

INSTITUTO TECNOLÓGICO DE BUENOS AIRES – ITBA

ESCUELA DE INGENIERÍA Y TECNOLOGÍA

PLATAFORMA INTELIGENTE PARA DESARROLLO DE VEHÍCULOS MÓVILES AUTÓNOMOS

AUTOR/ES: **Jurnet Berteloot, Alan (Leg. N° 51.162)**

PROFESOR RESPONSABLE / TUTOR: **Lerendegui, Norberto Marcelo**

TRABAJO FINAL PRESENTADO PARA LA OBTENCIÓN DEL TÍTULO DE INGENIERO MECÁNICO

BUENOS AIRES

PRIMERO Y SEGUNDO CUATRIMESTRE, 2017

Resumen

El presente informe detalla el proyecto final de la carrera de Ingeniería Mecánica con Focalización Mecatrónica “Plataforma inteligente para desarrollo vehículos móviles autónomos”. El proyecto fue desarrollado dentro del marco de la asignatura 31.58 Proyecto Mecatrónico entre marzo de 2017 y julio de 2018.

El informe comienza con una introducción a la temática que pone en contexto y justifica el diseño realizado a partir de una investigación sobre distintas plataformas de desarrollo. Parte del proyecto consiste en determinar las especificaciones de la plataforma en base a las necesidades detectadas durante la investigación, las cuales se acordaron en conjunto con la cátedra de proyecto mecatrónico. El objetivo principal del proyecto es alcanzar y en lo posible superar los requerimientos especificados inicialmente.

Luego de definido el alcance del proyecto, se describen propuestas que cumplen con los objetivos planteados. Inmediatamente después se presentan los módulos en los cuales se divide la plataforma, justificando todas las decisiones que se fueron tomando a lo largo del proyecto.

Posteriormente, se detalla la fabricación de la plataforma, donde se explica su implementación justificando las decisiones tomadas a partir de los métodos constructivos utilizados. Luego se explica la estructura modular de software encargada de controlar la plataforma, detallando y justificando decisiones de hardware.

La siguiente sección muestra los resultados de distintos ensayos bajo distintas condiciones de entorno.

Por último, se extraen las conclusiones del proyecto, donde en primer lugar se realiza un análisis de objetivos cumplidos, luego se plantean distintos puntos de mejora y cuáles serían los siguientes pasos para desarrollar una nueva versión de la plataforma.

Índice

1	Introducción	7
2	Investigación	8
2.1	Estructura	9
2.1.1	Dos ruedas paralelas con el mismo eje	9
2.1.2	Tres ruedas	11
2.1.3	Cuatro ruedas	12
2.1.3.1	Auto a escala comercial de tipo modelismo modificado	12
2.1.3.2	Modelo impreso 3D de auto modelismo.	14
2.1.4	Ventajas:	14
2.1.5	Desventajas:	14
2.1.6	Conclusiones	15
2.2	Hardware y software	15
2.2.1	ARMV7 – ROS	16
2.2.2	ARM – PYTHON/C++	17
2.2.3	ARM +	17
2.2.4	Microcontrolador	17
2.2.5	Arduino	17
2.2.6	Conclusión	18
3	Objetivos	19
4	Plataforma	19
5	Diseño mecánico	20
5.1	Definiciones básicas	20
5.1.1	Materiales	20
5.2	Diseño preliminar	21
5.2.1	Chasis	21
5.2.2	Módulo de tracción:	21
5.2.3	Módulo frontal:	21
5.2.4	Módulo de alimentación:	21
5.2.5	Módulo de control:	21
5.2.5.1	Core	22
5.2.5.2	Interfaz de sensores y actuadores	22

5.2.5.3	Drivers	22
5.2.5.4	Sensores:	22
5.2.5.5	Actuadores:	22
5.2.5.6	Comunicación	22
5.3	Diseño	23
5.4	Primera iteración de diseño	24
5.4.1	Problemas en el diseño y análisis	24
5.5	Segunda iteración de diseño	24
5.5.1	Problemas de diseño y análisis	25
5.6	Tercera iteración de diseño	25
5.7	Cuarta iteración de diseño	26
5.7.1	Dirección	27
5.7.1.1.1	Análisis del mecanismo	27
5.7.1.2	Unidades mínimas	28
5.7.1.3	Puntos límite de desplazamiento	29
5.7.1.4	Ángulos de transmisión	30
5.7.1.5	Análisis de fuerzas	32
5.7.1.6	Conclusión	34
5.7.2	Tracción	35
5.7.2.1	Motores brushless	35
5.7.2.2	Diferencial	36
5.7.2.3	Diseño	39
5.7.3	Chasis	41
5.7.4	Módulo de sensado (Radar)	41
5.7.5	Ensamble completo y módulo de control	43
6	Diseño electrónico	45
6.1	Arquitectura electrónica	45
6.1.1	Módulo central	45
6.1.2	Módulo de radar	46
6.1.3	Módulo de control	46
6.1.4	Modulo remoto	47
6.1.5	Arquitectura completa	48
7	Modelo	49

7.1	Modelo cinemático de la planta	49
7.2	Modelo de actuadores	50
7.2.1	Motor – Control de velocidad	50
7.2.2	Servo	51
7.3	Dead reckoning	52
7.3.1	Giroscopio	53
7.3.2	Encoder	54
8	Control de objetivo y detección de obstáculos	54
8.1	Control de objetivo	54
8.2	Detección de obstáculos	56
8.2.1.1	Sensores	56
8.2.2	Visual weight vector Estimation	58
8.3	Control de objetivos + detección de obstáculos	60
8.4	Arquitectura de software	61
8.4.1	Módulo central	62
8.4.2	Módulo de radar	63
8.4.3	Módulo de control	63
8.4.4	Módulo remoto	64
9	Fabricación	65
9.1	Impresión 3D	65
9.1.1	Consideraciones de diseño	65
9.1.2	Configuración de las impresiones	66
9.2	Electrónica	67
9.2.1	Módulo central	67
9.2.1.1	MPU9250	67
9.2.1.2	RF24	68
9.2.1.3	Arduino nano	69
9.2.1.4	Batería	69
9.2.1.5	Esc+Motor	70
9.2.1.6	Fuente step down	72
9.2.1.7	Encoder	73
9.2.1.8	Servo de dirección	74
9.2.2	Módulo de radar	75

9.2.2.1 Sensor Sharp GP2Y0A02YK0F	75
9.2.3 Módulo de control	76
10 Pruebas realizadas	77
10.1 Control	77
10.2 Comunicación	78
10.3 Tracción	78
10.4 Radar	79
10.5 Dirección	79
11 Resultados	80
11.1 Comandos	80
11.2 Tracción	80
11.3 Dirección	81
11.4 Chasis	81
11.5 Radar	82
11.6 Control	82
12 Conclusiones generales	83
13 Posibles mejoras	85
13.1 Mecánica	85
13.2 Electrónica	85
13.3 Software	86
14 Bibliografía	87
15 Anexos	88
15.1 Reproducción de la plataforma	88
15.2 Mecánica	88
15.2.1 Dirección	90
15.2.2 Tracción	91
15.2.3 Módulo de chasis	92
15.2.4 Módulo de radar	93
15.2.5 Módulo de control	94
15.2.6 Modulo remoto	95
15.2.7 Ensamble completo	96
15.3 Electrónica	97
15.3.1.1 Modulo central	97

15.3.1.2	Modulo radar	99
15.3.1.3	Módulo de control	100
15.3.1.4	Modulo remoto	101
15.4	Software	103
15.4.1	Comunicación	103
15.4.2	Uso de comandos – Matlab	112

1 INTRODUCCIÓN

Existe actualmente una iniciativa global en el tema de transporte orientada a reemplazar los vehículos motorizados y controlados manualmente por vehículos eléctricos autocontrolados. Distintas instituciones y empresas han desarrollado y continúan desarrollando distintas plataformas con el objetivo de investigar estrategias de control para luego implementarlas en vehículos comerciales.

Cada agente innovador se especializa en un modelo particular a automatizar; por ejemplo, existen plataformas que imitan autos comerciales y otras más simples. En general todas están compuestas por dos elementos igual de importantes: por un lado, el modelo mecánico del vehículo y, por el otro, el sistema de control del modelo.

En las sociedades actuales existen variados problemas asociados al incremento poblacional. Uno de los más graves y que demanda una pronta solución es el transporte. El crecimiento exponencial de la población se refleja en el crecimiento, igualmente exponencial, de la cantidad de vehículos comercializados en el mercado, generando problemas crecientes de abastecimiento de combustible y de congestiones en las vías.

Estos problemas fomentaron el nacimiento de cuatro tecnologías disruptivas: “diverse mobility”, “autonomous driving”, “electrification” and “connectivity”. Cada una de ellas trabajando en conjunto o individualmente, busca hacer más eficiente el transporte. Actualmente la necesidad de implementar estas tecnologías excede la capacidad de investigadores para desarrollarlas y se vuelve imperiosa una pronta solución.

Por otro lado, en estos últimos años surgieron distintos entornos de desarrollo masivos que facilitan la investigación y amplían el acceso a la última tecnología. Por ejemplo, entornos de desarrollo como Arduino, Raspberry Pi y plataformas de intercambio como GitHub potencian el desarrollo de nuevas ideas, expandiendo el conocimiento a lugares de acceso limitado a la tecnología.

Con el objetivo de hacer un aporte al desarrollo global de estas tecnologías, se buscó desarrollar una plataforma de bajo costo basada en entornos “open-source” que pudiera ser masificada, fácilmente reproducible y que potenciara el interés por desarrollar nuevas tecnologías en áreas con bajo presupuesto.

Para determinar cuáles eran las especificaciones a cumplir, con miras a lograr este objetivo macro, se realizó una investigación del estado actual del mercado y las tecnologías dedicadas a resolver estos problemas.

2 INVESTIGACIÓN

Según un estudio realizado en el “Institute for Transport Studies, University of Leeds” la cantidad de vehículos proyectados para el 2030 sera de aproximadamente 2 mil millones comparado con los 800 millones en el 2002, específicamente en china el crecimiento sera de 20 millones a 390 millones (Figura 2-1). Esta rápida expansión de vehículos en el mercado genera un crecimiento proporcional en la demanda de petróleo.

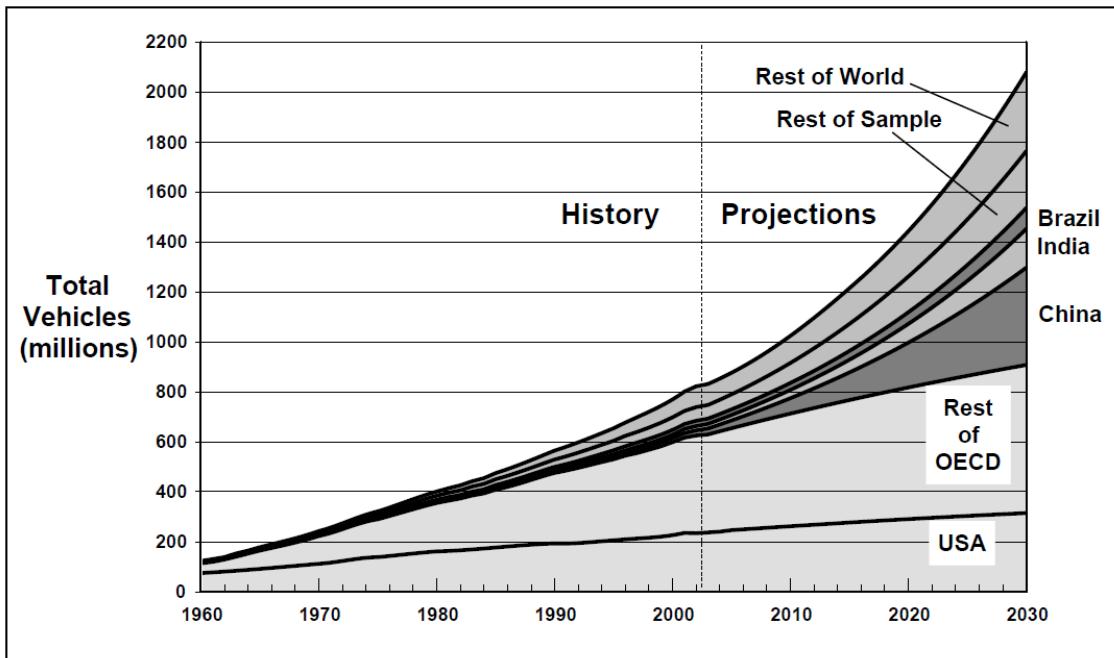


Figura 2-1: Resumen histórico y proyectado por regiones del crecimiento de vehículos en el mercado global.

Este incremento de vehículos genera dos problemas principales, por un lado, el incremento de la demanda de combustible y como consecuencia su agotamiento, y en segundo lugar el congestionamiento constante y creciente de grandes ciudades.

Según distintos estudios el primer problema fomento la tendencia de electrificación de vehículos, como se puede ver en la Figura 2-1 durante los últimos años distintas empresas, comenzando del rubro automotriz, realizaron inversiones en desarrollo de vehículos eléctricos dando la posibilidad a la reducción de la demanda de combustible.

El segundo problema es el más complejo de solucionar, pero existen distintas propuestas entre las cuales vale la pena mencionar:

- Autonomous Vehicle Technology: Esta tecnología de vehículos autónomos da la capacidad de una conducción más eficiente sin modificar infraestructura.
- Real-Time Traffic Feedback: Se centra en aumentar el valor de peajes e infracciones dinámicamente en base al flujo de tránsito. Es necesario infraestructura para medir el flujo.
- Car Sharing and Multi-modal Solutions: Apps para compartir viajes y reducción de tasas proporcional a la carga.

- V2I Smart Corridors: Generar corredores dinámicamente ante atascos de tráfico dando direcciones por cartelería. Necesaria la infraestructura.
- Adaptive Traffic Signals: Busca ajustar los tiempos de los semáforos e indicaciones según congestión. Necesaria la infraestructura.

Estos desarrollos no terminan de solucionar los problemas ya que la infraestructura existente sigue siendo la misma y es inviable actualizarla a la velocidad de crecimiento que tiene el mercado. La tendencia de electrificación automotriz solo resuelve en parte, la demanda de combustible, pero el tiempo de viaje y el congestionamiento continúa existiendo.

Centrados en estos inconvenientes, en distintos laboratorios de investigación se está desarrollando sistemas automáticos de control que puedan inteligentemente coordinar el flujo del tránsito y maximizar el uso de la infraestructura existente. Pero, para que esto sea viable y posible, cada vehículo debe tener la habilidad de poder navegar cualquier territorio autocontrolado y poder responder correctamente a cualquier imprevisto al mismo tiempo que recibe órdenes del camino que debe tomar.

La etapa 1 para lograr ese objetivo es obtener un sistema que pueda controlar a un vehículo estándar para que pueda llegar de un punto a otro evitando correctamente todos los obstáculos que se le presenten. En este primer paso se encuentran la gran mayoría de los laboratorios de control de universidades y empresas, planteando distintos tipos de sistemas de control implementados en distintas vehículos y plataformas.

En general los equipos de desarrollo se centran en el sistema de control sin centrarse en la mecánica del vehículo y termina generando un desacople de desarrollo. Existen una gran cantidad de plataformas educativas robóticas pero la mayor parte de desarrollo e investigación académico son sobre vehículos holonomicos. Por otro lado, la realidad es que más del 95% de los vehículos del mercado son no-holonomicos y no hay existe una plataforma educativa de referencia con este tipo de mecanismo.

Con el objetivo de determinar la estructura de la plataforma se realizó una investigación sobre distintas plataformas analizando ventajas y desventajas.

Se dividió la investigación en dos: la estructura por un lado y hardware y software por otro.

2.1 ESTRUCTURA

La dificultad principal de resolver el problema de control vehicular es lograr la conjunción de un sistema de control con el modelo mecánico del vehículo. Para simplificar el problema y poder obtener un sistema de control que funcione, distintos centros de desarrollo fueron proponiendo distintas plataformas mecánicas muy diferentes a un automóvil de pasajeros.

Dentro de los vehículos terrestres con ruedas se los puede clasificar por la cantidad de ruedas que poseen y cada uno tiene sus ventajas y desventajas para su desarrollo.

2.1.1 Dos ruedas paralelas con el mismo eje

Vehículos controlados con dos ruedas paralelas (Figura 2-2). Se popularizo en las últimas décadas, se utiliza principalmente para vehículos de transporte unipersonal recreativo. Debido a la necesidad de hardware especializado para mantener el balanceo se incluyó al mercado hace poco tiempo. El producto más icónico del mercado que funciona de esta manera es el Segway.

Características mecánicas:

- 2 ruedas paralelas impulsadas por motores eléctricos
- Autobalance
- Sin suspensión
- Máxima velocidad angular 132 rpm y un desplazamiento no mayor a 10 km-h para vehículos de transporte de personas



Figura 2-2: Plataforma de dos ruedas, con auto balanceo

Ventajas:

- Poco espacio cubierto

Desventajas:

- Flexibilidad
- Modularización
- Adaptabilidad
- Control
- Censado
- Limitaciones de concentración de carga (por posible desbalanceo)
- Reproducción

Conclusión:

Este tipo de vehículos están centrados en el transporte individual de personas en terrenos planos. La disposición de carga es crítica, por lo tanto, la modularización es relativa por la disposición de carga.

Los sensores y el procesamiento tienen que ser lo suficientemente veloces para corregir el desbalance.

Por otro lado, si bien el control es complejo debido a la inestabilidad propia del sistema existen hardware dedicado a resolver este problema.

