

UNIVERSIDAD POLITÉCNICA DE MADRID

**ESCUELA TÉCNICA SUPERIOR
DE INGENIEROS DE TELECOMUNICACIÓN**



**GRADO EN INGENIERÍA DE TECNOLOGÍAS Y
SERVICIOS DE TELECOMUNICACIÓN**

TRABAJO FIN DE GRADO

**DISEÑO E IMPLEMENTACIÓN DE UN ROBOT
MÓVIL GUIADO MEDIANTE VISIÓN ARTIFICIAL**

MIGUEL GARCÍA MAYA

2019

UNIVERSIDAD POLITÉCNICA DE MADRID

**ESCUELA TÉCNICA SUPERIOR
DE INGENIEROS DE TELECOMUNICACIÓN**



**GRADO EN INGENIERÍA DE TECNOLOGÍAS Y
SERVICIOS DE TELECOMUNICACIÓN**

TRABAJO FIN DE GRADO

**DISEÑO E IMPLEMENTACIÓN DE UN ROBOT
MÓVIL GUIADO MEDIANTE VISIÓN ARTIFICIAL**

MIGUEL GARCÍA MAYA

2019

TUTOR: ÁLVARO GUTIÉRREZ MARTÍN

Resumen

En los últimos años se han desarrollado una gran variedad de tecnologías de conducción asistida y autónoma. Muchas de las que están presentes en el mercado, hacen uso de cámaras y sensores que monitorizan el entorno.

En el siguiente Trabajo Fin de Grado (TFG), se abordará el diseño de un sistema navegación de un vehículo a escala autoconducido, que sea capaz de seguir un camino predefinido y detectar obstáculos mediante el uso de sensores. El documento contiene información sobre la implementación hardware y software del prototipo basado en la tecnología existente y en las limitaciones materiales. También se describirán los escenarios en los que se han realizado las pruebas del prototipo, los resultados obtenidos, y los problemas encontrados.

Abstract

In recent years, a wide variety of autonomous driving technologies have been developed. Many of whom have been launched on the market make use of cameras and sensors that scan the environment. The following thesis will approach the design of an scaled self driving vehicle, that will be able to follow a designed path and will also perform obstacle detection making use of sensors. The document has information about the hardware and software of the prototype based on the existing technology and the material constraints . Different scenarios will be described in which the prototype has been tested, and lastly the results, and problems that have been faced will be discussed.

Agradecimientos

A mis padres, por estar a mi lado cuando más lo necesitaba, por enseñarme el valor del esfuerzo, y por no haber dejado de creer en mí.

Contents

Resumen	iii
Resumen	iv
Agradecimientos	v
Índice General	vi
Índice de Figuras	ix
Índice de Tablas	x
Lista de acrónimos	xii
1 Introducción y Objetivos	1
1.1 Introducción	1
1.2 Objetivos	1
1.3 Estado del arte	2
1.3.1 Antecedentes	2
1.3.2 Situación actual	3
1.3.3 Tecnología utilizada	4
1.3.4 Inteligencia Artificial	8
1.3.5 Visión artificial	8
1.4 Organización del documento	9
2 Procesado de Imagen	11
2.1 Visión artificial	11
2.1.1 Algoritmos empleados para la detección de líneas	11
2.1.1.1 Obtención del área de interés	11
2.1.2 Detección y clasificación de señales de tráfico	16
2.1.2.1 Localización de las señales de tráfico	16
2.1.2.2 Clasificación de la señal	17
3 Implementación	21
3.1 Implementación del prototipo	21
3.1.1 Descripción general del sistema	21
3.1.2 Raspberry Pi 3	21
3.1.2.1 Sistema Operativo	22
3.1.3 Camera Module V2	22

3.1.4	Servomotor	23
3.1.5	PCA9685	24
3.1.6	Motor DC	24
3.1.7	Regulador de tensión	25
3.1.8	Sensor de ultrasonidos HC-SR04	26
3.1.9	Controlador de motor L298N	26
3.1.10	Sistema de alimentación	27
3.1.11	Pista de pruebas	27
3.2	Sistema de control. Guiado del vehículo	27
3.2.1	Distancia de la referencia	28
3.2.2	Diseño de un controlador P	28
3.3	Sistema de detección obstáculos	29
4	Experimentos	31
4.1	Determinación del error de la posición calculada con respecto a los extremos de la pista	31
4.1.1	Resultados experimentales utilizando la transformada de Hough	32
4.1.2	Resultados experimentales utilizando el método del alumno . .	33
4.2	Cálculo del error de posición del coche en movimiento en función de la distancia de la referencia de la pista	34
5	Conclusiones y líneas futuras	39
5.1	Conclusiones	39
5.2	Líneas futuras	39
	Bibliografía	42
	Anexos	45
A	Aspectos éticos, económicos, sociales y ambientales	47
B	Presupuesto económico	49

List of Figures

1.1	Simulación por ordenador del sistema TSR.	5
1.2	Componentes más importantes de un coche autónomo. (Joseph, 2017) . . .	6
1.3	Diagrama de bloques del software de un vehículo autónomo.	7
2.1	Diagrama de flujo para detección de líneas.	12
2.2	A la izquierda frame original, a la derecha región de interés.	12
2.3	Representación de la transformada de Hough.	14
2.4	Conjunto de rectas que pasan por un determinado punto.	15
2.5	Carril detectado.	16
2.6	Señales de tráfico usadas.	16
2.7	Parámetros de una circunferencia.	17
2.8	Imagen con diferentes colores.	18
2.9	Imagen con diferentes colores.	18
2.10	Regiones evaluadas para la clasificación del tipo de señal.	19
2.11	Resultados del algoritmo de detección y clasificación de señales de tráfico. . .	19
3.1	Raspberry Pi.	22
3.2	RaspiCam.	23
3.3	Servo Futaba-S3001(izda). Partes internas del servomotor(dcha).	23
3.4	Cicuito integrado PCA9685.	24
3.5	Encoder del motor.	25
3.6	Esquemático del circuito regulador de tensión.	25
3.7	Funcionamiento del módulo HC-SR04.	26
3.8	Módulo L298N.	27
3.9	Rangos de medida del sensor ultrasónico.	29
4.1	Posición calculada de la cámara frente a la posición real empleando la transformada de Hough.	32
4.2	Error de la posición calculada del vehículo comparada con la posición real. .	32
4.3	Posición calculada de la cámara frente a la posición real empleando el método del alumno.	33
4.4	Error de la posición calculada del vehículo comparada con la posición real. .	33
4.5	Cámara con una inclinación de 10°	34
4.6	Error normalizado para una distancia de 28 cm.	35
4.7	Error normalizado para una distancia de 23.5 cm.	36
4.8	Error normalizado para una distancia de 21.5 cm.	36
4.9	Comparativa de errores en función de la distancia.	37

Índice de tablas

3.1	Tabla de características del motor.	25
4.1	Equivalencia ángulo de inclinación de la cámara - Distancia	35

Lista de Acrónimos

PWM: Pulse Width Modulation.

GPIO: General Purpose Input/Output.

DC: Direct current.

UART: Universal Asynchronous Receiver-Transmitter.

Chapter 1

Introducción y Objetivos

1.1 Introducción

Los avances de los últimos años en capacidad de computación y electrónica están abriendo nuevos campos de investigación. Uno de los campos que se ha visto afectado por estas mejoras es el de la automoción. Cada año nuevos sistemas de seguridad electrónicos se incorporan a los vehículos, especialmente los que incluyen la detección del entorno del coche, como por ejemplo frenada de emergencia, aviso de salida del carril, o el asistente de velocidad inteligente. En esta línea de investigación, la conducción autónoma ha sido uno de los campos con un mayor desarrollo, sin embargo aunque a día de hoy ya se han lanzado algunos modelos comerciales, la gran complejidad técnica de obtener información de un entorno, muchas veces en condiciones adversas, y sobre todo el hecho de que haya vidas humanas dependientes de las decisiones que pueda tomar el coche de forma autónoma, hacen que la conducción autónoma plantee grandes exigencias de calidad.

1.2 Objetivos

El objetivo de este trabajo es el desarrollo de un prototipo de un vehículo autónomo, basado en un robot móvil guiado mediante visión artificial. Este robot deberá cumplir dos requisitos: el primero es que el prototipo tenga un coste asequible, por lo que los materiales deben poderse adquirir fácilmente en el mercado, el segundo requisito es que el prototipo se mueva en un entorno, que aunque simplificado, sea semejante a uno de conducción real, y que permita desarrollar unas técnicas de control similares a las que se desarrollan en la realidad. Teniendo en cuenta estas exigencias, se ha dispuesto que el prototipo esté basado en el ordenador de placa reducida raspberry pi, que equipada con una cámara, y haciendo uso de la librería de procesamiento de imágenes OpenCV, va a ser capaz de desplazarse por un circuito, que detectará mediante visión artificial.

Los puntos propuestos para poder cumplir este objetivo son los siguientes:

- Definición y diseño de la arquitectura del sistema para controlar todo el vehículo. Con especial atención al diseño eléctrico, a la hora de construir un sistema capaz de alimentar de energía a todo el prototipo y el uso de reguladores y controlador para ajustarnos a las exigencias de cada componente.

- Desarrollo de algoritmos de visión artificial. Ha sido necesario el aprendizaje de técnicas de procesamiento de imágenes en python, especialmente mediante la librería OpenCV, y la librería numpy, empleada para operaciones de cálculo con vectores.
- Desarrollo de un sistema de control de guiado del coche. Primero ha sido necesario volver a repasar los conceptos, que se refieren a los controladores proporcional, integral y derivativo. El diseño se ha realizado utilizando métodos empíricos de diseño de controladores.
- Diseño de pruebas y obtención de resultados, acotando correctamente todas las variables que han intervenido en el experimento.

1.3 Estado del arte

1.3.1 Antecedentes

El primer vehículo sin conductor fue desarrollado en el año 1926 por la compañía Achen Motor (Bimbrow, 2015). Aunque no podemos hablar de un vehículo totalmente autónomo, ya que este era controlado desde otro que lo seguía y le enviaba órdenes a través de un sistema de radio, leídas mediante un receptor instalado en el primer coche. Este prototipo no tuvo un mayor desarrollo, pero unos años más tarde, en la exposición Futurama de 1939, el diseñador industrial Norman Bel Geddes, con el patrocinio de General Motors, presentó un vehículo eléctrico autoconducido, que era controlado por un circuito eléctrico embebido en el pavimento de la carretera.

El primer hito significativo en la conducción autónoma vino de la mano del ingeniero alemán Ernst Dickmanns, quien a mediados de los años 80 y utilizando como base una furgoneta Mercedes-Benz, creó el primer vehículo completamente autónomo, capaz de moverse, hacer cambios y maniobras de frenado, haciendo uso de una sensores y cámaras (Dickmanns, 1997). Dado que la capacidad de computación era mucho menor que la actual, gran parte de la investigación se centró en desarrollar un sistema de visión artificial más eficiente. Para ello, se imitó el mecanismo que tiene un ojo humano de percibir su entorno, dotando a las plataformas, que cargaban las cámaras de la capacidad de realizar movimientos sacádicos para centrarse, en los puntos más importantes de la pista (Dickmanns, 1997). También se utilizaron algoritmos de filtrado de Kalman para eliminar el ruido en condiciones adversas. En 1987, esta furgoneta, realizó unas pruebas en Autobahn, la red de autopistas alemanas, llegando a alcanzar una velocidad de 90 km/h (Dickmanns, 1997).

Ese mismo año arrancó el proyecto Eureka PROMETHEUS (PROgramme for a European Traffic of Highest Efficiency and Unprecedented Safety), un proyecto de investigación a nivel Europeo, en el que participaron numerosas universidades y fabricantes de automóviles. Este proyecto contó con la participación de Dickmanns, y se realizaron grandes avances durante los ocho años de duración del proyecto. Los coches desarrollados por el equipo de Dickmanns, fueron probados en un entorno real de conducción en 1992. Se emplearon un tipo especial de computadores en paralelo, para cubrir las grandes demandas de capacidad de computación.

Los dos mayores hitos tuvieron lugar en 1994 y 1995 cuando el equipo de Dickmanns

automatizó un Mercedes-Benz Clase-S , en el cual se realizaron pruebas internacionales. El vehículo condujo de manera autónoma durante 1000 km, en una autopista de tres carriles a las afueras de París, bajo condiciones de tráfico denso, con una velocidad máxima de 130 km/h. El segundo hito fue un viaje de 1758 kilómetros desde Munich, en Alemania hasta Odense, en Dinamarca. Durante este trayecto se llegaron a velocidades de 175 km/h, en la autopista Autobahn (Dickmanns, 1997).

En el año 2004, con el fin de incentivar la investigación en el campo de la tecnología de conducción autónoma, la agencia del Departamento de Defensa de Estados Unidos responsable del desarrollo de nuevas tecnologías para uso militar, más conocida como DARPA, creó la competición de vehículos autoconducidos Grand DARPA (Buehler M, 2009). En esta competición los vehículos deben llegar desde un punto de los Estados Unidos hasta otro sin intervención humana y disponiendo únicamente de un listado de puntos intermedios entre el principio del circuito y el final. Las competiciones de DARPA, dieron una mayor visibilidad a esta tecnología y la industria del automóvil, empezó a invertir en los vehículos autónomos (Buehler M, 2009).

En el año 2009 Google comenzó a desarrollar su propio vehículo autónomo conocido como Waymo, que desde el año 2017 ya es capaz de circular por entornos urbanos (Davies, 2017).

