UNIVERSIDAD DE VALLADOLID

ESCUELA TÉCNICA SUPERIOR DE INGENIEROS DE TELECOMUNICACIÓN

TRABAJO FIN DE MÁSTER

MÁSTER EN INGENIERÍA DE TELECOMUNICACIÓN

Servicios de vehículo conectado y conducción autónoma en un Twizy

Autor:

D. Samuel Pilar Arnanz

Tutor:

Dr. D. Juan Carlos Aguado Manzano

D. Daniel Castaño Navarro

VALLADOLID, SEPTIEMBRE 2021

TRABAJO FIN DE GRADO

TÍTULO:	**Servicios de vehículo conectado y conducción autónoma en un Twizy**
AUTOR:	**D. Samuel Pilar Arnanz**
TUTOR:	**Dr. D. Juan Carlos Aguado Manzano,**
	D. Daniel Castaño Navarro
DEPARTAMENTO:	**Teoría de la Señal y Comunicaciones e Ingeniería Telemática**

TRIBUNAL

PRESIDENTE:	**Dr. D. Ignacio de Miguel Jiménez**
VOCAL:	**Dr. D. Ramón José Durán Barroso**
SECRETARIO:	**Dr. D. Ramón de la Rosa Steinz**
SUPLENTE:	**Dr. Dª Patricia Fernández Reguero**
SUPLENTE:	**Dr. Dª María Jesús Verdú Pérez**
SUPLENTE:	**Dr. D. Javier Aguiar Pérez**

FECHA: 03 de septiembre de 2021

CALIFICACIÓN:

Resumen del Trabajo Fin de Master

El propósito de este trabajo fin de máster es conseguir conducción autónoma en un modelo Renault Twizy en un entorno controlado con tecnologías baratas y añadirle una nueva función de vehículo conectado, como es FOTA (Firmware Over The Air). Por un lado, el dotar de autonomía al vehículo supone la implementación tanto a nivel hardware como software de nuevos elementos que le proporcionen percepción del entorno que le rodea. El detectar objetos no solo servirá para evitar colisiones sino también para poder seguir la banda magnética que describirá la ruta que debe seguir el coche. En el TFM se ha creado un simulador de la conducción autónoma del Twizy, antes de comenzar las pruebas de test, para analizar el resultado del seguimiento de la banda dependiendo de los parámetros configurables de los algoritmos de control empleados. Por otro lado, se ha implementado un sistema FOTA donde los usuarios serán capaces de entablar conexión remota con las ECUs (Electronic Control Unit) del vehículo, donde se podrá tanto leer como escribir parámetros. Además, se ha creado una página web donde se podrá realizar todas las operaciones relacionadas con este sistema.

Abstract

The purpose of this Master's Final Project is to achieve autonomous driving in a Renault Twizy and add a new connected vehicle function, such as FOTA (Firmware Over The Air). On the one hand, providing autonomy to the car means the implementation of hardware and software of new elements in order to provide the car the ability of perception of the environment that surrounds it. Detecting objects serve not only to avoid collisions but also to be able to follow the magnetic strip that will describe the route that the car must follow. A simulator of the autonomous driving of the Twizy will be created in this project, before starting the tests, in order to analyse the result of the tracking of the band depending on the configurable parameters of the control algorithms used. On the other hand, a FOTA system will be implemented where users will be able to establish a remote connection with the vehicle's ECUs (Electronic Control Unit), where it will be possible to read and write parameters. Furthermore, a web page is created which aims to carry out all operations related to this system.

Palabras clave

Autónomo, FOTA, módulo de comunicaciones, módulo de control, motor, multiprocesos, parking, sensores, simulador MATLAB, vehículo conectado, web, Twizy.

Agradecimientos

Primero, me gustaría agradecer a Juan Carlos Aguado Manzano, tutor de este trabajo fin de máster, por no solo haberme dado la oportunidad de participar en el TwizyContest2020 aquel octubre de 2019, sino también por seguir aportando su tiempo y ayuda en seguir adelante con el proyecto TwizyLine que creamos.

También son palabras de agradecimiento al tutor de Renault, Daniel Castaño Navarro, siempre sacando tiempo para poder echarme una mano para seguir avanzando dentro del proyecto y continuar aprendiendo del emocionante mundo de la automoción.

Quería hacer también mención de mis compañeros de equipo que han seguido realizando otra serie de tareas, tan importantes como los avances tecnológicos, para seguir dando a conocer el proyecto a distintas instituciones y seguir creciendo juntos, haciendo cada vez más grande el proyecto TwizyLine.

Por último, tanto la carrera como el máster no lo podría haber hecho sin el gran apoyo que supone para mí, mis padres y mi hermana, a los cuales dedico todos los triunfos conseguidos en mi vida. Una pena no poder compartir este momento con mis abuelos, que descansan en paz, si ahora me preguntaran con una sonrisa orgullosos de su nieto: "Pero hijo, ¿cuándo vas a dejar de estudiar? No paras", contestaría "¡Por fin!".

~La familia, sagrada como biblia~ José Álvaro Osorio Balvín

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1
Introducción

1.1 Contextualización y motivación del trabajo de fin de máster

En 1860 el ingeniero francés Étienne Lenoir construyó el primer automóvil que funcionaba gracias a un motor de combustión interna con gasolina [1]. Esto supuso un punto de inflexión dentro de la historia del automóvil y comenzó a competir por aquel entonces con el coche a vapor. Los coches a vapor eran más simples mecánicamente, la energía que producían era continua gracias a la presión del vapor, no tenían embrague o engranajes como un motor de combustión, pero también tenían desventajas como su gran peso debido a la gran caldera y el tanque de agua que perdía 3 litros por kilómetro como promedio o el tiempo que tardaba en calentarse antes de su uso [2].

En un principio el coche de combustión interna también lidió con el problema del arranque, pero la invención del encendido eléctrico supuso un gran avance y una gran ventaja frente a su competidor. Sin embargo, la clave del éxito del vehículo de gasolina en el primer cuarto del siglo XX fue su coste, ya que llegaba a ser incluso 6 veces más barato de uno propulsado a vapor. Esto dio lugar a dejar en el olvido el coche propulsado con vapor que dejó de ser usado conforme pasaban los años y el vehículo de combustión mejoraba las prestaciones [2].

El automóvil siguió evolucionado considerablemente hasta el día de hoy, cambiando la materia prima que necesita para funcionar, mejorando la tecnología que implementa, incrementando las dimensiones del vehículo… y sobre todo aumentando el confort y la seguridad.

Respecto a la evolución de los sistemas de seguridad, fue a partir de los años 60 cuando se comenzaron a introducir cambios importantes en los vehículos. Así, por ejemplo, en los años 60, se empezó a incorporar en los vehículos un chasis formado por un espacio de alta rigidez, rodeado de partes con menos rigidez más propensas a deformación en caso de colisión. Pese a que Bosh patentó en 1936 el sistema antibloqueo de ruedas (ABS, *Antiblockierensystem*), hasta 1970 no se comenzó a distribuir e implementar en los automóviles; este sistema, como su propio nombre indica, evitará que las ruedas se bloqueen cuando se está frenando al máximo posible. El ABS es obligatorio en la Unión Europea desde 2004 [3].

En 1983, el cinturón de seguridad se convirtió en obligatorio en España en todas las plazas delanteras y a partir de 1992, en las plazas traseras. Este simple pero efectivo sistema se estima que salva 100.000 muertes al año [4]; aun así, una encuesta sobre las actitudes de los usuarios en la vía recabó que todavía alrededor de un 20% de los pasajeros de un vehículo no lleva puesto el cinturón [5]. En la Tabla 1 se muestra la efectividad de llevar bien puesto el cinturón de seguridad a la hora de reducir la probabilidad de sufrir una lesión en caso de impacto. Se puede observar como en todas las situaciones reduce en más de un 20% la probabilidad de sufrir alguna lesión física.

Gravedad de la lesión	Tipo de colisión	Porcentaje de reducción	Intervalo de confianza (95%)
Conductores de vehículos ligeros (turismos y furgonetas)			
Muerte	Todas las colisiones	-50%	(-55%,-45%)
Lesiones graves	Todas las colisiones	-45%	(-50%,-40%)
Lesiones leves	Todas las colisiones	-25%	(-30%,-20%)
Cualquier lesión	Todas las colisiones	-28%	(-33%,-23%)
Pasajeros 1ª fila de vehículos ligeros (turismos y furgonetas)			
Muerte	Todas las colisiones	-45%	(-55%,-35%)
Lesiones graves	Todas las colisiones	-45%	(-60%,-30%)
Lesiones leves	Todas las colisiones	-20%	(-25%,-15%)
Cualquier lesión	Todas las colisiones	-23%	(-29%,-17%)
Pasajeros 2ª fila de vehículos ligeros (turismos y furgonetas)			
Muerte	Todas las colisiones	-25%	(-35%,-15%)
Lesiones graves	Todas las colisiones	-25%	(-40%,-10%)
Lesiones leves	Todas las colisiones	-20%	(-35%,-5%)
Cualquier lesión	Todas las colisiones	-21%	(-36%,-6%)

Tabla 1. Reducción en la probabilidad de lesión en caso de usar cinturón de seguridad [4].

El airbag frontal para el conductor se convirtió en obligatorio a partir de 1990, y hasta el año 2006 no lo fue para el copiloto. Para modelos de 2009 y posteriores, la Unión Europea obligó a los fabricantes de vehículos instalar el sistema de ayuda de frenado (BAS, *break assist system*), cuyo sistema incrementa la presión en el circuito de frenos cuando detecta que el conductor está realizando un frenado de emergencia, pero no está aplicando la suficiente fuerza al pedal del freno. Respecto a la mejora en el transporte seguro de bebés en los vehículos, se introdujo el sistema ISOFIX, inventado por la organización internacional de normalización (ISO), que tiene como objetivo la correcta sujeción de la silla de seguridad para niños siendo obligatorio en la Unión Europea desde 2011 [3].

Además de estos sistemas ya descritos, se han incorporado otro tipo de sistemas que son obligatorios para coches posteriores a 2010 como: ESP (sistema de control de estabilidad), AFS (alumbrado adaptativo), TPMS (sensor de presión de neumáticos), LKS (aviso de abandono de carril), DRL (luces de conducción diurna), eCALL (llamada de emergencia), SBR (sistema que avisa de que los cinturones donde haya pasajeros estén abrochados) [3].

Figura 1. Sistemas de seguridad obligatorios en la Unión Europea a partir de 2010.

A estos sistemas se les suma otro tipo de sistemas que pese a no ser obligatorios en la Unión Europea también se implementan para ofrecer mayor seguridad al conductor como son: DDM, sistema de vigilancia del conductor, mediante sensores o cámaras se analiza el nivel de concentración del conductor y se le avisa si se detecta fatiga; EBR, frenada automática de emergencia, si se detecta una colisión inminente se activa automáticamente una frenada de emergencia [3]; ISA, adaptación inteligente de velocidad, gracias a la conexión vehículo a infraestructura (V2X), la velocidad podría ser limitada mediante el cálculo de su velocidad por su posición GPS o el reconocimiento de señales inteligentes que comuniquen al coche la velocidad máxima permitida [6]. Cabe destacar también, que algunos de estos sistemas pueden ser desactivados manualmente, prevaleciendo siempre la voluntad del conductor sobre lo que indican estos sistemas, pudiendo evadir su funcionamiento.

Figura 2. Primer coche de pruebas del proyecto EUREKA [7].

Todo ese conjunto de sistemas que se han descrito trata de mejorar la seguridad mediante la automatización de algunos procedimientos involucrados en la conducción; por otro lado, también se ha estado desarrollando en otra línea de investigación la conducción completamente autónoma.

Ya en 1939 se consiguió la primera conducción autónoma de un vehículo, en este caso eléctrico, gracias al diseñador industrial Norman Bel Geddes junto su equipo hicieron que un coche sin conductor siguiera un circuito eléctrico que se localizaba en el pavimento de la carretera [8]. El coche autónomo de Norman Bel Geddes se presentó en la feria de muestras Futurama en Nueva York, donde se presentaban cómo sería la vida en un plazo de 30 años.

A partir de la década de los 80, se observan más avances dentro de este campo. En 1980 se inició el proyecto europeo EUREKA, cuyo objetivo era el crear un vehículo autónomo seguro que fuera capaz de recorrer largas distancias [9]. Hoy en día se puede encontrar empresas como Waymo, proyecto financiado por Google, cuyo objetivo era la creación de vehículos completamente autónoma. A partir de 2017 ofrece este servicio al mercado [10].

Figura 3. Coche autónomo de la compañía Waymo [11].

Norman Bel Geddes no sólo acertó con la idea del vehículo sin necesidad de un conductor, sino que también acertó con la materia prima necesaria para arrancar el coche, la electricidad. En los últimos años la situación provocada por el calentamiento global causada por emisión de gases contaminantes ha hecho que instituciones y organizaciones medioambientales aboguen por el uso del coche eléctrico, suponiendo un nuevo reto en la constante evolución del sector de la automoción.

Actualmente, la mayoría de países europeos tienen un plan de subvenciones para facilitar la compra de este tipo de vehículos [12]. En España el gobierno ha puesto en marcha el plan Moves 2020, cuyas ayudas pueden ascender hasta un descuento de 6.500 € para la compra de coches puramente eléctricos [13]. Pese a este tipo de ayudas, y a la mayor inversión en vehículos eléctricos por parte de las compañías fabricantes de automóviles, su acogida por la población está siendo lenta. Solo en los países nórdicos como Noruega, Finlandia o Suecia, los coches eléctricos empiezan a suponer un porcentaje significativo, superior al 10% de venta de coches total en 2018 [14]. Los datos obtenidos durante 2020 y 2021 hacen suponer una tendencia al alza de venta de estos vehículos. De hecho, pese a la reciente expansión mundial del coronavirus COVID-19 que ha supuesto un decremento en la venta global de vehículos de todo tipo, las ventas de vehículos eléctricos en todo el mundo se dispararon un 43% en 2020 (Figura 4) [15]. Otro ejemplo de esta tendencia al alza es España, donde según la consultora MSI Iberia se estima que en 2021 se matricularán un 18% más de coches eléctricos [16].

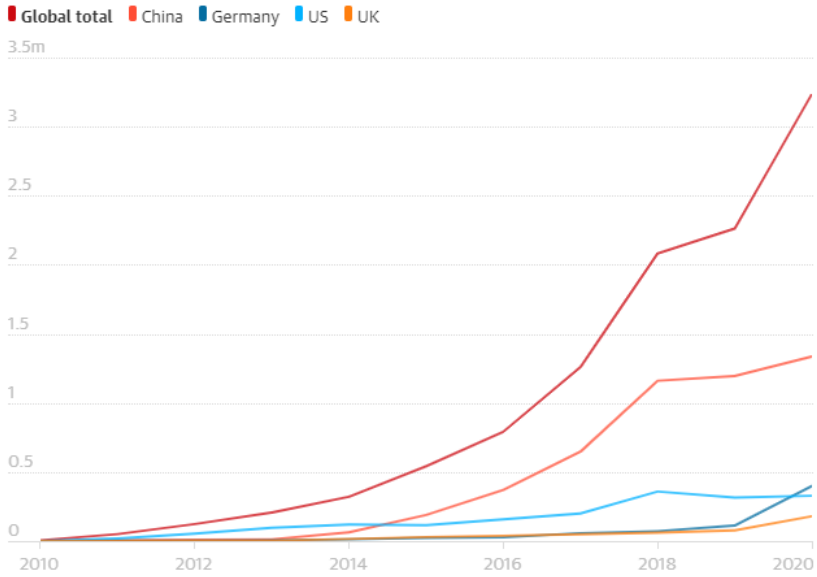


Figura 4. Gráfica del crecimiento de ventas del coche eléctrico [15].

En este contexto de concienciación por parte de instituciones y fabricantes de coches por un cambio en la movilidad, alentando a la población a usar coches eléctricos, junto con los continuos avances en la tecnología que se pueden implementar en un vehículo nace TwizyLine.

TwizyLine fue ideado por 4 egresados de la Universidad de Valladolid: Adrián Mazaira Hernández, Ignacio Royuela González, Mario Martín Fernández y Samuel Pilar Arnanz [17]. Este proyecto proponía un aparcamiento para coches autónomos donde los coches con la tecnología apropiada no solo serían capaces de aparcarse solos, sino también de cargarse solos [18].

Fue galardonado con varios premios como el primer premio a nivel nacional en el concurso organizado por Groupe Renault y Segula Technologies [19], consiguiendo posteriormente el segundo premio a nivel internacional de ese mismo concurso [20]; ganador de los premios Prometeo [21] y también ganador del concurso Campus Emprendedor [22].

Sin embargo, el proyecto TwizyLine no fue implementado al completo, aunque se consiguieron grandes avances que certificaban la posibilidad su desarrollo en un futuro. Las partes ya implementadas y documentadas en los trabajos fin de grado de los miembros del equipo fueron:

- Diseño y funcionamiento del parking [18].
- Desarrollo software de una aplicación para TwizyLine, en la cual los clientes podían solicitar un coche de los parkings y los administradores podían realizar operaciones de gestión [23].
- Implementación de dos módulos, similares a dos ECUs (unidad de control electrónica, Electronic Control Unit), uno de comunicaciones y otro de control. Además, se desarrolló un simulador para comprobar el funcionamiento de los módulos dentro del coche [24].
- Creación de un servidor capaz de controlar todos los elementos de todos los parkings, de dar soporte a la aplicación web, de ordenar aparcarse a los coches y almacenar la información que fuera recopilando en una base de datos [18].
- Se realizó un estudio de un plan de negocio adecuado al proyecto y se analizó su diseminación en los medios de comunicación [25].

Como se puede ver, gran parte del proyecto está diseñado y un parte importante implementado en los trabajos fin de grado de los integrantes del equipo, sin embargo, queda un trabajo importante de implementación y modificación de un vehículo. Esto será posible gracias a la donación de un Renault Twizy donado por la fundación Renault a la Universidad de Valladolid y a la colaboración del Grupo Renault con su asesoramiento sobre cómo hacer las diferentes modificaciones del vehículo.

Este trabajo fin de máster tiene como principal tarea realizar las modificaciones mecánicas y hardware necesarias al coche y conseguir que funcione de acuerdo al sistema TwizyLine. Por lo tanto, esto contempla adentrarnos en el campo de la conducción autónoma para dotar de esta capacidad al Twizy. Dicho campo es particularmente desafiante, ya que requiere un conjunto de test y evaluaciones antes de poder ser desplegado de forma generalizada en la sociedad [26]. No solo se requieren cambios a nivel técnico, sino que incluso afecta a las normas y leyes internacionales [25]. Está claro que antes de proceder a introducir de forma generalizada este tipo de vehículos es necesario desplegarlos en entornos controlados que permita a los usuarios ganar confianza en este tipo de tecnología. Debido a esa razón se propone un parking, siendo un entorno controlado donde los vehículos puedan circular de forma autónoma.

La tarea a llevar a cabo en este trabajo no se quedará ahí, sino que también ahondaremos dentro del campo del vehículo conectado. Los fabricantes quieren tener un control estricto de lo que está ocurriendo en los vehículos autónomos para solventar sobre la marcha cualquier tipo de problema que surja. Este tipo de servicio se considera crítico, dado que tiene un impacto muy alto en el prestigio y la confianza de los clientes en la marca y en la nueva tecnología. Los servicios que se pueden desplegar con el concepto de vehículo conectado van desde el denominado infotainment (información más entretenimiento) hasta cuestiones más serias como es la configuración y modificación de los parámetros de funcionamiento del vehículo.

Este último servicio está recibiendo recientemente bastante atención por parte de los fabricantes por los diferentes tipos de beneficios que pueden aportar. Por ejemplo, puede reducir el tiempo de lanzamiento de nuevos vehículos, al poder solapar diferentes fases de desarrollo que antes no era posible. Permitiría evaluar la situación del vehículo y comprobar si necesita algún tipo de arreglo o permitiría modificar algoritmos esenciales de control del vehículo [27]. Dada la importancia que tendrán este tipo se servicio en el futuro, en este proyecto de investigación se propone abordar también la temática del vehículo conectado. El desarrollo previo en el que se

basa esta propuesta, convierte en cierta medida al Twizy en un vehículo conectado, lo cual nos permitirá analizar y desarrollar algún caso de uso sobre vehículos conectados.

1.2 Objetivos

El objetivo general del proyecto de investigación es doble. Por un lado, se desea realizar las modificaciones en el vehículo donado por el Grupo Renault para se puedan probar funciones de vehículo autónomo y a largo plazo implementar el servicio definido por equipo TwizyLine. Por otro lado, abordar algunos de los problemas actuales a los que se enfrenta la industria de la automoción. Particularmente, en el proyecto estaremos interesados por un lado en analizar soluciones de comunicación de vehículo-proveedor, para poder dar aplicaciones de alto valor añadido. Dentro de estas aplicaciones cabe destacar por ejemplo el FOTA, necesario para poder realizar cambios en el software del Twizy de manera remota.

A continuación, se muestran los objetivos concretos que se deben conseguir para poder lograr los objetivos generales:

- Integración del módulo de comunicaciones en el Twizy. Con ello se tratará de proporcionar conectividad a internet mediante un modem 4G y así proporcionar al vehículo V2I.
- Integración del módulo de control. Comprobar que es capaz de controlar los distintos sensores a lo que está conectado.
- Integración de sensores. El Twizy estará formado por una serie de sensores que ayuden a la conducción autónomo y su correcto funcionamiento es esencial.
- Conseguir un control longitudinal y transversal del Twizy, mientras está operando en modo autónomo.
- Actualización del software de manera remota. Mientras el coche no esté en funcionamiento se dará paso a poder establecer comunicación con las diferentes ECUs (*Engine Control Unit*), para después poder leer y modificar parámetros de ellas.

1.3 Fases y métodos

Para la realización de este proyecto se hizo un estudio inicialmente sobre cuáles eran los pasos a tomar una vez se obtuvo el Twizy. Se comenzó tomando como punto de partida la implementación que se mostró en el TFG de Ignacio Royuela González [24], donde se puede observar cómo sería la estructura que podía seguir nuestro sistema y la cual se trató de seguir lo más fielmente posible.

Algunas de las fases que se mostrarán en la lista que aparece a continuación, estaban condicionadas a la llegada de nuevo material, por lo que se pretendió seguir avanzando con otras fases. De manera general podemos dividir el proyecto en 3 grandes fases.

Una primera parte, que servirá para montar la base sobre la cual luego se tratará de implementar la conducción autónoma y algún algoritmo de vehículo conectado. Esta parte a su vez se divide en:

1. Estudio de la infraestructura del vehículo Twizy mediante sus esquemas CAD ("*Computer-aided design*", diseño asistido por computadora) proporcionados por la empresa de Renault. Se trató de obtener la siguiente información:

   a. Acceso al sistema de alimentación, ya sea de la batería general situada en el chasis, o de la pila de 12V situada en la parte delantera.
   b. Posible localización de los módulos de control y comunicaciones, además de la controladora del motor.
   c. Posicionamiento óptimo de los sensores para que cumplan con su función.
   d. Posicionamiento adecuado del motor cerca de la columna de dirección, conforma al espacio libre que hay en la infraestructura en esa zona.
2. Integración hardware de los distintos dispositivos electrónicos que se deben incorporar al Twizy.
3. Desarrollo software para comprobar los distintos sensores y descarga e instalación de los programas ya desarrollados en el TFG de Ignacio Royuela González.
4. Testeo de los componentes añadidos a la infraestructura para comprobar su adecuada instalación y su correcto funcionamiento.

Tras realizar la primera parte, nuestro siguiente propósito era conseguir la conducción autónoma, para lo cual se siguieron los siguientes pasos:

5. Estudio y desarrollo de un entorno de simulación en Matlab para poder modelar la conducción autónoma y así optimizar los parámetros introducidos en el programa final.
6. Estudio y desarrollo software sobre la utilización del motor y la controladora del motor proporcionado por Maxon, que se utilizará para poder girar el volante de forma automática.
7. Implementar y afinar el control longitudinal y transversal.
8. Retocar el software de conducción autónoma para que la lógica esté de acuerdo al nuevo sistema de control longitudinal y transversal.
9. Caso de uso de conducción autónoma

En última instancia, tras llegar a este punto, los pasos finales para dar por concluido el proyecto han sido:

10. Implementar software necesario para poder realizar actualizaciones software de manera remota.
11. Caso de uso de FOTA ("*Firmware Over The Air*").

1.4 Inventario y lugar de trabajo

El desarrollo de este proyecto ha sido gracias a la donación del Renault Twizy por parte de la empresa "Grupo Renault España", y a la Escuela Técnica Superior de Ingenieros de Telecomunicación (ETSIT) que cedieron un espacio para poder tener aparcado y trabajar con el coche.

Por lo tanto, la mayoría del material va destinado a integrarlo en el coche. La lista de productos empleados:

- Renault Twizy.
- 2 Humming Boards, que harán las funciones de los módulos de control y comunicaciones respectivamente.
- Una Raspberry 3 Model B+.
- 2 arduinos nano.
- Sensor magnético A30 ASEL.061 V2 proporcionado por Astii Mobile Robotics.

- 3 sensores de ultrasonidos de proximidad.
- 1 hub USB de 4 puertos.
- 1 receptor GPS, Garmin GPS18x USB.
- Maxon EPOS4, controladora del motor.
- Motor Maxon, que incluye motor de 100W con su encoder MILE y la reductora con relación 1:81.
- 1 relé
- Cableado, conectores y crimps.

2
Estado del arte

En el capítulo introductorio se ha visto cómo ha evolucionado el sector del automóvil, tanto en el campo de la seguridad como en el del vehículo conectado a internet y autónomo, mostrando después el principal motivo del desarrollo de este trabajo fin de máster.

La primera parte de este capítulo comienza estudiando en profundidad el estado del arte del vehículo autónomo, después se mostrarán algunas de las empresas que ofrecen en el mercado coches autónomos y seguidamente se verá algunas implementaciones previas para conseguir un Twizy autónomo. La segunda parte se centrará en analizar las distintas tecnologías que operan dentro del campo del vehículo conectado y finalizaremos revisando posibles desarrollos de varias arquitecturas FOTA (*Firmware Over The Air*).

2.1 Estado del arte del vehículo autónomo

La Sociedad de Ingenieros de Automoción (SAE, por sus siglas en inglés) es la organización que se ha encargado de publicar algunos los principales estándares relacionados con los medios de transporte. Entre ellos se encuentran los estándares para conducción autónoma. De entre ellos nos interesa especialmente el estándar que recoge la taxonomía y la definición de términos de la conducción autónoma, el estándar SAE J3016 [28].

En este documento se definen las tareas a realizar con un coche autodirigido, denominadas DDT (“Dynamic Driving Task”). El SAE J3106 define el DDT como aquellas operaciones en tiempo real necesarias para controlar el vehículo en un escenario real, es decir, teniendo en cuenta el tráfico en carretera, otro tipo de medios de transporte como bicicletas y a los peatones. En resumen, las tareas que un vehículo autónomo debe realizar se podrían descomponer en tres procedimientos: planificación, percepción del entorno y control del vehículo. En estas operaciones se excluyen las funciones de programación de ruta y selección de trayectoria que se desea que siga el coche (Figura 5). Las tareas del DDT se pueden descomponer en las siguientes tareas [28]:

- Control del movimiento transversal del coche usando la dirección.
- Control del movimiento longitudinal del vehículo a través de la aceleración y deceleración.
- Monitorizar el entorno de conducción detectando, reconociendo y clasificando objetos o eventos.
- Procesar la respuesta frente la percepción de elementos externos, cuyo procedimiento consistirá en preparación y después ejecución.
- Planificación de maniobras.
- Mejorar la visibilidad del coche haciendo uso de elementos de iluminación, señalización (sonora u otro tipo).

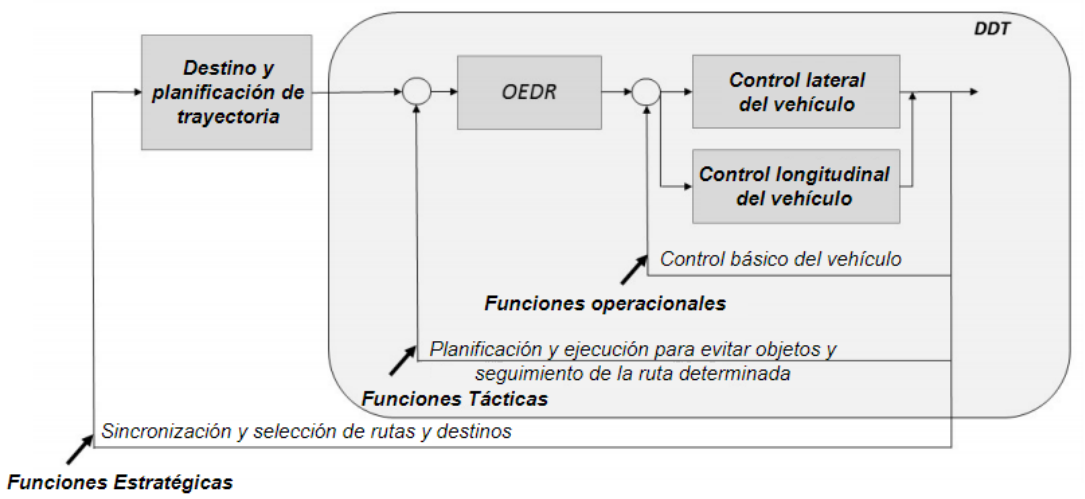


Figura 5. Esquemático de las operaciones de conducción determinando la parte DDT [28].

Todas estas subtareas que conforman el DDT están condicionadas al Dominio de Diseño Operativo (ODD, *Operational Design Domain*). El ODD determina cuales son las condiciones bajo las que va a funcionar el sistema. Se especifica en qué momento del día opera, bajo qué condiciones atmosféricas, sobre qué tipo de carretera, etc. Como se puede apreciar el ODD abarca un amplio rango de variables que será de vital importancia definir para conseguir una conducción autónoma segura [28].

También se debe tener en cuenta que las tareas definidas en el DDT son compartidas tanto por los agentes humanos como si se trata de coches autónomos. Se ha definido una escala de conducción autónoma que se basa en el porcentaje de implicación que tiene la persona que lo conduce y la propia máquina el SAE. Dicha escala diferencia entre 6 niveles de conducción autónoma (Figura 6) [28]:

- Nivel 0 (ninguna automatización): La gestión del DDT es llevada a cabo completamente por el conductor, es decir, es el encargado tanto del control lateral como longitudinal y de controlar la velocidad del vehículo.
- Nivel 1 (asistencia al conductor): Se implementa alguna función de control longitudinal o lateral (pero no ambas simultáneamente) que bajo las condiciones del ODD operaran, dejando el resto de funciones del DDT a cargo del conductor.
- Nivel 2 (automatización parcial de la conducción): Teniendo en cuenta las condiciones impuestas por el ODD, el control longitudinal y lateral podrá ser gestionado automáticamente simultáneamente. El conductor completará la tarea de detección y respuesta frente a eventos u objetos (OEDR, *Object and Event Detection and Response*).
- Nivel 3 (automatización condicionada): En el nivel 3 de autonomía, se consigue que el coche sea capaz de detectar y tomar las acciones consecuentes cuando detecte eventos u objetos (OEDR). El conductor debe estar atento ante posibles fallos o en las situaciones donde las condiciones del ODD no se cumplen y suponga un mal funcionamiento del sistema.
- Nivel 4 (automatización elevada): En este nivel superior de autonomía, aparte de tener las características descritas del nivel 3, el coche es capaz de gestionar situaciones de emergencia, donde las operaciones del DDT han fallado ya que tiene un sistema de respaldo para actúa en esas situaciones. El conductor puede recuperar el control del coche de forma opcional si es necesario y también cuando las condiciones del ODD no se cumplan.

- Nivel 5 (automatización completa): Este es el nivel de conducción autónoma completa, en él se incluyen las características descritas en el nivel 4 y además no es dependiente del ODD, es decir, no tiene condiciones específicas que restrinjan la conducción autónoma en alguna situación concreta.

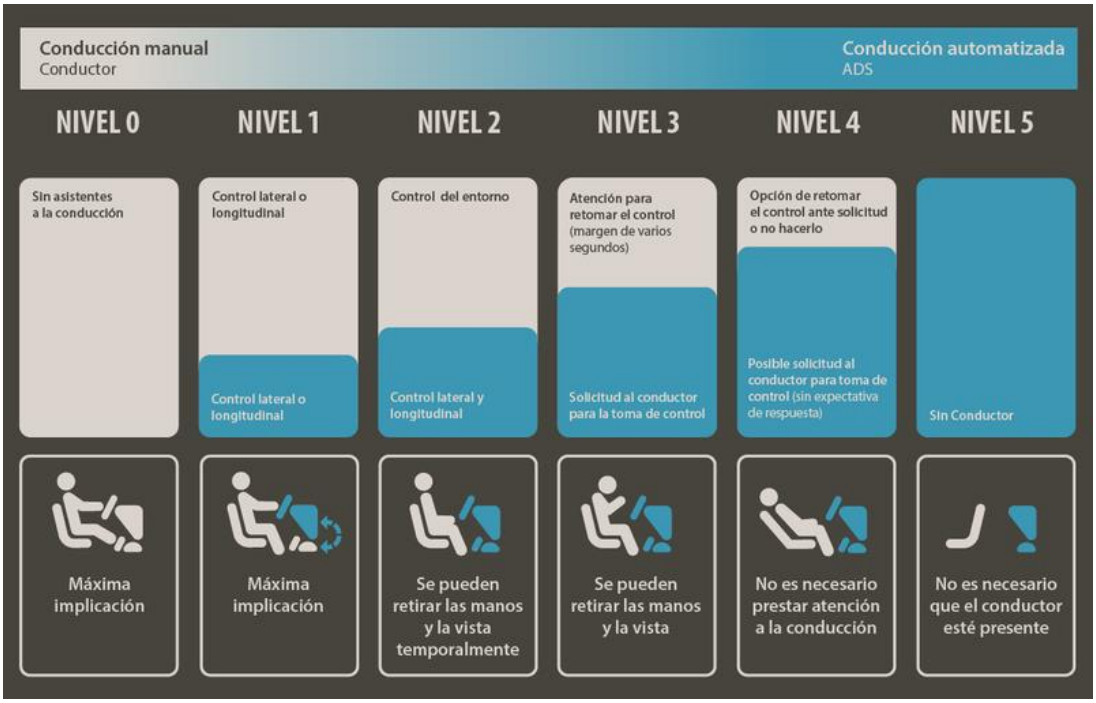


Figura 6. Niveles de conducción autónoma [29].

