Seguretat Vial. Disseny d'un sistema anticol·lisió amb Arduino.

Memòria de Projecte Final de Grau

Grau en Tecnologies de Telecomunicació

Menció de Telemàtica

Autor: Elisabet Sangrà Solé

Consultor: Antoni Morell Pérez

9 de juny de 2019

FITXA DEL TREBALL FINAL

Títol del treball :	*Seguretat Vial. Disseny d'un sistema anticol·lisió amb Arduino.*
Nom de l'autor :	*Elisabet Sangrà Solé*
Nom del consultor :	*Antoni Morell Pérez*
Nom del PRA :	*Pere Tuset Peiró*
Data de lliurament :	*06/2019*
Titulació o programa :	*Grau en Tecnologies de Telecomunicació*
Àrea del Treball Final :	*TFG - Arduino*
Idioma del treball :	*Català*
Paraules clau :	*Arduino, sensors, sistema anticol·lisió*

Resum del Treball:

El sector de l'automoció està canviant i cada vegada més tendeix cap a l'autonomia. És per això que en els darrers anys s'ha experimentat amb la sensorització dels vehicles. D'aquí han sorgit els primers vehicles intel·ligents i ara ja es parla dels vehicles autònoms com una realitat pròxima.

Un vehicle que capta dades de l'entorn i les processa pot ser de gran ajuda sobretot per al conductor i també per donar un extra de seguretat. Precisament en aquest treball es construeix un petit vehicle amb sensors la qual cosa permet que se li pugui implementar un sistema anticol·lisions. Aquest sistema és capaç de detectar la presència d'obstacles frontals i d'actuar de forma autònoma per evitar una col·lisió.

Per a conduir aquest projecte és important observar l'entorn i enregistrar dades tals que permetin calcular distàncies i velocitats. Això s'aconsegueix amb Arduino que és el cervell del vehicle i amb un parell de sensors: sensor ultrasò (per distàncies) i optointerruptor (per velocitats). També s'han usat altres dispositius però aquests són els més destacats.

El resultat és un sistema que processa i analitza les dades, determina els diferents escenaris i si hi ha situació de risc proporciona una resposta adequada per tal d'evitar un accident.

Abstract:

The automotive industry is increasingly changing and becoming more autonomous. In recent years, there has been an explosion in sensor car technology, which has led to the appearance of smart cars. With more and more advances in this field, the autonomous car will certainly come soon onto the market.

A vehicle which collects and processes data from the environment is of great help to the driver and ensures extra security. The project's aim consists of building a small car with sensors in order to implement a collision avoidance system, designed to prevent accidents by detecting the presence of vehicles in front. In case of an imminent collision, the system acts autonomously without any intervention by the driver.

The nature of this project makes it necessary to collect data from the surrounding environment in order to exactly calculate distances and speeds; which is achieved by Arduino, the car's brain, and by two primary sensors: ultrasound sensor (distances) and optointerruptor (speeds). Other devices are also used but these two are the most valuable.

As a result, the system processes and analyzes data, establishes different scenarios and provides an adequate response if there is risk of an accident.

Índex

Figures i taules

Índex de figures

Índex de diagrames

Índex de taules

Capítol 1: Introducció

1.1 Introducció

El futur és *smart*. Les dades han eclipsat la societat i han permès l'eclosió de noves tecnologies (IoT, Big Data, AI...). El món s'ha omplert d'aplicacions i sistemes tals que recullen dades i les monitoren. Aquestes dades no tenen cap valor de per sí, ara bé, la incorporació d'intel·ligència a aquests sistemes fa que sigui possible l'anàlisi i el processament de les dades recollides. L'objectiu final és que el sistema sigui capaç de prendre decisions de forma autònoma i eficient.

Gran part de l'èxit dels sistemes intel·ligents és degut als sensors que són capaços de mesurar fenòmens físics com la temperatura, les pulsacions o la velocitat i transformar-ho en un senyal elèctric. Això dona molt joc als sistemes electrònics quan utilitzen sensors en els seus dissenys, ja que es permet implementar una determinada resposta segons el que hagi detectat el sensor.

Avui en dia, l'ús de sensors juntament amb el tractament intel·ligent de les dades s'aplica en infinitat de camps: en robòtica, domòtica, *Smart Cities*, el sector industrial (*Industry 4.0*), el sector del transport, l'àmbit de la medicina... i en qualsevol tendència *smart*. Es pot dir que el món s'està tornant intel·ligent.

I si el món s'està tornant intel·ligent, també ho estan fent els nostres automòbils. Precisament el tema d'aquest Treball de Final de Grau és el disseny i implementació d'un sistema anticol·lisions proactiu elaborat a partir d'Arduino i sensors.

1.2 Descripció del projecte

El present treball consisteix en el disseny i implementació d'un sistema anticol·lisions proactiu, és a dir, un sistema que davant del perill de col·lidir reacciona de forma autònoma tot garantint la seguretat vial.

Aquest sistema es presenta prototipat en un robot en forma d'automòbil al qual se li ha incorporat tota una sèrie de sensors. Arduino és el nucli del robot, és qui processa i analitza les dades enregistrades pels sensors. És també qui identifica les diferents situacions i genera una resposta motriu adequada a través del moviment de les rodes.

El projecte parteix d'una situació de trànsit quotidiana: el vehicle de davant frena sobtadament. En aquesta situació, el més probable és que el sistema anticol·lisions aturi el vehicle evitant que aquest col·lideixi o com a mínim mitigui l'impacte. La qüestió és, es podria haver evitat aquesta situació? Es podria haver previst l'accident? Es podrien haver adoptat mesures preventives? És sempre la millor solució frenar? Què s'ha de tenir en compte per a prevenir i evitar un accident?

Un dels elements més importants d'un sistema anticol·lisió és que aquest pugui detectar obstacles i en pugui determinar a quina distància es troben. Aquest fet dona molt de joc per implementar el sistema, ja que segons la distància que separa el vehicle de l'obstacle es poden plantejar diferents escenaris i donar diferents respostes: adaptar la velocitat per mantenir la distància de seguretat, frenar per evitar una col·lisió imminent o no fer res. Tots aquests són aspectes que s'han anat treballant durant el projecte.

La primera part del projecte s'ha centrat en construir el robot, incorporar-li els sensors i programar l'Arduino per a què el robot es pogués moure en totes direccions. Com s'ha dit, és important poder detectar obstacles, per això, un dels sensors emprats és un mòdul ultrasò que a més permet calcular distàncies. La segona part del projecte ha consistit en implementar el sistema anticol·lisió i adaptar-lo a Arduino.

Des del punt de vista tècnic, aquest projecte combina electrònica amb programació. Dos aspectes que s'han anat treballant al llarg del grau a través de diferents assignatures. Arduino és la plataforma amb la que es porta a la pràctica el sistema anticol·lisions. Tot i les limitacions d'Arduino, s'ha escollit aquesta tecnologia pel gran ventall de possibilitats que ofereix, la modularitat dels seus sensors i pel magnífic tractament de les dades.

1.2.1 Abast

Es pretén dissenyar i implementar un sistema anticol·lisió que s'incorporarà a un robot en forma de vehicle de 4 rodes. Aquest sistema serà capaç de detectar la presència d'obstacles davant seu. Depenent de factors com la distància i la velocitat, el robot reaccionarà de diferent manera. Podrà aturar-se, mantenir la distància de seguretat o bé realitzar la maniobra d'avançament. Sempre anteposant la seguretat vial a tota acció.

1.2.2 Productes obtinguts

El producte final és un robot vehicle autònom capaç de reaccionar davant d'una possible col·lisió.

A més el projecte inclou una sèrie de lliurables:

- Informes de seguiment lliurats en cada Prova d'Avaluació Contínua
- Memòria
- Codi-font del robot
- Presentació
- Demostració del funcionament del robot

1.3 Objectius del projecte

Aquest projecte posa èmfasi en els vehicles intel·ligents i la seva aportació a la seguretat vial. L'objectiu principal és recrear una situació quotidiana on el vehicle es posa en perill i fer que aquest sigui suficientment autònom per a poder analitzar-la, respondre-hi i així evitar "un accident".

Objectius principals

- Primer. S'ha de muntar un robot amb sensors controlats per Arduino i proveir-lo de mobilitat. El robot ha de poder moure's cap endavant, aturar-se i girar.
- Segon. Amb l'ajuda dels sensor d'ultrasò s'ha de poder detectar un objecte. Les dades recollides per aquest sensor han de poder usar-se per calcular distàncies.
- Tercer. A més de distàncies també s'han de calcular velocitats. La velocitat a la qual circula el robot i la velocitat de l'obstacle que tenim al davant (si aquest és un mòbil).
- Quart. Implementar un sistema anticol·lisions on no es permeti que el robot impacti amb l'obstacle.
- Cinquè. S'han de definir les diferents situacions així com definir quina resposta cal donar: aturada, manteniment de carril, control de velocitats adaptiu, canvi de carril...

Objectius tècnics

- Les unitats amb les quals es desenvolupa el projecte són centímetres i segons. Tots els càlculs han de ser adaptats a aquestes unitats.
- Les dades provinent dels sensors, s'han de processar periòdicament. Si no es pot fer en temps real, que no passi d'1 segon (sobretot per a l'ultrasò i l'optointerruptor).
- Tant en el càlcul de distàncies com de velocitats, és important que no hi hagi un error superior als 5 cm (o 5 cm/s). De ser així, s'haurien de corregir les distàncies calculades amb l'ultrasò.
- El temps de reacció del sistema no ha de superar el segon.
- El sensor ultrasò ha de poder detectar un objecte a un mínim d'1 m.
- El sistema ha de començar a controlar les distàncies quan l' objecte es trobi a mig metre ja que s'inicia la fase de perill.
- El vehicle s'ha d'aturar sense col·lidir amb l'obstacle deixant una distància mínima de 5 cm.
- En termes de mobilitat, el robot ha de seguir una línia recta dissenyada amb cinta aïllant negra. No es pot sortir d'aquesta línia i si ho fa, s'ha de corregir la trajectòria. El gruix d'aquesta línia ha de ser entre 2 i 3 cm.
- En el loop d'Arduino s'ha d'evitar la instrucció delay(); per tal de no afectar al temps de reacció.

- A l'hora de programar el sistema, el càlcul de distàncies i velocitat s'han de realitzar en un segon pla i no han d'interferir en el moviment del vehicle.

Objectius secundaris (opcionals: atès al temps disponible)

- Millores per al sistema anticol·lisió
    - Manteniment de carril
    - Implementar un sistema d'avís acústic.
    - Implementar un sistema de control automàtic d'intermitents.
- Aplicar el sistema anticol·lisions en altres àmbits
    - Avançaments segurs
    - Avís d'angle mort
    - Assistent de pàrquing

Objectius personals

- Iniciar-se en el món d'Arduino.
- Aplicar els conceptes d'electrònica i programació vists durant el Grau de Tecnologies de Telecomunicació.

1.4 Metodologia

Aquest és un projecte emmarcat dins la temàtica Arduino.

El treball se centra en el desenvolupament d'un sistema anticol·lisió el qual prioritza la seguretat davant possibles accidents. Es parteix d'Arduino per construir un robot sensorial, dotar-lo de mobilitat i proporcionar-li intel·ligència. Es busca que aquest robot pugui respondre a l'entorn de forma òptima. El resultat final és un prototip que simula ser un cotxe intel·ligent amb un sistema anticol·lisió integrat.

La finalitat serà recrear tota una sèrie d'escenaris i observar com actua el robot de forma autònoma i quines decisions pren. Un exemple seria posar un obstacle davant del robot, el robot hauria de ser capaç de detectar-lo i reaccionar-hi ja sigui aturant-se sense impactar o avançant-lo.

Formalment, i com tot projecte de desenvolupament tècnic, aquest compta amb dues grans etapes:

- Disseny. L'etapa de disseny és una etapa més conceptual en la qual es treballen aspectes teòrics del projecte: com serà el robot, quins sensors necessitarem, quin serà l'esquema elèctric, quines funcionalitats tindrà...

- Muntatge i implementació. La part del muntatge consisteix a fabricar el robot de forma física prioritzant en l'esquema elèctric i la disposició dels sensors. La part d'implementació consisteix bàsicament en materialitzar totes aquelles idees i funcions a través de codi.

El projecte es validarà de forma pràctica a partir de tot un seguit de proves que comprovin que efectivament el robot actua correctament davant de determinades situacions.

1.5 Planificació

El tempo del projecte ve marcat per l'entrega de les diferents PACs on cada entrega marca una etapa del projecte.

DIAGRAMA DE FITES			
Nom	**Durada**	**Inici**	**Final**
Decisió del projecte	6 dies	20.02.2019	26.02.2019
PAC 1 – Planificació	6 dies	27.02.2019	05.03.2019
PAC 2 – Primer lliurament	34 dies	06.03.2019	09.04.2019
PAC 3 – Segon lliurament	41 dies	10.04.2019	21.05.2019
Entrega memòria	18 dies	22.05.2019	09.06.2019
Entrega presentació i codi	6 dies	10.06.2019	16.06.2019
Defensa del projecte	4 dies	17.06.2019	21.06.2019

Taula 1: Taula de fites

El procés de treball s'ha dividit en mòduls per tal de facilitar-ne la planificació. Cada mòdul tracta un aspecte important del projecte i té assignat una sèrie de tasques que són clau per al desenvolupament del projecte.

La implementació, donada la seva envergadura, consta de tres mòduls. Un primer mòdul (M.4) que inclou una sèrie d'instruccions bàsiques i que no té cap altra finalitat que validar que el disseny electrònic funciona. Un segon mòdul (M.5) amb introduccions avançades que són les que implementen el sistema anticol·lisió. I un tercer mòdul (M.6) que vol anar més enllà i aplicar el sistema anticol·lisions en altres àmbits.

