TRABAJO ESPECIAL DE GRADO

DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE RECONOCIMIENTO DE MATRICULAS VEHICULARES

Presentado ante la ilustre
Universidad Central de Venezuela
por el Br. José Alejandro Bracho Gorrín
para optar al título
de Ingeniero Electricista.

Caracas, octubre 2016

TRABAJO ESPECIAL DE GRADO

DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE RECONOCIMIENTO DE MATRÍCULAS VEHICULARES

TUTOR ACADÉMICO: Ing. Pedro Pinto

Presentado ante la ilustre
Universidad Central de Venezuela
por el Br. José Alejandro Bracho Gorrín
para optar al título
de Ingeniero Electricista.

Caracas, octubre 2016

CONSTANCIA DE APROBACIÓN

Caracas, 01 de NOVIEMBRE de 2016

Los abajo firmantes, miembros del Jurado designado por el Consejo de la Escuela de Ingeniería Eléctrica, para evaluar el Trabajo Especial de Grado presentado por el Bachiller José Alejandro Bracho Gorrín, titulado:

"DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE RECONOCIMIENTO DE MATRÍCULAS VEHICULARES"

Consideran que el mismo cumple con los requisitos exigidos por el plan de estudios conducente al Título de Ingeniero Electricista en la mención de Electrónica, Computación y Control, y sin que ello signifique que se hacen solidarios con las ideas expuestas por el autor, lo declaran **APROBADO**.

Jurado
PANAYOTIS TREMANTE

Jurado ALEJANDRO GONZALEZ

Pedro Pinto
Profesor guía

A Dios y a mi familia.

RECONOCIMIENTOS Y AGRADECIMIENTOS

A Dios por darme la perseverancia que necesitaba para culminar mis estudios.

A mi madre, padre y hermano por su amor y apoyo incondicional a lo largo de mi vida y mi carrera.

A todos los excelentes profesores que tuve la oportunidad de conocer en estos años de mi vida universitaria.

Agradezco a mi tutor y profesor guía Pedro Pinto, por todos los consejos y el apoyo que me brindó durante el desarrollo de este proyecto.

A todos las brillantes personas que conocí en los últimos años, que pasaron a formar parte de mi vida. Muchas gracias a Güette, Yvelc, Paglia, Marta, Lisgrett, José Manuel, Ortaz, Cen, Fabiola, Carlos G., Arroyo, Maitán, Jhonny, Joseph y Barillas por todo su apoyo.

A Rubén Mijares por todo el apoyo que me brindó en estos últimos años, y por facilitar su Raspberry Pi, lo cual permitió el desarrollo del prototipo para este proyecto.

Agradezco a todas y cada una de las personas que conocí en estos años de vida universitaria. Por todos aquellos momentos que pasamos juntos, los cuales atesoraré por el resto de mi vida. Gracias.

Bracho G., José A.

DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE RECONOCIMIENTO DE MATRÍCULAS VEHICULARES

Tutor Académico: Prof. Pedro Pinto. Tesis. Caracas. U.C.V. Facultad de Ingeniería. Escuela de Ingeniería Eléctrica. Ingeniero Electricista. Opción: Electrónica. Año 2016.

Palabras Clave: Visión Computarizada; OpenCV; Python; Tesseract-OCR; Raspberry Pi.

Resumen.- En el presente Trabajo Especial de Grado, se diseña un sistema de reconocimiento de matrículas vehiculares. El sistema en cuestión recibe una imagen digital de un vehículo, la cual será analizada utilizando diferentes algoritmos para localizar, filtrar, manipular y, finalmente, obtener el número de la matrícula. Para lograr este fin, se utilizaron diferentes funciones de la librería de visión computarizada OpenCV para analizar la imagen, y el software de reconocimiento Tesseract-OCR para obtener el número de la matrícula. Cabe destacar que se realizó la programación en Python, ya que ofrece numerosas facilidades de uso. Se realizaron diferentes pruebas para definir las condiciones de uso del programa. En conjunto con el desarrollo de la rutina, se logró la implementación de un módulo de adquisición de datos, el cual estaba conformado por una cámara USB, una tarjeta de desarrollo Raspberry Pi y un adaptador de red para enviar la imagen capturada al computador principal, el cual tiene una mayor capacidad de procesamiento que dicho módulo.

ÍNDICE GENERAL

CONSTANCIA DE APROBACIÓN....III
DEDICATORIA....IV
RECONOCIMIENTOS Y AGRADECIMIENTOS....V
RESUMEN....VI
ÍNDICE GENERAL....VII
LISTA DE FIGURAS....IX
LISTA DE TABLAS....XII
LISTA DE ACRÓNIMOS....XIII
INTRODUCCIÓN....1
CAPITULO I....3
1.1. PLANTEAMIENTO DEL PROBLEMA....3
1.2. OBJETIVOS....4
1.2.1. Objetivo General....4
1.2.2. Objetivos Específicos....4
CAPÍTULO II....5
2.1. Imagen digital....5
2.2.Procesamiento digital de imágenes....6
2.2.2. Mejoramiento de la imagen. Distorsión....11
2.2.3. Compresión de la imagen. Binarización y Erosión....15
2.2.4. Extracción de características. Patrones Binarios Locales....17
2.3. Visión Computarizada....19
2.3.1. Técnicas de localización de objetos en la imagen....20
2.3.2. Reconocimiento de objetos. Clasificadoras....24
2.4. Objeto a detectar: Matrícula Vehicular....27
2.5. OpenCV....29
2.6. Python....29
2.7. Antecedentes....30

CAPÍTULO III.....32
3.1. Descripción de hardware.....32
3.2. Descripción de software.....36
3.3. Descripción de la rutina de reconocimiento.....37
3.3.1. Localización de posibles áreas de la matrícula.....38
3.3.2. Reajuste de tamaño y filtrado de las posibles matrículas.....47
3.3.3. Binarización y erosión de la imagen.....47
3.3.4. Localización de área del número de la matrícula.....48
3.3.5. Reconocimiento de caracteres.....51
CAPÍTULO IV.....53
4.1. Rutina de localización.....54
4.1.1. Desempeño.....54
4.1.2. Vehículos en movimiento.....57
4.1.3. Distancia máxima de reconocimiento.....60
4.2. Desempeño OCR.....61
CONCLUSIONES Y RECOMENDACIONES.....66
REFERENCIAS BIBLIOGRÁFICAS.....70
ANEXO I.....73
MANUAL DE USUARIO.....73

LISTA DE FIGURAS

1. Representación matricial de una imagen.....6
2. Distribución Gaussiana del valor del pixel.....7
3. Ruido Gaussiano.....8
4. Ruido Impulsional.....8
5. Convolución de matrices.....9
6. Ejemplo de un kernel Gaussiano de 3x3.....10
7. Efecto de filtro Gaussiano.....10
8. Ejemplo de un kernel de filtro de media de 3x3.....11
9. Efecto de filtro de Media.....11
10. Distorsión Radial.....12
11. Distorsión de perspectiva.....12
12. Diferentes procesos de binarización.....16
13. Erosión de la imagen.....16
14. Matríz ejemplo.....17
15. Matríz de umbral.....18
16. Matríz de pesos.....18
17. Matríz LBP.....19
18. Aplicación de LBP para describir textura.....19
19. Diagrama de flujo ventana de búsqueda.....22
20. Ejemplo de una pirámide de imágenes.....23
21. Diagrama de flujo de la pirámide de imágenes.....24
22. Diagrama de la clasificadora en cascada.....25
23. Detección de vehículos con la clasificadora en cascada.....26
24. Ejemplo de una matrícula vehicular venezolana.....29
25. Esquema del equipo.....33
26. Cámara digital.....34

27. Raspberry Pi 2 modelo B.....35
28. Aplicación de LBP sobre la matrícula.....40
29. Ejemplos de muestras negativas.....42
30. Diagrama de flujo de la rutina de localización.....46
31. Binarización y Erosión.....48
32. Contornos.....49
33. Contornos de interés.....49
34. Algoritmos de localización del número de matrícula.....50
35. Área del número de la matrícula.....51
36. Utilización del Tesseract-OCR.....52
37. Secciones de matrícula.....55
38. Áreas de matrícula.....55
39. Matrícula con vegetación.....56
40. Matrícula dañada.....56
41. Matrícula con ángulo de visualización 1.....56
42. Matrícula con ángulo de visualización 2.....56
43. Dos vehículos en una imagen.....57
44. Matrículas reconocidas.....57
45. Vehículo en movimiento.....58
46. Imagen analizada.....58
47. Secciones obtenidas.....59
48. Efectos negativos en la imagen.....59
49. Matrícula detectada.....60
50. Matrícula detectada a 7 metros.....61
51. Detección errónea: Letra/Letra.....62
52. Detección errónea: Número/Letra.....62
53. Ejemplo matrícula inclinada 1.....64
54. Ejemplo de matrícula inclinada 2.....64

55. Matrícula desgastada.....64
56. Matrícula con vegetación.....65

LISTA DE TABLAS

Tabla 1. Especificaciones matrícula vehicular.....28
Tabla 2. Especificaciones de cámara USB.....33
Tabla 3. Especificaciones Cámara CyberShot.....34
Tabla 4. Especificaciones Raspberry Pi 2 modelo B.....35
Tabla 5. Especificaciones adaptador Wi-Fi.....36
Tabla 6. Especificaciones Computador.....36
Tabla 7. Resolución de cada nivel.....38
Tabla 8. Definición de la clase.....49
Tabla 9. Valores del parámetro psm.....51
Tabla 10. Resultados rutina de localización.....54
Tabla 11. Resultados OCR.....61
Tabla 12. Resultados OCR con corrección.....63

LISTA DE ACRÓNIMOS

ALPR: Reconocimiento automático de matrícula (*Automatic License Plate Recognition*).

BMP: Mapa de bits (*Bitmap*).

IDE: Ambiente de desarrollo integrado (*Integrated Development Environment*).

IDLE: Ambiente de desarrollo y aprendizaje integrado (*Integrated Development and Learning Environment*).

GIF: Formato de intercambio gráfico (*Graphics Interchange Format*).

GPL: Licencia pública general de GNU (*General Public License*).

HOG: Histograma de gradientes orientados (*Histogram of Oriented Gradients*).

JPEG: Grupo de expertos fotográficos unidos (*Joint Photographic Experts Group*).

LBP: Patrones binarios locales (*Local Binary Patterns*).

LPR: Reconocimiento de matrícula (*License Plate Recognition*).

MP: Mega píxel.

OCR: Reconocimiento óptico de caracteres (*Optical Character Recognition*).

OSD: Detección de orientación y escritura o letra (*Orientation and Script Detection*).

PNG: Gráfico portable para la red (*Portable Network Graphic*).

PSF: Fundación de software Python (*Python Software Foundation*).

PSM: Método de segmentación de página (*Page Segmentation Mode*).

ROI: Región de interés (*Region of Interest*).

SVM: Máquina de soporte vectorial (*Support Vector Machine*).

TIFF: Formato de archivo de imagen etiquetada (*Tagged Image File Format*).

USB: Bus serial universal (*Universal Serial Bus*).

INTRODUCCIÓN

Alrededor del mundo, los sistemas de reconocimiento visual se han utilizado para diferentes aplicaciones, desde controles de acceso por reconocimiento facial hasta sistemas de vigilancia de circuito cerrado. No es de sorprenderse que este tipo de sistemas esté ganando mucha popularidad, ya que, en términos de hardware, resulta muy sencillo de implementar, sólo se requiere de una cámara y un computador con ciertos requerimientos mínimos para procesar la información.

En los últimos años, con el mejoramiento de la capacidad de procesamiento de los computadores, se han realizado investigaciones orientadas al análisis de imágenes, para detectar ciertos elementos en las mismas. En estos últimos 15 años, se han diseñado rutinas para detección de objetos por forma, color, superficie, entre otras características. Gracias a los resultados de estas investigaciones, es posible diseñar e implementar un sistema que permita detectar cualquier objeto deseado.

Con esto en mente, se plantea el diseño de un sistema de reconocimiento de matrículas vehiculares. Este sistema de reconocimiento tiene múltiples aplicaciones, sin embargo, en este trabajo, se busca dar solución a una situación problemática, esa es, el robo de vehículos. En Venezuela, la verificación de datos del vehículo no está automatizada, y resulta complicado conocer los datos del vehículo, tales como propietario, infracciones e incluso, si está reportado como robado.

Como se dijo con anterioridad, el sistema estará conformado por una cámara, con la cual se realizará la captura de la imagen del vehículo y se enviará la información a una computadora para analizar la imagen, con el fin de localizar la matrícula y obtener el número de la misma.

En el **Capítulo 1** se expone el planteamiento del problema, junto con los objetivos general y específicos del presente trabajo de grado.

En el **Capítulo 2** se muestra el marco teórico, en donde se explican los conceptos, herramientas y métodos utilizados en el presente proyecto, junto con el respectivo marco legal o normativa que estipula las características principales del objeto a detectar.

En el **Capítulo 3** se ofrece una descripción general del proyecto, donde se explica el funcionamiento del equipo, los diferentes módulos que lo conforman, software utilizado y las diferentes etapas de la rutina de reconocimiento.

En el **Capítulo 4** se reportan las diferentes pruebas realizadas para validar el funcionamiento de la rutina, como también sus respectivos resultados.

Por último, se dará una serie de conclusiones basadas en los resultados obtenidos y numerosas recomendaciones para futuros proyectos.

CAPITULO I

1.1. PLANTEAMIENTO DEL PROBLEMA

Actualmente, en Venezuela, miles de vehículos son robados regularmente. La recuperación del vehículo presenta una gran dificultad ya que no existe un sistema automatizado que permita localizar, e incluso, obtener los datos del mismo.

