Universidad Politécnica de Madrid

ETSI SISTEMAS INFORMÁTICOS

POLITÉCNICA "Ingeniamos el futuro"

CAMPUS DE EXCELENCIA INTERNACIONAL

SISTEMA DE PREVENCIÓN Y DETECCIÓN DE COLISIONES EN CADENA DE VEHÍCULOS

Titulación: Grado en ingeniería de Computadores Plan 2014

Curso: 4º curso

Autor: Jose Antonio Alcaide Recio

Director-tutor: Javier García Martín

16 DE JULIO DE 2018

ETSISI - UNIVERSIDAD POLITÉCNICA DE MADRID

Índice

1. Resumen

La falta de atención en la carretera es una de las principales causas de accidentes de tráfico. Automatizar las tareas más críticas para la conducción es un problema complejo y preciso el cual implica una capacidad de cómputo grande debido a los datos masivos recogidos del medio que rodea al vehículo en ese preciso instante. Se han realizado diferentes enfoques entre ellos el Autopilot de Tesla que tiene el potencial de procesar gran cantidad de datos en Tiempo Real. Pero no solo eso, también se ha desarrollado un sistema de aviso en caso de accidente, el sistema eCall, capaz de alertar a los servicios sanitarios indicando la ubicación del accidente.

En este Proyecto he desarrollado un sistema que puede controlar la velocidad del vehículo en base a la distancia de seguridad que permita al automóvil frenar con seguridad en caso de emergencia. El otro principal objetivo de este sistema es que, en caso de accidente, envía la ubicación de este a los demás vehículos que dispongan de este sistema y les aparecerá en sus monitores un mapa con dicha ubicación para que puedan elegir otras rutas o estar atentos si se encuentran cerca del accidente. Este sistema ayuda a reducir la cantidad de accidentes debidos a conductores que no miden correctamente la distancia de seguridad, una práctica bastante habitual entre estos. También, disminuye el tiempo de reacción de los servicios sanitarios en caso de accidente, puesto que el aviso es instantáneo una vez colisionado el vehículo.

El sistema propuesto funcionará en tres etapas principales. En la primera etapa el sistema comprueba si el sistema ha sido activado por el usuario. En la segunda etapa comprueba cual es la acción que está tomando el conductor. En la última etapa si el sistema está activado el sistema recoge datos del medio que le rodea.

En base a los datos tanto del conductor como del medio que le rodea, el sistema toma decisiones para actuar sobre la velocidad del vehículo. Además, el sistema enviará una alerta en caso de accidente tanto si el sistema está activado como si no lo está.

1. Abstract

Driver inattention is one of the main causes of traffic accidents. Automating the most critical tasks for driving is a complex and precise problem which implies a large computing capacity due to the massive data collected from the environment surrounding the vehicle at that precise moment. Different approaches have been made, among them the Tesla Autopilot, which has the potential to process a large amount of data in Real Time. But not only that, a warning system has also been developed in case of an accident, the eCall system, capable of alerting emergency services indicating the location of the accident.

In this Project I have developed a system that can control the speed of the vehicle based on the safety distance that allows the car to brake safely in case of emergency. The other main aim of this system is that, in case of accident, it sends the location of this to the other vehicles that have this system and a map with that location will appear on their monitors so that they can choose other routes or be attentive if they are close to the accident. This system helps reduce the number of accidents due to drivers who do not correctly measure the safety distance, a fairly common practice among them. Also, it decreases the reaction time of the health services in case of accident, since the warning is instantaneous once the vehicle collides.

The proposed system will work in three main stages. In the first stage the system checks if the system has been activated by the user. In the second stage, check what action the driver is taking. In the last stage, if the system is activated, the system collects data from the environment that surrounds it.

Based on the data of both the driver and the environment that surrounds him, the system makes decisions to act on the speed of the vehicle. In addition, the system will send an alert in case of an accident, whether the system is activated or not.

2. Introducción

2.1. Descripción e importancia del problema

La conducción es una tarea compleja para las personas debido a que tienen que tomar decisiones y actuar en consecuencia en tan solo unos segundos, a veces incluso menos. A este rango de tiempo se le denomina **tiempo de reacción** y en personas en condiciones normales oscila entre 0,5 y 1,2 segundos [1] , varía según la edad y las condiciones físicas y psíquicas. Se compone de cuatro partes: **la percepción, la intelección, la emoción y la volición**. Si cualquiera de estas partes interfiere en el tiempo de reacción entonces tendremos un accidente. Los expertos concluyen que la elección de cada una de estas decisiones es la causa principal de los accidentes en carretera y diferente según el tipo de persona que sea, dependiendo de si están cansadas, distraídas, están bajo los efectos el alcohol...

El tiempo de reacción bajo los **efectos del alcohol** se puede multiplicar y si conducimos bajo efectos de fatiga este tiempo se puede multiplicar incluso por dos.

El tiempo de reacción de las personas puede variar según sus edades y el entorno dando lugar a una respuesta más rápida en personas más jóvenes. La siguiente tabla siguiente recoge los datos obtenidos por **la UPC ente 0,5 y 2 segundos**.

TIEMPO DE REACCIÓN			
Persona / Condiciones	De día	De noche	Noche y Tráfico en contra
Jóvenes hasta 35 años	0,5s	0,8s	1,2s
Adultos entre 36 y 60 años	0,8s	1,2s	1,5s
Tercera edad más de 60 años	1,2s	1,5s	2,0s

Otro factor condicionante es la **distancia de seguridad**, esto supone un serio problema a la hora de un posible accidente porque una incorrecta medida de seguridad puede causar la colisión con el coche delantero y posteriormente los coches sucesivos en cadena.

Muchas personas no calculan correctamente la distancia de seguridad y otros no la respetan. Un estudio llevado a cabo por la **Fundación Línea Directa en 2014** señala que cinco millones de conductores no respetan la distancia de seguridad y, lo que es peor, dos millones lo hacen sabiendo que la incumplen. En **2013** hubo más de **150 fallecidos por colisiones múltiples** por no respetar la distancia entre vehículos [2].

En 2002, la fundación Attitudes de Audi realizó un estudio que señalaba que en las conductas negligentes o incívicas reside el germen de la agresividad.

Entre las conductas irresponsables cuantificadas la de no respetar la distancia de seguridad es la que más **creció en 2013, hasta un 8%.**

Algunas veces podemos tener dificultades para establecer una medida de seguridad óptima, pero existe una operación sencilla que nos puede dar una idea de cuál debe ser la distancia que debemos guardar con el coche que nos precede y es la de calcula aproximadamente 0,5 metros por cada kilómetro/hora de velocidad que llevemos en ese instante, así mismo, si llevamos una velocidad de 100km/h la distancia de seguridad será de 50 metros. [3]

La norma establece que todo vehículo que circule detrás de otro habrá de hacerlo a una distancia que le permita detenerse en caso de frenazo brusco, sin colisionar con él, considerando la velocidad, el frenado y la adherencia.

Para realizar el **cálculo de la distancia de seguridad** de forma práctica en calzadas secas podemos apoyarnos en la **Regla del Cuadrado** que engloba la distancia de percepción y la distancia de reacción a una velocidad determinada.

Velocidad	**Distancia de seguridad (Regla del cuadrado)**
50 km/h	5^2 = 25 metros
90 km/h	9^2 = 81 metros
100 km/h	10^2 = 100 metros
120 km/h	12^2 = 144 metros

Si a esto le añadimos las diferentes **condiciones meteorológicas**, la distancia de seguridad variará dependiendo del agarre del neumático a la carretera, es por esto por lo que con lluvia la distancia de seguridad será mucho mayor que en un día soleado, entre otros.

La distancia de seguridad en una **calzada mojada** será la siguiente:

Velocidad	**Distancia de seguridad (Regla del cuadrado)**
50 km/h	2 x 5^2 = 50 metros
90 km/h	2 x 9^2 = 162 metros
100 km/h	2 x 10^2 = 200 metros
120 km/h	2 x 12^2 = 288 metros

La Dirección General de Tráfico (DGT) recomienda mantener una **distancia de tres segundos** que es el tiempo que se utiliza para realizar un frenazo de emergencia y prevenir un accidente.

El organismo Real Automóvil Club de España (RACE) determina que la distancia de seguridad se compone en esencia de dos distancias diferentes [4]:

- **La distancia de reacción**: es el espacio que recorre el vehículo en el tiempo que transcurre desde que nos percatamos que debemos frenar (este es el tiempo de reacción anteriormente descrito) hasta que actuamos en consecuencia para frenar el vehículo.

DISTANCIA DE REACCIÓN A 100 km/h

- **La distancia de frenada**: es el espacio en el que nuestro vehículo es capaz de detenerse por completo. Esto varía según el peso de la carga del vehículo, el estado de los neumáticos y del estado del asfalto.

La eficacia del sistema de frenado junto con la velocidad y el estado de la carretera juegan un papel fundamental en la distancia de seguridad. El sistema antibloqueo, las ayudas tecnológicas y un mantenimiento del sistema de frenado nos ayudará a disminuir la distancia de frenado.

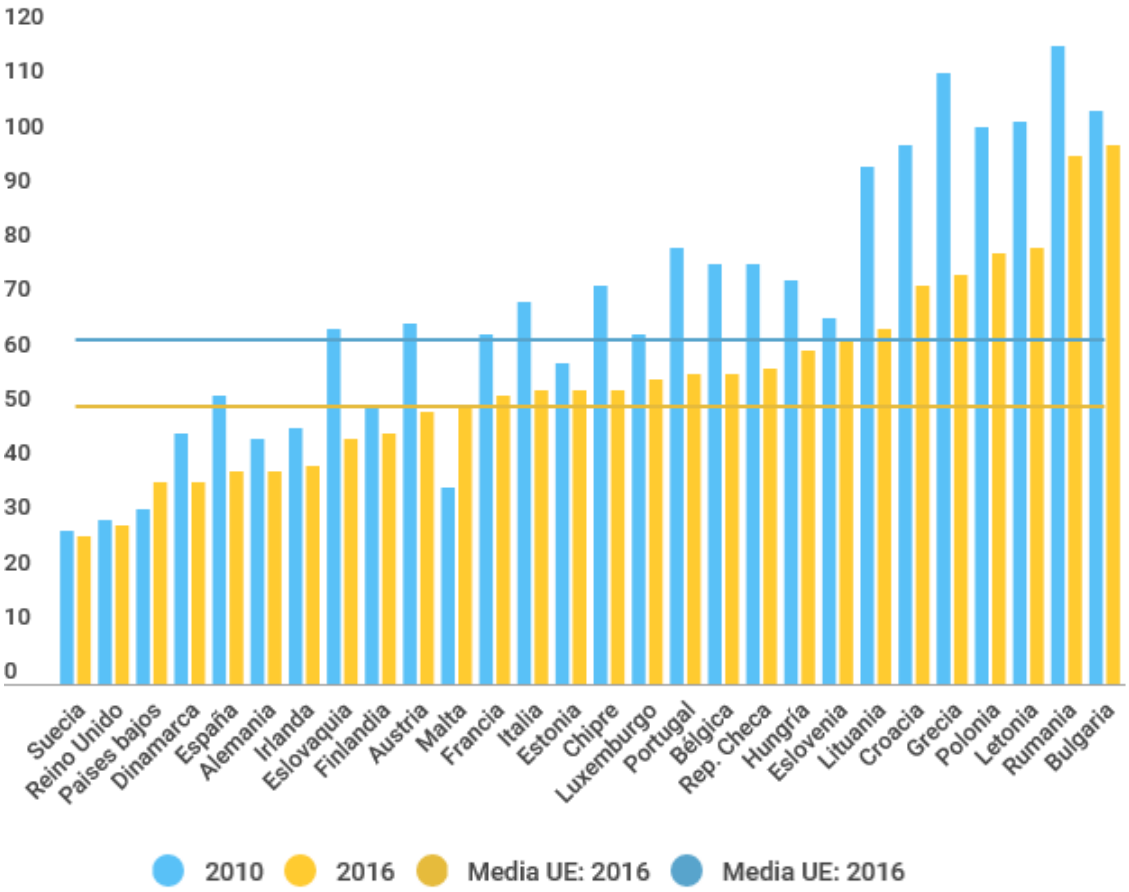
Todos estos problemas residen en un mismo conductor y en numerosas ocasiones estos accidentes provocan la muerte de los ocupantes del vehículo.

En 2016 murieron en accidente de tráfico 1810 personas, lo que representa un **7% más que en 2015.**

El balance definitivo de la siniestralidad vial de 2016 sitúa en 39 la tasa de fallecidos en accidente de tráfico por millón de habitantes, tres más que la registrada en 2015.

Aunque, para darnos un pequeño respiro nos encontramos por debajo de la media europea lo que no supone en ningún caso que esto no sea un verdadero problema para las vidas humanas.

En total, fallecieron 1810 personas en 102.362 accidentes, 121 personas más que en 2015 perdieron la vida en las carreteras [5].



Pero el **verdadero problema** está en cómo conseguir **reducir el número de víctimas en un accidente en cadena**, ya sea en el propio vehículo accidentado como en el resto de los coches involucrados en el mismo, porque es ahí donde nos encontramos un mayor número de víctimas.

Además, la misma solución encontrada para este tipo de accidentes se puede adoptar para los accidentes entre uno o dos vehículos.

Un **accidente en cadena** es aquel en el que se ven implicados dos o más vehículos, provocando daños tanto materiales como personales.

Para solucionarlo, debemos prevenirlo respetando las medidas de seguridad adoptadas por la Dirección General de Tráfico como la distancia entre vehículos o mejorar el tiempo de reacción con sistemas que sean capaces de actuar antes de la propia reacción del conductor, en caso de que el accidente se produzca alertar al resto de conductores para que se mantengan alerta y moderen su velocidad para así aumentar su tiempo de reacción y disminuir la distancia de frenado.

2.2. Trabajos previos que identifican el problema

Muchas investigaciones sobre sistemas de detección de distancias para automóviles han sido realizadas en los últimos años dejando un claro sistema denominado ADAS para ayudar a prevenir los accidentes de tráfico [6].

El Sistema Avanzado de Asistencia a la Conducción (ADAS) es un mecanismo que permite mejorar la seguridad del conductor y está dotado de tres tecnologías que permiten facilitar la conducción.

El Sistema de Mantenimiento de Carril (Sistema LKAS) controla a través de una cámara si el vehículo se está desviando de carril y el conductor no es consciente de ello.

El Control de Crucero Adaptativa (ACC) controla la distancia de seguridad con el vehículo que está delante de él a través de un radar de ondas milimétricas.

El Sistema de Mitigación de Impactos (CMBS) controla la velocidad del coche y avisa al conductor cuando este se aproxima demasiado al vehículo de delante.

Este sistema lo están integrando todas las marcas de coche debido a que proporciona una seguridad vital para el conductor.

Otro sistema importante a la hora de evitar un accidente por alcance es el **sistema de frenado automático** [7], también llamados sistemas de frenada de emergencia autónoma, cuando el conductor está distraído o incluso cuando no lo está, pero su capacidad de reacción es lenta o no responde.

Existen ciertos modelos de este sistema implementado en diferentes tipos de coches, algunos solo funcionan en ciudad hasta velocidades de 50km/h y otros son capaces de funcionar en carretera hasta velocidades de 210km/h sin garantizar el alcance del coche de delante, pero reducir, en mucho, la velocidad de impacto.

Para distancias comprendidas entre los diez y quince metros se suele utilizar una cámara, pero para distancias mayores se utiliza un radar o lidar para más precisión.

El sistema advierte al conductor con señales visuales, acústicas o un pequeño tirón de frenos, el sistema detecta si el usuario está frenando con la fuerza necesaria para frenar y en caso negativo el sistema frena con más fuerza.

Este sistema tiene sus limitaciones y por ello la actividad del conductor sigue siendo hoy en día necesaria.

Los sistemas basados en cámaras pueden no ser efectivos en situaciones de destello y por tanto no funcionan correctamente.

Los sistemas basados en radar pueden tener problemas si se acumula nieve o barro en la zona donde están colocados.

