Centro Universitario de la Defensa en la Escuela Naval Militar

TRABAJO FIN DE GRADO

Desarrollo de un sistema portátil de reconocimiento de matrículas para controles de seguridad

Grado en Ingeniería Mecánica

ALUMNO: Luis Morales Ridruejo

DIRECTOR: Miguel Rodelgo Lacruz

CURSO ACADÉMICO: 2016-2017

Universidade Vigo

Centro Universitario de la Defensa
en la Escuela Naval Militar

TRABAJO FIN DE GRADO

Desarrollo de un sistema portátil de reconocimiento de matrículas para controles de seguridad

Grado en Ingeniería Mecánica

Intensificación en Tecnología Naval

Cuerpo General

Universidade Vigo

RESUMEN

Los sistemas de control de acceso tradicionales, basados en pases y documentos, están siendo reemplazados por nuevos sistemas, basados en reconocimiento automático de imágenes. El presente trabajo pretende desarrollar un sistema portátil de reconocimiento de matrículas para controles de seguridad, que pueda mejorar la seguridad en el acceso a la Escuela Naval Militar. El sistema cuenta con una cámara y un pulsador que al ser accionado muestra visualmente si la matrícula está en la base de datos de vehículos autorizados. Para ello se ha realizado un estudio de las técnicas de detección y reconocimiento y se ha seleccionado la librería de reconocimiento de matrículas OpenALPR. Como plataforma hardware se ha utilizado una Raspberry Pi con su cámara infrarroja Pi NoIR, y se ha desarrollado una aplicación Pyton que automáticamente toma una fotografía, reconoce la matrícula, la coteja en una base de datos MySQL y muestra el resultado por medio de luces led. Por último, se han realizado pruebas de validación para la comprobación de su buen funcionamiento. El sistema desarrollado permitirá monitorizar los movimientos de vehículos a través de un control de seguridad para mejorar la seguridad del mismo y obtener información valiosa para la gestión de las instalaciones.

PALABRAS CLAVE

Detección, Matrícula, Raspberry Pi, Python, Picamera.

AGRADECIMIENTOS

A mi familia, en especial a mis padres, Isabel y Luis, por su educación y apoyo incondicional.

Al tutor de este trabajo, Don Miguel Rodelgo Lacruz, por su gran implicación y ayuda.

A mis compañeros de promoción por la ayuda prestada a lo largo de los años en la Escuela Naval Militar, en especial a Alejandro Carnicero Solanas por enseñarme todo lo que ha podido acerca del apasionante mundo de la programación.

Por último, y no por ello menos importante, a mi novia, Lidia, por el apoyo y los ánimos que me ha dado en estos años.

CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1 INTRODUCCIÓN

Existen muchas clases de seguridad, tantas como actividades sea capaz de realizar el ser humano. Este trabajo se centrará en la **seguridad física,** que se define como aquella que impide el paso al sistema a cualquier persona no acreditada. Se realiza a través de aplicaciones y procedimientos específicos cuyo objetivo es bloquear el paso a dichas personas. Por otro lado, la **seguridad lógica** que, a través de encriptación de códigos o códigos de autenticación (como pudieran ser las matrículas de los vehículos autorizados), permite el acceso solo a las personas acreditadas. Para elaborar un sistema de seguridad para el acceso a cualquier zona es fundamental tener en cuenta ambos tipos de seguridad y conseguir que encajen entre ellas sin impedir el correcto desarrollo de sus funciones.

En el presente trabajo final de grado se pretende desarrollar un sistema (hardware y software) portátil para el reconocimiento de matrículas en controles de seguridad, utilizando un sistema de visión artificial, que permita detectar automáticamente matrículas a partir de imágenes de coches, extraer el numero con un formato de texto legible por el ordenador mediante un dispositivo portable, con cámara y pulsadores, capaz de ejecutar la aplicación. El trabajo se dividirá en tres etapas:

- La **primera etapa** de adaptación con Python y, en concreto, con la librería de OpenALPR. Una vez familiarizados con las herramientas se elaborará un primer programa funcional que permita: cargar imagen, detectar la matrícula y devolver sus caractéres en formato texto.
- En la **segunda etapa** se elaborará un programa en Python y una base de datos MySQL, que permita realizar las funciones básicas del cotejado, en la base de datos, de las matrículas leídas.
- En una **tercera etapa** se creará un hardware cómodo de transportar, con el fin de hacer el sistema portátil.

La Escuela Naval Militar se utilizará como escenario para validar el desarrollo. Este sistema de seguridad consistirá en un control de acceso que pretende identificar la matrícula de un vehículo y determinar si está autorizado o no para acceder a las instalaciones.

1.1 Motivación

El control que se lleva a cabo a la hora de entrar o salir de un sitio determinado puede remontarse a la prehistoria. Ya los griegos y los romanos empleaban el uso de códigos o contraseñas que eran requeridas a la hora de entrar en ciertos lugares de gran interés, como podían ser los emplazamientos del emperador o el lugar de reunión de los filósofos. Dichos sistemas de autenticación se han ido mejorando con el paso de los años, por ejemplo, la construcción de un paso levadizo a la entrada de los castillos con el fin de realizar un reconocimiento visual antes de permitir el acceso.

Hoy en día estos sistemas de seguridad han evolucionado a una velocidad vertiginosa hasta llegar a los sistemas de reconocimiento automático de imagen o visión artificial.

La biometría es uno de los campos más populares de aplicación de los sistemas de reconocimiento automático de imágenes. Trata de encontrar sistemas de seguridad seguros, rápidos e infalibles, mediante la identificación de rasgos únicos de cada persona, como la huella dactilar o un escáner ocular, impidiendo la posibilidad de compartir claves, códigos de ingreso o tarjetas de acceso, con el fin de que la suplantación de identidad de una persona sea prácticamente imposible. Por ejemplo, en los aeropuertos se está implantando un reconocimiento facial para el control de los pasaportes [1].

No obstante, existen otros campos de aplicación para los sistemas de reconocimiento no biométricos. En el caso de los vehículos sería necesario buscar una característica única de cada uno de ellos. Esta característica podría ser el numero de bastidor o la placa de la matrícula. Es más sencillo de reconocer la segunda ya que se encuentra a simple vista, tanto por la parte delantera, como la trasera.

El sistema consta de un pequeño hardware portátil en el que se integran una Raspberry Pi con su módulo de cámara infrarroja, un sistema de leds y un pulsador para el funcionamiento de la aplicación. Se pretende que, con la ejecución de un programa realizado en Python, este sistema sea capaz de detectar las matrículas de los vehículos que accedan, en este caso, a la Escuela Naval Militar.

El reconocimiento automático de matrículas es un método de vigilancia en masa desarrollado para los medios de transporte por carretera, que es el medio de transporte mas utilizado por los ciudadanos, y el que ha sufrido un importante crecimiento en los últimos años. Están diseñados para una función específica, como puede ser la recaudación electrónica de peajes o la vigilancia de los aparcamientos y acceso controlado a los mismos. [1].

La Escuela Naval Militar, al ser un centro militar, tiene importantes requisitos de seguridad debidos, entre otras cosas, a la amenaza terrorista a la que se ve sometida o la tenencia de material clasificado en sus instalaciones. Esta necesidad, junto al anticuado sistema de control del que se dispone en la puerta de acceso de la Escuela Naval Militar, han sido los principales motivos por los que se ha decidido llevar a cabo este sistema. Como se muestra en la Figura 1-1, el sistema de control consta de un cuerpo de guardia comprobando que los vehículos que intentan acceder están en posesión de un pase acreditativo en papel, relativamente sencillo de falsificar, mostrado en la Figura 1-2.

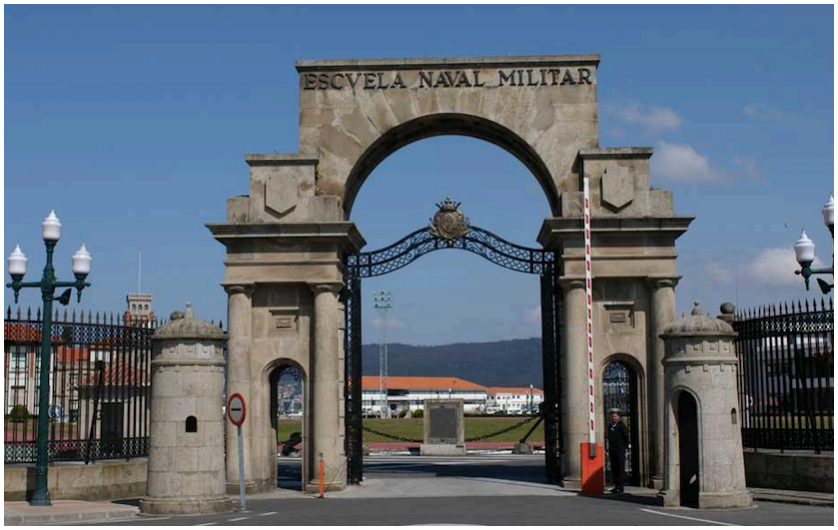

Figura 1-1 Puerta de Carlos I

La **visión artificial** es un subcampo de la inteligencia artificial para adquirir, procesar, analizar y comprender imágenes del mundo real y tiene como fin elaborar información (numérica o simbólica) capaz de ser entendida por un ordenador, que decidirá si la información es reconocida o no. La detección de objetos estudia cómo identificar la presencia de objetos determinados en una imagen,

sobre la base de su apariencia visual. Hay dos partes en el proceso: la extracción de características del contenido de la imagen y la búsqueda de objetos basada en dichas características [1].

Este sistema supondría un avance tecnológico, importante cara al público ya que la seguridad en las Fuerzas Armadas debe encontrarse entre las mejores del país para tranquilidad de sus ciudadanos. Además, proporcionaría numerosas ventajas, como incrementar la seguridad en el acceso a la Escuela Naval Militar, dar facilidades al personal de guardia en el caso de que este sistema se hiciese fijo y totalmente automatizado o incluso desarrollar aplicaciones futuras para la monitorización de los accesos, con el fin de realizar estudios y mejorar las prestaciones del aparcamiento de la Escuela Naval Militar.

Figura 1-2 Pase del vehículo para el acceso a la ENM

1.2 Objetivos

Los objetivos fundamentales de este trabajo serán el montaje del sistema hardware, consistente en una cámara que fotografíe las matrículas y que esté conectada con una Raspberry Pi que, a su vez, sea capaz de detectar, contrastar y monitorizar los datos obtenidos de las matrículas haciendo uso del software OpenALPR.

Para conseguir estos objetivos se han establecido los siguientes objetivos específicos:

- La instalación de un software en una Raspberry Pi.
- El desarrollo de una aplicación utilizando Python y ejecutada en la Raspberry Pi. El programa en Python debe ser capaz de activar la cámara y hacer una llamada a la librería de matrículas.
- Desarrollar un hardware con el propósito de hacer portátil el sistema atendiendo la necesidad de mantener fijos todos los elementos como la propia Raspberry Pi, la cámara y la batería externa.
- Realizar pruebas y validaciones que garanticen que el sistema funciona correctamente.

Raspberry Pi es un ordenador de placa reducida o de placa única. La ventaja de este ordenador es su software que es "open source", siendo su sistema operativo oficial Raspbian, que es una versión adaptada de Debian.

En nuestro caso detectaremos los códigos alfanuméricos de las matrículas de los vehículos con el software *OpenALPR*. Esos datos se contrastarán con la base de datos del departamento de seguridad de la Escuela Naval Militar.

OpenALPR es un software libre y gratuito, aunque su versión comercial no lo es. Es una aplicación C/C++ que según los comandos introducidos puede detectar matrículas de diferentes partes del mundo. Hace uso de OpenCV y las bibliotecas Tesseract de OCR.

El montaje se llevará a cabo en la puerta principal de acceso de vehículos de la Escuela Naval Militar, la puerta Carlos I. El listado de matrículas autorizadas se obtendrá de la Oficina de Seguridad de la Escuela Naval Militar. Para el desarrollo de las pruebas y las posteriores verificaciones se contará con la colaboración del personal de guardia en Carlos I.

Con el sistema montado, y consiguiendo su buen funcionamiento, se podría estudiar la posibilidad de cambiar el sistema de acreditación de pases en papel por este nuevo sistema, más moderno y seguro.

1.3 Recursos utilizados

En este apartado se enumeran los diferentes componentes hardware y software empleados.

1.3.1 Hardware

1. 1xRaspberry Pi.
2. 1xBatería externa.
3. 1xPlaca para soldar elementos electrónicos,
4. 1xPulsador.
5. 1xLed rojo.
6. 1xLed verde.
7. 1xCargador Mini-USB.
8. 6xCables para Raspberry Pi.
9. 1xCámara infrarroja para Raspberry Pi.
10. 1xCaja para albergar el sistema.

1.3.2 Software

1. Sistema operativo Linux.
2. Sistema operativo Raspbian Jessie para Raspberry Pi.
3. *Software* libre OpenALPR.
4. *Software Python.*
5. *Software* libre MySQL.

1.4 Organización de la memoria

La memoria está estructurada en cuatro capítulos, según se indica a continuación:

En el presente Capítulo se realiza una introducción al proyecto, se expone la motivación, objetivos y estructura del mismo, así como una enumeración y descripción de los distintos medios hardware y software, utilizados para el desarrollo.

En el Estado del arte se realiza una revisión del estado del arte dividida en las tres partes. En primer lugar, se expone el concepto de visión artificial y reconocimeinto de imágenes, con descripción de las mismas y de los procesos de visión artificial para la detección e identificación de elementos en las imágenes. En segundo lugar, se describen los diferentes procesos de visión artificial que se aplican para el reconocimiento de matrículas, así como las aplicaciones más usuales de dicho proceso y las librerías que existen en el mercado. En tercer lugar, se explica el hardware (Raspberry Pi 3) empleado en el proyecto.

En el Desarrollo del TFG se detallará el punto principal de este trabajo: la instalación y configuración de la Raspberry Pi, la configuración de la cámara, el programa en Python, como hacer la ejecución automática de la aplicación y como se ha construido el dispositivo móvil.

En el Capítulo de Validación y Pruebas se expondrán los resultados obtenidos con el dispositivo en las pruebas realizadas en la Escuela Naval Militar.

En las Conclusiones y líneas futuras se efectúa una valoración crítica del trabajo detallando las observaciones y las posibles mejoras para este trabajo.

Por último, en la Bibliografía se recogerán todas las fuentes de información consultadas para la realización del trabajo.

2 ESTADO DEL ARTE

2.1 Visión artificial y reconocimiento de imagen

2.1.1 Visión artificial

Dentro de la inteligencia artificial encontramos la visión artificial, que trata de imitar el proceso de visión e interpretación de los humanos. Por lo tanto, la visión artificial se compone de procesos que van, desde la toma de la imagen, hasta la obtención de la información buscada.

De un tiempo a esta parte, se han desarrollado numerosos métodos que incluyen adquisición, análisis y procesado, tanto de imágenes, como de datos multidimensionales del mundo real. Tienen como objetivo adaptar la información del mundo (imágenes) en valores numéricos. La visión artificial es aplicable a numerosos campos y a la realización de diferentes tareas.

En primer lugar, veremos cómo se forma y se almacena una imagen digital y, posteriormente, veremos los procesos que, con mayor frecuencia, se aplican a las imágenes para obtener el resultado deseado [2].

