenemos la siguiente descripción:

$$\mathrm{g}(x,y) = T[f(x,y)] \qquad \textbf{(2.4)}$$

donde, $f(x,y)$ es la imagen de entrada, $g(x,y)$ es la imagen procesada o de salida, y T es el operador de vecindad que se aplica sobre la imagen para la mejora. Si los vecinos que se encuentran en operación resultan ser una matriz de 1×1, se obtiene la definición simple de T, ya que $g(x,y)$ depende únicamente del pixel (x,y), el cual resulta ser el pixel que se está procesando. A todo este proceso se le conoce con el nombre de mapeo (*mapping*) y se puede definir como sigue:

$$\mathrm{s} = T(r) \qquad \textbf{(2.5)}$$

donde, r representa al pixel que está siendo procesado por el operador T, mientras que s se define como el pixel resultante.

Una *ventana*, *plantilla* o *kernel* se le denomina al conjunto de pixeles vecinos a cierto pixel. Una ventana de pixeles en vecindad se puede observar en la Figura 2.2, en donde se remarca el pixel actual como (x,y).

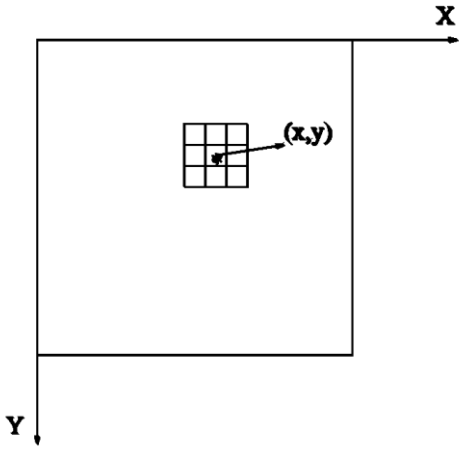

Figura 2.2 Ventana de pixeles vecinos de tamaño $\mathbf{3 \times 3}$.

Los algoritmos de realce o mejora de la imagen en el dominio del espacio más utilizado son: histograma de la imagen, negativo, ecualización del histograma, entre otros [52].

2.2.2.1.2 Algoritmos en el dominio de la frecuencia

Las técnicas de mejora en el dominio de la frecuencia tienen sus fundamentos en el uso de la transformada de Fourier de la imagen. En 1822, Fourier afirmó que: cualquier función que se repita a si misma periódicamente puede ser expresada como la suma de senos y/o cosenos de diferentes frecuencias, cada uno multiplicado por un coeficiente distinto [53].

Estas técnicas de representación proporcionan a detalle con qué frecuencia se repiten ciertas características en una imagen, consiguiendo representar información de la imagen de forma útil, ya que, teniendo esta frecuencia de repetición se puede detectar o alterar directamente elementos característicos dentro de la imagen, como lo es el ruido, bordes o texturas.

Las imágenes digitales son de tipo discreto, por lo tanto, se trabaja con la Transformada Discreta de Fourier (DFT, *Discrete Fourier Transform*). Para una imagen de tamaño $M \times N$, la DFT está definida por la ecuación 2.6:

$$\mathrm{F}(u,v) = \frac{1}{M\,N}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-j2\pi\left({}^{ux}/_{M}+{}^{vy}/_{N}\right)} \qquad \textbf{(2.6)}$$

donde $u = 0,1,2,\dots,M-1$ y $v = 0,1,2,\dots,N-1$ son las variables en el dominio de la frecuencia. Por otra parte, se tiene que $f(x,y)$ se obtiene al aplicar la transformada inversa como se muestra en la siguiente ecuación:

$$f(x,y) = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F(u,v)e^{j2\pi\left({}^{ux}/_{M}+{}^{vy}/_{N}\right)} \qquad \textbf{(2.7)}$$

donde $x = 0,1,2,\dots,M-1$ y $y = 0,1,2,\dots,N-1$ son las coordenadas espaciales (x,y) de la imagen digital.

Tomando en cuenta la información anterior, una representación gráfica del proceso de mejora de imagen en el dominio de la frecuencia se puede observar en la Figura 2.3, en la cual se muestra un ejemplo de los componentes de un proceso de filtrado en el dominio de la frecuencia.

Finalmente, es necesario mencionar que la selección de un método apropiado de realce o mejora de la imagen y la elección de los parámetros adecuados, dependen directamente de la imagen original y de la aplicación.

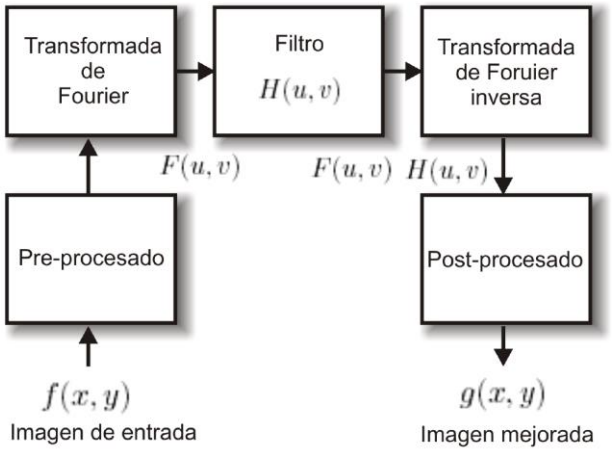

Figura 2.3 Ejemplo de proceso de filtrado en el dominio de la frecuencia [52].

2.2.2.2 Segmentación

La segmentación es una técnica de procesamiento que toma como entrada una imagen y genera como salidas atributos extraídos de ésta. Para conseguirlo, la segmentación subdivide la imagen en sus regiones u objetos constituyentes, de tal manera que los píxeles de esas regiones posean propiedades o atributos idénticos, como niveles de gris, contraste o texturas.

La mayoría de los algoritmos de segmentación están basados en dos propiedades básicas de intensidad de la imagen: la discontinuidad y la similitud [54].

En la categoría de segmentación mediante discontinuidad, el proceso se realiza dividiendo a la imagen basándose en cambios abruptos de intensidad, tal como ocurre con los bordes de una imagen.

Con respecto a la segmentación con base en la similitud, ésta es lograda mediante la partición de una imagen en regiones que son similares de acuerdo a un conjunto de criterios predefinidos.

En los sub-apartados siguientes de describen algunos algoritmos que forman parte de esta categoría. Desde que esta investigación se basa en un algoritmo que forma parte del tema de extracción de características y que el histograma es parte importante de este algoritmo, también se han incluido estos temas.

2.2.2.2.1 Detección de bordes

Es posible identificar las características de un objeto dentro de una imagen digital a través de la detección de su borde. Un borde o contorno es el límite entre regiones con pixeles de diferentes niveles de intensidad. Las técnicas clásicas de detección de bordes se basan en la aplicación de un operador derivativo local para identificar las discontinuidades en los niveles de intensidad de los pixeles, esto es, encontrar la derivada respecto a los ejes x y y, lo que se conoce como gradiente de imagen. Para una imagen continua $f(x,y)$ su derivada toma un máximo local en la

dirección del borde. Así, una técnica de la detección de bordes consiste en medir el gradiente de f a lo largo de r en la dirección de θ, esto quiere decir que:

$$\nabla f = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right) \tag{2.8}$$

$$\frac{\partial f}{\partial r} = \frac{\partial f}{\partial x}\frac{\partial x}{\partial r} + \frac{\partial f}{\partial y}\frac{\partial y}{\partial r} = f_x cos\theta + f_y sen\theta \tag{2.9}$$

La magnitud del gradiente (∇f) se calcula como se muestra en la ecuación 2.10.

$$|\nabla f| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2} \tag{2.10}$$

En otras palabras, el valor máximo de $\frac{\partial f}{\partial r}$ se obtiene cuando $\frac{\partial}{\partial \theta}\frac{\partial f}{\partial r} = 0$ (ecuación 2.11):

$$-f_x sen\theta_g + f_y cos\theta_g = 0 \Rightarrow \theta_g = tan^{-1}\left(\frac{f_y}{f_x}\right) \tag{2.11}$$

$$\left(\frac{\partial f}{\partial r}\right)_{max} = \sqrt{{f_x}^2 + {f_y}^2} \tag{2.12}$$

donde θ_g es la dirección del gradiente. En el caso discreto, se puede aproximar la derivada tomando de forma simple la diferencia entre dos valores contiguos. Considerando una sección de imagen de 2×2 elementos (Figura 2.4).

$I_{1,1}$	$I_{1,2}$
$I_{2,1}$	$I_{2,2}$

Figura 2.4 Sección de imagen de tamaño $\mathbf{2 \times 2}$.

Una aproximación discreta al gradiente en dicha región es entonces:

$$\frac{\partial f}{\partial x} = I_{1,2} - I_{1,1} \tag{2.13a}$$

$$\frac{\partial f}{\partial y} = I_{2,1} - I_{1,1} \tag{2.13b}$$

donde $\left(\frac{\partial f}{\partial x}\right)$ es el gradiente horizontal y $\left(\frac{\partial f}{\partial y}\right)$ es el gradiente vertical. Tambien se puede extender esta aproximación a un área de la imagen de 3×3, como se muestra en la Figura 2.5.

$I_{1,1}$	$I_{1,2}$	$I_{1,3}$
$I_{2,1}$	$I_{2,2}$	$I_{2,3}$
$I_{3,1}$	$I_{3,2}$	$I_{3,3}$

Figura 2.5 Sección de imagen de tamaño 3 × 3.

Entonces, el gradiente es aproximado como sigue:

$$\frac{\partial f}{\partial x} = \left(I_{1,3} + I_{2,3} + I_{3,3}\right) - \left(I_{1,1} + I_{2,1} + I_{3,1}\right) \quad \textbf{(2.14a)}$$

$$\frac{\partial f}{\partial y} = \left(I_{3,1} + I_{3,2} + I_{3,3}\right) - \left(I_{1,1} + I_{1,2} + I_{1,3}\right) \quad \textbf{(2.14b)}$$

Estas operaciones son frecuentemente implementadas mediante operadores o mascaras (ventanas) [55]. Las transformaciones basadas en diferencias entre pixeles vecinos son sensibles al ruido y para reducir este efecto se han propuesto diferentes variaciones de ventanas; las más usadas son: operador Roberts [56], operador Sobel [57] y operador Prewitt [58].

2.2.2.2.2 Extracción de características

Hoy en día la extracción de información de las imágenes digitales a través del procesamiento digital constituye un enorme campo de investigación en diversas ramas del conocimiento y con diferentes tipos de aplicaciones. En este sentido, investigadores de diversas áreas se mantienen realizando investigaciones en temas que van desde la aplicación de filtros lineales simples hasta la automatización del reconocimiento semántico de objetos. La visión artificial, correspondiente a la inteligencia artificial, es el sub-campo donde la detección automática de características sobre imágenes digitales ha encontrado un nicho para el desarrollo de una gran cantidad de métodos para tal propósito. A pesar de esta gran proliferación de trabajos en este sub-campo, no existe un método general para la extracción automática de características, sino que son los requerimientos directos del sistema en desarrollo son los que obligan a personalizar y desarrollar un método propio.

De acuerdo con Gonzales y Woods, el objetivo principal de este tipo de algoritmos es la obtención de elementos característicos implícitos y exclusivos de un objeto que permitan identificarlo y/o diferenciarlo de otros objetos que se encuentran en la misma imagen o en una diferente [40].

Una imagen digital contiene gran cantidad de datos pero estos no proporcionan la información suficiente para obtener una interpretación de la escena. Las características extraídas deben cumplir principalmente las siguientes condiciones [40]:

- Su extracción a partir de la imagen digital no debe representar para el sistema un costo excesivo de recursos. Así como también, debe presentar un tiempo de procesamiento lo más pequeño que sea posible sin representar una pérdida de exactitud.
- Su localización debe ser lo más exacta o precisa posible. El error de estimación cometido al realizar la extracción debe ser pequeño.
- Robustez y estabilidad son propiedades inherentes. Deben mantener sus cualidades a lo largo de una secuencia o procesos, además, la extracción debe ser insensible al ruido de captura e iluminación.
- Deben contener la máxima información posible de la escena.
- Tienen que poseer ciertas invarianzas dependientes del sistema en desarrollo como:
    - Traslación: los valores de las características son independientes de su posición en la imagen.
    - Rotación y escalado: tamaño y orientación del objeto.
    - Transformaciones no lineales de deformación, comúnmente llamado perspectiva.

Las características extraídas de la imagen pueden ser:

- Topológicas: agujeros, número de componentes conexas, etc.
- Geométricas: perímetro, área, curvatura, etc.
- Estadísticas: momentos entre otras.

La forma de representar las características extraídas de las regiones u objetos segmentados es mediante el uso de vectores de características normalizados, un ejemplo se muestra en la ecuación 2.15.

$$\boldsymbol{c} = [c_1, c_2, \dots, c_n]^T \tag{2.15}$$

donde $\boldsymbol{c}$ es el vector y c_i son las características. También se suele usar una representación en formato de árbol para características estructurales, pero, para este proyecto esta representación no será utilizada.

Una gran cantidad de algoritmos forman parte del paradigma de extracción de características de objetos inmersos en imágenes, como ejemplos se pueden mencionar, la transformada SIFT (*Scale Invariant Feature Transform*) [10], transformada LBP (*Local Binary Pattern*) [59], descriptores SURF (*Speeded Up Robust Features*) [60], transformada de Hough [61], momentos de Hu [62], momentos de Zernike [63], transformada Wavelet [64], descriptores HOG [1].

Los descriptores HOG son parte esencial de esta investigación, por tal motivo el apartado 3.1 los exponen detalladamente.

2.2.2.2.3 Características de histograma

Un método de gran importancia para algoritmos de extracción de características como SIFT, LBP, SURF, y en particular para HOG, es el cálculo del histograma.

Este tipo algoritmos de extracción de características se basa en el histograma de una región específica de la imagen. Sea *U* una variable aleatoria que representa un nivel de intensidad de gris en una región dada de la imagen digital, el cual es definido como

$$p_U(x) = P[U = x] = \frac{núm.\,de\ pixeles\ con\ nivel\ gris\ x}{núm.\,total\ de\ pixeles\ en\ la\ región}, \quad x = 0,1,\dots,L-1 \tag{2.16}$$

Las características comunes de $p_U(x)$ son sus momentos, momentos absolutos, la entropía está definida por la ecuación 2.17:

$$H = E[-log_2\, p_U] = -\sum_{x=0}^{L-1} p_U(x) log_2 p_U(x) \tag{2.17}$$

Algunas características típicas de histogramas son la media, varianza, valor cuadrático medio y coeficientes de asimetría y de curtosis, así como también la mediana y la moda. Las características del histograma son útiles para realizar análisis de la forma de objetos desde sus proyecciones, la varianza puede usarse para medir la actividad local en las amplitudes y un histograma estrecho representaría una región con bajo contraste [65].

2.3 Clasificación

Un clasificador es un procedimiento que permite determinar la posible clase a la que pertenece un objeto desconocido, sobre la base de un número determinado de casos de cada una de las clases conocidas, también llamado conjunto de entrenamiento (Figura 2.6).

La clasificación es el siguiente paso una vez que queda definida la extracción de características, la cual permite que cada instancia del conjunto de entrenamiento sea expresada como un vector de medidas. Estas medidas pueden ser definidas por ejemplo, a partir de las intensidades de los pixeles de la imagen digital, sin embargo, son utilizados métodos o procedimientos más complejos que permiten reducir la cantidad de medidas. Además, las medidas o características pueden ser valores de carácter cuantitativo (tipo real o enteros, como el área, intensidad lumínica, etc.), así como también, de tipo cualitativo o categórico (como la forma de los objetos y/o la regularidad de una imagen). Por lo tanto, el tipo de características y el tipo de aplicación en desarrollo, son los factores que definirán la forma en la que se procesen los patrones o características. A continuación se describen algunos de los procedimientos utilizados frecuentemente en la clasificación.

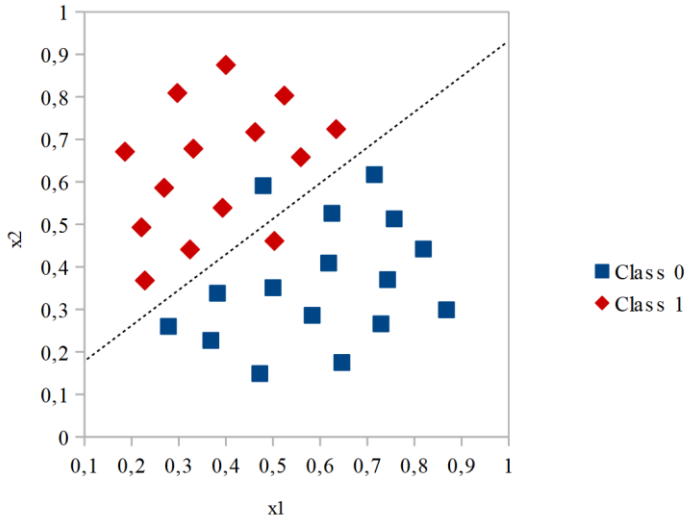

Figura 2.6 Imagen de un clasificador simple de dos clases.

La clasificación estadística o teoría de la decisión, está basada en las propiedades estadísticas de las características como lo puede ser la media, la varianza, promedio, etc. debido a que estudios iniciales en el área de reconocimiento de objetos están basados en estudios estadísticos [66].

Otro método es la clasificación sintáctica o estructural, la cual se basa en buscar relaciones entre las características de los objetos. Por su parte, el enfoque estructural contempla a las características de los objetos como puntos que componen estructuras geométricas, mientras que el enfoque sintáctico ve a las características como elementos de un alfabeto que al combinarse pueden o no ser parte de un lenguaje dado.

El modelar los problemas de forma real es el principal objetivo del enfoque lógico combinatorio [67]. Este enfoque toma en cuenta tanto variables cuantitativas como también variables cualitativas que interfieren en los problemas de clasificación.

Las redes neuronales es otro de los enfoque utilizados en el reconocimiento de objetos, estos sistemas explotan su intento por asemejar el funcionamiento del cerebro humano y logran clasificar a las características o patrones de prueba en el grupo correspondiente, la facultad principal es la de aprender a clasificar. A través del proceso de aprendizaje, las regiones de decisión van tomando forma y se ajustan al problema.

Existen en general dos tipos de casos de clasificación. El primero consiste en que, la clase perteneciente de cada instancia del conjunto de entrenamiento se encuentra disponible para el clasificador. En el segundo caso, no existe o no está disponible esta información y a esto se le conoce como clasificación sin supervisión o "*clustering*".

2.3.1 Clasificación supervisada

Como se mencionó anteriormente, en este tipo de clasificación para cada una de las instancias del conjunto de entrenamiento es conocida su clase perteneciente. El tipo de clasificadores supervisados más utilizado son los clasificadores binarios, los cuales distinguen únicamente entre dos tipos de objetos. Frecuentemente, generados a partir de un determinado número de clasificadores binarios combinados surgen los clasificadores multiclase. Los clasificadores supervisados comúnmente se representan por medio de un modelo genérico, al cual se le conoce como función discriminante.

Algunos ejemplos de clasificadores supervisados son:

- Discriminante lineal de Fisher
- Vecinos más cercanos
- Máquinas de Soporte de Vectores (SVM)
- Adaboost
- Redes neuronales artificiales

2.3.1.1 Clasificación binaria

Sea X un espacio de entrada, Y un espacio de etiquetas y Δ una distribución sobre X y dada una secuencia $S = \{\langle x_i, y_i \rangle\}_{i=1}^{m}$ de ejemplos etiquetados donde cada $x_i \in X$ independientes e idénticamente distribuidos de acuerdo a Δ y cada $y_i \in Y$ es asignado de acuerdo a una regla posiblemente estocástica. En este caso del problema del clasificador binario se restringe a $Y = \{0, 1\}$.

Una regla de clasificación llamada hipótesis, es una función $h: X \mapsto Y$ que asigna una etiqueta a cada elemento en el espacio de entrada. En el problema de clasificación binaria se tiene que $h: X \mapsto [0, 1]$, donde el valor de $h(x)$ es interpretado como la predicción de la etiqueta a ser asignada a la etiqueta x, mientras que la magnitud $|h(x)|$ es interpretada como la confianza de esta predicción. Por otra parte, una clase de hipótesis H es un conjunto compuesto por diferentes hipótesis en el espacio de entrada.

El desempeño de una hipótesis será evaluado utilizando el error de generalización R y el error empírico R_{emp} definidos como se muestra en las siguientes ecuaciones:

$$\mathrm{R}(h) = P_{(x,y)\sim\Delta}\{sgn(h(x)) \neq y\} \quad \textbf{(2.18)}$$

$$R_{emp}(h, S, D) = \sum_{i=1}^{m} D(i) I_{[sgn(h(x))\neq y]} \quad \textbf{(2.19)}$$

donde $D \in \mathbb{R}^m$ es una distribución discreta sobre el conjunto de muestras etiquetadas y $I_{[\cdot]}$ es la función indicadora.

2.3.1.2 Clasificadores Bayesianos

Son clasificadores que basan sus fundamentos a partir de la teoría de la probabilidad y estadística (análisis de varianza, covarianza, dispersión y/o distribución, entre otras). Deciden a que clase pertenece un cierto elemento mediante el cálculo probabilístico de pertenencia de dicho objeto hacia una clase (Figura 2.7).

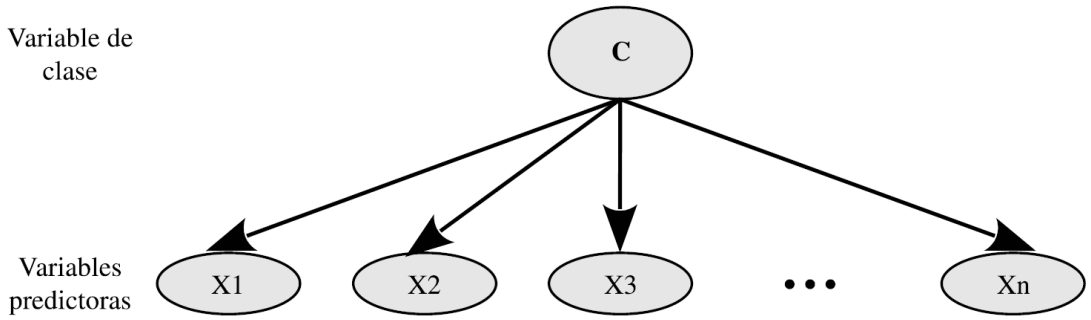

Figura 2.7 Esquema de un modelo Bayesiano [68].

2.3.1.3 Clasificadores lineales

Las funciones discriminantes de un clasificador supervisado son una serie de combinaciones lineales de los elementos del vector de características. Es decir:

$$g(x) = \sum_{i=1}^{d} w_i x_i \qquad \textbf{(2.20)}$$

donde, x es el vector de características, x_i son los elementos del vector de características, w es un vector de peso y d es el número de parámetros el cual es proporcional a la dimensión.

El espacio de características se dice que es N-dimensional debido a que frecuentemente tiene un gran número de dimensiones. Las fronteras de decisión dividen por grupos o regiones al espacio de características, a éstas se les conoce también como hiperplanos, por lo tanto, existirán tantos hiperplanos como número de clases de objetos diferentes (Figura 2.8).

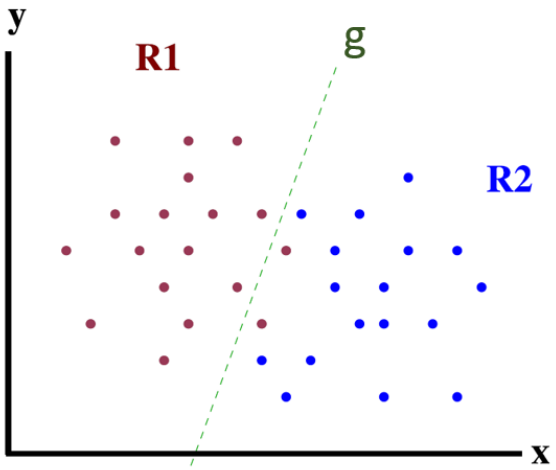

Figura 2.8 Representación de la frontera de decisión (g).

2.3.1.4 Clasificadores no lineales

En este tipo de clasificadores las fronteras de decisión son del tipo cuadrático, por lo tanto, estas pueden ser círculos, elipses, cónicas, parabólicas o hiperbólicas. Su representación matemática en forma matricial se muestra en la ecuación 2.21 donde: x es el vector de características, A es la matriz de pesos de tamaño $d \times d$ y B el vector de pesos de tamaño $1 \times d$.

$$g(x) = x^T Ax + x^T B + C, \;\; C = \omega_{n+1} \tag{2.21}$$

En la Figura 2.9 se muestra un ejemplo de este tipo de clasificadores.

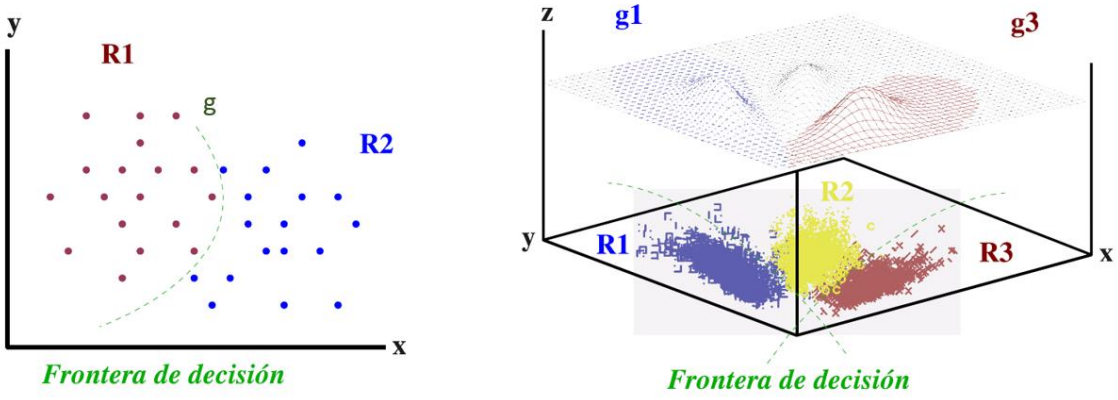

Figura 2.9 Clasificación de espacios de características en dos y tres dimensiones [68].

2.3.1.5 Algoritmo AdaBoost

El algoritmo AdaBoost de "*Adaptive Boosting*" es un algoritmo de aprendizaje que se basa en construir un clasificador "fuerte" a partir de una combinación lineal de clasificadores "débiles", así:

$$f(x) = \sum_{t=1}^{T} \alpha_t h_t(x) \quad \textbf{(2.22)}$$

donde, $h_t(x)$ es clasificador "débil", por lo tanto, el clasificador final o "fuerte" está dado por la ecuación 2.23.

$$H\big(f(x)\big) = sign\big(f(x)\big) \quad \textbf{(2.23)}$$

2.3.2 Clasificación no supervisada

La clasificación no supervisada se conoce también como clasificación sin aprendizaje. Su principal objetivo es identificar las formas que describen los agrupamientos de características. Un algoritmo de agrupamiento (del inglés "*clustering*") es un método de agrupación de una serie de vectores según determinados criterios de cercanía, la cual se define en términos de una determinada función de distancia; consiste en disponer los vectores de entrada de forma que se agrupen de forma más cercana a aquellos vectores que poseen características comunes. El número de *clusters* en los datos depende directamente de la resolución del clasificador. Un clasificador de *clustering* puede ser complicado ya que puede arrojar *clusters* de diferentes formas y tamaños, además de que un análisis no puede ser efectuado ya que la información o agrupación de *clusters* resulta no ser entendible a simple vista.

El cálculo de una medida de disimilitud es necesario para tener un criterio de lejanía, además del criterio de cercanía o similitud. La métrica *Lp* sirve para cuantificar estas medidas de disimilitud, esta es un ejemplo de alguna de las métricas usadas para este propósito. La siguiente definición pertenece a la métrica conocida como distancias de Minkowski [69].

$$d_p(x, y) = \left(\sum_{i=1}^{n} w_i |x_i - y_i|^p \right)^{\frac{1}{p}} \quad \textbf{(2.24)}$$

donde, w_i es el coeficiente de peso y p el tipo de norma. De esta función de distancia se derivan otras conocidas como: la distancia Manhattan con $p = 1$, la distancia euclidiana cuando $p = 2$ y la Chebychev con $p = \infty$. Otro punto importante a recordar es que la métrica se elige con base en el tipo de datos que se manejen y de la semántica que se les asocie.

Algunos de los algoritmos de agrupamiento o *clustering* se mencionan a continuación, debido a su gran extensión y a la cantidad de información que hay acerca de estos no serán descritos, se puede encontrar toda la información necesaria en [70], [71], [72], [73], [74].

- Agrupamiento mediante enfriamiento simulado.
- Método adaptativo.
- Algoritmo de Barchelor y Wilkins.

- Algoritmo K-means.
- Algoritmo GRASP.
- Algoritmo de agrupamiento secuencial.
- Algoritmo ISODATA.
- Métodos basados en grafos.

2.4 Redes Neuronales Artificiales

Para este trabajo de tesis, el enfoque neuronal es de particular interés, por tal motivo en este apartado se describen aspectos generales de las RNAs.

Se iniciara mencionando algunos hechos históricos relevantes de esta área que tienen una relación directa con la RNA tipo perceptrón multicapa y el algoritmo denominado retropropagación del error o propagación del error hacia atrás (EBP, *Error Backpropagation*), que representan la estructura de RNA y algoritmo de aprendizaje utilizados en este trabajo de tesis.

En 1943 Warren McCulloch y Walter Pitts proponen el primer modelo matemático de una neurona artificial [75]. Seis años después, Donald Hebb presenta una regla de aprendizaje, conocida como "Aprendizaje Hebbiano", que permite a las neuronas utilizar el refuerzo para fortalecer las conexiones de entrada más importantes [76]. Poco tiempo después, en 1954, Gabor presenta el "Aprendizaje filtrado", el cual usa el gradiente descendente para minimizar el error cuadrático medio entre las señales de salida actuales y las generadas anteriormente con la finalidad de obtener el peso sináptico óptimo [77].

En el camino a la estructura del MLP, en 1958 Frank Rosenblatt dio un paso importante al introducir un modelo neuronal, denominado perceptrón, formado por una red de unidades binarias de decisión y que actúa como una función que mapea un conjunto de patrones en un conjunto de clases [78]; cada unidad estaba constituida por una neurona basada en el modelo McCulloch-Pitts y un método de aprendizaje.

La RNA ADALINE (***ADA****ptive* ***LI****near* ***NE****uron* o ***ADA****ptive* ***LIN****ear* ***E****lement*) desarrollada por Bernard Widrow y Marcian Edward Hoff es considerada una evolución natural del perceptrón [79]. La diferencia entre estos modelos radica en el algoritmo de aprendizaje, ya que la red ADALINE, y su versión múltiple (MADALINE), utilizan la regla delta, regla Widrow-Hoff o regla del mínimo error cuadrado medio (algoritmo LMS, *Least Mean Square*), la cual supone que la actualización de los pesos sinápticos de la red es proporcional al error que la neurona comete, dicho error es determinado comparando la diferencia entre el valor deseado y la salida lineal obtenida.

En 1963, Novikoff desarrollo de un teorema de convergencia del perceptrón de Rosenblat [80].

Un hecho que redujo significativamente el interés y desarrollo sobre las RNAs se dio cuando en 1969 Marvin Minsky y Seymour Papert publicaron el libro "*Perceptrons: An Introduction to*

Computational Geometry" donde expusieron que el perceptrón solo es capaz de resolver problemas que son linealmente separables y que fallaba en problemas relativamente simples, como el problema XOR, que no cumplen esta característica [81].

Ese mismo año, de acuerdo a diversos investigadores, R. Hecht-Nielsen [82] y a Gori-Tesi [83] entre ellos, tiene sus orígenes el algoritmo EBP cuando Bryson y Ho desarrollaron un algoritmo en el campo de la teoría de control óptimo, muy similar al del EBP, para un control no-lineal adaptativo [84].