Hoy en día, existen múltiples maneras para rastrear el vehículo, siendo la más utilizada el sistema GPS. Sin embargo, a pesar de que este sistema resulte eficiente, los costos de instalación y servicio de localización son elevados, por lo que pudiera estar fuera del alcance del usuario o ciudadano promedio. Adicionalmente, su instalación requiere de una intervención del sistema eléctrico del vehículo, el cual puede ser manipulado por el delincuente para desactivar el sistema de localización.

Los sistemas de reconocimiento de matrícula por cámara ofrecen numerosos beneficios. En comparación con el sistema GPS, los costos de instalación son más bajos, ya que sólo se necesita instalar un conjunto de cámaras en una determinada área, que pudiera ser un estacionamiento, casetas de vigilancia, semáforos, en múltiples vehículos, etc. No requiere de una intervención directa con el sistema eléctrico del vehículo. Teniendo este sistema de reconocimiento, se puede diseñar una plataforma automatizada para reportar vehiculos robados, la cual, ante un reporte, busque el vehículo a través del sistema de cámaras y pueda notificar a las autoridades pertinentes la ubicación del mismo y hacer seguimiento.

Debido a los motivos antes explicados, y con el fin de dar una posible solución a esta problemática, se plantea el diseño de un sistema que permita obtener el número de la matrícula de un vehículo a partir de una foto digital.

1.2. OBJETIVOS

1.2.1. Objetivo General

Diseñar un sistema de reconocimiento de matrículas vehiculares.

1.2.2. Objetivos Específicos

1. Determinar los componentes, piezas y partes que integrarán el sistema.
2. Implementar el microsistema con su respectiva cámara.
3. Desarrollar la rutina para el reconocimiento del área de la matrícula.
4. Desarrollar la rutina para el reconocimiento del número de la matrícula.
5. Realizar pruebas y ensayos para validar funcionamiento del sistema.

CAPÍTULO II

MARCO TEÓRICO

2.1. Imagen digital

Una imagen es un arreglo matricial que se utiliza para almacenar información en forma de contenido visual. En otras palabras, una imagen es una función bidimensional de intensidad de iluminación *f(x,y)*, donde "*x*" e "*y*" representan las coordenadas espaciales en la imagen y el valor de "*f*" en el punto *(x,y)* es proporcional a la intensidad de iluminación de la escena en ese punto [1]. Para una imagen multiespectral (a color), la función de intensidad *f(x,y)* es un vector, donde cada componente contiene el valor de la intensidad correspondiente a cada banda espectral (roja, azul y verde).

Una imagen digital se obtiene al discretizar una imagen, tanto en el dominio espacial como en el valor de la intensidad almacenada. Es representada como un conjunto de arreglos matriciales (uno por cada banda espectral). Dicha representación se puede visualizar en la Figura 1.

El valor de la intensidad de iluminación digitalizada se denomina *nivel de gris*. Cada elemento o celda que conforma la matríz se denomina *pixel* o *pel*, que significa "elemento de la imagen". La cantidad total de pixeles, o las dimensiones del arreglo matricial se denomina resolución. Las extensiones de archivo más utilizadas para almacenar una imagen digital son: BMP, GIF, JPG/JPEG, TIF/TIFF y PNG.

$$f(x,y) = \begin{bmatrix} f(1,1) & f(1,2) & \dots & f(1,N) \\ f(2,1) & f(2,2) & \dots & f(2,N) \\ \vdots & \vdots & & \vdots \\ f(N,1) & f(N,2) & \dots & f(N,N) \end{bmatrix}$$

Figura 1. Representación matricial de una imagen.

2.2.Procesamiento digital de imágenes

El procesamiento digital de imágenes es un conjunto de técnicas o métodos desarrollados para manipular la información contenida en la imagen. Estos métodos consisten en aplicar diferentes operadores sobre la imagen, con los cuales se busca lo siguiente [1]:

- Restauración de la imagen: Mejorar la calidad de la imagen en una forma objetiva, como por ejemplo eliminación de ruido.
- Mejoramiento de la imagen: Mejorar la calidad de la imagen en una forma subjetiva, como por ejemplo incrementar el contraste, crear distorsión, etc.
- Compresión de la imagen: Involucra utilizar la menor cantidad de bits posible para representar la imagen, sin comprometer la calidad de la misma, tales como reducción del tamaño, binarización.
- Extracción de objetos: Para hacer explícitas algunas características en la imagen que puedan llevar a la detección de objetos en la misma, tales como la utilización de algoritmos descriptores.

2.2.1. Restauración de la imagen. Ruido y Filtrado.

En cualquier procesamiento de una señal eléctrica, se requiere de una etapa de filtrado para eliminar elementos no deseados, los cuales introducen ruido aditivo, interferencia aditiva y multiplicativa, imagen borrosa y de bajo contraste [1]. Estos elementos de ruido pueden tener diferentes causas, de las cuales destacan [2]:

- Falta de Iluminación.
- Características de la óptica de la cámara.
- Características del canal de transmisión.
- Condiciones generales del proceso de captura.

Los principales tipos de ruido que pudieran estar presentes en la imagen son los siguientes:

- Ruido Gaussiano: Este tipo de ruido es causado por efectos asociados a la electrónica de la cámara. Este tipo de ruido afecta la intensidad del pixel. En la Figura 2 se muestra la distribución Gaussiana del valor del píxel ante la presencia de ruido Gaussiano.

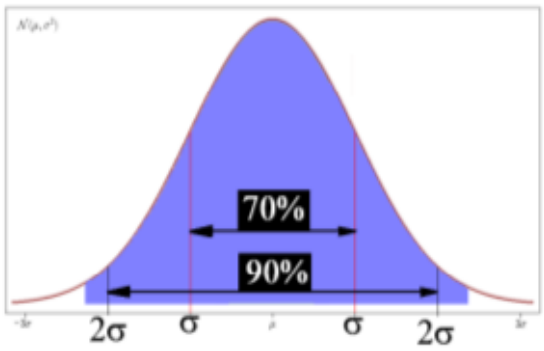

Figura 2. Distribución Gaussiana del valor del pixel.

En la siguiente Figura 3 se puede observar el efecto del ruido Gaussiano sobre la imagen.

Figura 3. Ruido Gaussiano. A la izquierda se muestra la imagen original, a la derecha la imagen con ruido Gaussiano.

- Ruido Impulsional (Sal y Pimienta): En la imagen se observan pixeles con valores muy elevados (255 si es una imagen de 8 bits) o muy bajos. En la Figura 4 se muestra el efecto de dicho ruido.

Figura 4. Ruido Impulsional. A la izquierda se muestra la imagen original, a la derecha la imagen con ruido "Sal y Pimienta".

El proceso de filtrado de una imagen se realiza mediante la convolución entre los distintos pixeles de la imagen y una matriz de convolución, la cual se denomina el "kernel" del filtro. Dependiendo del valor de cada elemento del kernel, se pueden obtener diferentes efectos en la imagen, tales como filtrado, detección de bordes, difuminación de la imagen, transformaciones morfológicas tales como erosión, dilatación, entre otras. En la Figura 5 se muestra la convolución entre matrices.

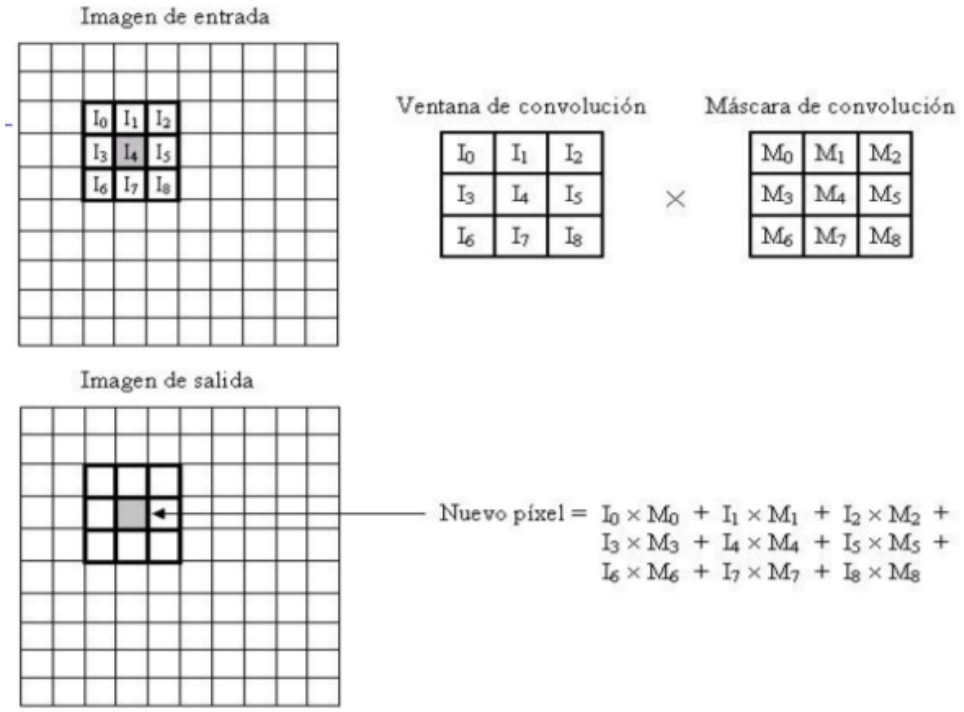

Figura 5. Convolución de matrices.

Los filtros se pueden clasificar según su linealidad (lineales y no lineales), según su función (suavizado, realce, diferenciales) y según las frecuencias que rechazan (pasa-bajo, pasa-alto y pasa-banda) [3]. Cabe destacar que si todos los elementos del kernel son positivos, el filtro se considera pasa-bajo, y se utiliza para "suavizar" la imagen. Si el kernel posee algunos elementos negativos, se considera pasa-alto, y se utiliza para detección de bordes y realce. Algunos filtros comunes son:

- Filtro Gaussiano: Se denomina Gaussiano ya que los elementos del kernel se aproximan a una distribución Gaussiana. Se utiliza para eliminar componentes de alta frecuencia [4]. Crean un efecto de "suavizado" en la imagen. Se muestra un ejemplo de filtro Gaussiano en la Figura 6, y su efecto sobre una imagen en la Figura 7.

$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Figura 6. Ejemplo de un kernel Gaussiano de 3x3.

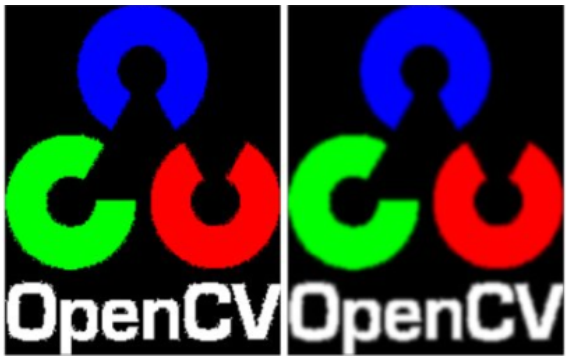

Figura 7. Efecto de filtro Gaussiano. A la izquierda se muestra la imagen original, a la derecha la imagen filtrada.

- Filtro de Media: Este tipo de filtro busca sustituir el valor del pixel central por la media aritmética de los pixeles vecinos. Se utiliza para eliminar el ruido impulsional [4]. En las Figuras 8 y 9 se pueden observar un ejemplo de filtro de media y su efecto sobre la imagen respectivamente.

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Figura 8. Ejemplo de un kernel de filtro de media de 3x3.

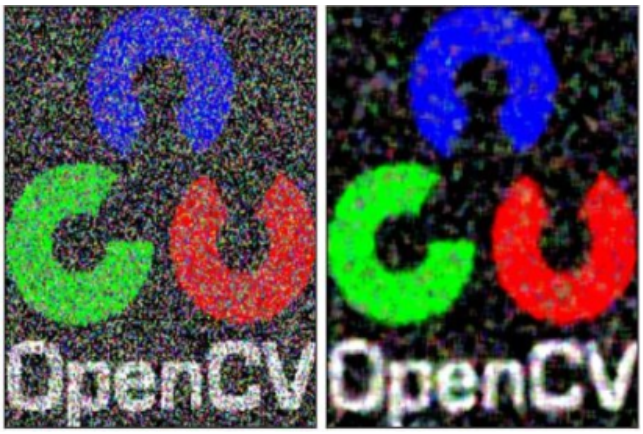

Figura 9. Efecto de filtro de Media. A la izquierda se muestra la imagen original, a la derecha la imagen filtrada.

2.2.2. Mejoramiento de la imagen. Distorsión.

La distorsión se puede definir como una alteración o deformación de la imagen. Este efecto es causado por las características de la lente y por las condiciones del proceso de captura de la imagen [1]. Los tipos de distorsión más comunes son:

- Distorsión radial u homogénea: Este tipo de distorsión depende de las características de la lente. Existe dos tipos de efectos: el efecto “cojín”, el efecto “barril” y el efecto “bigote”, el cual es una combinación de los anteriores.

- Distorsión tangencial o no homogénea: Este tipo de distorsión depende de las condiciones de captura, también conocido como distorsión por perspectiva. Dependiendo de las condiciones, objetos en la imagen pueden tener diferentes dimensiones a las reales, y líneas que son rectas se observan inclinadas.

Los diferentes tipos de distorsión antes mencionados se ilustran en las Figuras 10 y 11.