2.1.2 Adquisición de la imagen.

Las imágenes digitales están formadas por valores discretos, obtenidos a partir de valores continuos del mundo real. En el proceso de obtención de una imagen, existen dos etapas:

- Captura: Mediante un dispositivo óptico, se capta una escena continua de la vida real.
- Digitalización: La información obtenida con el dispositivo de captura se transforma en una información discreta, que es la señal digital.

2.1.2.1 Captura.

En la captura de la imagen, existen dos tipos de dispositivos fundamentales, según la forma de la captura, uno pasivo basado en el método de cámara oscura y otro activo basado en escaneo. Los primeros son más sencillos que los segundos, puesto que solo necesitan un dispositivo receptor, mientras que los segundos necesitan de un dispositivo emisor y otro receptor. En cuanto a la calidad de las imágenes, los segundos proporcionan una mejor calidad que los primeros, aunque con los nuevos avances esas diferencias de calidad están desapareciendo.

Los dispositivos de cámara oscura reciben las escenas tridimensionales de la vida real y las recogen sobre un conjunto de sensores plano. De este tipo son las cámaras que conocemos de

fotografía y video. De esta forma, se obtienen imágenes bidimensionales pero que, con el uso de varias cámaras, pueden convertirse en tridimensionales (imágenes estereoscópicas)

Los dispositivos de escaneo disponen de un elemento activo (emisor) que va recorriendo la escena que se desea obtener, recibiendo el receptor el haz del emisor. Para este tipo de dispositivos son necesarios dos dispositivos sincronizados.

2.1.2.2 Digitalización.

La digitalización consiste en la discretización de los datos recibidos (de analógico a digital). En la digitalización se dan dos procesos fundamentales: Muestreo y Cuantificación.

- **Muestreo**

El muestreo es la medición a intervalos respecto de alguna variable (tiempo o espacio). El parámetro que caracteriza este proceso es la frecuencia de muestreo, que representa el número de veces que se mide la variable por unidad de cambio. Por ejemplo, el número de muestras por unidad de espacio sobre el objeto original nos asocia al concepto de resolución espacial de la imagen, que se define como la distancia, sobre el objeto original, entre dos unidades de espacio adyacentes (píxel)

- **Cuantificación**

Consiste en asignar un valor a cada unidad de espacio que hemos obtenido en el muestreo. Los valores de cuantificación digital suelen ser potencias de dos, fácilmente representables en sistema binario para los ordenadores.

2.1.2.3 Representación digital de la imagen.

Todo lo que hemos visto hasta ahora son los procesos para la obtención de una imagen, pero, en realidad, una imagen está formada por puntos de intensidad luminosa cuyo valor de intensidad depende de la posición de cada punto.

$$\text{Imagen} = f(x, y)$$

Por lo tanto, lo que acabamos de definir es una matriz de intensidades luminosas, de M filas por N columnas, donde el producto de MxN lo conocemos como resolución de la imagen y a cada uno de los elementos (i,j) de dicha matriz de intensidad luminosa lo conocemos por píxel.

Si suponemos que la intensidad luminosa puede tomar valores entre 0 y 255 (gama de grises), donde el 0 se asigna al color negro y el 255 al color blanco y los representamos en una matriz de 16x16 obtendremos la imagen de la Figura 2-1 [3].

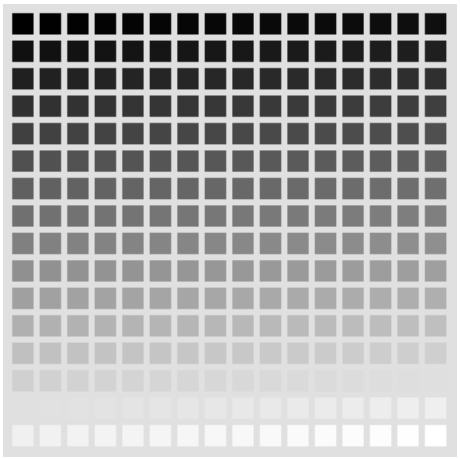

Figura 2-1 Cuadro de grises [3]

Dependiendo de los trabajos a realizar y la complejidad de los mismos, no siempre será mejor una imagen de gran tamaño, es decir de muchos píxeles, porque a mayor tamaño más cuesta mover y operar sus valores, mientras que una imagen de baja resolución resulta más fácil de operar. Por lo tanto, es importante definir el tamaño que necesitamos, que nos permita obtener los resultados buscados y con el mínimo de información para ahorrar en tiempo de computación.

Por lo tanto, una imagen es una matriz de datos, es decir, un conjunto ordenado de valores reales, donde cada elemento de la matriz es un píxel, con el que podemos hacer infinidad de operaciones matemáticas, lógicas, etc.

2.1.2.4 Tipos de imágenes.

Existen tres tipos básicos de imágenes, dependiendo de los valores que puedan tomar los elementos de la matriz y del número de datos que tengamos por cada elemento de la matriz.

Binaria. La matriz está formada por elementos binarios de tipo “boolean” donde, en esencia, cada valor se corresponde con blanco/negro. Normalmente, suele asignarse el valor cero al color negro, mientras que el valor 1 sería para el blanco. Así, cada elemento de la matriz o píxel es un bit, que puede tomar el valor 0/1.

De Intensidad: También llamada imagen de escala de grises, donde cada elemento de la matriz puede tomar un valor entero dentro del rango [0:255]. Los elementos de la matriz representan diferentes intensidades o niveles de gris, donde el valor cero suele ser asignado al negro, mientras que el valor 255 se reserva para el blanco. Cada elemento de la matriz se almacena en un byte, que puede almacenar un valor entre 0 y 255.

De Color: Se la conoce también como imagen RGB (Red-Green-Blue), donde cada elemento de la matriz queda representado por una terna de valores, representando, cada uno de esos tres valores, la intensidad de rojo, verde y azul, componiendo así el color final. Cada píxel se codifica con tres bytes por lo que la gama posible de colores alcanza el valor de 16,7 millones de colores. A este tipo de imágenes se la conoce como imagen multicanal, compuesta en este caso por tres canales [3].

Figura 2-2 Tipos de Imágenes

2.1.3 Procesos de tratamiento de imágenes.

Una vez adquirida una imagen comienzan los procesos de tratamiento de la imagen para poder obtener, de la misma, la información buscada.

Cada proceso estará formado por una serie de operaciones que se realizan sobre los valores de la matriz de intensidad luminosa de la imagen que estemos tratando, obteniendo una nueva imagen transformada. Los procesos más comunes en el tratamiento de imágenes son los siguientes:

2.1.3.1 Escalado de una imagen.

El escalado de una imagen consiste modificar las dimensiones de la imagen obtenida mediante la multiplicación las coordenadas de cada pixel por un factor de escala. Si el factor es menor de uno, se reducirá la imagen, mientras que, para valores mayores de uno, la imagen se aumentará.

El factor de escala puede ser fijo o variable y, según la variación del factor de escala se producirán distorsiones, deformaciones, etc. Por ejemplo, se puede aplicar un factor de escala horizontal y otro diferente en vertical, con lo que se produciría una modificación en la relación de la altura/anchura de la foto. Cuando el factor de escala es siempre el mismo se dice que el escalado es *isótropo*, mientras que si los coeficientes no son iguales el escalado será *anisótropo* [2].

Figura 2-3 Imagen Normal y Escalada

2.1.3.2 Modificaciones en el color de la imagen.

La modificación en el color de una imagen consiste en modificar los valores de cada uno de los elementos de la matriz de la imagen, según una función predeterminada. Así, una imagen RGB, como hemos comentado anteriormente, está representada por un conjunto de matrices M x N x P, donde M y N representan el tamaño en píxeles de la imagen y P representa el plano que se superpondrá y que, en RGB, se suele asignar 1 para color Rojo (Red), 2 para color Verde (Green) y 3 para color Azul (Blue).

La operación que nos permitiría pasar de una imagen RGB, en color, a una imagen en escala de grises consistiría en asignar, a cada pixel de cada matriz de cada color básico, el mismo valor. Este valor pudiera ser la media aritmética de los tres colores de cada píxel o una media ponderada de los mismos. Con esto obtendríamos la misma imagen en escala de grises.

Existen múltiples operaciones para modificar los colores de una imagen que, en definitiva, lo que buscan es resaltar aquellos elementos de la imagen que nos interesan para posteriormente poder hacer operaciones con ellos.

2.1.3.3 Binarización.

Como ya hemos comentado, las imágenes digitales consisten en una serie ordenada de valores de intensidad lumínica y que, para los procesos de binarización deben presentarse en escala de grises (recordemos que las matrices de color, en escala de grises, tienen todos los mismos valores en los mismos elementos).

La binarización es el proceso por el que se transforma una imagen de escala de grises a binaria mediante la transformación de los valores de intensidad lumínica de la matriz de la imagen, según la regla del valor límite de intensidad lumínica. Por lo tanto, a todos los píxeles que sean mayores se les

asigna el valor 1, mientras que a los menores se les asigna el valor 0. Así, hemos conseguido una imagen binaria Blanco/Negro.

Al valor límite de intensidad lumínica se le conoce como umbral de frecuencia y suele tomarse el valor medio de la tabla de frecuencias de las intensidades lumínicas. No obstante, esta sencilla técnica para binarizar empleando el valor medio de la tabla de frecuencias no es suficiente para muchas aplicaciones, como la que nos ocupa, para las que, existen otros métodos más sofisticados y que se describen más adelante [4].

Este proceso tiene grandes ventajas porque la imagen resultante ocupa menos memoria y, por lo tanto, resulta menos pesada para su manejo con un ordenador. Además, permite obtener propiedades geométricas y topológicas de una forma rápida y sencilla de los objetos que se encuentren en la imagen.

Cuando la binarización se realiza correctamente, apenas pierde información, permite separar los objetos del fondo de la imagen y resulta más sencillo detectar geometrías y realizar procedimientos topológicos, de una forma fácil y sencilla.

2.1.3.4 Histograma.

Para obtener una correcta binarización de una imagen en escala de grises, resulta fundamental elegir bien el umbral, que es la referencia para separar los objetos del fondo de los de la imagen. Conociendo la distribución de los diferentes valores de la gama de grises, de los píxeles oscuros y claros, resultaría más fácil determinar dicho valor de umbral. En general, estas distribuciones no son conocidas de antemano pero el histograma de la imagen nos permite obtener una representacón de dicha distribución de frecuencias.

El histograma de una imagen es la representación del número de píxeles que tiene la misma intensidad lumínica, según los niveles de gris que la componen. Dicha información resulta muy valiosa para la determinación del umbral de binarización.

Si, por ejemplo, el histograma de la imagen genera distribuciones de niveles de gris asociados a píxeles claros y oscuros muy separadas, se dice que el histograma es bimodal y, en ese caso, el umbral óptimo es aquel que se sitúa en el valle del histograma, ya que divide la imagen en los dos niveles de gris.

Según la tarea a realizar, se pueden adoptar dos estrategias para la determinación del umbral de binarización: fijo para todas las imágenes o dinámico para cada imagen, en función del histograma.

Cada estrategia tiene sus ventajas e inconvenientes. Si optamos por un valor fijo, ahorramos en procesos de computación, al no tener que calcular el valor para cada imagen y se emplea en procedimientos con imágenes muy poco cambiantes.

Por el contrario, el cálculo dinámico del umbral de binarización se suele emplear en procesos donde las imágenes tienen condiciones cambiantes de iluminación, por lo que nos proporciona un umbral más adecuado, pero se consumen más recursos de computación. No obstante, y gracias a la capacidad de procesamiento de los ordenadores de hoy en día, se va extendiendo esta segunda estrategia, por aportar valores más adecuados del umbral de binarización.

Por esta razón, últimamente, se está desarrollando la binarización adaptativa, que consiste en determinar el umbral de binarización para cada una de las partes en que se divida la imagen, lo que nos proporciona mejores resultados, no sin un coste computacional mucho más alto que cualquiera de las otras estrategias.

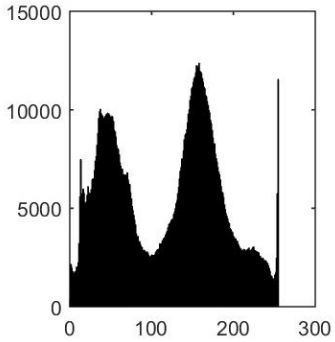

Figura 2-4 Histograma [3]

2.1.3.5 Etiquetado de elementos.

Empleamos la Visión Artificial para encontrar y clasificar determinados elementos que se encuentran dentro de una imagen. El etiquetado consiste en poner etiquetas a cada elemento que se encuentre en la imagen, normalmente, binarizada, una vez realizados los procesos para su localización. Estos elementos se localizan por formar un grupo de píxeles aislados en la imagen binaria.

Por lo tanto, el etiquetado asignará una etiqueta (normalmente un número) a todos los píxeles que forman parte de un elemento. Esto nos permitirá contar el número de elementos encontrados en la imagen, decidir si son aptos o no para nuestro objetivo y obtener algunas características de los que hayamos considerado aptos para su evaluación.

Es necesario trabajar con una imagen binaria porque con las imágenes en escala de grises o en color, sería imposible realizar esta tarea, ya que los píxeles no deben estar conectados entre sí, porque se considerarían como un único elemento. Para eliminar las conexiones impropias, se emplean operaciones morfológicas (según tamaño y forma), más concretamente de apertura o cierre de contornos.

Hay que prestar especial atención para diferenciar, a partir de una imagen binaria, cuál es el fondo y cuales son los objetos. Unos corresponderán a partes blancas y otros a partes negras. Los objetos que se etiquetan son las partes blancas y, si no coincidieran, se realizaría una inversión de la imagen.

El etiquetado es una operación delicada, ya que se necesita recorrer la imagen varias veces para etiquetar todos los elementos encontrados, pero teniendo cuidado porque podría proporcionar muchos elementos no deseados. Hay que intentar eliminar todo el ruido posible de la imagen, ya que hay que tener en cuenta que cada grupo aislado de píxeles se contará como un elemento, por lo que se hacen necesarias operaciones morfológicas para eliminar el ruido existente [5].

2.1.3.6 Extracción de características.

Para la visión humana, comparar tamaños y formas resulta relativamente fácil a simple vista, pero para la visión artificial resulta imposible y es por ello, por lo que se tiene que apoyar en ciertas características de los elementos para compararlos entre sí.

En cambio, para los humanos resulta más costoso y cansado obtener las características de los diferentes elementos, cuestión que resulta sencilla a la visión artificial, gracias a la capacidad computacional y a la ausencia de cansancio en operaciones repetitivas y monótonas.

De entre las características que se pueden medir, podemos destacar las siguientes: área, centro de gravedad, ejes de inercia, número de Euler, perímetro y número de agujeros. [5]

2.1.3.7 Operaciones de filtrado y relleno.

Una vez binarizada y etiquetada la imagen existen dos operaciones muy frecuentes que permiten homogeneizar los elementos detectados: la eliminación y el rellenado de elementos. Ambas técnicas son muy útiles porque acomodan la imagen para poder ejecutar algoritmos que proporcionen resultados más robustos.

- **Eliminación de regiones.**

Una vez detectados los elementos, se pueden diferenciar por algunas de sus características, como pueden ser el área o la forma. Así, por ejemplo, se pueden eliminar aquellos elementos que no alcancen el umbral del valor de área establecido.

De este modo, se eliminan aquellas regiones de la imagen que no podrán ser elementos evaluables para el objetivo buscado, lo que elimina, de forma eficiente, el ruido que proporcionan los objetos menores de la binarización [6].