Posteriormente en 1971, y en forma independiente, Werbos re-descubre al EBP cuando desarrolló un algoritmo de entrenamiento de retropropagación el cual publicó por primera vez en su tesis doctoral el año 1974 [85].

El trabajo de Werbos no fue ampliamente apreciado hasta que en 1982 Parker re-descubrió la técnica [86] y en 1985 escribió un reporte referente a este trabajo [87] cuando laboraba en el MIT.

Finalmente, en el año 1986 este algoritmo fue formalizado por el grupo PDP (*Parallel Distributed Processing Group*), integrado por Rumelhart, Hinton y Williams, como un método de aprendizaje global para una RNA tipo MLP [88].

Otro trabajo de importancia relacionado con el algoritmo EBP fue el propuesto por Le Cun en 1988 [89], el cual está basado en el trabajo de Bryson y Ho.

Con el tiempo, EBP se ha convertido en uno de los modelos neuronales más utilizados al demostrar ser una poderosa herramienta en aplicaciones de reconocimiento de patrones, modelado dinámico, análisis de sensibilidad, y el control de los sistemas en el tiempo, entre otros.

2.4.1 Modelo biológico

Las neuronas y las conexiones entre ellas llamadas sinapsis son la clave para el procesamiento de información. La neurona elabora una señal de salida a partir de la información recibida en las dendritas, una vez recibida la información, esta es procesada y enviada a través del axón, el cual es el camino de salida de la señal generada por la neurona [90].

Sinapsis son las unidades funcionales y estructurales principales que median entre las interacciones de las neuronas. Los neurotransmisores son unas sustancias químicas que se encuentran en las vesículas de las terminaciones sinápticas, su función es permitir la propagación de las señales electroquímicas de una neurona a otra [91] [92].

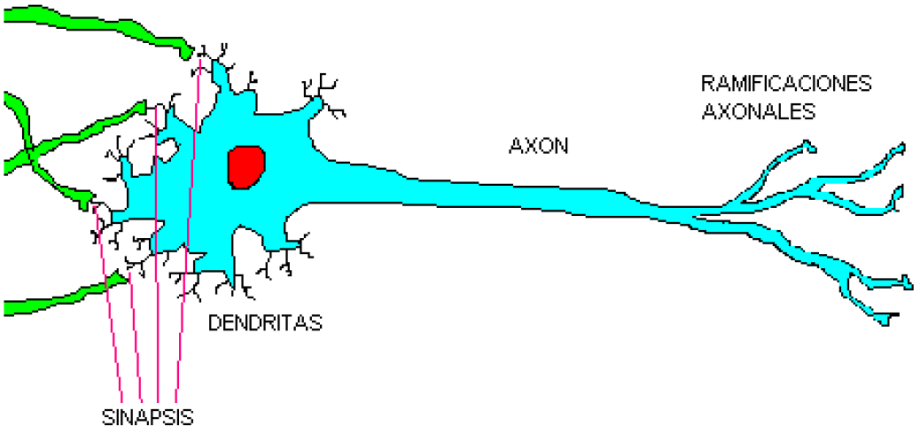

Figura 2.10 Componentes de una neurona [93].

El cerebro consta de billones de neuronas densamente conectadas. Aún en la actualidad no es conocida totalmente la forma en que interactúan las neuronas. Tomando en cuenta la información gráfica de la Figura 2.10, el funcionamiento de las neuronas se lleva a cabo cuando una neurona envía su información de salida por su axón, el axón lleva la señal por medio de ondas de corriente que depende del potencial de la neurona. Otra neurona (o neuronas) recoge las señales a través de sus dendritas mediante el proceso de sinapsis sumando todas las influencias excitadoras. Si dominan las influencias positivas, entonces, la neurona proporciona una señal también positiva la cual es enviada a través de su sinapsis de salida a otras neuronas.

2.4.2 Modelo artificial, conceptos generales

Las RNA son sistemas basados en el comportamiento del cerebro humano o animal. Están conformadas por elementos simples de procesamiento, denominadas neuronas artificiales, que poseen características inspiradas en el conocimiento o supuestos que se tienen acerca del funcionamiento de la neurona biológica, tanto en aspectos morfológicos como fisiológicos, y se basa en modelos matemáticos de su comportamiento.

De acuerdo a Martín y Sanz, se denomina procesador elemental o neurona artificial a un dispositivo simple de cálculo que, basados en modelos matemáticos del comportamiento de una neurona biológica y a partir de un vector de entrada procedente del exterior o de otras neuronas, proporciona una respuesta determinada [94].

La Figura 2.11 muestra los elementos que componen a una neurona artificial.

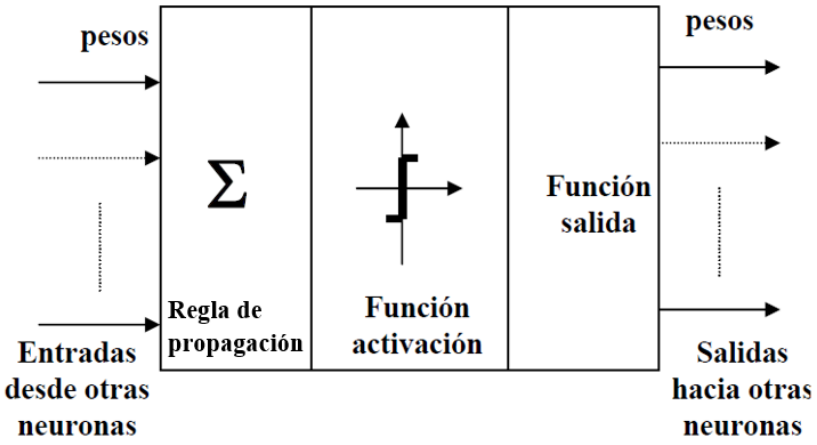

Figura 2.11 Esquema de una neurona artificial.

Un vector de entrada, que forma parte del conjunto de vectores representativo del sistema al que se adaptara la neurona, es canalizado al interior de la neurona artificial a través de sus **entradas**, las cuales están ponderadas por los denominados **pesos sinápticos**. Las entradas ponderadas pasan a la **regla de propagación** donde son sumadas para obtener el **potencial sináptico**. Éste es enviado a la **función de activación** donde se define el estado actual de la neurona. Cuando la **función de activación** es no-lineal, puede existir una **función de salida** que tiene por función umbralizar la salida, la cual es canalizada hacia otra neurona o el exterior a través de la **salida** de la neurona.

Por otro lado, una RNA consiste en un conjunto de neuronas artificiales interconectadas de una forma concreta. El interés del estudio de las RNA no se basa solamente en el modelo de la neurona artificial sino en cómo se interconecta con otras neuronas. La organización de las neuronas típicamente es por grupos llamados capas. Generalmente una red consiste en una secuencia de capas con conexiones entre capas adyacentes consecutivas [93], [90], [92].

En este sentido, existe una reciprocidad por parte de personalidades en el área de las RNA como Hecht-Nielsen, Caudill, Butler, Feldman, Ballard, Rumelhart, Hinton, Williams y Kohonen, en considerar a una RNA como un sistema de procesamiento de información que consta de un gran número de unidades simples de procesamiento, altamente interconectadas y jerarquizadas en capas o niveles, capaz de adaptarse a diversas aplicaciones para responder dinámicamente a estímulos externos [82], [95], [96], [88], [97].

Las capas de una RNA pueden ser categorizadas en capa de entrada, capa oculta y capa de salida (ver Figura 2.12).

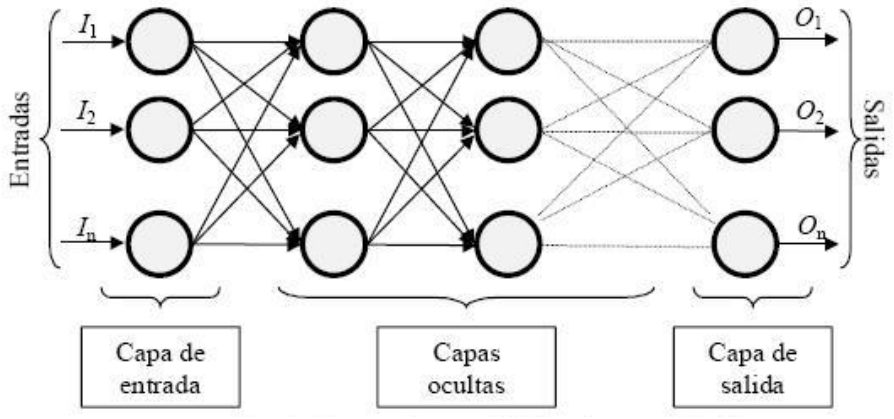

Figura 2.12 Esquema de una RNA totalmente conectada.

Una RNA está caracterizada por su:

- **Arquitectura.** Se denomina arquitectura de una RNA a la topología, estructura o patrón de conexiones que guardan las neuronas que la integran. Es decir, la arquitectura define la organización y forma en que están conectadas las neuronas que forman a la RNA y depende de los requerimientos de la aplicación para la que es diseñada.
- **Dinámica de cómputo.** Se encarga de determinar el valor que toman las unidades de proceso, con base en la **función de activación (o de transferencia)** de la neurona, las cuales especifican como se transforman las señales de entrada de la unidad de proceso en la señal de salida.
- **Algoritmo de entrenamiento o mecanismo de aprendizaje.** Como ya se mencionó una de las principales características de una RNA es su capacidad para aprender interactuando con su entorno o con alguna fuente de información. Esta función es cumplida por su mecanismo de aprendizaje o algoritmo de entrenamiento, el cual es un proceso adaptativo que modifica los pesos sinápticos de la red buscando mejorar su comportamiento y/o adaptarlo a una aplicación específica.

Finalmente, las siguientes características pueden ser consideradas como las principales de una RNA:

- **Aprender**. Las RNA pueden adquirir el conocimiento de un objeto por medio del estudio y la experiencia. Tienen la capacidad de cambiar su comportamiento con base en su entorno. Si les es presentado un conjunto de entradas, la red se puede ajustar para producir salidas consistentes.
- **Generalizar**. Las RNA son capaces de proporcionar respuestas o salidas correctas a entradas que presentan pequeñas variaciones, como por ejemplo ruido o distorsión, con respecto a las utilizadas durante su aprendizaje.
- **Abstraer**. Algunas RNA son capaces de abstraer los patrones esenciales de un conjunto de entradas que en apariencia no presentan aspectos comunes.

Al igual que los descriptores HOG, una RNA tipo MLP y el algoritmo de entrenamiento EBP son parte esencial de este trabajo de tesis, por este motivo el apartado 3.2 expone sus fundamentos teóricos y matemáticos.

2.5 Estado del arte

La inteligencia artificial es un tema muy interesante para el ser humano, desde el comienzo, la idea de poder crear un ente capaz de desarrollar las mismas funciones que las de él ha sido su primordial objetivo. El área de visión por computadora juega un papel muy importante en este sentido, ya que, la idea principal es la de crear un sistema capaz de entender su entorno como lo hacemos los seres vivos, por lo tanto, es de suma importancia crear sistemas capaces de procesar información visual lo más precisa posible.

A lo largo de muchas décadas se ha hecho uso de diferentes teorías para el reconocimiento de patrones por los sistemas de cómputo, comenzando con la intención de reconocer caracteres alfanuméricos, siguiendo con el intento por determinar las figuras básicas en una imagen hasta el reconocimiento de personas, pasando antes por la intención de detectar o reconocer cualquier tipo de objetos. A continuación se describen brevemente las investigaciones más afines a la detección de objetos mediante el algoritmo de histogramas de gradientes orientados, comenzando con un breve esbozo en los primeros trabajos de reconocimiento de objetos.

El tema de reconocimiento de objetos surgió debido a una interrogante, ¿es posible que una computadora logre representar los objetos dentro de una fotografía en un modelo tridimensional? La investigación realizada por Lawrence G. Roberts en 1963 [98], cuyo objetivo era el de poder transformar una imagen en dos dimensiones a una representación tridimensional de la misma, se basaba en el supuesto de que, una fotografía era el resultado de una proyección de un conjunto de objetos construidos con base en modelos tridimensionales. Roberts logró desarrollar un sistema que podía procesar una imagen (fotografía), convirtiéndola en un dibujo lineal, con esto y con el supuesto de que la información de los bordes de los objetos dentro de una imagen podrían reconstruir un modelo tridimensional con base en su topología, pudo reconstruir polígonos básicos y mostrarlos desde diferentes puntos de vista.

Con base en las investigaciones realizadas por Roberts surgieron nuevas propuestas, los más representativos fueron de entre todos, un enfoque de representación de formas como cilindros generalizados el cual es presentado en 1971 por Binford [99], mientras que David Lowe mostraba técnicas de emparejamiento geométrico en 1985 [100]. Así mismo, se desarrolló por ejemplo un método que consiste en alinear un objeto contenido en la base de datos con los elementos de la imagen, esto se realizaba mediante la etiquetación de solo dos puntos o tres puntos principales (para objetos bidimensionales o tridimensionales respectivamente), después de alineado el primer punto se realiza un rotación con el fin de empatar el segundo punto clave, una vez alineados estos dos puntos descriptivos se realiza un escalamiento del modelo base con la finalidad de empatar

ambas figuras, el resultado era el reconocimiento de diferentes objetos con diferentes formas, escalas y perspectivas, esta investigación tuvo lugar a finales de los 80 [101].

Hasta el momento se ha generalizado en cuanto al reconocimiento de objetos, este trabajo centra su estudio en el tema de reconocimiento de objetos con base en el histograma de gradientes orientados, por lo tanto, a continuación se describirán los trabajos referentes a este tema.

Los histogramas de orientación de bordes fueron presentados por Freeman y Roth en 1995 [102], su objetivo principal era reconocer gestos con las manos, empleando histogramas de orientación local como su vector de características. Ofrece una alta velocidad de cómputo y robustez a los cambios de iluminación en las escenas. Su vocabulario, es decir el número de gestos posiblemente reconocibles, era de cerca de 10 tipos diferentes de gestos. Como clasificador usaron la patente desarrollada por Robert K. McConnell publicada en el año de 1986 [103]. Ésta realizaba una transformación que convertía a la imagen en el vector de características, y posteriormente comparaba este vector con los vectores del conjunto de gestos de entrenamiento, utilizando la métrica de distancia Euclidiana.

Para el año 2001, S. Belongie *et al.* presentaron un enfoque novedoso para medir similitudes entre formas de objetos [104], con el objetivo de reconocer una variedad de objetos. Para esto introdujeron el término de contexto de formas (*shape context)* el cual, a partir de un punto de referencia captura la distribución de los puntos restantes relativos a este, ofreciendo con esto una caracterización discriminativa global. La premisa de que, puntos correspondientes a dos objetos similares tendrán el mismo contexto de forma les permitía resolver el problema de asignación de una forma óptima. Como clasificador utilizaron el enfoque del vecino más cercano. Así, lograron demostrar un rendimiento excelente en una amplia variedad de conjunto de datos, como siluetas, marcas, dígitos manuscritos y el conjunto de datos COIL [105].

En [10] David Lowe presenta un método para extraer características distintivas invariantes, mejor conocido como descriptores SIFT. Estos son usados para realizar búsquedas o emparejamientos entre diferentes vistas o puntos de vista de objetos. Este tipo de características son invariantes a rotaciones y/o escalamientos de la imagen, además, proveen un robusto emparejamiento frente a distorsiones, adición de ruido, cambio en la iluminación y/o cambio del punto de vista. El procedimiento de reconocimiento de objetos con este método consiste en: realizar una búsqueda o emparejamiento de características individuales dentro de una base de datos que contiene a las características de objetos ya conocidos, usando el algoritmo del vecino más cercano, seguido por la aplicación de la transformada Hough [61] para identificar los *clusters* que pertenecen a un único objeto, y por último una verificación por el método de mínimos cuadrados. La propiedad distintiva de los descriptores SIFT es lograda al ensamblar un vector de altas dimensiones, el cual está formado por el cálculo de gradientes de regiones locales de la imagen. En general este método utiliza ciertas regiones de un objeto para compararlas con su base de datos, estas regiones

son procesadas con el método de obtención de bordes y de esta forma obtiene los vectores característicos.

Navneet Dalal y Bill Triggs presentaron el enfoque de los histogramas de gradientes orientados (HOG) en el año de 2005 [1]. Luego de estudiar la variedad de métodos que existían acerca del reconocimiento de objetos, incluyendo sobre todo los recientemente mencionados, lograron demostrar experimentalmente que el conjunto de cuadriculas de descriptores de histogramas de gradientes orientados lograban superar los resultados de los conjuntos de características o métodos de reconocimiento existentes. Su objetivo fue el de reconocer personas, utilizando como base el conjunto de datos de peatones del MIT, pero que modificaron agregando una mayor cantidad de imágenes en las cuales los peatones o personas se mostraban en diferentes poses, ya que, la base de datos original no contemplaba esta variación en su conjunto. En resumen la idea principal o característica de este método consiste en que, la apariencia y forma de un objeto puede ser caracterizada de forma correcta a través de una distribución de los gradientes de intensidad locales o dirección de bordes (mencionados anteriormente). Introdujeron el término "*celda*" el cual consiste en dividir una ventana en pequeñas regiones, en donde se acumula el histograma de las direcciones del gradiente, éste consiste en un vector unidimensional. El gradiente de una imagen es el equivalente a la aplicación de un algoritmo de detección de bordes. Posteriormente, un conjunto o combinación de celdas es normalizado con el fin de mejorar la invarianza a la iluminación, a este conjunto o combinación de celdas normalizadas le denominaron "*bloque*". Entonces, al conjunto de bloques normalizados los llamaron descriptores HOG, cabe señalar que estos densos conjuntos de descriptores además se encuentran traslapados. Finalmente, el conjunto o rejilla de descriptores HOG queda representado como un vector de descriptores, el cual es procesado dentro de una máquina de vectores soporte (SVM) [106], la cual fue utilizada como su clasificador. En su trabajo dividieron la ventana de detección en celdas de tamaño de 8×8 pixeles y cada grupo de 2×2 celdas en la integración de los bloques. Cada celda consistía en el histograma de gradientes orientados en 9 direcciones (*bins*) y por lo tanto, cada bloque era la concatenación de los histogramas de cada celda. Definieron sus ventanas de detección de un tamaño de 64×128 pixeles, y tomando en cuenta un traslape entre bloques de 8 pixeles, obtuvieron 7×15 bloques, dando un total de 3780 características por ventana de detección. Todas estas características fueron finalmente usadas para entrenar el clasificador lineal SVM.

Una año después, Qiang Zhu *et al*. propusieron su algoritmo de detección rápida [11], en el cual integraron los enfoques de histograma de gradientes orientados junto con el de cascada de exclusores ("*cascade-of-rejectors*") con la finalidad de lograr un sistema rápido y preciso. Propusieron varias modificaciones al algoritmo HOG, la primera consistía en utilizar bloques de múltiples escalas, es decir, Dalal y Triggs [1] utilizaban únicamente bloques de tamaño 16×16 pixeles al concatenar celdas de 8×8 pixeles, para el equipo de Qian Zhu esto representaba una limitante y creían que tomando múltiples tamaños de bloques lograrían mejores resultados. Discretizaron la orientación y magnitud de cada pixel en histogramas de 9 *bins* pero, calcularon

y almacenaron la imagen integral [107] para cada bin, obteniendo un total de 9 imágenes, las cuales usaron para calcular rápidamente cualquier región rectangular de la imagen. Para acelerar el proceso de detección usaron una cascada de exclusores, utilizando el algoritmo AdaBoost [107] para escoger cuáles características serían evaluadas en cada etapa. Finalmente lograron desarrollar un detector de personas con un procesamiento de hasta 70 veces más rápido que el algoritmo HOG original, utilizando imágenes de 320×280 pixeles alcanzaron velocidades desde 5 hasta 30 cuadros por segundo dependiendo de la densidad del escaneo de la imagen.

Li Zhang *et al.* presentaron en la CVPR07 [108] un estudio en el cual comparaban las características Edgelet [109] y las características HOG [1], y también dos clasificadores diferentes, algoritmos de cascada AdaBoost [107] y SVM [106]. Esta comparativa la realizaron con base en imágenes infrarrojas, las cuales son comúnmente utilizadas en sistemas de vigilancia en los cuales no siempre está disponible la iluminación externa. Su enfoque consistía en tres etapas: extracción de características locales, selección de características y clasificación de objetos. Una diferencia entre HOG y Edgelets es que, ésta última puede contener un conjunto de características del orden de los millones, es por esto que la selección de características era parte fundamental de su método, aunque, gracias a esto notaron que existía cierta similitud entre algunas características de dichos algoritmos. Tomaron en cuenta que en el caso del algoritmo AdaBoost, el proceso de selección de características se encuentra intrínsecamente combinado con el aprendizaje del clasificador, mientras que para el algoritmo SVM, la selección de características es un proceso que se debe realizar explícitamente. Al finalizar sus experimentos, principalmente pudieron concluir que, lograron obtener una precisión de detección sobre imágenes infrarrojas comparable con las imágenes con representación en el espacio visual y que ambos algoritmos de extracción de características comparten propiedades comunes, principalmente porque ambos enfoques se encargan de capturar las siluetas o bordes de las imágenes.

En [110] se realizó una investigación en la cual se empleaba la extracción de características HOG desde todas las localidades de la rejilla en la imagen como candidatos del vector de características. El siguiente paso fue aplicar el algoritmo de análisis de componentes principales (*Principal Components Analysis, PCA*) el cual se describe en [111], este algoritmo tiene como finalidad reducir el número de dimensiones de los vectores de características, ya que esta técnica es muy conocida para realizar este proceso. Posteriormente un subconjunto apropiado de características PCA-HOG es seleccionado utilizando los algoritmos *Stepwise Forward Selection* (SFS) y *Stepwise Backward Selection* (SBS). SFS comienza con un subconjunto vacío y repetidamente agrega el vector de características en términos del bienestar del subconjunto. Por su parte, SBS inicia conjunto con todos los vectores de características incluidos y repetidamente remueve los vectores de características innecesarios por el bien del subconjunto. Cabe señalar que una región de una imagen que contenga todos sus pixeles en negro por ejemplo, sería un vector de características que podría o no ser incluido, según sea el caso. Como resultado final,

lograron reducir el número de características a menos de la mitad sin disminuir el rendimiento del algoritmo.

Particularmente, en sistemas de redes de cámaras distribuidas existe el problema de transmisión y almacenamiento de datos, esto se debe a que los sistemas de reconocimiento frecuentemente son realizados con un alto número de características, es por eso que Vijay Chandrasekhar *et al.* desarrollaron en el 2009 [112] un sistema de reconocimiento con descripción de características compreso CHOG. Propusieron un enfoque de cómputo de características de bajo rango de bits, el cual les permitió reducir hasta 20 veces la tasa de bits de transmisión. Usaron técnicas de codificación de árbol, específicamente utilizaron el algoritmo *Tree Fixed Length Coding* como compresor. Como clasificador utilizaron el algoritmo del vecino más cercano. Lograron mostrar cómo el cálculo eficiente de las distancias entre características en su representación compresa, elimina la necesidad de descomprimir la información.

En [12] y [13] se muestran detectores de personas con base en el algoritmo HOG y el algoritmo LBP como extractores de características. Como se ha mencionado hasta el momento, el algoritmo HOG puede realizar una representación efectiva de la apariencia y forma de un objeto, sin embargo, una única característica posiblemente no es lo suficientemente expresiva en entornos con mucho ruido en los bordes. Complementando el proceso de extracción de características con el modelo LBP atacaron este problema. El algoritmo LBP resulta ser un excelente descriptor de texturas debido a su invarianza de escala de grises y rotaciones. Ambos enfoques tuvieron excelentes resultados sobre la base de datos INRIA (generado por [1]), aproximadamente un 10% más precisos que el algoritmo HOG original. Como clasificadores utilizaron SVM y AdaBoost respectivamente.

Y. Socarrás *et al.* en [113] implementaron el algoritmo HOG añadiendo información de alto nivel proveniente de la segmentación de la misma imagen. Su idea principal fue re-ponderar el vector de características sin aumentar su tamaño. Los beneficios de la puesta en marcha de esta propuesta fueron mejorar el rendimiento del detector al enriquecer los vectores de características y por otra parte, tomar cierta ventaja de la información que proporciona el proceso de segmentación de imágenes. Usaron el método de segmentación "Mean-Shift" [114] y probaron diferentes métodos de re-ponderación, basados en color, luminancia por región o en texturas como el algoritmo LBP. Lograron un rendimiento del 4.47% en la tasa de detección comparado con el método original [1].

Capítulo 3

Algoritmo HOG, RNA-MLP y algoritmo EBP

Este trabajo de tesis propone un modelo de reconocimiento de objetos utilizando el algoritmo HOG en la etapa de extracción de características y el MLP, correspondiente al enfoque neuronal, como base en la construcción de la etapa de clasificación del sistema. Es por eso que, este capítulo versa sobre los fundamentos teóricos y matemáticos de los algoritmos HOG y MLP.

3.1 Algoritmo HOG

En el año 2005, N. Dalal y B. Triggs introdujeron el concepto de Histograma de Gradientes Orientados [1]. En su trabajo propusieron un nuevo algoritmo para la detección de personas basado en redes de histogramas de gradientes orientados como descriptores de características; los resultados experimentales demostraron que esta propuesta exhibe un desempeño que supera a los métodos existentes para la detección de personas. Además, en esta propuesta se muestra también el uso del algoritmo HOG como extractor de características para el sistema de reconocimiento de objetos.

Este método está basado en la evaluación de normalizaciones locales de los histogramas de gradientes orientados de una imagen en una densa cuadricula. Trabajos relacionados que han hecho uso similar de esta descripción de características se pueden encontrar en [102], [10], [115], [116]. El esquema del extractor de características basado en el algoritmo HOG se puede observar en la imagen de la Figura 3.1.

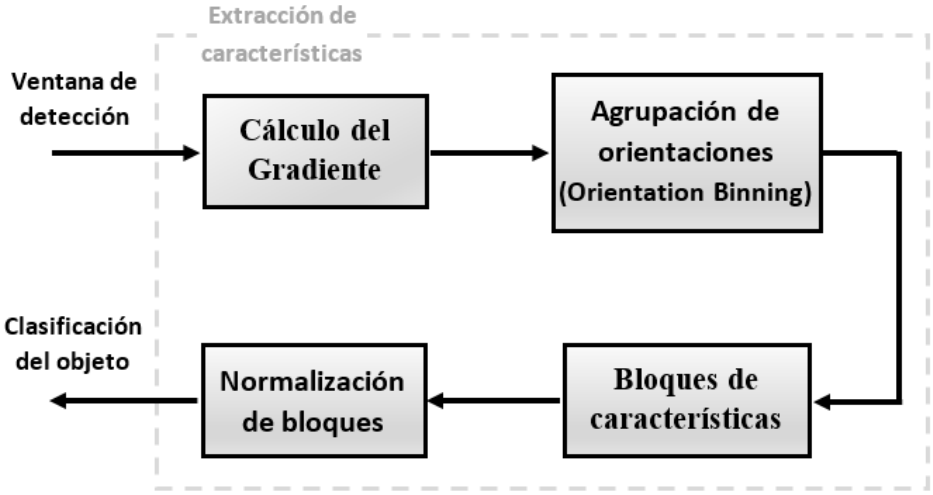

Figura 3.1 Esquema del proceso de extracción de características HOG.

En la Figura 2.1 se muestra el diagrama general de un sistema de reconocimiento de objetos, en el cual se incluyen los bloques de preprocesamiento de la imagen seguido por el bloque de segmentación, si bien el preprocesamiento de imágenes tiene en muchas ocasiones un impacto visual en las imágenes en las que se aplique, de acuerdo con Dalal y Triggs, para el algoritmo HOG este paso o proceso no es significativo en los resultados finales y por tanto puede ser omitido. La razón de lo mencionado es que la normalización de los bloques esencialmente cumple esta función. A continuación se describe cada bloque del algoritmo HOG.

3.1.1 Cálculo del gradiente

Frecuentemente, los bordes de los objetos o regiones en una imagen aparecen como un ligero salto de brillo y en otras ocasiones a lo largo de muchos pixeles. En imágenes que presentan ruido (comúnmente puntos con variaciones de brillo debido, en muchas ocasiones, a características del sensor de captura) algunos filtros mostrarán a los puntos mucho más intensos que los bordes de las áreas de interés, como por ejemplo filtros paso altas.

Dalal y Triggs [1] indican que el rendimiento de su sistema es sensible a la forma en que se calculan los gradientes, pero, según las pruebas realizadas notaron que los esquemas más simples para el cálculo del gradiente fueron los que tuvieron los mejores resultados. Para el cálculo del gradiente se pueden utilizar filtros Gaussianos seguidos por operadores derivativos de primer

orden. Operadores como el de Roberts [56], Sobel [57] o Prewitt [58] son los más populares, sin embargo, los resultados que obtuvieron demostraron que un operador derivador simple unidimensional ($[-1, 0, 1]$) y sin aplicar ningún filtro Gaussiano ($\sigma = 0$) resulto ser la combinación que genero los mejores resultados. Tomando en cuenta la teoría de detección de bordes del Capítulo 2 y que el sistema gira en torno a un procesamiento sobre imágenes en escala de grises, para el cálculo del operador derivador se utiliza una máscara o plantilla de tamaño 3×3 como la mostrada en la Figura 3.2.

0	$\boldsymbol{I_{1,2}}$	**0**
$\boldsymbol{I_{2,1}}$	**0**	$\boldsymbol{I_{2,3}}$
0	$\boldsymbol{I_{3,2}}$	**0**

Figura 3.2 Máscara de operador derivativo para cálculo de gradiente.

De la ecuación 2.14, se tiene que:

$$\frac{\partial f}{\partial x} = \left(I_{2,3}\right) - \left(I_{2,1}\right) \quad \textbf{(3.1a)}$$

$$\frac{\partial f}{\partial y} = \left(I_{3,2}\right) - \left(I_{1,2}\right) \quad \textbf{(3.1b)}$$

Entonces, se definen los gradientes direccionales de la siguiente forma [117]:

$$Gx_{x,y} = \frac{\partial f}{\partial x} = I(x+1, y) - I(x-1, y) \quad \textbf{(3.2a)}$$

$$Gy_{x,y} = \frac{\partial f}{\partial y} = I(x, y+1) - I(x, y-1) \quad \textbf{(3.2b)}$$

Así, se define la magnitud del gradiente $G_{x,y}$ como:

$$G_{x,y} = \sqrt{{Gx_{x,y}}^2 + {Gy_{x,y}}^2} \quad \textbf{(3.3)}$$

Y, la dirección del ángulo del gradiente respecto al eje x está dado por:

$$\theta = \tan^{-1}\left(\frac{G_y}{G_x}\right) \quad \textbf{(3.4)}$$

Supóngase que se tiene una imagen de tamaño $N \times M$, el gradiente de la imagen se calcula para todos y cada uno de los pixeles de la imagen, obteniendo de esta forma dos vectores de tamaño

$N \times M$, el primero se utiliza para almacenar el gradiente y el segundo se utiliza para almacenar el ángulo de éste. Para imágenes a color, se calcula el gradiente de forma separada para cada canal de color, y posteriormente se toma el que tenga la norma más grande como vector gradiente.

3.1.2 Agrupación de orientaciones

Un **histograma** es una representación gráfica de una variable en forma de un diagrama de barras (comúnmente), se utilizan frecuentemente para variables continuas o discretas que poseen un gran número de datos, y que además, se han agrupado en grupos, clases, características o **contenedores**. Un **voto**, frecuencia o muestra es una cantidad que se ubica en un determinado sub-rango de valores de una característica o clase. A partir de este momento se hará referencia a estos conceptos.

