UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIERÍA INDUSTRIAL VALENCIA

TRABAJO FIN DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

DESARROLLO DE UN PROTOTIPO A ESCALA DE UN VEHÍCULO PARA PRUEBAS DE LABORATORIO MEDIANTE IMPRESIÓN 3D

AUTOR: RUBÉN GARCÍA ARMERO

TUTOR: JUAN CARLOS BARAZA CALVO

COTUTOR: JOAQUÍN GRACIA MORÁN

Curso Académico: 2020-21

RESUMEN

En este proyecto se realizará el diseño y construcción de un prototipo de un vehículo con capacidades de conducción autónoma. Para la construcción del prototipo se utilizará la tecnología de fabricación mediante impresión 3D. En cuanto a la electrónica, se empleará un microcontrolador Arduino, así como una serie de sensores y actuadores determinados que le permitan recibir información sobre su entorno y actuar en consecuencia. El prototipo de vehículo será capaz de detectar obstáculos durante su trayectoria, tanto de avance como de retroceso, así como de seguir una ruta indicada con marcas sobre la vía. Se desarrollará la programación necesaria que permita comprobar el correcto funcionamiento del vehículo.

La finalidad de este proyecto es ofrecer al grupo de investigación al que pertenecen los tutores de este trabajo un vehículo sobre el cual poder desarrollar sus propios algoritmos de conducción autónoma y realizar pruebas de laboratorio.

Palabras clave: Impresión 3D, Arduino, conducción autónoma, electrónica, sensor, programación.

RESUM

En aquest projecte es realitzarà el disseny i construcció d'un prototip d'un vehicle amb capacitats de conducció autònoma. Per a la construcción del prototip s'utilitzarà la tecnología de fabricación amb impressió 3D. Pel que fa a l'electrònica, se emprarà un microcontrolador Arduino, així com una serie de sensors i actuadors determinats que li permetin rebre informació sobre el seu entorn i actuar en conseqüència. El prototip de vehicle serà capaç de detectar obstacles durant la seva trajectòria, tant en avanç com en retrocés, així com de seguir una ruta indicada amb marques sobre la via. Es desenvoluparà la programació necessària que permeti comprovar el correcte funcionament del vehicle.

La finalitat d'aquest projecte és oferir al grup d'investigació a què pertanyen els tutors d'aquest treball un vehicle sobre el qual poder desenvolupar els seus propis algoritmes de conducció autònoma i realitzar proves de laboratori.

Paraules clau: Impressió 3D, Arduino, conducció autònoma, electrònica, sensor, programació.

ABSTRACT

In this project, the design and construction of a prototype of a vehicle with autonomous driving capabilities will be carried out. For the construction of the prototype, the technology of manufacturing through 3D printing will be used. As for electronics, an Arduino microcontroller will be used, as well as a series of certain sensors and actuators that allow it to receive information about its environment and act accordingly. The prototype will be able to detect obstacles during its trajectory, both forward and backward, as well as to follow a route indicated by markings on the road. The necessary programming will be developed to check the correct operation of the vehicle.

The purpose of this project is to offer the research group to which the tutors of this project work belong a vehicle on which to develop its own autonomous driving algorithms and carry out laboratory tests.

Key words: 3D printing, Arduino, autonomous driving, electronics, sensor, programming.

ÍNDICE

DOCUMENTOS CONTENIDOS

- **Memoria**
- **Presupuesto**
- **Anexos**
- **Planos**

ÍNDICE DE LA MEMORIA

CAPÍTULO 1. INTRODUCCIÓN

1.1. GENERALIDADES

El presente documento recoge toda la información asociada al desarrollo y construcción de un prototipo a escala de un vehículo para pruebas de laboratorio mediante impresión 3D. Este proyecto ha sido ofertado por los tutores de este proyecto, investigadores en el Instituto ITACA, en el marco del desarrollo de una flota de vehículos a escala con capacidades para obtener y procesar información del entorno, con la finalidad de tener una base sobre la que desarrollar y probar programas de conducción autónoma. Para la fabricación de los componentes del vehículo se empleará la tecnología de impresión en 3D FDM (*Fused Deposition Modeling),* y para el control del vehículo se utilizará la plataforma de hardware de Arduino.

1.2. OBJETIVOS

Con este proyecto se pretende diseñar y construir un vehículo de pequeñas dimensiones, empleando las impresoras 3D del Departamento de Informática de Sistemas y Computadores, puesto que esta tecnología ofrece los menores tiempos y costes de fabricación del mercado para proyectos de prototipado o tiradas cortas.

El vehículo debe ser capaz de llevar a cabo dos funciones principales:

- Una conducción libre con sistemas de detección de obstáculos. Debe mantener un rumbo fijo, y en caso de encontrar un obstáculo ser capaz de sortearlo.
- Seguimiento de una ruta detectable por el vehículo. Debe ser capaz de detectar y seguir en tiempo real marcas en la vía (líneas que contrasten con el fondo del pavimento).

Por último, durante el diseño de los sistemas mecánicos (propulsión y dirección), se debe priorizar que tengan un funcionamiento lo más similar posible a los de un vehículo turismo convencional, con el fin de que los algoritmos de conducción autónoma que se desarrollen con el prototipo puedan ser más fácilmente aplicables a automóviles reales.

1.3. DEMANDAS DE USUARIO

Una vez adjudicado el proyecto, se llevó a cabo una reunión con los tutores de este con el fin de establecer los objetivos a alcanzar con este trabajo. Se comenzó mostrando dos versiones de otros prototipos adquiridos el año anterior, uno basado en Arduino y otro en Raspberry Pi, cuyas características no terminaban de satisfacer sus necesidades. Los principales problemas que presentan fueron analizados, y a partir de estos junto con nuevas necesidades que se expresaron durante la reunión, se elaboró la siguiente lista que compone las principales demandas de usuario:

- Debe tener un tamaño suficientemente grande para poder ser manipulado sin dificultad.
- Los componentes electrónicos deben estar a la vista y ser fácilmente accesibles. Además, deben estar suficientemente separados para ofrecer una disposición del cableado limpia, que no entorpezca las operaciones de conexión y desconexión.
- El diseño se debe hacer orientado a su posterior fabricación mediante impresión 3D.
- Se debe realizar un diseño flexible que permita futuras ampliaciones, variaciones o modificaciones.
- El vehículo debe disponer de espacio para ampliar sus prestaciones, de manera que se puedan colocar sensores adicionales en el futuro en la parte trasera y delantera.
- El montaje y desarme del vehículo debe ser lo más sencillo posible, que no requiera herramientas.

Una vez establecidos los objetivos y conociendo las demandas de usuario, el proceso de diseño debe comenzar convirtiendo esta información en las especificaciones de diseño sobre las que se apoyarán las decisiones que se tomen tanto en el momento de la representación digital como en la fabricación.

1.4. ESTRUCTURA DEL DOCUMENTO

El contenido de esta memoria a partir de este punto se divide en 5 capítulos. El primer capítulo se dedica a hacer una explicación de las tecnologías empleadas en este trabajo. En el segundo, se explica el proceso de diseño, fabricación y montaje del vehículo. El tercero servirá para presentar la programación llevada a cabo para las pruebas del funcionamiento del vehículo. El cuarto y quinto capítulos se reservan para presentar los resultados y las conclusiones obtenidas durante el desarrollo del proyecto.

Además de este documento, se presentarán más adelante el documento con los planos del vehículo, así como el presupuesto del proyecto completo y un anexo con el código desarrollado para controlar el vehículo.

CAPÍTULO 2. MARCO TECNOLÓGICO

Antes de comenzar con el proceso de diseño, se va a aprovechar este apartado para introducir brevemente algunos aspectos que es necesario asimilar para comprender en su conjunto este trabajo. En concreto se abordarán los fundamentos básicos de la tecnología de impresión 3D, haciendo hincapié en el tipo concreto que se empleó en este proyecto, así como las características fundamentales de los microcontroladores y el lenguaje Arduino.

2.1. IMPRESIÓN 3D

Está extendida la creencia de que la fabricación mediante impresión 3D es de reciente aparición, pese a que las primeras patentes con este nombre datan de mediados de los años 80. Esta creencia se debe al apogeo que experimentó durante la década de 2010 la tecnología de impresión *Fused Deposition Modeling* (en adelante FDM), tanto en el ámbito industrial como especialmente en el doméstico. Esta tecnología ha sido la utilizada en este proyecto, y en el siguiente epígrafe se explicarán sus características.

2.1.1 TECNOLOGÍA FDM

Al igual que casi todas las tecnologías de impresión 3D, la FDM construye las impresiones mediante una sucesión de capas de escaso grosor, que se van apilando hasta conformar la pieza en 3 dimensiones. La norma ISO/ASTM 52900 recoge las definiciones de 7 técnicas de impresión 3D. En el caso de la FDM, la define como un "proceso en el que el material se dispensa selectivamente a través de una boquilla u orificio" [1].

Existen diferentes variaciones en cuanto al sistema de movimiento del cabezal que alberga la boquilla. Para sintetizar, se explicará el sistema cartesiano, pues es el que emplean las impresoras Prusa i3 MK3 que fueron empleadas en el proyecto. En este modelo, el cabezal extrusor es movido por dos motores a lo largo de los ejes X y Z, mientras que el desplazamiento en el eje Y lo lleva a cabo otro motor que se encarga de mover la plataforma de impresión. Además, en el cabezal se encuentra otro motor, que se encarga de alimentar el extrusor del filamento necesario, que se encuentra almacenado en forma de bobina en la parte superior de la impresora.

Además de estos sistemas, este tipo de impresoras también está dotado de otros sistemas auxiliares que ayudan a conseguir mayor calidad en las impresiones. Entre ellos podemos destacar los siguientes, que se encuentran en el modelo utilizado:

- Ventiladores que, montados al cabezal, dirigen una corriente de aire hacia el material ya depositado para enfriarlo rápidamente. Este enfriamiento es beneficioso en algunos materiales, pero en otros es necesario desconectarlos.
- Una plataforma con control de temperatura o *cama caliente*, que permite mejorar la adherencia de las piezas a la plataforma.
- Un sistema de calibrado, que permite nivelar la plataforma y ajustar la distancia entre la boquilla del extrusor y la plataforma de manera automática, lo que permite reducir errores humanos con respecto a los sistemas de calibración manual.

Figura 1. Impresora Prusa i3 MK3 [12]

La gama de materiales que se pueden emplear en esta tecnología está limitada a polímeros termoplásticos, siendo el más popular el ácido poliláctico o PLA, aunque también se pueden utilizar el ABS, el PETG, polímeros flexibles como el TPU, o polímeros reforzados con fibra de carbono o nylon. En este proyecto se decidió emplear el PLA, puesto que presenta unas propiedades que lo hacen interesante para aplicaciones de prototipado y robótica:

- Facilidad de impresión. El PLA es el material que más fácil se imprime, puesto que no presenta problemas que se dan en otros materiales, como *warping* (despegue del material de la base de impresión), *stringing* (aparición involuntaria de filamentos de material extremadamente delgados), o *cracking* (separación entre capas por falta de adhesión).
- Economía. El precio del PLA es el segundo más económico, por detrás del ABS. El precio de mercado oscila entre los 13 y 25 €/kg, dependiendo de la calidad y del fabricante.
- Resistencia. Presenta una tensión de rotura de alrededor de 65MPa, si bien es cierto que su resistencia a fatiga y a altas temperaturas es limitada.
- Baja temperatura de fusión. Con un rango entre 190 y 220 grados, se evita un consumo excesivo de energía para calentar y enfriar el material.
- Origen vegetal. Al ser elaborado a partir de maíz, este material es más respetuoso con el medio ambiente que sus competidores.

Finalmente cabe hablar de como enviar las instrucciones de impresión a las máquinas. El lenguaje de trabajo que utilizan es el G-Code, el lenguaje por excelencia de las máquinas de control numérico. Sin embargo, para trabajar eficientemente, no se trabaja directamente programando en este lenguaje. En su lugar, se utilizan programas específicos llamados laminadores o *slicers.* Estos programas se encargan de importar archivos de modelos sólidos, generalmente de programas de diseño asistido por ordenador o *CAD*, y en su interfaz permiten ajustar diversos parámetros, dando como archivo de salida un archivo de instrucciones G-Code interpretable por la impresora.

Las impresoras de la marca Prusa cuentan con su propio programa de laminado, *PrusaSlicer*, y entre los parámetros que permite ajustar, se pueden destacar los siguientes, que son los más sencillos pero a su vez los de mayor importancia:

- Relleno. Normalmente cuando importamos un modelo sólido, este es completamente macizo. No obstante, en aplicaciones en las que el material no va a ser sometido a grandes esfuerzos mecánicos, es interesante crear un entramado con huecos en el interior de la pieza, con la finalidad de reducir el material empleado y el tiempo de impresión. Los programas permiten regular tanto la densidad del relleno, como el tipo de entramado. Cada tipo cumple una función, ya sea priorizar acortar el tiempo de impresión o mejorar las propiedades mecánicas del material, por ejemplo. En la Figura 2 se pueden observar diferentes tipos de rellenos.
- Altura de capa. Es uno de los parámetros fundamentales, puesto que repercute directamente sobre la calidad y resolución de la pieza, y el tiempo que se necesita para imprimirla. Para una boquilla estandarizada de diámetro 0.4 mm, los valores de altura de capa más frecuentemente utilizados varían entre 0.1 y 0.2 mm.
- Contorno. Se puede variar el número de contornos que se van a generar, lo que aumenta o disminuye el espesor de la pared exterior de la pieza. Asimismo, también es posible escoger cuantas capas sólidas deben tener las partes superior e inferior de la pieza.
- Uso de material de soporte. Este parámetro es fundamental en las piezas que presentan voladizos o partes que no conectan con la capa anterior. El software reconoce estas partes y coloca debajo de ellas material que lo sujete. Normalmente este material es fácil de eliminar posteriormente, aunque en ocasiones es necesario el uso de herramientas de desbastado.
- Velocidad de desplazamiento. Con este parámetro se puede optimizar el tiempo de impresión. Se pueden utilizar grandes velocidades (de 150 a 200mm/s) para el relleno dado que la calidad superficial no es relevante, mientras que en los contornos y las partes externas es preferible una velocidad menor (de 30 a 60 mm/s).

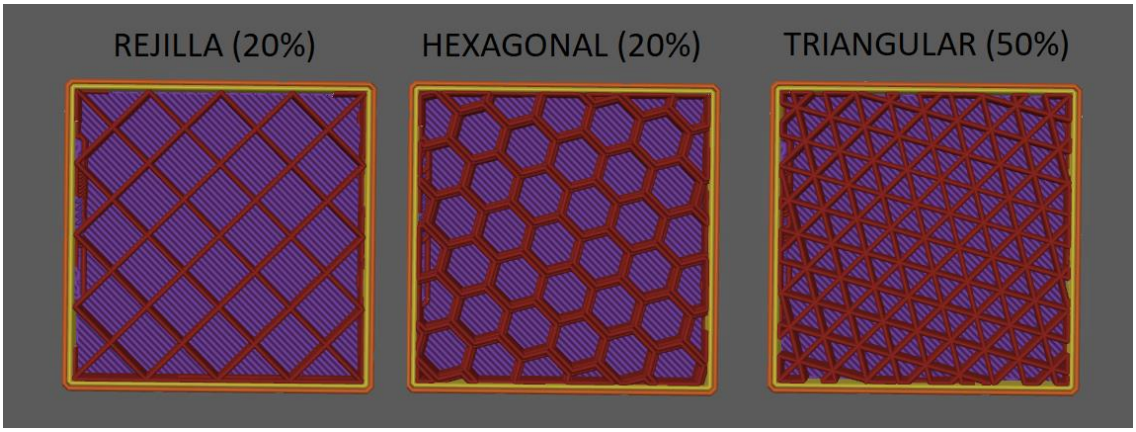


Figura 2: Distintos tipos de rellenos en PrusaSlicer

2.2. ARDUINO

Arduino es una plataforma de electrónica que engloba tanto el hardware que desarrolla la compañía como el software de código abierto que lo gobierna. Es ampliamente utilizado en proyectos que requieren de control electrónico, tanto por público entusiasta por su sencillez de uso, como por profesionales por las capacidades que ofrece.

En lo que respecta al hardware, este se compone de diferentes tipos de placas controladoras, así como de infinidad de accesorios (sensores y actuadores) que las complementan. Las placas son capaces de recibir señales de diferentes tipos de sensores y enviar señales a actuadores mediante su sistema de entradas y salidas. La relación entre las entradas y salidas se maneja a través del programa que el usuario puede enviar desde su ordenador a la memoria del microcontrolador.

En cuanto al software de los controladores, Arduino cuenta con un entorno de programación propio, el *Arduino Integrated Development Environment,* o *IDE,* que permite escribir código, compilarlo y enviarlo a cualquier controlador de la marca u otros sistemas compatibles. La programación en Arduino resulta sencilla de aprender gracias a la interfaz del IDE y a una amplia comunidad de programadores y educadores que comparten sus proyectos. Además, para programar los sensores y actuadores más comunes no suele requerirse la instalación de bibliotecas adicionales, y cualquiera que esté familiarizado con el lenguaje C no tendrá dificultad en elaborar código en Arduino.

Con todo esto, se pueden resumir las principales ventajas que ofrece Arduino con respecto a otras placas controladoras en los siguientes puntos:

- Sencillez: las placas Arduino ejecutan un único programa en bucle, por lo que, si bien es cierto que esto limita su versatilidad, lo hace idóneo para aplicaciones sencillas de control.
- Precio reducido: el precio base de la gama más accesible ronda los 20€, existiendo versiones de mayores prestaciones por el doble de este precio. También existen en el mercado placas clon compatibles con el software Arduino a mitad de precio, pero estas no son oficiales.

- Uso extendido: su popularidad ha ayudado a que se desarrolle hardware a medida. Además, al ser de código abierto, es sencillo encontrar proyectos y soluciones de programación elaboradas por otros usuarios.
- Portabilidad: su tamaño reducido y poder ser alimentado mediante pilas o baterías lo hace idóneo para proyectos donde el volumen es reducido o se necesita trasladar el equipamiento, como en aplicaciones de robótica o equipamiento médico portátil.

Figura 3. Arduino UNO [13]

2.3. ONSHAPE

Para el modelado de las piezas que se imprimen, así como para la visualización del conjunto ensamblado, se utiliza el software CAD *Onshape.* Este es un software en línea que permite el almacenado de los proyectos en la nube y además no necesita estar instalado en ningún dispositivo, puesto que al programa se accede a través de un navegador web. Esto permite trabajar desde diferentes dispositivos, incluso en ordenadores de bajas prestaciones. Además de las anteriores, cuenta con una serie de ventajas que hicieron que se seleccionara este software para este proyecto:

- Dispone de una característica llamada *Part Studio*, que permite diseñar varias piezas en el mismo archivo. Esto es muy útil para poder visualizar como van a interactuar y encajar las piezas antes de proceder con el ensamblaje.
- Permite exportar las piezas en diversos formatos, incluido el STL, un archivo ampliamente utilizado en impresión 3D que convierte la pieza en una malla de triángulos. Onshape además permite indicar la resolución de esta malla, según las necesidades de la pieza.
- Dispone de un sistema ramificado de versiones, que encapsula las diferentes iteraciones de una misma pieza en un solo archivo. Esto permite observar la evolución temporal del diseño, poder utilizar una pieza matriz que sirva de base para diversas piezas similares, o poder volver atrás a una versión anterior en caso de que se descarten las posteriores.
- Facilidad para importar desde la misma aplicación piezas de otros usuarios, así como para compartir tus propios proyectos. Existe también la opción de hacer tus proyectos visibles solo a usuarios determinados, lo que facilita el trabajo en grupo.
- Por último, cabe destacar que este software cuenta con licencias gratuitas para estudiantes y docentes con las mismas capacidades que la versión estándar.

Para la elaboración de los planos, pese a que Onshape dispone de herramientas para ello, he decidido emplear Autodesk Inventor, puesto que las plantillas normalizadas de los planos de la ETSII son más fáciles de implementar en esta herramienta.

CAPÍTULO 3. DISEÑO

3.1. ESPECIFICACIONES DE DISEÑO

En esta fase del proyecto, se trata de definir de una manera menos ambigua las características que se han especificado de manera cualitativa en la fase previa de obtención de las demandas del usuario. En este apartado se mostrarán las opciones que se barajaron durante el proceso de diseño y las que finalmente fueron adoptadas.

3.1.1. Actuadores

En primer lugar, se debe ofrecer una solución para el problema de los actuadores que propulsarán y controlarán el vehículo. Esto son los motores y los servomotores.

En el caso de los motores, en el mercado existe una gran variedad de tecnologías diferentes. En nuestro caso, era trivial que los motores debían ser de corriente continua dadas las dimensiones del vehículo y las corrientes de salida que proporcionan las placas controladoras Arduino. Sin embargo, dentro de los motores de corriente continua existen a su vez diferentes tipos dependiendo del uso al que vayan a ser destinados. La opción más sensata para un proyecto como este es emplear motores con escobillas, dado que es una opción robusta, muy económica y sencilla de controlar. Puesto que no se requiere que el vehículo alcance grandes velocidades, se descartan los motores sin escobillas, y los motores paso a paso se descartan por su tamaño, precio y excesiva complejidad. Finalmente se opta por adquirir 2 motores de escobillas con reductora, con tensión de funcionamiento de 3 a 6V, y que propulsarán al vehículo desde la parte trasera del mismo.

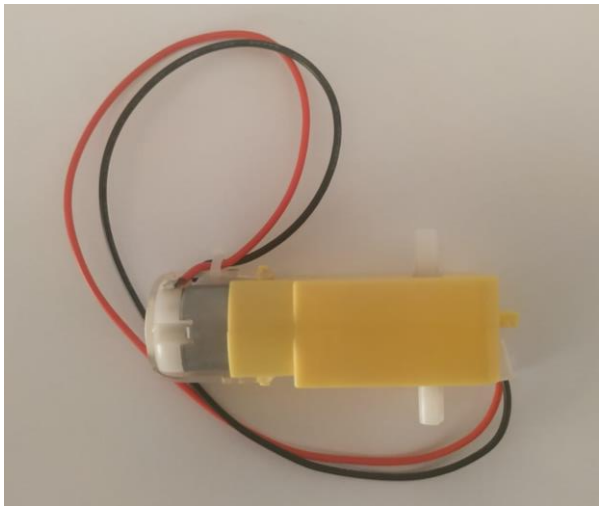


Figura 4. Motor de escobillas

Los otros actuadores que son necesarios para el control del vehículo son aquellos que se encargarán de la dirección. Para ello se barajaron dos posibilidades: emplear un servomotor que se encarga de mover el eje delantero, controlando así la dirección; o utilizar un sistema de tracción a las 4 ruedas, en las que a cada una acoplamos un motor. Al controlar cada motor de manera independiente, se puede controlar el sentido de la dirección de una manera mucho más versátil que con 2 motores y una servodirección, dotando al vehículo de menores radios de giro y por tanto mejor maniobrabilidad. No obstante, este sistema decide no implementarse, puesto que no es un sistema que se emplee en los vehículos turismo convencionales, entrando en conflicto con uno de los objetivos principales del proyecto. Por ello se opta por la primera opción, adquiriéndose 2 servomotores modelo SG90, tras descartarse el modelo MG995, de mayores prestaciones, por problemas de stock. En el siguiente apartado se explicará la función del segundo servomotor.

Figura 5. Servomotor SG90

3.1.2. Sensores

En el caso de los sensores, es necesario obtener 2 tipos de sensores, uno para cada una de las funciones principales de conducción autónoma que se quieren implementar: seguimiento de línea y detección de obstáculos.

Para la primera función se emplea un módulo que los tutores tenían de un kit de otro proyecto anterior, que cuenta con tres sensores de detección de luz infrarroja ITR20001. Su funcionamiento consiste en emitir luz infrarroja mediante una serie de emisores, y cuando esta luz incide sobre una superficie clara, se refleja y es captada por los receptores, mientras que, si incide sobre una oscura, la luz infrarroja es mayoritariamente absorbida, y el receptor no recibe una señal con tanta intensidad. Colocando en paralelo varios de estos pares de sensores y receptores se puede además determinar la posición en la que se encuentra una línea de cierto grosor que contrasta con el fondo.

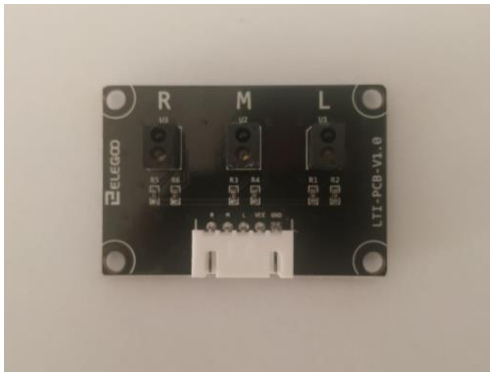


Figura 6. Módulo de seguimiento de línea

Para la función de detección de obstáculos, se decide optar por sensores de ultrasonidos. El funcionamiento es similar al anterior sensor, al tratarse de sensores con una parte emisora y otra receptora. No obstante, su funcionamiento es diferente, pues el emisor en este caso emite pulsos de sonido a alta frecuencia, que al rebotar sobre un objeto vuelven hacia el receptor. Este sensor además de detectar la presencia de un objeto delante de él es capaz de calcular la distancia a la que se encuentra, cronometrando el tiempo que tarda en recibir el pulso desde que se emitió. La distancia por tanto se calcula internamente en el sensor, mediante la fórmula:

$$d = \left(\frac{ct}{2}\right)$$

Donde:

La velocidad del sonido **c** suele ser una constante con valor 340m/s.

t es el tiempo que tarda el pulso en ir desde el emisor hasta el receptor.

d es la distancia entre el sensor y el objeto.

Se divide el producto de c y t por 2, puesto que el pulso recorre la distancia entre el sensor y el objeto 2 veces: una desde el emisor hasta el objeto, y otra desde el objeto hasta el receptor.

Con esto en mente, se decide optar por el modelo de sensor de ultrasonidos más popular del mercado, debido a su sencillez y precio económico, el HC-SR04. En una primera instancia se pensó en utilizar 4 sensores: 3 en la parte frontal y 1 en la trasera. De estos 3 sensores delanteros, uno estaría orientado enfocando al frente, y los otros dos ligeramente hacia los lados. Sin embargo, esta solución es difícil de implementar, puesto que requiere varias salidas de la placa, así como una gran complejidad a la hora de implementar el código. Es por esto por lo que se decide emplear un solo sensor, pero que sea capaz de variar su posición mediante el uso del segundo servomotor SG90 mencionado en el apartado 3.1.1.

Figura 7. Sensor de ultrasonidos HC-SR04

3.1.3. Alimentación

Para proveer al vehículo de la corriente necesaria para su funcionamiento, se barajaron 2 posibilidades: alimentación por pilas o mediante baterías. En un primer momento se optó por emplear 2 pilas recargables de ión litio de 3.7V, puesto que proporcionarían una tensión dentro del valor de funcionamiento de las placas Arduino (7-12V). Sin embargo, debido a una falta de stock estas fueron imposibles de adquirir en un plazo de tiempo adecuado, por lo que se optó por utilizar una batería adaptada a Arduino que el grupo de investigación ya poseía de un proyecto anterior. Esta cuenta con una entrada microUSB para su carga y una salida compatible con los pines de Arduino. En su interior se encuentra además dos baterías de ión litio de 3.7V, por lo que se mantiene la tensión esperada. Cuenta también con una carcasa que permite atornillarla.

