ESCUELA SUPERIOR POLITECNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

“Aplicación de visión por computador para el reconocimiento del número de placa de vehículos usando modelos de aprendizaje (OCR convencionales)”

INFORME DE MATERIA DE GRADUACION

Previo a la Obtención del Título de:

INGENIERO EN CIENCIAS COMPUTACIONALES

ESPECIALIZACION EN SISTEMAS DE INFORMACION

Presentada por:

María Fernanda Frydson Talledo

Richard Armando Gutiérrez Pilay

GUAYAQUIL - ECUADOR

Año: 2011

AGRADECIMIENTO

Agradezco a Dios y a mi madre quienes han sido mi fuerza y sostén, a la ESPOL, a todos mis profesores, mis amigos y compañeros y a todas las personas que de algún modo colaboraron a la realización y culminación de este trabajo.

Ma. Fernanda Frydson

AGRADECIMIENTO

Agradezco a Dios mi magnifico instructor, que brinda el aguante y las fuerzas para alcanzar mis metas.

Muy agradecido también a mis padres por brindar su apoyo incondicional y soportar los momentos más difíciles a su lado; así como mis amigos que resultaron ser hermanos y brindar alegrías en mi vida.

Richard Gutiérrez Pilay

DEDICATORIA

Quiero dedicar este trabajo principalmente a Dios, quien me ha dado la templanza y fortaleza para poder culminar este trabajo.

A mi madre, que con su amor, comprensión y su apoyo constante, supo enseñarme desde pequeña a luchar para alcanzar mis metas, ha sido y será el motor para seguir adelante; mi triunfo es suyo.

A mi hija Milena, quien ella es sin duda alguna lo mejor que nunca me ha pasado, es mi referencia para el presente y el futuro.

Ma. Fernanda Frydson

DEDICATORIA

Dedicado a mí Dios debido a que sin su amorosa guía no cumpliría mis metas, a mis padres por dar apoyo emocional y económico, así como a cada instructor y amigo que supo darme un valioso consejo.

Richard Gutiérrez Pilay

TRIBUNAL DE SUSTENTACION

_____	_____
Phd. Boris Vintimilla Burgos	**Phd. Sixto García Aguilar**
PROFESOR DE LA MATERIA DE GRADUACION	**PROFESOR DELEGADO DEL DECANO**

DECLARACION EXPRESA

“La responsabilidad del contenido de este Trabajo de Grado, me corresponden exclusivamente; y el patrimonio intelectual de la misma a la ESCUELA SUPERIOR POLITECNICA DEL LITORAL”

(Reglamento de Graduación de la ESPOL)

_____ _____

Fernanda Frydson Talledo **Richard Gutiérrez Pilay**

RESUMEN

En los últimos años la digitalización de la información (textos, imágenes, sonido, etc) ha devenido un punto de interés para la sociedad. En el caso concreto de los textos, existen y se generan continuamente grandes cantidades de información escrita, tipográfica o manuscrita en todo tipo de soportes. En este contexto, poder automatizar la introducción de caracteres evitando la entrada por teclado, implica un importante ahorro de recursos humanos y un aumento de la productividad, al mismo tiempo que se mantiene, o hasta se mejora, la calidad de muchos servicios.

El reconocimiento automático de matrículas (Automatic number plate recognition o ANPR) es una aplicación de la visión por computador que utiliza reconocimiento óptico de caracteres en imágenes para leer las matrículas de los vehículos. A diferencia de otras aplicaciones de visión por computador del entorno industrial, esta aplicación cuenta con la dificultad añadida de tener que trabajar con un entorno NO CONTROLADO, como lo es la iluminación que varía a cada hora del día, la inclinación y el estado de conservación de la placa, entre otras; por lo que a menudo se tiende a utilizar iluminación infrarroja para hacer posible que la cámara fotográfica capture tomas en cualquier momento del día y otras técnicas que posibiliten el reconocimiento de la matrícula vehicular, entre ellos el reconocimiento óptico de caracteres.

El reconocimiento óptico de caracteres (Optic Character Recognition OCR) realiza una simulación de la habilidad humana para el reconocimiento automático de patrones entre los diferentes caracteres alfanuméricos existentes mediante la creación y uso de modelos físicos y matemáticos.

El presente trabajo ha sido desarrollado como un proyecto de graduación, que comprende el análisis y desarrollo de una solución para el reconocimiento óptico de caracteres a partir de una imagen que previamente ha sido sometida a un proceso de binarización, segmentación, entre otras técnicas; con la finalidad de que pueda contribuir al reconocimiento automático del número de placa en un vehículo.

El proyecto ha sido dividido en dos etapas que son: la etapa de aprendizaje supervisado o entrenamiento, que consiste en la extracción de características de un carácter para que sean almacenadas en una librería de modelos, con la que finalmente, en la etapa de reconocimiento, se realice una verificación de los caracteres coincidentes siguiendo algún modelo matemático o estadístico y dependiendo del porcentaje de coincidencias se escogerá el carácter resultante del análisis.

INDICE GENERAL

Pág.

RESUMEN ..... **I**

INDICE GENERAL **3**

INDICE DE TABLAS ..... **VII**

INDICE DE FIGURAS ..... **VII**

INTRODUCCION ..... **IX**

CAPITULO 1

Revisión Bibliográfica

1.1 Resumen **1**

1.2 Introducción **2**

1.3 Técnicas OCR's **5**

1.4 Análisis de los OCR's **9**

1.5 Conclusiones **17**

CAPITULO 2

Placas Vehiculares en el Ecuador

2.1 Resumen **18**

2.2 Introducción **18**

2.3 Características generales **21**

CAPITULO 3

Tesseract

3.1 Resumen **23**

3.2 Introducción **23**

3.3 Entrenamiento paso a paso **24**

3.3.1 Generar librerias de entrenamiento **26**

CAPITULO 4

Entrenamiento y corrección de errores

4.1 Resumen **33**

4.2 Prueba inicial **34**

4.3 Resultados de las pruebas **35**

4.4 Conclusiones de los resultados obtenidos **51**

4.5 Características y clasificación de las placas **52**

4.5.1 Características de las imágenes **53**

4.5.2 Clasificación de las placas por su condición **54**

4.6 Detección de errores críticos **57**

4.6.1 Error en el análisis de las página **57**

4.6.2 Distinción entre carateres alfabéticos y numéricos **60**

4.7 Análisis y corrección de errores de caracteres **70**

4.7.1 Similitud de caracteres **70**

4.7.2 Patrones para caracteres “Especiales” **78**

4.7.3 Caracteres sesgados **80**

4.8 Conclusiones **86**

CAPITULO 5

Características del Tesseract

5.1 Resumen **88**

5.2 Determinación de la inclinación del carácter **89**

5.3 Determinación del ángulo de rotación vertical y horizontal de la placa **92**

5.4 Tesseract vs otros ANPRS **99**

CAPITULO 6

Pruebas de campo

6.1 Resumen **105**

6.2 Prueba de campo **105**

6.3 Resultado **107**

CONCLUSIONES 109

RECOMENDACIONES..... 110

REFERENCIAS BIBLIOGRAFICAS 111

INDICE DE TABLAS

Tabla 1.1 Ventajas y desventajas de OCRS's 8

Tabla 1.2 Comparacion de resultados para imagen 1 de prueba 16

Tabla 1.3 Comparación de resultados para imagen 2 de prueba 16

Tabla 2.1 Nomenclaturas para placas vehiculares en el Ecuador 20

Tabla 3.1 Herramientas visuales para edición de archivo BOX 28

Tabla 4.1 Resultados por cada placa utilizando lenguaje por default y lenguaje entrenado 51

Tabla 4.2 Resultados obtenidos con y sin variables de configuración 69

Tabla 4.3 Cuadro comparativo de tasa de reconocimiento 69

Tabla 5.1 Parámetros según condiciones ideales 92

Tabla 5.2 Pruebas de rotación en las placas sobre el eje vertical 96

Tabla 5.3 Pruebas de rotación en las placas sobre el eje horizontal .. 98

Tabla 5.4 Cuadro comparativo de OCR's comeriales 104

Tabla 6.1 Datos iniciales para las pruebas de campo 105

Tabla 6.2 Cuadro comparativo de pruebas de campo 108

INDICE DE FIGURAS

Figura 1.1 Estructura básica de una red neuronal 3

Figura 1.2 Imágenes fuente (de izquierda a derecha DRIVER-GOTHIC - LICENSE-PLATE) 9

Figura 1.3 Prueba con Imagen 1 para GOCR 10

Figura 1.4 Prueba con Imagen 2 para GOCR 11

Figura 1.5 Prueba con Imagen 1 para Ocrad 12

Figura 1.6 Prueba con Imagen 2 para Ocrad 12

Figura 1.7 Prueba con Imagen 1 para Tesseract con lenguaje default 13

Figura 1.8 Prueba con Imagen 1 para Tesseract con lenguaje personalizado 14

Figura 1.9 Prueba con Imagen 2 para Tesseract 15

Figura 1.10 Prueba con Imagen 2 para Tesseract con lenguaje personalizado 15

Figura 2.1 Placa Particular 20

Figura 2.2 Placa de alquiler 20

Figura 2.3 Placa gubernamental 21

Figura 2.4 Placa municipal 21

Figura 4.1 Imágenes plantilla 34

Figura 4.2 Cuadro comparativo de pruebas en Tesseract por default vs Tesseract entrenado 52

Figura 4.3 Secuencia de una placa vehicular 54

Figura 4.4 Ejemplos de placas con clasificación “Buena” ……..... 55

Figura 4.5 Ejemplos de placas con clasificación “Parcialmente Buena” 56

Figura 4.6 Ejemplos de placas con clasificación “Mala” 57

Figura 4.7 Imagen del primer carácter..... 57

Figura 4.8 Detección de errores en pruebas 58

Figura 4.9 Corrección en código fuente 59

Figura 4.10 Prueba para el reconocimiento del primer carácter 60

Figura 4.11 Corrección en código fuente utilizando variables de configuración 63

Figura 4.12 Prueba sobre la corrección utilizando variables de configuración 63

Figura 4.13 Plantilla..... 71

Figura 4.14 Plantilla “Q”, “D”, “U”, “O” originales 71

Figura 4.15 Prueba realizadas con los caracteres “Q”, “D”, “U”, “O” 72

Figura 4.16 Entrenamiento para los caracteres “Q”, “D”, “U”, “O” 73

Figura 4.17 Entrenamiento del carácter “D” 73

Figura 4.18 Entrenamiento del carácter “W” 74

Figura 4.19 Entrenamiento del carácter “Q” 74

Figura 4.20 Entrenamiento del carácter “1” 74

Figura 4.21 Entrenamiento del carácter “1”, “7” 75

Figura 4.22 Entrenamiento del carácter “5”, “6” 75

Figura 4.23 Entrenamiento del carácter “8” 75
Figura 4.24 Entrenamiento del carácter “1” 76
Figura 4.25 Entrenamiento del carácter “1” 76
Figura 4.26 Entrenamiento del carácter “8” 76
Figura 4.27 Entrenamiento del carácter “0” 77
Figura 4.28 Entrenamiento del carácter “4” 77
Figura 4.29 Entrenamiento del carácter “1” 77
Figura 4.30 Entrenamiento del carácter “2” 78
Figura 4.31 Entrenamiento del carácter “KYX” 78
Figura 4.32 Entrenamiento de caracteres “SJ”, “HM”, “PRB” 79
Figura 4.33 Entrenamiento de caracteres “NW”, “GB” 79
Figura 4.34 Entrenamiento de caracteres con sesgo a la izquierda 80
Figura 4.35 Entrenamiento del carácter “A” 81
Figura 4.36 Entrenamiento del carácter “w” 81
Figura 4.37 Entrenamiento de caracteres con sesgo a la derecha 81
Figura 4.38 Entrenamiento de caracteres con sesgo a la derecha 82
Figura 4.39 Entrenamiento de caracteres con sesgo a la izquierda 83
Figura 4.40 Entrenamiento de caracteres con sesgo y sin sesgo 84
Figura 4.41 Entrenamiento de caracteres con sesgo a la derecha 84
Figura 4.42 Entrenamiento de caracteres con sesgo a la izquierda 85
Figura 4.43 Entrenamiento de caracteres con sesgo a la derecha con fuente License Plate 86

Figura 4.44 Entrenamiento de caracteres con sesgo a la izquierda con fuente License Plate 86
Figura 5.1 Placa sesgada 89
Figura 5.2 Líneas base sobre toda la placa 89
Figura 5.3 Líneas base sobre el carácter 89
Figura 5.4 Ejemplo de imágenes de entrenamiento con inclinación 90
Figura 5.5 Errores en caracteres sesgados 90
Figura 5.6 Entrenamiento de caracteres sesgados 91
Figura 5.7 Placa de prueba 92
Figura 5.8 Angulo de inclinación de cámara 93
Figura 6.1 Regiones de reconocimiento 106

INTRODUCCION

Resumen

Se presenta una visión general del proyecto, así como la importancia que puede tener su aplicación en la sociedad. Además, se exponen las bases sobre las cuales se justifica el proyecto y los objetivos de su realización.

Al final se realiza un resumen de cómo el presente documento está organizado para alcanzar las metas propuestas.

Descripción del proyecto

Cuando se dispone de información impresa en algún tipo de soporte y se desea procesarla, existen dos opciones: una primera que consistiría en la labor larga y tediosa de introducirla manualmente a través del teclado. Otra posibilidad es automatizar esta operación por medio de un sistema compuesto de hardware y software adecuado que reduciría considerablemente el tiempo de entrada de datos.

El presente proyecto puede contribuir a un sinnúmero de aplicaciones de automatización en el ingreso de información, como lo son: la lectura de etiquetas, procesado de cheques bancarios, lectura de textos, reconocimiento de matrículas, entre otras; siendo este último la base del proyecto.

Cada una de las fases del reconocimiento de matrículas depende del resultado de la anterior, siendo el reconocimiento de caracteres la última fase, por lo que se necesita la mayor efectividad en las fases anteriores para lograr un resultado óptimo.

El reconocimiento de caracteres se basa en la comparación del carácter de entrada, extraído de una imagen de entrada que previamente fue sometida a un proceso de binarización, segmentación y adelgazamiento, con un alfabeto que contiene todos los caracteres posibles.

Se puede considerar que el corazón de la etapa de reconocimiento de caracteres es el proceso de extracción de características, esta tarea se basa en el hecho de que caracteres similares deben dar lugar a representaciones similares. A partir de cada uno de los caracteres obtenidos se asocian unas características que los van a diferenciar y que ayudaran a obtener una clasificación óptima. Entre las características a tomar en cuenta tenemos la superficie, inclinación, entre otras.

El proyecto se divide en dos etapas: la de entrenamiento, en la cual se extraen las características que represente a cada carácter modelo para que puedan ser almacenadas en una librería y que posteriormente sirvan para realizar la respectiva comparación con la imagen de entrada en la fase de reconocimiento, como se muestra en la figura.

- Control de camiones: situando un lector de matrículas junto a la báscula que mide la carga del camión.
- Inventariado de vehículos: además de capturar la imagen de la matrícula se podría adquirir imágenes adicionales del vehículo, para poder determinar el estado del mismo en el instante de ingreso al aparcamiento, en caso de sufrir algún siniestro.

Organización del informe

Para resolver el problema de reconocimiento óptico de caracteres, se prevé dividir el proyecto en dos etapas:

1) Entrenamiento; y 2) Reconocimiento.

La descripción detallada sobre la implementación de cada una de estas dos etapas está organizada como se menciona abajo:

- El capítulo I realiza un estudio bibliográfico para analizar los diferentes métodos que actualmente se utilizan en la solución del problema propuesto. Las ventajas y desventajas de cada técnica también son descritos en este capítulo.
- En el capítulo II se muestra mayor información acerca de las placas vehiculares como características y clasificación de las matrículas en el Ecuador.

- En el capítulo III se hace una revisión general acerca del OCR Tesseract y se explica los pasos para generar librerías de entrenamiento.
- En el capitulo IV se explica cómo se fue realizando el entrenamiento para el reconocimiento de las placas vehiculares, lo errores en el transcurso de las pruebas y las soluciones aplicadas.
- En el capítulo V, se establecen las características del OCR Tesseract que se deben considerar, como es la inclinación máxima de un carácter, su rotación horizontal y vertical. Finalmente se comparan los resultados de Tesseract vs ANPR’s Comerciales y OCR’s de código abierto que actualmente existen en el medio.
- En el capítulo VI, se muestran los resultados de las pruebas de campo, la cual se realizó con el apoyo del grupo de adquisición.

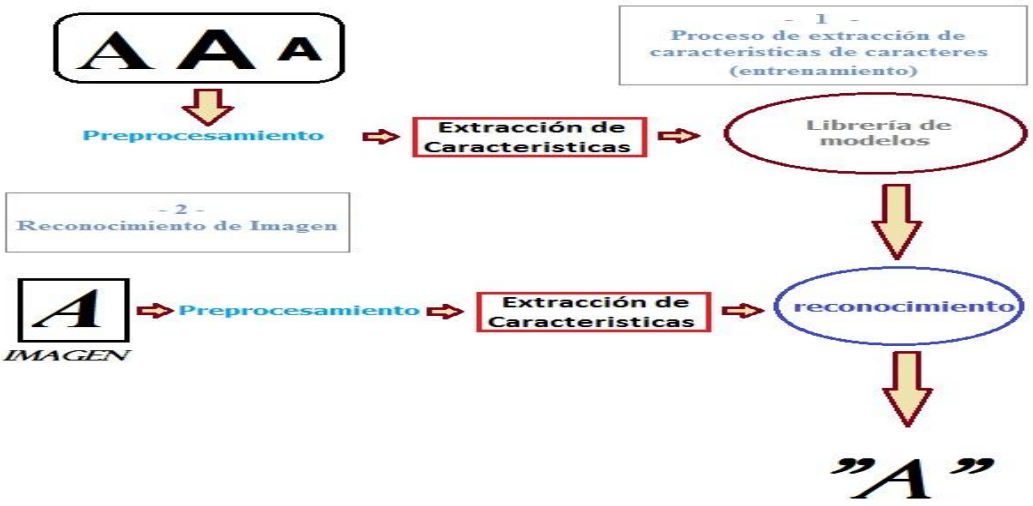
CAPITULO 1

REVISION BIBLIOGRAFICA

1.1 Resumen

En el presente capítulo se realiza una introducción general del reconocimiento de patrones, donde se resalta la importancia del proyecto propuesto.

Finalmente, se hace un análisis de las distintas técnicas para realizar el reconocimiento de caracteres, resaltando las ventajas y desventajas en el uso de cada una de las opciones expuestas anteriormente, y se indica qué técnica se va a utilizar junto con la justificación de su uso.



Objetivo

Desarrollar una solución que permita el reconocimiento de caracteres dentro de una imagen de una placa, que previamente se le aplicó diferentes técnicas para corregirla y optimizarla.

Justificación

La aplicación de OCR junto con otras técnicas ayudará al reconocimiento automático de matrículas que podrá ser utilizado para:

- La gestión de aparcamiento de abonados: usando la matrícula a modo de “llave” o “mando” para acceder a estacionamiento.
- Control de fraude en autopistas: para poder determinar si un vehículo fue plagiado o no es autorizado su circulación.
- Control de velocidad media en autopistas: situando lectores en varios accesos y salidas a la autopista.

1.2 Introducción

La capacidad de reconocer patrones es principalmente humana pues se relaciona con el reconocimiento o definición de un concepto. Los sistemas de reconocimiento de patrones artificiales simulan esta habilidad mediante la creación y el uso de modelos físicos o matemáticos. Sin embargo, dada una aplicación concreta, un sistema artificial es preferible al uso de seres humanos debido a su velocidad, exactitud y robustez.