- **Rellenado de figuras.**

Por otra parte, podemos tener el caso contrario, que nos encontremos con huecos indeseados dentro de los elementos detectados. Un hueco o agujero se produce cuando nos encontramos puntos negros de pequeña entidad, completamente rodeados por puntos blancos.

Con esta técnica conseguimos unas figuras más sólidas y compactas, pudiendo obtener el contorno con mayor facilidad.

2.1.3.8 Detección de contornos.

Una de las técnicas más productivas en la visión artificial es la detección de contornos o bordes. Un borde se puede definir como una frontera entre dos niveles de gris muy diferenciados

La detección de contornos se emplea en múltiples aplicaciones de visión artificial, podríamos decir que se encuentra en casi todos los procesos, ya que el análisis de los contornos y la comparación con figuras conocidas se demuestra muy revelador. Por ejemplo, en los procesos de fabricación, obtener el contorno de una determinada pieza nos permite compararla con la pieza patrón y obtener el nivel de similitud, lo que nos podría permitir desechar aquellas que no alcanzaran un valor mínimo establecido.

Para la detección de contornos se utiliza el operador gradiente, calculando todas las posibles variaciones de niveles de gris en todas las direcciones, recogiendo la dirección que produzca mayor valor.

En la práctica, se emplean máscaras o filtros a la imagen, que no son otra cosa que matrices que van recorriendo cada píxel de la imagen y realizando una determinada operación.

Como lo que se pretende encontrar son cambios de luminosidad, aparecen una serie de dificultades a la hora de encontrar el píxel del contorno buscado, de entre las que podemos distinguir:

- Superficies, de un mismo objeto, con diferente orientación.
- Efectos de iluminación: sombras, reflejos, etc.
- Cambios de textura.

Los operadores para la detección de contornos que más se utilizan en visión artificial son: Canny y Sobel.

Canny: Fue desarrollado por John F. Canny en 1986. Se utiliza para reducir la cantidad de información a procesar facilitando el reconocimiento de formas, búsqueda de patrones, seguimiento de objetos en movimiento o sistemas de reconocimiento OCR.

Consta de cuatro pasos:

- Búsqueda de los gradientes en la imagen: hace uso de cuatro gradientes (Horizontal, Vertical, y dos diagonales).
- Supresión de no-máximos: se toma un conjunto discreto de direcciones y elegiremos una dirección posible para el ángulo del gradiente. Los valores posibles a tener en cuenta son: Este-oeste, norte-sur, noroeste-sureste, noreste-suroeste.
- Mecanismo de doble umbral: En este paso, y habiendo calculado previamente gradientes y sectores a los que pertenecen, elegimos los píxeles candidatos a formar parte de los bordes.
- Proceso de Histéresis: De los posibles candidatos del paso anterior, establecemos los verdaderos bordes, contrastando las variaciones de color con sus vecinos.

Sobel: Se trata de un operador discreto que calcula una aproximación al gradiente de la función de intensidad de una imagen, en cada punto de la misma, determinando la dirección, sentido y módulo del vector.

Para calcular la aproximación utiliza un kernel (matriz de dimensiones 3x3) convolucionado con la imagen. Este operador tiene la propiedad de actuar como un supresor de ruido, lo cual resulta muy conveniente. El resultado es una imagen en escala de grises que representa la intensidad de los bordes, la cual se somete a un umbral para obtener una imagen binaria que identifica los bordes.

En matemáticas y, en particular, análisis funcional, una convolución es un operador matemático que transforma dos funciones f y g en una tercera función que en cierto sentido representa la magnitud en la que se superponen f y una versión trasladada y/o invertida de g. Una convolución es un tipo muy general de media móvil, como se puede observar si una de las funciones se toma como la función característica de un intervalo [5] [6].

2.1.3.9 Detección de rectas.

El más empleado en la visión artificial es el algoritmo de Hough, utilizado para la detección de ciertas formas, que sigan una ecuación determinada, dentro de las imágenes y, en particular, elementos rectos.

La versión más simple consiste en encontrar líneas dentro de una imagen binarizada. El procedimiento consiste en, para cada punto de la imagen, determinar si forma parte de una línea. De este modo, se calculan todas las posibles líneas, seleccionando aquellas que fueron las que más puntos posibles tuvieron, siendo éstas las líneas de la imagen. Como hemos comentado antes, se parte de una imagen binarizada y con los contornos definidos, previamente obtenida con un operador de contornos.

2.1.4 Reconocimiento de imagen

Esta es la tarea más desempeñada dentro del campo de la visión artificial, consiste en el procesado de la imagen determinando si dicha imagen contiene una característica, actividad u objeto. En la actualidad se pueden reconocer colores, caras humanas, caracteres escritos a mano o a máquina, objetos, vehículos, etc. No obstante, para poder reconocer dichos elementos, son necesarias una serie de condiciones específicas de entorno, posición relativa con respecto a la cámara, y de iluminación. Aunque para las personas este reconocimiento sea trivial, para un sistema de visión artificial se convierte en una tarea de cierta complejidad, si se quiere que desempeñe sus funciones en condiciones variables y de manera genérica.

La estructura básica de un proceso de reconocimiento utilizando la visión artificial se presenta en la Figura 2-5. En el subapartado 2.2.2 se detallarán cada una de las etapas.

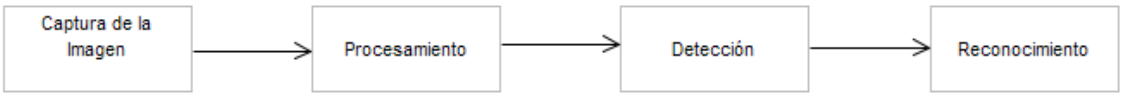

Figura 2-5 Estructura básica de reconocimiento de imágenes [4]

2.2 Reconocimiento de matrículas

2.2.1 Introducción

El reconocimiento automático de matrículas de vehículos es una manera de vigilancia en masa que hace uso del reconocimiento óptico de caracteres en imágenes (OCR). Estos sistemas son también conocidos como LPR (Licence Plate Recognition en inglés).

La integración masiva de las nuevas tecnologías, en los aspectos de la vida moderna, ha propiciado que los vehículos sean monitorizados a través de sus matrículas. El agente humano es una manera de realizar dicha monitorización, no obstante, un sistema de inteligencia artificial, es capaz de rescatar la información de las matrículas reales y plasmarla en medios digitales. Por ello, han sido desarrolladas diferentes aplicaciones de LRP que son usadas, tanto en ámbitos de seguridad, como de tráfico.

La visión artificial funciona con diferentes condiciones, debido a la variabilidad de la escena y a que todas las imágenes no van a tener las mismas condiciones, ni características. A la hora de adquirir una imagen, podemos encontrarnos con distorsión en la perspectiva debido a las variaciones y modificaciones de posición. Además, existen mas factores a tener en cuenta como los cambios de iluminación, los defectos en el objeto, etc.

La conclusión a sacar para este trabajo es que no siempre se va a tener una imagen en la que la matrícula salga en la misma posición, ni a la misma distancia, de modo que nos encontraremos ante diferentes tamaños y orientaciones. Puede ocurrir, también, que la matrícula tenga defectos añadidos como pueden ser suciedad, algún doblez o alguna fractura. El color del parachoques, sobre el que se sustente la matrícula, se puede considerar como otro factor añadido que nos dificulte la identificación de la misma.

Para sobreponerse a estas dificultades se proponen estas dos alternativas:

- Conseguir unas condiciones similares **adecuando** la adquisición de las imágenes. Para el ámbito en el que queremos la aplicación, el acceso a un parking, puede ser una buena solución ya que el sistema es concreto y bien definido. De esta manera se trabaja con un formato de imágenes que reúnen unos patrones similares.
- Conseguir la adaptación a un alto rango de variaciones **implementando** una aplicación de visión artificial **flexible**. Por lo tanto, hay que aplicar unas tecnologías capaces de mantener constantes los factores de orientación luminosidad, escala, etc. Esta solución resulta la mejor si queremos que el sistema funciones en varios ámbitos; por ejemplo, controles de accesos a diferentes parkings.

Para implementar un sistema más o menos flexible, hay que tener en cuenta todos estos factores. Adoptando la primera solución, el sistema desarrollado será más sencillo. Pero si no es conocido el emplazamiento final del sistema, se adoptará la segunda alternativa [7].

2.2.1 Sistema de detección de matrículas

Un sistema de detección de matrículas es un sistema basado en la visión artificial y que es utilizado para identificar a los vehículos a través de su matrícula, única para cada vehículo. Es muy común utilizarlo en aplicaciones de todo tipo, tanto de seguridad, como en el control de vehículos. La idea básica consiste en la obtención de los caracteres de la placa de la matrícula, almacenamiento en una base de datos y la realización de acciones según los datos obtenidos.

Como ya apuntamos antes, todos los vehículos poseen una matrícula única y fácilmente visible, por lo que se revela como el mejor método para identificar y controlar dichos vehículos. Este control ha sido siempre ejecutado por humanos, pero lo que se pretende es que, a través de la visión artificial, se pueda automatizar el reconocimiento de vehículos por su matrícula.

Hay que hacer una puntualización muy importante: Las matrículas de los vehículos varían en función del país y de la época en la que estén matriculados. Por ello, hay que tener en cuenta que, normalmente, el software va a ser creado para un país concreto, en este caso España. En nuestro caso valdría para todos los vehículos matriculados con la normativa europea cuyo formato consta de cuatro dígitos y tres consonantes. Esto va a implicar que las características del sistema van a estar orientadas al tipo de matrículas a tratar, en este caso las matrículas españolas con el formato europeo.

2.2.2 Proceso de detección de matrículas

El proceso comienza por la entrada de una imagen que, mediante diferentes etapas, en las que se aplica uno o varios algoritmos de visión artificial, se obtiene como salida una serie de posibles candidatos a ser la matrícula buscada. Estas etapas siguen un orden preestablecido y cada una tiene un objetivo concreto. Las etapas son:

- **Detección.**

Una vez obtenida la imagen, mediante el algoritmo LBP (generalmente utilizado para la detección de objetos), se encarga de recorrer todos los píxeles de la imagen para encontrar aquellas zonas candidatas a contener la matrícula. De cada región, se obtiene la coordenada x,y del píxel superior izquierdo y la anchura y altura para ser procesadas en etapas posteriores. Esta fase es la que requiere de mayor coste computacional, ya que tiene que hacer transformaciones de la imagen y comparación de patrones.

- **Binarización.**

En esta fase se realizan varias binarizaciones de cada región seleccionada previamente, modificando el umbral de binarización para encontrar la mejor imagen que nos permita la identificación de todos los caracteres de la placa de la matrícula. Una única imagen binarizada podría resultar muy clara o muy oscura, perdiendo parte de información necesaria, para ello se emplean diferentes métodos de obtención del umbral de binarización.

- **Análisis de caracteres.**

Esta etapa pretende encontrar zonas de tamaño predeterminado, según el tamaño original de los caracteres especificados, en la imagen binaria que se ha obtenido en el paso anterior. Esto se hace mediante la búsqueda de manchas o “blobs” conectados que se encuentren en la región central de la imagen. A continuación, busca manchas que mantengan una relación predeterminada de anchura y altura a las de un carácter de placa, que cumplan la condición de estar alineadas entre sí. Este análisis se realiza varias veces en cada una de las regiones, empezando por la búsqueda de caracteres pequeños, aumentando el tamaño en cada una de las pasadas. Si no se encuentran los objetos buscados, la región se descarta, mientras que si se encuentran objetos se sigue con la siguiente fase.

- **Búsqueda de bordes de matrícula.**

La siguiente fase consiste en que, si hemos encontrado objetos que pueden ser elementos de la matrícula, debemos ser capaces de encontrar los límites de la misma. Para ello, se toma, en este proceso, una región de la imagen algo mayor de la que manejamos en el proceso anterior. Se trata de encontrar los mejores bordes superior, inferior, izquierdo y derecho, en definitiva, los posibles límites horizontales y verticales de la matrícula.

Por lo tanto, se procede a buscar todas las líneas mediante el algoritmo de la transformada de Hough, con lo que obtendremos las posibles líneas horizontales y verticales de la imagen, aunque presenten una pequeña inclinación. Posteriormente, de entre todas las candidatas, y asignándoles unos pesos determinados a cada línea, se determina aquél borde que tiene más sentido y cumple con las condiciones preestablecidas de relación altura/anchura.

- **Búsqueda de límites verticales y horizontales.**

Esta técnica utiliza el filtrado de imágenes para realizar un cálculo de las estadísticas verticales de la misma.

El modo de actuación se basa en hacer una suma de las características verticales que hay en cada fila, de tal modo que la placa va a estar en el máximo de dicha suma como se observa en la figura 2-5. Se entienden como características verticales las variaciones más drásticas en el gradiente de la imagen. Para determinar los límites en las filas donde está la matrícula, hay que fijar un umbral.

Una vez que se tienen estos límites, se hace algo parecido para obtener los límites laterales, como se refleja en la Figura 2-6. Se vuelve a realizar una suma, como en el caso anterior, y se reconoce el límite izquierdo como el primer máximo por la izquierda y el derecho como el primer máximo desde la derecha.

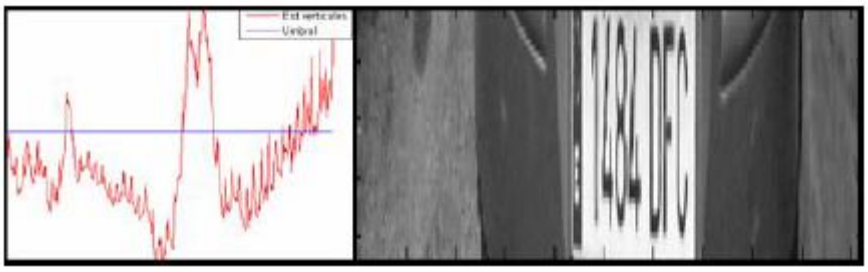

Figura 2-6 Detección vertical

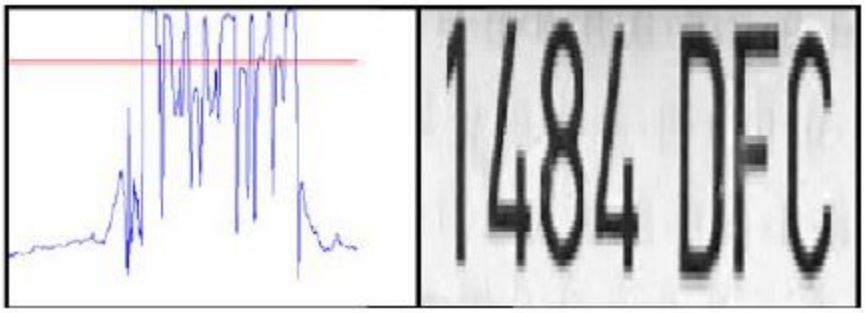

Figura 2-7 Detección de horizontales

En una fase posterior se aplica un algoritmo ORC cuando la placa está completamente detectada para extraer los caracteres de la misma.

El inconveniente de este sistema es que, al introducir un umbral para fijar los límites superior e inferior, se obliga a las que la imagen que se obtenga sea siempre a una distancia muy similar de la matrícula.

- **Enderezamiento.**

Como hemos comentado en el apartado anterior, al buscar los bordes de la placa de la matrícula, puede ser que las líneas detectadas, debido a la perspectiva de la imagen, no sean completamente horizontales y/o verticales, ni sean paralelas, por lo que se necesita realizar una transformación afín de la imagen para conseguir unos bordes lo más horizontales y verticales y que, además, sean lo más paralelos posibles entre sí.