2.1.2 Tres ruedas

Hay varios tipos de vehículos con tres ruedas, y se separan en:

- Delta: Dos ruedas tractoras en la parte trasera y una rueda directora en el frente.
- Tadpole: Rueda trasera impulsora, y dos delanteras directoras.
- Reverse trike: Dos ruedas tractoras en la parte delantera y una rueda directora trasera.
- Free Wheel: Dos ruedas independientes tractoras y una tercera de apoyo.

Este tipo de vehículos en el mercado existen en forma de motocicletas. Tienen un sistema de dirección similar al de las motos, y un sistema de tracción como el de los autos.

Las plataformas de desarrollo en los laboratorios de control generalmente son del tipo Free Wheel con dos motores independientes los cuales permiten el giro sobre su eje.

Características de la plataforma:

- Generalmente no tienen amortiguación
- Baja velocidad
- Alta rigidez

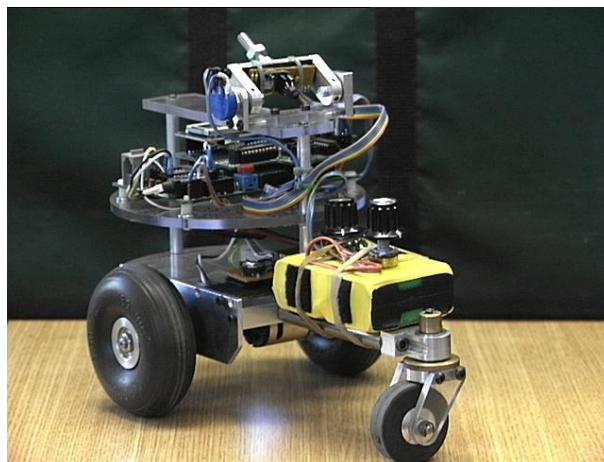


Figura 2-3: Dos motores independientes en las ruedas traseras y rueda libre delante

Ventajas:

- Simplicidad de diseño mecánico
- Control simple
- Modularización
- Censado
- Fácilmente reproducible
- Flexibilidad
- Bajo costo

Desventajas:

- Inestable al giro. Cuando el vehículo dobla, el centro de gravedad queda fuera del mismo y tiende al vuelco.
- Baja velocidad debido a la inestabilidad
- No hay vehículos comerciales que funcionen de esta manera.

Conclusiones

El diseño mecánico es simple, fácil de reproducir y el modelo es simple. En consecuencia, el control resulta más fácil que el de dos ruedas.

Por el contrario, la cantidad de vehículos de transporte de cargas o personas que sea semejante es muy baja y lo que existen tienen problemas de estabilidad.

2.1.3 Cuatro ruedas

Los vehículos de cuatro ruedas son los de mayor cobertura del mercado a nivel global. Vale la pena mencionar algunos de los tipos de vehículos de 4 ruedas:

- Dos ruedas traseras impulsoras y dos ruedas libres frontales
- Ruedas impulsoras paralelas (Tanque)
- Ruedas impulsoras trasera y 2 ruedas delanteras con dirección Ackermann.

Si bien el tipo correspondiente a 4 ruedas impulsoras es común en ciertos vehículos, la mayor cantidad de vehículos son del tipo Ackermann.

De este tipo se analizaron dos tipos de plataforma distintas que actualmente se utilizan para desarrollos:

- Auto a escala comercial de tipo modelismo modificado.
- Modelo impreso 3D de auto modelismo.

Ambos con diferencial mecánico y un solo motor eléctrico.

2.1.3.1 Auto a escala comercial de tipo modelismo modificado

Este tipo de plataformas están basados en una mecánica comercial ya establecida. Generalmente se utiliza como base mecánica de la plataforma autos de modelismo que se encuentran disponibles en el mercado.

Tiene como característica principal que el mecanismo está probado y funciona correctamente bajo condiciones variantes. Por otro lado, se corre con la desventaja que todo desarrollo sobre la plataforma sera dependiente de esa mecánica, la cual no es de libre uso.

Distintas universidades hacen uso de esta lógica, por ejemplo, el Instituto Tecnológico de Massachussets (MIT) tienen usa como plataforma de desarrollo los vehículos de Traxxas (Figura 2-4).

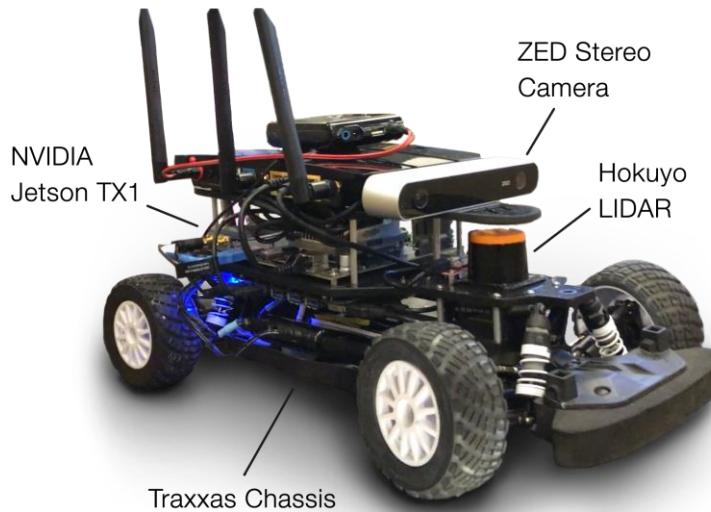


Figura 2-4: MIT Racecar. Plataforma del MIT basado en Traxxas

Mit Racecar

Ventajas:

- Comercial
- Diseño probado y validado
- Repuestos accesibles
- Suspensión en 4 ruedas
- Motor - Driver y baterías incluidos
- Diferencial mecánico
- Bajo costo
- Adaptabilidad a distintos terrenos
- Escalable
- Velocidades entre 0 y 180 km/h

Desventajas:

- Flexibilidad (No soporta modificaciones mecánicas drásticas)
- Modularización (Está sujeto al diseño comercial)
- Reproducción dependiente del mercado

Conclusión:

Es una buena alternativa ya que el diseño mecánico está casi resuelto. Existe la dificultad de que para colocar los sensores y sistemas de procesamiento es necesario realizar modificaciones sobre la estructura comercial, lo cual puede impactar negativamente en el momento de la reproducción de la plataforma ya que en caso de que no se produzca más la estructura el diseño queda obsoleto.

2.1.3.2 *Modelo impreso 3D de auto modelismo.*

La comunidad de modelismo es lo suficientemente grande como para existan desarrollos de todo tipo. Es destacable un desarrollo particular llamado *Open-RC* (Figura 2-5) el cual este compuesto por un diseño integral de un auto de modelismo escala 1/10 imprimible en 3D.

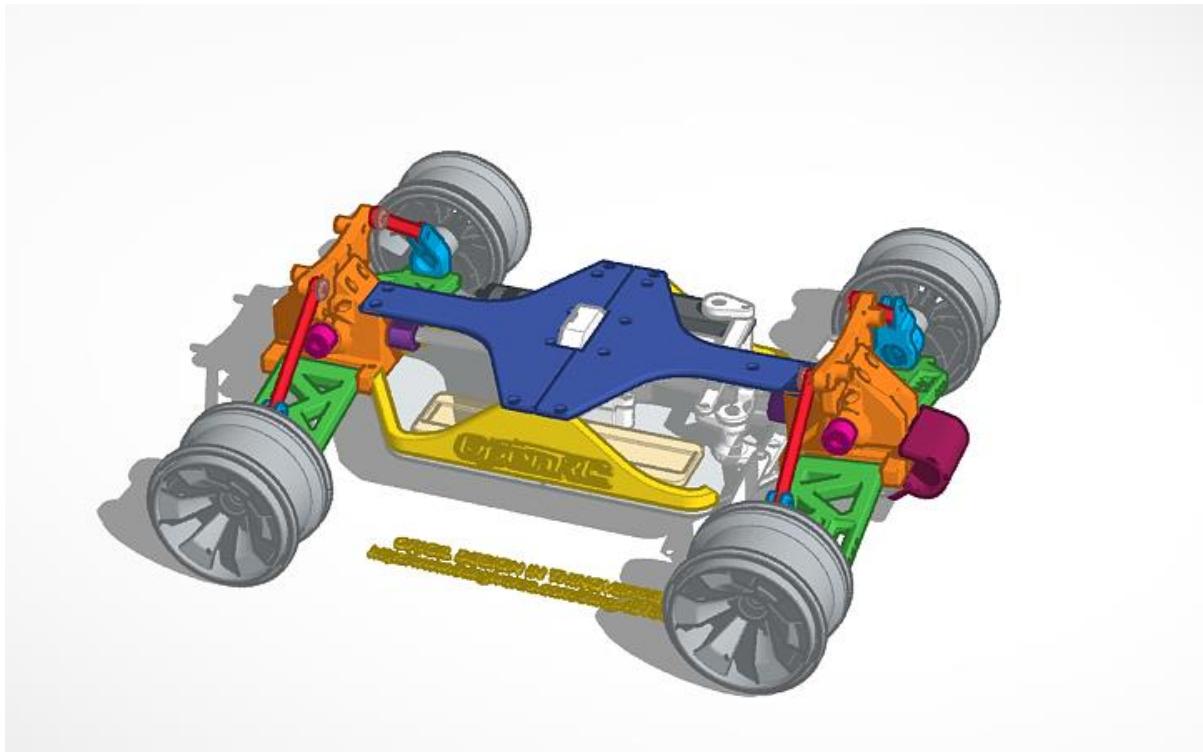


Figura 2-5: *Open - RC*

Este desarrollo facilita el acceso y aportan a la proliferación de desarrollos para vehículos reales. Por otro lado, este desarrollo solamente está orientado a reproducir un auto de modelismo. Esto hace que todas las piezas diseñadas cumplan ese objetivo y no otro, siendo necesaria una modificación del modelo en caso de utilizarlo para otro objetivo.

2.1.4 **Ventajas:**

- Adaptabilidad a distintos terrenos
- Fabricación local
- Diseño parcialmente resuelto

2.1.5 **Desventajas:**

- Necesidad de rediseño
- Flexibilidad
- No modular
- Baja vida útil debido a la mecánica impresa

Conclusión:

Es una buena alternativa ya que el diseño mecánico está casi resuelto. Existe la dificultad de que para colocar los sensores y sistemas de procesamiento es necesario realizar modificaciones sobre el modelo.

Por otro lado, toda la mecánica esta impresa incluyendo a los engranajes del diferencial. Esto implica que la vida útil estará sujeta a la calidad de la impresión y al material del cual está realizado. Las propiedades mecánicas del material en cuestión son muy pobres a la hora de soportar la fatiga, por lo tanto, es un punto importante por considerar.

2.1.6 Conclusiones

Como el objetivo es tener una plataforma flexible que pueda representar un vehículo del mercado es necesario que la plataforma tenga la posibilidad de tener un mecanismo de dirección Ackermann ya que la mayoría de los vehículos la tienen.

Los autos comerciales como el *MIT racecar* tienen la ventaja de estar validados, pero no es muy flexible en el momento de querer aplicar modificaciones y son dependientes de lo que ofrece el mercado. Por otro lado, la alternativa del diseño integral en 3D tiene la ventaja de la independencia de la oferta del mercado, pero la vida útil se ve altamente reducida.

2.2 HARDWARE Y SOFTWARE

El objetivo de esta plataforma es principalmente para desarrollar sistemas de control que permitan controlar un vehículo autónomamente y al mismo tiempo que sea fácil de reproducir, sea flexible y accesible.

Para cumplir con estos postulados es imprescindible que la plataforma sea flexible también del punto de vista electrónico y software, por lo tanto, se realizó una investigación y un análisis de costos para evaluar que plataforma de desarrollo es la indicada para implementar en este caso.

En primer lugar, se tuvieron ciertas consideraciones. La primera tiene que ver con el acceso al hardware, en Argentina no se encuentran disponibles en stock las mismas alternativas que en el resto del mundo debido a la dificultad de importación.

La segunda consideración es el costo, si bien es posible obtener cualquier plataforma del mercado global el costo de obtenerlo es de más del 100% del valor en el exterior.

La tercera consideración tiene que ver con el software. La plataforma será utilizada para desarrollos académicos, específicamente alumnos de ingeniería Mecánica, por lo tanto, debe de tener una barrera de entrada acorde a los conocimientos adquiridos en el instituto.

Luego se establecieron los requerimientos del conjunto hardware-software desde una perspectiva académica.

Requerimientos:

- Capacidad de soportar distintos módulos
- Comunicación entre módulos
- Simpleza para generar módulos nuevos
- Posibilidad de reemplazo de los módulos
- Fácil de escalar y ampliar
- Acceso para analizar los datos y debugging

- Soportar módulos descentralizados
- Robusto
- Open source
- Hardware independiente
- Capacidad de visualización de datos

Determinados los requerimientos y establecidas las consideraciones se realizó un análisis de distintas plataformas el cual se puede ver resumida en la Tabla 2-1.

HARDWARE SOFTWARE	Indices de evaluacion de caracteristicas								
	FACILIDAD DE OBTENCION	COSTO	FACILIDAD DE IMPLEMENTACION	MODULAR	CONECTIVIDAD	ESCALABILIDAD	POTENCIA	SOPORTE DE LA COMUNIDAD	TOTAL
ARMV7 - ROS	1	1	1	8	10	9	10	7	47
ARM - PYTHON/C++	8	3	4	8	10	5	8	9	55
ARM+MICRO CONTROLADOR - PYTHON/C/C++	4	3	4	7	10	5	8	7	48
ARM+ARDUINO - PYTHON/C/C++	8	3	4	7	10	8	8	8	56
MICROCONTROLADOR - C/C++	5	6	6	6	5	5	5	4	42
ARDUINO - C/C++	10	10	8	6	5	9	2	10	60

Tabla 2-1: Evaluación de características

Como se puede ver la opción con mejor desempeño es la de utilizar como hardware la placa de desarrollo Arduino. A continuación, se detalla el análisis de cada uno.

2.2.1 ARMV7 – ROS

Armv7 es la arquitectura de hardware que utilizan ciertas placas de desarrollo. Estas tienen la capacidad de correr UBUNTU, un sistema operativo basado en Linux, el cual permite la ejecución de ROS (Robot Operating System).

[..] ROS provee los servicios estándar de un sistema operativo tales como abstracción del hardware, control de dispositivos de bajo nivel, implementación de funcionalidad de uso común, paso de mensajes entre procesos y mantenimiento de paquetes. Está basado en una arquitectura de grafos donde el procesamiento toma lugar en los nodos que pueden recibir, mandar y multiplexar mensajes de sensores, control, estados, planificaciones y actuadores, entre otros. La librería está orientada para un sistema UNIX (Ubuntu (Linux)) aunque también se está adaptando a otros sistemas operativos como Fedora, Mac OS X, Arch, Gentoo, OpenSUSE, Slackware, Debian o Microsoft Windows, considerados como 'experimentales'. [...]

ROS fue desarrollado específicamente para desarrollo de robots, por lo que para esta plataforma sería la opción ideal.

EL hardware analizado para correr ROS fue la placa de desarrollo de Nvidia: la Jetson TK1.

Este hardware tiene una potencia comparable al de un pc y está preparada para interactuar con distintos dispositivos, pero debido a la complejidad, la dificultad de obtenerla y su costo se transformó en una alternativa no utilizable.

2.2.2 ARM – PYTHON/C++

Arm sin especificar versión se refiere a placas de desarrollo de menor potencia que las anteriores, como pueden ser las Raspberry Pi, las cuales no tienen la potencia para correr ROS.

Esta alternativa se basa en el análisis realizado utilizando una Raspberry Pi 3. Básicamente se propuso resolver el hardware mediante una única placa de desarrollo adaptando niveles lógicos para permitir la comunicación con distintos sensores y actuadores. Por otro lado, la programación basada en Python y/o C++ permite la flexibilidad y el acceso a librerías públicas.

Esta opción limita la posibilidad de modularizarían debido a la forzada centralización del hardware. Ademas, esta opción tiene la dificultad técnica que implica la configuración inicial del sistema operativo con una curva de aprendizaje que supera el tiempo establecido para el proyecto.

Si bien tiene una gran comunidad que simplifica la implementación, actualmente en la carrera no hay una iniciación en este tipo de plataformas. Es fácil obtener esta plataforma, pero el costo es un punto que considerar.

2.2.3 ARM +

Las versiones que incluyen el hardware con ARM tienen las mismas dificultades que las mencionadas previamente, pero en este caso se busca amplificar la modularización mediante la inclusión de un microcontrolador (Arduino u otro). Ademas, esto permite la adaptación de niveles lógicos y preprocesamiento en los microcontroladores.

También se descartó por las complejidades destacadas previamente.

2.2.4 Microcontrolador

Esta opción se refiere al uso de algún microcontrolador, por ejemplo, de Motorola o de Texas, el cual sea específicamente dimensionado según las tareas que realizaría el vehículo.

Es una opción válida para productos, pero no para plataformas de desarrollo debido al bajo soporte público que existe. Unos de los puntos de esta plataforma es que sea fácil reproducir y mantener. El elegir un microcontrolador específico se limita capacidad de desarrollo cooperativo.

Por otro lado, la potencia no es muy alta y es necesario un desarrollo extra de hardware para implementarlo en la plataforma.

2.2.5 Arduino

Arduino es una plataforma de desarrollo basada en distintos Microcontroladores de baja potencia bajo el concepto de “sistema mínimo”. Consiste en una serie de puertos que conectan con los pines del microcontrolador específicamente dispuestos para permitir la aplicación de shields (módulos de expansión).