Este paso se considera como el fundamento de la no-linearidad del sistema. Cada pixel procesa un voto ponderado para el histograma de orientación de borde basado en la orientación del gradiente de cada elemento, cada voto se acumula en los contenedores (*bins*) de la región espacial local donde fue calculada, a esto se le llama "*celda*" (Figura 3.3a). Las celdas pueden ser rectangulares o radiales. Los contenedores se espacian uniformemente según el tipo del gradiente, sin signo de $0° – 180°$, con signo de $0° – 360°$. Los votos son interpolados bilinealmente entre los centros de contenedores vecinos, tanto en orientación como en posición, esto con el fin de reducir el "*aliasing*" entre votos. El voto puede ser una función de la magnitud del gradiente, puede ser tanto la magnitud propia, como también su valor cuadrático, su raíz cuadrada o una forma recortada de la magnitud la cual podría representar la presencia/ausencia de un borde en ese pixel.

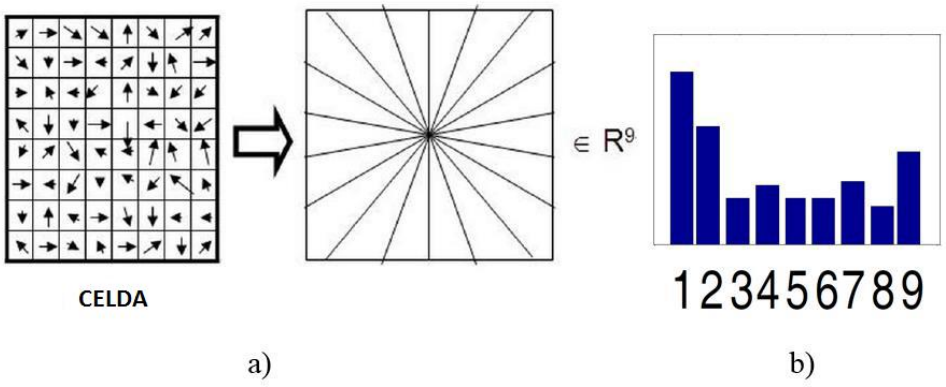

Figura 3.3 Representación de la agrupación de orientaciones de una celda. a) Celda de tamaño 8×8 con orientaciones y sus contenedores. b) Representación del vector de votos ponderados.

Para un buen rendimiento se necesita una excelente codificación de la orientación del gradiente. Los contenedores pueden cubrir un amplio rango de orientaciones (véase Figura 3.3b), sin embargo, el mismo valor puede no ser efectivo para el reconocimiento de todos los objetos, por

ejemplo, Dalal y Triggs [1] determinaron que para el caso de detección de personas, 9 *bins* eran adecuados para esta tarea.

En la Figura 3.3 se puede apreciar de una forma más clara la representación de los términos usados en esta sección, modificada de [1].

3.1.3 Bloques de características

Este paso de la extracción de características consiste en ordenar la información obtenida en los pasos anteriores; la forma en que se ordenan los histogramas de gradientes consiste en agrupar celdas para formar un "bloque". Cada ventana de detección, por lo tanto, está divida en una cantidad específica de bloques. Para el algoritmo HOG se consideran dos clases de geometrías: cuadradas o rectangulares, que particionan en cuadriculas las celdas, y las circulares, que particionan las celdas en forma logarítmica-polar. A estas clases se le denominan **R-HOG** y **C-HOG** para características HOG rectangulares y circulares respectivamente. Una característica importante de este paso consiste en que, todas las celdas de una ventana de detección están traslapadas por cierto factor de traslape de pixeles. Con esto se logra que cada celda aporte mayor información al conjunto en general.

Los bloques **R-HOG** se utilizan de una forma similar que los descriptores SIFT [10] pero con ciertas diferencias. Este tipo de bloques pueden ser descritos con sólo dos parámetros: largo y ancho. Dalal y Triggs indican que para el caso de detección de personas los mejores resultados los obtuvieron al utilizar bloques de tamaño 3×3 y 2×2 celdas, con celdas de tamaño 6×6 y 8×8 pixeles. Además de lo anterior, es útil realizar un adelgazamiento de los pixeles cercanos a los bordes de los bloques, aplicando para esto una ventana espacial Gaussiana para cada pixel antes de acumular los votos de orientación en cada celda, con $\sigma = 0.5 * ancho\_bloque$.

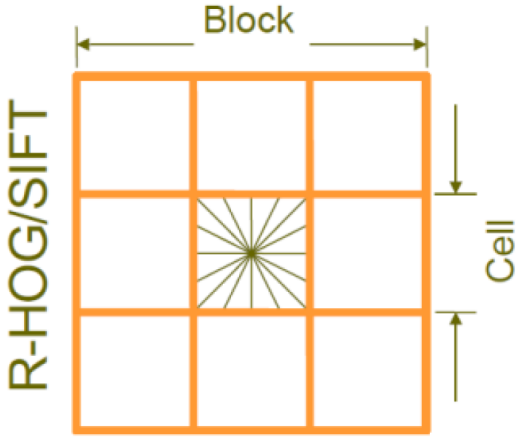

Figura 3.4 Representación de un bloque rectangular (**R-HOG**) [1].

Los bloques del tipo **C-HOG** están basados en los contextos de forma (*Shape Context*) definidos en [104]. Estos bloques tienen dos variaciones: con una sola celda central y con una celda central

dividida angularmente. Los bloques circulares pueden ser descritos por cuatro parámetros: el número de contenedores angulares y radiales, el radio del contenedor central, y el factor de expansión para el radio de los contenedores radiales adicionales. Dalal y Triggs indican que ambas variantes proporcionan el mismo rendimiento, dos contenedores radiales con cuatro contenedores angulares, un radio central de 4 pixeles, y un factor de expansión de valor 2 proporcionaron los mejores resultados para su sistema.

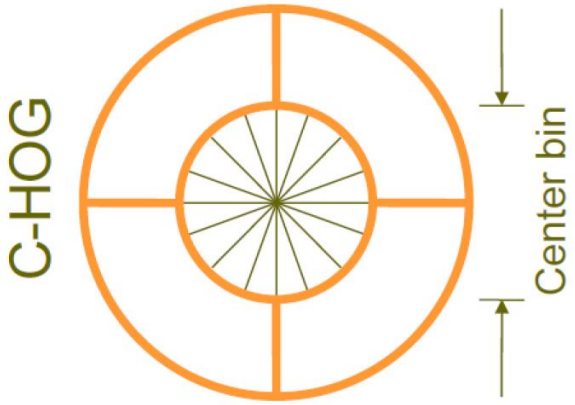

Figura 3.5 Representación de un bloque circular (**C-HOG**) [1].

Finalmente, la concatenación de todos los vectores de las celdas correspondientes al bloque (**R-HOG** o **C-HOG**) conforma el vector de características no normalizadas HOG, éste es transferido a la siguiente etapa.

3.1.4 Normalización de bloques

Las intensidades de los gradientes varían en un amplio rango debido a las variaciones de la intensidad o contraste del fondo de la imagen digital, para obtener un buen rendimiento es necesario realizar una buena normalización local.

Existen cuatro esquemas de normalización que pueden ser aplicadas en los bloques del algoritmo HOG para ambos tipos de geometría. Sea *v* el vector de características no normalizado, $\|v\|_k$ su *k*-norma para $k = 1,2$, y sea ε una constante pequeña. Los esquemas de normalización son los siguientes:

a) L2-norm:

$$v \rightarrow \frac{v}{\sqrt{\|v\|_2^2 + \varepsilon^2}} \tag{3.5}$$

b) *L2-Hys*: L2-norm, seguido por un recorte (limitar los valores de v a 0.2) y por último, re-normalizar (L2-norm).

c) L1-norm:

$$v \rightarrow \frac{v}{\|v\|_1 + \varepsilon} \tag{3.6}$$

d) L1-sqrt,

$$v \rightarrow \sqrt{\frac{v}{\|v\|_1 + \varepsilon}} \tag{3.7}$$

Dalal y Triggs indican que L2-Hys, L2-norm y L1-sqrt tienen el mismo rendimiento en su sistema, mientras que L1-norm lo reduce cerca del 27%. También indican que los resultados son insensibles al valor de ε en un amplio rango de valores. Finalmente, el vector normalizado de características corresponde al vector de características HOG, el cual se procesara para su clasificación, tanto en fase de aprendizaje como en la fase de detección.

3.2 RNA MLP y algoritmo EBP

Una de las principales características de las RNAs consiste en la capacidad de aprender a partir de una fuente de información que interactúa con el sistema. En este capítulo se describe el uso del MLP el cual es un tipo de RNA formado por múltiples capas. Éste es uno de los tipos de redes más comunes. El algoritmo de entrenamiento más utilizado para esta red neuronal es el conocido como algoritmo EBP y en conjunto tienen la capacidad de funcionar como clasificador de datos.

3.2.1 Perceptron Simple

Este modelo de RNA conocido como perceptrón o perceptrón Simple fue presentado por Rossenblatt en 1958 [78, 118] basado en el modelo de McCulloch y Pitts [75] y en una regla de aprendizaje basada en la corrección del error. Está basada en las primeras fases de procesamiento de los sistemas sensoriales de los animales. Una de las características principales de este modelo de RNA es la capacidad que tiene de aprender patrones. Esta RNA es unidimensional por naturaleza, está constituida por un conjunto de sensores de entrada que reciben la información a reconocer o clasificar y una neurona de salida la cual tiene la tarea de clasificar la información de entrada en dos clases con valores binarios (0 desactivada, 1 activada).

Supóngase que se tiene una función f de R^n en $\{-1,1\}$, la cual aplica un patrón de entrada $\mathbf{x} = (x_1, x_2, \dots, x_n)^T \in R^n$ en la salida deseada $z \in \{-1,1\}$, es decir, $f(x) = z$. La información que se dispone sobre esta función está dada por p pares de patrones de entrenamiento: $\{\mathbf{x}^1, z^1\}, \{\mathbf{x}^2, z^2\}, \dots, \{\mathbf{x}^p, z^p\}$, donde $\mathbf{x}^i \in R^n$ y $f(\mathbf{x}^i) = z^i \in \{-1,1\}$, $i = 1,2, \dots, p$. Dicha función realiza una separación en el espacio R^n de patrones de entrada. En una sección se encontraran los patrones con salida -1 y en otra los patrones con salida 1. Por lo tanto, la función f clasifica a los patrones de entrada en dos tipos de clases.

Una **unidad de proceso bipolar** es una función matemática que tiene como dominio el conjunto *n*-dimensional $\{-1,1\}^n$ y de rango el conjunto $\{-1,1\}$, definida como sigue:

$$f(x_1, x_2, \dots, x_n) = \begin{cases} 1 & si \quad w_1x_1 + w_2x_2 + \cdots + w_nx_n \geq \theta \\ -1 & si \quad w_1x_1 + w_2x_2 + \cdots + w_nx_n < \theta \end{cases} \quad \textbf{(3.8)}$$

donde los parámetros $w_1, w_2, \dots, w_n$, son conocidos como **pesos sinápticos** y estos son los pesos con los que se realiza la ponderación de los valores de entrada $x_1, x_2, \dots, x_n$, entonces, a la suma ponderada $u = w_1x_1 + w_2x_2 + \cdots + w_nx_n$ se le llama **potencial sináptico** y al parámetro θ se le conoce como **umbral**.

Análogamente, se puede definir una **unidad de proceso binaria** como una función matemática que tiene como dominio el conjunto *n*-dimensional $\{0,1\}^n$ y como rango el conjunto $\{0,1\}$, la cual está definida por la ecuación 3.9. En este caso, cuando la salida de la unidad de proceso es igual a 1 se dice que la unidad esta activada, en cambio, cuando la salida de la unidad es igual a 0 se dice que se encuentra desactivada.

$$f(x_1, x_2, \dots, x_n) = \begin{cases} 1 & si \quad w_1x_1 + w_2x_2 + \cdots + w_nx_n \geq \theta \\ 0 & si \quad w_1x_1 + w_2x_2 + \cdots + w_nx_n < \theta \end{cases} \quad \textbf{(3.9)}$$

A la función f se le conoce como la función de transferencia o activación, y en estos casos particulares se les conoce como función signo y función escalón, paso o de Heaviside, respectivamente. La representación gráfica de estas funciones se muestra en la Figura 3.6.

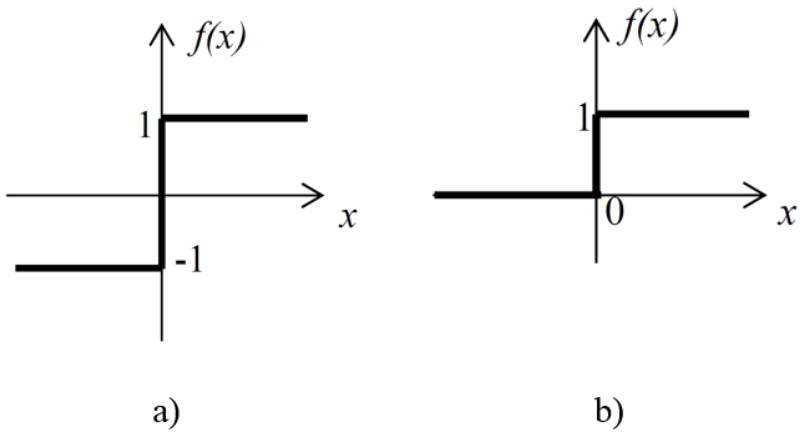

Figura 3.6 Funciones de activación a) Función signo. b) Función escalón.

Una unidad de proceso, neurona artificial, se muestra en la Figura 3.7, en la cual se indican también componentes a forma de analogía con la neurona biológica (Figura 2.10).

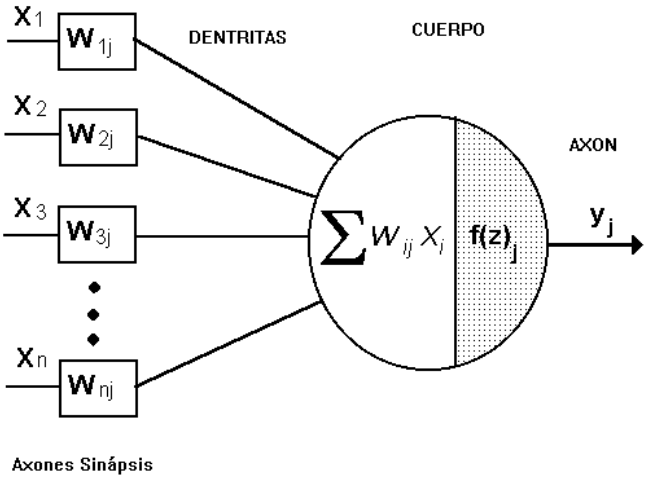

Figura 3.7 Unidad de proceso (Perceptron simple).

La **regla de aprendizaje del perceptrón simple** es un proceso adaptativo que sirve para determinar los valores de los pesos sinápticos y del umbral; consiste en comenzar con valores iniciales aleatorios e ir modificándolos de forma iterativa cuando la salida de la unidad de proceso no coincida con la salida esperada. Con esto se obtiene el error a cada salida como:

$$e_j(k) = z_j(k) - y_j(k) \tag{3.10}$$

donde, $y(k)$ es la salida generada por la unidad de proceso y $z_j(k)$ la salida deseada u objetivo.

La regla de aprendizaje está dada por la siguiente ecuación:

$$w_j(k+1) = w_j(k) + \Delta w_j(k), \;\; k = 1,2,\ldots \tag{3.11}$$

siendo:

$$\Delta w_j(k) = \eta(k)[z(k) - y(k)]x_j(k) \tag{3.12}$$

Esto significa que la variación del peso es proporcional al producto del error de la ecuación 3.10 por la componente j-ésima del patrón de entrada introducido en la iteración k, es decir, $x_j(k)$. El parámetro $\eta(k)$ es una constante de proporcionalidad que lleva por nombre **tasa de aprendizaje**, ya que cuanto mayor sea, más se modifica el peso sináptico y cuando este valor es pequeño, la red aprende poco a poco [119], [120].

3.2.2 Arquitectura del MLP

La arquitectura del perceptrón multicapa tiene como característica que todas sus neuronas están agrupadas u organizadas en capas de diferentes niveles. A cada capa le pertenece un conjunto de neuronas y existen tres tipos de capas diferentes: capa de entrada, capas ocultas y capa de salida, como se puede observar en la Figura 3.8.

Las neuronas de la capa de entrada actúan de forma diferente a las demás neuronas, estas se encargan de recibir la información o patrones de entrada y transmiten o propagan dicha información a todas las neuronas de la siguiente capa. La capa final o capa de salida de la red proporciona al entorno exterior la respuesta de la red neuronal para cada uno de los patrones de entrada. En las capas ocultas se lleva a cabo un procesamiento no lineal de los patrones que reciben.

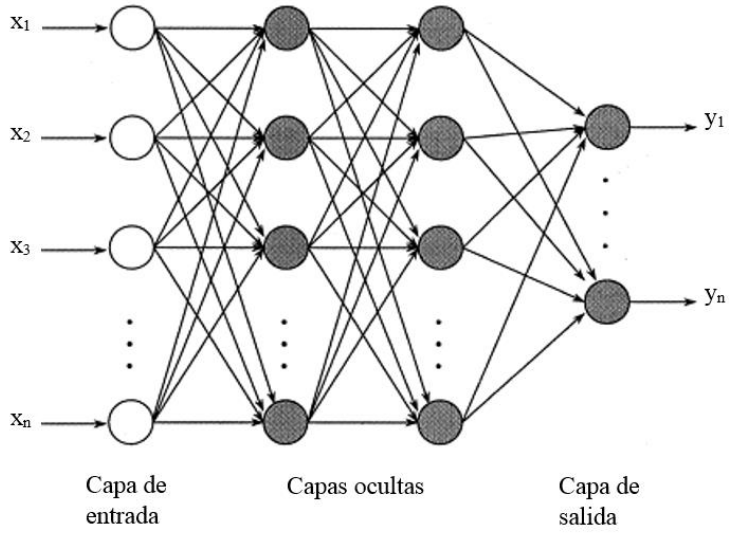

Figura 3.8 Arquitectura del Perceptrón Multicapa.

Las conexiones del MLP siempre se encuentran dirigidas hacia adelante, de aquí que a esta arquitectura también se le conozca con el nombre de red alimentada hacia adelante o redes "*feedforward*". Las conexiones que existen entre las neuronas llevan asociado un valor real, que lleva por nombre peso de la conexión o peso sináptico. Además, todas las neuronas tienen asociado un umbral o *bias*, éste se trata como una conexión más a la neurona y en este trabajo su entrada es constante e igual a 1.

Comúnmente todas las neuronas de una capa están conectadas a las neuronas de la siguiente capa, sin embargo, existen configuraciones de redes neuronales que pueden omitir conexiones entre neuronas, cuando este no es el caso, se dice que la red está totalmente conectada [119], [120].

3.2.2.1 Propagación de las señales de entrada

Para el MLP existe una relación en las variables de entrada y las variables de salida. Ésta se obtiene al propagar hacia adelante la información de las variables de entrada. La respuesta de cada neurona al procesar la información que recibe a su entrada se conoce como activación de la neurona, la cual es propagada a través de las conexiones de las neuronas.

Sea un MLP con C capas, $C-2$ capas ocultas y n_c neuronas en la capa c, para $c = 1,2,\dots,C$. Sea $W^c = (w_{ij}^c)$ la matriz de pesos asociada a las conexiones de la capa c a la capa $c+1$ para $c = 1,2,\dots,C-1$, donde w_{ij}^c representa el peso de la conexión de la neurona i de la capa c a la neurona j de la capa $c+1$; además, sea $U^c = (u_i^c)$ el vector de umbrales de las neuronas de la capa c para $c = 2,\dots,C$. Se denota a_i^c a la activación de la neurona i de la capa c. Las activaciones se calculan como sigue:

Activación de las neuronas de la capa de entrada (a_i^1):

$$a_i^1 = x_i \;\; para \;\; i = 1,2,\dots,n_1 \tag{3.13}$$

donde, $X = (x_1, x_2, \dots, x_{n_1})$ representa el patrón de entrada a la red.

Activación de las neuronas de la capa oculta c (a_i^c):

$$a_i^c = f\left(\sum_{j=1}^{n_{c-1}} w_{ji}^{c-1} a_j^{c-1} + u_i^c\right) para\ i = 1,2,\dots,n_c\ y \tag{3.14}$$

$$\boldsymbol{c = 2,3,\dots,C-1}$$

Activación de las neuronas de la capa de salida a_i^C:

$$y_i = a_i^C = f\left(\sum_{j=1}^{n_{C-1}} w_{ji}^{C-1} a_j^{C-1} + u_i^C\right) para\ i = 1,2,\dots,n_C \tag{3.15}$$

donde, $Y = (y_1, y_2, \dots, y_{n_C})$ es el vector de salida de la red neuronal y f es la función de transferencia o activación [119], [120].

3.2.2.2 Funciones de activación

Como se mencionó anteriormente, el perceptrón hacía uso de funciones lineales a la salida como la función signo o escalón, esto limita el rango de aplicaciones del perceptrón debido a su incapacidad por clasificar ciertos conjuntos o patrones. Al principio de su desarrollo se logró modificar su esquema con el uso de múltiples capas de neuronas, lo que dio a lugar al MLP, sin embargo este evento no hubiera funcionado sin el hecho de cambiar el tipo de funciones de transferencia o activación, lo que abrió camino a nuevos estándares de clasificación.

Actualmente, las funciones de activación más utilizadas para el MLP son la función sigmoidea y la función tangente hiperbólica. Estas funciones poseen como rango un conjunto de valores continuo dentro de los intervalos $\{0,1\}$ y $\{-1,1\}$, respectivamente. Estas funciones de activación están dadas por las siguientes expresiones:

- Función sigmoidea:

$$f_1(x) = \frac{1}{1 + e^{-x}} \tag{3.16}$$

- Función tangente hiperbólica:

$$f_2(x) = \frac{1 - e^{-x}}{1 + e^{-x}} \tag{3.17}$$

Tanto la función sigmoidea, como la función tangente hiperbólica, son crecientes con dos niveles de saturación: en el caso de la sigmoidea el máximo proporciona un valor de salida 1, y el mínimo un valor de salida 0, en el caso de la tangente hiperbólica el máximo proporciona valor 1, y el mínimo valor -1, estos comportamientos se pueden apreciar en la Figura 3.9.

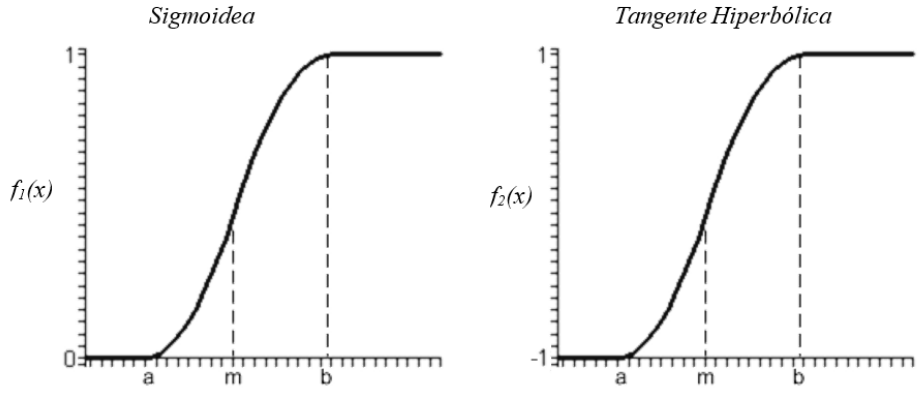

Figura 3.9 Funciones de activación del MLP.

Por lo general, las funciones de activación para el MLP son las mismas en todas las neuronas de la red (exceptuando el caso de las neuronas de entrada que se mencionó anteriormente), sin embargo, la elección de una u otra depende de la aplicación a la que está orientada la RNA, ya que depende de los valores numéricos que se quieran manejar durante el proceso o funcionamiento del sistema [119], [120].

3.2.3 Algoritmo BackPropagation

Un algoritmo de aprendizaje es un mecanismo o técnica que permite modificar o adaptar los parámetros (pesos sinápticos) de una RNA. El algoritmo de aprendizaje del MLP es del tipo supervisado, es decir, la modificación de los pesos de la red neuronal se realiza de tal forma que la salida de ésta sea lo más parecida a la salida deseada. Esto implica a su vez, que para cada patrón de entrada se disponga también de un patrón de salida.

El aprendizaje de la red neuronal se considera un problema de minimización de la siguiente forma:

$$Min_W E \tag{3.18}$$

donde, W es el conjunto de parámetros de la RNA (pesos y umbrales), E se considera la función de error que calcula la diferencia entre las salidas deseadas y las salidas de la red, frecuentemente está dada por la siguiente ecuación:

$$E = \frac{1}{N}\sum_{n=1}^{N} e(n) \tag{3.19}$$

donde N es el número de patrones de entrada y $e(n)$ es el error propagado por la red para el patrón de entrada n, el cual está dado por:

$$e(n) = \frac{1}{2}\sum_{i=1}^{n_C} (s_i(n) - y_i(n))^2 \tag{3.20}$$

donde $S(n) = (s_1(n), \dots, s_{n_C}(n))$ y $Y(n) = (y_1(n), \dots, y_{n_C}(n))$ son los vectores de salida esperada o deseada y salida de la red para el patrón presentado, respectivamente.

La meta de la regla de aprendizaje es alcanzada cuando W^* es un mínimo de la función de error E, ya que, en ese punto el error es cercano a cero, lo cual significa que la salida de la red es muy parecida a la salida esperada. Debido a que las funciones de transferencia son ahora no lineales, el problema de minimización es por lo tanto no lineal, esto lleva a buscar la solución por métodos no lineales. El método más conocido utilizado es el algoritmo EBP, también conocido como *método de descenso del gradiente* [121].

El algoritmo de aprendizaje debería realizarse para minimizar el error total de la ecuación 3.19, el proceso comúnmente utilizado se basa en métodos del gradiente estocástico [121], que consiste en la minimización sucesiva de los errores de cada patrón $e(n)$, en lugar del error total E. A este método se le conoce como descenso del gradiente, y se aplica de acuerdo a la siguiente ley de aprendizaje:

$$w(n) = w(n-1) - \alpha \frac{\delta e(n)}{\delta w} \tag{3.21}$$

donde $e(n)$ es el error para el patrón n según la ecuación 3.20, y α es la tasa de aprendizaje.

Debido a la arquitectura en capas de la RNA, es posible aplicar el método del descenso del gradiente de una forma eficiente, a este algoritmo se le dio el nombre de "*BackPropagation*" [88] o también se le conoce como la *regla delta generalizada* [119], [121], [122], [123].

3.2.3.1 Regla delta generalizada

Como se ha mencionado, esta regla es utilizada para el aprendizaje de RNA. Se distinguen dos casos de aplicación de esta regla: uno para los pesos sinápticos de la capa oculta $C-1$ a la capa de salida y para los umbrales de estos últimos, y otro caso para el resto de pesos y umbrales de la red. A continuación se muestra solo un resumen de las ecuaciones de esta regla, en [119], [121], [122], [123], [88], se puede encontrar a fondo el desglose y demostraciones de éstas.

- Parámetros de la capa oculta $C-1$ a la capa de salida

Pesos sinápticos de la capa $C-1$:

$$w_{j\,i}^{C-1}(n) = w_{j\,i}^{C-1}(n-1) + \alpha\delta_i^C(n)a_j^{C-1}(n) \quad para\ j = 1,2,\dots,n_C\ e\ i = 1,2,\dots,n_C \tag{3.22}$$

Umbrales u de la capa de salida:

$$u_i^C(n) = u_i^C(n-1) + \alpha\delta_i^C(n)\ \ para\ \ i = 1,2,\dots,n_C \tag{3.23}$$

Además, el error δ:

$$\delta_i^C(n) = -\big(s_i(n) - y_i(n)\big)f'\left(\sum_{j=1}^{n_{C-1}} w_{j\,i}^{C-1}a_j^{C-1} + u_i^C\right) \tag{3.24}$$

- Parámetros de la capa c a la capa $c+1$:
  Pesos:

$$w_{k\,j}^c(n) = w_{k\,j}^c(n-1) + \alpha\delta_j^{c+1}(n)a_k^c(n) \quad para\ k = 1,2,\dots,n_c, j = 1,2,\dots,n_c\ \ y\ \ c = 1,2,\dots,C-2 \tag{3.25}$$

  Umbrales de capa $c+1$ para $c = 1,2,\dots,C-2$:

$$u_j^{c+1}(n) = u_j^{c+1}(n-1) + \alpha\delta_j^{c+1}(n) \quad para\ j = 1,2,\dots,n_{c+1}\ \ y\ \ c = 1,2,\dots,C-2 \tag{3.26}$$

  Por otra parte:

$$\delta_j^{c+1}(n) = f'\left(\sum_{k=1}^{n_c} w_{k\,j}^c a_k^c + u_j^c\right)\sum_{i=1}^{n_{c+1}} \delta_i^{c+2}(n)w_{j\,i}^c \tag{3.27}$$

3.2.3.2 Derivada de la función de activación

Para calcular los valores de las ecuaciones 3.24 y 3.27, correspondientes a cada neurona es necesario el cálculo de la derivada de las funciones de transferencia o activación (ecuaciones 3.16 y 3.17), tomadas de [119].

- Función sigmoidea
  Derivando la ecuación 3.16 se tiene que:

$$f_1{}'(x) = \frac{1}{(1+e^{-x})^2}(-e^{-x}) = \frac{1}{1+e^{-x}}\frac{e^{-x}}{1+e^{-x}} \tag{3.28}$$

  Así,

$$f_1{}'(x) = f_1(x)(1-f_1(x)) \tag{3.29}$$

  Por lo tanto, de la ecuación 3.24 se define como:

$$\delta_i^c(n) = -\big(s_i(n) - y_i(n)\big)y_i(n)(1-y_i(n)) \tag{3.30}$$

  Mientras que para los valores de δ de las demás neuronas de la red (ecuación 3.27), se tiene la ecuación:

$$\delta_j^{c+1}(n) = a_j^c(n)(1-a_j^c(n))\sum_{i=1}^{n_{c+1}} \delta_i^{c+2}(n)w_j^c{}_i \tag{3.31}$$

- Función tangente hiperbólica
  Si se toma en cuenta que $f_2(x) = 2f_1(x) - 1$, entonces:

$$f_2{}'(x) = 2f_1(x)(1-f_1(x)) \tag{3.32}$$

  Por lo tanto, cuando se utilice la función tangente hiperbólica las ecuaciones 3.30 y 3.31 se tendrán que multiplicar por un factor de 2 únicamente.

Capítulo 4

S-ROHM, Sistema de Reconocimiento de Objetos basado en Histogramas de gradientes orientados y perceptrón Multicapa

En el presente capítulo se describe el sistema propuesto en este trabajo de tesis, el cual consiste en un modelo enfocado al reconocimiento de objetos que hace uso del algoritmo de Histogramas de Gradientes Orientados como extractor de características, seguido por un clasificador basado en el Perceptrón Multicapa, el cual denominaremos a partir de este momento S-ROHM. La unión de estos dos algoritmos tiene por finalidad desarrollar un sistema de reconocimiento de objetos que tenga la capacidad de detectar o reconocer más de un objetivo. Cabe aclarar que no se utiliza ningún pre-procesado o segmentación de la imagen extrínsecos al sistema, más sin embargo, estos dos pasos se calculan de forma intrínseca dentro del módulo de extracción de características.

El desarrollo de este trabajo fue realizado mediante el lenguaje de programación JAVA, se trata de un lenguaje de programación orientado a objetos desarrollado por Sun Microsystems a principios de los años 90´s.

4.1 Extracción de características mediante HOG

La propuesta realizada por N. Dalal y B. Triggs [1] es utilizada en el proceso de extracción de características del sistema presentado en este escrito. Este proceso consiste en implementar los módulos descritos en la sección 3.1 del capítulo anterior. Es necesario mencionar que se ha implementado una modificación al algoritmo a nivel inicial de procesamiento, la cual se detalla en la Sección 4.1.1.1. El algoritmo HOG, utilizado en la etapa de extracción de características, está constituido por cuatro fases: cálculo del gradiente, agrupación de orientaciones, obtención de los bloques de características y la normalización de dichos bloques. A continuación se describe la implementación de cada una de las fases y se muestran ejemplos de los resultados obtenidos de cada una de ellas.