La evolución de la tecnología que implementan los sensores es clave para conseguir el nivel 5 de autonomía, ya que esto llevará a una monitorización precisa de la situación en cada instante del coche propio y del entorno que le rodea. Por lo tanto, estos sensores se pueden separar en dos grandes grupos: los propioceptivos, que son aquellos que proporcionan información sobre la dirección, velocidad o aceleración entre otro tipo de parámetros propios del coche; y los exteroceptivos, que se encargan de captar la información de la zona exterior que rodea al vehículo, como pueden ser para detectar la carretera, peatones, crear un mapa de la ruta, etcétera. Algunos de los sensores que se instalan en los vehículos autónomos son [30]:

- Unidad de medición inercial (IMU, *Inertial Measurement Units*): Sensor cuya función es obtener la aceleración líneal y angular, con las cuales se puede determinar la velocidad y posición del vehículo.
- Sistema de satélite para navegación global (GNSS, *Global Navigation Satellite Systems*): Este dispositivo nos permite determinar la posición y velocidad del vehículo.
- Odometría: Otro sistema que también nos proporciona la velocidad y la dirección del vehículo, pero esta vez a través de codificadores en los motores y otra clase de técnicas.
- Lidar: Proporciona un mapa en 3D detallado de la zona que lo rodea (Figura 7). También es posible encontrarse con lidar 2D, empleados sobre todo para mapeado.

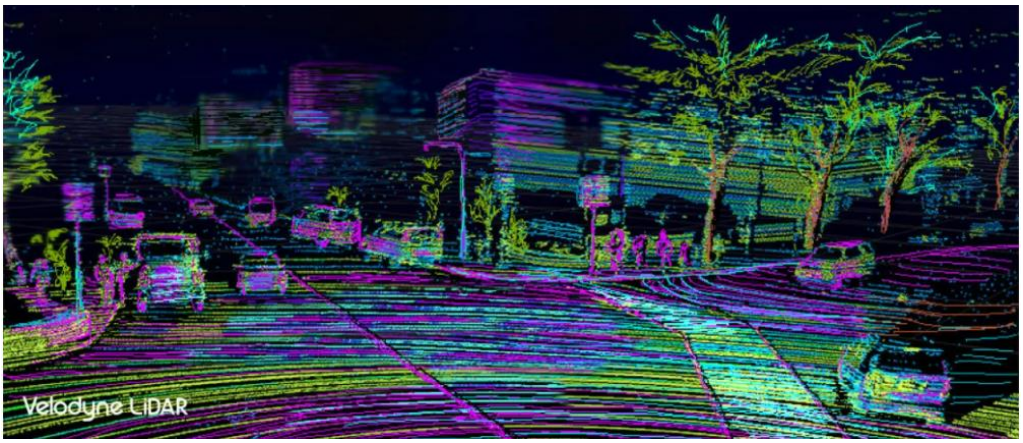


Figura 7. Imagen en capturada con la información extraída por un Lidar de la marca Vlodyne [31].

- Ultrasonidos: Muy utilizados en especial para aparcamientos autónomos debido a su sencillez y por su bajo coste.
- Radar: Permiten detectar objetos en escenarios de poca visibilidad.
- Cámaras: Permiten al coche obtener información sobre su entorno, detectando obstáculos y clasificándolos. Las cámaras estereoscópicas se emplean para conseguir información de la profundidad correspondiente a cada objeto percibido.

Figura 8. Imagen del interior de un Tesla Model S y de las imágenes de vídeo capturadas por las cámaras situadas en la parte izquierda, delantera y derecha del coche (de arriba abajo las imágenes de la derecha de la figura 8) [32].

La captura de información del exterior necesita un posterior procesado que será hecho por la "visión artificial", que se trata de una rama de la inteligencia artificial cuyo objetivo es conseguir información de alto nivel, es decir, saber reconocer y actuar frente al movimiento o posición de distintos objetos dependiendo de su posición o la velocidad a la que se muevan [33].

De hecho, según la página oficial de Tesla, empresa que ha hecho grandes avances en este campo (más adelante se verán algunos de sus proyectos), se enuncia que la única forma de lograr una solución general para la conducción autónoma total sea empleando un enfoque basado en una inteligencia artificial avanzada para la visión y la planificación, que lleve consigo un hardware que lo implemente eficientemente [34].

Otro punto importante a analizar es la seguridad de los coches autónomos. En la conducción se producen una gran variedad de escenarios a los que se tiene que enfrentar el vehículo autónomo, y deberá determinar las consecuencias de sus acciones calculando los riesgos que tiene cada una de ellas. Para determinar la probabilidad y el grado de severidad de los posibles accidentes y poder compararlos en caso de decisión se puede aplicar el análisis modal de fallos y efectos (FMEA, Failure Mode and Effect Analysis). Este mecanismo consiste en [35]:

- Identificar y reconocer fallos potenciales incluyendo sus causas y efectos.
- Evaluar los posibles fallos identificados.
- Sugerir acciones que puedan eliminar o reducir al máximo la probabilidad de que ocurran esos fallos potenciales.

Para evaluar los fallos, FMEA lo realiza de acuerdo a tres variables: severidad (S), ocurrencia (O), y detección (D). Con estas tres variables se calcula el número de prioridad de riesgo (RPN, *Risk Priority Number*):

$$RPN = S * O * D$$

En la página oficial de compañías que comercializan dentro del mercado del coche autónomo publican sus avances en el ámbito de la seguridad, como General Motors [36] o Waymo [37], ya que es uno de los puntos cruciales a definir y mejorar para una implantación a gran escala.

La complejidad de fabricar un coche autónomo, no solo viene determinada por cuestiones mecánicas, lógicas o de las condiciones del entorno sino también de la situación legal. En Europa las restricciones a este tipo de conducción se ven en muchos casos restringida debido al acuerdo firmado en la convención de Viena de 1968. España es una excepción ya que no se adhirió a ese acuerdo, y las pruebas de conducción autónoma han tenido mayor facilidad en realizarse en nuestro territorio [25].

En 2015 la DGT anunció un marco sobre el que exponía las condiciones para la realización de pruebas de vehículos autónomos en carretera. Según ese marco, se considera coche autónomo a aquellos coches con un nivel de autonomía mayor al nivel 3 de la escala estandarizada por la SAE. En ese mismo año se consiguió la primera experiencia de un vehículo autónomo, un C4 Picasso, recorriera en torno a 600 km que separan Vigo de Madrid, esta prueba fue un gran avance ya que se hizo en condiciones reales, es decir, en carreteras con tráfico [38].

El gobierno de España también ha hecho alianzas con los países vecinos, para realizar pruebas de coche autónomo entre los países, con el objetivo de normalizar la legislación del coche autónomo en los países europeos cercanos. En abril de 2018 se anunció el acuerdo entre el gobierno español y portugués para la creación de un corredor entre Vigo-Oporto para vehículos sin conductor [38]. En septiembre 2020 los respectivos ministros de transportes de Francia y España, firmaron un acuerdo de colaboración sobre la conducción autónoma facilitando ensayos de investigación con vehículos de este tipo en vías abiertas al tráfico en general [39].

2.1.1 Aplicaciones y mercado del coche autónomo

En esta subsección se mostrarán algunos de los desarrollos en coches autónomos de alto nivel (nivel 3 o superior) que podemos encontrar en el mercado. Algunas veces su conducción viene limitada por temas legales en algunos países como ya se explicó previamente.

En 2013 Tesla anunció su sistema de piloto automático. Todos los coches Tesla posteriores a 2014 tienen instalado al menos la primera versión de este software con su respectivo hardware. El piloto automático es como llama Tesla al sistema que facilita la tarea de conducción

al conductor, llevándola a un nivel 3 de autonomía. El primer modelo de Tesla venía dotado con las siguientes funcionalidades [40]:

- Gestión del carril, ya sea para mantenerse o cambiar de uno a otro activando el intermitente correspondiente.
- Gestionar la velocidad leyendo las señales verticales de limitación de velocidad.
- Control de la distancia con el coche de delante.
- Evitar choques laterales o frontales.
- Predicción de posibles accidentes activando el freno automáticamente.

Para el correcto funcionamiento de esta primera versión del piloto automático, el coche contaba con una cámara frontal, sensores ultrasonidos y un radar. En 2016 se comenzó a distribuir en los coches Tesla la segunda versión del piloto automático con más funcionalidades. Para empezar aquellos coches con "Autopilot 2.0" necesitan los siguientes dispositivos: 3 cámaras frontales, 4 cámaras laterales, sensores ultrasonidos mejorados y un radar con una distancia de 160 metros [41]. En la Figura 9 se puede observar la amplia zona de la cual el coche puede recabar información sobre objetos del exterior y actuar conforme a ello.

Figura 9. Zonas de percepción del sistema Autopilot 2.0 de Tesla [42].

Mercedes Benz es otro de los fabricantes de coches que ha conseguido llevar a cabo varios proyectos relacionados con el coche autónomo. En 2016 lanzó el sistema "Mercedes Benz Distronic Plus", para hacer competencia al sistema de Tesla. El sistema de la compañía alemana proveía al coche de las siguientes características [43]:

- Control de velocidad de crucero adaptativo.
- Ayuda al conductor a mantener constante la distancia entre su vehículo y el de delante, circulando a una velocidad inferior a 200km/h.

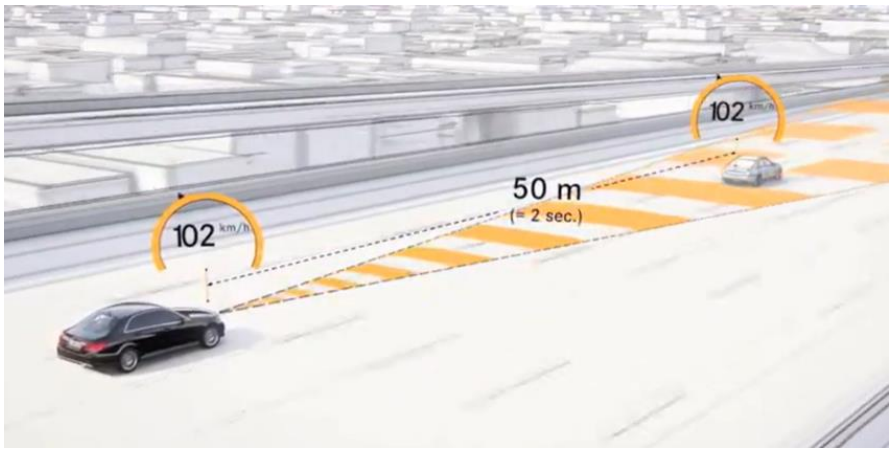


Figura 10. Fotograma del vídeo explicativo de Mercedes Benz sobre su sistema Distronic Plus, y su funcionalidad de automatizar el dejar el espacio entre dos vehículos [43].

- La servodirección inteligente ayuda al conductor a mantenerse en el centro del carril. En parte esto se consigue gracias a las cámaras estereoscópicas que detectan los carriles mientras el coche no supere los 200km/h y transmiten esa información a la servodirección eléctrica.
- En caso de una mala visión de las líneas, a velocidades inferiores a 60km/h, el coche es capaz de mantenerse centrado gracias a la posición de otros coches.
- Control de la velocidad, dependiendo de la velocidad máxima permitida y de la velocidad de los coches próximos a él.

En el caso de Distronic Plus, estaríamos hablando de un nivel de conducción autónoma de nivel 3. Pero Mercedes Benz también tiene un proyecto de autonomía de nivel 4. En 2019 Daimler, empresa matriz de Mercedes Benz, junto con Bosch consiguieron los permisos jurídicos por parte de las autoridades de Baden-Württemberg para implantar un aparcamiento completamente autónomo. Este sistema supone un nivel 4 de autonomía ya que se conduce solo sin la atención de una persona, pero solo dentro de un recinto concreto, por lo tanto, viene determinado por el ODD [25].

Figura 11. Interior del parking autónomo del museo de Mercedes Benz, con un Mercedes Benz Clase S dirigiéndose a su aparcamiento sin conductor [44].

Audi anunció en 2017 su sistema de ayuda a la conducción en atascos. Este sistema llamado “Audi AI Traffic Jam Pilot” permite tener un nivel de autonomía de nivel 3 en atascos. El conductor activará el botón de conducción autónoma y podrá dejar de tocar los pedales o el volante. Cómo se ha recalcado previamente, para activarlo se debe dar una serie de circunstancias: se debe circular por una carretera de 3 o más carriles, la velocidad máxima es de 37,3 km/h y en situaciones de atasco. Una vez finalicen las condiciones de operación y el sistema no consiga seguir con su funcionamiento avisará al conductor de que vuelva a tomar el control del vehículo [45].

En el mercado también podemos encontrar vehículos que alcanzan el nivel máximo en la escala definida por el SAE J1306, el nivel 5 de autonomía. A continuación, hablaremos de 3 empresas que ofrecen este tipo de servicio:

- Waymo: empresa filial de la compañía estadounidense Google, cuyo único objetivo es comercializar coches autónomos. En primer modelo de esta empresa el Waymo Firefly fue el primer prototipo en conseguir el nivel más alto de autonomía [46]. Actualmente la empresa Waymo ofrece este servicio en el modelo Chrysler Pacifica (Figura 12).

Figura 12. Un modelo Chrysler Pacifica de Waymo [47].

- En 2018 Toyota presentó en el CES de las Vegas su nuevo avance tecnológico llamado "Toyota platform 3.0". Este sistema permite llegar a un nivel 5 de autonomía gracias a la integración de numerosos sensores que daban la información suficiente al coche como para que tome decisiones por sí mismo. El Lexus LS con el que se presentó este proyecto en las Vegas, constaba de: 4 LiDAR orientados de hasta 200 m de distancia, 4 LiDAR orientados de corto alcance, más de 5 radares orientados en diferentes zonas del vehículo, sonars en torno al vehículo y un set de cámaras situadas junto al LiDAR de largo alcance delantero [48].

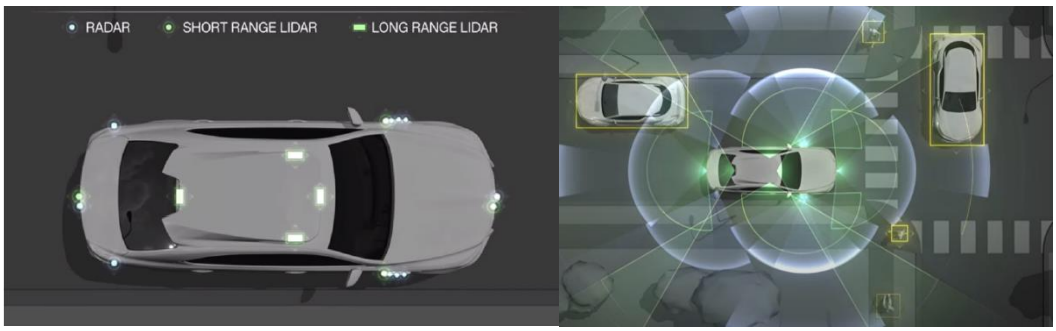


Figura 13. Posición de los sensores y alcance de ellos en el Lexus LS con el sistema Toyota Platform 3.0. [48]

- Uber: Otra de las empresas punteras en este ámbito, más conocidas por sus servicios de transporte privado, sufrió un retroceso en sus avances debido a un atropello en una fase de pruebas a un peatón a pesar de que el coche estaba preparado para prevenir estas situaciones [49]. Al igual que los demás vehículos de estas características contaba con una serie de sensores (Figura 14) que le permitía obtener control sobre el manejo del vehículo e información del exterior para actuar frente a los estímulos que percibía. A finales de 2020, Uber decidió abandonar este nicho de mercado vendiendo su división de coches autónomos a otra empresa [50].

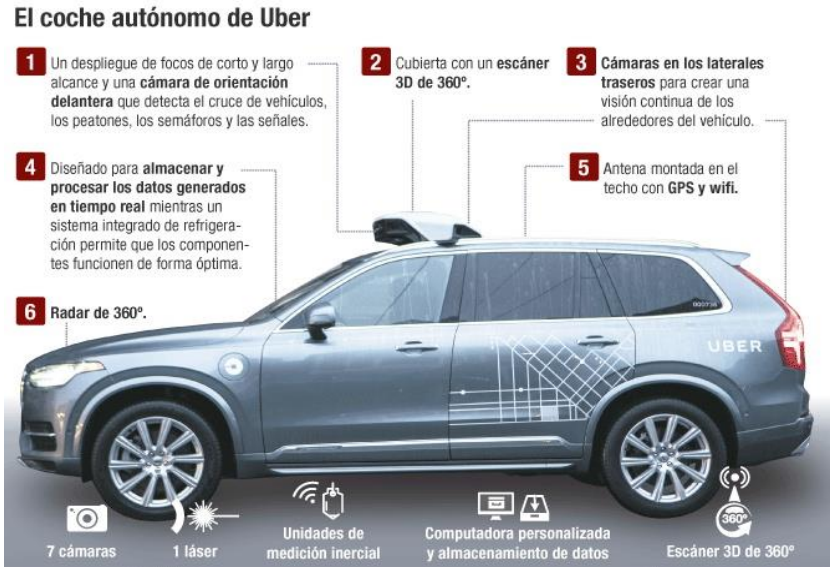


Figura 14. Características del coche autónomo diseñado por Uber [50].

2.1.2 Implementaciones previas del Twizy autónomo

Esta sección la dedicaremos para exponer diferentes proyectos previos de ámbito divulgativo e investigador cuyo objetivo era también dotar al Renault Twizy de una conducción autónoma de un nivel 3 o superior. Hablaremos de 3 proyectos, dos de ellos realizados en España (Sevilla y Cartagena) y otro hecho en Alemania (Dresden). Todos ellos realizan pruebas en entornos delimitados sobre rutas predefinidas, por lo tanto, estamos hablando de conducción autónoma sin necesidad de conductor, pero en un ODD restringido, es decir, de un nivel 4 dentro de la escala de la SAE.

Se comenzará exponiendo el proyecto alemán desarrollado en la facultad de ingeniería mecánica de la universidad técnica y de economía de Dresden en 2017. Para lograr la conducción autónoma realizaron los siguientes cambios en el Renault Twizy [52]:

- La columna de dirección del Renault Twizy fue reemplazada por la de un Renault Clio2. Esto permitió dotarlo de dirección asistida que no tenía la columna original, facilitando el control lateral del vehículo.

Figura 15. A la izquierda la columna de dirección de un Renault Clio2, y a la derecha la columna original del Renault Twizy del proyecto. [52]

- Accedieron al pedal del acelerador mediante un relé, que cuando se activa la conducción autónoma permite a la lógica de guiado acceder a él.
- Se accedió a las marchas del coche. Pudiendo poner el coche en directa (D), punto muerto (N) o marcha atrás (R).
- Se diseñó un programa en Matlab para simular la lógica de conducción y los parámetros necesarios para que sea autónoma.
- Añade al Twizy un módulo de comunicaciones V2X, una videocámara en la parte frontal y un láser en el morro delantero.

El proyecto de investigación desarrollado por la universidad politécnica de Cartagena plantea alguna modificación diferente. Por ejemplo, para conseguir la dirección asistida se le añadió un motor a la columna de dirección para ofrecerle menos oposición al giro cuando entre en modo automático. A diferencia del proyecto alemán, se realiza la modificación también en el freno (Figura 16), para conseguir automatizarlo [53].

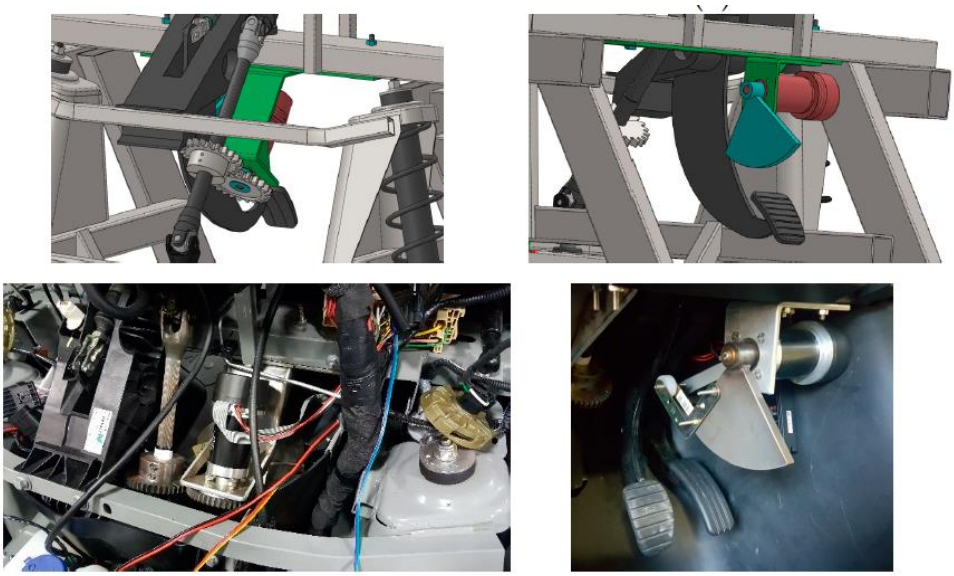


Figura 16. En la parte izquierda: arriba, esquemático de la posición del motor al lado de la columna de dirección; abajo, implantación final del final. A la derecha: arriba, esquemático del sistema empleado para el freno; abajo, el resultado final de ese sistema de freno autónomo. [53]

Sobre el sistema de percepción, lo dividen en dos grupos de sensores: de corto alcance, llegan hasta 10 m con respecto a la posición de delante y trasera y 3 m en con respecto a ambos lados; y de largo alcance, consiste principalmente en el lidar 3D de alta definición capaz de llegar a 100 m de distancia con una precisión de 2cm [53].

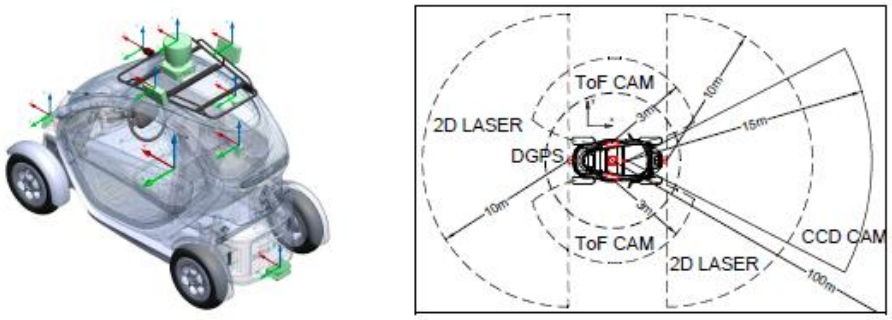


Figura 17. Sistema de percepción creado por la Universidad de Cartagena para le Renault Twizy [53].

El proyecto desarrollado en Sevilla se centra más en la parte software del algoritmo que controla la conducción del coche autónomo. Para ello emplea el entorno de simulación de robótica en 3D Gazebo, que viene integrado en ROS (*Robot Operating System*). ROS es de código abierto de forma que los usuarios pueden elegir la configuración de librerías y herramientas para que puedan interactuar con la máquina correspondiente. ROS se trata de un programa con muchas aplicaciones, en este caso enfocado al coche autónomo, las más interesantes son: percepción (disponen de una amplia gama de sensores de diferentes marcas con los que realizar las pruebas), identificación de objetos, movimiento automático, reconocimiento y control y planificación [54].

Figura 18. Entorno de simulación de ROS con el modelo Renault Twizy empleado por la universidad técnica superior de Sevilla [54].

2.2 Estado del arte del vehículo conectado

En 2010 el instituto de ingenieros eléctricos y electrónicos (IEEE, *Institute of Electrical and Electronics Engineers*) anunciaba un estándar para las comunicaciones inalámbricas en vehículos: 802.11p. El estándar IEEE 802.11p fue ideado para ser empleado por dispositivos que se encuentren en entornos donde las propiedades de la capa física pueden cambiar rápidamente y donde se necesitan intercambios de información en un periodo corto de tiempo. El objetivo de este estándar es proporcionar las especificaciones mínimas requeridas para garantizar la interoperabilidad entre los dispositivos inalámbricos que intentan comunicarse en entornos de comunicaciones potencialmente cambiantes y en los que las comunicaciones ocurren dentro de un marco de tiempo muy limitado [55].

En 2012 se publicó el estándar ITS-G5 que tiene como objetivo proporcionar diferentes formas gestión del tráfico y permitir a los usuarios estar mejor informados para aumentar la seguridad. Este estándar usa principalmente el protocolo IEEE 802.11p en las capas física y de enlace. Fue creado y desarrollado principalmente en Europa, y tuvo competencia por parte de otros estándares de objetivo final similar, cabe destacar, IEEE WAVE de Norte América y ARAB T109 desarrollado en Japón [56].

En 2016 la organización 3GPP (*3rd Generation Partnership Project*) lanzó su estándar C-V2X (*Celullar V2X*) como alternativa al estándar propuesto por IEEE. El término V2X, ya usado también en el documento de 802.11p, hace referencia a las comunicaciones vehículo con todo tipo de elementos, que se pueden clasificar en [57]:

- V2V: Vehículo a vehículo, se intercambian información sobre la posición, velocidad próximos movimientos, esto evita choques y aumenta la seguridad.
- V2I: Vehículo a infraestructura, comunicación con señales o semáforos.
- V2P: Vehículo a peatón, emitir alertas por la presencia de peatones o ciclistas.
- V2N: Vehículo a red, información del tráfico en tiempo real y conexión a servicios en la nube.

C-V2X suponía los pilares de la comunicación vehicular con 5G. Este estándar ha sufrido modificaciones conforme pasaban los años incorporándole nuevas funcionalidades (Figura 19). Este estándar ha tenido el apoyo de la principal asociación de estándares 5GAA (*5G Automotive Association*). 5GAA se trata de una organización mundial encargada de agrupar distintos sectores tecnológicos como el sector de la automoción y las telecomunicaciones para la estandarización del uso del 5G. Fue creada en 2016 y entre sus miembros se encuentran fabricantes de coches tales como: AUDI AG, Grupo BMW, Daimler AG, Ericsson, Huawei y Qualcomm entre otras [58].

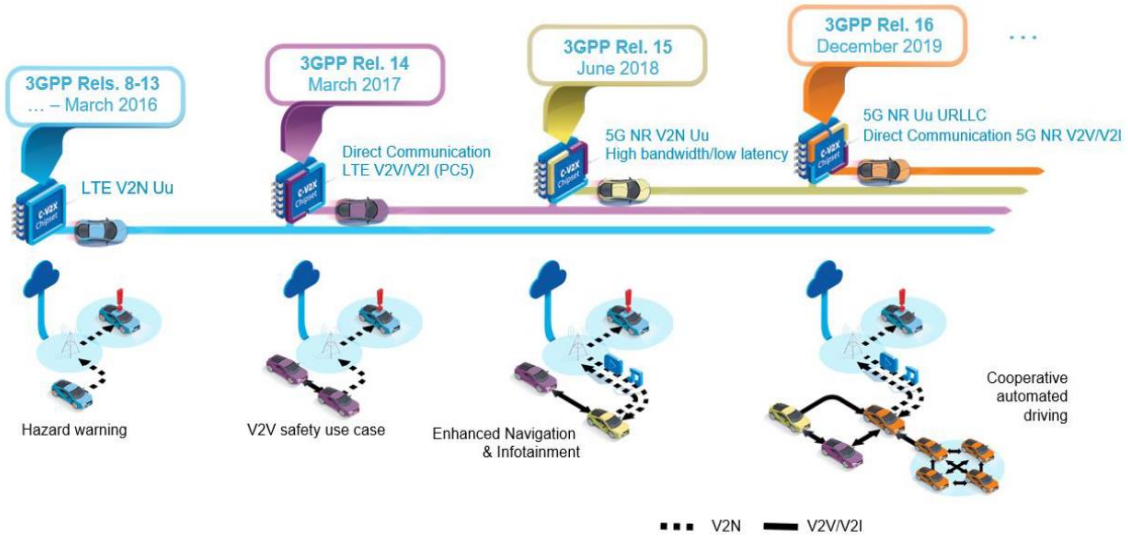


Figura 19. Evolución del estándar C-V2X de 3GPP [59].

A partir de la revisión número 15, anunciada a mediados de 2018, la estandarización propuesta por 3GPP se llama 5G-NR-V2X. Esta nueva versión, traía consigo nuevas capacidades a C-V2X para la conducción autónoma como son:

- Percepción: Alta transmisión de datos desde los sensores y una precisa modelación del mundo real.
- Planificación de rutas: Intercambio de trayectorias para realizar maniobras más seguras y rápidas.
- Posicionamiento: Geolocalización en tiempo real, compartiendo esa información con las infraestructuras pertinentes o con otros vehículos cercanos para evitar acercamientos peligrosos.
- Conducción coordinada: Intercambio de los datos obtenidos de los sensores para una conducción autónoma más predecible y coordinada.

La revisión número 16 supone una mejora de las capacidades que este estándar podía alcanzar, siendo:

- Alta tasa de transmisión: Alta eficiencia espectral para lograr una tasa binaria mayor.
- Baja latencia: Latencia menor a 1ms para situaciones de tiempo limitado.

- Alta fiabilidad: Soporta tanto unidifusión (*unicast*) como multidifusión (*multicast*).
- Vehículo a velocidades altas: El sistema está preparado para trabajar con altas tasas de transmisión mientras el coche no supere los 500km/h.
- Interoperabilidad: Puede coexistir con versiones anteriores del estándar y es compatible con ellas.

A comienzos de 2019 se expuso en el Mobil World Congress (MWC) de Barcelona varios casos de uso de coche con 5G. En este caso se trató de una colaboración entre Telefónica y Seat (Figura 20). Esto se consiguió en parte también por la solución de Edge Computing de Ericsson que hace posible el proporcionar una alta capacidad de cómputo al borde de la red y latencias mínimas. Esta prueba supone un nivel de autonomía de nivel 3, y no solo se probó las ayudas de 5G a la conducción sino también su utilidad en el campo del entretenimiento dando servicio a descargar contenidos en 4K gracias al gran ancho de banda que ofrece 5G [60].

Figura 20. Seat Ateca adaptado al uso de tecnologías 5G empleado en la demostración del MWC [60].

2.2.1 Estado del arte de FOTA

Uno de los servicios de vehículo conectado que proporcionan las marcas de coches se llama FOTA (*Firmware Over The Air*). Como ya se comentó en el capítulo de motivación, este sistema resulta de especial interés para los fabricantes de coches ya sea para acortar el tiempo de lanzamiento de los nuevos coches o permitir modificar algoritmos vitales para el funcionamiento del coche.

FOTA permite la actualización del software de un coche en remoto sin necesidad de conectarse físicamente a él. Esto sería posible realizarlo en aquellos coches con acceso a internet. Aunque la descripción del servicio puede resultar sencilla, los diferentes riesgos durante la operación del mismo que hay que tener en cuenta hace que su implementación tenga mayor complejidad de la que se puede esperar de su simple definición. Por ello, a continuación se verán las arquitecturas más comunes para implantar este servicio. Además, también se hablará sobre un estudio de posibles algoritmos de cifrado en este tipo de comunicaciones donde la seguridad es vital.

En 2018 la universidad de Canadá publicó un artículo donde plantea una arquitectura usando el protocolo MCC-FOTA para actualizar las ECUs en especial aquellas denominadas como HVAC (encargadas de calefacción, ventilación y aire acondicionado) [61].

Para testear esta arquitectura se realiza un análisis de datos de la empresa MCC (Mobile Climate Control Group) sobre IoT (*Internet of Things*) en vehículos para desarrollar un sistema de conectividad en la nube llamado MCC Fleet Tracker (MCC-FT). Pese a que este servicio

alojado en la nube ya proporciona servicios de análisis de mantenimiento predictivo, la actualización de las ECU automotrices aún requería la retirada de los vehículos de los clientes para poder acceder a ellas [61].

Por lo tanto, MCC anunció un protocolo de actualización de software por aire basado en tecnologías FOTA para MCC-FT para la reprogramación remota de ECU.

El hardware que MCCFleet Tracker implementa se puede ver en la Figura 21. Consta de dos partes el T35-1215, que es el módulo de procesamiento, y el T5320A+G que se conecta al otro dispositivo por RS-232, tiene un sistema GPS embebido y soporta comandos AT (conjunto de comandos Hayes, para configurar y parametrizas módems) [61].

MCC desarrolló su propia red de bus CAN basada en SAE J1939. Esta red CAN privada utiliza el estándar CAN 2.0B para la capa física y de enlace de datos, similar a la descrita en el SAE J1939. Todas las ECUs de MCC se comunican a través de ese bus CAN [61].

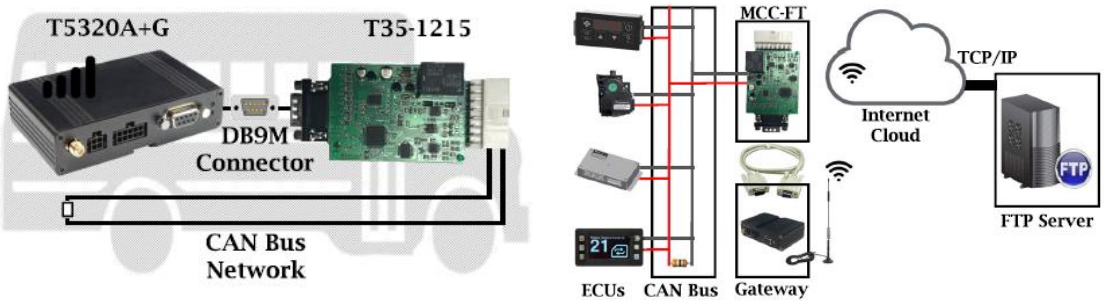


Figura 21. MCCFleet-Tracker a la izquierda y esquema de la implementación en un coche de ese hardware [61].