Todos los sistemas de frenado detectan vehículos de mucha masa como coches, furgonetas… pero dejan de surtir efecto cuando se trata de motos, bicicletas o peatones.

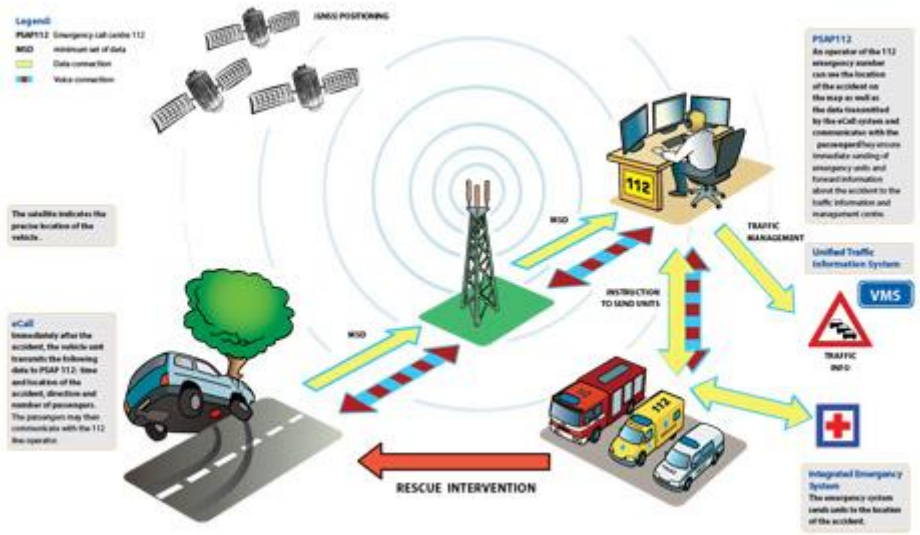
Estos sistemas no pueden controlar la presencia de farolas, vallas o guardarraíles ni tampoco son capaces de frenar ante un vehículo que se cruce delante de ellos perpendicularmente.

Lo ideal es combinar ambos sistemas, la cámara con el radar o el lidar para aumentar la fiabilidad del dato y actuar en consecuencia.

Pero no todo es conseguir que el coche evite un accidente, también se ha implementado el **Sistema eCall** [8] que tiene como objetivo ayudar a salvar vidas porque avisará de manera automática a **Emergencias 112** en caso de que se produzca dicho accidente.

Muchas veces el tiempo que se demora desde que se sufre un accidente hasta que llegan los servicios de emergencia puede ser tan largo que ya no se pueda hacer nada para salvar las vidas de las personas.

El sistema eCall detecta un posible accidente debido a los sensores instalados en el vehículo, en ese momento se graba un mensaje de ayuda que llegará a la central del 112 mediante GPS para facilitar la ubicación del vehículo. La llamada se podrá realizar de forma manual o automática si el accidente es grave.



Según diversos estudios científicos el **70% de las muertes** en accidentes de tráfico se produce a los 20-30 minutos.

Gracias al sistema eCall el tiempo de respuesta de los servicios de emergencia se podría disminuir un 50% en zonas rurales y un 40% en zonas urbanas.

El objetivo de este sistema es poder salvar hasta 2500 vidas al año gracias a su rapidez y eficacia según datos de la Comisión Europea y reducirla gravedad de las lesiones en un 15%.

Para efectuar el **despliegue del sistema eCall** se ha necesitado:

- Firmar el Memorándum de Acuerdo de eCall.
- Promover el uso del número europeo de emergencias 112.
- Modernizar los centros de emergencias para utilizar el sistema eCall.
- Mejorar los protocolos de actuación en caso de urgencia.

El sistema eCall requiere la coordinación de toda la red de comunicaciones, de manera que la llamada de urgencia pase por todos los canales y proveedores de telefonía móvil por los que necesite pasar con carácter prioritario y gratuito.

Pero estos sistemas no son los únicos implementados en los últimos años en los coches, también se ha implementado la capacidad de que un turismo se desplace automáticamente sin necesidad de conductor, este sistema lo han implementado varias firmas.

Por citar alguna de ellas, **Tesla con su sistema Autopilot** [9], es capaz de conducir y mantener una seguridad alta sin necesidad de conductor.

El sistema Autopilot consta de ocho cámaras para ofrecer una visión 360 grados alrededor del vehículo, doce sensores ultrasónicos, un radar delantero con procesamiento mejorado que le permite detectar objetos a través de lluvia intensa, neblina, polvo e incluso el vehículo que le antecede.

Todo este hardware tiene que ir de la mano de un software específico y para ello los desarrolladores de Tesla han aumentado hasta por 40 la potencia de procesamiento del nuevo ordenador integrado para comprender todos los datos recibidos por los sensores, este ordenador ejecuta una red neuronal para procesar visión, sonar y radar.

Este sistema brinda la capacidad de ver el mundo de una perspectiva única a la que un único conductor no podría tener acceso ya que este sistema recoge datos desde todos los ángulos y ve más allá de lo que vería un humano.

El sistema Autopilot de Tesla incorpora características de lo más comunes en 2018 como regular la velocidad según las condiciones el tráfico, mantenerse dentro de un carril… hasta otras características más innovadoras como **cambiar automáticamente de carril** sin dar previa instrucción por parte del conductor, **pasar de una autopista a otra**, **salir de la autopista**, **estacionarse** de forma autómata o ir a donde estés saliendo solo del garaje.

Además, se han desarrollado mejoras en el autogiro para realizar giros más pronunciados y conducir por sitios más estrechos.

Por supuesto, también tiene los ya mencionados sistemas anteriores como frenado de emergencia automático, advertencia de choque frontal o alerta lateral.

Por último, el vehículo es capaz de **buscar estacionamiento** totalmente solo una vez le haya dejado en su destino.

Tesla advierte que este sistema debe llevar un conductor que se mantenga atento ante cualquier imprevisto. Parece que el Autopilot ha supuesto un punto de inflexión claro en la seguridad de las carreteras.

2.3. Objetivos previos

Los **objetivos de este proyecto** son la **prevención de accidentes de vehículos en cadena** y, de producirse el accidente, el **aviso inmediato a la DGT y a los coches** posteriores al accidente para mantener a los conductores en alerta y así poder aumentar el tiempo de reacción.

Para ello se dispondrá de un **sistema que podrá ser acoplado a cualquier vehículo** sea o no nuevo.

Dicho sistema de prevención **controlará la velocidad del vehículo** y la distancia con el vehículo que le precede de forma similar a un sistema ADAS descrito anteriormente.

También tendrá control sobre el freno haciendo posible un **sistema de frenado de emergencia** o la reducción de la velocidad para **mantener la distancia de seguridad**.

Además, incorpora un **sistema de detección de lluvia** para moderar la velocidad en base a las condiciones meteorológicas.

El sistema tendrá **capacidad para activarse o desactivarse** cuando el conductor así lo desee.

No obstante, una vez producido el accidente, el vehículo estará dotado de un **sistema de detección de colisiones y GPS** con el que emitirá un aviso de su ubicación a la DGT y a los coches posteriores para alertarles del accidente y así minimizar el número de personas implicadas en él.

El sistema enviará el aviso a los demás vehículos sobre el accidente, aunque el sistema esté desactivado.

El sistema será capaz de detectar colisiones frontales, laterales y traseras, además de si se ladea bruscamente proporcionando inestabilidad en el vehículo.

El sistema podrá **recibir información sobre un posible accidente** notificándose así en un display y activando las señales acústica y luminosa.

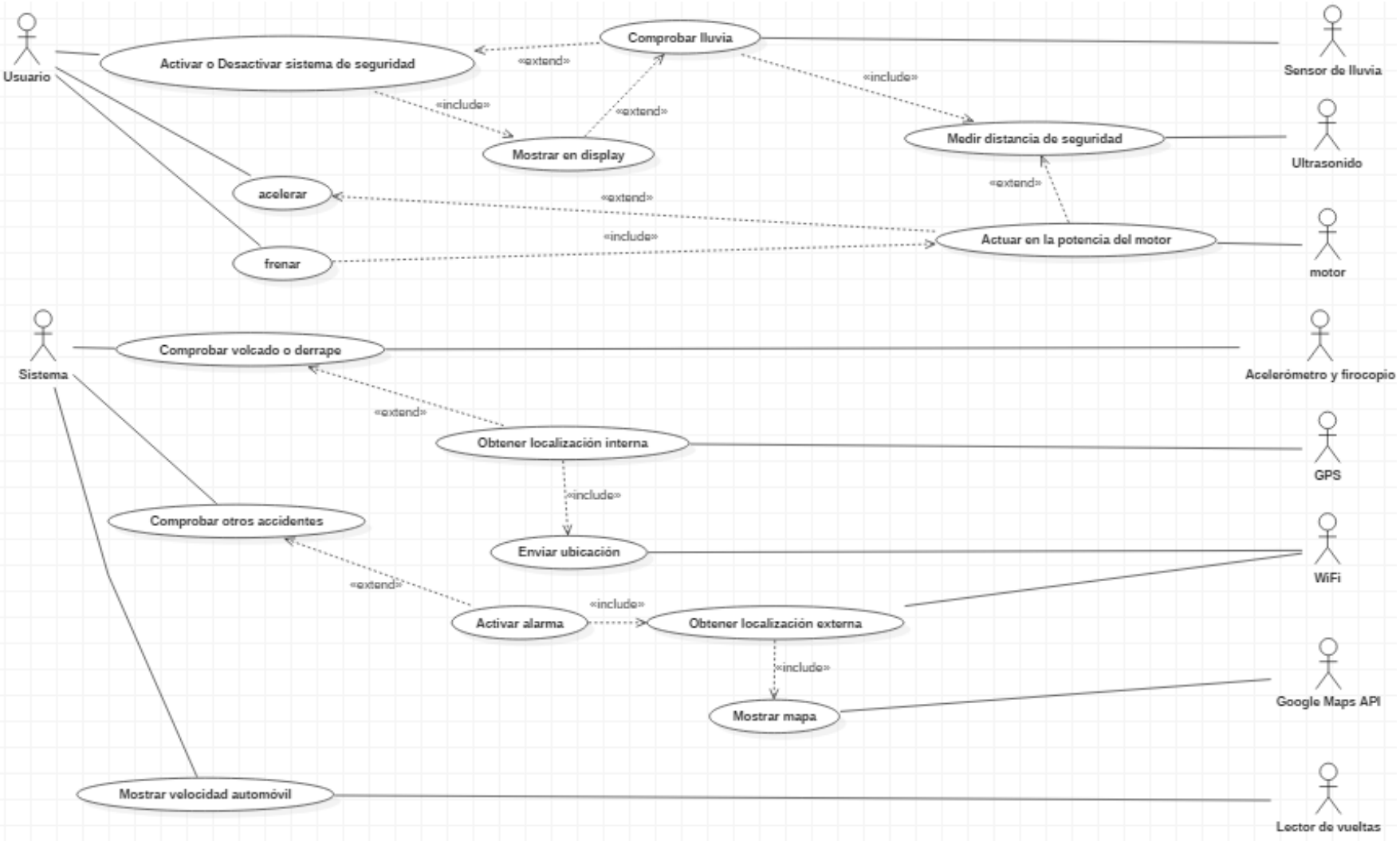
Gracias a tener los **vehículos conectados entre sí**, se pueden realizar **estadísticas** de la velocidad media de una carretera en tiempo real o de la distancia de seguridad media que llevan los coches, entre otros datos, con el objetivo de **prevenir un accidente** en cadena manteniendo a los usuarios informados con dichos datos.

2.4. Breve mención del método conseguido

El sistema obtenido alerta de un accidente a otros vehículos mostrándole su ubicación. Además, si esta activado es capaz de controlar la velocidad del vehículo para mantener una correcta distancia de seguridad y hacer posible un correcto frenado de emergencia.

3. Descripción detallada del sistema

3.1. Especificación



UC-0001	ACTIVAR O DESACTIVAR SISTEMA DE SEGURIDAD	
AUTORES	-	
FUENTES	-	
DESCRIPCIÓN	El usuario podrá activar o desactivar el sistema cuando lo desee	
PRECONDICIÓN	El sistema debe tener corriente	
SECUENCIA NORMAL	**Paso**	**Acción**
	1	El usuario pulsa el botón
	2	El sistema se activa
	3	El sistema muestra “ON” por el display
POS CONDICIÓN	Sistema encendido	
EXCEPCIÓN 1	**Paso**	**Acción**
	2	Si el sistema está activado entonces el sistema se desactiva
	3	El sistema muestra “OFF” por el display

UC-0002	COMPROBAR LLUVIA	
AUTORES	-	
FUENTES	-	
DESCRIPCIÓN	El sistema comprobará a través del sensor de lluvia si está lloviendo o no	
PRECONDICIÓN	El sistema de seguridad y el sensor de lluvia deben estar activados	
SECUENCIA NORMAL	**Paso**	**Acción**
	1	El sistema pide valores al sensor de lluvia
	2	El sensor de lluvia envía datos
	3	El sistema detecta que no llueve
	4	El sistema mide la distancia y ajusta la velocidad en consecuencia
POS CONDICIÓN	Conocimiento del estado meteorológico	
EXCEPCIÓN 1	**Paso**	**Acción**
	3	El sistema detecta que llueve
	4	El sistema mide la distancia y ajusta la velocidad en consecuencia
	5	El sistema muestra “It is raining” por el display
EXCEPCIÓN 2	**Paso**	**Acción**
	3	El sistema detecta que ya no llueve después de detectar lluvia anteriormente

	4	El sistema mide la distancia y ajusta la velocidad en consecuencia
	5	El sistema muestra “It is not raining” por el display

UC-0003	**MEDIR DISTANCIA DE SEGURIDAD**	
AUTORES	-	
FUENTES	-	
DESCRIPCIÓN	El sistema comprobará a través del sensor ultrasonido la distancia con el coche que le antecede	
PRECONDICIONES	El sistema de seguridad y el sensor de ultrasonido deben estar activados y tener conocimiento meteorológico	
SECUENCIA NORMAL	**Paso**	**Acción**
	1	El sistema pide valores al sensor de ultrasonido
	2	El sensor de ultrasonido envía datos
	3	El sistema detecta que la distancia de seguridad es menor que la necesaria para frenar el vehículo en caso de emergencia.
	4	El sistema ajusta la velocidad en consecuencia
POS CONDICIÓN	Ajuste de distancia de seguridad	
EXCEPCIÓN 1	**Paso**	**Acción**
	3	El sistema detecta que la distancia de seguridad es igual o mayor que la necesaria para frenar el vehículo en caso de emergencia.
	4	El sistema no hace nada

UC-0004	**ACTUAR EN LA POTENCIA DEL MOTOR**	
AUTORES	-	
FUENTES	-	
DESCRIPCIÓN	El sistema modificará la potencia del motor para dotar al vehículo de mayor o menor velocidad	
PRECONDICIONES	El sistema debe tener corriente	
SECUENCIA NORMAL	**Paso**	**Acción**
	1	El sistema verifica que el sistema de seguridad está activado

	2	No llueve y distancia de seguridad mayor o igual a la necesaria para frenar en caso de emergencia.
	3	El sistema modifica la potencia del motor en base al acelerador y la velocidad de seguridad máxima.
POS CONDICIÓN	Ajuste de velocidad	
EXCEPCIÓN 1	**Paso**	**Acción**
	2	Llueve y distancia de seguridad mayor o igual a la necesaria para frenar en caso de emergencia.
	3	El sistema modifica la potencia del motor en base al acelerador y la velocidad de seguridad máxima.
EXCEPCIÓN 2	**Paso**	**Acción**
	2	No Llueve y distancia de seguridad menor a la necesaria para frenar en caso de emergencia.
	3	El sistema modifica la potencia del motor en base a la meteorología y distancia de seguridad.
EXCEPCIÓN 3	**Paso**	**Acción**
	2	Llueve y distancia de seguridad menor a la necesaria para frenar en caso de emergencia.
	3	El sistema modifica la potencia del motor en base a la meteorología y distancia de seguridad.
EXCEPCIÓN 4	**Paso**	**Acción**
	1	El sistema verifica que el sistema de seguridad no está activado
	2	El sistema modifica la potencia del motor en base al acelerador.