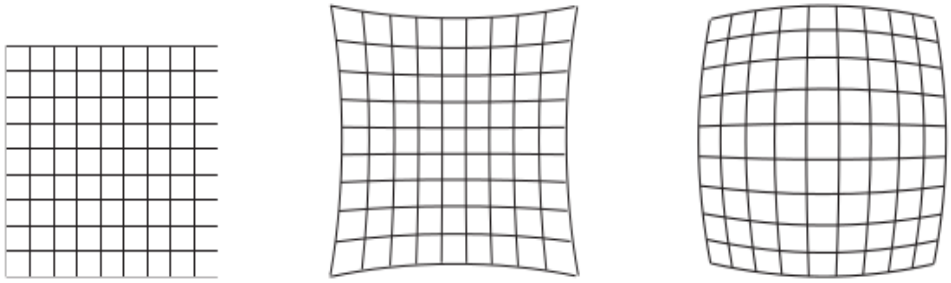

Figura 10. Distorsión Radial. De izquierda a derecha: Imagen original, efecto "cojín" y efecto "barril".

Figura 11. Distorsión de perspectiva.

Para compensar los efectos de distorsión, se realiza una transformación espacial seguida de una interpolación de nivel de gris [1].

- Transformación espacial: Este tipo de transformación busca corregir la ubicación del pixel. Suponga la posición correcta del pixel como (x_C,y_C) y la posición en la imagen distorsionada como (x_D,y_D), existirá una transformada que relacionará los dos conjuntos de coordenadas:

$$\begin{aligned} x_D &= O_X(x_C, y_C) \\ y_D &= O_Y(x_C, y_C) \end{aligned} \tag{1}$$

El tipo de transformada (global o local) a utilizar dependerá del tipo de distorsión presente (homogénea o no homogénea).

Para el caso de la distorsión homogénea, la cual es causada por las características de la lente, suele ser radialmente simétrica. Si se define el radio *r*, medido desde el centro de la imagen, como $r = \sqrt{x^2 + y^2}$, la relación entre el radio de la imagen distorsionada y la imagen corregida cumplen con la siguiente relación:

$$r_D = f(r_C) \tag{2}$$

La función $f(r_C)$ debe cumplir con las siguientes características:

1. La distorsión en el centro de la imagen debe ser nula, incrementando su valor hacia la periferia de la imagen.
2. Debe ser invertible.
3. Debe ser antisimétrica en "*x*" e "*y*".

Para cumplir con las condiciones antes mencionadas, se suele seguir el

siguinete modelo:

$$\begin{aligned} x_D &= x_C f(r_C) \\ y_D &= y_C f(r_C) \end{aligned} \tag{3}$$

Donde la función $f(r_C)$ depende de las características de la lente, como por ejemplo:

$$\begin{aligned} f(r_C) &= 1 + k_1 r_C \\ f(r_C) &= 1 + k_1 r_C + k_2 r_C^2 \\ f(r_C) &= \frac{1 + k_1 r_C}{1 + k_2 r_C^2} \end{aligned} \tag{4}$$

Para compensar el efecto de la distorsión homogénea, se requiere la utilización de patrones de calibración. Para el caso de la distorsión no homogénea, se suele utilizar la siguiente transformada:

$$\begin{aligned} x_D &= c_1 x_C + c_2 y_C + c_3 x_C y_C + c_4 \\ y_D &= c_5 x_C + c_6 y_C + c_7 x_C y_C + c_8 \end{aligned} \tag{5}$$

Donde $c_1, c_2, \ldots, c_8$ son parámetros que dependen de las características del proceso de captura, los cuales se determinan a través de la relación entre la ubicación en la imagen distorsionada y la imagen corregida de puntos clave.

- Interpolación de nivel de gris: El valor del pixel en la imagen corregida no puede tener el mismo valor que su equivalente en la imagen distorsionada, por lo que se debe aplicar un proceso de interpolación para determinar el nuevo valor del pixel en la imagen corregida, como por ejemplo interpolación bilineal, interpolación de vecinos cercanos, entre otras.

2.2.3. Compresión de la imagen. Binarización y Erosión.

Las imágenes digitales, generalmente, presentan un amplio rango valores de intensidad de pixel, que para el caso de una imagen de 8 bits, cada uno puede tener 256 valores diferentes [5]. Para reducir o limitar la cantidad de información analizada, se aplica un proceso denominado binarización, en el cual se compara el valor de intensidad del pixel con un valor de referencia, o umbral, y si resulta superior a esta referencia, se le asigna el valor lógico "1", y en caso contrario, se le asigna "0". Realizar el proceso de binarización presenta varias ventajas:

- Dado que se reduce la cantidad de información a procesar, se aprovecha mejor la capacidad de procesamiento del computador.
- Muchas de las propiedades geométricas de los objetos presentes en la imagen se pueden obtener rápida y fácilmente.

Existen 2 tipos de binarización [6]: la no adaptativa, en la cual se define el valor de umbral de manera arbitraria, esta siendo útil cuando los niveles de iluminación son constantes; y la adaptativa, en la cual se define el valor de umbral tomando como referencia el valor de los diferentes pixeles que conforman la vecindad de un pixel central, esta última se utiliza cuando los niveles de iluminación son variables.

Existen dos métodos de binarización adaptativa [6]:

- Media: En este tipo de binarización, el valor de umbral corresponde con la media aritmética de los pixeles circundantes.
- Gaussiana: En este tipo de binarización, el valor de umbral corresponde con la sumatoria de todos los coeficientes o "pesos" de los pixeles circundantes, los

cuales conforman una ventana gaussiana. Se muestra el efecto de diferentes binarizaciones en la Figura 12.

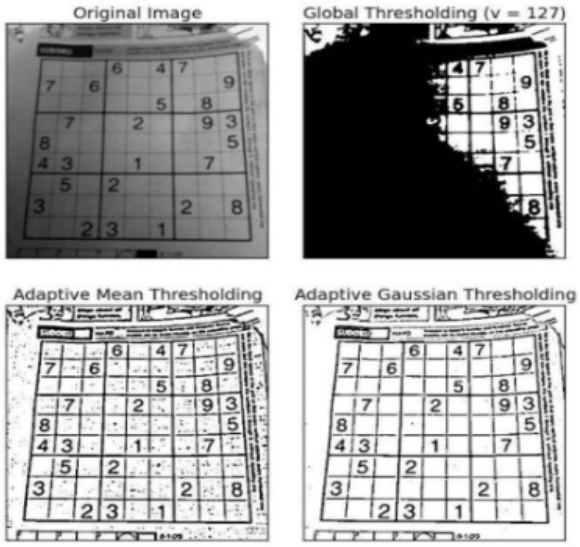

Figura 12. Diferentes procesos de binarización.

Una vez obtenida la imagen binaria, se pueden realizar diferentes operaciones para manipular su contenido. Una de las operaciones básicas que se pueden aplicar es la erosión [7], con la cual se busca definir un trazado más fino ("adelgazar" el trazado) de los diferentes contornos de la imagen binaria (Figura 13), con el fin de eliminar ciertos elementos no deseados y acentuar las características más importantes de algún objeto determinado.

Figura 13. Erosión de la imagen.

2.2.4. Extracción de características. Patrones Binarios Locales.

Los patrones binarios locales, también conocidos como características locales, o LBP, es un algoritmo descriptor que se utiliza para caracterizar imágenes u objetos de acuerdo a su textura o superficie. Este algoritmo fue implementado en el trabajo de Ojala y otros [8], donde se requería de un sistema para clasificar texturas. En versiones iniciales de este algoritmo, se realizaba la operación en matrices 3x3, en donde se ejecutaba la siguiente secuencia de pasos:

1. Se localiza el pixel central. Posterior a esto, se comparan los distintos valores de los pixeles de la vecindad con el valor del pixel central, y se construye una nueva matríz, cuyos valores de pixeles corresponden a "1" si el valor del píxel es mayor que el píxel central, y "0" de caso contrario.
2. Se crea una matríz de "pesos" (peso=2^k, $0 \leq k \leq 7$), donde la posición de cada uno se asigna arbitrariamente.
3. Los valores de los pixeles de la matríz "umbralizada" se multiplican por los valores de los pixeles de la matríz de "pesos". Esto operación arroja como resultado la matríz LBP.

A continuación, se ilustra los pasos mencionados anteriormente en el siguiente ejemplo:

$$\begin{bmatrix} 6 & 5 & 2 \\ 7 & \mathbf{6} & 1 \\ 9 & 8 & 7 \end{bmatrix}$$

Figura 14. Matríz ejemplo.

Al comparar los pixeles de la vecindad con el pixel central, se obtiene la matríz de umbral mostrada en la Figura 15:

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

Figura 15. Matríz de umbral.

Posterior a esto, se define la matríz de "pesos" (peso=2^k, $0 \leq k \leq 7$). Para este ejemplo, se asignó la posición en sentido horario. Dicha matriz se ilustra en la Figura 16.

$$\begin{bmatrix} 1 & 2 & 4 \\ 128 & & 8 \\ 64 & 8 & 16 \end{bmatrix}$$

Figura 16. Matríz de pesos.

Se realiza la siguiente operación para obtener la matríz LBP (ver Figura 17):

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & & 0 \\ 1 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 2 & 4 \\ 128 & & 8 \\ 64 & 8 & 16 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 128 & & 0 \\ 64 & 8 & 16 \end{bmatrix}$$

Figura 17. Matríz LBP.

Con la última operación realizada, se obtiene el patrón binario de la matríz: 11110001. Se muestra la aplicación del algoritmo descriptor sobre una imagen en la Figura 18.

Figura 18. Aplicación de LBP para describir textura.

2.3. Visión Computarizada

También conocida como visión artificial o visión por computador. Es una disciplina que involucra múltiples áreas, tales como la inteligencia artificial, óptica, procesamiento digital de imágenes, entre otras [1]; y se encarga de definir métodos o

algoritmos para adquirir, procesar y analizar imágenes fotográficas para generar información numérica que pueda ser tratada por un computador. Actualmente, existen múltiples aplicaciones relacionadas con esta área, entre las que destacan:

1. Reconocimiento, localización y seguimiento de objetos, tales como rostros, personas, entre otros.
2. Reconstrucción de escenas, también conocida como mapeo, o *Mapping*.
3. Clasificación de imágenes según su contenido.

2.3.1. Técnicas de localización de objetos en la imagen.

Con este conjunto de técnicas se realiza el proceso búsqueda del objeto de interés en la imagen. La etapa de búsqueda es la más importante de toda la rutina, y es la que más recursos computacionales exige, esto último se debe a las siguientes factores:

- Se desconoce la ubicación exacta del objeto: Esto quiere decir que el proceso de búsqueda se debe realizar en toda la imagen. Por lo general, en este tipo de aplicaciones, se asume o se limita la zona de búsqueda alrededor de un sector de la imagen, también conocida como región esperada de detección.

- Tamaño del objeto: El objeto de interés puede estar a cualquier distancia de la cámara, por lo que en la imagen se observará que el objeto ocupa un mayor porcentaje de la misma (cerca de la cámara), o un porcentaje menor (lejos de la cámara).

Para realizar el proceso de búsqueda, tomando en consideración los factores antes descritos, se plantea una combinación de dos técnicas de localización: ventana deslizante y pirámide de imágenes.

a) Ventana deslizante

Con este tipo de técnica, se desea localizar el objeto de interés haciendo un barrido a lo largo de toda la imagen [9]. Para ello, se debe utilizar una ventana de búsqueda con una dimensión constante (por ejemplo, 24x24 pixeles). Con este proceso de exploración se busca dividir la imagen en múltiples subconjuntos, los cuales serán procesados para determinar la presencia del objeto de interés en la imagen. El análisis del subconjunto de imágenes requiere de una menor capacidad de procesamiento, ya que se reduce la cantidad de información (pixeles) a procesar. La función en cuestión sigue el siguiente algortimo (Figura 19):

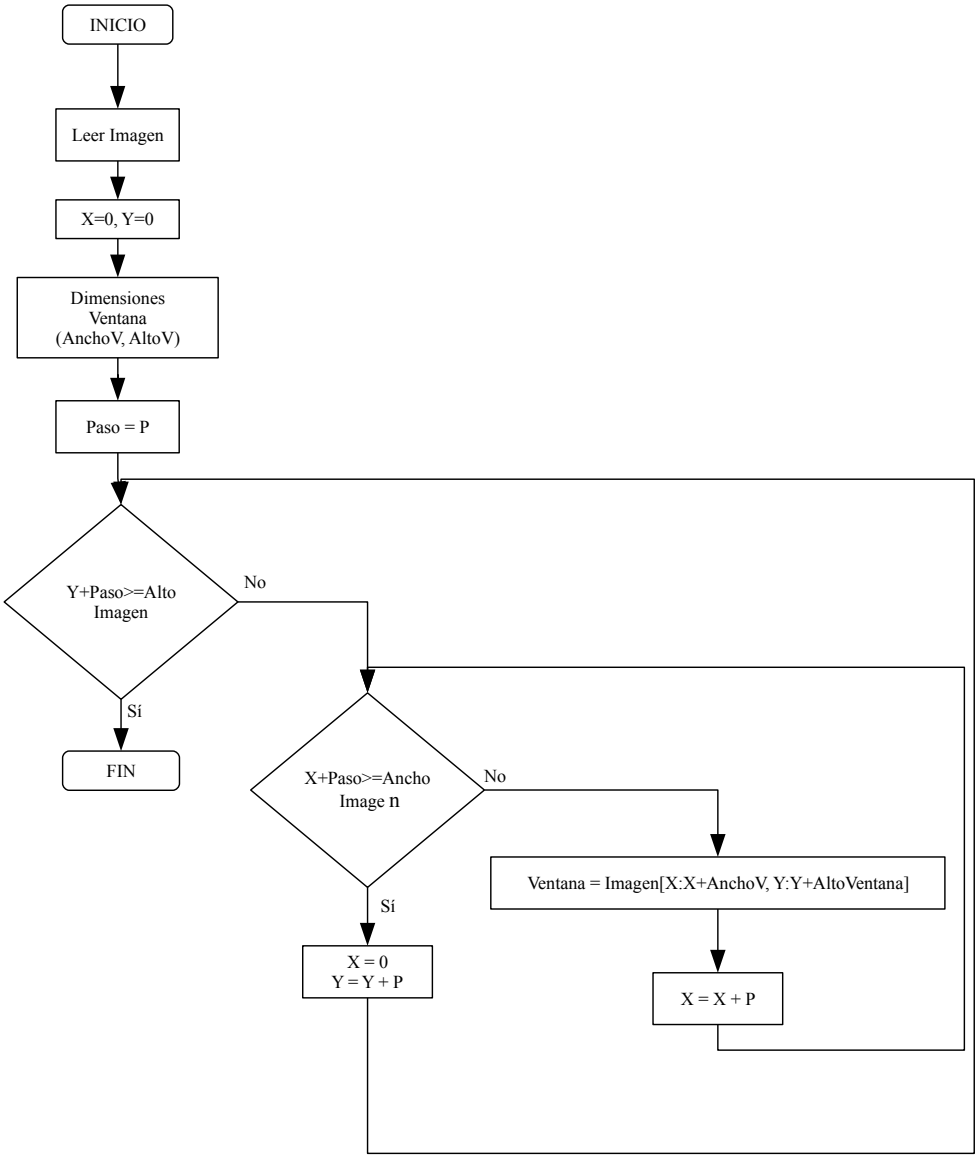

Figura 19. Diagrama de flujo ventana de búsqueda.