El protocolo de actualización, denominado MCC-FOTA, propuesto consiste en 6 pasos [61]:

1. La confirmación de que hay una actualización de firmware disponible a través del servidor FTP: Cuando se genera la actualización de software, el administrador del servidor cargará el archivo ".hex" correspondiente en el servidor FTP.
2. Descargar el archivo de actualización publicado en la memoria flash T5320A + G.
3. Almacenamiento y verificación de la actualización del firmware.
4. Identificación de la ECU objetivo: Cada paquete de actualización software incluye el número de identificación de la ECU correspondiente que se quiere actualizar.
5. Reprogramación de la ECU objetivo: Se transmitirá un ACK por difusión (broadcast) a través de la red de bus CAN privado de MCC, cuando el número de identificación coincide con una de las ECU del sistema. Este mensaje solicita a todas las ECU conectadas que envíen su número de identificación y ejecuten la versión de firmware en MCC-FT. Esto confirma que la ECU objetivo está en línea y disponible para reprogramación.
6. Verificación de la actualización de software realizada correctamente: A medida que se actualiza la versión de firmware, MCC-FT envía una notificación de éxito o fallo a la nube.

En la Figura 22 se muestra el diagrama de flujo del funcionamiento del sistema MCC-FOTA previamente descrito.

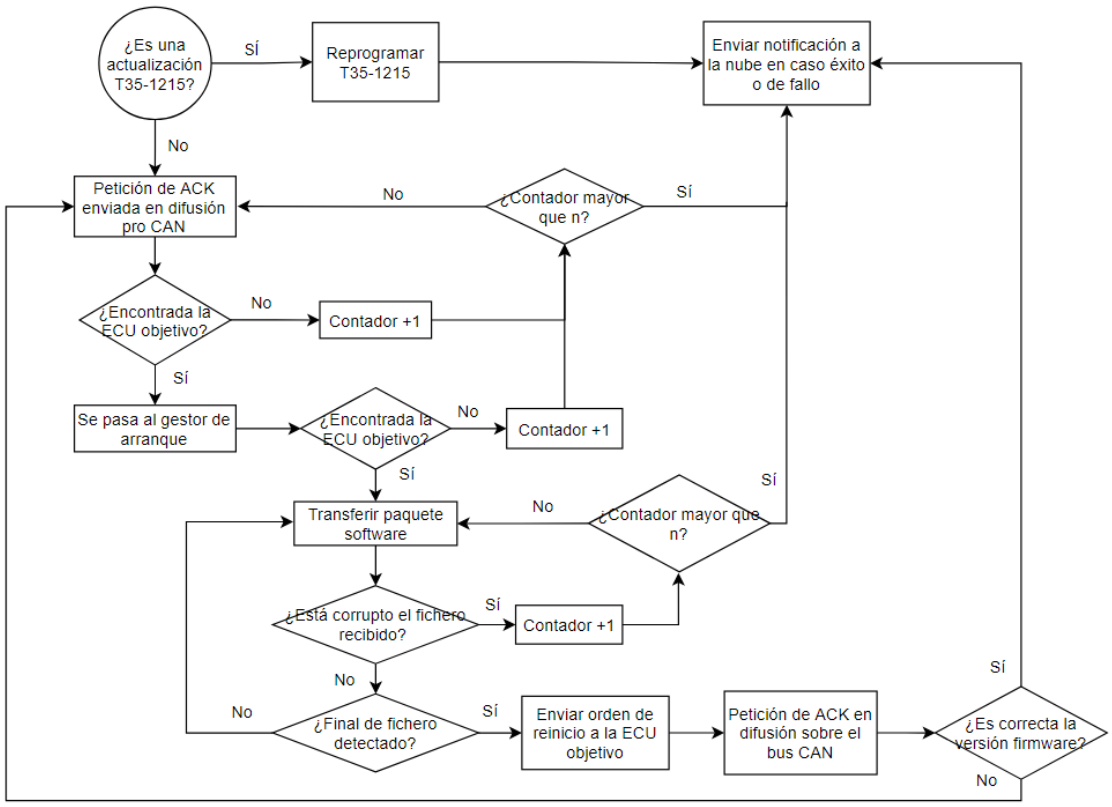


Figura 22. Diagrama de flujo del funcionamiento de MCC-FOTA [61].

A continuación, hablaremos del sistema FOTA implementado por Renault España. Se distinguen dentro de la ECU 3 partes: software, calibración y configuración. Existen dos tipos de FOTA: FOTA software, encargada de las dos primeras partes; y FOTA de configuración, se ocupa de la tercera parte. Esta sección se centrará en explicar el denominado como FOTA software que es el llevado a cabo dentro de la DIE (Dirección de Ingeniería España) de Renault en Valladolid.

Renault ha desplegado su sistema FOTA, siguiendo una estrategia de tres fases. La primera fase tiene como ECU objetivo la IBI. Esta ECU se caracteriza por su rapidez de procesamiento y por tener acceso directo a la TCU (*Unidad de control de transmisión* – encargada de ofrecer conectividad al exterior) y será la encargada de descargar las piezas de software llamadas deltas [1].

Se denomina delta, el paquete software necesario para dar un salto a una versión software más nueva. Para un salto de versión pueden existir varios tipos de deltas ya que cada tipo de coche tendrá sus características propias (funciones distintas, tamaño del tablero de a bordo diferente, etc.). Normalmente para no tener el mismo número de deltas que de coches (sería inviable el almacenamiento) se asigna una delta a un lote de coches con características similares.

En este primer paso la IBI puede actualizarse a sí misma y está puede actualizar una segunda ECU denominada IBC, ya que las dos están conectadas por el mismo CAN multimedia.

Entre los distintos buses CAN se incluyen ECUs que realizan la misma función que un router o Gateway en redes IP. Al separar los distintos buses también se separan las etapas FOTA.

[1]Información recabada durante una conversación con el tutor de Renault Daniel Castaño Navarro, líder en ingeniería de sistema conectividad y multimedia en Groupe Renault España.

La ‘Gateway’ (puerta de enlace) separa los distintos buses CAN, separando a su vez las etapas FOTA. En esta fase se distinguen 3 tipos de procedimientos [2]:

- Campaña de inventario: Se obtiene la información general de las ECUs, sobre el estado del coche, tipo de versión software tanto de la IBI como de la IBC.
- Campaña de logs: Se recuperan los logs que almacenan las operaciones realzadas y si han tenido éxito o algún fallo, para poder solventar problemas a la hora de la descarga de la delta.
- Campaña de actualización: Esta parte se puede hacer de forma manual, el empleado de Renault autoriza una actualización software de ese coche para que pueda conectarse al servidor donde se albergan las deltas, cuando haga la consulta el servidor tendrá que buscar en su base de datos qué delta debe enviar para ese coche específico. También se puede hacer de forma automática, el coche es el que pide una actualización a la versión más nueva, en caso de que tengan autorización previa. Hay que tener en cuenta que no se habilita que todos los coches puedan hacer peticiones continuamente para una actualización, ya que multiplicando el volumen de coches a los que da soporte Renault por una media de 2GB de descarga por delta, el procesamiento requerido por el servidor sería inmenso.

Las operaciones a realizar por parte de la IBI, aparte de descargar la delta y distribuirla a la ECU objetivo, serían la instalación y activación. A la hora de realizar estas operaciones dependerá del estado en el que se encuentre el vehículo y de qué tipo de actualizaciones se trata. La descarga puede hacerse incluso con el coche en movimiento, pero cuando se realiza la instalación depende si afecta a la seguridad. Por ejemplo, la actualización de la IBI se trataría de un proceso que no afecta a la seguridad, pero ni siquiera a otras funciones como los mapas, y seguiría un proceso como el descrito en la Figura 23. En esa imagen se observa:

- La descarga se hace mientras el coche está en movimiento.
- La instalación al no ser intrusiva en cuestiones de seguridad puede hacerse en movimiento.
- Si se apaga el coche mientras se estaba instalando, el procedimiento se queda en espera para que cuando vuelva a encenderse se termine la instalación.
- Cuando se vuelva a apagar el coche, se activa la actualización, lo que supone que el coche mantiene los calculadores (ECUs) despiertos durante el periodo necesario para la activación.
- Cuando el usuario vuelva a conducir su vehículo, ya lo tendrá actualizado.

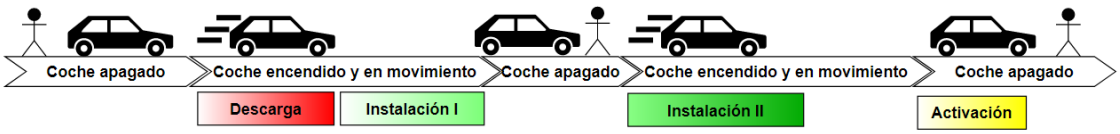


Figura 23. Ejemplo de un procedimiento FOTA empleado en Renault.

Hasta aquí hemos descrito la primera fase de despliegue del sistema FOTA que afecta a las ECU IBI e IBC. Sin embargo, el despliegue del sistema FOTA no se quedó estancado en la primera fase descrita, sino que se progresó a una segunda fase. En esta fase más avanzada, la IBI continúa haciendo la función de descarga de la delta de actualización y ahora también es capaz

[2]Información recabada durante una conversación con el tutor de Renault Daniel Castaño Navarro, líder en ingeniería de sistema conectividad y multimedia en Groupe Renault España.

de, a través de la Gateway, transmitir esa delta a otros calculadores fuera del CAN multimedia. Aquí aparece una de las primeras problemáticas a las que se enfrenta FOTA.

Como ya se comentó previamente una delta puede ser de 2GB de tamaño que para transferirlo entre ECUs por el bus CAN convencional (1Mbit/s, si es de alta velocidad) requiere un tiempo considerable. Frente a este canal de comunicación se abre el debate de utilizar otros tipos de redes y protocolos como son CAN-FD (5Mbit/s) o mismamente Ethernet (Ethernet Cat 5 tiene 100Mbit/s de tasa de transmisión), con el cuál la velocidad se vería incrementada en 100 veces más como mínimo.

El tamaño de estos paquetes de ficheros también plantea un dilema en la creación de los controladores. En ellos se albergan dos memorias, que denominaremos A y B, de tal manera que el software que se está ejecutando está en A, cuando llega una actualización se almacena en B y cuando se reinicie arrancará con el software de B. Si funcionase mal el arranque en B, el mismo controlador se reiniciaría utilizando de nuevo la memoria A. Esto supone un alto coste a los fabricantes, porque implica integrar memorias de gran almacenamiento y con una alta capacidad de procesamiento.

También cabe destacar los problemas derivados de las dependencias software entre controladores. Puede darse el caso de querer actualizar 8 de los 10 controladores disponibles, pero las nuevas versiones no son compatibles con las versiones de las dos ECUs sin actualizar. Esto a llevado a que se preparen paquetes de actualización para grupos de ECUs (en vez de individuales) que cubra y se ocupe de las dependencias de otros controladores.

Como se puede apreciar examinando las dos arquitecturas previas, la seguridad en este tipo de comunicaciones resulta una parte crucial para poder descargar correctamente los archivos de actualización o evitar cualquier tipo de intrusión de terceros al sistema. Una opción para solucionar esta problemática, es aplicar algoritmos de encriptado para mejorar la seguridad en el sistema FOTA de los vehículos.

Un ejemplo de cómo afrontar la problemática de la seguridad lo encontramos en el trabajo realizado en la universidad de Chongqing (China), donde se realiza varias pruebas para comprobar la eficiencia de 3 tipos de algoritmos distintos. De entre ellos se decide por sus prestaciones usar una encriptación híbrida: algoritmo AES (*Advanced Encryption Standard*) para encriptar la información en el transmisor más el algoritmo ECC (Elliptic Curve Cryptography) para encriptar la clave AES. También se muestra una comparación entre las principales técnicas de encriptación [62]:

- Encriptación simétrica: Para una gran cantidad de datos la eficiencia es mayor que la de la asimétrica. El algoritmo AES tiene mejores prestaciones frente a DES (*Data Encryption Standard*) o 3DES.
- Encriptación asimétrica: Mayor velocidad de encriptar/desencriptar que la simétrica. Dentro de esta categoría, cabe destacar por sus características el algoritmo ECC (frente a otros como RSA o ElGamal).
- Algoritmos de Hash: Usados para proteger la integridad de los datos.

3.

Servicios de conducción autónoma en un Twizy

Esta sección se ocupará de explicar todo el procedimiento llevado a cabo para conseguir tener una conducción autónoma en el Twizy en un ODD definido, es decir, bajo unas circunstancias y una infraestructura concreta.

Cabe destacar que esta sección se dividirá en tres subsecciones: una primera parte introductoria donde se expondrá las características del modelo Renault Twizy con el cual vamos a interactuar y una descripción del entorno de conducción autónoma; una segunda donde se explicará las modificaciones mecánicas que se han realizado en el Twizy, y una última donde ya se detallará el software empleado en cada uno de los elementos implicados para conseguir controlar adecuadamente el Twizy.

3.1 Estado inicial del proyecto

El objetivo de esta sección será la de explicar las propiedades del coche con el que vamos a trabajar. Un conocimiento previo de los componentes del coche y su estructura nos ayudará en apartados posteriores al posicionamiento de nuevos elementos o a poder modelar correctamente su movimiento. A continuación, se dará todo detalle del aparcamiento autónomo donde debe operar. Después se pasará a mostrar los cambios necesarios para que el Twizy tenga la opción de la conducción autónoma en determinadas situaciones que se vendrán limitadas por las operaciones realizadas en el parking.

3.1.1 Características del Renault Twizy y del parking autónomo

El Renault Twizy se trata de un vehículo eléctrico de dos plazas (Figura 91). Existen dos versiones de Twizy, con distintos límites de velocidad máxima: de 45km/h y el de 80km/h. El que se donó desde la Fundación Renault se trata del modelo de 80km/h.

Figura 24. Modelo Renault Twizy [63].

El vehículo consta de dos baterías. La batería principal es de ion de litio y se sitúa en la parte baja del chasis, y proporciona una autonomía de 90 km. Esta batería puede administrar una tensión de 48 o 56 V y un amperaje de 360A.

La segunda batería se encuentra en la parte baja del morro. Proporciona un voltaje de 12V y una intensidad de corriente de 12Ah. Cuando se quiera alimentar nuevos elementos se preferirá conectar a esta batería meramente por temas de seguridad, si la potencia de la pila es suficiente como para suministrar la energía necesaria.

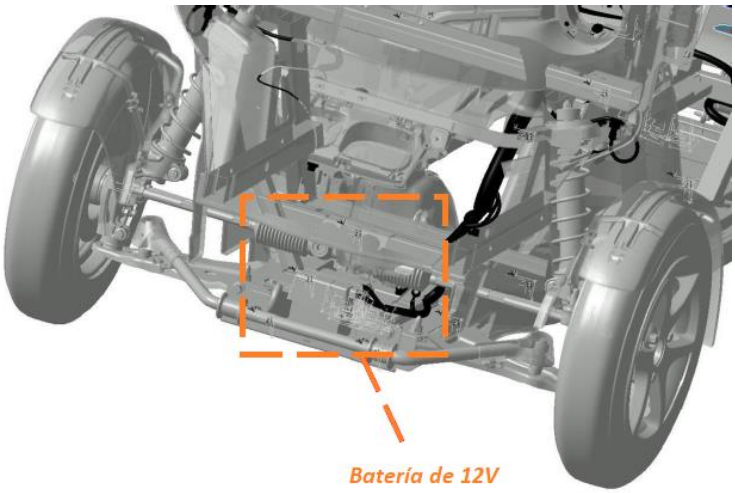


Figura 25. Posición de la pila de 12V.

En el Anexo 1 podemos ver más características sobre este vehículo como su envergadura, giros de volante máximos, tipo de caja de cambios y otros aspectos que nos serán de utilidad más adelante.

Al ser un coche eléctrico y de un tamaño relativamente pequeño comparado con un coche convencional de 5 plazas, resulta ideal para su uso en ciudades, donde el espacio es limitado y se está combatiendo contra la contaminación aérea. En este escenario es donde se quiere aplicar el parking ideado por el grupo de trabajo TwizyLine. Este parking se muestra en la Figura 26 y será el espacio en el cual el vehículo podrá circular en modo autónomo, por lo tanto, se está planteando una conducción de nivel 4, ya que, no se necesita ninguna persona para su manejo y está en un ODD limitado. El funcionamiento del parking sería el siguiente [18]:

- Al estar enviando el coche constantemente su posición al servidor, este ordenará abrir la barrera de un parking cuando éste se sitúa cerca de la “leaving zone” de algún parking.
- Una vez el coche dentro de la “leaving zone”, el coche leerá la tarjeta RFID situada en el suelo. El usuario se bajará, abandonará esa zona a través de un torno y el procedimiento de entrada del coche al parking se iniciará.
- La barrera que habilita la entrada al recinto se abrirá. El servidor ordenará entrar en modo autónomo a este coche. Se le ordenará aparcar en una posición.
- El coche se guiará gracias a la cinta magnética que se encuentra en el suelo. Esta cinta indicará por donde el coche puede circular. También se usará el detector RFID integrado en el coche y varias tarjetas RFID distribuidas por el parking que ayudarán al coche a decidir en las bifurcaciones y a detectar el final de la fila de aparcamientos.
- En la salida se aplica la misma tecnología de guiado, pero ahora el vehículo saldrá por la “pick-up zone”.
- El coche procederá a salir una vez sea solicitado por un usuario mediante la aplicación móvil.

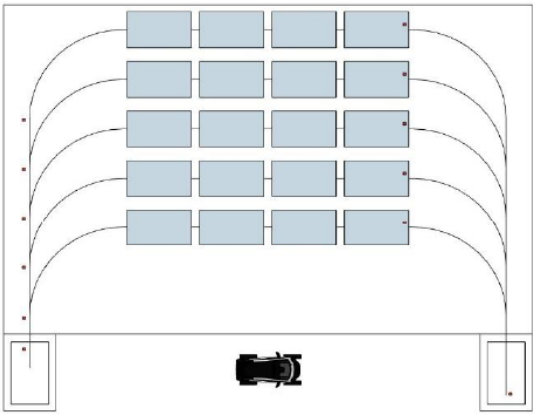


Figura 26. Parking autónomo ideado por TwizyLine, teniendo la leaving zone a la izquierda y la pick-up zone a la derecha, y representando las tarjetas RFID como puntos. [18]

3.1.2 Nuevas funcionalidades de acuerdo al ODD

Se deberán añadir una serie de modificaciones para que consiga cumplir con las funcionalidades necesarias para operar el parking. A continuación, se listará la serie de dispositivos seleccionados para integrar dentro del Twizy junto con la funcionalidad que completa [24]:

- Módulo de comunicaciones: Será una máquina capaz de procesar las comunicaciones con el exterior. En este caso las conexiones serán con el servidor TwizyLine.
- Módulo de control: Será la máquina capaz de controlar los movimientos del Twizy cuando este entre en modo autónomo dentro del parking.
- Dispositivo GPS: Este elemento servirá para conocer la posición del coche y detectar su proximidad a parkings de TwizyLine.
- Sensores de ultrasonidos: El Twizy ya dispone de sensor de ultrasonidos en la parte trasera, sin embargo, no dispone de ellos en la parte delantera. Estos dispositivos nos ayudarán a evitar choques con cualquier obstáculo de forma frontal cuando esté en modo autónomo.
- Sensor magnético: Este sensor será capaz de detectar la banda magnética que dibuja el circuito dentro de los parkings.
- Sensor RFID: Su objetivo será el detectar cuándo pase por encima de una tarjeta RFID.
- Motor: Este elemento deberá ser capaz de girar el volante para realizar los giros en modo autónomo, ya que, el Twizy no dispone de dirección asistida.

Algunos de estos elementos ya se disponían previamente, ya que fueron parte del trabajo final de grado de Ignacio Royuela González [24], como son el módulo de comunicaciones, el módulo de control y el dispositivo GPS.

La introducción de estos elementos en el vehículo supondrá primero una integración del hardware y después un desarrollo software para conseguir desarrollar las funciones deseadas, con el fin de que el Twizy esté operativo para funcionar en uno de los parkings previamente explicados.

3.2 Implementación hardware

Una vez se dispuso de los elementos necesarios para completar este proyecto, se hizo un estudio sobre las posibles ubicaciones de cada uno de ellos. Su posicionamiento debe ser dependiente de la función que va a realizar. A continuación, se mostrará dónde se han colocado primero el módulo de comunicaciones y el GPS, después el módulo de control y su conexión al resto de sensores, seguidamente hablaremos de esos sensores y su localización dentro del coche. En la Figura 27, se muestra un esquema general de cómo están conectados todos los elementos.

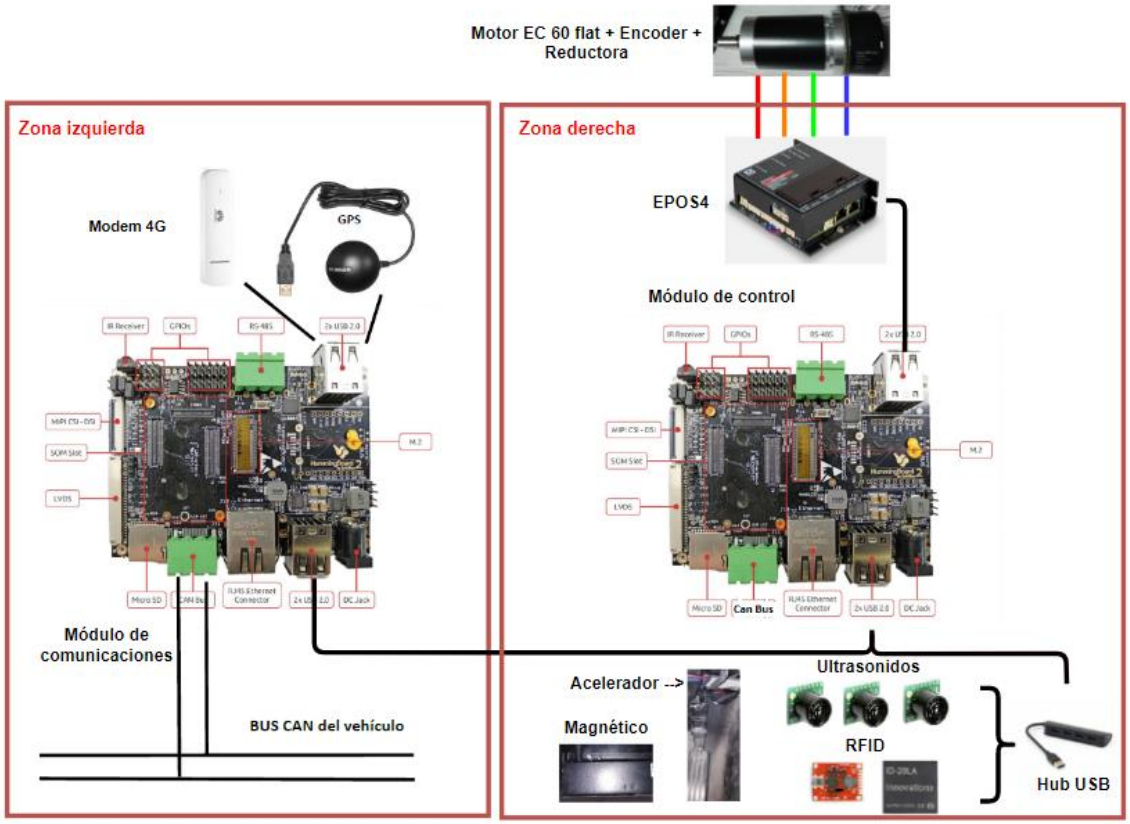


Figura 27. Esquema de conexionado en el Twizy.

3.2.1 Posicionamiento sensores

La ubicación de cada uno de los sensores será clave para que cumplan correctamente su cometido. Comenzaremos hablando sobre el sensor magnético. Este sensor fue proporcionado por la compañía Astii Robotics, y es capaz de detectar cinta magnética a una distancia máxima de 35mm [64]. Debido a sus características, fue colocado en la parte inferior delantera del coche según se aprecia en la Figura 28.

Figura 28. Sensor magnético situado abajo de la parte delantera del coche.

Para conseguir que se sujete en esa posición se ha diseñado una estructura anclada a un hierro de la parte del chasis inferior delantero. Además, esa estructura permite que el sensor se retire, en caso de no ser usado, con el fin de que no se dañe por algún obstáculo fuera del parking.

El sensor RFID está instalado de forma similar al sensor magnético, y es que necesita operar cerca del suelo para detectar las tarjetas RFID que se encuentra en el suelo del parking e indican al coche las bifurcaciones o los finales de línea. Como las tarjetas RFID se encuentra siempre a la izquierda de la línea se decide poner el sensor RFID en la posición mostrada en la Figura 29.

Figura 29. Sensor RFID sujetado al chasis a la derecha de la rueda izquierda.

Para poder acceder al pedal del acelerador, se diseñó un sistema de relés junto con dos potenciómetros digitales, que gobernados por el Arduino determinarán quién controla el acelerador (Figura 30). El relé habilitará la conexión con nuestro módulo de control una vez está en modo autónomo, y deshabilitará esa conexión cuando esté en modo manual y así el usuario del Twizy pueda llevárselo.

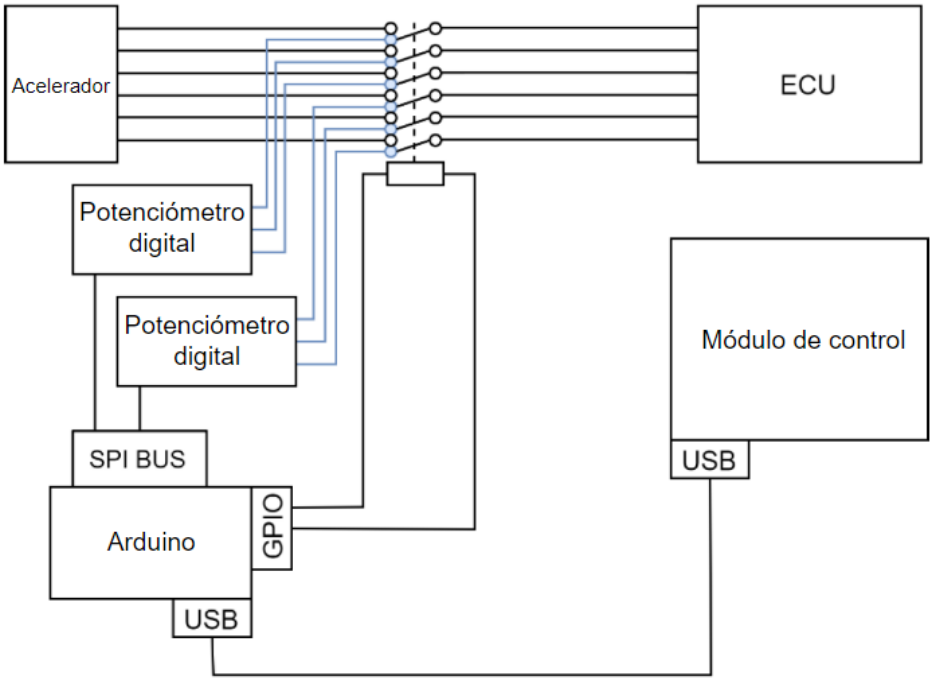


Figura 30. Esquema del conexionado para controlar el acelerador. [24]

El receptor GPS debe estar situado en una posición donde pueda tener una buena cobertura de ahí, que se decidiera en el techo del coche.

Figura 31. Receptor GPS Garmin colocado en el techo del coche, justo encima de la luna.

Por último, se dispone de 3 sensores de ultrasonidos cuya función es detectar obstáculos enfrente del coche y así evitar cualquier tipo de colisión. Por ello, se planteó y decidió su implantación en la parte delantera del morro del coche, tal y cómo se ve en la Figura 32.

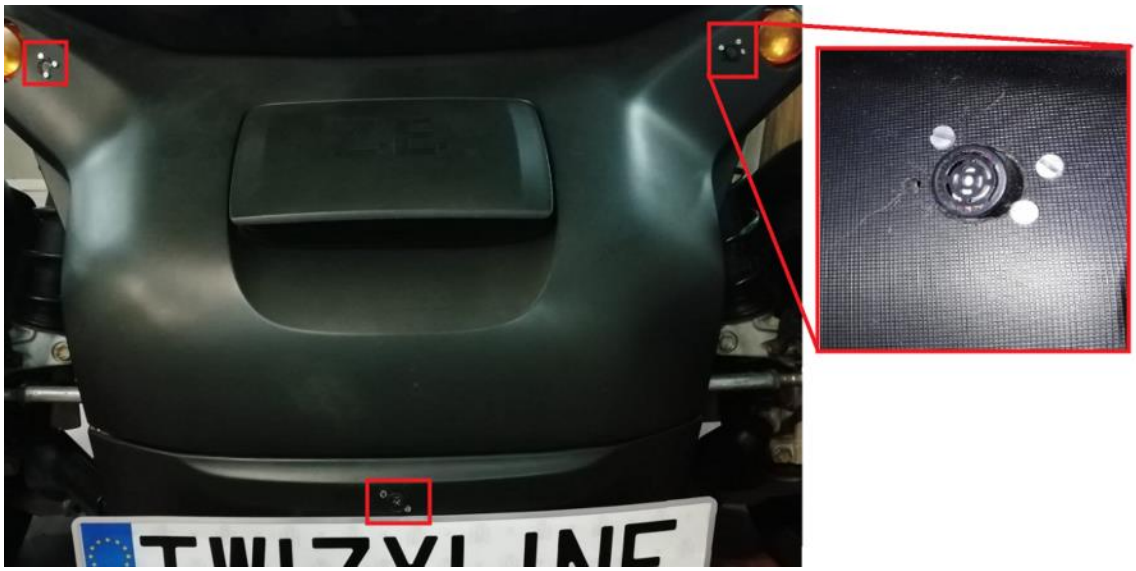


Figura 32. Sensores de ultrasonidos colocados en el morro delantero del vehículo.

Los sensores de ultrasonidos incluidos en la estructura del coche tienen un alcance de 6,45m en el punto máximo de su haz de detección. En la Figura 33 se puede ver cuál es el haz de detección de los sensores utilizados, teniendo en cuenta que son alimentados por 5V.

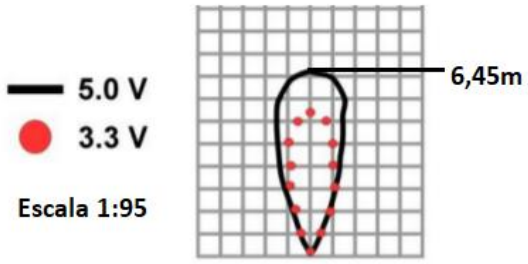


Figura 33. Haz de detección del sensor ultrasónico MB1010 [65].

3.2.2 Módulo de control y módulo de comunicaciones

El módulo de control será el encargado de procesar la información que se obtiene de los sensores. Por ello, se pensó en un lugar cercano a la parte delantera del coche donde se encuentran todos los sensores explicados en el apartado anterior. Por el otro lado, el módulo de comunicaciones, como su propio nombre dice, es el encargado de conectarse a internet y transmitir la información al servidor. Este módulo deberá poder comunicarse con el módulo de control para mandarle las órdenes pertinentes del servidor.

Tras analizar las diferentes partes del Twizy, y teniendo en cuenta que el módulo de comunicaciones estuviera cerca del de control para evitar usar un cableado muy largo, se decidió implantar los módulos en los recipientes situados en el salpicadero del coche En la Figura 34 se puede ver que el módulo de comunicaciones ha sido implantado en el contenedor izquierdo y el módulo de control en el contenedor derecho.

Figura 34. Zonas accesibles en el salpicadero para implantar nuestros módulos.

En el contenedor izquierdo está el conector OBD (*On Board Diagnosis*) que habilita la conexión con el bus CAN del coche. El módulo de comunicaciones no solo se conectará al OBD, sino que también llevará un módulo 4G, que permitirá conectarse a internet siempre que tenga cobertura. Además, el receptor GPS se conecta a este módulo también, de este modo, la zona derecha del salpicadero quedaría como muestra la Figura 35.

Figura 35. Zona derecha del salpicadero con los componentes que lleva incorporados.

En el contenedor derecho aparte del módulo de control, se decide añadir un hub USB para tener disponibles más puertos USB a los cuales podamos conectar los sensores. En concreto al hub irán conectados los sensores: magnético, RFID, ultrasonidos y acelerómetro (Figura 36).

Figura 36. Zona derecha del salpicadero, donde se observar de arriba a abajo: controladora EPOS4-70/15, módulo de control y hub de conexión para los sensores.

3.2.3 Hardware del motor

Para dotar al Twizy de dirección asistida se siguió como referencia el proyecto de Cartagena, previamente explicado en apartado 2.1.2, donde se integró un motor paso a paso que es capaz de girar el volante. La compañía Maxon Motor donó al proyecto un motor EC60 flat (brushless de100W de potencia) junto con el encoder MILE512 y la reductora planetario GP52C con relación 1:81, además de la controladora EPOS4-70/15(Figura 37).

Figura 37. A la izquierda podemos ver la controladora EPOS4 y a la derecha el motor EC60 flat (brushless) junto el encoder MILE512 y la reductora planetaria GP52C.

En el Anexo 3 se muestra cuál ha sido el conexionado entre la controladora y el motor. Ahora hablaremos sobre su posicionamiento dentro del Twizy. La controladora al tener que estar

relativamente cerca del motor y de la pila de 12V de la cual se alimenta se opta por situarla en el recipiente derecho del salpicadero junto al módulo de control al cual va conectado. El módulo de control se encargará de decir los parámetros característicos a los cuales el motor deberá funcionar lo cuales explicaremos después en el apartado de desarrollo software (3.3.2). En la Figura 36, se muestra cómo quedaría esa zona del vehículo.

El siguiente punto de interés es la posición del motor, que deberá estar cerca de la columna de dirección. Debemos tener en cuenta que para transmitir la potencia de giro del motor a la columna de dirección se necesitan dos engranajes los cuales se mandaron fabricar de acero. Además, se pensó en una estructura capaz de sujetar el motor al chasis del coche en la posición adecuada. En el Anexo 4, se muestra los planos del soporte y de los engranajes. El resultado final tras integrarlo en el vehículo es el mostrado en la Figura 38.