MÒDULS		
Tema	**Identificador**	**Descripció/Tasques**
Anàlisi	M.1	Estat de l'Art Selecció i compra de material Instal·lació del software
Disseny	M.2	Definició de les funcions del robot Desenvolupament cinemàtic Disseny electrònic
Muntatge	M.3	Muntatge físic del vehicle Muntatge electrònic
Implementació	M.4 – Instruccions bàsiques	Moviment de les rodes Girs del robot Direcció del vehicle i manteniment del carril
	M.5 – Instruccions avançades	Anàlisi de l'entorn i presa de decisions Detecció d'obstacles i aturada Adequació de la velocitat
	M.6 – Ampliació (opcional)	Avançament [Smart lane change] Aparcament assistit

		Sistema de llums intermitents Sistema d'alarma sonor
Demostració	M.7	Test i proves – El cotxe/robot actua tal com s'espera d'ell? – Els resultats pràctics són coherents amb els resultats teòrics?
Documents	M.8	Redacció de la memòria Preparació del codi Presentació power point Vídeo demostració

Taula 2: Classificació en mòduls

En termes de planificació, val dir que algunes tasques que estan catalogades com "opcionals". Realitzar-les dependrà en tot moment del bon funcionament del projecte, de si sorgeixen imprevistos i del grau de solució d'aquests. En cas d'imprevist es contempla utilitzar el temps destinat a aquestes tasques optatives per abordar les tasques que sí són considerades essencials.

A continuació es mostra la planificació del projecte a través d'un diagrama de Gantt i se'n detallen les etapes corresponents a la PAC 2 i a la PAC 3.

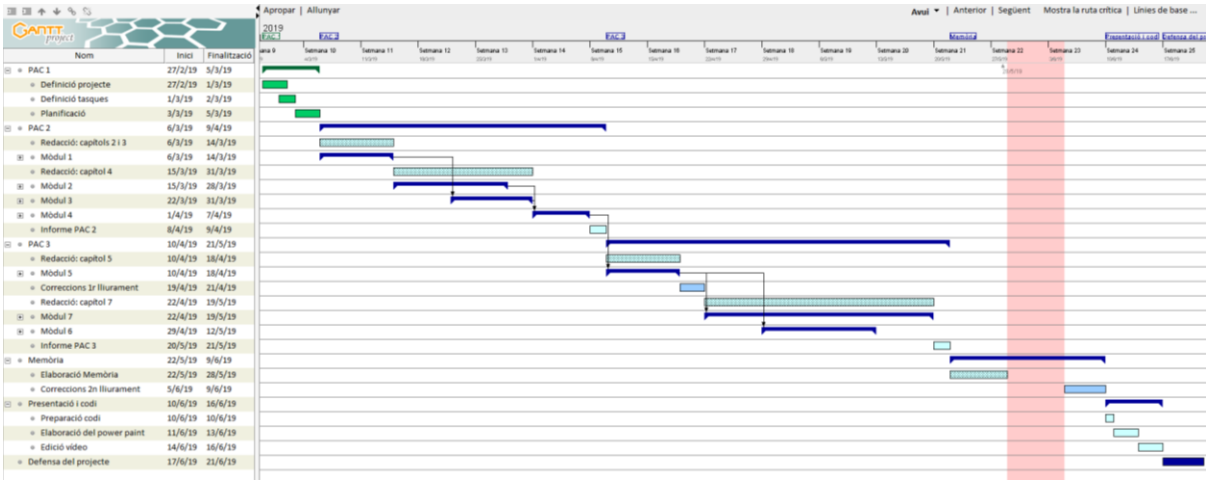
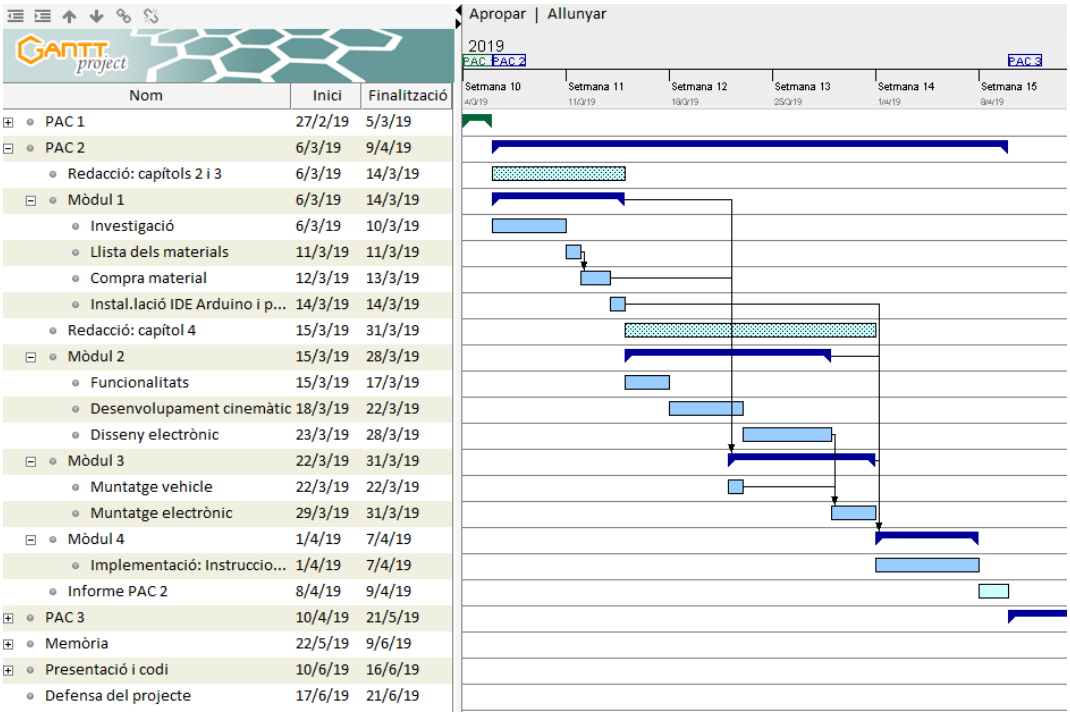
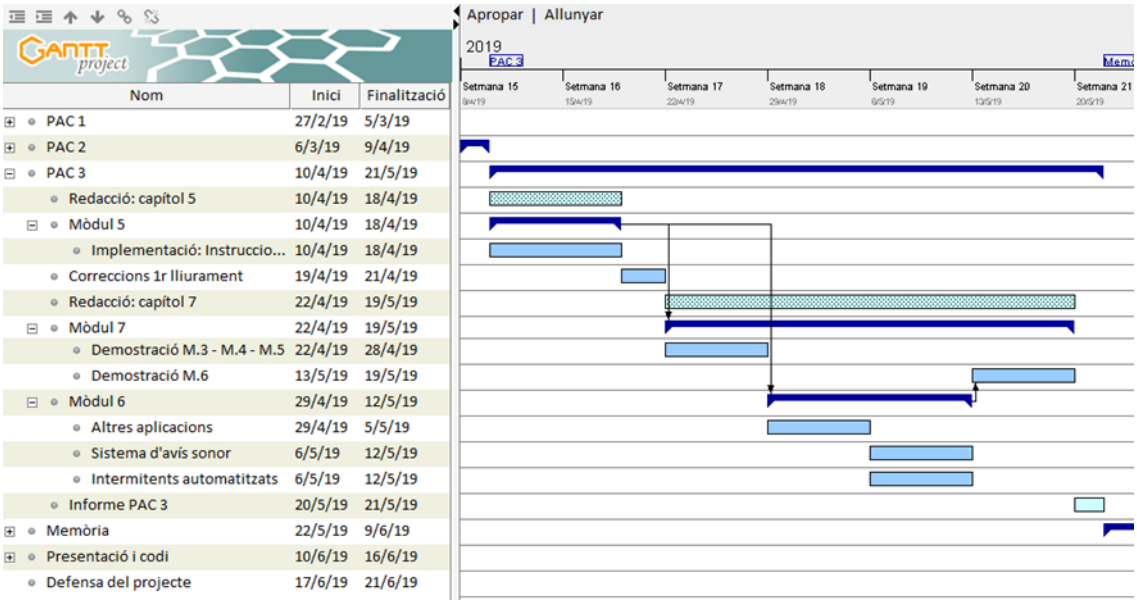


Diagrama 1: Diagrama de Gantt - Global



Nom	Inici	Finalització
PAC 1	27/2/19	5/3/19
PAC 2	6/3/19	9/4/19
Redacció: capítols 2 i 3	6/3/19	14/3/19
Mòdul 1	6/3/19	14/3/19
Investigació	6/3/19	10/3/19
Llista dels materials	11/3/19	11/3/19
Compra material	12/3/19	13/3/19
Instal.lació IDE Arduino i p...	14/3/19	14/3/19
Redacció: capítol 4	15/3/19	31/3/19
Mòdul 2	15/3/19	28/3/19
Funcionalitats	15/3/19	17/3/19
Desenvolupament cinemàtic	18/3/19	22/3/19
Disseny electrònic	23/3/19	28/3/19
Mòdul 3	22/3/19	31/3/19
Muntatge vehicle	22/3/19	22/3/19
Muntatge electrònic	29/3/19	31/3/19
Mòdul 4	1/4/19	7/4/19
Implementació: Instruccio...	1/4/19	7/4/19
Informe PAC 2	8/4/19	9/4/19
PAC 3	10/4/19	21/5/19
Memòria	22/5/19	9/6/19
Presentació i codi	10/6/19	16/6/19
Defensa del projecte	17/6/19	21/6/19

Diagrama 2: Diagrama de Gantt – PAC 2



Nom	Inici	Finalització
PAC 1	27/2/19	5/3/19
PAC 2	6/3/19	9/4/19
PAC 3	10/4/19	21/5/19
Redacció: capítol 5	10/4/19	18/4/19
Mòdul 5	10/4/19	18/4/19
Implementació: Instruccio...	10/4/19	18/4/19
Correccions 1r lliurament	19/4/19	21/4/19
Redacció: capítol 7	22/4/19	19/5/19
Mòdul 7	22/4/19	19/5/19
Demostració M.3 - M.4 - M.5	22/4/19	28/4/19
Demostració M.6	13/5/19	19/5/19
Mòdul 6	29/4/19	12/5/19
Altres aplicacions	29/4/19	5/5/19
Sistema d'avís sonor	6/5/19	12/5/19
Intermitents automatitzats	6/5/19	12/5/19
Informe PAC 3	20/5/19	21/5/19
Memòria	22/5/19	9/6/19
Presentació i codi	10/6/19	16/6/19
Defensa del projecte	17/6/19	21/6/19

Diagrama 3: Diagrama de Gantt – PAC 3

1.6 Pressupost i Viabilitat

Aquest projecte es podria desglossar en dues parts: d'una banda la implementació d'un sistema anticol·lisions i d'altra banda un prototip de baix cost fet amb Arduino.

Si ens cenyim al TFG, el pressupost del projecte no seria gaire elevat. Hauríem de considerar les hores treballades i el preu dels materials.

Hores treballades

Personal	Tasca	Hores invertides	Tarifa	Total
Cap de projectes	Planificació i seguiment del projecte	75 h	20€/h	1500
Desenvolupador tècnic	Funcionament del sistema anticol·lisions / Disseny i muntatge del robot	100 h	20€/h	2000
Programador	Implementació Arduino	175 h	25€/h	4375
Redactor	Documentació / Memòria / Presentació	40 h	10€/h	400
				Total: 8275 €

Taula 3: Recursos humans projecte

Preu del Robot

	Descripció	Compra	Data compra	Data prevista d'arribada	Preu
Microprocessadors					
Arduino Uno	Adquirit prèviament	-			0
Arduino Mega	Marca Elegoo	Amazon	11.03.19	17.03.19	14.99
Sensors i Actuadors					
Mòdul controlador de motors L298N		Conrad	11.03.19	-	9.90
Sensor d'ultrasons	HC-SR05	Conrad	11.03.19	-	5.90
Sensor d'infrarojos	Pack 5 unitats	Amazon	11.03.19	20.03.19	3.92
Optointerruptor		Reichelt	17.04.19	19.04.2019	3.60
Hardware					
Chasis		Amazon	11.03.19	14.03.19	14.99
Adaptador bateria- DC jack		Amazon	15.05.19	16.05.19	1.92
Bateria 9V	Pack 8 unitats	Amazon	15.05.19	16.05.19	8.99
Software					
IDE Arduino	Gratuït	Arduino website			0
Fritzing	Gratuït	Fritzing website			0
Accessoris i eines					
Cables M2M	Male to male	Conrad	11.03.19	-	4.99
Cables M2F	Male to Female	Conrad	11.03.19	-	4.99
Set de cables amb connector		Amazon	22.03.19	27.03.19	8.29
Pistola silicona		Amazon	22.03.19	25.03.19	9.98
Cinta aillant negra		Conrad	16.03.19	-	2.50
Cargols 3M	Pack 20 unitats	Obi	16.03.19	-	1.70
Set d'enroscadors		Amazon	29.03.19	01.04.19	14.83
					Total: 111.49 €

Taula 4: Pressupost robot

En total tindríem una despesa de poc menys de 8500 euros.

La cosa canviaria si volguéssim portar el projecte al mercat. Els més interessats en adquirir aquest sistema serien els grans fabricants d'automòbils. D'aquesta manera podrien complementar els seus vehicles amb sistemes anticol·lisions i així oferir un producte més segur.

En aquet cas, el projecte hauria de millorar qualitativament. S'ha de tenir en compte que aquest treball ha estat dissenyat pensant en Arduino i que Arduino té les seves limitacions. A més, en el món real els vehicles utilitzen processadors més potents. De totes maneres, aquest treball estableix una base per a desenvolupar un sistema anticol·lisions a nivell professional.

Per a desenvolupar un projecte d'aquesta envergadura, ens caldrien més recursos de personal i temps que els emprats. Per començar ens caldria tenir un equip de projecte que seria el que desenvoluparia i implementaria el sistema anticol·lisions. També haurien d'intervenir els departaments de màrqueting i el de ventes per tal de promocionar el producte i presentar-lo als possibles compradors.

Finalment s'hauria de crear un grup de suport i adaptació que es constituiria després de la compra del producte. La tasca del qual seria adaptar el sistema anticol·lisions al propi sistema de la companyia (cada marca té el seu propi software).