Existen diferentes enfoques a la hora de generar modelos matemáticos para el reconocimiento de patrones. Los más populares son:

- Redes Neuronales: Son sistemas de computación que permiten la resolución de problemas que no pueden ser descritos fácilmente mediante un enfoque algorítmico tradicional. Con las redes se expresa la solución de un problema, no como una secuencia de pasos, sino como la evolución de un sistema inspirado en el funcionamiento del cerebro y dotado, por tanto, de cierta "inteligencia". Tal sistema no es sino la combinación de una gran cantidad de elementos simples de procesos (nodos o neuronas) interconectados que, operando de forma masivamente paralela, consiguen resolver el problema.

  El funcionamiento de un nodo es similar al de las neuronas biológicas presentes en el cerebro. Suele aceptarse que la información memorizada en el cerebro está relacionada con los valores sinópticos

de las conexiones entre las neuronas. De igual forma, se dice que las redes neuronales tienen la capacidad de "aprender" mediante el ajuste de las conexiones entre nodos. Estas conexiones tienen un valor numérico asociado denominado peso, que puede ser positivo (conexiones de excitación) o negativo (conexiones de inhibición). En la figura 1.1 se muestra la estructura básica de una red neuronal. [7]

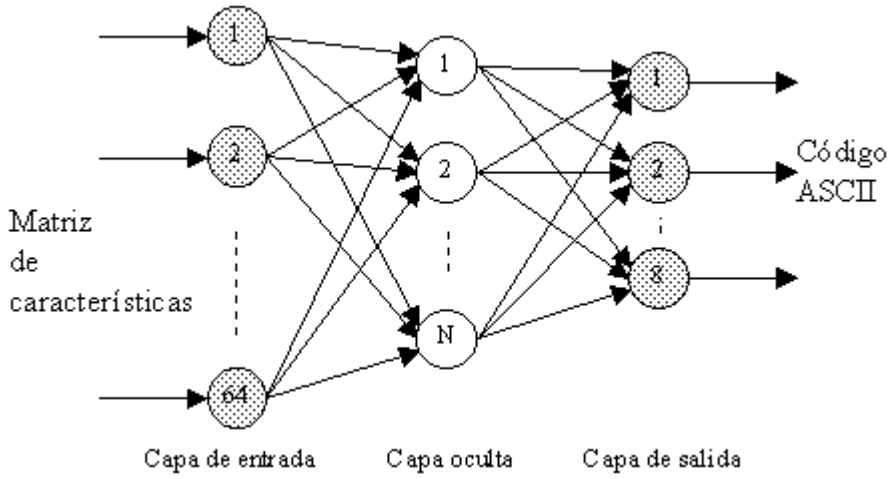


Figura 1.1.- Estructura básica de una Red Neuronal

- Método Lógico: Se basa en la idea de que el modelado del problema debe ser lo más cercano posible a la realidad del mismo, sin hacer suposiciones que no estén fundamentadas. Se utiliza conjuntos difusos y lógica simbólica, circuitos combinacionales y secuenciales, etc. Toman los valores de las variables de entrada, realizan algún procedimiento basados en un conjunto de reglas y deciden como modificar las variables de salida.

- Método Evolutivo: Es una estrategia de optimización estocástica, inspirada en la evolución de los seres vivos, en la cual, las especies naturales van evolucionando para adaptarse al medio que las rodea y aquellos individuos que tenga más éxito en tal adaptación tendrán mayor probabilidad de sobrevivir hasta la edad adulta y probablemente tendrán un número mayor de descendientes, que puede producir muchas veces un nuevo individuo mejor adaptado que cualquiera de sus padres, a las características de su medio ambiente. [18]

  En este método se define una estructura de datos que admite todas las posibles soluciones a un problema. Cada uno de los conjuntos de datos admitidos por esa estructura será un resultado. Unas serán mejores y otras peores. Las soluciones al problema son capaces de reproducirse entre sí, combinando sus características y generando nuevos resultados. En cada ciclo se seleccionan las soluciones que más se acercan al objetivo buscado, eliminando el resto. Las que fueron seleccionadas se reproducirán entre sí, permitiendo de vez en cuando alguna mutación o modificación al azar durante la reproducción. Dado que mantienen y manipulan varias en lugar de una única solución a lo largo de todo el proceso de búsqueda, suelen presentar tiempos de computación altos.

- Método Probabilista: Se basan en la teoría de la probabilidad y la estadística, utiliza análisis de varianzas, covarianzas, dispersión,

distribución, etc; para encontrar la solución que mejor resuelve un problema específico.

Supone que se tiene un conjunto de medidas numéricas con distribuciones de probabilidades conocidas y a partir de ellas se hace el reconocimiento. [6]

- Método Geométrico (Clustering): Es un método de agrupación de una serie de vectores de acuerdo con un criterio de cercanía. Esta cercanía se define en términos de una determinada función de distancia, geometría de formas, vectores numéricos, puntos de atracción, etc.

1.3 Técnicas OCR's

Existen muchos software's libres para el reconocimiento de patrones, entre ellos tenemos: Cuneiformes, el OCRopus, Tesseract, Ocrad, GOCR; también los hay de propietario como los son: ExperVision, FineReader, Microsoft Office Document Imaging, OmniPage, Readiris, ReadSoft, SimpleOCR, SmartScore.

A continuación se detallan las principales técnicas libres para el reconocimiento de caracteres:

- GOCR:

  Actualmente conocida como JOCR fue inicialmente desarrollada por Joerg Schulenburg, y su idea era conseguir un resultado rápido y aceptable sin aprender conocimientos teóricos. Más tarde fue mejorado. Este OCR es un método lógico que se basa en un conjunto de reglas, es utilizado para convertir y analizar un fichero gráfico de entrada en archivo de texto, en la cual reconoce cada carácter minimizando la distancia de los píxeles con un grupo de patrones que han sido entrenados previamente.

  GOCR puede ser usado con diferentes front-end y back-end con lo que se hace muy fácil el portarlo a diferentes sistemas operativos y arquitecturas, pero también hay una interfaz GTK en las fuentes del paquete.

  Además, el tamaño de las fuentes que soporta esta técnica es entre 20 - 60 píxeles y acepta muchos formatos de imágenes, como pnm, PBM, pgm, ppm, pcx, tga; otros formatos se convierten automáticamente utilizando netpbm-progs, estos tipos de imágenes incluyen png, jpg, tiff, gif, bmp, etc.

  Sin embargo, tiene inconvenientes con letras cursivas, texto escrito a mano, adicionalmente es muy sensible a imágenes que contenga ruido y grandes ángulos de inclinación, por lo que se necesita de un

gráfico de alta calidad para conseguir buenos resultados. También, es muy lento y tiene una tasa de error alto con respecto a otros software's.

- Ocrad:

  Creado por Antonio Díaz, esta técnica está basada en un método de extracción de características geométrico, que compara la forma de cada carácter (borde) con un modelo de clasificación binario donde agrupa símbolos por categorías. Incluye un analizador de diseño, capaz de separar las columnas o bloquees de texto que normalmente se encuentran en las páginas impresas.

  Ocrad lee una imagen en formato bmp, pgm o ppm y produce texto en formato byte o UTF-8. Puede ser usado como aplicación autónoma o como complemento de otros programas. Esta desarrollado sobre Unix. Es muy rápido, sensible al ruido y difícil de adaptar a nuevos símbolos.

- Tesseract:

  Desarrollado originalmente como software propietario en Hewlett-Packard y después de 10 años comprado, liberado y administrado por Google, está basado en el método de redes neuronales y es considerado uno de los software's libres de OCR más preciso, incluso mucho más potente que algunos software's de propietario, es multiplataforma y tiene un tiempo de ejecución aceptable, sin embargo

presentan problemas para la depuración en el caso de que haya un fallo en la segmentación.

El formato único que procesa es un TIFF y; se compila y ejecuta en Linux, Windows y Mac OS X[5].

Se muestra a continuación en la tabla 1.1 las ventajas y desventajas de las técnicas mencionadas anteriormente.

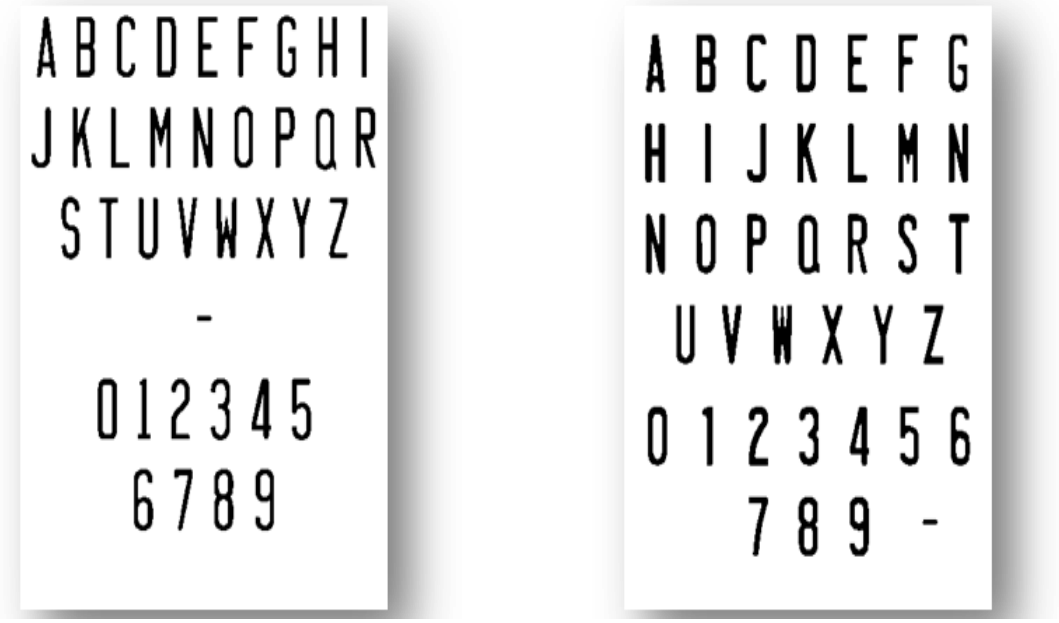
	Ocrad	**GOCR**	**Tesseract**
Precisión en el reconocimiento	Buena	Buena	Excelente
Tiempo de respuesta	Excelente	Mala	Buena
Sistemas Operativos desarrollado	Unix	Windows Linux	Windows Linux Mac OS
Lenguaje desarrollado	C++	C	C y C++
Licencia	GNU General Public License	GNU General Public License	Licencia Apache
Formato de imagen de entrada	Bmp, pgm, ppm	pnm, PBM, pgm y otros	TIFF
Calidad de imagen para análisis	Buena	Excelente	Buena
Versión estable disponible	Julio 2010	Agosto 2009	Sept. 2010

Tabla 1.1.- Ventajas y desventajas de OCR´s

1.4 Análisis de los OCR´s

Es muy importante saber las ventajas y desventajas que muestran cada uno de los algoritmos de OCR´s; sin embargo, para la determinación de qué algoritmo debe ser implementado en este caso, se debe hacer un análisis de ejecución y realizar una medición de los errores generados. Por lo que en este apartado se prueban y analizan los algoritmos mencionados anteriormente.

Para la prueba se utilizarán dos imágenes con diferentes tipos de fuente, muy comunes en las placas vehiculares, como se muestra en la Figura 1.2.



(a) **(b)**

Figura 1.2.- Imágenes fuente (de izquierda a derecha - DRIVER-GOTHIC - LICENSE-PLATE)

En ambas fuentes las letras “I” y “Q” han sido corregidas para ajustarse al diseño de las matrículas.

Los resultados obtenidos son los que se muestra a continuación:

- GOCR:

  Imagen 1

  Parámetros de configuración:

  - Se filtran los caracteres a reconocer por letras y números.
  - Se establece el símbolo “?” cuyos caracteres no se han podido reconocer (-u ¿).

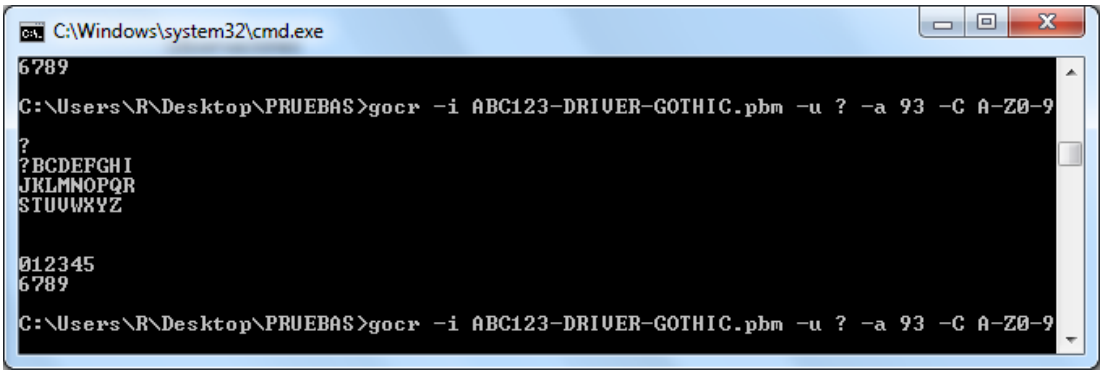


Figura 1.3.- Prueba con Imagen 1 para GOCR

Observaciones:

- No se pudo reconocer el carácter “A” ni el guión “-”
- Se muestra el símbolo “?” donde correspondía el carácter “A”, dicho carácter no existe en la imagen de prueba.

Imagen 2

Parámetros de configuración:

- Se establece el símbolo “?” cuyos caracteres no se han podido reconocer (-u ¿)

Observaciones:

- No se pudo reconocer 3 caracteres que son: “A”, “Q”, “W”

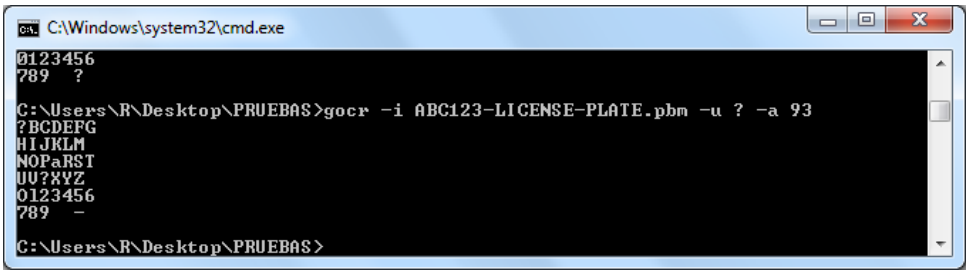


Figura 1.4.- Prueba con Imagen 2 para GOCR

- Ocrad:

  Imagen 1

  Parámetros de configuración:

  No existen parámetros de configuración que ayuden a optimizar el reconocimiento de caracteres.

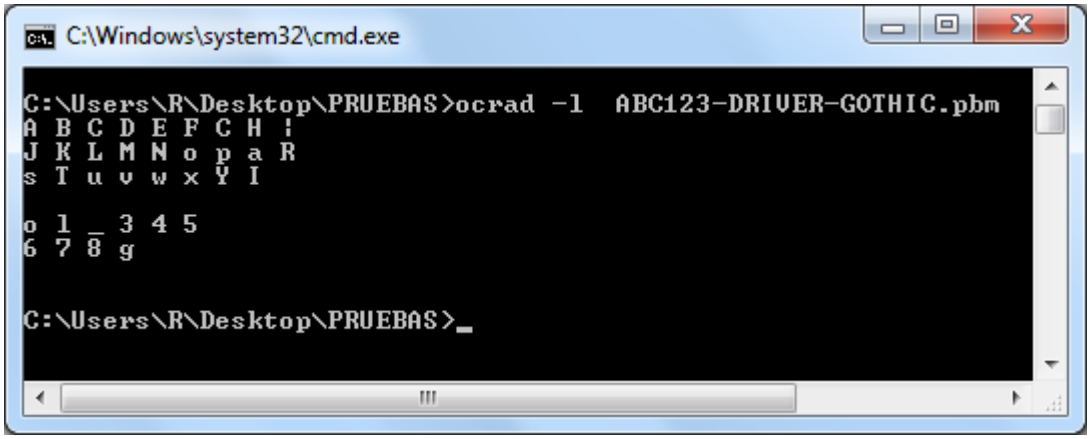


Figura 1.5.- Prueba con Imagen 1 para Ocrad

Observaciones:

- Se detectaron 8 caracteres que no fueron reconocidos que son: “I”, “G”, “Q”, “Z”, “0”, “2”, “9”, “-”.
- Otros caracteres fueron reconocidos pero en minúscula.

Imagen 2

Parámetros de configuración:

No existen parámetros de configuración que ayuden a optimizar el reconocimiento de caracteres.

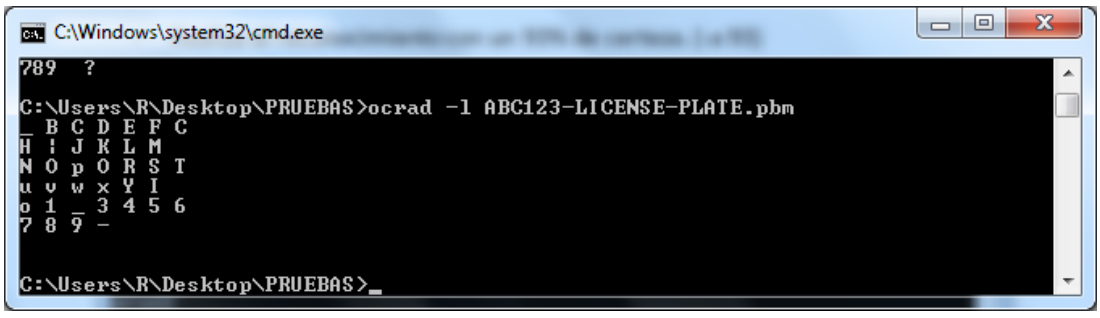


Figura 1.6.- Prueba con Imagen 2 para Ocrad

Observaciones:

El resultado obtenido muestra errores similares al de la imagen 1 de prueba, presenta 7 caracteres no reconocidos que son: “A”, “G”, “I”, “Q”, “Z”, “0”, “2”.

- Tesseract:

  Imagen 1

  Parámetros de configuración:

  En este ejemplo se utiliza el lenguaje por defecto identificado por “eng”, como se muestra en la Figura 1.7 y un lenguaje personalizado, creado a partir de un entrenamiento cuyo identificador es “mat”, como se muestra en la figura Figura 1.8.