La característica principal de Arduino es que poseen un bootloader que permite la programación mediante una interfaz amigable de alto nivel y un servicio de librerías de uso libre. Esto permitió a lo largo del tiempo una alta generación de librerías por parte de distintos usuarios potenciando la cogeneración de desarrollos. Ademas, son de muy bajo costo debido al alto volumen del mercado.

Por otro lado, la baja potencia limita el desarrollo, en especial en este proyecto la potencia disponible en un solo Arduino no es suficiente. Por eso se propuso utilizar una red de Arduino interconectados mediante el protocolo i2c. Cada uno con su función generando un módulo de hardware software y mecánica el cual se puede remover de la plataforma en su conjunto.

2.2.6 Conclusión

A partir del análisis realizado, se concluyó que la mejor opción para la plataforma es utilizar un arreglo de Arduino interconectados. Esta configuración compensa la baja potencia y aporta a la modularizarían manteniendo el costo y la complejidad baja.

3 OBJETIVOS

Como objetivo principal se pretende fabricar una plataforma basada en un vehículo real a escala. Y los requerimientos propuestos por la catedra es que sea de fácil fabricación en el ITBA. Con esto en mente se planteó utilizar nuevas herramientas como la impresión 3D que facilitan la reproducción.

Ademas, modelo tiene que ser modular, para el caso que se requiera mejorar o agregar nuevas modificaciones, en especial de hardware de experimentación.

Por otro lado, para mostrar el potencial de la plataforma se propone que el vehículo sea capaz de seguir una trayectoria definida de forma autónoma, con un margen de error acorde a las capacidades de los componentes con los que se fabrique la plataforma.

En conclusión, se pueden dividir la plataforma en 3 requerimientos básicos:

- Facilidad de reproducción y fabricación
- Modular
- Capacidad de control para alcanzar un objetivo y evitar obstáculos

4 PLATAFORMA

La plataforma está compuesta por tres elementos igual de importantes, por un lado, el modelo mecánico del vehículo por el otro, el hardware de control del modelo y por último el software de control.

El proyecto se dividió en 3 etapas secuenciales las cuales fueron solapadas con tal de completarlas.

1. Diseño mecánico.
2. Diseño electrónico.
3. Desarrollo de software.

5 DISEÑO MECÁNICO

El proceso de diseño mecánico resultó iterativo, se plantearon distintas alternativas y se diseñaron modelos tridimensionales los cuales fueron evaluados por personal experimentado. De acuerdo con cada devolución se fue adaptando y modificando el diseño para que pueda ser fabricado correctamente.

5.1 DEFINICIONES BÁSICAS

Antes de comenzar con el diseño de la plataforma se establecieron las dimensiones básicas de la plataforma y se establecieron las bases de diseño.

En primer lugar, como características principales se buscó seguir el diseño de un auto comercial con las mismas características, pero escalado una décima parte. Para esto se eligió tomar como parámetro al Toyota Corolla del 2018, auto que tiene dimensiones promedio y fue uno del más vendido nivel mundial.

A continuación, se detallan las dimensiones básicas las cuales están extendidas en el anexo 1.

Ancho	1.776 m
Largo	4.650
Distancia entre ejes	2.7 m
Diámetro de giro	10.8

Tabla 5-1: Dimensiones básicas exteriores del Toyota Corolla modelo 2018

Por otro lado, se limitaron las prestaciones. Como punto principal se estableció que no tendría un mecanismo de amortiguación. Debido a las bajas velocidades a las cuales se ensayarán la plataforma no es necesario que la misma disponga de un sistema de amortiguación ya que con la elasticidad de las ruedas es suficiente para amortiguar vibraciones.

5.1.1 Materiales

Se buscó en distintos modelos de distintos fabricantes como y de qué material fueron diseñadas las piezas que componen el vehículo ya que su diseño depende del tipo de fabricación. Dependiendo del precio y la calidad varían desde materiales plásticos, impresión 3d o inyectado, pasando por aluminio y distintas aleaciones hasta fibra de carbono.

Estas diferencias de materiales van de la mano del mercado al cual está destinado el producto:

- Los plásticos son de producción masiva y de baja potencia, la robustez es baja al igual que el precio. Además, están diseñados para que una única pieza plástica realice la mayor cantidad de funciones, de esta manera reducir la cantidad de piezas y reducir costos en fabricación. Estos diseños son complejos ya que todos son fabricados por inyección y tienen un diseño particular para permitir este tipo de fabricación.
- Impresión 3D: Este método de fabricación es relativamente nuevo y recientemente un grupo de personas basados en diseños de grandes fabricantes de autos a escala diseñó una réplica, pero adaptada para poder imprimirse en cualquier impresora 3D. Este diseño tiene como característica que no está fabricado íntegramente en plástico si no que las “partes fijas” son las que están fabricadas en 3D y unas pocas partes móviles. El resto de las piezas son estándar de vehículos a escala de hobbistas. El diseño es simple y está compuesto por muchas piezas distintas ensambladas y es altamente flexible ante modificaciones.

- Aluminio y otros metales: Este es el material más usado para autos a control remoto para hobbistas, hay piezas mecanizadas y algunas otras fundidas y luego mecanizadas. Los diseños son complejos y están optimizados para el modelo de auto particular, esto quiere decir que es no es muy flexible al momento de intervenir en el diseño.
- Fibra de carbono y otras fibras: Este material se usa generalmente en ciertas partes del auto, generalmente en las partes fijas y planas, diseñada con el objetivo de reducir el peso para aumentar la velocidad final del vehículo. Este tipo de material es complejo de trabajar y las piezas que se realizan con el mismo son planas.

Basados en este análisis y con el objetivo de la plataforma en mente se propuso diseñar un modelo que sea lo más flexible posible para ajustarlo fácilmente a los elementos de censado y control que se colocaran sobre esta.

Para esto se pensó en primer lugar para que sea fabricable mediante impresión 3D o corte de placas y que las partes con mecanismos móviles sean estándar de modelos a escala. El proceso de diseño fue iterativo, consultando con expertos en el tema para validarla.

Como se comentó previamente la plataforma debe ser modular, lo cual influenció en las decisiones de diseño. En la siguiente tabla se resumen las etapas de diseño y los distintos módulos que se plantearon en cada una, así como decisiones de fabricación y materiales.

5.2 DISEÑO PRELIMINAR

Con el objetivo de cumplir con los requerimientos de modularización y flexibilidad, se plantearon una serie de módulos funcionales.

5.2.1 Chasis

Tiene el objetivo de soportar el resto de los módulos y sea el componente central que soporte las solicitudes mecánicas.

5.2.2 Módulo de tracción:

Este módulo es el que contiene al motor, el diferencial y las ruedas traseras, lo suficientemente flexible para que pueda ser modificado para adaptaciones a otros sensores.

No tendrá amortiguación ya que se considera que las ruedas elásticas son suficiente para el control en bajas velocidades, pero se podrá agregar en caso de ser necesario.

5.2.3 Módulo frontal:

El módulo frontal contará con el mecanismo de 4 barras ackermann servo controlado y sin amortiguación.

5.2.4 Módulo de alimentación:

El módulo de alimentación contará con una batería acorde y un módulo de carga para que pueda automáticamente dirigirse al centro de carga.

5.2.5 Módulo de control:

El módulo de control será subdividido en distintos submódulos los cuales son dependientes del presupuesto y de las iteraciones del desarrollo del modelo de control.

Se proponen los siguientes submódulos:

- Core
- Interfaz de sensores y actuadores
- Drivers
- Sensores
- Actuadores
- Comunicación

5.2.5.1 *Core*

El core será el responsable de implementar el control, se propone utilizar un Arduino Mega que se comunicara con la interfaz de sensores y actuadores mediante el protocolo i2c. Esta placa de desarrollo es una de las más difundidas y con mayor soporte y comunidad, por lo tanto, es fácil adquirirla y es accesible para la reproducción.

5.2.5.2 *Interfaz de sensores y actuadores*

Se utilizará un Arduino nano el cual recibirá los datos de los sensores y realiza un preprocesamiento.

5.2.5.3 *Drivers*

Este módulo comercial es el responsable del control de potencia de los motores, tiene una interfaz propia y se comunicara con el Arduino nano de la interfaz.

5.2.5.4 *Sensores:*

Los sensores serán colocados en los lugares que corresponda para detectar lo que se desea medir. Como uno de los requerimientos de la plataforma es que sea reproducible y económica se propone que los sensores sean:

- 3 sensores ultrasónicos de distancia
- IMU para detectar aceleraciones y giros
- Encoder infrarrojo para medir velocidad del vehículo
- Sensores de posición para servos.

5.2.5.5 *Actuadores:*

Serán colocados en cada módulo correspondiente:

- Servo de dirección ackermann
- Motor de tracción

5.2.5.6 *Comunicación*

Para comunicarse con la interfaz y tener visualización de datos se utilizará un módulo RF de 2.4 Ghz y se desarrollará un control remoto capaz de controlar externamente el vehículo y con capacidad de transmitir los datos vía serial USB al pc.

5.3 DISEÑO

El proceso de diseño fue iterativo, se partió del diseño preliminar buscando cumplir con los módulos propuestos previamente establecidos y siendo flexible con el método de fabricación.

Como se puede ver en la siguiente tabla existieron 4 diseños dos de los cuales fueron fabricados y evaluados. Los primeros dos, fueron descartados en etapa de diseño luego de un análisis y feedback de expertos en diseño y fabricación de piezas.

Diseño 1		
Modulos	Material	Piezas comerciales
Modulo de alimentacion	Impresión 3D o fibrofacil con corte laser + Aluminio	Bateria + regulador
Chasis	Aluminio	
Modulo de traccion	Aluminio	Dogbone, shaft, rulemanes, Diferencial, motor, ruedas.
Modulo de direccion	Impresión 3D o fibrofacil con corte laser + Aluminio	Servo, masa, porta masa, mecanismo Ackermann, ruedas.
Modulo de control	Impresión 3D o fibrofacil con corte laser	Placa de desarrollo.
Modulo de sensado	Impresión 3D o fibrofacil con corte laser	Sensores.
Diseño 2		
Modulos	Material	Piezas comerciales
Alimentacion+Chasis+ Traccion	Impresion 3D	Bateria, Regulador, Dogbone, shaft, Rulemanes, Diferencial, Motor.
Modulo de direccion	Impresion 3D	Servo, masa, porta masa, mecanismo Ackermann, ruedas.
Modulo de control	Impresion 3D	Placa de desarrollo.
Modulo de sensado	Impresion 3D	Sensores.
Diseño 3		
Modulos	Material	Piezas comerciales
Traccion	Impresion 3D	Dogbone, shaft, Rulemanes, Diferencial, Motor.
Chasis	Impresion 3D	Bateria, placa de desarrollo
Paragolpe	Impresion 3D	Sensores
Modulo de direccion	Impresion 3D	Servo, masa, porta masa, mecanismo Ackermann, ruedas.
Modulo de control	Impresion 3D	Placa de desarrollo
Diseño 4		
Modulos	Material	Piezas comerciales
Traccion	Impresion 3D	Dogbone, shaft, Rulemanes, Diferencial, Motor, Ruedas.
Chasis	Impresion 3D	Bateria, placa de desarrollo
Modulo de direccion	Impresion 3D	Servo, rulemanes, ruedas.
Modulo de control	Impresion 3D	Placa de desarrollo
Modulo de sensado	Impresion 3D	Sensores, Servo.

Tabla 5-2: Resumen de diseño.

Cada módulo desmontable y reemplazable para poder cambiar las relaciones de tamaño con el fin de que sea semejante a un vehículo escala 1:1 particular.

A continuación, se muerta un poco más en detalle cada iteración y las razones de porque fueron descartados.

5.4 PRIMERA ITERACIÓN DE DISEÑO

En primer lugar, para poder realizar el desarrollo de la plataforma en el tiempo establecido por la cátedra se analizó la complejidad de fabricación y diseño de cada pieza del automóvil.

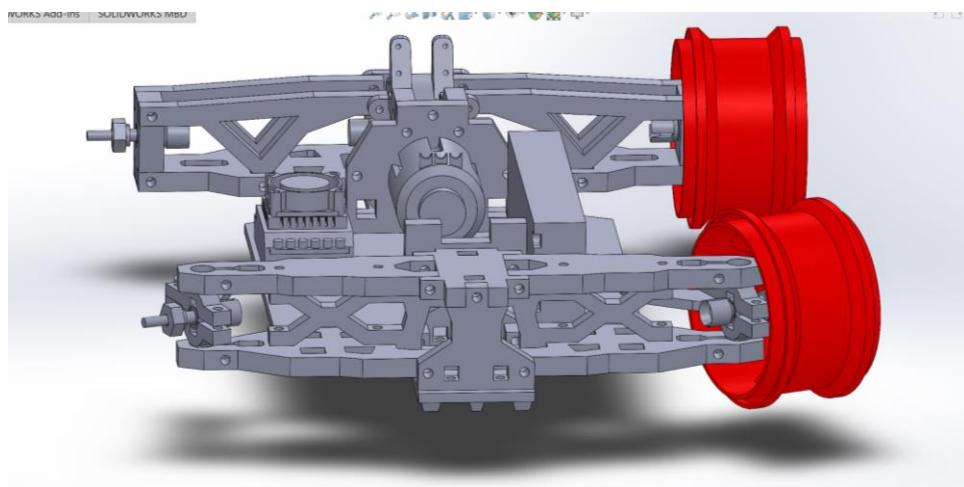


Figura 5-1: Primera iteración de diseño.

Como se puede ver en la figura, el diseño del chasis y de la dirección se realizó con piezas planas encastrables, fabricables en distintos materiales. Estas piezas pueden ser obtenidas mediante corte de placas o mecanizado, además también es posible imprimir las piezas o una combinación de las anteriores.

5.4.1 Problemas en el diseño y análisis

Como el diseño está limitado a cortes de placa genera una complicación con respecto al ensamble, para generar piezas con volumen apreciable es necesario el encastre y fijación de cada pieza. Si estas piezas son fabricadas en un centro de mecanizado las tolerancias serían las correctas y existiría mayor problema, pero debido a que se está planteando la posibilidad de una fabricación con corte de madera o impresión 3D, es imprescindible que el diseño sea tolerante a fallas de fabricación del orden de decimas de milímetro.

Al aumentar las piezas para poder generar un volumen apreciable los errores aumentan y no es posible garantizar un ensamble correcto. Por este motivo se decidió descartar la posibilidad de fabricarla con corte de placas y diseñar para un solo tipo de fabricación. Se estableció que se limitaría la fabricación del auto al de la impresión 3D. De esta manera es posible reducir drásticamente la cantidad de piezas y solucionar parcialmente el problema de la cantidad de encastres.

5.5 SEGUNDA ITERACIÓN DE DISEÑO

Partiendo de la premisa que será todo el auto impreso en 3D y que idealmente se tienen que reducir las piezas al mínimo para reducir los encastres se propuso el siguiente diseño.

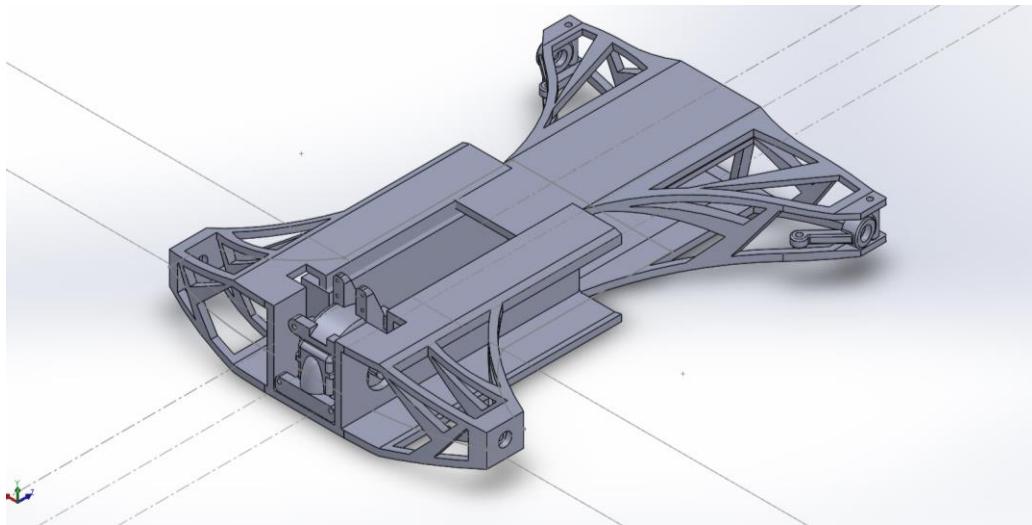


Figura 5-2: Segunda iteración de diseño.

Es un modelo de 2 piezas básicas en las cuales se monta el diferencial y motor y el sistema de dirección. Este diseño busca reducir al mínimo la cantidad de piezas necesarias para la plataforma.

5.5.1 Problemas de diseño y análisis

Por un lado, la reducción de piezas fue drástica, y es una ventaja respecto a errores de encastres, pero por el otro se aumentó la complejidad de diseño debido a que si bien hay menor cantidad de encastres son de mayor complejidad.

El diseño se consultó con distintos expertos en diseño mecánico e impresión 3D y las conclusiones fueron que no es conveniente imprimir piezas tan grandes y complejas, debido a que la posibilidad de falla es muy grande y es muy difícil que terminen cumpliendo las tolerancias estándar de la máquina.

Obtenidos estos resultados, e inputs se obtiene como conclusión que hay una relación de compromiso en el diseño de piezas para impresión 3D.