Una possibilitat seria tenir un equip de treball format per:

Personal	Tasca	Hores invertides	Tarifa	Total
Equip de projecte				
Cap de Projecte	Planificació i seguiment del projecte	150 h	40	6000
Analista tècnic	Disseny del sistema anticol·lisions	400 h	50	20000
Responsable tècnic	Disseny del sistema anticol·lisions	400 h	40	16000
Enginyer de Software	Implementació software	300 h	35	10500
Responsable de probes	Test i proves del sistema	50 h	30	1500
Departament de Comunicacions i Màrqueting				
Director de Màrqueting	Implementar la venta i anàlisi de mercat	100 h	40	4000
Editor	Presentació, vídeos, documentació	50 h	25	1250
Departament de ventes				
Agent de ventes	Contacte amb el departament de seguretat	50 h	30	1500
Agent de ventes	Contacte amb el departament d'innovació tècnica	100 h	30	3000
Responsable tècnic	Contacte departament enginyeria de software	50 h	40	2000
Departament Financer				
Director Financer	Control financer i estudi de viabilitat	25 h	50	1250
Comptable	Gestió de la comptabilitat del projecte	50 h	25	1250
Suport i adaptació del producte				
Responsable tècnic	Adaptació al sistema de la companyia	100 h	40	4000
Enginyer de software	Adaptació al sistema de la companyia	100 h	35	3500
Responsable de probes	Probes de seguretat	100 h	30	3000

Responsable de qualitat	Control de qualitat	50 h	30	1500
Altres despeses				
Desplaçaments i hotels				10000
Prototip				1000
Producció materials comunicació				2000
Registrar el producte				1500
Despeses de gestió				3000
				Total: 97750 €

Taula 5: Recursos humans projecte a gran escala

El sistema anticol·lisions resultant tindria cabuda dins del *pack* de seguretat que es pot adquirir addicionalment en la compra d'un automòbil. Aquest *pack* ronda entre 1000 i 1600 euros. Potser el preu patiria un petit increment ja que el sensor ultrasò i la seva instal·lació correria a càrrec del fabricant.

Hi hauria tres maneres de plantejar el model de negoci:

- Venta del sistema anticol·lisió (software) en la seva totalitat i permetent a la companyia fer tantes còpies com desitgi. El preu d'aquesta venta seria evidentment per un valor molt superior a aquests 97.750 euros de producció.
- Pagament de Royalties anuals a preu pactat. Per exemple 300.000 euros anuals a "x" anys.
- Negociar un petit percentatge per cada automòbil que se li instal.la el sistema anticol·lisió. Posem per cas que el sistema anticol·lisió forma part d'aquest *pack* extra de seguretat i que s'ha estipulat que ens emportem tan sols un 1%. Això representaria un mínim de 10 euros per vehicle. El retorn de la inversió l'obtindríem amb la producció de 10.000 vehicles.

Seguint amb aquest mateix model de negoci però ara prenent SEAT com la companyia que ens compra el sistema anticol·lisions. SEAT produeix en un any al voltant de 450.000 vehicles. Si cada vehicle portés incorporat el nostre producte (amb un 1%) obtindríem 4.500.000 euros, al que se li hauria de restar els 93.250 euros. Evidentment no tots els automòbils produïts portarien la tecnologia de sèrie. El mínim que necessitaríem per a fer front a la despesa, serien que 10.000 vehicles d'aquests 450.000 portessin incorporats el sistema. Xifra que representa un 2.3% del total dels vehicles produïts.

1.7 Estructura de la resta del document

Capítol 2. Anàlisi

El capítol 2 és un capítol de fons teòric que pretén donar una visió general sobre Arduino ja que és la tecnologia emprada en aquest projecte i alhora donar unes pinzellades sobre els sistemes anticol·lisions ja que aquests estan cridats a ser una de les tecnologies de futur en el camp dels *SmartCars* i dels vehicles autònoms.

Capítol 3. Hardware i Software

El capítol 3 analitza les necessitats del projecte a nivell de hardware i software. Es tracta de definir els diferents elements tant electrònics com mecànics que serviran per muntar el vehicle anticol·lisions. A més també es donaran unes pinzellades al programari i llenguatges que s'utilitzaran durant la implementació del projecte.

Capítol 4. Disseny del vehicle

El capítol 4 tracta aquells aspectes que són clau per a poder dissenyar el robot. La part fonamental d'aquest capítol és tot el disseny electrònic. En un segon pla, però també important, es defineixen les funcions que farà el robot i el desenvolupament de la física que requerirà el projecte.

Capítol 5. Implementació del robot

El capítol 5 resumeix el procés d'implementació del robot. Explica les decisions preses i fa un recorregut sobre les funcions implementades.

Capítol 6. Demostració

El capítol 6 emmarca el producte final. Posa el robot en marxa i comprova que actua de la forma esperada, és a dir, que realitza les funcions que hem definit i implementat en el projecte correctament.

Capítol 7. Conclusions i Línies futures.

El capítol 7 és el capítol de reflexió on es valora el que ha suposat l'elaboració d'aquest treball tant a nivell global com a nivell personal.

Capítol 2: Anàlisi

2.1 Arduino

El projecte Arduino és una plataforma de desenvolupament que consisteix en tota una sèrie de plaques electròniques que permeten realitzar de forma senzilla projectes interactius de qualsevol índole (robòtica, domòtica...). Darrerament ha esdevingut molt popular dins del món *maker* gràcies a la facilitat d'ús i a la flexibilitat que ofereix, sobretot perquè és un projecte de hardware de codi obert amb la qual cosa qualsevol fabricant pot modificar i adaptar els diagrames de la placa Arduino sense pagar llicència.

La placa Arduino està basada en un microcontrolador re-programable i tota una sèrie de pins d'entrada i sortida (E/S) que permeten establir connexions entre el microcontrolador i els diferents sensors i/o actuadors. Hi ha disponibles pins digitals que retornen dos estats HIGH i LOW i pins analògics que retornen rangs de tensió depenent del que el sensor capta.

D'altra banda, Arduino proporciona un entorn de desenvolupament basat en Processing i Wiring (C/ C++ simplificat). Destaca per la senzillesa a l'hora d'escriure el codi i perquè són llenguatges més comprensibles que els que normalment ofereixen els microcontroladors.

El resultat final és una placa que pot tenir infinitat usos. Es pot utilitzar per automatitzar qualsevol mecanisme com per exemple un motor que pugi o baixi les persianes segons la quantitat de llum que entra a l'habitació o per automatitzar el reg d'un hort segons la humitat del sòl. La idea darrere és captar dades de l'entorn a través dels sensors (input), processar-les i donar una determinada resposta a través d'un element que és controlat per la placa (output).

2.1.1 Elements

Si una cosa està clara és que Arduino serveix per a una gran quantitat de projectes i de diferents àmbits: domòtica, robòtica, audiovisual, eficiència energètica, moda, DIY... és per aquest motiu que existeixen tota una sèrie de sensors i actuadors que s'adapten als diferents tipus de projectes.

A continuació s'explica la diferència entre els diferents tipus d'elements que acompanyen a l'Arduino:

- Sensor

Un sensor és un dispositiu capaç de detectar magnituds físiques o químiques anomenades variables d'instrumentació i transformar-les en variables elèctriques. És a dir, el sensor recull informació de l'entorn i l'envia a l'Arduino en forma de voltatge (els microcontrolador només perceben voltatges).

Alguns dels sensors més habituals són: fototransistor, sensor de temperatura, sensor de pressió, sensor d'humitat, sensor ultrasònic, encoder...

- Actuador

L'Arduino és capaç de produir un senyal que activi un actuador. Aquest és un dispositiu controlat elèctricament el qual produeix una acció en el món físic. Un exemple podria ser l'activació de l'aire condicionador o l'activació d'un motor que mou alguna cosa.

Alguns dels actuadors més habituals són: LEDs, bronzidors, motors, bombes, vàlvules...

- Perifèric

Un perifèric és un dispositiu auxiliar i independent connectat a la unitat central de processament (Arduino). Es consideren perifèrics els dispositius de hardware a través dels quals Arduino es comunica amb l'exterior i els dispositius que serveixen com a memòria auxiliar.

Alguns dels perifèrics més habituals són: pantalles LCD, teclats, displays numèrics, memòries externes, micròfons, càmeres, impressores...

- Shields (escuts)

Un shield és un circuit imprès que afegeix funcionalitat a l'Arduino, podríem dir que és una placa extra que es connecta a l'Arduino. El shield són dues coses: hardware nou que s'adhereix al Arduino sense necessitat de cablejat i software nou que facilita la programació a través de la inclusió de llibreries per als components incorporats.

Alguns dels shields més habituals són: ethernet controller, motor drive shield, relay shield, protoshield, gameduino shield...

2.2 Vehicles amb sensors

La sensorització dels automòbils ha jugat un paper molt important per a poder desenvolupar aquestes tecnologies preventives d'accidents. Aquests sensors permeten recollir dades fiables del que succeeix al voltant del vehicle i determinar si hi ha possibilitat de risc. A partir d'aquí, és el propi vehicle el qui determina quina acció prendre ja sigui una acció passiva com és la notificació al conductor o una acció activa com seria l'accionament automàtic dels diferents sistemes del vehicle (sistema de frenat, sistema d'il·luminació...).

El vehicle intel·ligent és aquest vehicle dotat de sensors de diferent naturalesa (càmeres, ultrasons, infrarojos...) la combinació dels quals produeix un reconeixement fiable de l'entorn i del propi vehicle. Un pas més enllà trobem els vehicles connectats els quals permeten la comunicació entre vehicles (V2V) o fins i tot amb la infraestructura (V2I). D'aquí sorgeixen aplicacions interesants com la gestió intel·ligent dels semàfors o la notificació de via congestionada.

Ambdues tendències són el pas previ al vehicle autònom, aquell vehicle que es pot conduir sol i no requereix conductor. És evident que en matèria de seguretat la tecnologia aplicada ha de donar garanties molt altes. Les innovacions en els sistemes proactius són una bona base per donar aquestes garanties i una base on consolidar el vehicle autònom.

2.3 Sistemes anticol·lisió

La seguretat vial és sempre un tema prioritari per a la societat. Mentre que els governs adopten diferents mesures per intentar disminuir el nombre d'accidents, les innovacions en el sector automobilístic també se centren en aspectes de seguretat.

Les noves mesures en seguretat vial contemplen els sistemes de seguretat proactius. Aquests sistemes són capaços de fer que el vehicle actuï de forma autònoma davant d'una situació de risc. Tenint en compte que el 30% dels accidents es produeixen per distraccions al volant, aquests sistemes intenten esmenar l'error humà.

Un sistema anticol·lisió té com a objectiu detectar la presència d'altres vehicles a una distància prudencial. Quan aquesta distància s'acurta i la col·lisió és imminent, el sistema actua de forma autònoma ja sigui frenant el vehicle o prenent el control de la direcció. Mentre que els sistemes anticol·lisió per frenada són útils a baixes velocitats (velocitats urbanes), els sistemes anticol·lisió per control de direcció són més adequats per a velocitats més altes (velocitats interurbanes).

Els mateixos sensors que utilitzen els sistemes anticol·lisió també poden ser utilitzats per sistemes adaptius de velocitat. Aquests sistemes ajusten automàticament la velocitat del vehicle per mantenir la distància de seguretat amb el vehicle de davant.

També existeixen els sistemes adaptius de velocitat els quals complementen els sistemes anticol·lisió. Aquests sistemes utilitzen els mateixos sensors però en aquest cas s'utilitzen per ajustar automàticament la velocitat del vehicle i així mantenir la distància de seguretat amb el vehicle de davant.

Capítol 3: Hardware i Software

3.1 Hardware

3.1.1 Chasis

L'estructura del vehicle s'ha adquirit via Amazon. Es tracta d'un kit DIY (Do It Yourself) que inclou 2 plaques sobre les quals es munta l'Arduino i els diferents sensors, 4 rodes, 4 motors d'engranatge DC i la caixa de bateria.

Figura 1: Kit Smart Robot Car chasis 4WD

3.1.2 Arduino UNO vs Arduino MEGA

Existeixen una gran varietat de plaques Arduino. Les més usades però són Arduino UNO i Arduino MEGA. A continuació es detallen algunes característiques tècniques d'aquestes dues plaques.

	UNO	MEGA
Processador	ATmega328P	ATmega 2560
Memòria Flash (kB)	32 (0.5kB pel bootloader)	256 (8kB pel bootloader)
EEPROM (kB)	1	4
SRAM (kB)	2	8
Digital I/O pins	14	54
Digital I/O pins amb PWM output	6	14
Analog pins	6	16
Velocitat de rellotge (MHz)	16	16
Línies de comunicació	1	4

Compatibilitat amb Shields	Sí	Sí
Ethernet/WiFi/Bluetooth	Afegint mòduls	Afegint mòduls

Taula 6: Arduino UNO vs Arduino Mega

Un pas important és escollir sobre quina placa es fonamentarà el projecte. Com es pot veure en la taula, tant UNO com MEGA tenen unes característiques semblants. MEGA se sol usar per a projectes més potents on cal un gran nombre de pins. UNO acostuma a ser la placa recomanada per als *beginners* (els que s'inicien en aquest món i, per tant, realitzaran projectes més senzills).

Per al nostre projecte la decisió vindrà més endavant quan es faci tot el plantejament electrònic i es contempli el nombre de pins que necessitarem.

3.1.3 Actuadors i sensors

Mòdul controlador de motors L298N

Figura 2: Mòdul L298N

Aquest mòdul ens permet controlar la velocitat i la direcció de dos motors de corrent contínua (DC) gràcies al doble pont-H que porta integrat. Un pont-H (H-bridge) és un component format per 4 transistors que permet invertir el sentit del corrent. D'aquesta manera s'aconsegueix invertir el sentit de gir del motor.

Es treballa en un rang de tensions d'entre 3V i 35V i una intensitat de fins a 2A. A més, porta un regulador de tensió (*jumper*) que segons si està activat o desactivat el pin del V lògic (5V) serà d'entrada o sortida.

Per connectar els dos motors al mòdul es disposa de dues sortides, una per al motor A i una altra per al motor B. Addicionalment hi ha tota una sèrie de pins per controlar la velocitat de gir de cada motor (ENA i ENB) amb valors d'entre 0 i 255; i pins per controlar el sentit (IN1 , IN2, IN3, IN4) a través d'instruccions HIGH i LOW.

ENA		ENB		
IN1	IN2	IN3	IN4	Direction
L	L	L	L	Stop
H	L	H	L	Forward
L	H	L	H	Reverse
H	L	L	H	Left
L	H	H	L	Right

L = Low
H = High

Taula 7: Descripció pins L298N

Aquest mòdul serà d'utilitat per controlar les rodes de la part dreta i de la part esquerra del vehicle per separat. Es podrà determinar la direcció, el sentit i la velocitat.