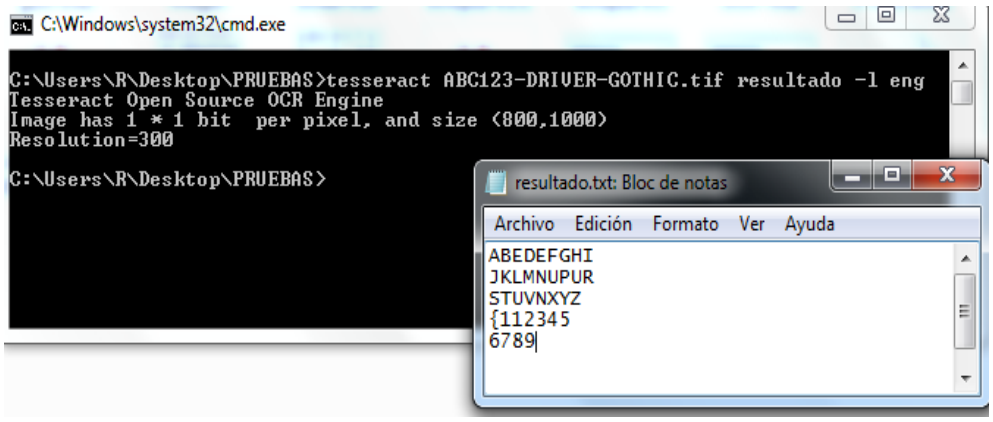


Figura 1.7.- Prueba con Imagen 1 para Tesseract con lenguaje default

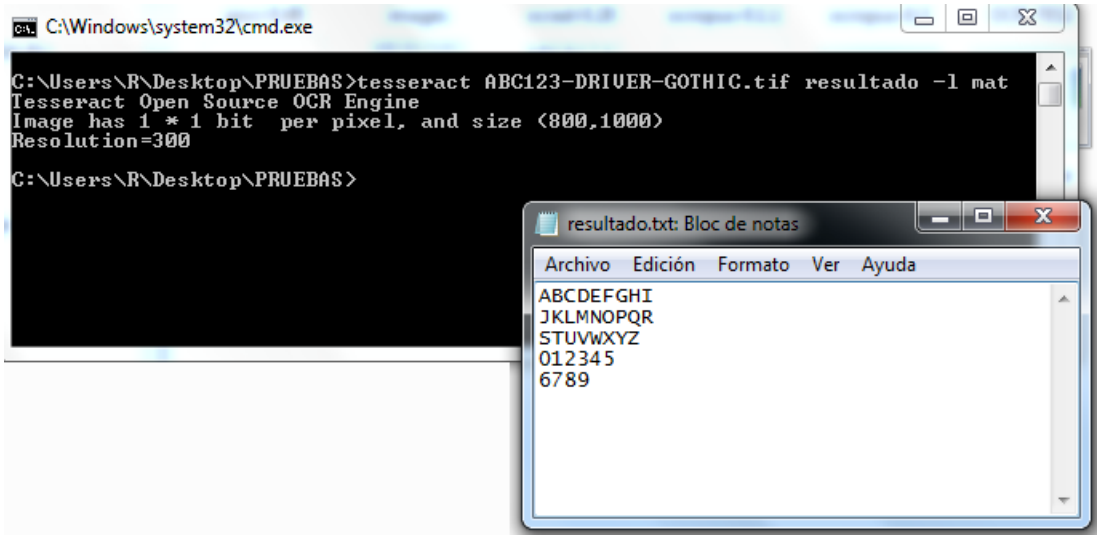


Figura 1.8.- Prueba con Imagen 1 para Tesseract con lenguaje personalizado

Observaciones:

- Son 6 caracteres que no fueron reconocidos siguiendo el modelo del Tesseract que esta por default, que son: “C”, “O”, “Q”, “W”, “0”, “-”.
- Empleando Tesseract personalizado fueron reconocidos todos los caracteres, a excepción del guión “-” que fue pasado por alto. Aquí, no se muestran errores, gracias a que se excluyen caracteres en minúscula, símbolos y signos que no se encuentran en una placa.

Imagen 2

Parámetros de configuración:

Al igual que en el ejemplo 1, se utiliza el lenguaje por defecto (-l eng) como se muestra en la Figura 1.9 y un lenguaje personalizado, creado a partir de un entrenamiento (-l mat), como se muestra en la figura Figura 1.10.

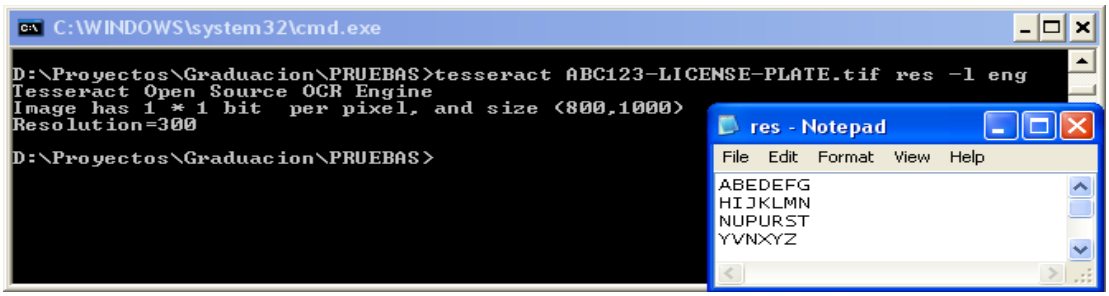


Figura 1.9.- Prueba con Imagen 2 para Tesseract

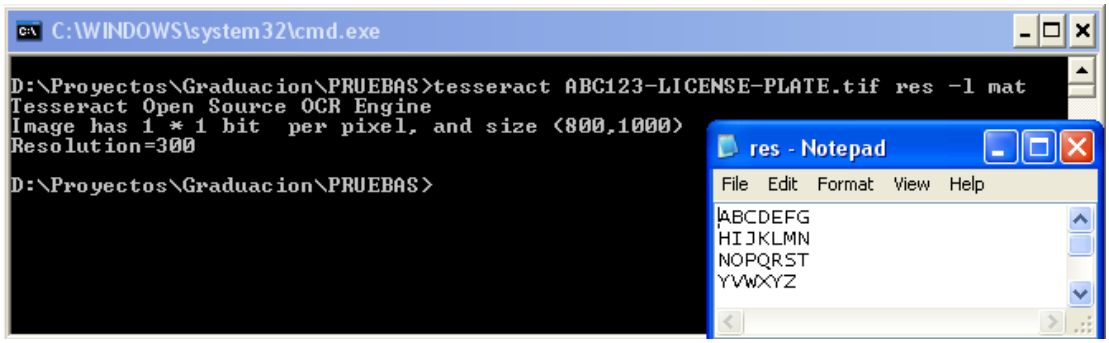


Figura 1.10.- Prueba con Imagen 2 para Tesseract con lenguaje personalizado

Observaciones:

- Son 6 caracteres que no fueron reconocidos siguiendo el modelo del Tesseract que esta por default, que son: “A”, “G”, “I”, “Q”, “Z”, “2”.
- Empleando Tesseract personalizado fueron reconocidos todos los caracteres. Aquí, no se muestran errores, gracias a que se excluyen caracteres en minúscula, símbolos y signos que no se encuentran en una placa.

A continuación se muestran las tablas 1.2 y 1.3, en donde se comparan los resultados obtenidos utilizando GOCR, OCRAD y Tesseract en las dos imágenes de prueba que se muestran en la Figura 1.2.

OCR	Total de Caracteres	Reconocidos Correctamente	No Reconocidos	Porcentaje de Error
GOCR	37	35	2	5.41
Ocrad	37	29	8	21.62
Tesseract	37	31	6	16.22
Tesseract con Lenguaje Personalizado	37	36	1	2.70

Tabla 1.2.- Comparación de resultados para la imagen 1 de prueba

OCR	Total de Caracteres	Reconocidos Correctamente	No Reconocidos	Porcentaje de Error
GOCR	37	34	3	8.10
Ocrad	37	30	7	18.92
Tesseract	37	31	6	16.22
Tesseract con Lenguaje Personalizado	37	37	0	0

Tabla 1.3.- Comparación de resultados para la imagen 2 de prueba

1.5 Conclusiones

A pesar de que GOCR dio muy buenos resultados, no es flexible en sus opciones para mejorar el reconocimiento; las cuales Tesseract si nos ofrece y que son muy útiles a la hora de personalizar un lenguaje, ya que nos permite realizar un entrenamiento previo mediante imágenes.

La tasa más baja por caracteres no reconocidos, es atribuible a Tesseract personalizado para ambas imágenes, con un 2.7% para la imagen 1 de prueba y 0% para la imagen 2 de prueba, gracias a que solo incluían alfanuméricos y el guión.

Según lo expuesto anteriormente y según los resultados obtenidos, se utilizara el Tesseract, ya que es un software libre y es robusto. Adicionalmente, es el mejor en cuanto a velocidad y precisión (en especial si la imagen es en blanco y negro), enfocándonos en el entrenamiento del algoritmo utilizando las herramientas que se dispone para así obtener los mejores resultados posibles.

CAPITULO 2

PLACAS VEHICULARES EN EL ECUADOR

2.1 Resumen

Las placas vehiculares son el registro que usan los vehículos automotores para su identificación y circulación legal en todo el territorio nacional. Por lo que en el presente capítulo se hablará un poco más extendido sobre sus características principales, las cuales se deben tomar en cuenta al momento de realizar el sistema que permita la detección de la placa y un óptimo reconocimiento de la misma.

2.2 Introducción

Las normas de tránsito contemplan que las entidades encargadas de otorgar las placas vehiculares son: Las Jefaturas Provinciales, Subjefaturas de

Tránsito y la Comisión de Tránsito del Guayas. Estas entidades gubernamentales deberán realizar la entrega de dos placas de identificación vehicular, que se colocan en la parte posterior y anterior del vehículo.

Todas las placas de identificación vehicular deben ser de una lámina metálica rectangular de 30 cm x 15 cm y deben cumplir con las normas de seguridad, recubrimiento y reflectancia que determine el Consejo Nacional de Tránsito.

Su diseño es único para todo el país, las letras y números están en relieve de 2 mm., en color negro mate, sobre el fondo reflectivo que indica el tipo de servicio y la pintura es de laca anticorrosiva. En la parte superior central lleva la palabra ECUADOR y contiene tres letras y de 3 - 4 dígitos, que son la clave necesaria para la identificación de cada vehículo.

La primera letra corresponde a la provincia donde ha sido matriculado, la segunda al tipo de servicio que presta el vehículo y la tercera es la letra del alfabeto a la que se le asignan de tres a cuatro dígitos que van desde el 0001 hasta el 9999, en ambos casos siguiendo el orden alfabético y numérico correspondiente.

A continuación se detallan las nomenclaturas que se utilizan en algunas provincias en la tabla 2.1.

Azuay - A Bolívar - B Cañar - U

Carchi - C	Galápagos - W	Manabí - M
Cotopaxi - X	Guayas - G	Morona - V
Chimborazo - H	Imbabura - I	Napo - N
El Oro - O	Loja - L	Pichincha - P
Esmeraldas - E	Los Ríos - R	Sucumbíos - K
Tungurahua - T	Zamora - Z	

Tabla 2.1.- Tabla de nomenclaturas para placas vehiculares en el Ecuador

También se deben considerar los colores de fondo para la identificación del servicio, que se determinan de la siguiente manera:

- Blanco - Plata: Particular

Figura 2.1.- Placa Particular

- Naranja: Alquiler

Figura 2.2.- Placa de Alquiler

- Oro: Función Legislativa, Función Jurisdiccional, Contraloría General del Estado, Procuraduría General del Estado, Superintendencia de Bancos, Superintendencia de Compañías, Consejo Nacional de Desarrollo, CONADE, Ministerios de Estado Tribunal Supremo Electoral

Figura 2.3.- Placa Gubernamental

- Rojo: Vehículos de Internación Temporal
- Verde Limón: las entidades que integran la administración provincial o cantonal dentro del régimen seccional, como los Consejos Provinciales y Municipios

Figura 2.4.- Placa Municipal

- Azul: Diplomáticos, Consulares, Asistencia Técnica y Organismos Internacionales

2.3 Características generales

Para el presente proyecto se ha tomado en consideración solo las placas vehiculares, que son con fondo blanco; sin embargo no habría ningún tipo de

impedimento para utilizar placas con otro tipo de fondo, ya que el motor de OCR trabaja con imágenes previamente binarizadas, es decir en blanco y negro, por lo cual no afectaría en el análisis final.

CAPITULO 3

TESSERACT

3.1 Resumen

El presente capítulo nos enseña paso a paso como se debe entrenar el motor OCR Tesseract para crear un nuevo idioma o un subconjunto de un idioma existente.

3.2 Introducción

Tesseract es un motor OCR libre, fue desarrollado originalmente por Hewlett Packard como software propietario entre 1985 y 1995. Tras diez años sin ningún desarrollo, fue liberado como código abierto en el año 2005 por

Hewlett Packard y la Universidad de Nevada, Las Vegas. Actualmente, Tesseract es desarrollado por Google y distribuido bajo la licencia Apache. Tesseract en 1995 fue considerado como uno de los tres motores OCR con mayor precisión en el reconocimiento de caracteres.

Tesseract fue originalmente diseñado para reconocer solo texto en inglés, sin embargo, actualmente Tesseract es capaz de procesar en otros idiomas como lo son el francés, italiano, alemán, español, portugués y puede ser entrenado para funcionar con otros idiomas más; siempre que estos idiomas se manejen de izquierda a derecha y de arriba hacia abajo. El árabe es una secuencia de comandos improbable de ser manejado por Tesseract, para esto debería tener algoritmos más especializados que al momento no dispone; con idiomas como el Chino suele ser más lento el reconocimiento pero funciona bien.

3.3 Entrenamiento paso a paso

Para entrenar al motor OCR Tesseract se debe crear algunos archivos de datos en el subdirectorio *tessdata, los cuales* deberán ser compactados en un solo archivo, utilizando la herramienta llamada *combine_tessdata.*

Siguiendo el estándar ISO 630-3 se debe tener un estilo de nombramiento de archivos, el cual se rige de la siguiente manera *código_lenguaje.nombre_archivo,* por ejemplo los archivos que son utilizados para el inglés son los siguientes:

- tessdata/eng.config
- tessdata/eng.unicharset
- tessdata/eng.unicharambigs
- tessdata/eng.inttemp
- tessdata/eng.pffmtable
- tessdata/eng.normproto
- tessdata/eng.punc-dawg
- tessdata/eng.word-dawg
- tessdata/eng.number-dawg
- tessdata/eng.freq-dawg

y los archivos más importantes son:

- tessdata/eng.traineddata
- tessdata/eng.user-words

El archivo *traineddata* es simplemente una concatenación de los archivos de entrada (archivo compactado).

Si solo se quiere reconocer una gama limitada de tipos de letra podría ser suficiente que solo se cree los archivos unicharset, intTemp, normproto, pfftable; si se requiere mejorar la precisión se puede utilizar el resto de archivos dependiendo de su aplicación.

Algunos de los procedimientos son inevitablemente manuales, se espera que en un futuro existan más herramientas automatizadas, que podrían requerir un proceso de instalación o generación un poco compleja.

3.3.1 Generar librerías de entrenamiento

El primer paso es determinar el conjunto de caracteres que van a ser utilizados, y diseñar una imagen con texto que contenga un conjunto de ejemplos. Los puntos más importantes que se deben tomar en cuenta a la hora de crear una imagen de entrenamiento son:

- Se debe tener como mínimo 10 muestras para los caracteres especiales, sin embargo puede ser suficiente con 5 muestras.
- Para los caracteres más frecuentes se deben crear muchas más muestras, por lo menos debe ser 20 muestras.
- Se debe hacer frases que tengan significado, no se debe hacer de la siguiente manera: 012345 ¡@#%^&, ya que no da muchas posibilidades de conseguir buenas mediciones en los caracteres especiales, en nuestro caso esto no aplica debido a que las placas no se generan en base a frases o palabras conocidas.
- En lo posible los datos de entrenamiento deben ser agrupados por tipo de letra.
- No es necesario crear muchas muestras por tamaño, 10 muestras son suficientes; sin embargo si se requiere para letras muy pequeñas, que están cerca de los 15 pixeles debe aumentar el número de muestras.

Se debe guardar el texto de entrenamiento en un archivo de texto UTF–8, para ser usado en el próximo paso. Por cada tipo de letra se debe crear una imagen de formato *TIFF*.

Para el segundo paso, Tesseract necesita un archivo “box”, en el que se encuentran el detalle de cada una de las imágenes de entrenamiento.

Este archivo “box” es un archivo de texto, que lista todos los caracteres que se encuentran en la imagen de entrenamiento, en orden y uno por línea. En la versión de Tesseract 3.0 muestra un archivo de texto con el formato requerido, pero pueden ser caracteres distintos a los de la imagen de entrenamiento, los cuales deben ser editados manualmente para que coincidan con los de la imagen, ya que en esto consiste el entrenamiento.

Se debe ejecutar el comando Tesseract sobre cada una de las imágenes de entrenamiento con la siguiente línea de comando:

Tesseract imagen_entrenamiento.tif imagen_entrenamiento batch.nochop makebox

De aquí se debe editar el archivo *imagen_entrenamiento.box (archivo generado mediante el comando tesseract)* y poner el carácter correspondiente, en el archivo, al comienzo de cada línea, en lugar del carácter incorrecto que Tesseract puso, un editor que entiende dicho código es el HTML, también el Bloc de Notas ++, que sería de gran ayuda para realizar las correcciones necesarias.

De igual manera existen varias herramientas visuales para la edición del archivo “box”, que se muestran en la tabla 3.1.

Nombre	Última actualización	Lenguaje
pytesseracttrainer	Enero 2011	Python, GTK2
cowboxer cowboxer	Enero 2011	C++, Qt
QT Box Editor	Enero 2011	C++, Qt
Tesseract-OCR boxfile AJAX editor	Sep 2010	herramienta en línea
owlboxer	Jul 2010	C++, Qt
Tessboxer	Jun 2009	.NET
boxfilereader.php	Mar 2009	php
tessboxes	Nov 2008	C
JTess eract	Octubre 2008	C#
wx-tetra	Sep 2008	perl, wx
bbtesseract	Jul 2008	VB.NET 2008

Tabla 3.1.- Herramientas visuales para edición de archivo BOX

Si se requiere entrenar un conjunto de caracteres, es una buena idea crear un buen archivo “box” en un tipo de letra, y luego ejecutar para todo el proceso de entrenamiento el resto de los archivos de la siguiente manera:

tesseract imagen_entrenamiento.tif imagen_entrenamiento -l nuevolenguaje batch.nochop makebox

Esto permitirá que el segundo archivo “box” sea más fácil de hacer, ya que hay una muy buena probabilidad que Tesseract reconozca la mayor parte del texto correctamente.

El siguiente pase es ejecutar Tesseract en modo de entrenamiento, mediante los siguientes comandos:

tesseract imagen_entrenamiento.tif imagen_entrenamiento nobatch box.train

o con el comando

tesseract imagen_entrenamiento.tif imagen_entrenamiento nobatch box.train.stderr

El propósito de este comando es obtener las características de los caracteres que fueron extraídas de las imágenes de entrenamiento, los cuales son almacenados en archivos “.**TR**”, los que a su vez se deben agrupar para crear prototipos. Estos prototipos son creados con la ayuda de los programas *mftraining* y *cntraining*.

Obtener el conjunto de caracteres

Tesseract necesita conocer el conjunto de caracteres posibles que podrá reconocer, los cuales son todos aquellos que se encuentran en los archivos

“box”. Para generar el archivo de datos **unicharset** utilizamos el programa **unicharset_extractor** (ejecutado sobre todos los archivos “box”):

unicharset_extractor imagen_entrenamiento1.box imagen_entrenamiento2.box ...

Comando para mftraining:

mftraining -U unicharset -O lang.unicharset imagen_entrenamiento_1.tr imagen_entrenamiento_2.tr ...

Mftraining dará como resultado tres archivos de datos: *intTemp*(prototipo de forma), *pffmtable*(número de características por cada carácter) y *Microfeat* que no se lo utiliza.

Comando para cntraining:

cntraining imagen_entrenamiento_1.tr imagen_entrenamiento_2.tr …

Al ejecutar este comandos se obtendrá un archivo de datos *normproto*(prototipo sensitivo a la normalización del carácter).

Tesseract puede utilizar hasta 5 diccionarios por cada idioma, cuatro de los archivos están codificados como DAWG y el último en archivo de texto UTF8. Para hacer los archivos diccionario DAWG es necesario una lista de palabras del idioma, esta lista se la pasa a formato UTF8. Luego se separa la lista de palabras en “palabras frecuentes” y “resto de palabras”, finalmente se usa el siguiente comando *wordlist2dawg* para construir el archivo DAWG:

wordlist2dawg frequent_words_list lang.freq-dawg lang.unicharset

wordlist2dawg words_list lang.word-dawg lang.unicharset

Se debe tener en cuenta que crear el diccionario de palabras es opcional, pero ayuda a mejorar el reconocimiento de caracteres.

Estos archivos no son utilizados para nuestro proyecto debido a que no se utilizan palabras frecuentes.