4.1.1 Gradiente de la imagen

Sea una imagen representada por la matriz $A = [a_{ij}]_{w \times h}$, donde w es el ancho y h el alto de la imagen; a representa el valor del *ij-ésimo* pixel de la imagen, para $a \in \{0,1,2, \dots, 2^b - 1\}$, donde b representa el número de bits utilizados para representar los valores de los pixeles (en nuestro caso 8 bits para escala de grises). Para el cálculo del gradiente se propone utilizar un operador derivador simple unidimensional del tipo $[-1,0,1]$, para cada pixel tanto de forma horizontal como de forma vertical en la imagen de acuerdo al pseudocódigo mostrado en la Tabla 4.1.

Tabla 4.1 Pseudocódigo del algoritmo que obtiene el gradiente de la imagen.

```
01 | método getGradiente(imagen A)
02 | inicio
03 |    para (j=0 hasta h-1) hacer
04 |       para (i=0 hasta w-1) hacer
05 |          Calcula la resta de pixeles vecinos en ambos ejes y los almacena en (mx, my)
06 |          N_xy ← sqrt(mx^2 + my^2), θ ← tan^-1(my/mx)
07 |          Almacena N_xy en el objeto magnitud de la clase imagen
08 |          Almacena θ en el objeto ángulo de la clase imagen
09 |       fin_para
10 |    fin_para
11 | fin_método
```

Este método regresa el conjunto de matrices gradiente el cual está conformado por la matriz magnitud $N = [n_{ij}]_{w \times h}$, y la matriz ángulo $Th = [th_{ij}]_{w \times h}$, del gradiente de una imagen, ambas de tamaño $w \times h$ y definidas por:

$$n_{ij} = \sqrt{\left[a_{i+1,j} - a_{i-1,j}\right]^2 + \left[a_{i,j+1} - a_{i,j-1}\right]^2} \quad \textbf{(4.1a)}$$

$$th_{ij} = \tan^{-1}\frac{a_{i,j+1} - a_{i,j-1}}{a_{i+1,j} - a_{i-1,j}} \quad \textbf{(4.1b)}$$

En la Figura 4.1 se muestran los resultados de aplicar a una imagen el algoritmo que permite obtener su gradiente, en estos resultados se pueden apreciar los contornos o bordes de toda la imagen.

Figura 4.1 Imagen de rostro humano a) Imagen en escala de grises. b) Magnitud del gradiente. c) Ángulos de orientación del gradiente (valor del ángulo proporcional a la intensidad de la imagen).

En esta fase no existen variables que puedan ser identificadas para su evaluación. Además, como se mencionó en la Sección 3.1.1, otros tipos de operadores derivativos pueden ser utilizados; para este trabajo se usa el operador propuesto en [1]. Por otro lado, buscando mejorar el rendimiento del algoritmo HOG, se implementó una modificación al cálculo del gradiente, descrita a continuación.

4.1.1.1 Mejora del gradiente de la imagen

Aunque, si bien es cierto que una segmentación no puede llevarse a cabo sin un reconocimiento parcial de los objetos que contiene una imagen y, a su vez, un reconocimiento no puede existir sin una segmentación de la misma, con el fin de mejorar el proceso de caracterización de la imagen, se presenta una modificación que tiene por finalidad mejorar la representación de las propiedades del algoritmo de cálculo de gradiente.

En la Figura 4.1 c), donde se representan gráficamente los ángulos de orientación del gradiente, se pueden percibir algunos rasgos de la imagen original, sin embargo también se percibe información que puede ser interpretada como ruido, esta información aparece debido a que el fondo de la imagen no es homogéneo; en otras palabras, el entorno o ambiente que también pertenece a la imagen pero que no forma parte del objeto de interés, introduce información no deseada. Buscando minimizar los efectos de esta información no deseada, la modificación implementada consiste en aplicar una umbralización a las matrices N y Th, representada como:

$$n_{ij} = \begin{cases} 0, & si\ n_{ij} < umbral\_grad \\ n_{ij}, & en\ otro\ caso \end{cases}$$
$$th_{ij} = \begin{cases} 0, & si\ n_{ij} < umbral\_grad \\ th_{ij}, & en\ otro\ caso \end{cases} \quad \textbf{(4.2)}$$

Aunque esta umbralización es totalmente empírica, se puede apreciar sobre todo en la Figura 4.2 c) cómo existe una diferencia significativa en cuanto a la representación del gradiente. Además, esta variación se implementa debido a que no representa una carga significativa de tiempo de procesado de la imagen comparada con otras técnicas.

Se pretende que con esta modificación el sistema tenga una mejor representación de las características debido a la aparición de zonas oscuras (con niveles de gris cero) en las nuevas imágenes. Esto se traduce en celdas y/o bloques vacíos, y que, debido a las características del clasificador se espera permitan una mejor separación ya que se elimina información no perteneciente al objeto de interés.

Figura 4.2 Imagen de rostro humano con mejora del gradiente a) Imagen en escala de grises. b) Magnitud del gradiente. c) Ángulos de orientación del gradiente (con mejora).

En este proceso se identifica la variable umbralGrad para su evaluación, que más tarde será agrupada junto con las demás variables que intervienen como parámetros de configuración del sistema para sus diferentes fases. Así mismo, cabe señalar que las pruebas y experimentos que se realizarán, incluirán esta mejora en el cálculo del gradiente, debido a que se observó una mejora sistemática en pruebas de rendimiento realizadas previamente.

4.1.2 Agrupación de orientaciones

La fase de agrupación de orientaciones consiste en calcular el histograma de los gradientes. Este proceso consiste en dividir las imágenes gradiente N y Th en celdas y para cada celda se calcula su histograma. El proceso inicia formando celdas de npx pixeles en el eje x, y npy celdas en el eje y.

También se define a $V_{cx,cy}$ como la matriz de celdas del tipo **R-HOG** de la imagen; donde $cx = 1,2,\dots,ncx$ y $cy = 1,2,\dots,ncy$, donde ncx y ncy representan el número de celdas en el eje x y y respectivamente, y los parámetros calculados $npx = w/ncx$ y $npy = h/cny$.

Además, para el caso en estudio, en la generación de agrupaciones de orientaciones se definieron 9 contenedores (*bins*), con ángulos sin signo de 0° a 180°, para cuantificar las orientaciones del gradiente (ver Figura 4.3 b); así, los valores centrales de los *bins* definidos son: $bin_0 = 10°$, $bin_1 = 30°$, $bin_2 = 50°$, $bin_3 = 70°$, $bin_4 = 90°$, $bin_5 = 110°$, $bin_6 = 130°$, $bin_7 = 150°$ y $bin_8 = 170°$; teniendo una distancia entre *bins* de 20°, denotada como d_{bins}.

Ahora, cada elemento de la matriz $V_{cx,cy}$, es un vector denotado por $v_{cx,cy}$, formado por tantos elementos como *bins* sean definidos (en este caso 9) cuyo valor inicial es cero, y que representa el vector del histograma de orientaciones de la celda cx, cy. Cada elemento de $v_{cx,cy}$ está asociado con un *bin*: $\{(v_{cx,cy}(0), bin_0), (v_{cx,cy}(1), bin_1), \dots, (v_{cx,cy}(8), bin_8)\}$.

Finalmente, $d_a = (th - bin_a)$ y $d_b = (bin_b - th)$ representan las distancias entre $th_{i,j}^{cx,cy}$, un elemento (ángulo de orientación) de la celda extraída de Th, y los *bins* contiguos bin_a, bin_b, considerando que $bin_a \leq th \leq bin_b$.

Así, el resultado de esta fase queda definido como

$$v_{cx,cy}(bin_a) = \begin{cases} v_{cx,cy}(bin_a) + (1 - \dfrac{d_a}{d_{bins}}) \cdot n_{i,j}^{cx,cy}, & si\ bin_a < th_{i,j}^{cx,cy} \\ v_{cx,cy}(bin_a) + n_{i,j}^{cx,cy}, & si\ bin_a = th_{i,j}^{cx,cy} \end{cases}$$

$$v_{cx,cy}(bin_b) = \begin{cases} v_{cx,cy}(bin_b) + (1 - \dfrac{d_b}{d_{bins}}) \cdot n_{i,j}^{cx,cy}, & si\ th_{i,j}^{cx,cy} < bin_b \\ v_{cx,cy}(bin_b) + n_{i,j}^{cx,cy}, & si\ bin_b = th_{i,j}^{cx,cy} \end{cases} \quad \textbf{(4.3)}$$

donde $v_{cx,cy}(bin_a)$ es el elemento asociado al bin_a y $v_{cx,cy}(bin_b)$ es el elemento asociado al bin_b; $n_{i,j}^{cx,cy}$ y $th_{i,j}^{cx,cy}$ pertenecen a la celda cx, cy, $i = 0,1,2, \dots, npx$ y $j = 0,1,2, \dots, npy$.

En la Tabla 4.2 y Tabla 4.3 se presentan los pseudocódigos que definen la operación de esta fase.

Tabla 4.2 Pseudocódigo del algoritmo que obtiene los histogramas de orientaciones.

```
01 | método getHistogramas(imagenes G')
02 | inicio
03 |    Divide imágenes en celdas de tamaño (npx, npy)
04 |    para (cada celda (cx, cy) de la imagen gradiente') hacer
05 |       para (y=0 hasta npy) hacer
06 |          para (x=0 hasta npx) hacer
07 |             método getVotoPonderado(th, nxy)
08 |             Almacena el voto ponderado en el bin v_cx,cy(h)
09 |          fin_para
10 |       fin_para
11 |    fin_para
12 | fin_método
```

Tabla 4.3 Pseudocódigo del algoritmo que obtiene los votos ponderados.

```
01 | método getVotoPonderado(real th, nxy)
02 | inicio
03 |    Encuentra bins vecinos al ángulo th
04 |    Calcula da, db entre bins vecinos y ángulo th
05 |    Calcula porcentaje de magnitud nxy para cada bin según distancias
06 |    regresa voto ponderado
07 | fin_método
```

Al finalizar el procesamiento de los algoritmos anteriores se obtiene una nueva matriz que contiene $ncx \times ncy$ celdas, cada celda corresponde a un vector de histograma local. Una mejor forma de comprender el proceso de agrupación de orientaciones y/o cálculo de histogramas se explica con el siguiente ejemplo tomando de referencia la Figura 4.3.

- Supóngase que se tiene una celda en la cual se cuantificaron las orientaciones del gradiente dentro de 9 contenedores (*bins*) con ángulos sin signo ($0° - 180°$)
    - El voto es la magnitud del gradiente por si misma
    - La interpolación lineal de los votos se procesa como sigue:
        - Si $\theta = 35°$.
        - La distancia a los *bins* centrales *Bin* 30 y *Bin* 50 es de 5 y 15 grados, respectivamente.
        - Por lo tanto, las proporciones son $15/20 = 3/4$, $5/20 = 1/4$

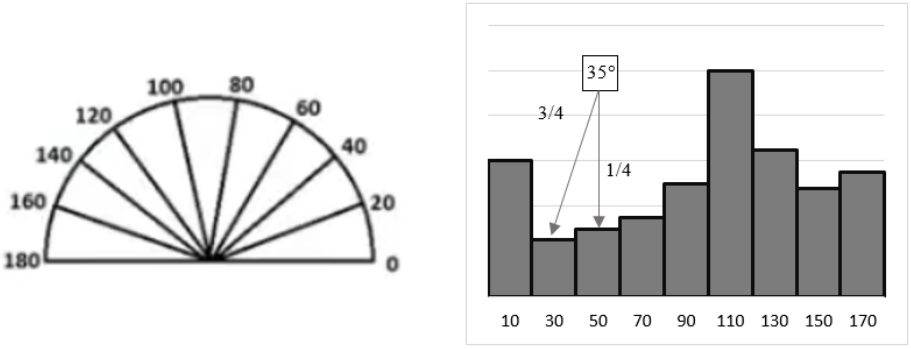

Figura 4.3 Representación de contenedores. a) Espaciado o separación de 9 *bins* uniformemente (20°). b) Histograma con base a los *bins* centrales.

La Figura 4.4 muestra el resultado de aplicar la fase de agrupación de orientaciones con los siguientes parámetros: voto lineal de la magnitud del gradiente (porcentaje de magnitud), 9 $bins$ de orientaciones ($0° - 180°$), 10×10 celdas y $\frac{ancho\_imagen}{núm\_celdas\_en\_x} \times \frac{alto\_imagen}{núm\_celdas\_en\_y}$ pixeles (npx y npy respectivamente).

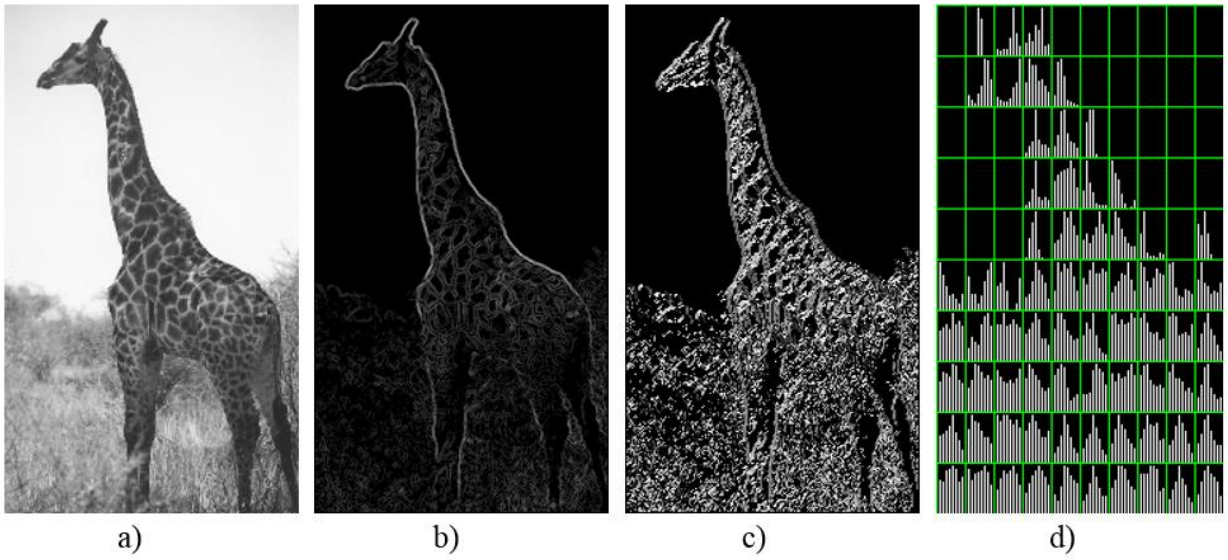

Figura 4.4 Imagen de la clase girafa. a) Imagen original en escala de grises. b) Magnitud del gradiente. c) Ángulos del gradiente. d) Celdas de histogramas R-HOG.

En esta fase se pueden identificar las siguientes variables: número de celdas por ventana en x ($numCelVentX$), número de celdas por ventana en y ($numCelVentY$), número de orientaciones o $bins$ ($numOrien$), signo de orientaciones ($sigOrien$). Otros parámetros son calculados utilizando estas variables y las propiedades de la imagen ($ancho$ y $alto$), los cuales son: número de pixeles por celda en x ($numPixCelX$), número de pixeles por celda en y ($numPixCelY$).

4.1.3 Bloques de características y normalización

Cuando se describió al algoritmo HOG (Sección 3.1), las fases de bloque de características y normalización de bloques fueron puntualizadas en secciones separadas (Sección 3.1.3 y Sección 3.1.4, respectivamente). Sin embargo, debido al estrecho vínculo existente entre ambas fases, éstas serán detalladas conjuntamente en la presente Sección.

En esta fase se agrupan las celdas en bloques conectados espacialmente, esto significa que únicamente celdas contiguas podrán ser agrupadas para formar un vector de características más largo.

Sea $V = [v_{ij}]_{w \times h}$ una matriz donde cada uno de sus elementos es un vector de histogramas, w representa el número de elementos a lo ancho, h el número de elementos a lo alto, $\boldsymbol{v}$ representa el ij*-ésimo* vector de histogramas de orientaciones. Así mismo, se tiene que $nctx$ y $ncty$ como el número de celdas traslapadas por bloque en x y y respectivamente. Se define la norma de un vector como sigue:

$$\|v\|_2 = \sqrt{\sum_{i=1}^{n} |v_i|^2} \qquad \textbf{(4.4)}$$

Esta norma (ecuación 4.4) es conocida como la norma ℓ_2 o norma euclídea. Se propone utilizar el tipo de normalización $L_2 - Hys$ del capítulo anterior, Sección 3.1.4, que consiste en utilizar la normalización $L_2 - norm$ (ecuación 3.5), seguida por el recorte de los valores del vector resultante, es decir, $max(\|v_i\|) = 0.2$, seguida por una re-normalización $L_2 - norm$, donde $\|\cdot\|$ hace referencia a la primera normalización del vector de características v. Por simplicidad en la codificación del sistema, de las ecuaciones 4.4 y 3.4 se obtiene la normalización ${L_2}' - norm$ como sigue:

$${L_2}' - norm(v) = \frac{v_i}{\sqrt{\|v\|_2^2 + \varepsilon^2}} = \frac{v_i}{\sqrt{\sum_{i=1}^{n}(|v_i|^2) + \varepsilon^2}} \qquad \textbf{(4.5)}$$

En la Tabla 4.4 y Tabla 4.5 se presentan los pseudocódigos propuestos para el desarrollo de esta fase, en la Figura 4.5 se muestra un ejemplo de la distribución de los bloques a lo largo de una imagen.

Las variables a evaluar en esta fase son: el número de celdas por bloque en x ($numCelBloqX$), el número de celdas por bloque en y ($numCelBloqY$), el número de celdas traslapadas en x ($numCelTrasX$) y el número de celdas traslapadas en y ($numCelTrasY$).

Tabla 4.4 Pseudocódigo del algoritmo que obtiene los bloques de características.

```
01 | método getBloquesHog(celdas B)
02 | inicio
03 |    para (j=0 hasta ncy con j=j+ncty) hacer
04 |       para (i=0 hasta ncx con i=i+nctx) hacer
05 |          para (l=0 hasta ncby) hacer
06 |             para (k=0 hasta ncbx) hacer
07 |                Agrega el vector histograma de la celda (i, j) al bloque bm
08 |             fin_para
09 |          fin_para
10 |          método normalizaHistograma(bm, ε)
11 |          Concatenar vector normalizado con vector de características hog
12 |       fin_para
13 |    fin_para
14 | fin_método
```

Tabla 4.5 Pseudocódigo del algoritmo que calcula la normalización de las características.

```
01 | método normalizaHistograma(bloque B, real ε)
02 | inicio
03 |    para (cada elemento del bloque B) hacer
04 |       Aplicar normalización L2'-norm
05 |       // Recortar el valor de cada elemento del vector bi
06 |       para (cada elemento del vector b_i) hacer
07 |          si (b_i(j) > 0.2) entonces
08 |             b_i(j) = 0.2
09 |          fin_si
10 |       fin_para
11 |       Aplicar normalización L2'-norm
12 |    fin_para
13 |    regresa bloque normalizado
14 | fin_método
```

Si se considera una ventana de detección de 8×16 celdas, 9 *bins* de orientaciones del gradiente, orientaciones sin signo $(0° - 180°)$, bloques de tamaño 2×2, traslape de 1 celda por eje, se tiene entonces: 7×15 bloques de 9 *bins*, cada bloque de $2\ x\ 2$ celdas. Por lo tanto, el vector de características final tendrá una dimensión de $7 \times 15 \times 4 \times 9 = 3780$ elementos o características.

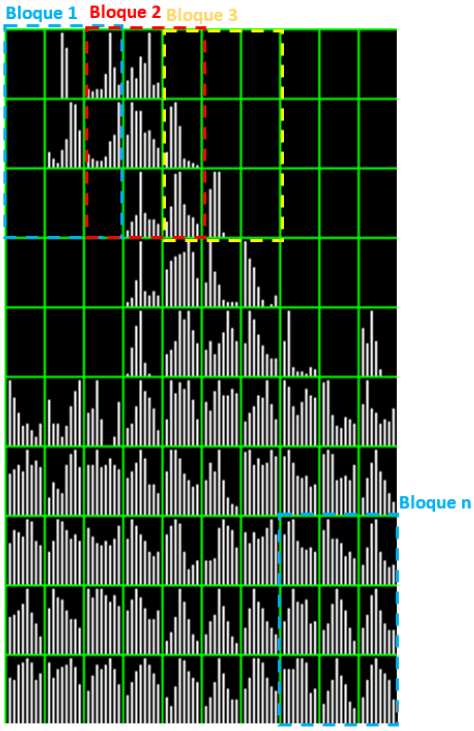

Figura 4.5 Obtención de bloques de caracteristicas (descripción gráfica).

4.2 Clasificador RNA-MLP y algoritmo EBP

Una RNA, con una arquitectura MLP y entrenada con el algoritmo *backpropagation*, es la responsable del proceso de clasificación que complementa al sistema S-ROHM. Como cualquier método de clasificación que emplee un enfoque neuronal, este proceso está integrado por una fase de entrenamiento y una fase de clasificación (operación).

En la fase de entrenamiento, el sistema hace uso de los vectores de características generados en el proceso anterior (extraídos de la base de datos o conjunto de entrenamiento), tanto para las clases de objeto positivas como también las negativas. Siendo una clase de objeto positiva el conjunto de imágenes que si contienen al objeto del cual se desea su futuro reconocimiento. Por su parte, el conjunto de clases de objetos negativas se refiere a las imágenes donde no existe un objeto específico, más sin embargo existe un entorno o un fondo diferente a cualquier objeto de estudio. En la fase de operación la RNA recibe la ventana de detección como vector de características y la clasifica de acuerdo a lo aprendido.

La arquitectura de la red utilizada en el sistema propuesto está integrada por una capa oculta con cinco neuronas, las funciones de activación son de tipo sigmoidea para las neuronas de las capas oculta y de salida. A partir de este momento se hará referencia al umbral de cada neurona como *bias*, del cual se habló en la Sección 3.2.2, éste se comporta como un peso sináptico más de la neurona con una entrada asociada igual a 1.

4.2.1 Fase de Entrenamiento

Con la finalidad de adaptar la RNA al reconocimiento de ciertos objetos, esta fase hace uso de todo el conjunto de entrenamiento, consiste en presentar todos y cada uno de los patrones de entrada (vectores de características) y patrones de salida (clases pertenecientes). En el sistema propuesto esta fase termina cuando ha alcanzado una cantidad determinada de iteraciones o generado un error mínimo, lo que ocurra primero.

En la Figura 4.6 se muestra un diagrama de bloques de la fase de entrenamiento, la cual está integrada por los siguientes módulos:

- **Crear RNA**: en este módulo se configuran los parámetros que definen las características de la RNA-MLP, tales como el número de neuronas en la capa oculta, número de neuronas en la capa de salida, funciones de activación, entre otros.
- El módulo **Inicializar Pesos Sinápticos** se encarga de asignar el valor inicial a los pesos sinápticos de las neuronas, tanto de la capa oculta como la de salida.
- El módulo **RNA-MLP** representa la estructura de la red, es decir define como interactúan entre si las neuronas que la componen; además, también integra a los sub-módulos auxiliares que permiten la ejecución del algoritmo *backpropagation*.
- **Entrenamiento**. Este módulo recibe el nombre de la fase a la cual pertenece debido a que en él se implementa el algoritmo *backpropagation*, teniendo por función adaptar a la RNA para que pueda reconocer al objeto en cuestión

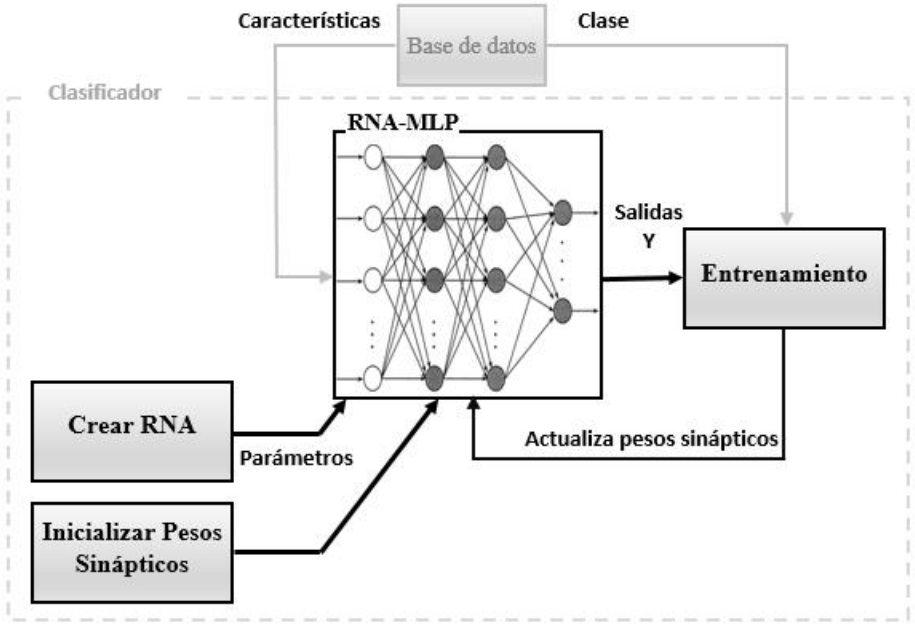

Figura 4.6 Esquema del Clasificador en fase de entrenamiento.

4.2.1.1 Crear RNA

El módulo **Crear** RNA tiene por función definir las características que permiten configurar la estructura de la red, en él se definen los siguientes parámetros de configuración:

- neuronasEntrada: número de neuronas de entrada a la red; este valor es equivalente a la cantidad de elementos que contiene cada vector de características.
- neuronasOculta: número de neuronas en la capa oculta.
- neuronasSalida: número de neuronas en la capa de salida; este valor es definido por el número de clases u objetos a reconocer.
- funcEntrada: define el tipo de función de activación para la capa de entrada.
- funcOculta: define el tipo de función de activación para la capa oculta.
- funcSalida: define el tipo de función de activación para la capa de salida.
- pesosEntrada: variable de selección para el tipo de pesos en la capa de entrada.
- pesosOculta: variable de selección para el tipo de pesos en la capa oculta.
- pesosSalida: variable de selección para el tipo de pesos en la capa de salida.
- biasEntrada: valor numérico para el *bias* de cada neurona en la capa de entrada.
- biasOculta: valor numérico para el *bias* de las neuronas de la capa oculta.
- biasSalida: valor numérico para el *bias* de las neuronas de la capa de salida.
- alpha: valor numérico para la tasa de aprendizaje.
- error: valor numérico para la función de error.
- numIteraciones: el número de iteraciones máxima que puede alcanzar la fase de entrenamiento.

Las funciones de activación utilizadas son de tipo sigmoidea y tangente hiperbólica (descritas en la Sección 3.2.2.2). Los pesos sinápticos son generados aleatoriamente y están definidos en ciertos rangos, por ejemplo $[0, 1]$, $[-1, 1]$, $[-0.1, 0.1]$, $[-0.25, 0.25]$, etc. El *bias* de cada neurona puede estar dentro del rango $[-1,1]$, sin embargo, el más utilizado en los sistemas de RNA's es -1. La tasa de aprendizaje es un valor pequeño (un valor típico es 0.2), sin embargo, puede ser definido como un valor decreciente, creciente o una función, todo depende del comportamiento del sistema.

4.2.1.2 Inicializar Pesos Sinápticos

El módulo **Inicializar Pesos Sinápticos** define los valores que los pesos sinápticos tendrán al inicio del entrenamiento. De acuerdo al parámetro especificado tipo_peso, el sistema genera valores numéricos aleatorios dentro del rango establecido. Aunque existe un parámetro de configuración de los pesos sinápticos de las neuronas de la capa de entrada, éstos se establecen con valor numérico 1 debido a que esta capa únicamente tiene como objetivo presentar todas y cada una de las entradas o elementos del vector de características a las neuronas de la capa oculta. Una descripción general del funcionamiento de este módulo se muestra en la Tabla 4.6:

Tabla 4.6 Pseudocódigo del algoritmo que inicializa los pesos sinápticos de la RNA-MLP.

```
01 |  método inicializaPesos(neuronas n, tipo_peso)
02 |  Inicio
03 |     // Para cada neurona n de la capa
04 |     para (j=0 hasta nn) hacer
05 |     // Para cada pesos p de la neurona n
06 |        para (i=0 hasta pm) hacer
07 |           Genera un valor aleatorio según tipo_peso
08 |           Almacena el valor en el peso i de la neurona j
09 |        fin_para
10 |     fin_para
11 |  fin_método
```

Además de lo descrito, el sistema S-ROHM está diseñado para poder leer desde un archivo los pesos sinápticos correspondientes a una ejecución anterior del sistema, ya que, éste almacena frecuentemente los pesos sinápticos durante el proceso de entrenamiento. La restauración de los pesos sinápticos se puede dar siempre y cuando coincidan los parámetros de configuración con las características de los pesos almacenados. Por defecto el sistema almacena dichos pesos en los archivos "Backp.PesosIni.txt" y "Backp.PesosFin.txt" para los pesos iniciales y finales respectivamente.

4.2.1.3 RNA-MLP en fase de entrenamiento

El módulo **RNA-MLP** hace referencia a la "conceptualización" de la RNA obtenida de los parámetros de configuración definidos en el módulo **Crear RNA**. Para el caso en estudio, se propone una red MLP con una capa oculta de 5 neuronas. La cantidad de neuronas de entrada depende directamente del número de elementos del vector de características, es decir, estará en función de los parámetros del extractor de características. Finalmente, el número de neuronas en la capa de salida dependerá del número de objetos diferentes que se desea identificar.

La Tabla 4.7 muestra el pseudocódigo del módulo **RNA-MLP** utilizado para la propagación de las señales de entrada.

Tabla 4.7 Pseudocódigo del algoritmo de propagación de señales de la **RNA-MLP**.

```
01 | método getPropagación(neuronas n, tipo_peso)
02 | Inicio
03 |    // Se presenta el patrón a la capa de entrada
04 |    para (cada neurona k de la capa de entrada) hacer
05 |       Presentar elemento k del conjunto de caracteristicas a la neurona k
06 |       Actualizar la salida de la neurona k con la nueva información → y_e(k)
07 |    fin_para
08 |    // Se presentan las salidas de las neuronas de entrada a la capa oculta
09 |    para (cada neurona j de la capa oculta) hacer
10 |       Presentar el conjunto y_e a la neurona j
11 |       Actualizar la salida de la neurona j → y_o(j)
12 |    fin_para
13 |    // Se presentan las salidas de las neuronas de la capa oculta a la capa de salida
14 |    para (cada neurona i de la capa de salida) hacer
15 |       Presentar el conjunt y_o a la neurona i
16 |       Actualizar la salida de la neurona i → y_s(i)
17 |    fin_para
18 | fin_método
```

4.2.1.4 Entrenamiento

La fase de **Entrenamiento** implementa el algoritmo *backpropagation*, descrito en la Sección 3.2.3, y auxiliándose de la salida generada por el módulo **RNA-MLP**, realiza la adaptación de la red al problema en estudio mediante la actualización de los pesos sinápticos. La Figura 4.7 muestra la representación gráfica que sigue el flujo de datos dentro del módulo **Entrenamiento**.

El bloque de **ajuste de pesos** se encarga de aplicar la regla delta generalizada (ver Sección 3.2.3.1). La operación de este módulo puede concluir debido a cualquiera de las siguientes causas:

- Número máximo de iteraciones: una iteración consiste en presentar el conjunto de entrenamiento a la red una vez.
- Error mínimo: El error puede ser calculado como promedio de los errores de una iteración (aprendizaje por lotes) o puede ser calculado cada presentación de patrón (aprendizaje en serie o en línea).

El modelo propuesto contempla ambos factores. El cálculo del error propuesto es realizado cada que se presenta un patrón.

Figura 4.7 Diagrama de flujo del módulo **Entrenamiento**.

4.2.2 Fase de Operación

Una vez concluida la fase de aprendizaje, la estructura de la red, representada por el módulo **RNA-MLP**, está completada y lista para empezar a trabajar en su fase de operación.