UC-0005	**ACELERAR**	
AUTORES	-	
FUENTES	-	
DESCRIPCIÓN	El usuario aumentará la velocidad del vehículo	
PRECONDICIONES	El sistema debe tener corriente	
SECUENCIA NORMAL	**Paso**	**Acción**
	1	El usuario pisa el acelerador

	2	El sistema verifica que el sistema de seguridad está desactivado
	3	El sistema modifica la potencia del motor en base al acelerador.
POS CONDICIÓN	El coche acelera	
EXCEPCIÓN 1	**Paso**	**Acción**
	2	El sistema verifica que el sistema de seguridad está activado
	3	El sistema verifica que la distancia de seguridad necesaria para frenar el vehículo junto con la meteorología es mayor
	4	El sistema modifica la potencia del motor en base al acelerador y la velocidad de seguridad máxima.

UC-0006	**FRENAR**	
AUTORES	-	
FUENTES	-	
DESCRIPCIÓN	El usuario disminuirá la velocidad del vehículo	
PRECONDICIONES	El sistema debe tener corriente	
SECUENCIA NORMAL	**Paso**	**Acción**
	1	El usuario pisa el freno
	2	El sistema verifica que el sistema de seguridad está desactivado
	3	El sistema modifica la potencia del motor en base al freno.
POS CONDICIÓN	El coche frena	
EXCEPCIÓN 1	**Paso**	**Acción**
	2	El sistema verifica que el sistema de seguridad está activado
	3	El sistema modifica la potencia del motor en base al freno.

UC-0007	**COMPROBAR VOLCADO O DERRAPE**	
AUTORES	-	
FUENTES	-	
DESCRIPCIÓN	El sistema comprobará si el vehículo sufre un accidente	
PRECONDICIONES	Debe estar activado el acelerómetro, giroscopio, GPS y WiFi	
SECUENCIA NORMAL	**Paso**	**Acción**
	1	El sistema obtiene datos del acelerómetro y giroscopio

	2	El sistema verifica que el vehículo está sufriendo un accidente
	3	El sistema obtiene localización del GPS
	4	El sistema envía localización a los demás vehículos y a la DGT a través de WiFi
POS CONDICIÓN	El coche envía localización del accidente a los demás usuarios de la vía y a la DGT	
	Paso	**Acción**
EXCEPCIÓN 1	2	El sistema verifica que el vehículo no está sufriendo ningún accidente

UC-0008	**OBTENER LOCALIZACIÓN INTERNA**	
AUTORES	-	
FUENTES	-	
DESCRIPCIÓN	El sistema obtendrá localización del GPS	
PRECONDICIONES	El vehículo está sufriendo un accidente y GPS activado	
	Paso	**Acción**
SECUENCIA NORMAL	1	El sistema pide datos al GPS
	2	El sistema obtiene localización del GPS
POS CONDICIÓN	El sistema obtiene localización	

UC-0009	**ENVIAR UBICACIÓN**	
AUTORES	-	
FUENTES	-	
DESCRIPCIÓN	El sistema comprobará si el vehículo sufre un accidente	
PRECONDICIONES	Debe estar activado el acelerómetro, giroscopio, GPS y WiFi	
	Paso	**Acción**
SECUENCIA NORMAL	1	El sistema obtiene datos del acelerómetro y giroscopio
	2	El sistema verifica que el vehículo está sufriendo un accidente
	3	El sistema obtiene localización del GPS
	4	El sistema envía localización a los demás vehículos y a la DGT a través de WiFi

POS CONDICIÓN	El coche envía localización del accidente a los demás usuarios de la vía y a la DGT	
EXCEPCIÓN 1	**Paso**	**Acción**
	2	El sistema verifica que el vehículo no está sufriendo ningún accidente

UC-0010	**COMPROBAR OTROS ACCIDENTES**	
AUTORES	-	
FUENTES	-	
DESCRIPCIÓN	El sistema comprobará si existe una alerta por accidente de otros vehículos	
PRECONDICIONES	Debe estar activado WiFi	
SECUENCIA NORMAL	**Paso**	**Acción**
	1	El sistema verifica que existe alerta de otro vehículo accidentado
	2	El sistema activa señal acústica y luminosa
	3	El sistema obtiene localización del otro vehículo vía WiFi
	4	El sistema envía una petición a Google Maps
	5	El sistema muestra mapa con la ubicación del accidente
POS CONDICIÓN	El usuario se mantiene alertado de un accidente externo	
EXCEPCIÓN 1	**Paso**	**Acción**
	1	El sistema verifica que no existe ningún accidente externo

UC-0011	**ACTIVAR ALARMA**	
AUTORES	-	
FUENTES	-	
DESCRIPCIÓN	El sistema activará una señal acústica y luminosa para alertar al conductor de un accidente en la vía	
PRECONDICIONES	Tiene que existir accidente externo	
SECUENCIA NORMAL	**Paso**	**Acción**
	1	El sistema activa la señal acústica y luminosa
POS CONDICIÓN	El usuario está alertado de un accidente externo	
EXCEPCIÓN 1	**Paso**	**Acción**
	1	El sistema apaga la alerta cuando el usuario cierra el mapa

UC-0012	OBTENER LOCALIZACIÓN EXTERNA	
AUTORES	-	
FUENTES	-	
DESCRIPCIÓN	El sistema obtendrá la localización del vehículo accidentado	
PRECONDICIONES	Debe estar activado WiFi y tiene que existir accidente externo	
SECUENCIA NORMAL	**Paso**	**Acción**
	1	El sistema obtiene datos del otro vehículo vía WiFi
	2	El sistema muestra un mapa con la localización
	3	El sistema borra la ubicación del accidente una vez se haya procesado
POS CONDICIÓN	El sistema obtiene localización del accidente	

UC-0013	MOSTRAR MAPA	
AUTORES	-	
FUENTES	-	
DESCRIPCIÓN	El sistema mostrará un mapa con la ubicación del coche accidentado al usuario	
PRECONDICIONES	Tiene que existir accidente externo y haber obtenido localización del accidente	
SECUENCIA NORMAL	**Paso**	**Acción**
	1	El sistema envía la localización a Google Maps
	2	El sistema recibe una imagen de un mapa con la ubicación del accidente
	3	El sistema muestra la imagen al usuario
POS CONDICIÓN	El usuario visualiza dónde está el accidente	
EXCEPCIÓN 1	Paso	Acción
	1	El sistema apaga la alerta cuando el usuario cierra el mapa

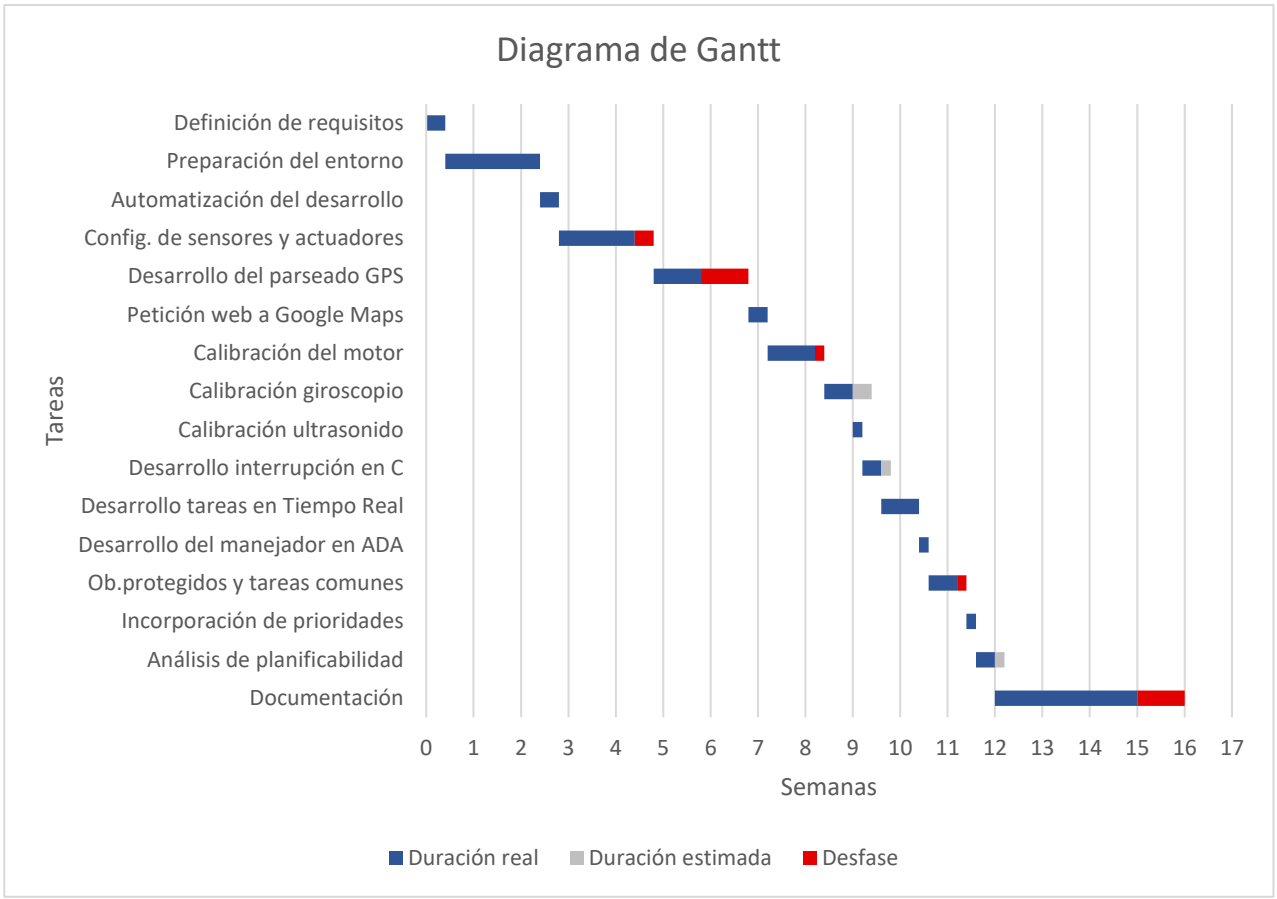
UC-0014	MOSTRAR VELOCIDAD AUTOMÓVIL	
AUTORES	-	
FUENTES	-	
DESCRIPCIÓN	El sistema mostrará la velocidad real que lleva el motor	
PRECONDICIONES	Tiene que estar activado el tacómetro	
SECUENCIA NORMAL	**Paso**	**Acción**

	1	El sistema recogerá cada dato del sensor
	2	El sistema realizará la conversión RPM a Km/h
	3	El sistema mostrará a los Km/h que va el motor
POS CONDICIÓN	El usuario comprueba la velocidad del vehículo	
EXCEPCIÓN 1	**Paso**	**Acción**
	1	Si el vehículo está detenido el sistema marcará 0 Km/h

3.2. Planificación del proyecto y diagrama de Gantt

Nº	Tarea	En qué se basa	Tiempo estimado (días)
1	**Definición de requisitos**	Definir los requisitos del sistema	2
2	**Preparación del entorno**	Creación de una imagen del sistema con todo lo necesario para poder compilar y ejecutar el sistema además de la comunicación y seguridad ente placas.	10
3	**Automatización del desarrollo**	Script para mantener un desarrollo mucho más ágil el cual descarga los archivos los compila y los ejecuta en la placa.	2
4	**Configuración de sensores y actuadores**	Toma de contacto con todos los sensores y actuadores para conseguir recibir o enviar datos de ellos o a ellos.	8
5	**Desarrollo del parseado GPS**	Conseguir la latitud y la longitud de todas las tramas recibidas por el GPS y decodificarlas en grados.	5
6	**Petición web a Google Maps**	Obtener una imagen del mapa con la ubicación que se quiere mostrar marcada en rojo.	2
7	**Calibración del motor**	Conseguir una función que simplifique la función real del motor para poder desarrollarla en el sistema.	5
8	**Calibración giroscopio y acelerómetro**	Calibrar el giroscopio y comprender un rango de valores en el cual el sistema acepte que no es un bache o una curva sino un accidente.	5
9	**Calibración ultrasonido**	Calibrar el ultrasonido para que detecte la distancia adecuada y configurarlo para que esa distancia sean metros y no centímetros.	1
10	**Desarrollo interrupción en C**	Comprender los semáforos y las interrupciones en C. Implementarlo para crear las 3 interrupciones necesarias.	3
11	**Desarrollo tareas en Tiempo Real**	Desarrollar todas las tareas unitarias que se requieren para el sistema.	4

12	**Desarrollo del manejador en ADA**	Desarrollar el manejador para despertar a las tareas de mucho cómputo.	1
13	**Desarrollo de objetos protegidos y tareas comunes**	Desarrollar los objetos protegidos y las tareas que requieren de varios valores en los objetos protegidos para realizar sus decisiones.	3
14	**Incorporación de prioridades**	Elegir que prioridad es la adecuada para cada tarea según el plazo en el que se necesita que sea ejecutado (DMS).	1
15	**Análisis de planificabilidad**	Comprobar que se cumplen todos los tiempos, que el sistema es planificable y que las tareas tienen una frecuencia de ejecución lógica.	3
16	**Documentación**	Realizar documentación por escrito y pruebas grabadas	15
		TOTAL	70 días



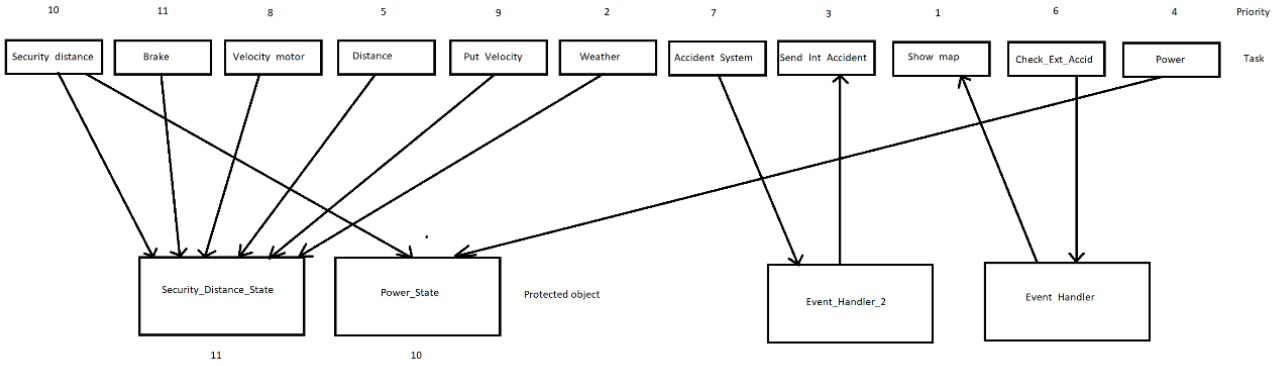
Se ha estimado un total de 70 días, pero algunas tareas se han visto retrasadas y esto ha ocasionado el retraso del proyecto completo debido a que solo hay una persona trabajando en dicho proyecto, es por lo que el total de días pasa de 70 a 80, haciendo un total de 16 semanas de trabajo.

3.3. Diseño del Sistema de Tiempo Real

En este sistema se ha seguido una planificación monótona en plazos (DMS) y por tanto las prioridades de las tareas se rigen por su plazo (Di), a mayor plazo menor prioridad.

Esto es así porque, se requiere de tareas que deben ser ejecutadas antes que su periodo (Pi), es decir: Di < Pi.

El mapa de procesos que se ha seguido para la realización del sistema es el siguiente:



Estimación WCET

Para obtener una estimación del peor tiempo de cómputo (WCET) se ha modificado ligeramente el código para poder poner tramas que recojan el tiempo de inicio y fin de la tarea.