El último archivo que Tesseract utiliza es *unicharambigs,* este representa las ambigüedades intrínsecas entre los caracteres o conjunto de caracteres, por ejemplo, se muestra en una imagen el siguiente conjunto de caracteres “rn” y lo reconoce como el carácter “m”.

Actualmente, este archivo se genera manualmente y al igual que el diccionario es opcional su generación. Tampoco es considerado en nuestro proyecto debido a que si puede darse algo parecido al ejemplo: “rm” y “m”, además al reconocer caracteres, solo se reconocerá un carácter a la vez, por lo que no es necesario el manejo de ambigüedades.

Finalmente se cogen todos los archivos (normproto, normproto, Microfeat, inttemp, pffmtable) y se los renombra con un prefijo *lang (es remplazado por el nombre de nuestro lenguaje),* donde ese prefijo lang consta de un código de 3 letras que representa el idioma y luego se ejecuta el comando *combine_tessdata* de la siguiente manera:

combine_tessdata lang

El resultado es un archivo *lang.traineddata, que* va directamente en el directorio *tessdata.*

Tesseract, puede reconocer utilizando el nuevo lenguaje utilizando el siguiente comando:

tesseract image.tif output -l lang

CAPITULO 4

ENTRENAMIENTO Y CORRECCION DE ERRORES

4.1 Resumen

En el presente capítulo se mostrarán las pruebas realizadas con el motor OCR Tesseract con el diccionario o librería que viene por defecto y con nuestro entrenamiento personalizado, una clasificación para determinar el estado de las placas y todas las correcciones a los problemas encontrados durante la etapa de pruebas.

4.2 Prueba inicial

Se obtuvieron 100 muestras de placas vehiculares, las cuales ya han sido segmentadas caracter por caracter y binarizadas (blanco y negro) para proceder directamente con el reconocimiento de patrones.

Este entrenamiento es catalogado como básico ya que se lo realizó utilizando solo las imágenes plantillas del tipo de fuentes similares a los de las matrículas vehiculares, en la Figura 4.1 muestra la plantilla que se utilizo para entrenar al motor.

El objetivo de esta prueba inicial es tener una mejor perspectiva del comportamiento del OCR al tratar con esta clase de imágenes, puesto que en las siguientes secciones se procederán a realizar una serie de pruebas y correcciones respectivas.

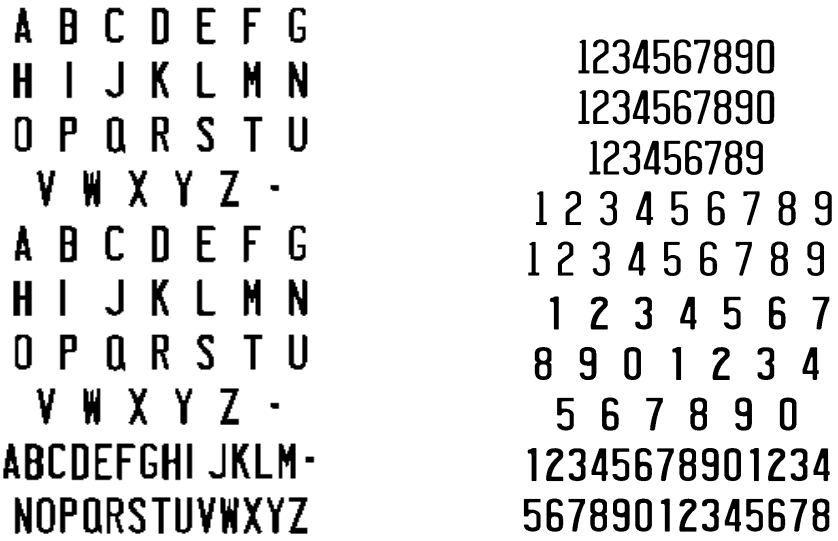


Figura 4.1.- Imágenes plantilla

4.3 Resultados de las pruebas

En la tabla 4.1 se muestra por cada placa un resultado usando el lenguaje que viene por default en el Tesseract, que se muestra en la primera fila y en la siguiente, muestra los resultados con el lenguaje al que Tesseract fue entrenado, y en cada resultado se muestra el porcentaje de acierto que se obtuvo.

Placa 1

Resultado:

E]	@	S	L3	B]	Q	17%
0	B	S	6	0	0	67%

Placa 2

Resultado:

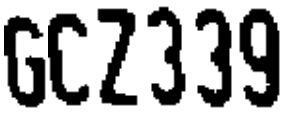
G	P	P	1	3	Z	83%
G	P	P	1	3	Z	83%

Placa 3

Resultado:

E	E	E]	Z	H	U	0%
E9	3	Q	2	0	0	17%

Placa 4



Resultado:

@	E	Z	3	E]	E]	33%
8	0	Z	3	0	61	33%

Placa 5

Resultado:

{5	[3	H	E]	L1	Q	0%
E9	0	0	0	53	0	0%

Placa 6

Resultado:

[II	****	Ki	D]	Q	M	0%
El1	****	0	0	0	0	33%

Placa 7

Resultado:

CB	Li	Ei	Q]	U	E	0%
E3	0	E3	0	0	0	0%

Placa 8

Resultado:

E	E	W	H	LH	U	17%
3	0	B	0	E9	0	0%

Placa 9

Resultado:

YH	Fi	PJ	****	H	****	0%
FH	F9	P4	****	0	****	0%

Placa 10

Resultado:

E	K]	I	E	H	U	0%
3	31	1	E3	0	0	17%

Placa 11

Resultado:

E	H]	****	2	Q	Q	17%
3	61	****	Z	0	0	0%

Placa 12

Resultado:

@	K]	U]	Q	E]	m	0%
9	61	01	5	0	01	17%

Placa 13

Resultado:

E	K!	E	H	E	E	0%
0	Q	3	0	0	D	17%

Placa 14

Resultado:

E	Q	[1	5	3	3	1	57%
61	0	0	5	3	3	1	57%

Placa 15

Resultado:

EH	C	E	Q]	E	****	33%
ED	C	6	9	8	****	33%

Placa 16

Resultado:

E	[I]	Y	E	E!	E	17%
3	El1	Y	6	0	6	33%

Placa 17

Resultado:

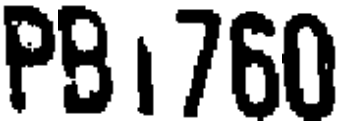
[5	H	II	[I	H	E	0%
0	0	0	0	0	R	0%

Placa 18

Resultado:

B]	****	IE	B]	7	H	17%
E9	****	0	61	7	0	17%

Placa 19



Resultado:

P	Q	****	7	[5	CU	33%
P	0	****	7	E8	Q	33%

Placa 20

Resultado:

W	H	****	H	U]	E	0%
Q	0	****	0	01	8	0%

Placa 21

Resultado:

B	EJ	N	5	El	****	33%
6	0	0	5	B	****	17%

Placa 22

Resultado:

G	[H	E	[5	El	El	17%
G	EQ	0	6	0	51	33%

Placa 23

Resultado:

[E	L	@	****	E1	F]	17%
EB	L	Q	****	61	0	17%

Placa 24

Resultado:

E	[G	T	2	E]	Li	33%
8	0	T	Z	0	G3	33%

Placa 25

Resultado:

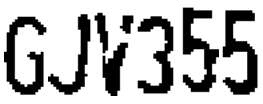
{E	E	G]	7	E]	I]	17%
E9	6	L0	7	0	0	17%

Placa 26

Resultado:

@1	II]	B	[H	[Q	****	17%
9	D1	B	ED	E9	****	17%

Placa 27



Resultado:

E]	****	H	3	E	Q	17%
6	****	Q	3	6	6	17%

Placa 28

Resultado:

E	E	W	[5	3	B]	17%
E9	0	V	O	3	0	33%

Placa 29 *Resultado:*

N	E]	[5	****	Pl	H	0%
Q	03	E9	****	P1	0	0%

Placa 30 *Resultado:*

G	Q	X	H	E1	E	33%
6	O	X	0	6	B	50%

Placa 31 *Resultado:*

[E	[G	M	[il	Ei	Q	0%
0	EQ	0	01	0	0	0%

Placa 32 *Resultado:*

[E	L	V	E	LU	B}	33%
0	L	V	0	0	81	33%

Placa 33 *Resultado:*

[E	B]	T	[H	2	il	33%
E8	01	T	01	2	0	33%

Placa 34 *Resultado:*

E	[H	H1	B	E	E	0%
0	LQ	0	0	0	6	0%

Placa 35 *Resultado:*

[G	ri]	Li	H	[S]	[II	0%
EQ	0	59	D1	E41	E Q	0%

Placa 36 *Resultado:*

E	LQ	H	EJ	****	H	0%
0	LQ	0	0	****	0	0%

Placa 37 *Resultado:*

@	H]	E	7	Q	7]	33%
0	O	E	7	9	0	67%

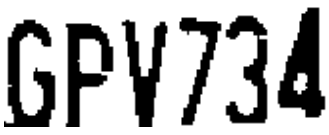
Placa 38 *Resultado:*

[B	IH	[Q	EI	[H	L1	0%
EB	61	E9	0	0	9	0%

Placa 39 *Resultado:*

H	E	J	IQ!	Li	E	17%
0	0	J	0	0	3	33%

Placa 40 *Resultado:*



[H	F5	W	7	3	E]	33%
0	F9	FI	7	3	0	33%

Placa 41

Resultado:

@	@	****	2	****	****	17%
0	Q	****	2	****	****	17%

Placa 42

Resultado:

B	H	E.]	[5	[ii	7	33%
6	H	61	6	63	7	33%

Placa 43

Resultado:

B	PJ	H	E	El	E	0%
6	E1	0	6	9	0	33%

Placa 44

Resultado:

[H	C	B	L2]	H	U	17%
0	C	0	61	0	0	17%

Placa 45

Resultado:

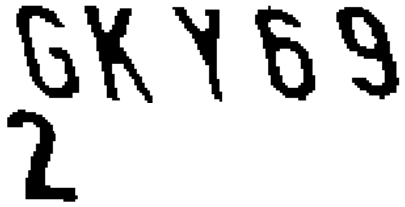
6	L	P	3	7	E	67%
6	4	P	3	7	6	50%

Placa 46

Resultado:

Ei	E	L!	E]	****	l]	0%
0	0	3	0	****	0	0%

Placa 47



Resultado:

B	*** *	`1	*** *	*3	7.	0%
6	*** *	Y	*** *	9	1	33%

Placa 48

Resultado:

{Q	W	C	E1	[1	E	17%
E9	E9	C	61	0	0	17%

Placa 49

Resultado:

B	\.	k	[1]	YS	\	0%
6	L	L	0	Q	X	17%

Placa 50

Resultado:

Bi	[U	****	@1	E]	E	0%
EB	E0	****	E9	0	3	0%

Placa 51

Resultado:

W	[B	K	5	E]	2	50%
0	0	K	5	9	2	67%

Placa 52

Resultado:

[li	li]	EI	Bi	E]	****	0%
E9	01	01	G3	E9	****	0%

Placa53

Resultado:

[5	Pl	X	H	E]	[H	17%
E9	F1	X	6	9	EQ	33%

Placa 54

Resultado:

Ya	****	S	T	B	Q	17%
6	****	S	T	0	D	17%

Placa 55

Resultado:

EH	B]	F	1	E]	1	50%
0	03	F	1	E9	1	50%

Placa 56

Resultado:

@	W	W	****	****	il	0%
B	F9	F9	****	****	0	0%

Placa 57

Resultado:

@	*E*	*{E*	*[l*	*[G*	*E]*	*0%*
9	5	E9	0	EQ	0	0%

Placa 58

Resultado:

H	E	M	E1	H	E	17%
3	Q	0	31	0	6	0%

Placa 59

Resultado:

[5	fl	U]	H	II	[E	83%
E9	FI	03	0	FI	E3	83%

Placa 60

Resultado:

{P]	IE	[5	li	****	E	0%
E9	I9	19	5	****	6	17%

Placa 61

Resultado:

E	R	E]	II	7	H	17%
0	0	0	0	7	9	17%

Placa 62

Resultado:

[H	B]	[I	LH	E]	Q	0%
E6	01	EI	0	0	0	0%

Placa 63

Resultado:

E	V	H	E	E	E	0%
13	F	0	6	D	6	0%

Placa 64

Resultado:

Li	E	V	B]	****	{B	17%
E9	0	V	01	****	E8	17%

Placa 65

Resultado:

W	****	I3}	EI	E1	1	17%
B	****	131	61	21	1	17%

Placa 66

Resultado:

[B	IH	[H	E1	****	EI	0%
E9	19	E8	0	****	0	0%

Placa 67

Resultado:

LE	[EI	H]	7	Q	H]	17%
03	D1	51	7	5	81	33%

Placa 68

Resultado:

Ei	H]	W	E1	PI	FI	0%
83	01	D1	5	0	FI	17%

Placa 69

Resultado:

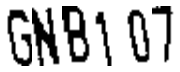
LE	H	E]	****	7	E	17%
LE	EI	0	****	7	6	17%

Placa 70

Resultado:

BE	W	K1	3	E	Li	8	29%
03	0	0	3	6	0	8	29%

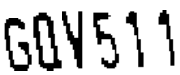
Placa 71



Resultado:

Li	RI	Q	****	****	****	0%
R	0	19	****	****	****	0%

Placa 72



Resultado:

Ki	N	****	E	X	****	0%
0	0	****	5	3	****	17%

Placa 73

Resultado:

E	E	[E	7	3	E	33%
3	3	0	7	3	0	33%

Placa 74

Resultado:

LE	****	Li	[H	EH	H	EH	0%
L9	****	19	I0	E0	I1	0	14%

Placa 75

Resultado:

E	E	C	LH	2	{5	33%
0	0	C	0	2	E3	33%

Placa 76

Resultado:

5	E	EI	Q	Q	H	0%
6	5	0	9	5	0	0%

Placa 77

Resultado:

[E	fi	[E	E]	3	EE	17%
E9	F3	0	0	3	6	0%

Placa 78

Resultado:

LH	Lu	U	3	H	EI	17%
L9	LQ	D	3	0	0	17%

Placa 79

Resultado:

I?	EI	B	Ei	EI	E	17%
61	0	B	E	0	0	0%

Placa 80

Resultado:

IE}	****	****	[I	EI	EI	0%
I9	****	****	EI	EI	61	0%

Placa 81

Resultado:

YH	I1	****	I}	EH	FH	0%
39	9	****	0	E9	19	0%

Placa 82

Resultado:

G	@	[Z]	1	EI	E	33%
6	B	E3	1	EI	9	17%

Placa 83

Resultado:

H	M	I	N	H	E	0%
Q	N	D	Q	0	6	0%

Placa 84

Resultado:

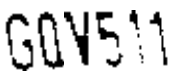
@	E!	1	M	li]	I	0%
B	EQ	1	0	0	2	33%

Placa 85

Resultado:

Q	E]	H]	5	[H	Q	17%
0	01	0	5	0	9	33%

Placa 86



Resultado:

F	H	****	E	****	N	0%
P	0	****	5	****	X	17%

Placa 87

Resultado:

E	Q	L	B}	****	E	17%
0	0	L	61	****	9	33%

Placa 88

Resultado:

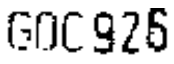
G	R	J	EI	2	[H	67%
G	R	J	0	Z	E0	50%

Placa 89

Resultado:

Ei	E	EE	E]	EE	****	0%
I39	C	03	0	8	****	17%

Placa 90



Resultado:

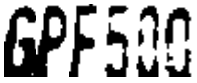
G	H	C	****	Z	hi	33%
G	Q	C	****	2	6	67%

Placa 91

Resultado:

G	****	****	[E	[5	****	[H	14%
G	****	****	ED	0	****	0	43%

Placa 92 *Resultado:*



F	Vi	E	H	{1	BI	0%
F	F9	Q	B	31	01	0%

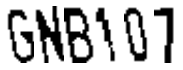
Placa 93 *Resultado:*

K1	G	EE	m	E	****	17%
61	G	0	0	6	****	50%

Placa 94 *Resultado:*

@	[U	E]	Z	[3	****	0%
3	E0	01	2	0	****	17%

Placa 95 *Resultado:*



E	N	H	X	KH	H	17%
0	W	B	K	Q	0	17%

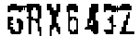
Placa 96 *Resultado:*

rÂ¤	[EI	E}	[H	7	G]	17%
D	0	E3	ED	7	63	17%

Placa 97 *Resultado:*

[E	li]	{E	E	E]	3	****	15%
0	13 1	0	6	61	3	****	29%

Placa 98 *Resultado:*



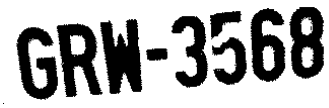
E	N	FJ	Lil	H	E	I.	0%
6	0	61	19	0	6	L	0%

Placa 99

Resultado:

6	V	/I	I	I	3		0%
G	V	/I	1	1	3		33%

Placa 100



Resultado:

*	B	IH	3	*	G	8	29%
G	B	H	3	*	6	8	43%

Tabla 4.1.- Resultados por cada placa utilizando lenguaje por default y lenguaje entrenado

4.4 Conclusiones de los resultados obtenidos

La Figura 4.2 muestra un gráfico comparativo de los resultados con las dos opciones de reconocimiento que brinda el Tesseract, utilizando las 100 imágenes de prueba. Según la gráfica mostrada existe un promedio del 15,70% de asertividad por cada placa en el modo del Tesseract que viene por default, esto quiere decir que en una placa vehicular en promedio solo se puede reconocer 1,10 caracteres; por otro lado se obtuvo el 21,17% de asertividad utilizando el Tesseract entrenado, lo cual significa que se puede reconocer 1,48 caracteres por placa.

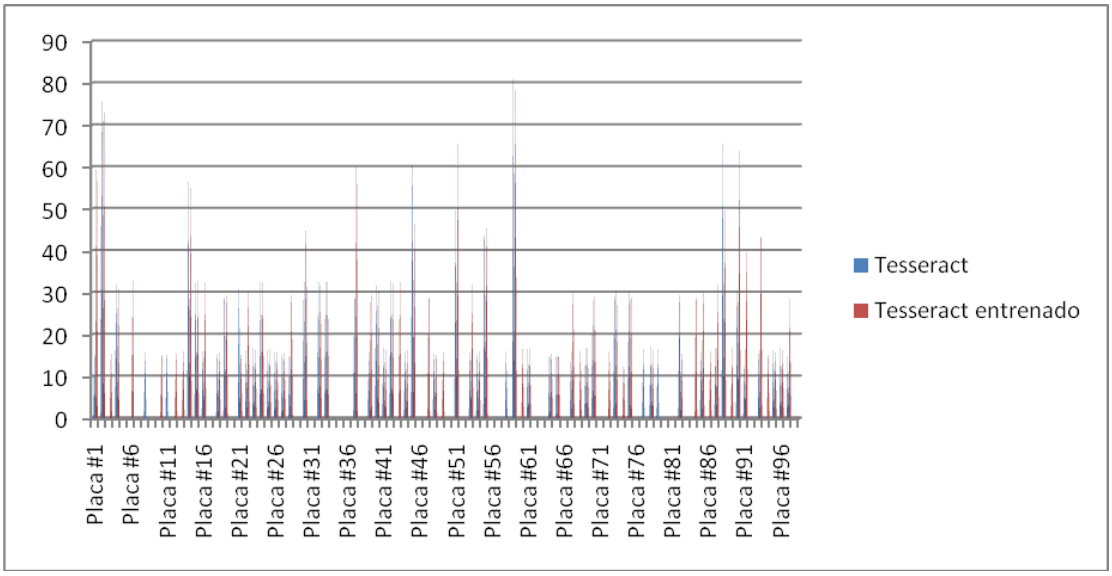


Figura 4.2.- Cuadro comparativo de pruebas en Tesseract por default vs Tesseract entrenado

En consecuencia, el Tesseract entrenado dio mejores resultados. Sin embargo, no se puede tomar como referencia y se debe realizar un mejor entrenamiento al motor para que se pueda hacer de manera satisfactoria el reconocimiento, ya que solo el 12% de las placas vehiculares tuvieron una asertividad mayor al 50% de los caracteres en la placa total. Adicionalmente, si se revisa los peores porcentajes de asertividad se puede concluir que hay menores posibilidades de reconocer caracteres cuando la imagen es sumamente pequeña, de dimensiones menores a 15 x 30.