Como se puede observar, a medida que se generen los valores de (*x,y*), se van extrayendo las subimágenes de la imagen principal.

b) Pirámide de imágenes

La pirámide de imágenes es una técnica que se utiliza para detectar objetos cuando no se conoce con certeza a que distancia se encuentra, por lo que las dimensiones del objeto pudieran ser superiores a las dimensiones de la ventana de búsqueda [10]. Para ello, se produce una serie de copias de la misma imagen, ajustadas a menores resoluciones (Figura 20), tal que los objetos más cercanos a la cámara se detecten en las imágenes de menor resolución, mientras que los objetos más lejanos se detecten en las imagenes de mayor resolución. La resolución de cada nivel se define como la resolución del nivel inferior entre un factor de reducción, el cual es constante. El proceso de creación de la pirámide de imágenes sigue el siguiente algoritmo (Figura 21):

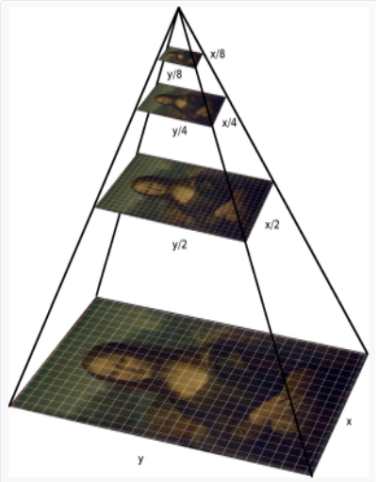

Figura 20. Ejemplo de una pirámide de imágenes.

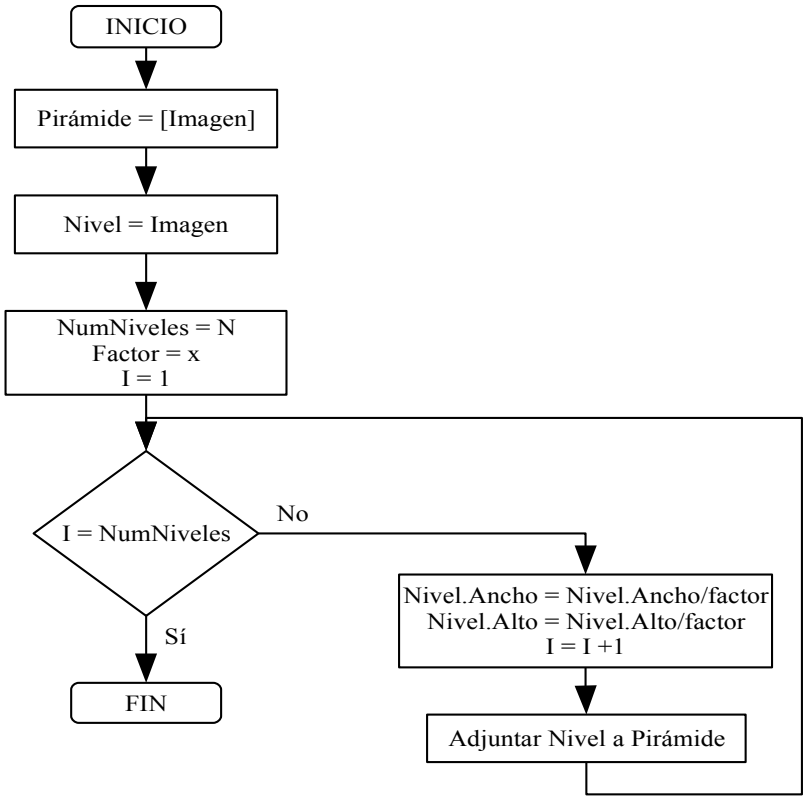

Figura 21. Diagrama de flujo de la pirámide de imágenes.

2.3.2. Reconocimiento de objetos. Clasificadoras.

Para detectar cualquier tipo de objeto en una imagen, se requiere utilizar un algoritmo descriptor, con el cual se define una serie de características que describen el objeto a detectar; y una clasificadora o máquina de aprendizaje supervisado, la cual debe ser entrenada con este conjunto de datos para el reconocimiento del objeto en una escena.

a) Clasificadora en Cascada

La clasificadora en cascada es un tipo de máquina de aprendizaje supervisado, la cual recibe un conjunto de datos de entrenamiento, y utilizando algoritmos de aprendizaje definirá una función o una serie de condiciones que le permitirá predecir a que conjunto pertenece un determinado dato de entrada. Este tipo de clasificadora fue implementada en [11], donde se requería de una clasificadora para detección de rostros que pudiera ser utilizada en sistemas de bajos recursos, tales como teléfonos, cámaras, entre otros. Se dice que la clasificadora es "en cascada", ya que está conformada por una serie de clasificadores más débiles, los cuales forman distintos niveles o etapas de clasificación. En cada etapa de clasificación, se realiza un proceso de evaluación del conjunto de muestras, y aquellas que cumplan con los criterios establecidos en la etapa, pasan al siguiente nivel de clasificación (ver Figura 22).

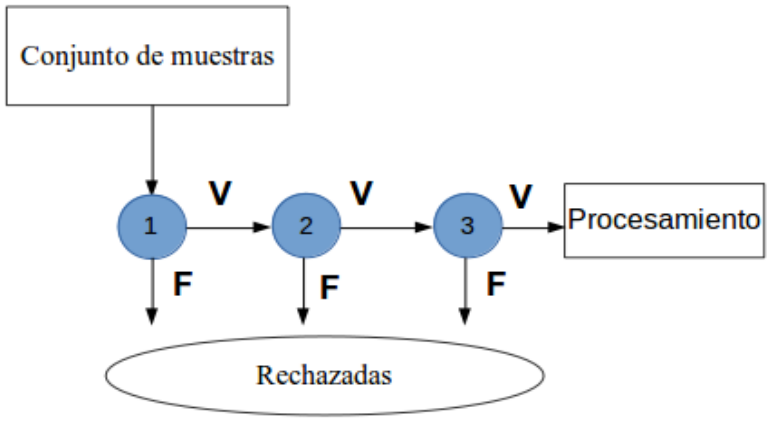

Figura 22. Diagrama de la clasificadora en cascada.

Tomando como ejemplo la Figura 22, la primera etapa descarta la mayor cantidad de muestras negativas, con muy poco procesamiento. Las etapas subsecuentes eliminan más muestras negativas, pero requieren de mayor procesamiento. Una vez finalizado el análisis multietapa, la cantidad de muestras ha

sido reducida drásticamente, por lo que se puede realizar análisis más complejos sobre este limitado conjunto de muestras.

Una imagen puede contener cientos de miles de características, o variables, que describen el objeto de interés. Para poder realizar el entrenamiento de la clasificadora, se deben limitar el número de características a estudiar, o en otras palabras, definir las variables más importantes que describen el objeto. Para ello, se utiliza un algortimo de optimización conocido como *AdaBoost* [12] (*Adaptive Boosting*).

Para cada etapa de entrenamiento, se selecciona un conjunto de muestras positivas y negativas, y se comparan las distintas muestras para determinar cuales son las variables que diferencia mejor los conjuntos, y en base a esto, se define las características de la clasificadora correspondiente a la etapa. Sin embargo, la clasificadora de la etapa no es perfecta, es posible que algunas muestras hayan sido clasificadas erróneamente, por lo que en el entrenamiento de la siguiente etapa, se hace más enfásis en estas muestras, y así sucesivamente hasta que se alcance el número de etapas deseado, o se alcance un porcentaje de error mínimo. Un ejemplo de reconocimiento de objetos se ilustra en la Figura 23.

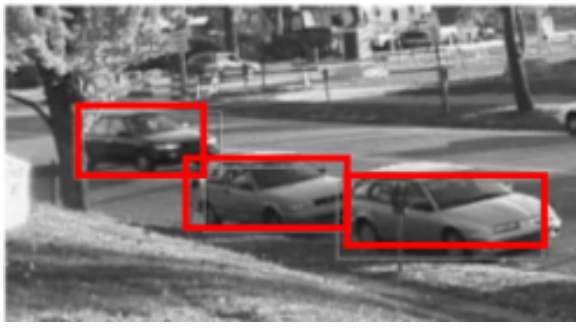

Figura 23. Detección de vehículos con la clasificadora en cascada.

b) Reconocimiento Óptico de Caracteres (OCR)

El reconocimiento óptico de caracteres, también conocido como OCR, es un caso particular de detección de objetos, en donde se utiliza un máquina de aprendizaje supervisado, previamente entrenada, para reconocer caracteres (letras, números, signos de puntuación, etc.). Este tipo de software se ha utilizado como una iniciativa para digitalizar libros o documentos a partir de una foto. Adicionalmente, se han empleado en diferentes aplicaciones de reconocimiento de texto, como por ejemplo: análisis de documentos de identificación, detección de señalización, obtención de números de matrícula, entre otras.

2.4. Objeto a detectar: Matrícula Vehicular

La matrícula es un elemento de identificación único para cada vehículo. En todo el mundo, los vehículos tienen prohibido transitar por la vía pública sin su respectiva identificación. En Venezuela, el organismo encargado de otorgar esta identificación y definir las leyes de tránsito es el Instituto Nacional de Transporte Terrestre (INTT).

Las características de las matrículas vehiculares están estipuladas en la Ley de Tránsito Terrestre, las cuales fueron aprobadas en la Gaceta Oficial N° 38944 del 3 de junio del 2008 [13]. En ella, se definen varias consideraciones y lineamientos, tales como las características de las matrículas dependiendo del tipo de vehículo (motocicletas, automóviles, servicio de carga, entre otras), mecanismos de seguridad, asignación de caracteres, etc. Entre todos estos lineamientos, destaca lo siguiente:

1. Las matrículas se clasifican en especiales y ordinarias.
2. Las matrículas ordinarias tendrán una serie de elementos de seguridad que

imposibilitan cualquier intento de falsificación o reproducción, tales como:

- Imágenes de seguridad tipo trenzas, que consistirán en 2 ondas sinusoidales en el centro de la placa, dispuestas verticalmente.
- Imágenes del mapa de la República Bolivariana de Venezuela, visibles únicamente a 30° sobre la perpendicular.
- Cinta de estampado en caliente, la cual cubrirá los caracteres alfanuméricos ubicados en el centro de la matrícula. El estampado contará con la palabra "VENEZUELA" con letra Arial Bold, de 5.5 mm.
- Cinta de estampado en caliente que cubrirá todas las demás partes de la matrícula, troqueladas a relieve.
- Código de identificación hecho a través de un sistema de marcación a láser.
- Características traslucidas especiales de las tintas, que pemiten el paso de luz, asegurando la retro-reflectividad de la lámina.

3. Las matrículas para automóviles, minibuses, camionetas de pasajeros y casas rodantes cumplen con las siguientes características:

Tabla 1. Especificaciones matrícula vehicular.

Dimensiones	Caracteres	Color	Transfondo
300mm x 150 mm	2 letras, 3 números, 2 letras	Fondo Blanco Caracteres Azules	Diseño difuminado de la bandera tricolor nacional

En la Figura 24 se muestra un ejemplo de matrícula para vehículos particulares en Venezuela.

Figura 24. Ejemplo de una matrícula vehicular venezolana.

2.5. OpenCV

OpenCV es una biblioteca libre de visión artificial originalmente desarrollada por Intel. Se ha utilizado en múltiples aplicaciones, desde sistemas control de acceso con reconocimiento facial, hasta aplicaciones de reconocimiento y seguimiento de objetos. Esto se debe a que su distribución se da bajo la licencia BSD, la cual permite que la librería sea usada libremente para propósitos académicos y/o comerciales. La librería en cuestión es multiplataforma, tiene versiones para C++, Python, Java, entre otras, y es compatible con los sistemas operativos Windows, Mac OSX y GNU/Linux. Ofrece infinidad de funciones para filtrado de imágenes, calibración de cámara, reconocimiento de objetos, visión estéreo y visión robótica [14].