Figura 8. Batería de ión litio

3.1.4. Actuadores

Una vez seleccionados los sensores y los actuadores, se puede seleccionar un controlador. Para ello hay que tener el número de entradas y salidas que necesitaremos, y la naturaleza de estas (digitales o analógicas). Para ello se enumeran los pines de cada componente a continuación:

- Sensores de ultrasonidos (2 unidades):
    - Entrada de alimentación de 5V
    - Masa GND
    - Entrada TRIG, digital
    - Salida ECHO, digital

- Sensor de seguimiento de línea:
    - Entrada de alimentación de 5V
    - Masa GND
    - 3 entradas analógicas, una para cada receptor

- Motores (2 unidades):
    - Entrada PWM
    - Masa GND

- Servomotores (2 unidades):
    - Entrada de alimentación de 5V
    - Masa GND
    - Salida de control PWM
- Batería:
    - Salida de tensión a la placa VIN
    - Masa GND

De modo que, en total, necesitaremos una placa que tenga al menos 3 pines analógicos, 4 pines PWM y otros 4 digitales. Teniendo en cuenta que la placa Arduino UNO, la más sencilla y económica del catálogo, ya dispone de 14 pines digitales, 6 de los cuales tienen capacidad para emitir una señal PWM, más otros 6 analógicos, se decide optar por este modelo. En la siguiente página muestran las características básicas de la placa, así como el diagrama de pines de esta, proporcionados por el fabricante, disponibles en la página oficial [2].

Microcontrolador	**ATMega328P**
Tensión de salida	5V
Tensión de entrada recomendada	7-12V
Entradas/Salidas digitales	14 (6 con salida PWM)
Entradas analógicas	6
Corriente de trabajo	20 mA
Memoria flash	32 KB
SRAM	2 KB
EEPROM	1 KB
Velocidad de reloj	16 MHz
Longitud	68.6 mm
Anchura	53.4 mm
Peso	25 g
Precio	20€

Tabla 1. Características de Arduino UNO [3]

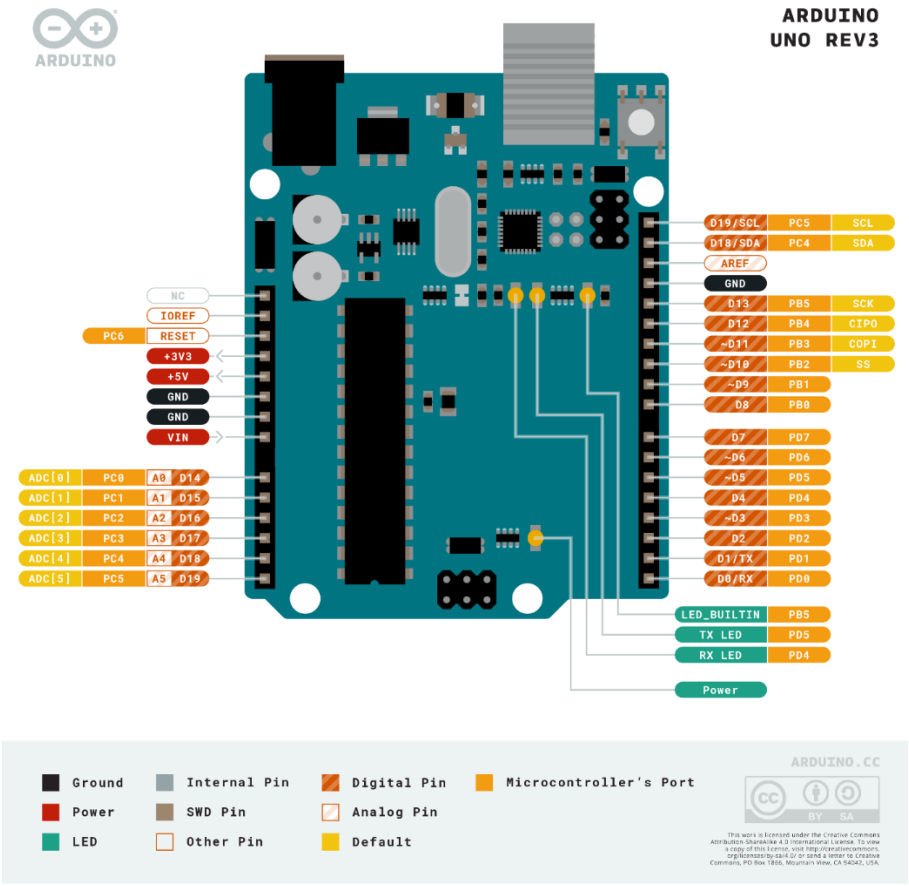


Figura 9. Diagrama de pines de Arduino UNO [14]

Se adquiere también una placa de expansión que facilita la conexión entre los sensores y actuadores con la placa. Esta incluye 4 slots de 2 pines para conectar motores, 2 slots de 4 pines para los sensores de ultrasonidos, 2 slots de 3 pines para conectar servomotores, un slot de 5 pines para el sensor de infrarrojos y un slot para la batería. Esta placa también dispone de los componentes necesarios para garantizar el correcto funcionamiento de los sensores y actuadores. Esto engloba tanto resistencias, condensadores o diodos, como el chip controlador de motores DRV8835[8]. Este chip proporciona una elevada corriente de hasta 1.5 amperios a los motores mediante un doble puente en H. Esto permite además regular tanto la velocidad como el sentido de giro de los motores. Por último, cuenta con un espacio para conectar un acelerómetro, que aunque en este proyecto no se va a emplear, puede ser interesante para el desarrollo de futuros algoritmos más avanzados.

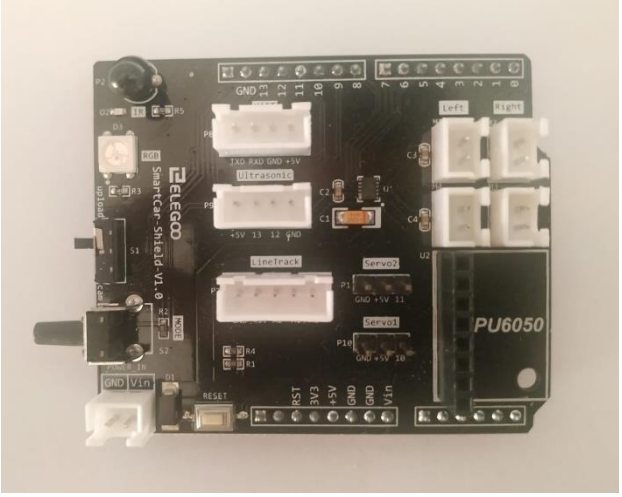


Figura 10. Placa de expansión para Arduino UNO

3.1.5. Estructura y disposición de los componentes

La forma en la que se disponen los componentes viene dada por las funciones que se exigen al vehículo. Se debe priorizar que los componentes se encuentren en un lugar donde su función se pueda desarrollar sin problemas, pero teniendo en cuenta que la relación de todos los componentes en conjunto sea óptima, para aprovechar el espacio disponible y los recursos materiales lo mejor posible.

La solución para la distribución de los componentes por la que se opta es por una división en 4 partes: dos niveles verticales (superior e inferior) y dos horizontales (delantera y trasera). De esta manera, se puede simplificar la labor de diseño, puesto que las piezas que se imprimirán harán la función de soporte y unión entre estos componentes y las partes mecánicas. La distribución por niveles será por tanto la siguiente:

- Nivel inferior delantero: dará soporte a las ruedas conducidas, así como al sistema de dirección. Además, el sensor de infrarrojos debe situarse aquí, puesto que, para su correcto funcionamiento, debe colocarse cerca del suelo. Si el sensor se encuentra demasiado lejos, la luz que emite se dispersa y puede no tener suficiente intensidad como

para ser detectada por el receptor. Además, debe situarse en la parte delantera, para poder detectar curvas y otras posibles variaciones de la línea lo antes posible.

- Nivel inferior trasero: en este caso serán las ruedas con tracción las que se deben situar en este nivel, así como los motores de escobillas que las accionarán. El sensor de ultrasonidos trasero lo situaremos también aquí, de manera que sea más fácil detectar obstáculos de baja altura durante las maniobras de marcha atrás.

- Nivel superior delantero: en este nivel se emplazará el servomotor de la dirección, que deberá tener acceso a la parte inferior para poder acoplarse al sistema de dirección. Asimismo, se colocará el sensor de ultrasonidos acoplado con el otro servomotor, que permite hacer un barrido de la zona delantera en busca de obstáculos.

- Nivel superior trasero: este nivel sustentará el microcontrolador Arduino UNO, junto con su placa de expansión, en una zona central para tratar que se encuentre lo más equidistante posible a todos los sensores y actuadores. Por último, la batería deberá situarse también en este nivel, cerca del controlador puesto que la longitud del cable de alimentación es de unos 10 centímetros.

3.1.6. Tamaño

La siguiente característica que se decide controlar es el tamaño final del vehículo. La primera aproximación que el usuario planteó para el área en planta del prototipo era de entorno a 50x25 cm. No obstante, el tamaño final tuvo que reducirse a 40x20 por una serie de motivos, que se enumeran a continuación:

- En primer lugar, se debe conocer el tamaño máximo de las piezas que pueden imprimirse en las impresoras de las que se dispone en el departamento. Se trata de los modelos I3 MK3 de Prusa, cuyo tamaño de cama es de 25x21 cm, y en la que la altura máxima de impresión es también de 21cm. Por esta razón, si se pretende construir una plataforma plana sobre la que se deben sostener los componentes electrónicos, resulta imposible realizarla tanto en 1 como en 2 piezas. Con las nuevas dimensiones es posible realizar 2 piezas que conformen la base, reduciendo la complejidad, problemas de tolerancias y limitando el uso de materiales de unión como tornillos y tuercas.
- En segundo lugar, una vez conocidos los componentes y su disposición, se observa que el espacio del que se dispone para albergarlos es excesivo, lo que implica un gasto innecesario de material.
- Por último, al disponer de cableado de tamaño normalizado (20cm), se observó que con una longitud de 50 cm no se podrían conectar los sensores a la placa con un solo cable, y conectar 2 cables seguidos implicaría un exceso de cable que complicaría las tareas de conexión y desconexión, así como posibles lugares de fallo en las zonas en las que se unen ambos cables. A estos problemas se le añade además el coste extra del empleo de mayor longitud de cable.

3.2. MODELADO EN CAD

Antes de comenzar con el diseño de las piezas, se debe tener en cuenta el proceso mediante el cual van a ser fabricadas. Por lo general, la impresión 3D tiene pocas limitaciones, en lo que respecta a la complejidad geométrica de las piezas, en comparación con procesos de fabricación tradicionales como el mecanizado o los procesos de colada. Sin embargo, hay diversos factores a tener en cuenta para optimizar el diseño orientado a impresión 3D.

En primer lugar, se debe prestar atención a la posición en la que se va a imprimir la pieza, puesto que las propiedades del material impreso no son isótropas. Las piezas impresas mediante tecnología FDM presentan una menor resistencia en el eje vertical, puesto que la menor adhesión entre las capas disminuye la resistencia a tracción en esa dirección. Por tanto, si se desea diseñar una pieza que va a someterse a cargas mecánicas no despreciables, se debe identificar el sentido de la carga y procurar orientar la pieza de manera que el eje vertical sea el menos cargado.

Otro factor a tener en cuenta son las tolerancias de fabricación de esta tecnología. La tolerancia de la tecnología FDM está estrechamente relacionada con su resolución o altura de capa. Además, es frecuente que la calibración de la impresora no sea perfecta, por lo que las piezas presenten errores de dimensionamiento del orden de las décimas de milímetros. Es por esto por lo que en el diseño enfocado a impresión 3D se suelen adoptar unos márgenes de holgura en las piezas que deben encajar unas con otras. Estos márgenes suelen estar comprendidos entre 0.1 y 0.5mm, en función del tipo de ajuste que se quiera dar. En el caso de este proyecto, se fabricó una pieza para comprobar la precisión de los agujeros impresos, y así ajustar el diámetro que debían tener en la pieza definitiva para dejar pasar con holgura suficiente los tornillos de métrica 3. La pieza, que se puede ver en la siguiente figura, presenta 7 agujeros, de 3 a 3.6mm de diámetro a intervalos de 0.1mm. Tras las pruebas, se observa que el mejor ajuste se da en el agujero de 3.4 mm, por lo que se utiliza esta medida para los agujeros por los que pasarán tornillos M3.

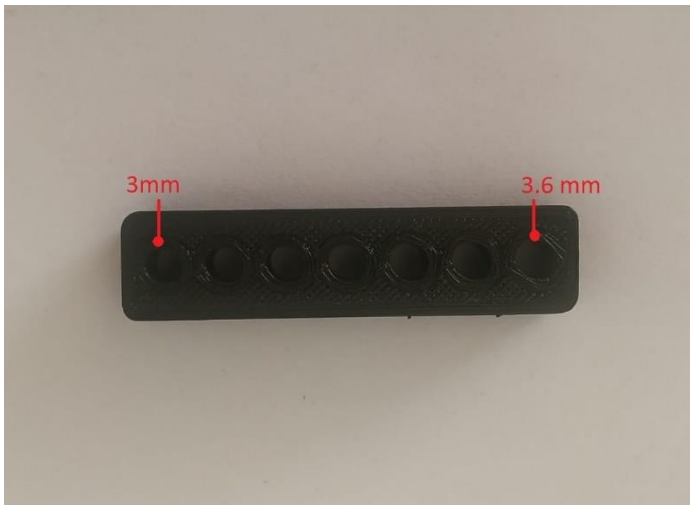


Figura 11. Probeta de tolerancias de agujeros

Por otro lado, aunque el empleo de material de soporte es en ocasiones inevitable, es una buena práctica tratar de orientar tanto el diseño como la posición de la pieza de manera que se minimice su uso. Una reducción en el material de soporte implica un menor tiempo de impresión, así como una

reducción en el gasto de material. Además, en ocasiones la retirada del material de soporte y la limpieza de la impresión puede no ser sencilla, requiriéndose emplear herramientas y mano de obra. Por estas razones es prudente considerar un diseño que evite, a menos que sea imprescindible, tener una geometría que lo haga susceptible de necesitar una gran cantidad de material de soporte.

En los siguientes apartados se desarrollará el proceso de diseño de las piezas que componen el vehículo. Primeramente, se agruparán las piezas en subconjuntos que cumplan una función determinada y a continuación se explicará cada pieza individualmente.

3.2.1 Base

Constituye la parte superior del vehículo. Este subconjunto está formado por dos piezas planas, dado que el tamaño total del subconjunto es mayor que el que puede imprimirse. Por tanto, se atornillarán entre sí junto con otras dos piezas de sujeción para evitar movimientos y flexiones.

Para el diseño de la base se parte de un boceto rectangular de las dimensiones en planta mencionadas en el apartado anterior (40x20cm). A este rectángulo le realizamos una operación de extrusión de 5mm. Se escoge este grosor puesto que en trabajos académicos anteriores se observó cualitativamente la resistencia de láminas de 3, 4 y 5 mm, y para las dimensiones de este proyecto este grosor se consideró óptimo por su escasa flexión. A continuación, gracias al repositorio de proyectos y piezas disponible en la comunidad pública de *Onshape,* se pudo obtener modelos precisos del controlador Arduino UNO[5], el microservo SG90[6] y el sensor de ultrasonidos HC-SR04[4]. Se orientaron estos modelos encima de la lámina para observar el espacio ocupado y el espacio sobrante, que posteriormente es recortado dando la forma final de la base. Hecho esto, se procede a dividir la base en dos piezas para adecuarla a las dimensiones de impresión. De este modo se divide en una pieza delantera de 160x200 mm y otra trasera de 230x200mm. Por último, se practican a las 2 piezas todos los agujeros necesarios para tornillería, cableado y para albergar el servomotor de la dirección:

- Base delantera (Plano 2)
  - 2 agujeros de diámetro 3.4mm en la parte delantera para sujetar el conjunto del servo y el sensor de ultrasonidos
  - 4 agujeros de diámetro 3.4mm para atornillar el tren delantero
  - 3 agujeros de diámetro 3.4mm para atornillar la base delantera con la trasera
  - 2 agujeros de 2mm de diámetro para atornillar el servo de dirección
  - 2 agujeros auxiliares de 3.4mm de diámetro en la parte delantera para facilitar futuras modificaciones
  - Un hueco rectangular de 23x13mm para alojar el servo de dirección
  - Un hueco cuadrado de 18mm de lado para facilitar el paso del cableado del sensor de seguimiento de líneas

- Base trasera (Plano 3)
  - 2 agujeros de diámetro 3.4mm en la parte trasera para sujetar el soporte del sensor de ultrasonidos
  - 4 agujeros de diámetro 3.4mm para atornillar el tren trasero

- 2 agujeros de diámetro 3.4mm para atornillar la batería
- 4 agujeros de diámetro 3.4mm para atornillar la placa controladora
- 3 agujeros de diámetro 3.4mm para atornillar la base delantera con la trasera
- Un hueco cuadrado de 18mm de lado para facilitar el paso del cableado del sensor de ultrasonidos
- 2 huecos cuadrados de 10mm de lado para facilitar el paso del cableado de los motores

La base está dotada además de piezas auxiliares, que van atornilladas a esta y desempeñan una función de sujeción, ya sea para unir las dos partes de la base, como para facilitar la colocación de los sensores.

- Láminas de sujeción (Plano 4)

Para unir y asegurar ambas partes de la base se diseñan dos tipos de láminas rectangulares. La primera, que se situará en la parte inferior de la base, tiene unas dimensiones de 200x50mm. Esta lámina cuenta con 6 agujeros de 3.4mm de diámetro, dispuestos de manera que coinciden con los practicados a las bases. Esta lámina es la que soporta mayores esfuerzos de flexión, por ello es de mayores dimensiones que las que se disponen en la parte superior. En este caso se diseñan 3 láminas de 40x15mm, cada una con dos agujeros de 3.4mm de diámetro, cada uno de los cuales se atornillará a una de las 2 partes de la base.

- Soporte del sensor de ultrasonidos (Plano 13)

Para la parte trasera también se diseña una pieza auxiliar para sujetar y orientar correctamente el sensor HC-SR04. Esta fue diseñada partiendo de una extrusión rectangular de 28x50x4mm. Se tomaron medidas de un modelo CAD de este sensor, disponible en el repositorio de archivos 3D GrabCad [4], para obtener las dimensiones y posición de los agujeros de la placa del sensor, así como del emisor y receptor de ultrasonidos, y otros componentes que sobresalen del plano. Se recortó la lámina conforme a las medidas tomadas, y se aplicó una corrección en el tamaño de los agujeros y los recortes, puesto que en la primera iteración de la pieza el ajuste era demasiado forzado. De modo que los recortes practicados para alojar el emisor y el receptor de ultrasonidos son de un diámetro 1mm mayor que las medidas originales, mientras que los agujeros para la tornillería son de 2.4mm. Por último, se añadieron dos pestañas que albergan 2 agujeros de 3.4mm de diámetro para poder atornillar este soporte a la parte trasera de la base, que cuenta con idénticos agujeros.

- Conjunto servomotor/soporte del sensor de ultrasonidos (Plano 15)

Para el diseño de esta pieza se recurre al anterior, pero practicándole una serie de modificaciones. En primer lugar, se cambia el diseño del soporte del sensor HC-SR04 de manera que la pieza impresa sirva como un marco al que atornillarlo. Posteriormente, se reemplazan las pestañas por una extensión de la lámina de 40x32mm, perpendicular a la lámina original. Aquí se realiza una perforación rectangular de 12x22mm, que corresponden a las medidas del servomotor. Estas medidas se amplían después a 12.2x22.5mm para facilitar un montaje más holgado, puesto que el servomotor se ajustará a la pieza mediante los dos tirafondos de métrica 2 que vienen con él. Se practican, además de los dos agujeros de 2.2mm para este ajuste, otros dos agujeros de 3.4mm de diámetro en las esquinas redondeadas de

la pieza, para poder dar soporte a futuras posibles ampliaciones. Por último, se diseñan dos fundas para los brazos del servomotor (Plano 17), que se encajan a presión, y que cuentan con un agujero de 3.4mm cada una, lo que permite anclar el conjunto a la base.

El servo de la dirección tan solo requiere como pieza auxiliar otra funda para el brazo, que le permitirá alojar un tornillo de métrica 3 para accionar la dirección como se verá en el siguiente apartado. Para el montaje del servo a la base no se necesita diseñar piezas auxiliares, se atornilla directamente a esta.

3.2.2 Tren delantero

Como ya se ha mencionado en apartados anteriores, este subconjunto (plano 8) constituye el soporte de las ruedas delanteras, así como el mecanismo de dirección y el sensor de seguimiento de líneas. Se sitúa debajo de la base del vehículo, a la que sujeta mediante 4 barras separadoras de aluminio de 40mm de longitud. Estos separadores están perforados con rosca métrica de 3mm, lo que permite atornillar por ambos extremos de la barra por un lado la base y al otro el soporte de las ruedas.

Con respecto a las ruedas, se debe mencionar que, en un primer momento se barajó la posibilidad de hacerlas impresas en 3D. Sin embargo, esta idea se descartó por una serie de motivos. El primero es que, para conseguir un mínimo de tracción se requiere de unos neumáticos de un material flexible y adherente. Se pensó en utilizar TPU o poliuretano termoplástico para ello, pues es un material que se comercializa en formato adecuado para la impresión en 3D, y cumple con las propiedades que se requieren. No obstante, al no disponer el departamento de este material, se optó por adquirir ruedas estandarizadas compatibles con los motores de escobillas (Plano 7). Al ser un modelo muy extendido se dispone de su modelo en 3D [7], lo que facilita la tarea del diseño de las piezas que deben unirse a las ruedas, puesto que el reducido tamaño de algunas partes de la rueda impide la toma de medidas.

Para conectar las ruedas al conjunto del vehículo, se diseñó un sistema de sujeción basado en el sistema de doble horquilla, pero de una manera mucho más simplificada, puesto que para esta aplicación no hay grandes esfuerzos mecánicos y no es necesario un sistema de amortiguación. La solución adoptada pasa por tanto por el diseño de las piezas que se explicarán más adelante.

- Soporte de la base delantera (Plano 9)

Es la pieza sobre la que se sustenta la base superior, como se explicó en la introducción de este apartado. La pieza parte de una lámina de 5mm de espesor de tamaño 100x100mm, sobre la que se practican los 4 agujeros M3 que permitirán atornillar los soportes hexagonales de 40mm. A esta lámina se le añaden dos salientes a los laterales con dos agujeros separados 160mm entre sí, que sujetarán los soportes de las ruedas. Se diseñan también 4 agujeros de 3.4mm en la parte delantera de esta pieza para asegurar el sensor de seguimiento de línea, con las medidas tomadas del propio sensor. Por último, se practica una perforación rectangular de 20x15mm para pasar el cableado del sensor hacia el controlador.

- Soporte de las ruedas (Plano 10)

Esta pieza proporciona 3 funciones mecánicas fundamentales: dar soporte al eje de la rueda, guiar el eje de dirección y anclar las ruedas delanteras al soporte. Se parte de una pieza rectangular de 40x31 mm, que se extruye hasta un grosor de 3mm. Se practica un agujero central de 13.2mm, que permite

un ajuste forzado a mano de los rodamientos F695ZZ adquiridos para este proyecto. Se diseñan, en el extremo de la pieza, dos brazos con 2 agujeros de 3.4mm de grosor, separado entre sí verticalmente 5.2mm. Estos brazos servirán para guiar el eje de dirección, que se unirá a estos brazos mediante un tornillo. Además de estos dos brazos, se diseñan otros dos en la parte central inferior de la pieza, con idénticos agujeros. En este caso estos brazos se unirán al soporte de la base. De esta pieza se necesitan 2 unidades, una para cada rueda. Sin embargo, las piezas no son idénticas, pero presentan una simetría especular, lo que permite el modelado de la segunda pieza a partir de la primera de manera casi inmediata con la herramienta de software CAD.

- Eje de la rueda (Plano 11)

Esta pieza permite la unión de la rueda con el rodamiento. Como las ruedas adquiridas están diseñadas para emplearse con los motores de escobillas, el hueco que presentan para introducir el eje no es de sección circular, sino que tiene dos chaflanes paralelos. De modo que el eje se diseña en dos tramos: un tramo de 18mm de longitud con el perfil del eje del motor de escobillas, y un tramo de 4.2mm de sección circular de 5mm de diámetro, para encajar con el rodamiento. A este último tramo se le practica un orificio de 3mm de diámetro, para montar a presión una tapa, que impedirá que el eje se desplace a lo largo de su longitud. De estas piezas también serán necesarias dos unidades, en este caso idénticas.

- Eje de dirección (Plano 12)

Esta pieza permite convertir el movimiento semicircular del brazo del servomotor en un movimiento rectilíneo. Este movimiento se aplica al soporte de las ruedas a una distancia del eje de la rueda, por lo que se volverá a convertir en un movimiento de giro que en este caso permite a las ruedas rotar sobre su eje. La pieza en sí es una lámina alargada de 175x15mm, de 5 mm de espesor. En sus extremos cuenta con dos agujeros de 3.4mm de diámetro, que permiten la unión con el soporte de las ruedas, como ya se explicó anteriormente. En la parte central presenta una extrusión de 13mm con forma de herradura, a la cual se acoplará el tornillo del servomotor.

3.2.3. Tren trasero

Este último subconjunto (plano 5), dedicado a la propulsión del vehículo, está formado por los motores de escobillas, con sus respectivas ruedas y el soporte del conjunto, que es la pieza diseñada (Plano 6). Esta pieza parte de una pieza rectangular de 200x100mm, con un grosor de 5mm. Sobre esta base se sitúan 4 agujeros de 3.4mm, sobre los que se colocarán otros 4 separadores de 40mm, que sujetarán la base trasera. A continuación, se extruyen en los extremos laterales dos prismas de 25x20x4mm; y a 19mm de estas, que corresponde a la anchura de los motores, se replica esta extrusión. Estas cuatro partes salientes cuentan con agujeros de 4mm alineados con los que presentan los motores para su sujeción mediante tornillería. Además, cuentan con una abertura en forma de arco, puesto que los motores presentan salientes cilíndricos en su superficie. Por último, se recorta material de la zona central, puesto que se observa un exceso de espacio no útil. Se barajó la posibilidad de hacer esta pieza en dos, eliminando completamente la zona central. Sin embargo, esto podría ocasionar que, al no haya nada que conecte las ruedas entres sí, estas podrían flectar ligeramente hacia afuera, en el eje vertical, de manera que se decide mantener una parte del material en la zona central.

Todas las esquinas, a excepción de las interiores de los huecos en los que se montan los servomotores, se han redondeado, estando los radios de acuerdo indicados en sus respectivos planos.

3.3. IMPRESIÓN

En este apartado se exponen los parámetros de impresión utilizados para la fabricación de las piezas. El material utilizado para ello, como se ha comentado anteriormente, es el PLA, de la marca RS PRO, puesto que la UPV tiene un contrato centralizado con RS Amidata para la compra de filamento para impresión. Se ha empleado el mismo filamento en dos colores, gris y negro, puesto que se comenzó imprimiendo las piezas en el modelo en negro, pero se terminó agotando. La boquilla empleada para la impresión de todas las piezas ha sido la estándar de 0.4mm de diámetro. La temperatura del extrusor y de la cama también ha sido común a todas las piezas, siendo de 215ºC y 60ºC respectivamente. La configuración de velocidad de movimientos se ha mantenido por defecto la ofrecida por PrusaSlicer.

A continuación, se presenta una tabla que sintetiza la configuración de impresión que se adoptó para cada pieza impresa. En esta tabla también se incluyen el tiempo de impresión y el material necesario para cada pieza, que será útil más adelante en el documento del presupuesto para establecer las mediciones.