4.5 Características y clasificación de las placas

Debido a que los grupos que nos anteceden no nos facilitaron imágenes únicamente con la región de la placa durante el proceso de nuestras pruebas

y desarrollo, ya que se ha venido trabajando en paralelo. Se procedió a tomar una muestra de 100 placas de vehículos, algunas obtenidas utilizando cámaras convencionales en diferentes condiciones y otras proporcionadas por nuestro profesor Phd. Boris Vintimilla.

4.5.1 Características de las imágenes

Las muestras tomadas se las procedió a binarizar y a segmentar sin realizarles mayores ajustes ni correcciones. Como resultado de la segmentación se obtuvieron de 6 a 7 imágenes por placa con las cuales se procedió a realizar las pruebas y un análisis para mejorar los resultados del OCR Tesseract.

Durante esta organización se estableció la secuencia para cada placa y sus respectivas imágenes. Cada placa es identificada con la letra “p” y un número de secuencia “n” seguido por el guión bajo y el número que a su vez seguirá de otra secuencia según la segmentación. En la Figura 4.3 se muestra un ejemplo de lo mencionado.

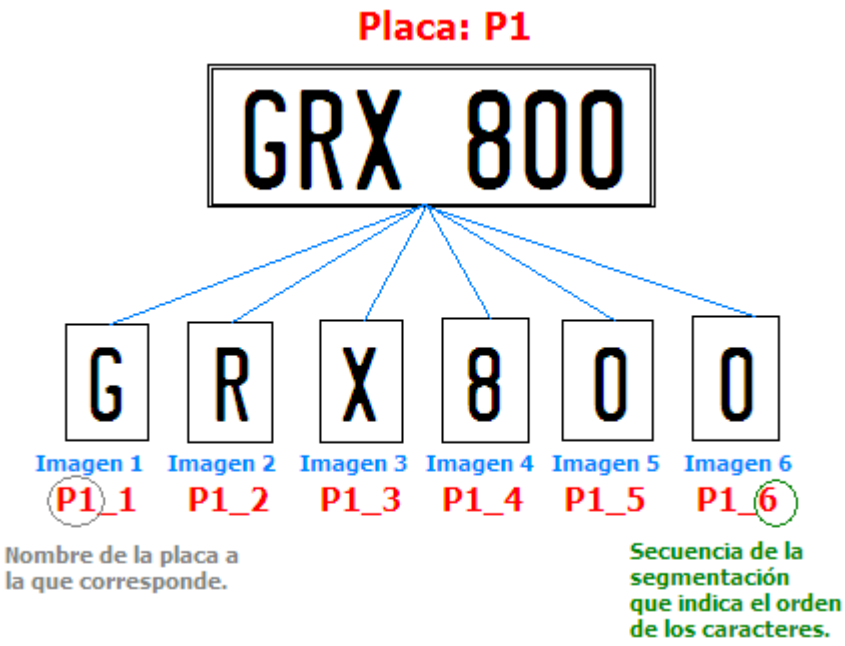


Figura 4.3.- Secuencia de una placa vehicular

Realizada esta organización cabe mencionar las siguientes observaciones:

- De las 100 Placas, se obtuvieron un total de 606 imágenes.
- Por cada placa existen 6 o 7 imágenes (para las nuevas placas), de los cuales las 3 primeras corresponden a los caracteres alfabéticos y las restantes corresponden a caracteres numéricos.
- Las imágenes se encuentran binarizadas (blanco y negro)
- El formato es TIFF, sin compresión
- La resolución varía para algunos casos, entre 72ppp y 300ppp.

4.5.2 Clasificación de las placas por su condición

Las 100 placas han sido clasificadas de la siguiente manera:

a) Buenas

- El ángulo de inclinación de los caracteres de la placa se encuentra en un rango permitido (+/-12 grados).
- Los caracteres se encuentran totalmente formados, es decir no existen caracteres cortados o incompletos.
- La distorsión de la imagen es despreciable.
- La altura de los caracteres se encuentra entre los 30 y 60px.

Figura 4.4.- Ejemplos de placas con clasificación “Buena”

b) Parcialmente Buenas

- El ángulo de inclinación de los caracteres de la placa se encuentra entre los valores máximos permitidos (+/- 14,15 grados).
- Las imágenes se encuentran levemente distorsionadas o pixeladas debido a su tamaño
- Presentan pequeños cortes o rupturas
- Ciertos caracteres apenas se puede apreciar la silueta, para casos como el 8, que pueden perder su curvatura y tener una tendencia en su silueta parecida a la de un 0
- La altura de los caracteres se encuentra entre los 25 y 60px

Figura 4.5.- Ejemplos de placas con clasificación “Parcialmente Buena”

c) Malas

- Aquellas cuyo ángulo de inclinación es superior al máximo permitido.
- Presentan grandes cortes o se encuentran incompletas
- Aquellas que se encuentran cubiertas por otros objetos (parachoques, tornillos)
- Deterioradas por el tiempo o choques
- La altura de los caracteres es inferior a los 20px.

Figura 4.6.- Ejemplos de placas con clasificación “Mala”

4.6 Detección de errores críticos

4.6.1 Error en el análisis de la página

El primer error durante la primera prueba del Tesseract V3 con un entrenamiento personalizado fue obtener como resultado nulo es decir una cadena vacía.

La imagen que se quería reconocer correspondía al primer carácter de la matrícula.

Figura 4.7.- Imagen del primer carácter

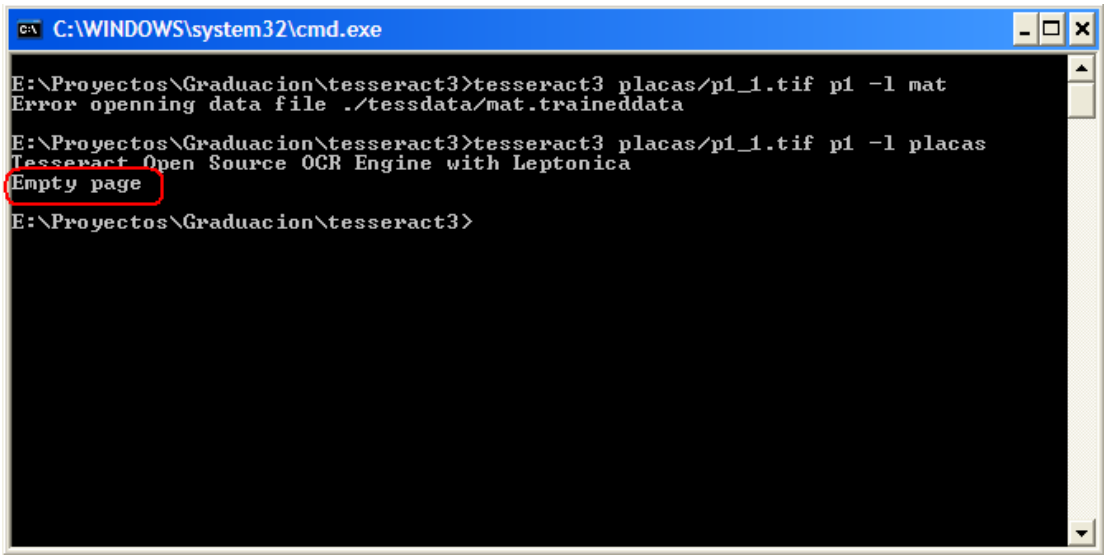


Figura 4.8.- Detección de errores en pruebas

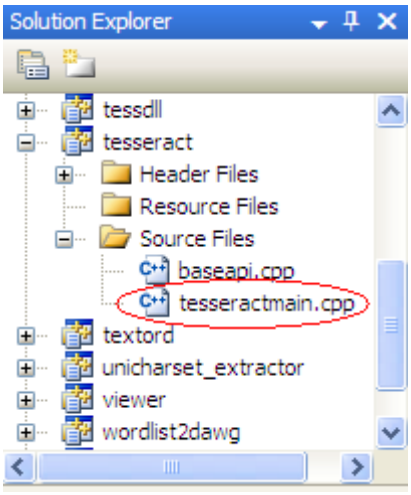
Causa del error

Tesseract tiene por defecto una etapa que consiste en la detección de segmentación de la página o imagen de entrada, así es capaz de detectar si la imagen consiste en una línea, una sola palabra, columnas de texto o simples caracteres. Así se determinó que para las imágenes segmentadas que utilizábamos no lo estaba reconociendo como un solo carácter sino más bien como un bloque o línea de texto y al no encontrar patrones en la imagen daba como resultado vacío.

Solución

Se procedió a modificar el código fuente del OCR Tesseract para que su reconocimiento sea puntual, estrictamente para reconocer caracteres sueltos.

De esta forma se estableció en la línea 223 del archivo fuente tesseractmain.cpp del proyecto Tesseract, el tipo de segmentación de la página que utilizaría, como se muestra en la Figura 4.9




```
const char* lang = "eng";
int arg = 3;
if (argc >= 5 && strcmp(argv[3], "-l") == 0) {
  lang = argv[4];
  arg = 5;
}

tesseract::TessBaseAPI  api;

api.SetOutputName(argv[2]);
api.Init(argv[0], lang, &(argv[arg]), argc-arg, false);
api.SetPageSegMode(tesseract::PSM_SINGLE_CHAR);
```

Figura 4.9.- Corrección en código fuente

El resultado de este cambio es claramente apreciado en la Figura 4.10, donde se testeo el primer carácter de la placa #1.

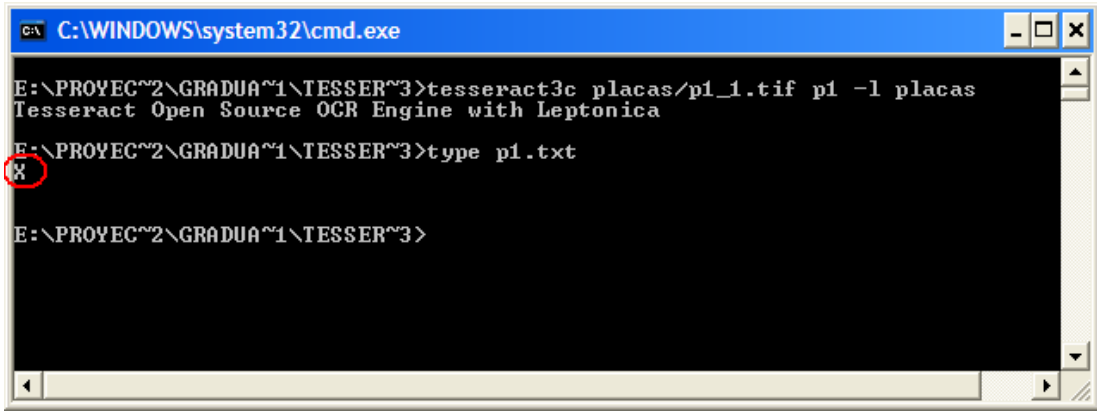


Figura 4.10.- Prueba para el reconocimiento del primer carácter

En reconocimientos masivos los tiempos de respuesta mejoran debido a que evitamos que el OCR trate de identificar que sea un bloque, una palabra o columnas de texto.

4.6.2 Distinción entre caracteres alfabéticos y numéricos

Se detectó que muchos caracteres alfabéticos son fácilmente identificados como numéricos, debido a varios factores como distorsión de la placa, mala segmentación, demasiado sombreo, caracteres difusos. A continuación se muestran los diferentes casos:

- La letra “B” fácilmente era reconocida como el número “8”.
- El cero “0” fácilmente era reconocido como la letra “O”, “Q” o “D”, incluso hasta con la “U”.
- El seis “6” fácilmente era reconocido como la letra “G”
- El número “1” fácilmente era reconocido con la “I”
- El “4” hallaba similitud con la letra “A”

- El número “5” hallaba similitud con la letra “S”
- El número “2” hallaba similitud con la letra “Z”.
- El número “7” hallaba similitud con la letra “T” o viceversa.

Causa

Este inconveniente se presenta debido a que los patrones de ciertos números son similares a los patrones de los caracteres alfabéticos (por ejemplo los patrones del número “2” con la letra “Z”) y es notable cuando los caracteres a reconocer no se encuentran bien definidos y geométricamente corregidos. Este caso en particular se aprecia notablemente al reconocer una placa debido a que esta se puede encontrar en diferentes condiciones (manchada, borrosa, rota, en pésimo estado, etc)

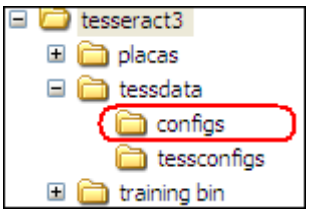
Soluciones

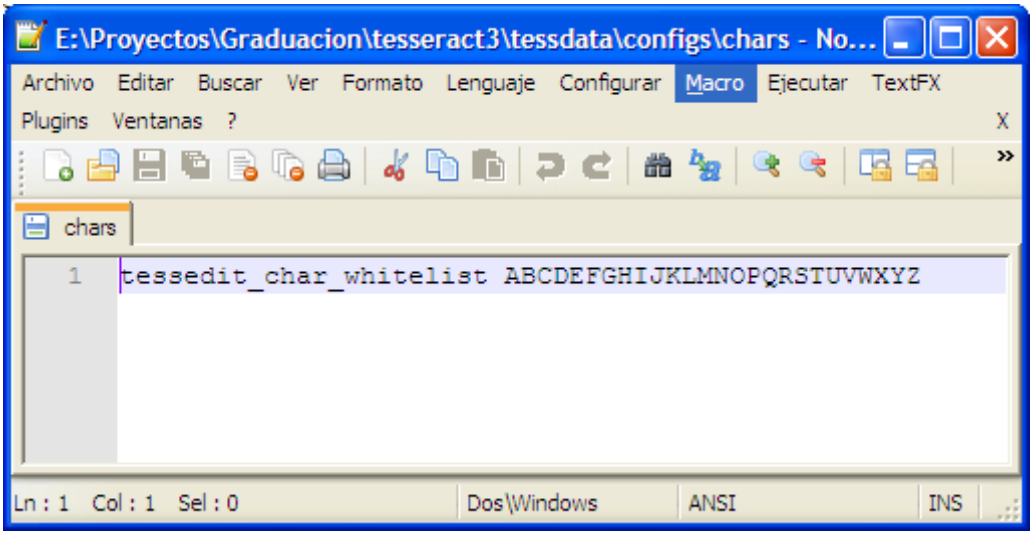
En un principio se planteo como solución crear 2 librerías de entrenamiento, una para carácter alfabéticos y otra por separado para caracteres numéricos pero esto implicaría realizar un trabajo innecesario y fue descartado por no ser óptimo.

Mediante revisión del OCR, como segunda solución se optó por crear un archivo de variables de configuración para especificar lo que se desea reconocer, sea números o letras.

Como ventaja del uso de un archivo personalizado de configuración se determinó que se puede omitir la creación de varios entrenamientos, solo indicando mediante línea de comando la configuración que se desea utilizar.

En la ruta *tessdata/config* se ubican los dos archivos de configuración “chars” y “nums”, como se muestra en la Figura 4.11.





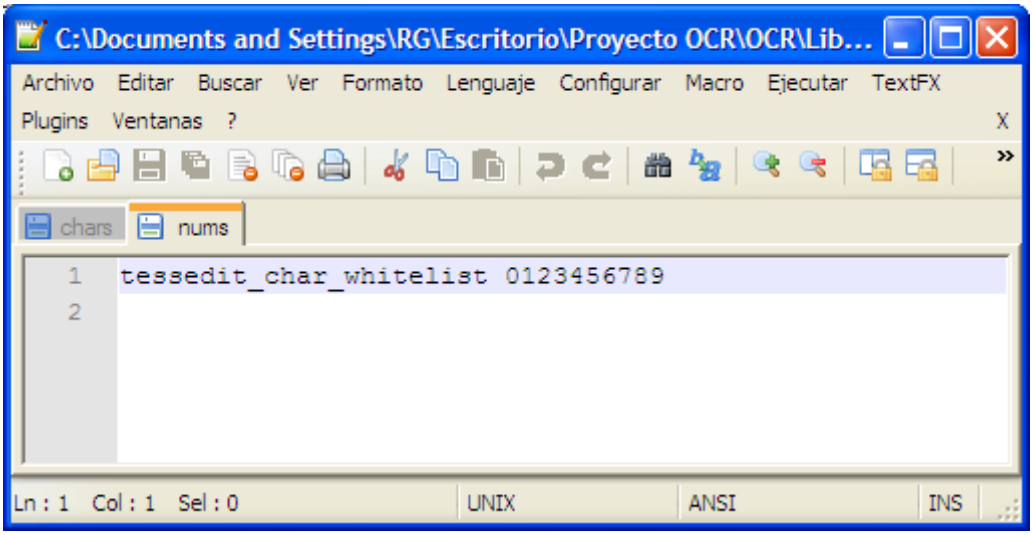


Figura 4.11.- Corrección en código fuente utilizando variables de configuración

Así, cuando se indique mediante comando el archivo de configuración “chars” solo tomará en cuenta los caracteres alfabéticos para el reconocimiento, de esta forma si existe similitud entre la “B” y el “8” optará por devolver la letra “B” ya que el carácter “8” no se encuentra en el archivo “chars”.

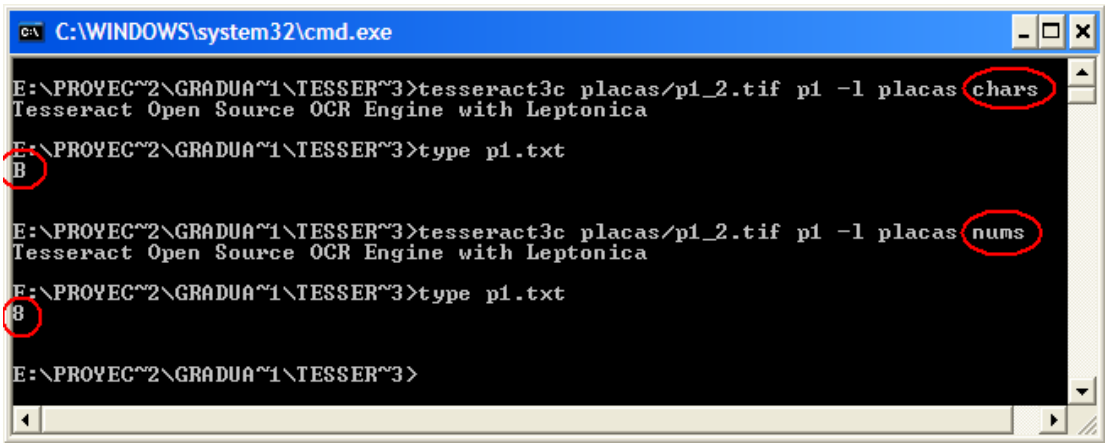


Figura 4.12.- Prueba sobre la corrección utilizando variables de configuración

Comparación de resultados

En la Tabla 4.2, se muestra como la placa es reconocida si no se utiliza variables de configuración en la columna “Todos los caracteres” y utilizando variables de configuración sobre la columna “Clasificación Letras & Números”. Un claro ejemplo de confusión entre caracteres alfabéticos y numéricos se lo aprecia en la Placa #2 sobre dicha tabla, en la cual Tesseract confunde la letra “Z” por el número “2”, error que fue corregido con las variables de configuración.