2.6. Python

Python es un lenguaje de programación [15] multiplataforma e interpretado, es decir, en vez de utilizar un programa que ejecute una etapa de compilación, se ejecuta directamente a través de un programa "intérprete". Fue creado en la década de 1990 por Guido Van Rossum en el Centro de Fundamento Matemático (CWI) en los Países Bajos, como un sucesor al lenguaje de programación ABC. Actualmente, el software

es distribuido, de manera gratuita bajo la licencia GPL, por la Python Software Foundation. Este lenguaje de programación ofrece múltiples ventajas, entre las que destacan:

- Evita el uso de caracteres especiales como { } $ / y \.
- Utiliza espacios en blanco para definir lineas de indentación para control del programa, en vez de utilizar caracteres como llaves ({}).
- Facilita la lectura y comprensión del programa.

2.7. Antecedentes

Los sistemas de reconocimiento de matrícula se han implementado en múltiples proyectos. Estos sistemas se basan en la utilización de una cámara, la cual está posicionada de tal forma que se garantice un enfoque directo a la matrícula, y un computador o servidor que reciba la información de dicha cámara para su posterior procesamiento. Cabe destacar que, existen 2 tipos de sistemas de reconocimiento de matrícula:

- Sistemas LPR: Este tipo de sistema solo permite obtener el número de la matrícula, no realiza ningún proceso de localización. En este tipo de sistema, es conocida la ubicación de la matrícula y se realiza un proceso de recorte y análisis alrededor de esa ubicación.

- Sistemas ALPR: En este tipo de sistemas no se conoce la ubicación de la matrícula. Se realiza un proceso de exploración en toda la imagen, o alrdedor de una región, para obtener la ubicación de la matrícula. Posterior a esto, se realiza el proceso de recorte y análisis como en los sistemas LPR.

El sistema de reconocimiento diseñado por Sajjad [16] aprovechaba las características de color de la matrícula para localizarla. Con respecto al proceso de captura, se posicionó la cámara de tal forma que se pudiera enfocar únicamente la parte frontal o posterior del vehículo. Realizando un proceso de binarización, y aprovechando las características de color de la matrícula, se logró obtener la localización de la misma, a la cual, se le aplicó el OCR para obtener dicho número. Cabe destacar que se utilizó Python y la librería de visión artificial OpenCV en dicho proyecto.

En el trabajo de Xin Li [17], se explica el diseño de un sistema de reconocimiento de matrículas. La técnica de exploración en la imagen es una combinación entre la técnica de ventana deslizante con pirámide de imágenes. Se utlizó el algoritmo descriptor HOG para definir las características más importantes de la matrícula, y con esos datos, realizó el entrenamiento de la clasificadora, la cual era un tipo de SVM. Adicionalmente, realizó el entrenamiento de una segunda clasificadora para el reconocimiento de los caracteres de la matrícula. Se realizaron diversas pruebas para determinar el desempeño de la rutina bajo diferentes condiciones de ruido y movimiento del objetivo.

Por último, en el trabajo de Soler [18], se diseñó una rutina de localización y reconocimiento de matrículas vehiculares. Se utilizó el algoritmo descriptor LBP para definir las características más importantes de la matrícula, con las cuales se realizó el proceso de entrenamiento de una clasificadora. En dicho proyecto, el proceso de exploración se basó en la técnica de ventana deslizante. La rutina en cuestión fue programada en C++, utilizando las librerías QT4, IPP y OpenGL, junto con Qvision (un framework para procesamiento de imágenes) y Octave.

CAPÍTULO III

DESCRIPCIÓN DEL PROYECTO

3.1. Descripción de hardware

Los sistemas de reconocimiento visual, usualmente, están conformados por una cámara con su respectivo módulo de control. El módulo en cuestión, dependiendo de su capacidad de procesamiento, puede analizar la imagen, o pudiera enviarla a un servidor de mayor capacidad para su procesamiento. Se deben considerar una serie de factores para la selección de los diferentes módulos que conformarán el equipo, tales como:

- Condiciones de iluminación.
- Ubicación y orientación.
- Características de la lente.
- Ubicación esperada del objeto a detectar.

Dado que, para el presente proyecto, no se tomaron en consideración las diferentes condiciones bajo las cuales se realiza el proceso de captura de la imagen, el criterio de selección de los diferentes módulos se limitó a minimizar costos para lograr el diseño e implementación de un prototipo. Con esto en mente, se plantea el esquema básico del prototipo a diseñar (ver Figura 25):

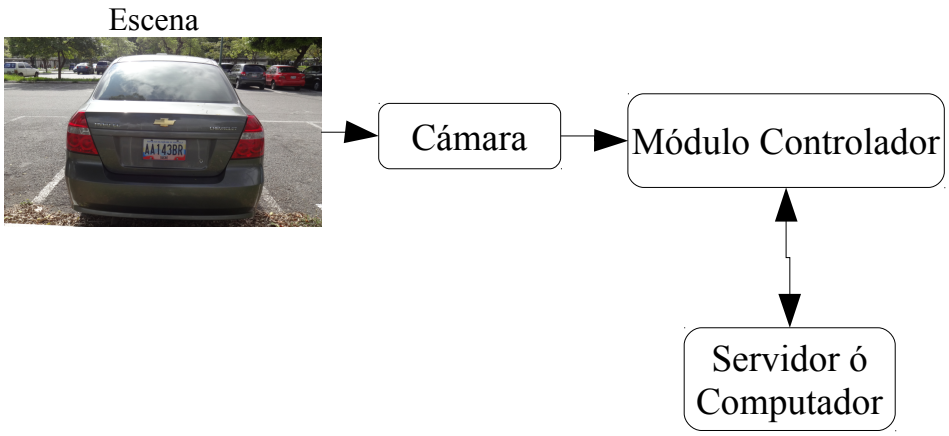

Figura 25. Esquema del equipo.

En base a las justificación antes mencionada, la cámara seleccionada fue una webcam USB de 2MP fabricada por la empresa TOOGOO (Figura 26), la cual presenta las siguientes características:

Tabla 2. Especificaciones de cámara USB.

Caracterísitca	Especificaciones
Fabricante	TOOGOO
Modelo	LE 20.0
Resoluciones Soportadas	640x480, 352x288, 320x240, 176x144, 160x120
Comunicación	USB 2.0
Memoria	No disponible
Batería	No disponible
Funciones especiales	Micrófono incorporado 6 LEDs alto brillo

Una de las ventajas que ofrece la cámara, además de su costo reducido, es que no requiere la instalación adicional de un software controlador o driver, por lo que es compatible con múltiples equipos y sistemas operativos.

Figura 26. Cámara digital.

Sin embargo, dado que la cámara soporta resoluciones muy bajas, se decidió utilizar otra cámara con mayor resolución para la adquisición de fotos de prueba para la rutina de reconocimiento. La cámara en cuestión presenta las siguientes características:

Tabla 3. Especificaciones Cámara CyberShot

Características	Especificaciones
Fabricante	SONY
Modelo	Cyber-Shot DSC-WX100
Resoluciones Soportadas	18MP, 13MP, 10MP, 5MP, 2MP y VGA
Conexión	USB 2.0
Memoria	Tarjeta SD-HC. 8GB. 15MB/s
Batería	NP-BN. 3,6V/630MAh.
Funciones Especiales	Zoom óptico hasta 10x Flash Opciones de auto-enfoque Pantalla LCD 460800 puntos

Cabe destacar que se desactivaron todas las funciones de auto-enfoque, mejoramiento de la imagen, zoom óptico y flash de la cámara para realizar el proceso de adquisición de muestras.

El módulo controlador seleccionado fue la tarjeta de desarrollo Raspberry Pi 2 modelo B (Figura 27), ya que cumple con los requerimientos mínimos para el diseño de un prototipo. La tarjeta de desarrollo en cuestión tiene las siguientes especificaciones:

Tabla 4. Especificaciones Raspberry Pi 2 modelo B.

Componente	Especificaciones
CPU	Broadcom BCM2836 900MHz quad-core ARM Cortex-A7
RAM	1GB
Puertos USB 2.0	4 puertos
Almacenamiento	Memoria MicroSD-HC. 16GB
Comunicación	Ethernet 10/100 Mbps
Fuente de Alimentación	5V/300mA. 3W. Micro USB
Sistema Operativo	Raspbian Jessie

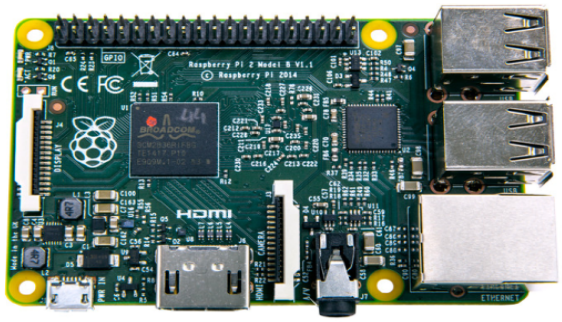

Figura 27. Raspberry Pi 2 modelo B.

Adicional a esto, el módulo Raspberry Pi [19] debe enviar la información a un computador con mayor capacidad de procesamiento. En el presente proyecto, se plantea enviar la información utilizando el protocolo de comunicación SSH. Para lograr el acceso remoto al módulo Raspberry Pi, se utilizó el siguiente adaptador:

Tabla 5. Especificaciones adaptador Wi-Fi.

Características	Especificaciones
Fabricante	EDIMAX
Modelo	EW-7811Un
Requisitos mínimos del sistema	Puerto USB 2.0, Windows 2000/XP/Vista/7/8/8.1, Mac OS 10.4-10.9, Linux Disco Duro de 100MB

Finalmente, las características del computador utilizado son las siguientes:

Tabla 6. Especificaciones Computador.

Característica	Especificaciones
Fabricante	ACER
Modelo	Aspire One D270
CPU	Intel Atom N2600
Almacenamiento	Disco Duro de 500GB
RAM	2 GB
Puerto USB	3 Puertos USB 2.0
Comunicación	Acer Nplify 802.11b/g/n
Sistema Operativo	Ubuntu 14.04.4 LTS Trusty Tahr con LXDE desktop

3.2. Descripción de software

A continuación, se detalla el software utilizado para el desarrollo del presente proyecto:

- Python: Se decidió utilizar este lenguaje de programación, ya que ofrece facilidades de uso y es de fácil comprensión. Versión: 2.7.4.

- OpenCV: Es una librería gratuita que contiene diferentes funciones para el procesamiento de imágenes. Versión: 2.4.8.
- Numpy: Numpy es un módulo básico de Python, el cual ofrece funciones para la resolución de operaciones matemáticas, facilita el proceso de manipulación de la imagen a través de operaciones matriciales. Versión: 1.8.2.
- Tesseract-OCR: Tesseract es un OCR gratuito propiedad de Google. Este software es utilizado para convertir el texto presente en la imagen en una variable de tipo *string* o cadena de caracteres. Versión: 3.03.
- Leptonica: Es un librería especializada en funciones de procesamiento de imágenes. Es un requisito tener instalada esta librería para la utilización del Tesseract-OCR. Versión: 1.7.

3.3. Descripción de la rutina de reconocimiento

La rutina de reconocimiento presenta las siguientes etapas:

1. Localización de posibles áreas de la matrícula.
2. Reajuste de tamaño y filtrado de las posibles matrículas.
3. Binarización y erosión de la imagen.
4. Localización de área del número de la matrícula.
5. Reconocimiento de caracteres.

Cabe destacar que para la selección del filtro con sus respectivos parámetros y del método de binarización, se realizaron diferentes pruebas evaluando la calidad de detección del OCR. El uso de un filtro o método de binarización inadecuado puede llevar a la eliminación de información crucial para la clasificación de los caracteres, o se pudiera introducir elementos de ruido que interfieran en el proceso de clasificación.

3.3.1. Localización de posibles áreas de la matrícula.

En esta etapa se realiza una exploración en toda la imagen para localizar posibles áreas donde se encunentre la matrícula. Inicialemente, se define una pirámide de imágenes de 4 niveles, esto para garantizar que la matrícula pueda ser reconocida a diferentes distancias. Las dimensiones de las diferentes imágenes que conforman dicha pirámide, representan la mitad de las dimensiones del nivel inferior. Para el presente proyecto, se utilizaron fotos en formato .JPG con una resolución de 13MP (4896x2752 pixeles). En base a esto, la resolución de cada nivel, de menor a mayor, son las siguientes:

Tabla 7. Resolución de cada nivel.

Nivel	Resolución (Pixeles)
1	612x344
2	1224x688
3	2448x1376
4	4896x2752

Una vez definida la pirámide de imágenes, se realiza un proceso de búsqueda en los diferentes niveles, el cual consiste en utilizar una ventana deslizante que realice un barrido en toda la imagen. La definición de las dimensiones de la ventana dependerá de la resolución de la imagen original, particularmente, del tamaño del objeto de interés. La ventana de búsqueda no debería ser muy reducida, ya que si el objeto se encuentra a una corta distancia de la cámara, en las imágenes de mayor resolución tendrá una mayor dimensión que dicha ventana, y en las imágenes de menor resolución se pierden detalles del objeto, por lo que no se garantiza la detección del mismo; adicionalmente, se aumentaría la cantidad de muestras a analizar, por lo que aumentaría el tiempo de procesamiento. Analizando el caso contrario, la ventana de búsqueda no debería ser muy amplia, ya que el tiempo requerido para procesar cada muestra aumentaría considerablemente, y esto pudiera

conllevar a un aumento en la detección de falsos positivos (imágenes que son detectadas como matrículas, pero no presentan ninguna matrícula).

Tomando en consideración los aspectos antes mencionados, se realizó un proceso de ensayo y error para determinar las dimensiones más adecuadas de la ventana de búsqueda, evaluando el tiempo de ejecución y la cantidad de muestras obtenidas. Luego de haber realizado diferentes pruebas, se decidió utilizar una ventana de exploración de 1500x750 pixeles. La relación entre ancho y alto de la ventana debe ser tal que sea apróximadamente igual a la del objeto a detectar, que según la Tabla 1, esta relación es de 2:1.