Sensor ultrasònic HC-SR04 / HY-SRF05

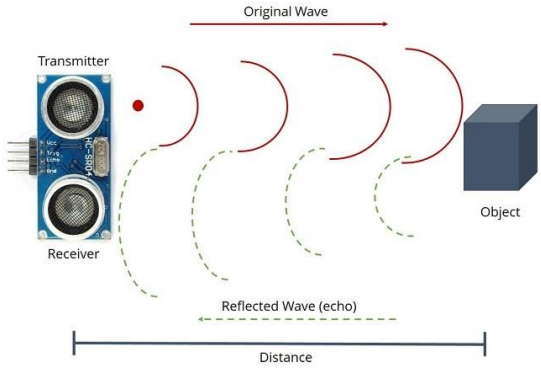


Figura 3: Sensor ultrasònic

Aquest és sensor de distància que s'emplea per detectar si hi ha la presència d'un obstacle davant. El seu funcionament consisteix en emetre un so ultrasònic per un transductor i esperar a què aquest ultrasò sigui rebotat (si hi ha presència d'obstacles). Si hi ha rebot es produeix un eco que és captat per un segon transductor i, per tant, es pot concloure que hi ha un obstacle. A partir de la demora de temps entre el moment en què s'emet l'ultrasò i el moment en què es rep l'eco, és possible calcular-ne la distància a la qual es troba l'obstacle.

El sensor HC-SR04 presenta 4 pins: VCC (5V d'alimentació), GND (*ground*), *trigger* (input - pin per disparar el senyal) i *echo* (output - pin per detectar l'arribada de l'eco). El rang de medició va des de 2 cm a 400 cm teòrics, a la pràctica són 200 cm.

Dins del projecte aquest sensor ens ajudarà a detectar la presència d'altres vehicles o qualsevol altre obstacle que hi hagi al davant del prototip. El càlcul de la distància serà imprescindible per aconseguir mantenir la distància de seguretat.

Optointerruptor

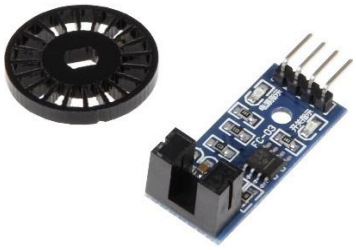


Figura 4: Optointerruptor

Els optointerruptors són sensors amb forma d' "U". Un dels extrems conté un díode emissor d'infrarojos; l'altre, un fototransistor receptor del senyal. Quan un objecte s'interposa entre els dos extrems, interromp el raig de llum infraroig de manera que el fototransistor no rep senyal.

Per al càlcul de velocitats, s'acostuma a acompanyar aquest sensor amb un codificador de velocitat - el que vindria a ser la "roda negra amb forats" de la imatge-. La idea és que aquest codificador s'instal·la en els motors d'engranatge de manera que gira de la mateixa manera que ho fa la roda. D'aquesta manera amb l'optointerruptor es pot calcular la rotació i les revolucions per segon i a partir d'aquestes dades obtenir-ne la velocitat.

L'optointerruptor té 4 pins: VCC (5V d'alimentació), GND (*ground*), D0 (sortida digital) i A0 (sortida analògica).

Gràcies a aquests dos elements es podrà calcular la distància recorreguda i la velocitat el vehicle.

Sensor IR seguidor de línies

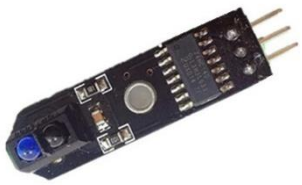


Figura 5: IR Linefollower

Un dels dispositius més utilitzats per al seguiment de línies és el sensor d'infrarojos. Aquest es compon d'un LED infraroig (emissor) i un fototransistor (receptor). Presenta un funcionament semblant al del sensor d'ultrasons però en aquest cas el que s'emet és una llum infraroja. Si la llum impacta contra una superfície blanca, aquesta es reflectirà i arribarà al fototransistor. Si per el contrari, la superfície és negra, el material absorbirà gran part de la llum i no arribarà al fototransistor.

El sensor IR presenta 3 pins: V+ (font d'alimentació), GND (*ground*) i S (output). El resultat que s'obté és 1 quan la superfície és blanca (HIGH) i 0 quan és negra (LOW). Alguns sensors d'infrarojos incorporen un potenciòmetre que permet regular la sensibilitat del fototransistor.

Dos d'aquests sensors instal·lats al davant del vehicle, permetrà al vehicle circular en línia recta (tot seguint una línia negra). Si en algun moment es detecta que el vehicle es desvia i surt de la línia, automàticament serà reconduït cap a aquesta línia negra.

3.2 Software

3.2.1 IDE Arduino

Per treballar amb Arduino generalment s'utilitza l'IDE d'Arduino. Un IDE (Entorn Integrat de Desenvolupament) és un entorn de programació a través del qual es poden crear programes en aquest cas per a la placa Arduino.

Les principals funcions que ofereix aquest entorn són: editor de codi, compilador, depurador i, molt important, la funció *upload*. Aquesta darrera funció serveix per carregar el programa a l'Arduino. Bàsicament el que fa és fer ús del *bootloader* del microcontrolador el qual permet carregar el codi sense necessitat de hardware addicional.

El programa creat per Arduino se'l coneix amb el nom d'*sketch*.

3.2.2 Llenguatges de programació

Arduino es basa en els llenguatges de programació Processing i Wiring. De fet l'estructura d'un sketch d'Arduino recorda a la de Processing i Wiring amb les dues funcions principals.

L'sketch s'estructura en dues funcions:

- Void setup(): inicialització de les variables. És la primera funció que executa el programa i només s'executa una vegada.
- Void loop(): és el nucli del projecte doncs conté el programa en sí. Després de la funció void, s'executa contínuament aquesta funció. Es llegeixen les entrades i s'activen les sortides.

Ambdues són imprescindibles per a què el programa funcioni.

El codi s'escriu en C++. No és un C++ pur sinó que està simplificat i prové d'avr-lib, una llibreria de C d'alta qualitat per als microcontroladors AVR d'AMTEL (els d'Arduino). A part d'usar les típiques estructures de C++, aquest entorn ofereix tota una sèrie de llibreries que faciliten la programació dels pins d'entrada i sortida així com diverses llibreries per a poder controlar els sensors i actuadors.

3.2.3 Programari complementari

- Ganttproject: programari de codi obert amb llicència GPL per a la planificació de projectes usant el diagrama de Gantt.
- Fritzing: programari lliure amb llicència GNU per a la realització d'esquemes elèctrics en projectes d'Arduino.
- Filmora9 i BeeCut: editor de vídeos. El compte gratuït permet exportar els vídeos amb marca d'aigua.

Capítol 4: Disseny del robot

4.1 Funcionalitat

El disseny del robot contempla les següents funcionalitats:

- Mobilitat del robot: cap endavant, cap endarrere, gir a la dreta, gir a l'esquerra i aturada.
- Seguiment de línia recta negra.
- Detecció d'obstacles frontals.
- Càlcul de distància entre el robot i l'obstacle frontal.
- Càlcul de velocitats: del robot i de l'obstacle.
- El robot no col·lideix amb l'obstacle frontal.
- Manteniment de la distància de seguretat.
- El robot pot variar la velocitat si ho requereix.
- Realització d'un avançament o canvi de carril.
- En cas de perill s'anul·la la maniobra d'avançament.

Aquest projecte parteix de la base que en presència d'un obstacle, s'ha de poder calcular la distància i les velocitats del robot i de l'obstacle (si aquest és un mòbil). Amb aquestes dades es treballarà per a definir els possibles escenaris i donar una resposta segons el cas.

Els escenaris estudiats són:

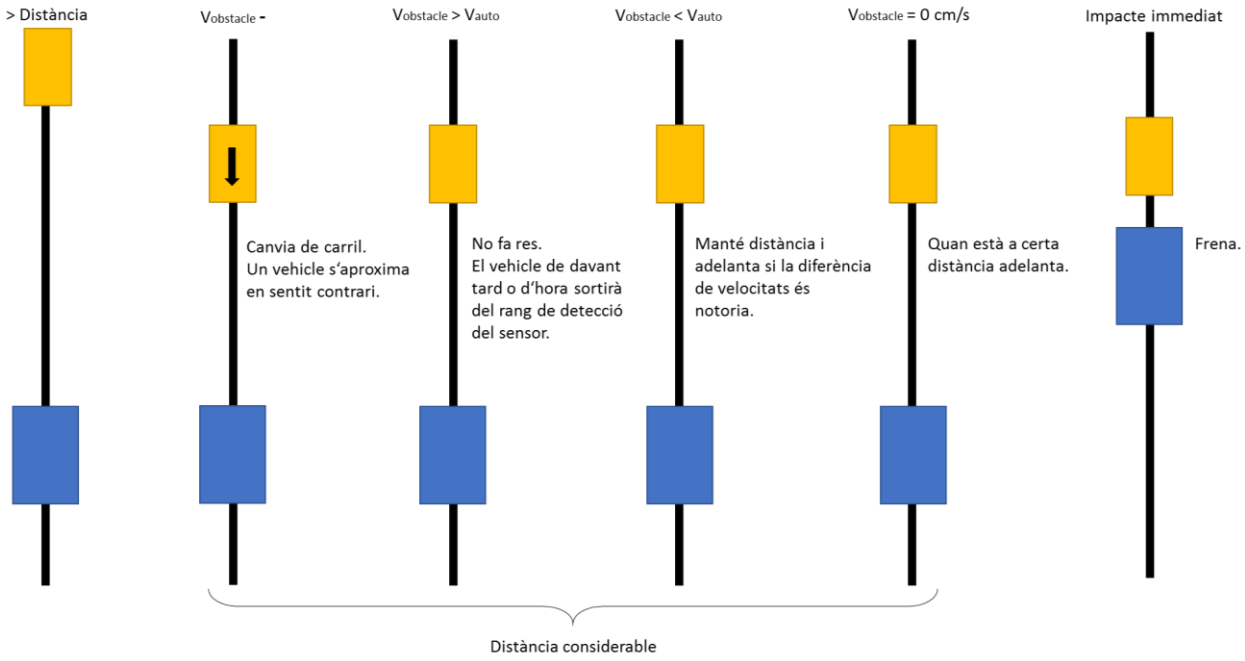


Figura 6: Escenaris sistema anticol·lisió

En funció de la distància:

- L'obstacle està fora del rang del sensor o a una distància massa gran – s'ignora.
- L'obstacle es troba a una distància considerable i s'ha de començar a controlar la resposta – l'acció dependrà de la comparació de velocitats.
- L'obstacle està massa a prop i hi ha perill de col·lisió – frena.

En el moment en què s'han de considerar les velocitats dels mòbils, tot dependrà de la diferència de velocitats. Si la velocitat de l'obstacle és bastant superior, el robot no adoptarà cap mesura, ja que tard o d'hora l'obstacle desapareixerà. Per contra si la velocitat de l'obstacle és bastant inferior, el robot haurà de procedir a realitzar un avançament (o reduir la seva velocitat).

4.2 Electrònica

En el capítol anterior s'han detallat els components que s'usaran en aquest projecte. Ara és el moment de veure com connectar-los entre sí per a què el robot funcioni correctament.

El material electrònic utilitzat és:

- Arduino Mega/UNO
- Motors DC (4 unitats)
- Motor H-Pont L298N
- Sensor ultrasò HC-SR05
- Optointerruptor
- IR seguidor de línies (2 unitats)
- Set de bateries (4 piles 1.5V)
- Bateria 9V entrada jack Arduino

Els motor DC són els que produeixen el gir de les rodes. Aquests han d'anar connectats al motor H-Pont L298N ja que aquest és el que submisitrarà l'energia necesària per a que els motors DC puguin fer girar les rodes. Dos motors DC -els corresponents a la part dreta del vehicle- aniran connectats a la sortida motor A; els altres dos motors DC -els corresponents a la part esquerra del vehicle-, a la sortida motor B.

El controlador de motors també ha de ser alimentat i per això la caixa de bateríes anirà als pins +12V i GND. El pin +5V (sense el jumper) actua com a input de manera que rep també 5V per part de l'Arduino.

La resta de pins van connectats directament a l'Arduino. Es detallen en la següent taula:

Arduino		
0 - RX		
1 - TX		
2	D0	Optointerruptor
3	Echo	Sensor ultrasò HC-SR05
4	Trig	Sensor ultrasò HC-SR05
5	S	IR seguidor de línies (dreta)
6	S	IR seguidor de línies (esquerra)
7		
8	ENA	Motor H-Pont L298N
9	IN1	Motor H-Pont L298N
10	IN2	Motor H-Pont L298N
11	IN3	Motor H-Pont L298N
12	IN4	Motor H-Pont L298N
13	ENB	Motor H-Pont L298N
5V	+5V	Motor H-Pont L298N

	V+	IR seguidor de línies (dreta)
	V+	IR seguidor de línies (esquerra)
	VCC	Sensor ultrasò HC-SR05
	VCC	Optointerruptor
GND	GND	Motor H-Pont L298N
	G	IR seguidor de línies (dreta)
	G	IR seguidor de línies (esquerra)
	GND	Sensor ultrasò HC-SR05
	GND	Optointerruptor

Taula 8: Distribució i connexió de pins a l'Arduino

Val a dir que donada la gran quantitat d'elements que han d'anar connectats al pin de 5V i al pin ground de l'Arduino, s'ha considerat emplear una mini-protoboard. D'aquesta manera els cables queden més estructurats.

A continuació es mostra l'esquemàtic del que seria el disseny del prototip.