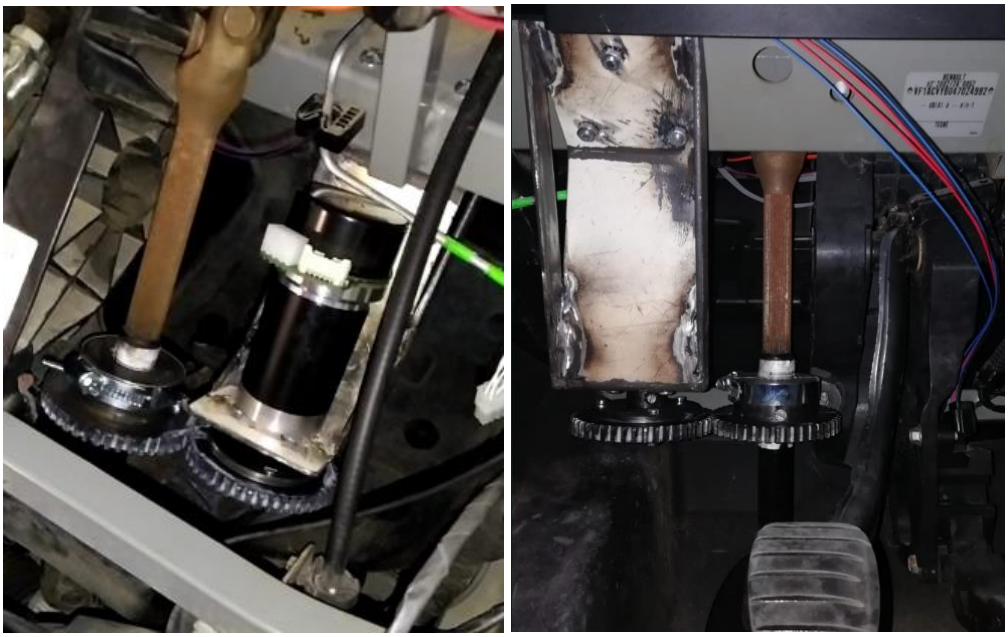


Figura 38. Implementación del conjunto del motor, los engranajes y el soporte en el Twizy. A la izquierda visión desde la parte frontal del Twizy, a la derecha vista desde dentro del coche.

Cabe destacar que para sujetar el engranaje que está en la columna de dirección se taladró ésta, por lo tanto, hay un tornillo que atraviesa la columna y evita que el engranaje de deslice hacia arriba o hacia abajo. El engranaje que está situado bajo el motor está sujeto al motor de tal forma que también evitamos que se deslice, para ello cortamos una pletina de metal redonda y la colocamos entre el engranaje y el tornillo de final del émbolo del motor (Figura 39).

Figura 39. Platina redonda metálica que evita que se deslice hacia abajo el engranaje.

3.3 Desarrollo Software

Tras haber explicado la instalación del hardware dentro del Twizy, ahora se explicará el desarrollo software llevado a cabo para poder integrar cada uno de los elementos y lograr que funcionen adecuadamente a las necesidades del proyecto. El software del módulo de comunicaciones ya fue desarrollado en su totalidad en el trabajo final de grado de Ignacio Royuela González [24] (a excepción del sistema de FOTA del cual ya se hablará en el capítulo 4), sin embargo, el software del módulo de control hay que completarlo para que funcione conjuntamente con los nuevos elementos.

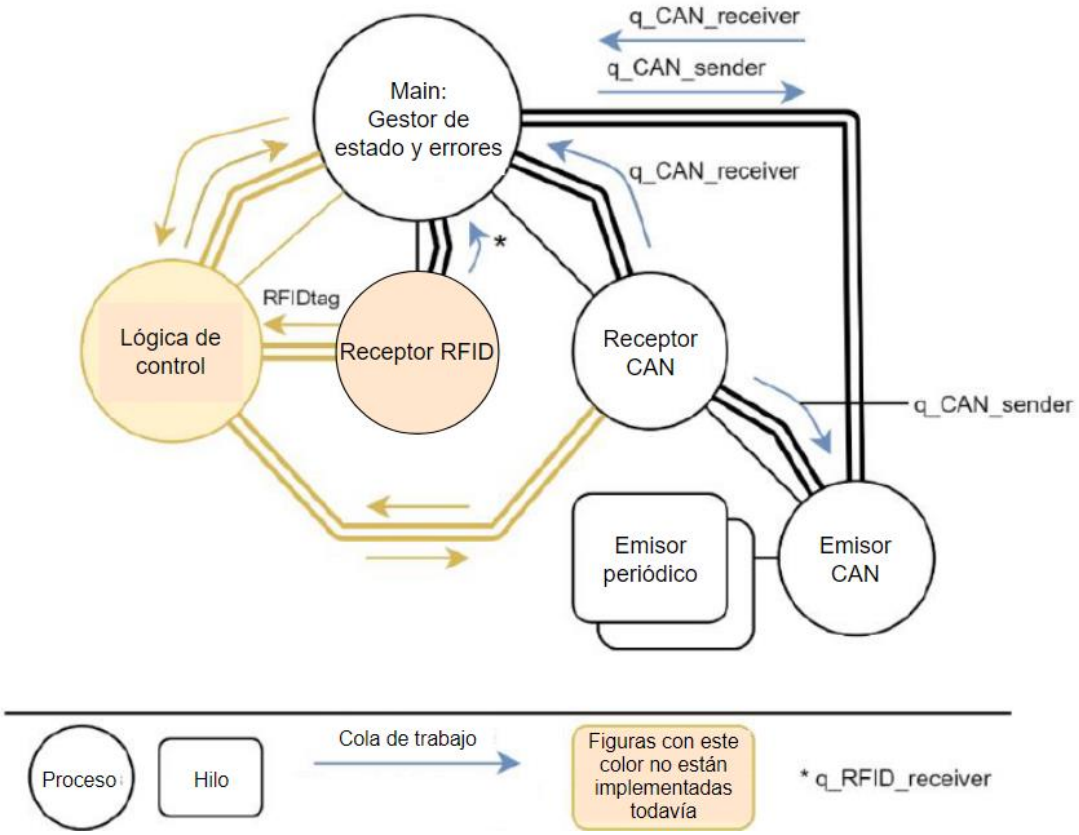


Figura 40. Esquema extraído del TFG de Ignacio Royuela donde se aprecia las partes ya creadas y las que se van a implementar en este TFM [24].

En la Figura 40 se observa el esquema del software que ya está implementado en el módulo de control y lo que queda por desarrollar que es lo realizado en este trabajo y se explicará más adelante. Además, se añadirán algunos procesos para controlar simultáneamente varios sensores a la vez. En concreto, nuestro objetivo será determinar qué tipo de operaciones deberá realizar el Twizy en modo autónomo.

Este apartado comenzará dando una explicación de cómo se estructurará el programa principal que se ejecutará en el módulo de control, seguidamente se mostrará cómo ha sido la puesta en marcha tanto de los sensores como del motor de Maxon y su funcionamiento, y finalmente, se expondrá el estudio realizado para poder realizar un simulador en MATLAB de la conducción autónoma del Twizy teniendo en cuenta las características del circuito que ha de seguir o de los dispositivos añadidos al vehículo. Todo el código desarrollado para este trabajo será almacenado en el repositorio GitHub del Grupo de Comunicaciones Ópticas (GCO) [66].

3.3.1 Arquitectura software

El programa principal del módulo de control será programado en Python, al igual que el del módulo de comunicaciones. El programa del módulo de control será llamado "mod-con.py" y deberá de cumplir las siguientes funciones:

- Establecer conexión con el módulo de comunicaciones.
- Determinar en qué estado se encuentra el vehículo.
- Procesar correctamente la información obtenida de los sensores.
- Diseñar un algoritmo de conducción tanto en sentido longitudinal como en sentido transversal, de acuerdo con los datos de percepción del entorno recibidos para que el Twizy pueda realizar una conducción autónoma.

De los 4 puntos antes mencionados, dos están ya desarrollados en el trabajo final de grado de Ignacio Royuela González [24]. Ambos módulos son capaces de comunicarse y además el módulo de control está programado para que fije en cuál de los 4 estados está el vehículo: Start-up (modo de arranque del sistema), modo normal (no autónomo), modo autónomo, y modo de fallos. Sobre este último punto habrá de realizarse algunos pequeños cambios para que funcione acorde a la nueva parte desarrollada, pero principalmente a continuación se explicará cómo se han programado los dos últimos puntos.

Para poder recibir y procesar la información de varios sensores simultáneamente se opta por una arquitectura de procesos hijo ejecutándose a la vez, usando la librería de multiproceso de Python (esta librería ya fue también usado tanto para el desarrollo del servidor [18] como para el desarrollo del módulo de comunicaciones). En la Figura 41 se muestra el esquema de procesos que se ejecutan dentro del programa "mod-con.py".

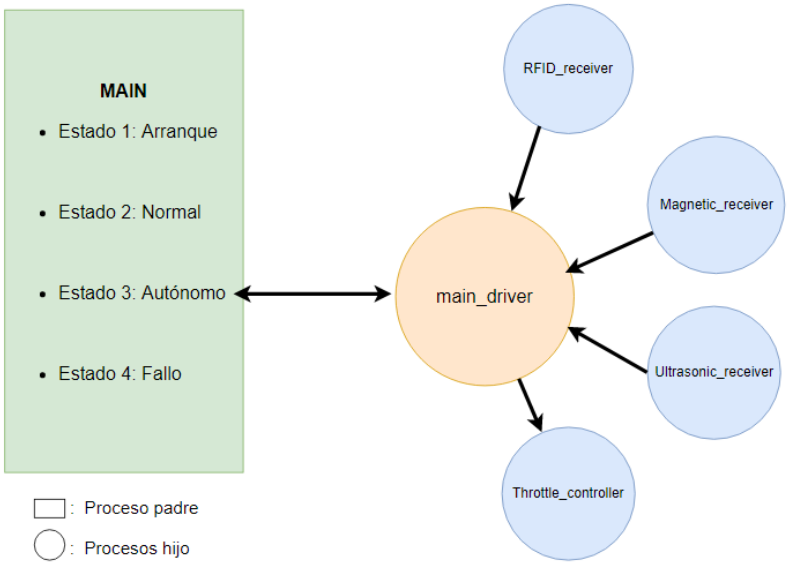


Figura 41. Esquema de procesos funcionando en el programa principal del módulo de control, las flechas indican el sentido de la comunicación.

El proceso hijo "main_driver" se encargará de toda la lógica de conducción una vez el Twizy entre en modo autónomo, que será cuando se cree este proceso. Al iniciarse este proceso dará lugar a otros 4 procesos hijo que se encargarán de cada uno de los 4 sensores indicados en la Figura 41. La finalidad de cada uno de estos 4 subprocesos hijo es la siguiente:

- "RFID_receiver": Procesar el ID del tag RFID adecuadamente (pasarlo a ASCII), y avisar al proceso padre "main_driver" que se ha leído un tag y cuál es su ID. El

proceso “main_driver” dependiendo del mensaje que haya recibido de la ruta que debe seguir, deberá prepararse para una bifurcación o para parar el coche.

- “Magnetic_receiver”: Su tarea será determinar si detecta la cinta magnética del suelo y decir en qué posición se encuentra el sensor con respecto a la cinta. El proceso “main_driver” deberá conseguir que el coche vaya por el centro de la banda en la recta o se desplace en sentido contrario a donde vaya realizar un giro para hacer mejor la curva.
- “Ultrasonic_receiver”: Deberá detectar los obstáculos que se encuentren dentro del rango de captación de estos sensores y avisar de la distancia a la que se encuentran. El proceso padre “main_diver” tendrá que determinar a qué distancia decide parar o considera peligroso que el coche siga en circulación ante una posible colisión.
- “Throttle_controller”: Su objetivo será simular la señal del pedal del acelerador y determinar a qué velocidad debe moverse el coche. Esa velocidad la determinará el algoritmo de control longitudinal que se encuentra en el proceso padre, “main_driver”.

El siguiente punto a tratar sería la lógica de conducción que el coche debería de seguir y que sería programada dentro del “main_driver” y cómo se comunica con el software de la controladora del motor. Antes de proceder a su explicación primero se explicará el software empleado para controlar los sensores y después se expondrá los pasos dados para la puesta a punto del software del motor y el estudio realizado en el simulador de MATLAB para determinar la lógica de conducción final en el “main_driver”

3.3.2 Software de los sensores

Se comenzará analizando el sensor magnético. Este sensor será controlado por un Arduino donde se ejecutará el código “sensor_magnetico.ino”. De este código cabe destacar que para obtener la información sobre a qué distancia está el coche de la banda se leen los siguientes pines:

- Pin 7 del sensor magnético: Proporciona una salida analógica de la posición del sensor respecto al centro de la banda. Se necesitará una conversión del valor de voltaje que nos ofrece a milímetros, para determinar en qué posición el sensor está leyendo la banda.
- Pin 3 del sensor magnético: Proporciona una salida digital que el 0 cuando detecta la banda y 1 si no la detecta.

La información obtenida por estos pines es leída por el proceso ‘Magnetic_receiver”, que a través de una cola de trabajo informará al “main_driver” sobre la posición del coche con respecto la banda y actuar conforme a ello.

Los siguientes sensores a explicar son los de ultrasonido. En este caso debemos tener en cuenta que vamos a recibir 3 datos de distancia, uno de cada sensor. Además, los sensores no pueden funcionar simultáneamente, dado que interferirían entre sí. En la Figura 42, vemos cómo están los sensores de ultrasonidos conectados entre sí, se ha seguido uno de los esquemas proporcionados por la hoja de especificaciones [65].

Para comenzar a funcionar, primero deben estar conectados a 5 V y después se debe de enviar un pulso, es decir poner en alta al pin ‘RX’ del primer sensor (el pulso debe estar en alta en un rango de entre 20µs y 48ms). Esto hará que el sensor envíe un eco y cuando reciba el eco de vuelta completo se enviará un pulso al pin ‘RX’ del siguiente sensor. De esta manera nunca

están funcionando los sensores simultáneamente. Esto evita que un eco de vuelta sea recibido por el sensor que no debe y haya cálculos de distancia erróneos.

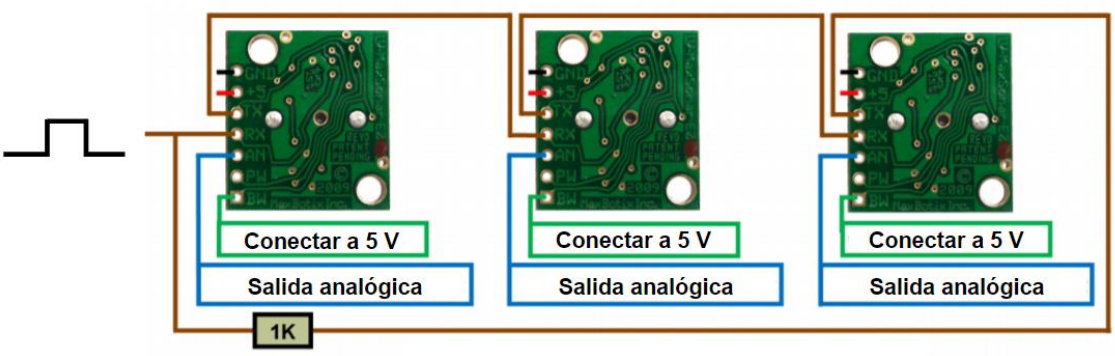


Figura 42. Esquema del conexionado de los sensores de ultrasonidos entre ellos.[65]

El proceso "Ultrasonic_receiver" se encargará de recibir una cadena de texto con estos datos y los separará en 3 para identificar el dato obtenido por cada sensor, para procesarlo adecuadamente. Recordemos que tendremos que poner un límite, que se fijará en el "main_driver", a partir del cual es coche comenzará a parar con el fin de evitar cualquier colisión.

A diferencia de los sensores previos, para el sensor RFID no se ha empleado ningún Arduino, sino que está conectado al módulo de control por un puerto serie mediante un conector RS-232. Una vez esa información es obtenida por el proceso "RFID _receiver", primero es pasada a hexadecimal para obtener el id de la tarjeta RFID leída, después se calcula el 'checksum' (algoritmo añadido a continuación del id que se lee para saber si la información es correcta) que si es correcto se pasa esta información por una cola de trabajo al proceso de control "main_driver" para que actúe conforme al id que ha recibido.

Finalmente se explicará el acelerador que será el elemento sobre el que actuaremos para modificar el control longitudinal. Un Arduino controlará el relé que hace que pasemos a controlar la señal del pedal del acelerador cuando el coche entra en modo automático y la potencia con la que queremos pisar el pedal (Figura 30). Se puede encontrar en GitHub [66] el código en el fichero "Accelerador_Arduino.ino"

Desde el proceso de "Throttle_controller" se indicará las operaciones que debe realizar el acelerador. Para gestionar el control longitudinal se hará uso de la librería PID de Python. Dentro de ese PID se meterán las constates Kp, Ki, Kd y el rango de valores de salida posibles. Usaremos este PID en dos situaciones:

- Enviar la señal al acelerador necesaria, para darle simular más o menos presión sobre el pedal del acelerador para mantener la velocidad a la prefijada.
- Regular la velocidad una vez se detecte un obstáculo con los sensores de ultrasonidos, ya que el Twizy deberá disminuir su velocidad hasta pararse completamente guardando una distancia de seguridad.

3.3.3 Software de los elementos del motor

El primer aspecto a tratar será la instalación del firmware en la controladora. Maxon proporciona en su página el programa EPOS Studio. Este software nos permitirá definir los parámetros del firmware de la EPOS4-70/15, conforme al motor que se esté controlando. Comenzaremos conectando la controladora a nuestro portátil e iniciaremos el procedimiento para instalar el firmware.

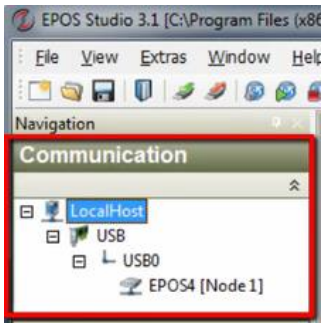


Figura 43. El sistema reconoce la EPOS4.

Una vez arranquemos el EPOS Studio deberemos crear una carpeta del proyecto, y deberá de reconocernos la controladora conectada (Figura 43). A partir de este momento, el motor deberá ya estar conectado correctamente según se especifica en el Anexo 3. El siguiente paso será acceder a *Wizards→Startup*, esta ventana nos permitirá configurar los parámetros iniciales de la controladora. A continuación, se mostrará una lista de los parámetros a modificar (aquellos que no aparezcan en la lista se dejará el valor por defecto) [67] [68]:

- Tipo de motor: *maxon DC motor*.
- Tensión nominal: *12V*.
- Constante de tiempo térmica del motor: *90 segundos*.
- Número de pulsos por vuelta: *4096 pulsos/vuelta.*
- Velocidad máxima permitida: *6000rpm*.

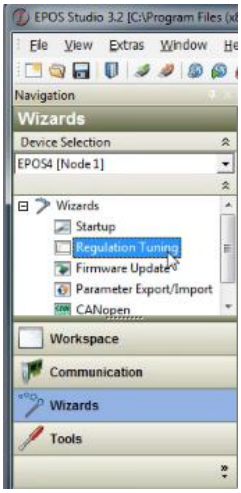


Figura 44. Menú de opciones en EPOS Studio.

El último paso para tener listo el firmware será la puesta a punto del motor ("*Regulation Tuning*", Figura 44), gracias a esta funcionalidad del EPOS4 Studio podemos optimizar el funcionamiento del motor, en concreto, se envía una señal de prueba para optimizar la respuesta eléctrica y la mecánica (velocidad y posición) [69]. Tras seguir estos pasos se tendrá la controladora preparada para enviar las órdenes pertinentes al motor.

El segundo aspecto a analizar es el programa donde se ejecuta la lógica de funcionamiento del motor. Maxon proporciona en su página oficial librerías para trabajar con sus motores para código C++. Esta librería ofrece una serie de comandos para funcionar el motor en diferentes modos. En la Figura 45, se muestra los modos por los que según la lógica de funcionamiento que se ha programado se encuentra el motor.

Figura 45. Estados por los que pasa el motor mientras está en funcionamiento.

A continuación, se explicarán cada uno de estos modos y los comandos utilizados en cada uno de ellos. Se comenzará explicando el primer y último estado, que como su nombre indican, sirven para establecer conexión o desactivarla con la controladora. En concreto, los dos comandos empleados son [70]:

- *VCS_OpenDevice*: Abre el puerto para enviar y recibir comandos de la EPOS4. Se puede utilizar una interfaz CAN, USB o RS232, en este caso, se emplea la interfaz USB que conecta la controladora con el módulo de control.
- *VCS_CloseDevice*: Cierra el puerto que se estaba usando para la comunicación entre la controladora y el módulo de control.

El modo denominado "*Machine State*" (máquina de estados), describe el estado en el que se encuentra la controladora y dependiendo de este estado determina los comandos que serán aceptados. En este caso, la controlará mientras esté en funcionamiento estará en dos estados, el estado activo o el estado desactivado, y los comandos empleados son [70][71]:

- *VCS_SetEnableState*: Cambia el estado de la controladora a activo, y permite pasar a un modo de funcionamiento y a aceptar sus respectivos comandos.
- *VCS_SetDisableState*: Cambia el estado del dispositivo a desactivado.

Antes de adentrarnos en el modo donde hacemos girar el motor y por consiguiente el volante, se debe tener en cuenta que se necesita un punto de referencia de posición del motor a partir del cual se determina hacia dónde y cuánto gira. Por defecto la controladora toma como punto de referencia del motor (también nos referiremos a ese punto como cero absoluto) la posición física en la que se encuentra cuando se abre el puerto de conexión entre la EPOS4 y el módulo de control (*VCS_OpenDevice*). Lo ideal sería que ese cero absoluto estuviera centrado o en uno de los extremos, es decir en un punto controlado, que nos permita saber dónde está realmente el punto de referencia. El problema es que cuando un usuario deje el coche en uno de nuestros parkings no se sabe en qué posición está el volante, es decir, cuando entre en modo autónomo ese cero se encuentra situado en una posición aleatoria.

El objetivo del modo "*Homing*" es conseguir definir el punto de referencia que se quiera y que servirá para modos posteriores en los que el motor entre en funcionamiento. Se pueden emplear varios métodos en este modo, la manera en la que está implementado es el método homing 7, llamado "*Home Switch Possitive Speed & Index*". Esta técnica lo que hace es girar hacia la derecha (sentido positivo), cuando el sistema llegue al tope (en este caso cuando el volante no pueda girar más), el sistema detectará un aumento exponencial de la corriente al intentar tener más potencia para poder girar. Cuando se supere el umbral de corriente máximo que se defina en el modo homing estaremos en uno de los extremos.

Después, se decidió que el punto de referencia fuera con las ruedas rectas, de tal manera que se debe dar 1,4 vueltas de volante desde el extremo (recordemos que el Twizy puede dar 2,8 vueltas de volante) para situar las ruedas centradas. Al terminar se guarda la posición en la que haya quedado el motor como nuevo punto de referencia. En la Figura 46 se puede ver en imágenes el procedimiento explicado seguido en el modo homing. En este procedimiento no hace falta detectar el otro extremo, ya que como ya sabemos de antemano cuales son las vueltas máximas de volante podemos limitar por software el número máximo de pasos que puede dar a izquierda

y derecha según el número de vueltas que admite el Twizy (1,4 vueltas a izquierda y a derecha desde el punto de referencia). Los comandos principales utilizados para este modo son:

- *VCS_ActivateHomingMode*: Se entra en modo homing y se habilita los comandos relacionado con este modo.
- *VCS_SetHomingParameter*: Se define las características del movimiento en este modo como la aceleración, la velocidad o el umbral de corriente máximo.
- *VCS_FindHome*: Comienza la búsqueda del tope y como argumento de entrada a este comando se indica qué método de Homing se utiliza.
- *VCS_WaitForHomingAttained*: Espera hasta que el modo homing termine o cuando el temporizador que se indica como argumento de entrada a esta función venza.
- *VCS_DefinePosition*: Se determina un nuevo punto de referencia.

Figura 46. Secuencia de fotos ilustrando el modo homing. De izquierda a derecha: orientación de las ruedas tras dejar un usuario el Twizy en el punto de aparcamiento; orientación de las ruedas una vez a entrado el Twizy en modo autónomo y ha llegado al tope girando a la izquierda; orientación final de las ruedas centradas cuando termina el modo homing.

Por último, quedaría por explicar el "*Profile Position Mode*". La librería proporcionada por Maxon nos ofrece dos modos de poder girar el motor y cuya elección depende de las necesidades del proyecto. Un modo sería el "*Profile Velocity Mode*", ese modo permite girar al motor con la velocidad y aceleración seleccionada continuamente el tiempo que se estime oportuno. Otro modo es el "*Profile Position Mode*" este modo nos permite decir, además de velocidad y aceleración, la posición a la que se quiere que gire el motor. Debido a las características del proyecto que se está llevando a cabo y al algoritmo de guiado elegido como se explicará más adelante, este segundo método nos resulta más útil para decir exactamente a qué posición queremos que gire.

Antes de explicar qué funciones se han utilizado en este modo cabe destacar las unidades empleadas con tal de saber qué argumentos introducir a cada función. Para la velocidad se emplea revoluciones por minuto (rpm), para la aceleración m/s^2, y para la posición número pasos. Como ya se ha comentado el motor da 4096 pulsos por vuelta, el encoder dispone de dos canales con dos cambios de flanco por cada periodo, y la reductora es de relación 1:81, por lo tanto, una vuelta de volante es de: 4096*2*2*81=1.327.104 pasos. Esto es verificado empíricamente introduciendo este valor en el motor y dando el volante una vuelta exacta. Los comandos utilizados es el modo de "*Profile Position Mode*" son:

- *VCS_ActivateProfilePositionMode*: Cambia el modo de funcionamiento a "*Profile Position Mode*", y habilita usar las funciones de este modo.

- *VCS_SetPositionProfile*: Define los parámetros de este modo de funcionamiento, la velocidad, la aceleración y la deceleración.
- *VCS_MoveToPosition*: El motor gira a la posición indicada en los argumentos de entrada. Además, esta función nos ofrece poder hacer el giro absoluto (con respecto al cero obtenido del modo homing) o el giro relativo (con respecto a la última posición). En este caso será de mayor utilidad usar el giro absoluto para obtener fácilmente la posición del volante (los pasos indicarán cuánto de lejos o de cerca estés de tener las ruedas rectas).
  También esta función permite decidir si hace el giro inmediatamente o espera a que el motor se desplace al punto donde se le haya indicado previamente. En este caso se seleccionará que haga el giro inmediatamente, ya que para la conducción autónoma será imprescindible que siga en todo momento las posiciones que indique el algoritmo de guiado, evitando de esta manera retardos en esperar a que llegue a la posición previamente indicada.
- *VCS_GetPositionIs*: Devuelve la posición actual del motor en el momento en el que se llama a la función.
- *VCS_GetVelocityIs*: Devuelve la velocidad del motor en revoluciones por minuto (rpm) en el momento en el que se llama a la función
- *VCS_GetTargetPosition*: Esta función permite obtener la posición que ha sido ordenado al motor a girar. Cabe destacar que a diferencia de *VCS_GetPositionIs*, cuando se mande girar 1,4 vueltas y se llama a *VCS_GetTargetPosition* siempre devolverá el mismo valor, mientras que si mientras gira se llama a *VCS_GetPositionIs* te dará todos los valores intermedios por los que vaya pasando el motor hasta llegar a 1,4 vueltas. Al final del giro, deberá dar el mismo valor ambas funciones.
- *VCS_HaltPositionMovement*: Para el movimiento del motor tan rápido como indique el parámetro de deceleración introducido en *VCS_SetPositionProfile*.

Como se mencionó previamente la librería que proporcionaba Maxon era para código C++, por lo tanto, todo el código respecto al funcionamiento de los modos motor antes explicado fue programado en C++, en un fichero llamado “HelloEposCmd.cpp”. La problemática aquí era cómo podía el fichero Python que se ejecutaba en el módulo de control (“mod-con.py”) acceder a los comandos del motor para poder girar el volante. Este problema se consiguió solucionar mediante la creación de una librería propia en Python (“shm.h”) y empleando memoria compartida.

El método de la memoria compartida permite a varios programas acceder al mismo espacio de memoria utilizando un semáforo para restringir o permitir el acceso a recursos compartidos en un entorno donde varios procesos operan simultáneamente. Se crearán un total de 4 memorias compartidas para que nuestro programa de Python y C++ se pueden comunicar:

- Memorias de escritura de “mod-con.py”: Una memoria para los pasos que debe girar el motor y otra para la velocidad. Estas 2 memorias servirán para enviar los parámetros objetivo, es decir, aquellos que se quieren conseguir.
- Memorias de escritura “HelloEposCmd.cpp”: Una memoria para los pasos actuales que está girando el motor y otra para la velocidad a la que se mueve. Estas otras dos memorias para informar al programa de Python en qué punto se encuentra del giro y a qué velocidad real está girando.

En C++ emplearemos la librería “shm.h” y en Python con tal de usar el mismo procedimiento que en “shm.h”, se creó una librería propia “motor” que permite acceder a las memorias compartidas que se creen. La librería de Python se programó utilizando el software

SWIG (*Simplified Wrapper and Interface Generator*) que conecta programas escritos en C o C++ con programas de alto nivel como Python, PHP, Javascript o Perl [72]. En esta librería creada se incluyen 6 funciones: dos para inicializar las dos memorias donde escribe el fichero Python, dos funciones para enviar datos a las memorias previamente inicializadas y dos funciones para leer de las memorias donde escribe el fichero de C++ (Figura 47).

```
# Librería motor importada
import motor
# Inicializamos la memoria para escribir en ella.
ID1=motor.keyRPM()
ID2=motor.keyDEG()
# Enviamos los rpm o los pasos objetivo al motor
motor.rpm_launcher([rpm],ID1)
motor.deg_launcher([pasos],ID2)
# Recibimos información de lo rpm o pasos actuales del motor
RPM=motor.rpm_rx()
DEG=motor.deg_rx()
```

Figura 47. Funciones disponibles con la librería 'motor' creada para compartir un espacio de memoria entre dos códigos con distinto lenguaje de programación.

Por último, cabe comentar que las funciones del motor ofrecidas por la librería de Python creada serán usadas dentro del proceso “main_driver”, donde se comenzará a ejecutar el fichero “HelloEposCmd.cpp”, después se enviará la posición y la velocidad a la que debe girar el motor, y cuando se salga de este proceso, es decir, cuando pase de modo autónomo a manual, se cerrará el programa en C++.

3.3.4 Modelo cinemático simulado y algoritmos de guiado

Habiendo llegado a este punto del proyecto, el siguiente aspecto a desarrollar era determinar un algoritmo de guiado suficientemente preciso para que el Twizy pueda seguir adecuadamente la banda magnética. Antes de probar los algoritmos directamente sobre el Twizy y que se explicarán a continuación, se decide crear un simulador en MATLAB. Este software se creará teniendo en cuenta el mayor número posible de parámetros físicos conocidos del Twizy y del motor de Maxon, de tal manera que la estimación dada por el algoritmo de MATLAB sea lo más parecida a lo que ocurra en la realidad cuando se implemente el algoritmo en el coche real. El código MATLAB que se ha desarrollado en esta parte se puede encontrar en la página de GitHub del Grupo de Comunicaciones Ópticas al igual que todo el código desarrollado en este proyecto [66].

El primer problema que se debía solventar era como se puede modelar el movimiento del Twizy de manera que se pudiera programar en MATLAB. Si analizamos los giros del coche, la dirección de las ruedas de los vehículos sigue la geometría de Ackerman [30]. Según se ve en la Figura 48, este axioma enuncia que las rectas perpendiculares al centro de la rueda cruzan todas en el mismo punto y ese punto es el centro de giro de cada rueda por separado.

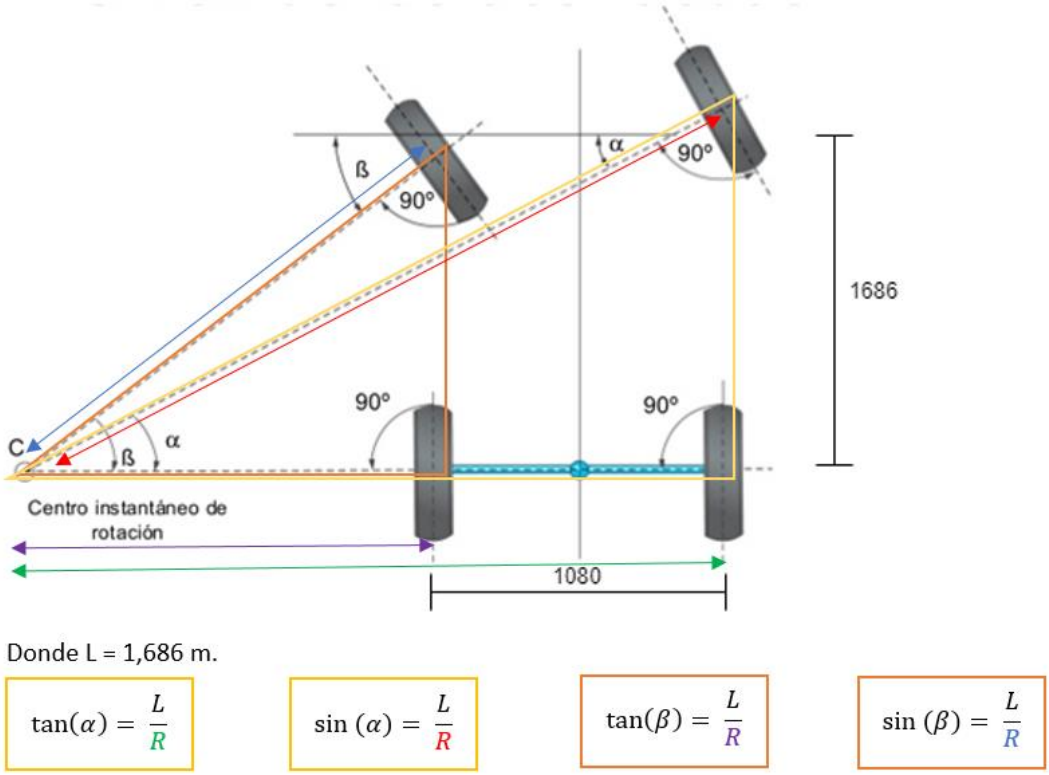


Figura 48. Diagrama ilustrando la geometría de Ackerman y cálculo del radio de giro de cada rueda.