- **Segmentación de caracteres.**

La segmentación de caracteres trata de aislar todos y cada uno de los caracteres que componen la imagen de la placa de la matrícula. Para ello se emplea un histograma vertical de la imagen binaria de la región donde, en las separaciones entre caracteres, el histograma presenta una discontinuidad. Una vez segmentados los caracteres, mediante procesos de eliminación de regiones, rellenado de objetos y etiquetado se obtiene una imagen limpia para ser procesada.

- **OCR (**Optical Character Recognition).

La fase OCR analiza cada carácter, extraído en la fase anterior, independientemente. El análisis consiste en la extracción de una serie de características de los objetos para compararlos con otros objetos patrón con unas características similares. Según el nivel de confianza obtenido de la comparación, se determina si encuentra resultado o no. Existen multitud de rutinas que realizan este proceso [7] [8].

2.2.3 Aplicaciones prácticas

Estos sistemas de reconocimiento de matrículas basados en la visión artificial tienen una alta versatilidad, pudiéndose utilizar en numerosas aplicaciones. A continuación, se hará referencia a las aplicaciones usadas de manera más frecuente.

2.2.3.1 Localización de vehículos

Esta es una aplicación muy demandada a la hora de encontrar vehículos en busca y captura. Estos vehículos suelen ser robados o que, con ellos, se ha cometido alguna infracción de cualquier tipo y la intención es localizarlos. Para esta aplicación se suelen utilizar puntos fijos, como carteles o farolas, pero también es posible ubicar el sistema en el interior de otro vehículo.

Figura 2-8 Localización de vehículos [9]

2.2.3.2 Alternancia en el acceso a núcleos urbanos

Este sistema está siendo utilizado en grandes ciudades como Madrid y promete ser una solución para la gestión del acceso alterno al centro urbano de los vehículos particulares en función de las características de su matrícula; por ejemplo, el método utilizado en la Comunidad de Madrid es limitar

el acceso para las matrículas pares o impares dependiendo del día del mes. Esto supondría una alternativa a la descongestión, además de una reducción de la contaminación, de los núcleos urbanos haciendo que todos los usuarios gocen de una igualdad de oportunidades.

2.2.3.3 Cobro de peajes urbanos

En algunos países se ha elegido dicha alternativa consiguiendo mejorar la movilidad en los núcleos urbanos. Existe la posibilidad de cobrar los peajes a los usuarios de las vías de la ciudad, sin que esto contribuya al Impuesto de Circulación de la misma.

Figura 2-9 Cobro de peajes urbanos [9]

2.2.3.4 Control de velocidad instantáneo

Aunque no es el mejor método durante el día, esta aplicación tiene frecuente uso en situaciones nocturnas. Consiste en detectar la matrícula del vehículo y a través de una secuencia de video se halla la velocidad a la que se aleja dicha matrícula.

Figura 2-10 Control instantáneo de velocidad [9]

2.2.3.5 Control de velocidad media

Es una alternativa para sustituir los controles de velocidad instantánea. Consiste en reconocer la matrícula en el primer punto, reconocer la matrícula en el segundo punto y calcular la velocidad media en dicho tramo. Son mas discretos y mas difíciles de burlar.

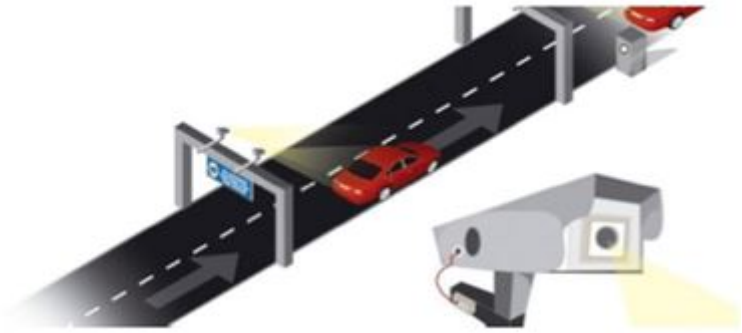

Figura 2-11 Control de velocidad media [9]

2.2.3.6 Optimización del tráfico urbano

Debido a la necesidad de optimizar la movilidad de los vehículos en las grandes ciudades, este sistema se presenta como una gran alternativa. Se trata de identificar los vehículos de forma dinámica, colocando estos sistemas en lugares estratégicos del centro urbano. De este modo se proporciona a los conductores información acerca de los tiempos que se va a emplear dependiendo de la ruta elegida y la velocidad media de la misma. Es una aplicación de optimización del tráfico en tiempo real.

2.2.3.7 Sistema de parking

Para aparcamientos privados, es de gran utilidad usar esta estrategia de reconocimiento de matrículas de los vehículos que quieran entrar. Otro uso es calcular, por ejemplo, el tiempo que un vehículo ha estado utilizando dicho aparcamiento, para el cobro posterior de un parking en el que no se tenga plaza en propiedad. Este tiempo se calcularía simplemente identificando la matrícula a la entrada y la salida del vehículo.

Figura 2-12 Sistema de parking [9]

2.2.4 Librerías

Una biblioteca de librerías de software consiste en un conjunto de funciones escritas en un lenguaje de programación específico, que permiten agilizar el trabajo y proporcionar una interfaz de desarrollo al programador o API (Application Program Interface). En el tema de visión artificial por ordenador, existen múltiples y diferentes bibliotecas.

2.2.4.1 LTI-Lib

Esta biblioteca está orientada a los objetos con estructuras y algoritmos de datos utilizadas en la visión y procesamiento de imágenes computacional. Fue desarrollado como parte de numerosos proyectos de investigación en visión artificial por la Universidad RWTH-Aachen. Estos proyectos se ocupan de la robótica, el lenguaje de señas, reconocimiento de gestos y reconocimiento de objetos. Proporciona un C++ orientado a objetos incluyendo la colección de algoritmos de alta velocidad para

poder ser utilizados en aplicaciones reales. Ha sido desarrollado utilizando GCC en Linux, y VisualC++ en Windows NT.

Figura 2-13 Logotipo de LTI-Lib [10]

2.2.4.2 VLX

VLX, desarrollado en C++, contiene los algoritmos mas usuales en visión computacional y es un conjunto de librerías que ofrecen una funcionalidad muy completa. Hacer un sistema rápido y estable fue el objetivo para el que se creó este conjunto de librerías, que dispone de la posibilidad de recurrir únicamente a las librerías que nos sean de interés de todas las que componen el conjunto.

El núcleo de librerías en VLX son:

- VNL (numéricos): contenedores numéricos y algoritmos, como pueden ser matrices, vectores, optimizadores, etc.
- VIL (imágenes): Nos permite cargar, guardar y manipular las imágenes sin importar el formato en el que se encuentre ni el tamaño que ocupe.
- VGL (geometría): Para la geometría de puntos, curvas y otros objetos elementales en una, dos o tres dimensiones.

Por otro, lado posee librerías capaces de trabajar con la geometría de la cámara, la manipulación de vídeo, equipos de música, estructura de recuperación por el movimiento, etc.

El principal inconveniente de esta librería es que no se encuentra de manera gratuita para trabajar con ella, aunque su uso no sea comercial, como ocurría también en la librería LTI-Lib.

Figura 2-14 Logotipo de VXL [10]

2.2.4.3 OpenALPR

OpenALPR no es un proyecto nuevo, no obstante, en los últimos meses ha aumentado su popularidad por las numerosas aplicaciones que puede tener. Se trata de un lector de matrículas de código abierto que, en el ámbito de vigilancia, ofrece multitud de posibilidades.

Esta librería es una aplicación C/C++ con enlaces en C#, Python y Java. El sistema es capaz de detectar diferentes tipos de matrículas de distintos países del mundo. Tiene un modo capaz de monitorear múltiples matrículas de un stream de video (MPEG) y devolver un paquete de datos JSON conteniendo el número de cada matrícula detectada.

En la Tabla 2-1 se muestran, en orden, las etapas de OpenALPR.

Fase de la canalización	Clase C ++	Descripción
Detección	Regiondetector.cpp	Encuentra las posibles regiones de la matrícula.
Binarización	Binarizewolf.cpp	Convierte la imagen de la región de la matrícula en blanco y negro.
Análisis de Carácter	Characteranalysis.cpp	Encuentra "blobs" de tamaño de carácter en la región de la matrícula.
Bordes de la placa	Platelines.cpp y platecorners.cpp	Encuentra la forma de la matrícula.
Corregir	Licenseplatecandidate.cpp	Transforma la perspectiva en una vista ideal de la matrícula.
Segmentación de caracteres	Charactersegmenter.cpp	Aísla y limpia los caracteres para que puedan ser procesados individualmente.
LOC	Ocr.cpp	Analiza cada imagen de carácter y proporciona múltiples confidencias posibles.
Pos procesamiento	Postprocess.cpp	Crea una lista n superior de posibilidades de placas basadas en confidencias OCR.

Tabla 2-1 Etapas de funcionamiento de OpenALPR

El software se puede utilizar de muchas maneras diferentes. Por ejemplo, con OpenALPR puede:

- Reconocer las matrículas de los flujos de cámaras. Los resultados son contrastados con bases de datos y pueden activar alertas. La base de datos puede estar en la nube o almacenarse enteramente dentro de su red en el local.
- Reconocer las matrículas de los flujos de la cámara y enviar los resultados a su propia aplicación.
- Procesar un archivo de vídeo y guardar los resultados de la matrícula en una base de datos.

La versión gratuita es suficiente para desarrollar una aplicación con un alcance considerable, pudiendo detectar un gran número de matrículas al día [11] [12].

2.3 Raspberry Pi 3 B

Como se ha indicado en el Capítulo 1, es un ordenador de placa reducida de bajo coste, considerada como un pequeño ordenador desarrollado en Reino Unido por la *Raspberry Pi Foundation,* que es una organización sin ánimo de lucro que dio sus primeros pasos como fundación en 2008, pero que, en realidad, llevaba gestándose desde mucho tiempo atrás. En 2011 desarrolló la Raspberry Pi como ordenador de bajo coste para facilitar la enseñanza de la informática en los colegios, pero hasta 2012 no comenzó a fabricarse. Desde entonces hasta hoy en día, se han desarrollado otros modelos, con mayores prestaciones, pero manteniendo la filosofía inicial del proyecto. Además, se ha creado una importante comunidad en Internet desarrollando sistemas operativos y aplicaciones, lo que la ha convertido en una potente herramienta.

En la Tabla 2-2 se analizan los distintos modelos de Raspberry Pi más modernos que se pueden encontrar:

Características	Modelo 2 B	Modelo 3 B
SoC	Broadcom BCM2836 (CPU + GPU + DSP + SDRAM + Puerto USB)	Broadcom BCM2837 (CPU + GPU + DSP + SDRAM + Puerto USB
CPU	900 MHz quad-core ARM Cortex A7	1.2GHz 64-bit quad-core ARMv8
GPU	Broadcom VideoCore IV 250MHz OpenGL ES 2.0	Broadcom VideoCore IV 250MHz OpenGL ES 2.0
RAM	1 GB LPDDR SDRAM 450 MHz	1 GB LPDDR SDRAM 450 MHz
Salidas USB 2.0	4	4
Almacenamiento	MicroSD	Micro SD
Conectividad de red	Ethernet 10/100 Mbps	Ethernet 10/100 Mbps, Wifi 802.11n, Bluetooth 4.1
Tamaño	85,60x56,5 mm	85,60x56,5 mm
Peso	45g	45g

Tabla 2-2 Características de los diferentes modelos de Raspberry Pi

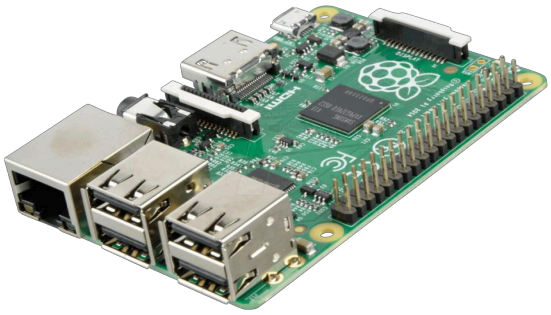

Figura 2-15 Raspberry Pi

En cuanto al hardware se pueden destacar los siguientes componentes del modelo 3 B:

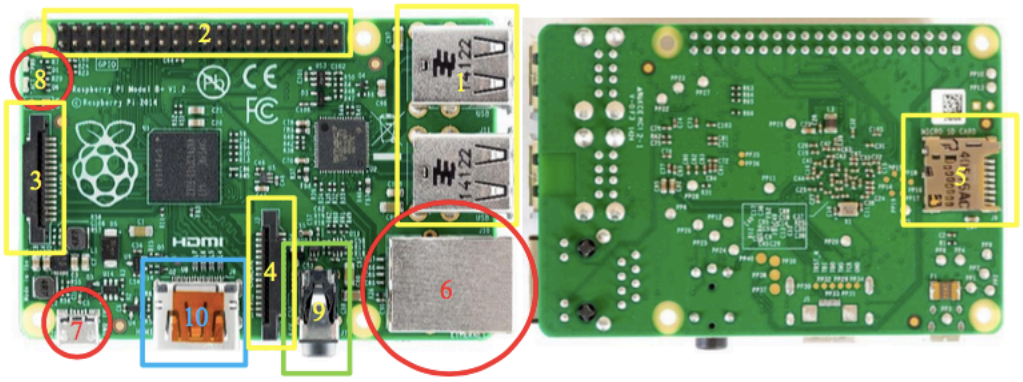

Figura 2-16 Elementos de Raspberry Pi

1. Puertos USB 2.0. El modelo 3 B posee cuatro puertos USB estándar 2.0 gestionados por el microchip LAN9514, especialmente diseñado para placas en las que se tienen que integrar dichos puertos, como es el caso de Raspberry. Gracias a estos puertos se pueden conectar por ejemplo el ratón y el teclado de manera simultánea facilitando la manera de trabajar con esta placa.

2. Pines GPIO.

Raspberry Pi B+ J8 Header

Pin#	NAME		NAME	Pin#
01	3.3v DC Power		DC Power 5v	02
03	GPIO02 (SDA1 , I2C)		DC Power 5v	04
05	GPIO03 (SCL1 , I2C)		Ground	06
07	GPIO04 (GPIO_GCLK)		(TXD0) GPIO14	08
09	Ground		(RXD0) GPIO15	10
11	GPIO17 (GPIO_GEN0)		(GPIO_GEN1) GPIO18	12
13	GPIO27 (GPIO_GEN2)		Ground	14
15	GPIO22 (GPIO_GEN3)		(GPIO_GEN4) GPIO23	16
17	3.3v DC Power		(GPIO_GEN5) GPIO24	18
19	GPIO10 (SPI_MOSI)		Ground	20
21	GPIO09 (SPI_MISO)		(GPIO_GEN6) GPIO25	22
23	GPIO11 (SPI_CLK)		(SPI_CE0_N) GPIO08	24
25	Ground		(SPI_CE1_N) GPIO07	26
27	ID_SD (I2C ID EEPROM)		(I2C ID EEPROM) ID_SC	28
29	GPIO05		Ground	30
31	GPIO06		GPIO12	32
33	GPIO13		Ground	34
35	GPIO19		GPIO16	36
37	GPIO26		GPIO20	38
39	Ground		GPIO21	40

Figura 2-17 Puertos GPIO [13]

Como se describe en la Figura 2-14 esta placa cuenta con dos filas de veinte pines cada una. Estos pines sirven de interfaz entre la Raspberry Pi y el exterior. Utilizan el sistema GPIO (General Purpose Input/Output) para dar un uso más amplio y realizar multitud de proyectos.

