ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

ESCUELA DE INGENIERÍA ELECTRÓNICA EN CONTROL Y REDES INDUSTRIALES

"DISEÑO DE UN SISTEMA EMBEBIDO PARA EL CONTROL DE INGRESO Y SALIDA DE VEHÍCULOS A TRAVÉS DE INTERNET, EN EL ACCESO PRINCIPAL DE LA ESPOCH."

Trabajo de titulación presentado para optar el grado académico de:

INGENIERO EN ELECTRÓNICA EN CONTROL Y REDES INDUSTRIALES

AUTOR: CARLOS ARTURO MANOSALVAS SALAZAR

TUTOR: ING. JAVIER JOSÉ GAVILANES CARRIÓN MSc.

Riobamba – Ecuador

2017

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

ESCUELA DE INGENIERÍA ELECTRÓNICA EN CONTROL Y REDES INDUSTRIALES

El Tribunal del trabajo de titulación certifica que: El trabajo de titulación: **DISEÑO DE UN SISTEMA EMBEBIDO PARA EL CONTROL DE INGRESO Y SALIDA DE VEHÍCULOS A TRAVÉS DE INTERNET, EN EL ACCESO PRINCIPAL DE LA ESPOCH,** de responsabilidad del señor Carlos Arturo Manosalvas Salazar, ha sido minuciosamente revisado por los Miembros del Tribunal del trabajo de titulación quedando autorizada su presentación.

ING. WASHINGTON LUNA E.
DECANO DE LA FACULTAD DE INFORMÁTICA Y ELECTRÓNICA. _____ _____

ING. FREDY CHÁVEZ V.
DIRECTOR DE LA ESCUELA DE INGENIERÍA ELECTRÓNICA, CONTROL Y REDES INDUSTRIALES. _____ _____

MSC. JAVIER GAVILANES C.
DIRECTOR DEL TRABAJO DE TITULACIÓN. _____ _____

ING. ALBERTO ARELLANO A.
MIEMBRO DEL TRIBUNAL. _____ _____

Yo, Carlos Arturo Manosalvas Salazar, declaro ser el autor del presente trabajo de titulación: **"DISEÑO DE UN SISTEMA EMBEBIDO PARA EL CONTROL DE INGRESO Y SALIDA DE VEHÍCULOS A TRAVÉS DE INTERNET, EN EL ACCESO PRINCIPAL DE LA ESPOCH"**, que fue elaborado en su totalidad por mí, bajo la dirección del Ingeniero Javier José Gavilanes Carrión, haciéndome totalmente responsable por las ideas, criterios, doctrinas y resultados expuestos en este Trabajo de Titulación, y el patrimonio intelectual de la misma pertenece a la Escuela Superior Politécnica de Chimborazo.

CARLOS ARTURO MANOSALVAS SALAZAR

DEDICATORIA

A Dios y a toda mi familia, en especial para mis padres Carlos e Inés quienes son autores de todo lo que llegaré a ser en la vida; gracias por ser mi fuerza, mi aliento para no decaer y por brindarme todo su apoyo incondicional e infinito amor. A mis hermanos Soledad, Gabriel y Odalis.

Carlos

TABLA DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE GRÁFICOS

ÍNDICE DE TABLAS

ÍNDICE DE ECUACIONES

ÍNDICE DE ANEXOS

RESUMEN

Se desarrolló un sistema embebido de tiempo real que basa su funcionamiento en un computador de placa reducida Raspberry Pi3 que en conjunto con una cámara ubicada en sitios estratégicos de los brazos mecánicos de la entrada y salida principal de la Escuela Superior Politécnica de Chimborazo, se adquieran imágenes de los vehículos que ingresan y abandonan el campus, a las mismas que a través de la implementación de un algoritmo de segmentación escrito bajo el lenguaje de programación Python en colaboración con las librerías de visión artificial que ofrece OpenCV, se aplican técnicas de procesamiento para que se extraiga desde el escenario la localización del sitio de interés, es decir la placa vehicular. Posterior a ello se emplea un algoritmo de Reconocimiento Óptico de Caracteres (OCR) conocido como Vecinos más Cercanos (KNN), el cual tras una fase de entrenamiento es capaz de reconocer las letras y números de la placa del automóvil, con el fin de que dicha información se almacene en una base de datos al ingreso y elimine de ella al momento de que éste abandone el sitio. Esta información se mantendrá almacenada en una base de datos anclada en la red, para que el propietario pueda conocer la estancia de su vehículo accediendo a los registros de la misma por medio de cualquier dispositivo que tenga acceso a la web. La implementación del sistema de registro de vehículos provee una alternativa para los sistemas de ingreso convencionales ya conocidos y presenta la ventaja de que no requiere el uso de dispositivos o sensores adicionales en el automóvil para su reconocimiento, a más de tener información en tiempo real de todos los datos adquiridos dentro de la web. Este sistema abre la posibilidad de desarrollar nuevas aplicaciones que añadan nuevas funcionalidades acordes al funcionamiento.

Palabras clave: <TECNOLOGÍA Y CIENCIAS DE LA INGENIERÍA>, <INTELIGENCIA ARTIFICIAL>**,** **<**INTERNET DE LAS COSAS (IOT)**>**, <OPENCV>, <PLACA VEHICULAR>, <RECONOCIMIENTO ÓPTICO DE CARACTERES (OCR)>, <SISTEMA EMBEBIDO>, <VISIÓN ARTIFICIAL>

ABSTRACT

The embedded real-time system was developed by using a reduced board computer Raspberry Pi3 which is an equipment with a camera placed in strategic points of the mechanic arms in the main entrance and exit of the Escuela Superior Politécnica de Chimborazo, this equipment captures images of the vehicles that enter and exit the campus and the information is extracted through the implementation of an algorithm of segmentation written in Python programming language and the collaboration of the artificial vision bookstores offered by OpenCV, processing techniques were applied to extract the vehicle plate from the location scenery. Then, an Optical Character Recognition (OCR) algorithm also known as K-Nearest Neighbours (KNN) was applied, which after a training phase identifies letters and numbers in the automobile plates, the information is storage in the entrance data base and deleted when exiting the site. This information is filed in a database anchored in the network, so the vehicle owner could know the time its vehicle stayed on campus by accessing to any devices with web service. The implementation of vehicles registration presents an alternative for the conventional vehicle entry systems and the advantage is that it does not require the use of device or additional sensors in the vehicle to recognize it; besides, it provides information in real time of all the data acquired on the web. This system opens the possibility to develop new applications which work according to the operation.

Key words: <ENGINEERING SCIENCES TECHNOLOGY>, <ARTIFICIAL INTELLIGENCE>, <INTERNET OF THINGS (IOT)>, <OPENCV>, <VEHICLE PLATE>, <OPTICAL CHARACTER RECOGNITION (OCR)>, <EMBEDDED SYSTEM>, <ARTIFICIAL VISION>

INTRODUCCIÓN

Actualmente los sistemas antirrobo de vehículos se han vuelto obsoletos frente a la delincuencia. Sistemas clásicos como el bloqueo de dirección, barra del volante o la alarma acústica suelen ser ineficientes en algunas situaciones. Del mismo modo métodos de rastreo satelital pueden ser vulnerados con un poco de conocimiento del funcionamiento del mismo y no siempre todos lo poseen debido a costos muy elevados.

En nuestro medio no se posee un método que permita conocer si un vehículo abandona o no un lugar sin conocimiento del propietario. Para tratar de solucionar esta dificultad se recurre a la seguridad privada pero a pesar de que en algo ayuda a opacar el problema la intervención de ellos no se destina solamente al cuidado de los vehículos.

Sistemas de paso con barras o tarifados no siempre suelen ser eficientes ante esta problemática. En lo que se refiere a la ESPOCH a parte de la seguridad privada no se tiene un sistema que permita conocer si el auto que ingresó permanece dentro del campus, por el hecho de que un gran número de éstos circula dentro de este, inclusive los sistemas de barras se encuentran obsoletos.

El rastreo satelital es el único sistema que controla la permanencia de un vehículo en cualquier lugar pero debido a su costo este es poco usado. Mucho menos existen alternativas económicas que aprovechen el uso de la red de internet para saber si un automóvil permanece en el lugar donde se lo dejó.

ANTECEDENTES

La tasa del parque automotor del país y en consecuencia de provincia muestra niveles de un inminente crecimiento anual. Solamente en el año 2013 según el Instituto Nacional de Estadística y Censo (INEC) se registraron 1 717 886, de los cuales solamente en la provincia de Chimborazo constan 43 511 lo que representa un 2.53% del total de vehículos matriculados.

A partir de esto varios problemas sociales han surgido en consecuencia a la cantidad de vehículos que circulan en el medio tales como: la contaminación ambiental, visual y auditiva, tráfico e inseguridad.

Según datos de la INTERPOL en el año 2014 a nivel mundial más de siete millones de automotores fueron sustraídos, donde gran parte de ellos fueron vendidos fuera de los países de

donde procedían, lastimosamente el Ecuador forma parte de dichas estadísticas. En el país la cifra de autos sustraídos en el año 2015 fueron de alrededor 7478 de los cuales solamente se pudieron recuperar 2695 según informes de Estadísticas de Seguridad Integral lo que generó una pérdida económica de aproximadamente USD 22 millones según investigadores locales.

FORMULACIÓN DEL PROBLEMA

¿Se poseen métodos que garanticen la permanencia de un vehículo dentro del campus de la ESPOCH, para evitar una posible sustracción del mismo?

JUSTIFICACIÓN TEÓRICA

Al ser testigos del auge que tiene hoy en día el uso de la red de Internet para acceder a todo tipo de información se puede aprovechar este fenómeno social para brindar una alternativa a quienes hacen uso de un automóvil como medio principal de transporte para conocer si el mismo no ha abandonado el lugar donde su propietario lo dejó simplemente usando un dispositivo que tenga acceso a la red.

Es por ello que se aplica el concepto de Internet of Things (IoT) o internet de las cosas para de esta manera mantener al sistema en una permanente conexión a la red, con el fin de que los dispositivos usados brinden la posibilidad de que el usuario interactúe con objetos cotidianos de la vida, en este caso mantener un contacto remoto con la presencia de un vehículo.

La visión por computador que es una rama de la inteligencia artificial se dedica al desarrollo de algoritmos computacionales capaces de emular al sentido de la vista, de tal manera que se aprovechen las técnicas y métodos que se aplican en el procesamiento digital de imágenes para obtener de ellas la información necesaria para cumplir un objetivo, como puede ser: el reconocimiento de patrones, objetos, tracking, visión robótica, visión estéreo, entre otras.

Parte esencial de la visión artificial son los algoritmos de reconocimiento de caracteres los que engloban un conjunto de técnicas y procesos basados en comparaciones de los mismos, empleados para distinguir de forma automática caracteres alfanuméricos. En realidad es muy complicado registrar los caracteres de un determinado alfabeto sino que es posible reconocerlos a partir de cualquier conjunto de formas o símbolos.

Al usar un sistema embebido se puede aprovechar su característica principal que es la de fusionar diversos dispositivos, sensores, microcontroladores y demás con la finalidad de que su trabajo en conjunto satisfaga la necesidad por la que fue creado; en este caso es de vital importancia que todo este proceso se lo realice en tiempo real ya que toda la información extraída por el sistema será inmediatamente subida a una base de datos alojada en la red de Internet.

JUSTIFICACIÓN APLICATIVA

El sistema embebido se instalará en el acceso principal de la Escuela Superior Politécnica de Chimborazo lugar propicio para obtener la información de interés de los vehículos que ingresan al mismo.

Al entrar un vehículo al campus la cámara dispuesta en un sitio estratégico tomará una foto de la escena donde se pueda obtener una imagen clara tratando de que factores como la distancia entre la cámara - objetivo, ángulo de disparo, entre otros, no interfieran de manera negativa en el proceso de adquisición. Cabe recalcar que al momento de que el vehículo abandone el campus realizará la misma acción.

Una vez que se obtiene la escena digitalizada ésta es procesada a partir de técnicas de procesamiento tales como: la segmentación, filtrado, eliminación de ruido y demás, se obtiene el área de interés que será la placa del vehículo y a partir de un algoritmo de reconocimiento de caracteres se extraerá la información de la placa que es el distintivo único que diferencia a cada automóvil.

Posterior al reconocimiento de la placa vehicular esta información será almacenada en una base de datos; ésta estará disponible en la red de para poder tener acceso a ella en cualquier momento desde cualquier lugar simplemente teniendo al alcance un dispositivo que permita el acceso a la web como puede ser: el celular, computador personal, tablet, y demás.

Todo este proceso será ejecutado por un sistema embebido para de esta manera adaptar todos los conceptos de visión artificial, bases de datos y redes de internet en un solo sistema robusto, confiable y eficiente capaz de ser implantado en cualquier sitio que requiera obtener información acerca de la permanencia de vehículos.

OBJETIVOS

OBJETIVO GENERAL

Diseñar e implementar un sistema embebido para el control de ingreso y salida de vehículos a través de Internet, para conocer la permanencia de ellos dentro del campus de la ESPOCH.

OBJETIVOS ESPECÍFICOS

- Emplear visión artificial para el registro de acceso o salida de vehículos en el campus de la ESPOCH.
- Analizar el funcionamiento de las librerías que ofrece OpenCV para el procesamiento de imágenes aplicado a la visión artificial.
- Aplicar un algoritmo computacional adecuado, para el reconocimiento de caracteres alfanuméricos.
- Diseñar una base de datos con acceso a la red, para almacenar la información extraída por el sistema.

CAPÍTULO 1

1 MARCO TEÓRICO

1.1 Sistemas embebidos

Se define a un sistema embebido o incrustado como cualquier equipo computacional programable, que cumple una función específica generalmente ejecutando sus tareas en tiempo real.

"Un sistema embebido es un sistema cuya función principal no es computacional, pero es controlado por un computador integrado. Este computador puede ser un microcontrolador o un microprocesador. La palabra embebido implica que se encuentra dentro del sistema general, oculto a la vista, y forma parte de un todo de mayores dimensiones." (Pérez, 2009, p.04)

Como todo sistema computacional éste se compone de un hardware y un software embebido como el elemento más importante. El sistema es independiente o parte de uno más grande y puesto que su software está embebido suele alojarse en memoria solo de lectura (ROM) por lo que prescinde de una memoria volátil. (Pérez, 2009, p.04)

En un sistema incrustado se distinguen tres características que definen y distinguen a este de otros sistemas computacionales.

- En este sistema se ejecutan tareas específicas de forma repetitiva, a diferencia de un sistema computacional ordinario que ejecuta cierta cantidad de programas.

- Guarda ciertas limitaciones en su implementación como: diseño, tamaño, costo desempeño, consumo energético, etc. "*Los sistemas embebidos generalmente deben ser poco costosos, poseer un tamaño reducido, tener un buen desempeño para procesar datos en tiempo real, y además consumir un mínimo de energía para extender el tiempo de vida de las baterías o prevenir la necesidad de elementos adicionales de enfriamiento"* (Pérez, 2009, p.05)

- Capacidad de reacción ante cambios ambientales, sin interrumpir su operación ni retrasar las funciones otorgadas.

1.1.1 Componentes

En un sistema embebido se distinguen tres componentes principales:

1.1.1.2 Hardware

Como en todo sistema, el término hardware se refiere a todo componente físico que constituye el mismo, que en conjunto con el software realizan tareas específicas. Dichos componentes se distinguen de los sistemas computacionales tradicionales en diversos aspectos como: tamaño, consumo energético, funcionalidad, capacidad de procesamiento, etc.

El hardware que comúnmente compone un sistema incrustado se describe a continuación:

- **Microprocesador / Microcontrolador:** Componente LSI (Large Scale Integration) que brinda la capacidad de cómputo al sistema y se encarga de realizar tareas o funciones en una sola pieza de circuito integrado, el mismo que se forma por miles de transistores y otros componentes electrónicos.

- **Sensores:** Dispositivo que se encarga de percibir las señales físicas de un fenómeno para traducirla a otro tipo de señales generalmente eléctricas.

- **Memoria:** Espacio de almacenamiento para la retención de datos y su posterior uso.

1.1.1.3 Software

Se define al software de un sistema embebido como la aplicación que ejecuta una tarea en particular de forma repetitiva.

"El software que se ejecuta en un sistema embebido es diseñado bajo algunas restricciones importantes: (i) cantidades pequeñas de memoria, generalmente en el orden de los KB, (ii) capacidades limitadas de procesamiento, generalmente los procesadores poseen velocidades que no superan los Mhz, (iii) la necesidad de limitar el consumo de energía en cualquier instante, bien sea en estado de ejecución o no." (Pérez, 2009, p.04)

1.1.1.4 Sistemas operativos embebidos

El sistema operativo brinda una capa de interacción entre el usuario y el sistema, además de ponerse a cargo en la administración de todos los recursos del mismo. En el caso de un sistema operativo embebido es aquel que se ejecuta bajo un sistema incrustado con características en tiempo real. Pérez lo define como un "software muy pequeño desarrollado específicamente para ser usado con un algún sistema embebido en particular, o en ocasiones puede ser una versión reducida de algún sistema operativo que se utiliza en una computadora de propósito general." (Pérez, 2009, p.11)

1.2 Visión artificial

Se considera a la visión como un sentido imprescindible para el ser humano, aunque no necesariamente exclusivo, puesto que no impide la concepción de actividades mentales. El motivo por el que nace el término visión artificial, básicamente fue el de dotar al ser humano la capacidad de mejorar la calidad de interpretación y el procesamiento de la información para los sistemas o máquinas autónomas, todo esto complementado con mecanismos sensoriales que puedan complementar al proceso. (Pajares y De la Cruz, 2007, p. 01)

Fu, menciona que: La visión artificial puede ser definida como los procesos de obtención, caracterización e interpretación de información de imágenes tomadas de un mundo tridimensional. (Fu et.al, 1989, p. 306)

Como un concepto general, Pajares y De la Cruz lo definen como "la capacidad de la máquina para ver el mundo que le rodea, más precisamente para deducir la estructura y las propiedades del mundo tridimensional a partir de una o más imágenes bidimensionales". (Pajares y De la Cruz, 2007, p. 02)

El proceso que describe en general a un sistema de visión artificial se lo puede sintetizar en la Figura 1-1, donde la escena tridimensional percibida por un sensor genera una información en forma de una imagen, para posteriormente someterla a un proceso de segmentación y obtener características que generen algún tipo de descripción, para finalizar con algún tipo de aplicación. (Pajares y De la Cruz, 2007, p. 02)

Figura 1-1: Proceso de un sistema de visión artificial

Realizado por: Manosalvas C. 2017

1.2.1 Arquitectura de un sistema de visión artificial

Un sistema de visión artificial está conformado de al menos cinco elementos:

- **Dispositivo de captura:** Unidad óptica que recepta las señales analógicas del mundo real.
- **Conversión A/D:** La señal analógica obtenida se convierte a una o varias señales digitales.
- **Memoria:** Donde se almacena la información obtenida.
- **Procesador:** Trabaja en conjunto con la memoria para operar sobre la imagen digitalizada.
- **Monitor:** Permite la visualización de los procesos realizados, aunque se puede prescindir del mismo.