Al tener cada rueda un ángulo y una distancia al centro de rotación distintos, la circunferencia que traza cada rueda será diferente. Según las especificaciones técnicas del Renault Twizy (Anexo 1), el coche tiene un radio de giro de 3,4m. Por lo tanto, calcularemos los radios de giro según se indica en la Figura 48 de cada rueda para saber a qué radio se refiere y lo compararemos con el radio de giro obtenido empíricamente del coche tomando como giro hacia la izquierda y teniendo en cuenta los valores del ángulo máximo de giro de la rueda delantera derecha que es 45º (ß) y en la rueda delantera izquierda que es 30º (α):

- Rueda delantera derecha: mediante el cálculo se obtiene 3,372 m de radio de giro ($R = \frac{1,686}{sen(\alpha)}$), mientras que empíricamente 3,41 m.
- Rueda delantera izquierda: mediante el cálculo se obtiene 2,384 m de radio de giro ($R = \frac{1,686}{sen(ß)}$), mientras que empíricamente 2.325 m.
- Rueda trasera derecha: mediante el cálculo se obtiene 2,92 m de radio de giro ($R = \frac{1,686}{tan(\alpha)}$), mientras que empíricamente 2,935 m.
- Rueda trasera izquierda: mediante el cálculo se obtiene 1,686 m de radio de giro ($R = \frac{1,686}{tan(ß)}$), mientras que empíricamente 1,61 m.

De este estudio sacamos dos conclusiones: la geometría de Ackerman se cumple teniendo en cuenta posibles pequeñas desviaciones en las pruebas empíricas realizadas; el radio de giro de 3,4 m proporcionado por el fabricante se refiere al radio de giro de la rueda delantera contraria al sentido del giro. Esto tendría sentido ya que el radio mayor te dará el espacio que necesita el coche para realizar un giro sin colisionar con ningún obstáculo.

Dado que las velocidades a las que nos vamos a mover son en general muy bajas (apenas 5 km por hora), es posible realizar la simplificación de que no existen fuerzas dinámicas en el movimiento de nuestro vehículo, y solo es necesario caracterizarlo desde un punto de vista

cinemático. Este hecho tiene una segunda ventaja, es posible aplicar el modelo cinemático de la bicicleta, que será mucho más sencillo de implementar, sin pérdida de exactitud. Este modelo, tal y como se muestra en la Figura 49, consiste en suponer que el vehículo tiene únicamente dos ruedas una entre las dos ruedas delanteras y otra entra las dos ruedas traseras. El nuevo ángulo δ se calcula: $\delta = \tan^{-1} \frac{1686}{1686+{}^{1080}/_{2}} = 37{,}14$ grados. Por lo tanto, el ángulo máximo de giro para el modelo de bicicleta son aproximadamente 37,14º (equivalente a 0,6545 radianes). El radio de giro en este caso resultaría: $R = \frac{1686}{sin(\delta)} = 2{,}769m$.

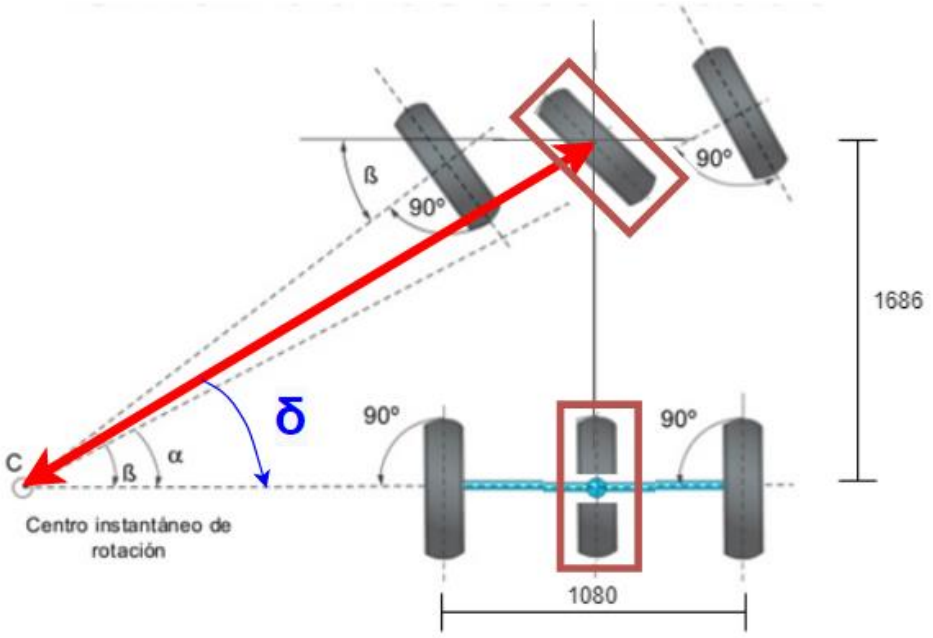


Figura 49. Esquema del modelo cinemático de una bicicleta.

El siguiente estudio que se realizó se trató de la implementación del método de Stanley. Fue utilizado por la universidad de Stanford en el DARPA (*Defense Advanced Research Projects Agency*) Grand Challenge en 2005, se le asignó el nombre de Stanley al método debido a que era el nombre del robot donde se implementó [30].

El objetivo de este algoritmo es conseguir el seguimiento de una ruta de un vehículo no tripulado, pero asegurando que independientemente de la velocidad del vehículo el tiempo de convergencia a la ruta es constante dado un cierto offset y ángulo de entrada inicial. Este método necesita dos parámetros de entrada: el primero es el error entre la orientación del trayecto y la orientación del vehículo, y el segundo denominado “cros-track error”, que se define como la distancia desde un punto de referencia (puede ser delantero, trasero o el centro de gravedad) hasta un punto cercano de la ruta establecida (Figura 50) [30].

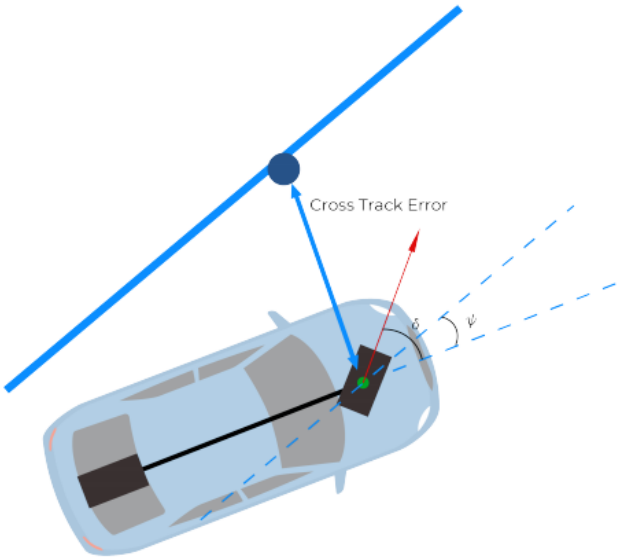


Figura 50. Algoritmo de Stanley, donde los dos parámetros característicos son: "cross Track Error" dibujado con una flecha azul y el ángulo ψ la diferencia de orientación entre la trayectoria y el vehículo [30].

En concreto debido a las características del proyecto solo contaremos con el primer parámetro. El segundo parámetro sería factible implementarlo teniendo una cámara o un Lidar capaz de visualizar la línea, objetos que hasta este punto no tiene incorporadas. Por lo tanto, uno de los objetivos de nuestra simulación es comprobar que usando solo el cross Track error es posible hacer seguir al vehículo la ruta, alineando el vehículo en cada bucle de control empezando el trayecto con el coche sobre la trayectoria preestablecida.

El algoritmo de control lateral se hace en dos fases. En la primera fase, que corresponde al algoritmo de Stanley propiamente dicho, se debería calcular el error entre el ángulo del eje del vehículo con respecto del ángulo de la ruta en el punto de referencia. A este primer término se le llamará $\delta_{orientación}$, y se recalculará en cada bucle como: $\delta_{orientación} = \theta_{vehículo} - \theta_{giro}$, siendo $\theta_{vehículo}$ la orientación del vehículo y θ_{giro} el ángulo de giro del trayecto. Teniendo en cuenta esto el ángulo que debería girar el coche sería: $\delta_{objetivo} = \delta_{orientación} + tan^{-1}(\frac{K_{stanley} * x}{v})$; siendo $K_{stanley}$ una constante de ganancia controlada, x la distancia del vehículo al trayecto y v la velocidad. Dos comentarios que añadir respecto a este método:

- Cuanto menor sea el valor del parámetro $K_{stanley}$, el control será más lento, pero en cambio se gana estabilidad y viceversa.
- El ángulo final de giro ($\delta_{objetivo}$) debe ser acotado entre dos límites que serán ±37,14º, que es el ángulo máximo de giro según lo analizado tras emplear el modelo cinemático de la bicicleta.

En la segunda fase, se aplica un PID sobre el control del motor Maxon para ganar estabilidad. Antes de proceder a realizar el simulador, se realizó una lectura y después comprensión de los controladores PID (proporcional, integrador y derivador). El control PID se trata de un algoritmo de control realimentado donde determinando las constantes proporcional, integral y derivativo adecuadas se logra una respuesta estable y robusta del sistema. El significado de cada constante dentro de cómo afecta al sistema [73]:

- El controlador proporcional (Kp) da forma a la curva de respuesta de la variable controlada. Produce un valor mayor cuanto mayor sea el error.

- El control integral (Ki) disminuye el error cuando el sistema se encuentre en estado estacionario, eliminándolo en determinados casos. Produce más salida cuanto más perdure el error.
- El control derivativo (Kd) mejora la estabilidad, aumenta la rapidez de la respuesta, prediciendo con antelación el valor del cambio del error y, frecuentemente, reduce el error ejecutando previamente las correcciones oportunas. Proporciona más salida cuanto más rápidamente se produce el error.

Esos parámetros a los que denominaremos Kp, Kd, y Ki se pueden ajustar siguiendo uno de estos 4 métodos de sintonización (buscar el valor óptimo): métodos de prueba y error, métodos basados en experimentos, métodos analíticos basados en modelos, métodos de sintonía automática [73]. Uno de los principales objetivos del simulador será hallar cuales son las constantes Kp y Kd (se supone Ki igual a 0).

La expresión matemática que describe el control PID se muestra en la Figura 51. Donde Kp es la constante de proporcionalidad, Ki es Kp entre el tiempo integral, e(t) el error cometido cuando el sistema se encuentre en régimen estacionario y Kd siendo Kp por el tiempo derivativo.

En nuestro caso, el PID se utilizará para modular la velocidad de giro del motor. Cuando el error encontrado entre el giro de las ruedas predicho por Stanley y el giro de ruedas conseguido en el bucle anterior sea alto, se incrementará la velocidad de giro para intentar corregir rápidamente el error, mientras que cuando ese error sea bajo, se intentará que la aproximación se haga de forma lenta para lograr la convergencia del algoritmo.

$$m(t) = \underbrace{K_p * e(t)}_{P} + \underbrace{\frac{K_p}{T_i}\int_0^t e\,(t)dt}_{I} + \underbrace{K_p * T_d * \left(\frac{de(t)}{dt}\right)}_{D}$$

Figura 51. Ecuación de implementación del algoritmo de control PID.[73]

3.3.5 Simulador MATLAB

Como ya se ha comentado en secciones previas, el simulador se trata de un programa MATLAB capaz de replicar el entorno y condiciones de conducción autónoma que se dan en este proyecto. Este programa integra el estudio teórico realizado en el apartado anterior y una vez se lleguen a unos resultados que confirmen el correcto seguimiento del coche por la línea, se implementará en Python dentro del proceso "main_driver" donde se alberga la lógica de conducción autónoma.

Cabe destacar que el simulador funcionará bajo las siguientes premisas:

- El vehículo tiene un sensor de movimiento que refresca su posición con un periodo de 'T'.
- El vehículo se mueve a velocidad constante 'v'.
- El movimiento del vehículo se puede describir con describir con ecuaciones cinemáticas (no hay derrapes ni similares).
- El modelo del vehículo está basado en el modelo de bicicleta visto en la sección anterior.
- El sensor devuelve la distancia del vehículo con respecto a la perpendicular de la curva, sino que mide la distancia desde la posición del vehículo teniendo en cuenta su ángulo de giro actual, al igual que lo hará el sensor real.

Lo primero que hace el programa es cargar una serie de ficheros Matlab donde se definen las características del sensor, simulador, motor y vehículo (en el Anexo 5 se muestra el contenido de los ficheros de configuración de estos elementos con sus respectivos parámetros característicos empleados en la simulación). Antes de comenzar la ejecución, el programa principal denominado "start.m" habilita dos modos de simulación:

- Única simulación: El usuario introduce los parámetros de: velocidad, radio de giro, Kp, Kd y K de Stanley.
- Barrido paramétrico: El usuario introduce la velocidad y el radio, y además a diferencia del anterior método se da un rango de posibles Kp, Kd y K de Stanley para buscar la combinación de parámetros óptima.

En ambos tipos de simulación el usuario podrá definir la ruta que debe seguir. Se puede definir rectas con su longitud, curvas con su radio de giro y el tag correspondiente que indicará que hay una curva a una determinada distancia, distancia que también se puede configurar. Para realizar las simulaciones y verificar la robustez del software desarrollado se hará las pruebas sobre un circuito con una trayectoria cambiante (varias curvas y rectas), en concreto la ruta es la mostrada en la Figura 52.

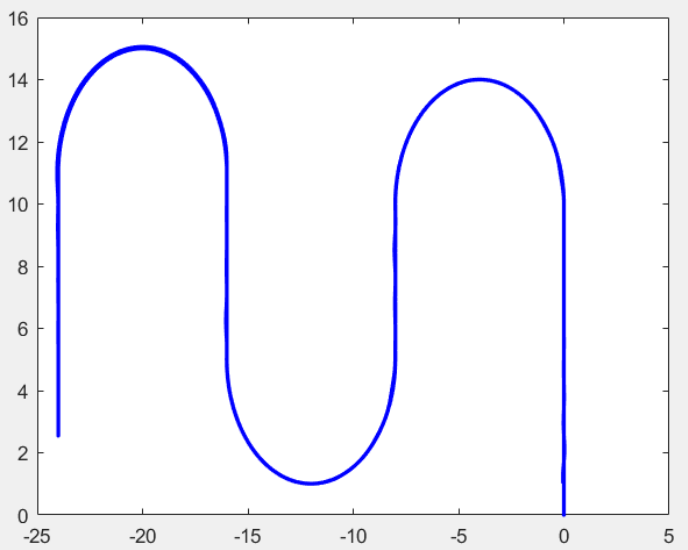


Figura 52. Ruta que debe realizar el Renault Twizy en las simulaciones.

Por último, del conjunto de fichero ".m" que conforman el simulador cabe también mencionar los ficheros "control_stanley.m" y "PID_control_lateral.m", donde se implementa los algoritmos de control y seguimiento de banda estudiados en la sección anterior.

En la Figura 53, se puede observar cómo se hace una pequeña modificación a la expresión dada por el algoritmo de Stanley, añadiéndole un segundo término. Este segundo término se añadió para mejorar el seguimiento de la banda por el coche en las curvas, como se aprecia en el código sobre si tag igual a 1 (es decir viene una curva) se suma ese segundo término. También decir con respecto a la curva que antes de encarar una curva, una vez lee el RFID que ordena giro, el Twizy deberá desplazarse unos milímetros hacia el lado contrario a la curva para realizar mejor la curva.

```
function delta_obj=control_stanley(distancia,delta_max,v,L,radius,tag)

global K_STANLEY;

% Algoritmo de stanley
delta_obj=atan(K_STANLEY*distancia/v)+atan(L/radius)*tag;
if(abs(delta_obj)>delta_max)
    delta_obj=sign(distancia)*delta_max;
end
```

Figura 53. Código del fichero "control_stanley.m".

Por otro lado, en la Figura 54 vemos el código que implementa el algoritmo PID. Si lo comparamos con la Figura 51 (expresión del PID), se aprecia que en este caso no hay variaciones. Se denomina error a e(t), 'error_anterior' a la derivada de e(t) en el tiempo y 'ERROR_ACUMULADO' a la integral de e(t) en todo el tiempo que es sistema esté operando.

```
function [rpm, sentido_giro]=PID_control_lateral(error,error_anterior,rpm_max)

global KP;
global KD;
global KI;
global ERROR_ACUMULADO;

ERROR_ACUMULADO = ERROR_ACUMULADO + error;
control_PID = KP * error + KD * error_anterior + KI * ERROR_ACUMULADO;

%rpm=rpm_max*abs(control_PID);
rpm=abs(control_PID);
if rpm>rpm_max
    rpm=rpm_max;
end
if(control_PID>0)
    sentido_giro = 1;
else
    if(control_PID < 0)
        sentido_giro = -1;
    else
        sentido_giro = 0;
    end
end
```

Figura 54. Código de "PID_control_lateral.m".

Se comenzará utilizando el segundo modo de simulación. Este barrido paramétrico permitirá obtener cuales son los valores de Kp, Kd, y K de Stanley adecuados para seguir la banda magnética. Fijaremos la velocidad constante en 3 km/h y el radio de giro teniendo en cuenta que el radio de giro del Twizy visto en el apartado anterior es de 3.4 m, se pondrá un radio de giro de 4 m las curvas que se introduzcan en la ruta. Para cada K de Stanley se calculará los resultados obtenidos con todas las combinaciones de Kp y Kd posibles dentro del rango definido (supone una alta carga computacional). Tras realizar varias simulaciones se llega a la conclusión que los rangos idóneos de simulación serán:

- Kp=[0:1000:30000], de 0 a 30000 en pasos de 1000.
- Kd=[0:1000:30000], de 0 a 30000 en pasos de 1000.

- K de Stanley=[3,8:0,2:8], de 3,8 a 8 en pasos de 0,2.

El barrido paramétrico nos arroja una serie de ficheros de datos ".mat", donde se genera uno para cada valor de K de Stanley analizado. En cada fichero ".mat" se guarda el valor máximo de la distancia entre el coche y la banda para cada combinación de Kp y Kd con la K de Stanley que se esté trabajando. Se va a representar los datos de cada fichero de datos para examinar en qué simulaciones qué combinación de parámetros resulta que la distancia máxima entre el vehículo y la banda se encuentran por debajo de 0,085 m, que es la distancia máxima a la que puede el sensor magnético detectar la banda magnética. Para una mejor visualización se eliminan todos aquellos resultados por encima de 0,15m, se normaliza a 0,085 y se representa en escala logarítmica, incluyendo un corte en el plano 0 para observar los valores por debajo de ese plano.

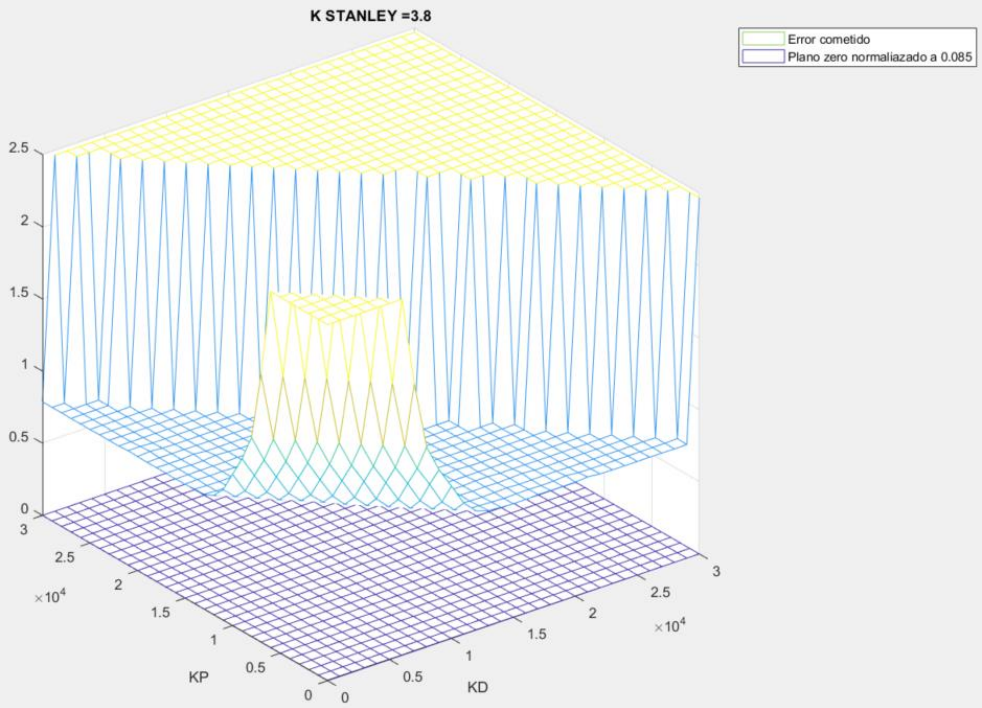


Figura 55. Resultado de la simulación con K de Stanley siendo 3,8, obteniendo ningún valor que cumpla las necesidades del sistema (ningún valor por debajo del plano 0).

En la Figura 55 se puede ver el resultado obtenido siendo la K de Stanley 3,8. Como se puede apreciar para ese valor de K de Stanley no se obtiene ninguna combinación que cumpla con los requerimientos del proyecto. En la Figura 56 se observa que ya para un valor de K de Stanley de 5 se obtiene unos valores de las constantes PID capaces de cumplir la restricción impuesta.

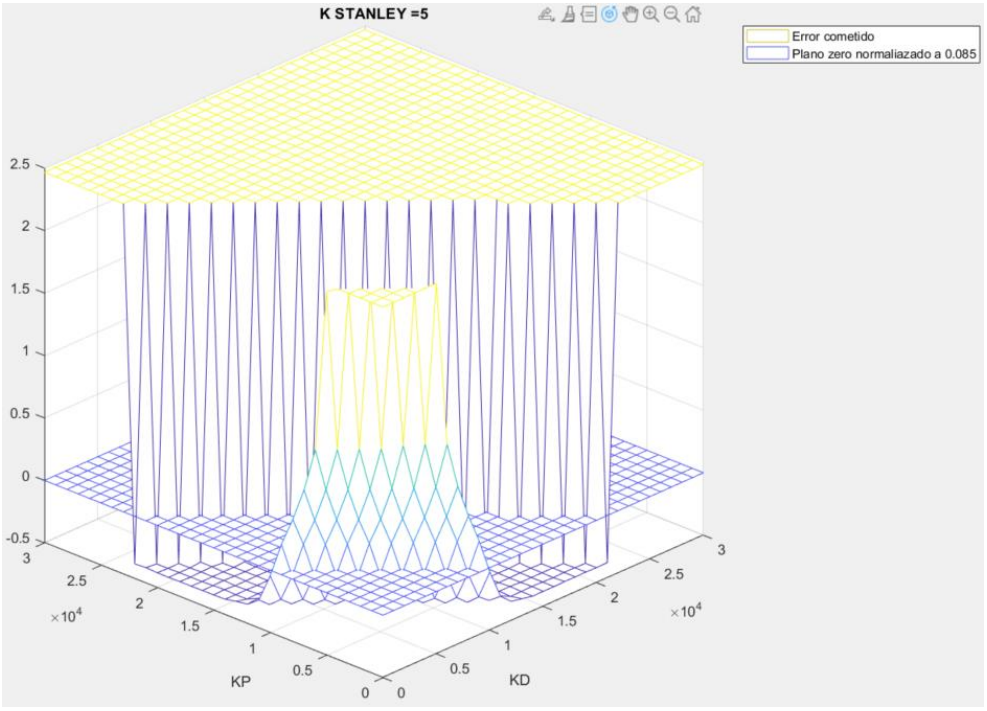


Figura 56. Resultado de la simulación para K de Stanley igual a 5, obteniendo un conjunto de valores que cumplen los requerimientos del proyecto (zona 1).

Conforme se aumenta la K de Stanley se comprueba que las características del conjunto de valores que cumplen la restricción (por debajo del plano 0), que forman una especie de valle en la representación cambian: el valle se estrechará, pero los valores de distancia máxima a la que se encuentran el coche y la banda decrecerán (Figura 57). Este efecto se cumple dentro de un rango, a partir de cierto valor de K de Stanley (por encima de 8,2 aproximadamente), el valle desaparece completamente. En la Figura 58 se aprecia como casi ya con K de Stanley igual a 8, el valle desaparece.

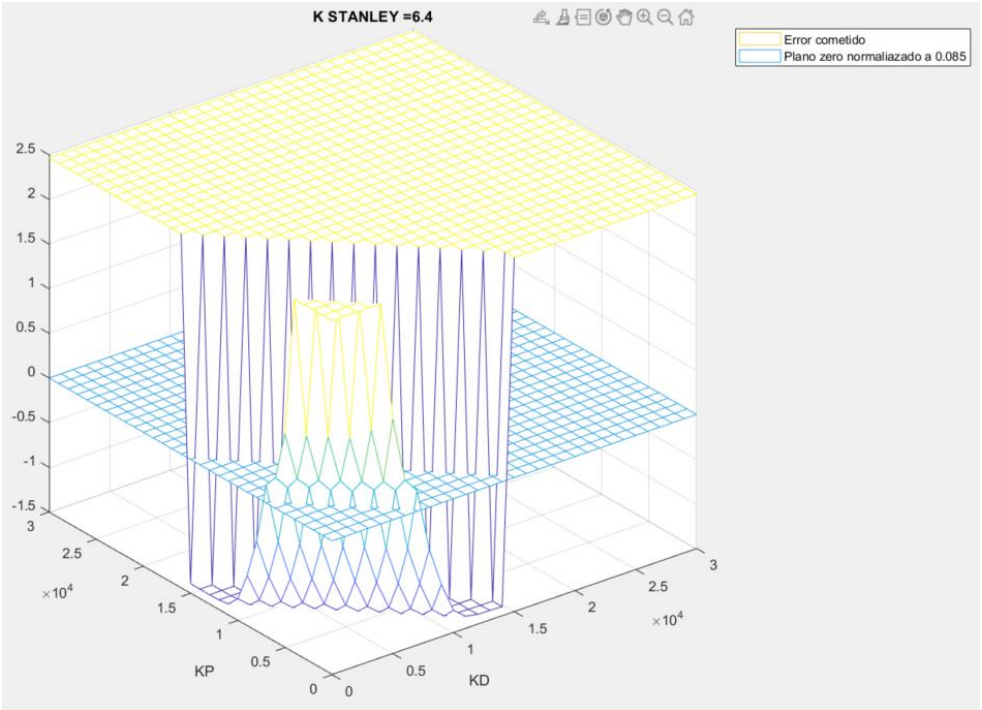


Figura 57. Resultado de la simulación con K de Stanley igual a 6,4 (zona 2).

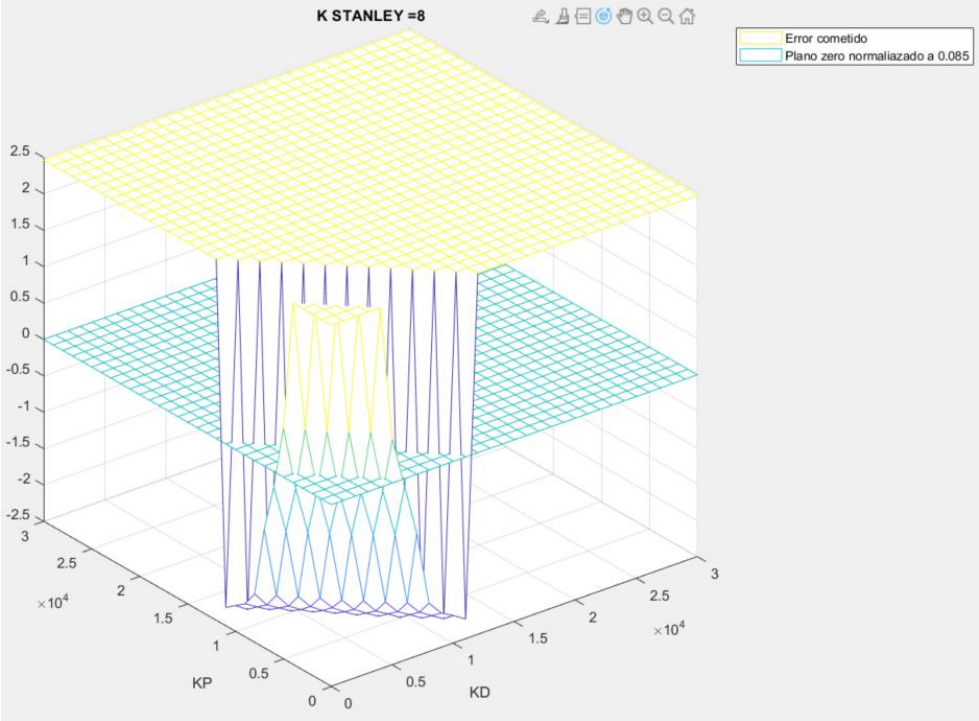


Figura 58. Resultado obtenido en la simulación con K de Stanley igual a 8 (zona 3).

Empleando el primer modo de simulación (única simulación) podemos verificar y comprobar que siguen la banda algunos de los valores que se encuentran por debajo del plano de 0. Por ejemplo, observando la Figura 57 con K de Stanley equivalente a 6,4 se ve que con Kp y Kd igual a 7000, el coche debería seguir el circuito. Este modo de simulación arroja tres representaciones: gráfica de rpm conseguidas por el motor vs de rpm objetivo enviadas, el circuito representado en azul, verde el camino descrito por el coche y rojo si se dista el coche de la banda más del máximo (0,085m) permitido; y la $\delta_{objetivo}$ del algoritmo de Stanley vs la $\delta_{conseguida}$ por el motor.

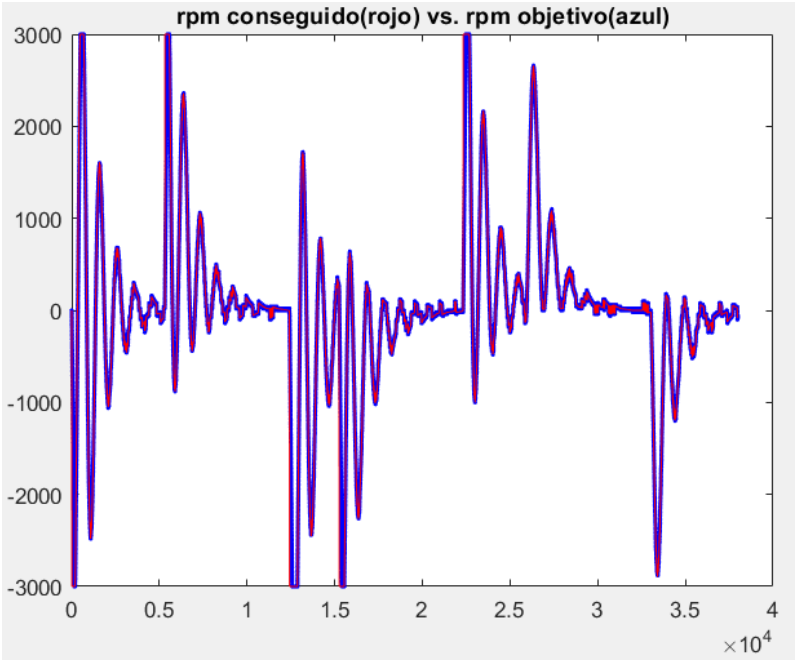


Figura 59. Con Kstanley 6,4, Kp=Kd=7000, v=3 y radio=4, la representación de las rpm conseguidas (rojo) vs las rpm objetivo (azul).

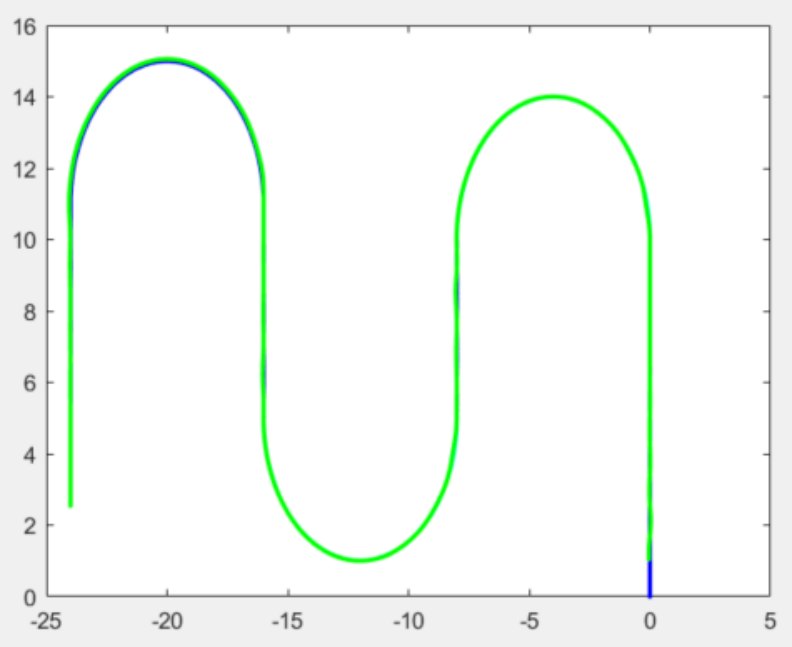


Figura 60.Con Kstanley 6,4, Kp=Kd=7000, v=3 y radio=4, la representación del circuito, estando de azul la ruta y en verde la trayectoria seguida hipotéticamente por el Twizy.