Con estos pines se pueden controlar pulsadores, leds, motores, etc. Para ello se deberán configurar adecuadamente como entrada o salida.
De los 40 pines que tiene la placa, 12 son de alimentación (5V o 3,3V) o tierra. El resto de pines se dividen entre GPIO normales, para emplear en proyectos, y GPIO especiales para usar como puerto SPI, UART o I2C.

3. Conector DSI Display Module. Puerto Display Serial Interface (DSI), que permite conectar a la GPU del dispositivo pantallas LCD directamente.

4. Conector CSI Camera Module. Se trata de un conector tipo bus de 15 pines usado para añadir un dispositivo compatible con la Camera Serial Interface. Con esta entrada se puede conectar la Pi NoIR (cámara de la Raspberry Pi), como la utilizada en el prototipo de prueba de este trabajo. Esta cámara carece del filtro infrarrojo por lo que es ideal para

condiciones nocturnas o de baja luminosidad.

5. Ranura con lector para microSD. Como la Raspberry Pi no dispone de un disco duro para almacenamiento, trae incorporado un lector de tarjetas de memoria microSD. El arranque del sistema se realizará desde la propia memoria microSD, por ello esta memoria deberá tener mínimo 2 GB de memoria para albergar el sistema operativo (Raspbian). En este trabajo se utilizará una tarjeta de 16 GB de capacidad.

6. Puerto Ethernet (RJ45). Este conector también esta integrado al microchip LAN9514 y proporciona una conectividad a 10/100 Mbps. El modelo 3 B tiene integrada una antena Wi-Fi por lo que este puerto puede ser utilizado para conectar la Raspberry Pi directamente a un PC sin necesidad de pasar por un router, conectando ambos equipos a través de un cable RJ45.

7. Alimentación micro USB. Al carecer de interruptores o pulsadores de encendido o apagado, esta placa dispone de un conector micro USB tipo B que proporciona 5 V de tensión. Esto es una ventaja ya que la mayoría de Smartphones utilizan cargadores que suministran más de 700 mA y son totalmente compatibles y capaces de dar tensión a la Raspberry Pi.

8. Leds indicadores. Con estos leds se puede saber si las Raspberry Pi está en funcionamiento o no, sin necesidad de tenerla conectada a una pantalla o monitor.

9. Salida de audio. Posee un conector de audio Jack 3,5 mm, además del propio HDMI. Si se utiliza el puerto HDMI, es sencillo obtener el audio: configurando adecuadamente el puerto, este transporta ambas señales, tanto video, como audio. Esto quiere decir que, simplemente conectando un único cable al monitor, es suficiente para obtener video y audio. No obstante, si el display carece de entrada HDMI se deberá utilizar la salida de audio Jack.

10. Salida HDMI. Permite la conexión de un dispositivo compatible con la interfaz HDMI 1.3 y 1.4 para la extracción de vídeo y audio.

En cuanto al software, la Raspberry ejecuta el sistema operativo GNU/Linux de código abierto. Hay varias versiones de Linux (distribuciones) que soportan la Raspberry Pi. La elegida para este trabajo ha sido Raspbian OS.

Raspbian Os es la distribución por excelencia para la Raspberry Pi. Es la más completa y documentada, además de optimizada, de las distribuciones existentes, es por eso que cuenta con el apoyo oficial. Esta distribución se basa en Debian 7.0 optimizado para la Raspberry Pi. Además, incorpora herramientas de desarrollo interesantes, como IDLE para Python o Scratch para programar videojuegos.

2.3.1 Pi NoIR

Esta cámara cuenta con el mismo sensor de 5 megapíxeles que su predecesora, pero sin el filtro infrarrojo, lo que le convierte en la herramienta perfecta para la captura de imágenes en la oscuridad.

La nueva Pi NoIR es ideal para aplicaciones que requieran visión nocturna (como en el caso de este trabajo para permitir el acceso de los vehículos no solo durante el día) e imágenes hiperespectrales, empleadas en astronomía. Su sensor es capaz de captar imágenes fijas con una resolución de 2592 x 1944, además de grabar vídeos en alta definición de hasta 1080p a 30 fotogramas por segundo, permitiendo a los usuarios de los modelos de Raspberry Pi A y B generar aplicaciones de vídeo. Tiene unas dimensiones de 20 x 25 x 9 mm y se conecta a la Raspberry Pi a través de los pines CSI libres, utilizando la interfaz de control I2C.

La cámara Pi NoIR es sensible a la radiación de IR de longitud de onda corta (alrededor de 880 nm) y incorpora una iluminación de IR para ver en la oscuridad.

Figura 2-18 Pi NoIR, cámara para Raspberry Pi [13]

3 DESARROLLO DEL TFG

En este Capítulo se describe, paso por paso, lo necesario para implementar el prototipo realizado en este trabajo. En primer lugar, se describe el diseño y los motivos por los cuales se ha hecho esa elección y en los siguientes subapartados se describirán los pasos seguidos para desarrollar, tanto el software, como el hardware.

3.1 Diseño

Para este trabajo se ha elegido:

- Como plataforma, la Raspberry Pi, ya que es una plataforma mundialmente conocida y muy utilizada. Gracias a eso, cualquier trabajo que se quiera realizar con ella tiene numerosas fuentes de documentación. También cuenta con la Pi Camera, la cámara oficial de Raspberry Pi que funciona con infrarrojos para que la detección de matrículas sea viable en periodos nocturnos. La distribución elegida es Raspbian OS, por ser la más completa y documentada, además de contar con el apoyo oficial de Raspberry Pi.
- Para la librería de detección de matrículas se ha escogido OpenALPR debido a que también es mundialmente conocida y utilizada. Es de código abierto y tiene enlaces en Python, que es el lenguaje de programación utilizado. Con esta librería se pueden identificar, con el mismo programa de Python, todas las matrículas europeas, sin importar el país.
- En cuanto a la base de datos, se ha escogido MySQL por ser una base de datos muy manejable desde la propia Raspberry Pi y sencilla de configurar a la hora de añadir o eliminar datos.
- Por último, para el hardware, se ha escogido un sistema que se acciona mediante un pulsador. Con este sistema se consiguen dos objetivos principales: que sólo se detecten las matrículas que desea el usuario y que la batería del prototipo tenga mayor duración. Para conseguir que el sistema funcione se ha de ejecutar automáticamente la aplicación nada más se encienda la Raspberry Pi.

3.2 Software

3.2.1 Instalación del Sistema Operativo

Dado que la distribución elegida es Raspbian, será necesario utilizar un ordenador y un lector de tarjetas SD. Con este ordenador, lo primero a realizar es descargar el sistema operativo desde la página oficial de Raspberry Pi [13].

Una vez descargado el sistema operativo, se descargará el software gratuito "Win32 Disk Imager", que será el que nos permita escribir en la tarjeta microSD de la Raspberry Pi. En dicho software se escoge el fichero que contenga la imagen del Sistema Operativo descargado anteriormente, y en Device la unidad asociada a la memoria microSD, teniendo cuidado de asegurarse que la elección es correcta. A través del botón Write, se copia el Sistema Operativo en la tarjeta microSD.

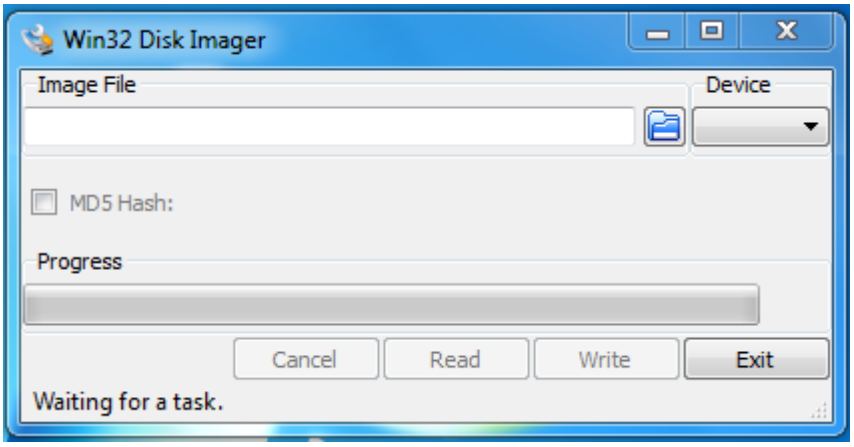

Figura 3-1 Captura programa Win32 Disk Imager

Una vez finalizado este proceso, la tarjeta será introducida en la Raspberry Pi y se podrá comenzar a trabajar en ella.

3.2.2 Raspberry Pi Camera

Cabe destacar que esta cámara es especialmente sensible a descargas electrostáticas, por ello se debe manipular con sumo cuidado.

Estando la Raspberry Pi apagada, se conecta un monitor por HDMI, el ratón y el teclado por USB y posteriormente se procede a conectar la cámara levantando el conector, insertando la cinta y volviendo a cerrar el conector como se muestra en la Figura 3-2.

Figura 3-2 Conexión de la Cámara

Una vez todo conectado, se enciende la Raspberry Pi y se introduce en la terminal los siguientes comandos:

```
sudo apt-get update
```

```
sudo apt-get upgrade
```

Para acceder al menú de configuración habrá que introducir:

```
sudo raspi-config
```

Se escoge la opción de "Enable Camera" y se reinicia la Raspberry Pi para poder dar uso a la cámara.

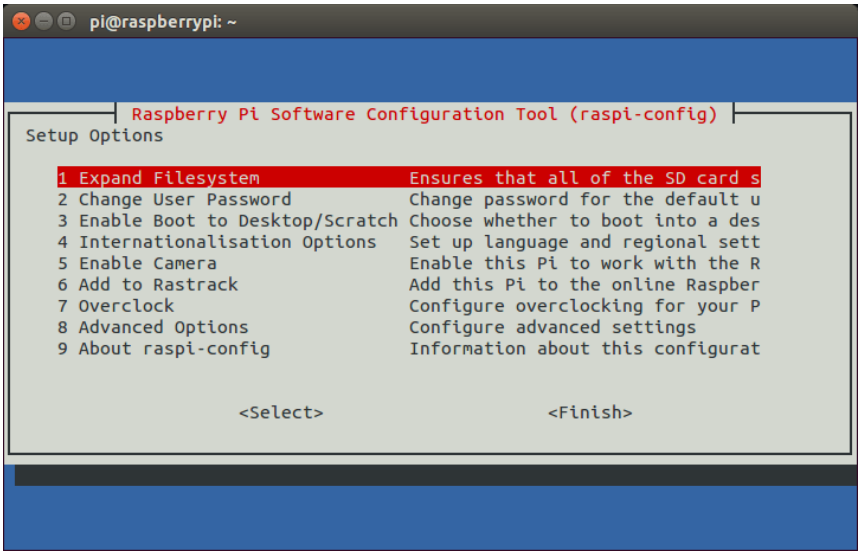

Figura 3-3 Menú de Configuración

3.2.3 Instalar la librería OpenALPR

Una vez configurada la cámara, se procede a instalar la librería OpenALPR con la que el sistema será capaz de detectar e identificar las matrículas de una fotografía.

Para ello se introducirá en la terminal lo siguiente:

```
sudo apt-get update && sudo apt-get install -y openalpr openalpr-daemon openalpr-utils libopenalpr-dev
```

De esta manera se instalará todo lo necesario para que la librería funcione correctamente. Se puede comprobar el buen funcionamiento descargando una foto de una imagen en internet y probar si la detecta e identifica correctamente. Para dicha prueba se introducirá lo siguiente:

```
wget http://plates.openalpr.com/ea7the.jpg
```

Con ese comando se descarga una imagen en formato JPG y la guardamos como *ea7the.jpg* para luego analizarla con:

```
alpr -c us ea7the.jpg
```

Con ese comando llamamos a la librería y le especificamos a través de "-c us" que la matrícula que queremos detectar es estadounidense. Para este trabajo se utilizará "-c eu" ya que las matriculas que queremos detectar son europeas.

3.2.4 MySQL

Se deberá crear el grupo que utiliza *apache* y darle los permisos:

```
sudo addgroup www-data
sudo usermod –a –G www-data www-data
```

Una vez concedidos los permisos, se actualizan los repositorios y se instala *Apache* y *PHP*:

```
sudo apt-get update
sudo apt-get install apache2 php5 libapache2-mod-php5
```

Para terminar, se reinicia el servidor apache con el siguiente comando:

```
sudo /etc/init.d/apache2 restart
```

Para la base de datos de donde se cotejarán las matriculas se utilizará MySQL que es una base de datos Open Source almacenando datos. La instalación se realizará a través del siguiente comando:

```
sudo apt-get install mysql-server mysql-client php5-mysql
```

Mientras se ejecuta la instalación se deberá establecer un usuario y una contraseña para el posterior acceso a la base de datos. Por defecto el usuario será *root* y la contraseña será *password*, en este trabajo el usuario se llamará *usuario* y la contraseña será *escuela*.

Para instalar phpMyAdmin se escribirá en la terminal lo siguiente:

```
sudo apt-get install libapache2-mod-auth-mysql php5-mysql
phpmyadmin
```

Durante este proceso de instalación se deberá especificar qué servidor web se utiliza, en este trabajo será *apache*, y si se quiere configurar una nueva base de datos facilitando una nueva contraseña.

Al terminar, será necesario modificar el siguiente archivo con el editor de texto "nano":

```
sudo nano /etc/php5/apache2/php.ini
```

Y se deberá escribir *extension=myqsl.so* antes de que comience la sección de "Dynamic Extensions". Recalcar la importancia de esta modificación en el archivo ya que al no realizarla la base de datos no funcionará correctamente y será una fuente de numerosos errores.

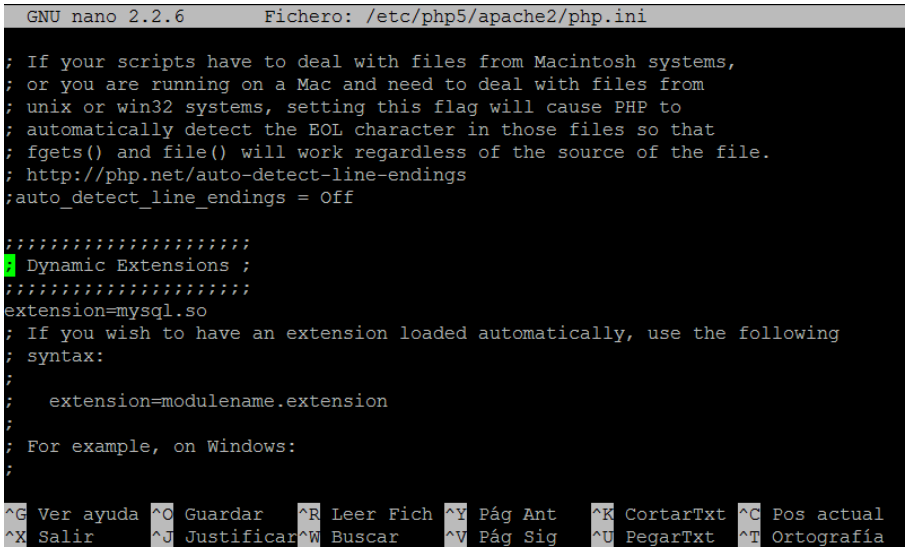

Figura 3-4 Sección en el documento a modificar

Cómo se ha creado una base de datos local, se puede acceder a ella sin necesidad de conexión a la red y se realizará de la siguiente manera:

```
mysql -u alumno -p
```

La terminal pedirá la contraseña que se estableció al crear la cuenta de MySQL, en este caso es *escuela*.