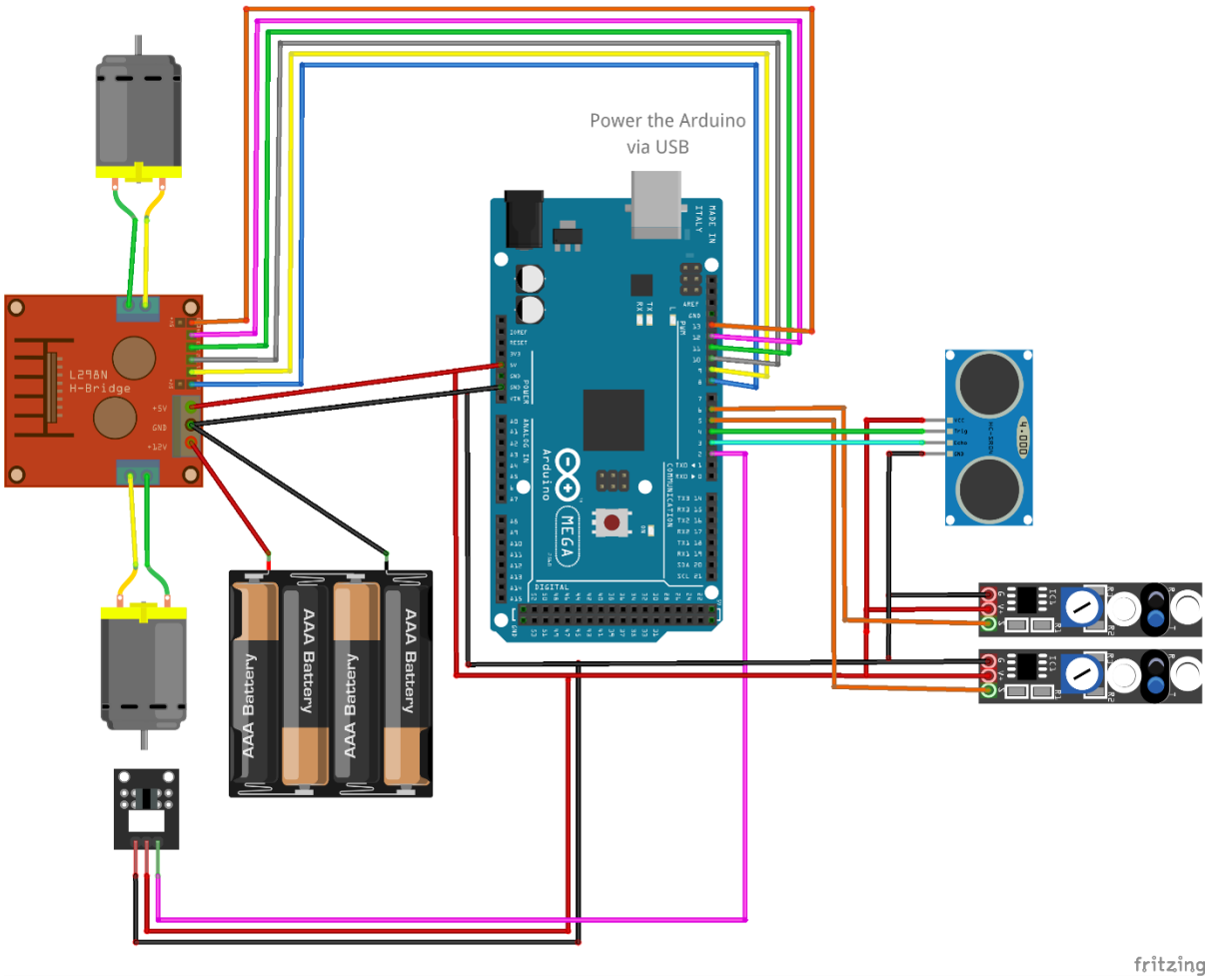


Figura 7: Esquemàtic robot

4.3 Muntatge

L'estructura del vehicle adquirida és de dos pisos:

- En el pis inferior s'han connectat les rodes al motor L298n el qual permet controlar per separat la part dreta i esquerra del vehicle. I també l'optointerruptor que ha d'anar a prop de la roda ja que mesura la velocitat a partir de la rotació d'aquesta.
- En el pis superior s'ha col·locat la placa de l'Arduino i una protoboard addicional que serveix per a alimentar tots els sensors ja que la placa no disposa de suficients pins de 5V i massa.
- En la part frontal s'ha col·locat el sensor ultrasònic i els sensors d'infrarojos que detecten la línia negra.

A continuació es mostren algunes imatges:

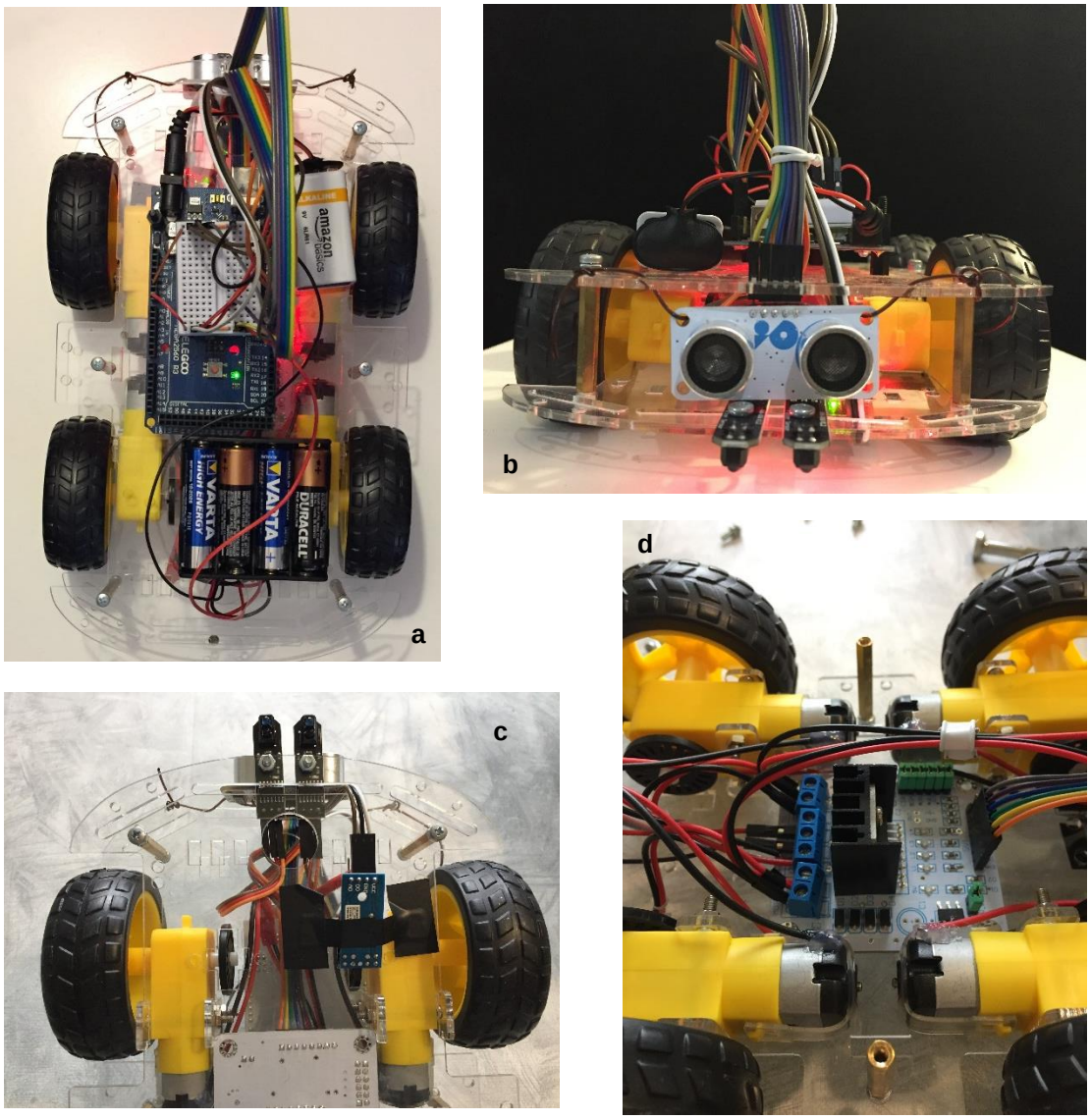


Figura 8: Muntatge robot. 8a: Primer pis. 8b: Frontal.
8c: Optointerruptor. 8d: Controlador de motors.

4.4 Cinemàtica

Mòdul ultrasò

Partint de la següent expressió cinemàtica podem trobar a quina distància es troba l'obstacle respecte a nosaltres.

$$v = \frac{distància\ recorreguda}{temps}$$

La velocitat (v) la coneixem perquè és la velocitat del so, 340m/s en condicions normals.
El temps és calculat pel sensor i correspon a la demora de temps que hi ha entre que l'ultrasò és enviat i rebut (després de ser rebotat) pel sensor.
La distància recorreguda que obtinguem serà el doble de la distància que hi ha a l'objecte. Doncs el temps obtingut pel sensor és el temps d'anada i tornada.

Velocitat del vehicle de davant

A partir de les dades obtingudes pel mòdul ultrasò, es pot calcular la velocitat mitjana a la qual circula el vehicle de davant. Només cal mesurar la distància dues vegades per a un interval de temps i amb les dades obtingudes la velocitat es pot calcular emprant la següent fórmula:

$$v_m = \frac{distància\ recorreguda_2 - distància\ recorreguda_1}{interval\ de\ temps}$$

Optointerruptor

La principal funcionalitat d'aquest sensor és poder calcular les Revolucions Per Segon (RPS):

$$rotació = \frac{Nombre\ d'interrupcions\ registrades\ (en\ 1\ segon)}{Nombre\ obertures\ del\ codificador\ de\ velocitat}$$

A partir del RPS o el que és el mateix, el número de voltes que fa una roda per segon, es pot calcular la distància que aquestes rodes han recorregut. Només cal conèixer el diàmetre de la roda per tal de saber-ne la longitud.

$$distància = \text{rotació} * (\text{diàmetre roda} * \text{pi})$$

I si aquesta distància l'ha recorregut en 1 segon, s'obté immediatament la velocitat.

Distància de seguretat

La distància de seguretat és la distància recomanada que un ha de mantenir amb el vehicle de davant i que garanteix que en moments de perill el vehicle s'aturarà abans de col·lidir. Mantenir aquesta distància és molt important per tal d'evitar accidents.

Aquest projecte pretén en tot moment que el vehicle mantingui aquesta distància.

Distància de reacció + Distància de frenat = Distància de detenció

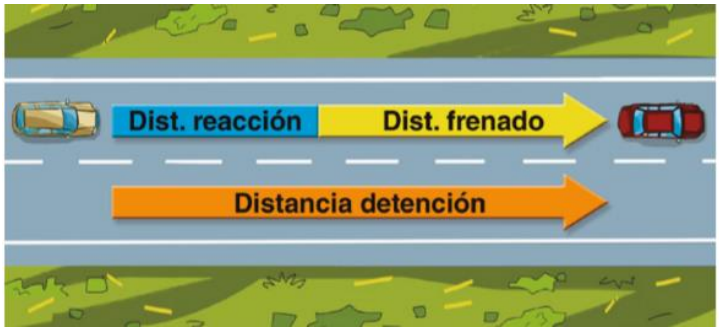


Figura 9: Distància de seguretat

La distància de reacció és la distància que el vehicle recorre entre que el conductor se n'adona del perill i prem el fre. Acostuma a ser d'entre 0.75 segons.

La distància de frenat és la distància que recorre el vehicle des de que es prem el fre fins que el vehicle s'atura per complet. A major velocitat, major distància de frenat.

Temps de gir

El vehicle es mou en línia recta però arriba un punt on ha de canviar de carril. En aquest moment la seva trajectòria ha de variar i això ho aconseguim desviant el vehicle amb girs (en direccions oposades).

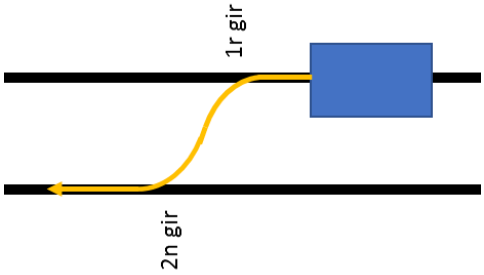
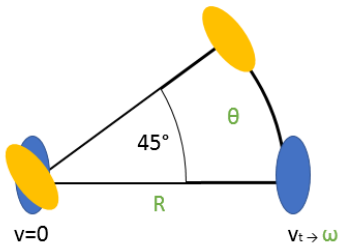


Figura 10: Trajectòria en el canvi de carril

Per a realitzar un gir, la velocitat és un paràmetre que cal considerar, ja que no és el mateix realitzar el gir a una velocitat baixa que a una velocitat alta. Per tal de trobar el temps que el robot ha de girar a una determinada velocitat, es pot fer a través de les següents equacions:



$$1\ volta\ = 2\pi\ rad = 360° \quad \rightarrow \quad 45° = 0.25\pi\ rad$$

$$v_t = \omega R; \quad \omega = \frac{R}{v_t}$$

$$\omega = \frac{\theta}{t}; \quad t = \frac{\theta}{\omega}$$

$$t = \frac{\theta \cdot R}{v_t}$$

Figura 11: Angle de gir

L'angle de gir vindrà donat com a paràmetre de la funció.

Aquest angle representa la variació de la posició de les rodes en cada gir. Donat que el robot construït no té eixos a les rodes, el gir serà menys suau que en la realitat on un simple toc de volant ja permet que el vehicle variï la trajectòria. En el cas d'Arduino s'ha de ser més creatiu i per aconseguir que el vehicle és desviï, cal moure una part del vehicle (part dreta o part esquerra) i mantenir bloquejada l'altra part.

L'angle de 45° és el que es considerarà com a referència, i a partir d'aquest és buscarà el millor angle per aconseguir el resultat desitjat.

Capítol 5: Implementació

5.1 Diagrama de blocs

A continuació es mostren dos diagrames de blocs on es pot veure la lògica del programa. El primer diagrama respon a la part implementada amb Arduino i el segon diagrama com funcionaria el sistema anticol·lisions en la seva totalitat.