Un ejemplo ilustrativo de ello es la tarea “Check_Ext_Accident” en la que se puede observar como se recoge el tiempo de inicio con t1 y el tiempo final con t2, posteriormente se realiza la diferencia de ambas y se muestra la duración de la tarea.

Cabe destacar que se ha asegurado que ninguna de las tareas ha sido interrumpida, ni expulsada de la CPU, ni bloqueada para estar seguros de medir únicamente el tiempo de CPU transcurrido.

```
task body Check_Ext_Accident is
    period      : constant Time_Span := milliseconds(50);
    current     : Time;
    init        : Time;
    accident    : Integer := 0;
    t1      : Time;
    t2      : Time;

    begin
        init    := Clock;
        current := init + period;
        t1 := Clock;
        loop

            accident := check_ext_location;
            if( accident = 1 ) then
                Event_Handler.Active;
            end if;
            t2 := Clock;
            put(Duration'Image(To_Duration(t2-t1))); put_Line("");
            delay until (current);
            t1 := Clock;
            current := current + period;
        end loop;
end Check_Ext_Accident;
```

Así mismo, se ha creado un fichero con al menos diez ejecuciones por cada tarea escogiendo como WCET de cada tarea el peor tiempo entre todas.

```
0.000016354
0.000015104
0.000013438
0.000013385
0.000013646
0.000046770          <<
0.000015000
0.000014062
0.000013958
0.000013698
0.000014479
```

Este es un ejemplo del fichero de salida “Ci_check_ext_accident” para la tarea “Check_Ext_Accident” siendo su WCET = 0.00004677 segundos

Tabla de tiempos

El protocolo seguido para este sistema es el **protocolo de techo de prioridad.** Ya que tenemos más de un recurso compartido podemos obtener ventajas frente a otros protocolos existentes.

A cada uno de los recursos compartidos del sistema se le asigna una prioridad equivalente a la mayor de todas las prioridades de las tareas que acceden a ducho recurso. A esta prioridad la llamaremos **Techo del Recurso,** es una prioridad estática, asignada en tiempo de compilación. [10]

Además de los Techos de los Recursos tendremos una variable, actualizada en tiempo de ejecución por el gestor de procesos, que contendrá el mayor de todos los Techos de los Recursos que en un momento dado están bloqueados por algún proceso. A esta variable se denomina **Techo Actual del Sistema.**

La tabla de tiempos en milisegundos es la siguiente:

					Objeto protegido	
Tarea	**Ci**	**Pi**	**Di**	**Pr**	**Security_Distance_State**	**Power_State**
Brake	0.2	300	2	11	x	-
Security_Distance	0.2	60	5	10	X	x
Put_Velocity	0.3	50	10	9	X	-
Velocity_motor	0.007	70	20	8	X	-
Accident_System	10	80	30	7	-	-
Check_Ext_Accident	0.04	100	100	6	-	-
Distance	192	240	230	5	X	-
Power	0.1	300	300	4	-	X
Send_Int_Accident	100	30000	1900	3	-	-
Weather	0.3	2000	2000	2	X	-
Show_Map	500	15000	10000	1	-	-

Tal como dijimos antes y basándonos en las tareas que acceden a los dos recursos de nuestro sistema el Techo de los recursos compartidos “Security_Distance_State” y “Power_State” es:

Objeto protegido	**Prioridad**
Security_Distance_State	11
Power_State	10

A pesar de que un botón tarda aproximadamente 200 milisegundos en ser pulsado continuamente, he puesto un periodo de 300 milisegundos a las tareas “Brake” y “Power” ya que con el pie la frecuencia al pulsar el botón disminuye y estos botones simularán el pedal de un coche y el encendido del sistema que con seguridad se producirá con una frecuencia menor.

La tarea “Distance” tiene un WCET de 192 milisegundos debido a que cuando detecta un objeto el tiempo en realizar un rebote es pequeño, en torno a 0.2 milisegundos, pero si no detecta ningún objeto el tiempo del rebote es mucho mayor produciéndose así un tiempo de cómputo de 192 milisegundos. En la realidad, tendremos muchas ocasiones en las que al vehículo no le anteceda ningún otro y por tanto este tiempo de espera si se va a producir con frecuencia.

La tarea “Send_Int_Accident” tiene un periodo de 30 segundos debido a que el sistema está preparado para que lance como máximo un envío de accidente cada 30 segundos y no envíe masivamente si por ejemplo diera vueltas de campana el vehículo.

Por último, la tarea “Show_Map” tiene un periodo de 15 segundos ya que tiene un tiempo de cómputo elevado, de 500 milisegundos, además estos 15 segundos los he puesto pensando en el conductor del vehículo y la posible distracción que conlleva mirar el mapa, ya que, si hubiera demasiados accidentes saltarían muchos mapas y la distracción del conductor sería mayor.

Una vez ajustados los tiempos de las tareas los resultados de las desviaciones locales de las tareas se pueden observar en el archivo “log_time_full_program”, aquí muestro un pequeño desglose del funcionamiento del sistema:

Tarea	Tiempo de ejecución (ms)	Desviación
START Put_Velocity	0.050147374	-
FINISH Put_Velocity	0.050389039	-
START Security_Distance	0.060110287	-
FINISH Security_Distance	0.060183984	-
START Accident_System	0.080117153	-
FINISH Accident_System	0.090287565	-
START Put_Velocity	0.100110947	-
FINISH Put_Velocity	0.100340737	-
START Check_Ext_Accident	0.100102614	No existe desvío porque Put_Velocity es muy rápida, pero al ser Check_Ext_Accident se ejecuta después
FINISH Check_Ext_Accident	0.100201988	-
START Security_Distance	0.120108595	-
FINISH Security_Distance	0.120180365	-
START Put_Velocity	0.150140405	-
FINISH Put_Velocity	0.150380664	-

Gracias a la baja latencia del procesador de la Raspberry Pi 3 y a los bajos tiempos de cambio de contexto, se puede considerar el tiempo de bloqueo (Bi) de los objetos protegidos 0 ya que el tiempo en los objetos protegidos es insignificante porque solo se realizan funciones get y procedimientos set, los cuales son muy rápidos .

Para que una tarea sea planificable con el método DMS debe verificar que el tiempo de respuesta (Ri) de la tarea sea estrictamente menor que el plazo de la misma. Ri < Di.

El tiempo de respuesta de una tarea se calcula de la siguiente forma (Klein p.4-35. Group 2, Tech 5):

$$R_i = C_i + B_i + \sum_{j \in hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil * C_j$$

Para contemplar las interrupciones y los procesos esporádicos se usa una ecuación similar basado en la utilización de CPU.

$$R_i = C_i + B_i + \sum_{j \in hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil * C_j + \left\lceil \frac{R_i}{T_{int}} \right\rceil * C_{int}$$

Este análisis basado en el tiempo de respuesta es válido para cualquier orden de asignación de prioridades fijas. El único requisito es que el orden de prioridad de los procesos esté especificado en el conjunto hp(i). [10]

Para el análisis de planificabilidad de las tareas he utilizado la herramienta RTA [11], ya que una buena automatización resulta más eficiente y segura a la hora de efectuar cálculos.

```
Response time analysis for task set Sample
-----------------------------------------

Id Task      A PR  Period  Offset  Jitter  Budget   Block Deadline Response Sch
-- --------- - -- ------- ------- ------- ------- ------- -------- -------- ---
 1 Brake     P 11 300.000   0.000   0.000   0.200   0.000    2.000    0.200 Yes
 2 Security_DP 10  60.000   0.000   0.000   0.200   0.000    5.000    0.400 Yes
 3 Put_VelociP  9  50.000   0.000   0.000   0.300   0.000   10.000    0.700 Yes
 4 Velocity_mP  8  70.000   0.000   0.000   0.007   0.000   20.000    0.707 Yes
 5 Accident_SP  7  80.000   0.000   0.000  10.000   0.000   30.000   10.707 Yes
 6 Check_Ext_P  6 100.000   0.000   0.000   0.040   0.000  100.000   10.747 Yes
 7 Distance  P  5 240.000   0.000   0.000 192.000   0.000  230.000  224.648 Yes
 8 Power     P  4 300.000   0.000   0.000   0.100   0.000  300.000  224.748 Yes
 9 Send_Int_AP  330000.000   0.000   0.000 100.000   0.000 1900.000 1672.448 Yes
10 Weather   P  22000.000   0.000   0.000   0.300   0.000 2000.000 1672.748 Yes
11 Show_Map  P  115000.000   0.000   0.000 500.000   0.00010000.000 9585.499 Yes

Priority ceilings for shared resources
--------------------------------------

Id Name      PR
-- --------- --
 1 Security_D11
 2 Power_Stat10

Total processor utilization :  97.27%
```

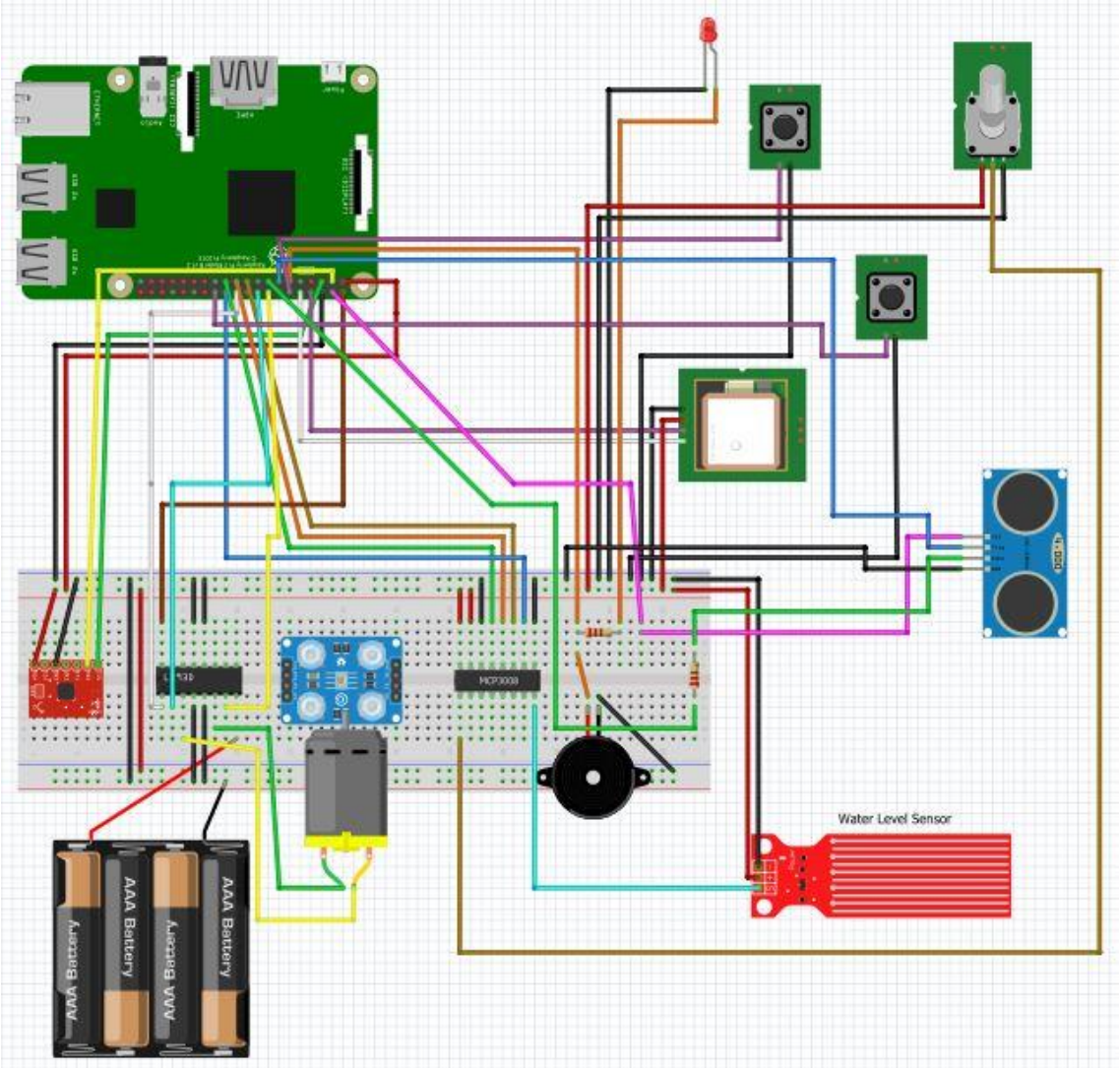
Como se puede ver en la salida del RTA **todas las tareas son planificables** (Sch = yes) y el porcentaje de utilización de **CPU es 97.27%** que está por debajo del 100% por lo que el **sistema es planificable.**

3.4. Diseño físico del sistema

Cada vehículo replicará este mismo diseño y mismo código software, esto es una de las ventajas ya que es fácil replicarlo.

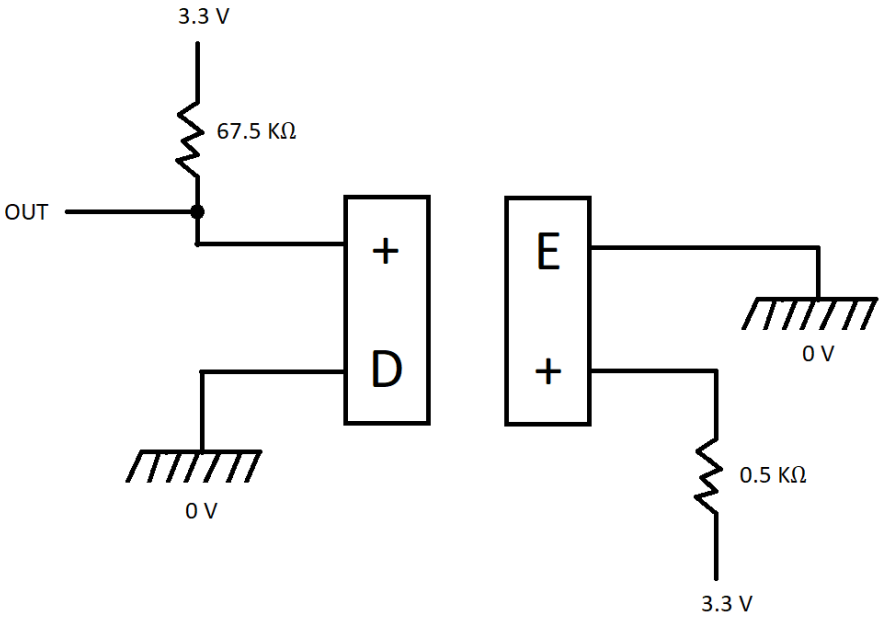
Cabe destacar que estos sensores y actuadores no serán los propuestos en una fase final, tan solo es un prototipo para comprobar el software, replicable y genérico a todos los vehículos.