Con el proceso de búsqueda antes mencionado, se obtiene un conjunto de subimagenes, las cuales deben ser analizadas para determinar si existe la presencia de una matrícula. Este conjunto de imágenes debe ser analizado por una clasificadora, la cual ha sido previamente entrenada para detectar el objeto en cuestión.

Para poder realizar el proceso de entrenamiento de cualquier clasificadora, se requiere de un conjunto de datos de entrenamiento, esto es, muestras o ejemplos del objeto que se desea detectar, a los cuales se les aplicará un algoritmo descriptor, para definir las características más relevantes. Para el presente proyecto, aprovechando las características de empastado (superficie) y mecanismos de seguridad en la matrícula vehicular, se utilizó el algoritmo descriptor LBP para definir las características o variables más importantes. En la Figura 28 se ilustra la aplicación del algoritmo sobre el objeto de interés.

Figura 28. Aplicación de LBP sobre la matrícula.

Se realizó un proceso de adquisición de muestras para el entrenamiento, el cual consistió en capturar imágenes de vehículos en diferentes estacionamientos al aire libre. Cada vehículo estaba sujeto a diferentes condiciones de iluminación, y se realizó la captura de imágenes a diferentes distancias y perspectivas, ya que el algortimo utilizado depende de la perspectiva y ángulo de inclinación de la matrícula, por lo que el conjunto de muestras, mientras más diverso sea, mejor será para el proceso de entrenamiento.

Se logró la adquisición de un total de 156 fotos. Con este conjunto de imágenes, se procedió crear el conjunto de muestras positivas y negativas para el entrenamiento de la clasificadora.

De las diferentes clasificadoras que están definidas en la librería OpenCV, se utilizó la clasificadora en cascada por las siguientes razones:

- Dado que la clasificadora está basada en árboles de desición, requiere de muy pocos recursos computacionales.
- Se puede realizar el proceso de entrenamiento a través de línea de comandos, por lo que el proceso es menos tedioso.
- Algoritmos descriptores disponibles para el entrenamiento: HAAR, HOG Y LBP.

Para realizar el proceso de entrenamiento, se debe crear primero un conjunto de directorios, siguiendo la siguiente estructura:

```
/Clasificadora
        /Etapas
                params.xml
        /Positivas/
        /Negativas/
        Positivas.txt
        Negativas.txt
        CascadeLBP
```

La carpeta *Positivas* contiene imágenes del objeto a detectar, en este caso, imágenes de las matrículas (y sólo las matrículas) extraídas de las diferentes fotos recolectadas. Las muestras positivas se ajustaron a un tamaño fijo de 100x50 pixeles, en escala de grises, nótese que el ajuste sigue conservando la relación entre ancho y alto del objeto.

La carpeta *Negativas* contiene imágenes de escenas donde no se encuentra el objeto de interés (Figura 29). Las fotos originales se editaron para cubrir el área donde se encontraba la matrícula, de esa manera, se obtiene una imagen de la que sólo se pueden extraer muestras negativas, como por ejemplo, los faros del vehículo, edificaciones, aceras, vegetación, personas, entre otros.

Figura 29. Ejemplos de muestras negativas.

En el archivo *Positivas.txt* se encuentra la ubicación de las imágenes positivas, siguiendo el siguiente formato:

Dirección 1 0 0 100 50

Donde *Dirección* es la dirección de la imagen; el siguiente número representa la cantidad de muestras positivas en la imagen, que en este caso, sólo hay una muestra positiva; y posterior al número de muestras, las coordenadas y dimensiones de cada muestra. Dado que la imagen es de la matrícula completa, el origen de la muestra es el origen de la imagen (0,0) y su dimensión es la de la imagen, que para este caso, se ajustó al tamaño de 100x50 pixeles.

En el archivo *Negativas.txt* se encuentra la dirección de las imágenes negativas, es decir, las imágenes donde no hay presencia de matrículas.

Una vez organizado este directorio, se utilizó la línea de comandos para realizar el entrenamiento de la clasificadora. OpenCV cuenta con 2 líneas de comando para realizar el entrenamiento:

1. *opencv_createsamples*: Con este comando se crea un archivo binario a partir del conjunto de imágenes positivas, el cual será utilizado para el entrenamiento de la clasificacora. Para crear el archivo binario, se deben definir los siguientes parámetros:

- -*vec*: Nombre del archivo binario (*CascadeLBP*).
- -*info*: Nombre del archivo que contiene las direcciones de las muestras positivas (*Positivas.txt*).
- -*maxdev*: Máxima desviación (Valor utilizado: 30).
- -*bg*: Nombre del archivo que contiene las direcciones de las imagenes de donde se extraerán las muestras negativas (*Negativas.txt*).
- -*num*: Número de muestras positivas (169 muestras).
- -*w*: Ancho de cada imagen positiva (100 pixeles).
- -*h*: Alto de cada imagen positiva (50 pixeles).
- -*maxxangle*: Máximo ángulo de inclinación de la muestra en la dirección “x”, dado en radianes (Valor utilizado: 0,5).
- -*maxyangle*: Máximo ángulo de inclinación de la muestra en la dirección “y”, dado en radianes (Valor utilizado: 0,5).
- -*maxzangle*: Máximo ángulo de inclinación de la muestra en la dirección “z”, dado en radianes (Valor utilizado: 0,5).

Con los parámetros antes definidos, la línea de comandos utilizada fue: *opencv_createsamples -vec CascadeLBP -info Positivas.txt -bg Negativas.txt -num 169 -maxidev 30 -maxxangle 0.5 -maxyangle 0.5 -maxzangle 0.5 -w 100 -h 50*

2. *opencv_traincascade*: Con este comando se realizó el proceso de entrenamiento de la clasificadora. Al igual con el comando *opencv_createsamples*, se definieron los siguientes parámetros:

- -*data*: Nombre del directorio donde se creará el archivo *.xml* con las especificaciones de la clasificadora resultante (*Etapas*).
- -*vec*: Nombre del archivo binario (CascadeLBP).
- -*bg*: Nombre del archivo que contiene las direcciones de las imagenes de donde se extraerán las muestras negativas (*Negativas.txt*).
- -*numPos*: Número de muestras positivas a utilizar en cada etapa de entrenamiento (Valor utilizado: 160 muestras).
- -*numNeg*: Número de muestras negativas a utilizar en cada etapa de entrenamiento (Valor utilizado: 80 muestras).
- -*w*: Ancho de cada imagen positiva (Valor utilizado: 100 pixeles).
- -*h*: Alto de cada imagen positiva (Valor utilizado: 50 pixeles).
- -*featureType*: Algoritmo descriptor a utilizar (Valor utilizado: LBP).
- -*numStages*: Número de etapas de entrenamiennto (Valor utilizado: 25 etapas).
- -*maxFalseAlarmRate*: Máxima tasa de falsas alarmas (Valor utilizado: 0,5).
- -*minHitRate*: Mínima tasa de aciertos (Valor utilizado: 0,999).
- -*precalcValBufSize*: Tamaño del búfer para valores de características precalculadas (Valor utilizado: 2048 Mb).
- -*precalcIdxBufSize*: Tamaño del búfer para índices de características precalculadas (Valor utilizado: 2048 Mb).
- -*acceptanceRatioBreakValue*: Valor de relación de aceptación para detener

sesión de entrenamiento (Valor utilizado: 0,0001).

- *-weightTrimRate*: Tasa de ajuste de peso o ponderación (Valor utilizado: 0,95).
- *-maxDepth*: Máxima profundidad de un árbol de desición débil (Valor utilizado: 1).
- *-maxWeakCount*: Máximo número de árboles de desición en cada etapa de clasificación (Valor utilizado: 300).

Con los parámetros antes definidos, la línea de comandos utilizada fue: *opencv_traincascade -data Etapas -vec CascadeLBP -bg Negativas.txt -numPos 160 -numNeg 80 -w 100 -h 50 -featureType LBP -numStages 25 -maxFalseAlarmRate 0.5 -minHitRate 0.999 -precalcValBufSize 2048 -precalcIdxBufSize 2048 -acceptanceRatioBreakValue 0.0001 -weightTrimRate 0.95 -maxDepth 1 -maxWeakCount 300*

Cabe destacar que el proceso de entrenamiento culminará cuando ocurra uno de los siguientes casos:

- Se alcance el número de etapas deseado.
- Se alcance el valor de relación de aceptación deseado.

Para el presente proyecto, se pudieron culminar las 25 etapas del proceso de entrenamiento. Una vez finalizado, se obtiene un archivo *cascade.xml*, el cual contiene todos los parámetros de la clasificadora resultante. Este tipo de archivo se utilizará para inicializar la clasificadora en la rutina de reconocimiento.

A continuación, en la Figura 30 se presenta el diagrama de flujo de la etapa de localización de la matrícula.

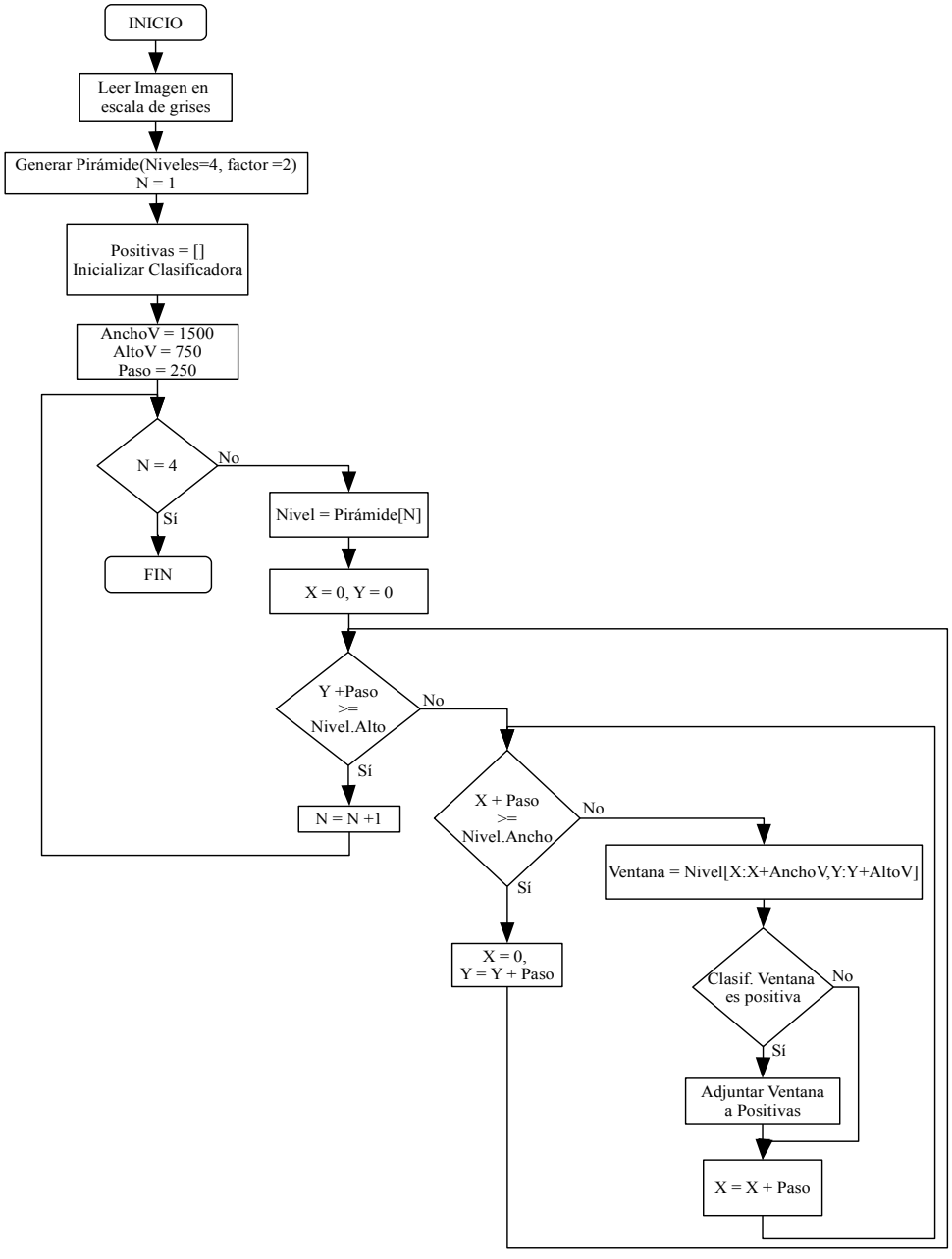

Figura 30. Diagrama de flujo de la rutina de localización.

La clasificadora determinará si en las diferentes ventanas existen candidatos potenciales de matrícula, para ello, se debe definir las dimensiones mínimas y máximas del objeto a detectar dentro de la misma imagen. En el presente proyecto, la función de detección se ajustó para detectar objetos cuya dimensión esté entre 150x75 y 1500x750 pixeles.

3.3.2. Reajuste de tamaño y filtrado de las posibles matrículas.

Dado que la función de detección de la clasificadora detecta objetos con diferentes dimensiones, se ajustó el conjunto de muestras a una dimensión fija, particularmente, a 800x400 pixeles.

Posterior a esto, se realizó un proceso de filtrado en todas la muestras ajustadas. Se utilizó un filtro Gaussiano, con una dimensión de kernel de 15x15 pixeles.