1.2.1.1 Dispositivos de captura de imágenes

Para adquirir una imagen a partir de una escena, es fundamental un dispositivo físico que sea sensible a cierta banda del espectro electromagnético (Figura 2-1), que produzca una señal eléctrica en proporción a la energía percibida por el sensor, para luego ser digitalizada. (Sucar y Gómez, 2003, p. 12)

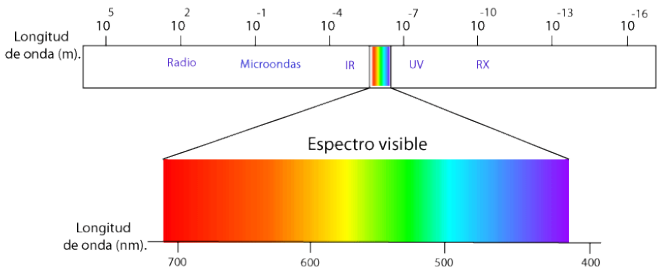


Figura 2-1: Espectro electromagnético

Fuente: http://www.aulaclic.es/fotografia-photoshop/graficos/espectro1.gif

Entre los principales dispositivos de captura se tienen: cámaras fotográficas, scanner, sensores de rango y ultrasonido, rayos X, imágenes de tomografías y resonancias, etc.

1.2.1.2 Digitalización de una imagen.

Es el proceso por el cual se convierte las señales analógicas obtenidas por el sensor a una señal continua, es decir a partir del muestreo de la imagen obtenida se convierte a una matriz discreta de MxN píxeles.

1.2.2 Representación de una imagen

La imagen obtenida por cualquier dispositivo de captura se presenta digitalizada en forma de una matriz de M x N elementos; representado a través de un sistema coordenado f(x,y); mostrando su origen en el extremo superior izquierdo (Figura 3-1). A cada elemento de este arreglo se lo denomina pixel (picture element). (Pajares y De la Cruz, 2007, p. 07)

Cada uno de los píxeles guarda cierta información, es decir en el caso de que la imagen esté en niveles de grises dicha información será el brillo. En imágenes a color se expresa la intensidad de cada uno de los componentes de la base de color.

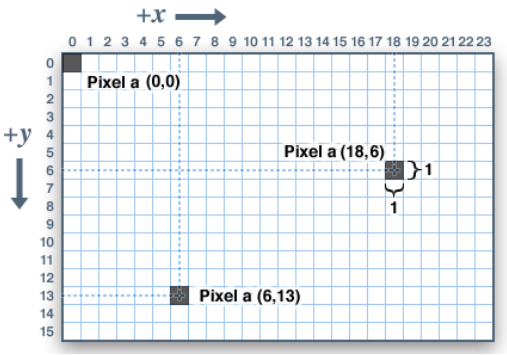


Figura 3-1: Sistema coordenado de una imagen

Fuente: http://www.diarioelectronicohoy.com/blog/lcds-graficos-pcd8544

Las coordenadas de un píxel (x,y) tiene dos vecinos horizontales y verticales, a este conjunto se le denomina, vecindad a cuatro de un píxel (Figura 4-1).

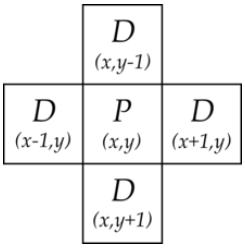


Figura 4-1: Vecindad a cuatro de un píxel.

Fuente: https://upload.wikimedia.org/wikipedia/commons/thumb.png

Cada uno de ellos tiene una distancia unitaria desde su posición original. Las coordenadas vienen dadas por la Ecuación 1-1.

$$(x+1,y),(x-1,y),(x,y+1),(x,y-1) \qquad \text{Ecuación 1-1}$$

Para los píxeles diagonales, conocidos como vecindad a ocho (Figura 5-1), las coordenadas vienen dadas por la Ecuación 2-1.

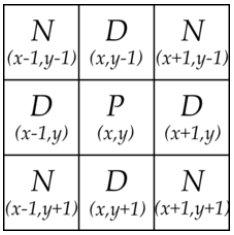


Figura 5-1: Vecindad a ocho de un píxel

Fuente: https://upload.wikimedia.org/wikipedia/com/thumb/f/fb/Achter.png

$$(x+1,y+1),(x+1,y-1),(x-1,y+1),(x-1,y-1) \qquad \text{Ecuación 2-1}$$

1.2.3 Segmentación

La imagen que arroja un dispositivo de captura generalmente no es lo suficientemente buena para extraer información de manera adecuada, por lo que se tiene que usar ciertas técnicas con el fin mejorar la misma y separar la imagen tratada en sus partes constitutivas, es decir apartar los fragmentos o características de la imagen para facilitar su análisis o reconocimiento.

La segmentación es el proceso por el cual a través de técnicas y procedimientos se extrae información útil de una imagen, para su posterior uso. Este proceso se basa en la similitud (se orienta hacia regiones) y discontinuidad (se orienta hacia bordes). (Pajares y De la Cruz, 2007, p. 08)

"La segmentación es el proceso que divide a una escena percibida en sus partes constituyentes u objetos. La segmentación es uno de los elementos más importantes de un sistema automatizado de visión, ya que este es el nivel de procesamiento en que los objetos se extraen de una imagen para su posterior reconocimiento y análisis." (Fu et.al., 1989, p. 376)

1.2.3.1 Binarización

La binarización es el proceso por el cual se compara los niveles de grises de una imagen con un valor determinado conocido como umbral (threshold). Si el nivel de gris presente en el píxel es menor al umbral, a este se le asigna un valor de cero lo que supone el color negro (Ecuación 3-1), de manera análoga, si el píxel es mayor al umbral este toma el valor de uno o blanco (Ecuación 4-1). (Ezqueda, 2002, p.24)

$$\boldsymbol{S}[\boldsymbol{x},\boldsymbol{y}] = 0\ (0), E[x,y] \leq T \qquad \text{Ecuación 3-1}$$

$$\boldsymbol{S}[\boldsymbol{x},\boldsymbol{y}] = 1\ (255), E[x,y] > T \qquad \text{Ecuación 4-1}$$

El problema principal de la binarización es el de hallar el valor de umbral adecuado para que se tome valores de cero o uno, ya que el nivel de gris de una imagen puede variar por diversos factores y eliminar información que podría ser útil para el sistema o inclusive la aparición de zonas no deseadas; es por esto que a este proceso se le puede definir como una técnica de prueba y error hasta hallar el valor adecuado de thresold. En la Figura 6-1 se aprecia el cambio de valor del umbral y el resultado de la imagen.

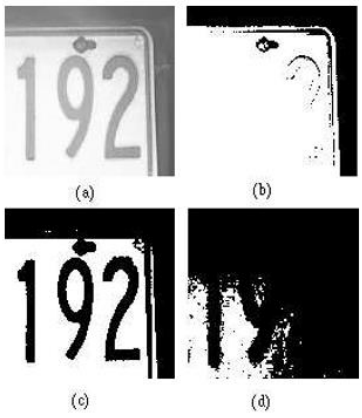


Figura 6-1: Binarización. (a) Imagen original. (b) Binarización umbral = 150. (c) Binarización umbral = 200.
(d) Binarización umbral = 250.

Fuente: Sucar & Gómez, 2002.

Fu considera que el uso de los umbrales en el tratamiento de imágenes es una de las principales técnicas usadas por los sistemas de visión industrial para la detección de objetos, especialmente en aplicaciones que requieran una cantidad elevada de datos. (Fu et. al, 1989, p. 370)

1.2.3.2 Detección de contornos

La detección de contornos es una técnica aplicada al procesamiento de imágenes de donde se extrae información de los límites de una forma o un objeto, que pueden ser usados posteriormente para la detección y análisis del mismo. Se basa en las variaciones de la intensidad de un pixel, si éste pasa de un valor de 255 (blanco) a 0 (negro) o viceversa, representa un contorno.

Para comprender el concepto de contorno, Vélez et al. mencionan que:

"El contorno de un objeto en una imagen digital corresponde al mínimo conjunto de píxeles que separa ese objeto del fondo o background de la imagen. Normalmente estos contornos se corresponden con los puntos donde se producen discontinuidades en los valores de píxeles adyacentes (cambios en el matiz o el brillo) o con los puntos donde cambia un patrón que se repite (cambios de textura)." (Vélez et al., 2003, p. 128)

La detección de contornos muestran las fronteras de los objetos, por lo que esta técnica es de gran utilidad para la identificación y segmentación de formas u objetos.

1.2.4 Operaciones morfológicas

Los procesos que abarcan la segmentación de la imagen no siempre garantizan que la región de interés sea la deseada, pueden presentarse problemas como presencia de ruido, discontinuidad de píxeles, bordes no deseados, etc. Las operaciones morfológicas se encargan del realce de la geometría de los pixeles, fundamentándose en la teoría de conjuntos tomando a una imagen binaria como un espacio R^2.

1.2.4.1 Erosión

Esta operación morfológica degrada la imagen fuente, eliminando los píxeles vecinos que toman valor de uno en la imagen fuente, como se aprecia en la Figura 7-1. Medina y Contreras definen a la erosión como el conjunto de todos los elementos x para los cuales B trasladado por x está contenido en A. (Medina y Contreras, 2015, p.34).

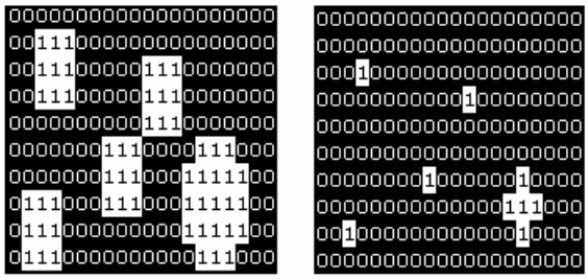


Figura 7-1: Erosión de una imagen binaria.

Fuente: http://eia.udg.es/~rafa/docencia/morfologia.pdf

Matemáticamente la erosión se define como:

$$\boldsymbol{A\theta B} = \{x/B_x \subseteq A\} \qquad \text{Ecuación 5-1}$$

1.2.4.2 Dilatación

La dilatación representa una ampliación de la imagen fuente, donde todos los píxeles de la vecindad toman un valor de uno (Figura 8-1). Esta operación da como resultado a un conjunto de elementos, donde al menos un elemento de B es contenido en A (Ecuación 6-1).

$$\boldsymbol{A\theta B} = \{x/(B^{\wedge})_x \cap A\} \qquad \text{Ecuación 6-1}$$

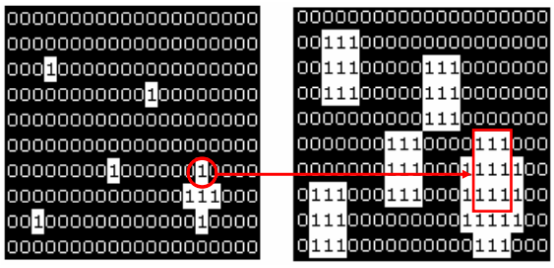


Figura 8-1: Dilatación de una imagen binaria

Fuente: http://eia.udg.es/~rafa/docencia/morfologia.pdf

1.3 OpenCV

OpenCV, acrónimo de las palabras Open Source Computer Vision, es un conjunto de librerías de código abierto desarrollado por Intel en 1999, que posee un amplio rango de módulos dedicados a resolver problemas de visión por computador. Esta librería está desarrollada bajo el lenguaje de

programación C y C++, ejecutado bajo las plataformas de Windows, MacOs y Linux, con la posibilidad de desarrollarlo en interfaces como Matlab, Python u otros lenguajes. Estas librerías fueron desarrolladas para aprovechar la eficiencia computacional en aplicaciones de tiempo real que requieren un enfoque total de todos sus recursos hacia un proceso.

El objetivo de OpenCV es el de proveer una infraestructura fácil de usar, para desarrollar proyectos sofisticados de visión de una manera rápida y simple, con la ayuda de alrededor de 500 funciones que abarcan muchas áreas como: inspección y control de calidad, seguridad, calibración de cámaras, visión estéreo, tratamiento de imágenes médicas, robótica, entre otras.

Hoy en día se ha expandido el uso de estas librerías, teniendo un aproximado de dos millones de descargas y el número crece en un promedio de 26 000 por mes. El grupo de usuarios alcanza los 20 000 miembros de los cuales se recibe las contribuciones de mejoramiento de los algoritmos y librerías de OpenCV. (Bradski y Kaehler, 2008, p.01)

1.3.1 Opencv – Python

Se considera a Python como un lenguaje de programación de alto nivel de propósito general, sencillo de aprender debido a su fácil codificación y sintaxis, evitando errores que son comunes al programar en diferentes lenguajes.

Para mejorar su comprensión a éste se lo maneja bajo un diseño visual con palabras claves inglesas que a diferencia de otros lenguajes de programación lo hacen con el uso de puntuación. Del mismo modo para delimitar un bloque de programa se omite el uso de caracteres como paréntesis, llaves u otros; en lugar de ellos se aumenta o disminuye la sangría del código lo que mejora su legibilidad.

A comparación con otros lenguajes como C o C++, Python suele ser relativamente lento debido a que se orienta a ser un lenguaje comprensible, es decir de alto nivel, por lo que toma más tiempo pasar desde el código fuente en el que está escrito hacia el lenguaje que es comprensible por la máquina o conocido como lenguaje de bajo nivel.

Debido a que todos los algoritmos que se usan en OpenCV están escritos bajo el lenguaje C++, estos se pueden vincular con Python, gracias a los enlaces generados dentro del programa que sirven como puente entre estos dos lenguajes, lo que quiere decir que se puede llamar a funciones

de C a través de Python y de esta manera poder resolver problemas o generar nuevos procedimientos y algoritmos en tiempo real para el campo de la visión artificial.

1.4 Reconocimiento óptico de caracteres (OCR)

Un OCR proveniente de las siglas en inglés Optical Character Recognition, es el proceso por el cual a partir de una imagen o símbolos obtenidos de un escenario, se adquiere el texto digitalizado correspondiente a un alfabeto que es reconocible por un ordenador, para su posterior procesamiento.

"La tecnología de reconocimiento de caracteres, OCR (Optical Character Recognition) engloba a un conjunto de técnicas basadas en estadísticas, en las formas de los caracteres, transformadas y en comparaciones, que complementándose entre sí, se emplean para distinguir de forma automática entre los diferentes caracteres alfanuméricos existentes" (Sánchez y Sandonís, 2010, p. 01)

Los factores que pueden derivar en un mal reconocimiento o la inefectividad de un algoritmo OCR implementado, no siempre son debido al sistema sino a causas como:

- Deterioro del objetivo.
- Letra borrosa.
- Fragmentación o solapamiento de letras.
- Tipografías extrañas.

1.4.1 Etapas de un OCR

El proceso básico que sigue un OCR se detalla en la Figura 9-1:

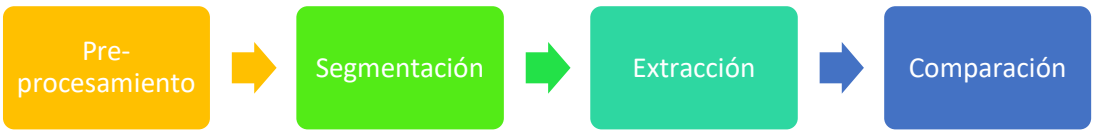


Figura 9-1: Proceso básico de un OCR

Realizado por: Manosalvas C. 2017.

La primera etapa de pre-procesamiento se la conoce también como adecuación de la imagen, donde el objetivo que se sigue es eliminar cualquier imperfección o ruido presente en la misma, para posteriormente incluir un proceso de binarización que delimitará notoriamente el carácter que se someterá a las siguientes etapas. (Sánchez y Sandonís, 2010, p. 01)

Posterior al pre-procesamiento continúa la segmentación, etapa en la cual se debe fragmentar las diferentes partes en donde los píxeles son adyacentes entre sí, para lograr una imagen lo suficientemente significativa para poder reconocer cada carácter dentro de la imagen binaria. (Sánchez y Sandonís, 2010, p. 02)

La tercera fase del proceso corresponde a la extracción, en donde se realizan operaciones morfológicas para obtener una imagen más reconocible. Operaciones como la erosión o dilatación son muy comunes en esta etapa, puesto que trabajan sobre la imagen eliminando los puntos de cada componente pero conservando su tipología y proporciones de la imagen original evitando la deformidad de la misma y facilitando su posterior reconocimiento o comparación.

El resultado de los anteriores procesos permite tener las características sobresalientes de la imagen a reconocer y analizar para comparar con los caracteres teóricos establecidos en dependencia al método de reconocimiento a implementar.

1.4.2 Algoritmos OCR

Los algoritmos más sobresalientes en cuanto al reconocimiento de caracteres se detallan a continuación:

1.4.2.1 Redes neuronales

Medina y Contreras definen a una red neuronal como:

"Las redes neuronales artificiales tratan de emular el funcionamiento del cerebro humano, creando conexiones entre unidades de procesamiento denominadas neuronas. La unidad central de la red es una neurona artificial que ha sido modelada matemáticamente a partir de una neurona biológica" (Medina & Contreras, 2015, p. 39)

Dicha red se compone de nodos o también conocidas como unidades que están unidos mediante conexiones, a las cuales se les asigna un peso numérico. Dicho peso constituye la memoria a largo

plazo de la red donde el aprendizaje se lo realiza con la actualización de dichos pesos con dependencia de que si estas unidades se han establecido como entrada o como salida de la red. Figura 10-1 (Russell & Norvig, 1996, p. 597)

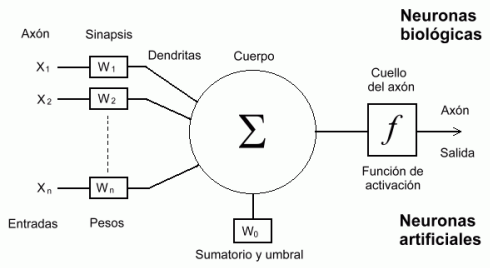


Figura 10-1: Neurona artificial

Fuente: http://www.um.es/LEQ/Atmosferas/Ch-VI-3/6-3-8.GIF

Una red neuronal necesita aprender para saber cómo reaccionar ante cualquier estímulo a su entrada. El proceso de aprendizaje es el algoritmo que utiliza la red para modificar los pesos de tal forma, que el carácter de entrada tenga a su salida un peso específico, por lo que se necesita un conjunto de entrenamiento de los patrones que se va a reconocer. (Bonilla, 2005, p. 22)

Una red de clasificación (Figura 11-1) se entrena para que los patrones almacenados se dividan en clases o conjuntos de entrenamiento; cuando hay un estímulo a la entrada y este pertenece al conjunto de clases, se asigna a la salida el código del patrón clasificado que generalmente se obtiene en código binario. (Bonilla, 2005, p. 21)

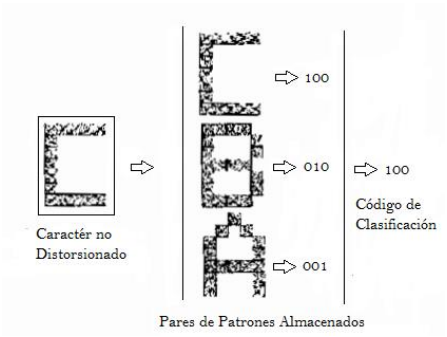


Figura 11-1: Red de clasificación

Fuente: Bonilla 2005.