1.3.2 Situación actual

En el año 2013, la NHTSA (National Highway Safety Administration) creó una clasificación de cinco niveles(NHTSA, 2013), en función de las capacidades de autoconducción del vehículo, como una base para que la guía para que la Administración legislara la correspondiente normativa. El nivel 0 correspondía a ninguna automatización, mientras que el vehículo con clasificación 4 podría realizar una conducción totalmente autónoma.

En 2014, la sociedad de ingenieros de automoción (SAE: Society of Automotive Engineers), realizó una nueva clasificación con 6 niveles (SAE, 2014), esta clasificación es la que se emplea globalmente a día de hoy, siendo adoptada por la propia NHTSA. La última revisión de dicho informe se realizó en septiembre de 2016(SAE, 2016).

- **Nivel 0:** Ninguna automatización. El conductor debe realizar todas las maniobras durante la conducción.
- **Nivel 1:** Conducción asistida. El automóvil puede tomar el control del volante o de los pedales, pero no de los dos a la vez. Dentro de esta categoría estas los coches con controles de velocidad adaptativos y asistentes de mantenimiento de carril.
- **Nivel 2:** El coche puede controlar tanto los pedales como la dirección, en determinadas situaciones. Pero la responsabilidad de la conducción recae completamente en el conductor, que debe permanecer atento a su entorno. Los modelos capaces de aparcar de forma completamente autónoma también se encuadran aquí.
- **Nivel 3:** Supone un salto cualitativo, ya que el coche es capaz de monitorizar el entorno 360 grados, y realizar maniobras tales como cambiar de carril, y resolver

situaciones que el conductor no sea capaz de resolver por el corto espacio de tiempo en el que se producen.

- **Nivel 4:** Automatización alta. Los automóviles serán capaces de hacer prescindible al conductor casi en su totalidad, y la tecnología del vehículo monitorizará constantemente el entorno. El conductor simplemente deberá indicar el lugar de destino, y podrá retomar el control del coche cuando lo desee. A día de hoy no existen en el mercado vehículos con esta categoría.
- **Nivel 5:** Automatización plena. El conductor se desentiende totalmente de la conducción y el vehículo no tiene volantes ni pedales. Lograr la conectividad vehículo a vehículo (V2V) y vehículo a infraestructura (V2I) vendrá condicionado por la implantación del sistema de comunicación 5G.

La legislación en esta materia está siendo renovada constantemente para adaptarse a los diferentes escenarios que se dan en esta industria. Estados Unidos ha sido el país que más pronto ha comenzado a legislar, ya hay 30 estados que han aprobado leyes o emitido órdenes que regulan las licencias, los seguros, las pruebas y/o el desarrollo de vehículos autónomos. Los estados de California y Arizona aprobaron en 2018 las pruebas en vía pública sin ningún ocupante dentro del vehículo.

En Europa, la legislación depende de cada país, ya que todavía no existe un marco común de la Unión Europea. Según un estudio de KPMG, Holanda sería el país más preparado del mundo para la implantación de esta tecnología (International, 2018). Su consejo de ministros aprobó en 2015 las primeras pruebas de esta tecnología en la vía pública, y ya está en marcha el primer autobús autónomo del mundo, además se está trabajando para adaptar para los coches autónomos más de 1.000 señales luminosas.

En España en el 2015 se publicó un marco de actuación para regular las pruebas en vías públicas. El plan de innovación lanzado por el Gobierno en 2017 incluye algunos pasos importantes para desarrollar el ecosistema requerido y promover la conducción autónoma en España.. Este plan incluye, por ejemplo, la planificación estratégica de las redes 5G en carreteras y ferrocarriles, y la estandarización técnica, desarrollo del marco legal y desarrollo de un libro blanco sobre la ética del vehículo autónomo, todo ello planeado para 2020.

Es por todo ello que la conducción autónoma ya es una realidad, y aunque sus aplicaciones en la sociedad actual sean reducidas, se espera una implantación gradual durante los próximos 15 años. (Litman, 2017)

1.3.3 Tecnología utilizada

En los últimos años han salido al mercado diferentes de sistemas de conducción asistida, la mayoría de los vehículos que salen al mercado actualmente cuentan con alguno de estos sistemas incorporados, no solo por razones de comodidad sino también por razones de seguridad. Algunos de los sistemas más populares son:

Los sistemas LDW (Lane Departure Warning) y el LKS (Lane Keeping System) son sistemas que detectan los carriles de la calzada y emiten una alarma cuando el sistema detecta el vehículo está fuera de control, en función del giro del volante o del uso de intermitentes. En el caso del LKW, el sistema también es capaz de corregir la

trayectoria.

El sistema TSR (Traffic Sign Recognition) detecta las señales de tráfico empleando una cámara de alta resolución, situada generalmente a la altura del espejo retrovisor interior, e indica en la pantalla del coche la señalización de la vía en tiempo real. El

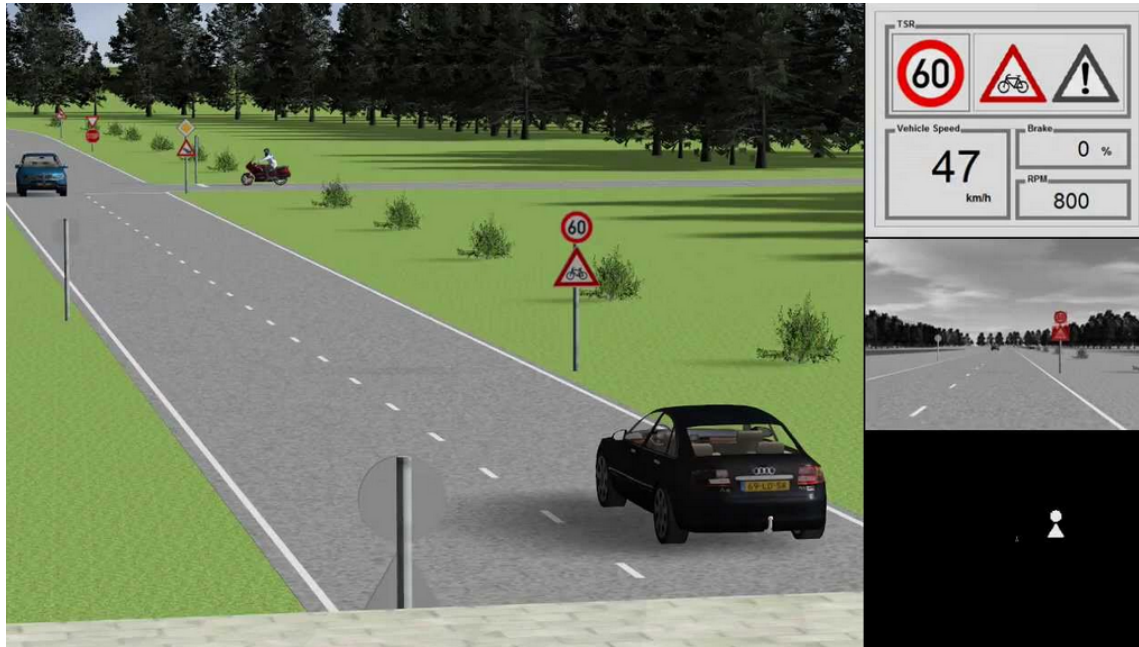


Figure 1.1: Simulación por ordenador del sistema TSR.

monitor de punto ciego, BLIS (Blind spot monitor), avisa de la presencia de vehículos en el ángulo muerto del vehículo. Algunos lo hacen de forma continua, mientras que otros se limitan a hacerlo sólo en el momento en el que el conductor activa el intermitente para efectuar un cambio de carril.

Con el objetivo de evitar muertes en la carretera, el Parlamento Europeo ha aprobado una directiva que obliga a instalar un paquete de diez medidas de seguridad en los modelos que se homologuen a partir de mayo de 2022. El más destacable ha sido el Asistente de Velocidad Inteligente (ISA), que se encargará de regular automáticamente la velocidad del coche. Lo hará registrando constantemente los límites de la vía y ajustando la marcha en consecuencia. A continuación se presenta una lista con los componentes más importantes de un vehículo autónomo (Joseph, 2017).

- LIDAR(Laser Imaging Detection and Ranging): Utilizando un haz láser pulsado genera en tiempo real una imagen tridimensional de nube de millones de puntos alrededor del vehículo, conociendo además para cada uno de ellos la distancia precisa hasta el foco emisor del LIDAR. Fabricantes como Google o Ford confían en este sistema como método principal para detectar los obstáculos. Este sistema se complementa con otros sensores y sobretodo con cámaras de vídeo que detectan todo lo que el LIDAR no puede ver, como las señales de tráfico o indicaciones en la calzada.Sin embargo este dispositivo tiene dos grandes inconvenientes, su alto precio (varía entre los 33.000 y los 70.000 €) y su difícil

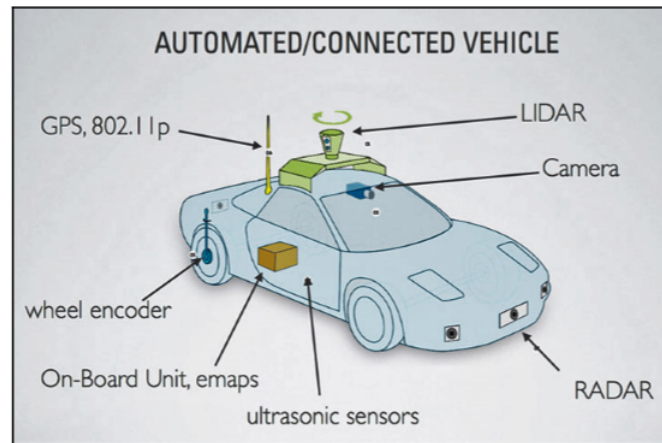


Figure 1.2: Componentes más importantes de un coche autónomo. (Joseph, 2017)

integración en el diseño del coche. Es por esos que otras marcas como Volvo o Tesla han sustituido el LIDAR por cámaras de video de alta definición repartidas por todo el vehículo, ya que con un software potente, se pueden llegar a medir las distancias a los objetos con la misma precisión que un LIDAR.

- Radar: El radar se utiliza con el mismo propósito que el LIDAR, la diferencia fundamental entre ambos es que el radar emplea ondas de radio. Tiene la ventaja de ser más ligero y ser capaz de operar en condiciones más adversas que el LIDAR (niebla, lluvia...), tiene un mayor alcance y es más barato. Sin embargo tiene un peor comportamiento en distancias cortas, siendo incapaz de distinguir dos objetos si estos están muy próximos.
- Sistema de Posicionamiento Global(GPS): Determina la posición del coche a través de la triangulación de posiciones proporcionadas por los satélites GPS. Generalmente se usa junto con la información proporcionada por la IMU (Unidad de Medición Inercial), y el encoder rotacional de las ruedas para estimar la posición del coche .Por ejemplo en el caso de un túnel, en el que se pierde la señal GPS, estos sensores aportan información de la dirección y velocidad.
- Encoder de las ruedas: Proporciona el número de vueltas de la rueda por segundo. Haciendo uso de la odometría se puede obtener información sobre la velocidad del coche y su posición.
- Sensores de ultrasonido: Estos sensores se utilizan para realizar las maniobras de aparcamiento, y para esquivar y detectar objetos en ángulos muertos. Tienen un alcance aproximado de unos 10 metros.
- Unidad de Medición Inercial: Se basa en un giroscopio y un acelerómetro y permite controlar la orientación en un espacio tridimensional y la velocidad y dirección de desplazamiento. Estos dos sistemas se complementan entre ellos.
- Ordenador de abordo: Es el núcleo del coche autónomo. Suele estar equipado con procesadores de alta gama como los Intel Xenon, y unidades de procesamiento gráfico o GPU (graphic processor unit), para el procesamiento

de todos los datos de los sensores. Todos los sensores están conectados a este dispositivo que predice la trayectoria y envía comandos de control del coche autónomo, como son el ángulo de giro, la aceleración o la frenada del coche autónomo.

A continuación se muestra la organización por bloques que conforma el software de un vehículo autónomo (Joseph, 2017).

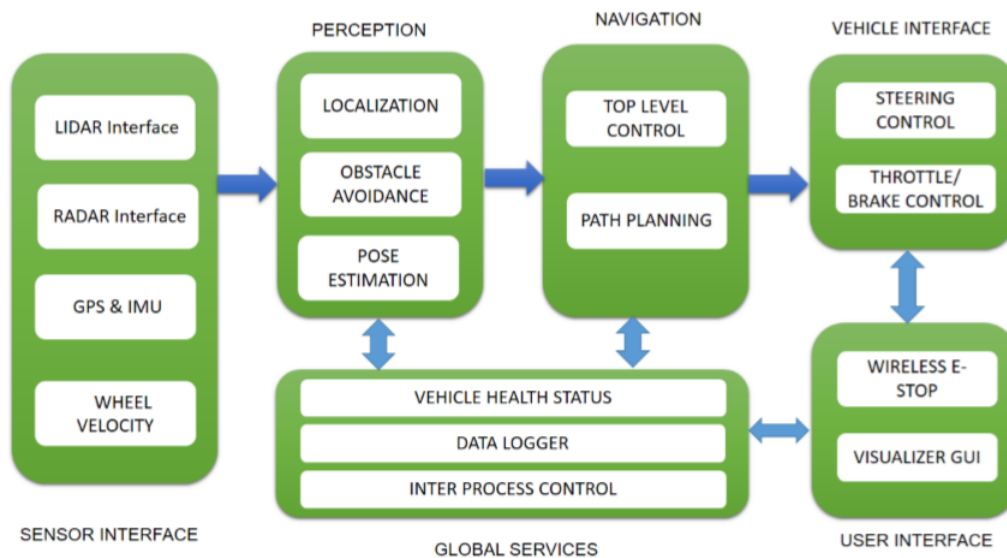


Figure 1.3: Diagrama de bloques del software de un vehículo autónomo.