- La complejidad de la pieza hace difícil la impresión y aumenta la posibilidad de fallas durante la fabricación.
- El tamaño de la pieza aumenta el tiempo de impresión y por lo tanto la posibilidad de fallas debido a la imposibilidad de controlar el entorno durante tanto tiempo. (Cortes de luz, temperaturas frías a la noche, etc.)
- Mientras más grandes sean las piezas menor resistencia tienen, ya que el relleno de la pieza no es completo pero las paredes si lo son.
- A mayor cantidad de piezas mayores encastres, y más problemas de tolerancias.

Con este análisis se planteó un nuevo diseño el cual balancea estos puntos.

5.6 TERCERA ITERACIÓN DE DISEÑO

Se tomaron las siguientes consideraciones de diseño:

- Menor cantidad de piezas posibles manteniendo el diseño simple.
- Diseñar, en lo posible, como si se pudiera fabricar en una fresadora de 3 ejes en una sola toma.
- Necesidad de tolerancia baja

Y se listaron los módulos mecánicos necesarios y que componentes básicos deben contener:

1. Tracción
 - a. Motor
 - b. Diferencial
 - c. Rodamientos
 - d. Masa
 - e. Eje
 - f. Porta masa
2. Dirección
 - a. Servo
 - b. Mecanismo Ackerman
 - c. Masa
 - d. Porta masa
3. Chasis
 - a. Porta batería
 - b. Porta ESC
 - c. Alojamiento de hardware
4. Paragolpes/Sensado
 - a. Sensores Ultrasónicos

Como cada uno de los componentes mencionados y sus dimensiones definen el diseño del modelo se partió modelando cada componente y diseñar en torno a estos. Este diseño fue fabricado y luego modificado por el diseño 4.

5.7 CUARTA ITERACIÓN DE DISEÑO

Partiendo del diseño de la tercera iteración se modificaron algunas piezas y se tomó la decisión de que el mecanismo de transmisión sea mayoritariamente metálico debido a las solicitudes mecánicas que debe soportar.

Ademas, se analizó la posibilidad de reemplazar los sensores ultrasónicos por sensores infrarrojos, con el objetivo de poder utilizar la tecnología ultrasónica para realizar ecolocalización de la plataforma en un espacio cerrado. Por otro lado, los sensores ultrasónicos presentan dificultades de censado ante piezas rectas. Como el sistema ultrasónico funciona midiendo el tiempo que tarda el eco de un pulso ultrasónico si la superficie en la cual incide se encuentra a 90 grados, el eco no vuelve y no detecta el obstáculo.

Por este motivo se removió del 4to diseño el paragolpes ultrasónico y se agregó un módulo de sensado compuesto por sensores infrarrojos.

Ademas, se estableció que existirá un módulo de control de velocidad y comunicación rf independiente del módulo de control que se encarga de evitar obstáculos. Esto se planteó de esta manera tal que se pueda controlar al vehículo con cualquier sistema embebido sin necesidad de reprogramar el sistema de dirección,

tracción o comandos inalámbricos. Ademas, tiene que ser posible de reemplazar este módulo, por lo tanto, se generó uno nuevo con ese fin.

El ultimo diseño consta de los siguientes módulos funcionales.

1. Dirección
2. Tracción
3. Chasis
4. Módulo de sensado
5. Módulo de control

5.7.1 Dirección

Las condiciones para la dirección son la limitación del movimiento del servo y el mecanismo Ackermann. A diferencia de la tracción, lo que primero se diseñó de la dirección fue el alojamiento del servo y el soporte de la porta masa.

Con esos puntos fijos, y la distancia entre ejes preestablecida se iteró el diseño del mecanismo para cumplir con los requisitos estándar de radio de giro de un vehículo escala 1/10 (radio menor a 500 mm). Una vez obtenidas las dimensiones de cada barra se diseñaron las fijaciones restantes.

A partir de este diseño básico, se analizó el mecanismo con el objetivo de verificar que el mismo es apto para las condiciones de trabajo y verificar si el servo elegido en primer lugar es suficiente para permitir el movimiento.

5.7.1.1 Análisis del mecanismo

El mecanismo esquematizado es un sistema de dirección Ackermann compuesto por barras y controlado con un servo motor CS-60 (Hobbico) el cual tiene un Ángulo útil de 180 grados.

Torque:	6.0V: 49.00 oz-in (3.53 kg-cm)
Speed:	6.0V: 0.16 sec/60°

Tabla 5-3: Propiedades del servomotor Hobbico

-El objetivo es analizar cuál es el límite de velocidad del vehículo para el cual sigue siendo posible controlar la dirección considerando que el radio de giro del auto es de 414 mm y la distancia entre ejes del auto es de 335 mm.

-Las cargas del modelo están determinadas por el Ángulo máximo que puede doblar las ruedas y los rpm de esta.

Las ruedas se considerarán sólidas de ABS.

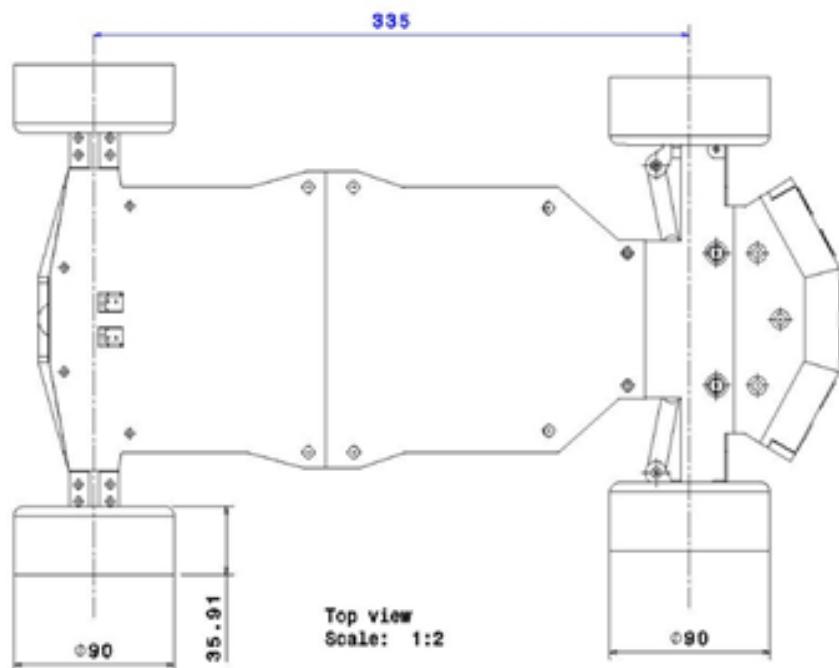


Figura 5-3: Dimensiones básicas del vehículo

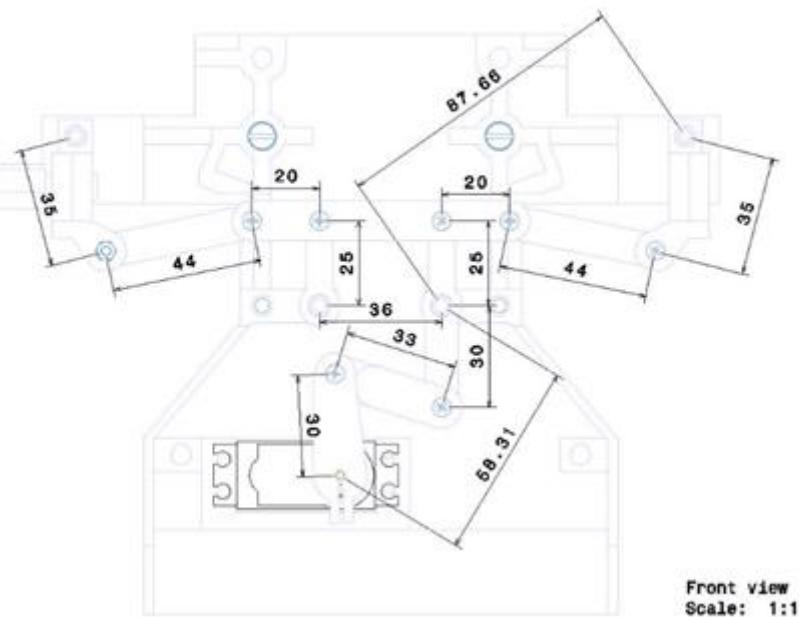


Figura 5-4: Dimensiones básicas del mecanismo

5.7.1.2 Unidades mínimas

El mecanismo se puede dividir en dos mecanismos de 4 barras y dos mecanismos de 5 barras como se puede ver en la figura.

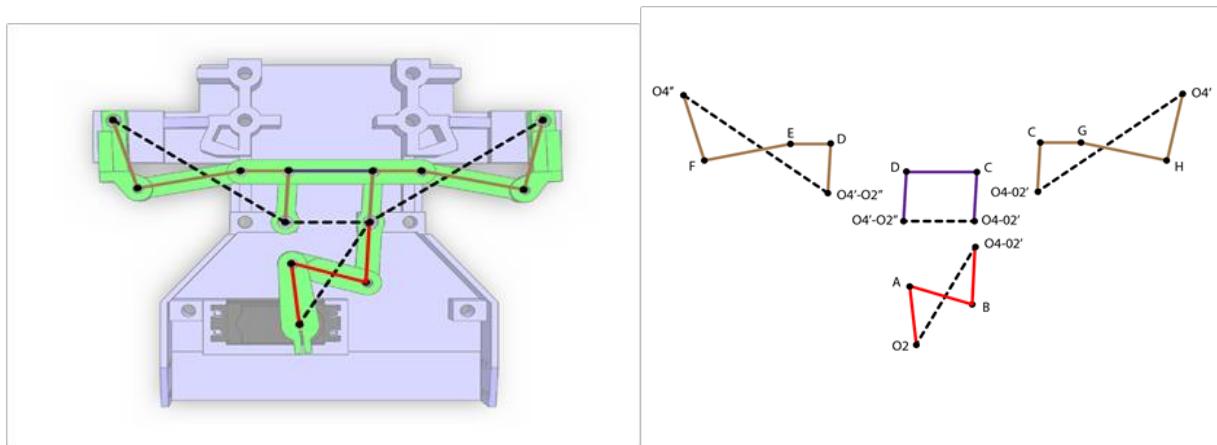


Figura 5-5: Cantidad de grados de libertad y condición de Grashof

Mecanismo O2-A-B-O4:

Mecanismo O2'-D-C-O4

Mecanismo O2'-C-G-H-O4'

Mecanismo O2''-D-E-F-O4''

La condición de Grashof muestra que solamente uno de los mecanismos rota completamente.

$$S + L < P + Q : \text{al menos una manivela}$$

$$S + L > P + Q : \text{triple balancín, ningún eslabón rota completamente}$$

$$S + L = P + Q : \text{al menos una manivela, pero con puntos de cambio cuando se alinean las barras}$$

Mecanismo O2-A-B-O4:

$$S = 30 \text{ mm} ; L = 58.31 \text{ mm}; P = 33 \text{ mm} ; Q = 30 \text{ mm}$$

$$30 \text{ mm} + 58.31 \text{ mm} > 33 \text{ mm} + 30 \text{ mm}$$

$$88.31 \text{ mm} > 63 \text{ mm}$$

Mecanismo O2'-C-D-O4':

$$S = 25 \text{ mm} ; L = 36 \text{ mm}; P = 25 \text{ mm} ; Q = 36 \text{ mm}$$

$$25 \text{ mm} + 36 \text{ mm} = 25 \text{ mm} + 36 \text{ mm}$$

$$61 \text{ mm} = 61 \text{ mm}$$

5.7.1.3 Puntos límite de desplazamiento

El mecanismo tiene puntos característicos en los que se producen inversiones no deseadas, por lo tanto, se limitó al mismo buscando que esas inversiones no se produzcan. Como se puede ver en la figura, existen dos puntos en los cuales el mecanismo está limitado.

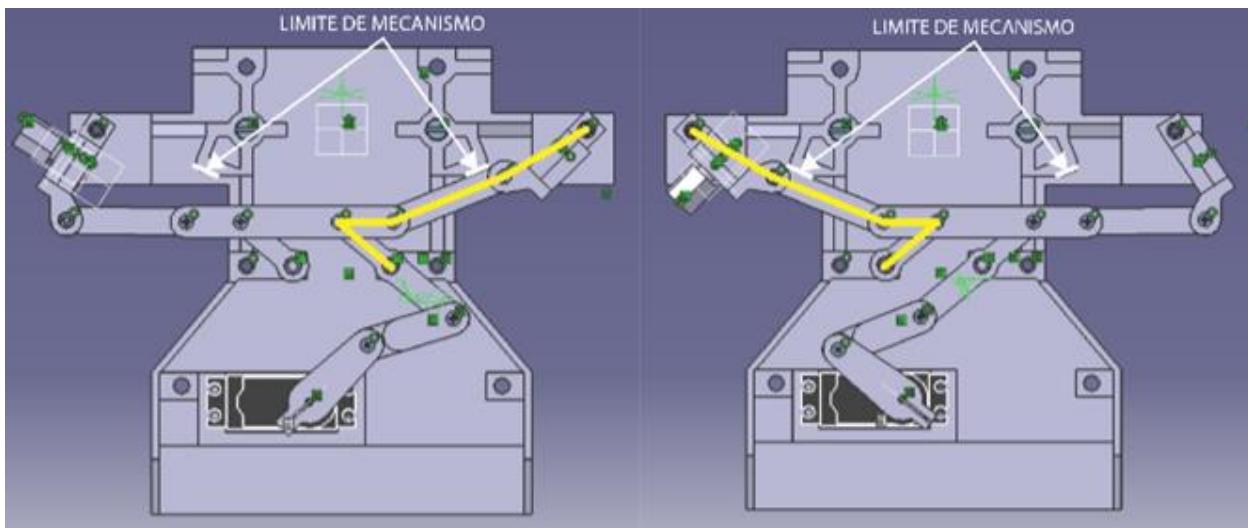


Figura 5-6: Limitaciones de mecanismo

-El mecanismo está limitado por la alineación de las barras GH Y H04' y la alineación de EF y F04''
 -Otros límites del mecanismo y puntos críticos teóricos que se darían cuando se alinea la barra O2'-O4' con C-D, pero no es posible que esto suceda debido a la geometría.

5.7.1.4 Ángulos de transmisión

En un mecanismo o una transmisión, se define el ángulo de transmisión como el ángulo entre la dirección de la fuerza (F) que un elemento o eslabón conductor realiza sobre otro y la dirección de la componente de dicha fuerza que es perpendicular a la velocidad en el punto de aplicación de dicha fuerza.

Siempre se busca que el ángulo de transmisión sea mayor a 45° , ya que valores bajos implican que una gran parte de la fuerza actuante sólo contribuye al aumento de las reacciones y del rozamiento entre los eslabones, pudiendo incluso llegar a bloquearse el mecanismo.

En este mecanismo, hay tres posiciones importantes a analizar:

- Dirección centrada (sin giro): para 2.789 grados de ángulo de entrada de servo el ángulo de transmisión mínimo es de 72.425 grados. (figura)
- Máximo giro: para 51.986 grados de ángulo de entrada de servo el ángulo de transmisión mínimo es de 7.62 grados. (figura)
- Mínimo giro: para -45.652 grados de ángulo de entrada de servo el ángulo de transmisión mínimo es de 7.62 grados. (figura)