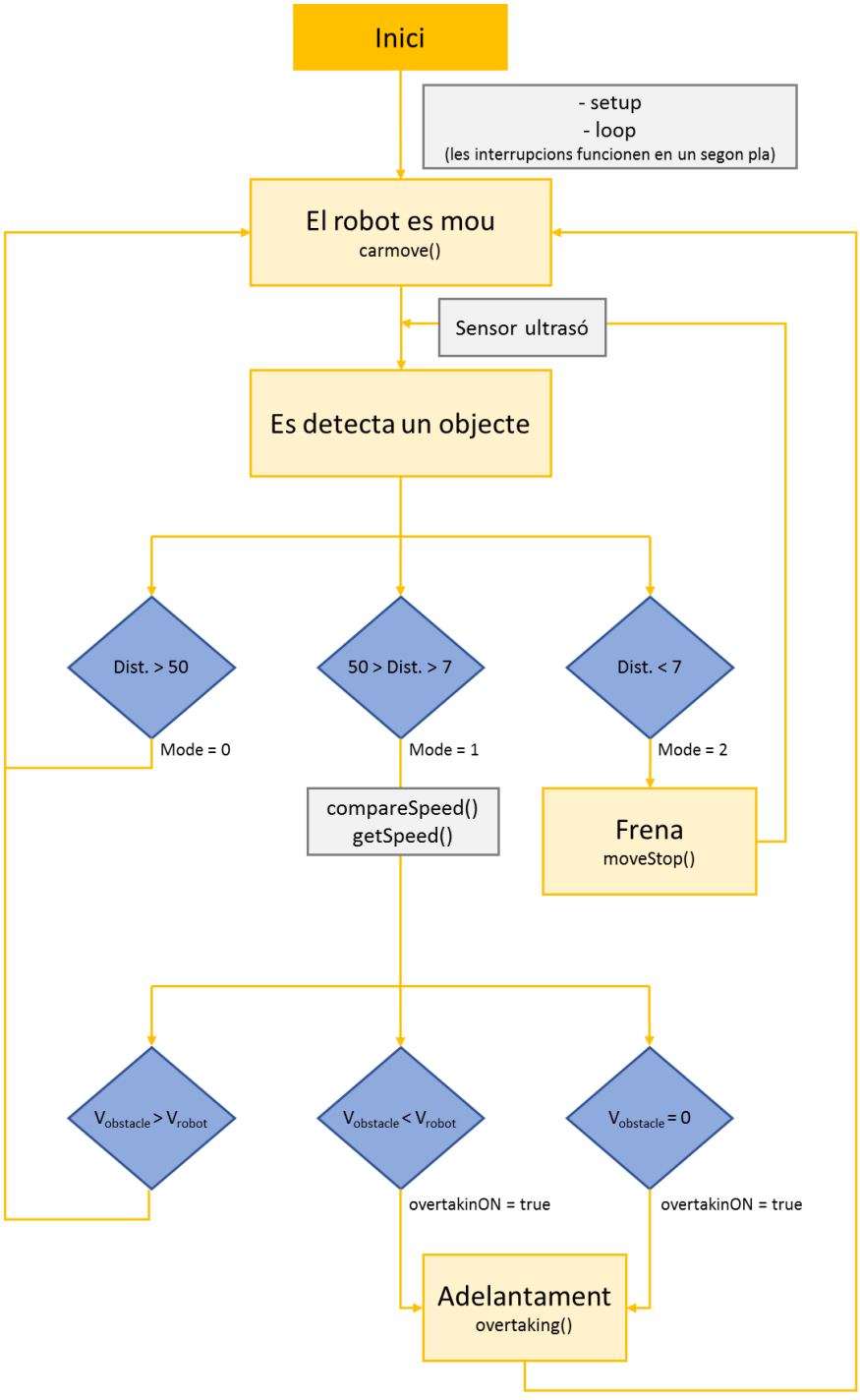


Diagrama 4: Diagrama de blocs de l'aplicació implementada

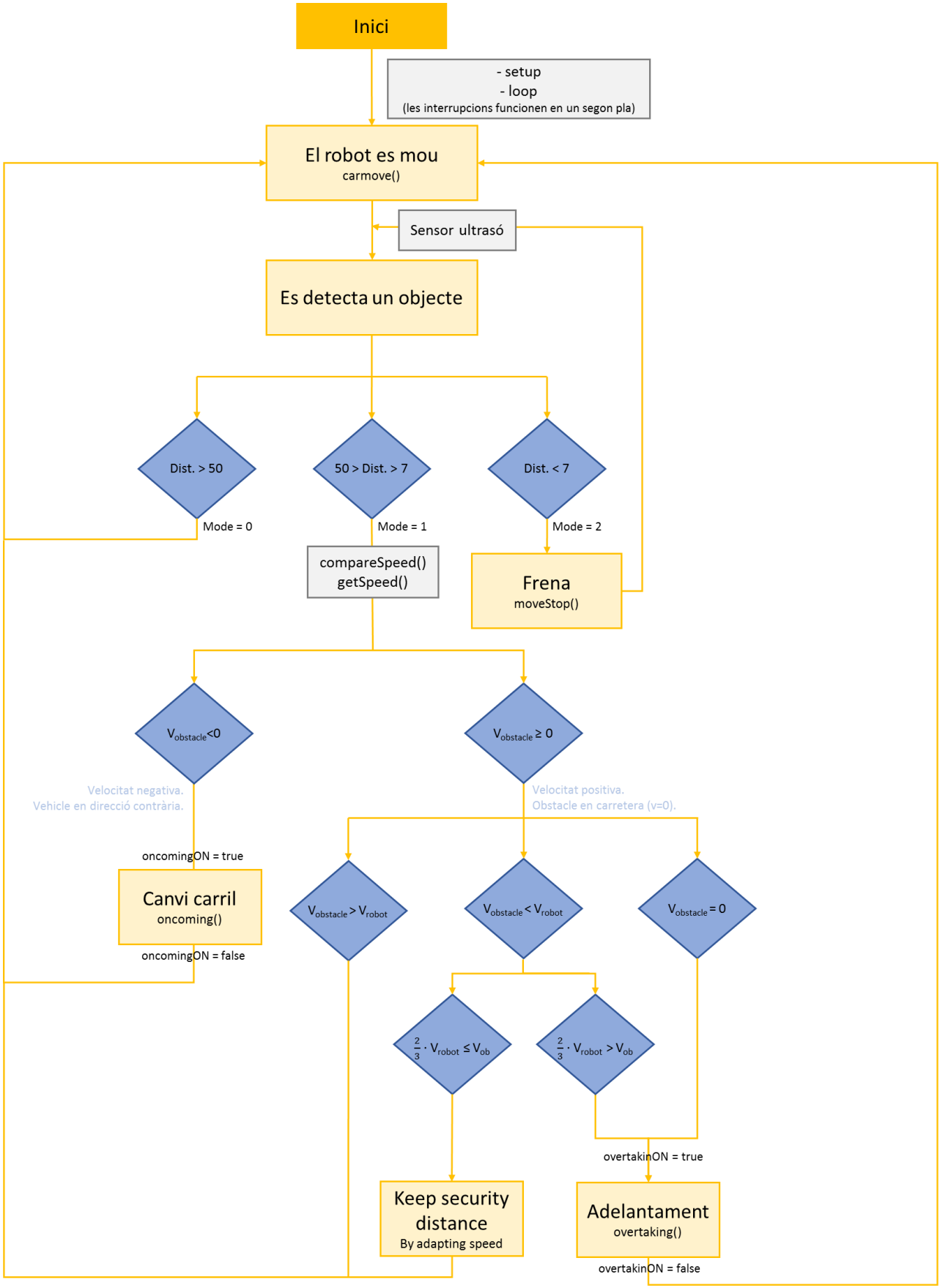


Diagrama 5: Diagrama de blocs de l'aplicació completa

5.2 Funcionament

El codi s'ha dividit en cinc parts:

- Definició de variables
- Interrupcions
- Setup
- Loop
- Funcions

Definició de variables

Primer de tot s'inclouen les llibreries TimeOne i TimeThree les quals s'han usat per temporitzar les interrupcions.

S'han declarat variables i constants.

Algunes de les variables declarades responen a la relació nom sensor amb el pin assignat. D'aquesta manera es facilita la comprensió del codi cada vegada que es vol llegir o escriure un pin ja que se'l crida per nom del sensor i no per número.

Algunes de les variables s´han definit com "volatile" això és degut a què són variables que es modifiquen en les interrupcions i, per tant, s'ha d'avisar al compilador que llegeixi la variable des de la RAM.

Interrupcions

L'ús d'interrupcions es debut a la necessitat d'obtenir una resposta immediata a les variacions dels sensors. En el nostre projecte és imprescindible obtenir en temps real la velocitat a la qual es mou el robot i la distància que hi ha del robot a l'objecte detectat.

Per això s'han implementat dues interrupcions:

- La interrupció 0.

  Aquesta calcula la velocitat, està enllaçada al pin 2 on s'ha connectat l'optointerruptor i es controla amb el TimeOne.

  Abans d'explicar com s'ha dissenyat aquesta interrupció és necessari veure el muntatge del robot. El codificador de velocitat s'ha col·locat a una roda de tal manera que aquest gira de la mateixa manera que ho fa la roda. Mentre el codificador gira, l'optointerruptor va detectant les interrupcions de llum entre els dos extrems.

Ara, per tal de calcular la velocitat, la interrupció presenta dues funcions. La primera és un comptador que comptabilitza durant un segon el nombre de vegades que es detecta una interrupció.

```
//Set up a counter.
//The sensor input is 1 when light is detected and 0 when there is no light.
//The oldValue variable helps to count a completed pulse change.
void count() {
  /*Serial.println ((int)digitalRead(blackwheel));
  Serial.print (" -- ");
  Serial.println (counter);*/
  if ((int)digitalRead(blackwheel) != oldValue){
    if ((int)digitalRead(blackwheel) == 1){
      counter++;
    }
    oldValue = (int)digitalRead(blackwheel);
  }
}
```

La segona és una funció que es crida una vegada el període de temps ha finalitzat. En aquesta es procedeix a calcular la velocitat tot considerant el nombre d'interrupcions comptabilitzat. Així es pot calcular les RPS necessàries per a obtenir la velocitat.

```
//Calculate car speed.
//First we get the number of spins that the wheel does in a second.
//Then we calculate
void speedISR() {
  Timer1.detachInterrupt();                          //stop the timer
  float rotation = counter / diskslots;              //get number of rotations a.k.a. RPS (Revolutions Per Second).
  float lenght = wheeldiameter * pi;                 //wheel lenght equals the distance run per rotation
  float f_currentSpeed = (rotation * lenght) + 0.5;  //get speed. Sum 0.5 to round the number before converting to integer.
  int currentSpeed = (int) f_currentSpeed;           //convert speed to integer
  speedAuto = currentSpeed;                          //update global variable
  /*Serial.println(counter);
  Serial.println(rotation);
  Serial.println(lenght);
  Serial.print("Speed: ");
  Serial.print(speedAuto);
  Serial.println(" cm/s");*/
  counter = 0;                                       //reset counter to 0
  Timer1.attachInterrupt(speedISR);                  //enable the timer
}
```

- La interrupció 1.

  Aquesta determina la distancia a l'objecte, està enllaçada al pin 3 on s'ha connectat l'echo del sensor ultrasò i es controla amb el TimeThree.

  Ja s'ha explicat que aquest sensor emet un ultrasò que, en cas de presència d'un obstacle, aquest és rebotat i captat pel transductor *Echo*.

  La implementació d'aquesta interrupció consta de tres funcions. Una primera funció que es crida cada vegada que hi ha un canvi en el pin *echo*. El temps que *echo* és manté High, és el temps que triga l'ona en anar, rebotar i tornar cap a l'echo. Per això en el moment en què es passa de Low a High, la funció enregistra el temps cridant a la funció micros(). Quan per el

contrari es passa de High a Low, l'ona ha arribat a l'echo i es torna a enregistrar el temps (en una altra variable). La diferència d'aquests dos temps serà el temps que l'ona ha trigat en anar i tornar. A partir d'aquí es pot calcular la distància.

```
//This function is called everytime the echo pin undergoes a change (H to L or L to H).
//From L to H: record starting time.
//From H to L: record ending time.
void echointerrupt(){
  switch (digitalRead(echo)){
    case HIGH:
      echoend = 0;                          //reset the end time
      echostart = micros();                 //records the starting time
      break;
    case LOW:
      echoend = micros();                   //records the ending time
      echoduration = echoend - echostart;   //calculates the echo duration (time the echo pin has kept HIGH)
      distance = echoduration/58;           //calculates the distance to the object | distance=(traveldistance/2)*speed of sound
      distanceOld = distanceNew;
      distanceNew = distance;
      /*Serial.print("Distance: ");
      Serial.println(distance);
      Serial.print("DistanceA: ");
      Serial.println(distanceA);
      Serial.print("DistanceB: ");
      Serial.println(distanceB);*/
      updatemode();
      break;
  }
}
```

La segona funció és l'activació del *trigger*. Un dels requeriments per a poder calcular correctament la distància és que l'ultrasò emès pel *trigger* tingui una duració de 10 µs. És important entendre que per aconseguir un tir net, cal utilitzar els delays(). Aquest és un dels motius pels quals s'ha optat per usar una interrupció i no el bucle. Perquè utilitzar endarreriments en el bucle faria que el sistema s'aturés amb la qual cosa es perdria efectivitat a l'hora de reaccionar.

```
//Triggers an ultrasound wave for 10us necessary to initialize measurement.
void distanceISR() {
  //Timer3.detachInterrupt();
  digitalWrite(trig,LOW);
  delayMicroseconds(2);
  digitalWrite(trig,HIGH);
  delayMicroseconds(10);
  digitalWrite(trig,LOW);
  delayMicroseconds(2);
  //Timer3.attachInterrupt(distanceISR);
}
```

La darrera funció, updatemode(), el que fa és assignar un mode segons la distància que hi ha entre el robot i l'objecte detectat.

Setup

En Setup s'inicialitzen les variables, es configuren els diferents pins per comportar-se com a input o output i es configuren les interrupcions.

```
//Seting up interruptions
//Timer 1
Timer1.initialize(1000000);                                          //Sets timer for 1s
attachInterrupt(digitalPinToInterrupt (blackwheel), count, CHANGE);  //Increases counter when speed sensor pin goes HIGH
Timer1.attachInterrupt(speedISR);                                    //Runs the speedISR function each time the timer period finishes.
//Timer3
Timer3.initialize(1000000);                                          //Sets timer for 1s
attachInterrupt(digitalPinToInterrupt (echo), echointerrupt, CHANGE); //starts/pauses the clock when there is a change in the echo pin
Timer3.attachInterrupt(distanceISR,5000);                            //Triggers a clean shot every half-second.
```

Loop

El bucle de control d'Arduino és el que executa el programa, en el nostre cas el que executa el sistema anticol·lisió. Aquí és on es presenten els diferents escenaris i on se'ls hi dona una resposta.

```
void loop()
{
  //Sets up the speed of the vehicle. Range from 0 to 255. Speed is set
  analogWrite(enA, 255);
  analogWrite(enB, 255);

  while (mode == 0){
    carmove();
    Serial.print ("Mode: ");
    Serial.println (mode);
  }

  while (mode == 1){
   compareSpeed();
   carmove();
    if (overtakingON == true){
      if (distanceB<30){
        Serial.print ("OVERTAKING");
      overtaking();
      }
    }else{
      carmove();
    }
    Serial.print ("Mode: ");
    Serial.println (mode);
  }
  while (mode == 2){
    moveStop();
    Serial.print ("Mode: ");
    Serial.println (mode);
  }
}
```

Les variables que intervenen directament són: mode i overtakingON.