- Interfaz de los sensores del coche: En este bloque se realiza toda la comunicación entre los sensores y el resto del coche. Todos los datos recibidos por los sensores convergen en este bloque.
- Módulo de percepción: En este bloque se procesan los datos de los sensores, para obtener información sobre los obstáculos en la vía o la posición del coche.
- Módulo de navegación: Determinan el control del coche. Haciendo uso del módulo de percepción, calculan la ruta más óptima para llegar de un punto A a un punto B.
- Interfaz del vehículo. Ejecuta los comandos de control del coche, después de haber sido calculada la ruta por el módulo de navegación.
- Interfaz de usuario. Permite al pasajero elegir el destino en el mapa, y generalmente se dispone también de un botón de parada de emergencia.
- Servicios globales: Se encarga de registrar los datos, y hace uso del sellado de tiempo, y del paso de mensaje para garantizar que el software funciona adecuadamente.

1.3.4 Inteligencia Artificial

La inteligencia artificial (IA), es la inteligencia llevada a cabo por máquinas. En ciencias de la computación, una máquina inteligente ideal es un agente flexible que percibe su entorno y lleva a cabo acciones que maximicen sus posibilidades de éxito en algún objetivo o tarea. Generalmente este término se utiliza para referirse a las máquinas que son capaces de imitar las funciones cognitivas que realizan otros seres humanos, como percibir, aprender razonar y resolver problemas.

Según la clasificación establecida por los informáticos Stuart Russell y Peter Norvig, (Orti et al., 1995), existen cuatro tipos de inteligencia artificial:

- Sistemas que piensan como humanos: Estos sistemas tratan de emular el pensamiento humano; por ejemplo las redes neuronales artificiales. La automatización de actividades que vinculamos con procesos de pensamiento humano, actividades como la Toma de decisiones, Resolución de problemas y aprendizaje.
- Sistemas que actúan como humanos: Estos sistemas tratan de actuar como humanos, es decir, imitan el comportamiento humano, como en el caso de la robótica. El objetivo último de estos sistemas es eliminar el error humano y el tiempo de ejecución de las tareas.
- Sistemas que piensan racionalmente: Es decir, con lógica (idealmente), tratan de imitar o emular el pensamiento lógico racional del ser humano; por ejemplo los sistemas expertos. El estudio de los cálculos que hacen posible percibir, razonar y actuar.
- Sistemas que actúan racionalmente : Tratan de emular de forma racional el comportamiento humano; por ejemplo los agentes inteligentes. Está relacionado con conductas inteligentes en artefactos.

Este trabajo estaría dentro de la segunda categoría, ya que pretende emular el comportamiento de un conductor, prescindiendo de cualquier tipo de intervención humana.

1.3.5 Visión artificial

La visión artificial es una disciplina científica cuyo objetivo es adquirir, procesar y analizar imágenes del mundo real a imitación del sentido de la visión humana. Esto se consigue gracias a distintos campos como la geometría, la estadística o la física.

Uno de las principales aplicaciones de esta tecnología, y la que se va a aplicar en este TFG, es la detección de objetos, esto es la parte de la visión artificial que estudia cómo detectar la presencia de objetos en una imagen sobre la base de su apariencia visual. Se distinguen dos fases en el proceso de detección de objetos la extracción de características del contenido de una imagen y la búsqueda de objetos basada en dichas características.

La extracción de características tiene como objetivo resumir el contenido de la imagen, eliminando aquellas características prescindibles con el objeto de simplificar el proceso de búsqueda de objetos. Estas características de la imagen son conocidas como descriptores.

Para el proceso de búsqueda de objetos lo más común es la aplicación de algoritmos de aprendizaje máquina, si los objetos a detectar son más sencillos (como por ejemplo formas geométricas), y las exigencias del sistema no son muy elevadas, se pueden aplicar algoritmos más sencillos, sin necesidad de recurrir al aprendizaje máquina.

1.4 Organización del documento

El documento está estructurado siguiendo el orden en el que se ha desarrollado el trabajo, teniendo en cuenta la secuencia de las tareas realizadas.

- Capítulo 2: se dedica a la investigación de los métodos existentes para la detección del entorno del coche y a su adaptación para la implementación en la Raspberry Pi. A su vez este capítulo va a estar dividido en dos secciones. En la primera sección se proponen diferentes sistemas para la detección del carriles, y en la segunda sección se desarrolla un sistema de detección de señales de tráfico.
- Capítulo 3: se describen los pasos seguidos para la implementación del prototipo. Este capítulo se divide en cuatro secciones. En la primera sección se presentan los componentes Hardware usados en la construcción del robot, a continuación se describen los circuitos usados para las pruebas del vehículo. el sistema de control del mismo, y el sistema de detección de obstáculos.
- Capítulo 4: se realizan los experimentos con el robot, se exponen los resultados obtenidos y se realizará una discusión sobre los mismos.
- Capítulo 5: se realiza un análisis del resultado global del trabajo y se proponen una serie de mejoras.

Chapter 2

Procesado de Imagen

En esta sección se emula un sistema básico de procesamiento de imágenes de un vehículo autónomo. El sistema constará de un identificador de caminos y un detector de señales de tráfico.

2.1 Visión artificial

Teniendo en cuenta las limitaciones técnicas del proyecto, y también la simplicidad, hemos escogido la librería OpenCV (Rosebrock, 2015). La librería OpenCV, cuyas siglas en inglés vienen de Open Source Computer Vision Library una biblioteca de visión artificial, publicada mediante licencia BSD y desarrollada por Intel. Está escrita en C++, pero permite su integración en distintos lenguajes como Java, Matlab, o Python (nuestro caso). Esta librería, debido a su naturaleza de código abierto está siendo constantemente actualizada y posee una gran variedad de funciones. En este TFG se presentan algunas de las más básicas.

Para el tratamiento de imágenes por ordenador es necesario que las imágenes estén cuantificadas y discretizadas. Así pues vamos a trabajar con matrices de píxeles, sobre las que se realizarán operaciones algebraicas. Por lo tanto las imágenes van a tener dos propiedades:

- Dimensión de la imagen: Vendrá dado en formato matricial, y se podrá acceder a a esta información mediante la propiedad shape.
- Número de canales: Esta información está ligada al color. Para las imágenes a color vamos a tener tres canales, y en el caso de imágenes en blanco y negro dispondremos de un solo canal.

2.1.1 Algoritmos empleados para la detección de líneas

En este apartado se describe el hilo de ejecución del algoritmo de detección de líneas.

2.1.1.1 Obtención del área de interés

La cámara va a captar una región bastante grande, con multitud de contornos y sombras, por lo que va a ser necesario, filtrar el área para dejar la región en la cual

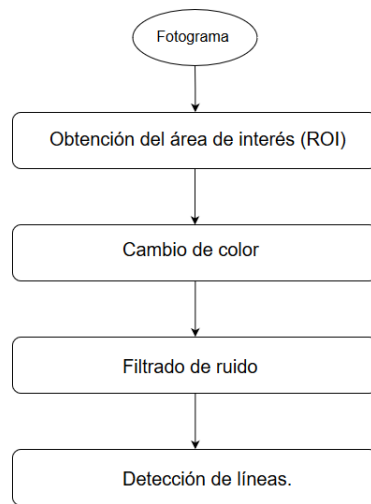


Figure 2.1: Diagrama de flujo para detección de líneas.

se detectan las líneas. Esta región va a tener forma rectangular para intentar captar la mayor información posible.

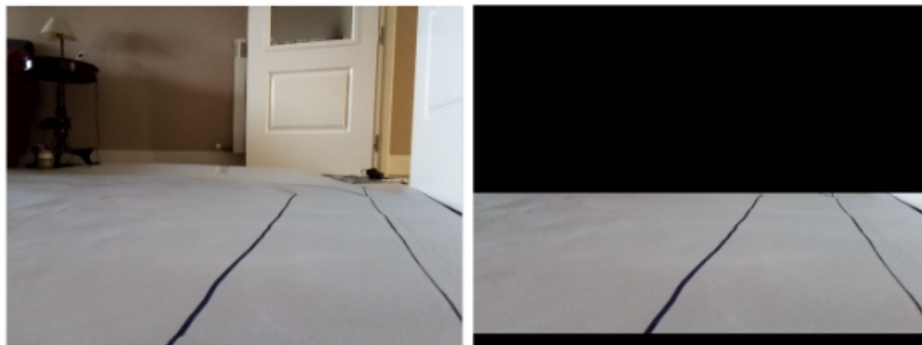


Figure 2.2: A la izquierda frame original, a la derecha región de interés.

Cambio de color

La unidad más pequeña, de una imagen es el píxel, así que cada imagen aparecerá representada como una matriz de píxeles. Además la cámara va a almacenar estas imágenes en formato RGB, esto significaba que cada píxel va estar formado por la superposición de tres matrices (rojo, verde, azul), cada uno de los cuales puede tomar valores de 0 a 255. OpenCV permite realizar la conversión de RGB a blanco y negro, en el que cada píxel va a estar representado por un único canal, la intensidad, que tomará valores de 0 a 255.

Filtrado Gaussiano

La operación de filtrado tiene como objetivo suavizar los contornos y hacer una imagen más clara. En el tratamiento digital de imágenes hay diferentes métodos para eliminar el ruido. Todos ellos tienen en común que utilizan la operación matemática de

convolución ésta consiste en ir recorriendo píxel a píxel una imagen con una máscara o kernel de $N \times N$. Este tamaño determina el número de píxeles con el que se va a trabajar.

En este caso se ha empleado el filtrado Guassiano, ya que es uno de los más efectivos y lo tenemos disponible para su uso en la librería OpenCV. Este tipo de filtrado emplea la campana de Gauss que en la práctica lo que hará es dar más importancia a los píxeles que están más cerca del centro de los que están más alejados. Para poder aplicarlo en una imagen debemos hacerlo en dos dimensiones, mediante una máscara o kernel de convolución.

Detección de bordes

La detección de bordes es una técnica muy utilizada que permite aislar los objetos y separarlos del fondo. Para ello se va a utilizar la función Canny disponible en la librería OpenCV. El proceso para detectar bordes con Canny se divide en 3 pasos:

- Detección de bordes con Sobel: se basa en el cálculo de la primera derivada. Esta operación matemática mide las evoluciones y los cambios de una variable. Básicamente se centra en detectar cambios de intensidad entre píxeles vecinos.
- Supresión de píxeles fuera del borde: El objetivo en esta fase es poder seleccionar aquellos bordes que cumplan cierta condición. En el detector de bordes Canny serán aquellos que tengan como grosor 1.
- Aplicar umbral por histéresis: La umbralización de imágenes permite segmentar una imagen en sus diferentes objetos. Básicamente consiste en determinar un umbral por el cual se decide si un píxel forma parte del fondo o forma parte de un objeto. Al contrario que el umbral simple, el umbral por histéresis se centra en establecer dos umbrales, uno máximo y otro mínimo. El píxel se considerará parte del borde si está por encima del umbral máximo, y no será borde si está por debajo del umbral mínimo, en el caso de estar entre el máximo y el mínimo, será parte del borde si está conectado con un píxel que forma ya parte del borde.

Detección de líneas

Este es el último paso de la visión artificial y también ha sido el más complejo, ya que dependiendo de la superficie sobre la que se vayan a detectar estos objetos, vamos a tener un gran número de errores. Así pues se han desarrollado dos métodos para la detección de líneas, siendo el primero un método completamente original del alumno, sin usar más funciones de la librería OpenCV, y el segundo haciendo uso de la transformada de Hough.

Método original del alumno A pesar de que en la mayor parte de la bibliografía se ha empleado la transformada de Hough para la detección de líneas, si la superficie de pruebas no tiene interferencias tales como ruido o reflejos en el suelo, ha sido más sencillo desarrollar nuestro propio algoritmo de detección de líneas.

La imagen sobre la que se vamos a identificar las líneas debería estar compuesta por dos únicas líneas, de color blanco situados sobre un fondo de color negro. Se van a presentar 3 casos diferentes:

- Líneas cuyo comienzo coincide con la parte inferior del área seleccionada.
- líneas cuyo comienzo, está situado en un lateral, por lo que no coincide, con la parte inferior del área seleccionada.
- una sola linea detectada.

Se va a realizar una iteración, por los bordes de la imagen, desde la esquina superior izquierda, en sentido antihorario, evaluando la intensidad de cada píxel. Recordemos que debido a que la imagen está en blanco y negro los dos únicos valores posibles serán 0 (negro) y 255 (blanco). Así pues, cuando detectemos un píxel de color blanco este será interpretado como un punto perteneciente a una línea. Aún así dado que, debido a reflejos, relieves, o sombras, se pueden obtener falsos blancos, se realizará una media de los valores obtenidos de los últimos 10 frames para darle una mayor estabilidad al sistema.

Detección usando la transformada de Hough El segundo método se va a emplear en entornos con más interferencias. Para desarrollar este algoritmo ha sido necesario el uso de la transformada de Hough, esta función está ya definida en la librería OpenCV. Para entender como funciona este método es necesario explicar el concepto que hay detrás de la transformada.

