



UNIVERSIDAD EUROPEA DE MADRID
ESCUELA DE ARQUITECTURA, INGENIERÍA Y DISEÑO

MÁSTER UNIVERSITARIO EN BIG DATA ANALYTICS
TRABAJO FIN DE MÁSTER

SISTEMA AVANZADO DE ASISTENCIA AL
CONDUCTOR (ADAS) BASADO EN
TÉCNICAS DE APRENDIZAJE AUTOMÁTICO
PARA LA DETECCIÓN Y TRANSCRIPCIÓN
DE CARTELES LUMINOSOS DE MENSAJE
VARIABLE

ALUMNO: GONZALO DE LAS HERAS DE MATÍAS

TUTOR: JAVIER SÁNCHEZ SORIANO

CURSO 2019-2020

Resumen

Dentro de las causas de los accidentes de tráfico, las distracciones son el motivo más habitual. Aunque en carretera exista numerosa señalización que contribuye a la seguridad, las señales luminosas de mensaje variable (VMS) requieren de una especial atención, que se transforma en distracción. Los dispositivos ADAS son sistemas avanzados que perciben el entorno y proporcionan una asistencia al conductor para su comodidad o seguridad. Este proyecto pretende desarrollar un prototipo de sistema de lectura de VMS mediante técnicas de aprendizaje automático, todavía no empleadas concretamente en este aspecto. El asistente consta de dos partes: una primera que reconoce la señal en el escenario y otra que extrae y reproduce el contenido. Para la primera, se ha confeccionado un conjunto de imágenes etiquetadas en formato PASCAL VOC, mediante anotaciones manuales, *scraping* y *data-augmentation*. Con este *dataset* se ha entrenado el modelo de reconocimiento de VMS, una RetinaNet con base de Restnet50 pre-entrenada sobre el *dataset* COCO. En el proceso de lectura, primeramente, se preprocesan y binarizan las imágenes para obtener la mejor calidad posible. Finalmente, la extracción se realiza mediante el modelo de OCR Tesseract en su versión 4.0 y la locución mediante el servicio *cloud* de IBM Watson Text to Speech.

Palabras claves

VMS, aprendizaje automático, ADAS, procesamiento de imágenes, percepción del entorno.

Abstract

Among the reasons of traffic accidents, distractions are the most common reason. Although there are many traffic signs on the road that contribute to safety, Variable Message Signs (VMS) require special attention, which is transformed into distraction. ADAS devices are advanced systems that perceive the environment and provide assistance to the driver for his comfort and safety. This project aims to develop a prototype of VMS reading system using machine learning techniques, still not yet used specifically in this aspect. The assistant consists of two parts: a first one that recognizes the signal on the stage and another one that extracts and reproduces the content. For the first one, a set of images were labeled in PASCAL VOC format, by manual annotations, scraping and data-augmentation. With this dataset, the VMS recognition model has been trained, a RetinaNet based of Restnet50 pretrained on the dataset COCO. In the reading process, firstly, the images were pre-processed and binarized to achieve the best possible quality. Finally, the extraction is made by the Tesseract OCR model, in its version 4.0 and the speech by the cloud service of IBM Watson Text to Speech.

Keywords

VMS, *machine learning*, ADAS, *image processing*, *environment perception*.

ÍNDICE DE CONTENIDOS

1. Introducción	1
1.1. Motivación y origen	1
1.2. Hipótesis	7
1.3. Objetivos	7
1.3.1. Objetivo General	7
1.3.2. Objetivos Específicos	7
1.4. Estructura del Documento.....	8
 2. Estado del Arte	 11
2.1. Sistemas de Seguridad en Vehículos	11
2.1.1. Primeros asistentes.....	11
2.1.2. ADAS	12
2.1.3. Vehículos semi/autónomos	14
2.2. Sistemas de Reconocimiento	17
2.2.1. Reconocimiento de Objetos	17
2.2.2. Reconocimiento de Texto	20
2.2.3. Detección en Carretera.....	21

3. Detección de VMS.....	23
3.1. Conjunto de Datos.....	23
3.1.1. Obtención de las imágenes.....	24
3.1.2. Conjuntos de Entrenamiento y Prueba	26
3.2. Reconocedor de VMS.....	27
3.2.1. Entrenamiento	28
3.2.2. Muestra de los resultados	30
4. Lectura del VMS	33
4.1. Preprocesamiento.....	33
4.1.1. Corrección del ángulo de giro	34
4.1.2. Recorte de la imagen	46
4.1.3. Ajuste de color para OCR	48
4.1.4. Muestra de los resultados	52
4.2. Reconomiento y Locución del Texto.....	53
4.2.1. Tesseract.....	53
4.2.2. Muestra de los resultados	54
4.2.3. Locución del Contenido	55
5. Conclusiones.....	57
5.1. Validación de Objetivos.....	57
5.2. Aportaciones.....	58
5.3. Trabajo Futuro.....	59

ANEXO A: Contenido del USB	61
ANEXO B: URLs del <i>Dataset</i>	63
Bibliografía	65

ÍNDICE DE TABLAS

Tabla 1. Parámetros del entrenamiento del modelo básico.	25
Tabla 2. <i>Hardware</i> empleado para el entrenamiento.	27
Tabla 3. Parámetros y resultado del entrenamiento principal.....	28
Tabla 4. Parámetros y resultado del reentrenamiento.....	30
Tabla 5. Puerta NOT aplicada a píxeles.	50
Tabla 6. Parámetros llamada IBM Watson Text to Speech.....	56
Tabla 7. Enlaces a los vídeos de los que se ha extraído las imágenes del <i>dataset</i>	63

ÍNDICE DE FIGURAS

Ilustración 1. Población, víctimas y vehículos registrados por RNB del país [4].....	1
Ilustración 2. Evolución de víctimas en España desde 1960 hasta 2018 [7].....	3
Ilustración 3. Evolución de víctimas mortales en España desde 1960 hasta 2018 [7]... Ilustración 4. Pilares de actuación propuestos por la OMS para mejorar la seguridad en	3
carretera [4].	4
Ilustración 5. Desempeño de una tarea vs dos tareas [17].	5
Ilustración 6. Ejemplo de VMS [23].	6
Ilustración 7. Aplicación de lectura de VMS [24].	6
Ilustración 8. Eras de los sistemas de seguridad en vehículos [15].	11
Ilustración 9. Taxonomía propuesta por [27].....	12
Ilustración 10. Niveles de conducción automática [31].	15
Ilustración 11. Comparativa entre Yolo v4 y otros detectores de objetos [45] sobre el <i>dataset</i> COCO [46].	19
Ilustración 12. Esquema de obtención del conjunto de datos.....	23
Ilustración 13. Métricas durante el entrenamiento del modelo básico.....	25
Ilustración 14. Ejemplo de sobreajuste durante el entrenamiento [65].	27
Ilustración 15. Cálculo del IoU.....	28
Ilustración 16. Métricas durante el entrenamiento principal.....	29

Ilustración 17. AP durante el entrenamiento principal.....	29
Ilustración 18. Convergencia con valores distintos de lr.....	29
Ilustración 19. AP durante el reentrenamiento.....	30
Ilustración 20. Ejemplo de VMSs reconocidos.....	31
Ilustración 21. Esquema de procesamiento del VMS hasta la locución.....	33
Ilustración 22. Resultados de las etapas del preprocesamiento de imágenes.....	34
Ilustración 23. Comparativa entre la imagen original y la convolucionada con el filtro gaussiano.....	36
Ilustración 24. Ejemplo del vector gradiente [69].....	38
Ilustración 25. Imágenes resultado del segundo paso [73].....	38
Ilustración 26. Ejemplo del Algoritmo de Canny: (a) imagen original, (b) orientación, (c) supresión no máxima, (d) histéresis de umbral [70].....	40
Ilustración 27. Resultado del Algoritmo de Canny en un VMS.....	40
Ilustración 28. (a) Representación de la recta PQ en el plano xy , (b) representación de todas las posibles rectas que pasan por P o Q y la recta PQ (punto de corte entre las rectas) el plano paramétrico ab [69].....	41
Ilustración 29. Representación gráfica de coordenadas polares [74].....	42
Ilustración 30. (a) Representación de una recta en el plano xy mediante coordenadas polares (ρ, θ) (b) Curvas sinusoides en el plano $\rho\theta$ que representan las posibles rectas que pasan por P o Q y la recta PQ en el punto de corte de ambas (c) División del plano $\rho\theta$ en celdas de acumulación [69].....	43
Ilustración 31. Resultado de la Transformada de Hough en un VMS.....	43
Ilustración 32. Resultado final del algoritmo de enderezamiento.....	46
Ilustración 33. Resultado final del algoritmo de recorte.....	48

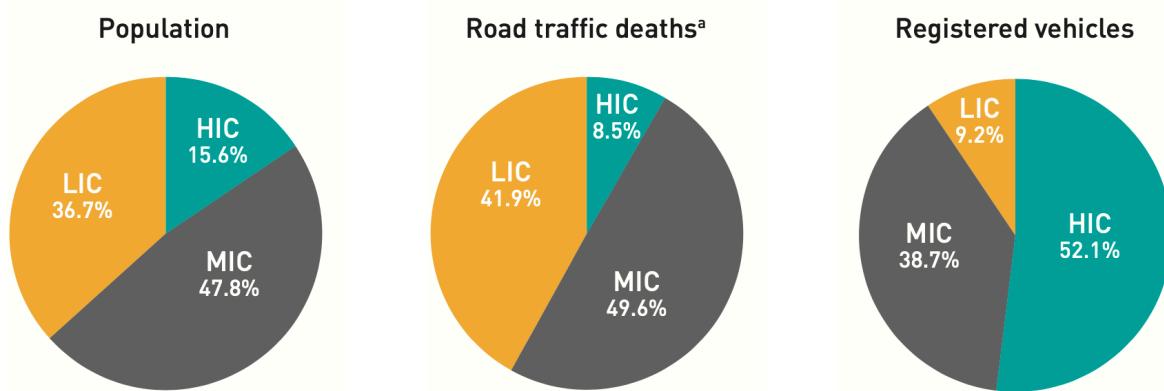
Ilustración 34. Resultado del primer paso del algoritmo de ajuste de color.....	50
Ilustración 35. (a) Imagen compuesta por un objeto A y un fondo B [69]. (b) Elemento estructurante [69]. (c) Traslaciones de <i>B</i> sobre <i>A</i> sin tocarla [69]. (d) $A \cdot B$ [69]. (e) Ejemplo de la operación morfológica de cerrado [83].....	51
Ilustración 36. Resultado del segundo paso del algoritmo de ajuste de color.....	51
Ilustración 37. Resultado final del algoritmo de ajuste de color.	52
Ilustración 38. Ejemplos del algoritmo de preprocesamiento (1º parte).....	52
Ilustración 39. Ejemplos del algoritmo de preprocesamiento (2º parte).....	53
Ilustración 40. Ejemplos de la transcripción con Tesseract (1º parte).....	54
Ilustración 41. Ejemplos de la transcripción con Tesseract (2º parte).....	55
Ilustración 42. Ejemplos de la transcripción sin parte del preprocesamiento.....	55

1. INTRODUCCIÓN

1.1. MOTIVACIÓN Y ORIGEN

Desde la democratización del vehículo privado, debido al abaratamiento de los costes de producción gracias a la fabricación en serie (conocido como fordismo), el parque automovilístico mundial no ha cesado de crecer. Prueba de ello, es el estudio [1], que con una muestra del 75% de la población mundial, estima que en 2002 las existencias eran de 800 millones de unidades y que en 2030 se alcanzarán los 2 billones. Analizando los datos de España, el total de vehículos se ha incrementado desde 15.696.715 en los años 90, hasta 33.729.982 en 2018 [2]. Teniendo en cuenta los datos del número medio de miembros de los hogares, recogido por el INE en 2018 [3], cada uno cuenta con casi 2 vehículos.

Population, road traffic deaths^a, and registered motorized vehicles, by income group



^a 30-day definition, modelled data. HIC = high-income countries; MIC = middle-income countries; LIC = low-income countries

Ilustración 1. Población, víctimas y vehículos registrados por RNB del país [4].

Este aumento ha traído consigo a la sociedad el problema de los accidentes de tráfico. Los datos de la Organización Mundial de la Salud (OMS) estiman que durante el periodo de 2011-2020, 1.1 millones de personas fallecieron a causa de accidentes de tráfico y entre 20 y 50 millones resultaron heridas [4]. Además, los datos muestran una distribución desigual según el nivel de Renta Nacional Bruta (RNB) del país. El 90% de los accidentes ocurren en países con niveles medios y bajos de RNB, aunque posean el 48% de los vehículos registrados en el mundo [4].

El estudio [4] clasifica los países según el método del World Bank Atlas [5]:

- **Low-Income Countries (LIC).** Aquellos con menos de 1.000 dólares de RNB.
- **Middle-Income Countries (MIC).** Aquellos entre 1.000 dólares y 12.000 dólares de RNB.
- **High-Income Countries (HIC).** Aquellos con más de 12.000 dólares de RNB.

En España, la Dirección General de Tráfico (DGT) ha elaborado una serie de anuarios estadísticos, que ilustran mediante una serie de gráficas la evolución desde 1960 hasta 2018 [6] [7]. La **Ilustración 2** muestra la variación del número de fallecidos, heridos hospitalizados y heridos no hospitalizados. En términos generales, el número de víctimas ha aumentado en los últimos años. La cifra de fallecidos y heridos hospitalizados se ha reducido mientras el número de heridos no hospitalizados se ha incrementado. Siguen produciéndose accidentes, pero cada vez la probabilidad de fallecimiento es menor. Esta reducción de las víctimas mortales se refleja en la **Ilustración 3**.

Las causas de los accidentes de tráfico se pueden clasificar según el factor de riesgo que los provoca. Se distinguen por factores humanos, mecánicos y del entorno (el estado del asfalto o de las señales de tráfico y las condiciones climatológicas). Según la DGT, en 2018 el 88% de los accidentes fueron consecuencia de comportamientos inadecuados del conductor [8] (conclusión es similar a la de [9] que afirma que el 90% se deben a causas humanas). En primer lugar, las distracciones (33%), seguidas del

exceso de velocidad (29%) y el consumo de alcohol (26%) [8]. La misma organización ha elaborado un documento que recoge las principales distracciones y expone cómo afectan a los accidentes [10]. En él se refleja que acciones como el uso del móvil, comer o fumar, son actividades que requieren de un tiempo y atención que reducen la concentración en la conducción. También el estado físico del conductor afecta a su tiempo de reacción y capacidad de distraerse. Esto conlleva un impacto directo en la distancia de frenado, lo que supone un riesgo grave. Además, muchas de estas conductas son conocidas por los conductores y muchos se declaran infractores [11].

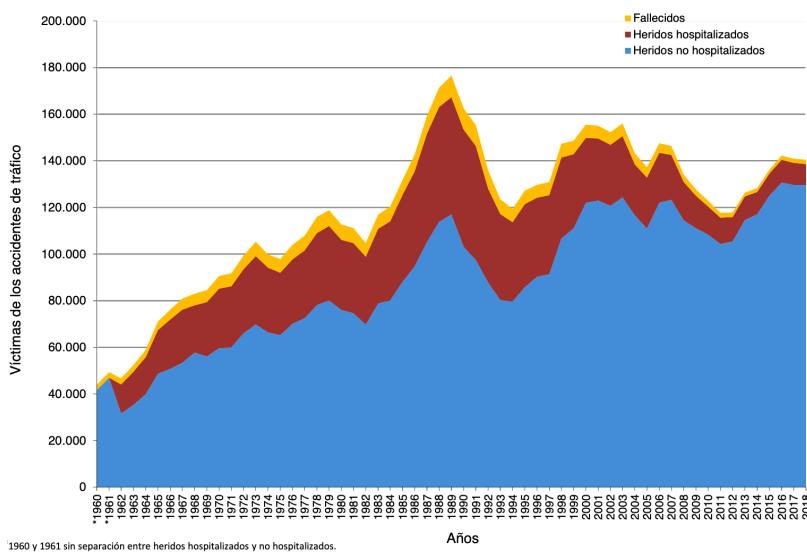


Ilustración 2. Evolución de víctimas en España desde 1960 hasta 2018 [7].

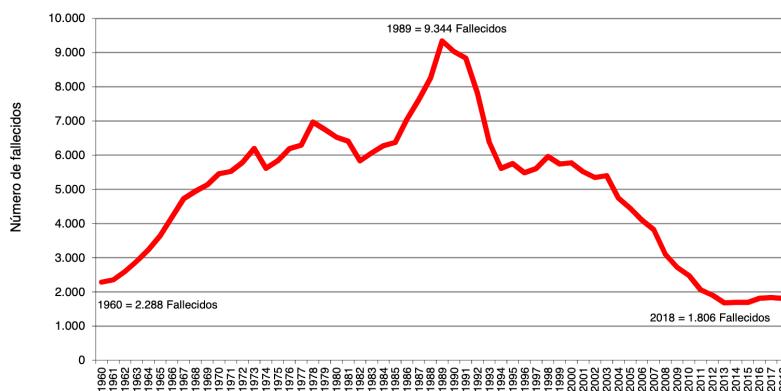


Ilustración 3. Evolución de víctimas mortales en España desde 1960 hasta 2018 [7].