El diseño global del sistema es el que se encuentra a continuación:



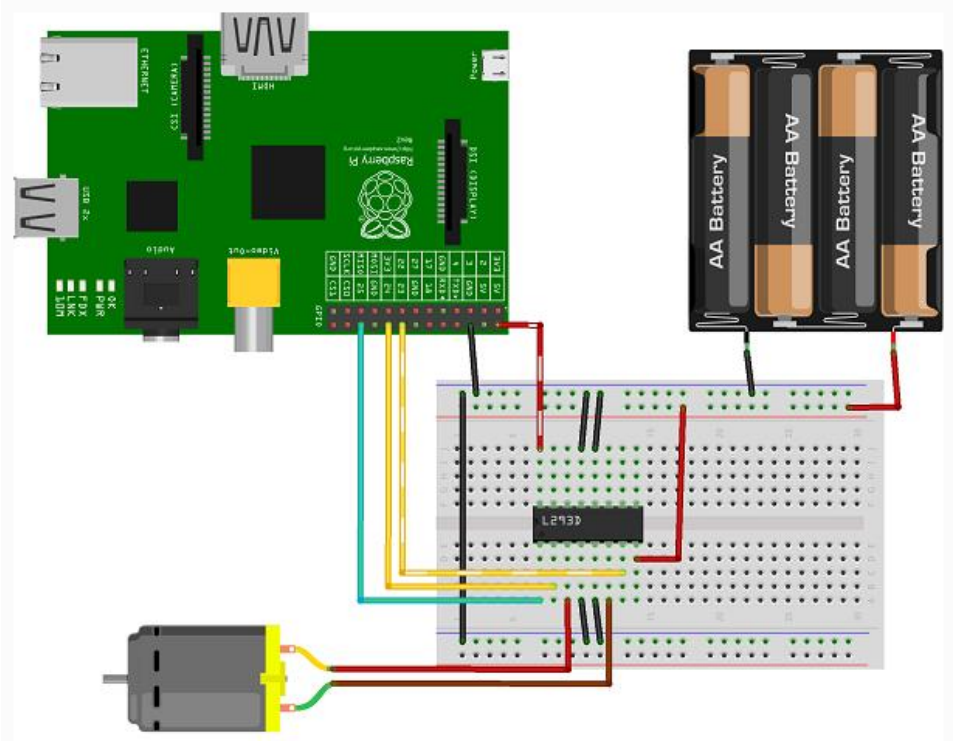
Diseño del tacómetro

Debido a que no existe un esquema específico para el tacómetro en el diseño global, el siguiente esquema detalla el diseño de este sensor:



Diseño motor

El diseño del motor es complejo ya que requiere de un microcontrolador L293D, baterías externas y voltaje a 5V sacado de la RPi3. [12]



El motor tiene varios problemas, uno de ellos es que la potencia del motor varía dependiendo del estado de la batería, siendo este más potente cuanto mayor amperaje tengan las pilas o dicho de otro modo cuanto más nuevas sean las pilas.

El otro problema es algo más complejo y viene relacionado con la capacidad que tiene el tacómetro de medir una vuelta en relación con la velocidad del motor.

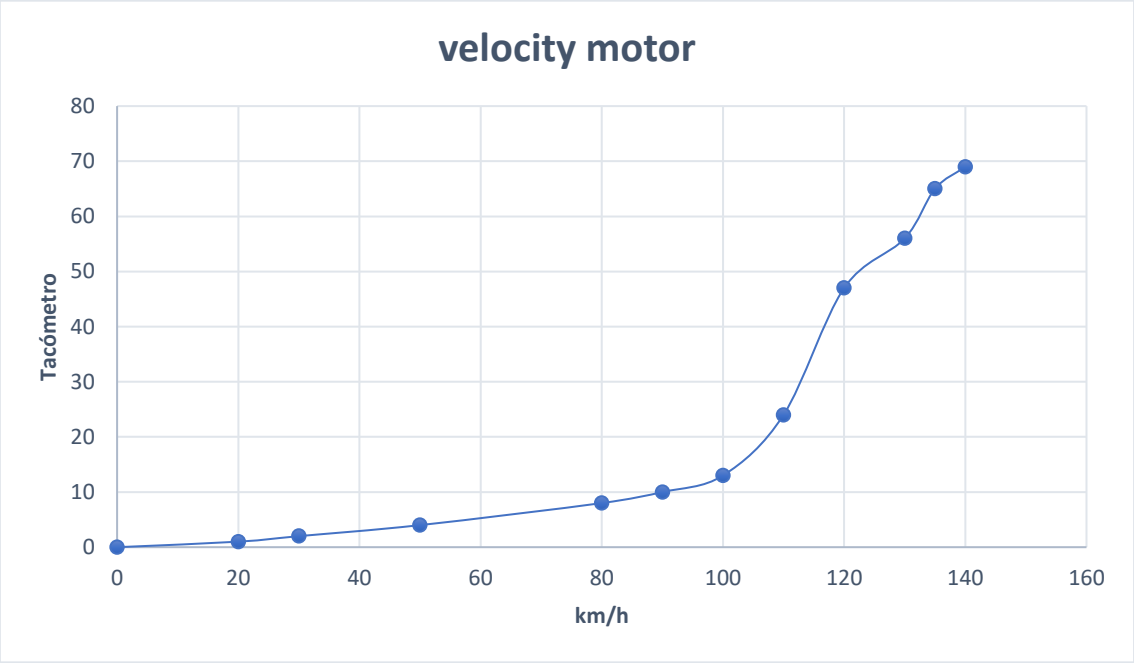
Por este motivo no se puede utilizar toda la potencia del motor, porque solo hasta una cierta velocidad es capaz de leer el tacómetro con mayor o menor claridad.

Utilizando esto como base y de forma empírica se obtienen resultados muy distintos ya que con velocidades por debajo de los 80km/h la velocidad es relativamente constante, pero a partir de este umbral la velocidad del motor varía bastante en muy poco rango de la velocidad del acelerador dejando incluso valores por los que ni siquiera va a pasar realmente.

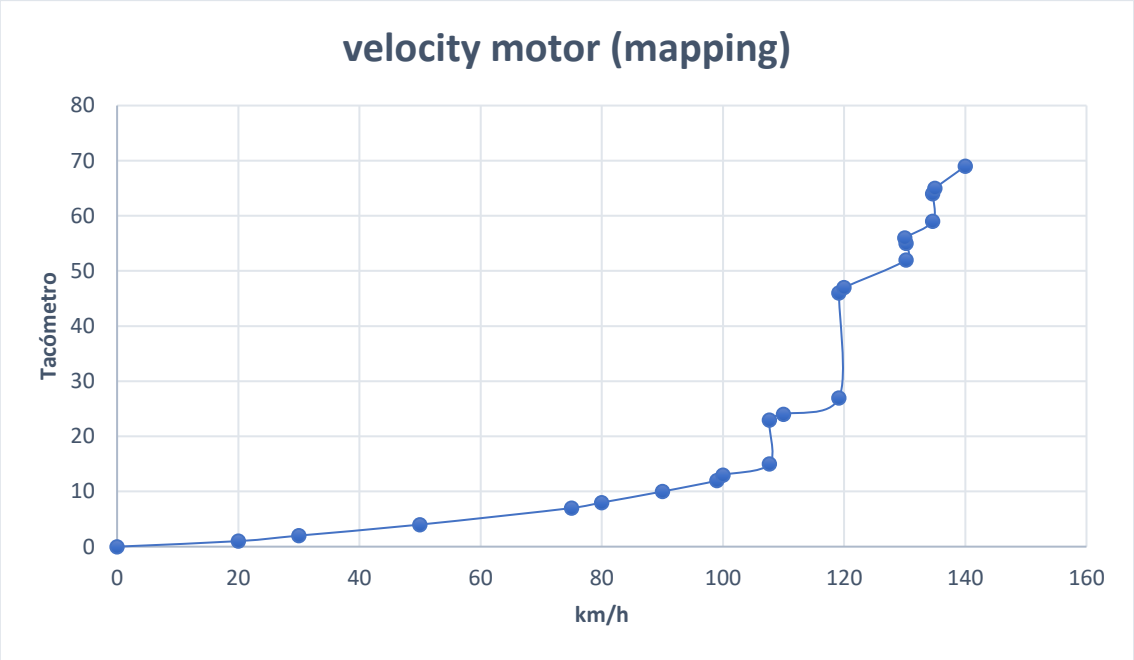
Para poder controlar el motor hay que mapearlo con el siguiente algoritmo: se comprueba una velocidad y la siguiente y se ve que obtiene el tacómetro en ambas, se divide para obtener el número que multiplicará linealmente hasta la siguiente otra velocidad que ya con este número se pase. Así obtenemos una linealidad en pequeños rangos. El rango excluido son valores que no corresponden a la función lineal.

Km/h	**Tacómetro**	**Nº multiplicador**	**Rango incluido**	**Rango excluido**
20	1	20	0-1	-
30	2	15	2-3	-
50	4	12.5	4-6	7
80	8	10	8-9	-
90	10	9	10-11	12
100	13	7.692	13-14	15-23
110	24	4.583	24-26	27-46
120	47	2.5531	47-51	52-55
130	56	2.3214	56-58	59-64
135	65	2.07	65-68	-
140	69	-	69+	-

La siguiente gráfica muestra la salida real del motor frente al tacómetro:



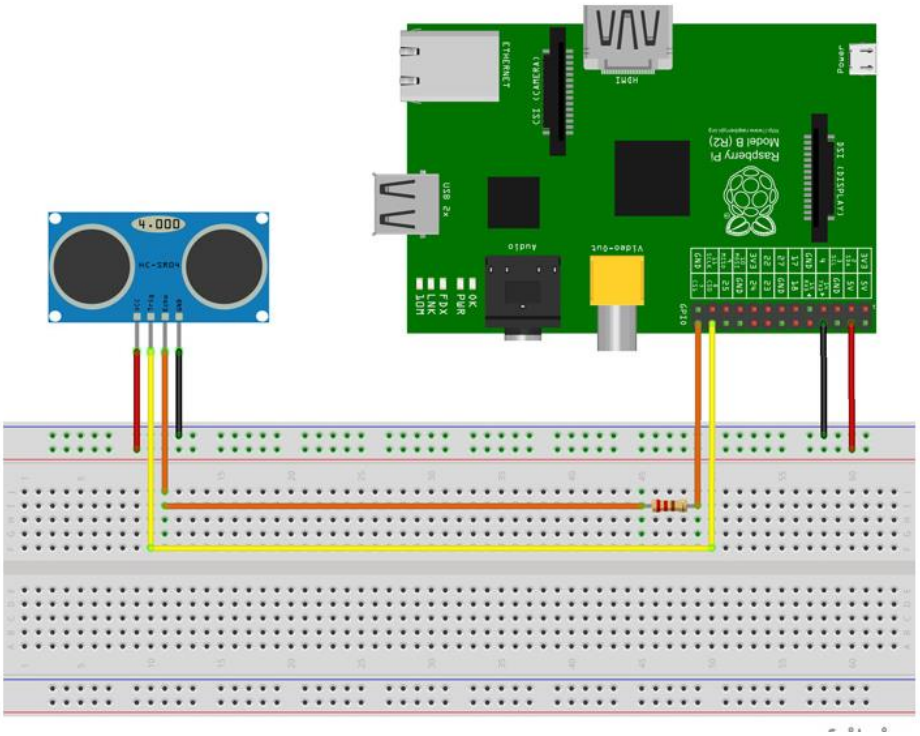
Esta gráfica muestra la salida después de ser mapeada con la tabla anterior:



Como se ve en la gráfica se consigue la función lineal en rangos cortos dado que muchos de los números del rango excluidos nunca se van a llegar a dar.

Diseño ultrasonido

El esquema de ultrasonido es delicado debido a que requiere de una resistencia de 1KΩ en el pin 'Echo' para que no se queme porque debe ir alimentado a 5V. [13]



Diseño GPS

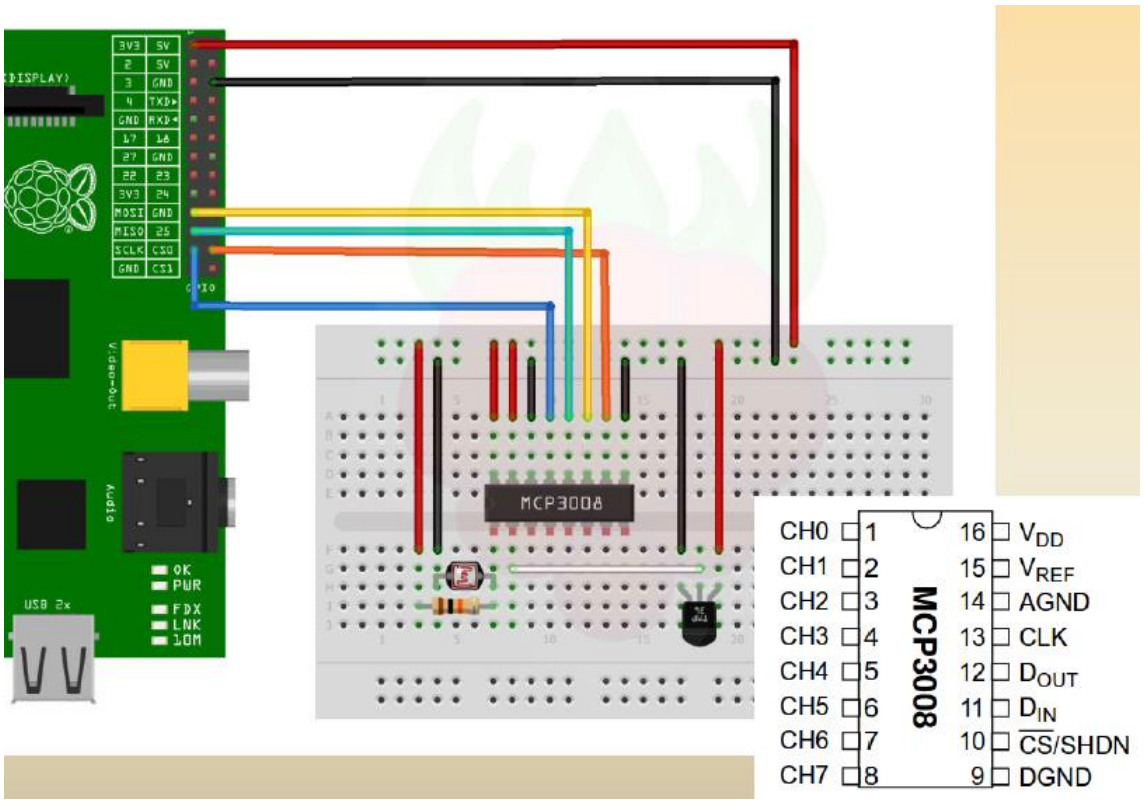
La conexión del GPS es mediante la **USART** de la Raspberry Pi. Pero hay que tener en cuenta que la conexión es opuesta, es decir [14]:

RPi3	**GPS**
5V	Vin/VCC
GND	GND
TX	RX
RX	TX

La RPi3 usa la USART AMA0 para el bluetooth para utilizarlo debemos cambiarla por la denominada mini USART y desconectar el bluetooth. Este proceso se explica con más detalle en el apartado "Cambiar UART y deshabilitar Bluetooth".

Diseño ADC

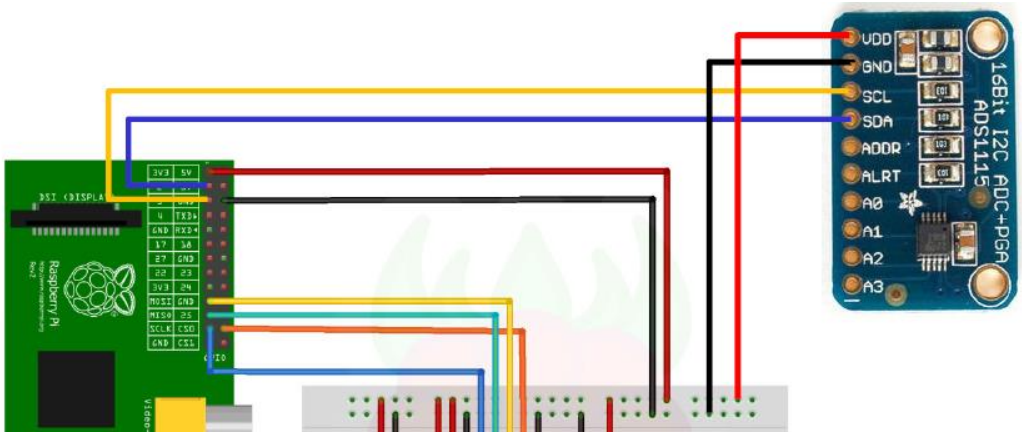
El ADC MCP3008 de Microchip se conecta a la RPi a través de **SPI** de la siguiente manera (Toma contacto RPi.pdf, 12):



Aquí irán conectados el sensor de lluvia y el potenciómetro que actuará de acelerador.