La ecuación matemática de una recta, expresada en el sistema de coordenadas polares, se puede representar de la siguiente manera $y = m \cdot x + n$. El parámetro ρ representa la distancia entre el origen de coordenadas y el punto (x,y), mientras que θ es el ángulo del vector director de la recta perpendicular a la recta original y que pasa por el origen de coordenadas. A cada recta le corresponderá un par (ρ, θ) único. Para

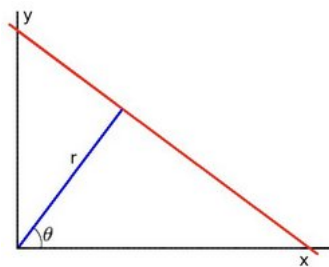


Figure 2.3: Representación de la transformada de Hough.

un punto cualquiera (x_o, y_o) , existe un conjunto de rectas que pueden pasar por ese punto, definidas por $\rho = x \cdot \cos\theta + y \cdot \sin\theta$. Esto corresponde a una curva sinusoidal en el espacio, única para ese punto (OpenCV, 2012). Si se toma un punto cualquiera y se representa esta senoide en el plano (ρ, θ) se obtiene la siguiente curva.

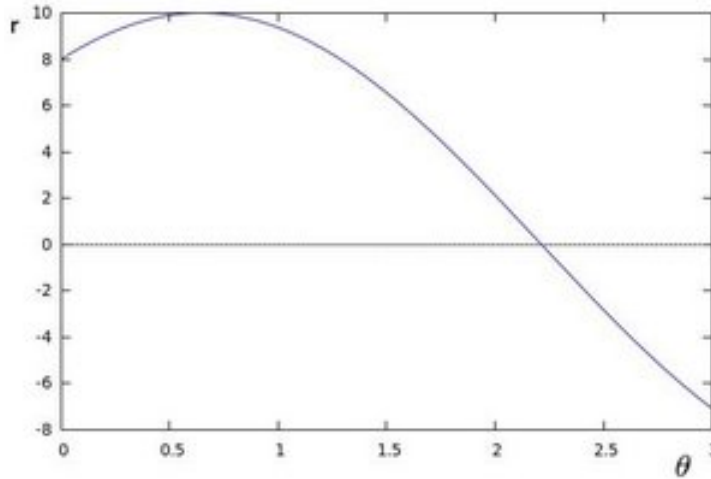


Figure 2.4: Conjunto de rectas que pasan por un determinado punto.

Si las curvas correspondientes a dos puntos se intersecan, el punto de intersección en el espacio de Hough corresponde a una línea en el espacio de la imagen que pasa por estos dos puntos. Por tanto, el problema de detectar puntos colineales se puede convertir en un problema de buscar curvas concurrentes.

Para cada punto (x, y) de una imagen el algoritmo comprueba cuáles rectas pasan por ese punto (x, y) . Cada recta tiene un contador de tal manera que cada vez que un punto (x, y) pertenece a una recta (ρ, θ) , el contador correspondiente se incrementa en 1 (OpenCV, 2012). Así pues si se evalúa un punto cualquiera (x_o, y_o) perteneciente a una recta, se obtendrán muchas rectas con parámetros (ρ, θ) que pasarán por ese punto y pondrán sus respectivos acumuladores en 1. Sin embargo sólo una recta de parámetros $\rho = \rho_o$ y $\theta = \theta_o$ seguirá acumulando por encima de 1. Por lo tanto mirando el conjunto de valores del contador de cada recta y obteniendo el valor máximo y el par (ρ, θ) correspondiente a dicho contador, se podrá definir la recta. La llamada a la función OpenCV, nos devolverá los valores de ρ y θ que caracterizarán a cada línea (OpenCV, 2012). Para consumir menos recursos haremos uso de la transformada probabilista, que en vez de todos los puntos solo toma un subconjunto de puntos al azar y eso es suficiente para la detección de líneas, únicamente habrá que disminuir el umbral. La función nos devolverá el par de puntos $(x_1, y_1), (x_2, y_2)$, que definen la recta.

De la imagen se van a extraer muchas líneas, el siguiente paso es filtrar aquellas líneas que no pertenezcan al carril, y agrupar las que si, en función de su pendiente. Aquellas líneas con pendiente negativa van a representar el borde izquierdo del carril, y las de pendiente positiva el borde derecha. En caso de no detectar ninguna línea se asumirá que las de los instantes anteriores se encontrarán en la misma posición de manera similar a cuando se encuentra nieve o no están bien definidos los carriles en la carretera.

Con cualquiera de los dos métodos deberíamos obtener las líneas de la imagen, no obstante experimentalmente se ha llegado a la conclusión de que la transformada de Hough es más fiable en suelos más irregulares, pero en carriles estrechos con curvas pronunciadas puede dar lugar a errores. El método del alumno detecta las líneas y

las curvas pronunciados siempre y cuando el suelo no sea muy heterogéneo en cuanto a formas y colores.

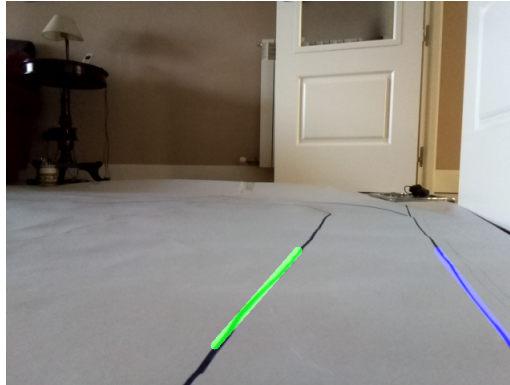


Figure 2.5: Carril detectado.

2.1.2 Detección y clasificación de señales de tráfico

En este proyecto se ha hecho uso de seis señales de tráfico, estas serán STOP, y sentidos obligatorios derecha, izquierda, recto, recto/derecha, recto/izquierda. La

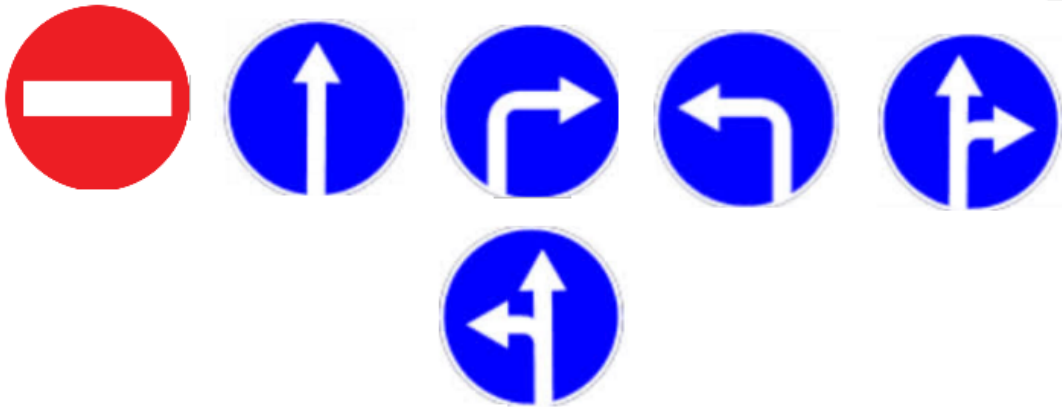


Figure 2.6: Señales de tráfico usadas.

detección de señales de tráfico va a constar de dos fases, la primera de ellas es la localización de la señal en el circuito y la segunda es la clasificación del tipo de señal.

2.1.2.1 Localización de las señales de tráfico

El primer paso es definir la zona en la que se van a encontrar las señales, que será la parte superior de la imagen. Se aplicará del mismo modo que en la detección de líneas una segmentación de la imagen mediante la aplicación de una máscara, esta vez en la parte superior.

El siguiente paso es encontrar la posición exacta de la señal, buscando en el espacio su forma geométrica característica (círculos). De forma similar a la detección de líneas, se va a realizar una conversión a escala de grises y posteriormente se va a aplicar un filtrado Guassiano para suavizar la imagen y eliminar ruido.

Posteriormente se va a aplicar la función disponible en OpenCV para la detección de círculos `HoughCircles`, cuyo funcionamiento se detalla a continuación: la expresión matemática que define una circunferencia es: $(x-cx)^2 + (y-cy)^2 = r^2$, donde r es el radio de la circunferencia, cx la coordenada en eje X del centro de la circunferencia y cy la coordenada en eje Y del centro de la circunferencia. Así pues la circunferencia queda definida por estos tres parámetros. Para cada píxel de contorno que aparece

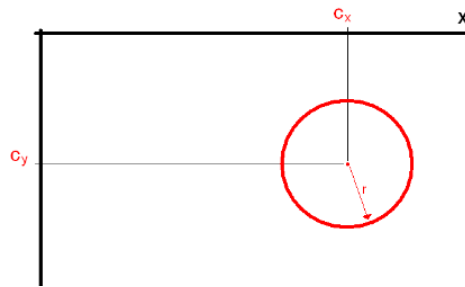


Figure 2.7: Parámetros de una circunferencia.

en la posición (x_0, y_0) existe una familia de circunferencias que pasan por este punto dadas por:

$$c_x = x_0 + \cos\Theta \cdot r$$

$$c_y = y_0 + \sin\Theta \cdot r$$

Una vez terminado este procedimiento se buscan los picos en el acumulador y con esto se obtienen el radio y el centro de la circunferencia. Si se conociera el radio de antemano, solo se necesitaría un acumulador de dos dimensiones. Cada píxel de contorno vota por todas las posibles circunferencias que pasan por él. La circunferencia que tenga un mayor número de votos es la que se considerará como la circunferencia de la imagen. Es importante destacar que para evitar detectar señales en la lejanía o falsos positivos, por otras formas o contornos, se va a escoger la circunferencia de mayor radio, y esta será la que será clasificada posteriormente.

2.1.2.2 Clasificación de la señal

La gran mayoría de los sistemas de reconocimiento de señales de tráfico incorporados en los vehículos realizan la detección de señales empleando una imagen plantilla, ya guardada en la base de datos. Para cada píxel de la señal, el coeficiente de correlación cruzada entre el patrón y la búsqueda de imágenes se calcula para cada canal de color (rojo, verde y azul). El coeficiente final es la media de los tres coeficientes RGB. El patrón con el mayor coeficiente corresponde a la señal registrada. Otro método más novedoso es el del reconocimiento de patrones usando redes neuronales. Sin embargo dadas las limitaciones de este proyecto y debido a que solo se va a hacer uso de 6 tipos de señales (todas circulares) en todo el proyecto, se ha optado por

desarrollar un sistema de clasificación basado en el color predominante de la señal o de ciertas secciones de dicha señal. Los valores de los tres canales (rojo, verde y azul), de cada pixel se pueden interpretar como coordenadas tridimensionales. Si se toma por ejemplo la imagen de la Figura 2.8 y se representa en un espacio tridimensional,

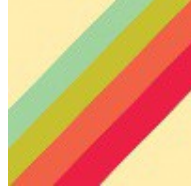


Figure 2.8: Imagen con diferentes colores.

siendo cada uno los ejes la intensidad de estos tres colores primarios.

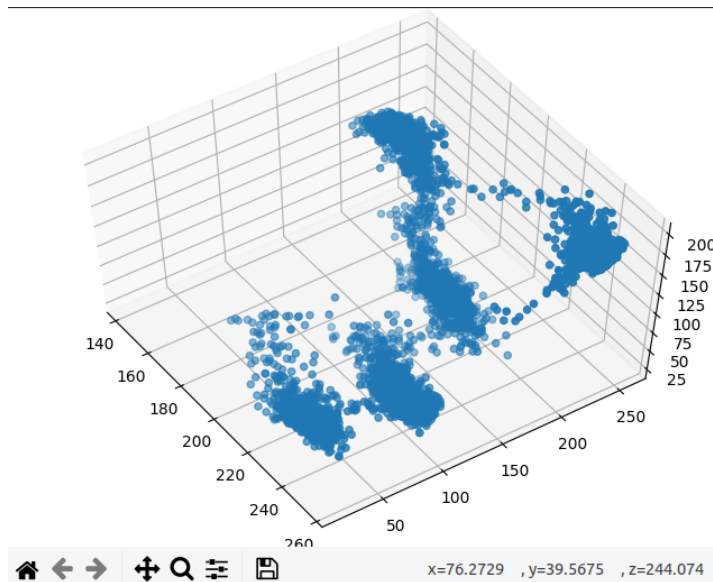


Figure 2.9: Imagen con diferentes colores.

En la Figura 2.9 puede observar que hay ciertas regiones en la gráfica en la que los puntos tienden a acumularse, formando grupos o clusters lo que puede interpretarse como los diferentes colores de la imagen.

Para la detección de color predominante se ha utilizado la función de agrupamiento de datos *kmeans* disponible en la librería OpenCV. El objetivo de este algoritmo es el de encontrar K grupos (clusters) entre n datos (Rosebrock, 2014). El algoritmo trabaja iterativamente para asignar a cada punto uno de los K grupos basado en sus características. De la ejecución de esta función se van a obtener dos valores: Una etiqueta identificativa de cada cluster y los centroides, compuestos por la media de todos los valores de cada cluster. En este caso nuestros objetos son los píxeles y sus características son las intensidades de rojo, verde y azul asociada a cada uno. Por lo tanto tenemos tantos datos u objetos como píxeles y tres características o propiedades

para cada píxel. Para poder aplicar k-means se va reconfigurar la matriz original de la imagen para crear una matriz con tantas filas como píxeles y cada fila/píxel va a estar representada por 3 columnas, una para cada intensidad de color. Por lo tanto tras la ejecución de k-means, los centroides representarán los colores resultantes de la media de todos los colores de su cluster.

El último paso es determinar cuantos píxeles contiene cada cluster, el valor centroeide del cluster que contenga un mayor número de pixeles se definirá como el color dominante.