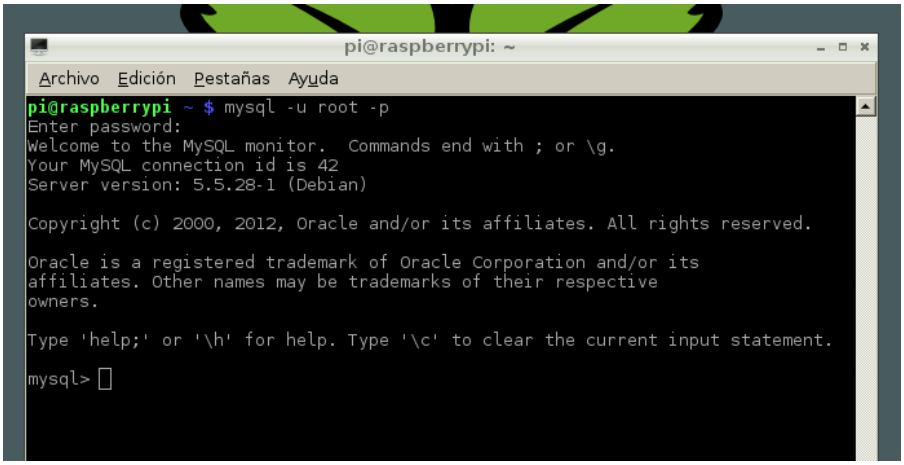

Figura 3-5 MySQL desde la terminal

Estando dentro del espacio de MySQL es imprescindible terminar cada línea de comandos con ";" ya que, si no, no ejecutará lo deseado. En primer lugar, se crea la base de datos y se indica que es dicha base la que se quiere utilizar:

```
create database tfg;
use tfg;
```

Una vez dentro de la base de datos se creará la tabla con los datos que queremos cotejar en este trabajo, con las variables que se quieren consultar. En este trabajo solo se consultarán las matrículas.

```
create table matricula(numero text);
```

Al final de la variable que se quiere crear, en este caso "numero" se deberá indicar el tipo de dato que será. Cómo la librería OpenALPR devuelve la matricula detectada en formato texto, sin diferenciar números y letras, se definirá el tipo de dato como texto poniendo al final "text".

Para introducir los datos en la tabla se escribirá:

```
insert into matricula values("4679HRZ");
```

De esta manera se está introduciendo la matrícula 4679HRZ como ejemplo. Para comprobar si se ha introducido correctamente se escribirá en la terminal:

```
select * from matricula;
```

Si se produjese algún error el sistema permite borrar datos de la tabla o directamente la tabla entera:

```
delete from matricula where numero=4679HRZ;

drop table matricula;
```

Para terminar, se deberán introducir todos los datos necesarios para la consulta de las matrículas y una vez finalizado ese proceso se escribirá para salir de MySQL lo siguiente:

```
quit;
```

3.2.5 Configuración de los puertos GPIO

Para la configuración de los puertos GPIO será necesaria la librería RPi.GPIO, por lo que lo primero a realizar será descargar y descomprimir la librería:

```
wget https://pypi.python.org/packages/source/R/RPi.GPIO/RPi.GPIO-
0.5.4.tar.gz
tar zxf RPi.GPIO-0.5.4.tar.gz
cd RPi.GPIO-0.5.4.tar.gz
sudo python setup.py install
```

Una vez instalada dicha librería lo único que se deberá hacer es importarlos en Python a la hora de escribir el script.

3.3 Script de Python

Este es uno de los puntos centrales del desarrollo realizado. Se trata de un programa de Python que, mediante el pulsado de un botón de manera manual, active la cámara, tome una fotografía, haga una llamada a la librería OpenALPR y en función de lo detectado en la fotografía analizada nos devuelva un resultado. El resultado se devolverá mediante el encendido de unos leds dependiendo si la matrícula ha sido detectada o no y en función de si la matrícula detectada se encuentra en la base de datos o no.

En la Figura 3-6 se muestra el diagrama de flujo del programa realizado. A continuación, se explica de manera detallada la función de cada una de las líneas que componen el programa.

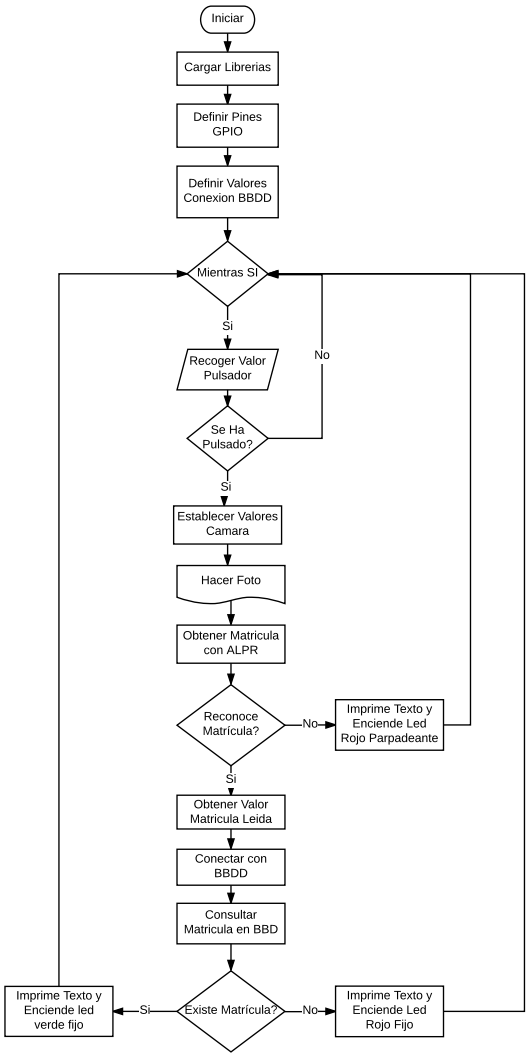

Figura 3-6 Diagrama de flujo de la aplicación (realizado en https://www.lucidart.com/es)

Es necesario recalcar que, en Python, las estructuras se forman mediante la orden terminada en dos puntos (:) y las líneas que pertenecen a la estructura, mediante el sangrado de las líneas. Esto quiere decir que, dependiendo de cómo estén escritas las líneas, unas quedaran sujetas a otras y en caso de no

escribirlas bien, al ejecutarlo, se informará de todos los errores. A continuación, se explicará paso a paso el programa.

En primer lugar, se deberá iniciar el escrito e importar las librerías necesarias para el funcionamiento de la aplicación. Para importar las librerías se escribe *import + Nombre de la Librería*. En este trabajo se han utilizado:

- MySLQ: para conectar con la base de datos.
- commands: para ejecutar los comandos en Linux.
- json: para poder utilizar los formatos json.
- RPi.GPIO as GPIO: para manejo de los pines GPIO de la placa.
- time: para el control de los tiempos a la hora de hacer las fotos.
- picamera: para el control y manejo de la cámara.

Una vez importadas las librerías, se definen los puertos GPIO a utilizar. En el caso de este trabajo serán: el puerto 21 como entrada (será el pulsador), y los puertos 22 y 27 como salidas (serán los diodos led). Con la línea "*GPIO.setmode(GPIO.BCM)*" se indica que el modo de numerar los pines será BROADCOM.

Se define entonces la conexión a la base de datos en MySQL. Se deberán seguir los siguientes pasos:

- Definir la variable con el nombre del servidor. En este trabajo se hará en un servidor local, aunque cabe la posibilidad de hacerlo en un servidor remoto.
- Definir la variable con el nombre del usuario. Recordar que en este trabajo será "usuario".
- Definir la variable con el valor de la contraseña de usuario que en este trabajo recordar que será "escuela".
- Definir la variable con el nombre de la base de datos, con el nombre "tfg".
- Por último, definir un array de conexión con estos cuatro datos.

```
#!/usr/bin/python

import MySQLdb
import commands
import json
import RPi.GPIO as GPIO
import time
import picamera
from subprocess import call


GPIO.setmode(GPIO.BCM)
GPIO.setup(21, GPIO.IN)
GPIO.setup(22, GPIO.OUT)
GPIO.setup(27, GPIO.OUT)



DB_SERVIDOR = 'localhost'
DB_USUARIO = 'usuario'
DB_PASSWORD = 'escuela'
DB_NOMBRE = 'tfg'

datos = [DB_SERVIDOR, DB_USUARIO, DB_PASSWORD, DB_NOMBRE]
```

Figura 3-7 Fragmento del script I

A partir de este punto es necesario entrar en un bucle infinito con la orden "*while true:*". Como se indicó anteriormente, todo lo que se escriba después de los dos puntos, estará incluido en dicho bucle.

En este trabajo se pretende que, al presionar el pulsador, la cámara haga una foto y a partir de ahí se analice. Para recoger la información del pulsador se deberá leer el pin GPIO que se haya asignado. En este caso será el pin 21 y se realizará mediante la orden "*inputValue=GPIO.input(21)*". De esta manera se indica que solo se actúa si se ha pulsado y por consiguiente *inputValue=True*.

Como primera medida se actúa sobre la cámara para definir las variables de brillo, contraste, etcétera y hacer la foto dentro de la estructura que se muestra en la Figura 3-8. Se pueden cambiar los valores dependiendo de la situación en la que se quiera tomar la foto, simplemente variando el valor numérico de cada línea. Destacar la importancia de ir realizando pruebas de funcionamiento y y de un ajuste correcto de los valores ya que a lo largo del desarrollo se ha comprobado que cambios en la luminosidad, como puede ser la presencia de luz artificial o de luz natural, hacen necesario tener que variar los parámetros de la cámara. Este no es un paso sencillo ya que existen infinidad de combinaciones posibles, pero sólo una es la idónea. En este trabajo, y debido a las condiciones de luminosidad, se establece el brillo al veinte por ciento y el contraste al cuarenta por ciento.

Además en la línea "*picam.capture('foto.jpg',resize=(1024,768))*" se indica que la foto que va a tomar se va a guardar con el nombre de foto.jpg. Más adelante, a la hora de obtener la matrícula se deberá indicar el nombre de la imagen para analizar. Además, se establece la resolución deseada para la imagen.

Para obtener la matrícula se hace uso de la librería "*commands*", para que se pueda ejecutar un comando en Linux que hace la llamada a la librería openALPR para obtener la matrícula de la imagen "foto.jpg" con la orden "*matricula=commands.getoutput("alpr –c eu –j foto.jpg")*". Tanto el comando "*-c eu*" como "*-j*" son comandos modificadores, el primero de ellos indica que la matrícula a analizar es europea y el segundo de ellos que se desea que el resultado sea devuelto en formato ".json". Se almacena el valor que contiene el json en el índice "results", que es el que tiene el resultado buscado con el número de la matrícula detectada y el nivel de confianza obtenido. Esto se consigue mediante las órdenes "*array=json.loads(matricula)*", "*resultado=array["results"]*" y "*longitud=len(resultado)*".

Si la longitud de "resultado" es igual a cero, la librería ALPR no ha sido capaz de identificar la matrícula que se le ha pasado en la imagen, por el contrario, si el resultado es distinto de cero, ha conseguido identificar la matrícula.

```
while True:

        inputValue = GPIO.input(21)

        if (inputValue == True):

                with picamera.PiCamera() as picam:
                        picam.start_preview()
                        picam.brightness= 20
                        picam.contrast= 40
                        picam.ISO =100
                        time.sleep(2)
                        picam.shutter_speed= 300000
                        picam.capture('foto.jpg',resize=(1024,768))
                        picam.stop_preview()
                        picam.close()

                matricula=commands.getoutput("alpr -c eu -j foto.jpg")

                array=json.loads(matricula)

                resultado=array["results"]

                longitud= len(resultado)
```

Figura 3-8 Fragmento del script II

En este punto se pueden dar dos casos: que no detecte matrícula o por el contrario que si la detecte. Dependiendo del caso que se dé, la variable "*longitud*" será igual a cero o no. En el primer caso se encenderá un led rojo parpadeante que irá conectado al pin 22. La orden para que el led parpadee se realiza mediante "*for i in range(10):*". Además, en caso de estar conectada la Raspberry Pi a un monitor, se imprimiría en pantalla "*NO HE PODIDO RECONOCER LA MATRICULA*".

En el caso de que sí se detecte la matrícula, se obtiene el valor de la matrícula identificada en la variable "*placa_resultante*" y el valor de confianza de dicha matrícula en la variable "*confianza*".

Con el valor obtenido, se define la variable "*query_sql*" donde se almacenarán las consultas que se realicen a la base de datos.

```
if (longitud==0):

    print "NO HE PODIDO RECONOCER LA MATRICULA"

    for i in range(10):

        GPIO.output(22, True)

        time.sleep(.5)

        GPIO.output(22, False)

        time.sleep(0.5)

else:

    placa_resultante=array["results"][0]["plate"]

    confianza=array["results"][0]["confidence"]


    query_sql = "SELECT * FROM matricula WHERE numero='%s'" % placa_resultante
```

Figura 3-9 Fragmento del script III

Para conectar y ejecutar la consulta a la base de datos se empleará una estructura "*try:-except:*" para controlar los posibles errores en el resultado de la consulta a la base de datos. Por lo expuesto, mientras no se produzca error, se ejecutará lo que esté situado debajo de "*try:*", mientras que, si se produce algún error, se ejecutará lo que esté situado bajo "*except:*".

En el caso de que no haya error, se realiza la conexión a la base de datos dándole los valores del array "*datos*", necesarios para la conexión.

Se obtiene únicamente el primer dato, puesto que no deberían existir matrículas repetidas en la base de datos.

Llegados a este punto se vuelven a dar dos posibles casos. El primero es que la matrícula no se encuentre en la base de datos y el segundo, por el contrario, que sí se encuentre. Con el valor de "*data*" se sabrá si la matrícula detectada pertenece o no al conjunto de valores de la base de datos. Si el valor "*data*" es nulo, la matrícula no existe en la base de datos. Si dicho valor no es nulo quiere decir que la consulta ha devuelto un resultado, por lo que la matrícula reconocida existe en la base de datos.

Es entonces cuando se empleará una estructura "*if-else*", dependiendo del valor de "*data*". Utilizando "*if (data==None):*" se está indicando que la matrícula leída no está en la base de datos por lo que se pasará a encender un led de color rojo, conectado al pin 22, de manera continua durante tres segundos y en el caso de que la Raspberry Pi estuviese conectada a un monitor, se imprimiría por este en color rojo: "*LA MATRICULA NO EXISTE EN LA BASE DE DATOS*".

Si la matrícula si que se encuentra en la base de datos, es decir, el valor de "*data*" no es nulo se ejecuta lo escrito debajo de "*else*". En este caso se encenderá un led de color verde, conectado al pin 27, de manera continua durante cinco segundos y en caso de estar conectada la Raspberry Pi a un monitor se imprimiría por este en color verde: "*MATRICULA: (número de la matrícula leída) (nivel de confianza en tanto por ciento)*".