Mode determina el mode en el qual ens trobem: 0, 1 i 2. Aquest és actualitzat cada segon per la interrupció 1. Per això l'ús de l'estructura while; mentre un mode estigui activat és realitzen determinades instruccions.

Els modes es defineixen de la següent manera:

- Mode = 0; Posada en marxa. No hi ha una resposta concreta perquè la presència o no presència d'obstacles és irrellevant. El vehicle segueix en funcionament (carmove()).
- Mode = 1; Mode de control. Aquest mode compren les distàncies entre 50 i 7 cm i depenent de les velocitats dels dos objectes (robot i obstacle) es determinarà una resposta o una altra. En primer lloc es crida a la funció compareSpeed() que a l'hora crida a getSpeed(). La segona obté la velocitat de l'obstacle i la primera compara les dues velocitats.

- Es considera que si la Vobstacle > Vrobot, en algun moment l'obstacle sortirà del rang de detecció del sensor.
- Es considera que si la Vobstacle < Vrobot, el robot en algun moment es trobarà amb l'obstacle. En aquest cas es plantegen dues situacions: si la velocitat del robot és molt superior a la de l'obstacle es procedirà a realitzar un avançament. Si, per contra, la velocitat del robot és semblant a la del obstacle, es reduirà la velocitat per tal de mantenir una distància prudent (la distància de seguretat).

  Cal tenir en compte que en la pràctica no s'ha fet aquesta distinció i el robot avança independentment de la diferència de velocitat entre els dos objectes.
- La Vobstacle = 0, no es tracta d'un mòbil sinó d'un obstacle pròpiament dit. En aquest cas es procedirà a realitzar un avançament.

```
//Compare speed to determine different scenarios
void compareSpeed()
{
  getSpeed(); //distanceA, distanceB, speedA & speedObstacle are updated

  //negative speed, a vehicle is coming towards us --> ONCOMING
  if (speedObstacle < 0){
    overtakingON = false;
    oncomingON = true;                    //Need to change to another lane. To the left is always better.
  }

  //positive speed
  if (speedObstacle > 0){
    if (speedObstacle >= speedA){         //Detected vehicle will eventually get out from our distance limit.
      overtakingON = false;
      oncomingON = false;
    }
    if (speedA > speedObstacle){          //Our speed is higher than the detected vehicle's speed.
      if((speedA*2/3) >= speedObstacle){  //Detected vehicle speed is too low --> OVERTAKING
        overtakingON = true;
        oncomingON = false;
      }else{                              //Detected vehicle speed has a similar speed --> Best option is to REDUCE SPEED.
        overtakingON = false;
        oncomingON = false;
        //analogWrite(enA, 255);          //Ideally here is where speeds are adapted. [This instruction is not implemented for this version of the prototype]
        //analogWrite(enB, 255);
      }
    }
  }

  //Detected vehicle is actually an obstacle (not a moving car) --> OVERTAKING
  if (speedObstacle == 0){
    overtakingON = true;
    oncomingON = false;
  }

  /*Serial.print("oncomingON : ");
  Serial.println(oncomingON);
  Serial.print("overtakingON : ");
  Serial.println(overtakingON);*/
}
```

Aquest mode activa l'avançament canviant el valor de la variable overtakingON a true. De totes maneres, aquesta maniobra no s'executa fins que la distància a l'objecte és de 30 cm.

- Mode = 2; Col·lisió imminent. El robot s'atura (moveStop()).

Algunes consideracions que s'han de fer d'aquest bucle són:

- no hi ha delays(). Ni tampoc en les funcions que són cridades.
- les interrupcions s'executen en un segon pla i no afecten al funcionament del loop més enllà de proporcionar-nos dades quan les necessitem.

Funcions

El programa es complementa amb tot un seguit de funcions. Hi ha funcions per controlar el moviment del vehicle, funcions per analitzar les dades i funcions que executen les respostes del robot.

Algunes d'aquestes funcions ja s'han explicat anteriorment donat que eren funcions cridades per les interrupcions o pel Loop. Tot i així, el que encara no s'ha explicat és com es mou el robot.

El codi es composa de moviments bàsics:

- moveForward() per moure's cap endavant
- moveRight() i moveLeft() per girar a dreta i esquerra respectivament
- moveStop() per aturar-se

Aquests moviments són controlats per l'Arduino i segons el que es vulgui aconseguir es transmet un HIGH o un LOW als pins corresponents. Així és com es mouen les rodes del robot [veure Taula 7].

Implementats aquests moviments, es creen funcions més complexes que es basen en aquests moviments senzills. Per exemple, per realitzar un canvi de carril, s'empren moveLeft(), moveForward() i moveRight().

5.3 Funcions implementades

A continuació es llisten totes les funcions implementades:

Interruptions		
Funció	**Descripció**	**Actualitza variables**
count()	Compta el nombre d'interrupcions que l'optointerruptor detecta durant un segons	counter
speedISR()	Calcula la velocitat a la qual circula el robot	speedAuto counter (reinicia a 0)
echointerrupt()	Calcula el temps en què el pin echo està HIGH	distance
distanceISR()	Activa el trigger durant 10 µs	-
Moviment del vehicle		
moveStop()	El robot s'atura	-
moveForward()	El robot es mou cap endavant	-
moveRight()	El robot gira a la dreta	-
moveLeft()	El robot gira a l'esquerra	-
carmove()	Moviment global del robot. Segueix línia negra.	-
changeLaneLeft()	Canvi al carril de l'esquerra	-
changeLaneRight()	Canvi al carril de la dreta	-
overtaking()	Avançament	overtakingON
Tractament de dades		
updatemode()	Canvia el mode del robot	mode
getSpeed()	Actualitza les velocitats (robot i objecte) i recull la distància en un interval d'1segon	distanceA, distanceB, speedA, speedObstacle
compareSpeed()	Compara les velocitats	overtakingON oncomingON
Funcions no usades		
moveBackwards()	El robot es mou cap endarrere	-
getTurnTime(int angle)	Determina el temps en que s'ha de produir un gir	-
oncoming()	Canvi de carril a l'esquerra quan un vehicle s'aproxima	oncomingON

Taula 9: Funcions implementades

Capítol 6: Resultats

En aquest capítol s'expliquen els resultats obtinguts un cop efectuades les diferents proves.

En el capítol 4 s'havien definit tota una sèrie de funcionalitats. No totes s'han pogut implementar, si més no, la gran majoria.

Funcionalitats	Implementació
Mobilitat del robot: cap endavant, cap endarrere, gir a la dreta, gir a l'esquerra i aturada.	✓
Seguiment de línia recta negra.	✓
Detecció d'obstacles frontals.	✓
Càlcul de distància entre el robot i l'obstacle frontal.	✓
Càlcul de velocitats: del robot i de l'obstacle.	✓
El robot no col·lideix amb l'obstacle frontal.	✓
El robot pot variar la velocitat si ho requereix.	✗
Manteniment de la distància de seguretat.	✗
Realització d'un avançament o canvi de carril.	✓
En cas de perill s'anul·la la maniobra d'avançament.	✗

Taula 10: Funcionalitats implementades

6.1 Què ha sortit bé

Mobilitat del robot

S'han definit els moviments per separat de tal manera que s'han pogut utilitzar segons les necessitats i s'han pogut crear funcions més complexes a partir d'aquests moviments bàsics.

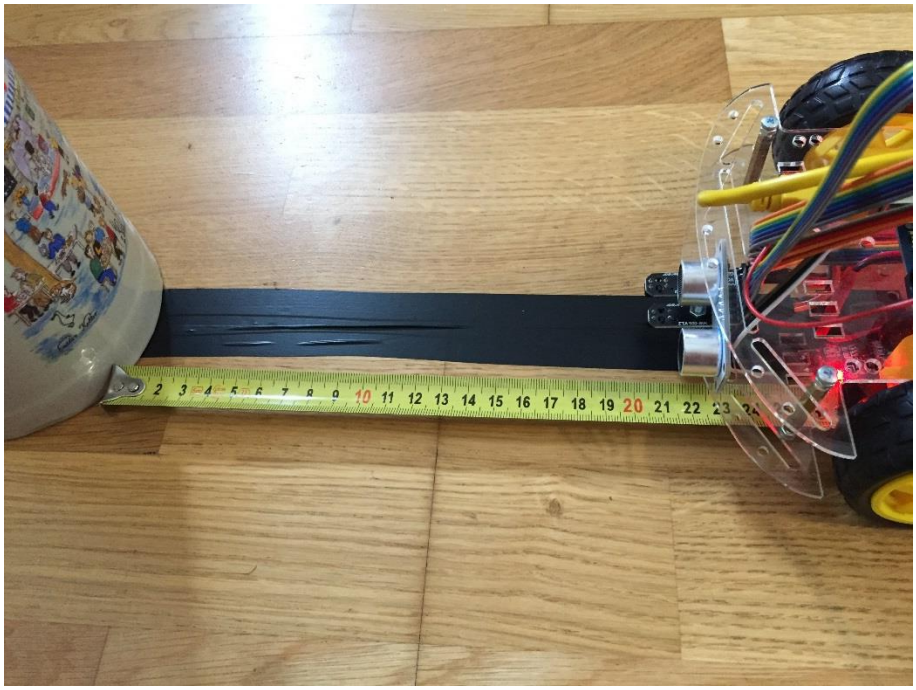
Seguiment de línia recta negra

El robot segueix la línia recta del circuit de proves i corregeix la trajectòria quan un dels sensors d'infrarojos surt de la línia. S'aconsegueix que el robot circuli en línia recta i que es desviï quan realitza un avançament.

Càlcul de distància

Les distàncies calculades amb el mòdul ultrasònic són correctes. Tenen un error d'entre 1 i mig cm, quant més a prop està el robot de l'objecte hi ha una millor precisió.

En el test he obtingut una distància de 22 cm per pantalla i la mesurada empíricament ha estat de 21.5.

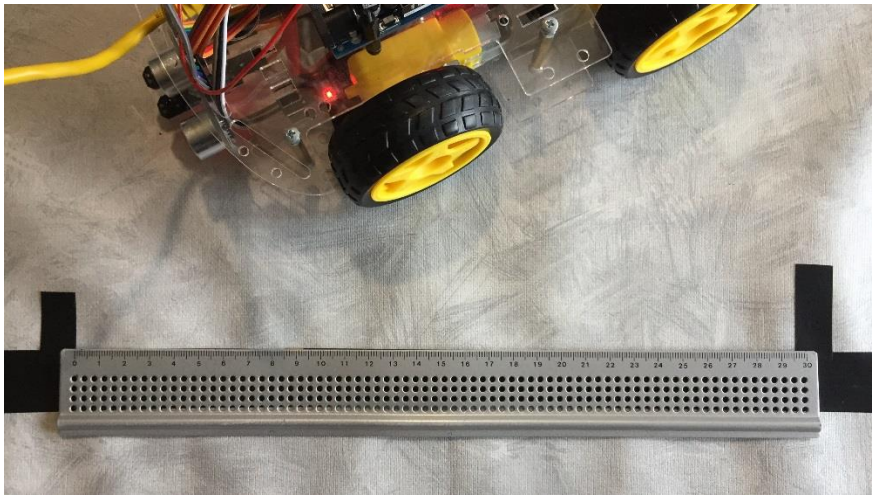


```
TEST
Distance: 22
```

Càlcul de la velocitat

Per dur a terme aquesta prova, s'ha calculat la distància recorreguda en 1 segon amb la qual cosa obtenim la velocitat real. En aquest cas les velocitats difereixen d'uns 2 a 3 cm/s. Això s'explica perquè en el moment en què es dona l'ordre d'aturar-se, el vehicle segueix movent-se. Recorre encara 2 cm de més fins a què s'atura per complert.

En el test he obtingut una velocitat de 27 cm/s per pantalla i la mesurada empíricament ha estat de 29.5 cm/s.



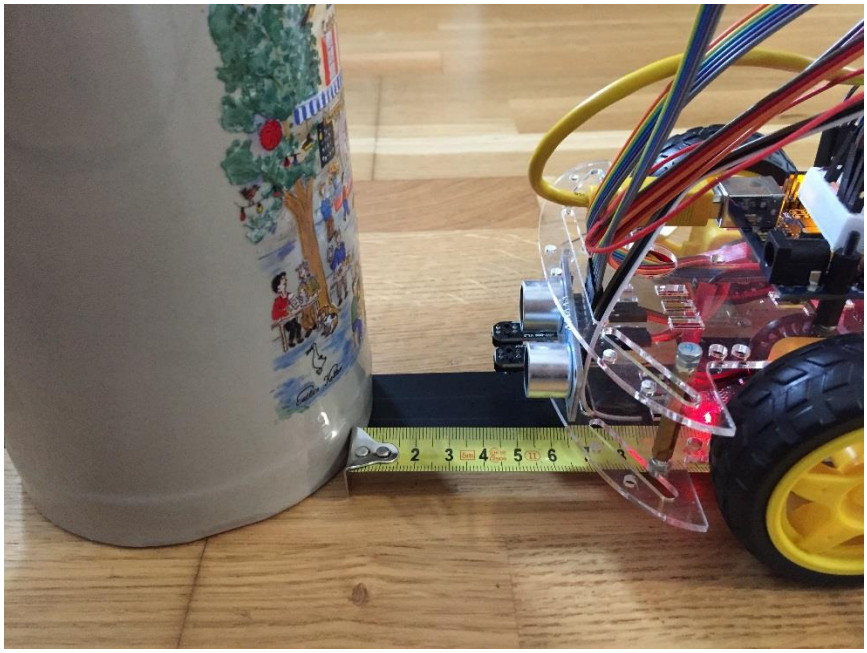
```
TEST
Speed: 27 cm/s
STOP
```

Observant aquests resultats podem determinar que la velocitat del vehicle és correcta però tindrem un problema quan el vehicle s'aturi i recorri aquests 2-3 cm de més.

El robot no col·lideix amb l'obstacle frontal

Donat que un dels objectius és aconseguir que el vehicle s'aturi a una distància mínima de 5cm, s'ha hagut de corregir la distància a partir de la qual el vehicle inicia el mode 2 - Col·lisió imminent. En comptes de definir aquesta distància a 5 cm, s'ha definit a 7 cm per allò que es comentava que el vehicle recorria 2-3 cm de més al aturar-se.