Nombre de la pieza	Unidades	Altura de capa (mm)	Densidad de relleno (%)	Material de soporte	Tiempo de impresión (minutos)	Material utilizado (g)
Base delantera	1	0,2	20	No	211	79,23
Base trasera	1	0,2	20	No	338	133,98
Pletina superior	3	0,2	20	No	8	1,68
Pletina inferior	1	0,2	20	No	80	26,82
Soporte de los motores	1	0,2	20	Sí	220	61,7
Soporte de la base delantera	1	0,2	20	No	112	38,27
Soporte de las ruedas	2	0,2	20	Sí	44	5,34
Eje de las ruedas	2	0,1	100	Sí	6	0,51
Eje de dirección	1	0,2	20	No	47	10,1
Soporte del sensor de ultrasonidos	1	0,15	50	Sí	46	5,48
Soporte del sensor de ultrasonidos y servo	1	0,15	50	Sí	68	7,55
Funda del servo	3	0,1	100	Sí	5	0,33
					1261	380.86

Tabla 2. Parámetros de impresión

Cabe mencionar que el tiempo real es superior al que se ve en la tabla, que es el que calcula PrusaSlicer. Para empezar, al tiempo hay que añadirle el trabajo de preparación de la impresión. Esto engloba tanto el tiempo que tarda el operador en conectar el equipo, preparar la superficie de la cama con adhesivo o laca para evitar que las piezas se desprendan, arruinando la impresión; como el tiempo que tarda la impresora en alcanzar la temperatura de trabajo. Asimismo, se debe contemplar el tiempo de procesado después de la impresión. Esto incluye el retirado de las piezas, la limpieza de la superficie de impresión y el retirado del material de soporte. Este último no es trivial puesto que en ocasiones se requieren de herramientas de desbastado o alicates.

Por último, hay que contemplar el gasto que supone, tanto de tiempo como de material, las impresiones fallidas, piezas que se puedan romper durante las pruebas, o piezas cuyas tolerancias dimensionales reales difieren de las diseñadas. Durante la fabricación de las piezas de este trabajo, se produjo la rotura del eje de la rueda, puesto que se imprimió en una orientación vertical, lo cual hizo a la pieza fallar por delaminación en su eje longitudinal. En la Figura 12 se puede observar la fractura, que ocurre por separación de las capas tras un ligero esfuerzo de flexión. Para evitar esto se repitió la impresión, esta vez girando la posición de impresión de la pieza 90 grados, y aumentando la densidad del relleno hasta el 100%. También se dieron problemas de ajuste en dos piezas: el soporte de las ruedas y la funda del servo. En la primera pieza, el problema de ajuste se produjo en el agujero central, que debe albergar con un ajuste forzado a mano el rodamiento F695ZZ. Se diseñó el agujero de la misma dimensión que el diámetro externo del rodamiento, para que, con un leve trabajo de desbastado, pudiese ajustarse sin problemas. Sin embargo, en una de las dos piezas se retiró demasiado material, por lo que el rodamiento se salía de su posición con el vehículo en funcionamiento, por lo que hubo que volver a fabricar la pieza. Con la funda del servomotor, debido a las pequeñas dimensiones de la pieza, el ajuste resultó complicado de llevar a cabo dada la limitada precisión de la impresora, por lo que llevó 4 iteraciones de la pieza hasta dar con el tamaño que ajustaba correctamente.

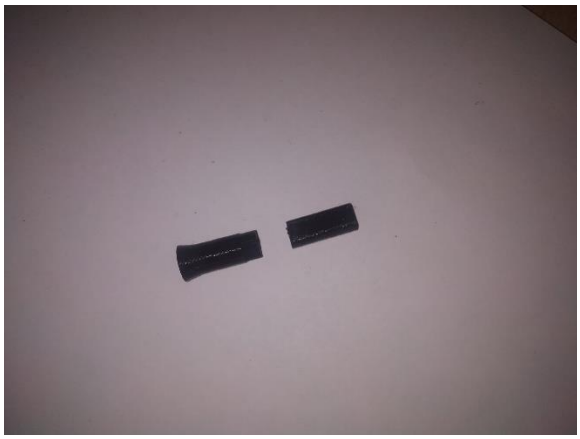


Figura 12: Rotura del eje de las ruedas

3.4. MONTAJE

En este apartado se describe el proceso de montaje. Se ha estructurado de manera que el montaje resulte sencillo y rápido, ya que una alteración del orden de las operaciones de montaje puede entorpecer operaciones posteriores.

Antes de comenzar con el proceso de montaje en sí, en la siguiente página se incluye una lista de las piezas del vehículo. Se incluyen también en la tabla las unidades necesarias de cada pieza, así como la categoría a la que pertenece: pieza impresa, tornillería o electrónica.

	Unidades	Categoría
Base delantera	1	Pieza impresa
Base trasera	1	Pieza impresa
Pletina superior	3	Pieza impresa
Pletina inferior	1	Pieza impresa
Soporte de los motores	1	Pieza impresa
Soporte de la base delantera	1	Pieza impresa
Soporte de las ruedas	2	Pieza impresa
Eje de las ruedas	2	Pieza impresa
Eje de dirección	1	Pieza impresa
Soporte del sensor de ultrasonidos	1	Pieza impresa
Soporte del sensor de ultrasonidos y servo	1	Pieza impresa
Funda del servo	3	Pieza impresa
Arduino UNO	1	Electrónica
Placa de expansión	1	Electrónica
Sensor HC-SR04	2	Electrónica
Sensor de seguimiento de línea	1	Electrónica
Batería	1	Electrónica
Servomotor SG90	2	Electrónica
Motor de escobillas	2	Electrónica
Cable puente de 20cm	13	Electrónica
Tornillo M3x30	2	Tornillería
Tornillo M3x16	10	Tornillería
Tornillo M3x14	8	Tornillería
Tornillo M3x12	5	Tornillería
Tornillo M3x8	16	Tornillería
Tornillo M2x8	4	Tornillería
Tornillo M1.6x8	8	Tornillería
Tuerca fina M3	41	Tornillería
Tuerca fina M2	4	Tornillería
Tuerca fina M1.6	8	Tornillería
Rodamiento F695ZZ	2	Tornillería
Espaciadores de aluminio M3x40	8	Tornillería
Separadores de plástico M3x3	8	Miscelánea
Rueda para motor de escobillas	4	Miscelánea

Tabla 3: Lista de piezas

Comenzamos el montaje con el tren trasero (Figura 13Montaje del tren trasero). En primer lugar, se colocan los motores de escobillas entre las pestañas salientes del soporte de los motores, hasta hacer coincidir el agujero de los motores con el de las pestañas. Se pasan los tornillos M3x30 por los agujeros, dejando la cabeza del tornillo apuntando hacia el exterior de la pieza, y se colocan sus respectivas tuercas. A continuación, se hacen pasar por los agujeros restantes 4 tornillos M3x8, y se roscan 4 separadores de aluminio a dichos tornillos. Por último, se insertan a presión las ruedas en los ejes de los motores.

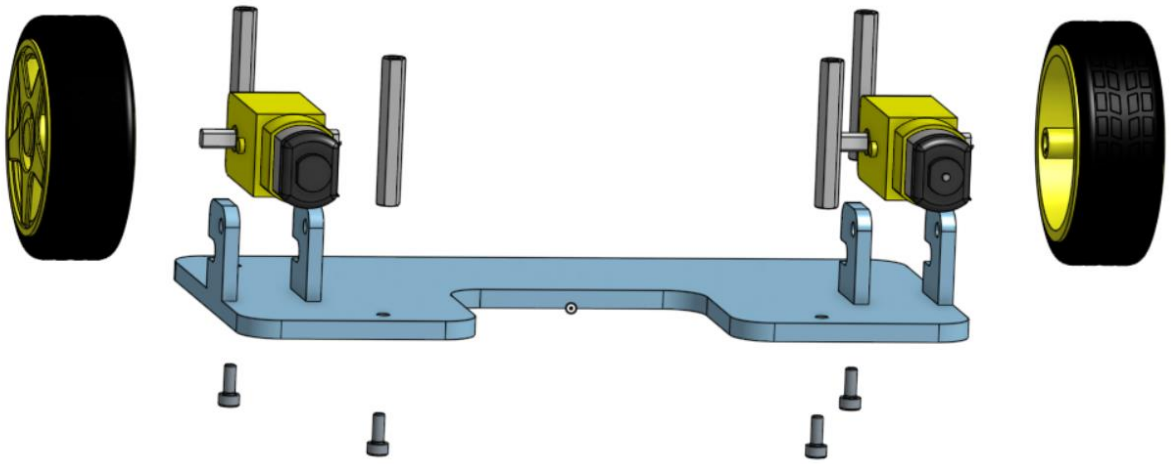


Figura 13: Montaje del tren trasero

Se continúa con el montaje del tren delantero. Primeramente, se debe atornillar el sensor de seguimiento de línea al soporte de la base delantera. Para ello, se colocan 4 separadores de plástico en los agujeros de la parte delantera del soporte, y a continuación se atornilla el sensor con 4 tornillos M3x14. Se colocan los separadores de aluminio del mismo modo que con el tren trasero, en los agujeros de las esquinas de la pieza.

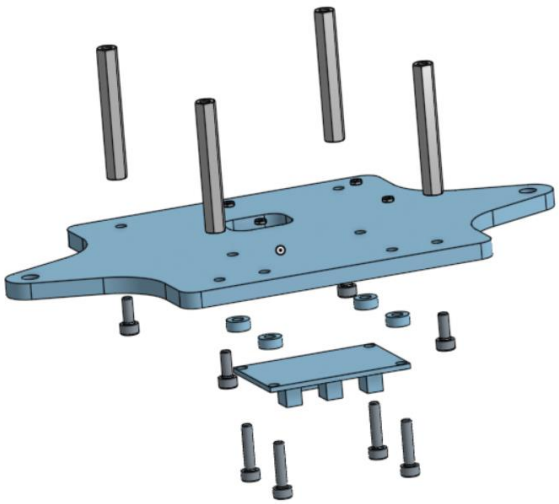


Figura 14: Montaje del tren delantero 1

Se prosigue con el montaje de los soportes de las ruedas (Figura 15). Se comienza colocando a presión los rodamientos en el agujero central de los soportes de las ruedas. Después se debe introducir la parte más alargada del eje de las ruedas en el hueco del rodamiento, hasta que haga tope, y a continuación colocar a presión la tapa del eje, que evita que se desplace. Hecho esto se pueden incorporar las ruedas al conjunto, que se acoplan también a presión con la parte del eje de sección plana.

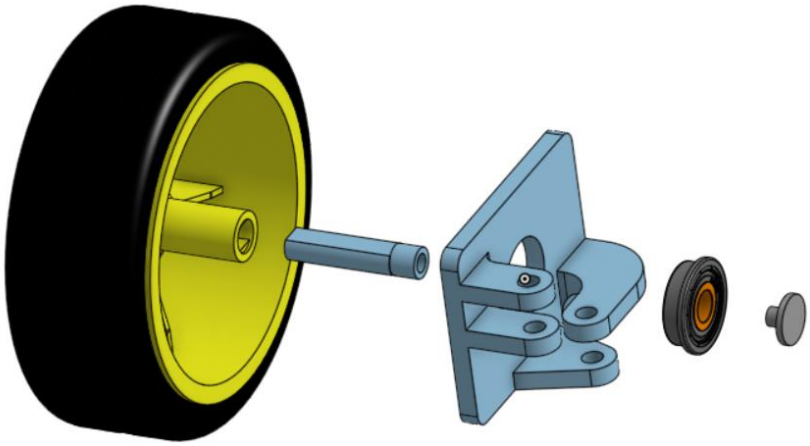


Figura 15: Montaje del soporte de las ruedas

Tras esto, se pueden atornillar los soportes de las ruedas al soporte de la base delantera, atornillando las pestañas de mayores dimensiones del soporte de las ruedas en los agujeros restantes de la base delantera con dos tornillos M3x16, y sus respectivas tuercas. Se continúa atornillando el eje de dirección a los agujeros restantes de los soportes de las ruedas, con otros dos tornillos M3x16 y otras dos tuercas. Es necesario que tanto estas, como las tuercas que aseguran los soportes de las ruedas al soporte de la base delantera no se aprieten demasiado, pues impediría el giro de las ruedas.

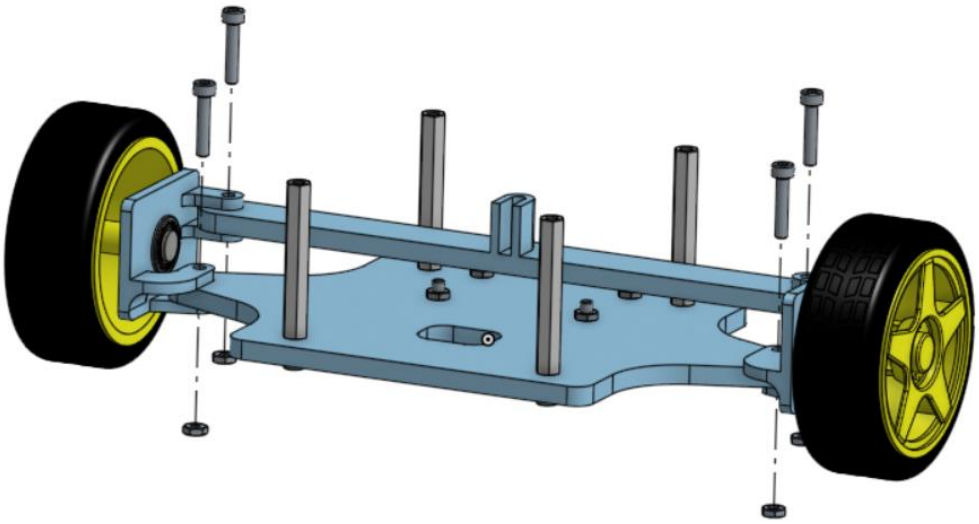


Figura 16. Montaje del tren delantero 2

Para el montaje de la base, se debe atornillar en primer lugar la base delantera y trasera. Para ello, se deben colocar ambas piezas en la orientación de la figura, puesto que no son simétricas. Se debe prestar atención que estén alineados el hueco para cables (1) y los agujeros del controlador Arduino (2).

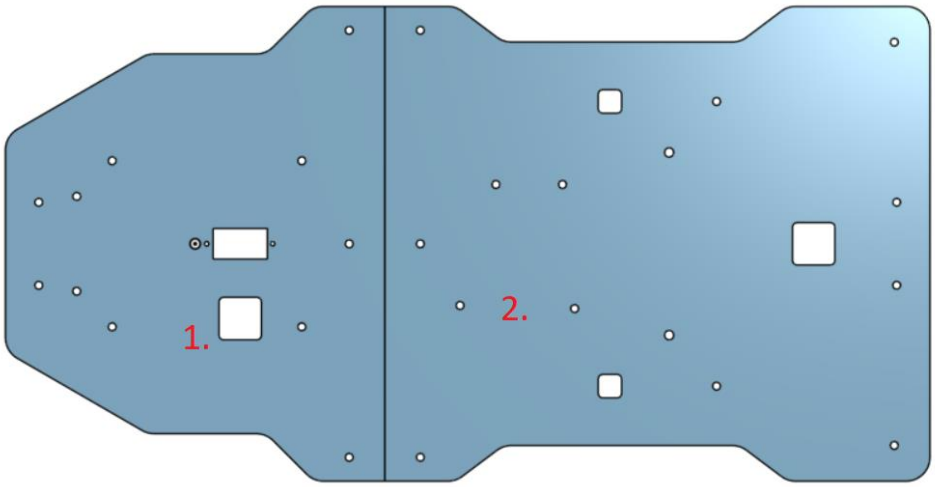


Figura 17: Alineación correcta de la base

A continuación, se debe colocar la pletina inferior, que queda en la parte inferior de la vista de la figura. Se pasan los 6 tornillos M3x16 por la pletina y ambas bases, y a continuación se colocan las 3 pletinas superiores sobre estos, y con 6 tuercas se termina de unir la base. Posteriormente, se montan sobre la base trasera el controlador Arduino y la batería. El controlador se sitúa en la parte más cercana a las pletinas, requiriéndose 4 separadores de plástico y 4 tornillos M3x14. Estos deben colocarse con la cabeza mirando hacia arriba, puesto que, en caso contrario, las tuercas se situarían en contacto con la placa Arduino, y debido a su tamaño no sería posible su montaje. La placa de expansión se encaja

montando sus pines macho sobre los pines hembra de la placa Arduino. La batería se atornilla a en sus agujeros con 2 tornillos de M3x12.

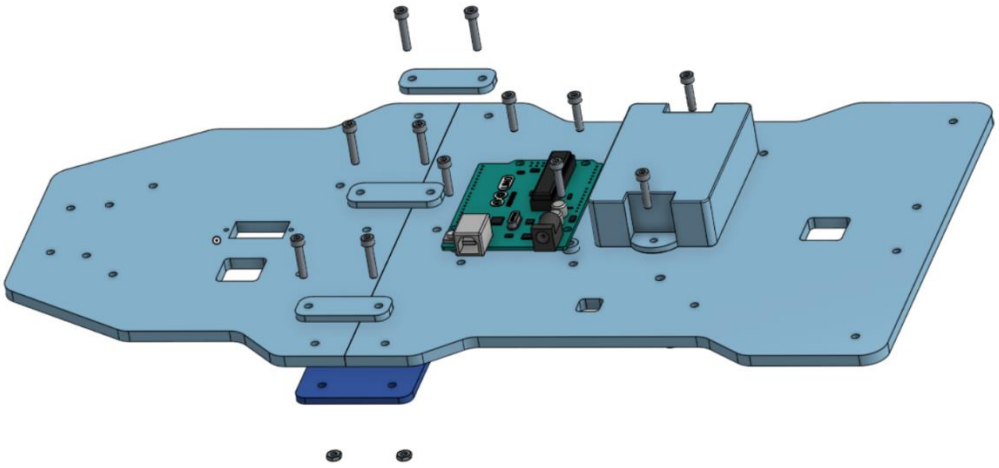


Figura 18: Montaje de la base

Para el correcto montaje del sensor HC-SR04, este debe atornillarse al soporte de sensor de ultrasonidos (plano 13) mediante 4 de los tornillos y tuercas M1.6. Se recomienda conectar los 4 cables puente al sensor antes de atornillarlo a la base trasera, puesto que será más fácil de manipular antes de estar anclado al conjunto. Finalmente, el soporte con el sensor se atornilla boca abajo en los dos agujeros centrales de la parte trasera de la base con 2 tornillos M2x12.

El servo de dirección se debe montar en el hueco rectangular de la base, comprobando que el brazo apunte hacia la parte delantera del vehículo. Posteriormente se atornilla con los tornillos M2 que incluye el propio servo. La funda del servo se encaja a presión en el brazo del servo, pero antes se debe introducir el tornillo M3x12 de forma que la cabeza quede apuntando hacia la base, y la tuerca hacia abajo.

Hecho esto ya se puede incorporar a la base ambos trenes. Para acoplar correctamente el tren delantero, hay que asegurarse que el sensor de seguimiento de línea se encuentra enfocando en el sentido de la marcha. Después, se debe colocar el tornillo que es soportado por la funda del servo en el hueco central con forma de herradura del eje de dirección. Hecho esto se puede atornillar la base a los separadores de aluminio con 4 tornillos M3x8. El tren trasero se atornilla a la base trasera del mismo modo.

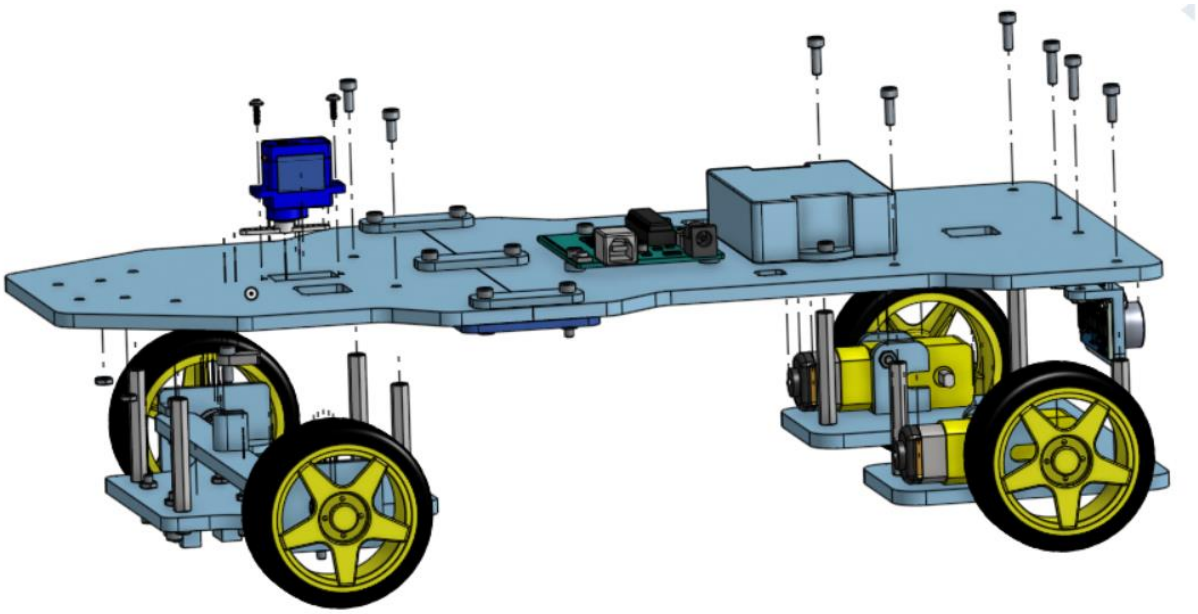


Figura 19: Montaje del sensor trasero, servodirección y unión de los trenes

Por último, queda el montaje del conjunto servo-sensor ultrasónico. El primer paso es colocar el sensor de ultrasonidos en su hueco, por la parte interior, dejando las pestañas de conexión en la parte superior. Este se atornilla con 4 tornillos y tuercas de M1.6. Después, el servomotor se introduce en su hueco rectangular, de manera que el eje del servo quede centrado con el eje de simetría del soporte, como se indica en la Figura 20. Hecho esto, se puede atornillar el servo al soporte con sus 2 tornillos M2. Finalmente, queda hacer pasar dos tornillos M3x12 por las 2 fundas del servo restantes y colocarlos en la segunda fila de agujeros de la base delantera. Finalmente, se introduce el brazo del servo en ambas fundas, y se procede a colocar las tuercas de M3, que deben quedar en la parte inferior de la base.

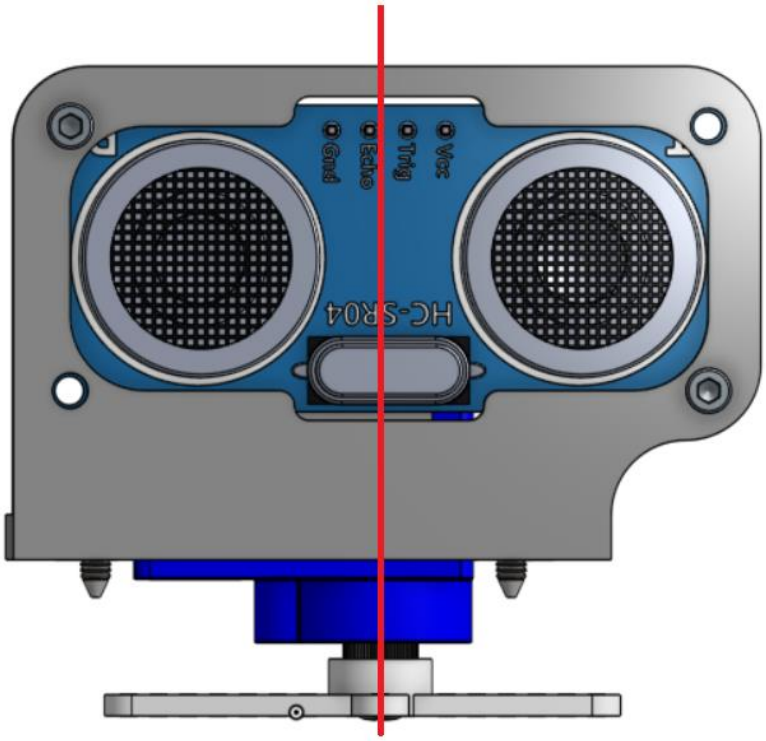


Figura 20: Alineación del servo con el soporte

Con este último paso concluye el montaje del vehículo. Siguiendo estos pasos, el montaje del vehículo completo, incluyendo las conexiones del cableado, llevó un tiempo aproximado de 20 minutos. En la Figura 21 se puede ver una imagen en perspectiva del vehículo real ya montado, y en la Figura 23 la vista en planta.

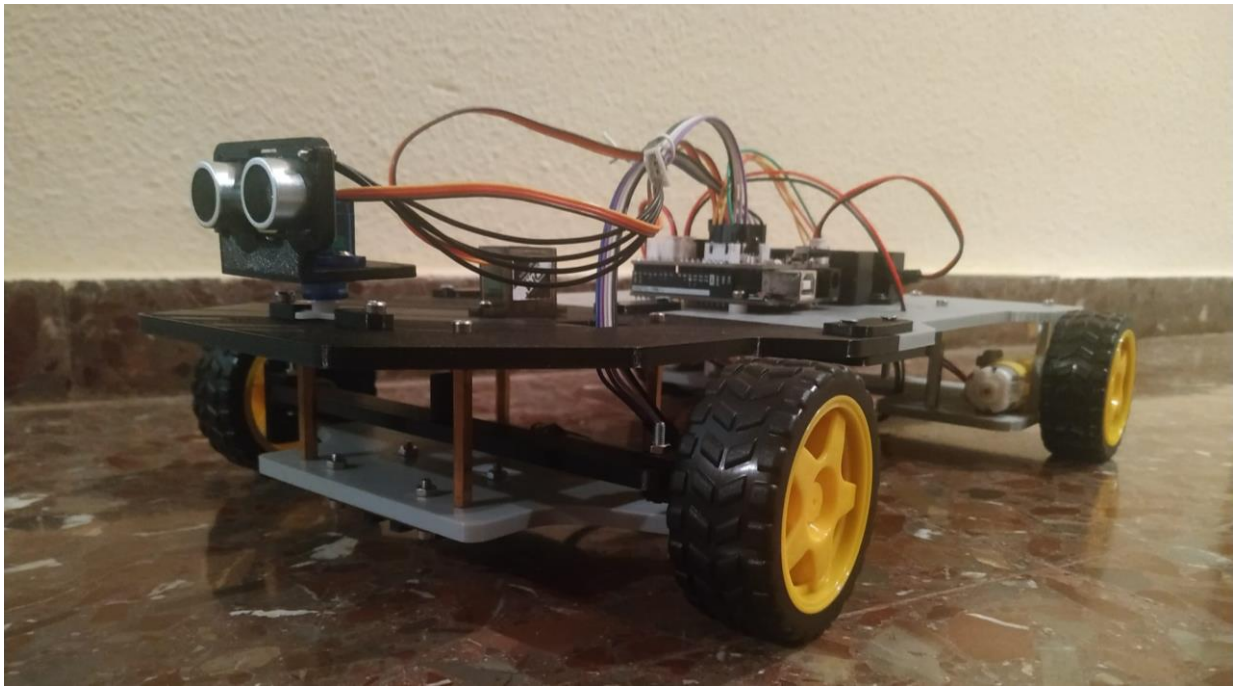


Figura 21: Vehículo montado

3.5. CONEXIONES ELÉCTRICAS

En este apartado se explica cómo se han realizado las conexiones de los distintos componentes con la placa Arduino. Para ayudar a la visualización, se emplea el programa Fritzing, que permite crear esquemas electrónicos en formato protoboard o esquemático. Este programa facilita la creación de esquemas de fácil visualización de manera rápida. Sin embargo, sus bibliotecas son limitadas, y no incluyen todos los componentes utilizados en este proyecto.