Por otro lado, un diferente tipo de red neuronal conocida como red de reconocimiento (Figura 12-1), trabaja con estímulos o entradas distorsionadas y/o incompletas, las que difieren del conjunto de entrenamiento que es almacenado, brindando a la salida el código binario correspondiente al

patrón que más se asemeja a la entrada. Este tipo de red se usa donde el ambiente presenta ruido. (Bonilla, 2005, p. 21)

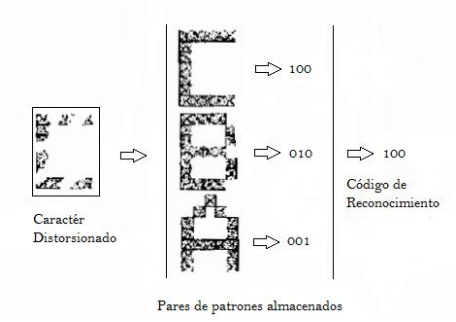


Figura 12-1: Red de reconocimiento

Fuente: Bonilla 2005.

1.4.2.2 Árboles de decisión

Un árbol de decisión es una técnica de búsqueda de datos que se puede aplicar en el reconocimiento de caracteres, basando su funcionamiento en una estructura jerárquica formada por nodos y hojas como se puede apreciar en la Figura 13-1. Cada nodo representa los atributos o patrones que se quieren evaluar y las hojas constituyen el lugar de almacenamiento de los resultados finales en base a la valoración de los nodos. La rama que conecta a un cierto número de nodos representa uno de los valores que posiblemente tomará la hoja en función al nodo padre. (Sánchez y Sandonís, 2010, p. 04)

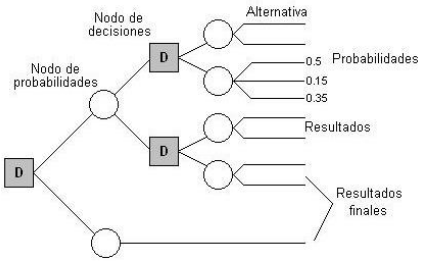


Figura 13-1: Estructura del árbol de decisiones

Fuente: http://www.monografias.com/trabajos106/analisis-sensibilidad-arboles-decisionpuntode-equilibrio/image005.jpg

La arquitectura de un árbol de decisión basa su funcionamiento en estructuras simples de programación IF, THEN, ELSE y éstas favorecen a la velocidad de cálculo pero su implementación puede ser muy costosa computacionalmente debido a la cantidad de ramas, nodos y hojas que intervienen en el proceso de selección. (Sánchez y Sandonís, 2010, p. 04)

El proceso de clasificación inicia en la raíz donde por medio de una función se realiza la evaluación de la característica particular del objeto en análisis. Cada una de las ramas que salen de la raíz representa los valores posibles que puede tomar el objeto inicial en base a sus atributos, y en base a ellos se continúa por cada nodo. En cada nodo se toma una decisión hasta que se alcanza una hoja, donde se etiqueta al objeto clasificado.

"En definitiva, si durante el recorrido se llega a una hoja, esa serie de patrones responden a un carácter reconocido y por tanto se devuelve. Si en el recorrido y después de evaluar todo el árbol no se llega a una hoja, se deduce que esa característica no ha sido consignada en el árbol y por tanto se debe crear una nueva hoja para guardar el nuevo dato." (Sánchez y Sandonís, 2010, p. 05)

1.4.2.3 Vecinos más cercanos

El método de los vecinos más cercanos KNN (K - Nearest Neighbors), basa su funcionamiento en la búsqueda del prototipo más cercano dentro de un conjunto de entrenamiento y modelos preestablecidos llamados clase, en donde se estima la probabilidad de que uno de ellos pertenezca a dicha clase. Ver Figura 14-1. (Cambronero, 2009, p.03)

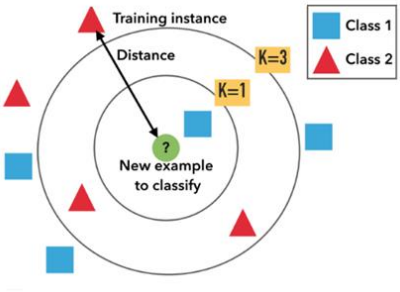


Figura 14-1: Algoritmo KNN bidimensional

Fuente: https://static1.squarespace.com/static/55ff6aece4b0ad2d251b3fee/t/5752540b8a65e246000a2cf9/1465017829684/

Si se posee un conjunto de objetos prototipos de los que a priori se conoce su clase o conjunto de muestra, y por otro lado se tiene un objeto desconocido, el objetivo del algoritmo es buscar dentro de las muestras, los "k" objetos más parecidos al nuevo, asignando la clase más numerosa entre los "k" objetos prototipos seleccionados. (Sánchez y Sandonís, 2010, p. 04)

En el Reconocimiento Óptico de Caracteres, el proceso de aprendizaje del algoritmo se inicia con la obtención de las clases prototipo de las cuales se esperan reconocer; en otras palabras son las imágenes de los caracteres que posteriormente se van a comparar con la muestra obtenida por el sistema; a este conjunto se lo almacena dentro de un vector junto con la clases asociadas a cada muestra del mencionado conjunto. “Las predicciones se realizan basándose en los ejemplos más parecidos al que hay que predecir.” (García & Gómez, 2009, p.03)

A partir de dicho vector se calcula la distancia Euclidea (Ecuación 7-1), desde cada muestra de entrenamiento hasta todas las posiciones siguientes del vector de donde se conoce su clase, de las cuales se toma la clase “k” más cercana y se separa la muestra de entrenamiento en la clase más frecuente a los que pertenecen los “k” vecinos que se adquirieron.

$$E = \sqrt{\sum_{i=1}^{N}(x_i - y_i)^2} \qquad \text{Ecuación: 7-1}$$

1.5 Placas vehiculares

Es el distintivo único que caracteriza y diferencia a cada vehículo, en conformidad con la Agencia Nacional de Tránsito, ente que dispone el color y caracteres que estarán presentes en la placa, acordes al uso del automotor y su lugar de matriculación respectivamente.

A partir del año 2012 se modificó las características de la placa (Figura 15-1), entre las que destacan: su dimensión de 154 mm de alto y 404 mm de ancho construidas de material reflectivo y en dependencia al tipo de vehículo varía el color del borde superior con fondo blanco. (Tabla 1-1)

Figura 15-1: Placa vehicular Ecuador

Fuente: Agencia Nacional de Tránsito

Tabla 1-1: Borde superior de la placa vehicular

Servicio	Color
Comercial o Público	Naranja
Organismo del Estado	Oro
Gobiernos Autónomos Descentralizados	Verde Limón
Diplomáticos y Organismos Internacionales	Azul
Internación Temporal	Rojo
Particular	Blanco

Fuente: Agencia Nacional de Tránsito

Los caracteres alfanuméricos se forman de tres letras y cuatro dígitos desde 0000 a 9999, con la palabra ECUADOR en la parte superior. La primera letra corresponde a la provincia en donde el automotor fue matriculado por primera vez (Tabla 2-1), la segunda identifica el tipo de matrícula y la tercera es relativa.

Tabla 2-1: Primera letra de la placa por provincia

Azuay	**A**	El Oro	**O**	Los Ríos	**R**	Orellana	**Q**
Bolívar	**B**	Esmeraldas	**E**	Manabí	**M**	Sucumbíos	**K**
Cañar	**U**	Galápagos	**W**	Morona Santiago	**V**	Tungurahua	**T**
Carchi	**C**	Guayas	**G**	Napo	**N**	Zamora Chinchipe	**Z**
Cotopaxi	**X**	Imbabura	**I**	Pastaza	**S**	Santa Elena	**Y**
Chimborazo	**H**	Loja	**L**	Pichincha	**P**	Santo Domingo	**J**

Fuente: Agencia Nacional de Tránsito

1.6 Raspberry Pi

Es un computador de placa reducida y bajo costo de uso libre, diseñado por la fundación Raspberry Pi con el fin de motivar al público a inmiscuirse en el mundo de la programación. Trabaja bajo el software de código abierto desarrollado por la misma empresa denominado Raspbian, aunque no es el único sistema operativo con el que pueda ejecutarse.

A partir de su creación se ha lanzado varias versiones, entre ellas la Raspberry Pi 1 modelo A, B y B+, Raspberry Pi 2 modelo B, y su última versión la Raspberry Pi 3 modelo B mostrada en la Figura 16-1, cada una de ellas difiere de su anterior versión por el perfeccionamiento de sus características y rendimiento.

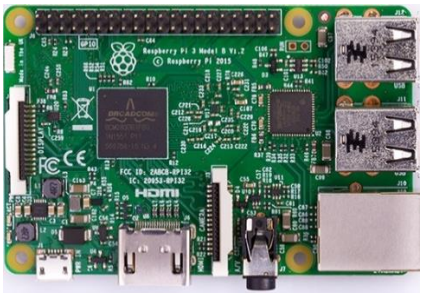


Figura 16-1: Raspberry Pi3

Fuente: https://www.raspberrypi.org/products/raspberry-pi-3-model-b/

1.6.1 Hardware RPi3

A este modelo se lo considera como la tercera generación de la Raspberry. Las características de hardware se muestran en la Tabla 3-1:

Tabla 3-1: Hardware Raspberry Pi3

Raspberry Pi 3 Modelo B	
Unidad central de proceso	• 1.2 Ghz • Arquitectura de 64 bits • Cuatro núcleos ARMv8
Juego de instrucciones	• RISC de 32 bits
Memoria (SD RAM)	• 1 Gb
Puertos USB 2.0	• 4
Entradas de video	• Conector MIPI CSI, que permite alojar el módulo de cámara
Salidas de video	• Conector RCA • High Definition Multimedia Interface • Interfaz DSI para panel LCD
Almacenamiento	• A través de tarjeta Micro SD
Conectividad a la red	• Ethernet RJ45 • WiFi 802.11n • Bluetooth 4.1
Periféricos de bajo nivel	• 40 x GPIO
Consumo energético	• 800 mA • 4 W

Fuente de alimentación	• 5V vía microUSB • GPIO Header
Dimensiones	• 85.60 mm x 53.98 mm

Realizado por: Manosalvas C. 2017.

1.6.2 Software RPi3

El software que ejecuta la placa Raspberry generalmente se basa en los sistemas operativos con distribuciones de Linux. Los sistemas oficiales que operan la RPi3 son:

- Raspbian Jessie.
- Raspbian Jessie Lite.
- Ubuntu Mate.
- Kali Linux.
- Windows 10.
- LibreELEC.
- OSMC.

1.6.2.1 Raspbian Jessie

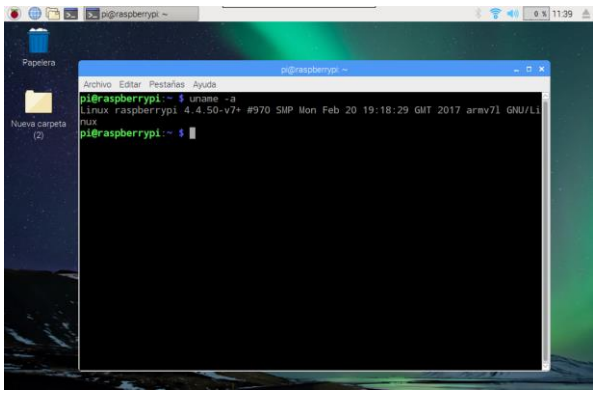


Figura 17-1: Entorno Raspbian Jessie

Realizado por: Manosalvas C, 2017.

Sistema operativo basado en Debian, que incluye utilidades y programas básicos capaces de controlar y optimizar los recursos hardware de la placa Raspberry. Cuenta con alrededor de 35000

paquetes y un software empaquetado y pre-compilado en un formato de instalación y manejo sencillo.

Al arrancar el sistema operativo carga por defecto su interfaz gráfica, que a diferencia de versiones anteriores, esto se lo realizaba por medio del terminal y se lo ejecutaba por medio de comandos manualmente. Si bien esto facilita el manejo de su entorno también supone un mayor uso de recursos.

Con el fin de brindar mayor facilidad para interactuar con el sistema, Raspbian Jessie incluye la posibilidad de ingresar a las opciones de configuración del sistema operativo por medio de una interfaz gráfica, la que permite habilitar y configurar preferencias como: habilitación del puerto para el módulo de cámara, comunicación SSH, entre otros.

1.6.3 Módulo de cámara

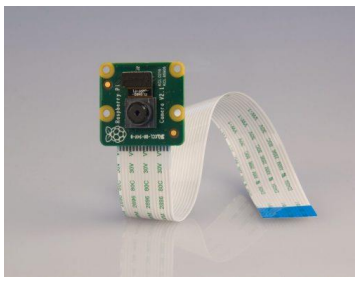


Figura 18-1: Cámara RPi3 8 Mp.

Fuente: https://www.raspberrypi.org/blog/new-8-megapixel-camera-board-sale-25/

El primer accesorio lanzado por Raspberry fue un módulo de 5 Mp, para tiempo después dar al mercado la cámara con un sensor de 8Mp lo que mejoraba la resolución y calidad de las imágenes adquiridas. Esta se conecta al controlador por medio de un puerto CSI lo que mejora la integración entre ellos. Las características que posee este módulo se detallan en la Tabla 4-1:

Tabla 4-1: Características cámara RPi3

Resolución	3280 x 2464 píxeles
Captura de video	1080p/30, 720p/60 y 640p x 480p/90
Sensor de imagen	Sony IMX219 de 8Mp
Dimensiones	25 x 23 x 9 mm
Marca	Raspberry Pi
Peso	18 g

Realizado por: Manosalvas C. 2017.

1.7 **Internet de las cosas (IOT)**

El Internet de las cosas, proveniente de las siglas IoT (Internet of Things), se refiere a la interconexión de los objetos cotidianos por medio del uso de la red, dotándolos con un cierto grado de inteligencia usando microcontroladores, transceptores para comunicación digital y protocolos que les permita interactuar entre ellos y con los seres humanos. Gracias a los avances tecnológicos y a la disponibilidad de acceder a la web en todo lugar, el Internet de las Cosas brinda un abanico de oportunidades para crear una infinidad de aplicaciones que pueden mejorar la calidad de vida.

El concepto de IoT además establece un fácil acceso e interacción con diferentes dispositivos como: cámaras, electrodomésticos, actuadores, vehículos, sensores y demás, pudiendo así desarrollar aplicaciones que provean nuevos servicios dentro de la domótica, automatización industrial, gestión inteligente de energía, gestión de tráfico de vehículos, redes inteligentes de sensores, asistencia médica y demás.

"Ahora debemos tener en cuenta que IdC representa la próxima evolución de Internet, que será un enorme salto en su capacidad para reunir, analizar y distribuir datos que podemos convertir en información, conocimiento y en última instancia, sabiduría. En este contexto, IdC se vuelve inmensamente importante." (Evans, 2011, p.01)

La popularización del Internet y el cómodo acceso que se tiene a él, facilita que los dispositivos y sensores puedan enviar grandes cantidades de datos a través de las redes fijas e inalámbricas para luego ser analizados por un computador, pretendiendo que dicho análisis sea cada vez más exhaustivo, para que el proceso que se realice sea optimizado y con ello evitar la intervención del ser humano.

Los cinco elementos fundamentales que debe tener un objeto que pretenda ser conectado a la red y ser considerado como inteligente son:

1. Sensores que permitan medir cualquier magnitud física del entorno que se pretende que esté inmerso en el IoT. Tomando en cuenta las limitaciones que tienen estos en consumo y tamaño.

2. El dispositivo de comunicación que permita transferir los datos obtenidos a través de la red. Existiendo muchas formas de comunicación, entre las cuales se tiene: M2M

(Machine to Machine), comunicación directa entre el objeto e Internet, comunicación con la nube, aplicando una de las tecnologías disponibles actualmente como: Bluetooth, NFC, ZigBee, WiFi, entre otras.

3. La fuente de alimentación que suministre la energía necesaria al sistema. Lo más común es colocar baterías debido a que estás pueden instalarse en lugares remotos donde no exista una red eléctrica. Se toma en cuenta el consumo de cada uno de los dispositivos/sensores/actuadores para dimensionar la batería.

4. El microcontrolador encargado de procesar y gestionar los datos que son intercambiados con los sensores/actuadores y el dispositivo de comunicación. Estos se pueden clasificar en tres categorías:

    4.1 Dispositivos de 8 bits. (Arduino)
    4.2 Sistemas basados en chips Ateros y ARM, con arquitectura de 32 bits.
    4.3 Microcontroladores de 64 bits. (Raspberry)

5. El lenguaje de programación implementado.

CAPÍTULO II

2 MARCO METODOLÓGICO

2.1 Obtención de imágenes

Al ser un sistema que se basa en visión artificial, el dispositivo considerado como principal es una cámara que permita adquirir imágenes del sitio de interés donde va a instalarse el sistema embebido, para posteriormente realizar el debido tratamiento de las imágenes conseguidas.

2.1.1 Selección del dispositivo de captura

Actualmente el mercado ofrece una infinidad de dispositivos de captura en dependencia a la función que se le va a otorgar. Se encuentra diversos tipos de ellas clasificándolos en función a su uso, conexión, tipo de sensor y demás características.

Al usar la Raspberry Pi3 como controlador principal se abre la posibilidad de usar la cámara que provee el fabricante debido a la facilidad de integración entre ellos. Se ha descartado la posibilidad de usar cámaras web debido al tamaño que ocuparía dentro del sistema ya que estas tienen dimensiones relativamente grandes, lo que implica un aumento en el espacio físico del sistema diseñado. Por otro lado una cámara IP presenta varias dificultades para trabajar con OpenCV si a éstas no se les configura previamente. Es por estas razones que se ha decidido trabajar con la cámara que provee el mismo fabricante de la tarjeta controladora.

2.1.2 Captura de imágenes

Luego de conectar el módulo de la cámara a través del puerto SCI, antes de iniciar con la captura de imágenes, este requiere una activación que se le puede realizar de dos maneras:

El entorno de Raspbian Jessie fue destinado a que exista una interfaz más amigable con el usuario, para que pueda administrar los recursos que este ofrece de una manera más sencilla. El módulo de la cámara no fue la excepción, basta con ingresar a la configuración del sistema y habilitar la opción de la interfaz de la cámara como se muestra en la Figura 1-2.

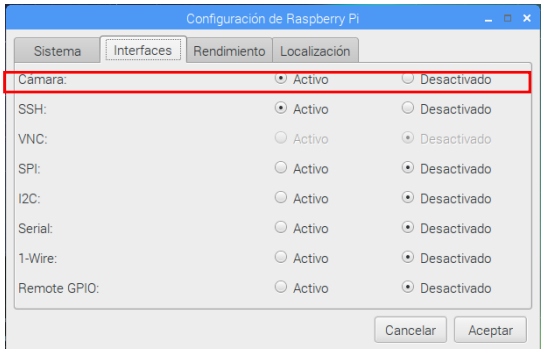


Figura 1-2: Ventana de activación de la cámara

Realizado por: Manosalvas C. 2017.