Ara s'entra en el mode 2 quan la distància a l'objecte és de 7 cm. En el moment en que s'atura el vehicle, aquesta distància hauria de ser de 5 cm com a mínim per complir els objectius.



```
TEST
Mode: 2
Distance: 5 cm:
```

Es comprova que efectivament el vehicle s'atura a 5 cm de l'objecte.

Durant les proves, es va detectar que en ocasions el vehicle impactava amb l'obstacle. Això es produïa perquè una de les funcions del mode 1 tenia una instrucció amb delay() de manera que s'entrava més tard en el mode 2. El temps de reacció del sistema superava el segon.

Un cop eliminada aquesta instrucció, el vehicle no ha col·lidit més.

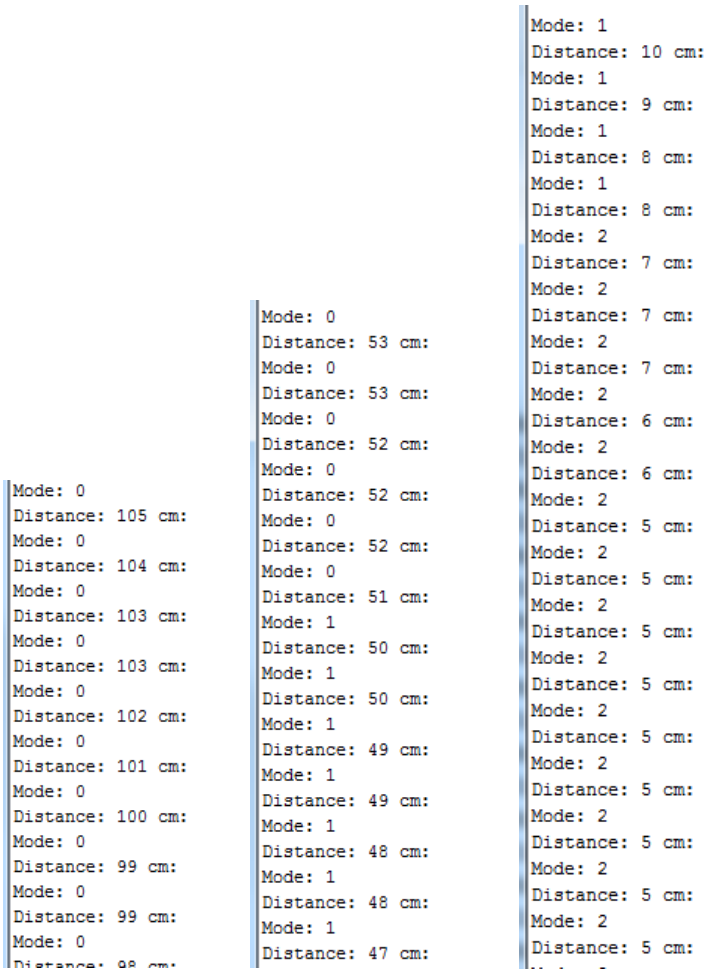
Entrada en els modes

Els modes venen determinats per la distància que hi ha entre el robot i l'obstacle.

- Mode 0: distàncies superiors a 50 cm.
- Mode 1: distàncies entre 50 i 7 cm.
- Mode 2: distàncies inferiors a 7 cm.

En aquest test, s'ha deixat que el robot circulés passant pels tres modes. En les següents captures es pot observar com el sensor detecta distàncies més enllà dels 100cm i, un altre cop, com el vehicle s'atura a 5 cm de l'obstacle.

A continuació es verifica l'entrada als modes:



6.2 Què no ha sortit bé

<u>El robot no varia la velocitat</u>

El principal problema que ha sorgit és que Arduino té les seves limitacions i controlar la velocitat és una d'elles. Quan s'assigna un valor d'entre 0 i 255 als motors, el que s'està fent és donar una modulació per amplada de polsos (PWM) i per tant modifiquem el cicle del senyal. En definitiva, el que estem modificant és la velocitat de gir de les rodes i no la velocitat lineal (al menys directament).

Experimentalment, he comprovat que el motor no té la força suficient per moure les 4 rodes amb un valor de 150. I la diferència de velocitats lineals que hi ha entre els valors 255 i 170 és de 36 cm/s i 31

cm/s respectivament. Amb la qual cosa, és una diferència molt baixa per poder treballar amb velocitats i aconseguir implementar que la funcionalitat velocitat adaptiva que volia implementar de cara a mantenir la distància de seguretat.

Implementar-la no hauria estat gaire difícil, només hauria d'haver creat una variable per a analogWrite(enA, *variable*); i analogWrite(enB, *variable*); i haver-li donat els valors desitjats en cada escenari. Per exemple, valor 0 quan el vehicle s'atura, valor 255 quan efectua un avançament, valor en funció de la velocitat de l'obstacle quan es vol mantenir la distància de seguretat...

Ara bé, el que m'hauria estat més complicat hauria estat transformar una velocitat lineal a un número entre 0 i 255.

Manteniment de la distància de seguretat

La idea és que per mantenir la distància de seguretat la velocitat s'ha d'adaptar i donat que aquesta funcionalitat no s'ha pogut implementar, tampoc s'ha pogut mantenir la distància de seguretat.

6.3 Què és millorable

Avançament i canvi de carril

El gir no és fluid i tampoc s'assimila a la realitat. En els vehicles actuals les rodes tenen eixos, en l'Arduino no. Per això el que s'ha fet és bloquejar una roda per a que el vehicle pugui girar lleugerament i així poder desviar la trajectòria.

De totes maneres, els valors que s'han usat per als girs han estat experimentals i no s'han basat en cap fonament teòric. Tenint present que s'ha de fer un moviment circular, la implementació d'aquest tampoc ha donat resultat.

6.4 Demostració

Arribat el moment de posar a proba el sistema anticol·lisió, s'ha creat un petit circuit amb dos carrils i dos obstacles. I s'ha configurat el robot per a què avanci el primer obstacle (mode 1 - overtaking) i s'aturi en el segon (mode 2 - stop).

A continuació es mostra una captura del que s'ha imprès per pantalla:

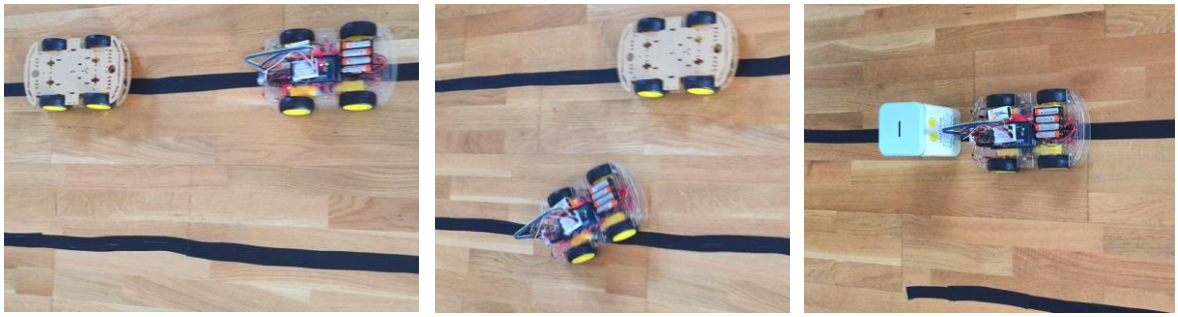
```
COM8 (Arduino/Genuino Mega or Mega 2560)

DEMO
This demo shows the actions of the
system depending on the distance to
the obstacle.
-----------------------------------
KEEP MOVING
Mode: 0
Distance: 74 cm
-----------------------------------
POSSIBLE RISK: CONTROL MODE
Mode: 1
Distance: 50 cm
OVERTAKING at distance 29 cm
 --> change to the left lane
 --> change to the right lane
END OVERTAKING
-----------------------------------
KEEP MOVING
Mode: 0
Distance: 123 cm
-----------------------------------
POSSIBLE RISK: CONTROL MODE
Mode: 1
Distance: 50 cm
 --> overtaking mode deactivated for testing purposes
-----------------------------------
ALERT: IMMINENT COLLISION
Mode: 2
Distance: 7 cm
CAR STOPPED at distance: 5 cm
-----------------------------------
```

S'observa com el robot passa per tots els modes:

- Mode 0 a distàncies superiors als 50 cm.
- Mode 1 a distàncies entre 50 i 7 cm. Overtaking a distàncies inferiors als 30 cm.
- Mode 2 a distàncies inferiors a 7 cm.

A més també es pot veure com es realitza l'avançament. Primer amb un canvi de carril a l'esquerra i seguidament amb el canvi de carril a la dreta.

Les següents imatges mostren els moments més importants del test.



Aproximació al primer obstacle, esquivar obstacle i aturada davant del segon obstacle (a 5cm de distància).

Capítol 7: Conclusions i línies de futur

7.1 Conclusions

El projecte ha presentat certes dificultats en quant al desenvolupament de l'aplicació. Els objectius s'han complert però no s'han pogut implementar totes les funcionalitats del sistema anticol·lisió. Si més no s'han establert unes bases teòriques i un disseny de com hauria de ser aquest sistema.

A nivell de projecte s'ha anat seguint la planificació però no hi ha hagut temps per realitzar aquelles tasques que s'havien definit com optatives. La part final de la implementació ha requerit una major dedicació de temps i, per tant, s'ha hagut de prioritzar.

El producte obtingut funciona perfectament. L'objectiu més important era evitar una col·lisió imminent i el robot aconsegueix detectar l'obstacle i aturar-se deixant un cert marge. Pel que fa el treball amb els sensors, aquest ha estat el més gratificant, doncs s'ha aconseguit controlar els diferents sensors i dispositius amb l'Arduino. A més, el càlcul de distàncies i velocitats s'assimilen a la realitat, amb un marge d'error mínim.

En termes generals, el grau de satisfacció amb el projecte és elevat. Si que es veritat que hi ha un parell d'aspectes millorables i que m'hagués agradat dedicar-li més temps per trobar-hi una solució però tampoc són bàsics. Arduino és una bona eina per realitzar projectes domèstics o prototips però per projectes més professionals té les seves limitacions.

7.2 Línies de futur

En aquest projecte s'ha presentat un sistema amb funcionalitats bàsiques. Altres funcionalitats que no s'han considerat imprescindibles, si que oferirien una millora qualitativa al producte. Algunes d'elles:

- Avís en angle mort.
- Control automàtic d'intermitents (hi ha un estudi del 2012 de l'IIHS, Insurance Institute for Highway Safety, que determina que aquesta característica aporta majors beneficis que per exemple el manteniment de carril).
- Avís acústic.

També considero interessant veure com es podria aplicar aquesta tecnologia en altres escenaris com per exemple quan un vehicle ve en contra direcció o què passaria si en comptes d'un sensor ultrasònic en tinguéssim més d'un.

Bibliografia

Llibres – Manuals

Brian W. Evans (2007). Arduino Notebook: A Beginner's Reference.

Articles

"Las nuevas tecnologías que ayudarán a la seguridad vial" (2018). GasExpress. http://gasexpress.es/blog/las-nuevas-tecnologias-que-ayudaran-a-la-seguridad-vial [consultat: 03/02/19]

"¿Cuáles son las medidas de seguridad proactivas de un coche?" (2018). canalMOTOR. https://www.motor.mapfre.es/consejos-practicos/seguridad-vial/medidas-de-seguridad-proactivas-de-un-coche [consultat: 03/02/19]

"Sistemas de detección en los coches para evitar accidentes" (2011). Ibanez. https://www.revistacesvimap.com/los-coches-autonomos-podran-liderar-una-reduccion-del-93-de-los-accidentes-en-2040 [consultat: 03/02/19]

"¿Cuáles son las medidas de seguridad proactivas de un coche?" (2018). canalMOTOR. https://www.motor.mapfre.es/consejos-practicos/seguridad-vial/medidas-de-seguridad-proactivas-de-un-coche [consultat: 03/02/19]

"Los coches autónomos podrán liderar una reducción del 93% de los accidentes en 2040" (2016). Cesvimap. https://www.revistacesvimap.com/los-coches-autonomos-podran-liderar-una-reduccion-del-93-de-los-accidentes-en-2040 [consultat: 05/02/19]

"Processing, Wiring y Arduino" (2017). Luis del Valle Hernández. https://programarfacil.com/blog/arduino-blog/processing-wiring-arduino/ [consultat: 17/03/19]

"Arduino DC Motor Control Tutorial – L298N | PWM | H-Bridge" (2017). Dejan Nedelkovski. https://howtomechatronics.com/tutorials/arduino/arduino-dc-motor-control-tutorial-l298n-pwm-h-bridge/ [consultat: 17/03/19]

"IR Line tracking sensor" (2017). Feiticeir. http://blog.whatgeek.com.pt/arduino/ir-tracking-sensor/ [consultat: 08/04/19]

"Medir distancias con Arduino" (2016). Victor Ventura. https://polaridad.es/medir-distancias-ultrasonidos-sr04-srf05-arduino/ [consultat: 20/04/19]

“Arduino Programming the HC-SR04 Project Interrupt Driven Software” (2014). Homediy electronics.http://homediyelectronics.com/projects/arduino/arduinoprogramminghcsr04withinterrupts/?p=4
[consultat: 13/05/2019]

“Todos estos coches se fabrican en España” (2017). ABC.
https://www.abc.es/motor/reportajes/abci-todos-estos-coches-fabrican-espana-201710082144_noticia.html
[consultat: 03/06/19]

"Advanced driver assistance systems 2018" (2018).). European Road Safety Observatory.
https://ec.europa.eu/transport/road_safety/sites/roadsafety/files/pdf/ersosynthesis2018-adas.pdf
[consultat: 03/06/19]

“Build a Robot Car with Speed Sensors” (2018). DroneBot Workshop.
https://dronebotworkshop.com/robot-car-with-speed-sensors/
[consultat: 05/06/19]

Web i blocs

Arduino: https://www.arduino.cc/
[darrera consulta: 09/06/19]

Wikipedia: https://wikipedia.org/
[darrera consulta: 09/03/19]

Luis Llamas: https://www.luisllamas.es/
[darrera consulta: 03/06/19]

Prometec: https://www.prometec.net/
[darrera consulta: 23/05/19]

Aprendiendo Arduino: https://aprendiendoarduino.wordpress.com/
[darrera consulta: 10/05/19]