3.3.3. Binarización y erosión de la imagen.

Una vez finalizado el proceso de filtrado, se realiza un proceso de compresión de la imagen para reducir la información a ser procesada por el OCR. Para ello, se realizó primero un proceso de binarización Gaussiana, con un kernel de dimensión 213x213 pixeles. Posterior a esto, se erosionó la imagen para obtener un trazado más fino del número de la matrícula, y para eliminar elementos no deseados, tales como puntos o contornos alrededor de los caracteres, lo cuales pueden interferir en el proceso de clasificación de dichos caracteres. Para el proceso de erosión, se utilizó un kernel de dimensión 5x5 pixeles (ver Figura 31).

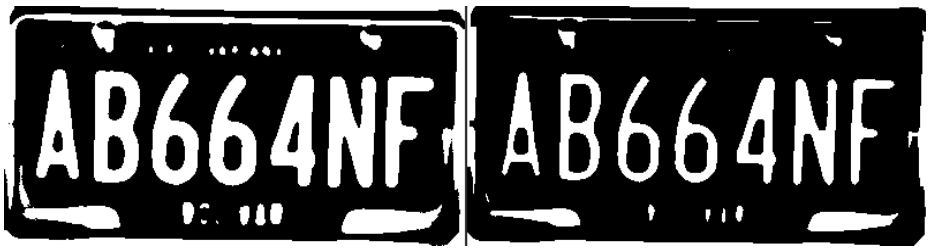

Figura 31. Binarización y Erosión.

3.3.4. Localización de área del número de la matrícula.

Como se puede observar en la Figura 31, el resultado final es una imagen binarizada, la cual contiene una serie de contornos, conformados por los caracteres de la matrícula y otros elementos. OpenCV cuenta con una serie de funciones para la extracción y análisis de contornos. En particular, para extraer cualquier contorno de la figura, se puede utilizar la función cv2.boundingRect(), la cual retorna 2 pares de parámetros:

- Las coordenadas de la esquina superior izquierda del rectángulo *(x,y)* que encierra dicho contorno.
- Las dimensiones de dicho rectángulo (*Ancho,Alto*).

En la Figura 32 se visualizan los diferentes contornos detectados en una matrícula. Al ajustar el tamaño de la imagen a 800x400 pixeles, se puede tener una noción de cuales son las dimensiones de los diferentes rectángulos que encierran el contorno de los caracteres (Figura 33). Para ese tamaño, los rectángulos que encierran los caracteres de la matrícula cumplen con las siguientes condiciones:

- Alto mayor o igual a 180 pixeles.
- Ancho mayor o igual a 55 pixeles.
- Área del rectángulo menor o igual a 18000 pixeles.

- Relación entre ancho y alto del rectángulo menor o igual a 0,57.

Figura 32. Contornos.

Figura 33. Contornos de interés.

Para facilitar el manejo del conjunto de contornos, se planteó la definición de la siguiente clase:

Tabla 8. Definición de la clase.

Nombre de la clase: *Letra*	
Variables	Descripción
x	Coordenada horizontal del contorno
y	Coordenada vertical del contorno
width	Ancho del contorno
height	Alto del contorno
image	Imagen contenida en el rectángulo
area	Área del rectángulo
aspectrate	Relación entre ancho y alto

Una vez obtenidos los diferentes contornos de los caracteres que conforman el número de la matrícula, se procedió a extraer dicha área. Para ello, se diseñó una rutina que realizara una comparación entre las coordenadas *x* de los diferentes contornos, y retornara la menor y mayor coordenada, las cuales corresponden con la ubicación del primer y último caracter de la matrícula, respectivamente. En la Figura 34, se muestran los diagramas de flujo del proceso de búsqueda de dichas coordenadas. A la izquierda, se muestra el proceso de localización de la menor coordenada, mientras que a la derecha se muestra la de la mayor coordenada.

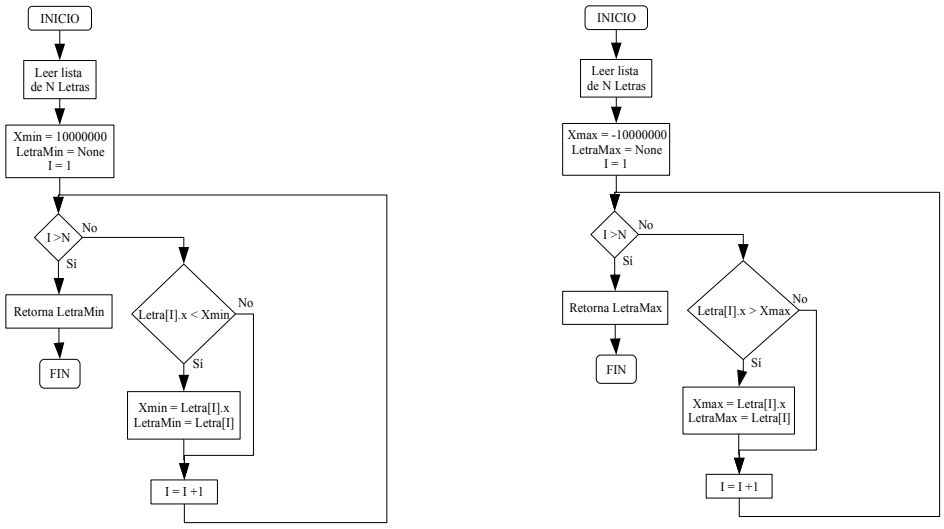

Figura 34. Algoritmos de localización del número de matrícula.

Con las coordenadas resultantes, se realiza un proceso de recorte de la imagen. Se obtiene un área rectangular, cuyo origen corresponde con las coordenadas del primer caracter (*Xmin,Ymin*), y sus dimensiones son determinadas por las coordenadas, ancho y alto del último caracter. Se puede observar el resultado del proceso de recorte en la Figura 35.

Figura 35. Área del número de la matrícula.

3.3.5. Reconocimiento de caracteres.

Una vez localizada el área donde se encuentra el número de la matrícula, se puede utilizar el software de reconocimiento de caracteres u OCR. Tesseract-OCR posee diferentes parámetros que deben ser configurados antes de realizar de lectura y clasificación. Se configuró el OCR para que reconociera texto en inglés (por defecto), ya que el caracter "Ñ" no es válido para la matrícula, y se configuró el método de segmentación de la página ó *psm*. A continuación, se presenta los posibles valores del parámetro *psm*.

Tabla 9. Valores del parámetro psm.

Valor	Descripción
0	Sólo OSD
1	Segmentación de página automática con OSD
2	Segmentación de página automática sin OSD ni OCR
3	Segmentación de página automática sin OSD
4	Asume una sola columna de texto de tamaño variable
5	Asume un solo bloque uniforme de texto alineado verticalmente.
6	Asume un solo bloque uniforme de texto
7	Trata la imagen como una linea de texto

8	Trata la imagen como una palabra
9	Trata la imagen como una palabra en un círculo
10	Trata la imagen como un solo caracter

Se configuró el valor del parámetro *psm* a 8, esto es asumir que la imagen a procesar es una palabra. Una vez finalizada la configuración del OCR, se procedió a analizar el área del número de la matrícula. Se muestra el uso del OCR en la Figura 36.

Figura 36. Utilización del Tesseract-OCR.

Como se puede observar en la Figura 36, al analizar el área obtenida en la etapa anterior, se reporta el texto detectado por la herramienta OCR mediante un mensaje en consola (texto en azul), que en este caso, coincide con el texto presente en la imagen.

CAPÍTULO IV

PRUEBAS DE VALIDACIÓN Y RESULTADOS

Una vez diseñada la rutina de reconocimiento, se realizaron diferentes pruebas para evaluar el desempeño del sistema, como también para definir las condiciones ópitmas bajo las cuales es posible lograr la detección de la matrícula. Se realizaron diversas pruebas para:

1. Evaluar el desempeño de la rutina de localización.
    - Determinar la posibilidad de detectar múltiples matrículas en una imagen.
    - Determinar la posibilidad de detectar la matrícula de un vehículo en movimiento.
    - Determinar distancia máxima de reconocimiento.
2. Evaluar el desempeño del OCR en función de los caracteres reconocidos.
    - Determinar desempeño bajo diversas condiciones de posición, estado, entre otras.

Cabe destacar que las diferentes pruebas se realizaron en imágenes con una resolución de 13MP. Como se mencionó con anterioridad, se deshabilitaron todas las opciones de flash, zoom óptico y mejoramiento de la imagen de la cámara.

4.1. Rutina de localización

4.1.1. Desempeño

Para evaluar el desempeño de la rutina de localización, se realizaron diversas pruebas, las cuales consistían en analizar diferentes imágenes de vehículos. En cada análisis, se obtenía una serie de muestras positivas, y en cada uno de estos conjuntos, se determinaba que porcentaje de esas muestras correspondía con el área de la matrícula. Al analizar cada imagen, se observó que la rutina detectaba múltiples muestras positivas, en algunos casos, se lograba la detección del área de la matrícula; sin embargo, en otras ocasiones detectaba secciones de la misma, e incluso, no se lograba la detección. A continuación, se presenta los resultados obtenidos:

Tabla 10. Resultados rutina de localización.

Número de imágenes analizadas	Número de muestras positivas	Número de secciones de matrícula obtenidas	Número de áreas de matrícula obtenidas	Matrículas localizadas con éxito
80	682	299 (43,84%)	84 (12,32%)	47 (58,75%)

Como se puede observar, cerca del 13% de las muestras obtenidas coresponden con el área donde se encuentra el número de la matrícula (Figura 38), mientras que un número cercano al 44% de las muestras son secciones de la matrícula (Figura 37), es decir, esquinas, bordes, áreas incompletas, entre otras. Se logró la localización de la matrícula en 47 imágenes, con un porcentaje de éxito cercano al 58,75%.

Figura 37. Secciones de matrícula.

Figura 38. Áreas de matrícula.

De acuerdo a los resultados obtenidos, cerca del 57% de las muestras detectadas están relacionadas con el área de la matrícula. Sin embargo, dado que la mayor parte de estas muestras corresponden con secciones de la misma, tales como bordes, esquinas, entre otras, no es posible obtener el número de dicha matrícula. En base a estos resultados, es evidente que la clasificadora requiere de más entrenamiento para el reconocimiento de este objeto. Los sistemas de reconocimiento visual, dependiendo de las características del objeto a detectar, por lo general requieren de miles de muestras para su entrenamiento.

Por otra parte, el sistema logró reconocer la matrícula en múltiples condiciones poco favorables, por ejemplo, se reconoció una matrícula que se encontraba en pésimas condiciones (Figra 40); y se logró el reconocimiento de una matrícula que se encontraba en una zona con vegetación (Figura 39). También se observó que el sistema logró reconocer la matrícula con un cierto ángulo de

inclinación (Figuras 41 y 42), ya sea causada por el posicionamiento de la matrícula en el vehículo, o causada por la perspectiva de captura de la imagen.

Figura 39. Matrícula con vegetación.

Figura 40. Matrícula dañada.

Figura 41. Matrícula con ángulo de visualización 1.

Figura 42. Matrícula con ángulo de visualización 2.

Adicionalmente, se determinó que el sistema es capaz de reconocer mútliples matrículas en una imagen, sin embargo, requiere de una mayor cantidad de muestras de entrenamiento, ya que reconoció secciones de las matrículas que no pueden ser utilizadas para determinar sus respectivos números. Se puede visualizar el resultado en las Figuras 43 y 44.

Figura 43. Dos vehículos en una imagen.

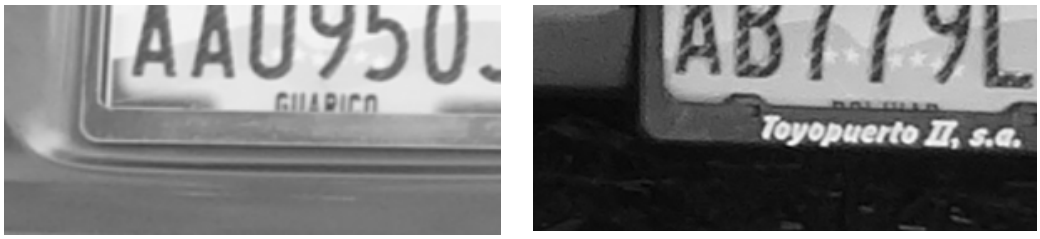

Figura 44. Matrículas reconocidas.

4.1.2. Vehículos en movimiento

Para determinar si la rutina puede reconcer matrículas de vehículos en movimiento, se realizó un proceso de adquisición de muestras, el cual consistía en tomar fotografías desde un vehículo particular en movimiento. Las imágenes fueron capturadas en una zona residencial, en la cual el vehículo transitaba a una velocidad variable entre los 40 y 60 Km/h. Una de las muestras obtenidas se puede observar en

la Figura 45.

Figura 45. Vehículo en movimiento.

Al analizar las diferentes imágenes obtenidas, se observó que el sistema mostró un desempeño similar al caso de los vehículos estáticos, es decir, en diversas ocasiones logró localizar la matrícula, sin embargo, detectó secciones de las cuales no se pudo obtener dicho número. Al analizar la imagen mostrada en la Figura 46, se obtuvieron los resultados ilustrados en la Figura 47.

Figura 46. Imagen analizada.

Figura 47. Secciones obtenidas.

A pesar de esto, se presentó el caso en que la rutina logró localizar correctamente el área de la matrícula. En este caso, se observó que los movimientos del vehículo junto con la interposición de la ventana frontal entre la cámara y el objetivo, introducen ruido adicional, más concretamente, difunan la imagen, y con esto, se cambia el dimensionamiento de los contornos detectados, por lo que interfiere en el proceso de recorte del área del número de la matrícula. Dicha prueba se visualiza en las Figuras 48 y 49.

Figura 48. Efectos negativos en la imagen.

En la siguiente imagen se presenta la matrícula detectada. Como se puede oberservar, por las causas antes mencionadas, se desmejora notablemente la calidad de la imagen, por lo que dificulta el proceso de obtención del número de la matrícula.