La otra manera para activar esta función es a través del terminal, ingresando el comando *sudo raspi-config* el que muestra las diferentes opciones de configuración de la tarjeta, entre ellas la activación del módulo de cámara, como se aprecia en la Figura 2-2.

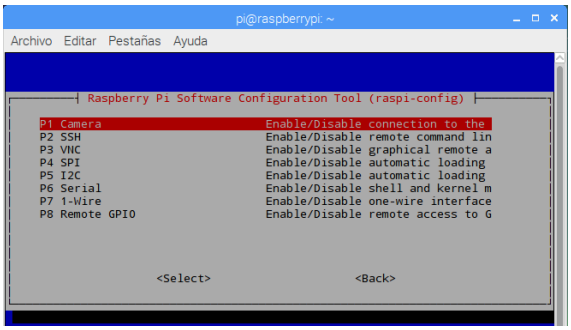


Figura 2-2: Activación de la cámara por terminal

Realizado por: Manosalvas C. 2017.

Para capturar las imágenes de los autos que ingresan al campus, es necesario tipear los comandos que se muestran en la Figura 3-2:

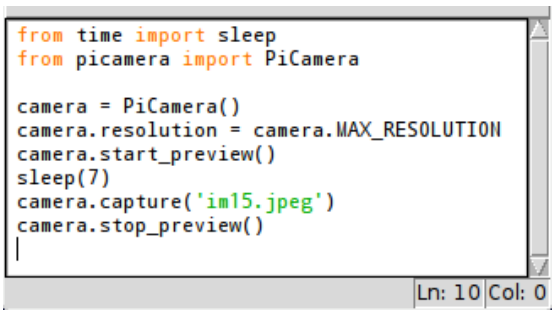


Figura 3-2: Comandos de captura de imágenes

Realizado por: Manosalvas C. 2017.

Se ha importado desde los paquetes *time* y *picamera*, las librerías *sleep* y *PiCamera* respectivamente. El paquete *time* brinda funciones relacionadas con tiempos de ejecución, en este caso la librería *sleep* se destina a un contador pausa o un retraso en el programa especificado en segundos. Del mismo modo la librería *PiCamera* obtenida de los paquetes de *picamera*, provee al lenguaje Python una interfaz entre este lenguaje y el módulo de la cámara.

Este módulo ofrece la posibilidad de adaptar a la necesidad del programador la resolución con la que se va a trabajar. En este caso se ha optado por 3280 x 2464 con una relación de aspecto de 4:3 que es la máxima permitida por la cámara, con el fin de que los detalles que presentan los vehículos no se pierdan. Elegir esta resolución ocupa más memoria en el almacenamiento y su tiempo de proceso relativamente más lento, pero esta diferencia de tiempos es prácticamente imperceptible si se elige una resolución menor.

Los comandos *start_preview* y *stop_preview* se usan para activar y desactivar la cámara respectivamente al comenzar un proceso, para ello se hace prescindible marcar el tiempo de trabajo del módulo, esto se ejecuta a través del comando *sleep* que lleva como argumento el tiempo en segundos.

Para almacenar la foto capturada se ejecuta el comando *.capture()*, el que lleva como argumento el nombre que se le va a asignar al archivo seguido de la extensión de la imagen. Se ha elegido el formato .png (Portable Network Graphics), ya que este a más de ocupar menos espacio en disco para su almacenamiento ofrece un contraste más profundo lo que resulta beneficioso para su posterior procesamiento.

2.2 Detección de la placa vehicular

Al culminar con el proceso de obtención de imágenes, es necesario que a través de varios procesos se determine el lugar en donde se encuentra la placa vehicular, objetivo con el cual se va a trabajar en lo posterior.

El diagrama de flujo mostrado en la Figura 4-2, muestra la secuencia de pasos que se llevan al cabo para extraer desde la imagen fuente la placa del vehículo.

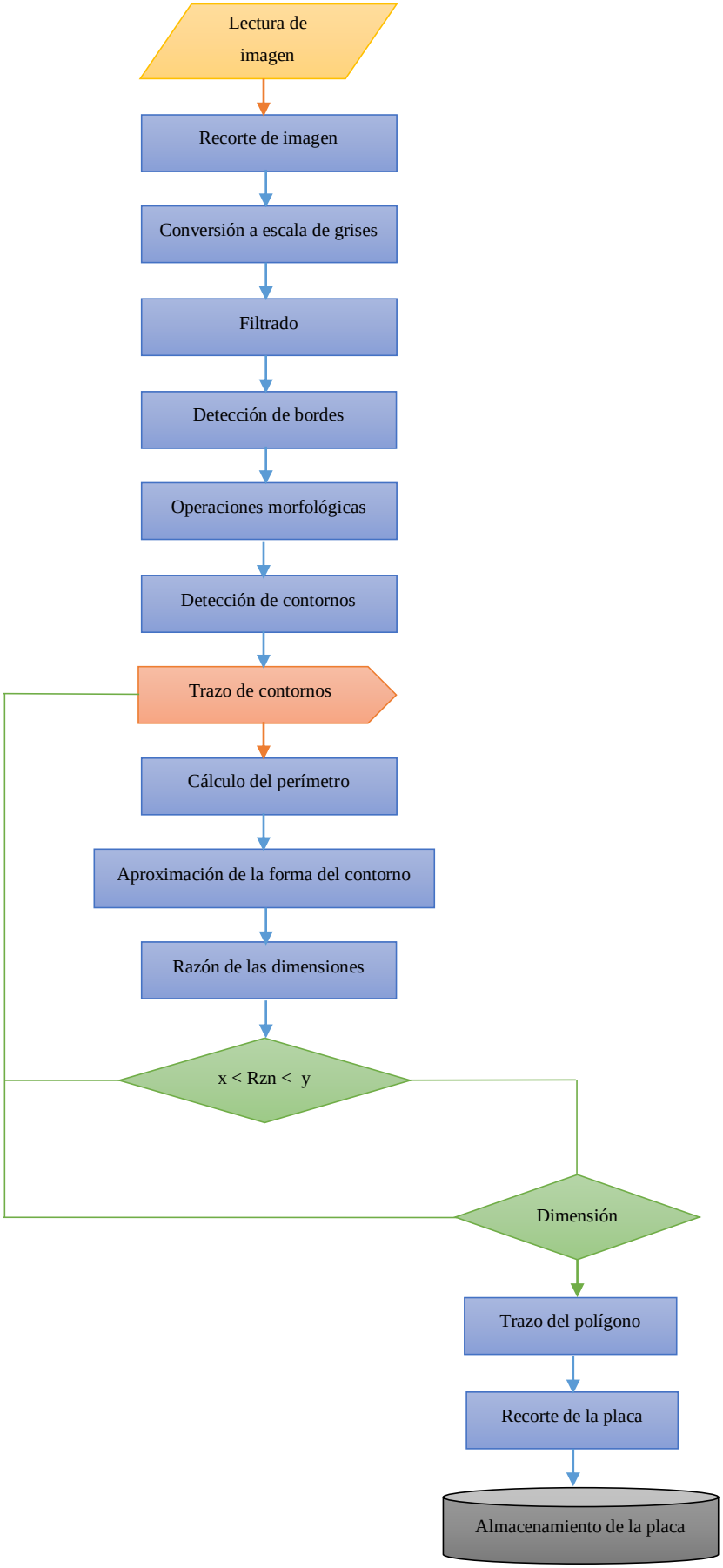


Figura 2-4: Diagrama de flujo de segmentación

Realizado por: Manosalvas C. 2017

2.2.1 Pre-procesamiento de la imagen

A partir de la escena obtenida, se somete a un procesamiento para obtener el área de interés del vehículo.

2.2.1.1 Lectura de imágenes

Con las imágenes almacenadas en memoria es necesario importarlas para someterles a su procesamiento, para ello se ejecuta el comando *imread*. Se especifica el nombre con el que se va a trabajar dentro del programa de Python (*nbr_img*), seguido del comando que da lectura a la imagen almacenada en memoria especificando el ella su nombre y extensión (*cv2.imread ('img_fnt . extensión')*).

nbr_img = cv2.imread ('img_fnt . extensión')

2.2.1.2 Recorte

A priori se conoce la dimensión en píxeles que la imagen posee, es por ello que para eliminar rasgos o sitios de ella que no van a representar significativamente información útil para su procesamiento se procede a recortar ciertas zonas de la imagen, sabiendo que la ubicación de la placa vehicular generalmente está en el centro del vehículo orientada hacia abajo. Al tener una fotografía de 3280 x 2464 se puede omitir de ellas las partes laterales y la superior.

Tomando en cuenta que el origen de coordenadas (0,0) de la imagen se encuentra en el vértice superior izquierdo, como primer argumento de la función de Python se introduce el punto de inicio del recorte hasta su incremento en el eje de las abcisas y de manera análoga en el eje de las ordenadas, es decir limitando el lugar donde se ubica la placa vehicular como se muestra en la Figura 5-2 en la zona delimitada en amarillo.

Figura 5-2: Recorte de la imagen.

Realizado por: Manosalvas C. 2017.

2.2.1.3 Conversión a escala de grises

Computacionalmente hablando es más fructífero trabajar con imágenes en escala de grises que imágenes a color BGR o YUW, de esta manera el procesamiento póstumo se hace de mejor manera ya que operar sobre una imagen con niveles de contraste entre 0 a 255 está por sobre la manipulación de los canales de una escena a color.

Para el paso de color a escala de grises se ejecuta el comando:

cv2.cvtColor('img_fnt',cv2.COLOR_BGR2GRAY). Donde la función *cv2.cvtColor* tiene como argumentos la imagen que será sometida al cambio, y la sentencia que transforma los canales RGB a una escala de grises. El resultado de ejecutar este comando a una imagen se muestra en la Figura 6-2.

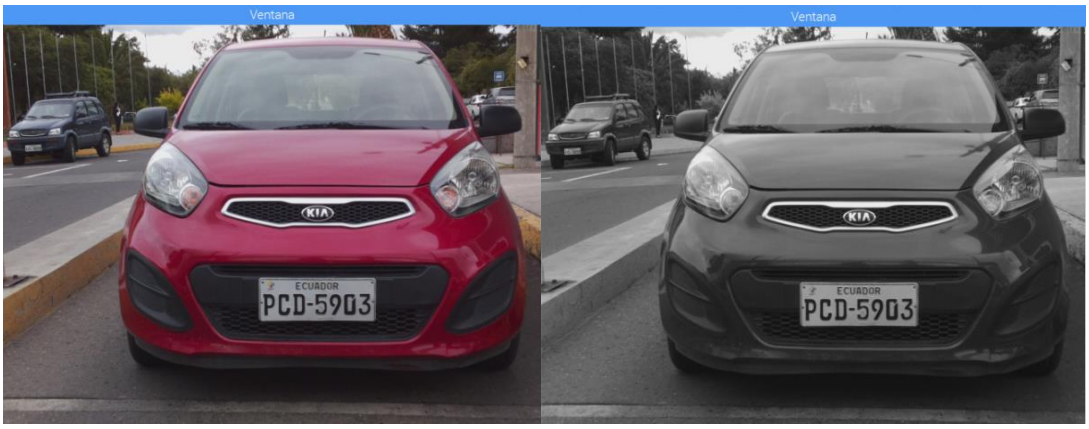


Figura 6-2: Conversión a escala de grises.

Realizado por: Manosalvas C. 2017

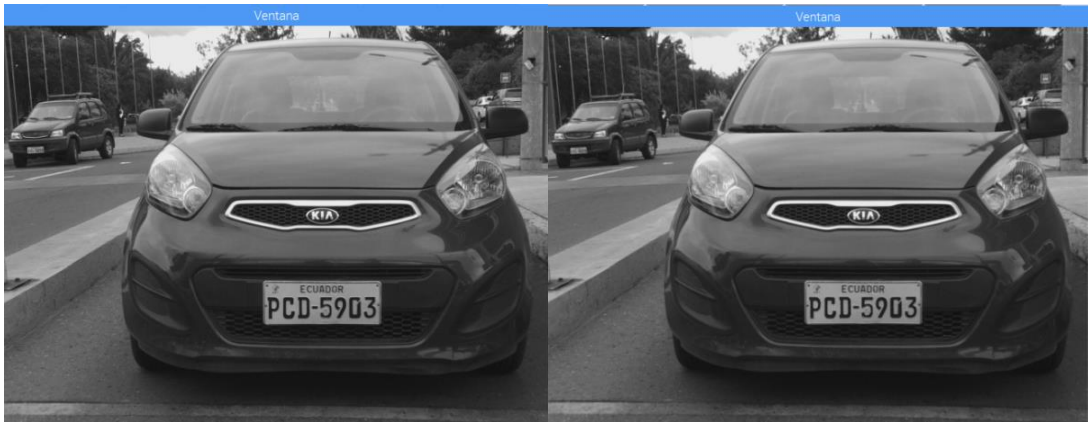
2.2.1.4 Filtrado de la imagen

En el proceso de filtrado se pretende eliminar, resaltar o destacar detalles en una imagen de una manera selectiva. Muchos de los filtros que se aplican en el procesamiento de imágenes sirven para el suavizado, es decir para reducir al máximo las variaciones de intensidades en la vecindad de un píxel o simplemente para la eliminación de ruido modificando los píxeles de diferente intensidad al de sus vecinos.

En este caso es necesario eliminar el ruido que se presenta en la imagen originado muchas veces por la variación de la iluminación, condiciones de temperatura o simplemente por la naturaleza de los dispositivos de captura. Se ha aplicado un filtro de Gauss el mismo que funciona al igual que un filtro de la media, pero a diferencia de éste, en lugar de calcular la media de cada uno de los píxeles vecinos se aproxima a una distribución Gaussiana. Si se considera la media igual a cero la función de trasformación de Gauss se muestra en la Ecuación 1-2.

$$\boldsymbol{T(x,y)} = e^{-\frac{(x^2-y^2)}{2\pi\delta^2}} \qquad \text{Ecuación 1-2}$$

El resultado de aplicar un suavizado a una imagen depende del tamaño de la máscara que se le aplica, en esta ocasión se ha aplicado una 5x5 la misma que no suaviza y elimina el ruido del cuadro obtenido. En la Figura 7-2 se aprecia una imagen filtrada por medio de Gauss con dos tamaños de máscara diferentes.



(a) (b)

Figura 7-2: Filtro de Gauss. (a) Máscara 3x3. (b) Máscara 9x9

Realizado: por Manosalvas C. 2017.

El comando ejecutado en Python para aplicar el filtro Gaussiano es: *cv2.GaussianBlur(img_fnt,(m x n),0).* Los argumentos que maneja *GaussianBlur* son la imagen fuente, el tamaño de la matriz de transformación y la desviación estándar que por general toma un valor de cero.

2.2.1.5 Detección de bordes

La detección de bordes resulta de mucha utilidad para la detección de objetos y segmentación de una imagen. Se ha usado el detector de bordes por medio del algoritmo de Canny.

Este algoritmo está considerado como uno de los mejores métodos de detección de contornos mediante el empleo de máscaras de convolución y basado en la primera derivada. Los puntos de contorno son como zonas de píxels en las que existe un cambio brusco de nivel de gris. En el tratamiento de imágenes, se trabaja con píxels, y en un ambiente discreto, es así que en el algoritmo de Canny se utiliza máscaras, las cuales representan aproximaciones en diferencias finitas. (Valverde, 2011, p.01)

A partir de la imagen filtrada que se ha obtenido anteriormente, se aplica el comando *cv2.Canny* el cual devuelve los contornos en donde no varía la intensidad de los píxeles como se muestra en la Figura 8-2.

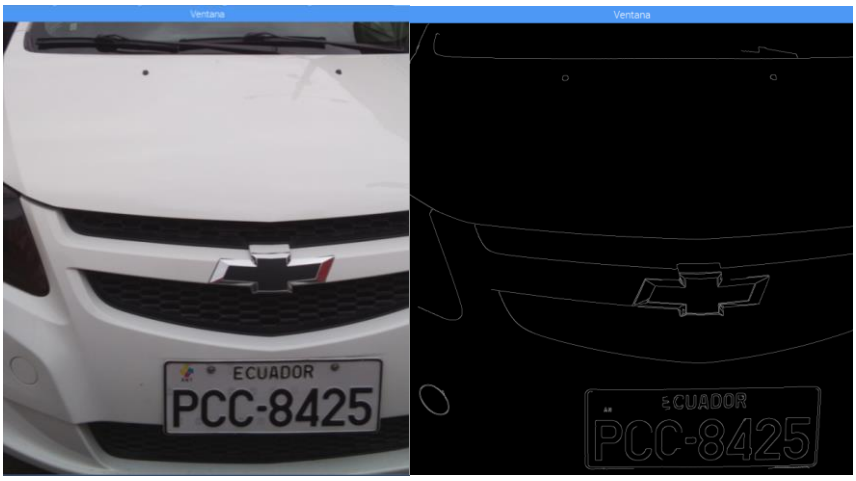


Figura 8-2: Detección de bordes de Canny.
Realizado por: Manosalvas C. 2017.

Se ha aplicado el detector de bordes para predecir y dibujar sobre la imagen el lugar en donde se ubica la placa.

2.2.1.6 Closing de la imagen

Para mejorar la calidad y detección de la placa vehicular se ha aplicado la operación de dilatación seguida de la erosión en la misma. Este proceso en conjunto se conoce como closing el cual es útil para cerrar pequeños agujeros o bordes dentro de los objetos que se encuentran en primer plano o del mismo modo puntos negros presentes que alteren los bordes antes obtenidos.

Al aplicar esta operación morfológica se acentúan y unen los todos los bordes presentes que presentan discontinuidad en la imagen de esta manera se clarifican y se definen estos para posteriormente detectar sus contornos. El comando correspondiente de OpenCV es:

cv2.morphologyEx(img_fnt,cv2.MORPH_CLOSE,kernel), donde los argumentos que se especifican son: la imagen fuente, el comando para especificar la operación morfológica y finalmente la dimensión de la matriz para realizar la erosión y la dilatación de la imagen; se ha considerado una matriz tamaño 6x6, la cual es suficiente para acentuar los bordes detectados, como se muestra en la Figura 9-2.

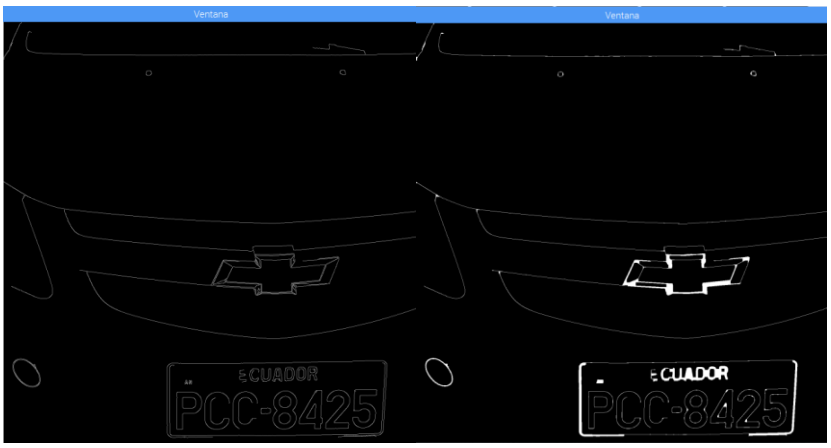


Figura 9-2: Closing de la imagen.