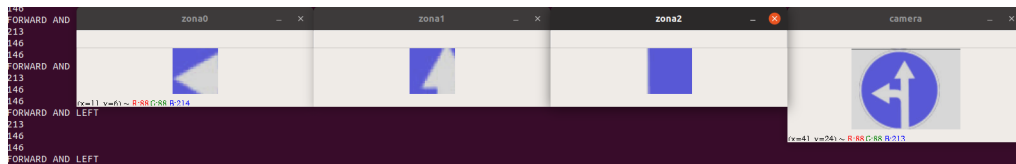


Figure 2.10: Regiones evaluadas para la clasificación del tipo de señal.

La primera clasificación discierne si el color dominante es el azul, o en cambio es el color rojo. Si el color es rojo, entonces automáticamente la señal queda clasificada como STOP, en caso contrario será necesaria una segunda clasificación, basada en la presencia de flechas (color blanco), en una región u otra de la imagen. Las flechas de dirección van a estar situadas en la región superior de la señal, así pues se hará una división en tres regiones, para determinar la dirección (derecha, izquierda o recto), en función de la mayor o menor presencia de blanco en la imagen. En la Figura 2.11 se puede observar la implementación en tiempo real, mediante la cámara de la Raspberry Pi.

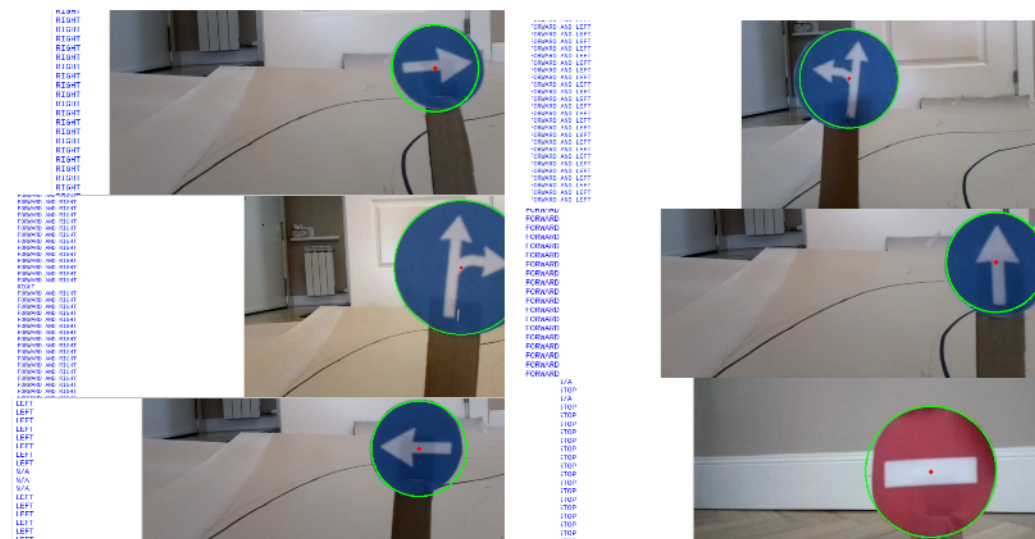


Figure 2.11: Resultados del algoritmo de detección y clasificación de señales de tráfico.

Este método de clasificación de señales ofrece generalmente buenos resultados,

pero bajo condiciones adversas de iluminación puede ser difícil detectar los bordes de la señal y en consecuencia realizar la clasificación de la misma. En el caso de las señales de dos direcciones, puede ser difícil determinar la dirección lateral si la imagen no está bien enfocada. En estos casos el programa indicará solo la dirección recta. Cuando el vehículo está en movimiento podemos encontrar que la señal no está completamente contenida dentro de la imagen, lo que puede causar fallos en el sistema. La señal más robusta frente a los errores es la señal de stop, ya que básicamente se trata de detectar un círculo rojo y no puede ser confundida con cualquier otra señal.

Chapter 3

Implementación

3.1 Implementación del prototipo

3.1.1 Descripción general del sistema

El prototipo va a estar basado en un robot móvil. El movimiento se realiza a través de un motor de corriente continua, y la dirección se controlará a través de un servomotor. El prototipo estará equipado con una cámara, RaspiCam, que irá obteniendo imágenes del entorno que serán analizadas por controlador principal (RaspberryPi 3 Model B). A su vez la Raspberry Pi se comunicará con los controladores de los motores, y recibirá información de los objetos cercanos a través del sensor de ultrasonidos.

3.1.2 Raspberry Pi 3

El robot va a estar controlado por el ordenador de placa reducida Raspberry Pi. El modelo seleccionado ha sido la Raspberry Pi 3 Model B. La Raspberry nos ofrece la posibilidad de utilizar varios lenguajes de programación (como C, C++, Python). Tiene un precio asequible y una capacidad de procesamiento más que suficiente para desarrollar los algoritmos de visión artificial. Este dispositivo cuenta con un procesador ARMv8 de cuatro núcleos que funciona a 1,2 GHz y utiliza instrucciones de 64 bits.

La placa también cuenta con 40 pines de entrada y salida de uso general (GPIO), para el control de sensores, motores y otros componentes, 4 puertos USB, un puerto Ethernet, una conexión HDMI, y la interfaz de la cámara CSI. Además lleva incorporado un módulo Bluetooth 4.1 Low Energy y un módulo wifi, lo que permite establecer una conexión remota mediante SSH y VNC.



Figure 3.1: Raspberry Pi.

Para la comunicación usando pines no definidos como UARTS se ha hecho uso de la librería WiringPi y ha sido necesario tener especial cuidado con la diferencia de voltajes de las conexiones.

3.1.2.1 Sistema Operativo

La Raspberry Pi 3 puede funcionar con diferentes tipos de sistema operativo Linux. En este TFG se va a usar raspbian, una distribución basada en Debian, creado específicamente para la Raspberry Pi, y que incluye varios paquetes instalados por defecto como por ejemplo python. Además, deberá instalarse la librería OpenCV, y otras bibliotecas matemáticas como numpy o matplotlib.

Para realizar todos estos procesos fue necesario la activación de los puertos SSH, VNC y la configuración de una red wifi con una dirección IP fija, para establecer la conexión a la placa desde el ordenador sin necesidad de conexión a un monitor.

La comunicación con la placa se ha hecho mediante SSH y VNC, por lo que ha sido necesaria la activación de estos puertos y la configuración de una dirección IP fija, para establecer la conexión a la placa desde el ordenador sin necesidad de conexión a un monitor.

3.1.3 Camera Module V2

Para detectar el entorno que se sitúa frente al robot, se va a utilizar la "Camera Module v2", diseñada específicamente para la Raspberry pi. Esta cámara nos va a permitir capturar imágenes con una resolución de hasta 8 megapíxeles. Está conectada a la Raspberry directamente a través del conector CSI (Camera Serial Interface).

A pesar de que su resolución de vídeo pueda llegar hasta los 1080p, se va a reducir la calidad de vídeo con el objetivo de aumentar la velocidad de cómputo.



Figure 3.2: RaspiCam.

Se ha elegido esta cámara debido a que por su reducido tamaño es perfecta para aplicaciones de robótica.

3.1.4 Servomotor

Un servomotor es un motor similar a un motor de corriente continua que tiene la capacidad de ubicarse dentro de cualquier posición dentro de su rango de operación, y mantenerse estable dentro de dicha posición. Para este proyecto se ha decidido que la solución más adecuada para el control de dirección del robot es el servomotor Futaba S3001.

El control interno que lleva el servomotor se hace en lazo cerrado y solo con un parámetro: la posición que tiene el potenciómetro. Cuando un servomotor recibe la señal de “cambiar de posición”, se crea un error en el sistema y su unidad de control activa el motor DC interno.

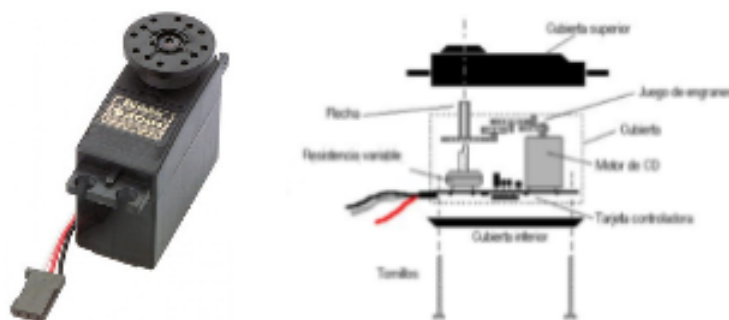


Figure 3.3: Servo Futaba-S3001(izda). Partes internas del servomotor(dcha).

Atendiendo a las especificaciones del componente vemos que tiene una velocidad de giro de 0.28 segundos por 60 grados para una entrada de 4.8 V, lo que es suficiente para nuestro proyecto. El servo posee tres cables: rojo, para la alimentación Vcc (atendiendo a la hoja de características 4.8- 6 V), negro, para la conexión a tierra y

por último blanco, por donde envía la señal PWM para comunicar el ángulo en el que se debe posicionar.

Los principales inconvenientes de estos motores es la falta de mecanismos de protección contra sobre-esfuerzos, altas temperaturas e inversiones de polaridad.

3.1.5 PCA9685

El controlador PCA9685 fue diseñado para controlar leds por PWM, pero también permite controlar servos, ya que estos también se controlan por PWM. El PCA9685 permite generar hasta 16 señales PWM, o controlar 16 servos, únicamente empleando 2 pines. La frecuencia del PWM es ajustable hasta a 1600 Hz, y la precisión de 12 bits.

La comunicación se realiza a través del bus I2C, por lo que es sencillo obtener los datos medidos.

Adicionalmente, la placa permite modificar su dirección I2C por defecto (0x40) mediante 6 pines, con lo que se pueden conectar hasta 62 placas iguales en el mismo bus (con 6 pines teóricamente tendríamos $2^6=64$ direcciones, pero dos de estas se reservan). De este modo se podría controlar un sistema de hasta 992 dispositivos PWM.

Para su uso en control de servos mediante la Raspberry Pi, se dispone de la librería `adafruit_servokit`, que mueve el servo a la posición deseada indicando el ángulo de giro (Adafruit, 2019).

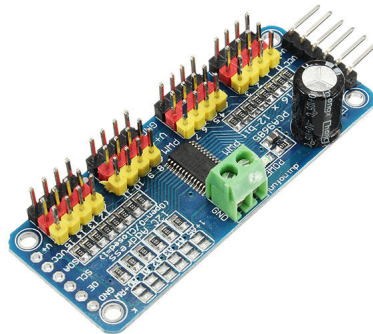


Figure 3.4: Circuito integrado PCA9685.

3.1.6 Motor DC

Se ha utilizado un motor de corriente continua para la tracción trasera del coche. Se ha elegido el motor de la compañía Maxom modelo A-max32 - 12V. A partir de la información obtenida de las hojas de características del fabricante, se obtiene la siguiente tabla de características.

El motor está equipado con un encoder rotativo modelo HEDS-5540. Este sensor nos da información sobre la posición del motor, proporcionando una serie de pulsos que pueden ser capturados mediante un microprocesador. Dependiendo de cual sea el mecanismo físico a través del cual se captura la posición, se establece una clasificación

Parámetro	Valor	Unidades
k_m , Constante de fuerza electromotriz	$24,3 \cdot 10^{-3}$	[Nm/A]
J_m , Inercia del rotor	$41,4 \cdot 10^{-7}$	[g · cm ²]
L_m , Inductancia del motor	$0,264 \cdot 10^{-3}$	[mH]
R_m , Resistencia terminal	2,23	Ω
n_o , Velocidad angular del motor sin carga	324,63	[rpm]
I_o , Corriente del motor sin carga	0,058	[A]

Tabla 3.1: Tabla de características del motor.

de encoders magnéticos, ópticos o capacitivos, el HEDS-5540 usa una fuente de LED por lo que será un encoder óptico. El encoder será absoluto o incremental en función de si genera un código inequívoco para cada posición o si en cambio genera un número determinado de pulsos por vuelta. Este motor dispone de un encoder incremental. La distribución de los pines del encoder es la siguiente:

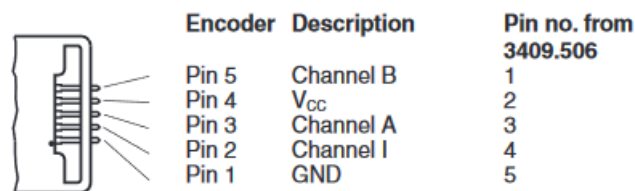


Figure 3.5: Encoder del motor.

3.1.7 Regulador de tensión

Los servomotores van a ser alimentados con una tensión de 5 V. Para obtener esta tensión de salida, haremos uso del circuito integrado 7805, que nos proporcionará 5V al recibir a la entrada los 12 V provenientes de nuestra batería. Nuestro regulador de tensión quedará tal y como se muestra en la Figura 3.6. Es importante el uso de condensadores de 0.33 μ F y 0.1 μ F, en la entrada y la salida respectivamente, para reducir riesgos de error por interferencias que puedan entrar por los cables.

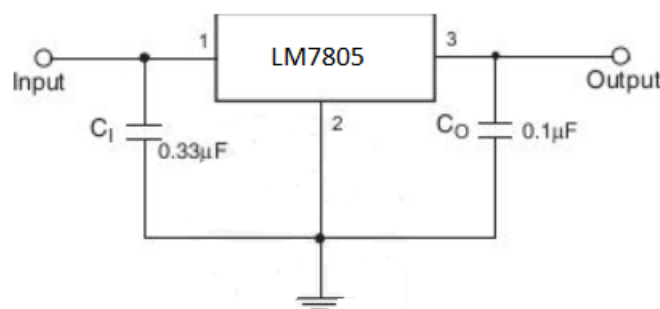


Figure 3.6: Esquemático del circuito regulador de tensión.