Figura 49. Matrícula detectada.

4.1.3. Distancia máxima de reconocimiento

Con respecto a la distancia máxima de reconocimiento, se utilizaron múltiples imágenes de un vehículo a diferentes distancias, las cuales estaban comprendidas entre 1 metro y 10 metros, en intervalos de 1 metro. Analizando este conjunto de imágenes, se determinó que la rutina puede localizar la matrícula de un vehículo ubicado a una distancia menor o igual a 7 metros, sin embargo, requiere de más entrenamiento para detectar la matrícula en su totalidad.

Cabe destacar que la distancia de reconocimiento está relacionada con la resolución de la cámara utilizada. Si se utiliza una cámara con la cual se pudiera obtener imágenes de mayor resolución, es posible aumentar la distancia máxima de reconocimiento, ya que se almacena una mayor cantidad de información, es decir, una mayor cantidad de pixeles conforman el área de la matrícula. En la Figura 50, se muestra el resultado obtenido a 7 metros del vehículo.

Figura 50. Matrícula detectada a 7 metros.

4.2. Desempeño OCR

Para evaluar el desempeño del Tesseract-OCR, se utilizó la rutina de localización del área del número de la matrícula (etapa final de la rutina de reconocimiento), aplicada a imágenes de matrículas, las cuales fueron recortadas del conjunto original de imágenes. A continuación, se presentan los resultados:

Tabla 11. Resultados OCR.

Número de Matrículas Procesadas	Promedio de caracteres reconocidos	Porcentaje promedio de caracteres reconocidos	Matrículas Reconocidas
107	4,81 (4-5 caracteres)	68,73%	22 (20,56%)

Durante el proceso de evaluación, se observó que el Tesseract, en múltiples ocasiones, logró reconocer todos los caracteres de la matrícula, sin embargo, en algunas situaciones no logró reconocer la matrícula en su totalidad, o realizaba detecciones erróneas. Algunas detecciones erróneas incluyen:

- Confusión entre letras, tales como “M” y “H” (Figura 51).

Figura 51. Detección errónea: Letra/Letra.

- Principalmente, confusión entre número y letra, tales como “0” y “U”, “1” e “I”, “2” y “Z”, “4” y “A”, “5” y “S”, “6” y “G”, “7” y “Y”, “8” y “B” y, finalmente, “9” y “Q”. También se presentó el caso contrario, reconocimiento de letras como números (Figura 52).

Figura 52. Detección errónea: Número/Letra.

Como se puede observar, el cero (0) presenta una imperfección en la parte superior, la cual se utiliza para hacer la distinción entre dicho número y el caracter "O".

- Confusión entre pares de números y letras, como por ejemplo, "14" y "M", "11" y "H".

Para corregir las detecciones erróneas entre letra y número, se diseñó una rutina que verifica si el serial de dicha matrícula cumple con el formato establecido (7 caracteres de la forma 2 letras, 3 números y 2 letras), y en caso contrario, aplica una corrección dependiendo del caracter detectado. Se obtuvieron los siguientes resultados:

Tabla 12. Resultados OCR con corrección.

Número de Matrículas Procesadas	Promedio de caracteres reconocidos	Porcentaje promedio de caracteres reconocidos	Matrículas Reconocidas
107	5,64 (5-6 caracteres)	80,58%	55 (51,4%)

Cabe destacar que en los casos donde ocurrió la confusión entre pares de números y letras, no se aplicó la rutina de corrección, ya que la cadena de caracteres detectada no contenían 7 elementos. La rutina no realiza correcciones entre letras (por ejemplo, confusión entre "M" y "H") .

Aunque el diseño de la rutina de corrección pudiera solventar el problema de las detecciones erróneas, se debe realizar un estudio exhaustivo de los diferentes casos que se pudieran presentar, lo cual no representa una solución eficiente. Por lo general, la detección errónea de caracteres por parte del OCR es un indicador de que el sistema requiere de un entrenamiento adicional para el reconocimiento de estos caracteres. En otras palabras, para mejorar el desempeño de la detección del OCR, se recomienda realizar un proceso de entrenamiento con una base de datos de los

posibles caracteres que se pueden encontrar en una matrícula.

Durante la etapa de pruebas, se observó que la rutina del OCR no arrojó ningún resultado en los casos donde la matrícula presentaba un cierto ángulo de inclinación. Esto se debe a que las dimensiones de los recuadros que encierran los contornos cambian, y por ende, interfiere con el proceso de recorte del área del número de la matrícula (ver Figuras 53 y 54).

Figura 53. Ejemplo matrícula inclinada 1.

Figura 54. Ejemplo de matrícula inclinada 2.

También se observó que el OCR no pudo reconocer una matrícula en mal estado (golpeada, desgastada). Se muestra dicha matrícula en la Figura 55.

Figura 55. Matrícula desgastada.

Cabe destacar que no debe existitr ningún tipo de obstáculo entre la cámara y la matrícula del vehículo (Figura 56). Al analizar la matrícula obtenida de la Figura 39, se observó que el OCR no arrojó un resultado, esto debido a la interposición de una rama, la cual modifica los contornos detectados e interfiere con el proceso de recorte y detección de texto.

Figura 56. Matrícula con vegetación.

CONCLUSIONES Y RECOMENDACIONES

Con este Trabajo Especial de Grado, se busca fomentar la utilización de herramientas open-source, tales como la librería de visión artificial OpenCV, y lenguajes de programación de alto nivel como Python. Gracias a las herramientas antes mencionadas, en conjunto con la plataforma Raspberry Pi, se logró el diseño de un prototipo de un sistema de reconocimiento de matrículas vehiculares.

Con respecto a la rutina de localización de la matrícula, ésta mostró un pocentaje de éxito cercano al 50%. Este desempeño era de esperarse, pues se utilizó una cantidad reducida de muestras positivas y negativas en el entrenamiento de la clasificadora. Las muestras detectadas por dicha rutina correspondían con secciones de la matrícula, de las cuales no se podía obtener el número de la misma. A pesar de este inconveniente, en múltiples ocasiones, se logró la detección del área completa de la matrícula, y posteriormente, se obtuvo dicho número. Se determinó que el uso de la rutina está mejor condicionado a el reconocimiento de matrículas en vehículos estáticos, ya que el movimiento del objeto disminuye la calidad de la imagen.

Se observó que el sistema logró reconocer la matrícula a una distancia máxima de 7 metros. Esto está relacionado con la resolución de la imagen digital analizada. En otras palabras, si se logra la captura de la imagen con una cámara de mayor resolución, es posible aumentar dicha distancia de reconocimiento.

Los programas de reconocimiento de caracteres como el Tesseract-OCR son un elemento crucial en este tipo de aplicaciones. Se observó que, en los casos donde se pudo detectar la matrícula completa, se logró el reconocimiento de más del 50% de

los caracteres; sin embargo, presentó problemas en la detección de números, los cuales fueron corregidos utilizando una rutina auxiliar, pero esto no presentó ser una solución definitiva. Es recomendable entrenar el OCR para el reconocimiento de estos carcateres, ya que en algunos casos reconoció múltiples letras o números como un solo caracter.

El módulo Raspberry Pi presentó ser una herramienta útil en el desarrollo de un prototipo en un lapso de tiempo corto, con un costo reducido. A pesar de esto, dicho sistema no está condicionado para uso en exteriores, y la cámara utilizada en conjunto con dicho módulo es de baja resolución, por lo que disminuye la máxima distancia de reconocimiento. Sin embargo, dicho prototipo cumple con la arquitectura básica del módulo a desarrollar.

El sistema de reconocimiento solo puede obtener el número de dicha matrícula si están presentes las siguientes condiciones:

- Ángulo de inclinación de la matrícula en cualquier dirección entre a 0-10°.
- Matrícula en buenas condiciones.
- No deben existir obstáculos entre la matrícula y la cámara.
- Objetivo de captura estático.
- Condiciones de iluminación favorables.
- Distancia de la cámara a la matrícula menor o igual a 7 metros.

Algunas recomendaciones para mejorar el desempeño del sistema pueden ser:

- Definir las condiciones bajo las cuales se realiza el proceso de detección para seleccionar los módulos que se adapten mejor a dicha situación. Se dice esto ya que, en el presente proyecto, la cámara que conforma el módulo fue de baja resolución, por lo que la distancia de reconocimiento se reduce drásticamente. Adicional a esto, no se realizó un estudio referente a los condiciones bajos las cuales se encontrará dicho módulo, por lo que no puede ser utilizado en exteriores, y tampoco puede ser utilizado en vehículos en movimiento.
- Dependiendo de las condiciones, utilizar rutinas de calibración para compensar efectos de distorsión introducidos por la lente de la cámara o por la perspectiva de captura.
- Obtener una mayor cantidad de imágenes de vehículos o matrículas para el entrenamiento de la clasificadora.
- Evaluar los algoritmos de las diferentes clasificadoras presentes en OpenCV, con el fin de elegir la más adecuada para la detección del objeto.
- Implementar el proyecto en un computador con una mayor capacidad de procesamiento.
- Conocer la posición esperada del objeto a detectar, con el fin de disminuir tiempo de procesamiento.
- Para disminuir tiempo de ejecución, se recomienda procesar todas las imágenes que conforman la pirámide de imágenes en paralelo.
- Rediseñar la rutina que recorta el área donde se encuentra el número de la matrícula, ya que pudiera no adecuarse a todas las situaciones.

- Analizar el conjunto de muestras positivas obtenidas en paralelo, con el fin de disminuir tiempo de ejecución.
- Realizar un entrenamiento adicional con el Tesseract para el reconocimiento de números y letras presentes en las matrículas venezolanas.

REFERENCIAS BIBLIOGRÁFICAS

[1] Petrou M. y Petrou C. *Image Processing: The Fundamentals*, (Libro).--Reino Unido: John Wiley & Sons. Segunda Edición, 2010, cap. 1-6.

[2] González, R.C., Wintz, P. *Procesamiento digital de imágenes*, (Libro). Addison-Wesley, 1996,Tema 3-4, pp. 89-269.

[3] S. Nedevschi, T. Mariţa, R. Dănescu, F. Oniga, R. Brehar, I. Giosan, S. Bota, A. Ciurte y A. Vatavu. *Image Processing – Laboratory* Guide, (Libro). UTPress. 2016, Cap. 9.

[4] González, Y. *Tema 5 - Filtrado de Imágenes.* (Presentación PowerPoint). [En línea].
<http://dmi.uib.es/~ygonzalez/VI/Material_del_Curso/Teoria/Tema5_Filtrado.pdf.>
[Consulta: Enero, 2016.]

[5] de La Fuente, E. y Trespaderne, F. *Visión Artificial Industrial. Procesamiento de Imágenes para Inspección Industrial y Robótica*, (Libro). --Valladolid: España: Secretariado de Publicaciones e Intercambio Editorial, Universidad de Valladolid, 2012, Cap. 4.

[6] *Image Thresholding Tutorial - OpenCV-Python Tutorials.* [En línea].
<http://docs.opencv.org/trunk/d7/d4d/tutorial_py_thresholding.html#gsc.tab=0>
[Consulta: Enero 2016]

[7] *Morphological Transformations - OpenCV-Python Tutorials.* [En línea]. <http://docs.opencv.org/3.0-beta/doc/py_tutorials/py_imgproc/py_morphological_ops/py_morphological_ops.html> [Consulta: Enero 2016]

[8] Timo O., Matti P. y Topi M. *Multiresolution Gray Scale and Rotation Invariant Texture Classification with Local Binary Patterns.* (Tesis).--Oulu: Finlandia: University of Oulu, Infotech Oulu, Machine Vision and Media Processing Unit, 2002.

[9] Rosebrock A. *Sliding Windows for Object Detection with Python and OpenCV*. Marzo 23, 2015. [En línea]. <http://www.pyimagesearch.com/2015/03/23/sliding-windows-for-object-detection-with-python-and-opencv/>

[10] *Image Pyramids – OpenCV-Python Tutorials.* [En línea]. <http://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_pyramids/py_pyramids.html>. [Consulta: Marzo 2015]

[11] Viola P. y Jones M. *Robust Real-Time Face Detection.* International Journal of Computer Vision, 57(2), (2004), 137–154 p.

[12] Rojas R. *AdaBoost and the Super Bowl of Classifiers. A Tutorial Introduction to Adaptive Boosting.* (Tesis).--Berlín, Alemania: Freie Universität Berlin, Computer Science Department, 2009.

[13] Reglamento de la Ley de Tránsito y Transporte Terrestre (1998, 26 de junio). Gaceta Oficial de la República Bolivariana de Venezuela, N° 38.944. Junio 3, 2008.

[14] *OpenCV Oficial Site*. [En línea]. <http://opencv.org/>. [Consulta: Julio 2016].

[15] *Python Software Foundation*. [En línea]. <https://www.python.org/psf/>. [Consulta: Mayo 2016]

[16] K. M. Sajjad. *Automatic License Plate Recognition using Python and OpenCV.* (Tesis).--Kerla, India: M.E.S. College of Engineering, Department of Computer Science and Engineering. Julio 2, 2014.

[17] L. Xin. *Vehicle License Plate Detection and Recognition*. (Tesis).--Missouri, Estados Unidos: University of Missouri, Faculty of the Graduate School. Diciembre 2010.

[18] Soler J. *Detección de Matrículas mediante Características Binarias Locales (LBP)*. (Tesis).--Murcia, España: Universidad de Murcia, Facultad de Informática, Departamento de Informática y Sistemas. Junio 2009.

[19] RaspberryPi Foundation Oficial Site. [En línea]. <https://www.raspberrypi.org/>. [Consulta: Junio 2016]