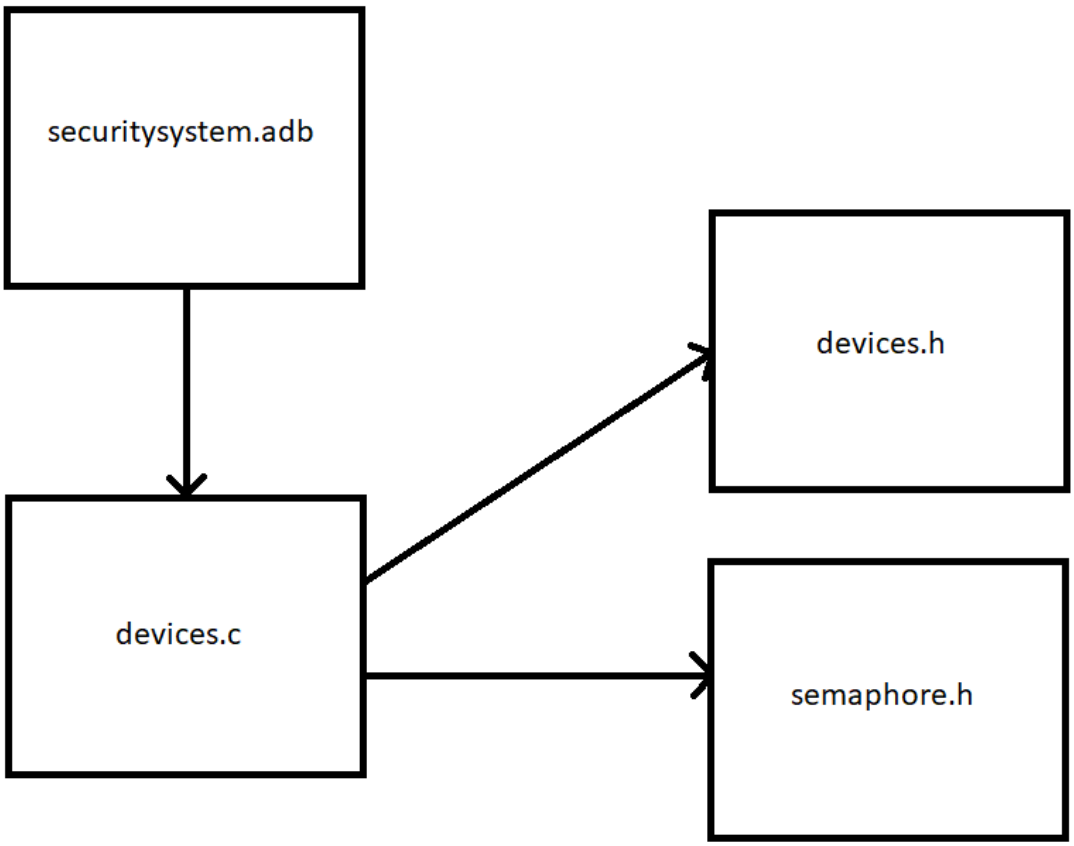
Diseño giróscopo y acelerómetro

Este sensor está conectado a la RPi a través de **I2C** para el que únicamente se necesitan dos vías, una de datos (SDA) y otra de sincronización del reloj (SCL).



3.5. Estructura del código

El código está compuesto por un Sistema de Tiempo Real desarrollado en ADA en el fichero “securitysystem.adb” y este se apoya en una librería de funciones para los sensores y actuadores desarrollada en C en el fichero “devices.c”. Además, el código en C se estructura en otros dos ficheros uno para las cabeceras, contantes y librerías a usar “devices.h” y otro con una librería para los semáforos de las interrupciones “semaphore.h”



En el fichero securitysystem.adb se definen las tareas y las prioridades además de pasar las funciones de C a su correspondiente en ADA y poder usarlas.

En el fichero devices.c se definen todas las funciones de la lectura de los sensores y activación de los actuadores, también se configura e inicializan los puertos y pines de la RPi, se crean los métodos de interrupción Wait y signal y se desarrollan los diversos parseadores.

En el fichero devices.h se encuentran las definiciones de las funciones la correspondencia de pines de cada sensor y actuador y las distintas librerias requeridas para el sistema.

En el fichero semaphore.h se desarrollan las funciones de inicialización del semáforo, wait y signal con llamadas al sistema para la correcta realización del semáforo.

3.6. Preparación y automatización del entorno de desarrollo

Conexión a internet con IP estática

Para configurar posteriormente el SSH debemos poner una IP estática, para ello:

- En RPi 1:

  Tenemos que modificar el fichero '**cmdline**' de la SD y poner una IP dentro del rango que nos otorga nuestro router pero sabiendo que esa IP no va a ser asignada a otro dispositivo de forma dinámica por el servicio DHCP, el modo más fácil es que si te asigna IP dinámicas cercanas a X.X.X.255 se ponga lo contrario X.X.X.4.

  Por ejemplo: Si te asignan una IP dinámica de 192.168.0.135, debes poner una IP estática de 192.168.0.7.

  Después de eso debemos modificar el fichero **/etc/network/interfaces** cambiando la IP, máscara, Gateway y Network a tu configuración personal.

  Una vez hecho eso resetea la Raspberry y escribe el comando: << **sudo dhclient eth0** >> para la configuración automática de DNS.

- En RPi 3 (WiFi) [15]:

  Antes de encender la RPi 3 por primera vez tenemos que crear el fichero '**wpa_supplicant.conf' en la micro SD** y dentro de él escribir:

```
# /etc/wpa_supplicant/wpa_supplicant.conf

ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1

network={
 ssid="nombre de tu router o SSID"
 psk="tu contraseña del wi-fi"
 key_mgmt=WPA-PSK
}
```

  Después iniciamos la RPi3 y verificamos que tenemos conexión WiFi.

  Para obtener la IP estática hacemos lo siguiente:

  Abrimos el fichero << **sudo nano /etc/dhcpcd.conf** >> y descomentamos y cambiamos el interfaz de eth0 a wlan0 poniendo la configuración de IP tal como estaba descrito en RPi 1:

```
interface wlan0
```

static ip_address=192.168.0.X/24

static routers=192.168.X.X

static domain_name_servers=8.8.8.8

static domain_search=8.8.4.4

Por último abrimos el fichero << **sudo nano /etc/network/interfaces** >>

Y escribimos lo siguiente:

interfaces(5) file used by ifup(8) and ifdown(8)

Please note that this file is written to be used with dhcpcd

For static IP, consult /etc/dhcpcd.conf and 'man dhcpcd.conf'

Include files from /etc/network/interfaces.d:

source-directory /etc/network/interfaces.d

iface wlan0 inet manual

Una vez hecho esto ya tendremos IP estática vía WiFi en nuestra RPi 3.

Cambiar contraseña del usuario pi

Lo primero que hay que hacer es cambiar la contraseña “raspberry” del usuario “pi” con el comando: << **passwd** >>.

Creación de un usuario

Necesitamos crear un nuevo usuario, distinto del predeterminado y darle permisos de administrador:

Vamos a crear el usuario “nuevoUsuario”: << **sudo adduser nuevoUsuario** >>

Te pide una contraseña y después datos personales que son opcionales.

Para dar permisos de administrador entramos en el fichero: << **sudo visudo** >>

Y escribimos debajo de la línea ‘pi’ lo siguiente: **nuevoUsuario ALL= (ALL) NOPASSWD: ALL**

Para que el usuario esté en el mismo grupo que el usuario pi debemos hacer lo siguiente:

Abrir el archivo << **sudo nano /etc/group** >> y escribir en cada apartado donde aparezca el usuario **pi** a continuación nuestro **nuevoUsuario** del siguiente modo:

Adm:x:4:pi, **nuevoUsuario**

SSH en la misma red

La configuración por realizar será una conexión SSH con una seguridad mediante clave privada y pública RSA 2048 y poder tener acceso, a nuestra Raspberry, tanto dentro de la misma red como fuera de la misma.

Instalación de clave pública y privada

La utilización de claves es un método más seguro que el uso de contraseña.

El método de utilización es sencillo, pongamos un ejemplo, tenemos un ordenador que se quiere conectar a una Raspberry, por lo que debemos generar una clave privada y también su clave pública asociada en el ordenador, la clave privada nunca debe ser compartida ni enviada, aquí reside la fortaleza de la utilización de las claves. La clave pública generada en el ordenador será ubicada en la Raspberry. De este modo, podremos conectarnos con nuestra clave privada desde nuestro ordenador a la Raspberry que tendrá nuestra clave pública.

Los pasos para conseguir con éxito esto en **Windows** son los siguientes:

Generamos la clave privada y pública en el ordenador con el que nos queremos conectar a la Raspberry, en Windows con **PuttyGen** y las guardamos.

A continuación, en la Raspberry creamos la carpeta **.ssh** y el fichero **authorized.keys** donde se guardarán todas las **claves públicas** de los distintos usuarios que quieran acceder a Raspberry en el usuario con el que accederemos posteriormente a la Raspberry desde otra máquina.

Los pasos para conseguir con éxito esto en **Linux** son los siguientes [16]:

Ejecutar el comando: << **ssh-keygen -t rsa** >>
Cuando nos pregunte esto:

Generating public/private rsa key pair.

Enter file in which to save the key (/home/test/.ssh/id_rsa):

Pulsar Enter
A continuación, preguntará:

Created directory '/home/test/.ssh'.
Enter passphrase (empty for no passphrase):

La intención en este proyecto es comunicar dos Raspberry por lo que no pondremos ninguna contraseña ya que más adelante se tendrán que conectar solas automáticamente por SCP.
Una vez finalizado este proceso habrá creado un par de claves en el directorio anteriormente indicado.

Como posteriormente replicaremos la imágen a otras RPi lo que haremos será utilizar la misma clave privada generada como clave para autenticar la RPi por lo que copiaremos la clave privada generada en el fichero '**authorized.keys' de la misma RPi que ha generado las claves.**
Si no existe este fichero lo creamos dentro de la misma carpeta .ssh

Tanto si hemos generado las claves en Windows como en Linux para securizar aún más nuestra red debemos hacer lo siguiente:

Modificamos el fichero de configuración **'/etc/ssh/sshd_config'** para proteger nuestra conexión:

Denegamos la entrada como root: **PermitRootLogin no**

Permitimos acceso al usuario con el que nos conectaremos: **AllowUsers nuevoUsuario**

Denegaremos el acceso a todos los usuarios que no sean con el que entraremos incluido el usuario pi: **DenyUsers pi**

Denegaremos el acceso por contraseña: **PasswordAuthetication no**

Permitiremos el acceso por clave pública, aceptaremos el algoritmo RSA e indicaremos la ruta donde se alojan las claves y que algoritmo es el utilizado:

RSAAuthentication yes

pubkeyAuthentication yes

AuthorizedKeysFile %h/.ssh/authorized_keys

Guardamos el archivo y reseteamos el servicio ssh: << **/etc/init.d/ssh restart** >>

Por último, nos conectamos a la Raspberry haciendo uso de nuestra clave privada, la IP estática que hayamos asignado y el usuario que hayamos dado permiso para entrar.

SSH distinta red

A la hora de conectarnos externamente a la Raspberry tenemos el problema de que la IP pública de la Raspberry es dinámica, para solucionar este problema nos vamos a apoyar en el servicio 'no-ip' que se configura en la Raspberry y está constantemente refrescando la ip pública de la Raspberry en un dominio asociado a nuestra cuenta no-ip y al que podemos conectarnos para saber la IP pública actual de la Raspberry.

Para configurar no-ip sigue estos pasos:

Entra en su web https://www.noip.com/ y regístrate es gratuito.

A continuación, crea un dominio poniendo el nombre que desees.

Una vez hecho esto pasamos a instalarlo en la Raspberry:

Descarga y descomprime no-ip:

<< **wget http://www.no-ip.com/client/linux/noip-duc-linux.tar.gz** >>

<< **tar -zxvf noip-duc-linux.tar.gz** >>

Accede a la carpeta, compila e instala el programa:

<< **sudo make** >>

<< **sudo make install** >>

Nos preguntará nuestro usuario y contraseña de no-ip, posteriormente al tiempo de espera nos hará una pregunta que se debe contestar NO.

Por último, configuramos el arranque automático del servicio no-ip y de dhclient:

<< **sudo nano /etc/init.d/noip2** >>

Copiar las siguientes líneas en el fichero, guardarlo y salir:

sudo dhclient eth0 (esto solo para RPi 1)

sudo /usr/local/bin/noip2

Añadir permisos, modificar fichero update-rc y ejecutarlo:

<< **sudo chmod +x /etc/init.d/noip2** >>

<< **sudo update-rc.d noip2 defaults** >>

<< **sudo /usr/local/bin/noip2** >>

Para poder acceder a la Raspberry tenemos que abrir el puerto en nuestro router, es conveniente cambiar el puerto público a uno que sepáis que el router no lo va a usar, generalmente números altos de no más de 4 cifras.

Una vez hecho esto se configura Putty poniendo en host name el nombre del dominio que hicimos en no-ip y el puerto público que pusimos en el router, además debéis indicar donde está vuestra clave privada para poder conectaros en el apartador SSh -> Auth de menú de Putty.

Cambiar UART y deshabilitar Bluetooth

En RPi3 nos encontramos el problema de que el puerto serie principal está usado por el Bluetooth del sistema por lo que tenemos que modificar esto para poder usar la UART como es debido.

Para configurar y cambiar el puerto serie en nuestra RPi 3 debemos hacer lo siguiente [17]:

Habilitar el puerto serie desde raspi-config indicando en 'interfaces' y después 'Serial' que el '**serial login Shell'** esté **deshabilitado** y el '**serial interface**' **habilitado**

La RPi3 dispone de los puertos serie:

/dev/ttyAMA0->Bluetooth.
/dev/ttyS0 -> GPIO serial port.

Pero el ttyS0 se trata en realidad de una mini UART que es inestable en baudios por lo que para una máxima fiabilidad escogeremos ttyAMA0 como UART principal.

Lo primero que debemos hacer es habilitar la UART:

<< **sudo nano /boot/config.txt** >>

Y abajo del todo escribimos: '**enable_uart=1**'

Una vez realizado esto verificamos que el serial 0 corresponde con ttyS0 y serial 1 corresponde con ttyAMA0 con el siguiente comando << **ls -l /dev** >>

Después tenemos que deshabilitar la consola realizamos lo siguiente:

<< **sudo systemctl stop serial-getty@ttyS0.service** >>

<< **sudo systemctl disable serial-getty@ttyS0.service** >>

Es necesario modificar el fichero de '**cmdline.txt**': << **sudo nano /boot/cmdline.txt** >>

Y eliminar la sentencia: '**console=serial0,115200**' si la hubiera.

A continuación, realizamos el swapping de puertos:

Abrimos el fichero: << **sudo nano /boot/config.txt** >> y escribimos al final: '**dtoverlay=pi3-miniuart-bt**'

Guardamos y **reiniciamos la RPi3**

Por últimos, comprobamos que efectivamente el cambio se realizó con éxito:

Serial0 = ttyAMA0

Serial1 = ttyS0

Instalar GNAT

Para saber que GNAT es el más actual procedemos a buscarlo dentro del repositorio con el comando:

<< **apt-cache search gnat | grep ^gnat** >>

En mi caso gnat 6 por lo que el comando para instalar sería: << **apt-get install gnat-6** >>

Existe la posibilidad de instalar GNAT GPS y otros componentes, pero para este proyecto solo necesito el compilador **GNU Ada Compiler**.

Instalar librería Nmea

La instalación de esta librería es necesaria para el parseado de las trazas que recibimos del GPS.

La instalación es la siguiente [18]:

<< **git clone https://github.com/jacketizer/libnmea.git** >>
<< **cd libnmea** >>
<< **make & make check** >>
<< **sudo make install** >>
<< **make unit-tests & make system-tests** >>

Hay que poner el flag -lnmea a la hora de compilar.

Instalacion I2C

Solo para RPi 1, en RPi 3 ya está instalado solo hay que dar permiso en configuración de Raspbian.

Nuestro sistema contiene sensores I2C por lo que tenemos que instalar y configurar la librería I2C de la Raspberry.

Instalar la librería con << **apt-get install i2c-tools** >>

Después ejecutar << **gpio load i2c** >> para poder acceder a la librería.