#	Original	Todos los caracteres	Resultado	Clasificación Letras&Números	Resultado
1	XBS609	XBS609	OK	XBS609	OK
2	GPP132	GPP13Z	-	GPP132	OK
3	GCV274	GCVZ7A	-	GCV274	OK
4	GCZ339	GCZ339	OK	GCZ339	OK
5	GKL358	GKL358	OK	GKL358	OK
6	GJK090	GJK090	OK	GJK090	OK
7	GMX672	GMX67Z	-	GMX672	OK
8	GKS949	GKS949	OK	GKS949	OK
9	GMF906	8-WF9Q6	-	BKF906	-
10	GEP797	GEP797	OK	GEP797	OK
11	GPR741	GPR7A1	-	GPR741	OK

12	GNU670	GNU67O	-	GNU670	OK
13	GMI288	GM1Z88	-	GMI288	OK
14	GMU540	GMU54O	-	GMU540	OK
15	GKC709	GKCTQ9	-	GKC709	OK
16	PBA5331	PBA5331	OK	PBA5331	OK
17	HCE924	HCE924	OK	HCE924	OK
18	GNY649	GHY6A9	-	GHY649	-
19	GMZ473	GMZA73	-	GMZ473	OK
20	GLR877	GLR877	OK	GLR877	OK
21	PBI760	PB-760	-	PB-760	-
22	GQI092	GQ2Q92	-	GQE092	-
23	PJM102	P-1M1O2	-	PUTM102	-
24	GHN581	GHN58-1	-	GHN581	OK
25	GQB649	GQB649	OK	GQB649	OK
26	GLP783	GLP783	OK	GLP783	OK
27	GOT235	G0TZ35	-	GOT235	OK
28	GRX6432	GH-K6-S3-2	-	GHW6432	-
29	GFM731	GFM731	OK	GFM731	OK
30	GMB082	GHBOB4	-	GHB084	-
31	GJV355	GJV355	OK	GJV355	OK

32	GGV633	GGW633	-	GGW633	-
33	POP727	E0P7Z7	-	EOP727	-
34	GQB078	GQB078	OK	GQB078	OK
35	GRX6432	GRX6432	OK	GRX6432	OK
36	GNB107	GHBKSST	-	GHB301	-
37	GQX465	GQX465	OK	GQX465	OK
38	GQD281	GQDZ8-	-	GQD289	-
39	GQD029	GQDOZ9	-	GQD029	OK
40	MGD065	MGOQ65	-	MGO065	-
41	GPF599	6PF5Q9	-	GPF599	OK
42	GLV798	GLV798	OK	GLV798	OK
43	GOT023	G0TOZ3	-	GOT023	OK
44	GOW935	GOW935	OK	GOW935	OK
45	GHS290	GHS29O	-	GHS290	OK
46	GOZ877	G0Z877	-	GOZ877	OK
47	GOE797	G0E797	-	GOE797	OK
48	GMB265	GHBZ65	-	GHB265	-
49	TCJ443	TCJA83	-	TCJ483	-
50	GPV734	GPV734	OK	GPV734	OK
51	G-P0810	G-PQ81O	-	G-P0810	OK
52	GOJ277	G0JZ77	-	GOJ277	OK

53	GHR567	GHR567	OK	GHR567	OK
54	GNR693	GHR693	-	GHR693	-
55	GCK974	GCK974	OK	GCK974	OK
56	RCO861	RC0B6-I	-	RCO861	OK
57	GLP378	GAP378	-	GAP378	-
58	GRL819	GRL819	OK	GRL819	OK
59	GRJ420	GRJ4Z0	-	GRJ420	OK
60	GJC181	GJC-I81	-	GJC181	OK
61	GKY692	GYXY6SZ	-	GYXY692	-
62	GPC947	GPC947	OK	GPC947	OK
63	GOC926	GOC9Z5	-	GOC925	-
64	GQV511	GQV5SN	-	GQV583	-
65	GLA901	GLK9QX	-	GLK961	-
66	GNL599	GNL599	OK	GNL599	OK
67	PDK592	PDK59Z	-	PDK592	OK
68	GOH697	G0H697	-	GOH697	OK
69	GPX590	GPX590	OK	GPX590	OK
70	GPS780	GR-ST8Q	-	GPS780	OK
71	MDF191	MDF19-I	-	MDF191	OK
72	GPB332	GEB337	-	GEB337	-
73	GMG103	GMG103	OK	GMG103	OK

74	GPM333	GPH333	-	GPH333	-
75	GLU216	GLUZ-I6	-	GLU216	OK
76	GQQ589	GQQ5B9	-	GQQ589	OK
77	GPH575	GPH575	OK	GPH575	OK
78	GDC055	S-DCQ55	-	GDC055	OK
79	GPW145	GPH345	-	GPH345	-
80	GPV473	GPVA73	-	GPV473	OK
81	GMF382	G-F3B2	-	G-F382	-
82	GOU938	GOU93B	-	GOU938	OK
83	GOZ324	G0Z3Z4	-	GQZ324	-
84	BFC335	6FC335	-	BFC335	OK
85	GHH534	G--CH-- E3-I	-	GBH534	-
86	GMA077	GMAO77	-	GMA077	OK
87	PVV444	PVV4AA	-	PVV444	OK
88	PLV018	PLV0-IB	-	PLV018	OK
89	PYE694	PH-E59-	-	PYE598	-
90	GOC926	GOC9Z6	-	GOC926	OK
91	GIP0810	G-PD8-I0	-	G-P0810	-
92	PFB421	PFB421	OK	PFB421	OK
93	GNB107	GWBNQ-1	-	GWB107	-

94	GOD737	G00737	-	GOO737	-
95	GPG514	GPG5-I4	-	GPG514	OK
96	GQV511	GQW5S-W	-	GQW533	-
97	GND758	G-D75-	-	G-D75-	-
98	GOD524	G0O57A	-	GOO574	-
99	GNA179	G-A179	-	G-A179	-
100	GRW3568	GRW3-568	-	GRW3568	OK

Tabla 4.2.- Resultados obtenidos con y sin variables de configuración

En esta etapa nos encontramos ya en un avance significativo del proyecto en cuanto a mejora en la tasa de reconocimiento, tal como se muestra en la tabla 4.3. La tasa de reconocimiento mejoró de un 28% a un 66%, estableciendo una clasificación de tipos de caracteres a reconocer.

Placas	Tasa de Reconocimiento	% de Rec. Con Clasificación
100 Placas (606 Imágenes)	28%	66%

Tabla 4.3.- Cuadro comparativo de tasa de reconocimiento

4.7 Análisis y corrección de errores de caracteres

4.7.1 Similitud de caracteres

Ahora se analizará los diferentes problemas de reconocimiento de caracteres encontrados, así como la corrección respectiva.

Mediante observación de resultados, se notó que el OCR tenía inconvenientes para distinguir las letras "D", "O", "Q", "U" ya que poseen patrones similares.

De esta forma se partió con el análisis de aquellos caracteres que presentaban problemas de reconocimiento por diferentes motivos como inclinación, sombreado, o que se encuentran sesgadas.

En la Figura 4.13 se muestra la "Q", "D" y "U" originales así como el resto de caracteres alfabéticos y numéricos.

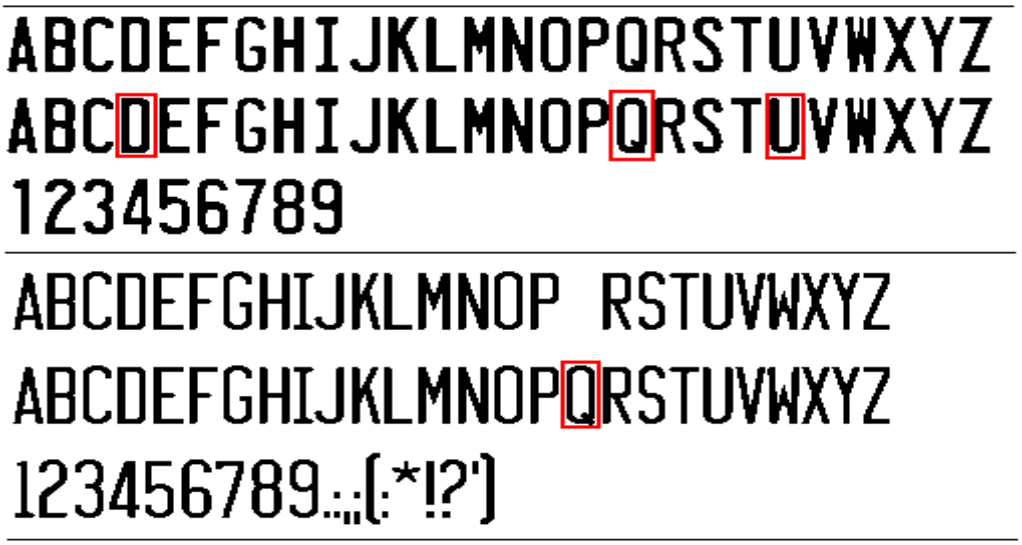
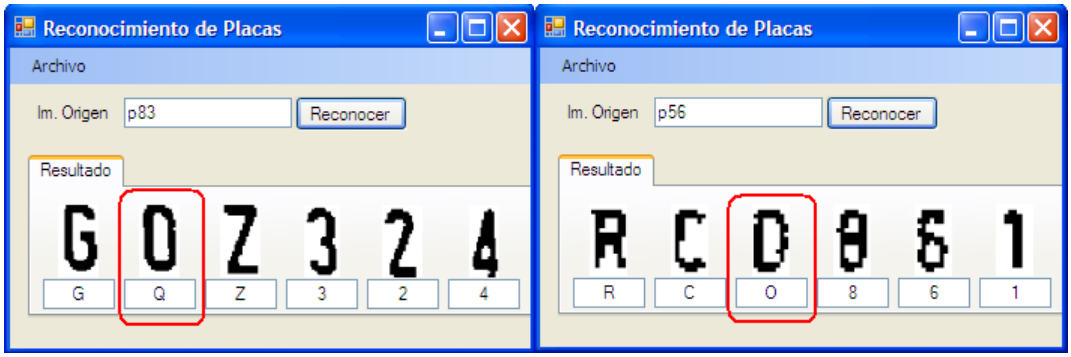


Figura 4.13.- Plantilla

Ya con mayor detalle observamos una diferencia significativa en la letra “Q” que es utilizada en las placas, como se muestra en la Figura 4.14.

Esta pequeña raya inclinada que caracteriza la letra “Q” ubicada en la esquina inferior derecha difiere de la “Q” utilizada en la placa. De igual forma ocurre con la letra “D”, “O” y “U”, como se lo observa en las siguientes pruebas realizadas que se muestran en la Figura 4.15.

Figura 4.14.- Plantilla “Q”, “D”, “U”, “O” originales



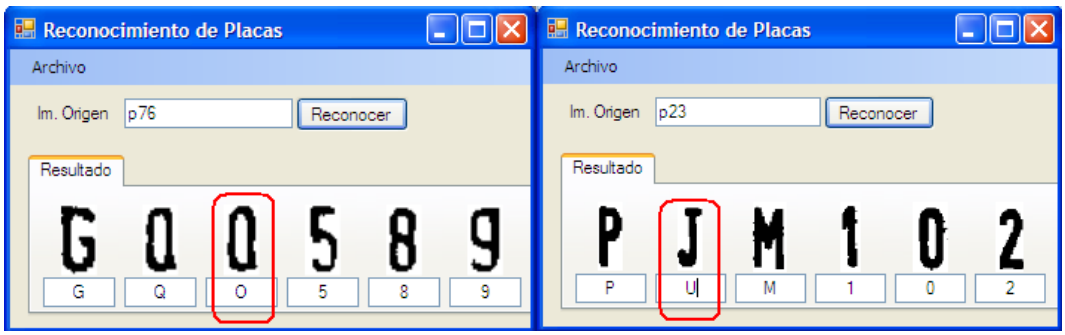


Figura 4.15.- Prueba realizadas con los caracteres "Q", "D", "U", "O"

Teniendo en claro la causa, se creó la imagen de entrenamiento para las letras "Q", "U", "D", "O" con diferentes alturas, corrigiendo la letra "Q" y bordes de la "D"; así se crearon las imágenes de entrenamiento: tr_q_01.tif, tr_q_02.tif, tr_q_03.tif y tr_q_04.tif que se muestran en la Figura 4.16.

Como altura mínima se estableció en 30 píxeles debido a que cuando son menores a dicho tamaño, las letras "D" y "O" dan resultado incierto puesto que la letra "D" tiende a hacerse "O" mientras su tamaño disminuye o si hay una mínima variación en forma de la letra "O" la puede convertir en "D".

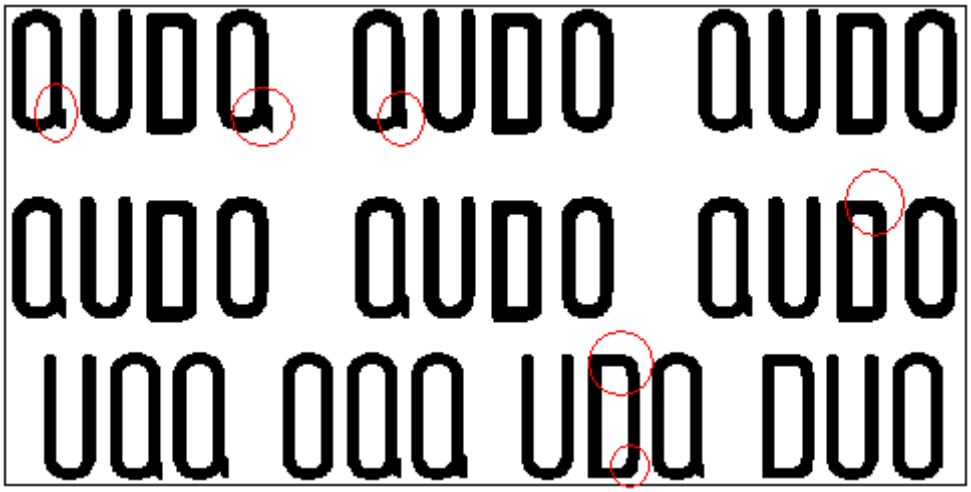


Figura 4.16.- Entrenamiento para los caracteres “Q”, “D”, “U”, “O”

Mediante una serie de pruebas que se hicieron, se detectaron muchos caracteres que erróneamente son reconocidos por tener patrones similares.

De esta manera se procedió con la creación de la imagen de entrenamiento *TR_ABC_CORR_1* con 644 caracteres en los cuales se corrigió lo siguiente:

- Carácter “D”: Con la corrección de sus esquinas.
- Carácter “W”: se hizo la corrección en el centro del carácter.
- Carácter “Q”: se realizó pequeñas variaciones pero muy significativas.

Figura 4.17.- Entrenamiento del carácter “D”

Figura 4.18.- Entrenamiento del carácter “W”

Figura 4.19.- Entrenamiento del carácter “Q”

Para los caracteres numéricos se crearon las imágenes *TR_NUM_CORR1, TR_NUM_CORR2* con 150 Y 141 caracteres numéricos respectivamente, cuyas correcciones fueron las siguientes:

- Caracter “1”: Se aumento el sombreado de dicho carácter

Figura 4.20.- Entrenamiento del carácter “1”

- Caracteres “1” y “7”: Debido a la similitud de dichos caracteres, se procedió a juntarlos y mezclarlos para obtener dichos errores y corregirlos, de ahí generar patrones para mejorar los resultados.

Figura 4.21.- Entrenamiento del carácter “1”, “7”

- Caracteres “5” y “6”: Dichos caracteres poseen patrones similares, así con una pequeña distorsión o ruido el carácter “5” tiende a un “6”, de ahí se procedió a aumentar el número de patrones y correcciones juntando ambos caracteres.

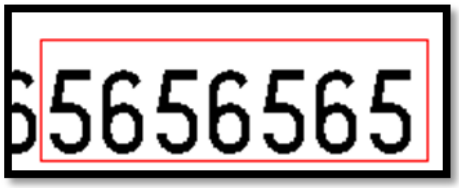


Figura 4.22.- Entrenamiento del carácter “5”, “6”

- Carácter “8”: A este carácter se le aumento el sombreado puesto que era muy común tener este problema al momento de segmentarla.

Figura 4.23.- Entrenamiento del carácter “8”

- Caracteres “1" y “8”: Se realizó otra corrección debido a que la imagen de entrada cuando es considerada pequeña o se encuentra en mal estado tenemos como resultado el carácter “1” con ciertas

características que se muestran en las Figuras 4.24 y 4.25. Lo mismo ocurre con el carácter “8”, como se muestra en la Figura 4.26.

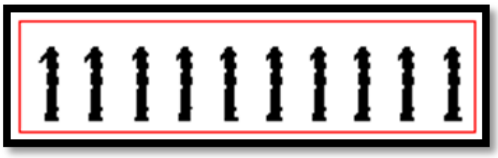


Figura 4.24.- Entrenamiento del carácter “1”

Figura 4.25.- Entrenamiento del carácter “1”

Figura 4.26.- Entrenamiento del carácter “8”

- Carácter “0”: Al igual que el caracter “1”, se aumentó el sombreado para este caracter con el objetivo de facilitar el reconocimiento de aquellas imágenes con esta similitud, además notamos que al aumentar su sombreado no causa que sea reconocido como otro carácter.

Figura 4.27.- Entrenamiento del carácter “0”

- Carácter “4”: Algo muy común es tener este carácter con aquella sombra que se encuentra señalada en la Figura 4.28, por esta razón se adicionó este error al entrenamiento.

Figura 4.28.- Entrenamiento del caracter “4”

- Carácter “1”: Por mala segmentación o adquisición se obtiene el caracter “1” como se muestra a continuación en la Figura 4.29, en esta corrección se procura dejar la parte señalada intacta para que los patrones que sean obtenidos se basen principalmente en la cabeza del carácter “1”.

Figura 4.29.- Entrenamiento del caracter “1”

- Caracter “2”: Debido a sombras, pérdida de color en placas se obtiene un ‘2’ como se muestra en la Figura 4.30. Dicho “2” al encontrarse cortado en su terminal tiende a reconocerse como un ‘7’.

Figura 4.30.- Entrenamiento del carácter “2”

4.7.2 Patrones para caracteres “especiales”

Se crearon las siguientes imágenes de entrenamiento: *TR_NUM_CORR3* con 240 caracteres numéricos, cuyo objetivo fue aumentar el número de patrones por carácter, *TR_ABC_CORR_ERROR1* con 317 caracteres. Se detectó un problema al reconocer la letra “X” pues con cierta variación en su forma, suele dar como resultado la letra “K”, o bien 2 letras “KY”.

En base a las correcciones realizadas anteriormente, se solucionó colocando en serie dichos caracteres, para que al momento de reconocerlos se indique en el entrenamiento que corresponden a caracteres diferentes.

Figura 4.31.- Entrenamiento de caracteres “KYX”

De forma similar se observó que al existir pocos patrones para las letras “S” y “J”, “H” y “M”, “P”,”R” y “B” estas tendían a dar malos resultados.

Otro caso fue el de la letra “N” y “W”, “G” y “B” pues durante las pruebas daban resultados erróneos, los cuales se corrigieron obteniendo patrones a partir de 12 caracteres de entrenamiento.