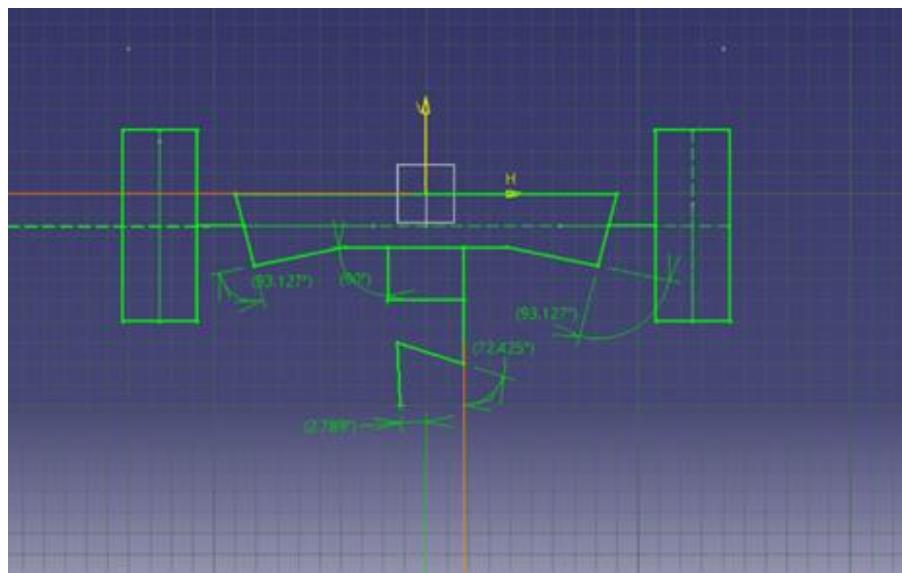


Figura 5-7: Dirección centrada

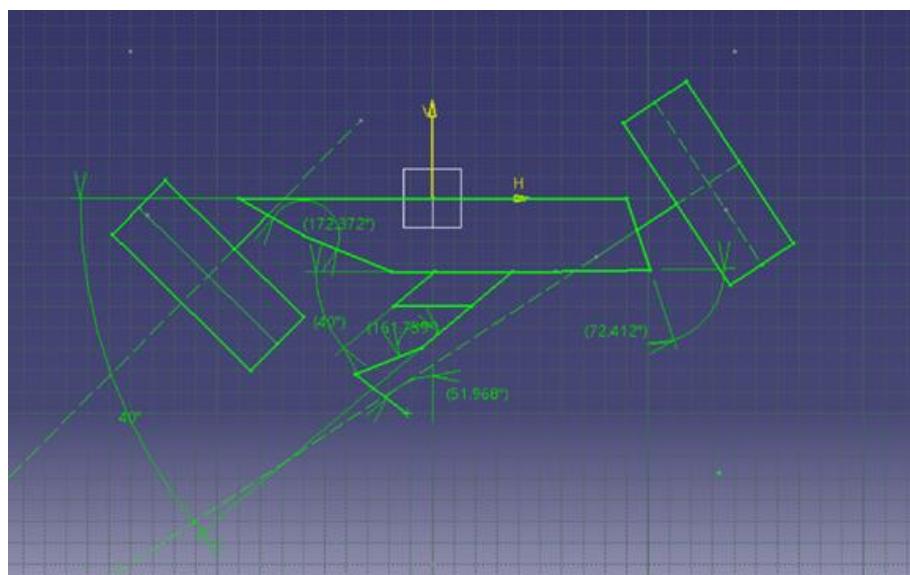


Figura 5-8: Dirección en máximo giro

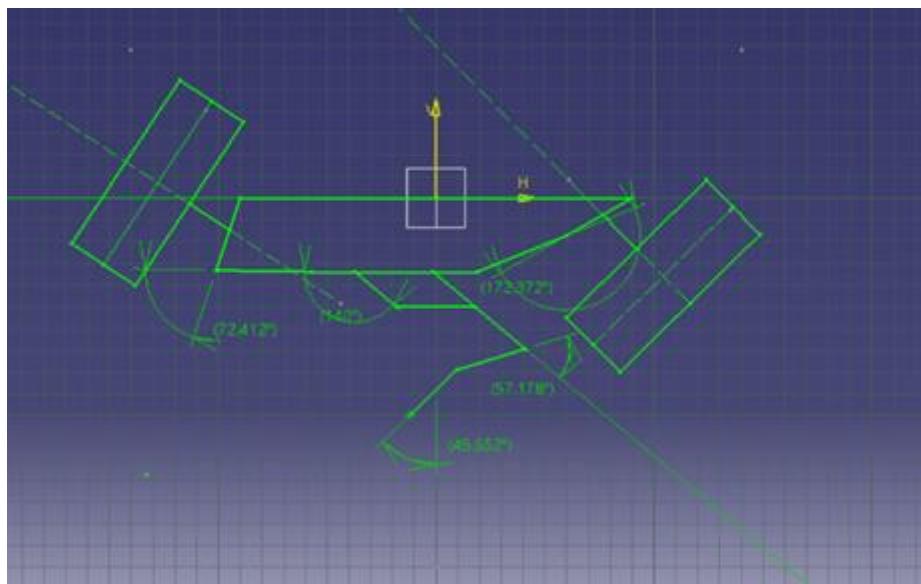


Figura 5-9: Dirección en mínimo giro

Como se puede ver no se cumple la recomendación de que el ángulo de transmisión no sea menor a 45 grados, esto indica que el servo tiene que hacer un toque máximo en los radios de giro mínimos y es posible que no se pueda alcanzar esa posición, para verificar si existe alguna limitación se realizó un análisis de fuerzas.

5.7.1.5 Análisis de fuerzas

En primer lugar, es necesario determinar las cargas sobre el mecanismo. Como se ve en la figura, las cargas dinámicas en las ruedas son de varios tipos, pero en este mecanismo, la carga más importante es la debida al giro a una dada velocidad.

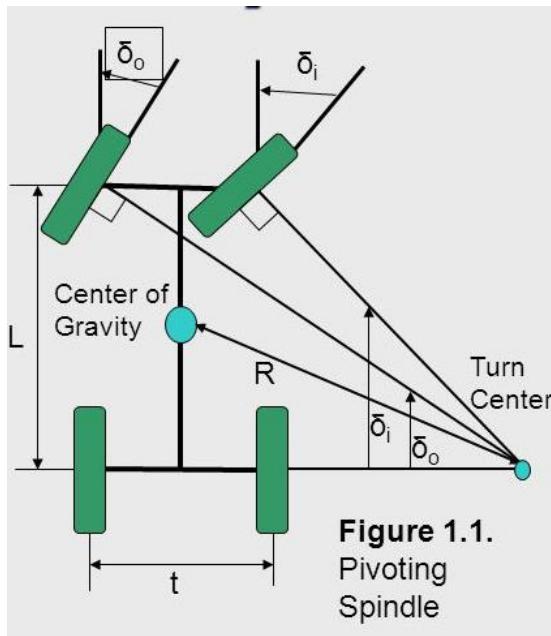


Figura 5-10: Dirección Ackermann

Cuando el vehículo se desplaza a una determinada velocidad y gira se genera una fuerza de roce estático en las ruedas contra el piso que hace que se permita el giro. Esta fuerza es la que el servo tiene que contrarrestar para que pueda mantener ese determinado giro.

Como cada rueda tiene un ángulo de giro y radios de giro distintos, la carga en cada lado del mecanismo difiere. Se modela de la siguiente manera:

$$F_0 = \frac{m}{4} V^2 / R_0 \quad F_1 = \frac{m}{4} V^2 / R_1$$

m: masa del vehículo

V: velocidad de desplazamiento

R₀: radio de giro de rueda izquierda

R₁: radio de giro de rueda derecha

F_i: Fuerza aplicada en dirección al eje de rotación sobre cada rueda

El radio de giro está determinado por el mecanismo y depende del ángulo de entrada del servo. La ecuación que modela la relación entre ambos depende del largo de cada barra y de las posiciones de los joints. Teniendo disponible el modelo, se simuló el mecanismo para obtener esa relación entre los ángulos y los radios y se calcularon las cargas a distintas velocidades.

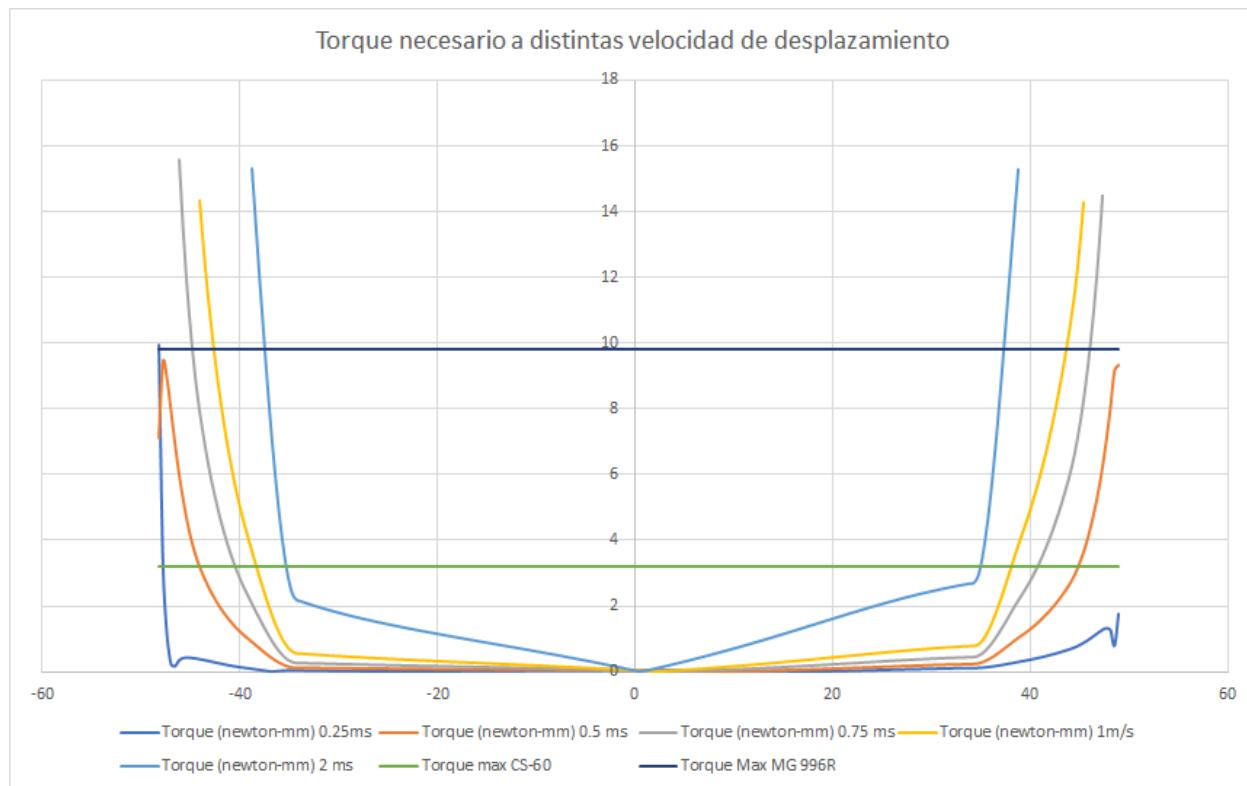


Figura 5-11: Torque necesario del servo a distintas velocidades de desplazamiento

Como se puede ver en el gráfico el torque necesario para mantener un radio de giro específico está íntimamente relacionado con la velocidad de desplazamiento.

Las dos líneas horizontales correspondientes a dos modelos de servo distinto indican cual es el torque máximo que pueden desarrollar cada uno en ese rango de ángulos.

5.7.1.6 Conclusión

Inicialmente se había elegido el servo CS-60 pero luego de este análisis se optó por el MG 996 R, el cual permite un control con mayor rango de ángulo a más velocidad.

Según este análisis se puede controlar en todo el rango, hasta 0.5 m/s de velocidad de desplazamiento de la plataforma.

5.7.2 Tracción

Como se buscó que la plataforma sea lo más semejante a un vehículo comercial se determinó que el sistema de tracción esté compuesto por un motor y un diferencial mecánico. Para diseñar la pieza que contiene los componentes fue necesario definir los componentes, adquirirlos y luego realizar modelo en tres dimensiones de cada uno.

Para eso se realizó un análisis de componentes para evaluar que motor era el correcto y era posible adquirir.

Se comenzó la búsqueda de acuerdo con los motores disponibles para vehículo a escala 1/10 utilizados en autos a escala de hobby.

Estos motores son del tipo brushless trifásicos y son controlados por un controlador de potencia acorde al motor y la alimentación mediante pwm. En el mercado argentino no están disponibles todos los modelos del mercado internacional, por lo tanto, se buscó un conjunto controlador-motor que esté disponible.

Existen dos tipos de motores brushless trifásicos para autos a escala: los llamados sensorless y los sensored.

5.7.2.1 Motores brushless

Este tipo de motores son similares al resto de los motores eléctricos sincrónicos de imanes permanentes, consta de un rotor formado por una serie de imanes permanentes y un estator con bobinas electrificadas. Como tiene un campo magnético constante no es necesario que tenga escobillas.

El rotor que tiene un campo magnético constante tiende a alinearse con el campo magnético generado por las bobinas, cuando este está por alinearse el campo magnético cambia tal que el rotor siga girando, energizando la bobina siguiente.

Cuando se presenta una carga en el eje del motor, la fuerza necesaria para que se alineen los campos magnéticos tiene que ser más grande, esto implica que la corriente de las bobinas tiene que ser mayor para que esta fuerza de Lorentz logre la alineación del campo. Esta situación se detecta mediante el censado de la posición del rotor o mediante la medición de tensión contraelectromotriz del resto de las bobinas. Si el sistema está en la posición correcta (alineado con el campo magnético) el control activa la siguiente bobina, si no lo está aumenta la corriente hasta lograr la alineación.

Este tipo de complicaciones son más evidentes durante el arranque del motor debido a que las cargas en ese momento son las mayores. Por otro lado, es posible diseñar el control para que el motor funcione correctamente sin censar la carga o la alineación aumentando la potencia suministrada arbitrariamente durante el tiempo de arranque.

En el caso de los motores brushleess de modelismo existen los dos tipos de motores, con sensor y sin sensor, ambos tienen un desempeño similar ya que la falta posible sincronización se soluciona aumentando la potencia de arranque. Esto tiene un costo energético sobre la batería, la cual tiene que ser capaz de soportar picos de corriente en todos los arranques, en contraposición con los que tienen sensores que incrementan la potencia en proporción a la carga.

En argentina la diferencia de los precios entre un sistema y el otro son de más de doscientos por ciento. Por este motivo se admitió utilizar un motor sin sensor para mostrar la utilidad de la plataforma.

El modelo de motor utilizado es el EZRUN MAX10 SENSORLESS el cual es un combo de motor + controlador. Las características técnicas se encuentran detalladas en el anexo, para el diseño solo hicieron falta las dimensiones.



Figura 5-12: Motor y controlador EZRUN MAX10

5.7.2.2 Diferencial

Los vehículos con dirección Ackermann de denominan no holonómicos. Un robot holonómico tienen los mismos grados de libertad efectivos que controlables. Por el contrario, un robot no polinómico tiene menos grados de libertad controlables que número total de grados de libertad, por último, si sucede lo contrario, el robot es redundante. Es decir, un vehículo de este tipo, en un plano tiene 3 grados de libertad totales: posición X, posición Y orientación. Pero tiene solo dos actuadores que controlan el ángulo de giro y la velocidad.

La consecuencia de esto es que existe una limitación de cual sera el camino entre dos estados en ese espacio. En el caso particular del vehículo Ackermann las trayectorias posibles que puede realizar están limitadas curvas circulares desde un radio mínimo denominado radio de giro.

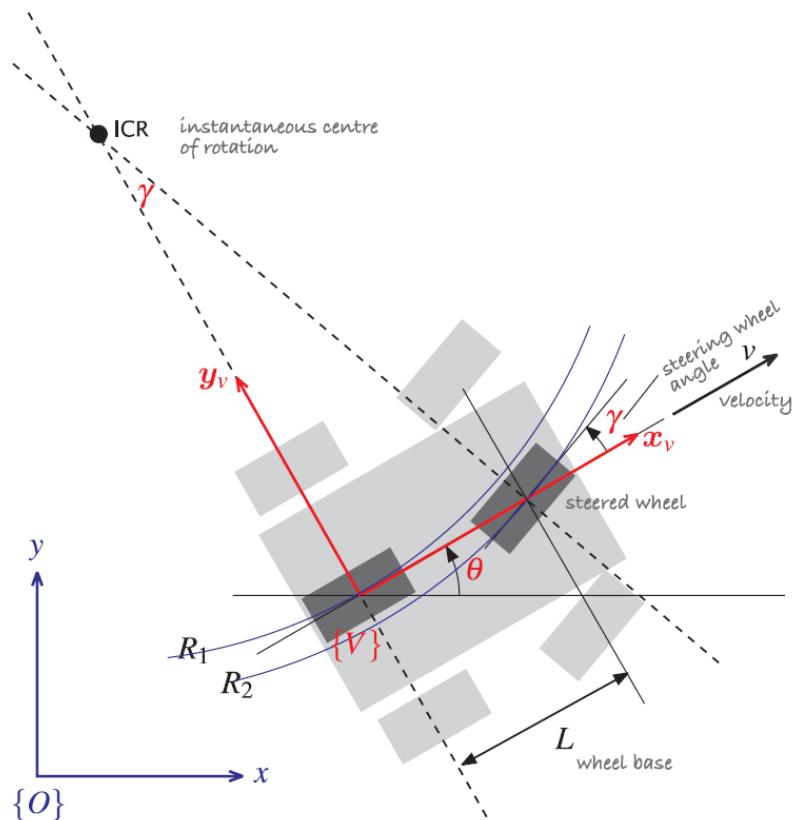


Figura 5-13: Modelo de vehículo con dirección Ackermann

Para que el Sistema se comporte correctamente con la menor Perdida de energía, las ruedas no patinen y gire efectivamente sobre la circunferencia, el vector del eje de rotación de cada una de las ruedas tiene que cruzarse en el eje de rotación Z del Sistema. Esto implica que la velocidad de desplazamiento de cada una será distinta, en consecuencia, la velocidad angular de cada una también lo será.

Debido a que el vehículo dispone de un solo motor y es indispensable que las ruedas tengan velocidades de giro distinto ante cambios de dirección, se colocó un diferencial que garantiza una transferencia de potencia tractora a la vez que permite giro a distinta velocidad.

En primer lugar, se pensó fabricar el diferencial utilizando tecnología de impresión 3D, pero luego de verificar las propiedades mecánicas del material se optó por adquirir un diferencial comercial. Si bien soportaría las tensiones de trabajo, la baja dureza del material implica una muy baja vida útil.

El diferencial adquirido es de tipo abierto, como se ve en la siguiente Figura.