Realizado por: Manosalvas C. 2017.

2.2.1.7 Detección de contornos

Para detectar los contornos existentes en la imagen, se ha precisado encontrar los lugares en donde existe un cambio en los niveles de gris lo que supone la existencia de o la silueta del objetivo. Tomando como base los bordes ya detectados en el proceso anterior, se aplica el comando *cv2.findContours()* que lleva como primer argumento la imagen sobre la cual se quiere operar,

seguido del modo de relación entre ellos y por último el argumento que permite especificar el método de compresión que se usará para generar un contorno.

Se debe incluir una variable de precisión con la que se va a hallar un contorno, bajo algunas pruebas realizadas con diferentes imágenes se ha optado que esta tome un valor de 0.07, para que se aproxime la forma del contorno a través de una figura cerrada. En otras palabras la precisión especificada predice una forma dentro de otra con el menor número de vértices posibles. Se puede observar en la Figura 10-2 una imagen encerrada con dos niveles de precisión distintos.

Figura 10-2: Aproximación poligonal de un contorno

Realizado por: http://docs.opencv.org/3.1.0/dd/d49/tutorial_py_contour_features.html

Los contornos que se han hallado en la imagen del automóvil varían en tamaño y número, debido a que los rasgos de cada vehículo son diferentes y muchas veces poseen accesorios u otros objetos que son detectados como bordes y posteriormente como contorno. En la Figura 11-2 se aprecia los diferentes bordes que fueron hallados y se muestran encerrados bajo un polígono con la precisión antes especificada, cada uno de ellos tiene la probabilidad de ser la placa del vehículo.

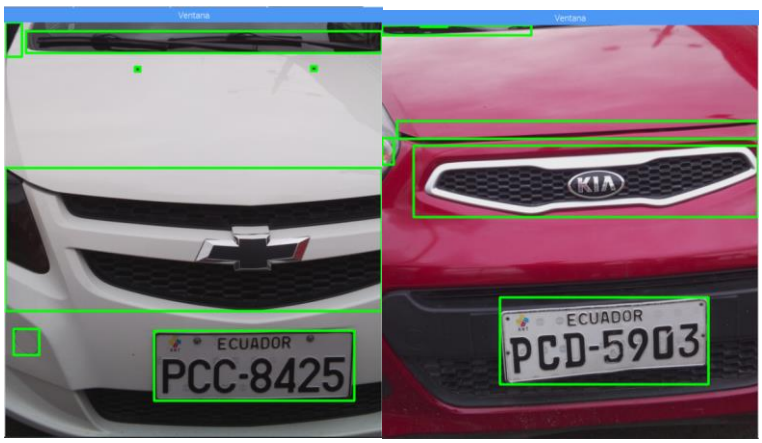


Figura 11-2: Posibles placas vehiculares.

Realizado por: Manosalvas C. 2017.

Tras la aparición de falsos positivos en la imagen, es decir que existen regiones encerradas por el polígono que no corresponden a la placa vehicular, es necesario eliminar estas con un proceso de depuración. En este caso se aplicarán dos filtros en la escena; el primero de ellos se basa en la dimensión de la placa la misma que se conoce a priori y no varía en ningún caso bajo los reglamentos establecidos para el tamaño de la misma. Al conocer que se tiene una longitud de 404 mm de largo por 154 mm de largo, se puede hacer una relación entre estas medidas (Ecuación 2-2), la misma que es independiente de la distancia en que se encuentre el objetivo.

$$\boldsymbol{R} = \frac{lrg}{anc} \qquad \text{Ecuación 2-2}$$

Dónde:
lrg = Largo de la placa
anc = Ancho de la placa
R = Relación obtenida de la placa

El resultado de aplicar este primer filtro en base a la relación establecida se muestra en la Figura 12-2.

Figura 12-2: Filtrado por relación de tamaño.
Realizado por: Manosalvas C. 2017.

A pesar de que el primer filtro en uno de los casos ha eliminado por completo los falsos positivos, se tienen contornos que coinciden con la relación establecida anteriormente, es por eso que se precisa aplicar otro filtro con relación a su longitud y ancho, pero esta ocasión en lugar de relacionar con una operación se hace una comparación entre sus dimensiones. Se puede prever los píxeles que ocupará la placa vehicular dentro de la imagen. Bajo este criterio se ha depurado

la escena, obteniendo simplemente los contornos de interés encerrados en el polígono. Ver Figura 13-2.

Figura 13-2: Detección de la placa vehicular.

Realizado por: Manosalvas C. 2017.

2.3 Segmentación de la placa vehicular

Al ya tener identificada la placa del vehículo se debe extraer esta de la escena principal, para posteriormente someterla a un procesamiento similar al realizado anteriormente con el objetivo de que los caracteres expuestos sean lo más legibles posibles para su reconocimiento. El diagrama de flujo correspondiente a todo este proceso se muestra en la Figura 14-2.

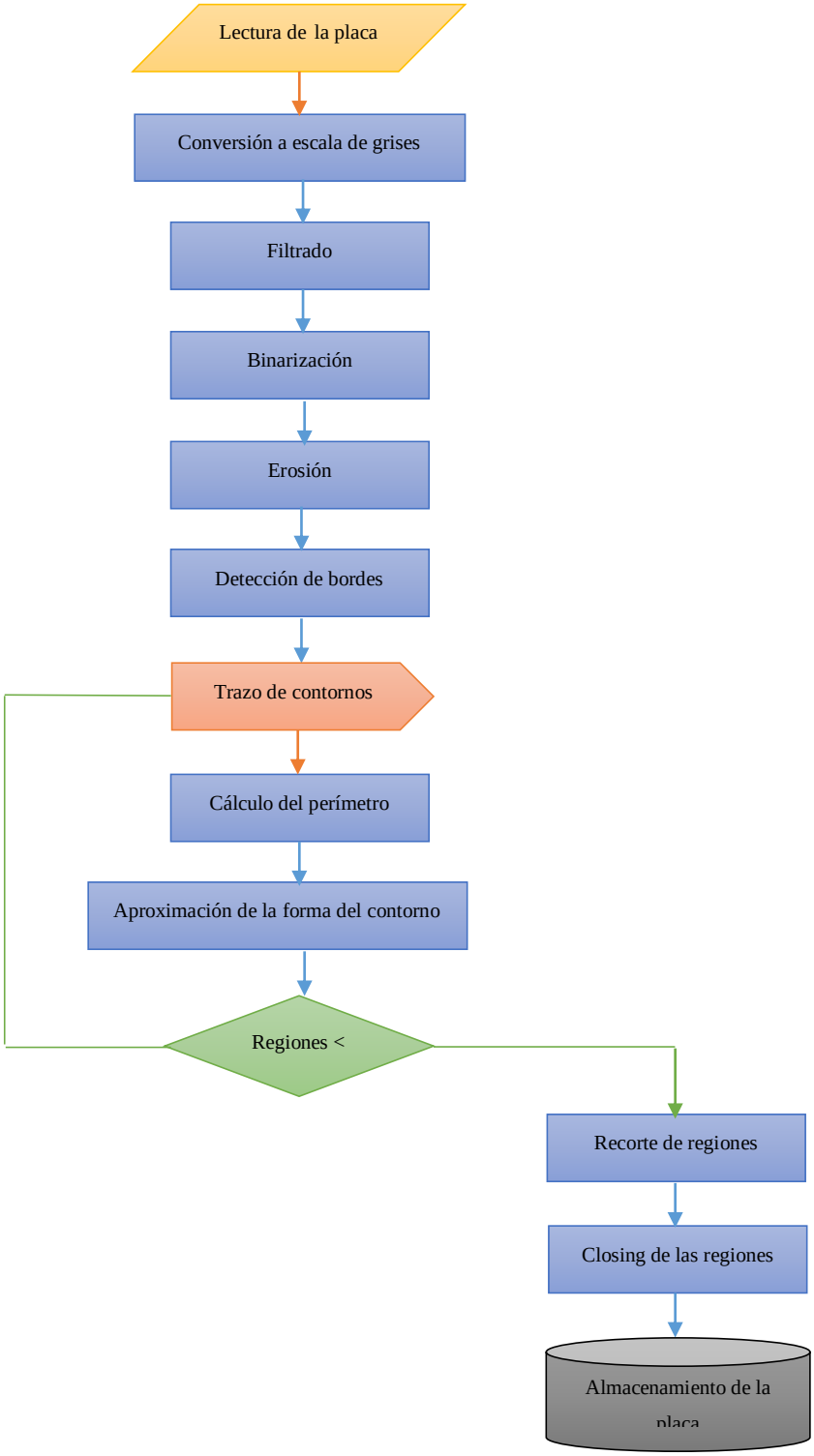


Figura 14-2: Diagrama de flujo procesamiento de la placa

Realizado por: Manosalvas C. 2017

2.3.1 Pre-procesamiento de la imagen segmentada

Como se ha mencionado, la legibilidad de los caracteres alfanuméricos es de suma importancia para que el algoritmo implementado en lo posterior presente la menor cantidad de errores al momento de reconocerlos. Se almacena en memoria la placa correspondiente al vehículo, a partir de ello se inicia con su procesamiento.

2.3.1.1 Conversión a escala de grises

Se opera sobre la imagen de la placa como se hizo en el procesamiento de la escena obtenida por la cámara. A partir de la imagen segmentada se cambia a escala de grises como se aprecia en la Figura 15-2. Es necesario prescindir del polígono que delimita la placa del vehículo con el propósito de eliminar información irrelevante.

Figura 15-2: Conversión a escala de grises de la placa.

Realizado por: Manosalvas C .2017.

2.3.1.2 Filtrado de la imagen

Se precisa eliminar ruido en la imagen obtenida en la placa, de la misma manera resulta conveniente suavizarla aplicando un filtro de Gauss con una matriz de trasformación de dimensión 5x5. El resultado se aprecia en la Figura 16-2.

Figura 16-2: Suavizado de la placa

Realizado por: Manosalvas C. 2017

2.3.1.3 Binarización

Computacionalmente trabajar niveles definidos de blanco (255) y negro (0) favorece al reconocimiento de la placa ya que segmenta el fondo de los caracteres a extraer. El inconveniente de este proceso es hallar el umbral adecuado para que no se pierdan rasgos o caso contrario permanezcan atributos inútiles en la placa vehicular. Definir un valor de *T*, presentaría errores en la segmentación, ya que las condiciones de iluminación, contraste de la imagen, distancia del objetivo o la legibilidad de los caracteres, entre otros, alteran dicha variable. Para evitar este problema el uso de la binarización por medio del algoritmo de Otsu resulta muy conveniente puesto que este elige el umbral óptimo tomando en cuenta la dispersión de los niveles de gris de cada pixel. Lo que da la ventaja de trabajar con un valor variable sin dependencia de los factores nombrados anteriormente. El resultado de aplicar este algoritmo a la placa vehicular se denota en la Figura 17-2.

Figura 17-2: Binarización por el Algoritmo de Otsu.
Realizado por: Manosalvas C. 2017.

El comando correspondiente para la binarización es *cv2.threshold()* que lleva como argumentos la imagen fuente, el valor que se le asignará a los valores que superen o se encuentren por debajo del umbral en este caso 0 y 255. Se especifica por medio de comandos de OpenCV el tipo de binarización como se señala anteriormente se aplica el algoritmo de Otsu, con los comandos *cv2.THRESH_OTSU+cv2.*

2.3.1.4 Erosión

Con el objetivo de acentuar los bordes de los caracteres se ha aplicado la operación morfológica de erosión con una matriz de transformación de dimensión 4x4, la Figura 18-2 muestra el resultado de esta operación. La desventaja que ha presentado es que las formas que no representan un carácter o número de la misma forma se han acentuado y aumentado en dimensiones debido a la naturaleza de esta operación.

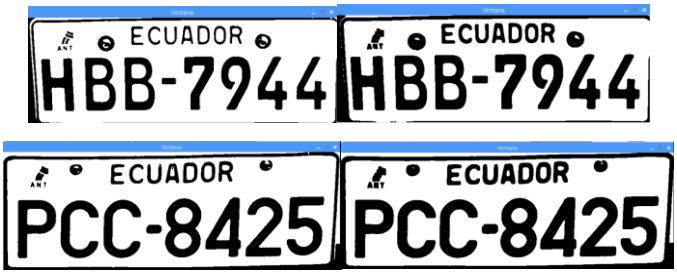


Figura 18-2: Erosión de los caracteres de la placa.

Realizado por: Manosalvas C. 2017.

2.3.1.5 Detección de bordes y contornos

Para identificar cada una de las formas en negro que constituyen la placa, entre ellas los caracteres y números que son el objetivo a analizar, es necesario hallar sus bordes y contornos. De esta manera se logra encerrar en un polígono cada una de ellas usando el proceso similar al hecho con la escena del automóvil. En consecuencia, primero se usa el algoritmo de Canny para detectar los bordes presentes en la imagen, tal como denota la Figura 19-2.

Figura 19-2: Bordes de la placa vehicular.

Realizado por: Manosalvas C. 2017.

Posterior a la detección de bordes, se encierra en un polígono cada contorno de la placa. Ver Figura 20-2.

Figura 20-2: Contornos de la placa

Realizado por: Manosalvas C. 2017.

2.3.1.6 Eliminación de falsos positivos

La detección de todos los bordes han mostrado todos los rasgos que se encuentran en la placa del vehículo, dentro de ellos están los caracteres que serán reconocidos posteriormente. Es necesario eliminar estos falsos positivos para evitar un mal reconocimiento de los caracteres alfanuméricos. En consecuencia, rasgos como la palabra ECUADOR, ANT y su logo correspondiente, el guion que separa los números de las letras, y por último los tornillos que sujetan a la placa. Todos estos detalles son eliminados con un filtrado por tamaños.

Se conoce que los caracteres alfanuméricos son los que gobiernan la placa en tamaño, por ende cada borde detectado que no cumpla con dimensiones establecidas será encerrado en un polígono exceptuando por obvias razones las letras y números. La separación de estos caracteres se aprecia en la Figura 21-2.

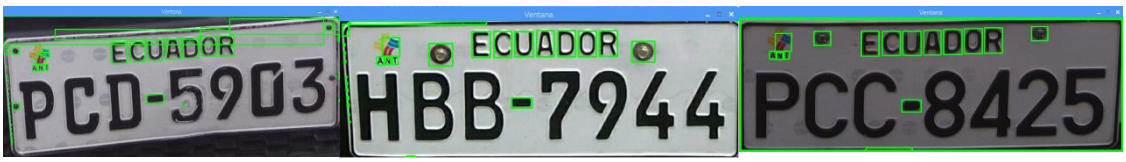


Figura 21-2: Selección de falsos positivos.

Realizado por: Manosalvas C. 2017.

Al tener demarcadas las zonas donde se ubican los detalles que no son de utilidad, se trabaja en cada una de ellas sobre una imagen binaria, en donde se somete a un closing con una matriz de transformación lo suficientemente grande para eliminar cada uno de los falsos presentes (Figura 22-2). En este caso se ha definido una matriz de tamaño 50x50.

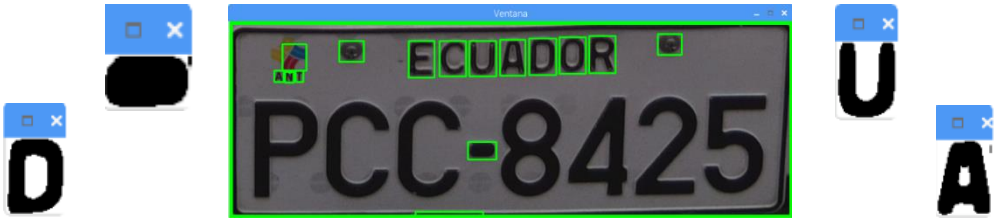


Figura 22-2: Eliminación de falsos positivos

Realizado por: Manosalvas C. 2017.

Posterior a esto, cada imagen sometida al closing regresa a su sitio original en coincidencia con la coordenada que le corresponde dentro de la imagen, lo que da un fondo completamente blanco, como se muestra en la Figura 23-2.

Figura 23-2: Caracteres alfanuméricos de la placa.

Realizado por: Manosalvas C. 2017.

2.4 Implementación del algoritmo KNN

Se ha implementado el algoritmo conocido como los vecinos más cercanos, el cual necesita de una fase de entrenamiento para predecir los caracteres y números de la placa vehicular.

2.4.1 Fase de entrenamiento del algoritmo

Al ser un algoritmo supervisado, es necesario tener en memoria cada una de las clases a las que pertenece cada carácter. Con el fin de entrenar al algoritmo se ha usado la imagen mostrada en la Figura 24-2, la cual se asemeja al tipo de letra que tienen los caracteres de la placa del automóvil. La ventaja es que al tener solo letras mayúsculas disminuye el tiempo que dura en entrenar y reconocer cada grafía. Estos caracteres son guardados en lo posterior como un array de vectores.

Figura 24-2: Clases de entrenamiento del algoritmo.

Realizado por: Manosalvas C. 2017.

Se carga la imagen al algoritmo, y se opera sobre ella con el objetivo de eliminar un posible ruido, suavizar y binarizar la imagen a la inversa, es decir letras blancas sobre un fondo negro para que el complemento de cada imagen corresponda a la misma binarizada a la inversa. Luego de ellos se declara cada uno de los caracteres que van a ser válidos para su reconocimiento, en este caso es todo el alfabeto exceptuando la letra Ñ, y los números del 0 - 9. Esto se lo realiza por medio del comando *ord* el cual devuelve el valor ASCII del carácter que esté en su argumento. Ver Figura 25-2.

```
CaracValid = [ord('0'), ord('1'), ord('2'), ord('3'), ord('4'), ord('5'), ord('6'), ord('7'), ord('8'), ord('9'),
              ord('A'), ord('B'), ord('C'), ord('D'), ord('E'), ord('F'), ord('G'), ord('H'), ord('I'), ord('J'),
              ord('K'), ord('L'), ord('M'), ord('N'), ord('O'), ord('P'), ord('Q'), ord('R'), ord('S'), ord('T'),
              ord('U'), ord('V'), ord('W'), ord('X'), ord('Y'), ord('Z')]
```

Figura 25-2: Declaración de caracteres ASCII.

Realizado por: Manosalvas C. 2017.

Para reconocer cada uno de los caracteres dentro de la imagen de entrenamiento se definen los contornos en cada una de ellas dibujando un rectángulo que demarque la zona o pixeles que corresponden a una letra o número, seguido a esto se recorta cada una de ellas para ser redimensionada, lo que va a favorecer a la consistencia del reconocimiento y el almacenaje de los vectores. El proceso realizado se muestra en la Figura 26-2. Cada carácter se guarda como un vector de números flotantes en un archivo plano usando el comando *numpy.savetxt()*, lo que almacena en sí es la intensidad de los píxeles de cada dígito y letra. Este archivo va a servir en más adelante para la comparación entre la placa y las clases preestablecidas.