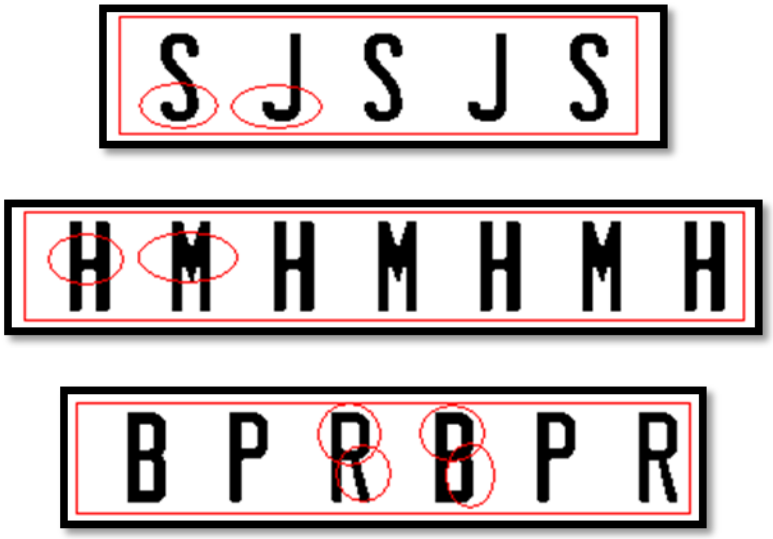


Figura 4.32.- Entrenamiento de caracteres “SJ”, “HM”, “PRB”,

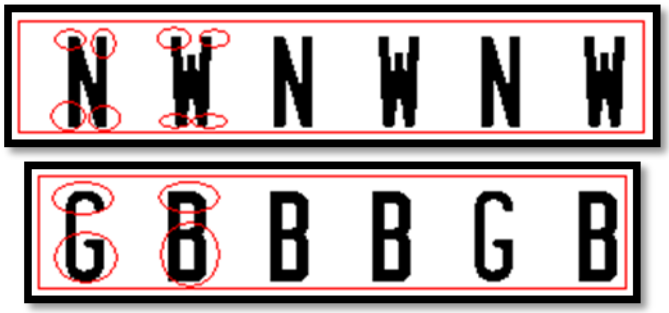


Figura 4.33.- Entrenamiento de caracteres “NW”, “GB”

4.7.3 Caracteres sesgados

Puesto que las tomas realizadas no fueron perpendiculares a la placa sino más bien a cierto ángulo, fue muy común encontrarse con placas a cierta inclinación o sesgo debido a la perspectiva. Dicho sesgo causaba que fácilmente una H sea confundida por una K o la letra M por la letra H, de igual forma con los caracteres numéricos, por ejemplo un 5 por un 6.

Con el problema que se detectó producido por el sesgo, se crearon las imágenes de entrenamiento:

TR_ABC_CORR2 con 667 de caracteres y una inclinación de 5 grados a la izquierda.

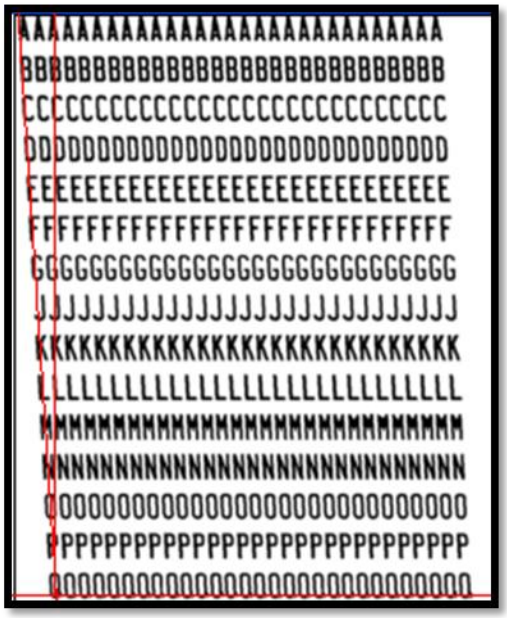


Figura 4.34.- Entrenamiento de caracteres con sesgo a la izquierda

Cabe mencionar que se realizó una corrección a la letra con respecto al tamaño del agujero que hay en el centro del carácter.

Figura 4.35.- Entrenamiento del carácter “A”

De manera similar se hizo una corrección en ciertos puntos al carácter “W”.

Figura 4.36.- Entrenamiento del carácter “W”

TR_ABC_CORR3 con 667 caracteres y un ángulo de inclinación de 5 grados a la derecha.

Figura 4.37.- Entrenamiento de caracteres con sesgo a la derecha

Se realizaron las mismas correcciones de la imagen *TR_ABC_CORR2.*

TR_ABC_CORR_ERROR2 con 317 caracteres, se basa en la imagen de entrenamiento “TR_ABC_CORR_ERROR1” y sus mismas correcciones, ahora con una inclinación o sesgo de 12 grados a la derecha.

TR_ABC_CORR_ERROR3 con 317 caracteres se basa en la imagen de entrenamiento “TR_ABC_CORR_ERROR1” y sus mismas correcciones, ahora con una inclinación o sesgo de 12 grados a la izquierda.

Figura 4.38.- Entrenamiento de caracteres con sesgo a la derecha

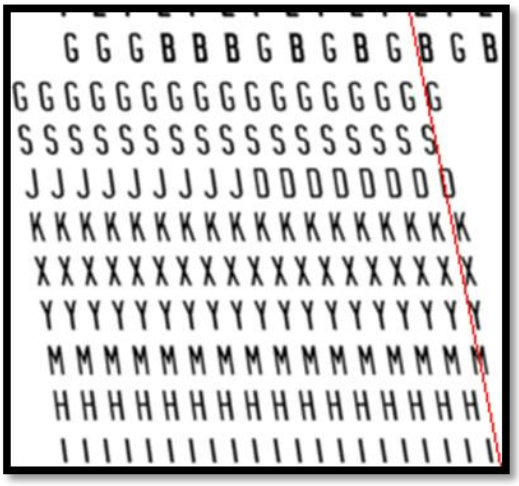


Figura 4.39.- Entrenamiento de caracteres con sesgo a la izquierda

TR_ABC_CORR_ERROR4F, con 614 caracteres de entrenamiento, es la agrupación de una serie de correcciones hechas en el transcurso de las pruebas, causadas por el entrenamiento con inclinación.

Por esta razón se muestran caracteres tanto con inclinación como sin ella para ciertos caracteres. Un caso especial fue que la letra “G” sesgada era reconocida como una “B”.

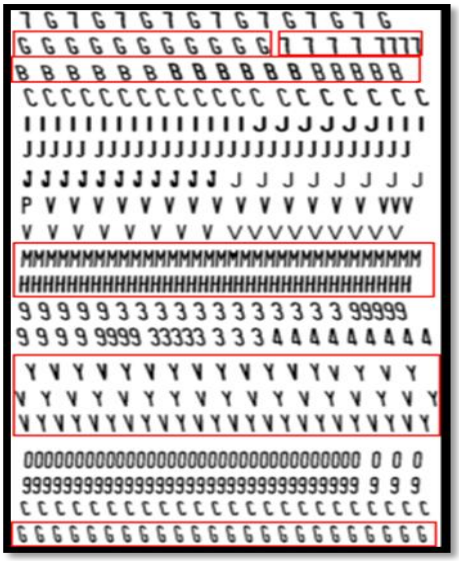


Figura 4.40.- Entrenamiento de caracteres con sesgo y sin sesgo

TR_NUM_DG_SESGO15DE, con 260 caracteres para entrenamiento, se adicionan estos patrones para solventar posibles errores de reconocimiento a causa de caracteres numéricos inclinados hacia la derecha hasta 15 grados.

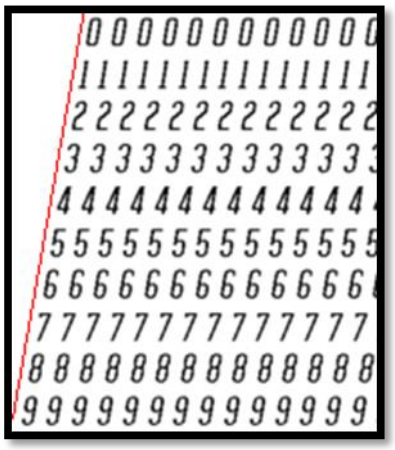
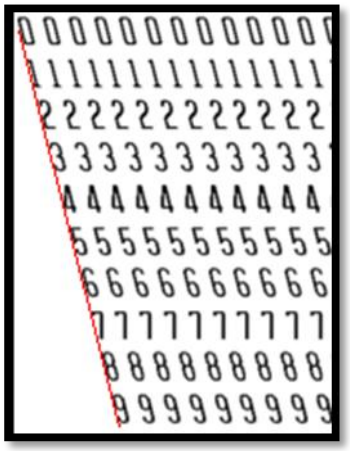


Figura 4.41.- Entrenamiento de caracteres con sesgo a la derecha

TR_NUM_DG_SESGO15IZQ, con 260 caracteres para entrenamiento, se adicionan estos patrones para solventar posibles errores de reconocimiento a causa de caracteres numéricos inclinados hacia la izquierda hasta 15 grados.



.

Figura 4.42.- Entrenamiento de caracteres con sesgo a la izquierda

*TR_NUM_LP_SESGO15DE***,** con 260 caracteres para entrenamiento, se adicionan estos patrones para solventar posibles errores de reconocimiento a causa de caracteres numéricos inclinados hacia la derecha hasta 15 grados, utilizando una fuente distinta (License Plate).

TR_NUM_LP_SESGO15IZQ, con 260 caracteres para entrenamiento, se adicionan estos patrones para solventar posibles errores de reconocimiento a causa de caracteres numéricos inclinados hacia la izquierda hasta 15 grados, utilizando una fuente distinta (License Plate).

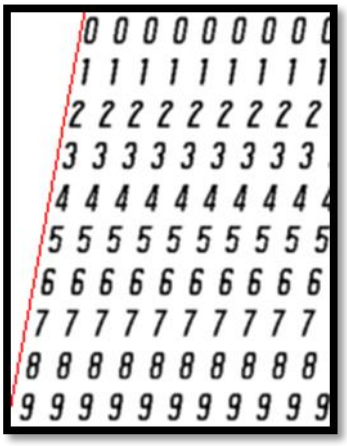


Figura 4.43.- Entrenamiento de caracteres con sesgo a la derecha con fuente License Plate

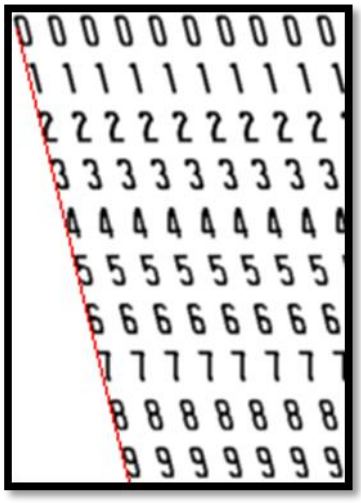


Figura 4.44.- Entrenamiento de caracteres con sesgo a la izquierda con fuente License Plate

4.8 Conclusiones

Al finalizar toda la etapa de entrenamiento de nuestro OCR, se obtuvo la librería de entrenamiento PV8, la cual se realizó con un total de

6.366 patrones; obteniendo finalmente buenos resultados sobre imágenes con una segmentación y corrección limitada.

Algo que podemos resaltar de todo el proceso de entrenamiento, es que no por utilizar mayor cantidad de imágenes de entrenamiento para obtener patrones nos va a garantizar mejores resultados; así se demostró que mientras más entrenábamos un caracter otros con características similares se veían afectados, por ejemplo con los caracteres “M” y “H”.

Es fácil deducir que si el reconocimiento se lo realiza a imágenes con previa corrección geométrica, correctamente segmentadas y libres de cualquier tipo de ruido, los resultados que se obtendrán serán en gran porcentaje mejores que los actuales obtenidos.

CAPITULO 5

CARACTERISTICAS DEL TESSERACT

5.1 RESUMEN

En el presente capítulo se determina la inclinación que tiene la imagen con respecto a la cámara, por lo cual se realizan diversas pruebas para determinar su máxima inclinación.

Adicional se realiza un breve resumen de los diversos ANPR`s comerciales que existen en la actualidad.

5.2 Determinación de la inclinación del carácter

Puesto que la inclinación de la placa durante el proceso de corrección geométrica no pueda ser correctamente mejorada, existe la posibilidad de recibir una placa como se muestra en la Figura 5.1.

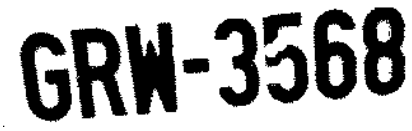


Figura 5.1.- Placa sesgada

Si se reconociera tal imagen como un todo no habría problema alguno en reconocerla ya que el algoritmo establece líneas base para hallar su inclinación, pero como el reconocimiento se lo realiza luego de una etapa de segmentación, las líneas bases quedan obsoletas sobre un solo carácter. En un solo carácter no es posible establecer líneas base ya que no existen referencias con otros caracteres

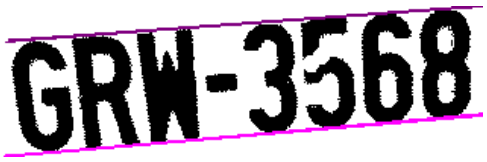


Figura 5.2.- Líneas base sobre toda la placa

Figura 5.3.- Líneas base sobre el carácter

Para intentar solucionar dicho problema se optó por añadir al entrenamiento, imágenes (caracteres numéricos y alfabéticos) con una inclinación de (+/-) 12 grados puesto que superior a esto, el porcentaje de reconocimiento baja considerablemente.

Figura 5.4.- Ejemplo de imágenes de entrenamiento con inclinación

El impacto que tuvo añadir dichas imágenes con los caracteres numéricos y alfabéticos, fue reducir el porcentaje de fiabilidad del reconocimiento, pues se encontraron los errores que se muestran en la Figura 5.5.

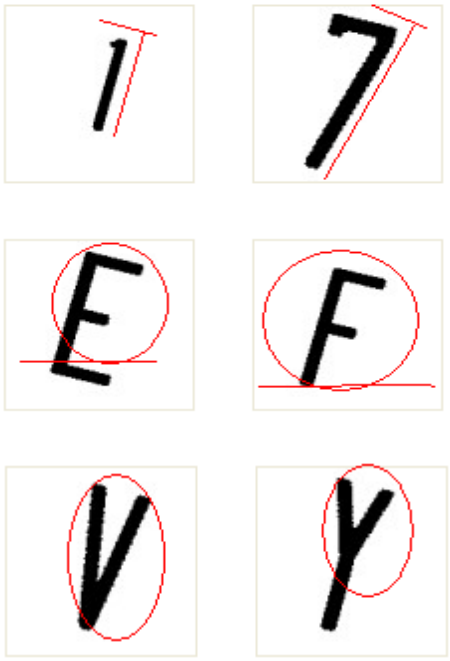


Figura 5.5.- Errores en caracteres sesgados

Al realizar el entrenamiento, caracteres como el “1” inclinado tomaba gran similitud con el “7” cuya característica principal que lo diferencia con el “1” es su inclinación.

Caracteres como la “F”, “E”, “V”, “Y” fueron afectados causando inconsistencia en los resultados. Como alternativa se crearon imágenes con gran cantidad de caracteres con un sesgo de 5 a 12 grados tanto para izquierda como para la derecha.

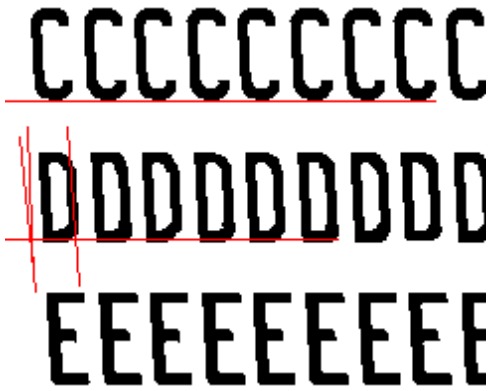


Figura 5.6.- Entrenamiento de caracteres sesgados

De esta manera se corrigió el problema anterior ya que al momento de obtener los patrones de cada caracter, éste ya posee información sobre su inclinación (basándose en las líneas de base).

Luego de las pruebas mencionadas se establece la inclinación aceptable en +/- 12 grados.

Se establece la inclinación máxima en +/- 15 grados para casos extremos.

5.3 Determinación del ángulo de rotación vertical y horizontal de la placa

El objetivo de estas pruebas es establecer el ángulo máximo que puede tener una placa en relación a la cámara.

Para ello se seleccionó una imagen en condiciones ideales, estas condiciones están dadas por la siguiente tabla 5.1 obtenida en las pruebas de campo.

	Datos 1	**Datos 2**	**Datos 3**	**Datos 4**	**Datos 5**	**Datos 6**	**Datos 7**	**Datos 8**
Ra	3,2	4,69	4,9	5,32	5,63	6,34	7,3	8,77
H	1,5	1,5	1,5	1,5	1,5	1,5	1,5	1,5
Rb	2,83	4,44	4,66	5,10	5,43	6,16	7,14	8,64
C	2,00	2,00	2,00	2,00	2,00	2,00	2,00	2,00
D	2,00	3,97	4,21	4,70	5,04	5,83	6,86	8,41
B	45,04	26,75	25,39	23,07	21,63	18,95	16,26	13,38
A	27,95	18,65	17,83	16,38	15,45	13,69	11,86	9,85
Placa (px)	134x126	115x63	115x59	104x55	99x50	90x47	79x39	70x32
Altura Caracter (px)	41,0	34,0	32,0	30,0	28,0	25,0	22,0	18,0

Tabla 5.1.- Parámetros según condiciones ideales

Figura 5.7.- Placa de prueba

Placa: GMX672 % de reconocimiento: 100%

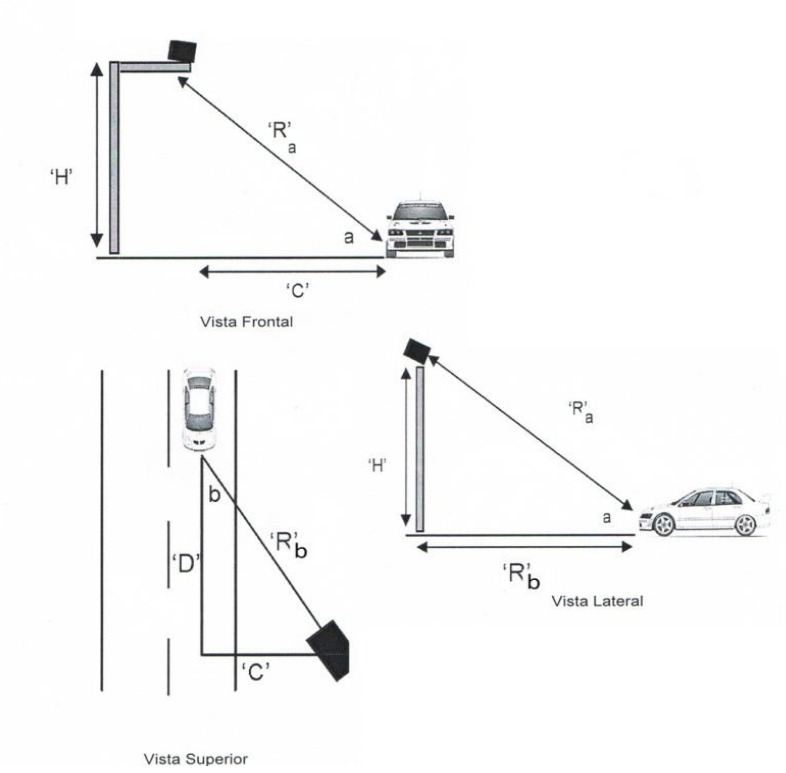
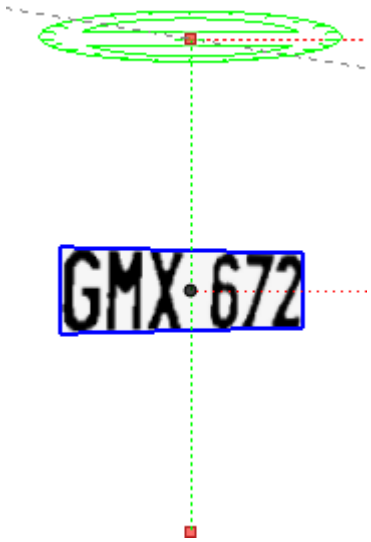
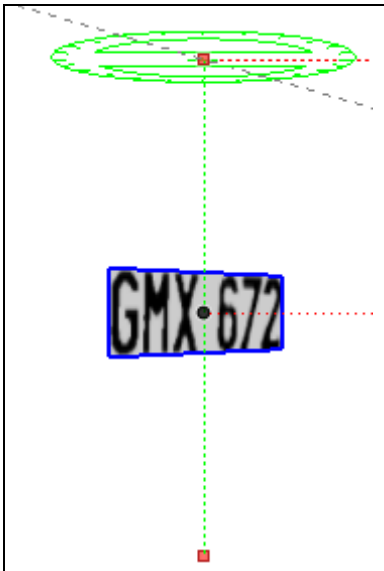
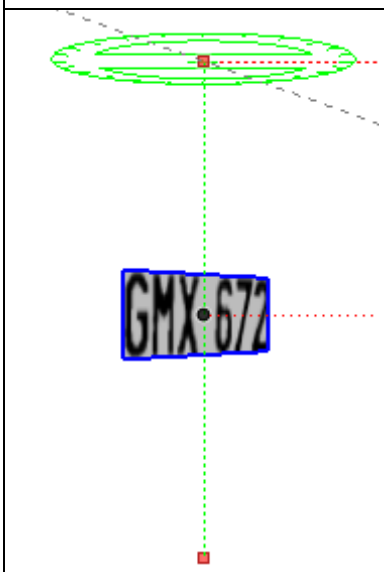


Figura 5.8.- Angulo de inclinación de cámara

Rotación horizontal de la placa

Se rotó la imagen a diferentes ángulos (de derecha hacia la izquierda), obteniendo los resultados según la Tabla 5.2.