La OMS en su informe de la década 2011-20 propone 5 pilares de actuación para mejorar la seguridad [4]:

1. **Gestionar la seguridad en carretera.** Desarrollar planes nacionales para el desarrollo de la seguridad vial.
2. **Carreteras más seguras.** Realizar inversiones en las infraestructuras automovilísticas para asegurar las mejores condiciones.
3. **Vehículos más seguros.** Automóviles con sistemas de seguridad más sofisticados.
4. **Conductores más seguros.** Campañas de concienciación y de formación.
5. **Mejores respuestas después del accidente.** Asegurar la rápida actuación de los servicios sanitarios y de emergencias.



Ilustración 4. Pilares de actuación propuestos por la OMS para mejorar la seguridad en carretera [4].

Ejemplos del 3º pilar son iniciativas como Prometheus [12] [13] creado por una asociación de fabricantes de vehículos e investigadores, o DRIVE (Dedicated Road Infrastructure for Vehicle safety in Europe) subvencionado por la Unión Europea [13]. Este último promulgó un gran número de trabajos sobre problemas fundamentales y prácticos, como GIDS (Generic Intelligent Driver Support) [14]. Su objetivo era “determinar los requisitos y las normas de diseño de una clase de sistemas de apoyo al conductor inteligente que se ajusten a los requisitos de información y a las capacidades de rendimiento de los conductores individuales” [14]. Era el inicio de lo que hoy en día conocemos como ADAS (Advanced Driver Assistance Systems), sucesores de los sistemas básicos de seguridad y posibilitadores en un futuro de la conducción autónoma [15]. Estos son una serie de funcionalidades que el vehículo ofrece al usuario para ayudarle en la conducción. Son capaces de asumir las decisiones del conductor y corregir

de manera proactiva y preventiva, sus errores. Algunos ejemplos son: la alerta de conductor agotado, aviso de colisión o la detección de señales de tráfico.

Los paneles luminosos de mensaje variable o VMS (Variable Message Signal) son dispositivos ATIS (Advanced Traveler Information System) situados sobre la carretera que constan de LEDs (Light-Emitting Diode) que destacan sobre un fondo negro. Son el mecanismo usado por los organismos de tráfico para comunicar información útil a los conductores, con el fin de mejorar su seguridad. Estos mensajes transmiten información mediante texto personalizado y/o pictogramas de señales de tráfico [16].

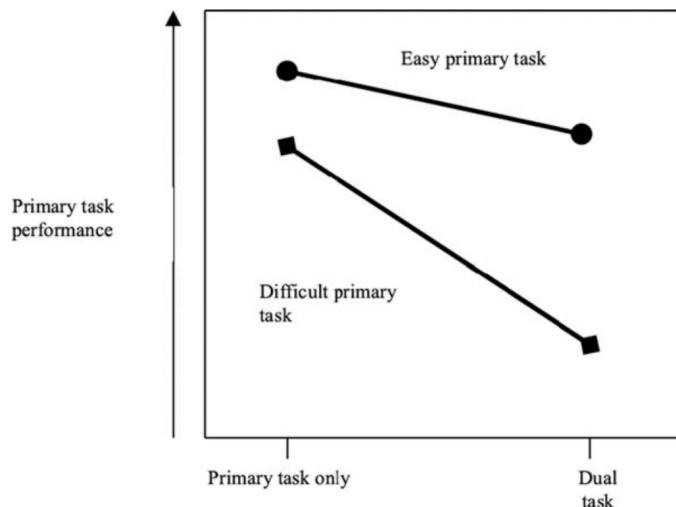


Ilustración 5. Desempeño de una tarea vs dos tareas [17].

Diversos estudios indican que los VMS tienen un impacto positivo en la conducción ya que reducen la velocidad [18] y descongestionan las retenciones causadas por accidentes o eventos [19]. El acto mismo de leer el VMS, provoca de por sí una reducción en la velocidad mientras se acercan a él [20]. No obstante, el hecho de invertir atención y tiempo en leer el mensaje y comprenderlo, es en sí misma una distracción y por tanto un riesgo. Además, si añadimos a una tarea principal, como es la conducción, la de lectura y entendimiento de la información, se obtiene una disminución de la efectividad de ambas tareas [17]. Existen planteamientos para reducir la atención requerida, simplificando mediante pictogramas o mensajes que consten de una sola

palabra. Estos últimos tienen una mayor efectividad a la hora de entender el mensaje que incluso los pictogramas, debido a que la comprensión depende del conocimiento previo del pictograma [21]. Existen convenciones como la de Viena [22] pero cada país es libre de alterar las señales, lo que dificulta el reconocimiento rápido de estos.



Ilustración 6. Ejemplo de VMS [23].



Ilustración 7. Aplicación de lectura de VMS [24].

Existen soluciones como READit VMS [24], que mediante una arquitectura cliente servidor y la geolocalización del usuario, realiza una locución del contenido del cartel o muestra un pictograma en una pantalla interna del vehículo. Estas aplicaciones requieren de una conectividad constante a la geolocalización y a internet para comprobar el VMS más próximo y pueden padecer problemas de latencia. Otro obstáculo es el estar limitados a los VMS registrados en el sistema. Debido a estas dependencias, no son sistemas autónomos que permitan una independencia del vehículo allá por donde transite. Los ADAS más parecidos son aquellos de reconocimiento de señales de tráfico que mediante técnicas más sofisticadas de visión por computador y aprendizaje automático, muestran al conductor la señal en una pantalla situada en el cuadro de mandos.

La motivación del presente proyecto, es aportar soluciones en el reto de la mortalidad en carretera, desarrollando un ADAS que intervenga en la mayor causa de accidentes, las distracciones [8] [11]. En carretera encontramos paneles con información que muchos estudios han reportado que ocasionan una reducción de la velocidad del vehículo. Sin embargo, la causa se debe a la atención requerida para leer y comprender

el mensaje [25] y que, añadido a la conducción, produce que esta sea menos eficiente [17]. Esta cuestión ha sido abordada mediante *software cliente servidor* [24] pero no con técnicas de aprendizaje automático y visión por computador. Ello permitirá al vehículo ser independiente de la latencia de la red, la geoposición y la base de datos de carteles. La solución consistirá en un reconocedor de VMS que reproduce el contenido del mensaje mediante voces sintéticas. Para ello, reconoce y segmenta los VMS para entregarlo al subsistema de OCR (Optical Character Recognition) que transcribe el contenido del panel y se anuncia mediante el servicio en la nube IBM Watson Text to Speech.

1.2. HIPÓTESIS

Es posible desarrollar un prototipo de sistema ADAS de lectura proactiva de VMS y aquellos conductores que dispongan de él, tendrán una conducción más segura al reducirse las distracciones y disponer de la información que transmiten.

1.3. OBJETIVOS

1.3.1. Objetivo General

El objetivo general es crear un prototipo de ADAS que mediante técnicas de aprendizaje automático sea capaz de detectar los carteles luminosos de mensaje variable colocados sobre la carretera y que reproduzca su contenido mediante voces sintéticas.

1.3.2. Objetivos Específicos

- **OE-1. *Dataset etiquetado.*** Recolectar imágenes de rutas de carretera con carteles luminosos y etiquetar la localización de este dentro de la instantánea.

- **OE-2. Modelo de detección de carteles.** Obtener un modelo de aprendizaje automático para la detección de los VMS
- **OE-3. Procedimiento de OCR.** Desarrollar un procedimiento de reconocimiento óptico de caracteres para la extracción y locución del contenido del cartel.

1.4. ESTRUCTURA DEL DOCUMENTO

El documento está dividido en 5 secciones principales, 2 anexos y la bibliografía:

- **Capítulo 1. Introducción.**

Se relata el origen y la motivación del proyecto, así como la hipótesis y el objetivo general subdividido en varios específicos.

- **Capítulo 2. Estado del Arte.**

Mediante un proceso de investigación bibliográfica, se describe el estado actual de la cuestión. Para ello, se han estudiado las soluciones actuales en el reconocimiento de objetos y texto, así como la situación de las distintas ayudas a la conducción y del vehículo autónomo.

- **Capítulo 3. Detección de VMS.**

Se expone el proceso de creación del conjunto de datos de imágenes etiquetadas necesario para obtener el modelo de reconocimiento de VMS, se explica la metodología del entrenamiento y se muestran los resultados obtenidos.

- **Capítulo 4. Lectura del VMS.**

Se describe detalladamente los algoritmos empleados en el preprocesamiento de imágenes, para aislar el contenido del cartel. También se desarrollan los fundamentos del modelo de OCR utilizado, detalla la configuración del servicio IBM Watson Text to Speech y muestra ejemplos de imágenes obtenidas.

- **Capítulo 5. Conclusiones.**

En este capítulo se realizan las últimas reflexiones del proyecto y se detallan los aportes realizados, así como las futuras líneas de actuación.

- **Anexo A. Contenido del USB.**

En este anexo se detallan los ficheros adjuntos del proyecto.

- **Anexo B. URLs del *Dataset*.**

En este anexo figura la información de los vídeos analizados para la creación del conjunto de entrenamiento y de prueba.

- **Bibliografía.**

En esta sección se enumeran todas las referencias bibliográficas citadas en todo el documento.

2. ESTADO DEL ARTE

2.1. SISTEMAS DE SEGURIDAD EN VEHÍCULOS

El informe [15] realizado por The Boston Consulting Group (BCG) para el The Motor & Equipment Manufacturers Association (MEMA) describe la evolución de los sistemas de seguridad en tres períodos:

1. Sistemas de ayuda y comodidad.
2. ADAS.
3. Vehículos semi/autónomos.

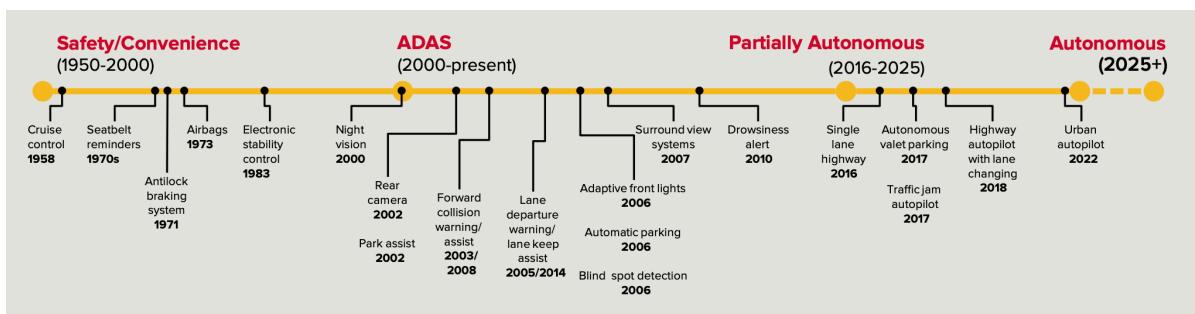


Ilustración 8. Eras de los sistemas de seguridad en vehículos [15].

2.1.1. Primeros asistentes

En el primer periodo se desarrollaron los primeros proyectos para mejorar la seguridad de los vehículos. Aunque puedan parecer simples, resultan de gran utilidad, puesto que no solo ayudan al conductor, sino que también le proporciona una mayor comodidad (aspecto estrechamente relacionado con la seguridad [26]).

Algunos de estos sistemas son:

- **Control de crucero**, para el mantenimiento automático de una velocidad prefijada sin intervención del conductor.
- **ABS (Antilock Braking System)** para evitar que durante el frenado alguno de los neumáticos se bloquee, lo que supondría una perdida de adherencia.
- **ESP (Electronic Stability Program)** que interviene en el comportamiento del vehículo para evitar perder el control.
- **Otras medidas de seguridad** como el airbag o el recordatorio del cinturón.

2.1.2. ADAS

Gracias al desarrollo de la tecnología, surgieron sistemas más avanzados que operaban sobre situaciones cada vez más complejas. El informe [27] propone una taxonomía basada en el tipo de sensor utilizado.

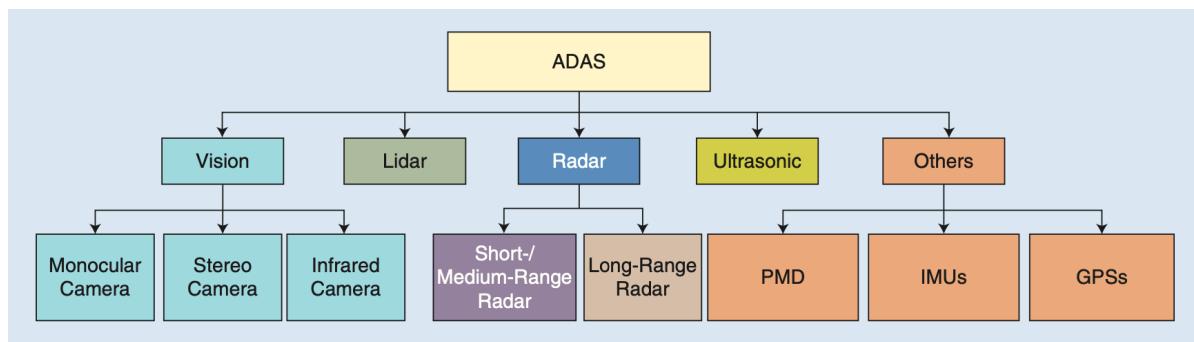


Ilustración 9. Taxonomía propuesta por [27].

- **Sistemas de visión.**

Disponen de cámaras (monoculares, estéreos e infrarrojas) situadas en puntos estratégicos del vehículo que aportan imágenes del entorno. Con las instantáneas recogidas, se extraen características de una imagen para obtener un conocimiento del entorno. Esta clase de sistemas tienen problemas con la profundidad y las obstrucciones del objetivo, sin embargo, son económicos [27].

- **LiDAR (Light Detection and Ranging).**

Es una tecnología que genera un entorno 3D proyectando rayos al entorno y midiendo la distancia a los distintos objetos. Esto permite al vehículo conocer en alta resolución los elementos que le rodean. Es una tecnología de vanguardia, pero a la vez costosa. No obstante, se estima que en 2025 habrá 35 millones de estas unidades [28].

Actualmente existe un debate entre el LiDAR y las cámaras convencionales. Empresas como Tesla, apuestan por la extracción de conocimiento mediante múltiples cámaras más otros dispositivos como radares. Otras como Waymo, creen que el LiDAR es la solución a futuro [29].

- **Radares.**

Los sistemas de radar miden la velocidad y la distancia de los objetos del entorno (gracias al efecto Doppler). Emiten una serie de microondas y miden el cambio de la frecuencia de onda. Un caso de uso es el Control de Crucero Adaptativo [27].

- **Ultrasonidos.**

Utilizando una serie de ondas sonoras, se mide la distancia a objetos cercanos. Un ejemplo del avisador de colisión durante el aparcado [27].

- **Otros ADAS.**

Todos estos ADAS son complementados con otra serie de funcionalidades para mejorar su precisión. Por ejemplo, los IMUs (Inertial Measurement Unit) o los GPSs (Global Positioning System), son sistemas auxiliares para la medición de distancias [27].

2.1.2.1. Implementaciones actuales

- **Detección de señales.**

Las diferentes señalizaciones como las de velocidad, stop o ceda al paso son reconocidas y presentadas al conductor en una pantalla a bordo. De esta manera el vehículo avisa de posibles excesos de velocidad o peligros.

- **Asistencia cambio de carril.**

Este sistema es una evolución del mantenimiento activo del carril. En el momento de cambiar, el asistente evalúa posibles colisiones y actúa en consecuencia.

- **Detección de peatones.**

Uno de los problemas en la circulación en vías urbanas son los posibles atropellos a peatones. Existen utilidades que los reconocen y los monitorizan para predecir un posible accidente y frenar antes de que ocurra.

2.1.3. Vehículos semi/autónomos

En la última era, que llega hasta nuestros días, el reto es crear automóviles capaces de ser conducidos por sí mismos. Con la ayuda de nuevos ADAS, como el piloto automático para atascos o el cambio de carril automático, estos son posibles. En 2025 se espera que haya 8 millones de vehículos autónomos y semiautónomos en todo el mundo [28].

El estándar J3016 “Levels of Driving Automation” de la Society of Automotive Engineers (SAE) establece 6 niveles para definir la autonomía de un vehículo. Van desde 0 (totalmente manual) hasta 5 (totalmente autónomo) [30].

A continuación, se detalla brevemente cada nivel:

Nivel 0: Sin Automatización. Aunque en este nivel el conductor es el encargado de realizar la “tarea de conducción dinámica”, puede contar con sistemas de ayuda. Un ejemplo sería el frenado de emergencia ya que técnicamente no “conduce” [30].

Nivel 1: Asistencia al Conductor. Este es el nivel más bajo de autonomía, en el que en ningún momento humano y vehículo manejan la dirección o los pedales al mismo tiempo. Asistentes como el Control de Crucero Adaptativo o el asistente de aparcamiento se encuentran en este nivel [30].