A continuación, tenemos que modificar el fichero /**etc/modules** y añadir **i2c-bcm2708** y **i2c-dev** para que se cargue el módulo I2C cuando arranque el sistema.

También comentar las líneas "**blacklist spi-bcm2708**" y "**blacklist i2c-bcm2708** "con # del fichero **/etc/modprobe.d/raspi-blacklist.conf**

Una vez cargado los módulos ejecutar **i2cdetect –y 0** ó **i2cdetect –y 1** para saber la dirección del dispositivo i2c conectado a la raspberry.

Bloqueo del usuario pi

Una vez hecho todo quitamos permisos de administrador al usuario Pi si no lo hemos hecho antes y lo bloqueamos de la siguiente forma:

<< **sudo visudo** >>

Comentar la sentencia '**pi ALL= (ALL) NOPASSWD: ALL**' con # y guardar el fichero.

<< **sudo usermod -L pi** >>

Script

El modo de trabajo está basado en la nube para poder trabajar en cualquier parte y tener siempre actualizado el proyecto además de tener la seguridad de que no se va a perder.

El proyecto se almacena en **OneDrive** en mi caso y desde cualquier dispositivo con conexión a internet y en un editor de texto se pueden modificar los ficheros.

Para automatizar todo el proceso de programación, compilación y ejecución del programa he creado un Script en Python 3.

Mediante un SSH a la Raspberry se lanza el Script "**DownloadFileRaspi.py**" el cual realiza las siguientes operaciones:

Primero elimina los ficheros binarios y ejecutables si los hay de anteriores compilaciones para realizar una ejecución limpia.

Después, descarga y sobrescribe en la Raspberry de los ficheros (pueden ser todos o seleccionados) en OneDrive.

A continuación, compila los ficheros en C con el compilador gcc y los ficheros de ADA con el compilador gnat. Cabe destacar que los flag -lm y -lnmea son para las librerías en C de libc6 y nmea respectivamente necesarias para nuestro proyecto.

Posteriormente se crean permisos de lectura, escritura y ejecución para el ejecutable de ADA.

Por último, se ejecuta el programa ADA.

Es **importante** entrar en el Script y modificar el PATH por vuestro path de proyecto y los archivos de descarga de OneDrive.

Para generar un **link de descarga directa en OneDrive** entrar desde el navegador a OneDrive, seleccionar un archivo y elegir la opción de insertar, creará un link html, hay que modificar la palabra **embed** por **download** y eliminar todo lo que no sea parte del URL.

Para ejecutar el script se debe poner << **sudo python3 DownloadFileRaspi.py** >> dentro de la carpeta del proyecto

Crear imagen SD

Una vez se ha configurado todo lo anterior correctamente es una opción valorable pero opcional el crear una imagen de la configuración que hemos realizado para no tener que repetirla en caso de que se produzcan fallos en la SD. Para ello haremos uso del programa: Win32 DiskImager para Windows.

Seleccionamos la unidad de la que queremos realizar la imagen y la ubicación donde queremos que se guarde acabado en .img

Por último, se selecciona "**Read**" y se espera hasta que finalice el proceso.

4. Resultados obtenidos

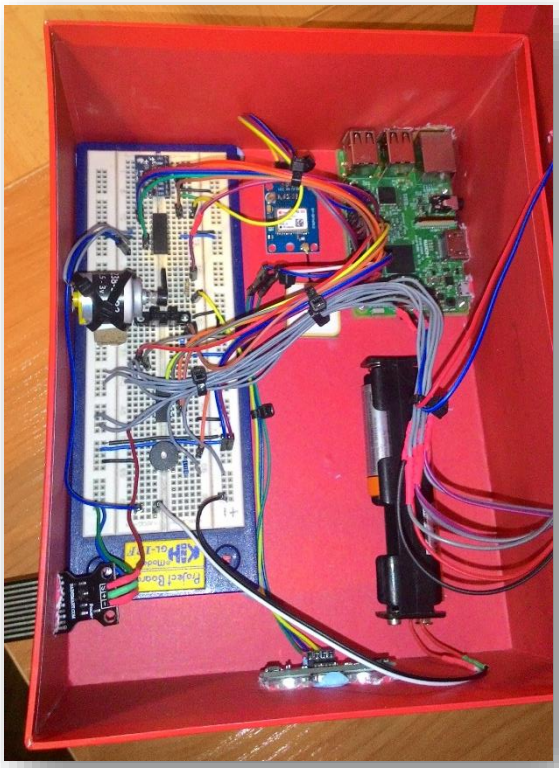
4.1. Producto que se ha conseguido

El producto conseguido es un prototipo de un vehículo, al cual se le a dotado de sensores de distancia, gps, giróscopo, acelerómetro, tacómetro, un motor simulando la velocidad del vehículo, un sensor de lluvia, un potenciómetro simulando el acelerador, botones para el freno y la activación del sistema y un sistema de alarma acústico y visual además de una pantalla donde se reflejan los parámetros en Tiempo Real del vehículo, así como un mapa con la ubicación de otro accidente.

El producto completo está comprendido por dos sistemas idénticos en hardware y software, para poder simular los distintos escenarios entre vehículos que veremos más adelante y una única pantalla conectada al sistema que recibirá el aviso de un accidente externo para poder visualizar el mapa en la pantalla.

El sistema funciona sobre Raspbian, una distribución Linux para Raspberry, esto nos permite flexibilidad a la hora de realizar otros usos siempre antes de poner en marcha el vehículo, como por ejemplo buscar rutas en Google Maps o incluso mandar un email si fuera necesario.

Tal como se ve en la imagen que se muestra a la debajo de este texto se puede observar que todos los componentes están usando una protoboard común para su interconexión con la RPi 3.



Los cables están diferenciados por bridas según si son externos o internos para hacer mucho mas fácil la apertura de la caja si fuera necesario y reducir el riesgo de desconexión de cables.

Tanto el motor como los actuadores de la tapa de la caja están soldados para mantener máxima fijación.

Incorpora una batería externa de 6V para el motor.

El producto final individual quedaría en forma de caja con toda su funcionalidad en el interior y solo lo imprescindible en el exterior reduciendo así el mal uso del sistema.

Finalmente, y por concluir la presentación del producto, hay que hablar del panel que muestra las métricas del sistema, en el cual en una sola línea se visualiza todo lo necesario para que de un vistazo el usuario sepa en todo momento que está pasando en el sistema.

En primer lugar, se encuentra el aviso de activación del sistema. Después viene la velocidad que queremos poner con el acelerador. A continuación, la velocidad real del motor, precediendo a esto viene el aviso del freno y por último la distancia con el vehículo de delante si lo hubiera, además también indicará si hay lluvia o si sufrimos un accidente consecutivamente en la misma línea.

```
bk0001@raspberrypi:~/Documents/TFG $ ./securitysystem
Starting the main program
gpio: Unable to load/unload modules as this Pi has the device tree enabled.
  You need to run the raspi-config program (as root) and select the
  modules (SPI or I2C) that you wish to load/unload there and reboot.
 I2C configured
 PIN configured
 PWM configured
 Interrupt ready
       Accelerator:  0 km/hbody of procedure Launcher_task
ON     Accelerator:  0 km/h   Motor Vel: 0 km/h      START   Distance:  323 m
```

4.2. Resultado de las pruebas

El resultado de las pruebas se encuentra en un fichero “log_time_full_program” para comprobar los tiempos de ejecución, desvíos locales y expulsiones. Además, he creado un vídeo demostrativo con cada una de las pruebas para que se vea mejor aún su ejecución, el vídeo se puede descargar o ver en el siguiente enlace:

(Haz clic aquí) vídeo demostrativo de pruebas

La explicación del resultado de cada prueba es la que viene a continuación:

1. **Activar y desactivar el sistema:**
   Esta prueba resultó satisfactoria ya que cumple su objetivo que es el de que al pulsar y despulsar el botón por primera vez el sistema se activa y muestra “ON” en la pantalla y pulsado por segunda vez el sistema se desactiva y muestra “OFF” en la pantalla. La velocidad de ejecución del botón es la adecuada y no se colapsa.
   La tarea no tendrá desviación local siempre que no sea bloqueada por otra tarea.

2. **Comprobar lluvia:**
   Esta prueba ha resultado satisfactoria, advierte de que está lloviendo mediante el display “It is raining” además de mandar la señal al sistema para que actúe en consecuencia. Del mismo modo, una vez que deja de llover advierte de nuevo al sistema e indica en el display “It is not raining”.
   La tarea no tendrá desviación local siempre que no sea bloqueada por otra tarea.

3. **Medir distancia de seguridad y ajustar velocidad:**
   Esta prueba ha resultado satisfactoria, mide la distancia de seguridad bien en todo momento, sin variaciones. Además, solo si el sistema está activado ajusta la velocidad del motor dependiendo de la distancia de seguridad del momento.
   Si la velocidad pasa los 120km/h y activamos el sistema de seguridad el vehículo pasará a tener una velocidad de 120km/h siempre que la distancia de seguridad se lo permita.
   La tarea no tendrá desviación local siempre que no sea bloqueada por otra tarea.

4. **Medir distancia de seguridad con velocidad menor que la máxima:**
   Esta prueba ha resultado satisfactoria, mide la distancia de seguridad bien en todo momento, sin variaciones. Además, solo si el sistema está activado ajusta la velocidad del motor dependiendo de la distancia de seguridad del momento y sin pasarse de la velocidad puesta en el acelerador.
   La tarea no tendrá desviación local siempre que no sea bloqueada por otra tarea.

5. **Acelerar:**
   Esta prueba ha resultado satisfactoria, aumenta la velocidad a medida que incrementa el acelerador (potenciómetro) su valor de 0km/h a 140km/h y disminuye su valor de igual forma. El sistema deja de incrementar la velocidad

del motor a pesar de lo propuesto por el acelerador cuando esta activado y la distancia de seguridad no permite esa velocidad. Además, marca en el display la velocidad a la que se quiere poner el vehículo.
La tarea no tendrá desviación local siempre que no sea bloqueada por otra tarea.

6. **Frenar:**
   Esta prueba a resultado satisfactoria, tanto con el sistema activado como desactivado el vehículo frena progresivamente. Cabe destacar que al ser solo un botón la progresividad la implementé en software a modo de estudio ya que si hubiera sido un pedal de freno se puede medir mejor la intensidad con la que se quiere frenar, es por esto que cuando freno y suelto el botón el motor se pone rápidamente a la velocidad que tenía antes, en un motor normal tardaría más en ejecutar esta acción e iría más progresivo. El sistema responde bien tanto al apretar el botón como al soltarlo si tenemos en cuenta que se usaría con el pie y no con la mano y la frecuencia es mucho más lenta. Además, muestra por pantalla cuando se pisa el freno y cuando se suelta.
   La tarea no tendrá desviación local siempre que no sea bloqueada por otra tarea.

7. **Detección de accidente:**
   - **Vuelco horizontal:** La prueba ha resultado satisfactoria, el sistema detecta que el vehículo ha sufrido un accidente cuando supera los 50% horizontales y envía la localización al resto de vehículos con efectividad y en el plazo indicado, no vuelve a enviar otro aviso como mínimo hasta que no pasen 30 segundos debido a que el accidente puede ocasionar varios giros en el vehículo, pero solo queremos un único aviso. Además, muestra en el display "vuelco horizontal".
   - **Vuelco vertical:** La prueba ha resultado satisfactoria, el sistema detecta que el vehículo ha sufrido un accidente cuando supera los 50% verticales y envía la localización al resto de vehículos con efectividad y en el plazo indicado, no vuelve a enviar otro aviso como mínimo hasta que no pasen 30 segundos debido a que el accidente puede ocasionar varios giros en el vehículo, pero solo queremos un único aviso. Además, muestra en el display "vuelco vertical".
   - **Derrape:** La prueba ha resultado satisfactoria, el sistema detecta que el vehículo ha sufrido un accidente cuando el sistema detecta una velocidad de giro anómala y envía la localización al resto de vehículos con efectividad y en el plazo indicado, no vuelve a enviar otro aviso como mínimo hasta que no pasen 30 segundos debido a que el accidente puede ocasionar varios giros en el vehículo, pero solo queremos un único aviso. Además, muestra en el display "Derrape".

   La tarea no tendrá desviación local siempre que no sea bloqueada por otra tarea.

8. **Velocidad del motor**

   La prueba ha resultado satisfactoria, la velocidad del motor cumple con lo establecido en el acelerador o la distancia de seguridad con un error del orden de ± 5 km/h. Además, se muestra la velocidad real del motor en el display. Cabe destacar que esta parte del sistema no será implementada en una fase final puesto que el motor y acelerador serán los propios del vehículo y se actuará sobre ellos.

   La tarea no tendrá desviación local siempre que no sea bloqueada por otra tarea.

9. **Recibir alerta de un accidente**

   La prueba ha resultado satisfactoria, el sistema recibe la alarma de accidente debido a que comprueba constantemente si a recibido alguna información externa, una vez recibida manda la orden de que debe ser mostrado el mapa con la localización recibida por el display y emite una señal luminosa y acústica en el coche receptor durante 5 segundos. No obstante, a pesar de mandar la petición de visualización del mapa el sistema se sigue ejecutando constantemente haciendo sus otras tareas ya que la apertura del mapa es costosa en tiempo de cómputo y no se podría dejar al sistema bloqueado mientras se abre el mapa en la pantalla. El mapa está disponible en el plazo estipulado al igual que todas las demás tareas.

   Para visualizar el mapa el sistema manda una petición a la API de Google Maps para recibir una imagen con la ubicación recibida y un punto rojo indicando dicha ubicación.

También se realizaron pruebas en el proceso de construcción del hardware como es el envío de valores de los sensores, la fiabilidad que estos tienen como puede ser el motor anteriormente descrito en las gráficas o incluso cuanto tiempo puede permanecer el sistema funcionando y con la caja cerrada, ya que se sobrecalienta y no tiene apenas ventilación.

Lo cierto es que relacionado a la temperatura del sistema es necesario unos disipadores de calor para la RPi3. No obstante, el hardware final no se va a implementar así ya que estos sensores no son un buen ejemplo de fiabilidad frente a sistemas críticos de tiempo real.

5. Presupuesto

5.1. Material

La siguiente tabla detalla el presupuesto del **prototipo** creado para este proyecto:

Componente	Precio	Nº elemento	Tienda
Led	0.4	1	led
Resistencia	0.14	3	resistencias
Motor	1.9	1	motor
Giroscopo y acelerómetro	4.5	1	giróscopo y acelerómetro
Raspberry Pi 3	55.45	1	Raspberry Pi 3
Sensor de agua	5.09	1	sensor de agua
GPS	41.27	1	GPS
Buzzer	4.55	1	Buzzer
L293D	3.94	1	L293D
Potenciómetro	1.8	1	Potenciómetro
Protoboard	9.85	1	Protoboard
ADC MCP3008	4.98	1	ADC
Cables	2.45	2	cables
Tacómetro	1.24	1	Tacómetro
Ultrasonido	2.92	1	Ultrasonido
Botones	1.83	1	Botón
TOTAL	**145.04 €**		

Es un total de 145.04 € por cada sistema, en mi caso tengo dos sistemas por lo que el precio sería de **290.08 €**

5.2. Horas de trabajo

Puesto que el precio de las horas de un ingeniero depende en gran medida del conocimiento que este tenga, he decidido no contabilizar el precio de un ingeniero en dinero sino en horas de trabajo realizadas.

La horas totales realizas para este proyecto, un único ingeniero desde cero, serían 16 semanas a 30 horas por semana, lo que hace un total de **480 horas de trabajo** efectivo sin contar reuniones con el cliente ni compra de material.

6. Impacto social o medioambiental

Los vehículos que dispongan de este sistema obtendrán un mayor nivel de seguridad en las carreteras y una reducción del número de accidentes.

Si todos los vehículos llevasen este sistema activado se hablaría de una red de información entre automóviles lo que facilitaría la labor de conducción autónoma.

Hoy en día, aun no es legal conducir vehículos sin supervisión constante del piloto, pero gracias a este sistema podemos asegurar que si está activado reducirá el tiempo de reacción en caso de frenado de emergencia haciendo posible que no se sufra ningún daño.