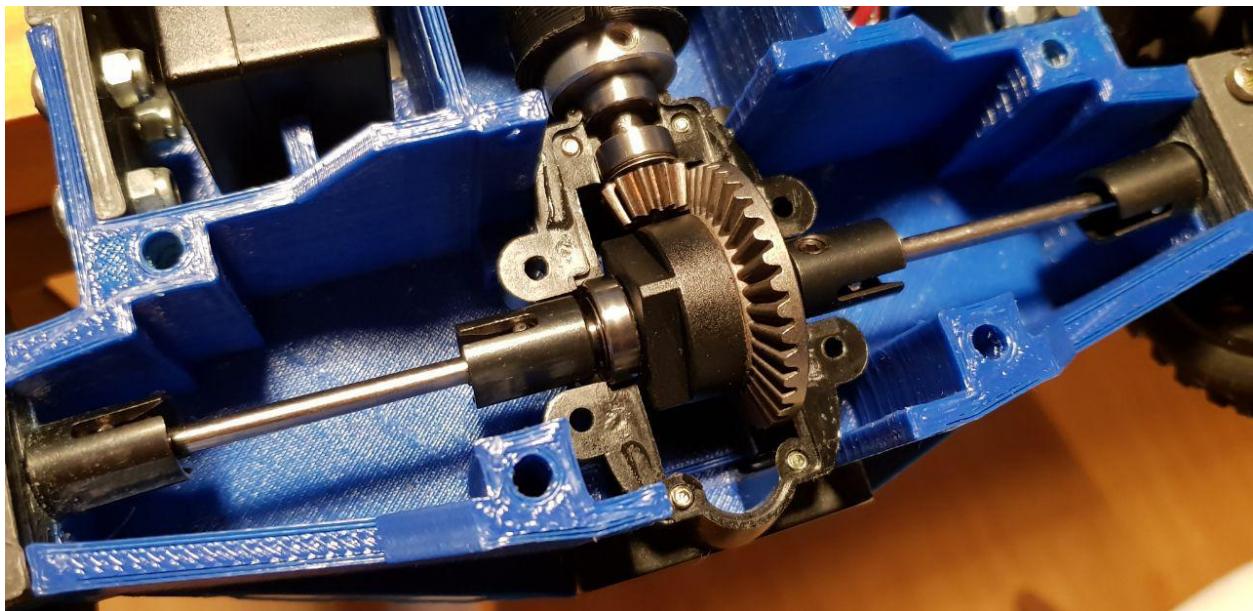


Figura 5-14: Diferencial abierto colocado en la plataforma junto con las cazoletas y los dogbone.

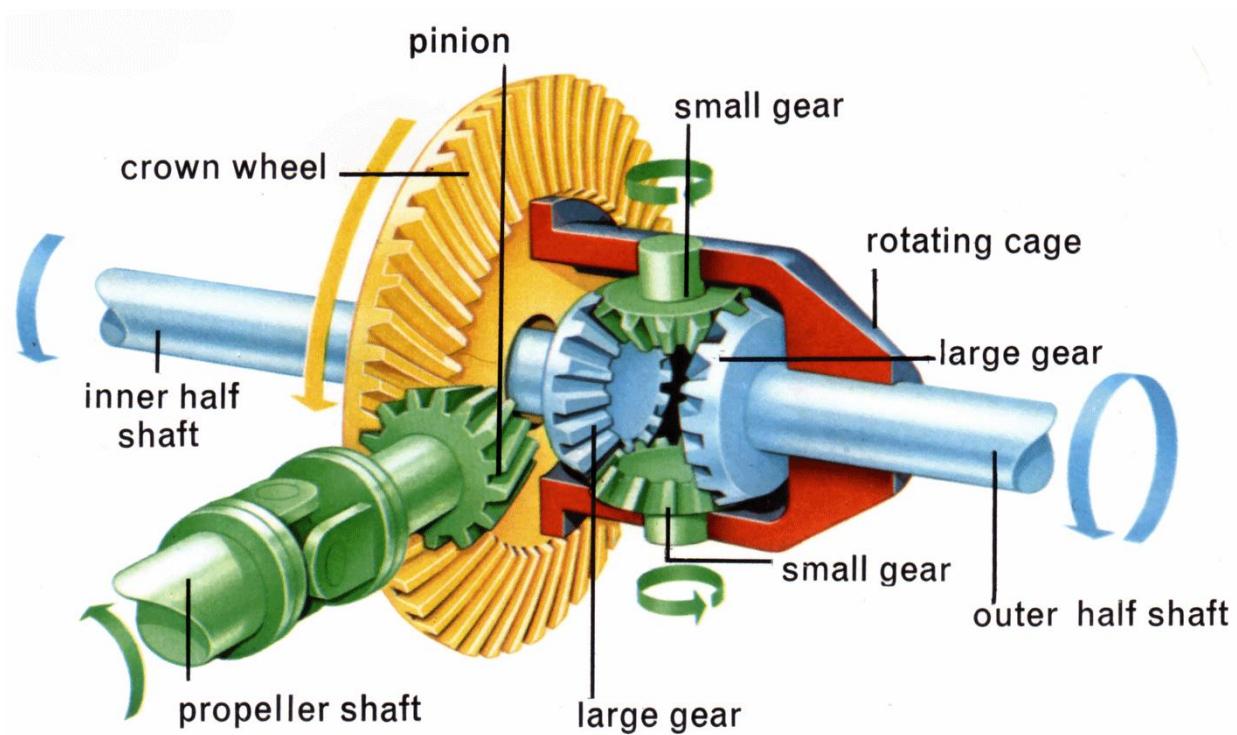


Figura 5-15: Detalle de diferencial Abierto

Los engranajes que gobiernan la relación de la caja son el piñón y la corona las cuales constan con la siguiente cantidad de dientes.

$$n_p = 11 \quad n_c = 35$$

Las cuales dan una relación de velocidades de:

$$e = \frac{n_p}{n_c} = 0.3142$$

Estos diferenciales son estándar para el mercado de vehículos a escala y es relativamente sencillo adquirir repuestos en caso de falla.

5.7.2.3 Diseño

Teniendo las piezas centrales se diseñó la pieza de soporte para que contenga al conjunto de trasmisión considerando, además:

- Rodamientos
- Acople entre eje de motor y diferencial
- Cazoletas
- Dogbone
- Distancia entre ruedas
- Rigidez
- Modular

Y por último que sea posible el montaje. En la siguiente figura se puede ver el diseño isométrico, para más detalle se puede ver el plano en el Anexo.

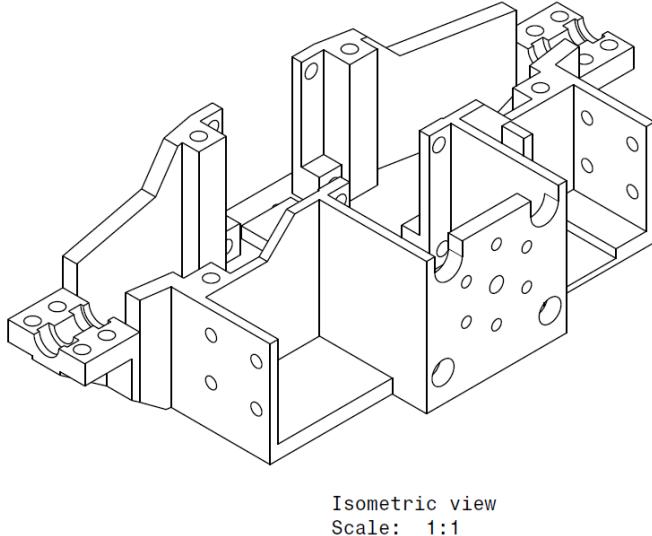


Figura 5-16: Modulo de tracción

Como cada módulo debe ser desmontable se diseñó con tal objetivo. Se tomaron las dimensiones del diferencial real y se modeló, lo mismo con el resto de las piezas comerciales.

A continuación, se puede ver el módulo de tracción con los componentes ensamblados.

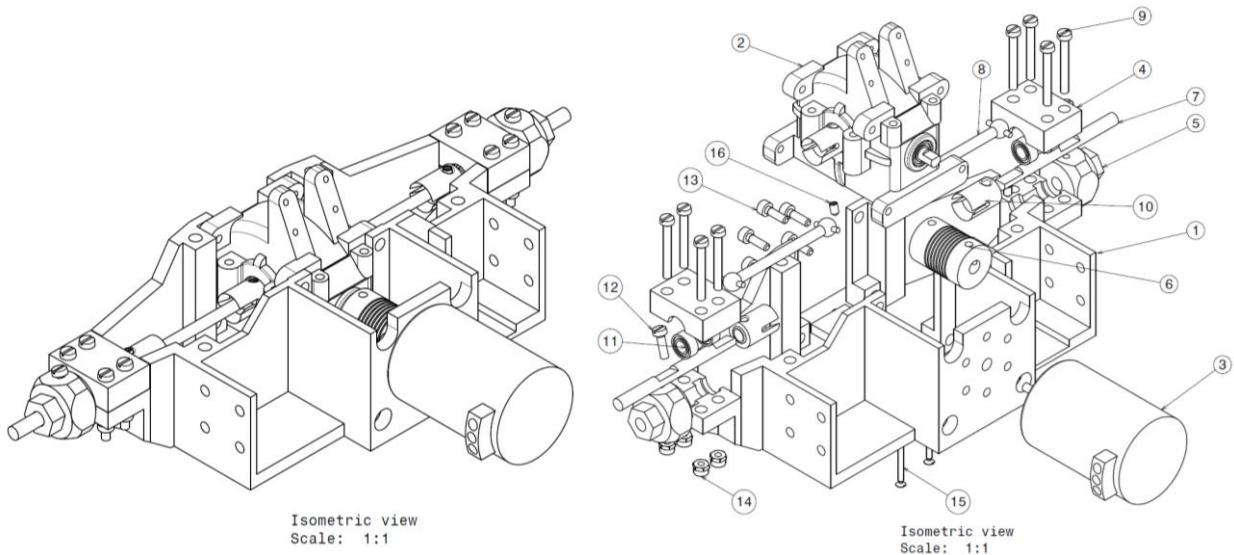


Figura 5-17: Modulo de tracción ensamblado

Para el detalle de los componentes del ensamble, se puede acceder al anexo. Como se puede ver el diseño consta de 8 agujeros en los laterales los cuales tienen la función de acople entre este módulo y el siguiente.

La característica principal del diseño es que fue realizado contemplando las capacidades técnicas del método de fabricación. Se utilizó la técnica de impresión 3D de depósito para la fabricación la cual tiene ciertas características que restringen el diseño.

Si bien se tiene la capacidad para imprimir virtualmente cualquier geometría, existen ciertas limitaciones que determinan la calidad del producto final. Durante el proceso de fabricación se explicarán los detalles.

5.7.3 Chasis

La siguiente pieza importante, se diseñó con el objetivo de dejar espacio suficiente para el alojamiento del resto de los componentes y con objetivos estructurales para unir la dirección con la tracción.

En la figura siguiente se puede ver la pieza, y se aprecian alojamientos los cuales están destinados a la batería y al controlador del motor.

Para su diseño se consideró el acople entre módulos, la rigidez y maximizar el espacio libre para colocar los sistemas de control.

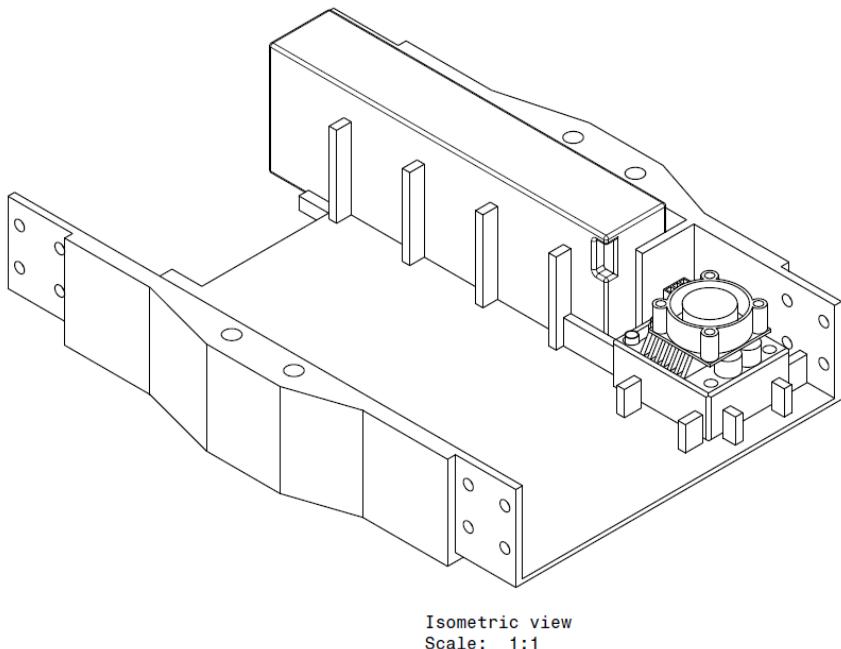


Figura 5-18: Render modulo chasis.

5.7.4 Módulo de sensado (Radar)

Para mostrar el funcionamiento de la plataforma se planteó como objetivo que la plataforma pueda alcanzar un punto en el espacio esquivando obstáculos, para esto fue necesario determinar cuál sería el sistema de detección del entorno. Inicialmente se evaluó la posibilidad de realizar la detección mediante la medición por ecos ultrasónicos, pero luego se descartó y se eligió en su lugar utilizar sensores infrarrojos.

Como el vehículo es no holonómico y tiene restringido su movimiento a curvas radiales se determinó que el sensado solo sería útil en un rango de 180 grados centrados en la dirección frontal.

Para este tipo de plataformas el mejor tipo de sensor son los denominados LIDAR, los cuales permiten un escaneo con un retraso inferior al milisegundo en 360 grados con una resolución angular menor a un grado. Debido a que en Argentina no es posible acceder a este tipo de tecnología con un presupuesto acotado se

buscó simular este tipo de sensor utilizando 5 sensores infrarrojos separados 45 grados montados sobre una plataforma rotatoria tal que pueda aumentar la capacidad de escaneo.

Los sensores utilizados son los Sharp GP2Y0A02YK0F los cuales entregan una señal analógica proporcional a la distancia medida. Para el actuador se utilizó el mismo servo que el utilizado para la dirección (MG996R).

En primer lugar, se fabricó un radar con los sensores separados 60 grados, dando una apertura de 300 grados, pero luego de distintos ensayos se optó por una separación de 45 grados el cual se puede ver en la siguiente figura.

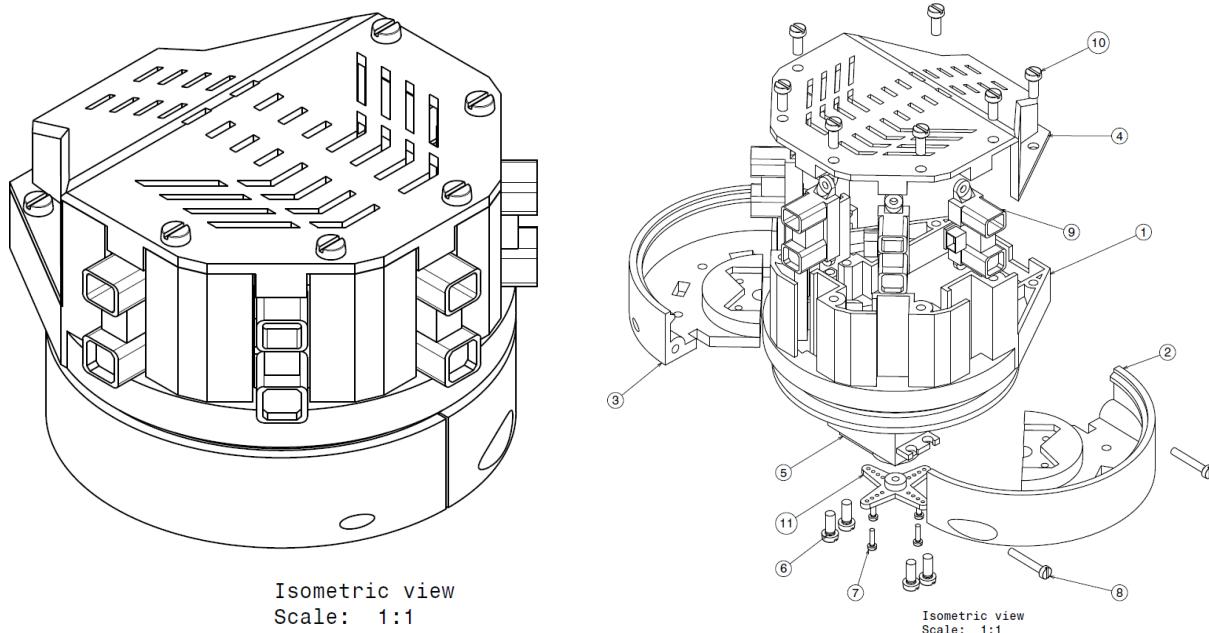


Figura 5-19: Radar móvil

Para fijar este módulo se diseñó una pieza con encastres genéricos tal que se pueda colocar en cualquier posición y coincida con los soportes.

5.7.5 Ensamble completo y módulo de control

Una vez diseñados cada módulo se montó cada uno en su posición y se verificó que todas las piezas fueran posibles de ensamblar y que cada una cumpla su función. Con respecto al módulo de control se diseñó una pequeña caja que contenga a un Arduino nano y sea posible programarlo aún colocado en el vehículo.

En la siguiente imagen se puede ver el ensamble completo, en el anexo se encuentran los detalles de cada módulo.