Nivel 2: Autonomía Parcial de la Conducción. En esta categoría se encuentran los vehículos que mediante sistemas ADAS, son capaces de tomar control de sí mismos en cualquier instante. En este nivel se incluye el Tesla Autopilot y el Cadillac Super Cruise [30].

Nivel 3: Autonomía Condicional. Estos vehículos cuentan con sistemas de detección del entorno y son capaces de llevar a cabo tareas que previamente se han informado al conductor. Esto es clave ya que este debe estar alerta para tomar de nuevo el control si el asistente es incapaz de llevarla a cabo. Un ejemplo es el sistema de Audi de conducción autónoma en atascos [30].

Nivel 4. Nivel Alto de Autonomía. Este nivel es muy similar al anterior, pero mucho más seguro, ya que el vehículo es capaz de manejar situaciones imprevistas sin poner en peligro a sus ocupantes. En esta categoría se encuentra el taxi autónomo de Waymo desplegado en Arizona [30].

Nivel 5. Autonomía Total. Los vehículos de este nivel son capaces de operar completamente sin la intervención humana. Pedales y volante no son necesarios ya que el pasajero se limita a indicar destino [30].

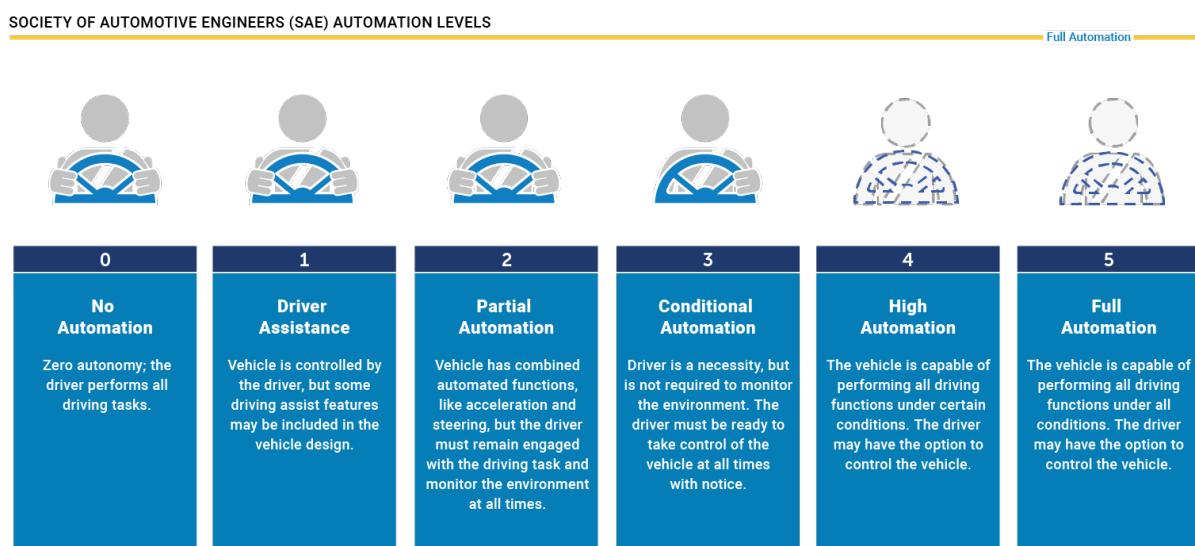


Ilustración 10. Niveles de conducción automática [31].

2.1.3.1. Proyectos comerciales

El coche autónomo es el gran anhelo de la industria automovilística. Actualmente, existen multitud de proyectos apoyados por estas compañías y en algunos casos se han creado alianzas con universidades, en pos de la investigación.

A continuación, se describen algunos de estos proyectos:

- **Google.**

La empresa norteamericana comenzó su proyecto de vehículo autónomo en 2009 dirigido por Sebastian Thrun, que también gestionaba un proyecto similar llamado Stanley (ganador del reto DARPA en 2005) en la Universidad de Stanford. Más tarde en 2016, crearon Waymo, una compañía a parte y exclusiva para el proyecto. El diseño de su vehículo está basado en una serie de sensores LiDAR, radares y cámaras de alta resolución para percibir el entorno [32].

- **Baidu.**

Es uno de los gigantes tecnológicos de China que está desarrollando un vehículo autónomo de código abierto llamado Apollo. Contiene entre otros, módulos de percepción, localización, planificación y control. El código fuente del proyecto se puede encontrar en GitHub y está apoyado por empresas como TomTom, Bosch, Intel, Ford, NVIDIA y Microsoft [32].

- **Tesla.**

Esta compañía dedicada a fabricación de coches eléctricos fue fundada en 2003, pero no fue hasta el año 2012 cuando comenzó a vender el primer vehículo, el Model S. La estrategia de Tesla para los coches autónomos es ofrecerlo como un ADAS, el cual se puede encender y apagar. El nombre designado para el sistema es Autopilot, el cual no contiene ningún sensor LiDAR en su diseño, ya que lo consideran caro e innecesario [32].

2.2. SISTEMAS DE RECONOCIMIENTO

2.2.1. Reconocimiento de Objetos

La historia de los reconocedores de objetos se divide en dos períodos: los modelos tradicionales y desde 2014, los basados en *deep learning* [33].

Los detectores de la primera etapa tuvieron que enfrentarse a la falta de recursos computacionales y de representación de características. Por esta razón, los algoritmos contenían rasgos hechos a mano y métodos que aprovechaban al máximo la potencia de la máquina [33].

- **Viola Jones** [34] [35].

Se trata de un reconocedor de caras extremadamente rápido, que deslizaba una ventana por toda la imagen, hasta que se identifica un rostro en alguna de las subsecciones.

- **HOG (Histogram of Oriented Gradients)** [36].

Este detector está diseñado para trabajar sobre una cuadrícula uniforme. Aunque puede utilizarse para detectar una variedad de objetos, fue motivado principalmente por el problema de la detección de peatones [33].

- **DPM (Deformable Part-based Model)** [37].

Este método es una extensión del detector HOG, en el que se aplica la estrategia de divide y vencerás. Por ejemplo, el problema de reconocer un coche se puede descomponer en localizar partes como las ruedas o las ventanas. Consiste en un filtro principal y varios secundarios configurados mediante aprendizaje supervisado como si tratasesen de variables latentes [33].

Con la evolución de las técnicas de aprendizaje automático surgieron las redes neuronales artificiales (ANN, Artificial Neural Network) y dentro de éstas, las Redes

Neuronales Convolucionales (CNN, Convolutional Neural Network) especializadas en el tratamiento de imágenes. Dentro de estas últimas, aquellas dedicadas a la detección de objetos se dividen dos grupos: los de una y dos etapas. Los primeros tratan la tarea como un problema de regresión aprendiendo las probabilidades de una clase y las coordenadas del cuadro delimitador. Los segundos en cambio, agrupan una serie de regiones de interés (1º paso) que son enviadas al clasificador de objetos y al delimitador de coordenadas (2º paso). Cada estrategia tiene puntos a favor y en contra. Por ejemplo, los de una etapa son más rápidos, pero tienen menos precisión [38].

Modelos de dos etapas:

- **R-CNN** [39].

Este sistema toma la imagen y la divide en unas 2000 regiones sobre las que se computa las características mediante una CNN. Finalmente, se clasifica cada región mediante SVMs (Support Vector Machine) lineales del tipo one-vs-rest [39].

- **Fast R-CNN** [40].

Apoyado sobre el modelo anterior, Fast R-CNN directamente extrae las características de la imagen completa, que son enviadas a la CNN para la clasificación y localización al mismo tiempo. Gracias a esta mejora, el tiempo de entrenamiento disminuye a la vez que aumenta la precisión [40].

- **Faster R-CNN** [41].

Este modelo elimina el cuello de botella que Fast R-CNN tenía a la hora de seleccionar la región de interés (RoI) [33], al usar una CNN llamada Region Proposal Network (RPN) para predecirla. Faster R-CNN fusiona la RPN y Fast R-CNN en una sola red, de manera que la primera le comunica a la segunda donde centrarse. Esto se consigue gracias a que comparten sus características convolucionales. De esta forma, la selección de la RoI es prácticamente a coste cero y el sistema es muy cercano al tiempo real [41].

Modelos de una sola etapa:

- YOLO (*You Only Look Once* v1 [42], v2/9000 [43], v3 [44], v4 [45]).

Es un sistema de reconocimiento de objetos en tiempo real gracias a que todo el proceso de detección es una sola red. El proceso consta de una fase en la que redimensiona la imagen a 488 x 488, para después ejecutar una sola CNN que devuelve la confianza del objeto detectado [42]. Existen diversas mejoras de este modelo que se centran en incrementar la precisión, pero manteniendo la rápida ejecución. La versión más reciente es la v4 [43] [44] [45].

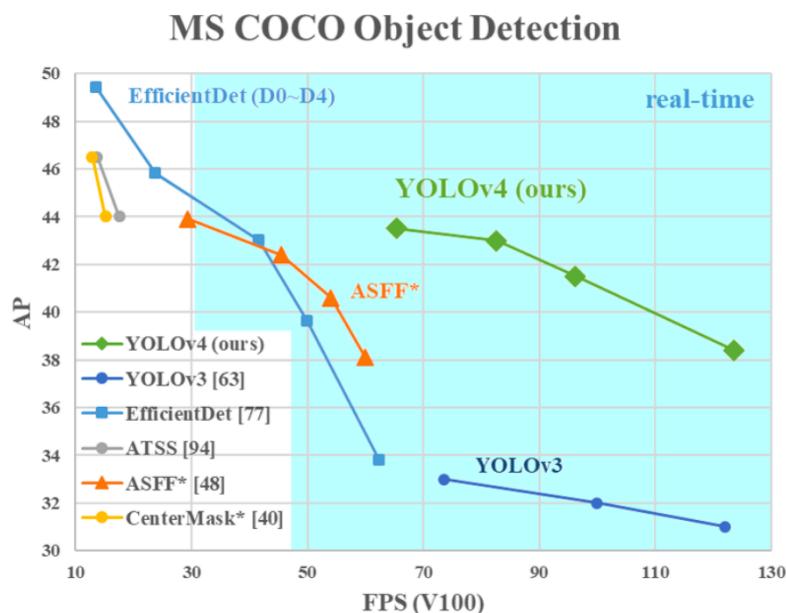


Ilustración 11. Comparativa entre Yolo v4 y otros detectores de objetos [45] sobre el *dataset* COCO [46].

- SSD (Single-Shot Detector) [47].

Su principal contribución es la introducción de las técnicas de detección de multi-referencia y multi-resolución, lo que mejora significativamente la precisión de la detección, especialmente para algunos objetos pequeños [33].

- RetinaNET [48].

Gracias a [48] se descubrió que el desequilibrio extremo de la clase de primer plano es la principal causa de que tengan menos precisión. Para solucionarlo,

introdujeron una nueva función de pérdida denominada “pérdida focal” para que el clasificador se centre en los ejemplos más difíciles de los mal clasificados. Así se consigue que este modelo alcance la precisión de los modelos de 2 etapas.

2.2.2. Reconocimiento de Texto

Al igual que con la detección de objetos, existen 2 épocas. Una primera en la que las técnicas se basaban en características “hechas a mano” para discriminar los caracteres y otra en la que predominan los modelos de aprendizaje automático [49] [50].

Era pre-aprendizaje profundo:

- **Connected Components Analysis (CCA).**

Estos clasificadores extraen primero los componentes candidatos para luego filtrar aquellos no textuales mediante reglas manuales o clasificadores entrenados [51]. Hay dos métodos Stroke Width Transform (SWT) y Maximally Stable Extremal Regions (MSER) [49].

- **Sliding Window (SW).**

Este modelo funciona deslizando una pequeña ventana multi-escala por todas las posibles localizaciones de la imagen, clasificando si hay texto o no [49].

En la era del *deep learning*, [50] plantea una taxonomía jerárquica dividida en detectores de texto, transcriptores, sistemas *end-to-end* y métodos auxiliares que mejoran la calidad de los modelos.

- **Detección.**

La detección de texto se puede definir como un subconjunto del problema de la detección de objetos en el que hay tres tendencias [50]:

- Reducción de *pipelines* para simplificar el proceso de entrenamiento y reducir el error.

- Descomposición en subtexto para después unirlas en una instancia completa.

- Reconocimientos específicos en casos como texto curvado, con formas irregulares o con fondos complejos.

- **Transcriptores.**

En los métodos tradicionales, el proceso consistía en preprocesamiento, segmentación y reconocimiento de caracteres. Sin embargo, la segmentación es costosa y alarga el tiempo de ejecución. Para evitar este paso, se usan métodos Connectionist Temporal Classification (CCT) [52] y mecanismos de atención [50].

- **Sistemas End-to-End.**

En lugar de dividir el problema principal en subproblemas de detección y reconocimiento, estos sistemas integran todo el proceso para realizar la lectura directamente de la imagen [50].

- **Técnicas auxiliares.**

Un ámbito importante son las técnicas que mejoran la calidad del entrenamiento como la creación de ejemplos sintéticos, la reducción de ruidos en la imagen o la incorporación de información del entorno [50].

2.2.3. Detección en Carretera

Uno de los ámbitos de aplicación de la detección de objetos son los sistemas de seguridad en vehículos. Algunos ejemplos son los siguientes:

- **Reconocimiento de Semáforos** [53] [54] [55].

Son asistentes que detectan este tipo de señalización, de manera que informan al conductor su estado actual. Incluso, si se conectara directamente al sistema de control del vehículo, este podría frenar automáticamente. Los principales retos que plantea este

ADAS están relacionados con la distinta tipología de los semáforos, ya que existen varios modelos según el país y a la existencia de intersecciones o varios carriles.

- **Reconocimiento de Señales** [56] [57].

La identificación de las señales tráfico es una de las tareas necesarias para la percepción del entorno. Estas son los principales mecanismos por los que los conductores reciben información (velocidad máxima, prohibiciones, intersecciones, etc.). Aunque actualmente existen ADAS comercializados (como el Toyota Road Sign Assist o RSA [58]), todavía es un reto en desarrollo. La principal problemática es la diversidad en tamaño y formas existentes.

- **Reconocimiento de Paneles** [59] [60] [61] [62].

Los paneles de información son un tipo de señalización situada encima de los carriles, que principalmente transmite la información mediante texto. Por lo tanto, el desafío para los asistentes radica en el reconocimiento de los caracteres, no solo en la identificación del objeto en el escenario.

3. DETECCIÓN DE VMS

El primer punto dentro del *pipeline* de reconocimiento es segmentar el VMS de la imagen original. En este capítulo se detalla el proceso de obtención del modelo de aprendizaje automático que lleva a cabo esta labor. Este consta de la confección del conjunto de imágenes y del entrenamiento del reconocedor.

3.1. CONJUNTO DE DATOS

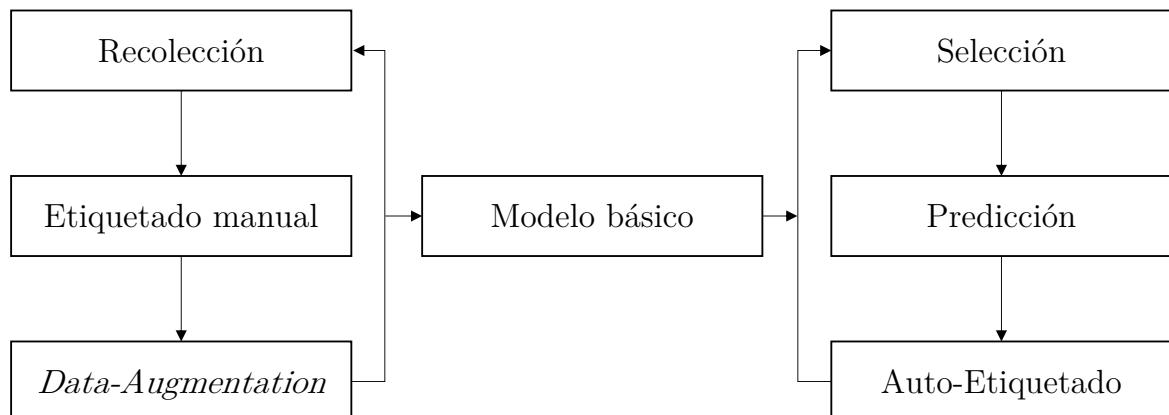


Ilustración 12. Esquema de obtención del conjunto de datos.

Gracias al estudio bibliográfico de la sección anterior, se ha descubierto la inexistencia de un conjunto de imágenes de VMS etiquetadas, por lo que su creación supone el primer paso del proyecto.

La estrategia es unir distintas fuentes para maximizar el número de ejemplos disponibles, con el menor trabajo manual. Este último es un punto clave, ya que cada imagen debe de ser anotada individualmente, lo que supone un gran gasto de tiempo.

Por ello, se ha diseñado un proceso en el que primero se obtendrá un *dataset* mínimo, para crear un modelo básico con el que poder procesar y etiquetar las imágenes iterativamente. Así, aunque la primera búsqueda será completamente manual, las posteriores consistirán en pequeños ajustes sobre imágenes extraídas de vídeos, que de otra manera conllevaría mucho trabajo.