Las conexiones eléctricas ejecutadas en el modelo real son triviales, puesto que, al disponer de una placa de expansión las conexiones son inmediatas. Por tanto, en este apartado tan solo se mostrará de forma esquemática las conexiones de los sensores y los actuadores conectados directamente a la placa Arduino, puesto que uno de los componentes que no están disponibles en la librería es precisamente la placa de expansión. De este modo, se puede visualizar qué pines se han asignado a cada componente, lo cual facilitará la comprensión posterior del código desarrollado. En la Esquema de conexiones completo2 se encuentra el esquema eléctrico al completo. Por ausencia de estos elementos en la biblioteca del programa, la placa de expansión se ha sustituido por una placa de prototipos, con el controlador de los motores DRV8835 por separado. La representación de la batería tampoco corresponde a la utilizada en la realidad.

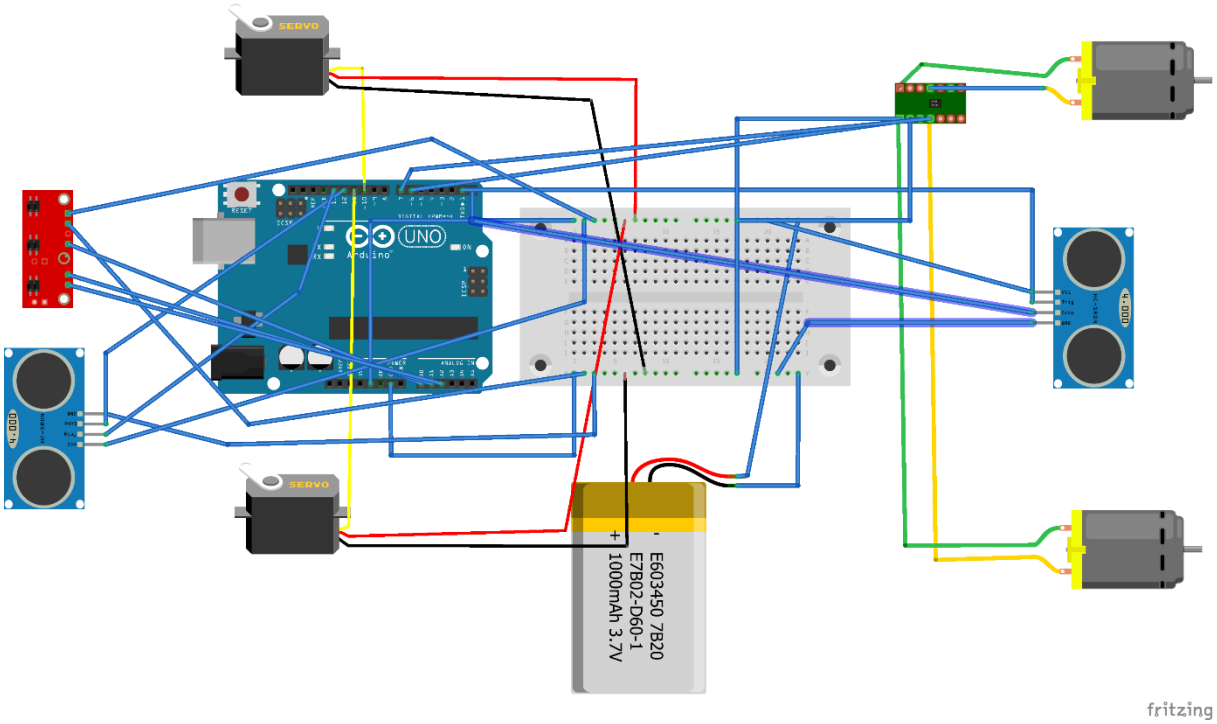


Figura 22: Esquema de conexiones completo

Como el esquema al completo resulta complejo de comprender, se dará una explicación de las conexiones de cada elemento:

- El sensor de seguimiento de línea cuenta con 5 conexiones. La entrada de tensión para alimentar los emisores de infrarrojos y la masa se conectan a sus respectivos pines en la placa controladora. Las 3 entradas de los receptores de infrarrojos derecho, izquierdo y central se conectan a los pines analógicos A0, A1 y A2 respectivamente.

- El sensor de ultrasonidos delantero, además de las conexiones de Vcc y GND, cuenta con un pin TRIG y un pin ECHO. El primero, se conecta como salida al pin digital 13 de la placa, mientras que el segundo se conecta como entrada al pin número 12. De igual manera, el sensor trasero conecta su TRIG al pin 1 y el ECHO al 0. A estos pines se les denomina seriales, y se utilizan para comunicar la placa Arduino con otros dispositivos. No es ideal utilizar estos para conectar un sensor de este tipo, pero al ser los que están conectados a la placa de expansión no queda más remedio. El uso de estos pines en la placa se debe a que permitiría en un futuro conectar otro tipo de dispositivos, como por ejemplo una cámara de vídeo.

- Los servomotores tienen sus pines de tensión y masa conectados igual que los anteriores, mientras que las señales de control se conectan a pines con capacidades PWM. En el caso del servo de la dirección, se escoge la salida 11, y para el servo que mueve el sensor de ultrasonidos delantero se utiliza la número 10.

- Los motores se conectan al controlador DRV8835, y compartirán sentido de giro y velocidad puesto que deben funcionar como si compartieran el mismo eje como ocurre en los automóviles convencionales. El controlador dispone de la entrada EN, que regula la velocidad de giro de los motores, que se conectará como salida al pin 6 del controlador, puesto que se requiere una señal PWM. El sentido de giro de los motores se regula en el controlador con una señal digital, por lo que se asigna el pin 7 a esta función. En la placa de expansión, al disponer de 4 conexiones para motores, se conectan los dos motores de los que se disponen a los pines izquierdos.

Las conexiones en el modelo real, efectuadas a través de la placa de expansión, quedan mucho más ordenadas como se puede apreciar en la figura 23:

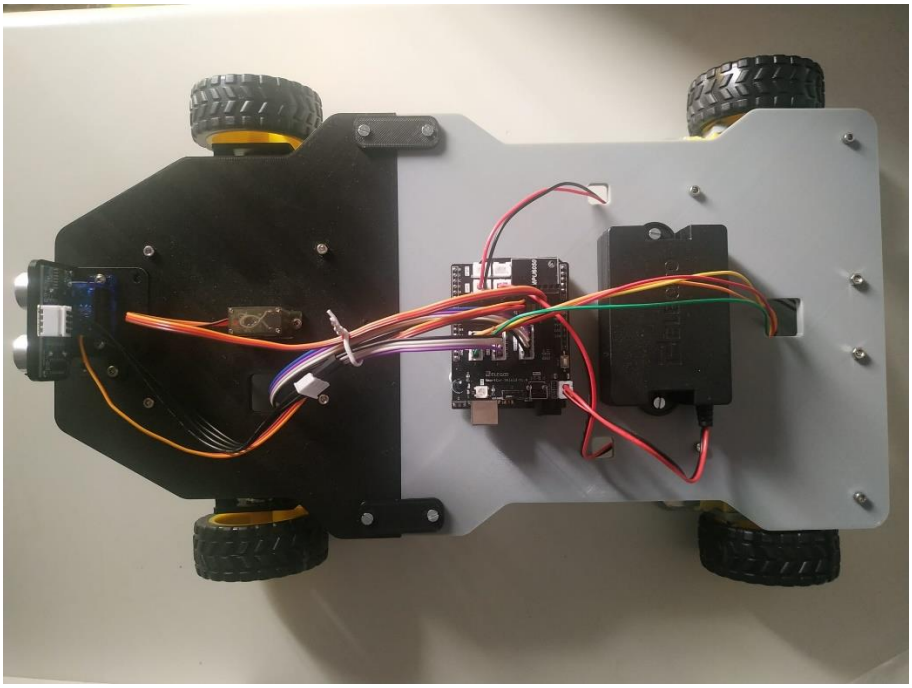


Figura 23: Vista superior del vehículo

Además de las ventajas que ofrece la placa de expansión en términos de organización del cableado y facilidad de montaje, se pueden destacar también otras ventajas que nos ofrece frente a utilizar el controlador de Arduino junto con una placa de prototipos. En primer lugar, la placa de expansión está diseñada para encajar sus pines con los del controlador. De este modo, se reduce el espacio en planta que ocuparía una placa de prototipos, puesto que la de expansión se coloca verticalmente sobre el Arduino, no añadiendo ningún espacio necesario más en la base del vehículo.

El espacio en planta también se reduce al llevar integrado en la placa el controlador de los motores. En las primeras iteraciones del vehículo no se empleaba la placa de expansión, y por tanto los diseños incorporaban el controlador de motores L298N, que ocupaba una parte significativa del espacio en planta del vehículo.

Por otro lado, la desventaja principal de la placa de expansión es que limita el vehículo a tener una estructura poco flexible, puesto que las posibilidades que ofrece es una combinación de tres sensores, dos servomotores, 4 motores y un acelerómetro-giroscopio. Esto implica que, en caso de que se necesite una configuración diferente a la que se presenta en este proyecto, sea necesario adquirir un modelo diferente de placa de expansión o utilizar solamente el controlador.

Pese a todo, en la zona central del vehículo se dispone de espacio para que, en un futuro, se puedan aprovechar los tornillos de la unión de las bases para incorporar una pieza que permita incorporar una placa de prototipos de pequeñas dimensiones.

CAPÍTULO 4. PROGRAMACIÓN

En este capítulo se presentan los programas desarrollados para comprobar el correcto funcionamiento del vehículo. Se debe matizar que estos programas van a presentar una estructura sencilla, puesto que será el grupo de investigación de los tutores de este TFG quien se encargará de desarrollar los algoritmos más robustos y completos para este modelo de vehículo. El objetivo de estos programas es por tanto observar la respuesta mecánica del vehículo y verificar que esta es adecuada, o en su defecto observar los defectos de este para plantear una solución posterior.

Se comenzará explicando las funciones desarrolladas para cada acción que deba tomar el vehículo, y posteriormente se agruparán en programas completos que permitan una conducción con detección de obstáculos y otra con seguimiento de líneas.

Antes de nada, es necesario conocer los pasos a seguir antes de poder enviar el programa a la memoria de la placa controladora. Para el desarrollo del código, se ha empleado el software en línea de Arduino [9]. Este cuenta con una ventana en la parte izquierda de la pantalla en la que se puede acceder a los programas creados por el usuario, bibliotecas, ejemplos, el monitor del sistema y los ajustes y preferencias. Si se selecciona el apartado *sketchbook,* se puede acceder a los distintos programas que ya se hayan creado, y se abre una ventana en la que se puede introducir código. Encima de esta ventana se encuentran los botones de compilado, guardado y una pestaña que permite seleccionar el dispositivo al que se pretende enviar el *sketch* y el puerto por el cual se va a enviar. En este caso, se selecciona la placa Arduino UNO y el puerto COM9, que permite la conexión entre el ordenador y la placa a través de su puerto USB tipo B. Hecho esto se puede comenzar a explicar las funciones que se van a implementar.

4.1. CONTROL DE LOS MOTORES

El primer grupo de funciones servirán para indicar la marcha del vehículo. Se deben implementar de manera que el vehículo pueda avanzar, detenerse y retroceder. Para ello es de vital importancia conocer el esquema de movimiento que se presenta en la tabla 4. En ella se puede ver que para que los dos motores avancen, se debe enviar una señal analógica a las entradas habilitadoras, y una señal de nivel alto a la salida INB. Para retroceder, la señal de nivel alto se debe enviar a la salida INA. Por último, para detener los motores, ambas entradas deben estar a nivel bajo.

Las señales analógicas pueden tener un valor entre 0 y 255, valores entre los cuales se puede indicar la de velocidad de giro del motor.

ENA (salida 5)	ENB (salida 6)	INA (salida 7)	INB (salida 8)	Función
0	0	X	X	Parar
1	1	0	1	Avanzar
1	1	1	0	Retroceder

Tabla 4: Control de dirección del controlador DRV8835

Conociendo esto, se procede a convertir la información de la tabla en el código de las tres funciones que se pueden desarrollar para los movimientos de los motores. Más adelante se combinarán con otras funciones para dotar al vehículo de control de dirección.

```
109
110 /*mueve los motores hacia delante*/
111 void delante(){
112   analogWrite(ENA, velocidad);
113   digitalWrite(INA, LOW);
114   analogWrite(ENB, velocidad);
115   digitalWrite(INB, HIGH);
116 }
117 /*mueve los motores hacia detras*/
118 void detras() {
119   analogWrite(ENA, velocidad);
120   digitalWrite(INA, HIGH);
121   analogWrite(ENB, velocidad);
122   digitalWrite(INB, LOW);
123 }
124 /*detiene los motores*/
125 void parar() {
126   analogWrite(ENA, 0);
127   digitalWrite(INA, HIGH);
128   analogWrite(ENB, 0);
129   digitalWrite(INB, LOW);
130 }
```

Figura 24: Funciones de control de los motores

Estas funciones, así como las que se van a presentar en apartados posteriores, podrán ser llamadas más adelante por el programa principal, lo que permitirá facilitar la comprensión y organización del conjunto.

4.2. CONTROL DE LOS SERVOMOTORES

En este apartado se mostrarán las funciones que gobernarán tanto el servo de la dirección como el que moverá el sensor de ultrasonidos delantero. Para ello nos valdremos de la biblioteca específica de Arduino *servo.h*, que facilita el control de servomotores, puesto que permite el empleo de las siguientes funciones [10]:

- `Servo nombre_servo`: declara una variable de tipo servo.

- `attach(pin, pulso_minimo, pulso_maximo)`**:** establece un pin de Arduino como el pin que controlará el servomotor. Asimismo, establece los valores mínimos y máximos, en microsegundos, de la duración del pulso de la señal PWM que enviará dicho pin. Por lo general, los servos funcionan con una señal de 20 milisegundos de periodo. Variando la duración del pulso de esta señal, se consigue modificar la posición del eje del servo. En el caso del servo SG90 del que se dispone, la posición de 0 grados corresponde a una señal de 500 microsegundos, y la de 180 grados a una señal de 2400 microsegundos.

- `nombre_servo.write(angulo)`**:** permite posicionar el eje del servo en la posición que se desee, recibiendo como parámetro de entrada el ángulo al cual queramos posicionarlo.

Cabe mencionar, antes de presentar el código para cada servo, que los servos que se utilizan, tanto el de dirección como el del sensor de ultrasonidos, parten desde una posición de reposo central. Esto implica que, antes de montarlos al conjunto, es necesario que los posicionemos en su posición central, es decir, a 90 grados. Para ello basta con ejecutar un programa de prueba con la función `nombre_servo.write(90)`, y atornillar el brazo del servo en la posición central. Para el servo de dirección, el brazo debe atornillarse paralelo a la línea que une a los tornillos del servomotor, mientras que, para el servo del sensor, se debe colocar en posición perpendicular.

En la Figura 25 se exponen las funciones desarrolladas para el control de los servomotores:

```
/*Control del servo de direccion*/

/*se mueve el eje de direccion 45 grados hacia la izquierda*/
void giroizquierda() {
  for (i = 90; i >= 45; i--) {
    servo2.write(i);
    delay(20);
  }
}
   /*se mueve el eje de direccion 45 grados hacia la derecha*/
  void giroderecha() {
    for (i = 90; i <= 135; i++) {
    servo2.write(i);
    delay(20);
    }
  }
  /*el servo vuelve a la posicion de centro desde la posicion izquierda*/
  void centradoizquierda() {
    for (i = 45; i <= 90; i++) {
    servo2.write(i);
    delay(20);
    }
  }
  /*el servo vuelve a la posicion de centro desde la posicion derecha*/
  void centradoderecha() {
    for (i = 135; i >= 90; i--) {
    servo2.write(i);
    delay(20);
    }
  }
```

Figura 25: Funciones del servo de dirección

Se puede observar que la función de dirección solo permite efectuar un giro con un ángulo determinado. Se podrían implementar funciones que permitan ángulos de giro variables, pero se incrementaría considerablemente la complejidad del programa final, y la extensión de los documentos. Dado que el objetivo de esta programación es efectuar una comprobación de que los sistemas electrónicos, mecánicos y sus interacciones cumplen con los requisitos del departamento, se opta por no desarrollar algoritmos excesivamente complejos.

En cuanto a la función del servo del sensor, en el siguiente apartado se completará el código añadiéndole las funciones de medición del sensor.

4.3. FUNCIONES DEL SENSOR DE ULTRASONIDOS

Para poder operar con el sensor HC-SR04 primero hay que conocer cómo utilizarlo correctamente, conocer las funciones necesarias e implementar las características físicas de las señales del sensor en el código.

Como ya viene explicado en la página 10, la relación entre la distancia entre el sensor y el objeto viene dada por la expresión:

$$d = \left(\frac{ct}{2}\right)$$

Si se tiene en cuenta que, para una aplicación como esta, la distancia la mediremos en centímetros y que la variable de tiempo, como comentaremos a continuación, se recibirá en microsegundos, se obtiene una constante para calcular la distancia en función del tiempo de la forma:

$$d = \left(\frac{1}{2} 340 m/s \frac{100\ cm}{m} \frac{1s}{1000000 \mu s}\right) = 0.017 cm/\mu s$$

Esta constante se utilizará en el código, multiplicando al valor de tiempo recibido por la función `pulseIn`. Esta función requiere dos parámetros de entrada, de la forma `pulseIn(pin, HIGH)`. Su función es contar, en intervalos de 10 microsegundos, el tiempo que tarda en colocarse a nivel alto el pin que se haya indicado y devolver ese tiempo. El pin que se dará como parámetro de entrada en la función será por tanto aquel que esté conectado al terminal ECHO del sensor, pues es el encargado de recibir los pulsos de ultrasonidos.

Se puede observar que esta función solo calcula el tiempo desde que es llamada hasta que el pin recibe la señal, pero no se tiene información de cuando se ha emitido el pulso. Es por esto por lo que el intervalo de tiempo entre la emisión del pulso y la llamada a la función debe ser lo más reducido posible. Esto se consigue con la función `digitalWrite(pin, estado)` y la función `delay(tiempo)`. Primero se debe llamar a la primera función, dándole como parámetros de entrada el pin del controlador asignado al terminal TRIG del sensor y el estado HIGH. Después, se utiliza la función `delay` antes de volver a poner el terminal a nivel bajo. Se empleará un retardo de 1 microsegundo, que es el mínimo que permite la función. El error que se comete entre la emisión del pulso y la recepción es insignificante en esta escala, puesto que sería de 0.017 centímetros. Además, el uso de intervalos de 10 microsegundos de la función `pulseIn` hace que el error en la medición oscile entorno a los 0.17 centímetros.

Por último, queda establecer la distancia a la que se pretende detener el vehículo respecto del objeto que encuentre. Tras una serie de pruebas, se establece esta distancia en 30 centímetros, pues permite una parada a una distancia cómoda para poder maniobrar sin dejar demasiado lejos el obstáculo. Se desarrollan en total 3 funciones diferentes: una función que detecta obstáculos directamente delante del vehículo cuando este se encuentra en marcha, otra que los detecta cuando el vehículo da marcha atrás y otra que detectará los obstáculos cuando el coche se encuentra en reposo y el conjunto servo-sensor hace un barrido, indicando en qué dirección se encuentra el obstáculo más cercano.

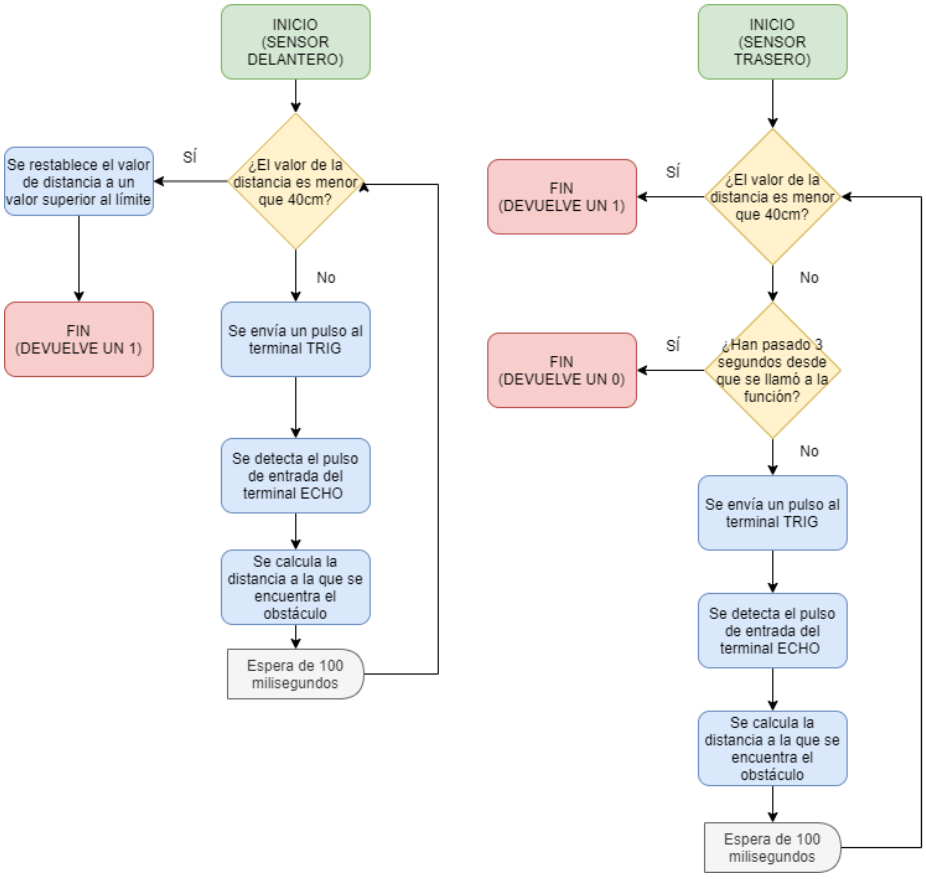


Figura 26: Diagramas de flujo de las funciones de detección de obstáculos

```
/* Control del sensor cuando el coche se encuentra en marcha hacia delante*/

int ultradelante(){
  while(distancia>40){
    digitalWrite(Trigdelante, HIGH);
    delayMicroseconds(5);
    digitalWrite(Trigdelante, LOW);
    tiempo = pulseIn(Echodelante, HIGH);
    distancia = tiempo*0.017;
    delay(100);
}
  distancia = 1000;
  return 1;
}

/* Control del sensor cuando el coche se encuentra en marcha hacia detras*/

int ultradetras(){
  for(i=0; i<=10;i++){
    digitalWrite(Trigdetras, HIGH);
    delayMicroseconds(5);
    digitalWrite(Trigdetras, LOW);
    tiempo = pulseIn(Echodetras, HIGH);
    distancia = tiempo*0.017;
    delay(300);
   if (distancia<40){     /*Si encuentra un obstáculo, la función devuelve un 1*/
      distancia = 1000;
      return 1;
    }
  }
      return 0;       /*Si no encuentra un obstáculo, la función devuelve un 0*/
 }
```

Figura 27: Funciones de detección de obstáculos

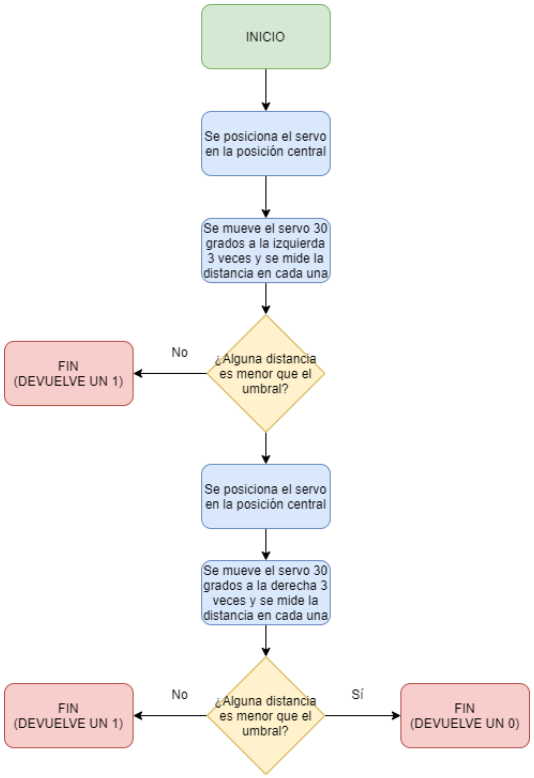


Figura 28: Diagrama de flujo de la función de barrido de ultrasonidos

```
/*Control del sensor cuando el vehiculo ha encontrado un obstáculo y se detiene para realizar un barrido */

int ultrabarrido(){
  servo1.write(90);           /*se posiciona el servo del sensor en posicion 0*/
  distanciamin = 1000;
  for(i=3; i<=6; i++){
    servo1.write(30*i);
    delay(200);
    digitalWrite(Trigdelante, HIGH);
    delayMicroseconds(5);
    digitalWrite(Trigdelante, LOW);
    tiempo = pulseIn(Echodelante, HIGH);
    distancia = tiempo*0.017;
    delay(500);
      if (distancia<distanciamin){      /*Almacenamos en la memoria de la variable  distanciamin el valor mas bajo*/
        distanciamin=distancia;
      }
  }
  if (distanciamin>29){
    servo1.write(90);                   /*Si no hay obstaculos a menos de 19 cm, la funcion devuelve un 1 (girar a la izquierda)*/
    return 1;
  }
  if(distanciamin<29){
    distanciamin = 1000;
  for(i=6; i>=0; i--){
    servo1.write(30*i);
    delay(200);
    digitalWrite(Trigdelante, HIGH);
    delayMicroseconds(5);
    digitalWrite(Trigdelante, LOW);
    tiempo = pulseIn(Echodelante, HIGH);
    distancia = tiempo*0.017;
    delay(500);
    if (distancia<distanciamin){
      distanciamin=distancia;
    }
  }
  }
  if (distanciamin>29){
    servo1.write(90);
    return 2;       /*Si no hay obstaculos a menos de 19 cm, la funcion devuelve un 2 (girar a la derecha)*/
  }
 if (distanciamin<29){
      servo1.write(90);
    return 0;       /*Si hay obstaculos a menos de 19 cm, a ambos lados la  funcion devuelve un 0 (dar marcha atras)*/
        }
      }
```

Figura 29: Barrido de ultrasonidos

Para esta última función, el servo del sensor de ultrasonidos se coloca en la posición central. Después, realiza 3 movimientos de 30 grados hacia la izquierda, tomando una medición de distancia con cada movimiento. Si no detecta un obstáculo, la función devuelve un 1 y termina su ejecución. Si lo detecta, realiza otros 3 movimientos de 30 grados hacia la derecha, desde la posición central, con sus respectivas mediciones. Si no detecta un obstáculo la función devolverá en este caso un 2. Si el sensor detecta obstáculos en ambos casos, la función devolverá un 0.

Con esta función concluyen todas las funciones que componen el código del programa de detección de obstáculos mediante ultrasonidos. En el anexo 1 se encuentra el código principal, con la declaración de las variables, funciones y pines, así como los códigos de configuración inicial (o *setup*) y el bucle principal de control del módulo de ultrasonidos.