Por ultimo, debe contemplarse la posibilidad de que se produzca un error. Esta posibilidad se contempla con "*except:*". En caso de que se produzca un error, se activará el comando "*except:*" mostrando en la pantalla: "*Error: No se pudo obtener el dato.*" y se cerraría la conexión con la base de datos con la orden: "*con.close()*".

```
try:

    conn = MySQLdb.connect(*datos)
    cursor = conn.cursor()

    cursor.execute(query_sql)

    data = cursor.fetchone()

    if (data==None):

        print chr(27) + "[7;31m" + "LA MATRICULA NO EXISTE EN LA BASE DE DATOS" + chr(27) + "[0;0m"

        print chr(27) + "[7;31m" + "ENCIENDO EL LED ROJO DURANTE 3 SEGUNDOS" + chr(27) + "[0;0m"

        GPIO.output(22, True)
        time.sleep(3)
        GPIO.output(22, False)

    else:

        print chr(27)+"[0;36m"+"MATRICULA: " + chr(27)+ "[7;36m" + placa_resultante + chr(27) + "[0;0m" + " -- " + format(confianza)

        print chr(27) + "[7;32m" + "ENCIENDO EL LED VERDE DURANTE 5 SEGUNDOS" + chr(27) + "[0;0m"


        GPIO.output(27, True)
        time.sleep(5)
        GPIO.output(27, False)

except:

    print "Error: No se pudo obtener el dato"

    conn.close()
```

Figura 3-10 Fragmento del script IV

3.4 Ejecución automática de la aplicación

Este es necesario si lo que se desea es comenzar a detectar las matrículas sin necesidad de ejecutar nada, ni introducir nada en la Raspberry Pi. Lo que es de vital importancia en un sistema portátil

alimentado con una batería externa como en el caso de este trabajo. Además, evita la necesidad de estar conectado a un monitor.

El primer paso será crear un script que se encargue de arrancar automáticamente nuestro software. En nuestro caso es el script *detector-init*. Para crearlo se hará:

```
sudo nano /etc/init.d/detector-init
```

En dicho script se deberá escribir:

```
#! /bin/sh
# /etc/init.d/detector-init

### BEGIN INIT INFO
# Provides:          detector-init
# Required-Start:    $all
# Required-Stop:     $remote_fs $syslog
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Description:       Script para arrancar el detector de matrículas
### END INIT INFO


# Dependiendo de los parámetros que se le pasen al programa se usa una opción u otra
case "$1" in
  start)
    echo "Arrancando detector-init"
    # Aquí hay que poner el programa que quieras arrancar automáticamente
    /usr/bin/python /home/pi/prueba_final1.py
    ;;
  stop)
    echo "Deteniendo detector-init"

    ;;
  *)
    echo "Modo de uso: /etc/init.d/detector-init {start|stop}"
    exit 1
    ;;
esac

exit 0
```

Figura 3-11 Script para el arranque automático

En el caso de este trabajo, el programa que queremos arrancar automáticamente será *prueba_final1.py* con lo que se deberá indicar en la línea correspondiente indicando también la ruta de carpetas en la que se encuentra.

Una vez realizado el programa, se da permisos de ejecución al fichero:

```
sudo chmod 755 /etc/init.d/detector-init
```

Se realiza la comprobación de que todo se ejecuta correctamente:

```
sudo /etc/init.d/detector-init start
```

En caso de querer parar la aplicación, se escribirá:

```
sudo /etc/init.d/detector-init stop
```

Para terminar, se activa el arranque automático:

```
sudo update-rc.d detector-init defaults
```

Una vez hecho esto, al reiniciar la Raspberry Pi, esta comenzará a detectar las matrículas siempre que esté encendida y se cumplan las condiciones del script principal *prueba_final1.py*.

3.5 Dispositivo hardware

Para hacer el montaje del hardware se han utilizado los siguientes elementos:

- Una Raspberry Pi modelo 3 B.
- Una Pi Camera Infrarroja.
- Un pulsador.
- Tres resistencias de 1kΩ.
- Dos diodos led, uno verde y uno rojo.
- Diferentes cables para conectar los elementos entre si.
- Una batería externa.

El montaje para este trabajo se realizó, en primer lugar, sobre una *protoboard* para realizar las pruebas y el montaje definitivo se llevó a cabo sobre una placa para soldar los elementos. El montaje sigue un esquema como el mostrado en la Figura 3-12, dicho esquema ha sido realizado en *Fritzing*, un programa gratuito para realizarlos diseños sobre la protoboard, de manera esquemática o en PBC.

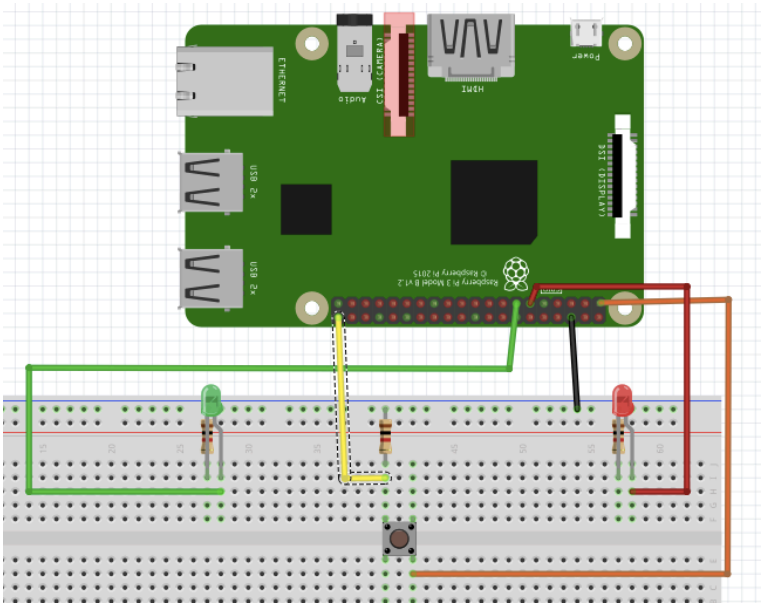

Figura 3-12 Esquema del montaje

Además, el sistema ha sido colocado en una caja de plástico con diferentes soportes de madera con el fin de facilitar su transporte y de esta manera permitir que el sistema sea válido no solo en la puerta de Carlos I.

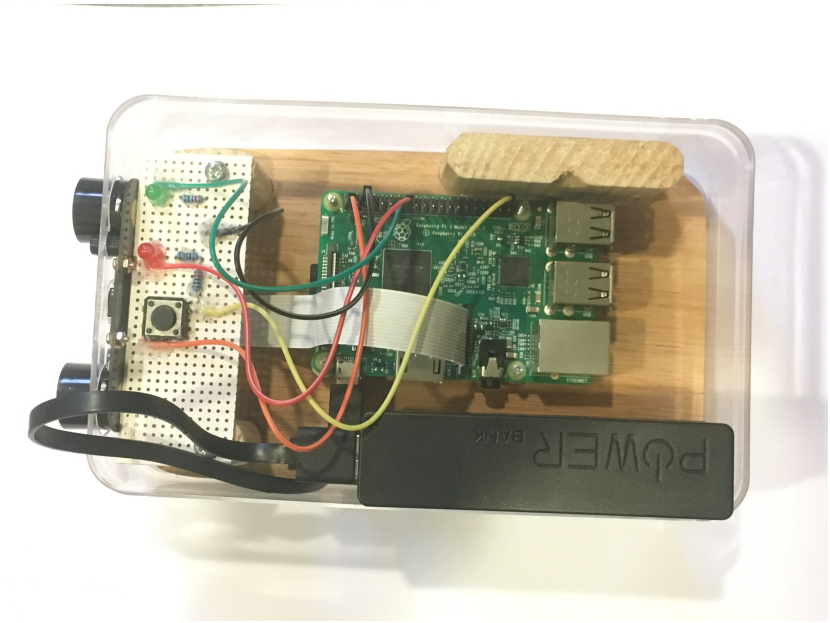

Figura 3-13 Montaje definitivo I

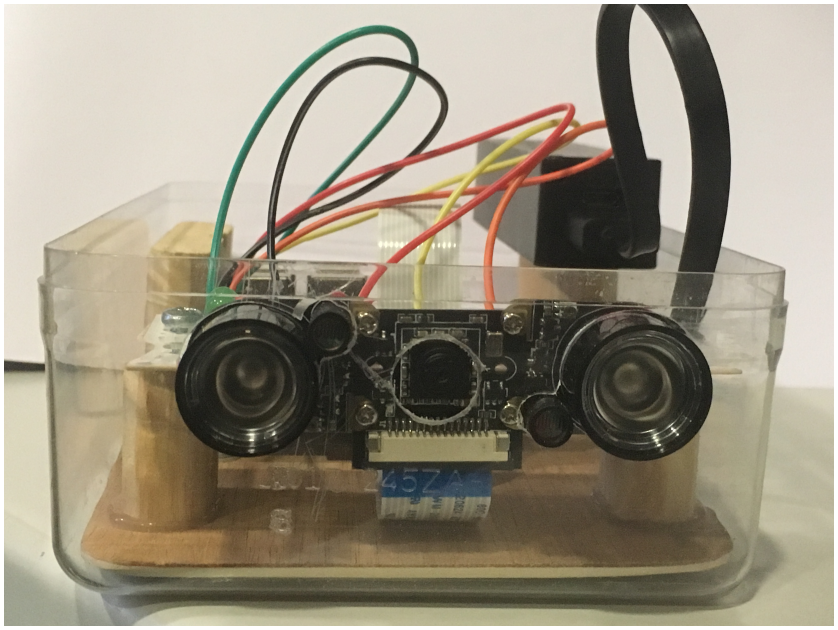

Figura 3-14 Montaje definitivo II

4 VALIDACIÓN Y PRUEBAS

En este Capítulo se describen las pruebas realizadas con el prototipo para evaluar su funcionalidad y validar su funcionamiento.

Para ello, se han realizado las pruebas variando los parámetros de la luz ambiente (día o noche), variando el ángulo desde el que está tomada la fotografía y variando la distancia a la que se toma la imagen. Con lo cual, para cada distancia probada se deberán realizar cuatro pruebas distintas.

Las pruebas están realizadas con un número total de doce matrículas diferentes pertenecientes a los vehículos de los alumnos que se alojan en el cuartel Marqués de la Victoria.

Las pruebas se realizaron fotografiando aleatoriamente los vehículos, para las imágenes centradas se realizaron treinta fotos y para las fotografías en perspectiva se realizaron un total de ochenta fotografías teniendo en cuenta la perspectiva desde ambos lados. Se ha tomado un umbral del 75% de matrículas acertadas para considerar la prueba satisfactoria. Para esta aplicación no es necesario que el porcentaje de aciertos sea muy elevado ya que si no se detecta una matrícula o se detecta incorrectamente se notificará al usuario mediante el encendido de los leds para que realice un nuevo intento.

4.1 Pruebas de distancia: a 1,5 metros

4.1.1 Durante las horas de luz natural

Alternando los vehículos fotografiados se realizaron un total de treinta fotografías con la matrícula totalmente centrada. Esta prueba resultó ser un éxito ya que se detectaron correctamente el 100% de las matrículas.

Por otro lado, se realizaron otras ochenta fotografías variando el ángulo de la cámara con respecto al coche. Con los resultados de esta prueba se puede afirmar que, a la distancia de metro y medio, este aparato funciona de manera fiable tomando fotografías con un ángulo de, como máximo, quince grados a ambos lados respecto a la perpendicular de la matrícula. Con ángulos mayores, la eficacia del aparato se reduce por debajo del 75% y en lugar de una mejora para la comodidad, produciría un efecto contrario.

La siguiente imagen muestra el vehículo, de día, a la distancia de 1,5 m.

Figura 4-1 Fotografía tomada a 1,5 metros.

4.1.2 Durante el período nocturno

Debido a que la cámara instalada cuenta con infrarrojos, las pruebas realizadas durante la noche ofrecieron resultados prácticamente idénticos a las pruebas realizadas durante el día.

Se realizaron un total de treinta fotografías tomadas con la matrícula totalmente centrada y el resultado fue inmejorable, 100% de matrículas detectadas correctamente.

En cambio, con las fotografías tomadas con perspectiva los resultados ofrecidos fueron ligeramente peores que los obtenidos durante el día. Se tomaron un total de ochenta fotos y se llego a la conclusión que, en este caso, el máximo ángulo aceptable es de diez grados también tomados desde la perpendicular de la matrícula. Con las fotografías tomadas a quince grados, el porcentaje de aciertos se reducía al sesenta por ciento, no obstante, con las imágenes tomadas con diez grados, el porcentaje de aciertos fue del 77,5% como muestra la Tabla 4-1.

ÁNGULO	**PERIODO**	
	DIURNO	**NOCTURNO**
0º	30/30 = 100% ACIERTOS	30/30 =100% ACIERTOS
5º	72/80 = 90% ACIERTOS	70/80 = 87,5% ACIERTOS
10º	64/80 = 80% ACIERTOS	62/80 = 77,5% ACIERTOS
15º	61/80 = 76% ACIERTOS	56/80 = 70% ACIERTOS
20º	57/80 = 71.2% ACIERTOS	50/80 = 62,5% ACIERTOS
25º	52/80 = 65% ACIERTOS	31/80 = 38,7% ACIERTOS

Tabla 4-1 Resultados a 1.5 metros

4.2 Pruebas de distancia: a 3 metros

Los resultados obtenidos a esta distancia han sido prácticamente iguales que los obtenidos en la distancia de metro y medio. Además, se han detectado ciertas particularidades que se expondrán en apartados posteriores.

Figura 4-1 Fotografía tomada a 3 metros

4.2.1 Durante las horas de luz natural

Las pruebas se han realizado de la misma manera que el resto de pruebas, fotografiando primeramente treinta vehículos de manera aleatoria y con la matrícula centrada, en estas condiciones se obtuvieron un total de veintinueve matrículas detectadas correctamente y una detectada incorrectamente.

En segundo lugar, se pasó a fotografiar los vehículos buscando el ángulo límite para un funcionamiento eficaz. Se realizaron un total de ochenta fotografías con la misma aleatoriedad, concluyendo que el máximo ángulo con el que el sistema detecta correctamente un porcentaje superior al setenta y cinco por ciento de matrículas es veinte grados.

Se puede observar cierta mejoría a la hora de realizar la detección en perspectiva, aunque un ligero empeoramiento a la hora de realizar la detección de manera perpendicular a la matrícula.

4.2.2 Durante el periodo nocturno

Durante las pruebas nocturnas a esta distancia se obtuvieron mejores resultados que a distancias inferiores.

Para las pruebas tomando imágenes sin perspectiva se hicieron un total de treinta fotografías obteniendo el cien por cien de las matrículas correctamente detectadas.

Por otro lado, las fotografías tomadas con perspectiva concluyeron que el máximo ángulo al que el aparato funciona correctamente es de veinte grados. Se tomaron un total de ochenta fotografías obteniendo, para dicho ángulo, un setenta y siete por ciento de matrículas correctamente detectadas.

La Tabla 4-2 muestra los resultados para todas las combinaciones de ángulos y luminosidad.

ÁNGULO	PERIODO	
	DIURNO	NOCTURNO
0º	29/30 = 96.6% ACIERTOS	30/30 =100% ACIERTOS
5º	75/80 = 93.7% ACIERTOS	77/80 = 96.2% ACIERTOS
10º	69/80 = 86.2% ACIERTOS	69/80 = 86.2% ACIERTOS
15º	64/80 = 80% ACIERTOS	62/80 = 77.5% ACIERTOS
20º	62/80 = 77.5% ACIERTOS	61/80 = 76.2% ACIERTOS
25º	50/80 = 62.5% ACIERTOS	31/80 = 38.7% ACIERTOS

Tabla 4-2 Resultados a 3 metros

4.3 Pruebas de distancia: a 5 metros

A esta distancia se han obtenido los peores resultados, como muestra la Tabla 4-4, concluyendo que, a una distancia de cinco metros y con los ajustes iniciales de la cámara, el sistema no es eficaz, ni durante el día, ni durante la noche. A su vez, tampoco es eficaz ni con perspectiva, ni sin perspectiva.