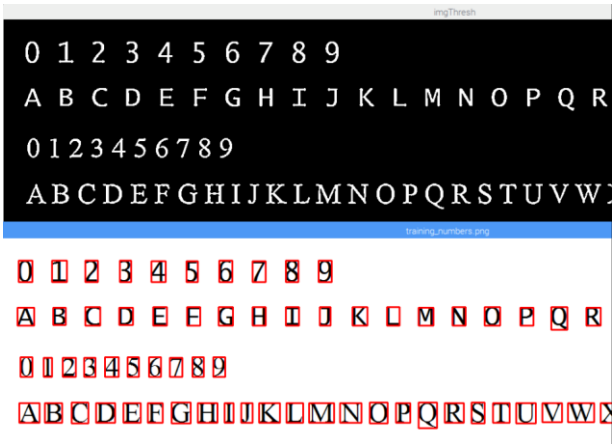


Figura 26-2: Reconocimiento de caracteres de entrenamiento.

Realizado por: Manosalvas C. 2017.

2.4.2 Comparación de caracteres

Al ya contar con el algoritmo entrenado, y los vectores que caracterizan a cada uno de los números y letras en dependencia de su intensidad de píxeles se procede a comparar las clases con un objetivo, para ello se comienza cargando en el programa el archivo plano a través del comando *numpy.loadtxt()*. Se inicializa el objeto knearest, por medio del comando *cv2.KNearest_create()*, para luego obtener las características del entrenamiento con *cv2.Knearest.train()* el que lleva en su argumento el archivo de los vectores antes obtenidos. Ver Figura 27-2.

```
try:
     VecTra = np.loadtxt("vectores.txt", np.float32) # Lectura de archivo plano.
except:
    print "Error. No existe el archivo de entrenamiento\n" # Error de entrenamiento
    os.system("pause") # Delay del sistema
    return
kNearest = cv2.ml.KNearest_create() # Creación del objeto K-Nearets
kNearest.train("vectores.txt")
```

Figura 27-2: Comandos de comparación de caracteres.

Realizado por: Manosalvas C. 2017

2.5 Diseño de la base de datos

Se ha planteado que todos los caracteres de la placa de cada automóvil se almacenen en una base de datos, para que estos se dispongan en la red. Como primer paso es gestionar dicha información con el software necesario, es por ello que se ha elegido a MySQL debido a su fácil adaptación al entorno de Debian.

2.5.1 MySQL

MySQL es un gestor de base de datos de código abierto, considerado como uno de los más populares del mundo debido a su fácil integración con aplicaciones web y su unificación con varias interfaces de programación entre las cuales destacan C, C++, Java, Python, entre otros.

2.5.1.1 Creación de base de datos

Se debe crear una base de datos que aloje las placas de los vehículos, esto se lo realiza a través del comando: *create database "nombre_bd";*

El comando *create database* permite introducir una base de datos en memoria, seguido del nombre que se le va a dar a la misma, en este caso lleva como identificación *vehículos.* Se puede comprobar si la base creada se aloja en el sistema por medio del comando *show databases;* este proceso se lo aprecia en la Figura 28-2.

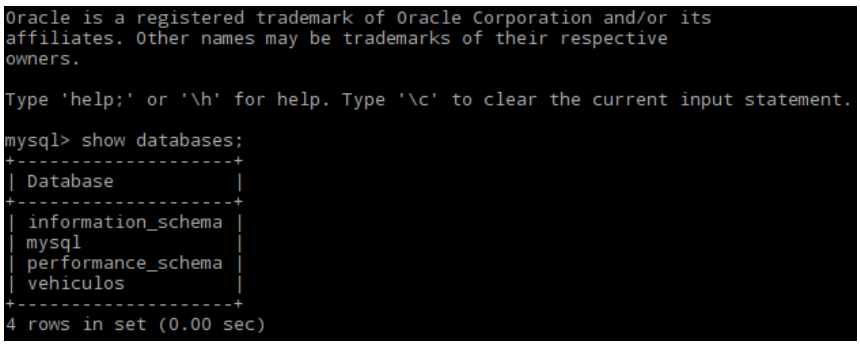


Figura 28-2: Creación de la bases de datos.

Realizado por: Manosalvas C. 2017.

Dentro de la base introducida se ha creado la tabla PLACAS que contiene el campo CARACTERES de tipo varchar extensión 10 el que recepta texto y cadenas. Se permite que sea nulo en el caso de algún error en la lectura de la placa. (Ver Figura 29-2.)

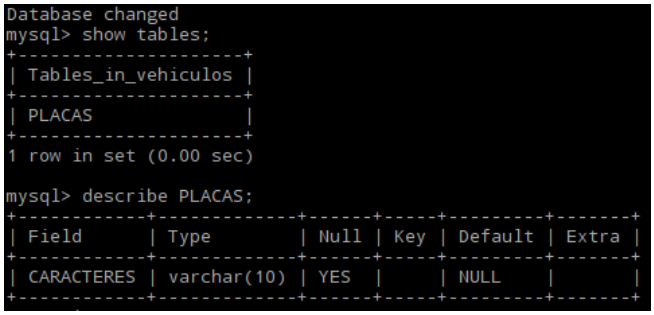


Figura 29-2: Descripción de la BD.

Realizado por: Manosalvas C. 2017.

Debido a la flexibilidad e integración que caracteriza a MySQL, la conexión entre esta y el lenguaje de programación Python resulta relativamente sencilla basta con ejecutar el comando *MySQLdb.connect*, el que lleva como argumentos el lugar donde se recopilan los datos, el usuario, el password y por último la base de datos a donde se dirige. La creación del cursor es primordial para poder apuntar a cada campo dentro de la base el que permite escribir y borrar datos. Finalmente con sentencias SQL se insertan o borran los datos adquiridos. Los comandos correspondientes a este proceso se especifican en la Figura 30-2.

```
bd = MySQLdb.connect("localhost","root","liguista","vehiculos")
cursor = bd.cursor()
sql = "INSERT INTO PLACAS (CARACTERES) VALUES('%s')" %(licPlate.strChars)
```

Figura 30-2: Comandos de conexión a la BD.

Realizado por: Manosalvas C. 2017.

2.5.2 Anclaje a la red

El uso de servidores gratuitos que se ofrece en la Web, proveen funciones limitadas en cuanto a acceso remoto, memoria, administración, velocidad de conexión, saturación de datos y demás. A parte de que su dominio tiende a ser ligeramente extenso, difícil de memorizar y lo que se pretende es que esta dirección sea sencilla de recordar para tener un fácil acceso. Proaño y Suárez muestran en la Tabla 1-2 algunos servidores que se encuentran en la red y sus características.

Tabla 1-2: Tipos de hosting web.

	GRATUITO			PAGO	
Sitio web	Eshost	Hostinger	Eshost	GoDaddy	Hosting de Pago
Dominio	controlado	controlado	libre	libre	Libre
Dirección	http://eshost.com.ar/	http://www.hostinger.es/	http://eshost.com.ar/	https://es.godaddy.com/	https://www.hostingdepago.com/
Espacio de disco duro	1000 MB	2GB	5GB	100GB	5GB
Transferencia mensual	10 GB	100GB	300GB	Ilimitado	Ilimitado
Soporta	FTP, PHP, MySQL, Correo	FTP, PHP, MySQL, Correo	PHP y MySQL	Correo, FTP, MySQL, PHP, Javascript, Apps	Correo, FTP, MySQL, PHP, Javascript, Apps
Publicidad	No	No	No	No	No
Costo mensual	0$	0$	2.50$	1.00$	3.95$

Fuente: Proaño J. y Suarez M., 2016.

Estos han sido los motivos por los cuales se ha optado por usar un servidor de paga en donde se pueda alojar la página web, sin que ocasione ninguno de los problemas antes mencionados. La dirección del dominio que se ha otorgado es 143.95.106.248; bajo esta dirección IP se ha trabajado en la creación de la página web y anclaje de la base de datos.

Para realizar la conexión remota entre la base local con la anclada en la red, es necesario simplemente otorgar los permisos necesarios dentro del panel de control del servidor web; en otras palabras se especifica las direcciones IP de las máquinas en donde se alojan los datos e ingresando a la dirección http://143.95.106.24/cpanel (servidor web), se accede a "*Manage Databases & Users" - "Remote MySQL®"*, donde se configura todos los parámetros antes citados. (Figura 31-2)

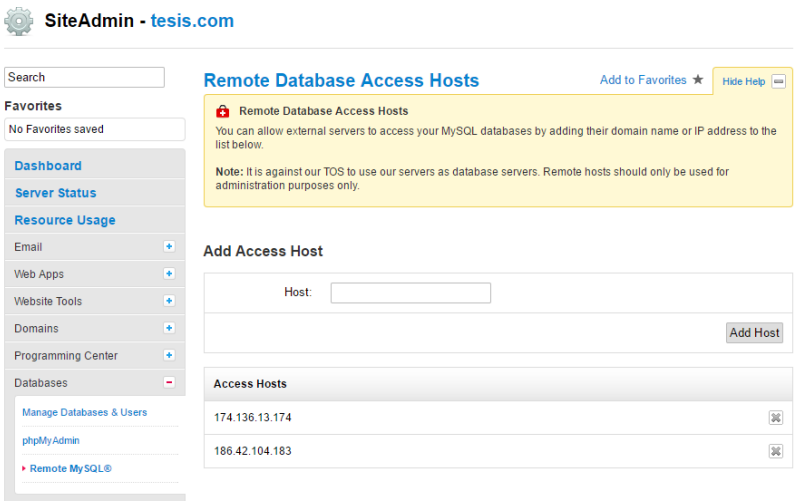


Figura 31-2: Host de acceso remoto

Realizado por: Manosalvas C. 2017.

Al tener configurado el servidor web se debe especificar dentro de la Raspberry la dirección en donde se van a alojar los caracteres de la placa. Los comandos correspondientes a este proceso se detallan en la Figura 32-2.

```
bd = MySQLdb.connect("143.95.106.248","tesiscom","*****","vehiculos")
cursor = bd.cursor()
sql = "INSERT INTO PLACAS (CARACTERES) VALUES('%s')" %(licPlate.strChars)
sql = "DELETE FROM WHERE LIKE ('%s')" %(licPlate.strChars)

try:
    cursor.execute(sql)
    bd.commit()

except:
    bd.rollback()

bd.close()
```

Figura 32-2: Conexión de la base de datos a la red.

Realizado por: Manosalvas C. 2017.

La conexión de MySQL lleva como argumentos la dirección del servidor, el nombre de usuario que se provee al configurar el acceso remoto, su contraseña y por último la base de datos. Las sentencias que corresponden a la escritura de los caracteres se realizan mediante el comando INSERT especificando la tabla, el campo a donde se dirigen y su cadena de caracteres que se ha almacenado en la variable *licPlate.strChars.* Por otro lado, para eliminar las placas que residen en la web, es decir para los vehículos que salen del campus se usa el comando DELETE. Se hace una comparación con el comando WHERE LIKE y se elimina de los registros.

2.5.2.1 Diseño de la página web

La interfaz gráfica que se ha de mostrar al usuario para que se acceda a los registros de las placas vehiculares debe ser sencilla de operar. La misma se ha implementado por medio del lenguaje PHP. Este permite crear páginas web de contenido dinámico con acceso a información almacenada en una base de datos brindando la posibilidad de una programación orientada a objetos.

Accediendo a la dirección IP descrita anteriormente se muestra la página web donde se alojan las placas. Para esto se ha usado un textbox para ingresar los caracteres que se desean buscar y el botón que entregará el mensaje de que si el auto correspondiente a la placa tipeada, se encuentra dentro del campus.

La interfaz gráfica que se muestra en la dirección IP dada, se refleja en la Figura 33-2.

Figura 33-2: Interfaz gráfica del sitio web.

Realizado por e: Manosalvas C. 2017.

CAPÍTULO III

3 MARCO DE RESULTADOS Y DISCUSIÓN

3.1 Escenario

El sistema creado ha sido instalado y probado en la puerta principal, acceso número uno de la Escuela Superior Politécnica de Chimborazo ubicada en la Panamericana Sur Km ½. Ver Figura 1-3.

Figura 1-3: Ingreso principal ESPOCH.

Fuente: http://bibliolatino.com/portadas/espoch/1.jpg

3.1.1 Funcionamiento y estado actual del sistema de ingreso

El grupo Procelec, empresa dedicada al diseño e instalación de sistemas de control fue la encargada de implantar un sistema de ingreso en las puertas de ingreso del campus. Este contaba inicialmente con cuatro brazos mecánicos dos para el ingreso y dos para la salida accionados por cilindros hidráulicos, dispensadores de tickets donde se imprimían caracteres y códigos de barras correspondientes a cada vehículo, el que era detectado por el sensor de presencia ubicado en el piso, por último la lectura de tarjetas electromagnéticas y accionamiento del pulsador ubicado al alcance de los conductores permitía la apertura de cada brazo.

Este sistema que actualmente se encuentra implementado en la puerta principal de la Espoch, tras varias indagaciones acerca del tema con el Departamento de Mantenimiento y Desarrollo Físico y Jefe de Seguridad, ha manifestado que éste fue instalado hace aproximadamente 12 años y hoy en día se encuentra obsoleto debido a los siguientes factores:

- Inexistencia de las tarjetas de acceso.
- Deterioro y destrucción de las lectoras de tarjeta.
- Deterioro y destrucción de los brazos mecánicos.
- Inexistencia del protocolo de comunicación entre el software y el sistema.
- Falta de mantenimiento preventivo y correctivo del sistema en su totalidad.

El factor más importante por el cual este sistema ha quedado en desuso fue el de la falta de un estudio de proyección y crecimiento del parque automotor; es decir, para aquella época el número de automóviles que ingresaban al campus era menor a la cantidad que hoy en día se puede apreciar, motivo por el cual la Agencia de Tránsito solicitó que este sistema debía salir de operación debido a los problemas de movilidad y tráfico que este ocasionaba en las calles aledañas al ingreso.

Todos estos inconvenientes han opacado la implementación en su totalidad del sistema propuesto, motivo por el cual las pruebas realizadas se han hecho con la intervención del ser humano.

3.1.2 Ubicación del sistema embebido

De los cuatro brazos mecánicos que existían inicialmente, dos han permanecido ilesos uno corresponde a la entrada y otro a la salida, y es ahí donde se va a instalar este sistema por lo que es necesario conocer las medidas para conocer la ubicación idónea.

El ancho de cada carril es de 3.50 metros y los brazos tienen una longitud de 3 metros medidos desde su eje lo que supone un margen de error con respecto al centro de la placa de alrededor 25 centímetros. El diámetro de éste es de aproximadamente de 8.6 centímetros. Finalmente la altura en posición horizontal del brazo con respecto al piso es de 90 centímetros, todas estas medidas se aprecian en la Figura 2-3.

Figura 2-3: Medidas de los carriles.

Realizado por: Manosalvas C. 2017.

El lente de la cámara se ha ubicado justo en el centro del carril es decir a 1.5 m para así abarcar completamente el sitio donde se ubica la placa del vehículo. Otro factor que favorece a la correcta toma del objetivo, es el ángulo de apertura de 120 grados que ofrece el sensor. La carcasa se ha diseñado con una ligera inclinación de para ubicar la cámara con un ángulo de depresión de 110 grados. Ver Figura 3-3.

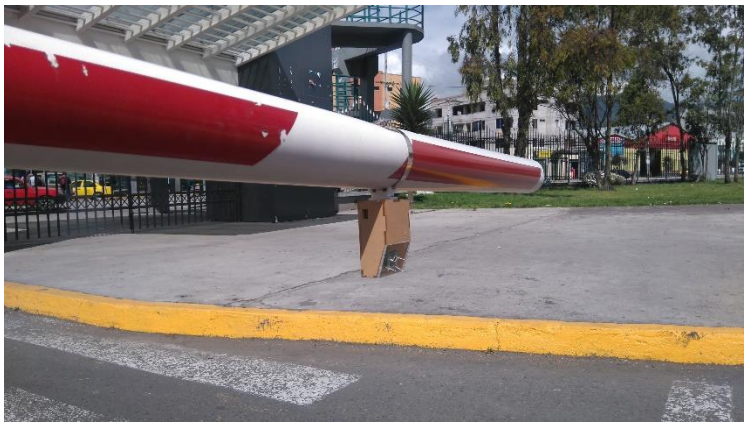


Figura 3-3: Posición del sistema en el brazo mecánico.

Realizado por: Manosalvas C. 2017.

3.2 Pruebas del algoritmo KNN

Para evaluar cómo se comporta el algoritmo de los vecinos más cercanos KNN, y comprobar que el entrenamiento del mismo ha sido satisfactorio se realizan pruebas con imágenes de caracteres que se asemejan a una placa vehicular.

Se han tomado letras y números al azar de diferentes fuentes de tal forma que se emulen a una placa vehicular, es decir en total siete caracteres alfanuméricos. En la Figura 4-3 se muestra el resultado de la evaluación:

Figura 4-3: Evaluación del algoritmo KNN

Realizado por: Manosalvas C. 2017.

El entrenamiento del algoritmo arroja una eficiencia del 100% ya que de las muestras aleatorias tomadas de todo el alfabeto y los números correspondientes del 0-9,se han reconocido satisfactoriamente en condiciones perfectas para que se ejecute el algoritmo, es decir no existe ilegibilidad, discontinuidades, cambio de iluminación, etc. Lo que no sucede al identificar una placa vehicular ya que al examinar cada una de ellas pueden existir factores que desfavorezcan en la comprensibilidad de los caracteres. El tiempo cronometrado en el reconocimiento es de 2.65 segundos, desde que lee la imagen fuente hasta visualizar la variable que la contiene.

3.2.1 Pruebas de reconocimiento de placas de vehículos

Al comprobar que el funcionamiento del algoritmo implementado no presenta errores, ahora se deben tomar modelos reales de las placas, para esto se aplica un método estadístico de muestreo

probabilístico aleatorio donde todos los vehículos tienen la probabilidad de ingresar al campus pero no lo hacen al mismo.

Dentro de la ciudad de Riobamba según datos de la Dirección de Gestión de Movilidad, Tránsito y Transporte se han matriculado alrededor de 60000 vehículos en el año 2016, lo que representa al universo para obtener la muestra por medio de la Ecuación 1-3 que denota la fórmula matemática para calcular el tamaño de una muestra para datos globales.

$$\boldsymbol{n} = \frac{Z^2.N.P.Q}{E^2(N-1)+Z^2.P.Q} \qquad \text{Ecuación 1-3}$$

Donde:

n = Tamaño de la muestra (Valor a encontrar).
Z = Nivel de confianza.
N = Población o universo (Número de vehículos).
P = Probabilidad de acierto (0.5).
Q = Probabilidad de error (0.5).
E = Error muestral. (10% según la distribución normal correspondiente a un valor de Z = 1.65)

Aplicando la ecuación se obtiene:

$$\boldsymbol{n} = \frac{Z^2.N.P.Q}{E^2(N-1)+Z^2.P.Q}$$

$$\boldsymbol{n} = \frac{1.65^2(60000)(0.5)(0.5)}{0.1^2(60000-1)+1.65^2(0.5)(0.5)}$$

$$\boldsymbol{n} = 67.98 \approx 68$$

Conociendo que la muestra óptima para el universo es de 68 se toma como prueba el 25% de los mismos a consecuencia de que se considera que este porcentaje es el que ingresa al campus en un tiempo determinado.