El funcionamiento del bucle es el siguiente: primero se pone en marcha el vehículo y se ejecuta la función que detecta obstáculos en la dirección de la marcha. Si se detecta un obstáculo, el vehículo se detiene y ejecuta la función de barrido. Si esta devuelve un 0, el vehículo da marcha atrás durante 3 segundos o hasta que encuentre un obstáculo. En ambos casos, el vehículo se detiene y realiza una esquiva hacia la izquierda y continua con la marcha. Si la función barrido devuelve un 1, el vehículo ejecuta directamente la maniobra de esquiva hacia la izquierda, mientras que si devuelve un 2, la ejecuta hacia la derecha.

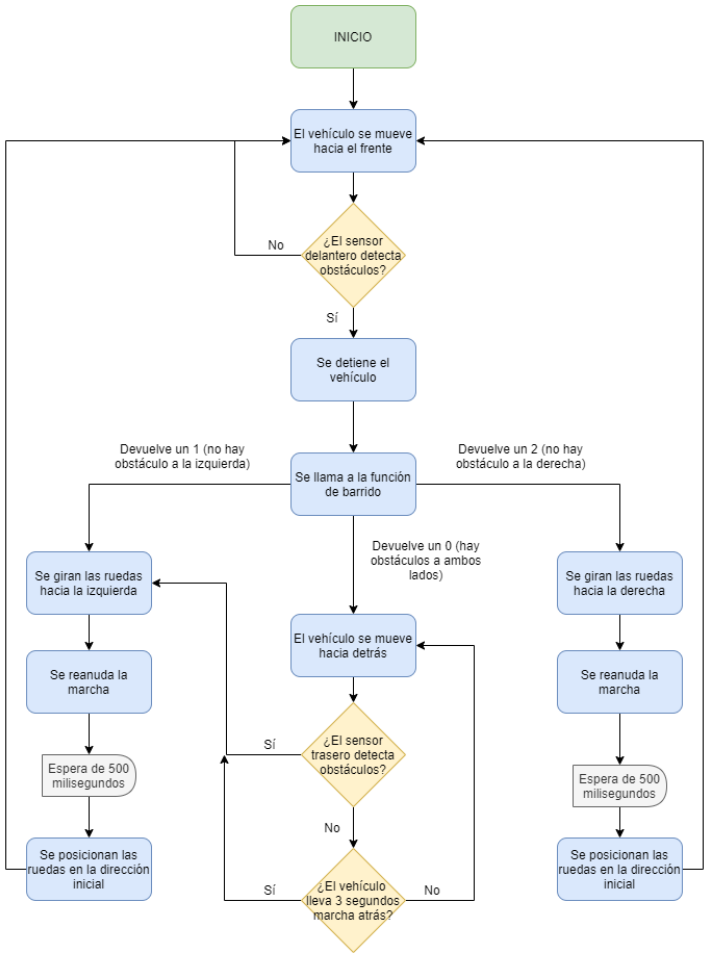


Figura 30: Diagrama de flujo del bucle de control de ultrasonidos

4.4. FUNCIONES DEL MÓDULO DE SEGUIMIENTO DE LÍNEA

Las últimas funciones que quedan por definir son aquellas que permitirán al vehículo realizar el seguimiento de línea. Antes de comenzar con la función de detección, se realiza un código de prueba para observar cuales son los valores de entrada que proporcionan los sensores ITR20001 cuando apuntan a una superficie de alta y baja reflectividad. Para ello, se situó una hoja de papel con una cinta adhesiva negra, y se coloca el módulo en diferentes posiciones. El código siguiente (figura 31) proporciona la lectura de los sensores, y permite la visualización de los datos en el monitor serial [11], como se muestra en la Figura 32.

```
#define SensorIzq A2
#define SensorMed A1
#define SensorDer A0

float lecturaIzq;
float lecturaMed;
float lecturaDer;

void setup() {
  Serial.begin(9600);
  pinMode(SensorIzq, INPUT);
  pinMode(SensorMed, INPUT);
  pinMode(SensorDer, INPUT);
}

void loop() {
 lecturaIzq = analogRead(SensorIzq);
 lecturaMed = analogRead(SensorMed);
 lecturaDer = analogRead(SensorDer);
  Serial.print(lecturaIzq);
  Serial.print("\t");
  Serial.print(lecturaMed);
  Serial.print("\t");
  Serial.print(lecturaDer);
  Serial.print("\n");
  delay(3000);
}
```

Figura 31: Lectura de datos del módulo de seguimiento de línea

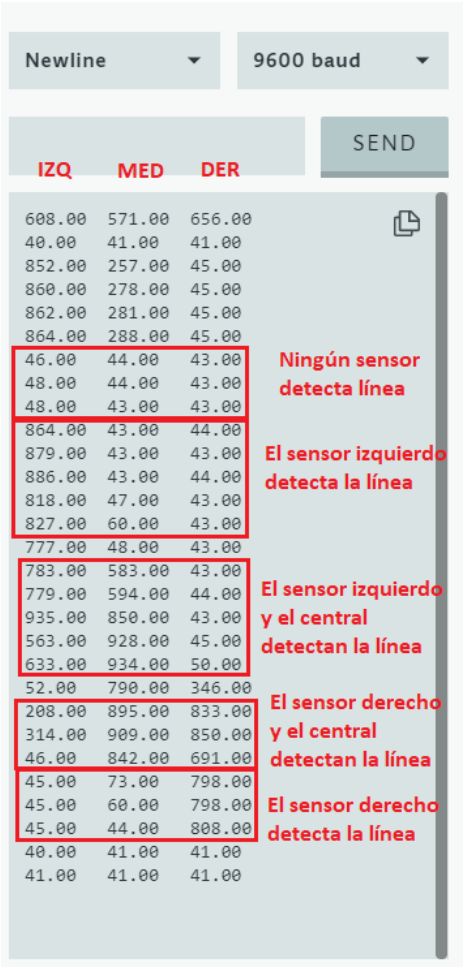


Figura 32: Visualización de datos de entrada del módulo de seguimiento de línea a través del monitor serial

Como se observa en la figura 32, cuando el sensor incide sobre una superficie reflectiva, éste manda una señal a la placa que proporciona valores bajos, entre 40 y 60, mientras que cuando incide sobre una superficie no reflectiva, devuelve valores entre 550 y 1000. Con estos datos se puede por lo tanto definir cuáles serán los umbrales de detección que se implementarán en el código. Hay que tener en cuenta que, en condiciones reales, la reflectividad del suelo no será tan alta como la de una hoja de papel en blanco, por lo que los límites inferiores tendrán que abarcar un mayor rango que el que se muestra en esta prueba. Además, en suelos de colores oscuros, la línea deberá ser de colores claros para contrastar, y consecuentemente se tendrán que realizar ajustes al código.

Para la lectura en suelos reflectantes, se realiza la función mostrada en la Figura 33. Se ha aplicado un umbral que permite al programa asimilar como una detección de la línea cuando el valor del sensor supera un valor de 200, puesto que las mediciones del suelo en el que se llevaron a cabo las pruebas este valor no era mayor de 100 en ningún caso. El código cuenta también con una variable “estado” que almacena un valor en función de que sensor está activado en cada momento, y una variable “estadoanterior” que almacena el valor del último estado del módulo en caso de que este no detecte la línea con ninguno de sus 3 sensores. Esto permite al vehículo buscar la línea en caso de que esta tome un giro demasiado brusco.

```
61 void lectura(){
62   if(estado!=1){
63  estadoanterior=estado;
64   }
65  lecturaIzq = analogRead(SensorIzq);
66  lecturaMed = analogRead(SensorMed);
67  lecturaDer = analogRead(SensorDer);
68
69  if (lecturaMed>=UmbralInf){                        /*sensor central detecta linea*/
70   estado = 1;
71 }
72  if(lecturaMed<UmbralInf&&lecturaIzq>=UmbralInf){    /*sensor izq detecta linea*/
73   estado = 2;
74  }
75   if(lecturaMed<UmbralInf&&lecturaDer>=UmbralInf){   /*sensor der detecta linea*/
76   estado = 3;
77  }
78  if(lecturaMed<UmbralInf&&lecturaIzq<UmbralInf&&lecturaDer<UmbralInf){
79    estado=estadoanterior;
80  }
81
82 }
```

Figura 33. Código de lectura del módulo de seguimiento de línea

Como las funciones de movimiento y giro son las mismas que se utilizan en la función de detección de obstáculos, ya se puede presentar el código completo, que se encuentra en el anexo 2. El funcionamiento del bucle de control es sencillo: se pone en marcha el vehículo y se modifica la posición del servo en función de los valores que tome la variable estado. Un valor 1 significa que el sensor central detecta la línea, por lo que se mantiene su trayectoria, y los valores 2 y 3 indican que el sensor central no detecta la línea, pero si el izquierdo y el derecho respectivamente. Esto se completa con la variable “estadoanterior” que ya hemos explicado, la cual solo puede tomar los valores 2 y 3, para que, en caso de no detectar la línea, el vehículo la busque girando su dirección.

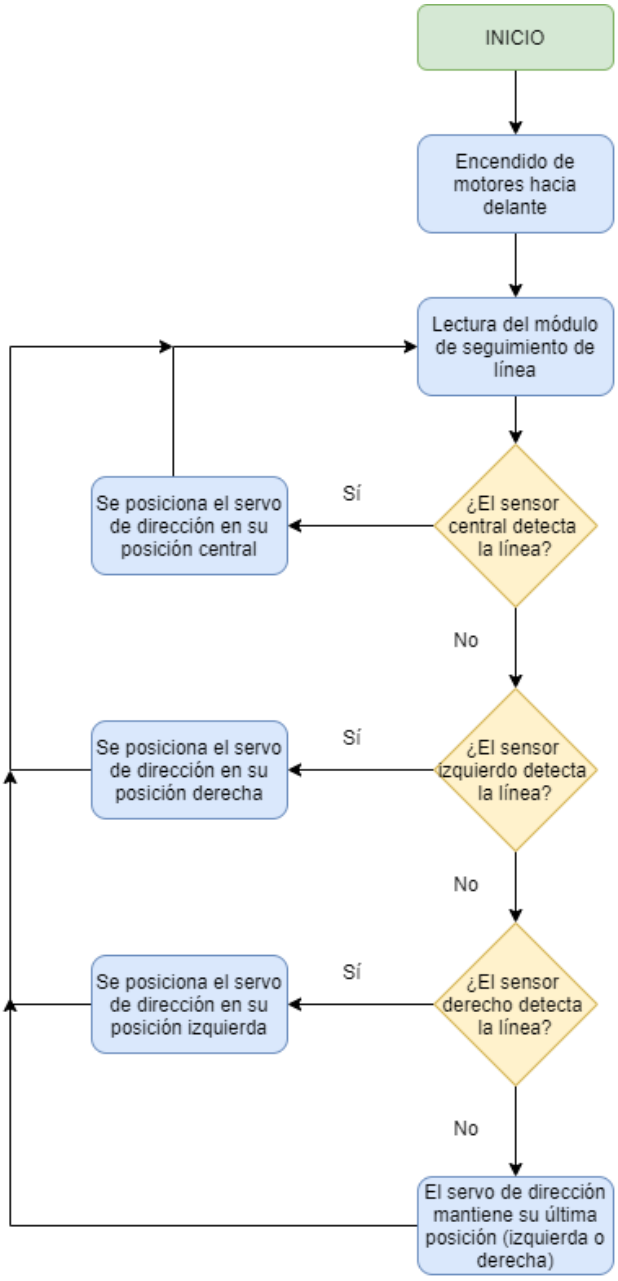


Figura 34: Diagrama de flujo del control por seguimiento de línea

CAPÍTULO 5. RESULTADOS

5.1. PRUEBAS

Una vez desarrollados los códigos de prueba, se llevaron a cabo las pruebas del vehículo. Estas se realizaron en un aparcamiento local, puesto que dispone de un amplio espacio asfaltado y nula presencia de vehículos durante el tiempo que duraron las pruebas. Por otro lado, el terreno presentaba pequeños desniveles, que desequilibraban la trayectoria del vehículo, así como irregularidades en la superficie, como pequeños fragmentos de grava, que dificultaban las tareas de lectura de datos en el caso del programa de seguimiento de línea.

Se realizaron 3 pruebas en total:

- En la primera, el controlador se carga con el programa de detección de obstáculos. Se dispone en el camino del vehículo un obstáculo de pequeñas dimensiones, que es capaz de detectar, y el vehículo se detiene. A continuación, efectúa la rutina de barrido de ultrasonidos y al no detectar obstáculos a su izquierda efectúa la maniobra de esquiva en esa dirección.
- En la segunda, con el mismo programa, se dispone frente al vehículo un obstáculo de mayores dimensiones, y tras él otro obstáculo más. Al efectuar la maniobra de barrido frente a este, y detectar que el obstáculo no puede ser sorteado, procede a dar marcha atrás durante 3 segundos o hasta que encuentre un obstáculo. Al detectar el obstáculo con el sensor trasero, el vehículo modifica su trayectoria y ejecuta la maniobra de esquiva.
- En la última prueba, se ejecuta el programa de seguimiento de línea. Para ello, se realizó una línea continua con cinta adhesiva negra y se tomaron medidas de la reflectividad del pavimento. Al tratarse de asfalto, estas medidas eran irregulares y de no muy alta reflectividad, por lo que se tuvo que modificar el código para aumentar el umbral de detección, para evitar falsas señales de detección. Esta medida tiene como consecuencia la no detección de algunas señales correctas de detección, por lo que se disminuyó el tiempo entre lecturas en 20 milisegundos. De esta manera se consigue solventar en cierta medida el problema de las lecturas erróneas, como se verá a continuación en el vídeo de demostración.

Las pruebas fueron grabadas, y se puede acceder al vídeo a través del siguiente enlace:

https://youtu.be/tJZO1QCP07g

5.2. OBSERVACIONES

Durante las pruebas se observaron comportamientos no deseados en el vehículo, que, si bien son de carácter leve y se pueden corregir, cabe mencionarlos para que se tengan en cuenta y realizar las correcciones y mantenimientos pertinentes.

Una vez concluidas las pruebas del vehículo en el modo de detección de líneas, se observó que, al volver a ejecutar el programa de detección de obstáculos, la dirección era menos precisa que al principio. Se examinó el vehículo y se observó que los tornillos y tuercas que aseguran el eje de dirección no se encontraban correctamente apretados, e incluso una de las tuercas se perdió durante las pruebas. Además, la funda del servo de la dirección dejó de encajar correctamente con el brazo del servomotor. Estos problemas se presumen que se produjeron debido a las altas velocidades de conmutación del servo cuando se carga el programa de seguimiento de línea, que acabaron desgastando los citados componentes.

Otro de los inconvenientes encontrados es que el radio de giro del vehículo es elevado, pero este ya se anticipaba, puesto que se trata de una maqueta de grandes dimensiones que dispone de un sistema de servodirección en lugar de una configuración de 4 motores, que permiten radios de giro más pequeños.

CAPÍTULO 6. CONCLUSIONES

En líneas generales, se puede concluir que los objetivos marcados al inicio del proyecto se han cumplido satisfactoriamente. Se ha podido desarrollar una plataforma en la que se ha comprobado que cuenta con las bases para desarrollar algoritmos de conducción autónoma enfocada a vehículos turismo. Además, la fabricación en 3D ha permitido fabricar los componentes en menos de 24 horas y con un coste reducido, al tratarse de un proyecto de tirada única. Haber desarrollado las piezas en 3D ha permitido también que sea fácil aplicar sobre el vehículo base modificaciones y ampliaciones. El montaje es sencillo e intuitivo, y no requiere herramientas. Por último, se ha comprobado que el vehículo cuenta con una gran accesibilidad de los componentes, y resulta sencillo manipular las conexiones electrónicas de manera ordenada.

Por todo esto, se espera que este proyecto impacte de manera positiva al grupo de investigación de Sistemas Tolerantes a Fallos, al que pertenecen los tutores del presente TFG, en su labor de investigación, y que ayude a desarrollar programas de conducción autónoma que mejoren la comprensión de su funcionamiento a sus desarrolladores, para que posteriormente puedan aplicar los conocimientos adquiridos en vehículos comerciales, y conseguir mejorar la seguridad en las carreteras.

BIBLIOGRAFÍA

[1] Norma ISO/ASTM 52900:2015, *iso.org.* Fecha de consulta: 7 de junio de 2021. Disponible en https://www.iso.org/obp/ui/#iso:std:iso-astm:52900:ed-1:v1:en

[2] Página oficial de Arduino: https://arduino.cc

[3] Especificaciones técnicas de Arduino UNO REV 3, *arduino.cc*, 2020. Fecha de consulta: 12 de Julio de 2021. Disponible en https://store.arduino.cc/arduino-uno-rev3

[4] Modelo 3D del sensor HC-SR04, *grabcad.com*, 2020. Fecha de consulta: 25 de mayo de 2021. Disponible en https://grabcad.com/library/ultrasonic-sensor-hc-sr04-3

[5] Modelo 3D de la placa Arduino UNO, *grabcad.com*, 2015. Fecha de consulta: 25 de mayo de 2021. Disponible en https://grabcad.com/library/arduino-uno-4

[6] Modelo 3D del servomotor SG-90, *grabcad.com*, 2017. Fecha de consulta: 25 de mayo de 2021. Disponible enhttps://grabcad.com/library/sg90-micro-servo-9g-tower-pro-1

[7] Modelo 3D de ruedas de 65mm para motores de escobillas, *grabcad.com*, 2019. Fecha de consulta 25 de mayo de 2021. Disponible en https://grabcad.com/library/wheel-d65x25-1

[8] Hoja de características del chip DRV8835, Texas Instruments, *ti.com.* Fecha de consulta 4 de agosto de 2021. Disponible en https://www.ti.com/lit/ds/symlink/drv8835.pdf

[9] Enlace al editor de código de Arduino IDE en la nube, *arduino.cc*, 2021. https://create.arduino.cc/editor

[10] Funciones de la librería servo de Arduino, *arduino.cc*. Fecha de consulta 25 de Julio de 2021. Disponible en https://www.arduino.cc/reference/en/libraries/servo/

[11] Comunicación serial en Arduino, *arduino.cc*, 2021. Fecha de consulta 25 de julio de 2021. Disponible en https://www.arduino.cc/reference/en/language/functions/communication/serial/

FIGURAS NO ORIGINALES

[12] Impresora prusa i3MK3, *prusa3d.es.* Disponible en https://shop.prusa3d.com/es/impresoras-3d/180-kit-original-prusa-i3-mk3s.html#

[13] Placa microcontroladora Arduino UNO, a*rduino.cc.* Disponible en https://store.arduino.cc/products/arduino-uno-rev3/

[14] Diagrama de pines de Arduino UNO, a*rduino.cc*. Disponible en https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

PRESUPUESTO

1. MATERIAL Y EQUIPOS

En este apartado se incluyen los costes asociados a los objetos físicos que componen el proyecto, así como las herramientas necesarias para su ejecución. Este último grupo engloba tanto las herramientas físicas como las herramientas de software utilizadas.

1.1. MATERIAL Y HERRAMIENTAS DE IMPRESIÓN

Este apartado lo conforman tanto el material utilizado para fabricar las piezas como los materiales y herramientas necesarios para el preparado de la superficie de impresión y la retirada del material de soporte. Además, se debe calcular el coste de amortización de la impresora, siguiendo el siguiente criterio:

El precio base de la impresora es de 769€. Se considera que su vida útil es de 3000 horas, que equivale a un funcionamiento continuado de 8 horas diarias durante un año. Esto conlleva un coste por hora de $\frac{769}{3000} = 0.26€/h$. El tiempo de impresión total de las piezas es de 21 horas.

Concepto	Cantidad	Coste unitario (€)	Subtotal (€)
PLA RS Pro	0.381kg	36.15/kg	13.77
Amortización Impresora Prusa I3MK3	21h	0.26/h	5.46
Alicates	1	1.95	1.95
Laca adhesiva	1	5	5
Coste total del material y herramientas de impresión			**26.18 €**

1.2. ELECTRÓNICA

Concepto	Cantidad	Coste unitario (€)	Subtotal (€)
Elegoo Arduino UNO R3	1	9.99	9.99
Placa de expansión Elegoo	1	1.5	1.5
Batería litio 3.7V	2	6.12	12.24
Módulo de seguimiento de línea de 3 canales	1	15	15
Sensor HC-SR04	2	1.8	3.6
Servomotor SG-90	2	3.33	6.66
Motor de escobillas	2	1.63	3.26
Cable puente 20cm	13	0.05	0.65
Coste total del material electrónico			**52.9 €**

1.3. FERRETERÍA

Concepto	Cantidad	Coste unitario (€)	Subtotal (€)
Tornillo M3x30	2	0.07	0.14
Tornillo M3x16	10	0.06	0.6
Tornillo M3x14	8	0.05	0.4
Tornillo M3x12	5	0.05	0.25
Tornillo M3x8	16	0.05	0.8
Tornillo M2x8	4	0.04	0.16
Tornillo M1.6x8	8	0.04	0.32
Tuerca fina M3	41	0.05	2.05
Tuerca fina M2	4	0.05	0.2
Tuerca fina M1.6	8	0.05	0.4
Rodamiento F695ZZ	2	1.21	2.42
Espaciadores de aluminio M3x40	8	0.81	6.48
Rueda para motor de escobillas	4	0.26	1.04
Coste total de ferretería			**15.26 €**

1.4. SOFTWARE

Gracias a las versiones de uso para estudiantes que ofrecen Onshape, Autodesk y Microsoft, y que el resto del software es gratuito en su forma estándar, el gasto en software ha resultado nulo en este proyecto. Se presentan los precios de las licencias de uso profesional para que se aprecie el impacto que habría tenido sobre el coste total la ausencia de estas versiones para estudiantes. El coste unitario que se muestra es el de una licencia anual, mientras que la cantidad se dispone de manera que equivale a 3 meses de uso.

El coste del equipo informático utilizado para la realización del proyecto no se incluirá en el presupuesto al disponer el alumno de este con anterioridad.

Concepto	Cantidad	Coste unitario (€)	Subtotal (€)
Licencia de Onshape	0.25	1282.38	320.6
Autodesk Inventor	0.25	2886	721.5
Arduino IDE	0.25	0	0
Paquete Microsoft Office	0.25	149	37.5
PrusaSlicer	0.25	0	0
Coste total de software			**1079.6 €**

Estos costes no se incluirán en el presupuesto final al no corresponderse con un gasto realizado.

Concepto	Subtotal (€)
Material y herramientas de impresión	26.18
Electrónica	52.9
Ferretería	15.26
Software	0
Coste total de material y equipos	**94.34€**

2. CONSUMO ELÉCTRICO

En este apartado se incluirá únicamente el gasto eléctrico derivado de la impresión de las piezas que componen el proyecto. Se toma como precio unitario el PVPC medio del mes de julio, que se estableció en 0.1632€/kWh. La potencia que absorbe la impresora Prusa I3MK3 de la red no es un valor constante, puesto que depende de la temperatura ambiente o las operaciones que realice la impresora en función del tipo de pieza y la configuración. De manera que se calcula el gasto energético en función de su potencia nominal, 240W.

Concepto	Potencia (w)	Tiempo de impresión (h)	Gasto energético (kWh)	Precio unitario (€/kWh)	Total (€)
Consumo eléctrico Prusa I3MK3	240	21	5.04	0.1632	0.82

3. MANO DE OBRA

Concepto	Coste unitario (€/h)
Ingeniero graduado en Tecnologías Industriales	25
Ingeniero Tutor UPV	60

Concepto	Horas	Coste unitario (€)	Subtotal (€)
Estudio preliminar y documentación	25	25	625
Diseño del prototipo	150	25	3750
Programación	50	25	1250
Desarrollo de los documentos del proyecto	75	25	1875
Supervisión del proyecto	30	60	1800
Coste total de la mano de obra			**9300 €**

4. COSTE TOTAL DEL PROYECTO

Concepto	**Importe (€)**
Material y equipos	94.34 €
Consumo eléctrico	0.82
Mano de obra	9300
Presupuesto de ejecución material	**9395.16**
Gastos generales (13%)	1221.37
Presupuesto de ejecución	**10616.53**
IVA (21%)	2229.47
Presupuesto base de licitación	**12846 €**

ANEXOS

ANEXO 1. CÓDIGO DETECCIÓN DE OBSTÁCULOS

```
1
2  #include <Servo.h>
3  #define Echodetras 0
4  #define Trigdetras 1
5  #define Echodelante 12
6  #define Trigdelante 13
7  #define ENA 5
8  #define INA 8
9  #define ENB 6
10 #define INB 7
11 #define velocidad 100
12 #define pinservoultra 11
13 #define pinservodireccion
14 Servo servo1;
15 Servo servo2;
16 int pulsomin = 500;
17 int pulsomax = 2400;
18 int i = 0;
19 int tiempo;
20 float distancia = 30;
21 int contador = 0;
22 float distanciamin = 30;
23 int detecta;
24 int barrido = 0;
25
26 void delante();
27 void detras();
28 void parar();
29 void giroderecha();
30 void giroizquierda();
31 void centradoderecha();
32 void centradoizquierda();
33 int ultradelante();
34 int ultradetras();
35 int ultrabarrido();
```

Anexo 1.1. Definición de pines, declaración de variables y funciones

```
void setup() {

  pinMode(INA, OUTPUT);
  pinMode(ENA, OUTPUT);
  pinMode(INB, OUTPUT);
  pinMode(ENB, OUTPUT);
  servo1.attach(pinservoultra, pulsomin, pulsomax);
  servo1.write(90);
  servo2.attach(pinservodireccion, pulsomin, pulsomax);
  servo2.write(90);
  pinMode(Echodetras, INPUT);
  pinMode(Trigdetras, OUTPUT);
  pinMode(Echodelante, INPUT);
  pinMode(Trigdelante, OUTPUT);}
}
```

Anexo 1.2. Función de inicio del programa de detección de obstáculos

```
void loop() {
  delante();
  detecta = ultradelante();
  if (detecta==1){
    parar();
    barrido = ultrabarrido();
  }
  switch(barrido){
    case 0:{
      detras();
      detecta = ultradetras();
      if (detecta==0){
        parar();
        giroizquierda();
        delante();
        delay(500);
        centradoizquierda();
        delay(250);
        giroderecha();
        delay(250);
        centradoderecha();
      }else if (detecta==1){
        parar();
        giroizquierda();
        delante();
        delay(500);
        centradoizquierda();
        delay(250);
        giroderecha();
        delay(250);
        centradoderecha();
    } break;
  }
```

Anexo 1.3. Bucle de detección de obstáculos 1/2

```
88      case 1:{
89        giroizquierda();
90          delante();
91          delay(500);
92          centradoizquierda();
93          delay(250);
94          giroderecha();
95          delay(250);
96          centradoderecha();
97      }break;
98      case 2:{
99        giroderecha();
100         delante();
101         delay(500);
102         centradoderecha();
103         delay(250);
104         giroizquierda();
105         delay(250);
106         centradoizquierda();
107     }break;
108 }
109 }
```

Anexo 1.4. Bucle de detección de obstáculos 2/2

ANEXO 2. CÓDIGO DE SEGUIMIENTO DE LÍNEA

```
#include <Servo.h>
#define SensorIzq A2
#define SensorMed A1
#define SensorDer A0
#define UmbralInf 200
#define velocidad 100
#define ENA 5
#define INA 8
#define ENB 6
#define INB 7
#define pinservodireccion 10
Servo servo2;

int estado, estadoanterior;
float lecturaIzq;
float lecturaMed;
float lecturaDer;
float vectordatos[3] = {0, 0, 0};
int pulsomin = 1000;
int pulsomax = 2000;

void setup() {
  pinMode(SensorIzq, INPUT);
  pinMode(SensorMed, INPUT);
  pinMode(SensorDer, INPUT);
  pinMode(INA, OUTPUT);
  pinMode(ENA, OUTPUT);
  pinMode(INB, OUTPUT);
  pinMode(ENB, OUTPUT);
  servo2.attach(pinservodireccion, pulsomin, pulsomax);
  servo2.write(90);
  estado=1;
  }
```