3.1.8 Sensor de ultrasonidos HC-SR04

Para la detección de obstáculos se ha hecho uso del sensor de ultrasonidos HC-SR04. Este sensor emite pulsos ultrasónicos que, al reflejarse en un objeto, produce un eco que es recibido por el sensor y éste lo convierte en señal eléctrica. El sensor interpretará la distancia en función del tiempo transcurrido entre el pulso de emisión y el pulso de eco recibido.

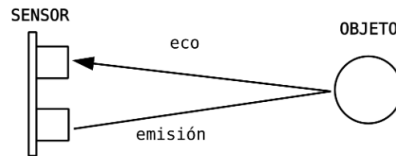


Figure 3.7: Funcionamiento del módulo HC-SR04.

Cuando recibe un pulso en su entrada TRIG, este módulo emite ocho pulsos a 40kHz y recibe los ecos de los mismos. La salida ECHO genera un ancho de pulso proporcional a la distancia al obstáculo.

Para obtener la distancia al obstáculo calcularemos el rango a través del intervalo entre la señal enviada del trigger y la señal recibida del eco. La velocidad del sonido a 20º de temperatura es 340m/s. Como la señal tiene que ir y volver, la distancia será la mitad de la recorrida. La siguiente ecuación, es la que se ha empleado para su cálculo:

$$d(cm) = \frac{t(\mu s) * 340(m/s)}{2} \quad (3.1)$$

Un aspecto importante a tener en cuenta al utilizar este modelo de sensor en una Raspberry Pi, es que funciona con 5V, tanto para su alimentación como para su salida hacia la Raspberry Pi, la cual trabaja a 3,3V. Para solucionar el problema y no estropear nuestro dispositivo, tendremos que hacer un divisor de tensión entre la salida del sensor y la Raspberry, de esta manera podremos fijar una tensión intermedia entre el nivel de alimentación y el nivel de tierra.

3.1.9 Controlador de motor L298N

Los motores son controlados en velocidad y sentido de giro por una señal modulada en ancho de pulso (PWM). El módulo controlador de motores L298N H-bridge nos permite controlar la velocidad y la dirección de dos motores de corriente continua o un motor paso a paso de una forma muy sencilla, gracias a los 2 puentes-H que monta. El módulo cuenta con todos los componentes necesarios para funcionar sin necesidad de elementos adicionales, entre ellos diodos de protección y un regulador LM7805 que suministra 5V a la parte lógica del integrado L298N. Cuenta con jumpers de selección para habilitar cada una de las salidas del módulo (A y B) para cada uno de los dos motores. La salida A está conformada por OUT1 y OUT2 y la salida B por OUT3 y OUT4. Los pines de habilitación son ENA y ENB respectivamente.

El módulo tiene dos modos de operación en función de la tensión de alimentación.

Si la tensión de trabajo se encuentra entre 6 y 12 V, el regulador se encontrará

activo y el pin V lógico proporcionará una salida de 5V, que se podrá usar como tensión de alimentación para otro dispositivo siempre que no requiera una tensión de alimentación mayor a 500 mA.

Si la tensión de trabajo se encuentra entre 12 y 35 V, el regulador está inactivo, y será necesario alimentar la parte lógica del L298N, con 5 V.

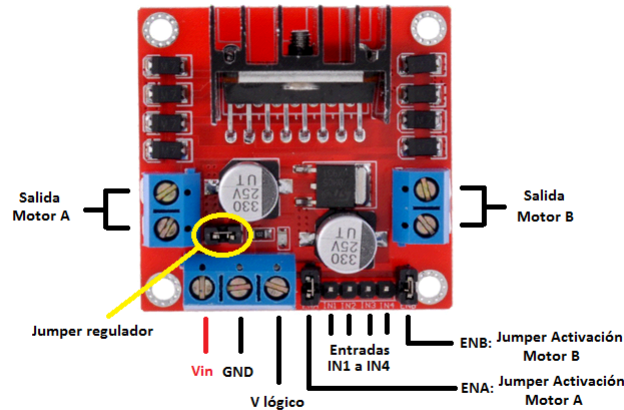


Figure 3.8: Módulo L298N.

3.1.10 Sistema de alimentación

La alimentación del prototipo, utiliza dos baterías, para la alimentación de la raspberry pi y para la alimentación de los motores. La opción elegida para alimentar la Raspberry Pi, es una batería de Litio recargable con una capacidad de 10000mAh, y capaz de suministrar una salida de 5V/2.1A lo que será suficiente para los periféricos que irán conectados a la raspberry. Para la alimentación del motor de corriente continua, y de los servos, vamos a emplear una batería de 12 V. Esta batería se va a montar mediante la conexión serie de 10 baterías de niquel-cadmio, de 1,2 V y una capacidad de 2 mAh.

3.1.11 Pista de pruebas

Originalmente se utilizó cinta aislante negra colocada sobre el suelo, para diseñar la pista, pero tanto el parqué como el suelo de baldosas introducían muchas interferencias en la imagen. Así que finalmente se ha optado por dibujar la pista sobre superficies de papel. Se han utilizado dos pistas, la primera para la fase inicial del proyecto es una pista recta de 4 metros, y la siguiente, es una pista curva de 4,5 metros sobre las que se realizará las operaciones con el coche en movimiento.

3.2 Sistema de control. Guiado del vehículo

El objetivo es mantener el vehículo posicionado en el centro de la pista, para ello lo primero que se ha de hacer es obtener el punto que vamos a definir como el centro de

la pista. Se define el centro de la pista como el punto medio entre los puntos inferiores de las dos líneas.

$$x_{mid} = \frac{x_{iz} + x_{der}}{2} \quad (3.2)$$

El punto medio de la imagen se considera la posición del coche, así que la distancia entre el centro de la pista y el coche se define como:

$$\Delta x = x_{mid} - \frac{w_{imagen}}{2} \quad (3.3)$$

Puesto que solo se va a realizar el control en la dirección del vehículo, es necesario fijar una velocidad a la cual se van a hacer las pruebas. Se ha implementado un controlador proporcional para controlar la dirección. Dada la escasa información medible de la que se dispone, el controlador se ha ajustado mediante técnicas empíricas.

3.2.1 Distancia de la referencia

El centro de la pista, hacia donde se va a dirigir el coche esta situada a una determinada distancia del coche que se va a definir como la proyección ortogonal de la posición del vehículo en el camino. Cambiar esta distancia va a producir cambios significativos en el movimiento del vehículo. Si elegimos una distancia demasiado corta, el vehículo oscilaría y tendríamos un comportamiento inestable. Si se aumenta esta distancia, el vehículo realizará un seguimiento más suave, sin embargo, se corre el riesgo de perder precisión durante las curvas.

Este parámetro se puede ajustar de dos formas, bien modificando en el programa la región de interés con la que vamos a trabajar, o bien ajustando el ángulo de inclinación de la cámara. Esta distancia en principio va a ser de 15 cm, pero podrá estar sujeta a modificaciones en función del tipo de pista. En la sección de experimentos se desarrollará más detenidamente su impacto en la conducción del coche.

3.2.2 Diseño de un controlador P

Un sistema de control proporcional es un tipo de sistema de control lineal. El controlador proporcional da una salida que es proporcional al error: $u(t) = K_p \cdot e(t)$, donde K_p es la ganancia proporcional ajustable. Es el controlador más simple, ya que solo se puede utilizar el valor de K_p para modificar sus especificaciones. Los efectos de aumentar la constante de proporcionalidad pueden ser: aumentar la velocidad de respuesta del sistema, disminuir el error del sistema en régimen permanente y finalmente aumentar la inestabilidad del sistema. Este último es un efecto no deseado, por lo que se debe minimizar y tener especial precaución con la entrada de ruido que puede desestabilizar más al sistema.

Se ha asumido que la distancia máxima entre el centro de la pista y el coche es de 70 píxeles, así que se establece que a un valor de error respecto al centro de la pista menor o igual que -70 le corresponde un ángulo de 0° grados en el servo y a un valor mayor o igual que 70 un ángulo de 180°. La ecuación para obtener el ángulo será la siguiente:

$$angulo = \frac{-52 \cdot error}{80} + \frac{52 \cdot 110}{80} \quad (3.4)$$

Tras varias pruebas se ha comprobado que para errores en el intervalo $[-40,40]$ el coche se encuentra totalmente centrado en la pista y el giro solo desestabiliza el coche, así que no se aplicará ninguna acción de giro para este rango de valores.

3.3 Sistema de detección obstáculos

Para la detección de obstáculos en la vía se ha hecho uso de los sensores de ultrasonidos HC-SR04, ya que tienen un alcance suficiente. Aunque estos tienen un alcance máximo de cuatro metros, no es necesario usar distancias tan amplias para un circuito de pruebas de un tamaño tan pequeño, sobretodo teniendo en cuenta que los tramos en línea recta del circuito no tienen más de 3 metros. Además, puesto que el trayecto discurrirá cerca de muebles fijos, se deberá ajustar la distancia para evitar que se detecten erróneamente objetos situados fuera de la pista, se escogerá una distancia máxima de medida que se encontrará dentro de la zona de visión del vehículo.

Se establecerá también una distancia inferior bajo la cual se encuentra la zona de parada. En cualquier circunstancia, el vehículo se detendrá si un objeto es detectado en esa zona.

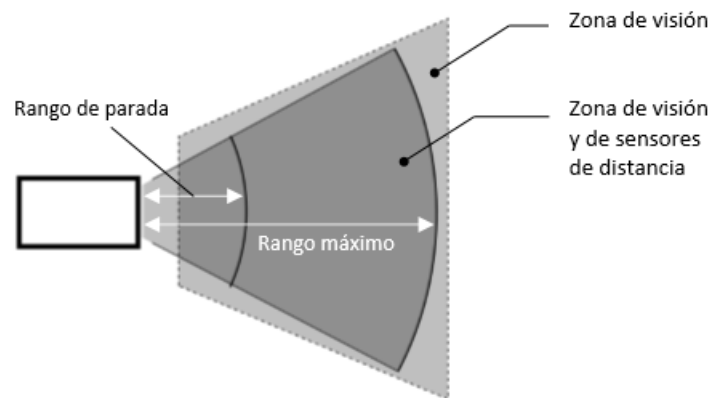


Figure 3.9: Rangos de medida del sensor ultrasónico.

En condiciones normales de operación en que el objeto es detectado por algún sensor de distancia, el vehículo reducirá la velocidad hasta que el objeto quede en la zona de parada, entonces el vehículo se detendrá. Si el objeto se retira de la pista o se desplaza de tal manera que quede fuera de la zona de parada del sensor de distancia, el vehículo reanudará la marcha de forma normal.

Chapter 4

Experimentos

4.1 Determinación del error de la posición calculada con respecto a los extremos de la pista

El siguiente experimento se realiza con el objetivo de cuantificar el error de la posición calculada del vehículo, más concretamente la posición de la cámara, situada en el centro del prototipo, con respecto al borde izquierdo de la pista. El vehículo fue movido a diferentes distancias con respecto a la línea izquierda del carril y se realizaron un total de 10 mediciones por posición. Se van a emplear los dos algoritmos de detección de líneas para detectar cual de los dos es más preciso en este escenario.

Para contrarrestar el error humano de la colocación del coche en la pista, se ha repetido tres veces el experimento. Se han realizado 5 mediciones, en intervalos de 50 mm. Para el cálculo de la posición del coche con respecto a la pista, se ha creado la función `PixelsToMillimeter`, que realiza la conversión de píxeles a milímetros mediante la siguiente ecuación.

$$distancia[mm] = \frac{|x_{izq} - centroImg|}{x_{der} - x_{izq}} \quad (4.1)$$

donde x_{der} y x_{izq} son los puntos inferiores de las rectas y $centroImg$ es el centro de la imagen que representa la posición de la cámara del coche.

En las Figuras 4.1 y 4.3, se representa la posición respecto al borde izquierdo calculada por el algoritmo (en mm) frente a la posición real de la cámara para los dos algoritmos de detección de líneas. En las Figuras 4.2 y 4.4, se observa la variación del error normalizado en función de la posición del coche en la pista. Para el calculo del error se asume que el máximo error es de 200 píxeles cuando el vehículo se encuentra totalmente centrado en la línea del borde del carril.

4.1.1 Resultados experimentales utilizando la transformada de Hough

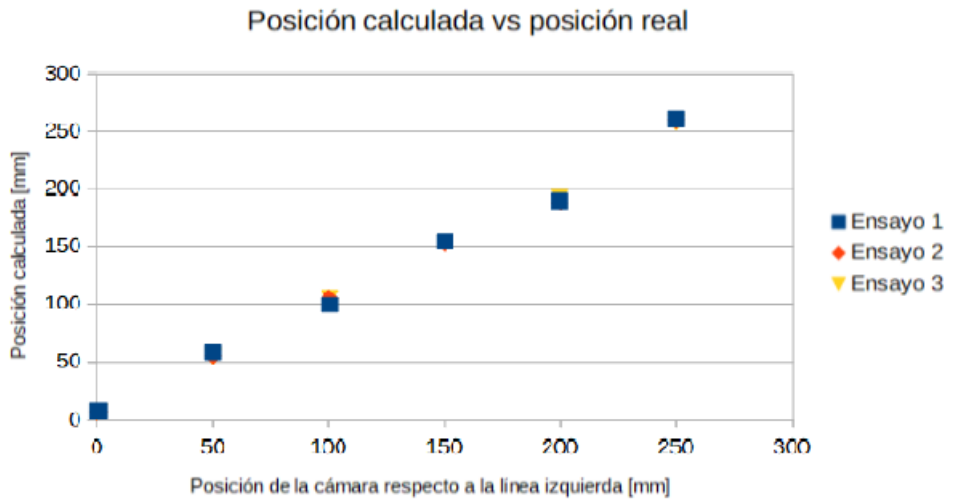


Figure 4.1: Posición calculada de la cámara frente a la posición real empleando la transformada de Hough.