Por este motivo se decidió ajustar los parámetros de brillo y contraste de la cámara, como se detalla en el Capítulo 3, subapartado 3.3. Con estos cambios, a la distancia de cinco metros, se obtienen resultados por encima del umbral de aciertos establecidos, tomando las fotos de manera perpendicular y con un ángulo máximo de entre veinte y veinticinco grados, tanto de día como de noche. Las pruebas se realizaron de la misma manera que las anteriores obteniéndose durante el día de manera perpendicular un 93,3% de aciertos y con el ángulo máximo de entre veinte y veinticinco grados como se puede observar en la Tabla 4-3.

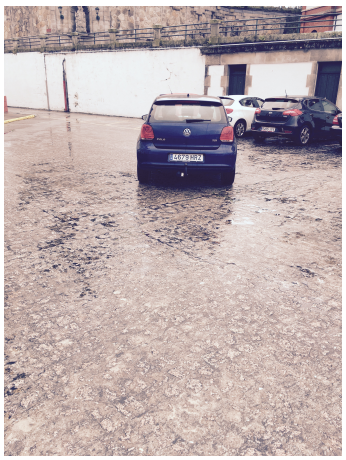

Figura 4-2 Fotografía tomada a 5 metros

Durante la noche, de manera perpendicular se obtuvieron resultados ligeramente mejores, de un 93,3% de aciertos, y con las fotografías en perspectiva se obtuvo de nuevo el mismo ángulo.

Sin embargo, con los parámetros nuevos, el sistema no es eficaz a las distancias de tres metros y metro y medio de ninguna de las maneras posibles.

Como conclusión, se volvieron a establecer los parámetros iniciales destacando que la distancia máxima eficaz de funcionamiento del sistema es de tres metros. Además, se recomienda a los usuarios utilizar el aparato a esta distancia ya que les permite un ángulo de desvío mayor.

ÁNGULO	A 1,5 METROS		A 3 METROS		A 5 METROS	
	PERIODO		PERIODO		PERIODO	
	DIURNO	NOCTURNO	DIURNO	NOCTURNO	DIURNO	NOCTURNO
0º	20/30=66,6% ACIERTOS	20/30=66,6% ACIERTOS	21/30=70% ACIERTOS	22/30=73,3% ACIERTOS	28/30=93,3% ACIERTOS	28/30=93,3% ACIERTOS
5º	55/80=68,7% ACIERTOS	53/80=66,2% ACIERTOS	58/80=72,5% ACIERTOS	57/80=71,2% ACIERTOS	72/80=90% ACIERTOS	71/80=88,7% ACIERTOS
10º	53/80=66,2% ACIERTOS	51/80=63,7% ACIERTOS	57/80=71,2% ACIERTOS	57/80=71,2% ACIERTOS	70/80=87,5% ACIERTOS	69/80=86,2% ACIERTOS
15º	50/80=62,5% ACIERTOS	38/80=47,5% ACIERTOS	57/80=71,2% ACIERTOS	45/80=56,2% ACIERTOS	71/80=88,7% ACIERTOS	71/80=88,7% ACIERTOS
20º	31/80=38,7% ACIERTOS	33/80=41,2% ACIERTOS	41/80=51,2% ACIERTOS	44/80=55% ACIERTOS	68/80=85% ACIERTOS	67/80=83,7% ACIERTOS
25º	27/80=33,7% ACIERTOS	25/80=31,2% ACIERTOS	38/80=47,5% ACIERTOS	40/80=50% ACIERTOS	67/80=83,7% ACIERTOS	66/80=82,5% ACIERTOS

Tabla 4-3 Resultados con parámetros ideales para 5 metros

ÁNGULO	PERIODO	
	DIURNO	NOCTURNO
0º	21/30 = 70% ACIERTOS	22/30 =73,3% ACIERTOS
5º	54/80 = 67,5% ACIERTOS	55/80 = 68,7% ACIERTOS
10º	52/80 = 65% ACIERTOS	54/80 = 67,5% ACIERTOS
15º	52/80 = 65% ACIERTOS	51/80 = 63,7% ACIERTOS
20º	49/80 = 61,2% ACIERTOS	47/80 = 58,7% ACIERTOS
25º	44/80 = 55% ACIERTOS	41/80 = 51,2% ACIERTOS

Tabla 4-4 Resultados obtenidos con los parámetros definitivos a 5 metros

4.4 Observaciones

Durante las pruebas realizadas se observaron una serie de problemas añadidos:

- El sistema no es capaz de detectar ninguna matrícula en el caso de que en la imagen tomada haya más de una matrícula con posibilidad de ser detectada. Por esta razón se recomienda utilizar este aparato para una única fila de vehículos.
- El sistema comete errores de detección en caso de que la matrícula esté mojada o sucia. En el caso de la lluvia se comprobó que, si en la placa hay muchas gotas, el sistema no suele detectar la matrícula correctamente. Por esta razón se recomienda al usuario utilizar el aparato en un lugar donde el vehículo se encuentre a cubierto de la lluvia en el momento de tomar la foto.
- A vehículos con matrículas especiales de las Fuerzas Armadas, tales como los vehículos de la guardia, el sistema no es capaz de reconocer la matrícula. Este inconveniente, en nuestro caso, no es relevante debido a que los vehículos de la Escuela Naval Militar no disponen de pase (su matrícula es totalmente reconocible a simple vista).

5 CONCLUSIONES Y LÍNEAS FUTURAS

5.1 Conclusiones

De acuerdo con los objetivos iniciales propuestos se puede concluir que estos se han alcanzado satisfactoriamente. El sistema operativo Raspbian funciona correctamente, la aplicación desarrollada utilizando Python satisface todas las necesidades propuestas y el dispositivo hardware cumple con sus funciones de comodidad y portabilidad. Todos estos objetivos han sido evaluados en el Capítulo 4. No obstante, se han presentado algunos problemas a lo largo del mismo:

Como aspectos positivos se pueden destacar el reducido coste del sistema ya que solo consta de una Raspberry Pi, una batería externa, una cámara con sensores infrarrojos y componentes electrónicos de bajo coste, como pueden ser leds, pulsadores o resistencias y cables. Además, presenta unos muy buenos resultados alcanzando con éxito los objetivos propuestos como la detección de casi el 100% de las matrículas tomando fotografías frontales, tanto de día, como de noche a una distancia de hasta tres metros y una detección superior del 75% con ángulos de hasta veinte grados. El alcance se puede aumentar hasta los cinco metros utilizando ajustes específicos para la cámara.

No obstante, se han presentado algunos inconvenientes de fácil solución. La cámara utilizada presenta ciertos problemas referentes a la distancia y las condiciones de la placa. Aún así, los resultados obtenidos son satisfactorios. En ocasiones es necesario tomar la fotografía una segunda vez, debido a que el aparato es portátil y requiere cierta técnica para el enfoque de la cámara. No obstante, esta técnica no es difícil de adquirir.

Es necesario prestar atención a una serie de condicionespara el buen funcionamiento del sistema. En el caso de que se fuese a llevar a cabo para su implantación en la escuela, sería necesario la construcción de un nuevo hardware mas robusto y estanco. Como se ha detallado en el Capítulo 4, este sistema no es viable para el control de los vehículos ya estacionados debido a que podrían aparecer en la misma fotografía mas de una matrícula.

Por otro lado, también se han extraído algunas conclusiones del proceso de desarrollo. A la hora de realizar el desarrollo es necesario tener, al menos, conocimientos básicos de Python y el esquema de funcionamiento de los puertos GPIO de la Raspberry Pi. Para programar en Python es de vital importancia respetar los sangrados para una correcta estructura. Un fallo de este tipo, por leve que sea, puede cambiar el funcionamiento de la aplicación de manera drástica o incluso evitar que pueda ser ejecutado. Es más que recomendable realizar tutoriales acerca de lo que se va a realizar, esto evitará muchos problemas a la hora de avanzar en el proyecto. Numerosos problemas detectados en las primeras fases del trabajo son debidos a la falta de experiencia, pero, a medida que el trabajo iba avanzando, los errores y deficiencias se corregían de una manera mas rápida y eficaz. En cuanto al

código en Python, resulta de gran utilidad desglosarlo en pequeñas partes con funciones especificas y comprobar que funcionan correctamente. Una vez comprobadas todas las partes resulta muy sencillo unirlas para completar el desarrollo de la aplicación. A su vez, es necesario tener claro el esquema de los puertos GPIO de la Raspberry Pi ya que a la hora de configurarlos no todos ofrecen las mismas posibilidades. La documentación que la comunidad Raspberry Pi cuelga en Internet, de manera desinteresada, ha resultado de mucha ayuda. No obstante, es esencial no tomar esta información como referencia infalible y contrastar la información con fuentes oficiales.

El hábito adquirido de documentar y referenciar cada uno de los pasos realizados ha facilitado la tarea de redactar la memoria. También ha facilitado el avance del trabajo el hecho de realizar esquemas en papel con las ideas principales para luego ir añadiendo mejoras secundarias.

Para la comprensión del funcionamiento y de los circuitos existen programas gratuitos y páginas web disponibles en Internet, como *Fritzing* o *www.lucidart.com* que permiten realizar esquemas y diagramas de manera gratuita.

5.2 Líneas futuras

A continuación se proponen algunas líneas futuras para mejorar las prestaciones del sistema desarrollado o incluir nuevas funcionalidades:

- La construcción de un dispositivo fijo en el lugar donde se quiera instalar. Esto acompañado de sus respectivas pruebas permitiría establecer una zona de espera para los vehículos de tal manera que la librería fuese capaz de detectar correctamente el cien por cien de las fotografías tomadas.
- En dicha zona de espera, construir una marquesina o alguna estructura similar para proteger a los vehículos de la lluvia a la hora de tomar la fotografía.
- En el caso en que se desease implementar en la Escuela Naval Militar, se podría realizar otra aplicación para la identificación del personal a bordo del vehículo. Estas aplicaciones podrían ser de reconocimiento facial o de lector de TIM (Tarjeta de Identificación Militar).
- Otro campo de mejora, sería el autoguardado de los accesos de vehículos a las instalaciones con el fin de realizar posteriores estudios estadísticos como por ejemplo las horas en las que más vehículos entran o salen, saber en todo momento que personal se encuentra a bordo de la Escuela Naval Militar, realizar estudios de la necesidad de ampliación de los aparcamientos en la Escuela Naval Militar o incluso detectar accesos anómalos haciendo uso de sistemas de inteligencia artificial.
- Se podría alojar la base de datos en un servidor remoto y desarrollar una aplicación con el fin de actualizar las altas de nuevos vehículos o las bajas de vehículos que ya no tengan la autorización de pase de una manera más sencilla y cómoda. A este respecto indicar que en este trabajo ya se ha realizado una prueba exitosa de conectividad utilizando la interfaz Wi-Fi de la Raspberry Pi.

6 BIBLIOGRAFÍA

[1] «La biometría en la seguridad,» http://seguridadseat.com/articulos-seguridad/objetivos-seguridad-biometrica.html. [Último acceso: 20 Febrero 2017].

[2] «Comparativa visión humana y visión artificial,» http://sabia.tic.udc.es/gc/Contenidos%20adicionales/. [Último acceso: 14 Enero 2017].

[3] «Imágenes en escala de grises.,» http://www.aloj.us.es/galba/digital/cuatrimestre_ii/imagen-pagina/ 2elementos4d.htm . [Último acceso: 15 Enero 2017].

[4] «Binarización de imágenes en opencv,» http://es.scribd.com/doc/94351721/OpenCV . [Último acceso: 25 Febrero 2017].

[5] « Etiquetado y extracción de características,» http://catarina.udlap.mx/u_dl_a/tales/documentos/ lis/nieto_b_d/capitulo2.pdf . [Último acceso: 20 Enero 2017].

[6] «Etapas en el procesamiento de una imagen,» 2014. http://www.sites.upiicsa.ipn.mx/polilibros/portal/Polilibros/P_proceso/Nuevas_tecnologias_Areli_Araos_Pe%C3% B1aloza/procesoimagenes/proc_info.htm . [Último acceso: 14 Enero 2017].

[7] «Diferentes aspectos sobre el uso de ocr,» http://geekland.hol.es/fundamentos-usos-limitaciones-ocr/ . [Último acceso: 24 Febrero 2017].

[8] «Diversos algoritmos ocr» http://www.dspace.espol.edu.ec/bitstream/123456789/20617/1/ D-91508.pdf. [Último acceso: Febrero 2017].

[9] «Aplicaciones de la visión artificial,» http://dmi.uib.es/~ygonzalez/VI/Material_del_Curso/ Teoria/Aplicaciones_VC.PDF. [Último acceso: 19 Enero 2017].

[10] «Página Oficial de Animetrics, empresa que desarrolla sistemas de reconocimiento artificial,» http://animetrics.com/. [Último acceso: 18 Enero 2017].

[11] «Tutorial opencv,» eupt.unizar.es/ctmedra/tutorial_opencv.pdf. [Último acceso: 15 Enero 2017].

[12] «Página oficial opencv,»
http://opencv.org/. [Último acceso: 2017 Enero 2017].

[13] Raspberry Pi, «Raspberrypi,»
https://www.raspberrypi.org/downloads/. [Último acceso: 12 Enero 2017].

[14] «Sistemas de detección de matrículas,»
http://www.tecoyse.net/reconocimiento_matriculas_ por_cctv.php . [Último acceso: 14 Enero 2017].

[15] M. Shah, Fundamentals of computer vision 1, 1997.

[16] «Tutorial Python,»
http://docs.python.org.ar/tutorial/pdfs/TutorialPython2.pdf. [Último acceso: 22 Enero 2017].

[17] «Tutorial para Raspberry Pi,»
https://geekytheory.com/tutorial-raspberry-pi-gpio-parte-2-control-de-leds-con-python. [Último acceso: 22 Enero 2017].

[18] «Tutorial Python,»
http://www.iaa.es/python/curso-python-para-principiantes.pdf. [Último acceso: 20 Enero 2017].

[19] «Tutorial Python,»
https://media.readthedocs.org/pdf/entrenamiento-python-basico/latest/entrenamiento-python-basico.pdf. [Último acceso: 23 Enero 2017].

[20] «Tutoria Python,»
https://wiki.python.org/moin/SpanishLanguage. [Último acceso: 26 Enero 2017].

[21] «Tratamiento de imágenes opencv,»
http://www.informatica-juridica.com/trabajos/Informe_ OpenCV_Tratamiento_Imagenes.pdf . [Último acceso: 28 Enero 2017].

[22] «GitHub, OpenCV»
https://github.com/opencv/opencv. [Último acceso: 13 Enero 2017].

[23] «Surf feature detector,»
http://courses.cs.washington.edu/courses/cse576/13sp/projects/
project1/artifacts/woodrc/index.htm. [Último acceso: 16 Febrero 2017].

[24] «Librería tesseract para algoritmo ocr,»
https://code.google.com/p/tesseract-ocr/ . [Último acceso: 23 Febrero 2017].

[25] D. Lelis Baggio, S. Emami, D. Millan Escriva, K. Ievgen, N. Mahmood, J. Saragih y R. Shil
Mastering OpenCV with Practical Computer Vision Projects, Birmingham: Packt Publishing 2012.