Anexo 2.1. Declaraciones y configuración de inicio del código de seguimiento de línea

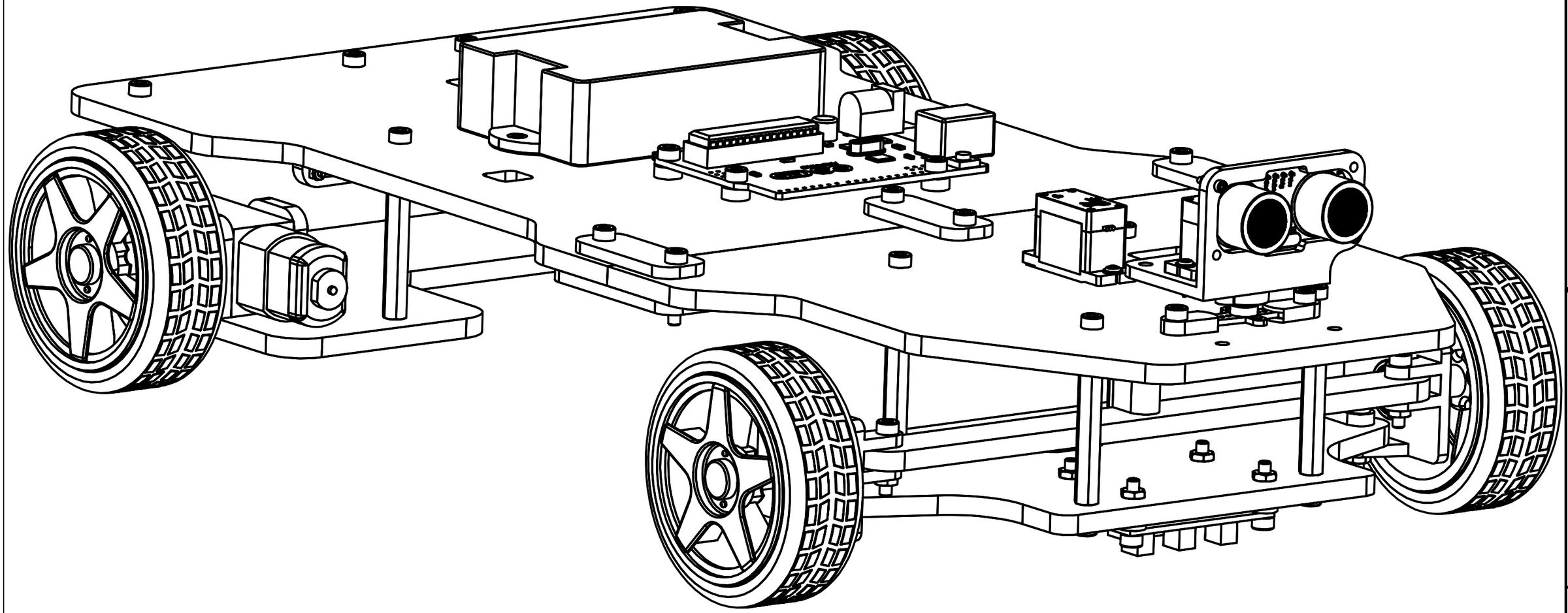
```
void loop() {
  delante();
  lectura();
  switch(estado){
    case 1:{
      servo2.write(90);
      delay(50);
      break;
  }
  case 2:{
      servo2.write(45);
      delay(50);
      break;
  }
  case 3:{
      servo2.write(135);
      delay(50);
      break;
  }
}
}
```

Anexo 2.2. Bucle de seguimiento de línea

```
void lectura(){
  if(estado!=1){}
 lecturaIzq = analogRead(SensorIzq);
 lecturaMed = analogRead(SensorMed);
 lecturaDer = analogRead(SensorDer);

 if (lecturaMed>=UmbralInf){                          /*sensor central detecta linea*/
  estado = 1;
}
 if(lecturaMed<UmbralInf&&lecturaIzq>=UmbralInf){     /*sensor izq detecta linea*/
  estado = 2;
 }
  if(lecturaMed<UmbralInf&&lecturaDer>=UmbralInf){    /*sensor der detecta linea*/
  estado = 3;
 }
 if(lecturaMed<UmbralInf&&lecturaIzq<UmbralInf&&lecturaDer<UmbralInf){
   estado=estadoanterior;
 }

}

void delante(){
  analogWrite(ENA, velocidad);
  digitalWrite(INA, LOW);
  analogWrite(ENB, velocidad);
  digitalWrite(INB, HIGH);
}
```

Anexo 2.3. Funciones de lectura y movimiento

PLANOS



TRABAJO FINAL DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIERÍA INDUSTRIAL VALENCIA

Proyecto:
DESARROLLO DE UN PROTOTIPO A ESCALA DE UN VEHÍCULO PARA PRUEBAS DE LABORATORIO MEDIANTE IMPRESIÓN 3D

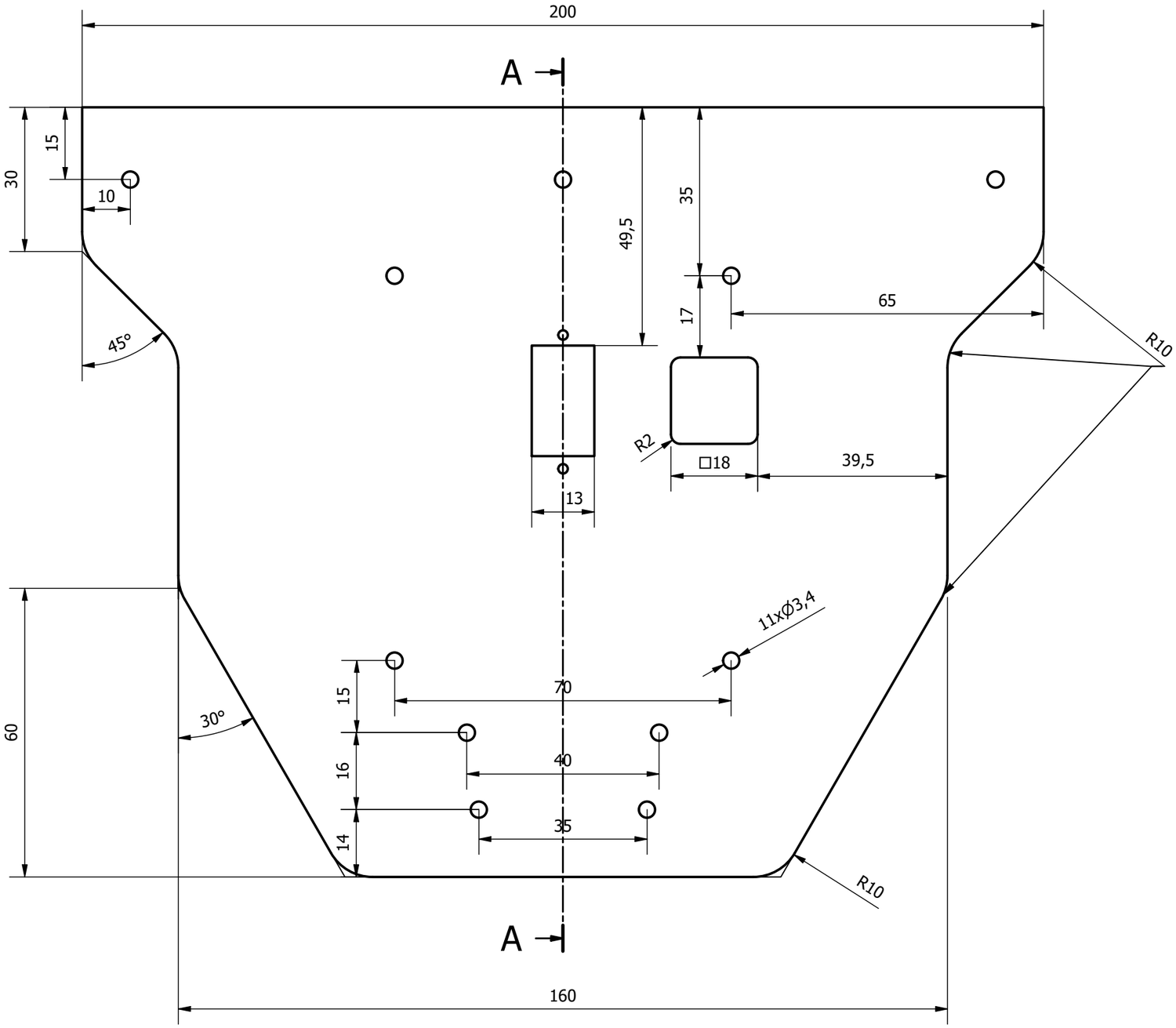
Plano: Ensamblaje

Autor: Rubén García Armero

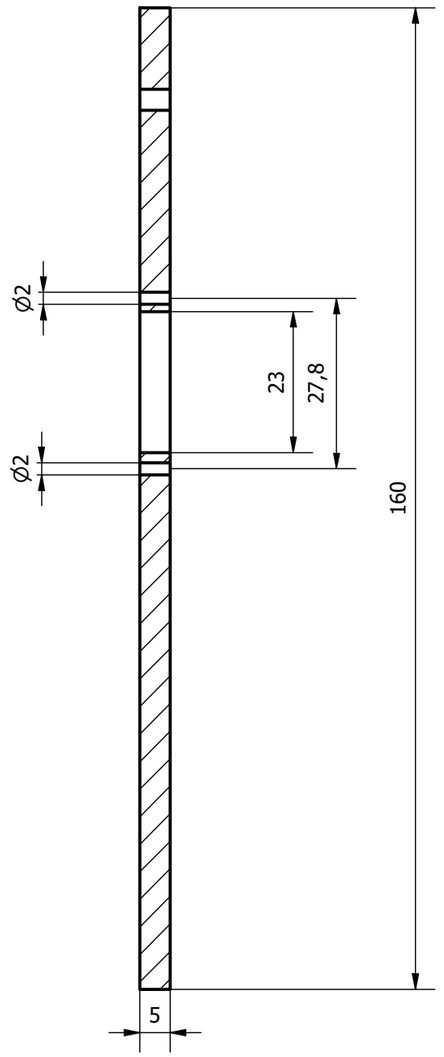
Fecha: Julio 2021

Escala: 1:1

Nº Plano: 1/17



A-A (1 : 1)



TRABAJO FINAL DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIERÍA INDUSTRIAL VALENCIA

Proyecto: DESARROLLO DE UN PROTOTIPO A ESCALA DE UN VEHÍCULO PARA PRUEBAS DE LABORATORIO MEDIANTE IMPRESIÓN 3D

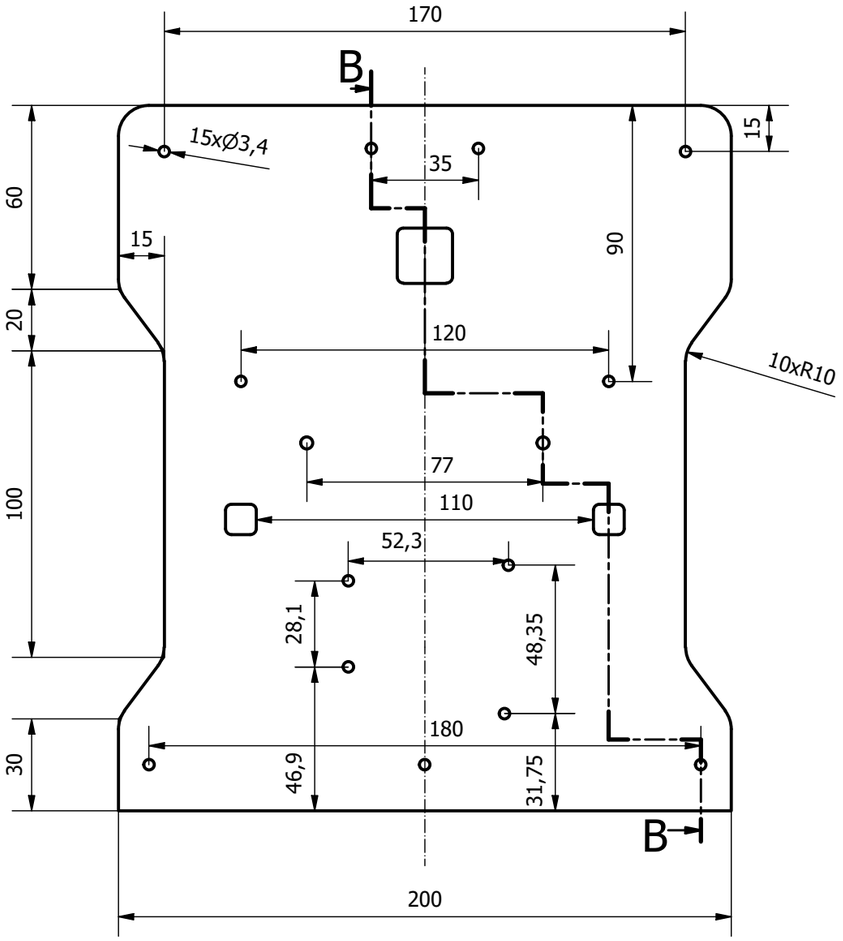
Plano: Base delantera

Autor: Rubén García Armero

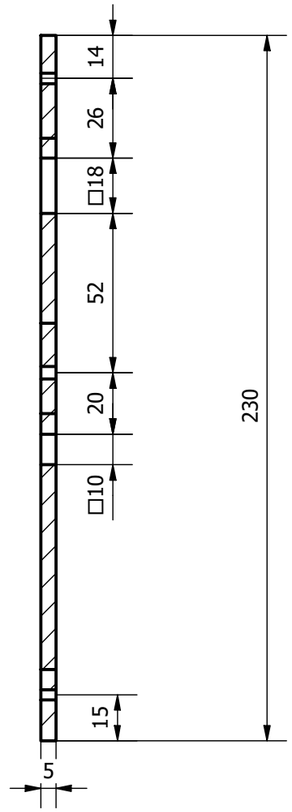
Fecha: Julio 2021

Escala: 1:1

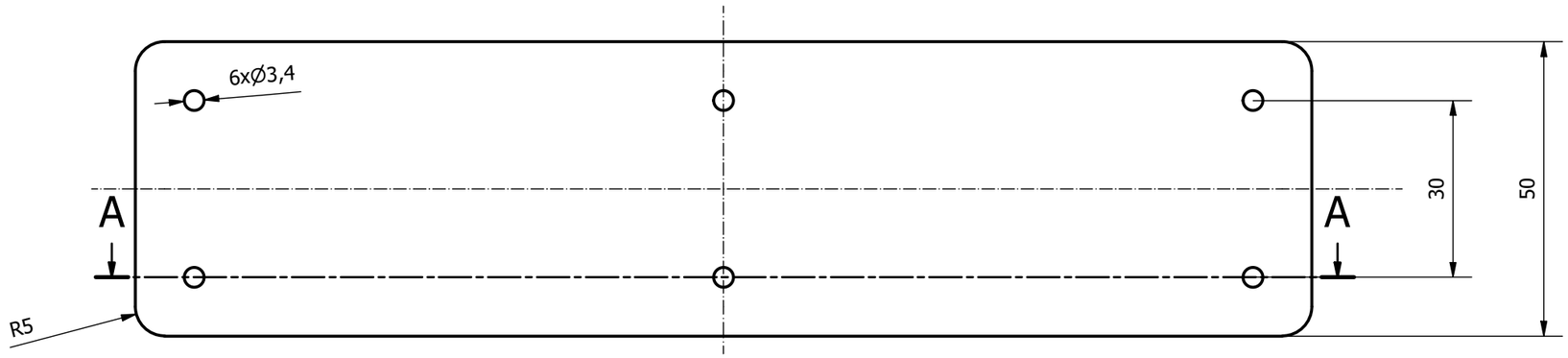
Nº Plano: 2/17



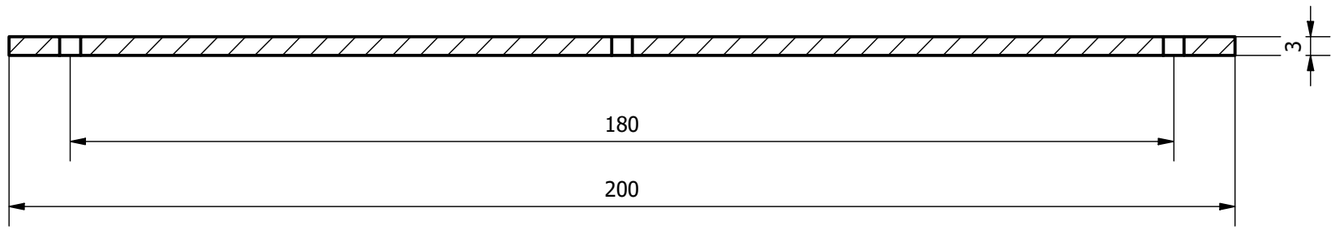
B-B (1 : 2)



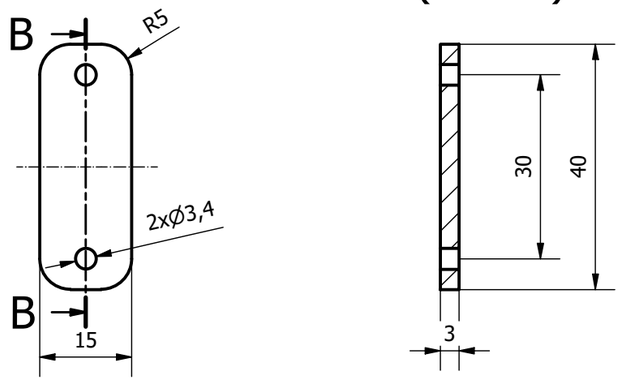
TRABAJO FINAL DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES UNIVERSITAT POLITÈCNICA DE VALÈNCIA ESCUELA TÉCNICA SUPERIOR INGENIERÍA INDUSTRIAL VALENCIA	Proyecto: DESARROLLO DE UN PROTOTIPO A ESCALA DE UN VEHÍCULO PARA PRUEBAS DE LABORATORIO MEDIANTE IMPRESIÓN 3D	Plano: Base trasera Autor: Rubén García Armero	Fecha: Julio 2021 Escala: 1:2	Nº Plano: 3/17



A-A (1 : 1)



B-B (1 : 1)



TRABAJO FINAL DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIERÍA INDUSTRIAL VALENCIA

Proyecto: DESARROLLO DE UN PROTOTIPO A ESCALA DE UN VEHÍCULO PARA PRUEBAS DE LABORATORIO MEDIANTE IMPRESIÓN 3D

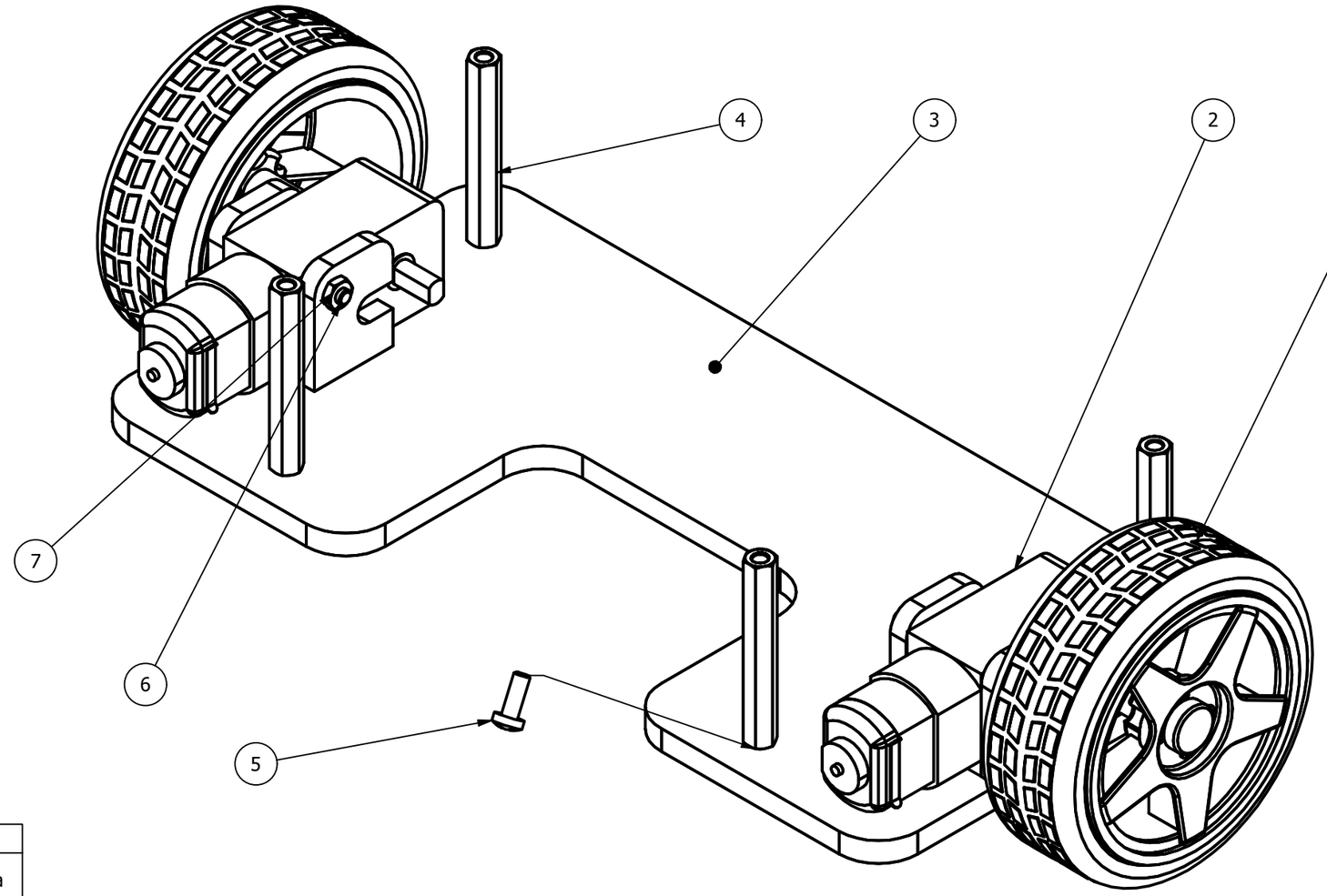
Plano: Pletina inferior y pletina superior

Autor: Rubén García Armero

Fecha: Julio 2021

Escala: 1:1

Nº Plano: 4/17



LISTADO DE PIEZAS		
Pieza	Unidades	Referencia numérica
Rueda	2	1
Motor de escobillas	2	2
Soporte motores	1	3
Separadores hexagonales M3x40	4	4
Tornillo M3x8	4	5
Tornillo M3x30	2	6
Tuerca M3	2	7