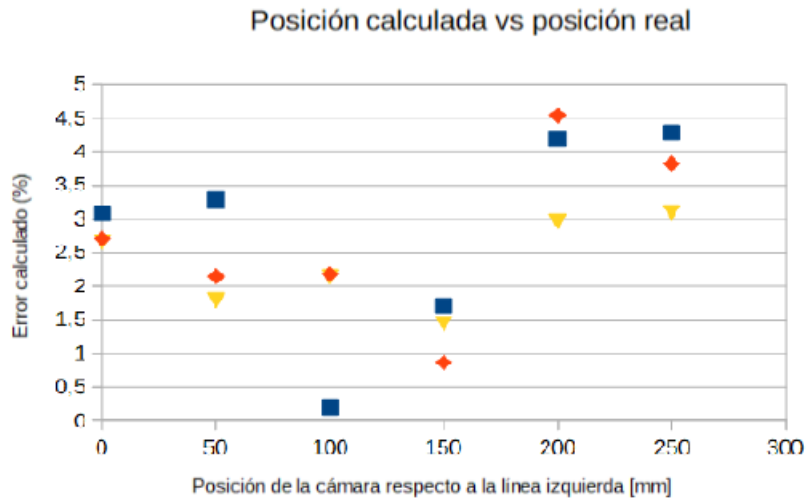


Figure 4.2: Error de la posición calculada del vehículo comparada con la posición real.

En la Figura 4.1, se observa que no hay una correspondencia exacta entre los valores calculados y los reales, ya que la posición calculada siempre es ligeramente superior a la posición real. Se observa que el error se ha mantenido prácticamente igual en los tres ensayos. De la variación del error presentada en la Figura 4.2 se observa que el error aumenta cuando el vehículo se encuentra en alguno de los extremos, lo que se debe a que las pendientes de los extremos del carril tienden a igualarse siendo difícil

discernir cual es el carril izquierdo y el carril derecho en función de las pendientes.

4.1.2 Resultados experimentales utilizando el método del alumno

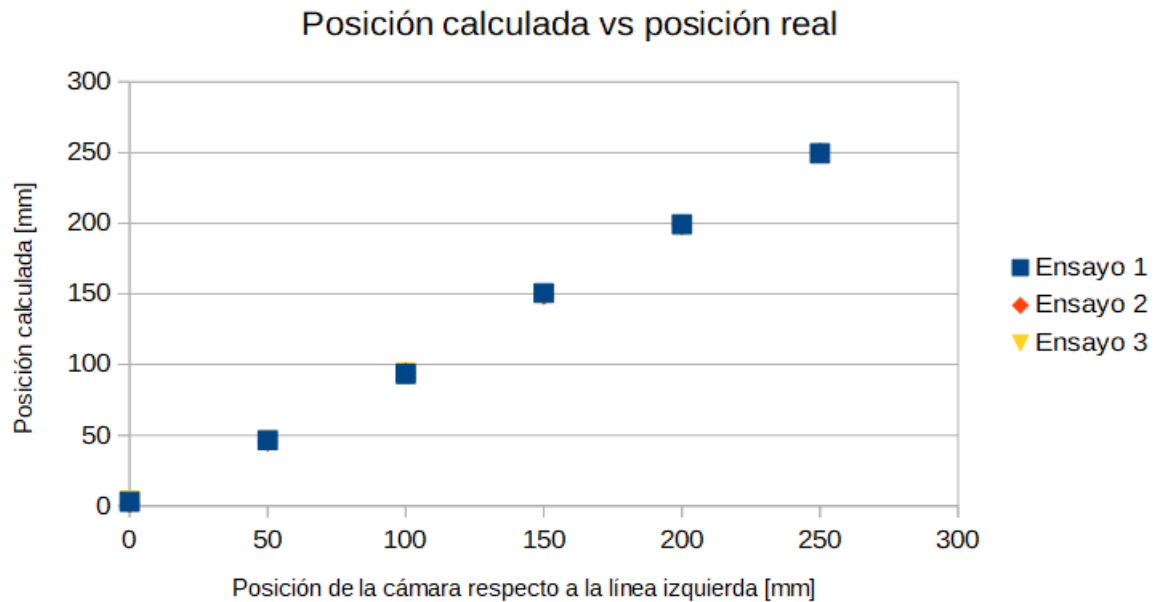


Figure 4.3: Posición calculada de la cámara frente a la posición real empleando el método del alumno.

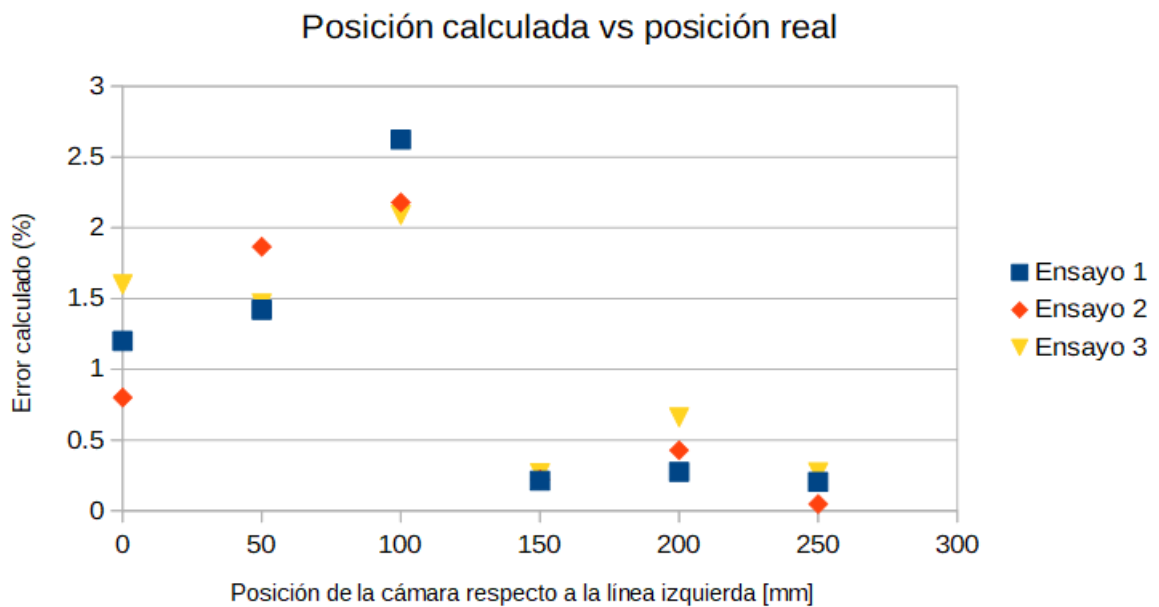


Figure 4.4: Error de la posición calculada del vehículo comparada con la posición real.

En la Figura 4.3, se observa una línea recta que se traduce en una correspondencia prácticamente ideal entre los valores calculados frente a los valores reales. La dispersión de los resultados del ensayo es aún menor que cuando se usa la transformada de Hough. En la Figura 4.4, se observa que la evolución del error es totalmente opuesta a la obtenida mediante la transformada de Hough, debido a que, los extremos del carril se detectan sin problemas, y cuanto más cerca esté el vehículo de la línea, menor será el error. En las posiciones centrales de la pista la referencia se aleja y el error aumenta .

En conclusión, para pruebas con el vehículo parado con un suelo liso, se ha comprobado que funciona mejor el algoritmo desarrollado por el alumno, ya que los parámetros de las líneas empleando la transformada de Hough no se mantienen constantes y los valores de posición del coche oscilan unos centímetros.

Otras fuentes de error pueden venir dadas por la variación de la luz, que puede alterar el funcionamiento del algoritmo de detección de bordes, la ondulación del papel donde se ha hecho la pista, el error humano de colocación de la cámara en el centro del vehículo, y su ángulo de visión, que esta ligeramente movido a la derecha.

4.2 Cálculo del error de posición del coche en movimiento en función de la distancia de la referencia de la pista

En este experimento se pretende determinar la distancia óptima a la que debe estar situada el centro de la pista para obtener un proceso lo más estable posible. Del experimento anterior se ha concluido que el algoritmo óptimo para la pista de pruebas es el desarrollado por el alumno, así que de ahora en adelante se va a prescindir de la transformada de Hough. La pista de pruebas va a ser una trayectoria curva de 4 metros de largo con un ancho de carril de 255 mm.

La distancia de visión del coche se cambia variando la inclinación de la cámara del vehículo. Se van a probar 3 ángulos de inclinación distintos (ver tabla 4.1 y Figura 4.5):

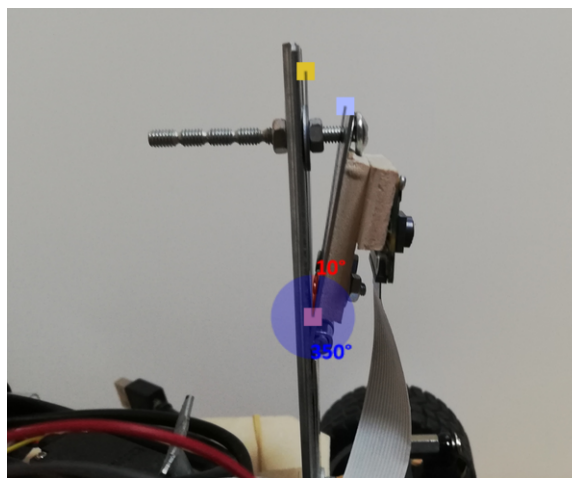


Figure 4.5: Cámara con una inclinación de 10°.

Ángulo de inclinación	Distancia
10°	28
15°	23,5
20°	21,5

Tabla 4.1: Equivalencia ángulo de inclinación de la cámara - Distancia

En las Figuras 4.6, 4.7 y 4.8, aparece representado el error de posición del vehículo respecto al centro de la pista en función del tiempo, para un intervalo de tiempo de aproximadamente 9 segundos, ya que el tiempo que ha tardado el vehículo en recorrer la pista de prueba varía dependiendo de las condiciones de cada ensayo.



Figure 4.6: Error normalizado para una distancia de 28 cm.

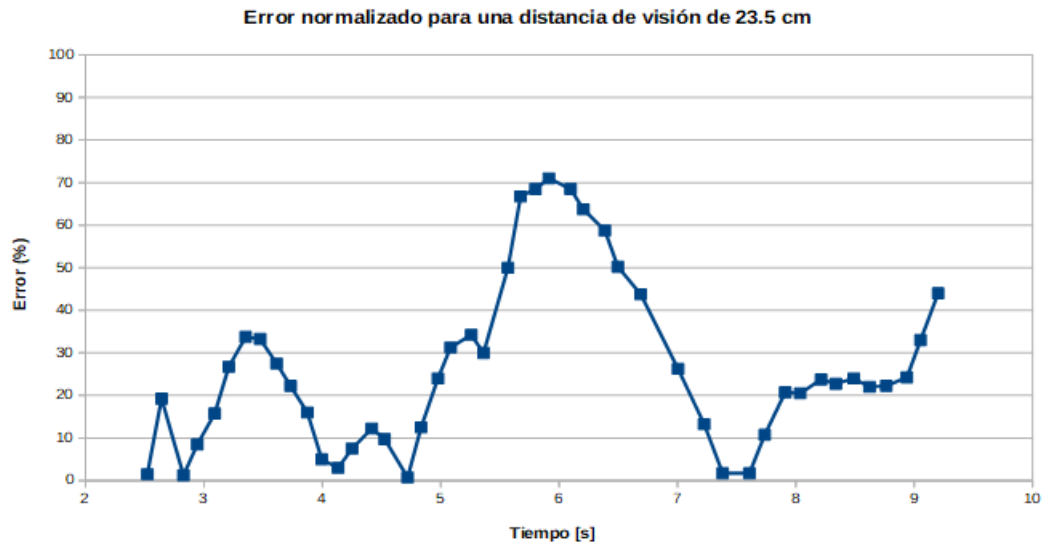


Figure 4.7: Error normalizado para una distancia de 23.5 cm.

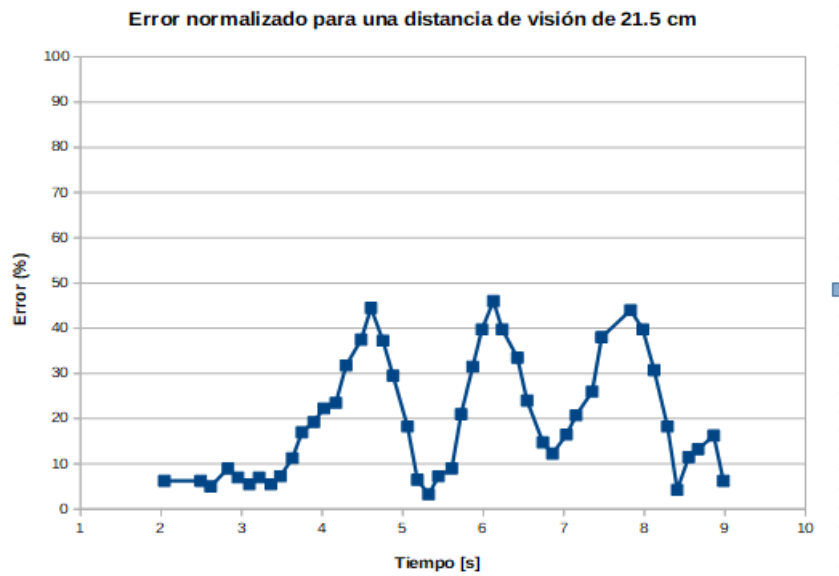


Figure 4.8: Error normalizado para una distancia de 21.5 cm.