Para un mejor reconocimiento de los caracteres se han planteado ciertos requerimientos que deberían cumplir las placas de los vehículos, entre los cuales destacan:

- Ubicación correcta de la placa vehicular como dictan las normas de la Agencia Nacional de Tránsito.
- No poseer marcos, adornos o extras que dificulten la apreciación de los caracteres.
- No poseer placas deterioradas.
- Legibilidad de los caracteres de la placa.

En base a la muestra probabilística obtenida la Tabla 1-3, se muestran los resultados obtenidos con 17 vehículos que ingresaron al campus:

Tabla 3-1: Resultados de implementación de KNN.

TABLA DE RESULTADOS				
Caracteres de la placa	**Adquisición del algoritmo**	**Caracteres acertados**	**Porcentaje de acierto**	**Tiempo de respuesta**
PCC-8425	PCC8425	7	100%	5.1 s
HBA-8296	HBA8296	7	100%	5 s
HBB-4240	HBBA2A0	5	71.40%	6.33 s
HBB-5719	HBB5719	7	100%	5.2 s
HBA-3931	H8A3931	6	85.70%	6.1 s
HBA-3871	HBA387I	6	85.70%	6. 36 s
PCO-7154	PCO7154	7	100%	6 s
HBB-9896	H889896	5	71.40%	6.45 s
AFI-883	AFI8B3	5	83.33%	6.53 s
HBB-6335	HBB6335	7	100%	5.23 s
HBA-3677	HBA3677	7	100%	5.33 s
HBB-1445	HBB1445	7	100%	5.13 s
HBB-7944	HBB7944	7	100%	6.35 s
PCD-5903	PCD5903	7	100%	6 s
GSE-4826	OSE480	5	71.40%	6.33 s
PBD-6320	PBD6320	7	100%	5.23 s
HBB-3827	HBB3B27	6	85.70%	7 s
HCM-370	IICM370	5	83.33%	6.03 s
			Promedio: 91%	**Promedio:** 5.84 s

Realizado por: Manosalvas C. 2017.

En el Gráfico 1-3 se muestran los tiempos obtenidos en el reconocimiento de las muestras adquiridas, esta da un promedio de 5.84 segundos desde que se lee la imagen fuente hasta almacenar ésta en una variable.

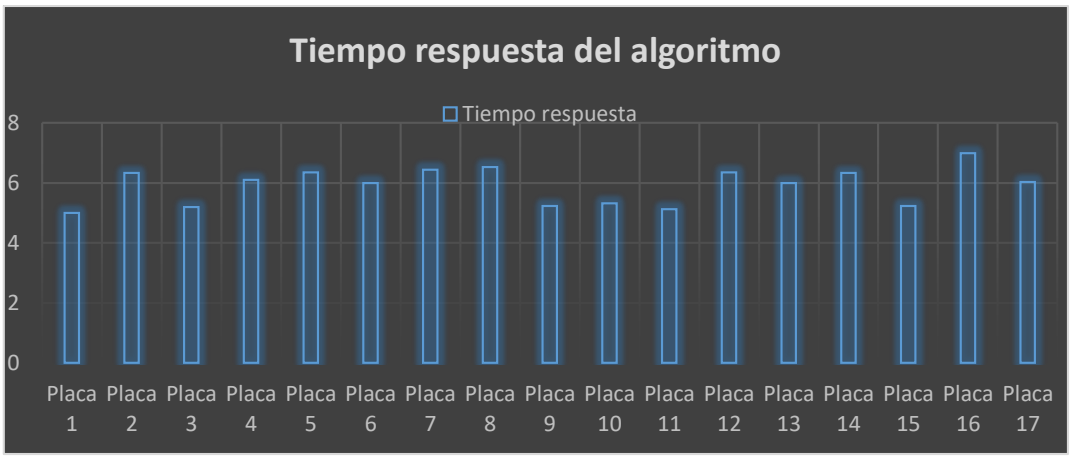


Gráfico 1-3: Tiempo de respuesta.

Realizado por: Manosalvas C. 2017.

El Gráfico 2-3 denota el porcentaje de acierto en la identificación de las placas vehiculares. El promedio que arroja es de 91%, lo que puede establecer que dado un margen de error del 10% el nivel de confianza efectivamente es de 90% con una tolerancia del ± 1%.

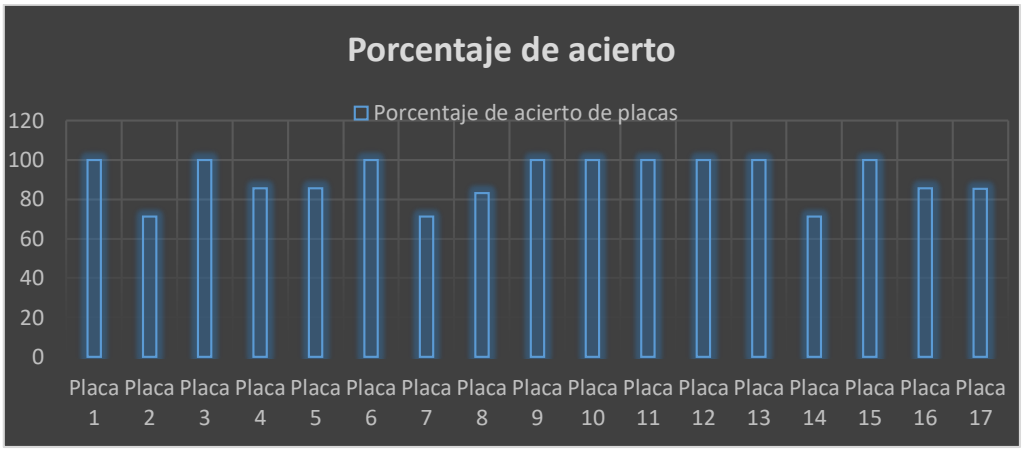


Figura 2-3: Porcentaje de acierto.

Realizado por: Manosalvas C. 2017.

En la Figura 5-3 se muestran algunos de los vehículos, que respondieron en un 100% al reconocimiento de caracteres aplicando el algoritmo KNN, la consola imprime como resultado los caracteres que corresponden al mismo para posteriormente alojar en la base de datos.

Figura 5-3: Reconocimiento satisfactorio de caracteres.

Realizado por: Manosalvas C. 2017

Debido a varios factores señalados anteriormente, varias placas no han sido reconocidas en su totalidad. La Figura 6-3 manifiesta algunos ejemplos de caracteres que no concuerdan con las del vehículo.

```
LOS CARACTERES SON: AFI8B3     LOS CARACTERES SON: IICM370  LOS CARACTERES SON: OSE480
----------------------------   ----------------------------  ----------------------------
```

Figura 6-3: Datos erróneos de la placa.

Realizado por: Manosalvas C. 2017.

3.3 Pruebas de conexión con la base de datos

Se tomó las pruebas correspondientes a la conexión con la web especificando las direcciones IP que la red asigna a cada uno de los módulos Raspberry para que se envíe los datos a través de ella. Desde el CPanel del servidor se modifica las conexiones de cada uno de ellos otorgando los permisos correspondientes para que remotamente se lea y escriba las placas vehiculares.

En la Figura 7-3 se muestra la página web, resultado de realizar la búsqueda de una placa que ingresó al campus.

Figura 7-3: Búsqueda de placa de ingreso.

Realizado por: Manosalvas C. 2017.

Por otro lado las placas de los autos que salieron del campus o que no registran alguna entrada muestran el aviso de la Figura 8-3.

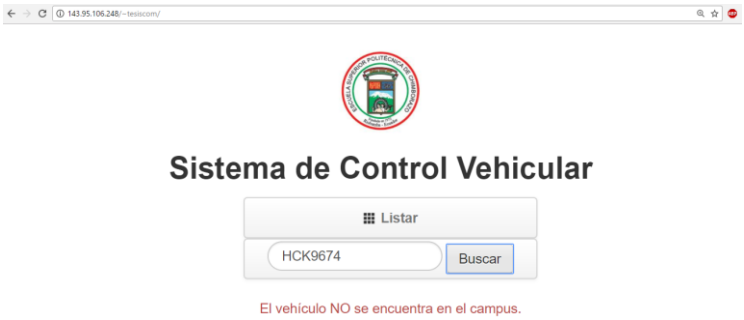


Figura 8-3: Búsqueda de placa de salida

Realizado por: Manosalvas C. 2017.

3.4 Análisis de resultados obtenidos

3.4.1 ¿Se pueden aplicar técnicas de visión por computador para registrar un vehículo?

En la actualidad, muchos sistemas de registro necesitan de cámaras para almacenar datos de un auto a través de sensores o dispositivos externos. El sistema desarrollado se basa en el uso de la placa como principal medio para reconocer y diferenciar a un vehículo de otro, sin tener la necesidad de adquirir recursos adicionales que pueden conllevar a un gasto extra para acceder un beneficio.

Las técnicas de pre-procesamiento, procesamiento y segmentación se consideran como las más relevantes dentro de la visión artificial, ya que estas son aplicadas en su mayoría para diversas tareas dentro de este campo, por lo que el registro de vehículos no ha sido la excepción y sometiendo a técnicas de procesamiento de imagen sobre la escena obtenida del vehículo se ha hallado el sitio en donde se encuentra la placa del mismo, tomando en cuenta el factor iluminación considerado como el más influyente dentro de varios métodos que se aplican en el proceso en general.

Dicho esto, se puede manifestar que efectivamente las técnicas de visión artificial sobre el procesamiento de imágenes resultan de mucha utilidad y pueden sustituir procedimientos que lo puede realizar el ser humano, en el caso de un registro de vehículos se puede optar por obviar al personal que es el encargado de tomar nota para llevar un registro continuo de vehículos dentro del lugar que lo requiera.

3.4.2 ¿De qué manera influyen los algoritmos y librerías de visión artificial para el desempeño del sistema?

OpenCV es considerado como un pilar dentro de la visión artificial debido a su fácil integración con muchas plataformas y lenguajes de programación que le han permitido que su uso se extienda en diversas aplicaciones. Su eficiencia dentro del tratamiento de imágenes se fundamenta en el principio matemático que caracteriza a cada función, por ejemplo la detección de bordes por medio del algoritmo de Canny que usa un operador de primera derivada tanto en vertical como en horizontal para devolver un borde dentro de una imagen. El algoritmo para determinar el valor recomendable de umbral dentro de la binarización se lo hace por medio de métodos estadísticos; el más representativo es el método de Otsu que usa la varianza de los niveles para determinar un valor óptimo.

De esta manera, innumerables librerías y funciones que ofrece OpenCV influyen de manera positiva en el la creación de proyectos que requieran de visión artificial, esto se debe a que procesos complejos se pueden simplificar a pocas líneas de código teniendo un resultado mucho mejor a lo que se pretende.

3.4.3 ¿Cómo se puede usar la red de Internet para registrar los vehículos que ingresan y abandonan el campus?

El auge del concepto del Internet de las cosas ha permitido que tareas cotidianas se realicen por medio de dispositivos tecnológicos que ofrecen conectividad por lo que se ha aprovechado el este fenómeno para alojar los datos extraídos por el sistema dentro de una base de datos en la red. De este modo simplemente accediendo a la dirección proporcionada se conocerá la estancia del vehículo que ingresó al campus de la ESPOCH.

CONCLUSIONES

OpenCV se convierte en una herramienta poderosa dentro del campo de la visión artificial, se puede operar sobre imágenes con procesos complejos en pocas líneas de código. Además de su flexibilidad e integración con varios lenguajes de programación en varios sistemas operativos.

El algoritmo de los vecinos más cercanos no simplemente se usa en el reconocimiento de caracteres, este es aplicado en diversas áreas debido a su sencilla forma de operar y a su bajo consumo de recursos tanto en software como en hardware.

Al ser un algoritmo supervisado, KNN requiere de una fase de entrenamiento exhaustiva, ya que de esto depende el reconocimiento óptimo de los caracteres. Mientras más datos de entrenamiento por clase se implementen mejor será el resultado final.

Se ha implementado el sistema embebido en la puerta principal de la ESPOCH, lo que no limita a que este pueda ser instaurado en los demás ingresos solamente direccionando la base de datos a los “n” sistemas que se requieran.

Las características del hardware de la Raspberry Pi3, limitan de cierta manera la ejecución de los procesos encomendados y mucho más si se trata de procesamiento de imágenes ya que estos requieren de altos gastos computacionales. En contraste a ello, esta tarjeta presenta una fácil integración con otros dispositivos, en este caso el uso de la cámara no genera retardos hasta adquirir las imágenes, como es el caso de las cámaras IP y USB.

Se ha tomado en cuenta las características del escenario en donde se ha puesto en funcionamiento el sistema. El cambio de iluminación es el principal problema con el que tiene que lidiar el procesamiento de la imagen es por eso que la segmentación se basa en la detección de bordes y contornos más no en detecciones de regiones.

El sistema embebido propone una alternativa de solución para otros sistemas de acceso que tienen el mismo objetivo. A diferencia de otros comerciales, no se deben adquirir dispositivos adicionales que permitan registrar un vehículo.

RECOMENDACIONES

La adquisición de un dominio para alojar la base de datos perfeccionará el sistema embebido, puesto que es más fácil recordar una dirección Web que una dirección IP para ingresar a las consultas de las placas vehiculares.

Se deben tomar en cuenta las versiones que son instaladas en el hardware, tanto en el lenguaje de programación Python como en las librerías de OpenCV. Una incompatibilidad de las versiones de estos genera varios conflictos al momento de compilar el programa.

Se puede reemplazar el dispositivo que captura la imagen por otro que dentro de sus características posea estabilizador de imagen, puesto que al tener movimiento de los vehículos muchas veces distorsiona el objetivo generando ruido y deformación.

Un entrenamiento recurrente de las muestras del algoritmo KNN va a mejorar el reconocimiento de los caracteres, puesto que mientras más clases se tengan abarcará más prototipos de distintas letras y números a clasificar.

Se puede tomar como base el trabajo desarrollado para implementar una interfaz que permita ingresar los caracteres de la placa de manera manual, puesto que sólo se muestra en consola un error de lectura para quien administra el programa más no para el usuario.

BIBLIOGRAFÍA

BONILLA MARTÍNEZ, Eduardo Gabriel. *Reconocimiento de caracteres mediante redes neuronales con MatLab.* [En línea]. (Tesis de pregrado). Escuela Politécnica Nacional, Quito – Ecuador. 2005. pp. 21-22. [Consulta: 03 de mayo de 2017]. Disponible en: http://bibdigital.epn.edu.ec/bitstream/15000/5503/1/T2472.pdf

BRADSKI, Gary; KAEHLER Adrian. *Learning OPENCV Computer Vision with the OpenCV Library.* [En línea]. Instituto Tecnológico de Massachusetts, Cambrigde - Estados Unidos. 2008, p. 01. [Consulta: 15 de abril de 2017]. Disponible en: https://books.google.es/books?hl=es&lr=&id=seAgiOfu2EIC&oi=fnd&pg=PR3&dq=opencv&ots=hTM28kfGPa&sig=UFVsPjlcijaWWKPS1n7tpoufW9A#v=onepage&q=opencv&f=false

CAMBRONERO GARCÍA, Cristina; & GÓMEZ MORENO, Irene. *Algoritmos de aprendizaje: KNN & KMEANS.* [En línea]. Universidad Carlos III, Madrid – España. 2009 p. 03 [Consulta: 02 de mayo de 2017]. Disponible en: http://www.it.uc3m.es/jvillena/irc/practicas/08-09/06.pdf

EVANS, Dave. *Internet de las cosas. Como la próxima evolución del Internet lo cambia todo.* [En línea]. Cisco Systems Inc, San José – California. 2011, p. 02. [Consulta: 18 de abril de 2017]. Disponible en: http://www.cisco.com/c/dam/global/es_mx/solutions/executive/assets/pdf/internet-of-things-iot-ibsg.pdf

EZQUEDA ELIZONDO, José. *Fundamentos de Procesamiento de Imágenes.* [En línea]. Universidad Autónoma de Baja California, Tijuana – México. 2002, p. 24 [Consulta: 02 de abril de 2017]. Disponible en: http://www.academia.edu/16801512/Fundamentos_de_procesamiento_de_im%C3%A1genes_digitales

FU, K; et al. : *Robótica: Control, detección, visión e inteligencia.* Ciudad de Juárez – México: McGraw-Hill, 1989. p. 06, 370,376.

MEDINA SÁNCHEZ, Martín; & CONTRERAS HURTADO, Pedro. *Desarrollo de un sistema de acceso para parqueaderos y cobro tarifario utilizando reconocimiento de patrones.* (Tesis de pregrado). Escuela Superior Politécnica de Chimborazo, Riobamba – Ecuador. 2015. pp. 34-39.

PAJARES MARTINSANZ, Gonzalo; & DE LA CRUZ GARCÍA, Jesús. *Visión por computador, imágenes digitales y aplicaciones.* 2da. Ed. Madrid – España: Ra-Ma, 2007, pp. 01-08.

PEREZ, David. *Sistemas embebidos y sistemas operativos embebidos.* [En línea]. Universidad Central de Venezuela. Caracas – Venezuela. 2009, pp. 04-11. [Consulta: 05 de mayo de 2017]. Disponible en: http://www.academia.edu/16523506/Info_Sistemas_Embebidos

PROAÑO VALLEJO, Jessica; & SUAREZ CHANCHAY, Mario. *Implementación de un prototipo de sistema electrónico mediante comunicación inalámbrica para supervisión y detección de inundaciones.* [En línea] (Tesis de pregrado). Escuela Superior Politécnica de Chimborazo, Riobamba, Ecuador. 2016. pp. 21-22. [Consulta: 28 de mayo de 2017]. Disponible en: http://dspace.espoch.edu.ec/bitstream/123456789/6446/1/98T00133.pdf

RUSSELL, Stuart; & NORVIG, Peter. *Inteligencia Artificial: Un enfoque moderno.* Ciudad de Juarez – México: Prentice Hall. 1996, p. 597.