El diagrama de bloques de la fase de operación puede ser apreciado en la Figura 4.8. El funcionamiento de esta fase es mostrado en el pseudocódigo de la Tabla 4.8 y está formado por los siguientes pasos:

- Un vector de características correspondiente a una ventana de detección es presentado a la red.
- El algoritmo de propagación de señales de la **RNA-MLP** (Tabla 4.7) actua y genera el resultado del proceso de clasificación y se envia al módulo de decisión.
- El bloque de decisión define la clase a la que pertenece el objeto presentado. La operación de este bloque es mostrada mediante el pseudocódigo de la Tabla 4.8.

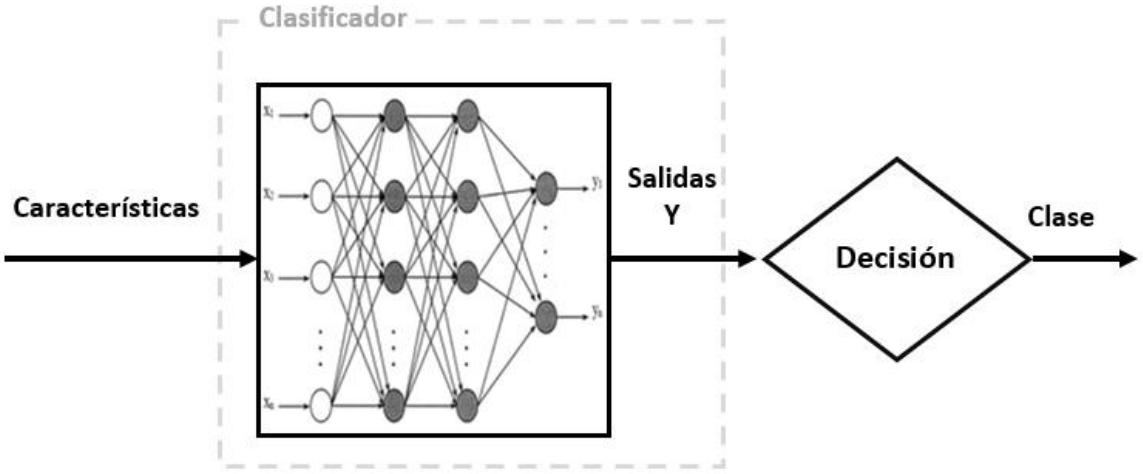

Figura 4.8 Esquema del Clasificador en fase de operación.

Tabla 4.8 Pseudocódigo del algoritmo de decisión.

```
01 | método decisión(vector salidasY)
02 | Inicio
03 |     // Se evalua cada salida
04 |     para (cada salida y_i) hacer
05 |         si (y_i > umbral) entonces
06 |             Establecer como posible objeto de clase i
07 |         de_lo_contrario
08 |             No es un objeto de clase i
09 |         fin_si
10 |     fin_para
11 |     De la lista de posibles objetos, elegir el que tenga la distancia menor al objeto
12 |     fin_para
13 | fin_método
```

4.3 Integración del sistema

Una vez diseñados e implementados los procesos de extracción de características y de clasificación, fueron utilizados para generar clases en el lenguaje JAVA, denominadas "características" y "clasificador" respectivamente. Buscando proporcionar un entorno visual sencillo que permita al usuario acceder y manipular las características que presenta el sistema propuesto, estas clases fueron integradas a una interfaz gráfica de usuario (GUI, *Graphical User Interface*). Debido a las características del lenguaje JAVA la integración del sistema se reduce únicamente a la creación de objetos de cada clase, una sección de control que configura los

parámetros de cada módulo y la interfaz gráfica que gestiona el funcionamiento del sistema en general.

El funcionamiento del sistema S-ROHM se compone en tres modos: entrenamiento, operación y reconocimiento.

El **entrenamiento** busca adaptar al sistema para que sea capaz de reconocer ciertos objetos; se trata de una implementación secuencial de las clases "características" y "clasificador", es decir la base de datos completa (conjunto de entrenamiento) es presentada al proceso de extracción de características, una vez calculados o generados los vectores de características, son almacenados para posteriormente presentarlos al clasificador, de esta forma no tiene que calcularse el mismo vector de características de cada imagen en cada iteración del clasificador.

En el modo de **operación** se realiza una prueba inicial al sistema, la cual consiste en presentarle una base de datos, que contiene las imágenes nuevas, que permite hacer una estimación global del rendimiento del sistema. Las imágenes empleadas en esta prueba tienen características similares a las imágenes usadas en el entrenamiento; por ejemplo, pueden poseer la misma relación de aspecto, pero no necesariamente las mismas dimensiones de imagen.

Finalmente, el modo **reconocimiento** procesa imágenes de entornos completos donde el objeto en estudio está integrado a una escena que comparte con otros objetos, este procesamiento se realiza mediante un barrido de la imagen a través de ventanas de detección de diferentes dimensiones.

En las Secciones siguientes se describen los elementos que integran la GUI del sistema propuesto y que definen la forma en que éste opera.

4.3.1 Parámetros de configuración de S-ROHM

Para que S-ROHM pueda iniciar su operación, es necesario realizar su configuración. Esta sección tiene por finalidad describir los diferentes parámetros que definen el funcionamiento de S-ROHM. La Tabla 4.9 incluye todos los parámetros de configuración del sistema. Es importante mencionar que algunos parámetros son definidos con alguna representación numérica, por ejemplo, el signo de orientaciones puede ser configurado en el rango de $\{0° - 180°\}$ o $\{0° - 360°\}$ pero para la configuración del sistema se utiliza el valor 0 o 1 según sea el rango elegido.

Tabla 4.9 Parámetros de configuración del sistema.

Parámetro	Descripción	Valor(es)
	Parámetros del Extractor de características	
numCelVentX	Número de celdas por ventana de detección en el eje X	[8 - 16]
numCelVentY	Número de celdas por ventana de detección en el eje Y	[8 - 16]
numCelBloqX	Número de celdas por bloque en el eje X	[2-3]
numCelBloqY	Número de celdas por bloque en el eje Y	[2-3]
numCelTrasX	Número de celdas traslapadas en el eje X	[1-2]
numCelTrasY	Número de celdas traslapadas en el eje Y	[1-2]
numOrien	Número de orientaciones del gradiente	[9, 18]
sigOrien	Signo de orientaciones de los ángulos del gradiente	[0, 1]
umbralGrad	Umbral para la mejora del gradiente	0.2
	Parámetros del Clasificador	
neuronasOculta	Número de neuronas en la capa oculta	[5-20]
funcEntrada	Tipo de función de activación para la capa de entrada	Identidad
funcOculta	Tipo de función de activación para la capa oculta	Sigmoidea
funcSalida	Tipo de función de activación para la capa de salida	Sigmoidea
pesosEntrada	Tipo de pesos en la capa de entrada	1.0
pesosOculta	Tipo de pesos en la capa oculta	[-0.25, 0.25]
pesosSalida	Tipo de pesos en la capa de salida	[-0.25, 0.25]
biasEntrada	Bias de cada neurona en la capa de entrada	-1.0
biasOculta	Bias de las neuronas de la capa oculta	-1.0
biasSalida	Bias de las neuronas de la capa de salida	-1.0
alpha	Tasa de aprendizaje	[0.1, 0.25]
error	Función de error	0.000000001
umbral	Vector de umbrales para clases	0.98046875,...
	Parámetros generales del sistema de reconocimiento	
tamVentDetIniX	Tamaño de ventana de detección inicial en el eje X	[100-400]
tamVentDetIniY	Tamaño de ventana de detección inicial en el eje Y	[100-400]
tamVentDetFinX	Tamaño de ventana de detección final en el eje X	[100-400]
tamVentDetFinY	Tamaño de ventana de detección final en el eje X	[100-400]
incTamVentX	Tasa de incremento en el tamaño de la ventana de detección en el eje X	[20-100]
incTamVentY	Tasa de incremento en el tamaño de la ventana de detección en el eje Y	[20-100]
despVentX	Desplazamiento en pixeles de la ventana de detección en el eje X	[10-100]
despVentY	Desplazamiento en pixeles de la ventana de detección en el eje Y	[10-100]

4.3.2 Interfaz Gráfica de Usuario de S-ROHM

La GUI de S-ROHM se compone de una ventana principal, una barra inferior con los parámetros de configuración y un panel de procesamiento que integra a tres pestañas con información de los

diferentes procesos del sistema. La ventana principal contiene al panel de pestañas de los procesos, así como la barra con los parámetros de configuración del sistema, esta barra es común para las pestañas de los procesos con parámetros relacionados a éstos.

4.3.2.1 Barra de parámetros de configuración

La barra de parámetros de configuración permite al usuario definir los valores de los parámetros que determinarán el comportamiento de los procesos que integran al S-ROHM (mostrados en la Tabla 4.9). En la barra se mostrarán los parámetros de los procesos, por lo que la barra incluye la sección de parámetros HOG, la de parámetros RNA y la de parámetros de reconocimiento (ver Figura 4.9).

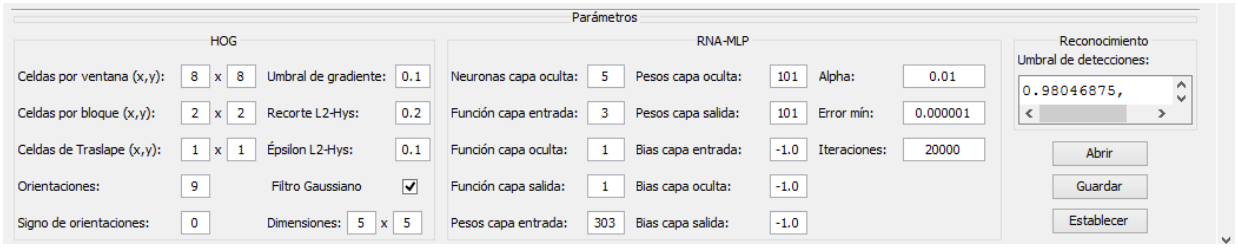

Figura 4.9 Panel o barra de parámetros de configuración.

Los parámetros de configuración **HOG** son principalmente los referentes a los tamaños de los descriptores HOG, celdas, bloques, número y signo de orientaciones, aplicación de filtro gaussiano antes de la agrupación de orientaciones, etc. Todos los componentes de esta sección son valores numéricos que corresponden directamente al valor deseado para la configuración, excepto por el valor del signo de las orientaciones ya que este se elige entre 0 para ángulos de 0 a 180 grados y 1 para ángulos de 0 a 360 grados.

Con respecto a los parámetros de configuración de la **RNA**, incluyen aquellos que definen el número de neuronas de entrada, las funciones de activación de las capas de neuronas, pesos sinápticos y *bias* de las mismas, coeficiente de aprendizaje (Alfa), error mínimo, y el número de iteraciones de la fase de entrenamiento. Las funciones de activación y los pesos de las neuronas son codificados como se muestra en la Tabla 4.10, debido a la variedad de valores disponibles para cada parámetro, los pesos sinápticos se componen de un código de tres dígitos.

Finalmente, el umbral de detecciones correspondiente a la sección **reconocimiento** tiene como finalidad establecer el valor máximo (entre 0.0 y 1.0) en el cual el objeto procesado o ventana de detección no es un objeto correspondiente a la base de datos positiva, o lo que es lo mismo $1.0 - umbral\_de\_detecciones$ es el umbral que determina que tan parecido es el objeto procesado a uno de los objetos de la base de datos positiva.

Tabla 4.10 Códigos de los parámetros de configuración del clasificador.

Parámetro	Código	Valor(es)
Función de activación	1	Función sigmoidea
	2	Función tangente hiperbólica
	3	Función identidad
Pesos sinápticos	10X	Pesos aleatorios positivos-negativos
	20X	Pesos aleatorios positivos
	30X	Pesos fijos
	XX1	[0, 0.01]
	XX2	[0, 0.10]
	XX3	[0, 0.25]
	XX4	[0, 0.50]
	XX5	[0, 1.00]

Adicionalmente, la barra de configuración de parámetros incluye tres botones: Abrir, Guardar y Establecer. Los primeros dos botones son para manipular el archivo "Parametros.txt", en el cual pueden ser almacenados todos los parámetros de configuración HOG y RNA, así como los correspondientes a la ventana de detección (mismos que se incluyen en la segunda pestaña de la interfaz y de los cuales se hablará más adelante), para posteriormente poder cargarlos mediante el botón "Abrir". Por su parte, el botón "Establecer" fija los parámetros capturados o cargados desde archivo dentro del sistema para su procesamiento.

4.3.2.2 Panel de procesamiento

El panel de procesamiento permite al usuario seleccionar el proceso a ejecutar, estos procesos están disponibles a través de pestañas y son la pestaña "HOG", la pestaña "Reconocimiento" y la pestaña "Visor".

4.3.2.2.1 Pestaña "HOG"

La pestaña HOG (ver Figura 4.10) permite al usuario aplicar el algoritmo HOG a una imagen positiva y a una negativa y, con la finalidad de poder realizar un análisis, acceder a los resultados de cada una de las fases que lo integran. Específicamente las etapas observables son el gradiente de la imagen, tanto magnitud como ángulos y la agrupación de orientaciones en la cual se agrupan las orientaciones para formar las celdas con los histogramas de gradientes orientados, esto debido a que por la naturaleza de la obtención y normalización de los bloques de características no es posible obtener alguna información gráfica de estas fases ya que, como se detalló en capítulos anteriores, estos módulos son descritos por vectores numéricos.

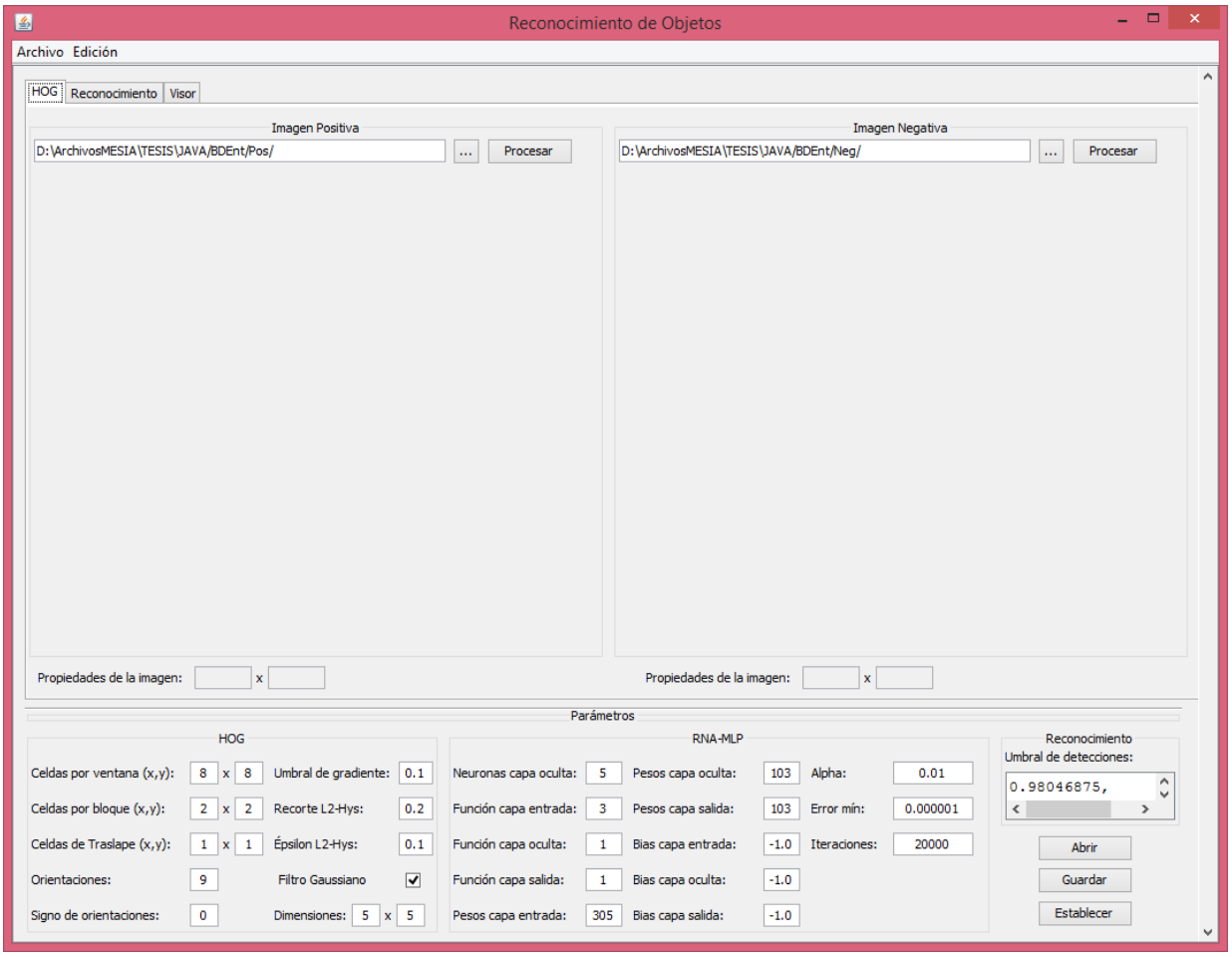

Figura 4.10 Ventana del módulo de extracción de características HOG.

La Figura 4.11 muestra los resultados de las fases del algoritmo HOG aplicado a una imagen de la clase Rostros y una imagen Negativa. Para procesar una imagen es necesario abrirla desde la ventana de selección de archivos que se muestra al hacer clic izquierdo sobre el botón "…" correspondiente a cada panel, una vez seleccionada la imagen es suficiente con hacer clic izquierdo sobre el botón "Procesar", los formatos aceptados por el sistema son: JPG, PNG y/o GIF. En la parte inferior de las imágenes resultantes se presenta el tamaño de la imagen procesada, esta información por sencilla que parezca resulta útil ya que puede ser usada al definir los parámetros de extracción de características como por ejemplo, el número de celdas en cada eje. Para cada panel, donde se sugiere que se procese una imagen positiva (panel izquierdo) y una negativa (panel derecho) existen los botones de selección de archivos y procesamiento como se muestra en la Figura 4.12.

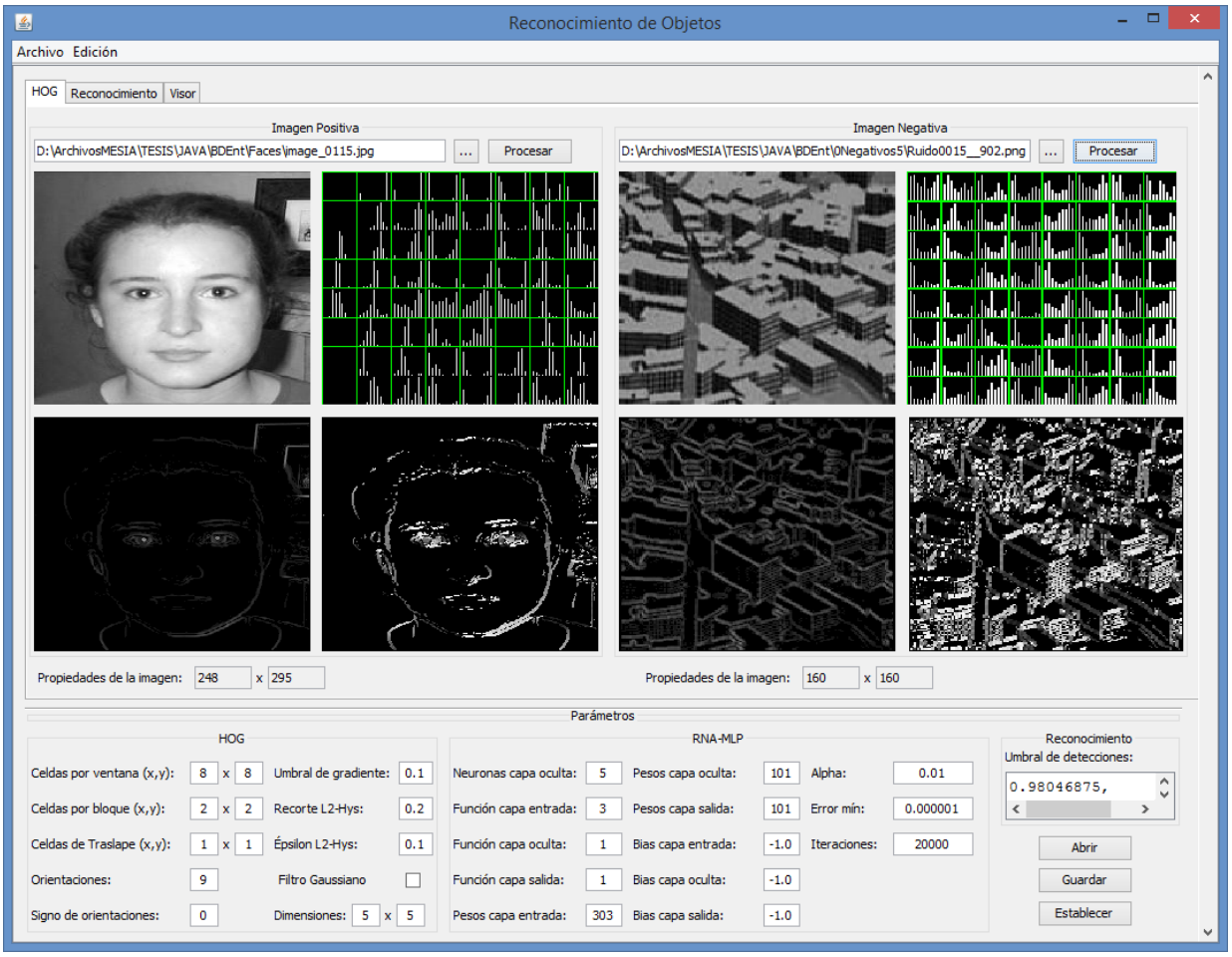

Figura 4.11 Ventana del módulo de extracción de características HOG procesando imágenes.

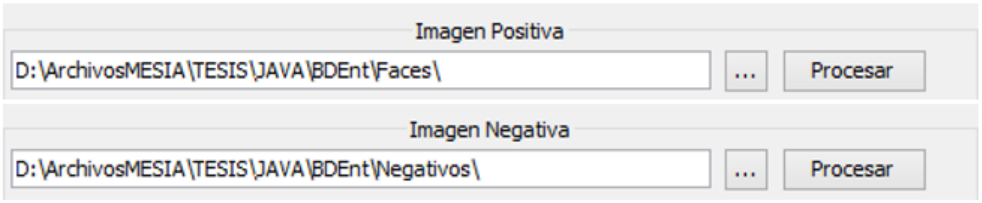

Figura 4.12 Paneles de procesamiento individual de imágenes positivas (superior) y negativas (inferior).

4.3.2.2.2 Pestaña "Reconocimiento"

La pestaña Reconocimiento corresponde al procesamiento general del sistema, las fases que integran este procesamiento son: entrenamiento, operación del sistema y reconocimiento de objetos inmersos en imágenes. La Figura 4.13 muestra la apariencia inicial de la pestaña Reconocimiento.

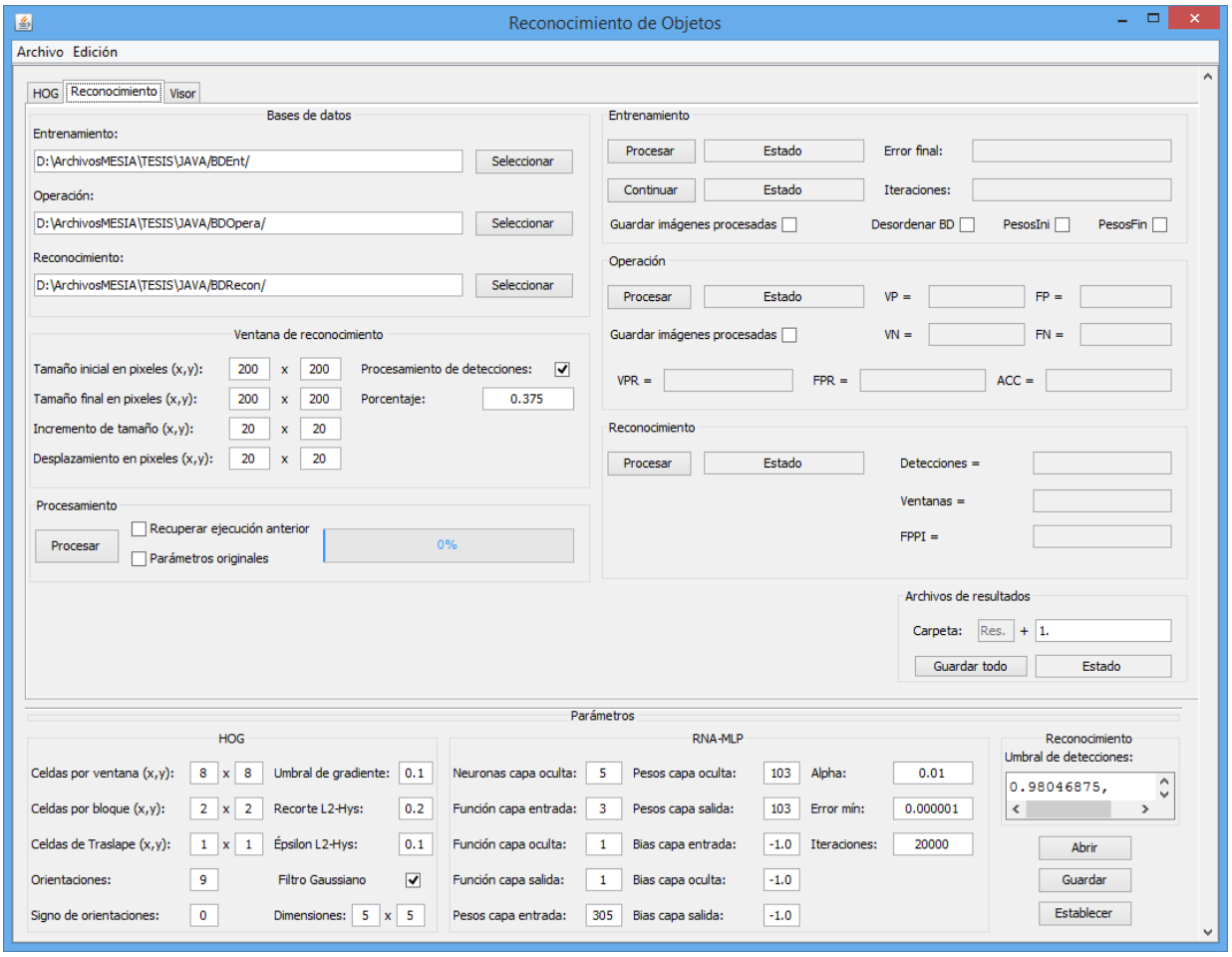

Figura 4.13 Ventana de procesamiento del sistema de reconocimiento de objetos.

Antes de iniciar la operación de reconocimiento se deben elegir las carpetas donde se encuentran las bases de datos. Esta tarea se lleva a cabo a través del botón "Seleccionar" correspondiente a cada selector según se muestra en la sección "Bases de datos" de la pestaña de procesamiento del sistema "Reconocimiento" (Figura 4.14), al hacerlo se muestra el selector de archivos correspondiente al sistema operativo.

Al iniciar la ejecución del sistema se crean las carpetas: "BDEnt", "BDOpera" y "BDRecon", donde se sugiere se almacenen las bases de datos de los procesos de entrenamiento, operación y reconocimiento respectivamente, sin embargo el usuario puede seleccionar las carpetas que desee o donde existan resultados de ejecuciones anteriores, la única condición que debe ser cumplida es que se encuentren en el mismo directorio.

Figura 4.14 Sección Base de datos de la pestaña Reconocimiento.

Para un procesamiento de la **fase de entrenamiento** se deben seleccionar tanto la carpeta de la base de datos de las imágenes negativas, como la carpeta de imágenes positivas, si se selecciona una o ninguna carpeta, el sistema elige como base de datos la carpeta predetermina y toma de ahí todas las carpetas que se encuentren en ella. La **fase de operación** procesa imágenes con el fin de determinar a qué clase pertenecen, en este punto las imágenes por procesar deben ser similares en cuanto a la relación de aspecto a las imágenes utilizadas para la fase de entrenamiento. La selección de las carpetas se realiza de la misma forma que en la fase anterior, basta con seleccionar las carpetas de clases positivas y negativas. En cuanto a la **fase de reconocimiento** se debe seleccionar la carpeta o directorio en donde se encuentren las carpetas con imágenes de dimensiones desconocidas, en donde se realizará un barrido de las mismas con diferentes ventanas de reconocimiento, ventanas que tendrán como propiedades las características que sean definidas en la sección "Ventana de reconocimiento" (Figura 4.15) de esta misma pestaña.

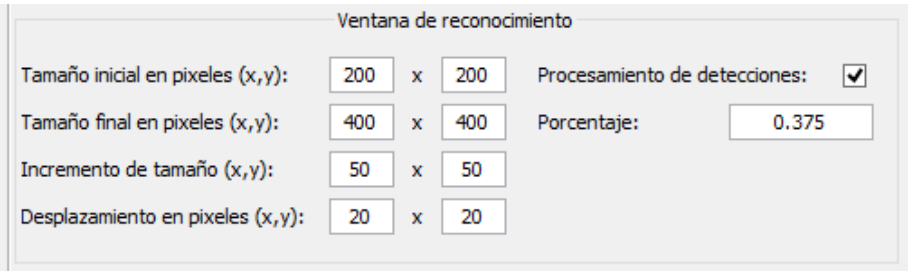

Figura 4.15 Sección Ventana de reconocimiento de la pestaña Reconocimiento.

Las propiedades de la "Ventana de reconocimiento" consisten básicamente en las dimensiones de la ventana inicial y de la ventana final, el incremento de tamaño en pixeles para llegar desde el tamaño de ventana inicial al tamaño de ventana final y el desplazamiento de la ventana de detección. Además, en esta sección son agregados dos parámetros más: la selección de un procesamiento de detecciones y su porcentaje. El procesamiento de detecciones consiste en determinar si dos o más detecciones corresponden al mismo objeto y, de ser cierto, el algoritmo elimina las detecciones que se encuentren, hasta el valor de porcentaje establecido, más cercanas a la ventana de detección centrada en la imagen.

Una vez establecidos los parámetros correspondientes y siendo elegidas las bases de datos por procesar, se cuenta con los paneles de ejecución de las fases de procesamiento como se muestra en la Figura 4.16.

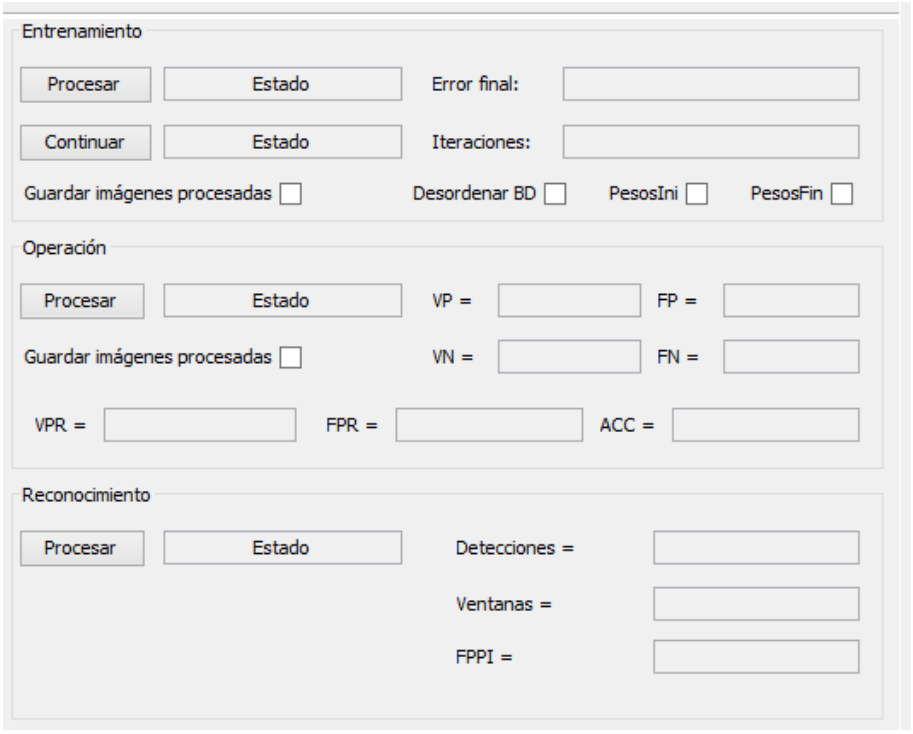

Figura 4.16 Sección de paneles de ejecución de la pestaña Reconocimiento.