3.1.1. Obtención de las imágenes

Como se ha comentado anteriormente, la adquisición de las primeras imágenes supone una tarea con una gran carga manual. No obstante, la técnica de *scraping* facilita esta labor. El *scraping* consiste en la extracción automática de información residente en internet mediante el uso de *scripts* personalizados. Para este proyecto se ha utilizado el lenguaje Python para su codificación.

La adquisición inicial se puede dividir en 3 pasos:

- 1. Recolección.**

Mediante búsquedas y recortes manuales combinados con *scripts* de *scraping*. Las fuentes de información consultadas son Google Imágenes, YouTube y varias webs.

- 2. Etiquetado.**

Cada imagen se anota manualmente mediante el *software* [63], que genera un fichero XML (Extensible Markup Language) en formato PASCAL VOC (Visual Object Classes).

- 3. Data-Augmentation.**

El *data-augmentation* es un método muy extendido y consiste en aplicar modificaciones a la imagen (rotaciones, recortes, traslaciones, etc.) con el objetivo de crear instancias aparentemente nuevas. Para este proyecto, debido a que el VMS se encontrará siempre en la posición superior de la imagen, se ha optado por voltear la

imagen sobre el eje y. De esa manera, los carteles que se encuentren a un lado se situarán en el opuesto, generando un nuevo ejemplo.

Una vez conseguida la primera versión del *dataset* (134 ejemplos de VMS), se ha entrenado con ella una RetinaNet [48] sobre un modelo Resnet50 [64] pre-entrenada sobre COCO [46]. Estos son los resultados.

Entrenamiento modelo básico	
Épocas	25
Nº de imágenes entrenamiento	134
Tiempo	\approx 01:30:00
Coeficiente de aprendizaje	1e-5
<i>Loss</i>	0.174

Tabla 1. Parámetros del entrenamiento del modelo básico.

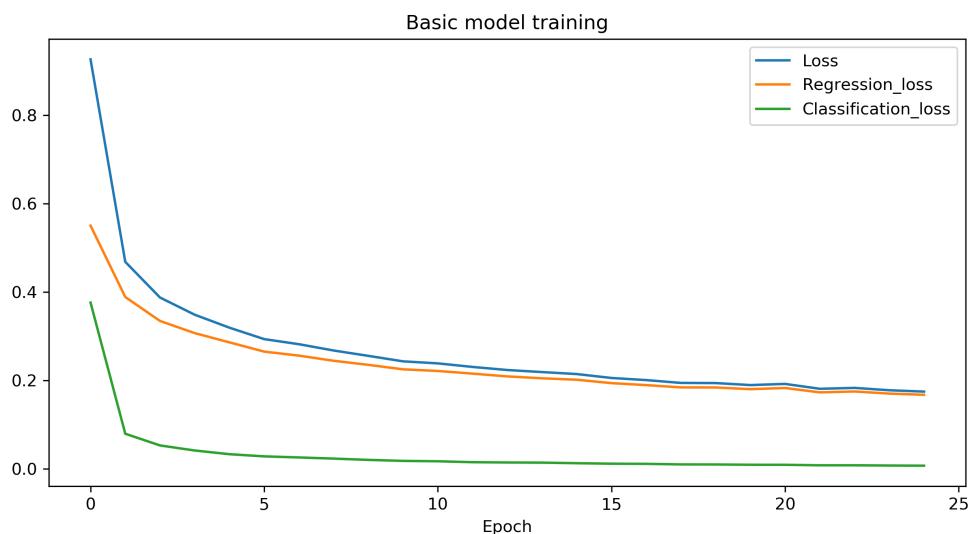


Ilustración 13. Métricas durante el entrenamiento del modelo básico.

Gracias a este modelo, comienza un proceso iterativo en el que se obtienen más rápidamente nuevas imágenes etiquetadas.

Este tiene 2 vías:

1. Manual.

Al igual que en la primera adquisición, se seleccionan manualmente las imágenes de VMS. La evolución reside en que el etiquetado lo realiza el modelo básico.

2. Semiautomático.

En este caso, se seleccionan vídeos que se analizarán mediante el modelo básico, para extraer una serie de imágenes candidatas ya etiquetadas de horas de grabación, que de otra forma resultaría mucho más tedioso.

Ya que el modelo no es perfecto (ni se pretende que lo sea) es necesario revisar la selección y detección automática. Por último, una vez validadas las imágenes con sus anotaciones, estas se someten al mismo *data-augmentation* (volteo sobre el eje Y).

3.1.2. Conjuntos de Entrenamiento y Prueba

Todo algoritmo de aprendizaje automático es sensible a sobre ajustar sus parámetros a los datos con los que ha sido entrenado. En esta situación, el modelo memoriza esta información, lo que le impide generalizar y, por tanto, tener un buen desempeño en situaciones reales.

Para evitar esta situación, se ha dividido el conjunto de datos en una parte exclusiva para el entrenamiento y otra para la validación. Este método es una práctica habitual y muy extendida para medir correctamente la bondad del modelo.

La siguiente imagen ilustra cuando se produce el sobreajuste (*overfitting* en inglés). Llegada cierta época, la generalización se transforma en memorización del conjunto de entrenamiento. Esto se manifiesta en el incremento del error en la validación después de una tendencia descendente, mientras que el del entrenamiento se reduce hasta casi desaparecer. Finalmente, el mejor modelo se encuentra justo antes de que esto se produzca.

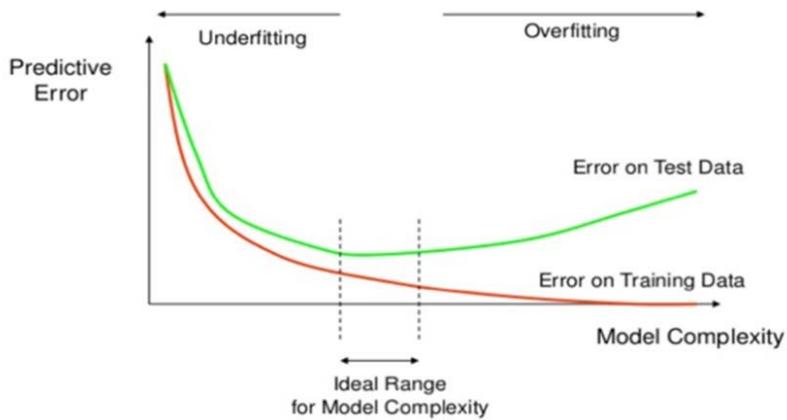


Ilustración 14. Ejemplo de sobreajuste durante el entrenamiento [65].

El conjunto de entrenamiento consta de 706 (324 con VMS) imágenes extraídas en parte de 19 vídeos de YouTube con una duración total de 05:19:27. El de prueba contiene 153 (56 con VMS) imágenes revisadas manualmente para asegurar la mejor comparativa.

Para más información acerca de las fuentes del conjunto de datos, consultar el Anexo B: URLs del *Dataset*.

3.2. RECONOCEDOR DE VMS

A continuación, se detalla el proceso de entrenamiento realizado para obtener el modelo final. Se ha utilizado una distribución pública llamada keras-retinanet [66] que trabaja sobre TensorFlow 2.0 [67] y el siguiente *hardware*:

Recursos <i>hardware</i>	
Procesador	Intel i7 9800K 3.6 GHz
Memoria RAM	32 GBs
Tarjeta Gráfica	Nvidia RTX 2080 Ti
Disco duro	1 Tb SSD M2

Tabla 2. *Hardware* empleado para el entrenamiento.

3.2.1. Entrenamiento

Se ha establecido como indicador a maximizar el AP (Average Precision), que es área bajo la curva de cobertura-precisión dada una IoU (Intersection over Union). La IoU indica el grado de solapamiento del área reconocida y el área real. Se emplea como umbral para hallar los Verdaderos Positivos (TP), los Falsos Positivos (FP) y los Falsos Negativos (FN) que definen el valor de la precisión y la cobertura.

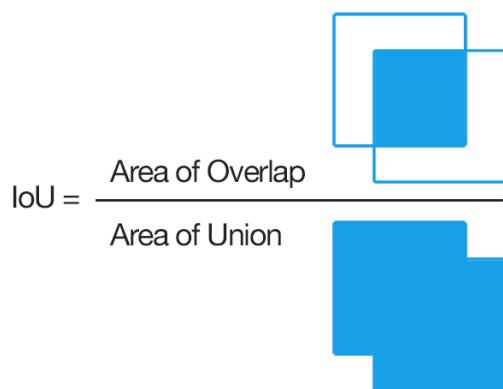


Ilustración 15. Cálculo del IoU.

Los parámetros y resultados del entrenamiento son los siguientes:

Resultado entrenamiento. Tiempo \approx 01:20:00			
Épocas	16	Mejor Época	7
Loss	0.008	Loss (7º época)	0.024
AP	1e-5	AP (7º época)	0.703
IoU	0.5	lr	1e-5

Tabla 3. Parámetros y resultado del entrenamiento principal.

Una vez terminado el primer entrenamiento, este se puede reanudar reduciendo el coeficiente de aprendizaje (lr) para mejorar ligeramente el modelo. La explicación reside en que el lr guía el descenso del gradiente por el espacio de errores hasta alcanzar el mínimo local (o en el caso óptimo, el mínimo absoluto). Un valor alto del lr provoca

la divergencia de la red, mientras que uno bajo, aunque requiera de más tiempo, convergerá.

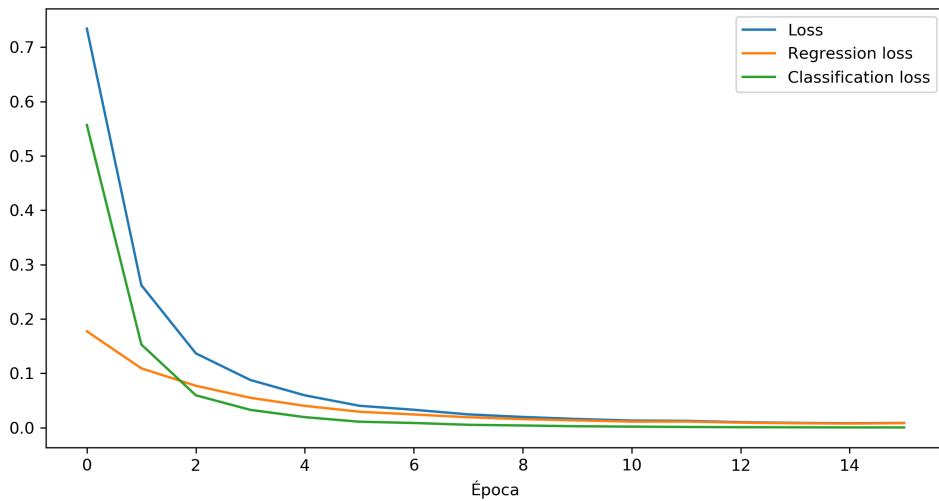


Ilustración 16. Métricas durante el entrenamiento principal.

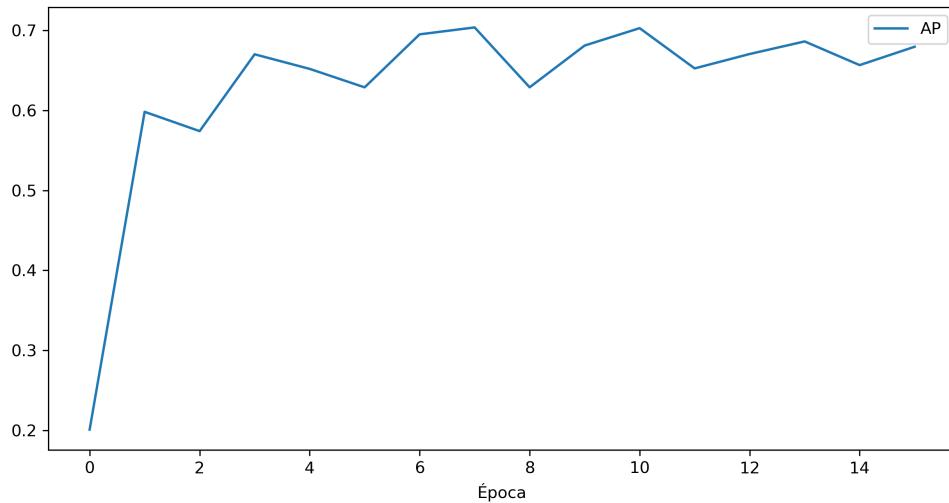


Ilustración 17. AP durante el entrenamiento principal.

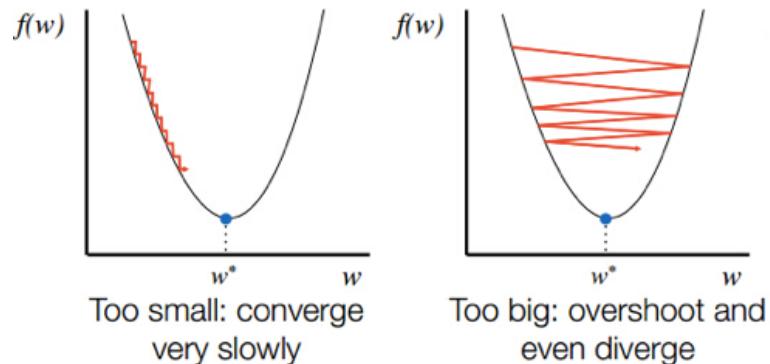


Ilustración 18. Convergencia con valores distintos de lr.

Los parámetros y resultados de la continuación del entrenamiento son los siguientes:

Resultado reentrenamiento. Tiempo \approx 01:15:00			
Épocas	14	Mejor Época	7
Loss	0.009	Loss (7º época)	0.024
mAP	1e-7	AP (7º época)	0.703
IoU	0.5	lr	1e-7

Tabla 4. Parámetros y resultado del reentrenamiento.

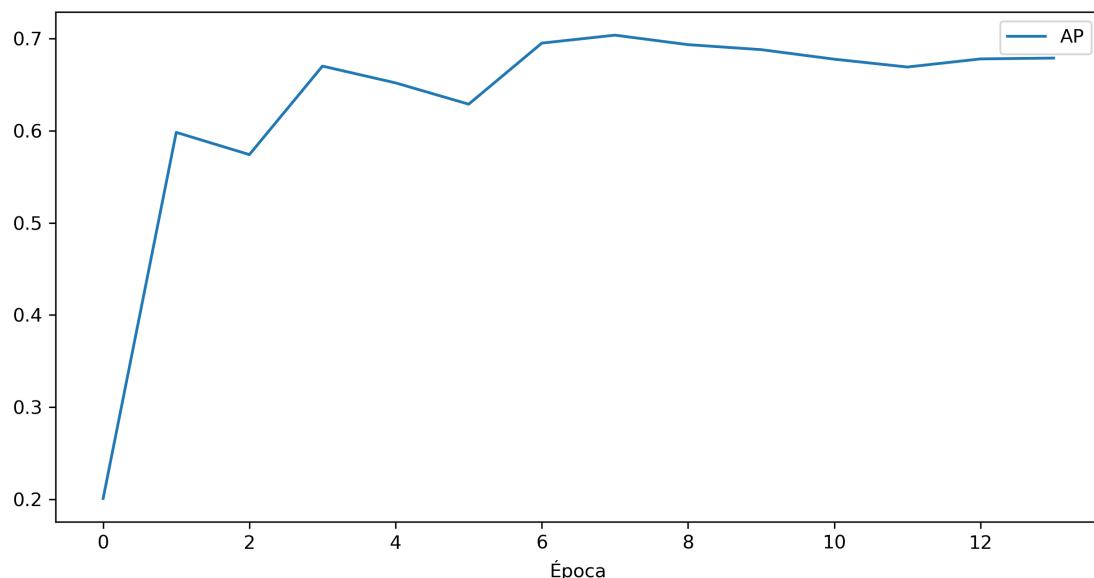


Ilustración 19. AP durante el reentrenamiento.

Vistos los resultados del reentrenamiento, se concluye que el modelo con el mejor AP sigue siendo el conseguido en la época 7. La reducción de lr no ha surgido el efecto buscado.

3.2.2. Muestra de los resultados

A continuación, se muestran algunos de los resultados obtenidos, junto con el grado de confianza.



Ilustración 20. Ejemplo de VMSs reconocidos.

4. LECTURA DEL VMS

Continuando en el *pipeline*, este capítulo se centra en la extracción y transcripción del contenido de los VMSs previamente reconocidos. Esta tarea se divide en 2 partes: el preprocesamiento de las imágenes y el reconocimiento del texto mediante el modelo OCR.