SÁNCHEZ FERNANDEZ, Carlos; & SANDONÍS CONSUEGRA, Víctor. *Reconocimiento Óptico de Caracteres (OCR).* [En línea]. Universidad Carlos III, Madrid – España. 2010 pp. 01-05. [Consulta: 26 de abril de 2017]. Disponible en: http://www.it.uc3m.es/jvillena/irc/practicas/08-09/09.pdf

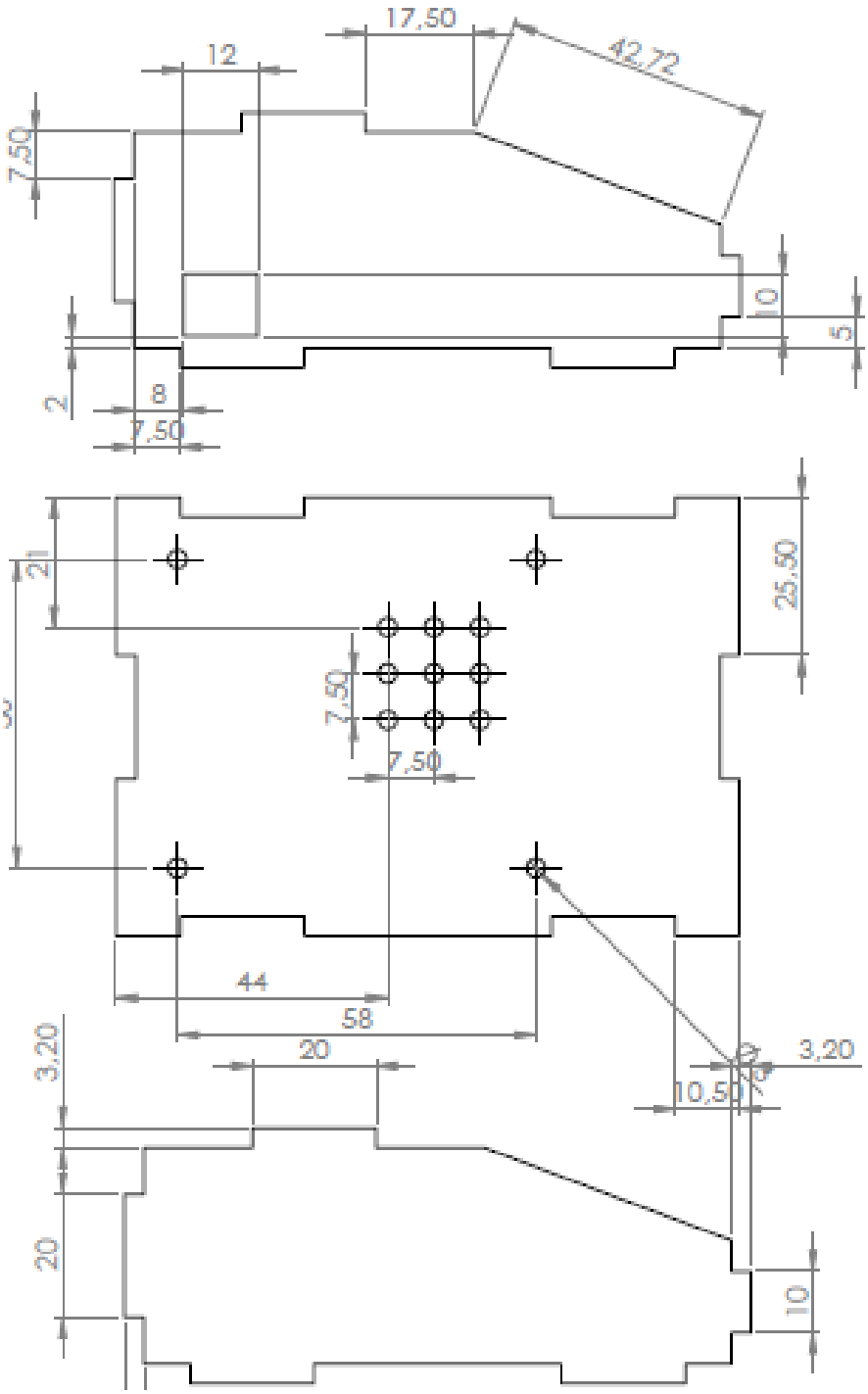
SUCAR, Enrique; & GOMEZ, Giovani. *Visión Computacional.* [En línea]. Instituto Nacional de Astrofísica, Óptica y Electrónica, Puebla – México. 2002, p. 12 [Consulta: 01 de abril de 2017]. Disponible en: http://ccc.inaoep.mx/~esucar/Libros/vision-sucar-gomez.pdf

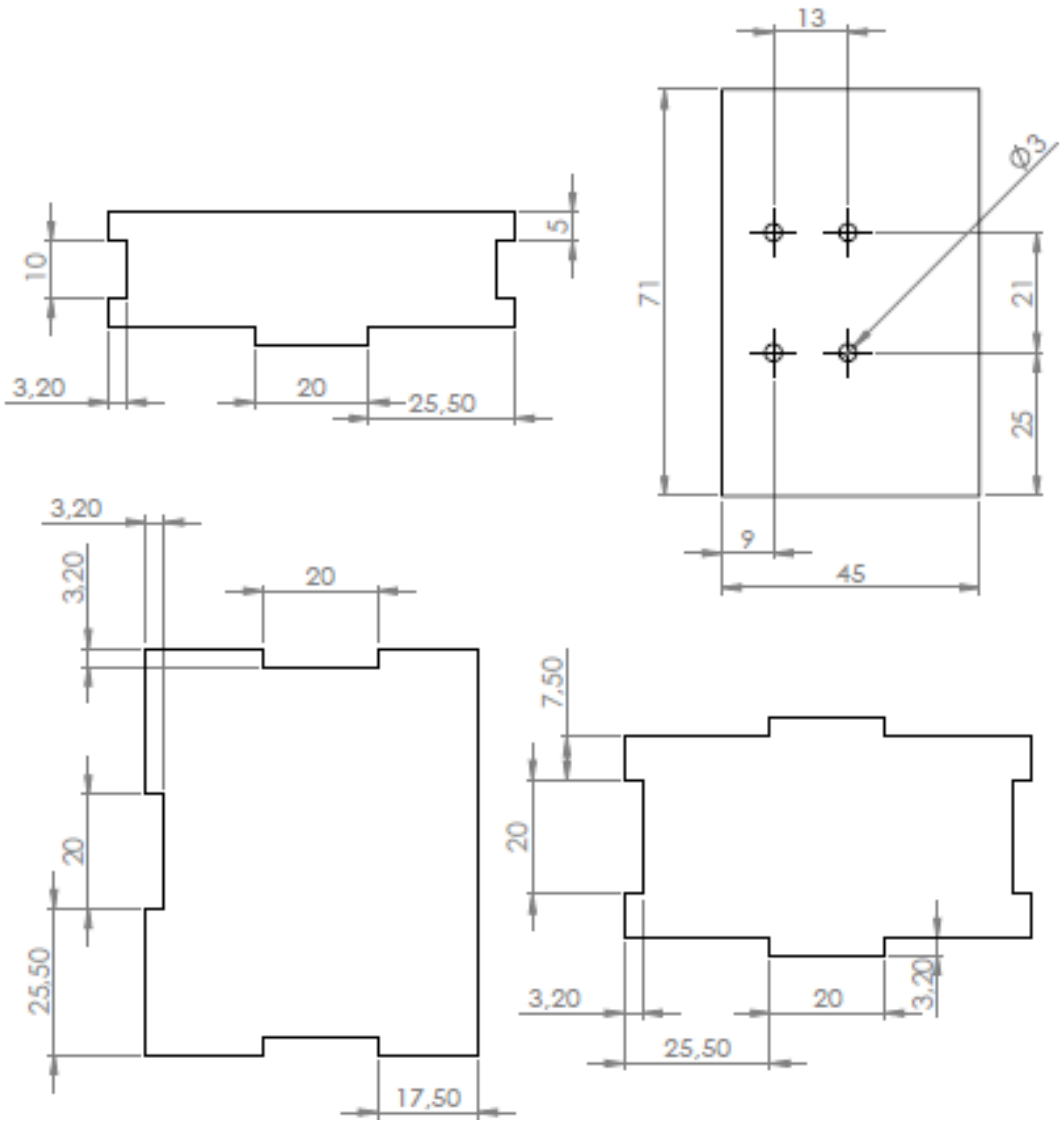
VALVERDE REBAZA, Jorge. *Detección de bordes mediante el algoritmo de Canny*. [En línea]. Universidad Nacional de Trujillo, Trujillo – Perú, 2011, p. 01. [Consulta: 15 de mayo de 2017]. Disponible en: https://www.researchgate.net/profile/Jorge_Valverde-Rebaza/publication/267240432_Deteccion_de_bordes_mediante_el_algoritmo_de_Canny/links/548dd1ae0cf225bf66a5f636/Deteccion-de-bordes-mediante-el-algoritmo-de-Canny.pdf

VÉLEZ, José; et al. : *Visión por computador.* [En línea]. Universidad Rey Juan Carlos, Madrid – España. 2003, p. 128 [Consulta: 09 de abril de 2017]. Disponible en: http://www.visionporcomputador.es/libroVision/VisionPorComputador.pdf

ANEXO A

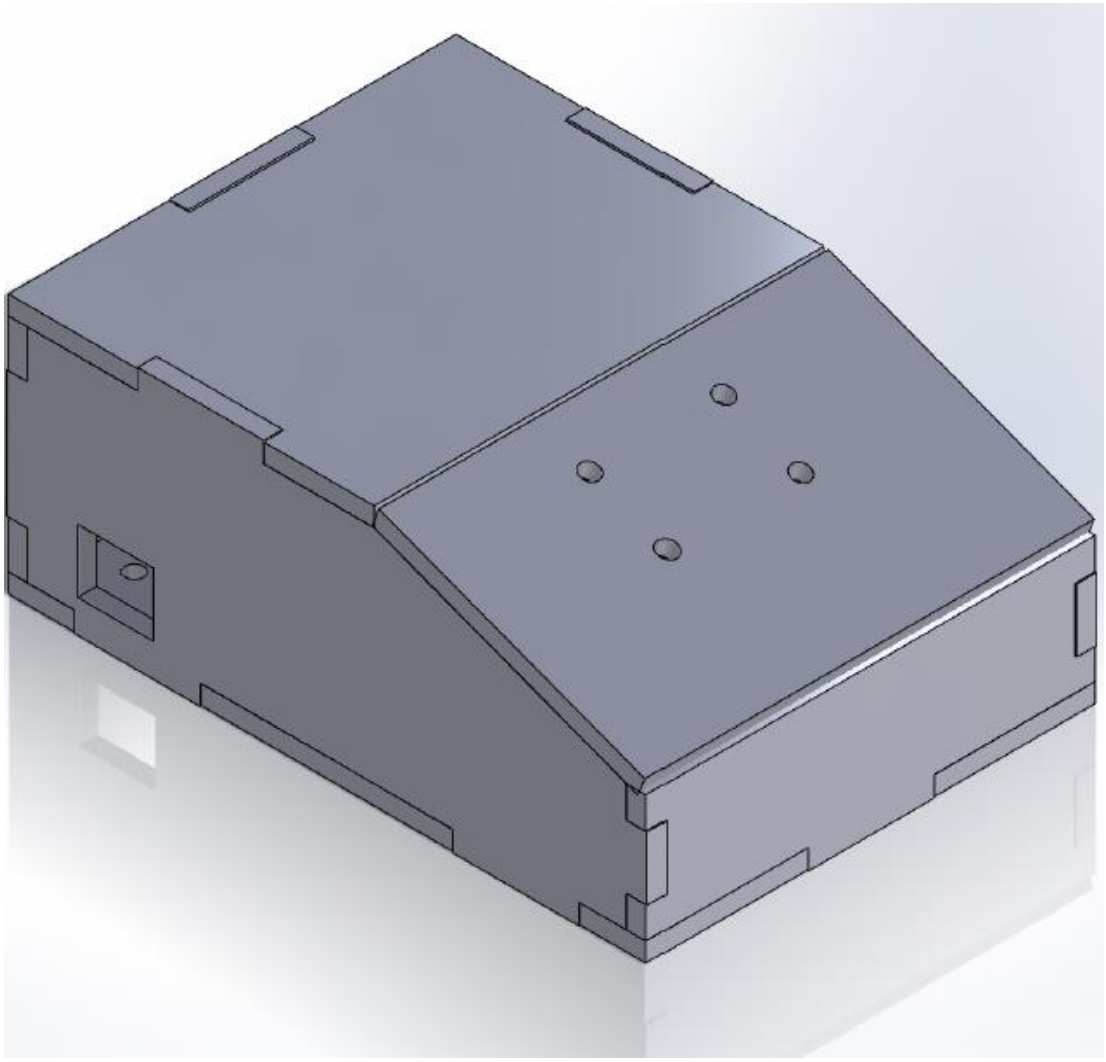
DISEÑO DE LA CARCASA DEL SISTEMA.




13
Ø3
71
21
25
9
45
5
10
3,20
20
25,50
3,20
3,20
20
20
25,50
17,50
7,50
20
3,20
20
3,20
25,50

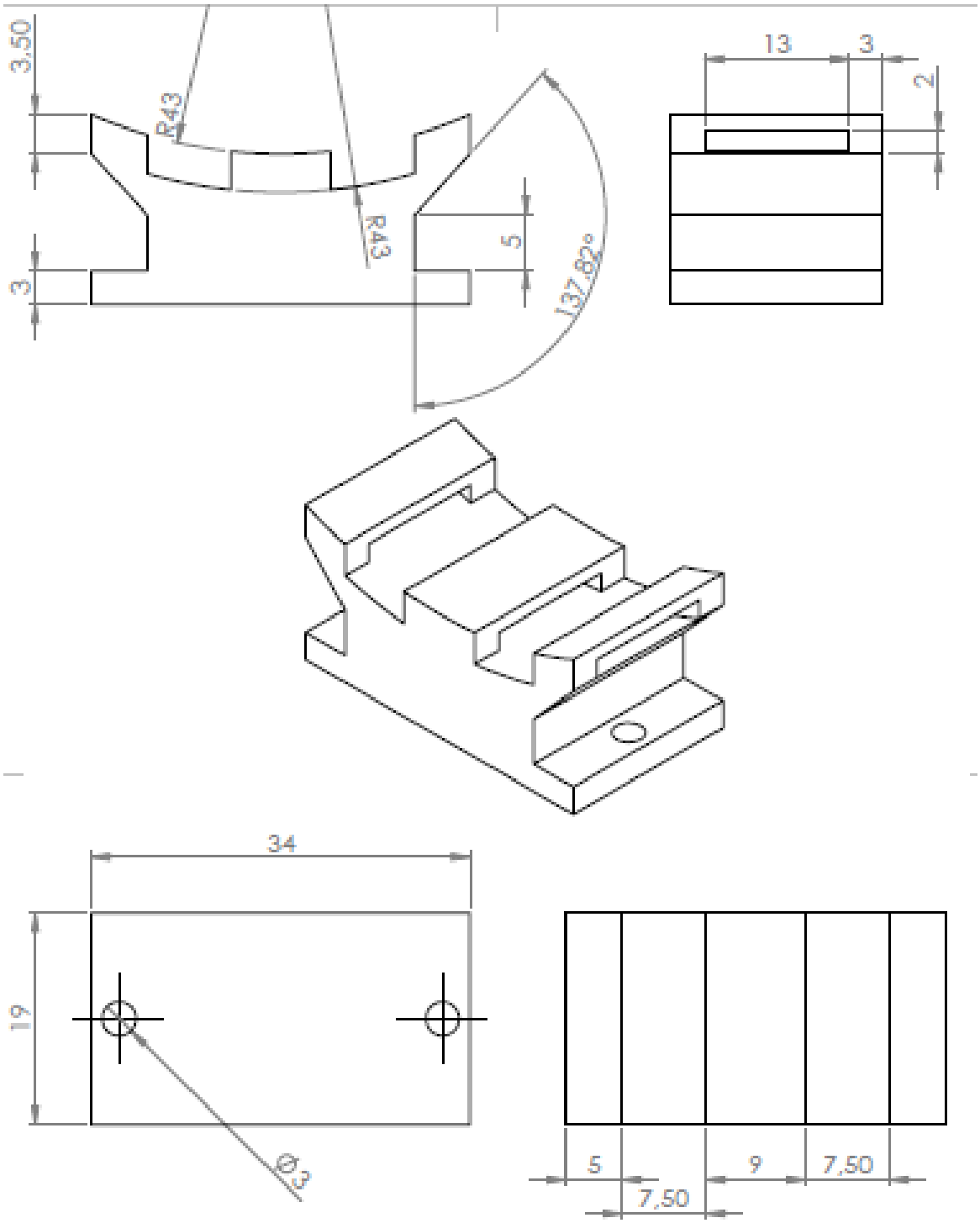
ANEXO B

DISEÑO EN 3D DE LA CARCASA.



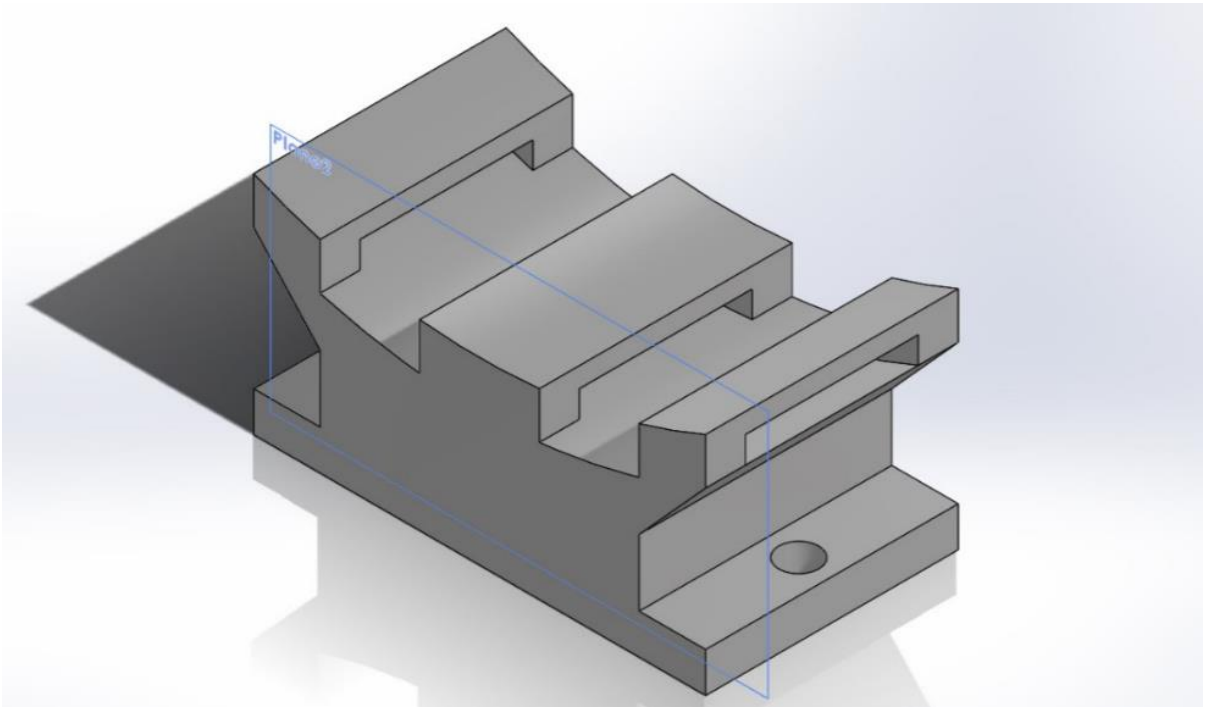
ANEXO C

DISEÑO DEL ACOPLE.



ANEXO D

DISEÑO EN 3D DEL ACOPLE



ANEXO E

CORTE DE PIEZAS DE LA CARCASA EN ACRÍLICO.

ANEXO F

IMPRESIÓN DEL ACOPLE.

ANEXO G

MANUFACTURA COMPLETA.