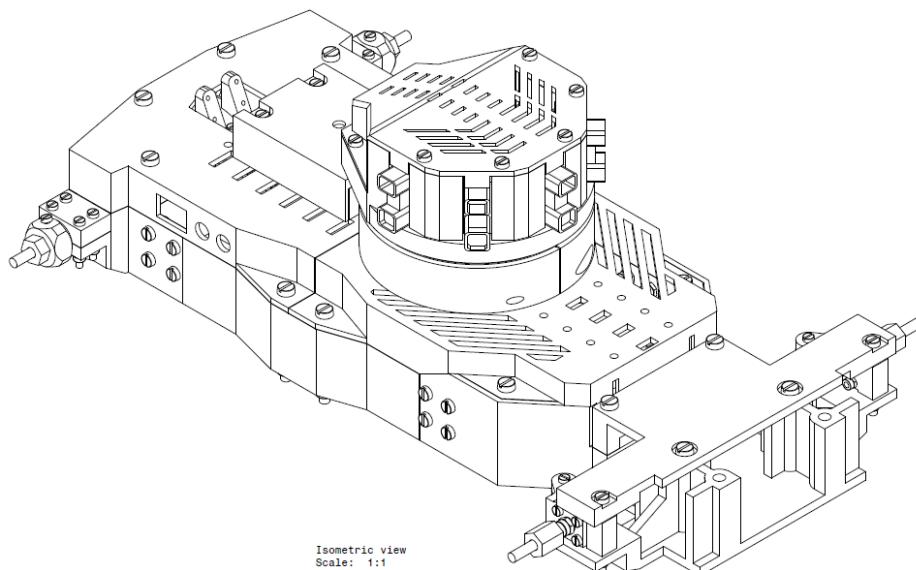
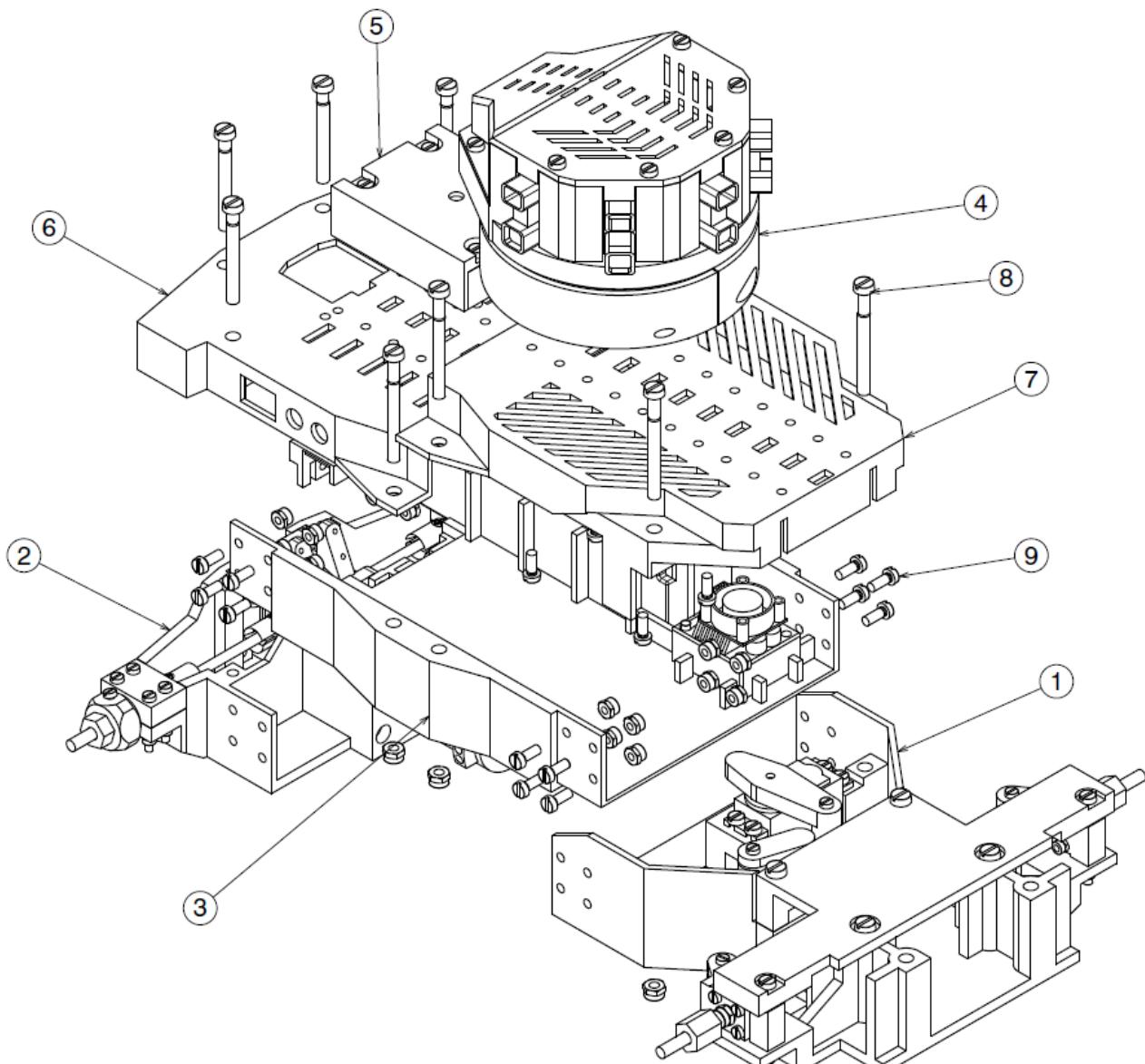


Figura 5-20: Plataforma ensamblada



Isometric view
Scale: 1:2

Numero	Codigo	Descripcion	Cantidad
1	P01-D04	Modulo de direccion	1
2	P01-T04	Modulo de traccion	1
3	P01-CH04	Modulo de chasis	1
4	P01-R02	Modulo de radar	1
5	P01-CT01	Modulo de control	1
6	P01-TF04-I3D-L04-W20-IF50	Tapa frontal	1
7	P01-TP04-I3D-L04-W20-IF50	Tapa posterior	1
8	ISO-1207-M5-X50	Bulon M4X50	10
9	ISO-1207-M4-X10	Bulon M4X10	20

Figura 5-21: Plataforma explotada en módulos

6 DISEÑO ELECTRÓNICO

6.1 ARQUITECTURA ELECTRÓNICA

La electrónica del vehículo sera modular intentando seguir la modularización de diseño mecánico por lo tanto se diseñaron tres módulos de hardware individuales internos al vehículo y un módulo externo para control remoto del mismo:

- Modulo central
- Modulo radar
- Módulo de control
- Modulo remoto

Los primeros tres se intercomunican mediante i2c y el módulo central se comunica mediante RF con el módulo Remoto.

6.1.1 Módulo central

El módulo central es el encargado de manejar las comunicaciones, comandar la velocidad del motor y la dirección de las ruedas, sensar la posición, el heading del vehículo y leer el encoder del motor.

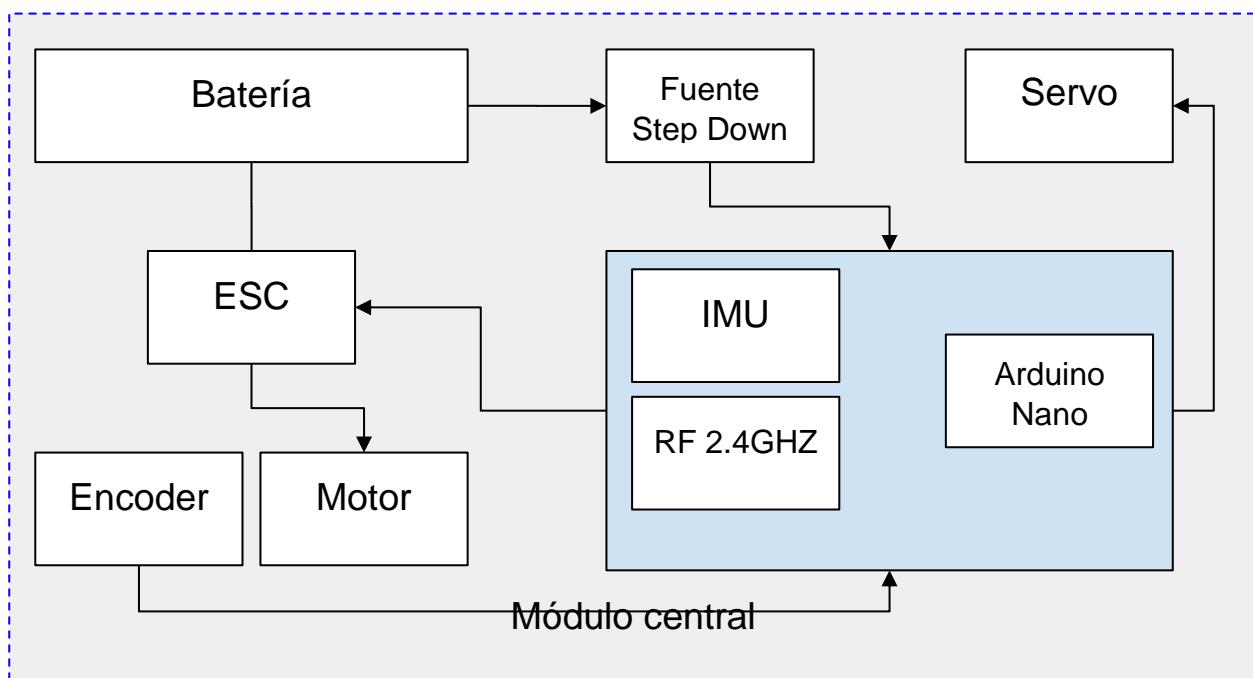


Figura 6-1: Arquitectura electrónica del módulo central

Este módulo esta a su vez conectado con los otros dos módulos restantes mediante I2C.

6.1.2 Módulo de radar

El módulo de radar es el encargado de obtener las mediciones de los sensores de distancia, filtrarlas, comandar al servo de dirección y obtener la posición angular respecto a la posición original (mirando hacia el frente).

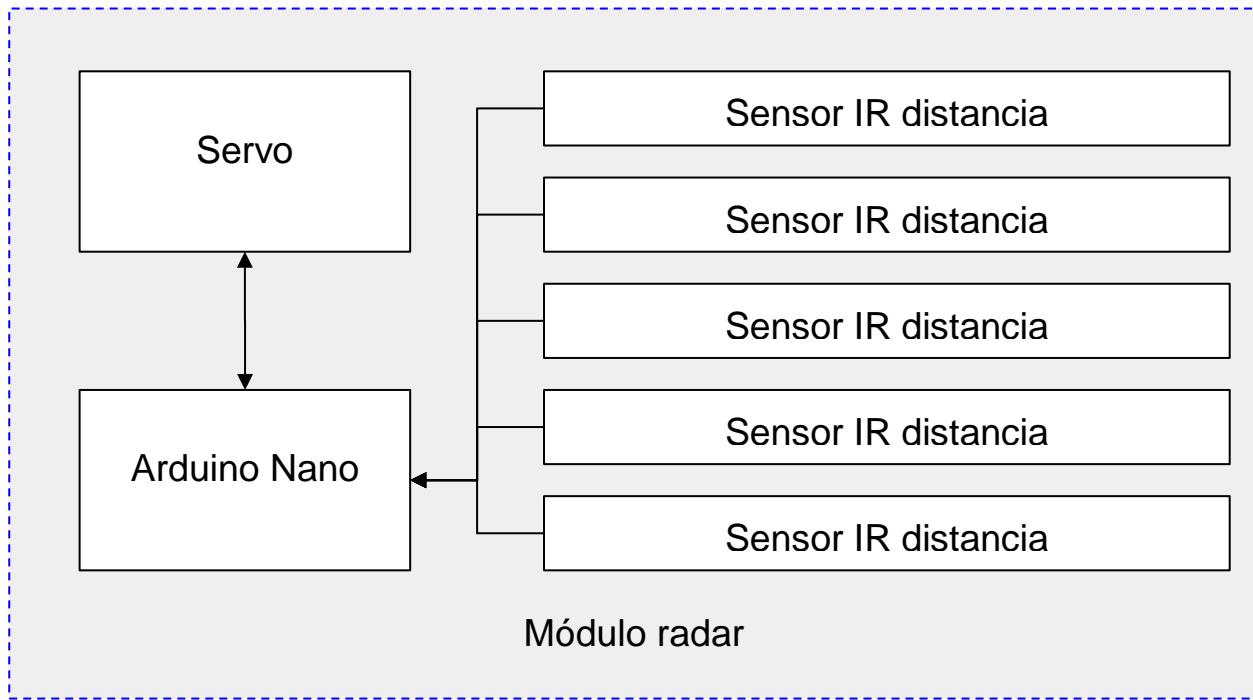


Figura 6-2: Arquitectura electrónica del módulo de radar

6.1.3 Módulo de control

Este módulo consta de solamente un Arduino nano el cual recibe los datos de los sensores pre-procesados por los otros módulos y devuelve la velocidad y giro hacia el módulo central para que ordene al motor y al servo dirección el nuevo setpoint.

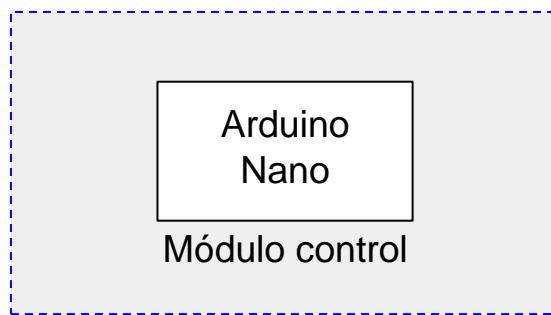


Figura 6-3: Arquitectura electrónica del módulo de control

6.1.4 Módulo remoto

El módulo remoto consta de un Arduino nano un módulo RF y un joystick con un botón para poder interactuar con el vehículo de forma remota sin necesidad de que el control remoto esté conectado a una PC.

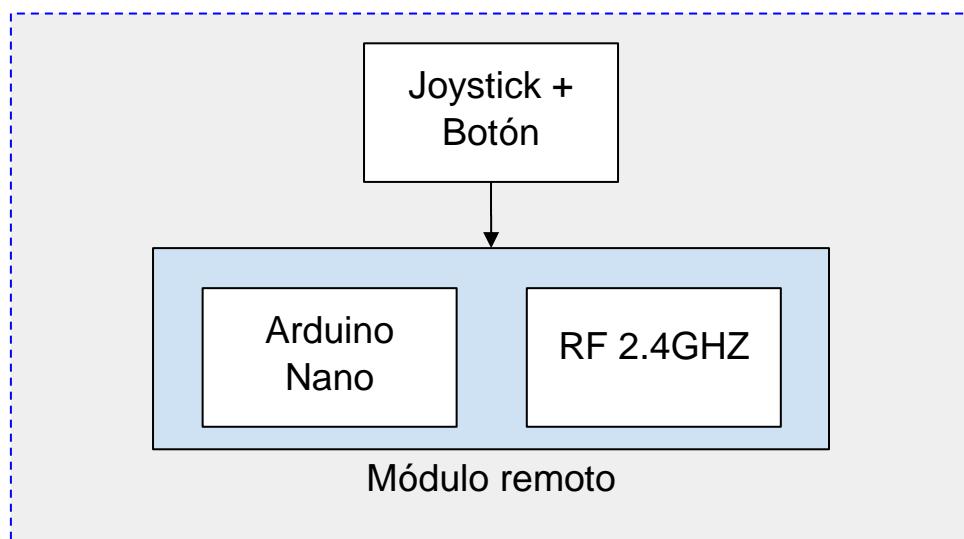


Figura 6-4: Arquitectura electrónica del módulo remoto

6.1.5 Arquitectura completa

En la arquitectura completa de hardware que se muestra a continuación se pueden ver flechas de colores distintos las cuales representan los tipos de comunicación entre módulos.

Las flechas rojas muestran una comunicación I2C bidireccional, la azul una comunicación Serial y la Gris punteada es una comunicación por radio frecuencia también bidireccional.

El objetivo con este tipo de arquitectura es que sea posible reemplazar los módulos, dejando el central, y que pueda seguir funcionando.

Por ejemplo, es posible remover el módulo de control y colocarle una raspberry pi conectada con i2c y el sistema sigue funcionando de la misma manera, pero con un “cerebro más grande”. Lo mismo es posible realizar con el módulo de sensores.

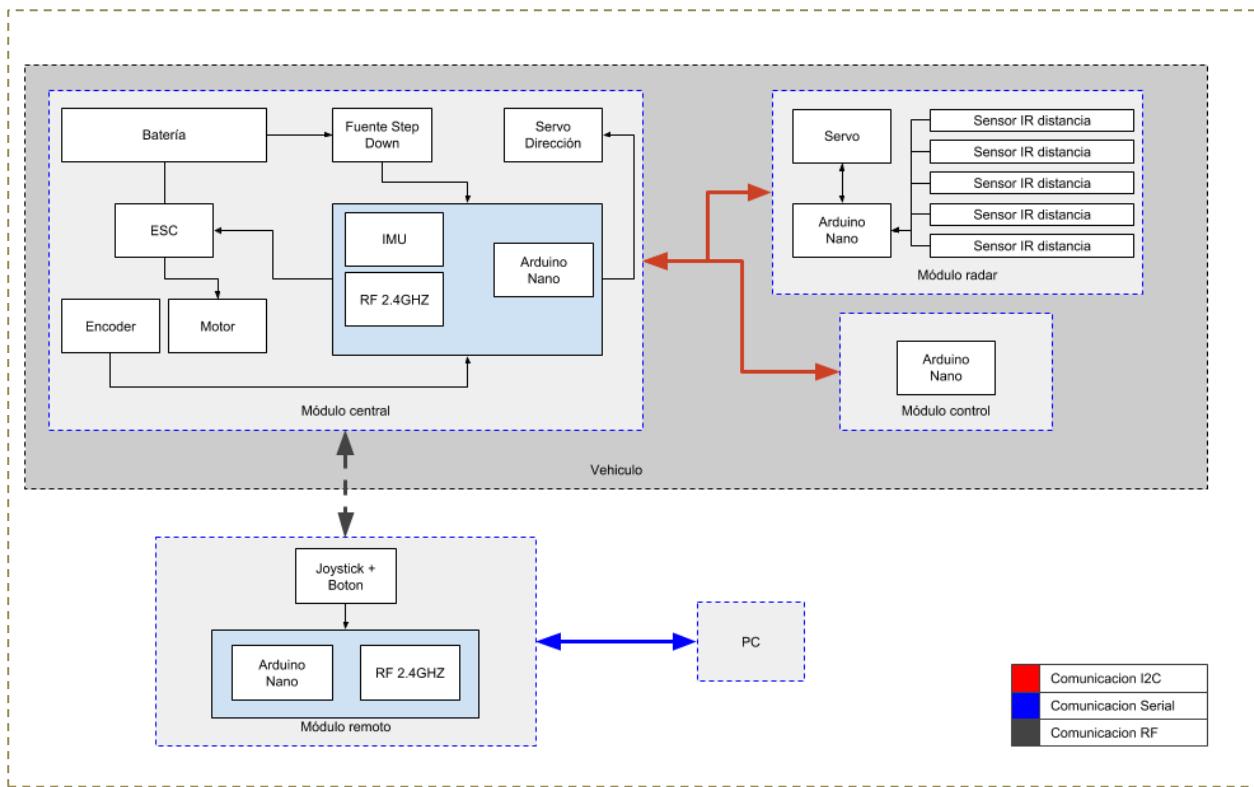


Figura 6-5:Arquitectura completa

7 MODELO

Antes de definir la arquitectura de software se determinó un modelo que permita realizar el control requerido.