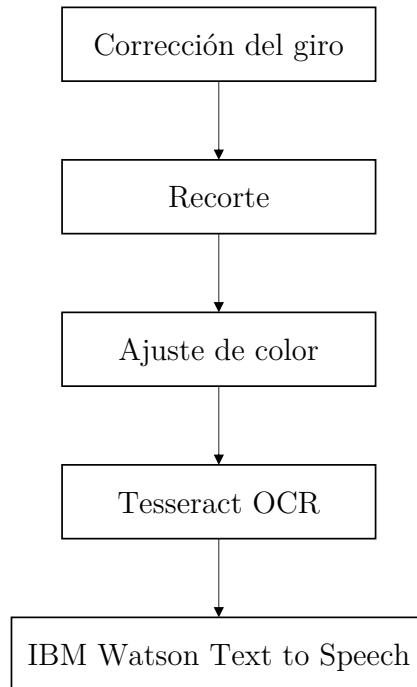


Ilustración 21. Esquema de procesamiento del VMS hasta la locución.

4.1. PREPROCESAMIENTO

Antes de obtener el texto, es preciso realizar una serie de operaciones sobre la imagen del panel, para asegurar la correcta transcripción del contenido. Estas consisten en la rotación de la imagen, el recorte de bordes y el ajuste del color.



Ilustración 22. Resultados de las etapas del preprocesamiento de imágenes.

4.1.1. Corrección del ángulo de giro

La primera cuestión por resolver es el ángulo de giro que el texto puede presentar, ya que una excesiva inclinación puede afectar al reconocimiento. La estrategia para resolverlo es inspeccionar la imagen para obtener rectas, cuyas pendientes se usarán para estimar el ángulo de rotación. Esto se logrará gracias a la Transformada de Hough, operación con la que se obtienen dos puntos de la recta con los que se puede calcular la pendiente de la recta que pasa por ellas. No obstante, antes de aplicar esta técnica es necesario ejecutar el Algoritmo de Canny para procesar la imagen marcando los bordes de los objetos que contiene. En los siguientes puntos se detallan los fundamentos matemáticos de ambos.

4.1.1.1. Algoritmo de Canny

El Algoritmo de Canny [68] [69] [70] [71] es una técnica desarrollada por John Canny en 1986 cuyo objetivo es detectar bordes dentro de una imagen cumpliendo los siguientes criterios:

- **Bajo ratio de error.** Todos los bordes se tienen que identificar correctamente sin que se produzcan falsos positivos.

- **Buena localización.** Los puntos que definen el borde reconocido tienen que encontrarse lo más cerca posible al auténtico borde.
- **Respuesta mínima.** El detector debe identificar solo un punto por cada punto real perteneciente al borde.

El algoritmo utiliza máscaras de convolución y se basa en el uso de la primera derivada, ya que su comportamiento en los cambios de intensidad resulta muy interesante en la detección de bordes (cuando haya cambios en la intensidad, el valor será constante durante la transición y 0 cuando no varíe [72]). Así pues, un cambio brusco se reflejará de igual manera en el valor de la primera derivada.

El trabajo de Canny consistía en expresar matemáticamente los anteriores criterios y encontrar soluciones óptimas. No obstante, debido a que no es tarea fácil, se descubrió que la derivada de una función gaussiana es una buena aproximación para el detector [69].

Primera derivada de una función gaussiana unidimensional:

$$\frac{d}{dx} \frac{1}{\sigma^2 \sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}} = \frac{-x}{\sigma^3 \sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}} \quad (1)$$

El Algoritmo de Canny se resume en 4 pasos:

1. Reducción de ruido.

El primer paso consiste en suavizar la imagen para reducir el posible ruido, no obstante, no hay que excederse ya que puede resultar en una pérdida de información de los contornos. Para ello, se convoluciona un filtro gaussiano $G(x, y)$ con la imagen de entrada $I(x, y)$ generando la nueva imagen con menos ruido $I_{suavizada}(x, y)$ tal como se indica en (2).

$$I_{suavizada}(x, y) = G(x, y) * I(x, y) \quad (2)$$

Utilizando fórmula (3) que indica una función gaussiana en 2 dimensiones, se desarrolla (4), que describe el cálculo del valor de cada elemento de un filtro de $(2k + 1) \times (2k + 1)$, para llegar finalmente a (5), en donde se indica la convolución con el valor aproximado del filtro. En este caso, se ha optado por un filtro de 5x5 (lo que implica que el valor k sea 2) y $\sigma = 1$.

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \quad (3)$$

$$G(x, y) = H_{ij} = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{(i-(k+1))^2 + (j-(k+1))^2}{2\sigma^2}\right); 1 \leq i, j \leq (2k+1) \quad (4)$$

$$I_{suavizada}(x, y) = \frac{1}{159} \begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix} * I(x, y) \quad (5)$$

(1) Imagen original $I(x, y)$



(2) Imagen convolucionada

$$I_{suavizada}(x, y)$$



Ilustración 23. Comparativa entre la imagen original y la convolucionada con el filtro gaussiano.

Es importante resaltar que el tamaño del filtro afecta al rendimiento del detector.

Cuando más grande, menos sensible será al ruido y el error de localización será más alto.

2. Calcular la magnitud del gradiente.

El siguiente paso es hallar la orientación y magnitud del vector gradiente de cada píxel (x, y) de $I_{suavizada}$, que viene dado por la ecuación (6).

$$\nabla I_{suavizada}(x, y) = \text{grad}[I_{suavizada}(x, y)] = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x} I_{suavizada}(x, y) \\ \frac{\partial}{\partial y} I_{suavizada}(x, y) \end{bmatrix} \quad (6)$$

Para estimar el valor de la derivada por cada eje, g_x y g_y , se aplica el operador de Sobel [69] que consiste en convolucionar la imagen y un filtro de 3x3 como se indica en (7) (8).

$$g_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} * I_{suavizada}(x, y) \quad (7)$$

$$g_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * I_{suavizada}(x, y) \quad (8)$$

Cada vector gradiente es bidimensional y su dirección está orientada hacia la máxima variación de intensidad, por tanto, es perpendicular al borde. Su magnitud $M_{suavizada}(x, y)$ se calcula mediante la ecuación (9) y la dirección $\alpha(x, y)$ se obtiene mediante la función \tan^{-1} (10).

$$M_{suavizada}(x, y) = \|\nabla I_{suavizada}(x, y)\| = \sqrt{g_x^2(x, y) + g_y^2(x, y)} \quad (9)$$

$$\alpha(x, y) = \tan^{-1} \left(\frac{g_y(x, y)}{g_x(x, y)} \right) \quad (10)$$

Tanto $\|\nabla I_{suavizada}(x, y)\|$ como $\alpha(x, y)$ son imágenes del mismo tamaño que la de entrada. Finalmente, para cada valor de α se redondea uno de los siguientes ángulos: 0° , 45° , 90° o 135° correspondientes a (d_1, d_2, d_3, d_4) .

En la Figura 14 encontramos un ejemplo del resultado de este paso.

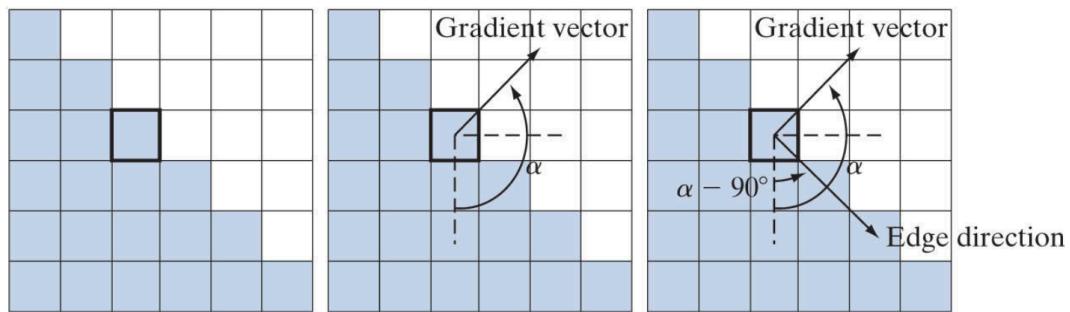
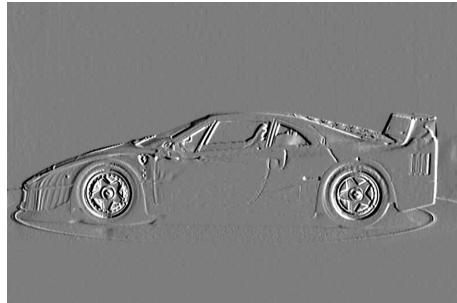


Ilustración 24. Ejemplo del vector gradiente [69].

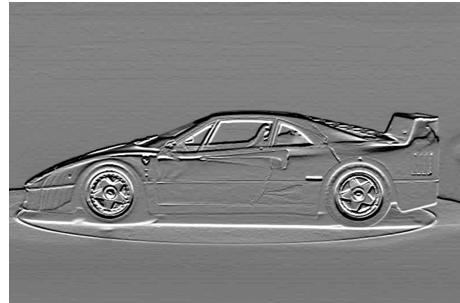
(1) Imagen suavizada $I_{suavizada}$



(2) Imagen g_x



(3) Imagen g_y



(4) Imagen final g_N



Ilustración 25. Imágenes resultado del segundo paso [73].

3. Supresión no máxima a la magnitud del gradiente.

El siguiente paso es afinar la imagen gradiente $\|\nabla I_{suavizada}\|$ mediante la supresión no máxima de la magnitud, ya que los bordes detectados son demasiados

gruesos debido a un ligero desenfoque. Para ello, se comparan el valor de la magnitud del gradiente de cada píxel, $\|\nabla I_{suavizada}\|$ en (x, y) , con el de sus dos vecinos en la dirección d_k que indique $\alpha(x, y)$. Si el valor del actual píxel es más pequeño que al menos uno de los dos, se le asigna el valor 0, $g_N(x, y) = 0$. En caso contrario, se asigna el valor de la magnitud del gradiente, $g_N(x, y) = \|\nabla I_{suavizada}\|$.

Al final este paso, tenemos $g_N(x, y)$ que es igual que $\|\nabla I_{suavizada}\|$ pero con bordes más delgados.

4. Histéresis de umbral a la supresión no máxima.

El último paso consiste en reducir los falsos positivos aplicando a $g_N(x, y)$ una función umbral. Algoritmos como el de Marr-Hildreth [69] emplean un único umbral que conlleva una serie de errores. Si este valor resulta demasiado bajo, seguirán existiendo puntos del borde falsos y si por el contrario es excesivamente alto, varios auténticos puntos del borde serán eliminados (falsos negativos). Canny resuelve parcialmente esta situación mediante el uso de la histéresis, que consiste en el empleo de un umbral inferior T_L y otro superior T_H , con una relación entre ellos dentro del rango 2:1 y 3:1.

$$g_{NH}(x, y) = g_N(x, y) \geq T_H \quad (11)$$

$$g_{NL}(x, y) = g_N(x, y) \geq T_L \quad (12)$$

Después de aplicar el umbral (11) (12), se eliminan de $g_{NL}(x, y)$ los valores de $g_{NH}(x, y)$ superiores a 0 (13), ya que los valores de este último ya se encuentran en el primero.

$$g_{NL}(x, y) = g_{NL}(x, y) - g_{NH}(x, y) \quad (13)$$

Considerando los valores de $g_{NH}(x, y)$ y $g_{NL}(x, y)$ distintos de 0 como bordes “fuertes” y “débiles”, los “fuertes” de $g_{NH}(x, y)$ son asumidos y marcados como puntos

de bordes válidos. No obstante, dependiendo del valor de T_H los bordes de $g_{NH}(x, y)$ tendrán saltos. Para rellenarlos, se sigue el siguiente procedimiento:

1. Localizar el siguiente píxel P no visitado perteneciente a un borde en $g_{NH}(x, y)$:
2. Marcar como borde válido todos los píxeles “débiles” de $g_{NL}(x, y)$ conectados a P en δ -connectivity.
3. Si todos los píxeles con valor superior a 0 $g_{NH}(x, y)$ han sido visitados, se salta al punto 4. En caso contrario, se vuelve al paso 1.
4. Establecer el valor 0 para todos los píxeles de $g_{NL}(x, y)$ no marcados como bordes válidos.

Finalmente, la imagen de salida del Algoritmo de Canny se consigue fusionando $g_{NL}(x, y)$ y $g_{NH}(x, y)$.

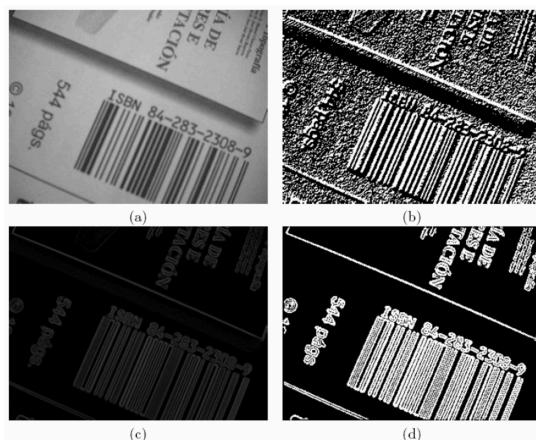


Ilustración 26. Ejemplo del Algoritmo de Canny: (a) imagen original, (b) orientación, (c) supresión no máxima, (d) histéresis de umbral [70].

(1) Imagen original



(2) Resultado final



Ilustración 27. Resultado del Algoritmo de Canny en un VMS.

4.1.1.2. Transformada de Hough

La transformada de Hough [69] [74] [75] [76] [77] es una técnica empleada para detectar figuras dentro de una imagen que se puedan expresar matemáticamente. Debido a que para el caso de uso de este proyecto solo interesan líneas rectas, esta explicación se centrará en ellas.

Considerando el punto $P(x_i, y_i)$ un punto en el plano xy , la ecuación general de una recta que pase por P es $y_i = ax_i + b$. Como por P pueden pasar infinitas rectas y todas ellas tienen que cumplir la ecuación anterior, esta se puede reescribir manteniendo fijo P para encontrar los valores a y b de todas dichas rectas, $b = -ax_i + y_i$ que se representa como una recta del plano ab . Tomando otro punto $Q(x_j, y_j)$ de xy , la expresión de todas las rectas que pasan por él será $b = -ax_j + y_j$. Para hallar en ab la recta PQ se debe resolver el sistema de ecuaciones y dos incógnitas o lo que es lo mismo, el punto (a', b') de corte entre las dos rectas.

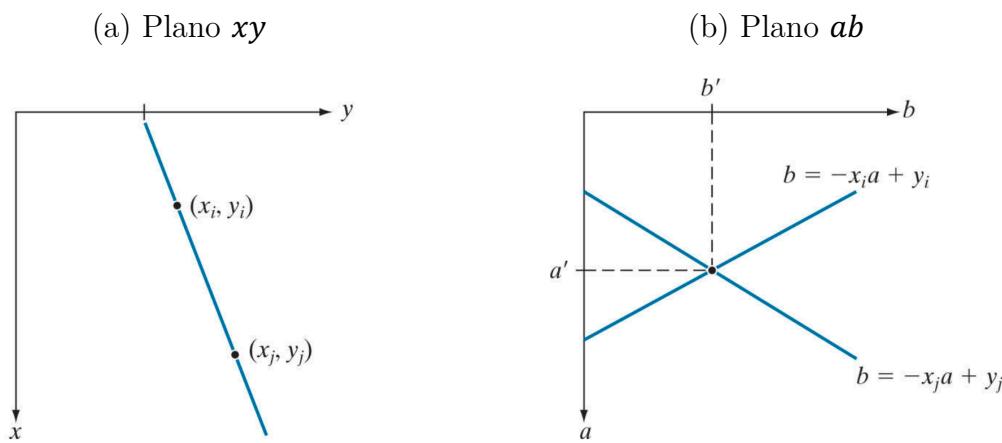


Ilustración 28. (a) Representación de la recta PQ en el plano xy , (b) representación de todas las posibles rectas que pasan por P o Q y la recta PQ (punto de corte entre las rectas) el plano paramétrico ab [69].

Esta representación en un espacio parametrizado presenta un problema. Según la recta se va transformando en vertical u horizontal, los valores de a y b se aproximan a infinito. Transformando las coordenadas a notación polar se resuelve, ya que un punto se representa mediante un par (ρ_θ, θ) en el que $\rho_\theta \geq 0$ y $\theta \in [0, 2\pi]$. Para transformar las coordenadas cartesianas se utilizan las fórmulas (14) y (15)

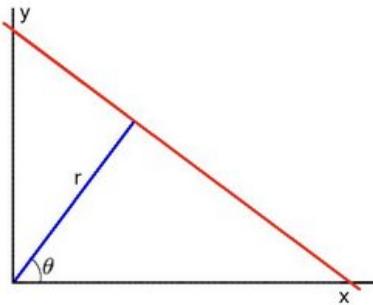


Ilustración 29. Representación gráfica de coordenadas polares [74].

$$\rho = \sqrt{x^2 + y^2} \quad (14)$$

$$\theta = \tan^{-1}\left(\frac{y}{x}\right) \quad (15)$$

Trasladando el anterior razonamiento en al espacio polar, utilizando la expresión (16) que indica una recta genérica y manteniendo x e y fijos, se obtiene la ecuación polar de todas las rectas dependientes de θ y ρ que pasan por (x, y) .

$$x \cos \theta + y \sin \theta = \rho \quad (16)$$

Análogamente al plano xy , la recta polar PQ se encuentra en (ρ', θ') , punto de corte entre las curvas sinusoides $x_i \cos \theta + y_i \sin \theta = \rho$ y $x_j \cos \theta + y_j \sin \theta = \rho$, que representan todas las rectas que pasan por P y Q respectivamente.