Otra opción positiva del sistema es que sabremos en todo momento los accidentes que hay en la carretera y podremos tomar otras rutas alternativas evitando así el colapso de la carretera en la que se encuentre el accidente y haciendo más efectiva la llegada de personal sanitario al punto del accidente. Esto es bastante importante puesto que los primeros minutos de sufrir el accidente son cruciales si el daño es grave.

Las personas están ociosas por saber cuál será la próxima nueva tecnología y siempre intentan tener lo último en el mercado, es por lo que, con los nuevos coches autónomos la gente se piensa dos veces si comprarse un coche de este estilo o uno normal, este sistema, al ser de un ambiente genérico, lo que consigue es el hecho de poder dotar a cualquier automóvil de esta posibilidad, haciendo posible, eficiente y segura la conducción en carretera de todos los vehículos. Además, como se ha mencionado con anterioridad de esta manera se podría conseguir la red de información entre vehículos en menos tiempo que si tenemos que esperar a que todos los coches sean sustituidos por otros nuevos con estos sistemas de serie.

Otro de los puntos fuertes del sistema es la capacidad de detectar y avisar a los demás vehículos y, porque no, a la DGT y servicios sanitarios de un accidente justo en el momento en el que se produce, es decir, no solo previene accidentes si no también alerta a los demás cuando este se produce haciendo que el flujo vial sea mucho más seguro.

Pero no todo es un impacto social también gracias a este sistema se puede lograr una conducción más eficiente y con ello se reducirán las emisiones tóxicas, lo que hará que las enfermedades derivadas de la contaminación se reduzcan.

Gracias a todos los datos enviados a la DGT a través de este sistema se pueden realizar estadísticas para saber qué zonas hay más afluencia de coches y más contaminación para así poder tomar medidas y reducir estos niveles que son un gran problema en este siglo XXI.

7. Valoración

7.1. Interpretación y justificación de los resultados obtenidos

Todos los resultados obtenidos en las pruebas han resultado satisfactorios, pero hay ciertas consideraciones a tener en cuenta en los resultados:

Como los sensores son para un **prototipo** se han tenido que configurar para que consideraran que tenían rangos reales de, por ejemplo, 140km/h en el motor o 100 metros de distancia dado que, por ejemplo, el sensor de distancia propuesto para este proyecto no alcanza más de 2 metros.

Gracias a la **baja latencia** del procesador de Raspberry pi 3 se consigue un porcentaje de **consumo de CPU menor al 100%** ya que es capaz de realizar la tarea más rápida del diseño en tan solo 0,1 milisegundos y la más lenta en 500 milisegundos, algo que ha supuesto una gran ventaja a la hora de la planificabilidad del sistema.

Cabe destacar que la tarea 'Show_Map" requiere más tiempo de cómputo y tuve que ponerle un plazo mayor al que realmente quería en un principio, ya que de no ser así no podía asegurar que la tarea fuera a ejecutarse en los tiempos establecidos. La conclusión de esto no ha sido otra que la de aceptar estos tiempos para que la tarea sea planificable debido a que **no es una tarea crítica**, es una tarea de información y realmente no importa el tiempo de ejecución, más allá de la espera del usuario por ver el mapa, aun así, he realizado pruebas y siempre se ejecuta en un plazo menor a 2 segundos con el sistema funcionando al completo a pesar de que su plazo sea de 10 segundos por lo tanto es un tiempo aceptable.

Los **resultados obtenidos** están basados en los valores de sensores para el prototipo y en ningún caso se podrá incorporar este software, ni mucho menos el hardware, a un vehículo directamente, habrá que modificar diversos factores como motor, acelerador, entre otros comentados con anterioridad para que el conocimiento del medio que rodea al vehículo sea más fiable y por ende también habrá que modificar el software que ello conlleva.

No obstante, el **software** generado para este sistema es un **buen punto de partida** para dicho sistema real y con total seguridad reaccionará en los tiempos estipulados ya que es un sistema planificable por lo que se asegura que el sistema actuará tal y como se requiere.

El **diseño del sistema** será utilizado casi en su totalidad por el sistema final ya que apoyándonos en los resultados obtenidos se ha conseguido una estabilidad y fiabilidad del sistema.

7.2. Hasta donde se ha conseguido llegar de los requisitos originales

Respecto de los requisitos originales se ha conseguido abarcar un gran número de estos, pero bien es cierto que aún quedan otros que no:

Se ha conseguido abordar la **prevención de accidentes de vehículos en cadena** y, de producirse el accidente, el **aviso inmediato a los coches** posteriores al accidente para mantener a los conductores en alerta y así poder aumentar el tiempo de reacción. Pero no se ha llegado a implementar el aviso a la DGT, aunque no supondría un gran problema implementarlo más allá de crear una buena red de comunicaciones y emisión de información en broadcast.

Se ha conseguido crear un **sistema genérico** que podrá ser acoplado a cualquier vehículo sea o no nuevo.

Más en detalle se ha logrado que el **sistema de prevención controle la velocidad del vehículo y la distancia** con el vehículo que le precede de forma similar a un sistema ADAS y el **control sobre el freno** haciendo posible un sistema de frenado de emergencia o la reducción de la velocidad para mantener la distancia de seguridad.

Además, se ha incorporado un **acelerador** para poder controlar el sistema de forma manual para no hacerlo simulado como en un principio se pensó ya que así hay más interactuación con el usuario.

También se ha incorporado el **sistema de detección de lluvia** para moderar la velocidad en base a las condiciones meteorológicas.

Al sistema se le ha dotado de un botón con capacidad para **activar o desactivar el sistema** cuando el conductor así lo desee. Sin embrago, enviará un aviso a los demás vehículos si ha sufrido un accidente, aunque el sistema esté desactivado.

Otro de los logros del sistema es que tiene un **sistema de detección de colisiones y GPS** con el que emitirá un aviso de su ubicación a los coches posteriores para alertarles del accidente y así minimizar el número de personas implicadas en él.

El sistema es capaz de detectar vuelcos horizontales y verticales, además de si se ladea bruscamente proporcionando inestabilidad en el vehículo.

El sistema tiene incorporado un sistema para **recibir información sobre un posible accidente** notificándose así en un display y activando las señales acústica y luminosa.

A pesar de tener **vehículos conectados** entre sí, no se han conseguido realizar envíos de información a la DGT para que esta realice estadísticas de velocidad media de una carretera en tiempo real o de la distancia de seguridad media que llevan los coches, entre otros datos, con el objetivo de prevenir un accidente en cadena manteniendo a los usuarios informados con dichos datos.

7.3. Métodos que no han sido efectivos para resolver el problema

En un principio intenté incorporar el sistema a una **RPi 1** lo que resultó una odisea tanto por la **escasez de pines**, que finalmente no tenía para todos los sensores, como por la latencia y lentitud del procesador en sí. Además, el sistema de ficheros estaba continuamente corrompiéndose y tenía que volver a formatear todo el sistema.

Otro aspecto negativo de la RPi 1 es que no incorpora WiFi y por tanto tenía que usar la Raspberry por Ethernet o con una tarjeta WiFi auxiliar por USB.

Pensé en considerar desarrollar el software del proyecto dentro de la RPi 1 pero debido a los sucesivos formateos y pérdida de datos decidí generar un repositorio en OneDrive el cuál, me aceleró mucho el proceso de desarrollo.

Otro método no efectivo fue utilizar la miniUART ttyS0 de la RPi 3 para el GPS y por tanto tuve que cambiarlo en la configuración inicial.

A la hora de analizar la perioricidad de las tareas me di cuenta de que tenía dos tareas que tardaban demasiado, “Show_Map” y “Accident_System”, por la visualización del mapa y el envío de la localización vía WiFi respectivamente, por lo que el sistema me salía no planificable además de ineficiente.

Para resolverlo, lo que hice fue dividir esas tareas en otras dos para que las nuevas tareas creadas solo chequearan si existe o no accidente interno o externo y en caso de ser así despertaran a las otras dos tareas que si tardaban más en ser ejecutadas.

De este modo solventé el problema y además el sistema es más eficiente y rápido.

Por último, intenté usar la velocidad del motor y no la velocidad del acelerador para comprobar la velocidad máxima a la que se podría ir con cierta distancia de seguridad, pero esto no fue efectivo debido a que primero hay que comprobar que velocidad queremos que tenga el vehículo y si es apta entonces actuar sobre el motor para que el vehículo adquiera dicha velocidad y no al revés.

7.4. Indicación de si los resultados están de acuerdo o no con los resultados anteriores

Todos los resultados están de acuerdo con los resultados previos que se estimaban antes de ejecutar las pruebas finales.

Aunque es cierto que para obtener estos resultados se hicieron pruebas previas las cuales no todas fueron satisfactorias en primera instancia como el motor el cual me esperaba una salida lineal y resultó darme una salida lineal y exponencial en algunos rangos para lo cual tuve que mapearlo tal como expliqué con anterioridad.

Cabe destacar que el resultado de localización del GPS es mayor del esperado suponiendo que me daría una localización más global pero lo cierto es que tiene un error por debajo del esperado, por debajo de los 300 metros. Lo malo de este sensor GPS es que tarda mucho en coger señal del exterior y en interiores ni siquiera coge señal, es por lo que, este sensor en una versión final también debería cambiarse a uno mucho más fiable, rápido y con mejor antena.

Otro sensor que también le tuve que hacer un ajuste fue el sensor de distancia el cual medía 2cm por encima, para ello ajusté esos dos centímetros y después realicé una conversión de centímetros a metros para hacer que la velocidad que llevaba el coche en kilómetros por hora tuviera relación con la distancia de seguridad percibida por el sensor.

Por último, el botón que simula el pedal de freno tiene un rebote que cuando se ejecuta con la mano a veces se ejecuta erróneamente si lo pulsamos demasiado rápido, esto no ocurre si lo hacemos con el pie ya que la frecuencia del pie es menor que la de la mano, es por esto que la prueba del freno salió satisfactoria, porque se tiene en cuenta que su activación es con el pie.

7.5. Resumen general de las conclusiones que se han obtenido

El prototipo es fiable y garantiza que todas las tareas cumplen sus tiempos y se ejecutan correctamente, algo que es bastante importante en un sistema crítico de Tiempo Real.

Respecto a implantarlo en un vehículo habría que realizar diversas modificaciones en el diseño hardware, pero no muchas en el software ya que la lógica del diseño persigue la planificabilidad de dichas tareas.

Es cierto que la implementación de este tipo de sistemas supone un coste mucho más elevado de pruebas y otros requisitos más ambicioso y ajustados, pero es un buen punto de partida para empezar.

Por otro lado, he comprobado que un sistema de esta magnitud no puede ser incorporado a un sistema tan escaso como una Raspberry pi 3, es necesario un hardware mucho más potente o dos procesadores para reducir la carga de CPU frente a tareas pesadas y que no colapsen el sistema.

A pesar de ello el prototipo ha conseguido realizar todas las operaciones con éxito lo que indica que es robusto y asegura el 100% del potencial del sistema para garantizar la máxima seguridad acorde con los requisitos establecidos.

7.6. Implicaciones futuras

Este sistema es un prototipo y la primera implicación futura es desarrollar un diseño hardware para un vehículo real, el software sobre el que se va a asentar ese hardware será el implementado en este proyecto, aunque no en la versión final ya que, claro está se requiere de una mayor precisión y mecanismos que aseguren una certificación y legalidad de uso de estos sistemas.

La mayor implicación futura será la modificación del procesador haciendo este exclusivo para sistemas de Tiempo Real e incorporando un sistema operativo certificado que ayude a la planificación de las tareas como ARINC 653 (Avionics Application Standard Software Interface) para aviónica.

Otro factor importante para tener en cuenta en trabajos futuros es el uso de herramientas automáticas para pasar pruebas exhaustivas de alto y bajo nivel.

No obstante, y puesto que este sistema no es específico de un único vehículo, sino que se conectan entre sí, se debe de crear una infraestructura entre estos vehículos mediante GPRS o algún otro modelo en el que se obtenga un nivel de seguridad elevado debido a que si alguien consiguiese entrar en esta red podría ser fatal para el correcto funcionamiento del sistema incluso proporcionando accidentes, es por lo que hay que tener especial hincapié en esta parte del proyecto.

Por último, sería un gran avance no solo que los coches sean capaces de comunicarse entre sí, si no también enviar datos de velocidad distancia de seguridad, ente otros a la DGT para que los analice y sea capaz de realizar métricas de las carreteras por si hubiese alguna anomalía comunicarlo a través de los paneles de las carreteras o interactuar de algún otro modo. De esta forma, las carreteras estarían mucho más controladas y serían más fiables, por tanto, se reduciría considerablemente el número de accidente.

Se sabe que hoy en día ningún sistema puede sustituir al piloto, pero sí puede ayudarle a tomar decisiones, prevenir accidentes, alertar de otros posibles accidentes o incluso, por qué no, de interactuar con otros vehículos indicándoles cual va a ser su próximo destino. Aún queda mucho por hacer, pero esto es un buen comienzo para empezar a ver cómo se comportan estos sistemas.

8. Referencias

[1] [En línea]. Available: https://www.prevencionintegral.com/comunidad/blog/inseguridad-cuando-prevencion- fracasa/2017/02/21/alexis-evdemon-reflejos-tiempo-reaccion-evitacion-accidente. Accedido el 10/07/2018

[2] [En línea]. Available: https://www.autobild.es/noticias/los-conductores-no-respetan-distancia-seguridad-240547. Accedido el 10/07/2018

[3] [En línea]. Available: https://www.seguridad-vial.net/conduccion/reglas-circulacion/66-distancia-de-seguridad. Accedido el 10/07/2018

[4] [En línea]. Available: https://www.race.es/seguridadvial/formacion-race/en-carretera/distancia-de-seguridad. Accedido el 10/07/2018

[5] [En línea]. Available: http://cadenaser.com/ser/2017/07/14/sociedad/1500032366_435466.html. Accedido el 10/07/2018

[6] [En línea]. Available: https://www.seguridad-vial.net/vehiculo/seguridad-pasiva/156-el-sistema-adas-ayuda-a-prevenir-accidentes-de-trafico-a-los-conductores. Accedido el 10/07/2018

[7] [En línea]. Available: http://www.circulaseguro.com/sistemas-de-frenado-automatico-ante-que-frenan-y-ante-que/. Accedido el 10/07/2018

[8] [En línea]. Available: https://www.seguridad-vial.net/vehiculo/seguridad-pasiva/120-que-es-el-ecall. Accedido el 10/07/2018

[9] [En línea]. Available: https://www.tesla.com/autopilot. Accedido el 10/07/2018

[10] J. García Martín, Sistemas de Tiempo Real, ETSISI - Universidad Politécnica de Madrid.

[11] [En línea]. Available: www.dit.upm.es. Accedido el 10/07/2018

[12] [En línea]. Available: https://business.tutsplus.com/es/tutorials/controlling-dc-motors-using-python-with-a-raspberry-pi--cms-20051. Accedido el 10/07/2018

[13] [En línea]. Available: http://fpaez.com/sensor-ultrasonico-hc-sr04-para-raspberry-pi/. Accedido el 10/07/2018

[14] [En línea]. Available: http://blog.whatgeek.com.pt/2015/03/connect-a-gps-to-the-raspberry-pi/. Accedido el 10/07/2018

[15] [En línea]. Available: https://raspberryparatorpes.net/comandos/como-configurar-el-wi-fi-antes-de-iniciar-la-raspberry-pi-por-primera-vez/. Accedido el 10/07/2018

[16] [En línea]. Available: fuenteabierta.teubi.co/2013/02/asegurando-el-ssh-en-la-raspberrypi-o.html. Accedido el 10/07/2018

[17] [En línea]. Available: https://spellfoundry.com/2016/05/29/configuring-gpio-serial-port-raspbian-jessie-including-pi-3/. Accedido el 10/07/2018

[18] [En línea]. Available: nmea.io/. Accedido el 10/07/2018