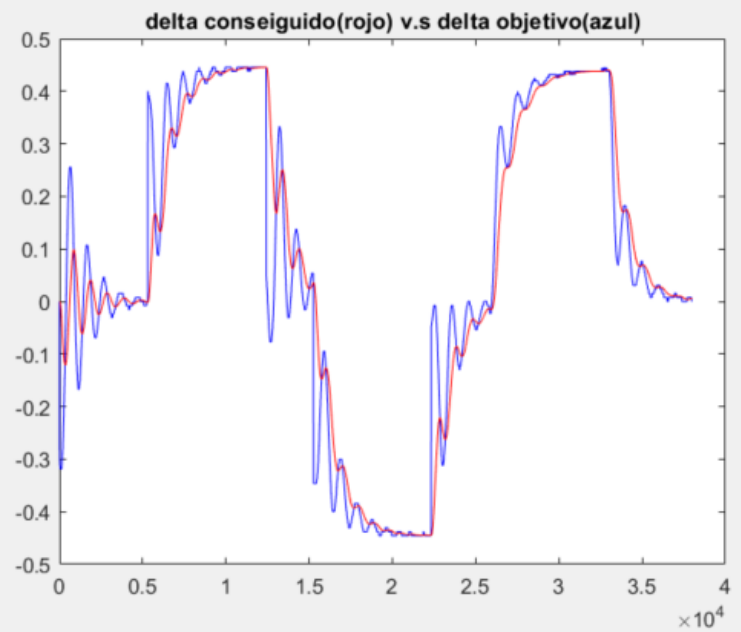


Figura 61. Con Kstanley 6,4, Kp=Kd=7000, v=3 y radio=4, la representación de la delta conseguida (rojo) vs el delta objetivo calculada con el método de stanley.

Si se hubiera probado con otra combinación de parámetros que estuviera por encima del plano, por ejemplo, empleando la Figura 55 con K de Stanley igual a 3,8, se utiliza ese valor y el resto de parámetro igual que en la anterior simulación. En la Figura 62 se ve como con esos valores el coche pierde la línea al comienzo de la tercera curva, en cuanto el coche pierda la línea se parará [24] ya que no será capaz de volver a encontrar la banda (lo dibujado a partir del punto donde pierde la línea es un representación ideal de la trayectoria del coche si el sensor fuera capaz de leer cualquier offset con respecto a la ruta).

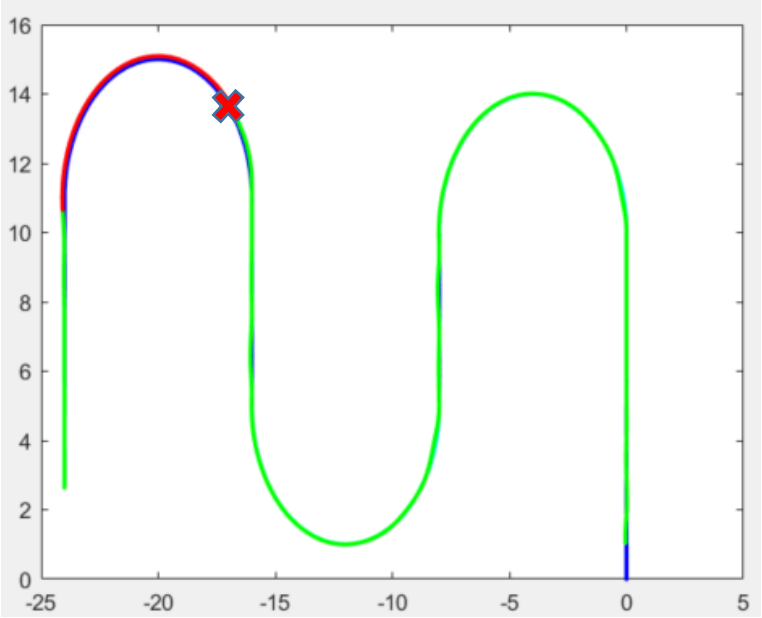


Figura 62. Con Kstanley 3,8, Kp=Kd=7000, v=3 y radio=4, la representación del circuito, estando de azul la ruta, en verde la trayectoria seguida hipotéticamente por el Twizy y en rojo cuando la distancia entre banda y sensor magnético es superior al máximo permitido. La X representa el punto donde el coche perdería la banda y debería pararse.

Una vez realizado este estudio, se trató de incrementar la velocidad de movimiento poniendo a 4 km/h, 5km/h y 6km/h. El procedimiento a seguir fue igual que el seguido para 3km/h, pero los resultados fueron diferentes. Para el rango de K Stanley analizado no era viable ninguna de esa velocidad, ya que el simulador determinaba que el coche se saldría del circuito. En la Figura 63 se puede ver el resultado de ir a 6km/h para los valores empleados en la Figura 60, el coche a 6km/h no consigue seguir el circuito.

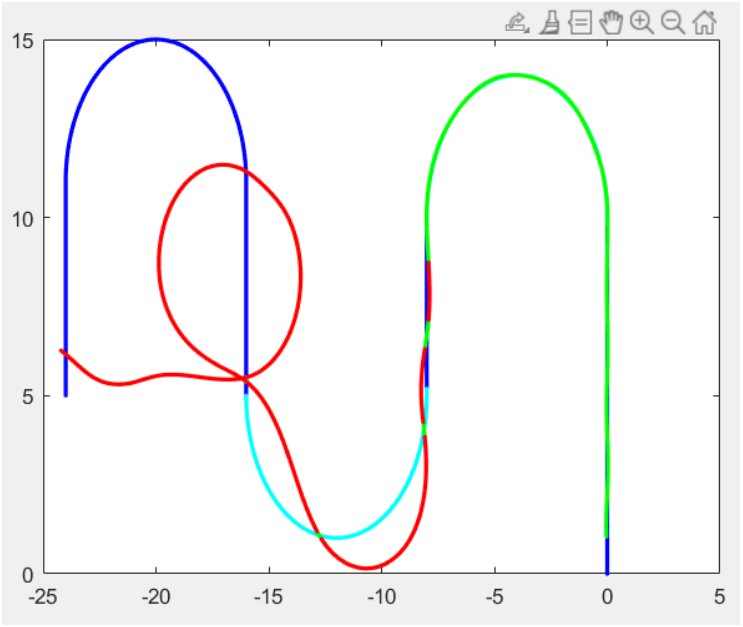


Figura 63. Con Kstanley 6,4, Kp=Kd=7000, v=6 y radio=4, la representación del circuito, estando de azul la ruta, en verde la trayectoria seguida hipotéticamente por el Twizy y en rojo cuando el coche a perdido la banda.

3.4 Integración y pruebas

Esta es la última fase dentro del plan de trabajo del campo de vehículo autónomo. Integraremos y adaptaremos el simulador realizado en Matlab dentro del programa "mod-con.py" programado en Python para comprobar si los resultados obtenidos se cumplen. Se han probado varias combinaciones de las que se consideraron según el simulador correctas para realizar el seguimiento. Se puede resumir en 3 grupos de combinaciones:

- Zona 1: Situación ilustrada en la Figura 56, donde los valores debajo del plano 0 forman un valle amplio, pero con poco margen respecto al plano.
- Zona 2: Situación ilustrada en la Figura 57, donde los valores debajo del plano 0 forman un valle más estrecho que en la zona 1, pero lo valores son más distantes del plano 0 que marca el límite.
- Zona 3: Situación ilustrada en la Figura 58, donde los valores debajo del plano forman un pico (desaparece el valle), y los valores son más bajos que los obtenidos en la zona 2.

Se implementó un circuito simple con la cinta magnética proporcionada por Astii Robotics (Figura 64) en el cual se trató de demostrar los siguientes puntos:

- El coche debe ser capaz de seguir en línea recta la banda.
- El vehículo debe ser capaz de escoger la ruta correcta en caso de encontrarse en una bifurcación tras leer un RFID.
- El Twizy debe ser capaz de realizar una curva de radio 4 o mayor.

Figura 64. Twizy de la universidad en el escenario de las pruebas.

Los resultados obtenidos tras realizar las pruebas fueron los siguientes, teniendo en cuenta que se introdujo velocidad 3km/h y radio igual a 4 en la curva como constantes iguales para todas las pruebas:

- Prueba con valores de la zona 1:
  - Valores ejemplo: K de Stanley igual a 5, Kd igual a 9000, Kp igual a 10000.
  - Resultado: Se obtiene mayor error en rectas, muchas veces apura tanto al extremo de la banda que acaba perdiéndola. En las bifurcaciones no decide bien la ruta preestablecida alguna vez.
- Prueba con valores de la zona 2:
  - Valores ejemplo: K de Stanley igual a 6,4, Kd igual a 7000, Kp igual a 6000.

- Resultado: Esta zona era la que mejor respondía al realizar los movimientos el Twizy. El coche corregía antes que en la zona 1 la desviación lo que evitaba que saliera, las bifurcaciones siguen siendo un problema porque no se consigue un 100% de acierto.

- Prueba con valores de la zona 3:
  - Valores ejemplo: K de Stanley igual a 8, Kd igual a 4000, Kp igual a 6000.
  - Resultado: El Twizy intenta corregir todo el tiempo la desviación lo más rápido posible lo que supone girar al máximo el volante, y cuando pasa por el medio la banda y quiere girar hacia el otro lado el tiempo en lo que tarda el volante en dar 2,8 vueltas es demasiado como para que el sensor magnético siga detectando.

Figura 65. Renault Twizy efectuado un giro a la derecha en una bifurcación.

El siguiente paso a dar en la fase de la prueba consistió en dibujar con cinta magnética un circuito más largo en un terreno más similar al de un aparcamiento. Para ello, se utilizó una zona de parking al lado del edificio UVainnova (Figura 66) , donde recreamos un escenario real donde el coche se aparcara solo. Las pruebas se realizaron con los parámetros obtenidos de la zona 2, ya que fueron los que mejor resultado dieron en la primera parte de la fase de pruebas.

Figura 66. Zona de pruebas al lado del edificio UVainnova.

El test fue infructuoso ya que el coche no consiguió seguir la banda magnética tras 3 o 4 metros. Durante el test pareció evidente que el coche no conseguía mantener la velocidad constante tal y como se había visto en las pruebas anteriores, y como se había supuesto para las simulaciones de ordenador. Las razones para que la velocidad variara eran varias, entre las que se destacan que el firme era relativamente irregular, que el parking tenía una ligera pendiente y que las ruedas del Twizy no tenían la presión adecuada. Todos estos factores unidos contribuyeron a que la variación de velocidad del Twizy fuera notable incluso a simple vista. La pregunta que surgió fue si esta situación podía dar lugar a que el algoritmo de control fracasara en la intención de mantener el Twizy siguiente la banda magnética.

Por lo tanto, el siguiente paso fue intentar modelar en el simulador la situación de variación de la velocidad. Dado que resulta difícil modelar una variación aleatoria de la velocidad a partir de unas cuantas observaciones de campo, se modelo la variación de la velocidad como una señal periódica con cuatro formas diferentes: rectangular, sinusoidal, dientes de sierra crecientes y dientes de sierra decrecientes. De estas cuatro variaciones, dos de ellas resultaron mostrar los peores resultados, siendo la variación rectangular y los dientes de sierra crecientes los que más dificultaban el control del vehículo. En la Figura 67, se aprecia una forma de cómo se modela esa velocidad no constante y a continuación se mostrará los resultados obtenidos si varía así la velocidad.

También se observó que el periodo de tiempo elegido para la variación de la velocidad era esencial para la pérdida de control. Esto resulta relativamente fácil de explicar, para periodos pequeños (frecuencias altas), dado que en ese caso para el algoritmo será como trabajar a una velocidad media inferior, a la que ya sabemos que es capaz de converger.

En la siguiente gráfica se muestra una simulación concreta para observar el efecto de la variación de la velocidad. Para ello se eligió que durante la parte rápida del periodo el vehículo circulara a 3 km/h, exactamente el valor que habíamos elegido para que circulara, y en la parte lenta se eligió que circulara al 15% de dicha velocidad, esto es, 0.45 km/h. Además, se le dio un periodo de dos segundos.

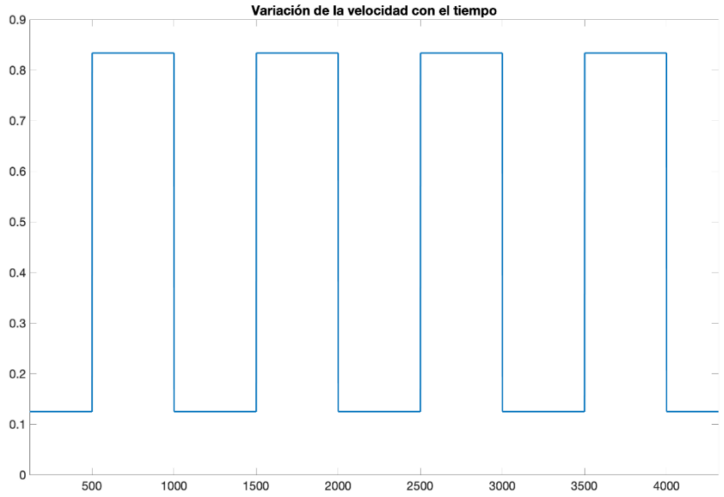


Figura 67. Variación de la velocidad, con un periodo de 2 segundos y velocidad máxima de 3km/h.

Se emplea el modo de simulación única y se comprueba si con los valores de la zona dos y con este nuevo escenario, el coche consigue seguir la banda magnética. En la gráfica de la Figura 68, como el parámetro de delta conseguido es incapaz de seguir el delta objetivo que debería alcanzar.

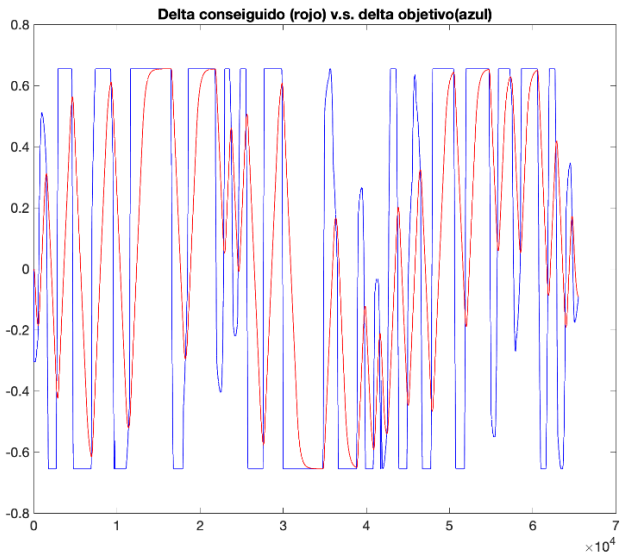


Figura 68. Con Kstanley 6,4, Kp=Kd=7000, velocidad variable y radio=4, la representación de la delta conseguida (rojo) vs el delta objetivo calculada con el método de Stanley.

En la Figura 69 se puede ver como el coche en la simulación es incapaz incluso de hacer la primera recta, tal y como, nos pasó en las pruebas realizar en la segunda fase en el terreno de al lado de UVainnova.

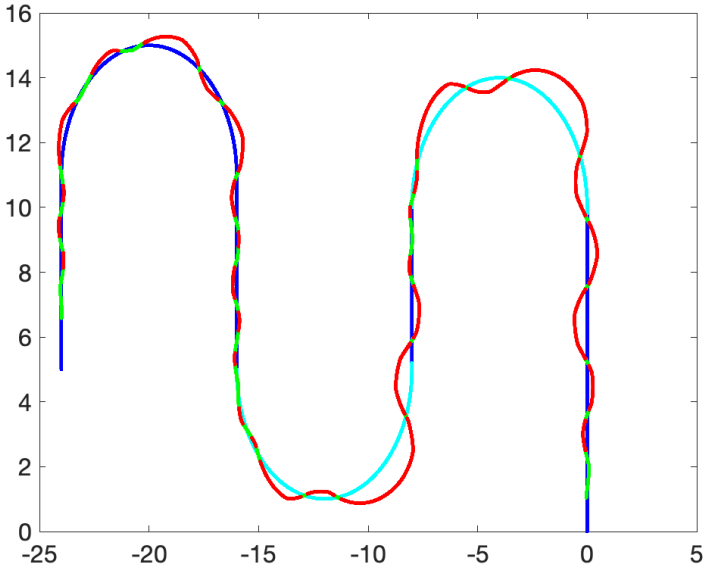


Figura 69. Con Kstanley 6,4, Kp=Kd=7000, velocidad variable entre 3km/h y un 15% de 3km/h (0,45km/h) y radio=4, la representación del circuito, estando de azul la ruta, en verde la trayectoria seguida hipotéticamente por el Twizy y en rojo cuando el coche a perdido la banda.

La explicación a este problema probablemente se encuentra en varios factores. Se debe tener en cuenta que estamos exigiendo al vehículo una presión lateral muy alta, de solo 8,5 cm a cada lado. Además, el algoritmo de Stanley que ha sido elegido para realizar el control lateral depende de dos parámetros, por un lado, el offset y por otro lado el ángulo del eje del vehículo con respecto a la tangente de la ruta en el punto de medida del offset. De estos dos parámetros solo disponemos de uno de ellos, contribuyendo a que el algoritmo no funcione como debería.

Con los sensores introducidos hasta el momento en el proyecto no es posible corregir el segundo problema, esto es, la falta de uno de las entradas de control de algoritmo de Stanley. Si se obtuviera los resultados mejorarían al tener más información sobre la posición del vehículo

respecto al circuito que describe la banda magnética. Esto podría conseguirse mediante una cámara o con un LiDAR que detectara esos dos parámetros y se calibrara mejor el algoritmo de conducción autónoma.

4. Servicios de vehículo conectado

Este capítulo se dedicará a mostrar un caso de uso de coche conectado. Se explicará la implementación que se ha realizado para poder realizar un procedimiento de FOTA. FOTA (*firmware over the air*) se trata de un sistema de actualización en remoto de las ECUs del vehículo. En nuestro caso, no se implementará un servicio FOTA completo, sino que reduciremos nuestro servicio a los primeros pasos del servicio FOTA, tal y como se vio en el capítulo de estado del arte. Por lo tanto, se tratará de obtener una respuesta con los parámetros más característicos de las ECUs con las que queramos comunicarnos y además se modificarán algunos de esos parámetros para demostrar que podemos realizar una actualización software remotamente.

Todo esto se podrá realizar a través de una página web, en la cual seleccionaremos el coche y la ECU con la que queremos comunicarnos. Finalmente se mostrará un caso de uso de cómo usar la interfaz gráfica proporcionada ya sea para leer o modificar dichos parámetros de la ECU indicada. A lo largo de este capítulo, se identificará por pseudónimos algunos datos y otros aparecerán tapados en las imágenes, debido a la confidencialidad de datos prestada por Renault.

4.1 Configuración con el DDT

En este proyecto se realizará un procedimiento FOTA de configuración. En este caso no hay Gateway que separe los buses CAN, por lo que cada controlador será maestro y esclavo a la vez. El objetivo final de este proceso de actualización en remoto será poder comunicarse con las ECUs, para ello es necesario saber qué mensajes CAN debemos enviarles para que respondan a nuestras peticiones. Para tratar de descifrar esos mensajes haremos uso del DDT (*diagnosis data tool*, Figura 70). Este dispositivo junto con su correspondiente programa DDT conectándose al bus CAN es capaz de entablar conexión y enviarle comandos de lectura o escritura a las ECUs que estén en el vehículo. El programa de DDT se trata de un software, que tras descargar una base de datos con las ECUs que tiene cada modelo de coche, es capaz primero de identificar si esa ECU responde y después si se recibe respuesta, poder enviarla las órdenes que estén disponibles.

Figura 70. Conector DDT.

Con tal de examinar los mensajes introducidos en CAN a través de las peticiones hechas por el DDT, se usará el programa CANoe. Este software se empleará como un analizador de paquetes, donde podremos filtrar todos los paquetes leídos y fijarnos solo en aquellos enviados y recibidos por el DDT. La conexión entre el ordenador donde se ejecuta CANoe y el bus CAN se hace con una CANcaseXL.

Por último, al tener que conectar dos conectores CAN cuando el Twizy solo dispone de una interfaz OBD, se conectará a dicha interfaz un cable multiplexor OBD 2:1, pudiendo realizar la conexión de todos los elementos indicados. En la Figura 71 se muestra un esquema de esta configuración.

Esta configuración podría hacerse con un solo ordenador, conectando tanto el DDT como el CANoe al mismo ordenador, sin embargo, debido a que la licencia del software de DDT solo está en el ordenador prestado por Renault y en él no se dispone de CANoe se opta por la configuración descrita.

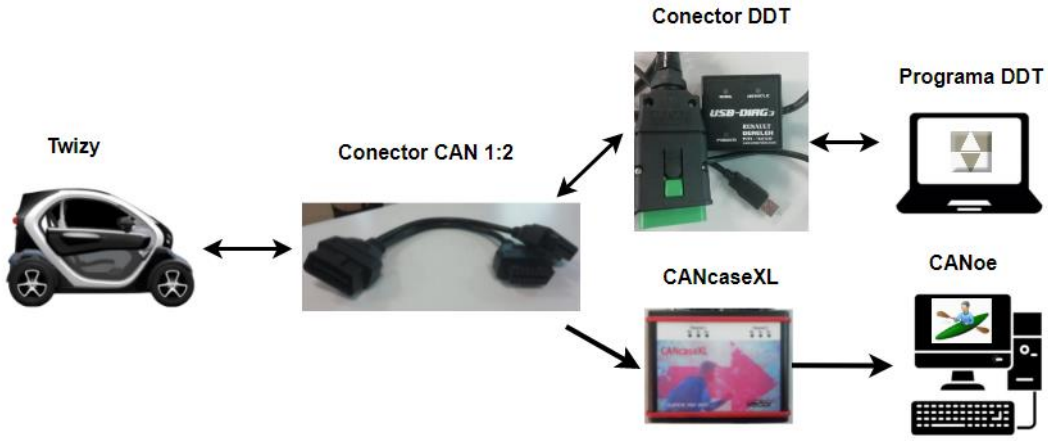


Figura 71. Esquema de trabajo utilizado en la primera fase de comprensión de los mensajes intercambiados.

Tras conectar todo correctamente, se procede a arrancar el vehículo. Primero ejecutamos y configuramos CANoe. En la Figura 72 podemos ver la interfaz gráfica de CANoe, en la parte central comenzarán a aparecer tramas que se envían periódicamente en el coche. No seleccionamos la opción "*Toggle display mode*", para diferenciar con el color qué trama es nueva, ya que veremos que aparecerá una nueva de color negro, frente a las demás tramas que saldrán de color gris claro. En este primer paso nuestro objetivo será determinar los IDs del mensaje CAN destinados a las ECUs con las que podemos comunicarnos.

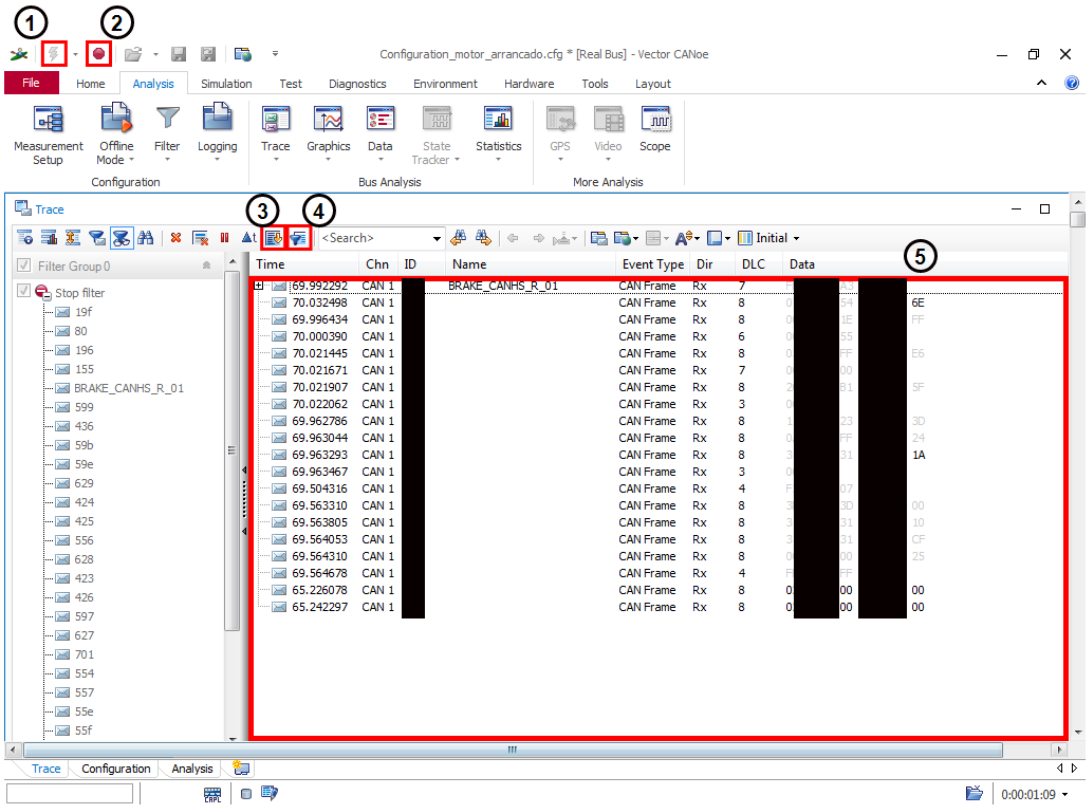


Figura 72. 1. Botón para comenzar a analizar los paquetes; 2. Botón para pausar la recepción de nuevos mensajes; 3. Toogle display mode, sirve para dejar que muestren todas las tramas, aunque sean repetidas o que se solapen aquellas con ID igual; 4. Activar o desactivar los filtros por ID; 5. Ventana donde aparecerán las tramas recibidas, en la columna de data aquellos datos que se repiten tornan un color grisáceo claro frente a los nuevos datos en color negro.

Ya tenemos el CANoe preparado, ahora se configura el software del DDT. Iniciamos el programa y en la pantalla inicial *Scan* seleccionamos el modelo del coche con el que se está trabajando, en este caso: Renault Twizy. Nos aparecerán todas las ECUs que pueden estar instaladas en un coche de ese modelo y el bus CAN, por lo tanto, deberemos determinar qué ECUs son las que tiene. Para ello, al haber 11 ECUs se prueba a mandarles un mensaje de identificación a cada una de ellas y ver cuáles contestan. Finalmente, se obtiene que podemos tener acceso a 3 ECUs: tablero de a bordo, controlador de la batería de litio y al módulo del cargador de la batería.

Para comprender los datos de los mensajes CAN, se ha hecho un estudio previo de los protocolos que intervienen en esta comunicación. Así pues, teniendo como referencia el modelo OSI (*Open Systems Interconnection*): en la capa de aplicación tenemos el protocolo UDS (*Unified Diagnosis Services*); en la capa de transporte y de red el protocolo ISO-TP; y el protocolo CAN engloba tanto la capa física como la capa de enlace.

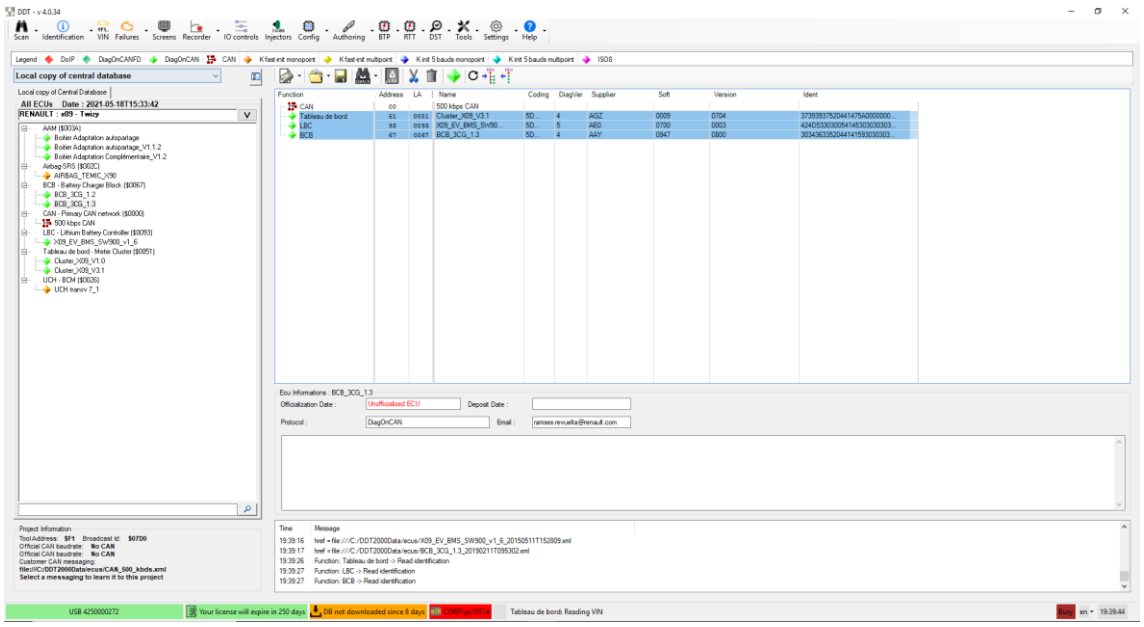


Figura 73. Interfaz gráfica del programa DDT.

Accediendo a la pantalla *Identification* podremos ver la información general de cada una de las 3 ECUs a las que accedemos. Esto generará que en la pantalla de CANoe se hayan recibido 6 tipos nuevas de tramas con IDs diferentes a los previamente recibidos por CAN. Como este procedimiento que se ha realizado se analiza cada ECU por separado pudiendo discernir los ID, a los cuales se hará referencia de aquí en adelante por temas de confidencialidad de la siguiente manera:

- *IDtxTDB*: ID de aquellos mensajes CAN necesarios para enviar peticiones a la ECU del tablero de a bordo.
- *IDrxTDB*: ID de aquellos mensajes CAN enviados por la ECU del tablero de a bordo con respuesta a las peticiones realizadas.
- *IDtxLBC*: ID de aquellos mensajes CAN necesarios para enviar peticiones a la ECU de la batería de litio.
- *IDrxLBC*: ID de aquellos mensajes CAN enviados por la ECU de la batería de litio con respuesta a las ordenes previamente enviadas.
- *IDtxBCB*: ID de aquellos mensajes CAN necesarios para enviar peticiones a la ECU del cargador de batería.
- *IDrxBCB*: ID de aquellos mensajes CAN enviados por la ECU del cargador de batería con respuesta a las peticiones realizadas.

El siguiente punto a tratar son las órdenes, es decir, cómo configurar el campo de datos del mensaje CAN para que nos responda la ECU con lo que se quiere. De los mensajes de identificación, previamente citados se deduce el DID que llamaremos como *did-ident*, que es aceptado por cada una de las tres ECUs. Con esto se tendría ya una orden de lectura de parámetros de la ECU, ahora se trata de conseguir otro DID de escritura.

En la ventana "*Tools→Request and Responses*" se pueden enviar los dids disponibles cada ECU. También se puede optar por la ventana *Screens* que proporciona una interfaz más amigable donde podemos ver qué parámetros están disponibles para modificar en cada ECU. Finalmente, tras indagar por esta interfaz, y con el fin de que se pueda demostrar visualmente que se está realizando cambios, se opta por analizar los mensajes CAN intercambiados para cambiar los kilómetros que indica el odómetro y las unidades que utiliza, pudiendo variar entre kilómetros o millas. Ambas operaciones de escritura serían sobre la ECU del tablero de a bordo.

Teniendo ya conocimiento de los IDs de los mensajes CAN para cada ECU y los DIDs necesarios para proceder a realizar las operaciones descritas anteriormente, volveremos a CANoe con el fin de poder ver toda la secuencia de mensajes que se están enviando en cada operación. Desmarcaremos la opción "*Toggle display mode*" para ver todas las tramas, aunque sean con el mismo ID, y aplicaremos un filtro dependiendo de la ECU que queramos analizar para solo quedarnos con ese intercambio de mensajes entre la ECU a examinar y el programa DDT.

En la Figura 74 se muestra el esquema que se implementa en CANoe, donde gracias al bloque Logging se crean ficheros de texto con las tramas recibidas para poder ser guardadas y analizadas posteriormente.

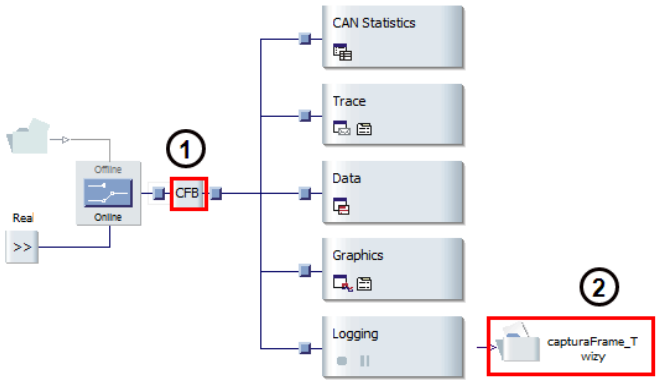


Figura 74. Esquema de configuración de CANoe: 1. Bloque de filtros por ID de mensajes CAN; 2. Guardamos las diferentes tramas que obtenemos en un fichero de texto "capturaFrame_Twizy".

A continuación, se mostrará en forma de diagrama de secuencia los mensajes intercambiados indicando en cada caso qué operación se está realizando, y explicando el significado de cada uno de los mensajes.

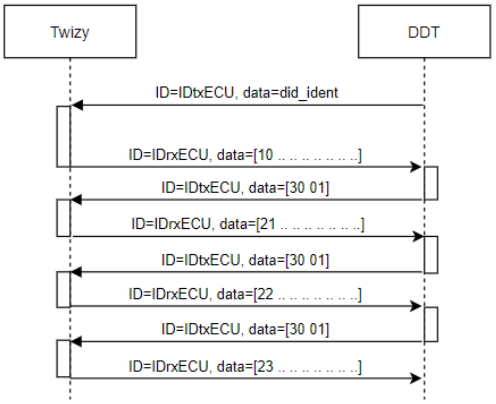


Figura 75. Secuencia de mensajes tras una petición de identificación a una ECU.