La Trasformada de Hough emplea este concepto. Si dentro de una imagen existe una línea recta r , todas las curvas de todas las posibles rectas que pasan por cada punto de r se deben cortar en el mismo punto (ρ', θ') del espacio $\rho\theta$. Por lo tanto, cuanto más curvas se intersequen en un mismo lugar, más certeza habrá de la existencia de r dentro de la imagen.

Computacionalmente, Hough divide el espacio $\rho\theta$ en celdas llamadas acumuladores, en el que el rango de los parámetros es (ρ_{min}, ρ_{max}) y $(\theta_{min}, \theta_{max})$ cumpliendo $-90^\circ \leq \theta \leq 90^\circ$ y $-D \leq \rho \leq D$, siendo D la distancia máxima entre las esquinas opuestas de la imagen.

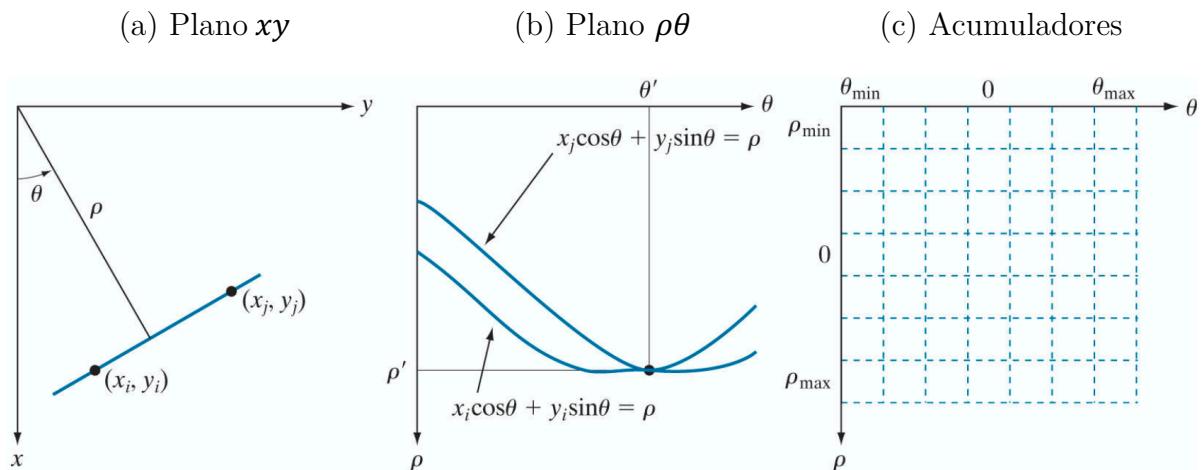


Ilustración 30. (a) Representación de una recta en el plano xy mediante coordenadas polares (ρ, θ) (b) Curvas sinusoides en el plano $\rho\theta$ que representan las posibles rectas que pasan por P o Q y la recta PQ en el punto de corte de ambas (c) División del plano $\rho\theta$ en celdas de acumulación [69].

En la subdivisión, la celda correspondiente a las coordenadas (i, j) está asociada al espacio de coordenadas (ρ_i, θ_j) y tiene un valor inicial de 0. Entonces por cada uno de los puntos (x_k, y_k) de la imagen que no pertenezcan al fondo, se calculan los valores ρ y θ de la sinusoide $x_k \cos \theta + y_k \sin \theta = \rho$ (que representa todas las rectas que pasan por dicho punto) redondeados a la celda (i, j) más cercana, la cual aumenta su valor en 1.

Finalmente, cada acumulador (i, j) tiene un valor de K que indica el número de puntos del plano xy que están en la recta definida por $x \cos \theta_j + y \sin \theta_j = \rho_i$. Estableciendo un umbral para K , se obtienen los valores ρ y θ de las rectas de la imagen.

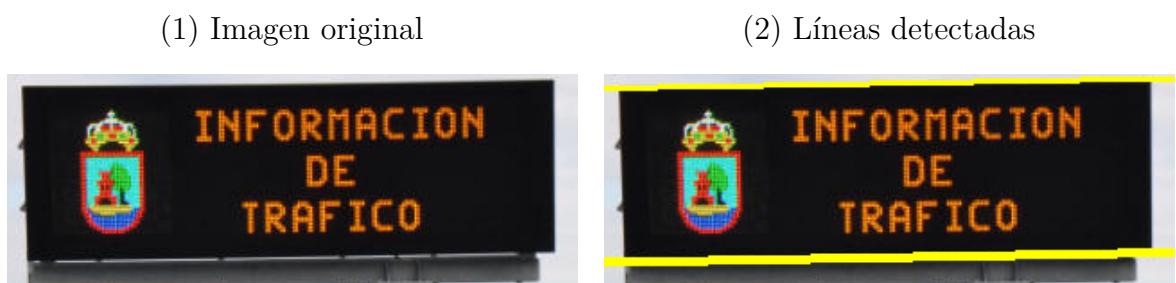


Ilustración 31. Resultado de la Transformada de Hough en un VMS.

4.1.1.3. Algoritmo de enderezamiento

Como se ha comentado anteriormente, la imagen del VMS puede presentar un pequeño ángulo de giro que afecta al OCR. Para corregirlo, se ha diseñado el siguiente procedimiento basado en el Algoritmo de Canny y la Transformada de Hough.

Algoritmo 1: Enderezamiento de la imagen

1. Detectar los bordes de la imagen de entrada mediante el Algoritmo de Canny.
 2. Encontrar las coordenadas polares (ρ, θ) de cada recta contenida en la imagen gracias a la Transformada de Hough.
 3. Calcular el ángulo de giro θ (en grados) considerado como la media aritmética de las pendientes de las rectas que superen el umbral establecido.
 4. Calcular la matriz de rotación R con respecto a θ y aplicarla a la imagen de entrada.
-

1. Detección de bordes.

Esta tarea se lleva a cabo gracias al Algoritmo de Canny sobre una imagen transformada a escala de grises, a la que previamente se ha aplicado un filtro gaussiano de 5x5 para reducir el ruido (aunque Canny ya aplique uno por defecto).

La parametrización de Canny utilizada está inspirada en [78]. Los umbrales se calculan automáticamente de la siguiente manera:

1. Obtener la intensidad media de los píxeles v .
2. Aplicar las siguientes fórmulas con $\sigma = 0.33$ para hallar el umbral inferior y superior:
 - 2.1. $T_l = \max (0, (1 - \sigma) \cdot v)$
 - 2.2. $T_H = \min (255, (1 + \sigma) \cdot v)$

2. Reconocimiento de rectas dentro de la imagen.

Se aplica la Transformada de Hough sobre la imagen de salida del Algoritmo de Canny, obteniendo una lista de pares (ρ, θ) . Los parámetros establecidos son:

- Distancia del acumulador en el eje $\rho = 1$.
- Distancia del acumulador en el eje $\theta = \frac{\pi}{180}$ radianes $= 1^\circ$.
- Umbral $T = 100$.

3. Cálculo de ángulo de giro θ .

Por cada par (ρ, θ) se aplica la fórmula (17) para hallar la ecuación de la recta en el plano xy . De ella se obtiene la pendiente a la cual es necesaria transformar a grados mediante (18) y se introduce en una lista. El ángulo de giro θ se estima mediante la media aritmética de todos los pendientes de las rectas detectadas.

$$y = \left(-\frac{\cos \theta}{\sin \theta} \right) x + \left(\frac{r}{\sin \theta} \right) \quad (17)$$

$$\text{grados} = \frac{a \cdot 180}{\pi} \quad (18)$$

4. Cálculo de la matriz de rotación R .

Por último, mediante una matriz de rotación R (19) aplicada a la imagen original se obtiene la imagen enderezada. Para ello, es necesario calcular α y β mediante (20) (21) sabiendo que $\text{centro} = \left(\frac{\text{ancho}}{2}, \frac{\text{alto}}{2} \right)$, $\text{escala} = 1$ y θ es el valor obtenido en el paso 3.

$$R = \begin{bmatrix} \alpha & \beta & (1 - \alpha) \cdot \text{centro}.x - \beta \cdot \text{centro}.y \\ -\beta & \alpha & \beta \cdot \text{centro}.x + (1 - \alpha) \cdot \text{centro}.y \end{bmatrix} \quad (19)$$

$$\alpha = \text{escala} \cdot \cos \theta \quad (20)$$

$$\beta = \text{escala} \cdot \sin \theta \quad (21)$$

A continuación, se muestra el resultado del algoritmo de corrección del giro:

(1) Imagen original



(2) Líneas detectadas



(3) Imagen rotada



Ilustración 32. Resultado final del algoritmo de enderezamiento.

4.1.2. Recorte de la imagen

Una vez corregido el ángulo de inclinación, el siguiente paso es recortar la imagen para que solo se muestre el interior del VMS. El objetivo es identificar las líneas que delimitan el panel y que marcan los puntos de corte. El siguiente algoritmo detalla el procedimiento a seguir.

Algoritmo 2: Recorte de la imagen

1. Hallar las ecuaciones de las rectas contenidas en la imagen.
 2. Calcular el punto de corte con los límites de la imagen.
 3. Identificar los puntos de recorte y extraer la subsección.
-

1. Hallar las ecuaciones de las rectas contenidas en la imagen.

Mediante los pasos 1, 2, y 3 del Algoritmo 1 se obtiene (ρ, θ) de las líneas horizontales (entre 0° y 1° de inclinación) r_{hi} y verticales (entre 88° y 92°) r_{vj} en la imagen. Seguidamente, se calculan las ecuaciones en el plano xy .

2. Calcular el punto de intersección con los límites de la imagen.

- Límites laterales.

Por cada recta r_{hi} , se calcula la intersección con los límites verticales $x = 0$ (22) y $x = w$ (23), siendo w el ancho de la imagen, para guardar la coordenada y de cada corte en la lista l_h . De esta manera, cada elemento de l_h es candidato a ser el límite del recorte horizontal.

$$\text{Corte lateral izquierdo} = \begin{cases} x = 0 \\ y = a * 0 + b = b \end{cases} \quad (22)$$

$$\text{Corte lateral derecho} = \begin{cases} x = w \\ y = aw + b \end{cases} \quad (23)$$

- Límites superior e inferior.

Por cada recta r_{vj} , se calcula la intersección con los límites horizontales $y = 0$ (24) e $y = h$ (25), siendo h el alto de la imagen, para guardar la coordenada x de cada corte en la lista l_v . De esta manera, cada elemento de l_v es candidato a ser el límite del recorte vertical.

$$\text{Corte superior} = \begin{cases} x = \left(\frac{0 - b}{a}\right) \\ y = 0 \end{cases} \quad (24)$$

$$\text{Corte inferior} = \begin{cases} x = \left(\frac{h - b}{a}\right) \\ y = h \end{cases} \quad (25)$$

Nota: El eje y crece positivamente hacia abajo.

3. Identificar los puntos de recorte y extraer la subsección.

- Recorte horizontal.

Identificar los puntos de corte superior I_{Hh} e inferior I_{Lh} de l_h que cumplan:

- $I_{Hh} = \max(p)$ siendo $p \in l_h$ y $0 \leq p \leq \frac{h}{6}$.

- $I_{Lh} = \min(p)$ siendo $p \in l_h$ y $\left(\frac{5}{6} h\right) \leq p \leq h$.

- Recorte vertical.

Identificar los puntos de corte izquierdo I_{Lv} y derecho I_{Rv} de l_v que cumplan:

- $I_{Lv} = \max(p)$ siendo $p \in l_v$ y $0 \leq p \leq \frac{h}{10}$.
- $I_{Rv} = \min(p)$ siendo $p \in l_v$ y $\left(\frac{9}{10} h\right) \leq p \leq h$.

Se ha establecido experimentalmente el rango de valores de p para I_{Hh} , I_{Lh} y I_{Lv} , I_{Rv} , así como los siguientes incrementos:

- $I_{Hh} = I_{Hh} + 0.03I_{Lh}$ y $I_{Lh} = I_{Lh} + 0.05I_{Lh}$.
- $I_{Lv} = I_{Lv} + 0.03I_{Rv}$ y $I_{Rv} = I_{Rv} + 0.05I_{Rv}$.

(1) Imagen rotada



(2) Líneas horizontales



(3) Líneas verticales



(4) Imagen final



Ilustración 33. Resultado final del algoritmo de recorte.

4.1.3. Ajuste de color para OCR

Una vez aislado el contenido VMS, la imagen está en condiciones de ser procesada por el Algoritmo 3 de cara al OCR. El objetivo es crear una nueva instantánea binarizada, es decir, con el texto en negro sobre fondo blanco.

Algoritmo 3: Ajuste de color para OCR

1. Binarizar la imagen.
 2. Unir trazos discontinuos en las letras.
 3. Ecualizar el histograma.
-

1. Binarizar la imagen.

1.1. Convertir a escala de grises.

Aplicando la siguiente fórmula (26) [79] se obtiene el valor del gris (siendo R el valor del canal de rojos, G el valor del canal de verdes y B el valor del canal de azules).

$$Gris = 0.299R + 0.587G + 0.114B \quad (26)$$

1.2. Aplicar el Método de Otsu.

La Binarización de Otsu [69] [80] [81] [82] es un método sin parámetros y no supervisado que consiste en encontrar automáticamente un umbral T que minimice la varianza $\sigma_{Within}^2(T)$ dentro de un mismo. Esto se define como la suma ponderada de las varianzas de ambas clases (27), siendo $n_B(T)\sigma_B^2(T)$ la del fondo (por debajo de T) y $n_F(T)\sigma_F^2(T)$ la del primer plano (por encima de T).

$$\sigma_{Within}^2(T) = n_B(T)\sigma_B^2(T) + n_F(T)\sigma_F^2(T) \quad (27)$$

En donde $n_B(T)$ y $n_F(T)$ son la probabilidad acumulada de que un píxel pertenezca a una clase y $[0, N - 1]$ son los posibles valores en la escala de grises.

$$n_B(T) = \sum_{i=0}^{T-1} p(i) \quad (28)$$

$$n_F(T) = \sum_{i=T}^{N-1} p(i) \quad (29)$$

Minimizar σ_{Within}^2 también se puede expresar como maximizar la varianza entre las clases $\sigma_{Between}^2$ que viene dada por (27).

$$\sigma_{Between}^2(T) = \sigma^2 - \sigma_{Within}^2(T) \quad (30)$$

1.3. Invertir el color de la imagen.

Como salida del Método de Otsu se obtiene una imagen de texto blanco sobre fondo negro. Por ello, es necesario aplicar la puerta lógica NOT sobre cada valor de esta, para invertirlo.

Píxel	Salida
0	255
255	0

Tabla 5. Puerta NOT aplicada a píxeles.

(1) Imagen recortada



(2) Imagen binarizada



Ilustración 34. Resultado del primer paso del algoritmo de ajuste de color.

2. Unir trazos discontinuos en las letras.

La imagen binarizada puede presentar pequeñas discontinuidades en el trazo de las letras. Para corregir estas imperfecciones que afectan al reconocimiento, se ha empleado la transformación morfológica de cerrado [69] [83] [84] que consigue paliar este problema.

Las transformaciones morfológicas son operaciones que trabajan sobre una imagen normalmente binarizada, desplazando un *kernel* por ella (de manera similar a la convolución 2D). El cerrado (31) consiste en una dilatación que llena los pequeños agujeros en el trazo, seguida de una erosión que corrige los píxeles no deseados que la primera operación ha aumentado.

$$A \cdot B = (A \oplus B) \ominus B \quad (31)$$

La dilatación transforma el valor un píxel en 1 si todos los píxeles por debajo del kernel son 1 y la erosión cuando al menos uno tiene el valor 1.