TRABAJO FINAL DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIERÍA INDUSTRIAL VALENCIA

Proyecto: DESARROLLO DE UN PROTOTIPO A ESCALA DE UN VEHÍCULO PARA PRUEBAS DE LABORATORIO MEDIANTE IMPRESIÓN 3D

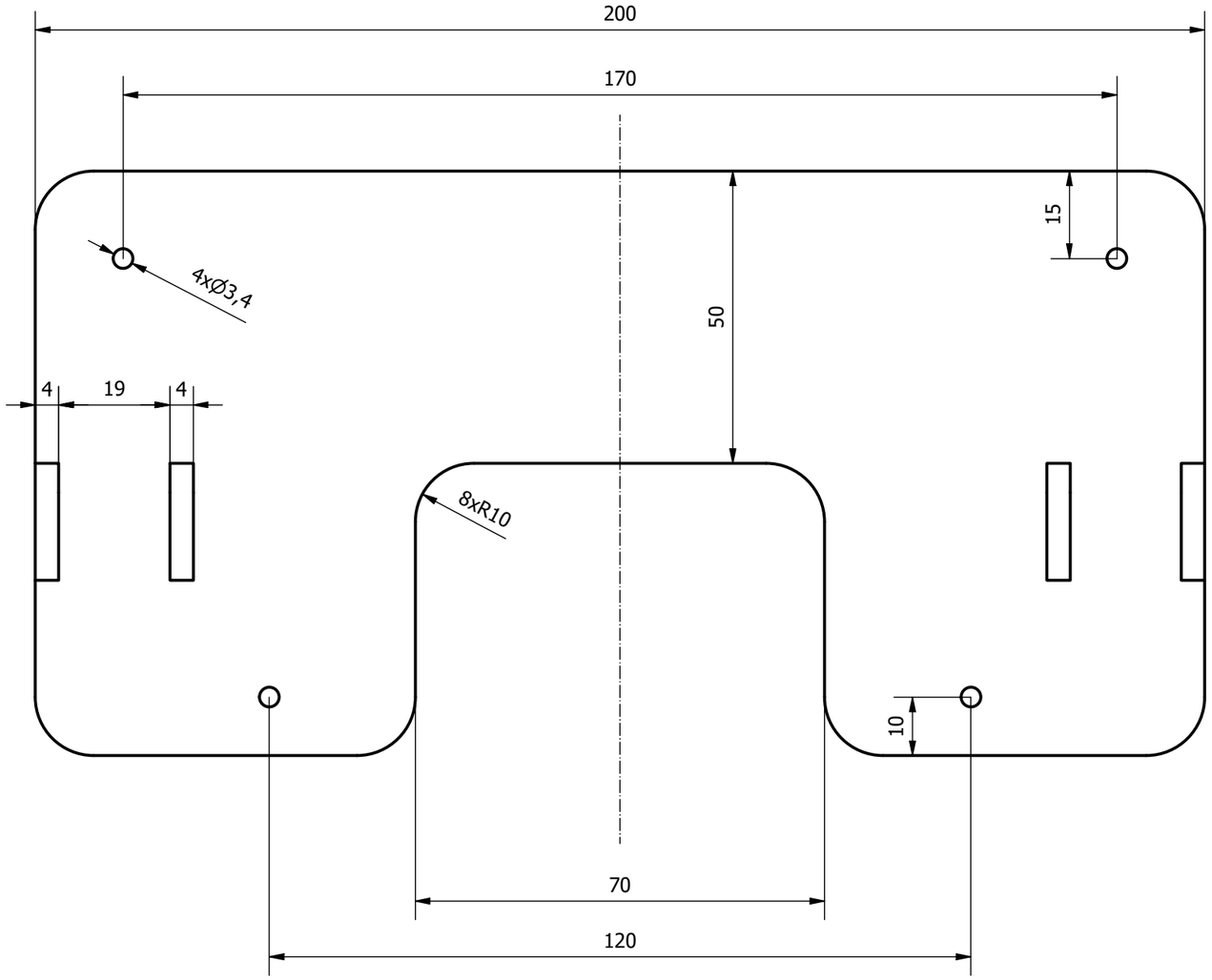
Plano: Tren trasero

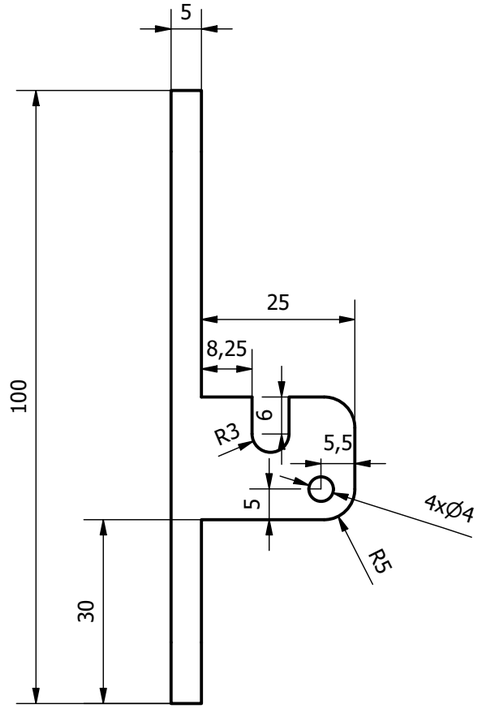
Autor: Rubén García Armero

Fecha: Julio 2021

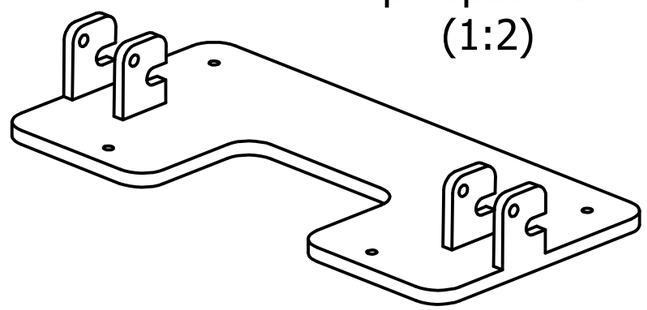
Escala: 1:1

Nº Plano: 5/17

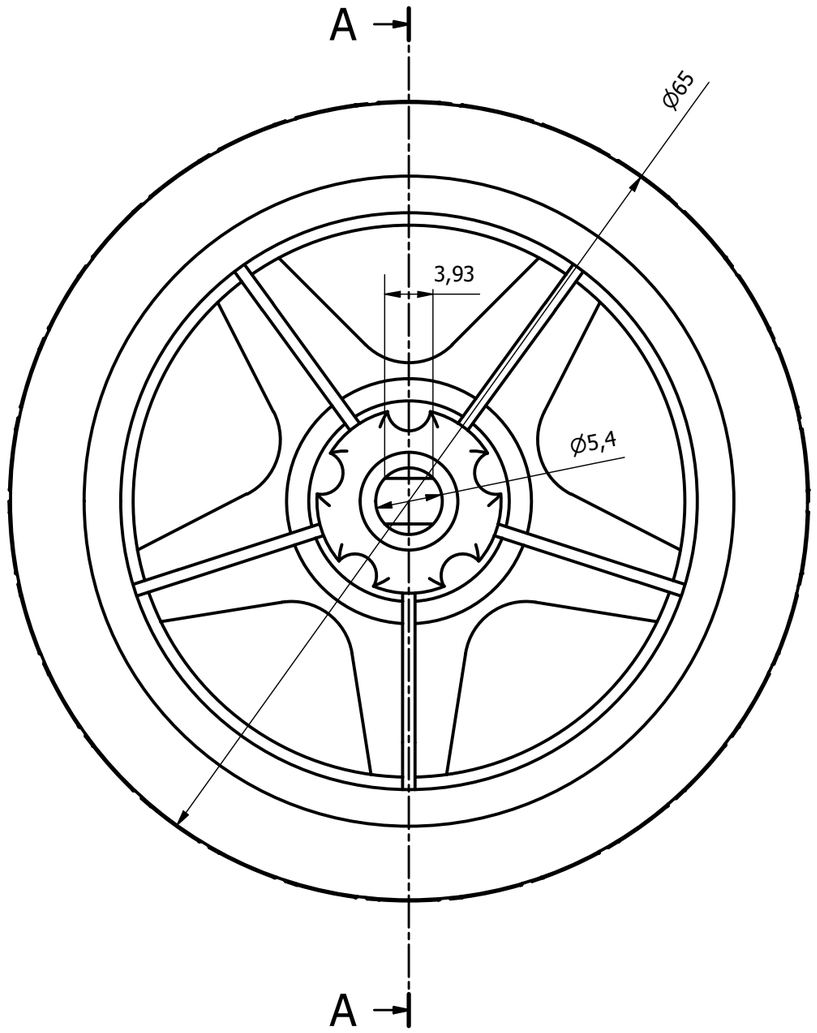




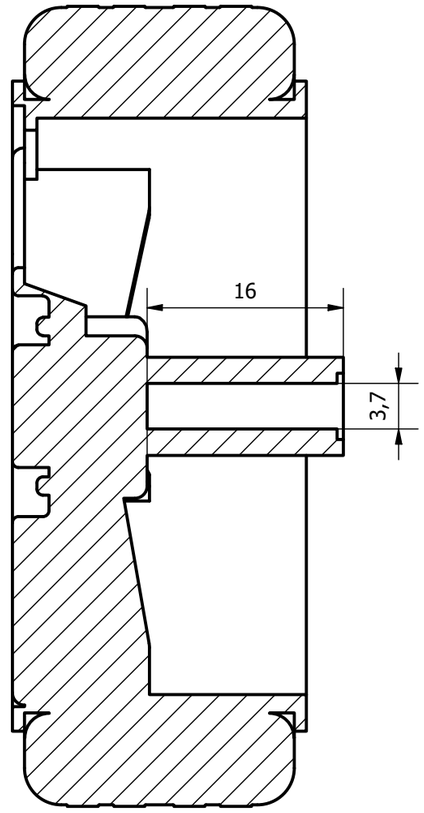
Vista en perspectiva (1:2)



TRABAJO FINAL DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES UNIVERSITAT POLITÈCNICA DE VALÈNCIA · ESCUELA TÉCNICA SUPERIOR INGENIERÍA INDUSTRIAL VALENCIA	Proyecto: DESARROLLO DE UN PROTOTIPO A ESCALA DE UN VEHÍCULO PARA PRUEBAS DE LABORATORIO MEDIANTE IMPRESIÓN 3D	Plano: Soporte motores Autor: Rubén García Armero	Fecha: Julio 2021 Escala: 1:1	Nº Plano: 6/17



A-A (2 : 1)



TRABAJO FINAL DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIERÍA INDUSTRIAL VALENCIA

Proyecto: DESARROLLO DE UN PROTOTIPO A ESCALA DE UN VEHÍCULO PARA PRUEBAS DE LABORATORIO MEDIANTE IMPRESIÓN 3D

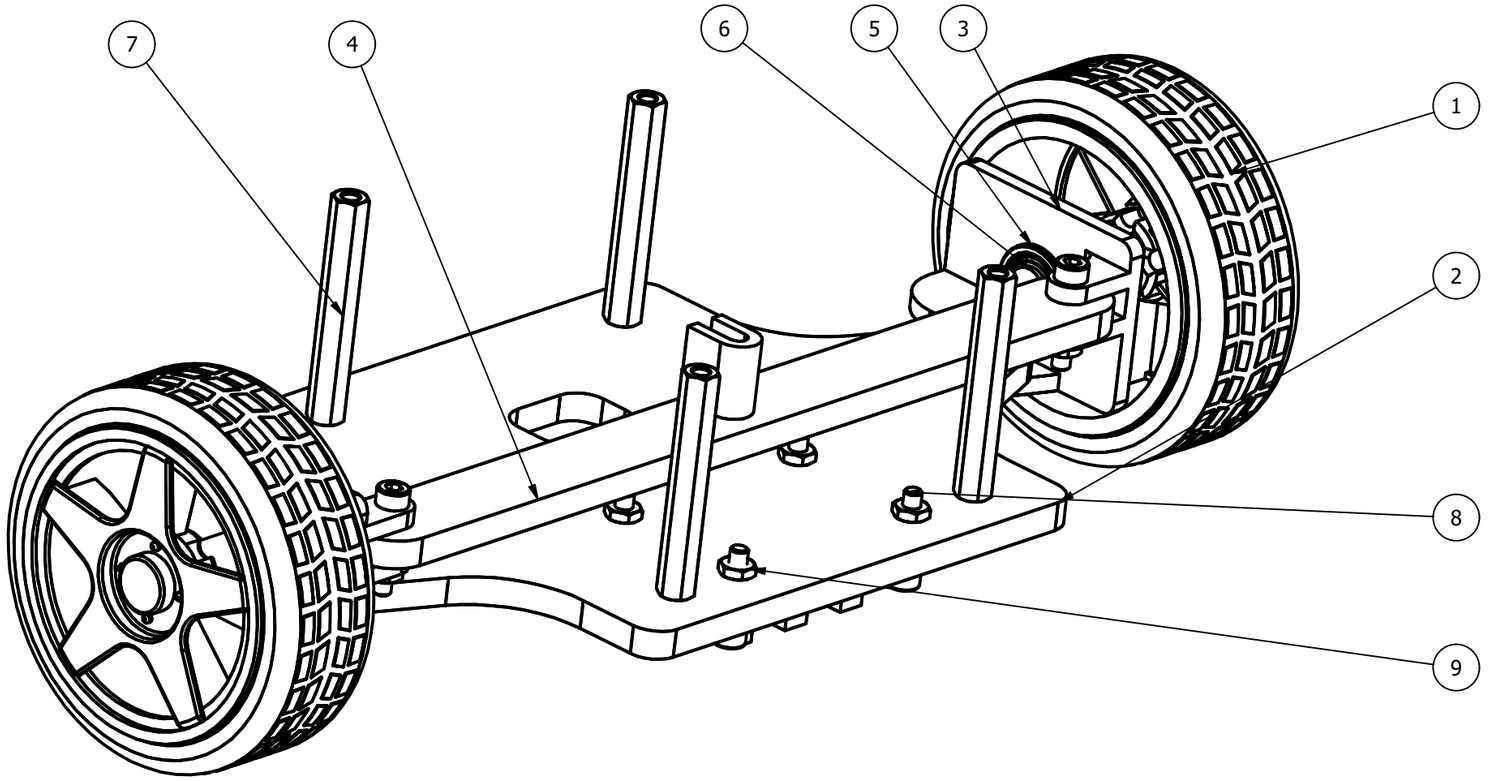
Plano: Rueda

Autor: Rubén García Armero

Fecha: Julio 2021

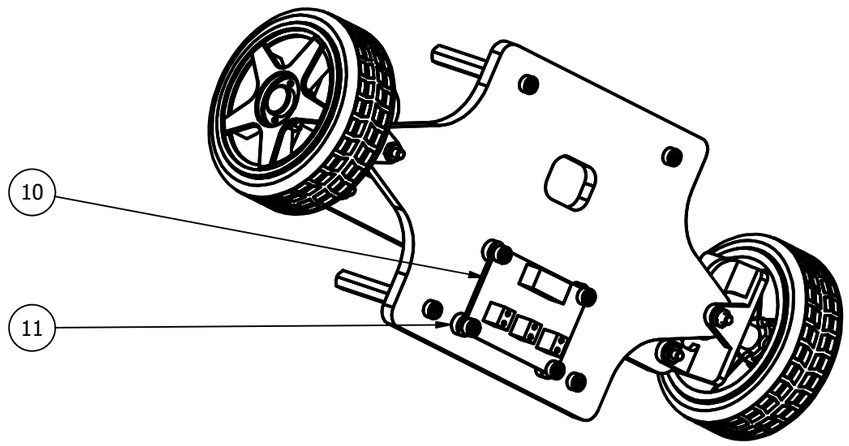
Escala: 2:1

Nº Plano: 7/17



LISTADO DE PIEZAS		
Pieza	Cantidad	Identificador
Rueda	2	1
Soporte base delantera	1	2
Eje de dirección	1	3
Soporte Ruedas	2	4
Rodamiento	2	5
Eje de la rueda	2	6
Separadores hexagonales M3x40	4	7
Tornillos M3x16	12	8
Tuercas M3	8	9
Sensor de seguimiento de línea	1	10
Separadores M3x3	4	11

Vista inferior
(1:2)



TRABAJO FINAL DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIERÍA INDUSTRIAL VALENCIA

Proyecto: DESARROLLO DE UN PROTOTIPO A ESCALA DE UN VEHÍCULO PARA PRUEBAS DE LABORATORIO MEDIANTE IMPRESIÓN 3D

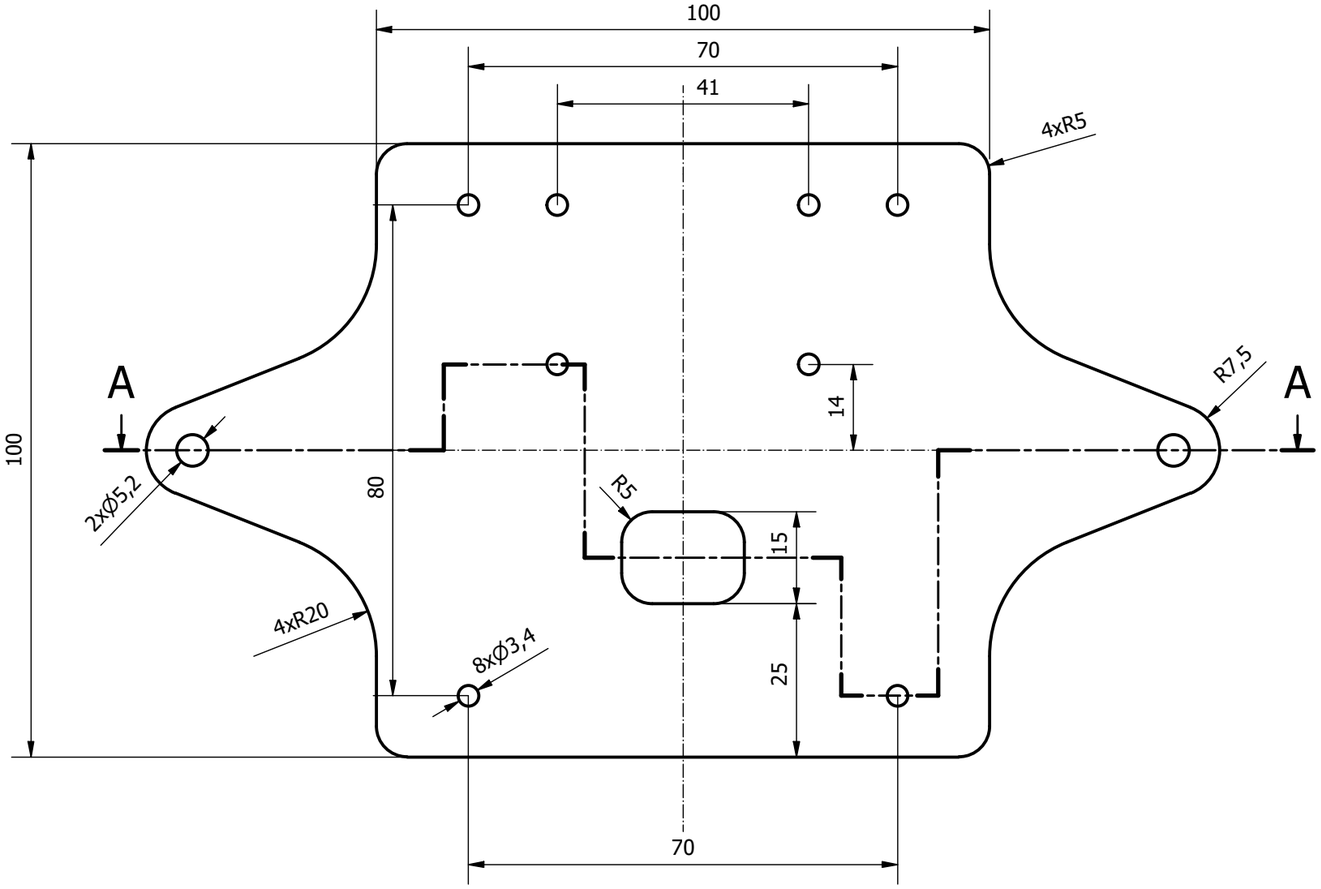
Plano: Tren delantero

Autor: Rubén García Armero

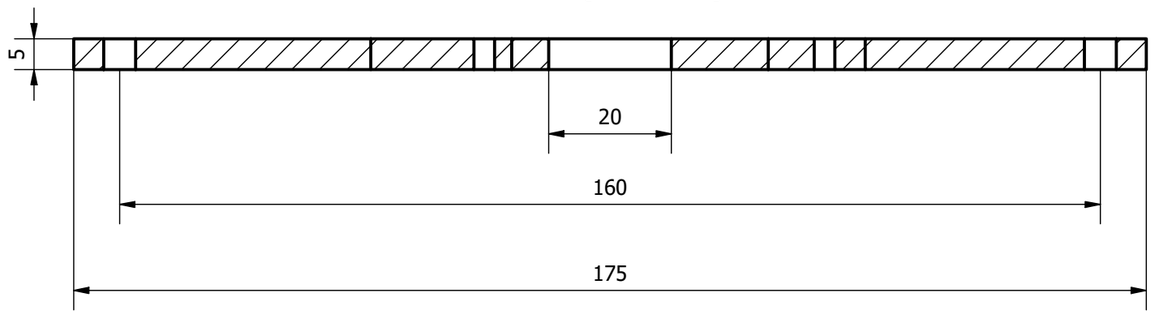
Fecha: Julio 2021

Escala: 1:1

Nº Plano: 8/17



A-A (1 : 1)



TRABAJO FINAL DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIERÍA INDUSTRIAL VALENCIA

Proyecto: DESARROLLO DE UN PROTOTIPO A ESCALA DE UN VEHÍCULO PARA PRUEBAS DE LABORATORIO MEDIANTE IMPRESIÓN 3D

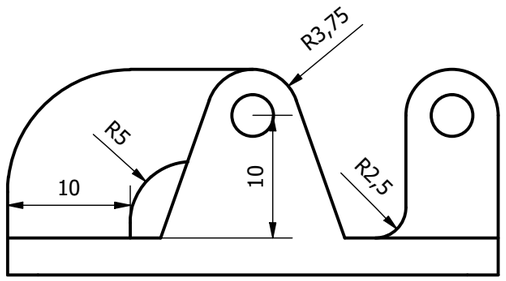
Plano: Soporte de la base delantera

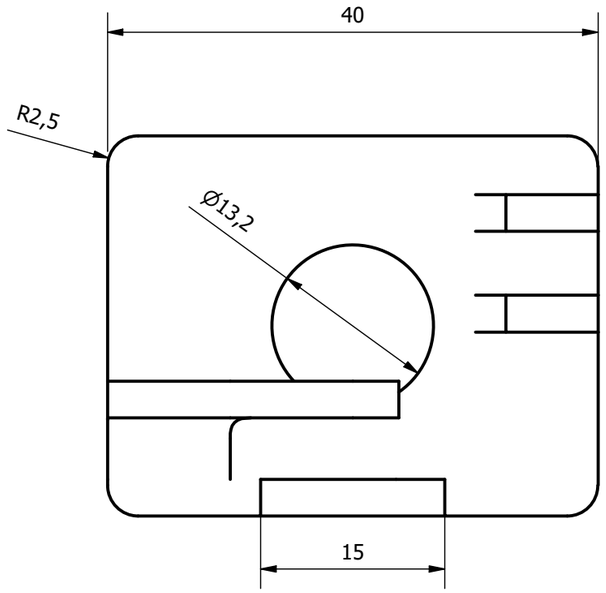
Autor: Rubén García Armero

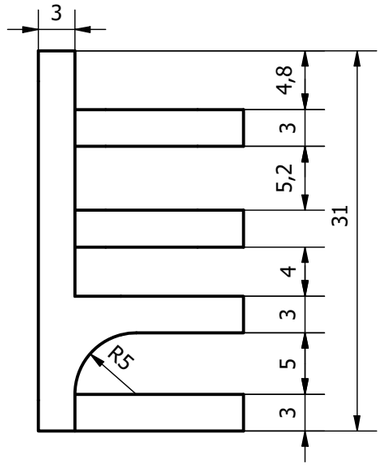
Fecha: Julio 2021

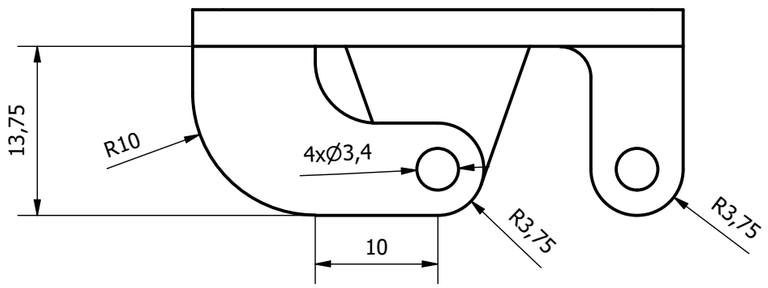
Escala: 1:1

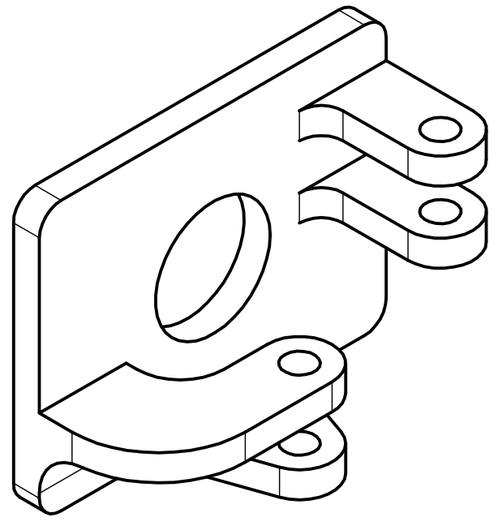
Nº Plano: 9/17



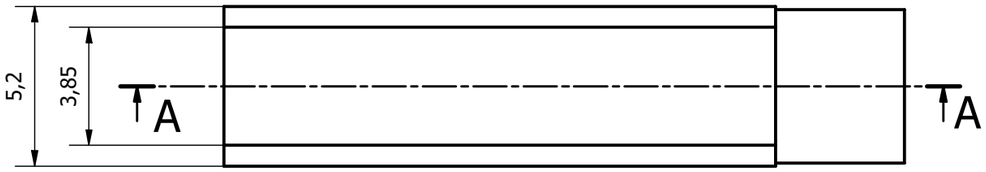


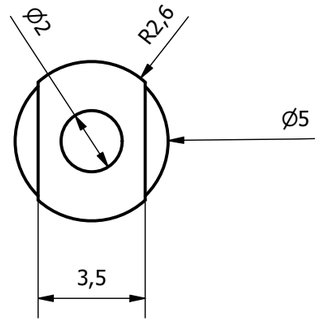


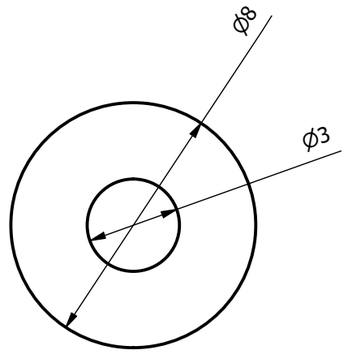




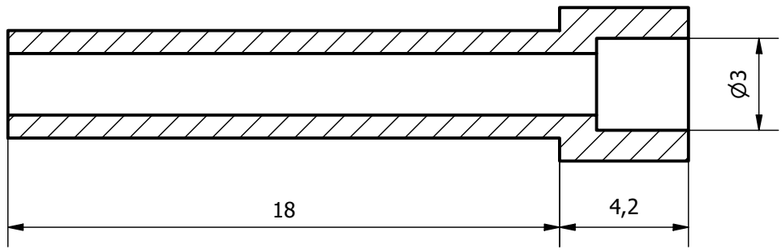
TRABAJO FINAL DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES UNIVERSITAT POLITÈCNICA DE VALÈNCIA ESCUELA TÉCNICA SUPERIOR INGENIERÍA INDUSTRIAL VALENCIA	Proyecto: DESARROLLO DE UN PROTOTIPO A ESCALA DE UN VEHÍCULO PARA PRUEBAS DE LABORATORIO MEDIANTE IMPRESIÓN 3D	Plano: Soporte ruedas Autor: Rubén García Armero	Fecha: Julio 2021 Escala: 2:1	Nº Plano: 10/17

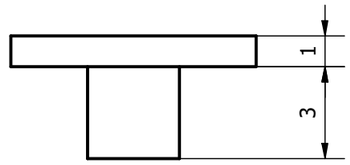


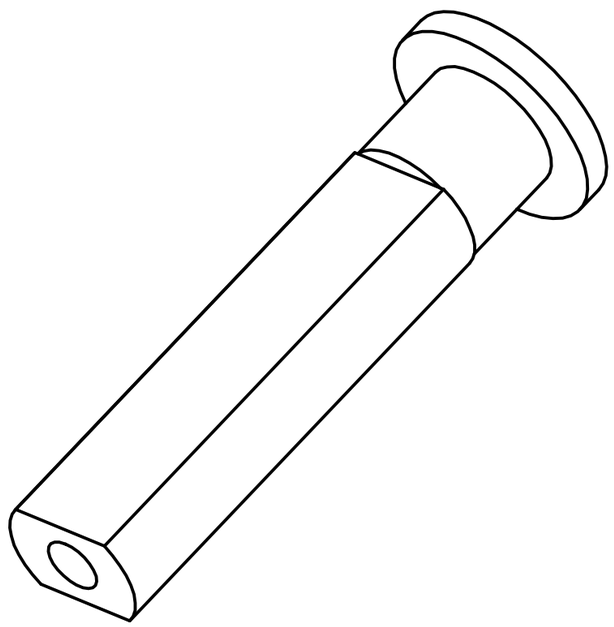




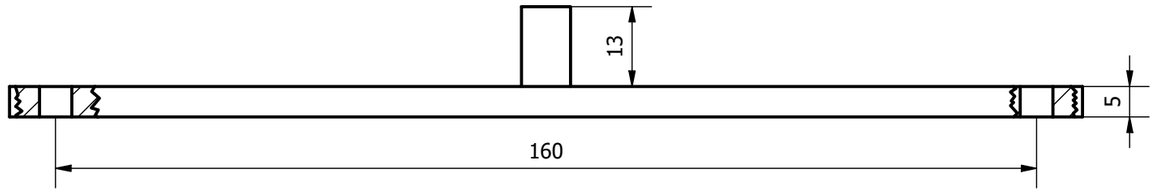
A-A (5 : 1)

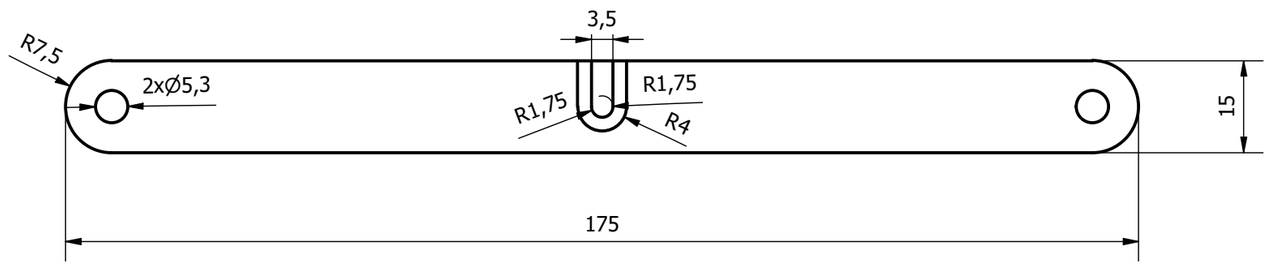






TRABAJO FINAL DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES UNIVERSITAT POLITÈCNICA DE VALÈNCIA ESCUELA TÉCNICA SUPERIOR INGENIERÍA INDUSTRIAL VALENCIA	Proyecto: DESARROLLO DE UN PROTOTIPO A ESCALA DE UN VEHÍCULO PARA PRUEBAS DE LABORATORIO MEDIANTE IMPRESIÓN 3D	Plano: Eje de las ruedas Autor: Rubén García Armero	Fecha: Julio 2021 Escala: 5:1	Nº Plano: 11/17