En la Figura 75, se muestra la secuencia de mensajes que se intervienen en la comunicación entre una ECU y el DDT, a la hora de obtener la información general de la ECU. Como se aprecia primero el DDT debe enviar el mensaje UDS con el ID de la ECU a la que queremos entablar conexión y la orden o DID, para saber qué tipo de operación se quiere realizar. Después la ECU responderá con el ID de respuesta, cabe destacar que al ser la información de la respuesta más larga que la que se puede encapsular en un mensaje CAN, se envía en 4 mensajes distintos, significando el primer byte: 10, primera trama de una secuencia; 21, trama consecutiva numero 1; 22, trama consecutiva número 2; y 23, trama consecutiva número 3. Entre estos mensajes el DDT envía un mensaje cuyos datos son 30 01, esto significa control de flujo (30) y que permite el recibo de un mensaje más de la secuencia (01) [74].

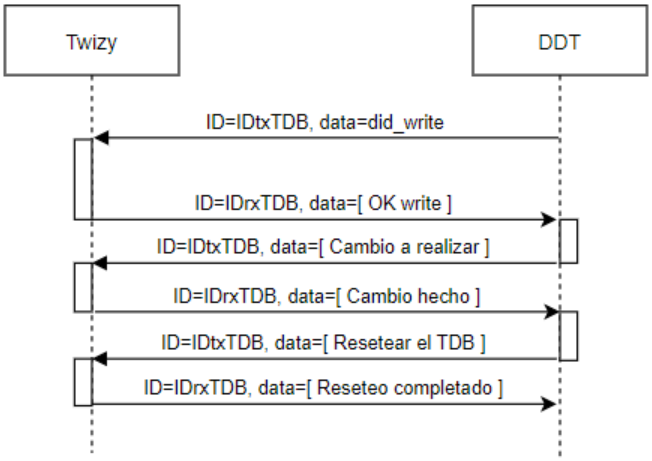


Figura 76. Secuencia de mensajes intercambiados cuando se escribe en la ECU del tablero de a bordo (TDB).

En la Figura 76 podemos ver la secuencia de mensajes necesaria para que se produzca un cambio en el tablero de cambio, tras haber modificado alguno de sus parámetros. Por lo tanto, esta secuencia de mensajes se dará tanto para cambiar las unidades del odómetro como para cambiar los kilómetros que indique el odómetro.

4.2 Arquitectura FOTA implementada

Ahora ya sabemos los mensajes intercambiados entre la ECU y el DDT para podernos comunicar con la ECU. El siguiente objetivo será sustituir el DDT por el sistema de FOTA creado, que tendrá las mismas funciones: ser capaz de comunicarse con la ECU, poder realizar operaciones tanto de lectura como de escritura, y proporcionar una interfaz gráfica que muestre al usuario el resultado de las operaciones realizadas. A continuación, se mostrará la arquitectura y el procedimiento que sigue el sistema de FOTA implementado.

La arquitectura del sistema de FOTA es la mostrada en la Figura 77. El protocolo de comunicación es MQTT (*Message Queing Telemetry Transport*), protocolo IoT que se usa para establecer la conexión entre el servidor y la flota de coches que tenía disponibles el sistema TwizyLine [18].

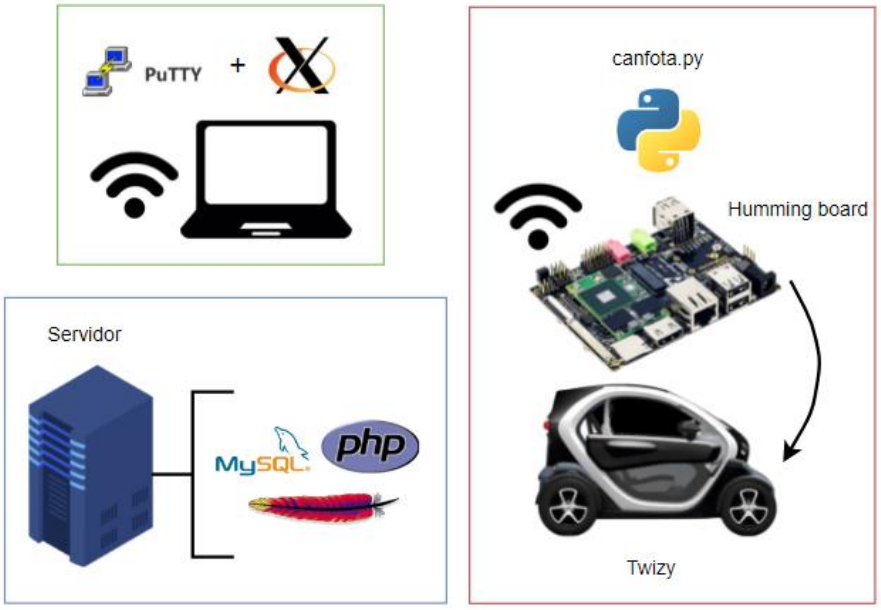


Figura 77. Arquitectura FOTA implementada.

En el vehículo se ha creado un script en Python "canfota.py" que se encargará de gestionar las comunicaciones entre las ECUs y el servidor. Este fichero, que se ejecuta en el módulo de comunicaciones, tiene como tarea: decidir si el coche está disponible para hacer una actualización FOTA, enviar por el bus CAN los mensajes CAN que reciba por MQTT, así como leer y devolver las respuestas a esos mensajes también por MQTT al servidor. En el Anexo 2, se muestran los tópicos usados en cada caso.

En el servidor está alojada una página web. A través de esta página se accede al sistema FOTA y a sus funcionalidades. Para poder alojar la página web en el servidor (una Raspberry) fue necesario instalar el servidor HTTP Apache (servidor web), el compilador de PHP y phpMyAdmin (conectar PHP con la base de datos MySQL). Para acceder a esta página primero se necesitará activo algún visor gráfico como Xlaunch, que es el que se usó en este proyecto. Después, se accede al servidor mediante SSH, para lo cual se usará el software PuTTY habilitando la opción de dejar pasar tráfico al visor gráfico de Xlaunch (Figura 78).

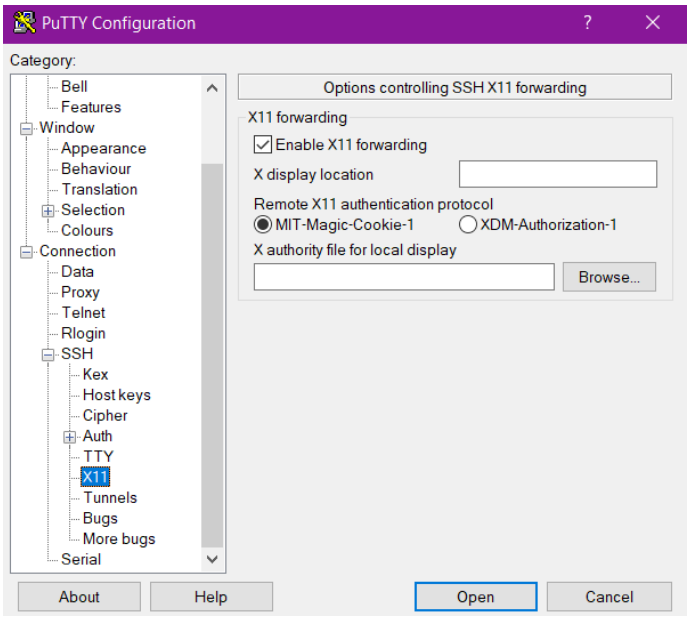


Figura 78. Configuración de una conexión PuTTY habilitando el tráfico al X server que esté activado.

Sabiendo la dirección IP pública, y las credenciales del servidor (usuario y contraseña) se podrá acceder al servidor tras habilitar la opción antes mencionada. Ya dentro del servidor se accederá a un navegador, en este caso se usará Google Chromium que viene ya preinstalado es el sistema operativo de Raspbian. Teclearemos "`chromium-browser`" y se abrirá el navegador en el Xlaunch. La página principal del sistema FOTA está alojada en la URL: "http://localhost/init.php" (Figura 79).

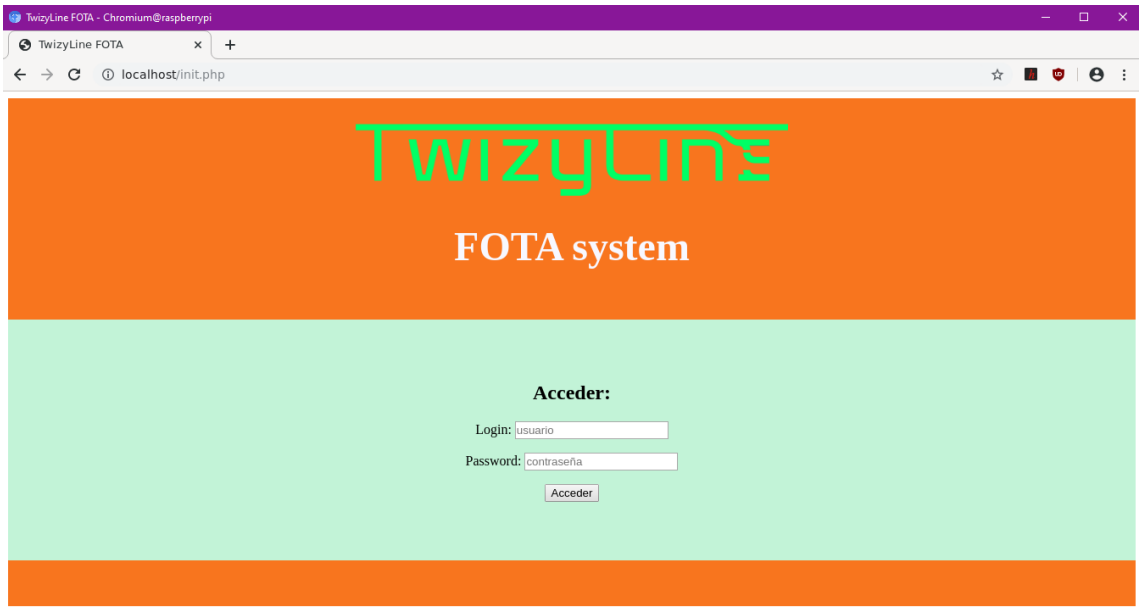


Figura 79. Página inicial de FOTA.

A este sistema solo tendrán acceso aquellas personas que estén ya previamente registradas como administradores en el sistema de TwizyLine, es decir, aquellos usuarios que estén registrados en la base de datos como administradores [18]. Actualmente hay un administrador, cuyas credenciales son: 'samuel' y contraseña '1234'. Una vez accedamos, la pantalla que aparecerá será la de la Figura 80, que explica qué hace el sistema al que se está accediendo y se pide pulsar el botón "Buscar coches actualizables" para ver qué coches están disponibles para conectarse.

Figura 80. Página web del sistema de FOTA tras haber iniciado sesión.

En el siguiente apartado mostraremos dos ejemplos de lectura y otros dos de escritura de los parámetros de la ECU del Renault Twizy que tenemos disponible, mostrando cada uno de los pasos a dar en la interfaz gráfica diseñada.

4.2.1 Caso de uso FOTA usando la API

Esta sección se dedicará a mostrar 4 ejemplos de FOTA que se pueden realizar con la página web descrita. Tal y como se explicó en el apartado anterior lo primero será iniciar sesión en la plataforma y después ya tendremos acceso a buscar los coches disponibles para actualizar.

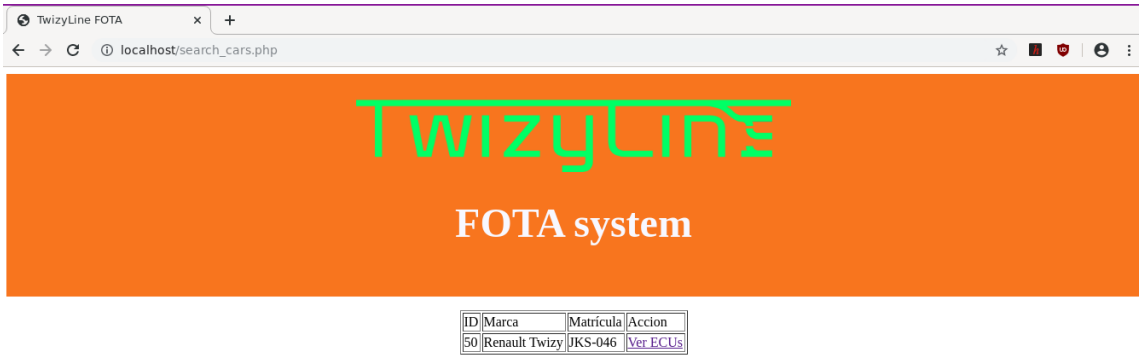


Figura 81. Coches disponibles en la página web para hacer un procedimiento de FOTA.

El único coche disponible para hacer las pruebas es el Renault Twizy que tenemos a nuestra disposición en la ETSI. En este coche ya tenemos integrado las humming boards donde se ejecuta el fichero “canfota.py” que habilita las comunicaciones entre las ECUs y el servidor usando el protocolo MQTT. Procederemos a ver las ECUs a las cuales tenemos acceso clicando sobre “Ver ECUs”.

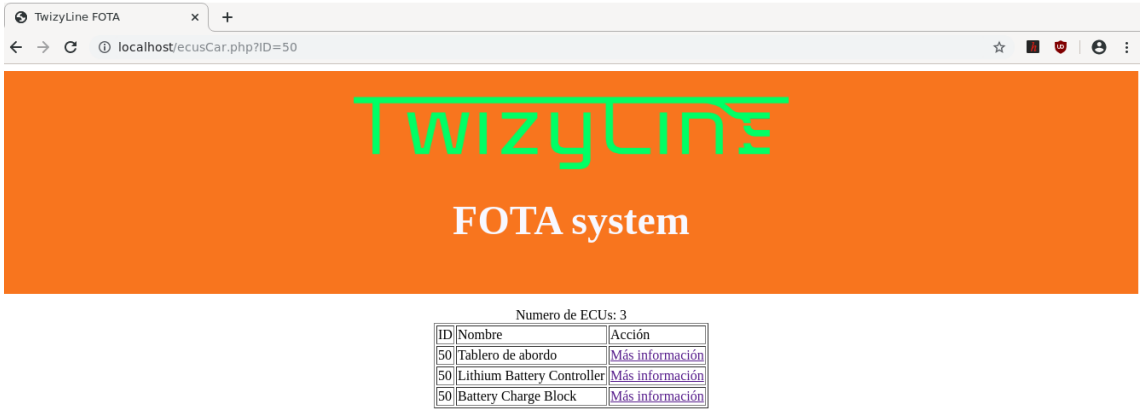


Figura 82. ECUs con las que podernos comunicarnos del coche con ID 50.

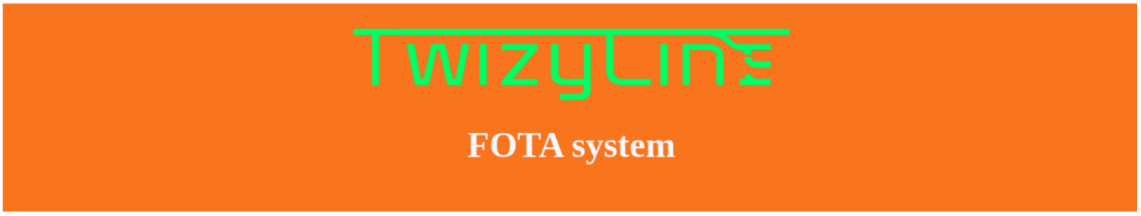
En la Figura 82, se ven las ECUs con las que se puede establecer conexión. Si nos fijamos son las mismas tres a las que teníamos acceso en el apartado 4.1 con el DDT. A continuación, se mostrarán dos ejemplos de lectura de parámetros de dos de las ECU, comenzando por el de la ECU encargada de gestionar la carga de la batería (Figura 83).

INFO de Battery Charge Block del coche 50

Part. Number Ref.:	0463R
Diag. Version:	04
Supplier:	AAY
Hardware Ref.:	0000R
Soft.:	0947
Version:	0800
Calibration Ref.:	0001
Reserved:	Pn 2 Hn 1 An 0
Format:	88

Figura 83. Información de la ECU del cargador de batería.

La información que se muestra en la Figura 83 es la obtenida tras un mensaje de identificación. En la Figura 84, se ve la información general de la ECU del tablero de a bordo. Este a diferencia del anterior sí que te permite modificar dos de los parámetros: el odómetro y la unidad de medida tanto del odómetro como del velocímetro.



INFO de Tablero de abordo (Cluster) del coche 50

Part. Number Ref.:	7997R		
Diag. Version:	04		
Supplier:	AGZ		
Hardware Ref.:			
Soft.:	0009		
Version:	0704		
Calibration Ref.:	0000		
Reserved:	Pn 2 Hn 1 An 0		
Format:	88		
Odometro:	27674	distance	Enviar
Odometers unit:	KM	KM ML	

Figura 84. Información de la ECU del tablero de a bordo.

Ahora se procederá a mostrar dos ejemplos de escritura. Se va a añadir un kilómetro más a lo ya marcado, por lo tanto, teclearemos 27675 en el campo de texto y se dará a enviar (Figura 85). Al dar a enviar lo que se está produciendo es que se envíe un mensaje MQTT al coche pidiéndole el cambio de kilómetros al número introducido por el usuario.

INFO de Tablero de abordo (Cluster) del coche 50

Part. Number Ref.:	7997R	
Diag. Version:	04	
Supplier:	AGZ	
Hardware Ref.:		
Soft.:	0009	
Version:	0704	
Calibration Ref.:	0000	
Reserved:	Pn 2 Hn 1 An 0	
Format:	88	
Odometro:	27674	27675 Enviar
Odometers unit:	KM	KM ML

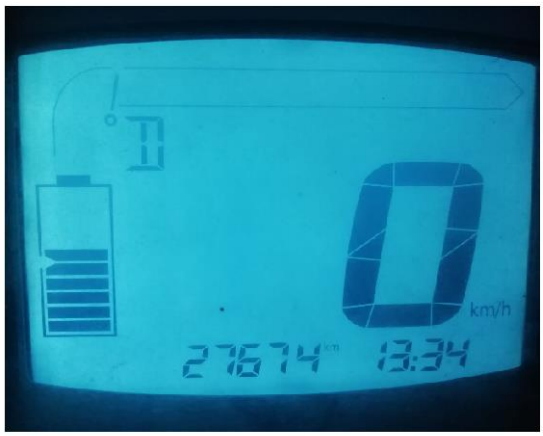


Figura 85. El odómetro marca 27674 km, tal y como se indica en la tabla, y se va a añadir 1 km más.

En la Figura 86, se aprecia que el resultado del cambio de los kilómetros que marca el odómetro ha sido satisfactorio.

INFO de Tablero de abordo (Cluster) del coche 50

Part. Number Ref.:	7997R	
Diag. Version:	04	
Supplier:	AGZ	
Hardware Ref.:		
Soft.:	0009	
Version:	0704	
Calibration Ref.:	0000	
Reserved:	Pn 2 Hn 1 An 0	
Format:	88	
Odometro:	27675	distance Enviar
Odometers unit:	KM	KM ML

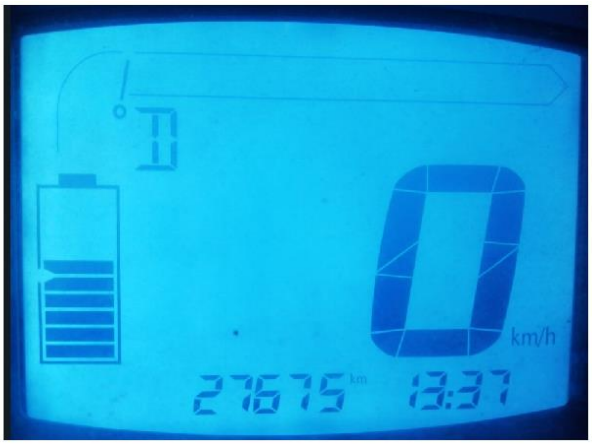


Figura 86. Tras realizar el cambio, el odómetro marca 27675 km, tanto en la página como en la pantalla de a bordo.

El último ejemplo que se quiere mostrar es el cambio de unidades de kilómetros a millas. Para ello hay que pulsar en el botón “ML” de la fila “Odometers units”. Esto generará un mensaje MQTT que se enviará al coche 50 para que cambie sus unidades. En la Figura 87 se muestra que, en ambos lados, página web y pantalla de a bordo, la unidad de medida son las millas.

INFO de Tablero de abordo (Cluster) del coche 50

Part. Number Ref.:	7997R	
Diag. Version:	04	
Supplier:	AGZ	
Hardware Ref.:		
Soft.:	0009	
Version:	0704	
Calibration Ref.:	0000	
Reserved:	Pn 2 Hn 1 An 0	
Format:	88	
Odometro:	27675	distance Enviar
Odometers unit:	ML	KM ML

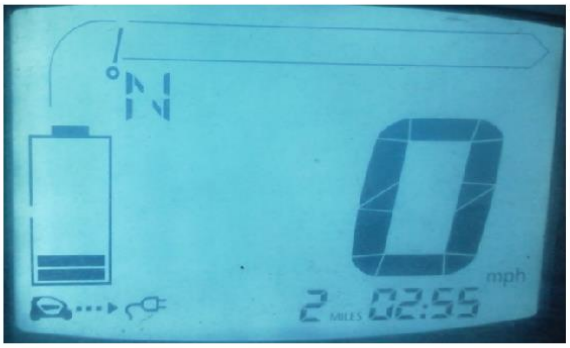


Figura 87. Las unidades del odómetro han sido cambiadas a millas.

5. Conclusiones y líneas futuras

5.1 Conclusiones

Considerando los objetivos de este trabajo fin de máster definidos en la sección 1.2, se puede concluir que se ha logrado llegar a las metas prefijadas. Durante el desarrollo de este proyecto, se ha dotado al Renault Twizy, donado por la fundación Renault a la universidad de Valladolid, de conducción autónoma bajo las condiciones descritas dentro del proyecto TwizyLine. Además, se ha provisto al vehículo de nuevas funcionalidades dentro del campo de coche conectado, habilitando la comunicación remota para la actualización del software de las ECUs que operan dentro del Twizy.

Como ya se comentó a lo largo del documento, este trabajo supone una continuación del proyecto TwizyLine. A pesar de haber terminado el concurso para el que fue ideado y galardonado con un segundo puesto en su fase internacional, el proyecto ha seguido dando pasos hacia adelante, no sólo en el ámbito tecnológico, que se ha ocupado este trabajo fin de máster de ello, sino también en el ámbito de diseminación del proyecto o económico a nivel de viabilidad del proyecto de cara a montar una empresa propia, que ha sido llevado a cabo por todos los integrantes del equipo.

Sobre la implementación de este trabajo fin de máster, cabe mencionar la tarea de investigación y estudio de las diferentes tecnologías que se podían emplear, especialmente a la hora de proporcionar una conducción autónoma al Twizy, tomando como referencia proyectos anteriores con objetivo final similar, como fue el proyecto de la universidad de Dresden o en la implementación hecha en la universidad de Cartagena, ambos explicados en la sección 2.1.2.

Sobre esta primera parte del trabajo realizado cabe destacar la consecución de los siguientes puntos:

- Integración de los sensores implantados a nivel hardware y software.
- Instalación del motor Maxon en el chasis de coche. Desarrollo software tanto de la controladora como de las ordenes necesarias para manejar debidamente el control transversal.
- Estudio y posterior implementación de diferentes algoritmos de guiado y control para la conducción autónoma.
- Realización de un simulador en Matlab donde se podía probar diferentes parámetros para observar si eran adecuados para seguir la línea, antes de introducirlos en el vehículo.
- Lograr la conducción autónoma del Twizy y la toma de decisiones en las bifurcaciones dependiendo de la orden enviada.

Sobre la segunda parte de este proyecto, se centró en el desarrollo de un sistema FOTA para habilitar al Twizy de una nueva funcionalidad dentro del campo del vehículo conectado. Para esta parte fue clave la ayuda ofrecida por el tutor de Renault que proporcionó el conocimiento y herramientas necesarias para llevarlo a cabo.

El desarrollo del sistema FOTA supuso por un lado la creación de una página web alojada dentro del servidor de TwizyLine, para dotar al sistema de una interfaz gráfica desde donde poder comunicarse con el coche. Los principales objetivos alcanzadas en esta fase se resume en:

- Creación de una interfaz gráfica para que el trabajador acceda al sistema FOTA.
- Establecer comunicación con las ECUs disponibles en el Renault Twizy, para que podamos leer su configuración actual.
- Modificar parámetros de la configuración de las ECUs.

Por último, es necesario enfatizar que la realización de este trabajo fin de máster ha supuesto la adquisición de nuevo conocimiento y por supuesto también, el uso de ciertas áreas ya estudiadas durante mi estancia en la Universidad de Valladolid, tanto durante el grado de Ingeniería de Tecnologías de la Telecomunicación como durante el Master de Ingeniero de Telecomunicación. Además, ha supuesto gran avance dentro del proyecto TwizyLine, despertando el interés de otras corporaciones en nuestro sistema y abriéndonos las puertas a nuevos eventos.

5.2 Líneas futuras

El proyecto TwizyLine debe seguir dando pasos hacia adelante. En este trabajo fin de master se abre dos líneas de investigación donde poder seguir avanzando: la conducción autónoma y el vehículo conectado. El progreso dentro de estos dos campos significará la capacidad de crear un producto más innovador de cara a su futura comercialización.

A continuación, se plantea nuevas líneas de desarrollo:

- Integrar nuevos sistemas de detección de línea en el Twizy, como LiDARs o cámaras que permitan la implementación completa del algoritmo de Stanley.

- Desarrollo hardware y software de motor con su estructura correspondiente para activar el freno del vehículo.

- Mejorar el simulador realizado en Matlab, introduciendo nuevos datos obtenidos tras las pruebas realizadas empíricamente o sustituir algunos de los parámetros utilizados en caso de emplear otro tipo de sensores o elementos que ayuden a la conducción autónoma.

- Integrar en una sola página la dedicada al proyecto TwizyLine y la creada para el sistema FOTA, con tal de que los empleados puedas acceder al sistema FOTA sin necesidad de ssh o servidor X.

- Incrementar las funcionalidades del sistema FOTA, no solo pudiendo hacer un FOTA de la configuración, sino también un FOTA software.

- Estudiar posibles soluciones a los problemas que plantea los sistemas FOTA actuales, en relación a: velocidad del canal de comunicación dentro del vehículo, capacidad de procesamiento de las ECUs o dependencias entre ECUs.

5.3 Premios y otros méritos

El proyecto TwizyLine, tras ser el ganador en la fase nacional, fue el representante español en la final internacional del TwizyContest2020 (Figura 88) en diciembre de 2020, donde se alzó con la segunda posición.

Figura 88. Logo del concurso TwizyContest2020.

Además, el proyecto también fue galardonado en los premios "Iniciativa Campus Emprendedor 2020" (Figura 89), dentro de la categoría de "Idea Innovadora de Negocio", organizado por el T-CUE (Transferencia de Conocimiento Universidad Empresa).

Figura 89. Logo del concurso "Iniciativa Campus Emprendedor".

6.
Bibliografía

[1] M. Ruiza, T. Fernandez y E. Tamaro, «Biografías y vidas. La enciclopedia biográfica en línea.,» 2004. [En línea]. Disponible: https://www.biografiasyvidas.com/biografia/l/lenoir.htm. [Último acceso: Mayo 2021].

[2] BBC News, «¿Por qué ya no conducimos coches de vapor?,» 23 Febrero 2013. [En línea]. Disponible: https://www.bbc.com/mundo/noticias/2013/02/130215_coches_vapor_ap. [Último acceso: Mayo 2021].

[3] Autofacil, «La evolución de la seguridad en el automóvil,» 2014. [En línea]. Disponible: https://www.autofacil.es/seguridad/2014/06/05/evolucion-seguridad-automovil/19056.html. [Último acceso: Mayo 2021].

[4] Fundación Mafre, «¿Cuántas vidas salvan el cinturón de seguridad y las sillitas?,» Noviembre 2016. [En línea]. Disponible: https://www.seguridadvialenlaempresa.com/seguridad-empresas/actualidad/noticias/cuantas-vidas-salvan-cinturon-y-sillitas.jsp#:~:text=Seg%C3%BAn%20Volvo%2C%20el%20cintur%C3%B3n%20de,1%2C25%20millones%20de%20personas. [Último acceso: Mayo 2021].

[5] M. Muñoz, « El cinturón de seguridad se usa menos de lo que nos parece » 15 Abril de 2021.[En línea]. Disponible: https://soymotor.com/coches/noticias/el-cinturon-de-seguridad-aun-se-usa-menos-de-lo-que-nos-parece-975613#:~:text=La%20Encuesta%20sobre%20Actitudes%20de,el%20del%20resto%20de%20pasajeros. [Último acceso: Mayo 2021].

[6] Instituto de investigación de vehículos: Centro Zaragoza, «Los sistemas de Adaptación Inteligente de la Velocidad» 2010. [En línea]. Disponible: http://www.centro-zaragoza.com:8080/web/sala_prensa/revista_tecnica/hemeroteca/articulos/R38_A7.pdf. [Último acceso: Mayo 2021]

[7] J. Billington, « The Prometheus project: The story behind one of AV's greatest developments » 2018. [En línea]. Disponible: https://www.autonomousvehicleinternational.com/features/the-prometheus-project.html [Último acceso: Mayo 2021].

[8] B. Gringer, « History of the autonomous car » 2018. [En línea]. Disponible: https://www.titlemax.com/resources/history-of-the-autonomous-car/ [Último acceso: Mayo 2021]

[9] D. Sanders, « European Eureka Project - PROgraMme for a European Traffic of Highest Efficiency and Unprecedented Safety (PROMETHEUS) » 2018. [En línea]. Disponible: https://researchportal.port.ac.uk/portal/en/projects/european-eureka-project--programme-for-a-european-traffic-of-highest-efficiency-and-unprecedented-safety-prometheus(7258e5d4-51ec-4114-84b1-f107e2dd9cb5).html [Último acceso: Mayo 2021]

[10] D. Galán, «Que acelere el coche autónomo: Waymo recibe una inyección de 2.250 millones de dólares para impulsar el adiós al conductor» 2020. [En línea]. Disponible: https://www.motorpasion.com/tecnologia/que-acelere-coche-autonomo-waymo-recibe-inyeccion-2-250-millones-dolares-para-impulsar-adios-al-conductor [Último acceso: Mayo 2021]

[11] Página oficial Waymo, «Waymo Stoty», 2020. [En línea]. Disponible: https://waymo.com/company/#story [Último acceso: Mayo 2021]

[12] E. de Aragón, «Repasamos las ayudas que los gobiernos europeos están ofreciendo para la adquisición de vehículos electrificados», 2020. [En línea]. Disponible: https://movilidadelectrica.com/ayudas-europeas-vehiculos-electrificados/ [Último acceso: Mayo 2021]

[13] Página Oficial Plan MOVES II, «PLAN MOVES II », 2020. [En línea]. Disponible: https://www.idae.es/ayudas-y-financiacion/para-movilidad-y-vehiculos/plan-moves-ii. [Último acceso: Mayo 2021].

[14] Statista, «Los automóviles eléctricos, todavía en minoría en Europa», 2019. [En línea]. Disponbile: https://es.statista.com/grafico/16084/proporcion-de-coches-electricos-sobre-matriculaciones-totales/. [Último acceso: Mayo 2021].

[15] D. Carrington, «Global sales of electric cars accelerate fast in 2020 despite pandemic», 2021. [En línea]. Disponible: https://www.theguardian.com/environment/2021/jan/19/global-sales-of-electric-cars-accelerate-fast-in-2020-despite-covid-pandemic. [Último acceso: Mayo 2021].

[16] T. Fuentes, «Las ventas de coches subirán un 18% en 2021», 2021. [En línea]. Disponible: https://www.cocheglobal.com/mercado/ventas-coches-previsiones-evolucion-subida-2021-2022_441108_102.html. [Último acceso: Mayo 2021].

[17] Página oficial de TwizyLine, «Drawing The Future», 2021. [En línea]. Disponible: http://twizyline.com/. [Último acceso: Mayo 2021].

[18] S. Pilar Arnanz, trabajo final de grado: «Back-end implementation for an automatized car parking», 2020.

[19] Página oficial del TwizyContest2020, «TwizyContest2020 Off the Street», 2021. [En línea]. Disponible: https://www.twizycontest.com/. [Último acceso: Mayo 2021].

[20] M. Martín Fernández, «Renault and the University of Valladolid welcome TwizyLine», 2020. [En línea]. Disponible: http://twizyline.com/renault-and-the-university-of-valladolid-welcome-twizyline. [Último acceso: Mayo 2021].

[21] Prometeo, «Prometeo: Resolución 2019/2020», 2019. [En línea]. Disponible: https://innovacion.funge.uva.es/wp-content/uploads/2019/11/Acta-resoluci%c3%b3n-PROMETEO-2020_Firmada.pdf. [Último acceso: Mayo 2021].

[22] T-CUE, « Ganadores de la edición 2020 del concurso "Iniciativa Campus Emprendedor" », 2020. [En línea]. Disponible: https://www.redtcue.es/campus/campus-2020/ganadores. [Último acceso: Mayo 2021].

[23] A. Mazaira Hernández, trabajo final de grado: «Front-end implementation for an automatized car parking », 2020.

[24] I. Royuela González, trabajo final de grado: «Four level autonomous vehicle for an automatized parking », 2020.

[25] M. Martín Fernández, trabajo final de grado: «Plan de Comunicación de TwizyLine: plataforma de carsharing con aparcamiento autónomo», 2020.

[26] T. de Francisco Aparicio, «Simulación de la Unidad de Control Motor y Sistemas de Asistencia a la Conducción dentro de una plataforma de integración electrónica de Renault Captur», 2020.

[27] T. Systems, «Actualizaciones OTA», 2021. [En línea]. Disponible: https://www.t-systems.com/es/es/industries/automotive/connected-mobility/over-the-air-update. [Último acceso: Mayo 2021].

[28] Página oficial SAE, «SAE Standards News: J3016 automated-driving graphic update», 2019. [En línea]. Disponible: https://www.sae.org/news/2019/01/sae-updates-j3016-automated-driving-graphic [Último acceso: Mayo 2021].