El botón "Procesar" de la fase de entrenamiento genera una nueva red neuronal considerando los parámetros de la barra de "Parámetros de configuración", entrena dicha red neuronal con la base de datos y al finalizar indica el error final obtenido. La caja de selección "Guardar imágenes procesadas" disponible tanto en la sección de entrenamiento y de operación define si serán almacenadas las imágenes resultantes de la extracción de características (como las mostradas en la pestaña "HOG"), en la carpeta "*.../Resultados/img2*". Se tiene para esta misma fase la opción de desordenar la base de datos, es decir, mezclar tanto imágenes positivas como imágenes negativas en el orden en que será entrenada la red neuronal. También se tiene la opción de entrenar la red con los pesos iniciales o con los pesos finales (al procesar un número de iteraciones) de algún procesamiento anterior, siempre y cuando los parámetros de configuración sean los mismos. Por otra parte, el botón "Continuar" lleva a cabo la continuación de un procesamiento anterior, es decir, si una primera ejecución terminó a las 1000 iteraciones (sin, por ejemplo, haber obtenido el error deseado), el sistema es capaz de retomar este procesamiento y continuar a partir de la iteración 1001.

En la fase de operación se encuentra el botón "Procesar" el cual lleva a cabo la ejecución del reconocimiento de objetos en imágenes, las cuales son una sola ventana de detección del mismo tamaño que la imagen en proceso. Una vez terminado el procesamiento de este apartado, el

sistema muestra los resultados de la operación, datos como el número de verdaderos positivos y falsos positivos entre otros, son mostrados en pantalla. Así como otros resultados que serán explicados en el siguiente capítulo y que serán utilizados para determinar el rendimiento del sistema.

Por su parte, la fase de reconocimiento procesará las imágenes con objetos inmersos en éstas, esta fase entregará como resultados el número total de detecciones en todas las imágenes, el número de ventanas procesadas y un aproximado del número de falsos positivos por ventana.

Finalmente se encuentran en esta pestaña un panel para el procesamiento de las tres fases descritas anteriormente y un panel para guardar los archivos generados por el sistema en una determinada carpeta (Figura 4.17). El primero realiza una ejecución de todas las fases sin tener que procesar fase por fase, permite seleccionar si se requiere recuperar o continuar con una ejecución anterior y además, permite seleccionar si se realizará el procesamiento con los parámetros actualmente establecidos en la barra de "Parámetros de configuración" o con los parámetros originales de la ejecución anterior (desde un archivo "Parametros.txt"). El segundo panel permite guardar los archivos creados por el sistema en cada fase de procesamiento, la Tabla 4.11 muestra los archivos o carpetas generados por el sistema y la descripción de los mismos.

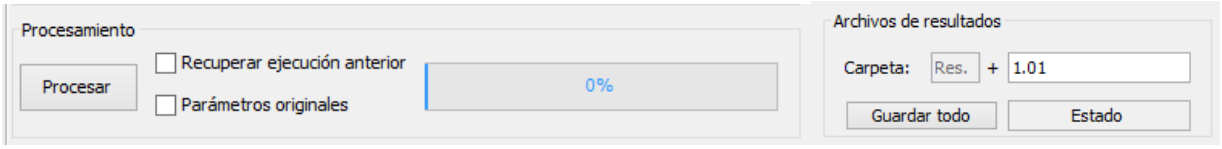

Figura 4.17 Sección de paneles procesamiento general y almacenamiento de archivos de la pestaña Reconocimiento.

Tabla 4.11 Archivos generados por el sistema de reconocimiento de objetos.

Nombre	Tipo	Contiene
Parametros.txt	Archivo	Parámetros de configuración HOG, RNA y Reconocimiento.
BaseDatosEnt.txt	Archivo	Nombres de las imágenes utilizadas para la fase de entrenamiento.
HoGEnt.txt	Archivo	Vectores de características de la base de datos de entrenamiento.
Backp.PesosIni.txt	Archivo	Pesos iniciales de todas las neuronas de la RNA.
Backp.PesosFin.txt	Archivo	Pesos finales de todas las neuronas de la RNA.
BaseDatosOpera.txt	Archivo	Nombres de las imágenes utilizadas para la fase de operación.
HoGOpera.txt	Archivo	Vectores de características de la base de datos de operación.
TablaFaseOpera.csv	Archivo	Tabla de resultados de la fase de operación.
BaseDatosRecon.txt	Archivo	Nombres de las imágenes utilizadas para la fase de operación.
TablaFaseRecon.csv	Archivo	Tabla de resultados de la fase de reconocimiento.
/img/	Carpeta	Imágenes de resultados de la fase de reconocimiento.
/img2/	Carpeta	Imágenes resultantes de las fases de entrenamiento y operación.

La Figura 4.18 muestra esta misma pestaña después de realizar el procesamiento de alguna información.

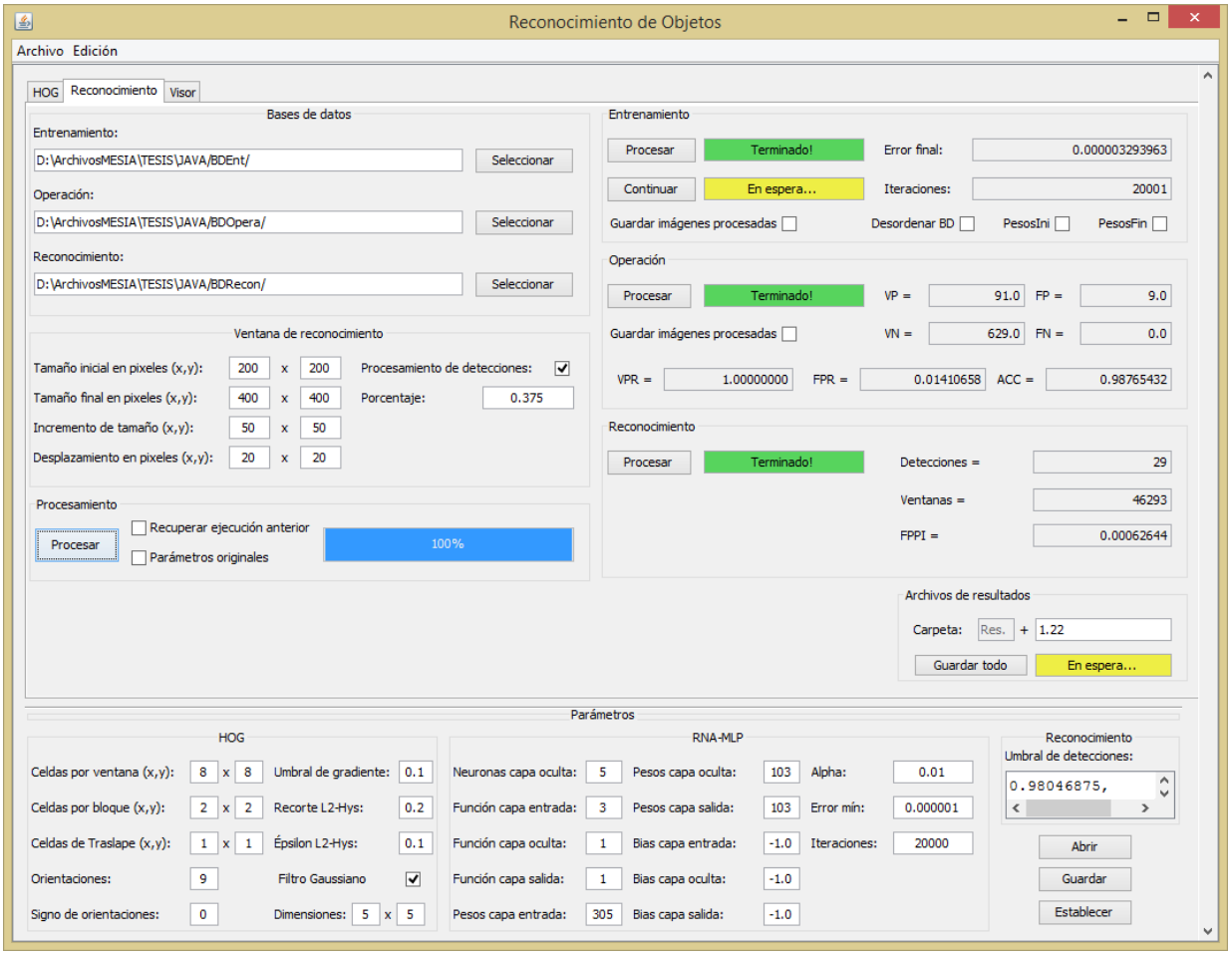

Figura 4.18 Ventana de la pestaña Reconocimiento despues de procesar información.

4.3.2.2.3 Pestaña "Visor"

Esta pestaña tiene como objetivo principal funcionar como un visor de imágenes de resultados, mostrará las imágenes que hayan sido procesadas por el sistema en la fase de reconocimiento, las cuales incluirán las detecciones de los objetos inmersos en ellas. No es necesario seleccionar la carpeta donde se encuentren dichas imágenes ya que el sistema está configurado para tomarlas desde la carpeta correspondiente. En la figura 4.19 se muestra esta pestaña después de procesar algunas imágenes.

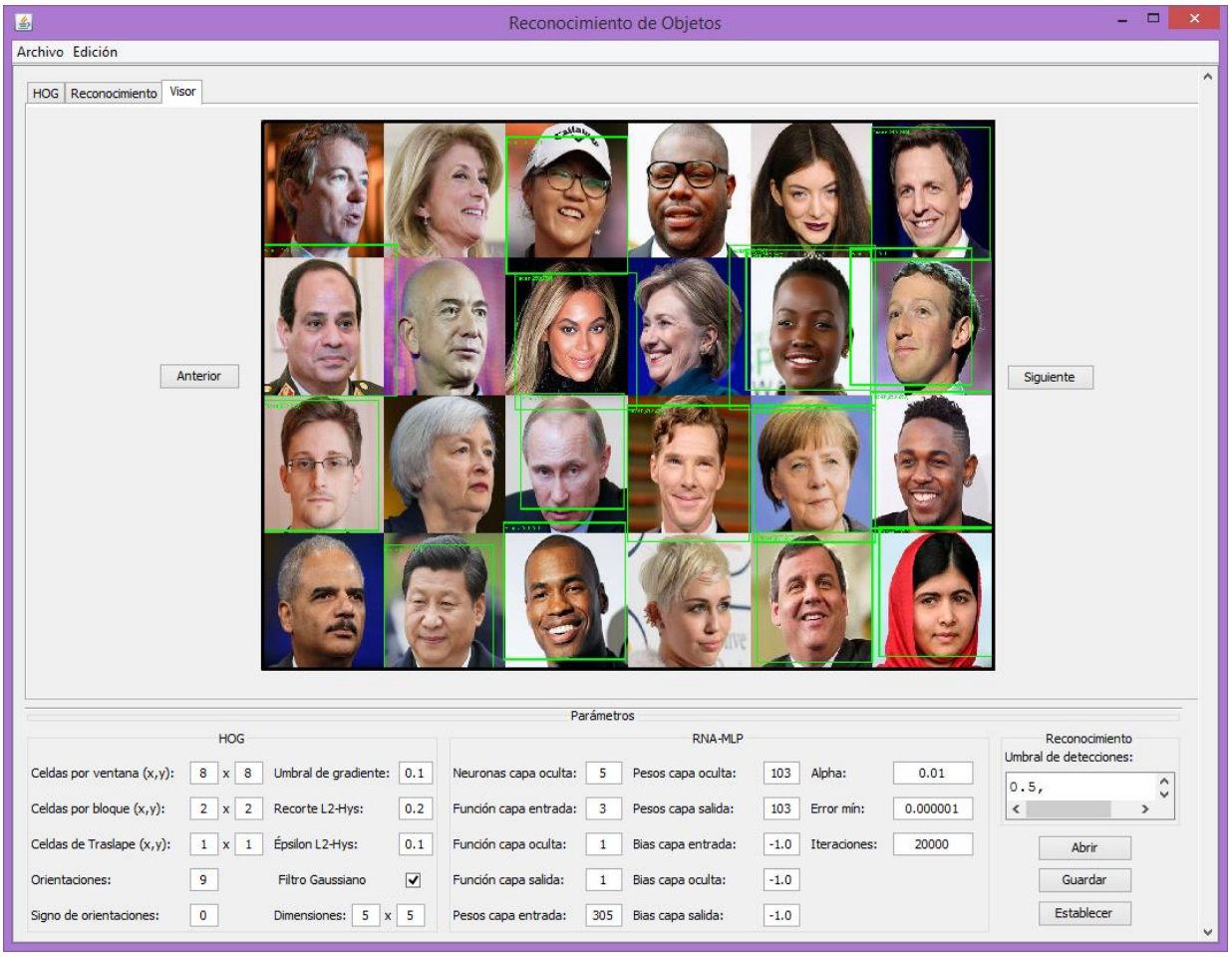

Figura 4.19 Ventana de la pestaña Visor despues de procesar información.

Capítulo 5

Resultados experimentales y discusión

En este capítulo se presenta el funcionamiento de S-ROHM mediante una serie de experimentos que permiten determinar su desempeño. En el primer experimento se estudia el rendimiento del sistema para varios conjuntos de imágenes diferentes, denominadas clases de objetos, donde cada conjunto de imagen es evaluada de manera individual para observar el comportamiento del sistema cuando se le presenta una sola clase a reconocer o clasificar. En el segundo experimento se realiza una comparación con el sistema HOG desarrollado por Dalal y Triggs [1]. Finalmente, en un tercer experimento se estudia el comportamiento del sistema cuando se le configura para clasificar múltiples clases al mismo tiempo.

5.1 Aspectos generales de la experimentación

Las sub-secciones siguientes describen los aspectos a considerar en los experimentos realizados para determinar el desempeño del sistema propuesto. En general, se describen las bases de datos utilizadas en los experimentos, la metodología de entrenamiento del sistema, así como también la metodología aplicada para medir el rendimiento del S-ROHM.

5.1.1 Base de datos del sistema S-ROHM

Para realizar los experimentos se tomaron diferentes conjuntos de imágenes de la popular base de datos Caltech de Fei-Fei *et al* [124]. Esta base de datos contiene 101 clases diferentes con un conjunto variante en tamaño de cada una. Se tomaron sólo algunas de ellas para realizar los experimentos, en la Figura 5.1 se puede observar algunas imágenes tomadas de esta base de datos para el conjunto de imágenes positivas, mientras que en la Figura 5.2 se muestran algunas del conjunto de imágenes negativas. Cabe señalar que para el conjunto de imágenes negativas se recopilaron imágenes tanto de la base de datos Caltech como de la base de datos INRIA para detección de personas[1], entre otras. Se generaron 1179 imágenes negativas para el conjunto de entrenamiento del sistema (I_{-}^{train}) y 1179 imágenes negativas para el conjunto de prueba o fase de operación del sistema (I_{-}^{test}). El conjunto en general de imágenes negativas está compuesto por imágenes de diferentes dimensiones, color RGB y formato jpg o png.

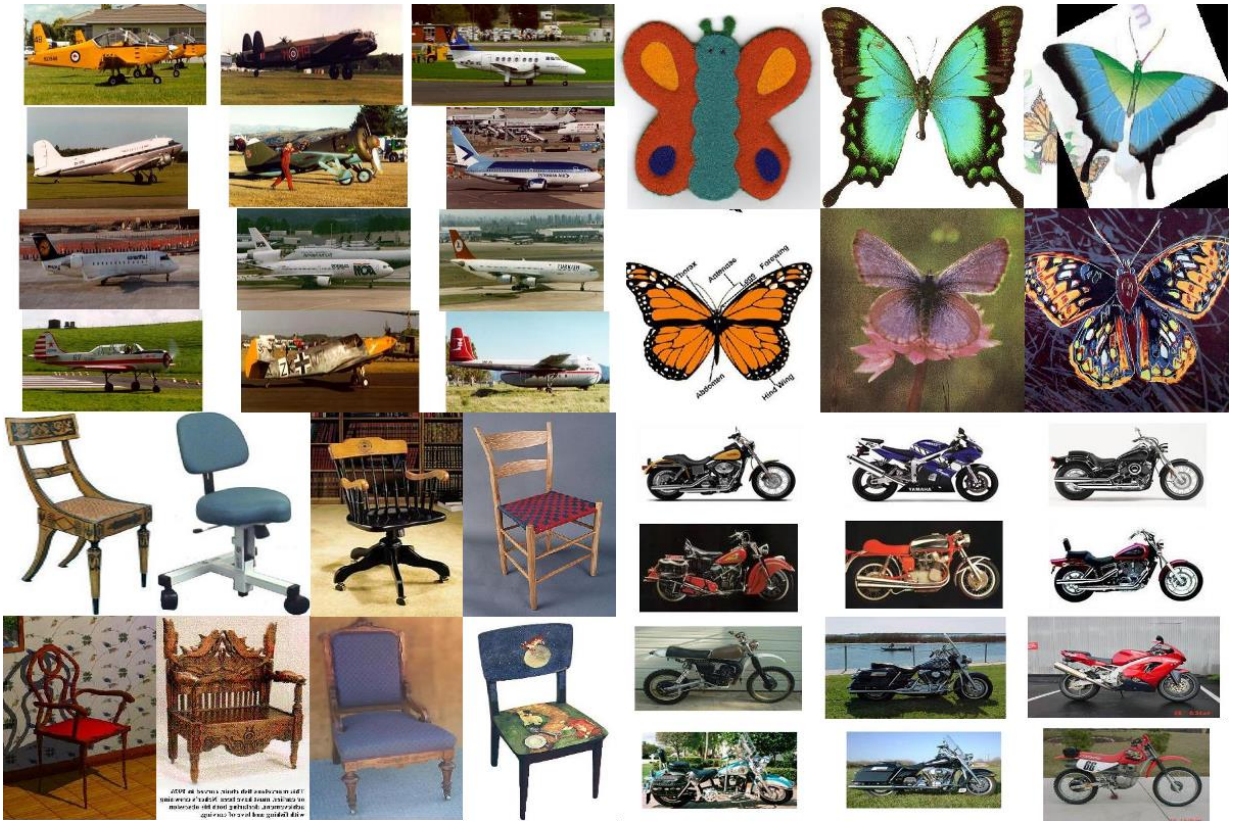

Figura 5.1 Imágenes positivas de algunas categorías de la base de datos Caltech 101.

Tanto las imágenes positivas de entrenamiento como las negativas son de un tamaño mediano y cuentan con diversas resoluciones, las cuales dependen de la forma del objeto. Por ejemplo, la clase *Airplane* cuenta con un tamaño de imagen promedio de 390×150 pixeles, en cambio, la clase *Chair* tiene un promedio de imagen de 200×300 pixeles.

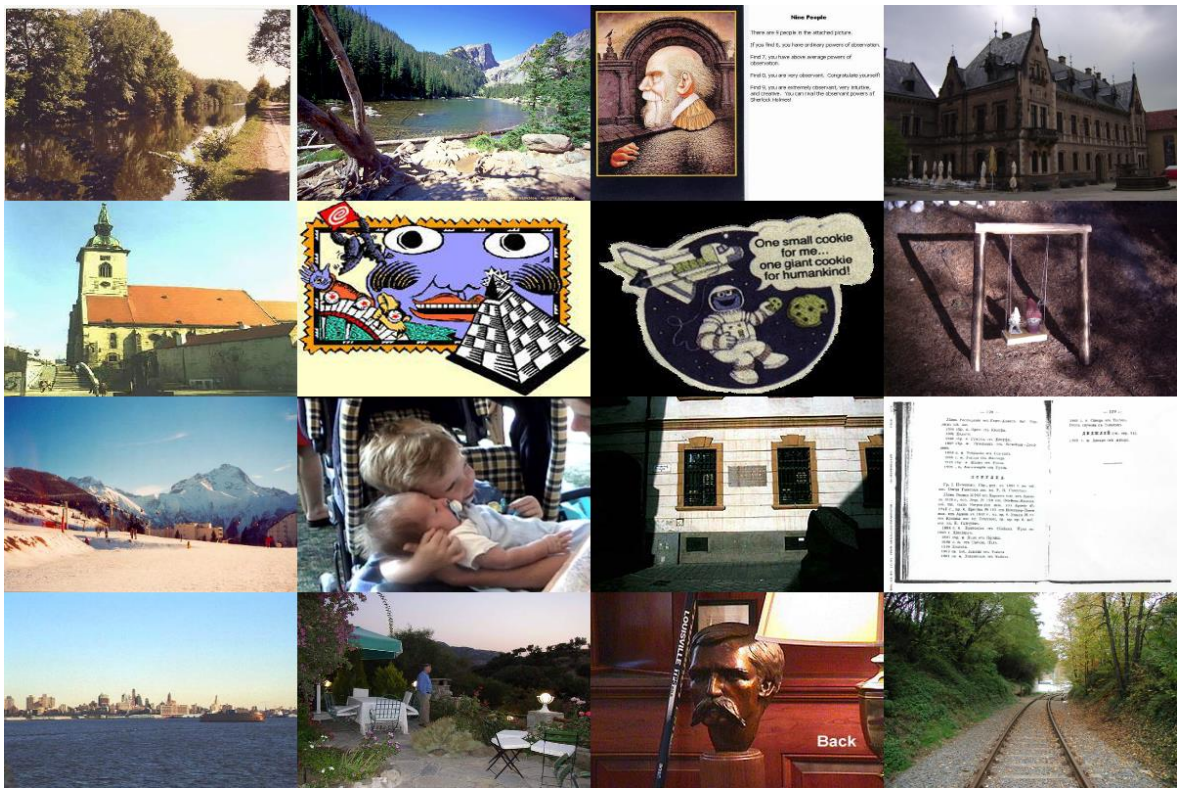

Figura 5.2 Imágenes negativas de la base de datos Caltech 101.

En la Tabla 5.1 se detalla la cantidad de imágenes que contiene cada clase (conjunto de imágenes positivas W_+).

Tabla 5.1 Detalles de las bases de datos utilizadas.

Clase	**Cantidad de imágenes**	
	Entrenamiento W_+^{train}	**Prueba W_+^{test}**
Airplane	200	200
Butterfly	40	40
CarSide	86	86
Chair	30	30
ElectricGuitar	30	30
Faces	100	100
Helicopter	40	40
Horses	85	85
Ketch	50	50
Laptop	40	40
Motorikes	200	200
PersonINRIA	2416	1126
Piano	45	45
Revolver	40	40
SoccerBall	30	30
Watch	100	100

Cada clase de imágenes positivas contiene dos conjuntos de imágenes, es decir, las imágenes del conjunto de entrenamiento son totalmente diferentes a las imágenes del conjunto de prueba de cada clase. En el caso de la clase "PersonINRIA" se tomaron las imágenes de la base de datos INRIA [1] en sus totalidad, tanto imágenes positivas como negativas, esto con el fin de comparar el rendimiento de S-ROHM con el trabajo presentado por Dalal y Triggs. Las imágenes positivas tienen una razón de aspecto de acuerdo a su forma de objeto, por lo que las dimensiones de cada imagen varían ligeramente pero mantienen cierta correspondencia, a diferencia del sistema propuesto por Dalal y Triggs, el sistema S-ROHM permite incluir en el procesamiento imágenes con diferentes tamaños, escalas o dimensiones tanto para la misma clase en proceso o de manera general para cualquier forma de objeto, esto con el conocimiento de que, estandarizar tamaños, márgenes y dimensiones puede o no mejorar el rendimiento del sistema (Dalal y Triggs plantean que al definir sus parámetros característicos de su ventana de detección para imágenes positivas en la detección de personas, se aumenta el rendimiento de su sistema [1]). Así mismo, cada imagen positiva en ambos conjuntos representa una ventana de detección de nuestro sistema, es decir, no se encuentran inmersas en un fondo (*background*), sin embargo, los márgenes que rodean al objeto pueden contener o no algún tipo de fondo ya que las imágenes recopiladas no han tenido proceso de segmentación alguno como puede observarse en la Figura 5.1.

La base de datos INRIA consta de: 1218 imágenes negativas de entrenamiento (llámense II_{-}^{train}), 2416 ventanas positivas de entrenamiento (llámense IW_{+}^{train}), 453 imágenes negativas de prueba (llámense II_{-}^{test}) y por último, 1126 ventanas positivas de prueba (llámense IW_{+}^{test}). Las ventanas de entrenamiento que incluye la base de datos INRIA se encuentran estandarizadas con unas dimensiones de 96×160 pixeles y tanto los conjuntos de entrenamiento y prueba contienen únicamente imágenes de personas con 16 pixeles de borde o fondo, como se explica en [1].

5.1.2 Entrenamiento del sistema S-ROHM

Para el entrenamiento del sistema se utilizaron los conjuntos de entrenamiento I_{-}^{train} y W_{+}^{train} (se excluye a la clase "PersonINRIA"). Primero, del conjunto de imágenes negativas tanto de entrenamiento como de prueba fueron extraídas ventanas aleatorias de cada una de las imágenes con el fin de aumentar el conjunto y añadir robustez al sistema. Se generaron 1743 ventanas para el conjunto I_{-}^{train} y 1909 ventanas para el conjunto I_{-}^{test}. Ambos subconjuntos de ventanas (llámese W_{-}^{train} y W_{-}^{test} para las ventanas de entrenamiento y prueba respectivamente) fueron añadidos a su respectivo conjunto, dando los totales de 2922 y 3088 elementos para el conjunto de imágenes negativas de entrenamiento (I'^{train}_{-}) y de prueba (I'^{test}_{-}) respectivamente. Una vez obtenidos estos conjuntos de datos, son presentados al sistema los conjuntos I'^{train}_{-} y W_{+}^{train} para calcular las características HOG y posteriormente aplicar la fase de entrenamiento usando el clasificador MLP. Entonces, se ejecuta la fase de operación o prueba del sistema a los conjuntos I'^{test}_{-} y W_{+}^{test} con la finalidad de evaluar el rendimiento del sistema utilizando cada una de las

imágenes de dichos conjuntos como ventana de detección. Finalmente, se ejecuta la fase de reconocimiento en la cual se utilizan algunas imágenes con tamaños mayores a los 512×512 pixeles, en las cuales se realiza un barrido para ventanas de detección a diferentes escalas.

Para el caso de detección de personas de la base de datos INRIA, cabe señalar que únicamente se utilizaron las imágenes contenidas en dicha base de datos. Se usó el procedimiento estándar de entrenamiento para éste conjunto de datos, fueron extraídas un promedio de 10 ventanas aleatorias del conjunto de imágenes negativas, del tamaño predefinido de la ventana de detección canónica (96×160 pixeles), haciendo un total de 14948 ventanas/imágenes de detección (llámense II'^{train}_{-}), incluyendo el conjunto original de imágenes negativas. Las imágenes positivas de entrenamiento que se incluyen en la base de datos INRIA ya se encuentran estandarizadas en un tamaño de 96×160 pixeles, por lo que solo fue necesario presentárselas al sistema para los procesos de extracción de características y entrenamiento. En la fase de operación o prueba del sistema S-ROHM se tomó como base las imágenes negativas de prueba II^{test}_{-}, de las cuales se realiza una extracción de ventanas de detección a diferentes escalas (el número de ventanas depende de las dimensiones de la imagen, aproximadamente se obtienen 536 ventanas en promedio). Para el conjunto de imágenes positivas de prueba fue necesario hacer un escalamiento de dicho conjunto ya que estas imágenes se encontraban en un tamaño de 70×134 pixeles.

5.1.3 Evaluación del sistema S-ROHM

Para medir el rendimiento del sistema presentado en este trabajo de tesis, se utilizan los siguientes términos:

- **Verdadero Positivo** (VP): Si la clase de salida es p' y pertenece realmente a la clase p, se tiene una clasificación exitosa.
- **Verdadero Negativo** (VN): Si la clase de salida es n' y corresponde realmente a la clase n, se tiene un rechazo exitoso.
- **Falso Positivo** (FP): Si la clase resultante es p' pero en realidad pertenece a la clase n entonces se tiene una falsa detección.
- **Falso Negativo** (FN): Si la clase resultante es n' pero en realidad pertenece a la clase p entonces se tiene un falso rechazo.

De estos términos obtenemos las siguientes ecuaciones que aportan más información acerca del sistema de reconocimiento de objetos. La ecuación 5.1 muestra el término conocido como razón de verdaderos positivos o sensibilidad del sistema:

$$VPR = \frac{VP}{VP + FN} \tag{5.1}$$

A la ecuación 5.2 se le conoce como razón de falsos positivos e indica la proporción de objetos clasificados erróneamente.

$$FPR = \frac{FP}{FP + VN} \tag{5.2}$$

En la ecuación 5.3 se presenta el término precisión o *accuracy* (ACC) la cual puede evaluar la tendencia del sistema para acertar verdaderos positivos:

$$ACC = \frac{VP + VN}{P\# + N\#} \tag{5.3}$$

donde, $P\#$ representa la cardinalidad del conjunto de ventanas de W_{+}^{test}, mientras que $N\#$ representa la cardinalidad del conjunto de imágenes negativas y su subconjunto de ventanas $I_{-}^{test} \cup W_{-}^{test}$. La correspondiente tasa de fallos (*miss rate*) queda definida como:

$$MR = 1 - VPR \tag{5.4}$$

Por último se presenta un cálculo sencillo que indica la cantidad de fallos por ventana de detección, falsos positivos por ventana (FPPW – *False positive per window*):

$$FPPW = \frac{FP}{N} \tag{5.5}$$

Así mismo, para complementar la evaluación del sistema, se recalca que el clasificador propuesto MLP regresa un valor real para cada ventana de detección dada, el cual es umbralizado con un(os) valor(es) fijo(s) u con la finalidad de poder decidir si es o no un objeto perteneciente a determinada clase. Por lo tanto, MR y $FPPW$ son funciones dependientes de u, lo cual permite graficar las curvas de evaluación ROC (ecuación 5.6), las cuales muestran la compensación (del inglés *tradeoff*) entre la tasa de fallos y los falsos positivos por ventana para cada umbral [125].

$$E(u) = \big(FPPW(u), MR(u)\big) \tag{5.6}$$

5.2 Experimento 1. Desempeño del S-ROHM para clasificación binaria

Este primer experimento consistió en presentar individualmente al S-ROHM los conjuntos o clases de imágenes de entrenamiento con la finalidad de adaptar al sistema para su posterior reconocimiento. Para este fin, se pueden distinguir dos conjuntos de imágenes, el primer conjunto (imágenes negativas) como se ha mencionado contiene imágenes en las cuales no existe algún objeto de los predispuestos a clasificar. Cabe señalar que este mismo conjunto se utilizó para todo el desarrollo del experimento, es decir, este conjunto fue presentado como conjunto negativo de todas las clases. En este grupo existen imágenes que representan paisajes, formas o en general entornos aleatorios. El segundo conjunto de imágenes corresponde a cada una de las clases descritas en la Tabla 5.1 y que representan los objetos que deben ser reconocidos.

La Tabla 5.2 muestra los resultados obtenidos cuando se presentan por separado clases de objetos diferentes y con cierta configuración en cuanto al número de celdas de la rejilla del algoritmo HOG. Los parámetros HOG que se mantienen constantes son: tamaño de bloques de 2×2 celdas, traslape entre celdas de 1×1 celdas, 9 orientaciones (*bins*). Los parámetros RNA-MLP que fueron utilizados para la clasificación de los objetos son: 5 neuronas en la capa oculta, función sigmoidea en la capa oculta y de salida, pesos aleatorios en el rango de $[-0.25,0.25]$, *bias* o umbral de neuronas de -1.0, $\alpha = 0.01$ (tasa de aprendizaje) y en todos los casos fueron suficientes 20000 iteraciones en la fase de entrenamiento de la red neuronal.