TRABAJO FINAL DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIERÍA INDUSTRIAL VALENCIA

Proyecto: DESARROLLO DE UN PROTOTIPO A ESCALA DE UN VEHÍCULO PARA PRUEBAS DE LABORATORIO MEDIANTE IMPRESIÓN 3D

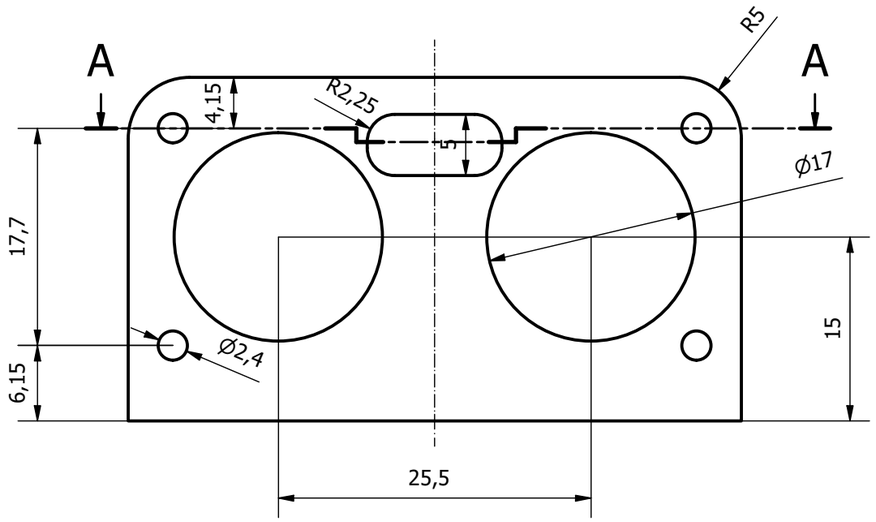
Plano: Eje de dirección

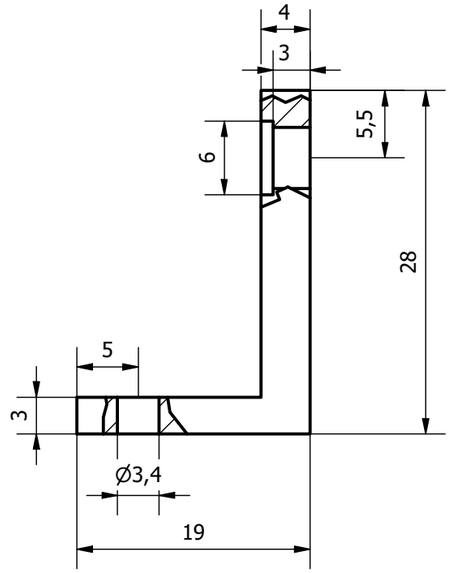
Autor: Rubén García Armero

Fecha: Julio 2021

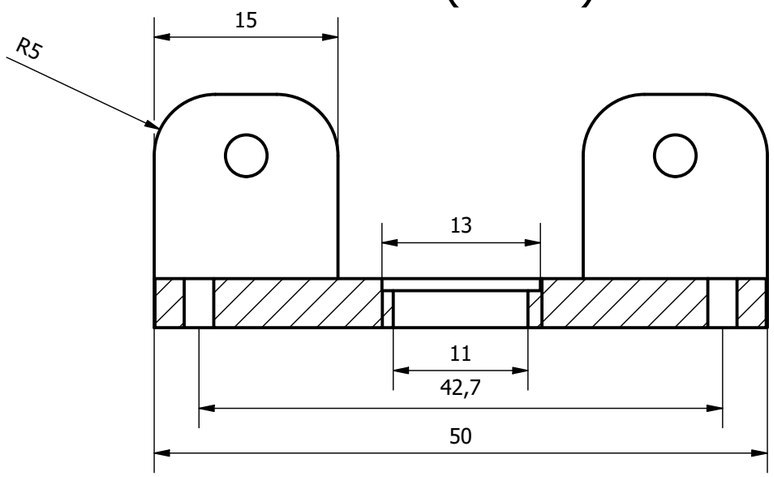
Escala: 1:1

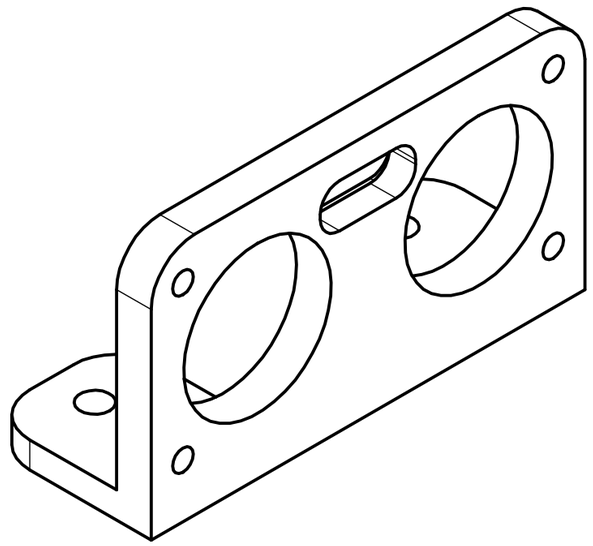
Nº Plano: 12/17



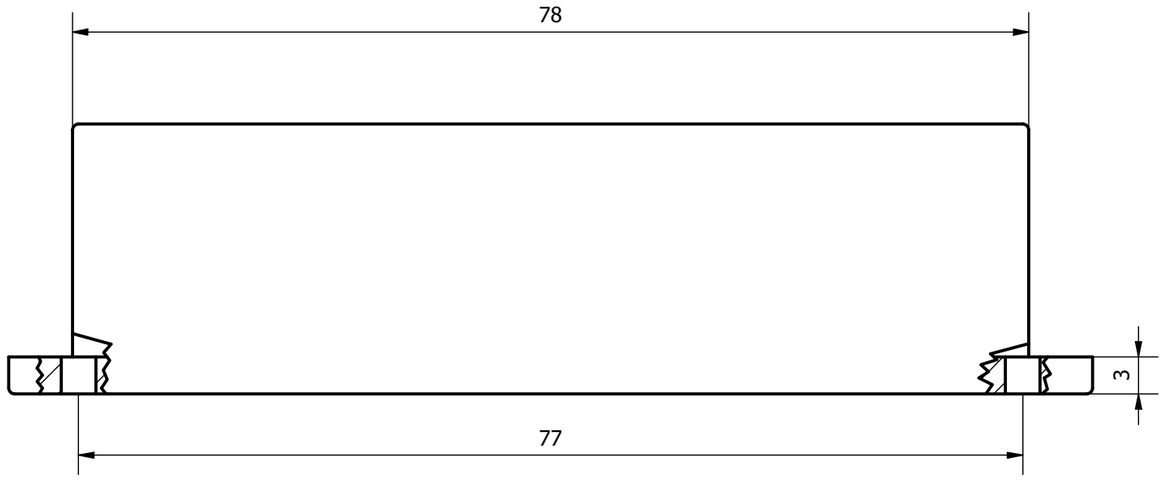


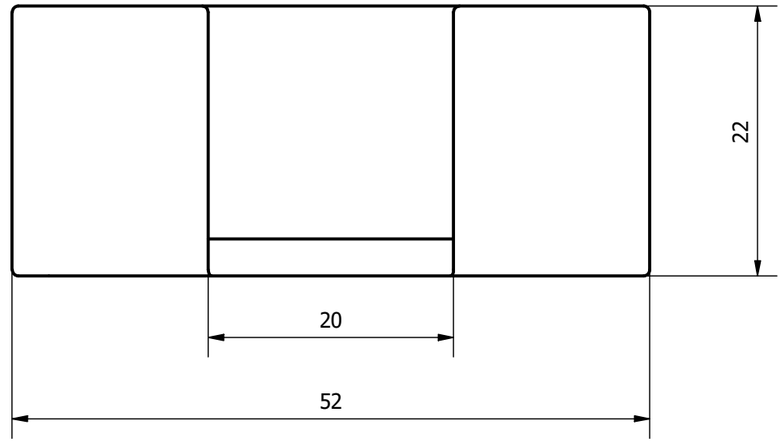
A-A (2 : 1)

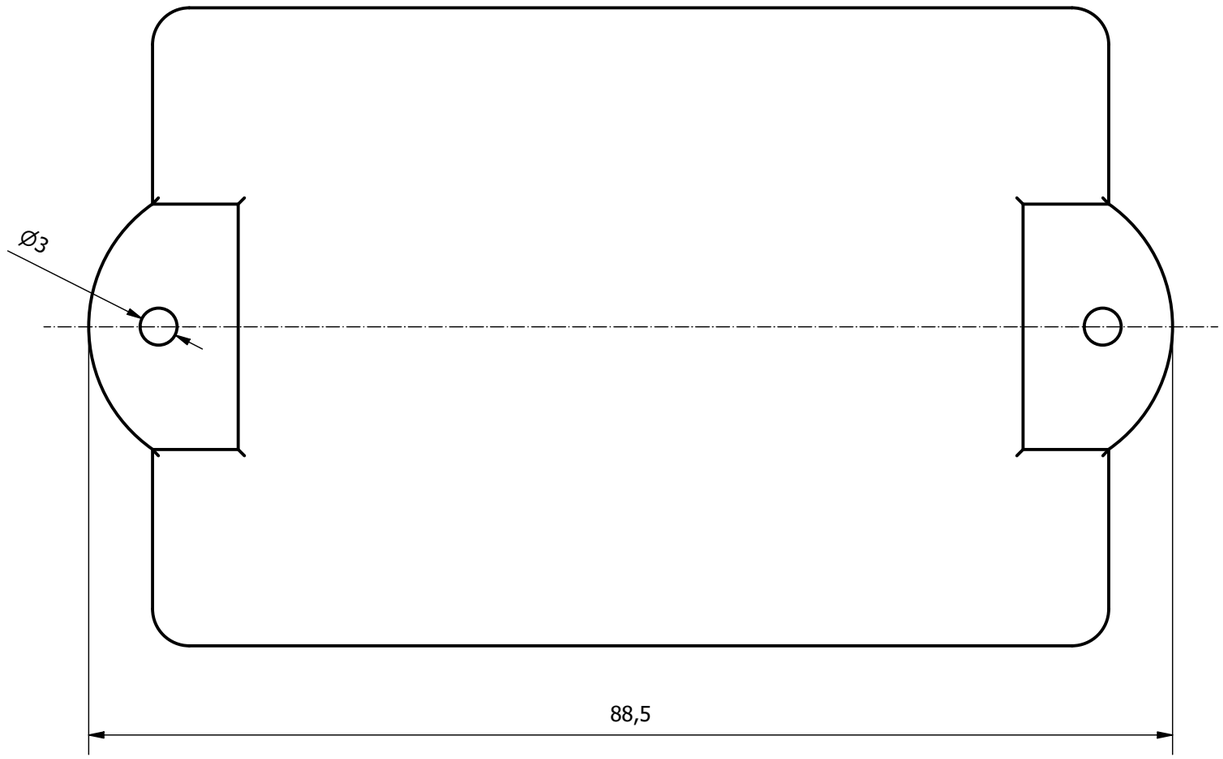




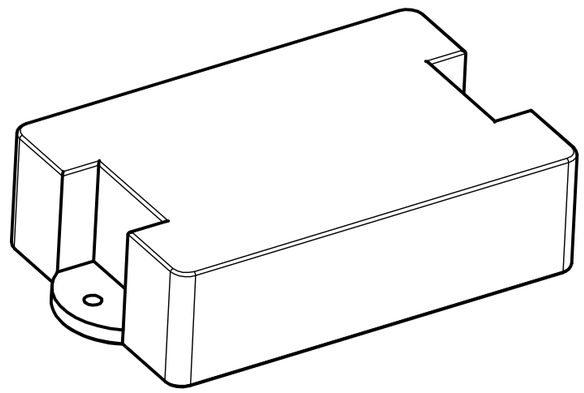
TRABAJO FINAL DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES UNIVERSITAT POLITÈCNICA DE VALÈNCIA — ESCUELA TÉCNICA SUPERIOR INGENIERÍA INDUSTRIAL VALENCIA	Proyecto: DESARROLLO DE UN PROTOTIPO A ESCALA DE UN VEHÍCULO PARA PRUEBAS DE LABORATORIO MEDIANTE IMPRESIÓN 3D	Plano: Soporte sensor ultrasonidos Autor: Rubén García Armero	Fecha: Julio 2021 Escala: 2:1	Nº Plano: 13/17







Vista en perspectiva
(1:1)



TRABAJO FINAL DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIERÍA INDUSTRIAL VALENCIA

Proyecto:
DESARROLLO DE UN PROTOTIPO A ESCALA DE UN VEHÍCULO PARA PRUEBAS DE LABORATORIO MEDIANTE IMPRESIÓN 3D

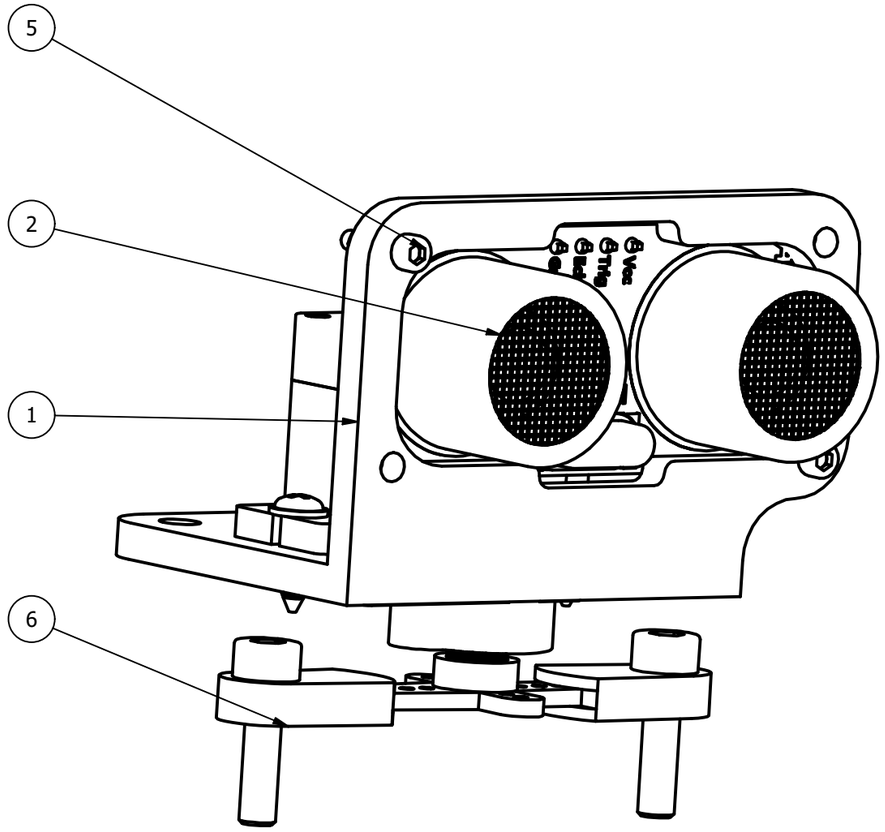
Plano: Batería

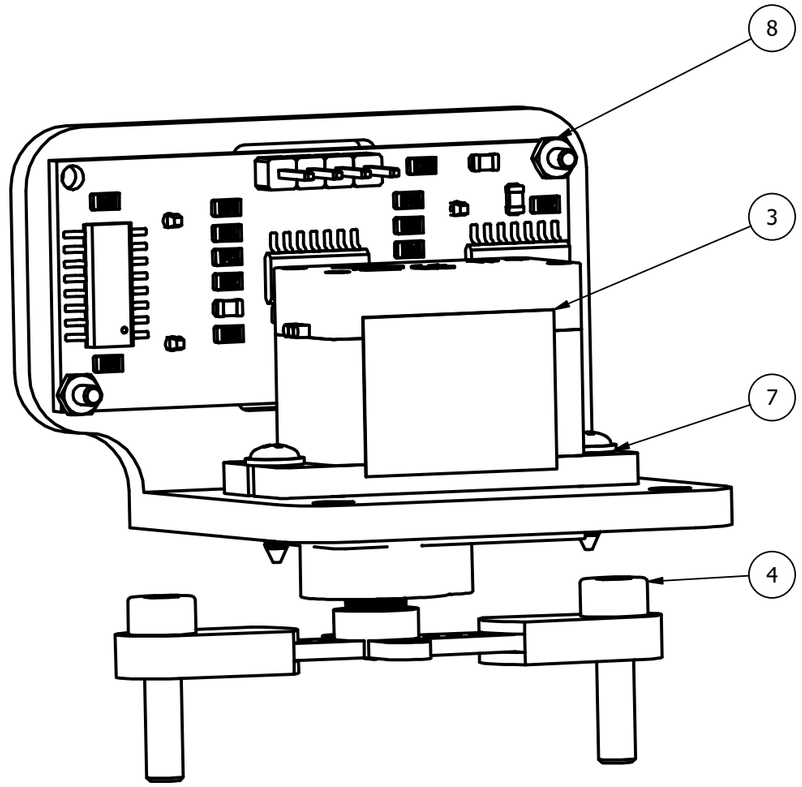
Autor: Rubén García Armero

Fecha: Julio 2021

Escala: 2:1

Nº Plano: 14/17





LISTADO DE PIEZAS		
Pieza	Unidades	Referencia numérica
Soporte sensor-servo	1	1
Sensor HC-SR04	1	2
Servomotor SG90	1	3
Tornillo M3x12	2	4
Tornillo M1.6x8	2	5
Funda servomotor	2	6
Tornillo M2x8	2	7
Tuerca M1.6	2	8

TRABAJO FINAL DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIERÍA INDUSTRIAL VALENCIA

Proyecto: DESARROLLO DE UN PROTOTIPO A ESCALA DE UN VEHÍCULO PARA PRUEBAS DE LABORATORIO MEDIANTE IMPRESIÓN 3D

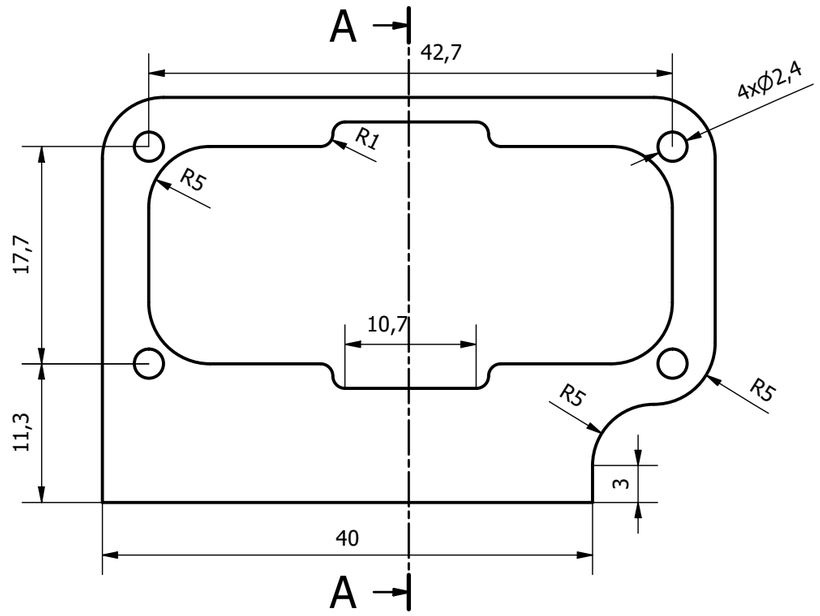
Plano: Conjunto Servo-Sensor Ultrasonidos

Autor: Rubén García Armero

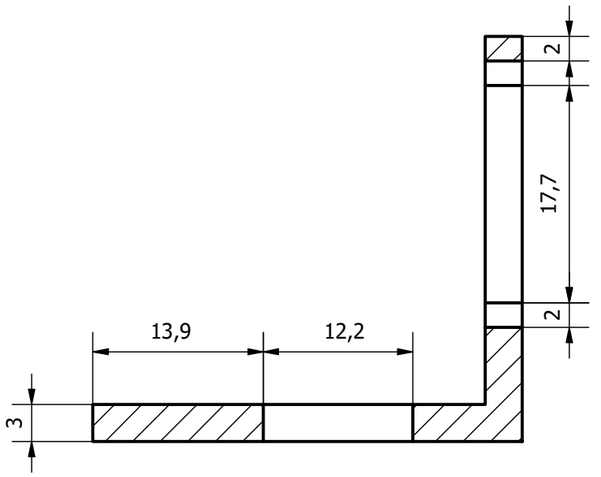
Fecha: Julio 2021

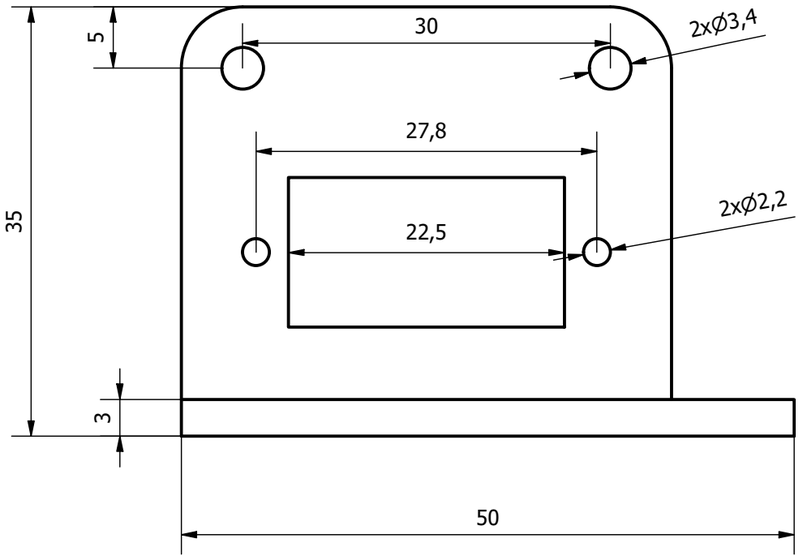
Escala: 2:1

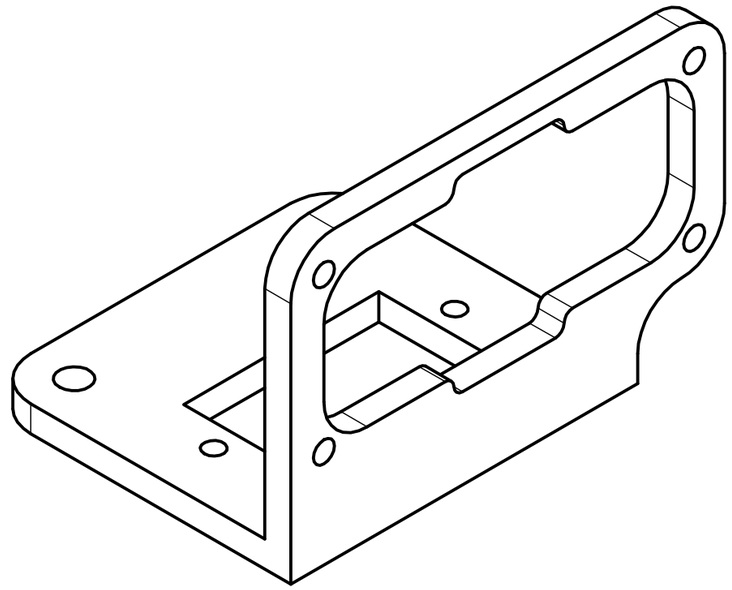
Nº Plano: 15/17



A-A (2:1)







TRABAJO FINAL DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIERÍA INDUSTRIAL VALENCIA

Proyecto:
DESARROLLO DE UN PROTOTIPO A ESCALA DE UN VEHÍCULO PARA PRUEBAS DE LABORATORIO MEDIANTE IMPRESIÓN 3D

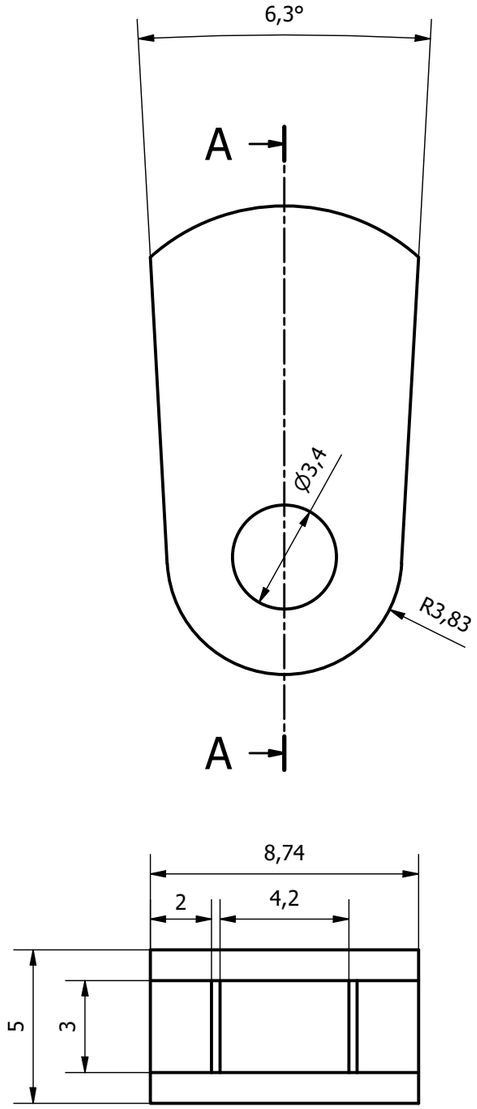
Plano: Soporte sensor ultrasonidos-servo

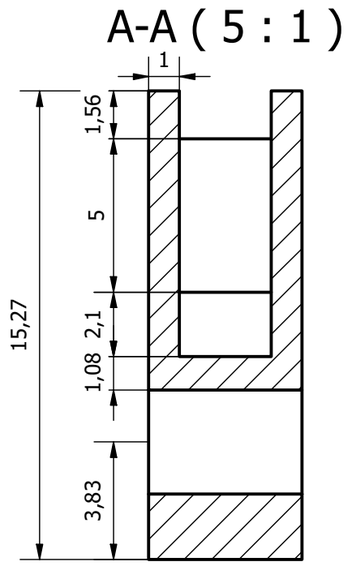
Autor: Rubén García Armero

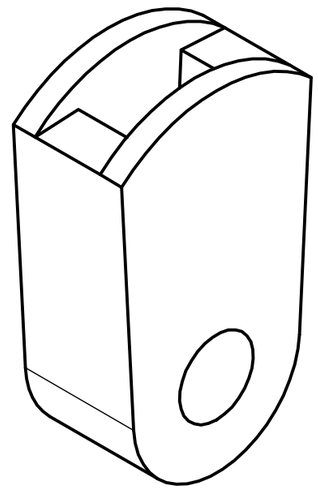
Fecha: Julio 2021

Escala: 2:1

Nº Plano: 16/17







TRABAJO FINAL DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIERÍA INDUSTRIAL VALENCIA

Proyecto: DESARROLLO DE UN PROTOTIPO A ESCALA DE UN VEHÍCULO PARA PRUEBAS DE LABORATORIO MEDIANTE IMPRESIÓN 3D

Plano: Funda servo

Autor: Rubén García Armero

Fecha: Julio 2021

Escala: 5:1

Nº Plano: 17/17