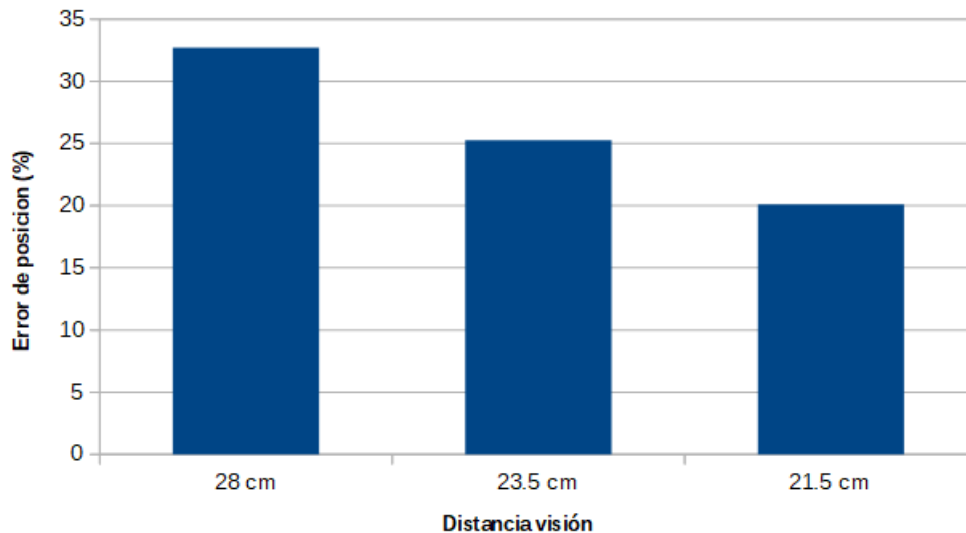


Figure 4.9: Comparativa de errores en función de la distancia.

En los tres casos el error comienza siendo prácticamente nulo, pero pronto el vehículo comienza a oscilar, aumentando el error progresivamente, hasta llegar al máximo, cerca de los 6 segundos, lo que coincide con el punto de máxima curvatura. A partir de entonces el error disminuye progresivamente conforme se entra de nuevo en una recta. Se observa que el error se va atenuando conforme se reduce la distancia de la referencia de la pista con respecto al coche, siendo el valor óptimo 21,5 cm (20° de inclinación de la cámara). Finalmente se probó con un último ángulo de inclinación de 25° pero se observó que el vehículo perdía a los pocos segundos la referencia de las dos líneas, saliéndose de la pista y dando valores incongruentes de error.

Se observa también que el tiempo que se tarda en recorrer la pista aumenta proporcionalmente con el error, ya que las oscilaciones que realiza el vehículo son mucho más amplias.

Al cabo de un tiempo realizando los experimentos se observa que los resultados van empeorando debido al agotamiento de las baterías que ralentizan el movimiento del servo.

Chapter 5

Conclusiones y líneas futuras

5.1 Conclusiones

El TFG ha conseguido el objetivo principal, la detección de caminos y señales de tráfico y finalmente la implementación en un robot guiado por el camino. El mayor reto lo ha planteado la detección de líneas, ya que la presencia de reflejos en el suelo ha propiciado que se tomasen soluciones mucho más simples para el control del vehículo no pudiéndose implementar por ejemplo el algoritmo “pure pursuit”. Al final la solución ha pasado por diseñar una pista sobre una superficie de papel, lo que ha obligado a realizar un trazado menos completo de lo que se pretendía en un principio. El control de la dirección del vehículo se ha presentado como un proceso no lineal e inestable, pero para la velocidad que se ha manejado en el vehículo, el controlador P ha cumplido la función deseada. Se ha comprobado la respuesta más brusca del controlador P empleando al inicio, lo que se ha resuelto estableciendo un umbral mínimo a partir del cual comienza a funcionar el controlador. La detección de señales de tráfico ha funcionado sin errores para las distancias manejadas, y en un entorno con buenas condiciones de iluminación.

El montaje ha planteado también un reto debido a la falta de espacio disponible en el coche, y al cableado necesario. Para hacer más cómoda la alimentación del prototipo y la carga de las baterías se han empleado una serie de conectores macho-hembra fácilmente desmontables.

El prototipo cumple además con el objetivo inicial de tener un bajo coste, exceptuando el motor de corriente continua. No obstante no se ha necesitado toda la potencia disponible, por lo que puede ser sustituido por otro de menor precio. El coche se puede redimensionar en tamaño, ya que el sistema de control es escalable, y debido a su estructura modular se pueden ir añadiendo nuevos componentes o sustituir los ya existentes de una manera rápida.

5.2 Líneas futuras

Dado que los problemas más graves en el desarrollo del proyecto han estado centrados en la parte de la visión artificial, las futuras líneas de investigación deben ir encaminadas en mejorar la obtención y el tratamiento de imágenes.

Se propone un análisis más detallado sobre las diferentes técnicas existentes para la eliminación de ruido en las imágenes, y más concretamente en la eliminación de

reflejos, haciendo uso por ejemplo de un filtrado homórfico, como el propuesto para su posterior implementación software. También sería recomendable una mejora en el algoritmo detección de líneas para descartar las líneas producidas por otros elementos tales como las paredes o sombras, posibilitando así la navegación en entornos más variados.

Para mejorar el sistema de reconocimiento de señales de tráfico se podría crear una base de datos con las imágenes plantilla de todas la señales, y desarrollar un algoritmo para identificarlas mediante correlación cruzada.

Una vez mejorada la imagen, se podría mejorar algoritmo para el cálculo de la trayectoria, haciendo uso por ejemplo de la geometría de dirección de Ackerman, del algoritmo “pure pursuit”, y de otros elementos como una unidad de medición inercial o un sensor de guiñada para un mayor control del vehículo, especialmente en los giros. Estos algoritmos podrían ser probados en un simulador donde se podría modelar el vehículo al completo. Se podrían añadir también más sensores al coche, por ejemplo sensores de ultrasonidos en la parte trasera del coche para poder monitorizar el entorno en 360 grados. Se podría dotar al vehículo de un modo de conducción manual, para ello habría que disponer de una interfaz para conducirlo, lo que se podría conseguir mediante una aplicación móvil conectada por Bluetooth con la Raspberry. También se podría ampliar la gama de señales a detectar mediante otros algoritmos como la correlación cruzada o usando redes neuronales para el reconocimiento de patrones.

La ”Raspberry Pi 3, a pesar de haber cumplido sus funciones para el desarrollo del prototipo de bajo coste, tiene ciertas limitaciones, por lo que si se quiere sacar este prototipo al mercado, sería recomendable escoger otro microcontrolador para obtener una mejora en la calidad del sistema.

Bibliography

- Adafruit (2019). Adafruit 16 channel servo driver with raspberry pi. <https://cdn-learn.adafruit.com/downloads/pdf/adafruit-16-channel-servo-driver-with-raspberry-pi.pdf>. [Online; accessed 1 de Mayo de 2019].
- Bimbraw, K. (2015). Autonomous cars: Past, present and future - a review of the developments in the last century, the present scenario and the expected future of autonomous vehicle technology. *ICINCO 2015 - 12th International Conference on Informatics in Control, Automation and Robotics, Proceedings*, 1:191–198.
- Buehler M, Iagnemma K, S. S. (2009). *The DARPA urban challenge. Vehicles in city traffic*.
- Davies, A. (2017). Waymo has taken humans out of its self driving cars. <https://www.wired.com/story/waymo-google-arizona-phoenix-driverless-self-driving-cars/>. [Online; accessed 4 de Abril de 2019].
- Dickmanns, E. (1997). Vehicles capable of dynamic vision. pages 1577–1592.
- International, K. (2018). Autonomous Vehicles Readiness Index. <https://assets.kpmg/content/dam/kpmg/xx/pdf/2019/02/2019-autonomous-vehicles-readiness-index.pdf>. [Online; accessed 25 de Abril de 2019].
- Joseph, L. (2017). *ROS Robotics Project*. Packt Publishing Ltd.
- Litman, T. (2017). Autonomous Vehicle Implementation Predictions. Implications for Transport Planning. https://orfe.princeton.edu/~alaink/SmartDrivingCars/PDFs/VictoriaTransportAV_Predictionsavip.pdf. [Online; accessed 25 de Abril de 2019].
- NHTSA (2013). Policy on Automated Vehicles.
- OpenCV (2012). Hough Line Transform. https://docs.opencv.org/2.4/doc/tutorials/imgproc/imgtrans/hough_lines/hough_lines.html. [Online; accessed 12 de Marzo de 2019].
- Orti, E., Bredas, J., and Clarisse, C. (1995). *Inteligencia Artificial: Un Enfoque Moderno*. Prentice Hall.

Rosecbrock, A. (2014). Opencv and python k-means color clustering. <https://www.pyimagesearch.com/2014/05/26/opencv-python-k-means-color-clustering/>. [Online; accessed 1 de Mayo de 2019].

Rosecbrock, A. (2015). How to install opencv 3 on raspbian jessie.

SAE (2014). Taxonomy and Definitions for Terms Related to On-Road Motor Vehicle Automated Driving Systems. https://www.sae.org/standards/content/j3016_201401/preview/. [Online; accessed 4 de Abril de 2019].

SAE (2016). Taxonomy and Definitions for Terms Related to On-Road Motor Vehicle Automated Driving Systems. https://www.sae.org/standards/content/j3016_201601/preview/. [Online; accessed 4 de Abril de 2019].

Anexos

Anexos A

Aspectos éticos, económicos, sociales y ambientales

Se estima que el desarrollo de la automatización eliminará 75 millones de empleos para el año 2025, pero además esta misma tecnología tiene previsiones de generar más de 133 millones de nuevos roles de trabajo, con lo que se generarán 58 millones de nuevos empleos, que requerirán un mayor grado de especialización.

En el caso particular de la conducción autónoma, esta tecnología podría sustituir en un futuro a los conductores de autobuses, taxis o trenes, pero el mantenimiento y la mejora de estos vehículos, además de aumentar la oferta de transporte facilitando el movimiento de personas, requerirá de técnicos y especialistas que mantengan dichos vehículos.

Dejando a un lado el transporte de pasajeros, los robots autónomos de pequeñas dimensiones están proliferando rápidamente debido a la gran variedad de aplicaciones industriales que tienen, por ejemplo en la gestión de almacenes. En los hospitales estos robots también tienen un gran potencial, para el transporte de medicamentos y otros materiales delicados. El prototipo desarrollado tiene la potencia de motor y su batería tiene autonomía suficiente como para ser aplicado en alguno de estos proyectos.

La investigación y desarrollo en el sector de la automoción y de la robótica es fundamental el desarrollo de modelos a escala. por lo que, utilizando como base este robot, se pueden abrir nuevas líneas de investigación en la tecnología de conducción autónoma.

Es evidente pues que la automatización y concretamente la conducción autónoma es un campo con un gran potencial en diferentes áreas, como puede ser la logística, el sector servicios, o la investigación académica, y ofrece además grandes oportunidades de negocio. El interés del prototipo desarrollado se encuentra principalmente en que forma parte de una tecnología muy reciente, en la que cualquier línea de investigación puede aportar algo nuevo, y cuya industria asociada tiene un enorme potencial de crecimiento, por lo que es un buen momento para posicionarse dentro de la misma.

Anexos B

Presupuesto económico

En este apartado se realiza una estimación de los costes asociados al proyecto.

Costes de personal

Descripción	Salario anual (€)	Dedicación (horas)	Total (€)
Ingeniero recién graduado	22.000	400	5.985,77
			5.985,77

Costes del Hardware del robot

Descripción	Cantidad	Precio Ud. (€)	Total(€)
Raspberry Pi 3 Model B	1	36,4	36,4
Módulo L298N	1	2,09	2,09
Chasis	1	52,98	52,98
Raspberry Pi Camera Module v2.	1	23,98	23,98
Motor DC A-max 32	1	160	160
Servomotor Futaba s3001	1	12,95	12,95
Módulo PCA9685	1	2,66	2,66
Sensor de ultrasonidos HC-SR04	1	0,62	0,62
Regulador de tensión 7805	1	0,45	0,45
Condensador	2	0,2	0,4
Resistencia	2	0,18	0,36
Protoboard	1	2	2
Pack de 120 cables de Arduino	1	6,39	6,39
Conector de 2 Vías macho-hembra Faston	4	1,42	5,68
Batería USB - 2200 mAh.	1	12	12
Pack de 10 baterías SubC 1.2 Voltios 3,8 Amperio	1	49,85	49,85
			368,81

Costes del circuito de prueba

Descripción	Cantidad	Precio Ud. (€)	Total(€)
Rollo de papel kraft (50 m x 1 m)	1	13	13
Rotulador permanente – Color Negro	1	2,5	2,5
			15,5

Costes de equipo y amortizaciones

El coeficiente lineal máximo de amortización para equipos destinados a procesos de información es de un 25% anual.

Descripción	Cantidad	Precio Ud. (€)	Uso (meses)	Amortización (años)	Total (€)
Ordenador personal	1	1650	7	5	240,625
					240,625

Coste total

SUBTOTAL (€)	6.610,71
IVA	21,00 %
PRESUPUESTO TOTAL (€)	7.998,95