Tabla 5.2 Resultados del experimento 1 (clasificación binaria).

	Rendimiento					**Parámetros**	
Clase	**VPR**	**FPR**	**ACC**	**FP**	**FN**	**# Celdas**	**Umbral**
Airplane	0.955	0.00097	0.9963	3	9	16x8	0.60
Airplane	0.945	0.00060	0.9960	2	11	8x8	0.90
Butterfly	0.45	0.00130	0.9916	4	22	12x8	0.90
Butterfly	0.45	0.00190	0.9910	6	22	8x8	0.90
Carside	0.767	0.00097	0.9927	3	20	12x8	0.90
Carside	0.732	0.00097	0.9918	3	23	8x8	0.90
Chair	0.133	0.0	0.9916	0	26	8x12	0.90
Chair	0.233	0.00032	0.9923	1	23	8x8	0.90
ElectricGuitar	0.400	0.00032	0.9939	1	18	12x8	0.50
ElectricGuitar	0.400	0.00032	0.9939	1	18	8x8	0.50
Faces	0.960	0.00010	0.9987	0	4	8x16	0.90
Faces	0.760	0.00010	0.9924	0	24	8x8	0.90
Helicopter	0.375	0.00356	0.9884	11	25	16x8	0.20
Helicopter	0.500	0.00453	0.9891	14	20	8x8	0.19
Horses	0.741	0.0	0.9930	0	22	12x8	0.90
Horses	0.823	0.00388	0.9914	12	15	8x8	0.90
Ketch	0.680	0.00226	0.9926	7	16	10x10	0.50
Ketch	0.740	0.00064	0.9952	2	13	8x8	0.50
Laptop	0.650	0.00032	0.9952	1	14	16x16	0.50
Laptop	0.625	0.00064	0.9945	2	15	8x8	0.50
Motorbikes	0.960	0.00032	0.9972	1	8	16x8	0.98
Motorbikes	0.980	0.00001	0.9987	0	4	8x8	0.90
PersonINRIA	0.703	0.00501	0.9931	862	334	8x16	0.90
Piano	0.800	0.00064	0.9964	2	9	12x8	0.50
Piano	0.777	0.0	0.9968	0	10	8x8	0.50
Revolver	0.700	0.00194	0.9942	6	12	16x8	0.50
Revolver	0.675	0.00129	0.9945	4	13	8x8	0.50
Soccerball	0.241	0.00032	0.9926	1	22	12x12	0.90
Soccerball	0.241	0.00064	0.9923	1	22	8x8	0.90
Watch	0.660	0.00226	0.9871	7	34	12x8	0.50
Watch	0.700	0.00259	0.9880	8	30	8x8	0.50

Tomando en cuenta los parámetros de configuración del algoritmo HOG que presentan Dalal y Triggs en su artículo [1], además de las diversas y variadas pruebas experimentales llevadas a cabo para las diferentes clases, se determinó empíricamente que el número de celdas a utilizar y que son mostradas en este trabajo de tesis (véase Tabla 5.2) corresponden a una configuración de 8×8 celdas, 16×8 celdas y 8×16 celdas en la mayoría de los casos. Estas configuraciones son determinadas con base en las dimensiones de los objetos y además, fueron las que obtuvieron los resultados más llamativos. Así mismo, esto explica por qué se presentan un par de resultados para cada clase evaluada en la Tabla 5.2. Sin embargo, no fue aplicada esta misma evaluación a la clase "PersonINRIA", en la cual se utilizó únicamente la metodología aplicada por Dalal y Triggs en su método de detección de personas.

Con la finalidad de mostrar el comportamiento del sistema de reconocimiento de objetos de forma gráfica, en la Figura 5.3 se muestran algunas detecciones sobre imágenes con entornos. Por su parte, en la Figura 5.4 se muestran algunos falsos positivos del sistema.

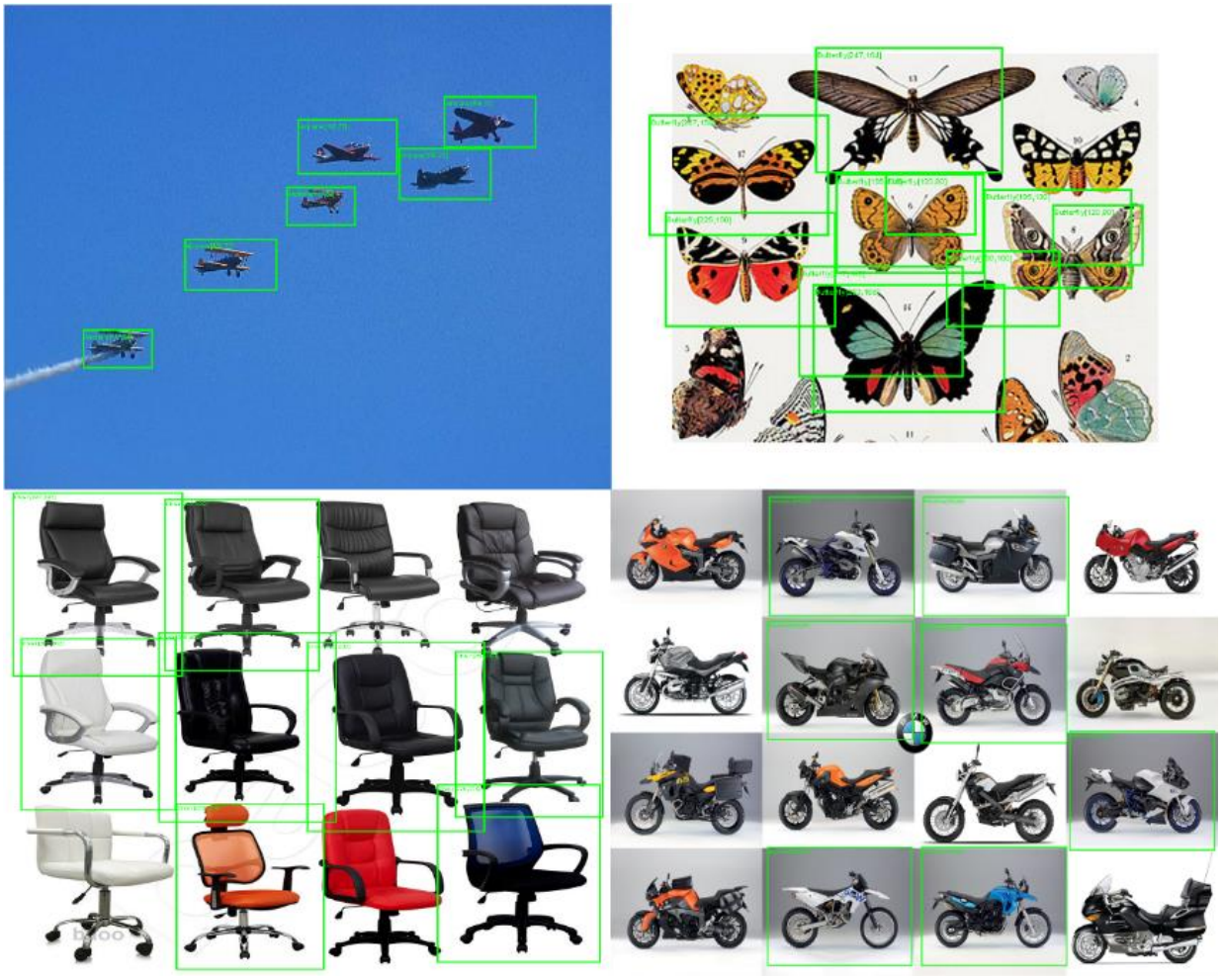

Figura 5.3 Detecciónes para proceso de clasificación binaria en imágenes con entornos completos (se detectan las clases de izquierda a derecha y de arriba abajo: *Airplane, Butterfly, Chair, Motorbikes*).

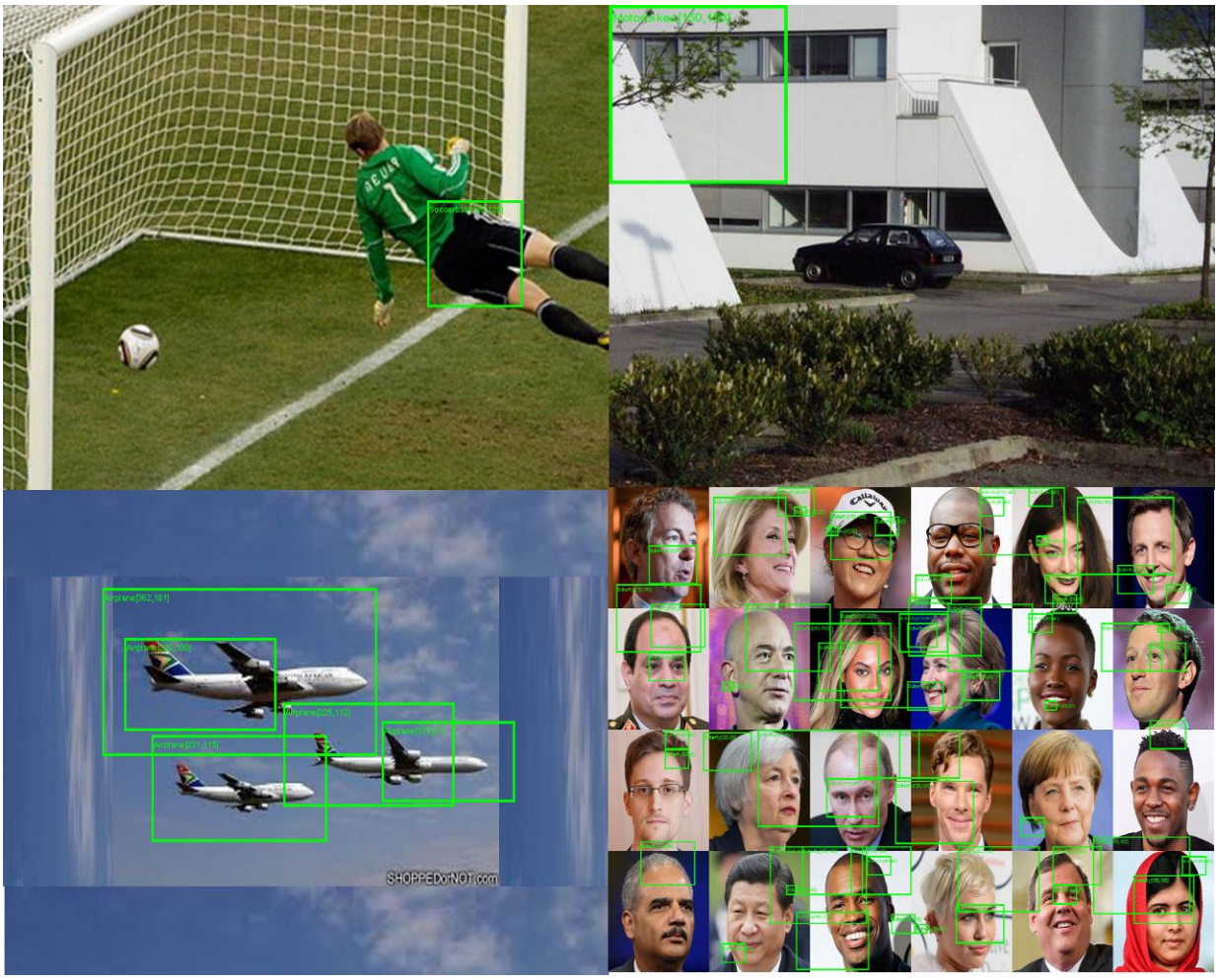

Figura 5.4 Falsos positivos en el proceso de clasificación binaria (se detectan las clases de izquierda a derecha y de arriba a abajo: *SoccerBall*, *Motorbikes*, *Airplane*, *Butterfly*).

Los resultados de la Tabla 5.2 muestran el rendimiento del sistema para ciertos umbrales específicos, las siguientes figuras (Figura 5.5 a Figura 5.8) muestran las curvas de rendimiento (ROCs) para ciertas clases seleccionadas, en donde el procesamiento se realizó mediante un barrido del umbral de detección.

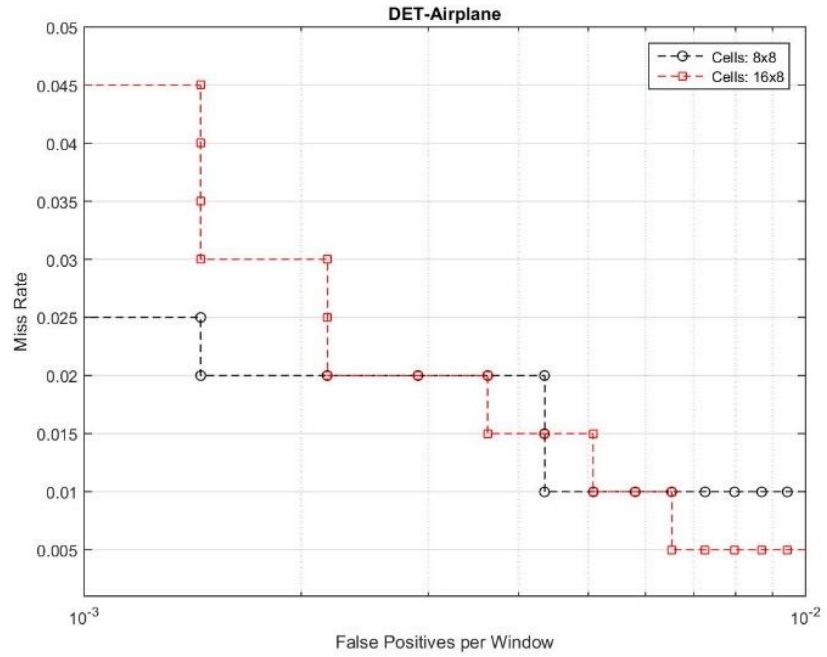

Figura 5.5 Gráfica de evaluación de rendimiento ROC usando escalas logarítmicas (menos es mejor), diferencia entre dos configuraciones de números de celdas por ventana para la clase *Airplane*.

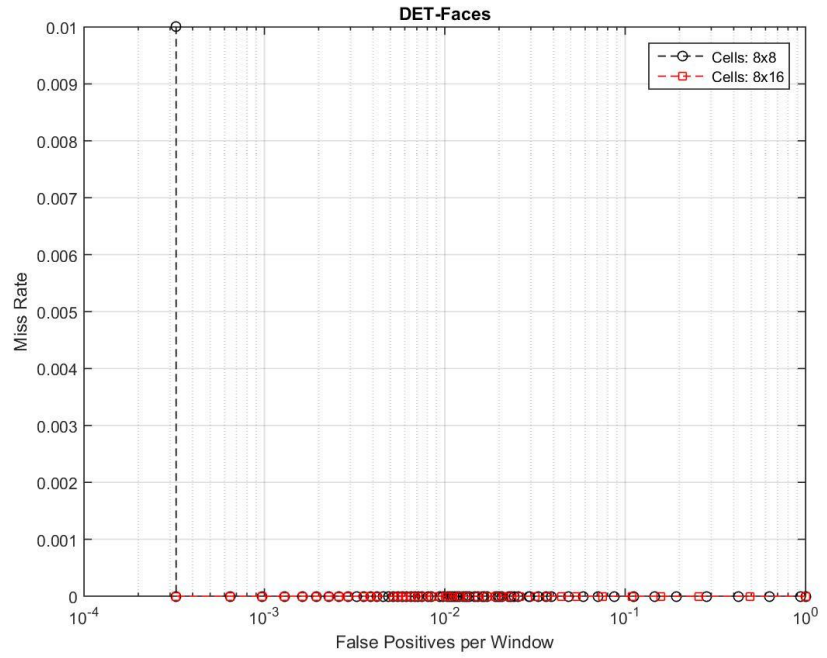

Figura 5.6 Gráfica de evaluación de rendimiento ROC usando escalas logarítmicas (menos es mejor), diferencia entre dos configuraciones de números de celdas por ventana para la clase *Faces*.

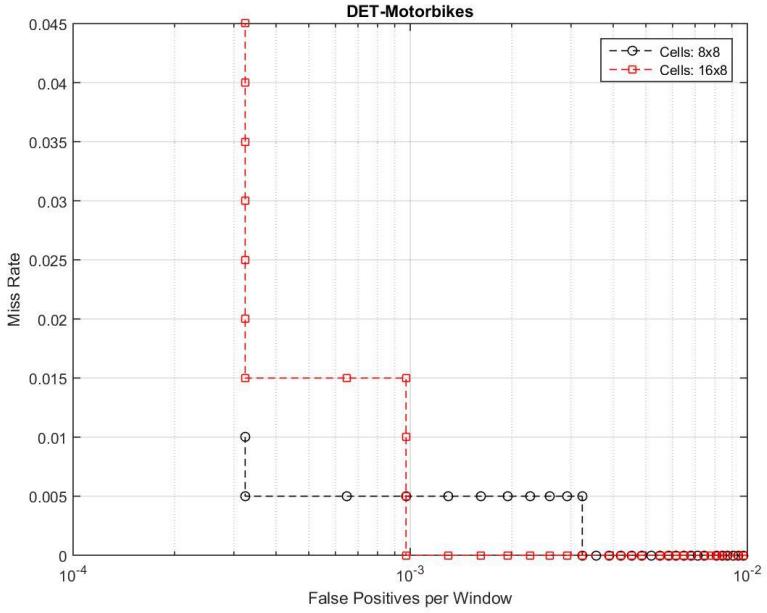

Figura 5.7 Gráfica de evaluación de rendimiento ROC usando escalas logarítmicas (menos es mejor), diferencia entre dos configuraciones de números de celdas por ventana para la clase *Motorbikes*.

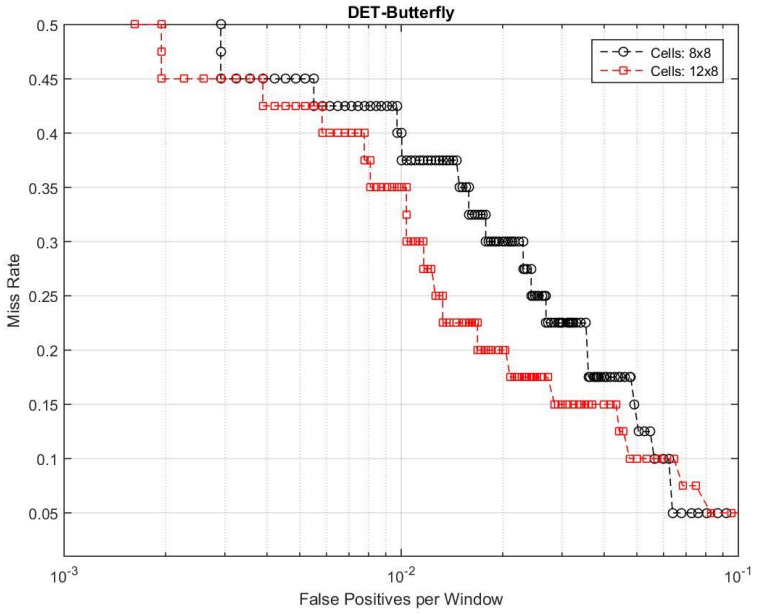

Figura 5.8 Gráfica de evaluación de rendimiento ROC usando escalas logarítmicas (menos es mejor), diferencia entre dos configuraciones de números de celdas por ventana para la clase *Butterfly*.

Particularmente podría parecer que la gran cantidad de variables que componen al sistema son las que afectan el desempeño del mismo, sin embargo, muchas de ellas pueden no afectarlo. Como en el caso de las neuronas de la capa oculta, se hicieron pruebas de 10, 15, 20 neuronas y en todas las pruebas los resultados eran muy similares, variando en 0.5% de $10^{-3}FPPW$, lo que

demostró que trabajar con 5 neuronas era suficiente para tener resultados comparables y con una carga computacional razonable. Sin embargo, un dato que realmente afecta el desempeño del sistema es la inicialización de los pesos sinápticos, se realizaron pruebas con diferentes valores: pesos sinápticos fijos en 1.0, fijos en 0.5, aleatorios de entre [0,1], aleatorios de entre $[-1,1]$, entre otros, y en todos eran resultados muy variados, hasta el punto de no tener una clasificación. Por lo que, analizando esta situación se determinó que valores cercanos a 0 y tomando en cuenta valores positivos y negativos eran los pesos sinápticos adecuados que permitían una clasificación satisfactoria. La función de activación tangente hiperbólica no representó mejora o disminución del rendimiento, esto puede ser comprensible debido a que los valores de salida necesarios para definir la clase perteneciente de la ventana de detección en proceso, se limitan a una umbralización y dependerá totalmente del umbral que se defina. En el caso del *bias* o umbral de las neuronas, no afecta el porcentaje de aciertos su modificación, sin embargo, es necesario incluirlo para que los pesos sinápticos puedan converger en la fase de aprendizaje. El valor de α afecta la velocidad de aprendizaje, a mayor valor, mayor número de iteraciones serán necesarias en diversos casos. En cuanto a los parámetros del extractor de características, dependerán de las propiedades del objeto, sus dimensiones definirán el tipo y características de la rejilla utilizada como: el número de celdas, número de celdas por bloque y número de celdas traslapadas. En ciertos casos o para ciertas clases se podría utilizar un mayor número de *bins* (18), ya que si no, puede no existir una clasificación satisfactoria, por lo que, como se había mencionado antes, no se puede establecer un regla general que permita clasificar todo tipo de objetos.

Cabe señalar que se realizaron diferentes pruebas variando los parámetros mencionados, sin embargo, los resultados que resultaron más competitivos fueron los que se reportan en la Tabla 5.2. Si bien Dalal y Triggs mencionan en su artículo [1] que ciertas configuraciones (como el número de orientaciones por ejemplo) son las ideales para ciertos tipos de objetos, en los resultados mostrados en el presente trabajo de tesis se pudo demostrar que es posible una clasificación con un rendimiento aceptable, utilizando una configuración estándar del sistema S-ROHM (véase Tabla 5.2). Sin embargo, se le aconseja al lector que no se tomen estos resultados presentados como del todo óptimos y únicos para los objetos descritos.

5.3 Experimento 2. Desempeño del S-ROHM para reconocimiento de personas con la base de datos INRIA

Las pruebas realizadas en este experimento consistieron en comparar el rendimiento del S-ROHM con el sistema de reconocimiento de personas propuesto por Dalal y Triggs en [1]. Para esto, fue realizado el proceso de entrenamiento y prueba descrito en la Sección 5.1.2. Los resultados calculados por el sistema propuesto se pueden apreciar en la gráfica (a) de la Figura 5.9, por otra parte, la gráfica (b) muestra el rendimiento de otros sistemas de reconocimiento de personas, los cuales fueron comparados con el algoritmo HOG de Dalal y Triggs, tomada de [1].

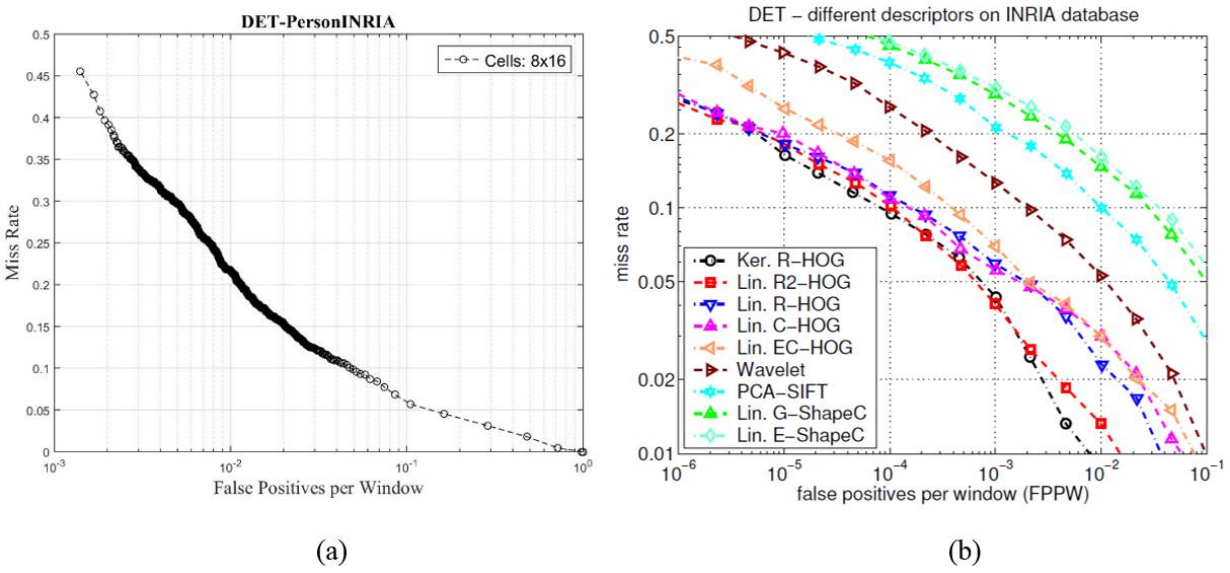

Figura 5.9 Resultados del proceso de reconocimiento de personas. (a) S-ROHM para la base de datos INRIA. (b) Diferentes sistemas comparados con HOG para la base de datos INRIA [1].

Los parámetros de extracción de características de este experimento son los mismos que se utilizaron con el sistema de detección de personas HOG en [1]. Consiste en 8×16 celdas por ventana, 2×2 celdas por bloque, traslape entre bloques de 1×1 celdas, 9 *bins* u orientaciones, los ángulos del gradiente fueron tomados sin signo ($0° - 180°$), en este proceso fue usada la umbralización del gradiente (Sección 4.1.1.1). Por su parte, se decidió experimentar con tan solo 5 neuronas para el clasificador MLP, función sigmoidea como función de transferencia de las neuronas, pesos entre $[-0.25, 0.25]$ y únicamente 20000 iteraciones para la fase de entrenamiento del sistema. Esto representa el uso de pocos recursos, ya que el número de neuronas utilizadas permitió si bien no una clasificación que sobrepase los resultados de los sistemas existentes, si permite una buena clasificación con resultados competitivos.

5.4 Experimento 3. Desempeño del S-ROHM para clasificación multiclase

En este experimento se realizaron pruebas con diferentes clases de objetos de la misma base de datos utilizada en el experimento 1. Como se observó en la Tabla 5.2, se realizaron pruebas con rejillas (número de celdas) adecuadas a las características de la imagen o clase (de 8×16 o 16×8 por ejemplo), pero además se hicieron pruebas con una configuración común a la mayoría de las clases (8×8), esta serie de pruebas se realizó de esta forma con el fin de determinar los cambios que sufría el sistema al establecer una configuración estándar de la ventana de detección, preparando de esta forma el comportamiento que tendría en esta fase de pruebas. La Tabla 5.3 muestra los resultados de un reconocimiento multiclase.

Tabla 5.3 Resultados del experimento 3 (clasificación multiclase).

Grupo	Rendimiento													
	VPR[1]	**ACC[1]**	**VPR[2]**	**ACC[2]**	**VPR[3]**	**ACC[3]**	**VPR[4]**	**ACC[4]**	**VPR[5]**	**ACC[5]**	**VPR[6]**	**ACC[6]**	**FPR**	**# Celdas**
BUTTERFLY[1] CHAIR[2] FACES[3] KETCH[4] LAPTOP[5] SOCCERBALL[6]	0.025	0.976	0.03	0.979	1.000	0.989	0.660	0.983	0.75	0.985	0.103	0.980	0.011	8x8
AIRPLANE[1] CARSIDE[2] ELECTRICGUITAR[3] MOTORBIKES[4] REVOLVER[5] WATCH[6]	0.930	0.988	0.670	0.983	0.100	0.983	0.970	0.991	0.600	0.987	0.630	0.981	0.0074	8x8
AIRPLANE[1] CARSIDE[2] ELECTRICGUITAR[3] MOTORBIKES[4] REVOLVER[5] WATCH[6]	0.960	0.993	0.770	0.989	0.030	0.986	0.995	0.996	0.680	0.992	0.630	0.984	0.0038	12x8
AIRPLANE[1] BUTTERFLY[2] CHAIR[3] FACES[4] LAPTOP[5] MOTORBIKES[6]	0.960	0.993	0.180	0.985	0.130	0.987	0.970	0.99	0.450	0.988	0.975	0.994	0.0042	8x8
AIRPLANE[1] CARSIDE[2] HELICOPTER[3] MOTORBIKES[4]	0.935	0.994	0.710	0.990	0.350	0.990	0.955	0.995	N/A	N/A	N/A	N/A	0.0016	8x8
AIRPLANE[1] CARSIDE[2] HELICOPTER[3] MOTORBIKES[4]	0.945	0.993	0.670	0.988	0.28	0.987	0.975	0.995	N/A	N/A	N/A	N/A	0.0032	12x8
AIRPLANE[1] MOTORBIKES[2]	0.950	0.995	0.980	0.997	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	0.0013	8x8
AIRPLANE[1] MOTORBIKES[2]	0.975	0.997	0.980	0.997	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	0.0016	12x8
FACES[1] REVOLVER[2]	0.930	0.997	0.600	0.994	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	0.0	8x8
AIRPLANE[1] CARSIDE[2] MOTORBIKES[3]	0.920	0.993	0.700	0.989	0.950	0.995	N/A	N/A	N/A	N/A	N/A	N/A	0.0019	8x8
AIRPLANE[1] CARSIDE[2] MOTORBIKES[3]	0.930	0.994	0.690	0.990	0.960	0.996	N/A	N/A	N/A	N/A	N/A	N/A	0.0013	16x8

Para este experimento fueron utilizados los siguientes parámetros de configuración del extractor y del clasificador: 2×2 celdas por bloque, 1×1 celdas de traslape, 9 orientaciones del gradiente, gradiente sin signo, por su parte, 5 neuronas en la capa oculta, función sigmoidea, pesos sinápticos entre $[-0.25, 0.25]$, de la RNA. El parámetro variable en este experimento fue el número de celdas por ventana utilizada (véase Tabla 5.3). Cabe señalar que no fueron las únicas pruebas, se realizaron pruebas con diferentes configuraciones como por ejemplo, aumentando el número de celdas por ventana, modificando la inicialización de los pesos sinápticos, entre otras. Sin embargo en este experimento se logró observar que el aumento del número de neuronas en la capa oculta de la RNA, así como el incremento del número de iteraciones en la fase de entrenamiento de la RNA, aumentan el desempeño del sistema logrando aumentar la razón de verdaderos positivos. Por su parte, realizar una variación en la tasa de aprendizaje de la RNA (α) acelera o desacelera la convergencia del aprendizaje de forma proporcional.

Los resultados presentados en la tabla anterior muestran una diferencia en cuanto a la razón de falsos positivos comparada con los resultados mostrados en el experimento 1, en este experimento se obtuvieron valores mayores, lo que significa que aunque el sistema en general muestra una clasificación exitosa, existe una mayor cantidad de falsos positivos en las imágenes procesadas. En parte, esto se le atribuye a la diferencia de imágenes de entrenamiento usadas por cada clase, ya que, en ciertos casos existe gran diferencia de una clase a otra. Así mismo, se observó que en la fase de entrenamiento de la red neuronal artificial el error mínimo no era tan pequeño (para las 20000 iteraciones usadas) comparado con los resultados del experimento 1, esto significa que el aprendizaje es mucho más lento en la clasificación multiclase. Cambios que pueden mejorar el rendimiento del sistema acerca de este comportamiento consisten en: aumentar el número de iteraciones de la fase de entrenamiento de la RNA, aumentar la cantidad de imágenes negativas de entrenamiento y/o aumentar el número de imágenes positivas de entrenamiento.