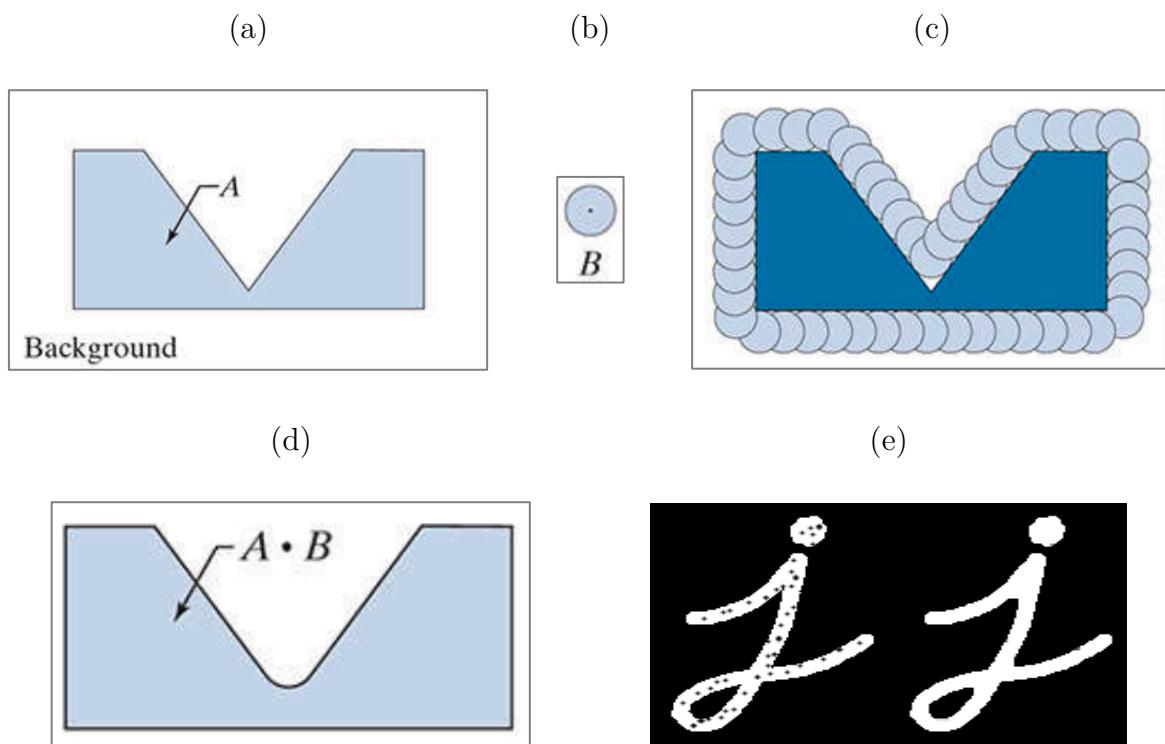


Ilustración 35. (a) Imagen compuesta por un objeto A y un fondo B [69]. (b) Elemento estructurante [69]. (c) Traslaciones de B sobre A sin tocarla [69]. (d) $A \cdot B$ [69]. (e) Ejemplo de la operación morfológica de cerrado [83].

(1) Imagen binarizada



(2) Imagen detectadas

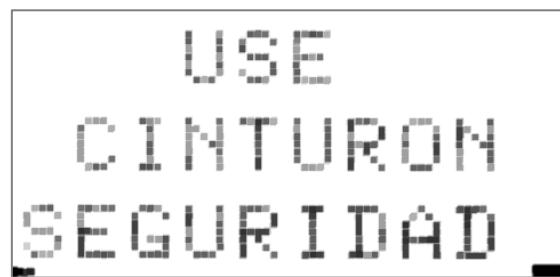


Ilustración 36. Resultado del segundo paso del algoritmo de ajuste de color.

3. Ecualización de histograma.

Por último, es necesario aumentar el contraste para que el posterior modelo de OCR sea capaz de reconocer el texto. Para ello, se ha ecualizado el histograma [85] [69] de la imagen $H(i)$, mapeándola a la distribución acumulativa normalizada $H'(i)$, que es más uniforme.

(1) Imagen rotada



(2) Imagen ecualizada



Ilustración 37. Resultado final del algoritmo de ajuste de color.

4.1.4. Muestra de los resultados

A continuación, se muestran algunos resultados del preprocesamiento.

(a) Imagen original



(b) Imagen final



VELOCIDAD
CONTROLADA
POR RADAR



58 MUERTOS
PUENTE
PILAR 2003



SALIDA 4
RAMON Y CAJAL
P CONGRESOS

Ilustración 38. Ejemplos del algoritmo de preprocesamiento (1º parte).



Ilustración 39. Ejemplos del algoritmo de preprocesamiento (2º parte).

4.2. RECONOCIMIENTO Y LOCUCIÓN DEL TEXTO

Una vez preprocesada la imagen del VMS, esta se encuentra en disposición de ser transcrita mediante el modelo de OCR Tesseract [86], y posteriormente locutado gracias al servicio *cloud* IBM Watson Text to Speech [87].

4.2.1. Tesseract

Tesseract [86] [88] [89] es un motor de reconocimiento óptico de caracteres que comenzó como un proyecto de tesis doctoral en los laboratorios de HP. La motivación era crear un sistema de OCR de calidad ya que no existía ninguno en la época. Además, para HP podría suponer un complemento de valor para su línea de escáneres. La versión

utilizada en el proyecto es Tesseract 4.0, que implementa redes neuronales recurrentes LSTM (Long Short-Term Memory), consiguiendo mejores resultados y mucho más rápido.

4.2.2. Muestra de los resultados

A continuación, se muestran algunos resultados del reconocimiento.

(a) Imagen procesada

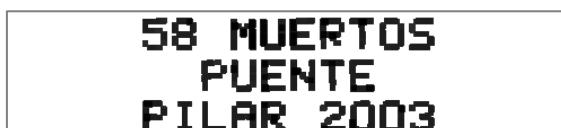


(b) Texto reconocido

PANEL EN PRUEBAS



VELOCIDAD CONTROLADA
POR RADAR



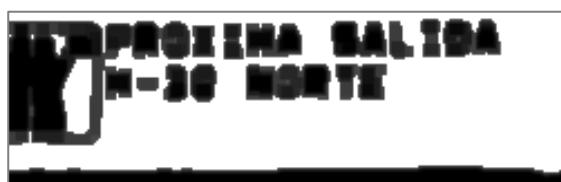
58 MUERTOS PUENTE PILAR 2003



SALIDA Il jearos Y casal
PP" CONGRESOS



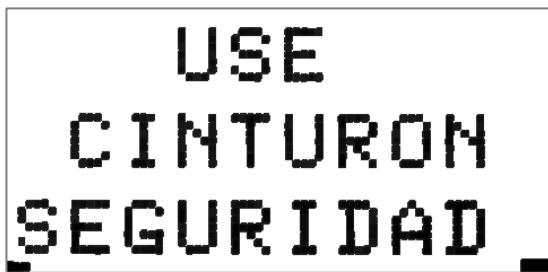
SE RESPONSABLE
maprio Y Y OMEQUEDOENCASA



PANA

Ilustración 40. Ejemplos de la transcripción con Tesseract (1º parte).

(a) Imagen procesada



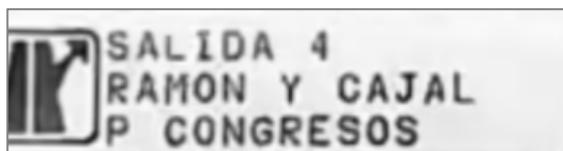
(b) Texto reconocido

USE CINTURON DE SEGURIDAD

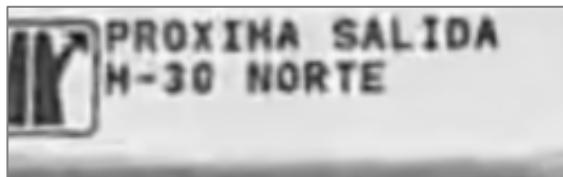
Ilustración 41. Ejemplos de la transcripción con Tesseract (2º parte).

Se ha detectado que, en algunas imágenes como las dos siguientes, el modelo de OCR tiene mucho mejor desempeño sin la última parte del algoritmo de preprocessamiento. Concretamente, sin la última fase de ajuste de color, las instancias con menor resolución tienen una mejor transcripción.

(a) Imagen procesada



(b) Texto reconocido

SALIDA 4 | RAMON Y CAJAL
P CONGRESOS

PROXIMA SALIDA M-30 NORTE

Ilustración 42. Ejemplos de la transcripción sin parte del preprocessamiento.

4.2.3. Locución del Contenido

El último paso del *pipeline* es la locución del contenido. Gracias al servicio *cloud* de IBM Watson Text to Speech [87] esta tarea es muy sencilla. Este ofrece al usuario una API REST que recibe el texto a transcribir y devuelve un fichero de audio.

Los parámetros de la llamada son los siguientes:

Parámetros llamada IBM Watson Text to Speech

Endpoint GET /v1/synthesize

Voz es-ES_EnriqueVoice

Formato de Audio wav

Servidor Londres

Tabla 6. Parámetros llamada IBM Watson Text to Speech.

Un ejemplo de una llamada sería el siguiente:

```
curl \  
-X GET \  
-u 'apikey: <API_KEY>' \  
--output <NOMBRE_FICHERO_SALIDA> \  
<URL>
```

5. CONCLUSIONES

La seguridad vial es un aspecto importante en una sociedad en la que cada vez hay más vehículos. Gobiernos, como el español, se han centrado en la creación de instituciones que velan por ella, como la DGT, además de asignar cuantiosas partidas presupuestarias destinadas a infraestructuras o campañas de concienciación. Por otro lado, el sector comercial se inclina por la investigación en tecnología que aumente la seguridad de los vehículos, los ADAS. Existe gran diversidad de estos asistentes que actúan en diversas situaciones, como posibles colisiones, cambios de carril, reconocimiento de señales, etc. En esa línea, este proyecto se ha centrado en crear un sistema ADAS, para la distracción que supone leer los VMS, mediante el uso de técnicas de aprendizaje automático que todavía no se han aplicado a este problema concreto.

5.1. VALIDACIÓN DE OBJETIVOS

A continuación, se analiza la conformidad de los objetivos marcados mediante los resultados obtenidos.

- **OE-1. *Dataset* etiquetado.** Se ha obtenido un conjunto de datos con 706 imágenes etiquetadas para el entrenamiento y 153 para la evaluación.
- **OE-2. Modelo de detección de carteles.** Se ha obtenido un modelo de detección de VMS un AP de 0.703.
- **OE-3. Procedimiento de OCR.** Se ha creado un procedimiento de extracción del texto del VMS, mediante la aplicación de sucesivos algoritmos.

5.2. APORTACIONES

Tras el análisis de los resultados del proyecto, las principales aportaciones son las siguientes:

1. Las varias tipologías de VMS afectan al procesamiento.

Dependiendo de la antigüedad y cantidad de información del VMS, este presenta distintos aspectos en su contenido. Se pueden encontrar paneles básicos o con señales de tráfico y logos en los laterales, además de carteles con la matriz de LEDs con más o menos resolución. Estos últimos se ha observado que el modelo de OCR detecta mucho mejor el texto sin aplicar el método de Otsu.

2. *Dataset* de VMS.

Antes de la realización del proyecto no existía un conjunto de imágenes etiquetadas con el que trabajar. Como resultado de este, ahora se dispone de un *dataset* con 706 instancias etiquetadas y 153 ejemplos de mayor calidad para una mejor evaluación.

3. Modelo de reconocimiento de VMS.

Se ha entrenado una red neuronal de tipo RetinaNet con base de Resnet50 y una precisión media de 0.703, que reconoce los VMS en una imagen e indica la localización al que asocia un grado de confianza. Este modelo se ha implementado mediante el repositorio público keras-retinanet [66] que opera sobre el *framework* de aprendizaje automático TensorFlow 2.0 [67] creado por Google.

4. *Pipeline* de procesamiento para la extracción de texto.

Gracias a la aplicación en cadena de diversos algoritmos sobre la imagen segmentada del VMS, se ha conseguido un prototipo para extraer el texto, que posteriormente el modelo de OCR Tesseract 4.0 transcribe. Este procesamiento consiste

en la corrección del ángulo de rotación, el recorte de los límites del panel y el ajuste de color.

5. Nueva línea de trabajo.

En definitiva, se ha abierto una nueva línea de trabajo e investigación acerca de la lectura de los paneles luminosos con mensaje variable, mediante las técnicas más vanguardistas de *machine learning*.

5.3. TRABAJO FUTURO

Durante el desarrollo del proyecto se han abierto nuevas líneas de investigación, que incluyen mejoras en el sistema actual y nuevas vías de actuación. Algunas de ellas se detallan a continuación:

- **Ampliar y mejorar el conjunto de imágenes.**

Uno de los aspectos clave a la hora de entrenar modelos de *deep learning* es el número de casos de ejemplo utilizados. Aunque con 706 imágenes se ha obtenido un modelo con 0.703 de AP sobre las 153 instancias de validación, estas cantidades se antojan escasas. Además, debido a la situación extraordinaria vivida durante la pandemia causada por el COVID-19, muchas de las imágenes han sido extraídas de la red y algunas tienen una resolución excesivamente baja. Para paliar esta situación, una de las líneas de actuación es la recolección de imágenes de calidad y la revisión más exhaustiva del etiquetado.

- **Mejora del modelo de reconocimiento.**

Los resultados experimentales de la detección muestran que el modelo confunde con cierta frecuencia los carteles estáticos con los VMS. En trabajos posteriores, se propone incrementar el nivel de precisión mediante el uso de varios modelos de aprendizaje automático.

- **Inclusión de señales de tráfico.**

Como se ha mencionado anteriormente, algunos VMS cuentan con señales de tráfico que complementan el mensaje de texto. En el prototipo actual, esta información es desechada, por lo que en el futuro se podría incluir esta información para completar el mensaje que las autoridades de tráfico quieren transmitir.

- **Sistema embebido.**

Para la comercialización del asistente, sería interesante crear un pequeño sistema dedicado con *hardware* de coste relativamente bajo. Para ello, se propone desarrollar con una Nvidia Jetson Nano y diversos periféricos necesarios, como una cámara y altavoz, una plataforma autónoma de lectura de VMS.

ANEXO A: CONTENIDO DEL USB

En este anexo se detallan los ficheros adjuntos del proyecto:

- Memoria.pdf
- dataset/
 - images/
 - 00_original/
 - 01_clean/
 - 02_resized/
 - 03_train/
 - 04_autolabeling/
 - 05_test/
 - data.csv
 - template.xml
 - labeling_tool/
 - 01_load_new_images.py
 - 02_resize_and_flip.py
 - commands.txt
 - train.csv
 - reconocedor_vms/

- keras_retinanet/
- notebook_data/
- scripts/
 - 03_image_autolabeling.py
 - 04_youtube_recognition.py
 - commands.py
- snapshots/
 - pretrained/
 - vms_model.h5
- test_ocr/
- utils_ocr/
 - detectedLine.py
 - functions.py
- utils_vms/
 - configuration.py
 - functions.py
 - visualization.py
- annotations_test.csv
- annotations_train.csv
- classes.csv
- Model_evaluation.ipynb
- Model_train.ipynb
- OCR.ipynb

ANEXO B: URLs DEL *DATASET*

En este anexo figuran la información de los vídeos analizados para la creación del conjunto de entrenamiento y de prueba.

URL	Duración	URL	Duración
youtu.be/MFzuxq4V0XI	00:06:59	youtu.be/dJcH8YFuvY4	00:25:12
youtu.be/GREMRp7rvoY	00:08:16	youtu.be/M2rvG-e04HE	00:16:22
youtu.be/lNhy2mT94Ao	00:05:22	youtu.be/8ifk3BHz1_c	00:13:35
youtu.be/37YgdfidwkA	00:19:58	youtu.be/6bkOZcBECsk	00:12:37
youtu.be/JctnDDdoy0A	00:22:56	youtu.be/afCyz52txC0	00:06:40
youtu.be/H1gxWeWsa_E	01:20:25	youtu.be/vU82-jnUi_E	00:07:57
youtu.be/UZFLDp_LLj4	00:27:23	youtu.be/D5RHKJNhw7I	00:08:01
youtu.be/tz8bEIirIx4	00:10:50	youtu.be/S1DE3pvnG8s	00:12:33
youtu.be/QJ_XSlOeCBw	00:10:48	youtu.be/XLQbclKjNrw	00:03:36
youtu.be/4s-WfvYUbPM	00:19:57		

Tabla 7. Enlaces a los vídeos de los que se ha extraído las imágenes del *dataset*.

BIBLIOGRAFÍA

- [1] J. Dargay, D. Gately y M. Sommer, «Vehicle Ownership and Income Growth, Worldwide: 1960-2030,» *The Energy Journal*, vol. 28, pp. 143-170, 2007.
- [2] Dirección General de Tráfico - Ministerio del Interior, «Anuario Estadístico General,» 2018. [En línea]. Available: <http://www.dgt.es/es/seguridad-vial/estadisticas-e-indicadores/publicaciones/anuario-estadistico-general/>.
- [3] Instituto Nacional de Estadística, «Encuesta Continua de Hogares,» 2018. [En línea]. Available: https://www.ine.es/prensa/ech_2018.pdf.
- [4] Organización Mundial de la Salud, «Decade of Action for Road Safety 2011–2020,» 2011. [En línea]. Available: <https://www.who.int/publications/i/item/decade-of-action-for-road-safety-2011-2020>.
- [5] The World Bank, «The World Bank Atlas method - detailed methodology,» [En línea]. Available: <https://datahelpdesk.worldbank.org/knowledgebase/articles/378832-the-world-bank-atlas-method-detailed-methodology>.