[29] Página de km77, «Conducción autónoma | Niveles y tecnología», 2018. [En línea]. Disponible: https://www.km77.com/reportajes/varios/conduccion-autonoma-niveles [Último acceso: Mayo 2021].

[30] R. Saúl Cova Rocamora, «Navegación y conducción autónoma de vehículos con geometría Ackermann», 2019. [En línea]. Disponible: http://rua.ua.es/dspace/handle/10045/94755 [Último acceso: Mayo 2021].

[31] M. Murphy, «128-laser LiDAR sensor significantly sharpens autonomous cars' vision», 2017. [En línea]. Disponible: https://newatlas.com/velodyne-lidar-vls-128-sensor/52453/ [Último acceso: Mayo 2021].

[32] Página Oficial de Tesla, «Tesla Self-Driving Demostration», 2016. [En línea]. Disponible: https://www.tesla.com/videos/autopilot-self-driving-hardware-neighborhood-long. [Último acceso: Mayo 2021].

[33] A. Díez Ramírez, «Conducción autónoma: Estudio del estado del arte, impacto sobre la movilidad y desarrollo de simulador de tráfico.», 2019. [En línea]. Disponible: http://oa.upm.es/53520/1/TFG_ALFONSO_DIEZ_RAMIREZ.pdf [Último acceso: Mayo 2021].

[34] Página Oficial de Tesla, «Piloto automático», 2019. [En línea]. Disponible: https://www.tesla.com/es_ES/autopilotAI [Último acceso: Mayo 2021].

[35] M. Ben-Daya et al, «Handbook of Maintenance Management and Engineering», 2009. [En línea]. Disponible: https://link-springer-com.ponton.uva.es/content/pdf/10.1007%2F978-1-84882-472-0.pdf [Último acceso: Mayo 2021].

[36] General Motors, «Self-Driving Safety Report », 2018. [En línea]. Disponible: https://www.gm.com/content/dam/company/docs/us/en/gmcom/gmsafetyreport.pdf [Último acceso: Mayo 2021].

[37] Waymo, «Waymo Safety Report», 2020. [En línea]. Disponible: https://storage.googleapis.com/sdc-prod/v1/safety-report/2020-09-waymo-safety-report.pdf [Último acceso: Mayo 2021].

[38] C. Prego, «Qué dice la legislación española sobre los coches autónomos: una instrucción y muchas incógnitas», 2019. [En línea]. Disponible: https://www.xataka.com/vehiculos/que-dice-legislacion-espanola-coches-autonomos-instruccion-muchas-incognitas [Último acceso: Mayo 2021].

[39] Ministerio del Interior (España), «Acuerdo para el desarrollo del vehículo autónomo», 2020. [En línea]. Disponible: https://revista.dgt.es/es/noticias/nacional/2020/09SEPTIEMBRE/0924Espana-y-Francia-acuerdo-conduccion-automatizada.shtml [Último acceso: Mayo 2021].

[40] The Tesla Team, «Dual Motor Model S and Autopilot», 2014. [En línea]. Disponible: https://www.tesla.com/blog/dual-motor-model-s-and-autopilot?redirect=no [Último acceso: Mayo 2021].

[41] T. Bensmann, «Tesla Autopilot 2.0 vs. 1.0 – status», 2017. [En línea]. Disponible: https://bensmann.no/tesla-autopilot-2-vs-1-status/ [Último acceso: Mayo 2021].

[42] The Tesla Team, «El futuro de la conducción», 2017. [En línea]. Disponible: https://www.tesla.com/es_ES/autopilot [Último acceso: Mayo 2021].

[43] Mercedes Benz España, «Sistema DISTRONIC PLUS | Mercedes-Benz España», 2017. [En línea]. Disponible: https://www.youtube.com/watch?v=Z7u3m0h8C1k [Último acceso: Mayo 2021].

[44] Vida Premium, «Aparcamiento automatizado», 2017. [En línea]. Disponible: https://www.vidapremium.com/aparcamiento-automatizado-2472.htm#4 [Último acceso: Mayo 2021].

[45] Audi Singapore, «The Audi AI traffic jam pilot technology», 2017. [En línea]. Disponible: https://www.youtube.com/watch?v=nUlK6fpveXg [Último acceso: Mayo 2021].

[46] A. Au, C. Chung, «Autonomous vehicles», 2017. [En línea]. Disponible: https://www.newmediabusinessblog.org/index.php/Autonomous_Vehicles#Firefly [Último acceso: Mayo 2021].

[47] A. Callejo, «Waymo hace un pedido de 62.000 Chrysler Pacífica, mientras que FCA se plantea vender coches autónomos al gran público», 2017. [En línea]. Disponible: https://forococheselectricos.com/2018/06/waymo-hace-un-pedido-de-62-000-chrysler-pacifica-mientras-que-fca-se-plantea-vender-coches-autonomos-al-gran-publico.html [Último acceso: Mayo 2021].

[48] Toyota Research Institute, «Platform 3.0 - TRI's New Automated Driving Research Vehicle», 2017. [En línea]. Disponible: https://www.youtube.com/watch?v=CNdmvscHcgU&t=4s [Último acceso: Mayo 2021].

[49] R. Álvarez, «Se confirma que el coche autónomo de Uber que mató a una mujer no estaba programado para detectar y actuar ante peatones imprudentes», 2019. [En línea]. Disponible: https://www.xataka.com/automovil/se-confirma-que-coche-autonomo-uber-que-mato-a-mujer-no-estaba-programado-para-detectar-actuar-peatones-imprudentes [Último acceso: Mayo 2021].

[50] El Pais, «Uber abandona y vende su división de coches autónomos», 2020. [En línea]. Disponible: https://elpais.com/economia/2020-12-08/uber-abandona-y-vende-su-division-de-coches-autonomos.html [Último acceso: Mayo 2021].

[51] CincoDías, «Uber se pone en cabeza en la carrera del vehículo autónomo», 2016. [En línea]. Disponible: https://cincodias.elpais.com/cincodias/2016/12/14/empresas/1481721756_459319.html [Último acceso: Mayo 2021].

[52] C. Dunkel, «Adaption eines Versuchsträgers und Entwicklung von Funktionprototypen für automatisierte Fahrfunktionen», 2017. [En línea]. Disponible: https://docplayer.org/55196856-Diplomarbeit-adaption-eines-versuchstraegers-und-entwicklung-von-funktionsprototypen-fuer-automatisierte-fahrfunktionen.html [Último acceso: Mayo 2021].

[53] R. Borraz, P. M. Alcover, P. J. Navarro Lorente, «Cloud Incubator Car: A Reliable Platform for Autonomous Driving», 2018. [En línea]. Disponible: https://www.mdpi.com/2076-3417/8/2/303 [Último acceso: Mayo 2021].

[54] B. K. Cóndor Romero, «Control autónomo del Renault Twizy y su modelo 3D en Gazebo», 2019. [En línea]. Disponible: https://idus.us.es/handle/11441/99547 [Último acceso: Mayo 2021].

[55] IEEE Computer Society, «802.11p: Amendment 6, Wireless Access in Vehicular Environments», 2010. [En línea]. Disponible: https://ieeexplore-ieee-org.ponton.uva.es/stamp/stamp.jsp?tp=&arnumber=5514475 [Último acceso: Mayo 2021].

[56] ETSI, «automotive Intelligent Transport Systems (ITS)», 2010. [En línea]. Disponible: https://www.etsi.org/technologies/automotive-intelligent-transport [Último acceso: Mayo 2021].

[57] Qualcomm Technologies, «How NR based sidelink expands 5G C-V2X to support new advances use cases», 2020.

[58] 5GAA, «About us», 2016. [En línea]. Disponible: https://5gaa.org/about-5gaa/about-us/ [Último acceso: Mayo 2021].

[59] 5GAA, «Timeline for deployment of C-V2X- Update», 2019. [En línea]. Disponible: https://5gaa.org/wp-content/uploads/2019/01/5GAA_White-Paper-CV2X-Roadmap.pdf [Último acceso: Mayo 2021].

[60] Telefónica, «Telefónica y Seat mostrarán en el MWC varios casos de uso de coche conectado con 5G en un entorno de ciudad para una conducción más segura», 2019. [En línea]. Disponible: https://www.telefonica.com/es/web/sala-de-prensa/-/telefonica-y-seat-mostraran-en-el-mwc-varios-casos-de-uso-de-coche-conectado-con-5g-en-un-entorno-de-ciudad-para-una-conduccion-mas-segura [Último acceso: Mayo 2021].

[61] T. Mirfakhraie, G. Vitor, K. Grogan, «Applicable Protocol for Updating Firmware of Automotive HVAC Electronic Control Units (ECUs) Over the Air», 2018. [En línea]. Disponible: https://ieeexplore.ieee.org/document/8726718 [Último acceso: Mayo 2021].

[62] A. Cheng, J. Yin, D. Ma, X. Dang, «Application and Research of Hybrid Encryption Algorithm in Vehicle FOTA System», 2020. [En línea]. Disponible: https://ieeexplore.ieee.org/document/9164481 [Último acceso: Mayo 2021].

[63] Autobild, «Renault Twizy 2012», 2020. [En línea]. Disponible: https://www.autobild.es/coches/renault/twizy/twizy-0-2012 [Último acceso: Mayo 2021].

[64] Asti Mobile Robotics, «A30 ASEL.061 V2» 2019.

[65] Lv-MaxSonar-EZ Series, «MB1010», 2010. [En línea]. Disponible: https://docs.rs-online.com/9885/0900766b8153438a.pdf [Último acceso: Mayo 2021].

[66] GitHub del Grupo de Comunicaciones Ópticas (GCO), «TwizyContest», 2020. [En línea]. Disponible: https://github.com/GCOdeveloper/Twizycontest [Último acceso: Julio 2021].

[67] Maxon Motor Ibérica S.A.U, «EC 60 flat Ø60mm, Conmutación electrónica (Brushless), 100W», 2020.

[68] Maxon Motor Ibérica S.A.U, «Encoder MILE 512-4096 ppv, 2 canales, con line driver», 2020.

[69] Maxon Motor Ibérica S.A.U, «Start-up with EPOS Studio», 2015.

[70] Maxon Motor Ibérica S.A.U, «EPOS: Command Library Documentation», 2021.

[71] Maxon Motor Ibérica S.A.U, «EPOS: Firmware Specification», 2020.

[72] SWIG, «Welcome to SWIG» 2020. [En línea]. Disponible: http://www.swig.org/ [Último acceso: Mayo 2021].

[73] S. Podar Cristea, «Tema4: Acciones básicas de control», 2020.

[74] J. C. Aguado Manzano, A. N. Gentil Otero, N. H. Cabello Martínez, «7. Diagnosis», 2019.

[75] Renault Group, «Manual de Renault Twizy», 2019.

[76] Maxon Motor Ibérica S.A.U, «EPOS4 70/15: Hardware Reference», 2020.

Anexo 1

En este anexo se introducirá una hoja de especificaciones con las características de nuestro Renault Twizy. En la Figura 90, podemos ver las características generales del vehículo.

	TWIZY 45	TWIZY 80
Homologación	**Cuadricicio ligero (L6e)**	**Cuadricicio pesado (L7e)**
TVV		
Nivel de emisión	Cero emisiones: 100 % eléctrico	
Número de plazas	2 (1 en Cargo)	
MOTOR		
Tipo motor	3CG - eléctrico asíncromo	
Potencia kW CEE (cv)	4 (5)	13 (17)
Par máx. Nm CEE (m.kg)	33	57
Régimen par máx. (r.p.m.)	de 0 a 2.050 r.p.m.	de 0 a 2.100 r.p.m.
Carburante	Eléctrico	
CAJA DE VELOCIDADES		
Manual - Automática	Automático	
Tipo	Reductor	
Relación de desmultiplicación	1 : 13,4	1 : 9,23
Número de relaciones A.V.	1	
PRESTACIONES		
Velocidad máx. (km / h)	45	80
50 m salida parada (s)	7,5	6,6
0-45 km/h (s)	9,9	6,1
30-60 km/h (s)	5 (hasta 45 km/h)	8,1
CONSUMO CICLO URBANO ECE-15 (en l/100 km y g/km)		
CO_2 (g / km)	0	
Autonomía certificada* ECE-15 (km)	100	90
Autonomía real** (km)	75 a 85	60 a 70
Wh / km	58	63
DIRECCIÓN		
Asistida	cremallera directa	
Ø de giro entre aceras (m)	6,8	
Número de giros del volante	2,8	
TRENES		
Tipo tren delantero	Pseudo-Mc Pherson - Combinado muelle/ amortiguador/ eje flexible	
Tipo tren trasero	Pseudo-Mc Pherson - Combinado muelle/ amortiguador/ eje flexible	
Ø barra estabilizadora delantera/ trasera (mm)	Delantera y trasera: diámetro 23 mm	
RUEDAS Y NEUMÁTICOS		
Llantas de referencia	33 cm (13")	
Dimensiones neumáticos delanteros	Continental EcoContact 125/80 R13	
Dimensions neumáticos traseros	Continental EcoContact 145/80 R13	
FRENOS		
Tipo de circuito	Circuito simple	
Discos delanteros plenos (Ø en mm)	214	
Discos traseros plenos (Ø en mm)	204	
AERODINÁMICA Y CAPACIDAD		
SCx/ Cx	0,64	
Capacidad de energía (kWh)	6,1	
MASAS (kg)		
En vacío en orden de marcha (sin batería)	446 (375)	474 (375)
En vacía en orden de marcha delante	197	206
En vacío en orden de marcha atrás	249	268
Total (MTR)	685	690
Carga útil (CU)	110	115
Masa máx. remolcable frenada	0	
Masa máx. remolcable no frenada	0	

Figura 90. Características generales para los dos modelos de Renault Twizy [75].

En la Figura 91 tenemos las dimensiones del Renault Twizy, las cuales fueron de gran necesidad a la hora de diseñar el parking autónomo del Twizy. Y en la Figura 92 se muestra el esquema eléctrico del Twizy, donde se puede apreciar en la parte inferior de color verde la pila general de 48 V que suministra energía la coche cuando está en movimiento, y en la parte delantera de color azul claro se puede ver la pila de 12V, cuyo uso sirve para arrancar el coche entre otras funcionalidades.

VOLUMEN PORTAOBJETOS (dm³)	
Volumen del maletero	31 (180 en Cargo)
Volumen guantera izquierda	3,5
Volumen guantera derecha	5

MEDIDAS (mm)		
C	Voladizo delantero	313
A	Batalla	1686
D	Voladizo trasero	339
B	Longitud total	2338
E	Vía delantera	1094

MEDIDAS (mm)		
F	Vía trasera	1080
	Anchura delantera	1237
G	Anchura trasera	1232
	Anchura con retrovisores/ con puertas	1381/1396
H	Altura en vacío	1454
H1	Altura en vacío con puertas abiertas mín./máx.	1818/1980
K	Altura libre en carga	120
L	Reglaje longitudinal asiento delantero	200
P	Altura delantera	908
Q	Altura trasera	843

Figura 91. Dimensiones del Renault Twizy [75].

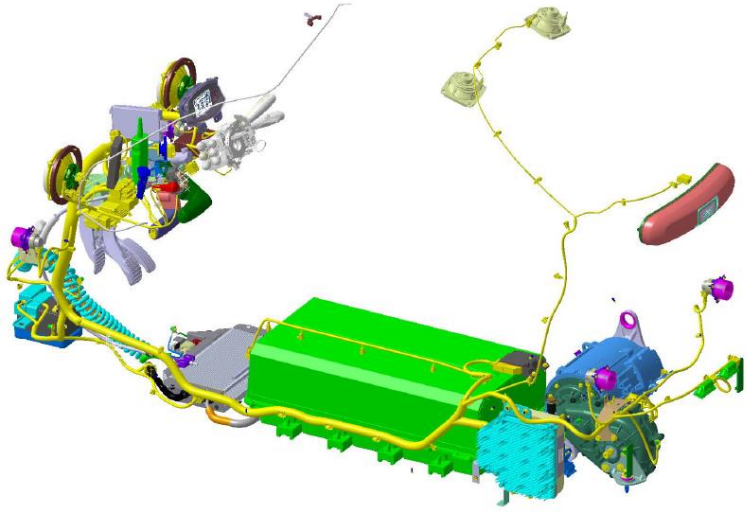


Figura 92. Arquitectura eléctrica del Twizy. [75]

Anexo 2

Este anexo se dedicará a exponer los tópicos usados en MQTT para el intercambio de los diferentes mensajes entre el vehículo y el servidor. El parámetro 'id_car' se refiere al id asignado a cada coche en dentro del sistema TwizyLine.

Publicador	Subscriptor	Tópico	Descripción
Servidor	Twizy	id_car/fotaReady	El servidor pregunta al coche con id igual a id_car si está preparado para un procedimiento de FOTA.
Twizy	Servidor	id_car/carReady	El coche responde que sí que está disponible la configuración FOTA tras ser preguntado por su disponibilidad.
Servidor	Twizy	id_car/fotaECU	El servidor pregunta por la información general de una ECU. El mensaje contiene el ID de la trama CAN, para poder comunicarse con la ECU seleccionada.
Twizy	Servidor	id_car/fotaGinfo	El coche envía toda la información general de la ECU al servidor.
Servidor	Twizy	id_car/fotaUnits	El servidor ordena cambiar las unidades a las seleccionadas en la interfaz gráfica por el usuario. En el mensaje se envía "ML" o "KM" dependiendo de la unidad elegida.
Servidor	Twizy	id_car/fotaSpeed	El servidor ordena modificar los kilómetros o millas que marca el odómetro. En el mensaje MQTT se especifica el nuevo número introducido por el usuario.

Tabla 2. Tópicos empleados en las comunicaciones MQTT.

Anexo 3

En este anexo se mostrará cuál ha sido el cableado necesario y las conexiones realizadas entre la controladora y el motor.

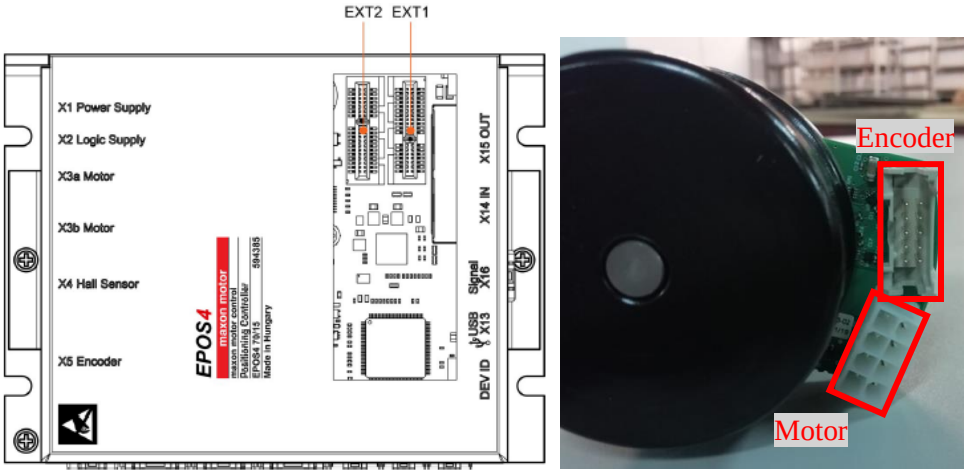


Figura 93. A la izquierda esquema de las conexiones de la controladora EPOS4-70/15 y a la derecha las conexiones del motor EC60 flat (más encoder y reductora).

En la Figura 93 se muestra las conexiones disponibles en los dos elementos que queremos conectar. Seguiremos el esquema de conexión general de la hoja de especificaciones de la EPOS4-70/15 [76].

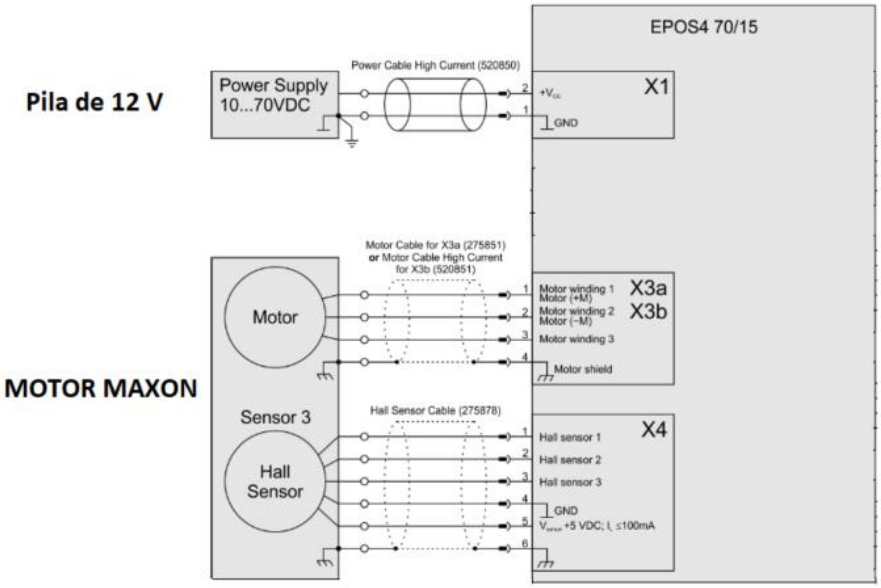


Figura 94. Esquema de las conexiones entre la el motor y la EPOS4 con su respectiva alimentación [76].

A continuación, se enumerará el material necesario para cada una de las conexiones realizadas, cabe destacar que las medidas de los cables vienen dadas por la referencia estadounidense de sección de cable, llamada AWG (*American Wire Gauge*):

- Fuente de alimentación y la EPOS4: Conector macho de sección 2x2,5mm^2, cable de 14 AWG (2,5mm^2), 2 crimps de AWG 14.

- Pines del motor: Un conector macho de sección 4x0,75mm^2, un conector macho de sección 4x2,5mm^2 (estos dos se enchufan a la EPOS4) y un conector de 6x0,75mm^2 (a los pines del motor). Cable AWG 14 y 20, además de los respectivos crimps para esa sección de cable.
- Encoder: 2 conectores macho de sección 10xAWG28, se encuentra bajo el nombre de DIN 41651 IDC. Bus de 10 cables de AWG 28. En caso de usar el conector mencionado no hace falta crimps.

Anexo 4

Este anexo servirá para mostrar con más detalle los elementos usados para mantener el motor en la ubicación en la cual finalmente se decidió implementarlo.

Se comenzará hablando del soporte, que está atornillado al chasis del coche para sujetarse en esa posición. En primera instancia se creó un soporte de prueba realizado en PLA (un tipo de plástico) por el FABLAB de la FUNGe (Figura 95). Este diseño ayudó a determinar las medidas exactas del soporte final.

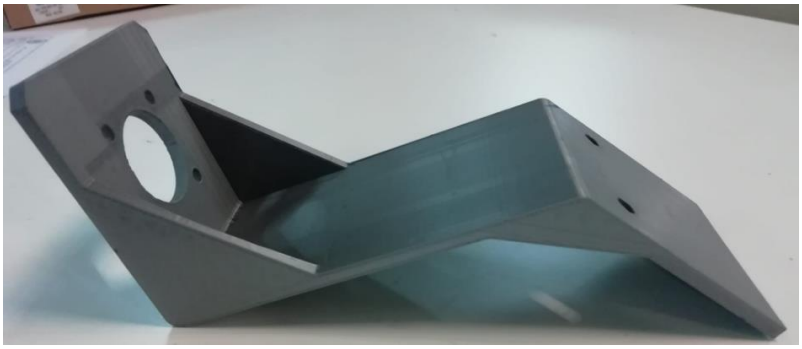


Figura 95. Soporte de prueba con material PLA.

El diseño final se realizó sobre metal, las medidas del soporte son parecidas, pero se añadió nuevas soldaduras para conectar las diferentes partes del soporte para que sea capaz de soportar el efecto rotatorio del motor y no se mueva. En la Figura 96, se aprecia como se han añadido dos varillas de metal para unir la tapa superior con la conexión entre la pared vertical y la tapa inferior. Además, se mantiene la soldadura entre la parte superior de la tapa inferior y la pared vertical. Los agujeros de la tapa superior son para introducir los tornillos que lo sujeten al chasis del coche, los agujeros de la tapa inferior son para sujetas el motor al soporte. En la Figura 38, se puede observar cómo es la integración final de este soporte.

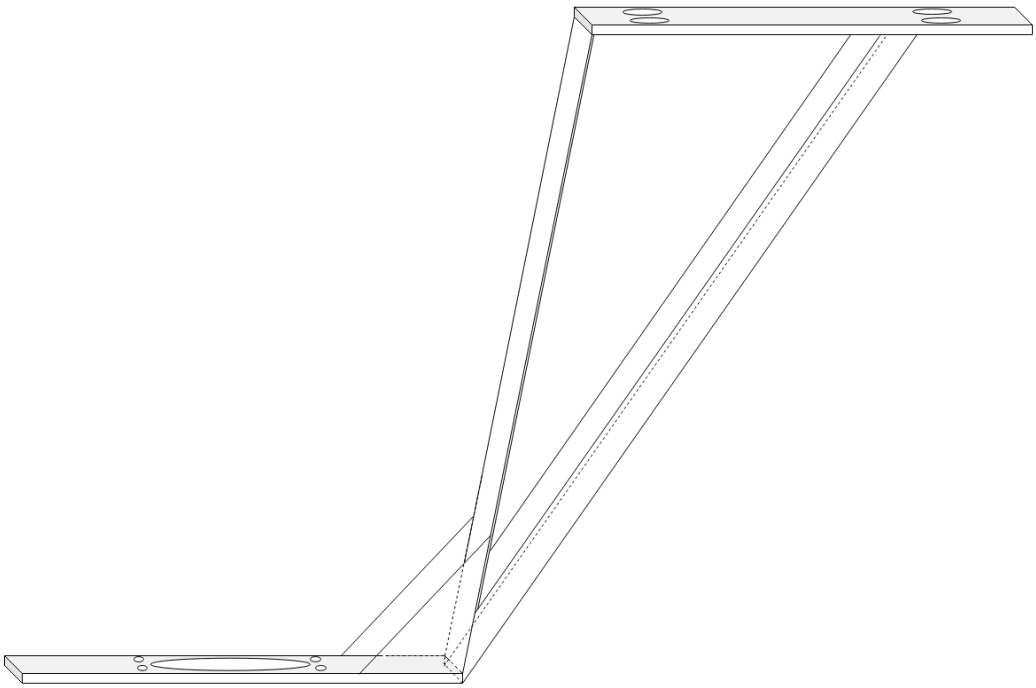


Figura 96. Estructura del soporte final, implementado en metal.

Para la transmisión del efecto rotatorio del motor a la columna de dirección se mandó fabricar dos engranajes donados por la empresa Luce IT. En la Figura 97, se observa un esquema de los engranajes que se han utilizado, siendo el de la izquierda es que va sujeto a la columna de dirección, se necesitó taladrarla para introducir unos tornillos y así evitar que se desplace. La pieza del centro se utiliza para unirse al motor a través de la cara con la muesca y se une al engranaje de la izquierda con tornillos. Por último, el engranaje de la izquierda aparte de estar unido a la pieza del centro, se encaja con el otro engranaje, para transferir el efecto rotatorio.

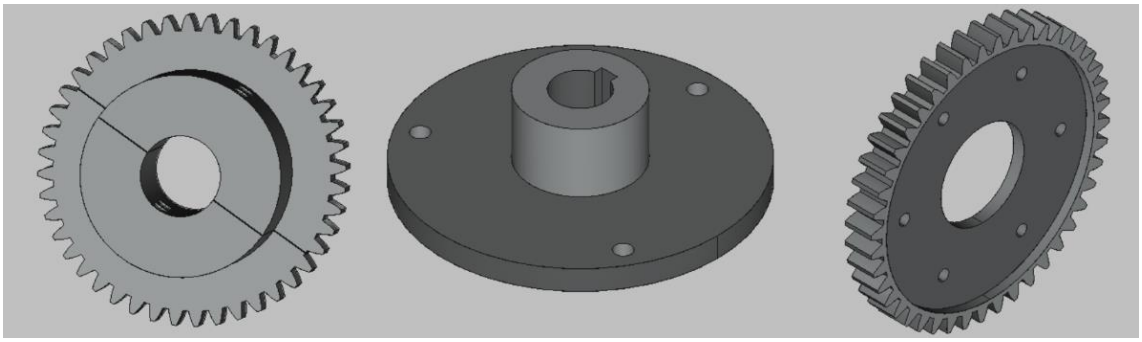


Figura 97. Esquema de los engranajes implementados en la columna de dirección (izquierda), y en el motor (engranaje del centro).

A continuación, se muestra donde se consiguió mantener el engranaje del motor sujeto a este sin que deslizase hacia abajo cuando estaba en movimiento. En la Figura 98 se puede ver las partes que conforman la estructura ideada para bloquear el movimiento hacia abajo del engranaje. Cabe destacar que el motor tenía al final del embolo una ranura para introducir un tornillo que sujeta la pletina de metal. En la Figura 39 se podía ver la implementación final de cada parte.

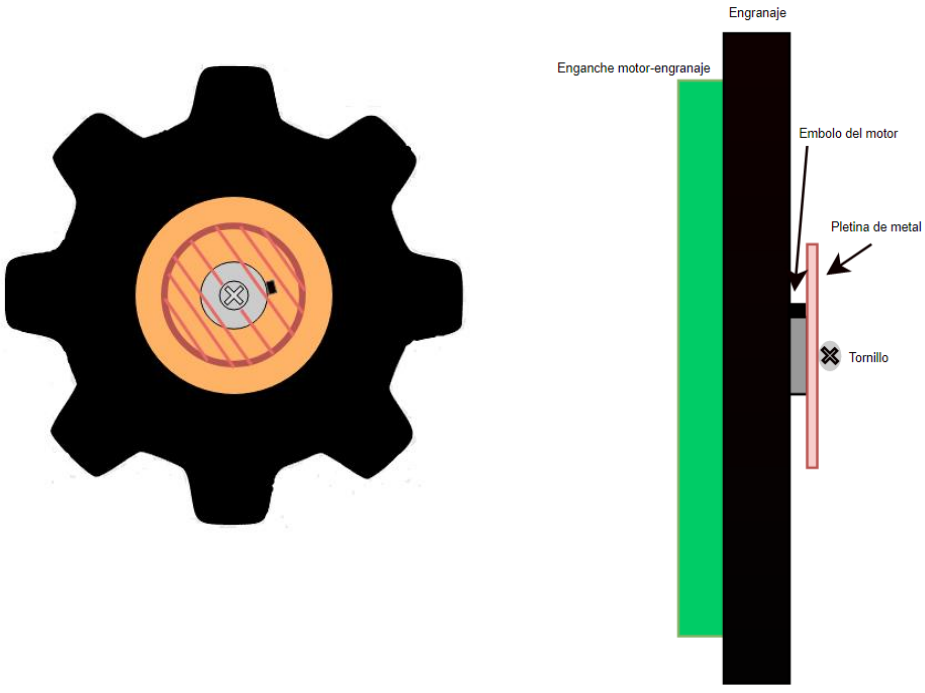


Figura 98. Partes implicadas en sujetar el engranaje del motor, vista al alza (izquierda) y vista de perfil (derecha).

Anexo 5

Este anexo se empleará para mostrar el contenido de los ficheros que caracterizan los elementos empleados en la simulación. Estos ficheros se llamaron "configuracion_XX.m" y se encuentran al igual que todo el código empleado en este proyecto en el GitHub del Grupo de Comunicaciones Ópticas [66]. A continuación, se mostrará en tablas (Tabla 3, Tabla 4, Tabla 5 y Tabla 6) cada uno de esos ficheros junto al parámetro, valor y descripción.

"configuracion_motor.m"		
Velocidad angular máxima	wm = 3000 rpm	Máximas revoluciones por minuto a las cuales el motor aguantaba, sin entrar en modo de error (lo que supone la desactivación del motor y perder el control sobre el volante).
Factor de la reductora	G = 81	Según las especificaciones las reductora planetaria GP52C.
Máxima aceleración	aceleracion_revoluciones_maxima = 10000 m^2/seg	Aceleración máxima disponible por el motor según sus especificaciones.
Modo de operación	control_motor='angulo---'	Como ya se explicó, podemos funcionar en con un perfil de posición o de velocidad. En este caso, se controla por posición.
Aceleración infinita	control_aceleracion='ON-'	Según el motor tenga aceleración infinita o no.

Tabla 3. Parámetros del fichero configuracion_motor.m

"configuracion_sensor.m"		
Refresco del sensor	S = 0,01 seg	Periodo de refresco del valor del sensor.
Distancia de la posición del sensor respecto al eje delantero	d = 0,2 m	Medidas tomadas del prototipo tras su integración.
Precisión	precisión_sensor = 0,001 m	Precisión que tiene el sensor magnético
Longitud	lsensor=0,085 m	Longitud del sensor magnético

Tabla 4. Parámetros del fichero configuracion_sensor.m

"configuracion_vehiculo.m"		
Distancia entre ejes	L = 1,69 m	Medido en el prototipo que tenemos a disposición.
Delta máximo objetivo	delta_max=0,6545 rad	Giro máximo de las ruedas, equivalente a 37,5º.
Vueltas máximas	vueltas_max = 1,3	Máximo número de vueltas que puede dar a cada lado del vehículo.
Centro del eje	lr = L/2 m	Suponemos que el centro está en mitad de los ejes.
Centro de masas	lf = L-lr	Distancia del eje delantero al centro de masas.

Tabla 5. Parámetros del fichero configuracion_vehiculo.m.

"configuracion_simulador.m"		
Pasos temporales	paso_tiempo = 0,002 s	Periodo del tiempo en el simulador.
Pasos espaciales	paso_distancia = 0.01 m	Distancia entre dos puntos de la ruta.
Tiempo de medición de la distancia	paso_sensor_magnetico = round(S/paso_tiempo) s	Periodo de medición de la distancia entre el coche y la banda magnética, siendo S el periodo de refresco del sensor magnético especificado en su fichero de configuración.

Tabla 6. Parámetros del fichero configuracion_simulacion.m