Un punto muy importante a resaltar es que, aunque el propósito de este desarrollo de tesis no fue principalmente realizar una detección de múltiples objetos en una sola imagen, las características y naturaleza de una RNA permitieron llevar a cabo este experimento dando resultados realmente sorprendentes, ya que, si bien los resultados no muestran una clasificación perfecta, éstos muestran que es posible realizar detecciones de múltiples objetos con pocos recursos computacionales (haciendo referencia al uso de únicamente 5 neuronas en la capa oculta de la RNA). Este último punto también propicia el desarrollo de una posible línea de investigación en cuanto a la detección de múltiples objetos en imágenes, ya que en estos momentos existe poca bibliografía que aborde este tema, sobre todo con el uso de RNAs.

Capítulo 6

Conclusiones y trabajo futuro

El presente capítulo describe las conclusiones generadas por este trabajo de tesis, a través de los experimentos realizados a las diferentes configuraciones y modos de operación del sistema de reconocimiento de objetos denominado S-ROHM. Finalmente, se incluye el trabajo futuro que dará continuidad a esta investigación.

6.1 Conclusiones

- Se logró implementar en el lenguaje de alto nivel JAVA el algoritmo de extracción de características denominado HOG. Así mismo, se realizó una explicación detallada de este algoritmo junto con ejemplos gráficos que permiten la comprensión del mismo. El desarrollo de este algoritmo en un lenguaje de alto nivel representa una alternativa a las diferentes propuestas existentes, además, su desarrollo fue totalmente propio sin el uso de librerías existentes, por lo cual se adecua a la aplicación y permite posibles optimizaciones.
- Aunado a esto se llevó a cabo el desarrollo, en el mismo lenguaje, de un clasificador basado en redes neuronales artificiales tipo MLP, constituido por: una capa de entrada cuyo número de neuronas equivale al número de características extraídas por el algoritmo HOG, una capa de neuronas oculta con un mínimo de cinco neuronas pero que permite su

ampliación a cualquier valor deseado y finalmente una capa de salida de una neurona para el caso de clasificación binaria, mientras que el número de neuronas es equivalente al número de clases a clasificar para el caso multiclase. Esta unión representa al S-ROHM, producto final de este trabajo de tesis.

- Se generó una interfaz gráfica de usuario que permite interactuar con las diferentes etapas que constituyen a S-ROHM. Esta interacción se logra a dos niveles. Primero, permite al usuario configurar cada una de las etapas de S-ROHM y también visualizar, guardar o recuperar los resultados de cada una de ellas. Segundo, debido a que el diseño de S-ROHM guarda una estructura modular, el usuario puede modificar y/o substituir las diferentes fases que lo integran, lo que permite probar nuevos algoritmos en cada etapa o mejorar los existentes.
- Se comprobó experimentalmente que establecer una configuración estándar del algoritmo HOG, permite una clasificación satisfactoria para las diferentes clases de objetos, sin embargo, queda claro que un análisis profundo permitiría obtener los parámetros de configuración adecuados que permitan así, obtener los resultados óptimos para la clasificación de cada clase.
- A través de los resultados obtenidos se comprobó que es necesario una gran base de datos de imágenes negativas para reducir el índice de falsos positivos.
- En este trabajo no fueron usados filtros ni modificaciones en las imágenes de entrenamiento, pero el uso de éstos, así como el incluir una fase de segmentación previa podría aumentar el rendimiento del S-ROHM.
- Sistemas existentes como el algoritmo HOG muestran que para obtener sus resultados es necesario contar con una gran cantidad de imágenes negativas para el proceso de entrenamiento. En este trabajo de tesis fue comprobado ya que las fases de pruebas iniciales se realizaron con una base de datos con muy pocas imágenes negativas, lo que genero resultados pobres. Por lo que para mejorar el rendimiento fue necesario tomar imágenes negativas tanto de la base de datos INRIA como de la base de datos Caltech.
- La razón por la cual se tomaron algunas de las clases que contaban con una pequeña cantidad de imágenes, es debido a que se esperaba obtener resultados satisfactorios demostrando que si bien los resultados óptimos se obtendrían al contar con una extensa base de datos que permitiera cubrir la variabilidad en apariencia, es posible obtener resultados con pocos datos disponibles.
- Dos casos específicos que llamaron la atención fueron la clase *Butterfly* y la clase C*hair*, ya que ambas contaban con una pequeña base de datos de imágenes positivas, sin embargo, los resultados obtenidos fueron inquietantes debido a que si bien el sistema no fue capaz de reconocer la mayor cantidad de imágenes de prueba, si fue capaz de reconocer que objetos no pertenecían a dichas clases. En otras palabras, el sistema no fue capaz de reconocer mariposas o sillas, pero si fue capaz de reconocer que partes de la

imagen estática no pertenecían a esas clases y por lo tanto, a forma de discriminación permitía reconocer una mariposa o una silla, como se puede observar en la Figura 5.3.

- Por otra parte, queda claramente demostrado que las clases que cuentan con una mayor cantidad de imágenes positivas de entrenamiento, son las clases que permiten que S-ROHM presente un rendimiento mayor, obteniendo la mayor cantidad de verdaderos positivos así como la menor cantidad de falsos positivos.
- Para los experimentos fueron utilizadas imágenes positivas de las diferentes clases que no se encontraban estandarizadas, no contaban con márgenes fijos, inclusive algunas presentaban rotaciones, sin embargo el sistema era capaz de aprender y obtener una clasificación satisfactoria.
- A diferencia del sistema de reconocimiento de personas de Dalal y Triggs el sistema desarrollado en esta tesis es capaz de analizar, procesar y reconocer objetos sin un tamaño estándar. En este sentido, se propuso desde un inicio desarrollar un sistema que fuese capaz de procesar imágenes sin limitarse en cuanto a tamaños, bordes y márgenes. Por lo tanto, la estandarización, el preprocesado, la segmentación y otras herramientas, podrían incrementar el rendimiento del sistema.
- Se propuso una optimización en el algoritmo de cálculo del gradiente que permite una mejor discriminación para el sistema clasificador, la cual consiste en realizar una umbralización, tanto de la magnitud como de los ángulos del gradiente. Esta optimización logra reducir la razón de falsos positivos.
- Se tomaron los parámetros óptimos presentados por Dalal y Triggs como base para la clasificación de las clases presentadas en el presente trabajo, esto debido a que el análisis desarrollado por ellos demostró que el uso de una rejilla densa de celdas permite tener un sistema robusto de descriptores de imagen. Experimentalmente se comprobó que una mayor cantidad de celdas por ventana de detección usada, se traduce en una excesiva carga computacional, pero no en todas las clases significa un aumento sustancial del rendimiento.
- Fue comparado de forma básica el rendimiento del sistema S-ROHM contra el sistema de reconocimiento de personas que utiliza el algoritmo HOG de Dalal y Triggs mostrando resultados competitivos. Un análisis más complejo permitiría una mejor comparación ya que la base de datos INRIA que fue encontrada en el enlace que se muestra en su artículo, presentaba ciertas discrepancias en cuanto a lo expuesto en sus experimentos. Sin embargo, la RNA del sistema fue configurada con una cantidad mínima de neuronas y, también, fue entrenada a una cantidad baja de iteraciones. Se prevé que un análisis profundo permitiría obtener la configuración óptima que permita sobrepasar los resultados propuestos en su sistema en todos los casos.
- Experimentalmente fue comprobada la propiedad de clasificación múltiple de una red neuronal artificial, mostrando que con los parámetros de configuración establecidos para una clasificación binaria, es posible efectuar una clasificación multiclase. Como se puede

observar en la Tabla 5.3, existe una clasificación muy similar al caso de clasificación binaria.

- Con base en la Tabla 5.2 y Tabla 5.3, se puede concluir que es posible una clasificación satisfactoria para casi cualquier tipo de objeto.
- Finalmente, el incluir el enfoque neuronal en la etapa de clasificación del sistema de reconocimiento de objetos presenta resultados muy competitivos y que mejora el rendimiento de estos sistemas en el sentido de que con una base de datos con pocas imágenes positivas, es capaz de aprender y reconocer para diferentes clases de objetos. Así mismo, agrega la capacidad o funcionalidad de detección multiclase, que podría tener mayores aplicaciones y que representa un valor agregado al S-ROHM.

6.2 Trabajo a futuro

Para darle seguimiento a las contribuciones hechas por el trabajo presentado en esta tesis, se proponen los siguientes temas de investigación.

- En la propuesta de mejora del gradiente de la ventana de detección se comprobó que su uso fue de utilidad al disminuir la razón de falsos positivos FPR, por lo tanto, se podría realizar un análisis detallado de la optimización propuesta con el fin de adecuar el umbral propuesto, así como también proponer alguna función que elimine de igual forma el ruido de la imagen.
- En este trabajo no se realizó ningún análisis de tiempos de ejecución, sin embargo, en tareas de análisis de imágenes es muy conocido que los tiempos de procesamiento son elevados y dependen de las características del hardware utilizado, es por esto y, debido a las nuevas tecnologías en los sistemas de cómputo, los cuales están equipados actualmente con más de un procesador lógico que, se propone el uso de hilos (*threads*) en el desarrollo del sistema para la ejecución en paralelo de, por ejemplo, múltiples ventanas de detección o el trabajo en paralelo de las neuronas de la RNA.
- En este trabajo se propuso el uso de una sola capa oculta de neuronas para la RNA tipo MLP, los resultados obtenidos al experimentar con la clasificación multiclase muestran que, al aumentar el número de neuronas de esta capa solo se disminuye la cantidad de iteraciones necesarias para alcanzar el error mínimo. Se propone el uso de dos o más capas ocultas de neuronas para la RNA, lo cual podría aumentar el desempeño del sistema. Las ecuaciones 3.14 y 3.15 proporcionan la información necesaria para adecuar la implementación de una nueva capa.

$$a_i^c = f\left(\sum_{j=1}^{n_{c-1}} w_{j\,i}^{c-1} a_j^{c-1} + u_i^c\right) \; para\; i = 1,2,\dots,n_c \; y \; c = 2,3,\dots,C-1$$

$$y_i = a_i^C = f\left(\sum_{j=1}^{n_{C-1}} w_{j\,i}^{C-1} a_j^{C-1} + u_i^C\right) \; para \; i = 1,2,\ldots,n_C$$

- Finalmente se propone la implementación del sistema en dispositivos lógicos programables como lo puede ser un FPGA, el paralelismo que proporciona esta herramienta se adecua a las funcionalidades del S-ROHM, el bajo uso de recursos en la RNA puede ser de gran utilidad en alguna aplicación con sistemas autónomos que requiera un sistema de reconocimiento de objetos.

Referencias

[1] Dalal N. and Triggs B. "Histograms of Oriented Gradients for Human Detection", *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol. 1, No 1, pp. 886-893, 2005.

[2] Ballard, D.A. and Brown, C.M. *Computer vision*. Englewood Cliffs, NJ, USA: Prentice-Hall. 1982.

[3] Sobrado, E.A., Sistema de visión artificial para el reconocimiento y manipulación de objetos utilizando un brazo robot. Tesis de Maestría, Pontificia Univ. Católica del Perú, Lima, Perú. 2003.

[4] Mak, K.L., Peng, P., and Yiu, K.F.C. "Fabric defect detection using morphological filters", *Image and Vision Computing*, Vol. 27(10), pp. 1585–1592. 2009.

[5] Abouelela, A., Abbas, H.M., Eldeeb, H., Wahdan, A.A., and Nassar, S.M. "Automated vision system for localizing structural defects in textile fabrics", *Pattern Recognition Letters*, Vol. 26(10), pp. 1435–1443. 2005.

[6] Baluja, S. and Pomerleau, D. "Dynamic relevance: vision-based focus of attention using artificial neural networks". *Artificial Intelligence*, Vol. 97(1-2), pp. 381-395. 1997.

[7]Wang, J. and Asundi, A.K. "A computer vision system for wineglass defect inspection via Gabor-filter-based texture features", *Information Sciences*, Vol. 127(3–4), pp. 157–171. 2000.

[8] Li, O., Wang, M., and Gu, W. "Computer vision based system for apple surface defect detection", *Computers and Electronics in Agriculture*, Vol. 36(2–3), pp. 215–223. 2002.

[9] Saeidi, R.G., Latifi, M., Najar, S.S., and Saeidi, A.G. "Computer vision-aided fabric inspection system for on-circular knitting machine", *Textile Research Journal*, Vol. 75(6), pp. 492–497. 2005.

[10] Lowe D.G. "Distinctive image features from scale-invariant keypoints", *IJCV*, Vol. 60, No 2, pp. 91-110, 2004.

[11] Zhu Q., Avidan S., Yeh M. and Cheng K. "Fast Human Detection Using a Cascade of Histogram Oriented Gradients", *IEEE Computer Society Conference of Computer Vision and Pattern Recognition*, TR2006-068, 2006.

[12] Wang X., Han T.X. and Yan S. "An HOG-LBP Human Detector with Partial Occlusion Handling", *In ICCV*, pp. 32-39, 2009.

[13] Yang S., Liao X. and Borasy U. "A Pedestrian Detection Method Based on the HOG-LBP Feature and Gentle AdaBoost", *Int. Joutnal of Advancements in Computing Technology*, Vol. 4, No 19, 2012.

[14] Ozbay, S. and Ercelebi E. "Automatic Vehicle Identification by Plate Recognition". *Proceedings of World Academy of Science, Engineering and Technology*. 2005.

[15] Kocer, H.E. and Cevik, K.K. "Artificial neural networks based vehicle license plate recognition", *Procedia Computer Science*, Vol. 3, pp. 1033–1037, 2011.

[16] Comaniciu, D., Ramesh, V., and Meer, P. "Real-time tracking of non-rigid objects using mean shift", *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Vol. 2, pp. 142-149. 2000.

[17] Sivic, J., Russell, B.C., Efros, A.A., Zisserman, A. and Freeman, W.T. "Discovering objects and their location in images", *Proceedings of Tenth IEEE International Conference on Computer Vision, 2005*, Vol. 1, pp. 370-377. 2005.

[18] McKenna, S.J., Jabri, S., Duric, Z., Rosenfeld, A. and Wechsler, H. "Tracking Groups of People", *Computer Vision and Image Understanding*, Vol. 80(1), pp. 42–56. 2000.

[19] Comaniciu, D., Ramesh, V. and Meer, P. "Kernel-based object tracking*", IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 25(5), pp. 564-577. 2003.

[20] Baluja, S. and Pomerleau, D. "Expectation-based selective attention for visual monitoring and control of a robot vehicle". Robotics and Autonomous Systems, Vol. 22(3–4), pp 329–344. 1997.

[21] Murray, D. and Little, J.J. "Using Real-Time Stereo Vision for Mobile Robot Navigation", *Autonomous Robots*, Vol 8(2), pp. 161-171. 2000.

[22] Royer, E., Lhuillier, M., Dhome, M., and Lavest, M-C. "Monocular Vision for Mobile Robot Localization and Autonomous Navigation", *International Journal of Computer Vision*, Vol. 74(3), pp. 237-260. 2007.

[23] Thrun, S., Fox, D., Burgard, W., and Dellaert, F. "Robust Monte Carlo localization for mobile robots", *Artificial Intelligence*, Vol. 128(1–2), pp. 99–141. 2001.

[24] Chaumette, F., and Hutchinson, S. "Visual servo control. I. Basic approaches", *IEEE Robotics & Automation Magazine*, Vol. 13(4), pp. 82-90. 2006.

[25] Pietikäinen, M., Hadid, A., Zhao, G., and Ahonen T. *Computer Vision Using Local Binary Patterns*. Springer London Dordrecht Heidelberg New York. 2011.

[26] Jiménez, A., Sistema de reconocimiento y localización de objetos cuasi-esféricos por telemetría láser. Aplicación a la detección automática de frutos para el robot Agribot", Tesis de doctoral, Universidad Complutense de Madrid. 2000.

[27] Viola, P., and Jones, M.J. "Robust Real-Time Face Detection", *International Journal of Computer Vision*, Vol. 57(2), pp. 137-154. 2004.

[28] Muñoz-Salinas, R., Aguirre, E., and García-Silvente M. "People detection and tracking using stereo vision and color", *Image and Vision Computing*, Vol. 25(6), pp. 995–1007. 2007.

[29] Haihong Zhang, Bailing Zhang, Weimin Huang, and Qi Tian. "Gabor Wavelet Associative Memory for Face Recognition", IEEE Transactions on Neural networks, Vol. 16(1), pp. 275-278. 2005.

[30] Yilmaz, A., Javed, O., and Shah, M. "Object tracking: A survey", *Journal ACM Computing Surveys*, Vol. 38(4), 2006.

[31] Miura, J., Kanda, T., and Shirai, Y. "An active vision system for real-time traffic sign recognition". *Proceedings of Intelligent Transportation Systems*, pp. 52-57. 2000.

[32] Forsyth D. A. and Ponce Jean. *Computer Vision: A Modern Approach*. Prentice Hall. USA, ISBN 978-0136085928. 2011.

[33] Uttal W. R. and Hillsdale N. J. *On Seeing Forms.* Lawrence Erlbaum Associates, Inc. England, ISBN 978-1-84872-435-8. 1988.

[34] Tou J. T. and Gonzalez R. C. *Pattern Recognition Principles*. Addison-Wesley Publishing Co, Inc. USA, ID 19750040001. 1974.

[35] Papageorgiou C. P., Oren M. and Poggio T. "A general framework for object detection", *IEEE International Conference on Computer Vision*, pp. 555, 1998.

[36] Michael A. y Andrade-Cetto J. "Cómputo de Características Invariantes a la Rotación para el Reconocimiento de Distintas Clases de Objetos", *Universitat Autónoma de Barcelona*, España, 2006.

[37] Luna M. D. and Tudela P. *Percepción Visual*. TROTTA. Spain, ISBN 978-848-16487-20. 2007.

[38] Nevatia R. and Binford T. O. "Description and Recognition of Curved Objects", *Artificial Intelligence*, Vol. 8, No 1, pp. 77-98, 1977.

[39] Maravall D. *Reconocimiento de formas y visión artificial*, Addison Wesley Iberoamericana, 1994.

[40] González R. and Woods R. *Digital Image Processing*. Prentice Hall. New Jersey, ISBN 0-201-18075-8. 2002.

[41] Jiménez F. M. "Reconocimiento de objetos con realidad aumentada. App iWhatslt para la atención de la diversidad funcional en visión". Tesis de licenciatura. Departamento de Lenguajes y Sistemas Informáticos, Universidad de Granada. 2013.

[42] Sánchez Raul. "Diseño e implementación de un sistema detector de objetos para el robot Asibot". Tesis. Departamento de Ingeniería de Sistemas y Automática, Universidad Carlos III de Madrid. 2011.

[43] Sossa Azuela J. H. *Rasgos descriptores para el reconocimiento de objetos*. Instituto Politécnico Nacional, México, ISBN 970-36-0318-1. 2006.

[44] Pratt W. K. *Digital Image Processing*. John Wiley and Sons, Inc, Third edition. USA, ISBN 0-471-37407-5. 2001.

[45] Guzmán E. "Compresión de imágenes mediante memorias asociativas". Tesis doctoral. Centro de Investigación en Computación, IPN. 2008.

[46] Petrou, M., P. Bosdogianni. *Image Processing: the fundamentals*. ISBN 0-471-99883-4, USA: John Wiley and Sons, 1999.

[47] Acharya, T., A. K. Ray. *Image Processing. Principles and Applications*. ISBN-13 978-0-471-71998-4, USA: John Wiley & Sons, 2005.

[48] Acharya T. and Tsai P-S. JPEG 2000 Standard for Image Compression. Concepts, Algorithms and VLSI Architectures. John Wiley & Sons. USA, ISBN 0-471-48422-9. 2005.

[49] Bövik, A. Handbook of Image and Video Processing. Principles and Applications. ISBN 0-12-119790-5, USA: Academic Press, 2000.

[50] Zanuy, M. F. *Tratamiento Digital de Voz e Imagen*. ISBN 970-15-0651-0, USA: Alfaomega grupo editor, 2001.

[51] Jahne, B. *Digital Image Processing*. 5th and extended edition. ISBN 3-540-67754-2, Germany: Springer, 2002.

[52] Mendoza M. A. "Procesamiento y análisis digital de imágenes mediante dispositivos lógicos programables". Tesis de licenciatura. Universidad Tecnológica de la Mixteca. 2009.

[53] Fourier J. B. J. *Théorie Analytique de la Chaleur*, 1822.

[CAS02] Castleman K. *Digital Image Processing*. Prentice Hall. New Jersey. 1996.

[54] Gonzalez, R. C., R. E. Woods. *Digital Image Processing Using MATLAB*. ISBN 81-7758-898-2, USA: Pearson Educational, 2006.

[55] Sucar E. and Gómez G. *Visión Computacional [en línea]*. Instituto Nacional de Astrofísica, Óptica y Electrónica. Puebla, México. 2005 [fecha de consulta: 10 Agosto 2014]. Disponible en: <http://ccc.inaoep.mx/~esucar/>.

[56] Roberts L. "Machine Perception of 3-D Solids", *Optical and Electro-optical Information Processing*, MIT Press, 1974.

[57] Boyle R. and Thomas R. "Computer Vision: A First Course", *Blackwell Scientific Publications*, pp. 48-50, 1988.

[58] Prewitt J. M. S. "Object enhancement and extraction", *Picture processing and Psychopictorics*, Academic Press, pp. 75-149, 1970.

[59] Ojala, T., Pietikäinen, M., Harwood, D. "A comparative study of texture measures with classification based on feature distributions". *Pattern Recognition*, Vol. 29, No. 1, pp. 51-59. 1996.

[60] Bay H., Tuytelaars T., and Van Gool L., "SURF: Speeded up robust features," *in Proceedings of the European Conference on Computer Vision*, pp. 404–417, 2006.

[61] Hough P. V. C. *Method and means for recognizing complex patterns*. USA: Pantent 3, 069 654, 1962.

[62] Hu M-K. Visual pattern recognition by moment invariants. *IRE Transactions on Information Theory*, 8(2), pp. 179-187, 1962.

[63] Reed Teague M. Image analysis via the general theory of moments. *Journal of the optical society of America*, Vol. 70(8), pp.920-930, 1980.

[64] Aboufadel E. A wavelets approach to voice recognition. *Grand Valley State University*, 2001

[65] Bustos O, Mallea A. y Herrera M. *Introducción al procesamiento de imágenes digitales*. Universidad Nacional de Córdoba, Argentina, 2006.

[66] Rutovitz D. "Pattern Recognition", *Journal of the Royal Statistical Society. Series A (General)*, No 129, pp. 504-530, 1966.

[67] Carrasco J. "Enfoque Lógico Combinatorio al Reconocimiento de Patrones, I Selección de Variables y Clasificación Supervisada", *Computación y Sistemas*, Vol. 4, No 1, pp. 68-69, 2000.

[68] Sánchez K. V. "Descriptores de imágenes digitales con momentos de Zernike". Tesis de maestría. Facultad de Ingeniería Eléctrica, UNAM. 2011.

[69] Van de Geer J. P. *Some Aspects of Minkowski Distances*. Department of Data Theory, University of Leiden, Research Report RR-95.03. 1995.

[70] Hartigan J. A. *Clustering algorithms*. John Wiley & Sons, Inc. USA. 1975.

[71] Hartigan J. A. "Algorithm AS 136: A K-Means Clustering Algorithm", *Journal of the Royal Statistical Society*, Vol. 28, No 1, pp. 100-108, 1979.

[72] Kaufman L. and Rousseeuw P. J. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley & Sons, Inc. USA, ISBN 978-0471878766. 2005.

[73] Jain A. K. and Dubes R. C. *Algorithms Clustering Data*. Prentice Hall. USA, ISBN 978-0130222787. 1988.

[74] Selim S. Z. and Alsultan K. "A simulated annealing algorithm for the clustering problem", *Journal Pattern Recognition*, Vol. 24, No 10, pp. 1003-1008, 1991.

[75] McCulloch W. S. and Pitss W. "A logical calculus of the ideas", *Bulletin of Mathematical Biophysics*, Vol. 5, 1943.

[76] Hebb D. O. The Organization of Behavior: A Neuropsychological Theory. John Wiley & Sons. USA, ISBN 1-4106-1240-6. 1949.

[77] Gabor D. "Theory of Communication", *Journal of Institute for Electrical Engineering*, Vol. 93, No 26, pp. 429-457, 1954.

[78] Rosenblatt F. "The Perceptron: A probabilistic model for information storage and organization in the brain", *Psychological Review*, Vol. 65, No. 3, pp. 386-408, 1958.

[79] Widrow B. and Hoff M. "Adaptive Switching Circuits", *IRE WESCON Convention Record*, Vol. 4, pp. 96-104, 1960.

[80] Novikoff A. B., "On Convergence Proofs for Perceptron", *Proceedings of the Symposium on the Mathematical Theory of Automata*, Vol. 12, pp. 615-622, 1963.

[81] Minsky M. and Papert S. Perceptrons. An Introduction to Computational Geometry. M.I.T. Press, Cambridge. USA, ISBN 0036-8075. 1969.

[82] Hecht-Nielsen R. "Theory of the Back Propogation Neural Network", *Proceedings of the International Joint Conference on Neural Networks*, pp. 593-608, 1989.

[83] Gori M. and Tesi A. "On the Problem of Local Minima in Backpropagation", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 14, No 1, 1992.

[84] Bryson A. and Ho Yu-Chi. "Applied Optimal Control: Optimization, Estimation and Control", *Blaisdell Publishing Company or Xerox College Publishing*, pp. 481, 1969.

[85] Werbos P. "Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences". Tesis doctoral. Universidad de Harvard. 1974.

[86] Parker D. *Learning Logic*. USA: Stanford University, Invention Report S81-64, File 1, Office of Technology Licensing. 1982.

[87] Parker D. *Learning Logic*. Technical Report TR-47. Center for Computational Research in Economics and Management Science, MIT. 1985.

[88] Rumelhart D. E., Hinton G. E. and Williams R. J. "Learning Internal Representations by Error Propagation", *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Vol. 1, pp. 318-362, 1986.

[89] Le Cun Y. and Becker S. "Improving the convergence of back-propagation learning with second order methods", *Proceedings of the 1988 connectionist models summer school*, pp. 29-37, 1988.

[90] Hilera J. R. y Martínez V. J. *Redes Neuronales Artificiales*. Alfaomega-Rama. España, 2000.

[91] Kung S. Y. *Digital Neural Networks*. Prentice Hall. USA, ISBN 978-0136123262. 1993.

[92] Basogain X. *Redes Neuronales Artificiales y sus aplicaciones*. Escuela Superior de Ingeniería de Bilbao, EHU, España, 2006.

[93] Valencia M. Yáñes C. y Sánchez L. *Algoritmo Backpropagation para Redes Neuronales: conceptos y aplicaciones*. México: Centro de Investigación en Computación, IPN, Serie Verde, No 125. 2006.

[94] Martín, B.B. y Sanz, M.A. *Redes Neuronales y Sistemas Borrosos*. Alfaomega. pp 20-40. 2007.

[95] Caudill M. and Butler C. "Understanding Neural Networks", *Computer Explorations*, Vol. 1, pp. 155-218, 1993.

[96] Feldman J. and Ballard D. "Connectionist models and their properties", *Cognitive Science*, Vol. 6, No 3, pp. 205-254, 1982.

[97] Kohonen T. "An introduction to neural computing", *Neural Networks*, Vol. 1, No 1, pp. 3-16, 1988.

[98] Roberts L. G. "Machine Perception of Three-Dimensional Solids". PhD Thesis. Massachusetts Institute of Tecnology. 1963.

[99] Binford T. "Visual perception by computer", *IEEE Conf. On Systems and Controls*, Miami, 1971.

[100] Lowe D. G. "Visual recognition from spatial correspondence and perceptual organization", Courant Institute of Mathematical Sciences, New York University. 1985.

[101] Huttenlocher D. and Ullman S. "Object Recognition Using Alligment", Artificial Intelligence Laboratory, Massachusetts Institute of Tecnology. 1987.

[102] Freeman W. T. and Roth M. "Orientation Histograms for Hand Gesture Recognition", *IEEE Intl. Wkshp. On Automatic Face and Gesture Recognition*, 1995.

[103] McConnell R. K. *Method of and apparatus for pattern recognition*. U. S. Patent No. 4567610A. 1986.

[104] Belongie S., Malik J. and Puzicha J. "Matching shapes", *The 8th ICCV, Vancouver, Canada*, pp. 454-461, 2001.

[105] Murase H. and Nayar S. "Visual learning and recognition of 3-D objects from appearance", *Int. Journal of Computer Vision*, Vol. 14, No 1, pp. 5-24, 1995.

[106] Joachims T. "Making large-scale SVM learning practical". In B.Schlkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods*. MIT Press, pp. 42-56, 1999

[107] Viola P. and Jones M. "Rapid object detection using a boosted cascade of simple features", *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2001.

[108] Zhang Li., Wu B. and Nevatia R. "Pedestrian Detection in Infrared Images based on Local Shape Features", *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[109] Wu B. and Nevatia R. "Detection of Multiple, Partially Occluded Humans in a single Image by Bayesian Combination of Edgelet Part Detectors", *International Conference on Computer Vision*, Vol. 1, pp. 90-97, 2005.

[110] Kobayashi T., Hidaka A. and Kurita T. "Selection of Histograms of Oriented Gradients Features for Pedestrian Detection", *Published in: Neural Information Processing*, ISBN 978-3-540-69159-4, pp. 598-607, 2008.

[111] Ke Y. and Sukthankar R. "PCA-SIFT: A more distinctive representation for local image descriptors", *Proc. Of Computer Vision and Pattern Recognition*, pp. 66-75, 2004.

[112] Chandrasekhar V., Takacs G., Chen D., Tsai S., Grzeszczuk R, and Girod B. "CHoG: Compressed Histogram of Gradients A low Bit-Rate Feature Descriptor", *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2504-2511, 2009.

[113] Socarras Y., Vázquez D., López A. M., Gerónimo D. and Gevers T. "Improving HOG with Image Segmentation: Application to Human Detection", *Int. Conf. on Advanced Concepts for Intelligent Vision Systems*, pp. 178-189, 2012.

[114] Cheng Y. "Mean Shift, Mode Seeking, and Clustering", *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 17, No 8, 1995.

[115] Freeman W. T., Tanaka K., Ohta J. and Kyuma K. "Computer Vision for Computer Games". *2nd International Conference on Automatic Face and Gesture Recognition, Killington*. VT, USA, pp. 100-105, 1996

[116] Mikolajczyk K. and Schmid C. "Scale and affine invariant interest point detectors", *International Journal of Computer Vision*, Vol. 60, No 1, pp. 63-86, 2004.

[117] Pérez-Aguila R. *Procesamiento de Imágenes: Apuntes*. Universidad Tecnológica de la Mixteca, México, 2011.

[118] Rosenblatt F. "Perceptron simulation experiments", *Proceedings of the IRE*, Vol. 18, No. 3, pp. 301-309, 1960.

[119] Isasi P. y Galván I. M. *Redes de Neuronas Artificiales: Un enfoque práctico*. Pearson Educación, S.A., España, ISBN 84-205-4025-0. 2004.

[120] Flórez R. y Fernández J. M. Las Redes Neuronales Artificiales: Fundamentos teóricos y aplicaciones prácticas. Netbiblo. 2008.

[121] Rojas R. *Neural Networks: A Systematic Introduction*. Springer. Berlin. 1996.

[122] Zagaceta M. T. "Filtrado Digital Adaptivo Integrado". Tesis doctoral. Centro de Investigación en Ciencia Aplicada y Tecnología Avanzada, IPN. 2009.

[123] Bottou Léon. "Stochastic Gradient Descent Tricks", *Microsoft Research*, Redmont, WA, 2012.

[124] Fei-Fei R., Fergus R. and Perona P. "Learning generative visual models from few training examples: an incremental Bayesian approach tested on 101 object categories", *IEEE CVPR 2004, Workshop on Generative-Model Based Vision*, 2004.

[125] Anwer R. M., Vázquez D. and López A. M. "Opponent Colors for Human Detection", *Pattern Recognition and Image Analysis Lecture Notes in Computer Science*, Vol. 6669, pp. 363-370, 2011.