	**45 Grados** **Corrección geométrica:** Ninguna **Resultado:** GMX 672 **100%** Caracteres Reconocidos **Imagen Binaria B/N - TIFF Sin compresión:** 
	**55 Grados** **Corrección geométrica:** Ninguna **Resultado:** GMX 672 **100%** Caracteres Reconocidos **Imagen Binaria B/N - TIFF Sin compresión:** 

	**60 Grados** **Corrección geométrica:** Ninguna **Resultado:** GMX 672 **100%** Caracteres Reconocidos **Imagen Binaria B/N - TIFF Sin compresión:** 
	**65 Grados** **Corrección geométrica:** Ninguna **Resultado:** GMX 672 **100%** Caracteres Reconocidos **Imagen Binaria B/N - TIFF Sin compresión:** 

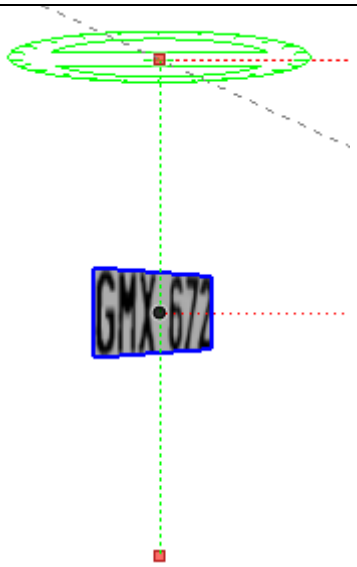
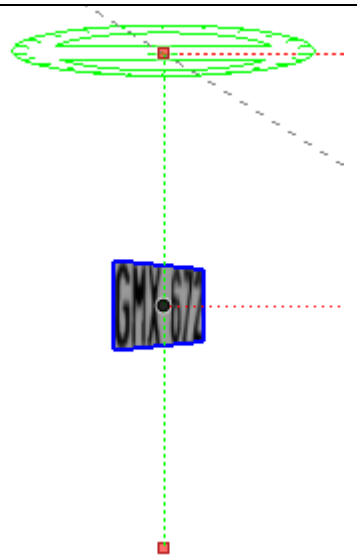
	**70 Grados** **Corrección geométrica:** Ninguna **Resultado:** GMX 672 **100%** Caracteres Reconocidos **Imagen Binaria B/N - TIFF Sin compresión:** 
	**75 Grados** **Corrección geométrica:** Ninguna **Resultado:** GMX 612 **83,33%** Caracteres Reconocidos **Errores:** Mientras el ángulo de rotación horizontal aumenta, los caracteres tienden a tomar similitud con otros como es el caso del 1 y 7, M y H. **Imagen Binaria B/N - TIFF Sin compresión:** 

Tabla 5.2.- Pruebas de rotación en placa sobre el eje vertical

A partir del ángulo 75 se presentan errores de reconocimiento, por lo que se establece como el ángulo limite de rotación horizontal en 70 grados, el cual mediante corrección geométrica mejoraría significativamente el resultado del reconocimiento.

Rotación Vertical

En la tabla 5.3 muestra el detalle de las pruebas realizadas para determinar el ángulo límite de rotación vertical (la placa rota sobre el eje horizontal x).

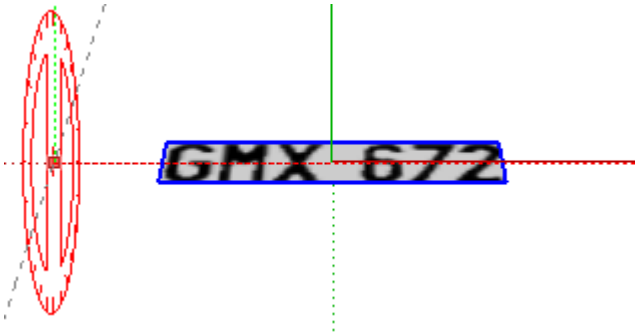
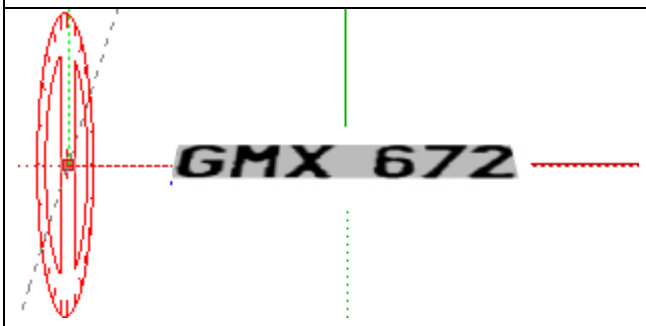
	**45 Grados** **Corrección geométrica:** Ninguna **Resultado:** GMX 672 **100%** Caracteres Reconocidos **Imagen Binaria B/N - TIFF Sin compresión:** GMX 672
	**60 Grados** **Corrección geométrica:** Ninguna **Resultado:** GMX 672 **100%** Caracteres Reconocidos

	Imagen Binaria B/N - TIFF Sin compresión: 
	**65 Grados** **Corrección geométrica:** Ninguna **Resultado:** GMX 672 **100%** Caracteres Reconocidos **Imagen Binaria B/N - TIFF Sin compresión:** 
	**70/75 Grados** **Corrección geométrica:** Ninguna **Resultado:** GIIIIX 672 **83,33%** Caracteres

	Reconocidos **Imagen Binaria B/N - TIFF Sin compresión:** **Errores:** Mientras aumente su rotación vertical, luego de la binarización, los caracteres tienden a verse aplastados causando problemas de reconocimiento y como en el presente ejemplo la “M” la reemplaza por “III” o simplemente no la reconoce. GMX 672 **(70º)** GMX 672 **(75º)**

Tabla 5.3.- Pruebas de rotación en placa sobre el eje horizontal

Se establece como ángulo máximo de rotación vertical en 65º. A partir de esta medida se consideran como no reconocibles.

5.4 Tesseract vs otros ANPR’s

A continuación se muestran diferentes ANPR’s comerciales junto con sus características, y un cuadro comparativo en el que se resumen sus principales ventajas a lado del OCR Tesseract.

CARMEN

El sistema informático equipado con el software “CARMEN® ANPR” para el reconocimiento automático de matrícula es capaz de leer el número de licencia de cualquier vehículo en cualquier parte del mundo con la más alta exactitud en menos de una fracción segundo.

Características

- Tiene una tasa de reconocimiento de 98,5%
- Puede capturar imágenes incluso a velocidades del vehículo de hasta 250km/h

Es importante tener en cuenta que todos estos software “CARMEN” no son programas de aplicación sino SDK.

SIRAM

Con tecnología Neuronal Propia y en Constante Evolución y mejora, tiene las siguientes características:

- Tasa de Fiabilidad del 98% (incluyendo matrículas dañadas, etc).
- Tiempo de Procesado 100 ms.
- Reconoce Matrículas de 2 Líneas.
- Retorna Fiabilidad por Matrícula.
- Retorna Fiabilidad Por Carácter.
- Posibilidad de retornar hasta 8 matrículas en una imagen (OPCIONAL).
- Tolerante a Perspectiva

- Lectura de Buffer, BMP y JPGs.
- Independiente del hardware (Cámaras, capturadoras, etc).
- Integración Inmediata con cámaras IP (AXIS, Mobotix, JVC, Lilin, Vivotek, Sony, Panasonic, etc).
- Rutina de pre-corrección de perspectivas extremas.

E-NETCAMANPR

Es un sistema de reconocimiento de matrículas y gestión de la misma basada en tecnología IP.

Para disponer de un sistema de reconocimiento de matrículas, además de la correspondiente licencia de e-netcamCLIENT, será preciso disponer de una licencia de enetcamANPR por cada vial en el que se encuentre una cámara de reconocimiento.

Se conoce por “Vial” el área de tránsito de vehículos, por lo que en una puerta de entrada y salida, habrá dos viales.

Características

- Puede ser arquitectura local, parcialmente distribuida o distribuida
- La distancia de la cámara con respecto al automóvil debe ser de unos 9mts
- La altura de la cámara del suelo será de unos 50-60 cmts.
- Se intentará reducir siempre la proyección de la matrícula (perspectiva).

- El tamaño mínimo de los caracteres de las matrículas debe ser de 20 píxeles.
- Con el fin de testear si la posición de la cámara en el vial es la correcta existe una funcionalidad en la aplicación que permite controlar este aspecto
- Para casos en los que el vehículo no se detiene, se deberán colocar cámaras especiales. Las cámaras con “Reset Asíncrono” permiten obtener una imagen nítida del vehículo a 20 km/h o a 120 km/h.

SLAM

Comprende de cámaras de altas prestaciones para capturar las imágenes de las matrículas, de captura progresiva para obtener imágenes nítidas en movimiento, de tecnología digital con mayor inmunidad al ruido, con el foco de infrarrojo integrado en la carcasa, grado de protección IP55 para trabajar en el exterior, etc. El sistema está dotado de la más alta tecnología, lo que permite su utilización en vías con vehículos a alta velocidad y en ambos sentidos. Presenta cierta inmunidad frente a variaciones medioambientales como es la luz solar gracias al foco de luz infrarrojo integrado. La luz infrarroja funciona en modo flash, para alargar su vida media, y es totalmente invisible por los conductores.

Características

- Identifica todas las matrículas independientemente de su posición, color, tipo de caracteres y nacionalidad (en el ámbito europeo).
- Funcionamiento diurno y nocturno (Flash de iluminación IR integrado)
- Lectura en vehículos parados o en movimiento.
- Rapidez de reconocimiento (t<0.5 seg).
- Orientación de la cámara: +/- 15º con el plano de la matrícula.
- Rango de Temperatura: -5ºC hasta 45ºC
- Alimentación 90VAC-260VAC.
- Fiabilidad del 95%.
- Sistema modular y escalable.
- Diseño compacto, fácil de instalar

A continuación se muestra en la Tabla 5.4 un gráfico comparativo de los OCR's comerciales antes mencionados.

	% Reconocimiento (Fiabilidad)	Tiempo de Procesado	Corrección de Perspectivas	Tamaño Mínimo de Caracteres
CARMEN	98.5	t < 1s	-	-
SIRAM	98	100ms	SI	-
E-NETCAMANPR	-	-	SI	20

SLAM	95	t < 0.5s	SI	-
TESSERACT V3	98.1 *	(0.2 < t < 0.96)s	SI	15
OCRAD	0 *	0.18	-	-
GOCR	10 *	0.2	-	-

* Obtenido sobre las 100 placas de pruebas.

Tabla 5.4.- Cuadro comparativo de OCR`s Comerciales

CAPITULO 6

PRUEBAS DE CAMPO

6.1 Resumen

En este capítulo se muestran los resultados obtenidos de las pruebas de campo realizadas con apoyo del grupo de adquisición, en el cual se realizó un reconocimiento sobre 24 puntos.

6.2 Prueba de campo

Puntos	24
Tomas	240
Corrección	Alineamiento horizontal

Tabla 6.1.- Datos iniciales para las pruebas de campo

Sobre 24 puntos señalados por el grupo de adquisición en base a parámetros establecidos por ellos, se procedió a realizar el reconocimiento sobre las placas tomadas en cada uno de los puntos.

La región mostrada de color amarillo en la Figura 6.1, encierra los puntos en los cuales se obtuvo un reconocimiento aceptable. Los puntos encerrados en el círculo rojo son aquellas cuyo porcentaje de reconocimiento fue el 100% es decir la placa fue en su totalidad reconocida por el OCR; mientras que los puntos encerrados en circulo de color naranja, su reconocimiento fue parcial, el cual se debe a la condición de la imagen, probablemente no existe una muy buena corrección en general de la placa.

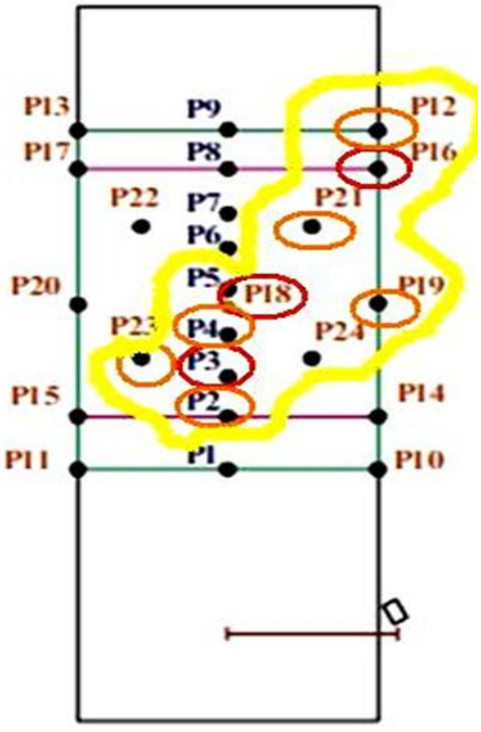


Figura 6.1.- Regiones de reconocimiento

6.3 Resultado

Es importante mencionar que el resultado obtenido en la tabla 6.2 (porcentajes de reconocimiento) varía según el proceso previo al reconocimiento. Para nuestras pruebas se realizó una corrección geométrica manual y una binarización utilizando métodos convencionales; por lo que si se aplicara un buen algoritmo de corrección geométrica y binarización, las placas catalogadas como en mal estado, podrían llegar a ser clasificadas como parcialmente buenas aumentando el porcentaje de reconocimiento.

Punto	Ángulo (b)	% Reconocimiento	Observaciones
P1	45.04	66,67%	
P2	36.25	83,33%	
P3	30.65	100,00%	
P4	26.68	83,33%	
P5	23.68	66,67%	
P6	21.33	0,00%	Placa en mal estado
P7	19.42	0,00%	Placa en mal estado
P8	17.84	0,00%	Placa en mal estado

P9	16.50	0,00%	Placa en mal estado
P10	61.93	66,67%	
P11	7.13	0,00%	Placa en mal estado
P12	29.05	83,33%	
P13	2.12	66,67%	
P14	53.95	33,33%	
P15	5.23	0,00%	Placa en mal estado
P16	31.09	100,00%	
P17	2.3	0,00%	Placa en mal estado
P18	24.06	100,00%	
P19	39.93	83,33%	
P20	3.19	50,00%	
P21	28.26	83,33%	
P22	11.88	50,00%	
P23	17.34	83,33%	
P24	38.59	66,67%	

Tabla 6.2.- Cuadro comparativo de pruebas de campo

CONCLUSIONES

1. En el presente trabajo se ha presentado los diferentes métodos que existen para el reconocimiento de caracteres, los tipos de OCR de libre distribución y ANPR comerciales, los cuales se podrían utilizar para el reconocimiento de placas vehiculares.

2. Se realizó un análisis entre los OCR de código abierto para determinar el más conveniente y el que más facilidades nos brinda para cumplir con nuestro propósito de reconocer con mayor porcentaje de asertividad las placas vehiculares.

3. El OCR seleccionado, Tesseract, cuyo método de reconocimiento es basado en patrones, nos ofrece la posibilidad de entrenarlo y hasta realizar modificaciones comprensibles en el código fuente, modificando parámetros de reconocimiento desde variables de configuración.

4. Respecto al entrenamiento, mediante imágenes muy similares a placas vehiculares y un análisis de dichas placas, como rotación, sesgo y tamaño, se pudo conseguir realizar una librería de entrenamiento con excelentes resultados al momento de reconocer una placa, superando considerablemente al resto de OCR de código abierto.

RECOMENDACIONES

1. Como posibles futuras mejoras es imperativo mencionar que límites de reconocimiento podrían extenderse a otros tipos de placas, por ejemplo de otros países, las cuales su tipografía es variada, única para cada país. Se podría crear un entrenamiento mucho más robusto en base a lo analizado en el presente trabajo.

2. En el presente proyecto se realizó la toma de imagen de manera manual, por lo que se debe considerar como futuro trabajo la unificación de los módulos del seminario, principalmente de Corrección y Segmentación, para que el porcentaje de acierto aumente satisfactoriamente y sea de manera autónoma.

BIBLIOGRAFIA

[1] Wikipedia, “Reconocimiento de Patrones”, http://es.wikipedia.org/wiki/Reconocimiento_de_patrones, 21/12/2007

[2] Wikipedia, “Reconocimiento Automatico de Matriculas”, http://es.wikipedia.org/wiki/Reconocimiento_autom%C3%A1tico_de_matr%C3%ADculas, 16/07/2007

[3] ANPR, “Automatic License Plate Reconigtion, Automatic Number Plate Recognition”, http://www.anpr.net/, 1993

[4] Roadtraffic Techonology, “ANPR and Journey Time System Help Understand Traffic Movements across Northamptonshire”, http://www.roadtraffic-technology.com/contractors/detection/counters/press5.html, 12/02/2010

[5] Wikilingue, “Reconocimiento Óptico de Caracteres”, http://es.wikipedia.org/wiki/Reconocimiento_%C3%B3ptico_de_caracteres, 16/04/2010

[6] Hilera y Martinez, “Redes Neuronales Artificiales”, RA-MA Editorial, 1995

[7] Fogel, David B; "Evolutionary Computation: Toward a New Philosophy of Machine Intelligence", Revista IEEE, 17/04/1997

[8] Manuel Herrán Gascón, “Arena Sensible”, Editorial RED Científica, 01/04/2005

[9] Jose Madrigal Garcia, “Computacion Evolutiva”, http://www.monografias.com/trabajos14/comput-evolutiva/comput-evolutiva.shtml, 2010

[10] Google, “Training Tesseract 3”, http://code.google.com/p/tesseract-ocr/wiki/TrainingTesseract3, 2010

[11] Maps Informática Industrial, “Sistema de Lectura Automática de Matriculas SLAM”, http://www.maps.es/fntesp/Lectura%20Matriculas.htm, 2010

[12] Mike Constant, “An Introduction to ANPR”, http://www.cctv-information.co.uk/i/An_Introduction_to_ANPR, 2010

[13] Jörg Schulenburg, “GOCR/JOCR”, http://jocr.sourceforge.net/developers.html, 29/03/2011

[14] Jörg Schulenburg**,** “GOCR Documentation”, http://aist1.pisem.net/gocr.html, 03/06/2002

[15] CTE, “Reglamento de Tránsito del Guayas”, http://www.cte.gob.ec, 2010

[16] ENRIQUE ALBA, MANUEL LAGUNA Y RAFAEL MARTÍ, “Métodos Evolutivos”, www.uv.es/sestio/TechRep/tr03-04.pdf, 2010