- [6] Dirección General de Tráfico - Ministerio del Interior, «Las Principales Cifras de la Siniestralidad Vial,» 2018. [En línea]. Available: <http://www.dgt.es/es/seguridad-vial/estadisticas-e-indicadores/publicaciones/principales-cifras-siniestralidad/>.
- [7] Dirección General de Tráfico - Ministerio del Interior, «Anuario Estadístico de Accidentes,» 2018. [En línea]. Available: <http://www.dgt.es/es/seguridad-vial/estadisticas-e-indicadores/publicaciones/anuario-estadistico-accidentes/>.
- [8] Dirección General de Tráfico - Ministerio del Interior, «Las distracciones causan uno de cada tres accidentes mortales,» 2018. [En línea]. Available: http://www.dgt.es/es/prensa/notas-de-prensa/2018/20180917_campaña_distracciones.shtml.
- [9] A. Smiley y K. A. Brookhuis, «Alcohol, Drugs and Traffic Safety. Road Users and Traffic Safety,» pp. 83-104, 1987.
- [10] Dirección General de Tráfico - Ministerio del Interior, «Distracciones al Volante,» [En línea]. Available: http://www.dgt.es/PEVI/documentos/catalogo_recursos/didacticos/did_adultas/Distracciones_al_volante.pdf.
- [11] Dirección General de Tráfico - Ministerio del Interior, «Las distracciones son la causa de uno de cada cuatro accidentes,» 2019. [En línea]. Available: <http://www.dgt.es/es/prensa/notas-de-prensa/2019/Las-distracciones-son-la-causa-de-uno-de-cada-cuatro-accidentes.shtml>.

- [12] J. Billington, «The Prometheus project: The story behind one of AV's greatest developments,» 2018. [En línea]. Available: <https://www.autonomousvehicleinternational.com/features/the-prometheus-project.html>.
- [13] K. A. Brookhuis, D. de Waard y W. H. Janssen, «Behavioural impacts of Advanced Driver Assistance Systems-an overview.,» *European Journal of Transport and Infrastructure Research.*, vol. 1, 2001.
- [14] J. A. Michon, «Generic Intelligent Driver Support: A comprehensive report on GIDS,» pp. 3-18.
- [15] BCG, «A Roadmap to Safer Drivingthrough Advanced Driver Assistance Systems,» 2015. [En línea]. Available: https://image-src.bcg.com/Images/MEMA-BCG-A-Roadmap-to-Safer-Driving-Sep-2015_tcm9-63787.pdf.
- [16] S. Nygårdhs y G. Helmers, «VMS – Variable Message Signs. A literature review,» 2007.
- [17] G. Gopher, «Attentional Allocation in Dual Task Environments.,» *Attention and Performance III.*, 1990.
- [18] V. G. B. Kolisetty, T. Iryo, Y. Asakura y K. Kuroda, «Effect of Variable Message Signs on Driver Speed Behavior on a Section of Expressway under Adverse Fog Conditions - A Driving Simulator Approach,» *Journal ofAdvanced Transportation*, vol. 40, n° 1, pp. 47-74, 2005.

- [19] S. Peeta y J. Ramos, «Driver response to variable message signs-based traffic information,» *Intelligent Transport Systems, IEE Proceedings*, 2006, pp. 2-10.
- [20] A. Calvi, M. Blasiis y C. Guattari, «The Effectiveness of Variable Message Signs Information: A Driving Simulation Study,» *Procedia - Social and Behavioral Sciences*, 2012, pp. 693-702.
- [21] J. Roca, B. Insa y P. Tejero, «Legibility of Text and Pictograms in Variable Message Signs: Can Single-Word Messages Outperform Pictograms?,» *Human factors*, vol. 60, pp. 384-396, 2018.
- [22] P. Simlinger, S. Egger y C. Galinski, «Proposal on unified pictograms, keywords, bilingual verbal messages and typefaces for VMS in the TERN,» International Institute for Information Design, 2008. [En línea]. Available: https://ec.europa.eu/transport/road_safety/sites/roadsafety/files/pdf/projects_sources/in-safety_d2_3.pdf.
- [23] Autopistas.com, «Paneles de mensajería variable,» [En línea]. Available: <https://www.autopistas.com/blog/paneles-de-mensajeria-variable/>.
- [24] Universitat de València, «READit VMS,» 2018. [En línea]. Available: <https://www.uv.es/uvweb/estructura-investigacion-interdisciplinar-lectura/es/productos-tecnologicos/productos-tecnologicos/readit-vms-1286067296453.html>.
- [25] A. Erke y F. H. R. Sagberg, «Effects of route guidance variable message signs (VMS) on driver behaviour,» *Transportation Research Part F: Traffic Psychology and Behaviour.*, vol. 10, pp. 447-457, 2007.

- [26] Dirección General de Tráfico - Ministerio del Interior, «Cuestiones De Seguridad Vial,» 2018. [En línea]. Available: <http://www.dgt.es/Galerias/seguridad-vial/formacion-vial/cursos-para-profesores-y-directores-de-autoescuelas/XXI-Cuso-Profesores/Manual-II-Cuestiones-de-Seguridad-Vial-2018.pdf>.
- [27] V. K. Kukkala, J. Tunnell, S. Pasricha y T. Bradley, «Advanced Driver-Assistance Systems: A Path Toward Autonomous Vehicles,» IEEE Consumer Electronics Magazine, 2018, pp. 18-25.
- [28] O. Bay, «ABI Research Forecasts 8 Million Vehicles to Ship with SAE Level 3, 4 and 5 Autonomous Technology in 2025,» ABI Research, 2018. [En línea]. Available: <https://www.abiresearch.com/press/abi-research-forecasts-8-million-vehicles-ship-sae-level-3-4-and-5-autonomous-technology-2025/>. [Último acceso: 2020].
- [29] S. O'Kane, «How Tesla and Waymo are tackling a major problem for self-driving cars: data,» The Verge, 2018. [En línea]. Available: <https://www.theverge.com/transportation/2018/4/19/17204044/tesla-waymo-self-driving-car-data-simulation>.
- [30] Synopsys, «The 6 Levels of Vehicle Autonomy Explained,» [En línea]. Available: <https://www.synopsys.com/automotive/autonomous-driving-levels.html>. [Último acceso: 2020].
- [31] NHTSA, «Automated Vehicles for Safety,» [En línea]. Available: <https://www.nhtsa.gov/technology-innovation/automated-vehicles-safety>. [Último acceso: 2020].

- [32] C. Badue, R. Guidolini, R. Vivacqua Carneiro, P. Azevedo, V. Brito Cardoso, A. Forechi, L. Jesus, R. Berriel, T. Paixão, F. Mutz, L. Veronese, T. Oliveira-Santos y A. Ferreira De Souza, «Self-Driving Cars: A Survey,» *arXiv:1901.04407v2* , 2019.
- [33] Z. Zou, Z. Shi, Y. Guo y J. Ye, «Object Detection in 20 Years: A Survey,» *arXiv:1905.05055v2*, 2019.
- [34] P. Viola y M. J. Jones, «Robust real-time face detection,» *International journal of computer vision*, vol. 57, nº 2, p. 137–154, 2004.
- [35] P. Viola y M. J. Jones, «Rapid object detection using a boosted cascade of simple feature,» *Proceedings of the 2001 IEEE Computer Society Conference*, vol. 1, 2001.
- [36] N. Dalal y B. Triggs, «Histograms of Oriented Gradients for Human Detection,» *IEEE Computer Society Conference*, vol. 1, p. 886–893, 2005.
- [37] P. Felzenszwalb, D. McAllester y D. Ramanan, «A Discriminatively Trained, Multiscale, Deformable Part Model,» *Computer Vision and Pattern Recognition*, pp. 1-8, 2008.
- [38] P. Soviany y R. T. Ionescu, «Optimizing the Trade-off between Single-Stage and Two-Stage Object Detectors using Image Difficulty Prediction,» *arXiv:1803.08707v3*, 2018.

- [39] R. Girshick, J. Donahue, T. Darrell y J. Malik, «Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation,» *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2013.
- [40] R. Girshick, «Fast R-CNN,» *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1440-1448, 2015.
- [41] S. Ren, K. He, R. Girshick y J. Sun, «Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,» *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, nº 6, pp. 1137-1149, 2017.
- [42] J. Redmon, S. Divvala, R. Girshick y A. Farhadi, «You only look once: Unified, real-time object detection,» *Proceedings of the IEEE conference on computer vision and pattern recognition*, p. 779–788, 2016.
- [43] J. Redmon y A. Farhadi, «YOLO9000: Better, Faster, Stronger,» *arXiv:1612.08242v1*, 2017.
- [44] J. Redmon y A. Farhadi, «YOLOv3: An Incremental Improvement,» *arXiv:1804.02767*, 2018.
- [45] A. Bochkovskiy, C.-Y. Wang y H.-Y. Mark Liao, «YOLOv4: Optimal Speed and Accuracy of Object Detection,» 2020.
- [46] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár y C. Zitnick, «Microsoft COCO: Common Objects in Context,» *arXiv:1405.0312v3*, 2014.

- [47] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu y A. C. Berg, «Ssd: Single shot multibox detector,» *European conference on computer vision*, p. 21–37, 2016.
- [48] T.-Y. Lin, P. Goyal, R. Girshick, K. He y P. Dollár, «Focal Loss for Dense Object Detection,» *IEEE transactions on pattern analysis and machine intelligence*, vol. 42, n° 2, pp. 318-327, 2018.
- [49] H. Lin, P. Yang y F. Zhang, «Review of Scene Text Detection and Recognition,» *Archives of Computational Methods in Engineering*, vol. 27, p. 433–454, 2020.
- [50] S. Long, X. He y C. Yao, «Scene Text Detection and Recognition: The Deep Learning Era,» *arXiv:1811.04256*, 2018.
- [51] Y. Zhu, C. Yao y X. Bai, «Scene text detection and recognition: recent advances and future trends,» *Frontiers of Computer Science*, vol. 10, 2015.
- [52] A. Graves, S. Fernández, F. Gomez y J. Schmidhuber, «Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks,» *ICML 2006 - Proceedings of the 23rd International Conference on Machine Learning*, vol. 2006, pp. 369-376, 2006.
- [53] R. Charette y F. Nashashibi, «Real Time Visual Traffic Lights Recognition Based on Spot Light Detection and Adaptive Traffic Lights Templates,» *IEEE Intelligent Vehicles Symposium, Proceedings*, pp. 358-363, 2009.

- [54] N. Fairfield y C. Urmson, «Traffic Light Mapping and Detection,» *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 5421 - 5426, 2011.
- [55] Y.-C. Chung, J.-M. Wang y S.-W. Chen, «A vision-based traffic light detection system at intersections,» *Journal of Taiwan Normal University: Mathematics, Science and Technology*, vol. 47, pp. 67-86, 2002.
- [56] Y. Lu, J.-M. Lu, S. Zhang y P. Hall, «Traffic signal detection and classification in street views using an attention model,» *Computational Visual Media*, vol. 4, n° 3, p. 253–266, 2018.
- [57] Z. Zhu, D. Liang, S. Zhang, X. Huang, B. Li y S. Hu, «Traffic-Sign Detection and Classification in the Wild,» *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2110-2118, 2016.
- [58] Toyota Motor Sales, «Toyota Safety Sense™ comes standard on many new Toyotas,» [En línea]. Available: <https://www.toyota.com/safety-sense/animation/drcc>. [Último acceso: 28 07 2020].
- [59] Á. González, L. M. Bergasa y J. J. Yebes, «Text Detection and Recognition on Traffic Panels From Street-Level Imagery Using Visual Appearance,» *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, n° 1, pp. 228-238, 2014.
- [60] A. Vazquez-Reina, R. Sastre, S. Arroyo y P. Gil-Jiménez, «Adaptive traffic road sign panels text extraction,» *Proceedings of the 5th WSEAS Int. Conf. on Signal Processing, Robotics and Automation*, pp. 295-300, 2006.

- [61] Á. González, L. M. Bergasa, M. Gavilan, M. A. Sotelo, F. Herranz y C. Fernandez, «Automatic information extraction of traffic panels based on computer vision,» *2009 12th International IEEE Conference on Intelligent Transportation Systems*, pp. 1-6, 2009.
- [62] Á. González, L. M. Bergasa, J. J. Yebes y J. Almazán, «Text recognition on traffic panels from street-level imagery,» *2012 IEEE Intelligent Vehicles Symposium*, pp. 340-345, 2012.
- [63] Tzutalin, «LabelImg,» GitHub, [En línea]. Available: <https://github.com/tzutalin/labelImg>. [Último acceso: 8 7 2020].
- [64] K. He, X. Zhang, S. Ren y J. Sun, «Deep Residual Learning for Image Recognition,» *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, nº 770-778, 06 2016.
- [65] H. Al-Behadili, K. Ku-Mahamud y R. Sagban, «Rule pruning techniques in the ant-miner classification algorithm and its variants: A review,» *2018 IEEE Symposium on Computer Applications Industrial Electronics (ISCAIE)*, pp. 78-84, 2018.
- [66] Fizyr, «keras-retinanet,» GitHub, [En línea]. Available: <https://github.com/fizyr/keras-retinanet>. [Último acceso: 12 07 2020].
- [67] Google, «TensorFlow,» [En línea]. Available: <https://www.tensorflow.org/tutorials/quickstart/beginner>. [Último acceso: 12 07 2020].

- [68] J. Canny, «A Computational Approach to Edge Detection,» *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, nº 6, pp. 679-698, 1986.
- [69] R. C. Gonzalez y R. E. Woods, Digital Image Processing, Pearson.
- [70] J. Valverde-Rebaza, «Detección de bordes mediante el algoritmo de Canny,» 2007.
- [71] OpenCV. Open Source Computer Vision, «Canny Edge Detector,» [En línea]. Available:
https://docs.opencv.org/master/da/d5c/tutorial_canny_detector.html. [Último acceso: 21 06 2020].
- [72] G. Pajares y J. M. de la Cruz, Visión por Computador. Imágenes digitales y aplicaciones., RaMa.
- [73] J. Irving Vásquez, Visión Computacional. Edges, Consejo Nacional de Ciencia y Tecnología, 13 de febrero de 2020.
- [74] OpenCV. Open Source Computer Vision, «Hough Line Transform,» [En línea]. Available: https://docs.opencv.org/master/d9/db0/tutorial_hough_lines.html. [Último acceso: 21 06 2020].
- [75] A. Shehata, S. Mohammad, M. Abdallah y M. Ragab, «A Survey on Hough Transform, Theory, Techniques and Applications,» 2015.
- [76] P. V. C. Hough, «Methods and Means for Recognizing Complex Patterns». USA Patente 3,069,654, 1962.

- [77] D. H. Ballard y C. M. Brown, Computer Vision, Englewood Cliffs, NJ: Prentice Hal, 1982.
- [78] A. Rosebrock, «Zero-parameter, automatic Canny edge detection with Python and OpenCV,» 06 04 2015. [En línea]. Available:
<https://www.pyimagesearch.com/2015/04/06/zero-parameter-automatic-canny-edge-detection-with-python-and-opencv/>. [Último acceso: 21 06 2020].
- [79] OpenCV. Open Source Computer Vision, «Color conversions,» [En línea]. Available:
https://docs.opencv.org/3.4/de/d25/imgproc_color_conversions.html. [Último acceso: 2020 06 28].
- [80] N. Otsu, «Fronts Propagating with Curvature-Dependent Speed: Algorithms Based on Hamilton-Jacobi Formulations,» *IEEE Trans. Systems, Man, and Cybernetics*, vol. 9, nº 1, p. 62–66, 1979.
- [81] OpenCV. Open Source Computer Vision, «Image Thresholding,» [En línea]. Available:
https://docs.opencv.org/master/d7/d4d/tutorial_py_thresholding.html. [Último acceso: 28 06 2020].
- [82] B. S. Morse, «Lecture 4: Thresholding (Brigham Young University),» [En línea]. Available:
http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/MORSE/threshold.pdf. [Último acceso: 28 6 2020].

- [83] OpenCV. Open Source Computer Vision, «Morphological Transformations,» [En línea]. Available:
https://docs.opencv.org/trunk/d9/d61/tutorial_py_morphological_ops.html. [Último acceso: 28 06 2020].
- [84] K. Sreedhar y B. Panlal, «Enhancement of Images Using Morphological Transformations,» *International Journal of Computer Science and Information Technology*, vol. 4, nº 1, p. 33–50, 2012.
- [85] OpenCV. Open Source Computer Vision, «Histogram Equalization,» [En línea]. Available:
https://docs.opencv.org/3.4/d4/d1b/tutorial_histogram_equalization.html. [Último acceso: 30 06 2020].
- [86] R. Smith, «An Overview of the Tesseract OCR Engine,» *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, vol. 2, pp. 629-633, 2007.
- [87] IBM, «Watson Speech to Text,» [En línea]. Available:
<https://www.ibm.com/es-es/cloud/watson-speech-to-text>. [Último acceso: 11 7 2020].
- [88] tesseract-ocr, «Tesseract,» GitHub, [En línea]. Available:
<http://code.google.com/p/tesseract-ocr>. [Último acceso: 28 07 2020].

- [89] A. Kumar, «Performing OCR by running parallel instances of Tesseract 4.0 : Python,» 30 06 2018. [En línea]. Available: <https://appliedmachinelearning.blog/2018/06/30/performing-ocr-by-running-parallel-instances-of-tesseract-4-0-python/>. [Último acceso: 28 07 2020].