7.1 MODELO CINEMÁTICO DE LA PLANTA

Como se explicó previamente la plataforma es no holonimica por lo tanto es necesario definir cuál es la transferencia de la planta. Para simplificar el modelo se utilizó el comúnmente llamado “Modelo de bicicleta” el cual reemplaza las 4 ruedas por dos en serie como muestra la figura siguiente.

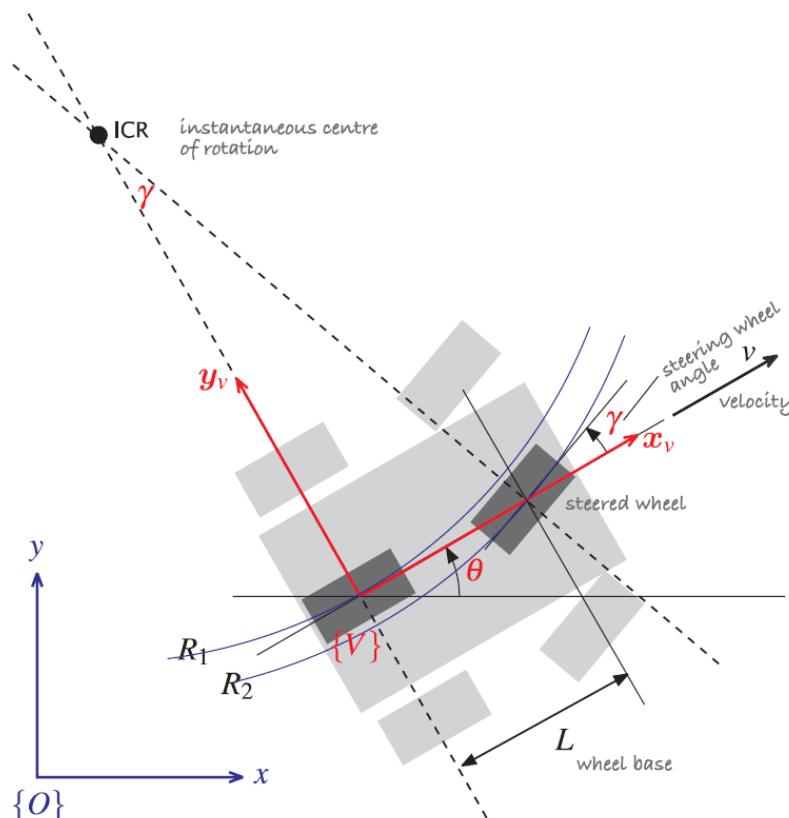


Figura 7-1: Modelo de bicicleta

Como se puede ver en la Figura 7-1: Modelo de bicicleta se definen 3 parámetros que dependen de la geometría del sistema: la distancia entre ejes “ L ”, y los radios de giro R_1 y R_2 los cuales están determinados por la intersección entre los vectores de rotación de las ruedas virtuales.

La posición en el plano $\{O\}$ está determinada por la posición del sistema de referencia $\{V\}$. En este plano el vehículo solo puede moverse en x a una velocidad v mientras que en y tiene velocidad nula. En resumen:

$$v_x = v \quad v_y = 0$$

Por otro lado, la velocidad de giro está determinada por la velocidad de desplazamiento y la distancia al centro de rotación ICR.

$$\dot{\theta} = \frac{v}{R1} \quad R1 = \frac{L}{\tan(\gamma)}$$

El ángulo gamma es el ángulo de giro establecido por el mecanismo impulsado por el servomotor.

De acuerdo con este planteo las ecuaciones cinemáticas en el plano {O} son:

$$\dot{x} = v \cdot \cos(\theta)$$

$$\dot{y} = v \cdot \sin(\theta)$$

$$\dot{\theta} = \frac{v}{L} \cdot \tan(\gamma)$$

Como se puede ver es un sistema no lineal y no es posible obtener una relación entre el ángulo θ y las coordenadas x e y. Ademas se puede observar que cuando la velocidad es nula, no es posible que el vehículo gire. Es decir, si $v=0$ entonces $d\theta/dt = 0$. Por otro lado, es posible obtener la velocidad angular ($d\theta/dt$) mediante la medición de un giroscopio.

De acuerdo a los ensayos realizados en [...] de mostro que el modelo cinemático se comporta mejor que el modelo dinámico para sistemas de control. Por este motivo, se obviará el desarrollo dinámico de la plataforma, debido no solo a que no es necesario si no que la incertidumbre incluida en los coeficientes de las propiedades mecánicas de los componentes ampliarían el error haciéndolo aún menos performante.

7.2 MODELO DE ACTUADORES

En este sistema existen dos actuadores bien definidos: el motor, encargado de otorgarle velocidad lineal a la plataforma, y el servomotor de dirección que se encarga de la orientación de esta.

Ambos actuadores tienen como entrada de control una señal PWM. En el caso del servo, el duty cycle corresponde a una posición angular definida y está controlada por un sistema de control interno.

7.2.1 Motor – Control de velocidad

En el caso del motor, la señal PWM viaja al controlador y este se encarga de transformarla en una secuencia de magnetización para el motor trifásico a una tensión determinada por el PWM. A falta de una función transferencia entre el PWM y la velocidad angular y la falta de información sobre el controlador ESC se realizó un sistema de control de velocidad externo que ajusta el PWM de acuerdo con la velocidad media en la salida del eje del motor con un encoder óptico.

Se aplicó un controlador PID ajustado mediante ensayos para lograr un arranque suave sin sobre pico.

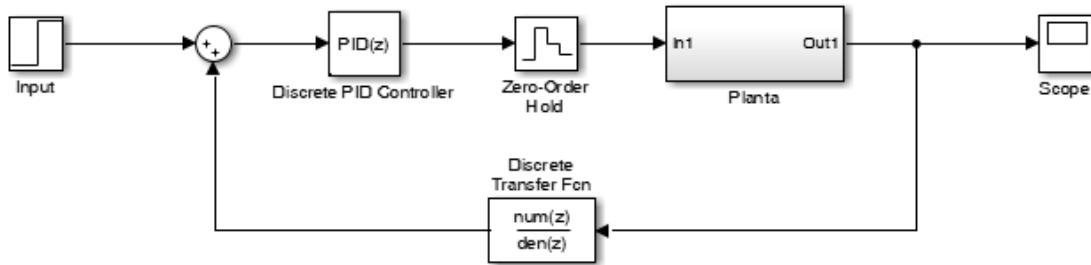


Figura 7-2: Modelo de control de motor brushless

El sistema se planteó como se muestra en la Figura 7-2 siendo la Planta una caja negra y el encoder con una trasferencia de primer orden sin retraso.

El encoder fue colocado sobre el acople flexible y es capaz de entregar un pulso por vuelta de motor, lo cual implica con la relación del diferencial:

$$e = \frac{n_p}{n_c} = 0.3142$$

Un pulso cada 0.3142 vueltas de la rueda, aproximadamente 3.18 pulsos por revolución. Midiendo el tiempo T entre cada pulso se puede obtener la velocidad lineal v en el plano {Y} mediante:

$$v = \frac{2\pi}{T} * r * \frac{1}{e}$$

Siendo r el radio de la rueda y e la relación del diferencial.

7.2.2 Servo

El servo de dirección tiene su propio PID para garantizar su posición, pero por otro lado la salida del mecanismo es el ángulo *gamma* que se busca controlar. Para eso es necesario determinar una relación entre el ángulo η del servo con el ángulo de giro de las ruedas.

El mecanismo este compuesto por 3 sub-mecanismos de 4 barras los cuales plantean un desafío a la hora de obtener una relación analítica, debido a la complejidad de cálculo y a la diferencia angular de cada rueda. En el modelo cinemático de bicicleta planteado el giro individual de cada rueda se tiene que transformar en un giro único.

Para simplificar este problema se tomó una relación lineal entre el ángulo de entrada y ángulos de salida y luego se promedió. La relación lineal se tomó a partir de la simulación del sistema buscando los ángulos mínimos y máximos de rotación posibles. Esta relación está definida por la siguiente tabla.

	Angulo mínimo	Angulo central	Angulo máximo
Servo	44.027	87.06	141.76
Rueda izquierda	-33.393	0	45.43
Rueda derecha	-45.43	0	33.393
Promedio	-39.41	0	39.41

Como se puede ver en la tabla, hay una diferencia importante entre la dimensión del rango de giro hacia la izquierda en relación con el giro hacia la derecha. Esto es debido al diseño del mecanismo de barras el cual fue realizado de esta manera para poder utilizar el mayor rango posible del servo sacrificando la simetría. Esto tiene como consecuencia que sean necesario dos ajustes lineales diferentes para los dos giros.

Tomando los valores de la tabla se obtuvo la siguiente función que relaciona el ángulo del servo η y el ángulo del modelo cinemático.

$$\eta = \begin{cases} 1.388 * \gamma + 87.06 & \gamma \geq 0 \\ 1.0919 * \gamma + 87.06 & \gamma < 0 \end{cases}$$

7.3 DEAD RECKONING

Para estimar la posición se utilizaron 2 sensores, un giroscopio y un encoder. Se utilizó la técnica de dead reckoning la cual consta en la integración de las mediciones de posición y giro a partir de la posición original utilizando el modelo cinemático para obtener la posición de la plataforma en el plano {O}.

Esta técnica tiene la ventaja que es simple su implementación y no requiere de referencias externas, pero tiene una gran desventaja por el incremento del error a medida que se estiman las mediciones.

Básicamente este método funciona sumando las posiciones previas a la actual.

$$\begin{aligned} x_{k+1} &= x_k + (\Delta x_k + \xi_x) \\ y_{k+1} &= y_k + (\Delta y_k + \xi_y) \\ \theta_{k+1} &= \theta_k + (\Delta \theta_k + \xi_\theta) \end{aligned}$$

Si se expande cada sumatoria se puede ver que el término ξ se va sumando en cada iteración, este representa el error en la medición k.

Cuando k crece, el error crece, por lo tanto, la posición real de la plataforma en relación con la estimada a medida que pasa el tiempo diverge. Este problema se suele solucionar mediante la utilización de otros sensores de referencia absoluta como el GPS o reconocimiento de referencias mediante procesamiento de imágenes que corrigen la estimación “reiniciando” el error.

En este caso no se utilizaron sensores de referencia, por lo tanto, es de esperar que el error se acumule.

Utilizando el modelo cinemático presentado previamente, mediante la diferenciación del modelo se pudo obtener la posición y la orientación de la plataforma a partir de las mediciones de los sensores.

$$\dot{x} = v \cdot \cos(\theta)$$

$$\dot{y} = v \cdot \sin(\theta)$$

$$\dot{\theta} = \frac{v}{L} \cdot \tan(\gamma)$$

Utilizando la siguiente ecuación y la medición de los sensores:

$$v_k = \frac{2\pi}{T_k} * r * \frac{1}{e}$$

$\dot{\theta}$: Sensado de giroscopio

Y las mediciones de los sensores:

$$\dot{x} = \frac{\Delta x}{\Delta t} \quad \dot{y} = \frac{\Delta y}{\Delta t} \quad \dot{\theta} = \frac{\Delta \theta}{\Delta t}$$

Reemplazando

$$\begin{aligned}
 \Delta\theta_k &= Sense * \Delta t & \Delta x &= v_k \cdot \cos(\theta_k) * \Delta t & \Delta y &= v_k \cdot \sin(\theta_k) * \Delta t \\
 \theta_{k+1} &= \theta_k + (\Delta\theta_k + \xi_\theta) & x_{k+1} &= x_k + (\Delta x_k + \xi_x) & y_{k+1} &= y_k + (\Delta y_k + \xi_y) \\
 \theta_{k+1} &= \theta_k + (Sense * \Delta t & x_{k+1} &= x_k + (v_k \cdot \cos(\theta_k) & y_{k+1} &= y_k + (v_k \cdot \cos(\theta_k) \\
 &\quad + \xi_\theta) & * \Delta t + \xi_x) & * \Delta t + \xi_x) & * \Delta t + \xi_y)
 \end{aligned}$$

De esta manera se obtiene la posición en el plano y la orientación. Siempre y cuando el error se mantenga acotado los valores serán representativos. Estos errores dependerán principalmente de las condiciones de funcionamiento de la plataforma, para poder estimar el error sera necesario una serie de ensayos. Inicialmente este error se supondrá lineal con el tiempo y menor al 10 % durante la duración de los ensayos estimados en el orden de un minuto.

7.3.1 Giroscopio

El giroscopio utilizado es el incluido en el IMU – MPU9250 y se lo configuro para medir a escala completa de 2000 grados por segundo tal que teniendo mayor precisión se pueda reducir el error y poder tener una medición correcta el mayor tiempo posible.

Ademas se estableció que el sensor sume en la secuencia solo cuando el vehículo está en movimiento, es decir, cuando existe un comando al controlador del motor.

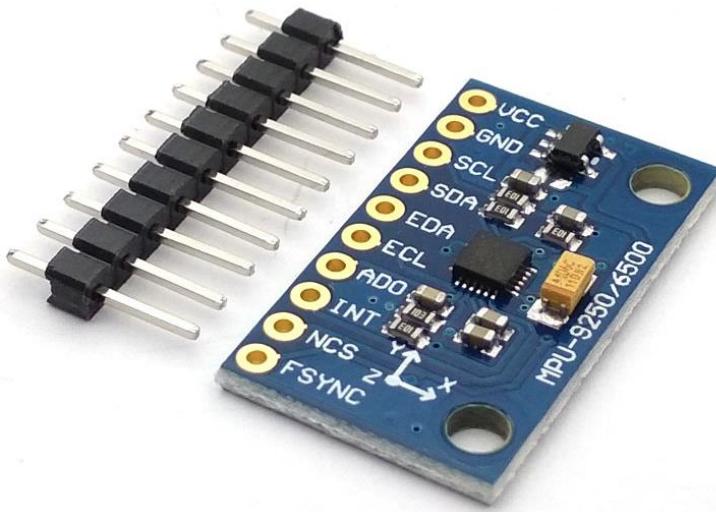


Figura 7-3: MPU-9250

El sensor, está conectado al módulo central mediante los pines SCL y SDA de comunicación y reciben alimentación por los pines VCC y GND. Esta conexión permite la comunicación i2c con el Arduino del módulo central, el valor de la velocidad de giro sera enviado como un paquete de 2 bytes el cual representa un rango de 2000 grados por segundo. Esto quiere decir que el valor en unidades SI deberá ser convertido como:

$$Sense = raw_sense * \frac{2000}{32767.5} \frac{rad}{s}$$

7.3.2 Encoder

El encoder es utilizado para medir la velocidad angular del eje del motor. Para esto se imprimió una pieza que se colocó fija al acople flexible la cual interrumpe un haz infrarrojo del módulo de encoder mostrado en la siguiente figura.

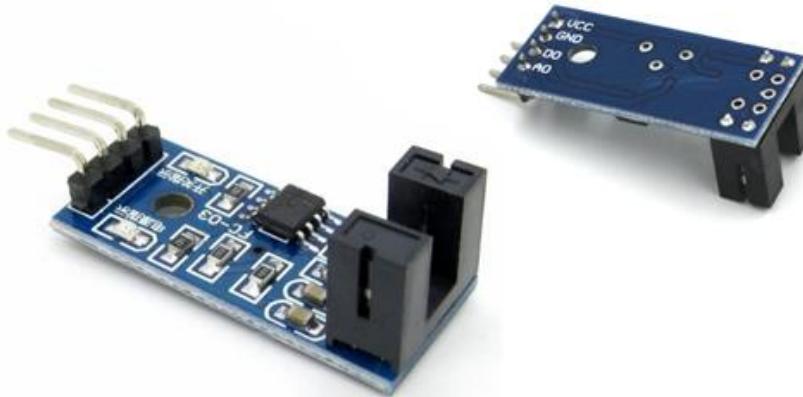


Figura 7-4: Sensor infrarrojo de barrera. Encoder.

Este dispositivo es alimentado por el módulo central y por el pin D0 devuelve un pulso cada vez que un objeto interrumpe el haz infrarrojo. Este pin fue conectado a uno de interrupción del Arduino central y medirá en segundos el tiempo T.

8 CONTROL DE OBJETIVO Y DETECCIÓN DE OBSTÁCULOS

8.1 CONTROL DE OBJETIVO

Teniendo la capacidad de determinar la ubicación de la plataforma es posible determinar un destino con respecto al origen del plano {0}.

Si se determina como un punto (x_1, y_1) al objetivo y al (x_0, y_0) a la posición del vehículo, existirá un vector (x_1-x_0, y_1-y_0) que definirá el ángulo β y la distancia ρ que el vehículo debe tomar para alcanzar ese objetivo.

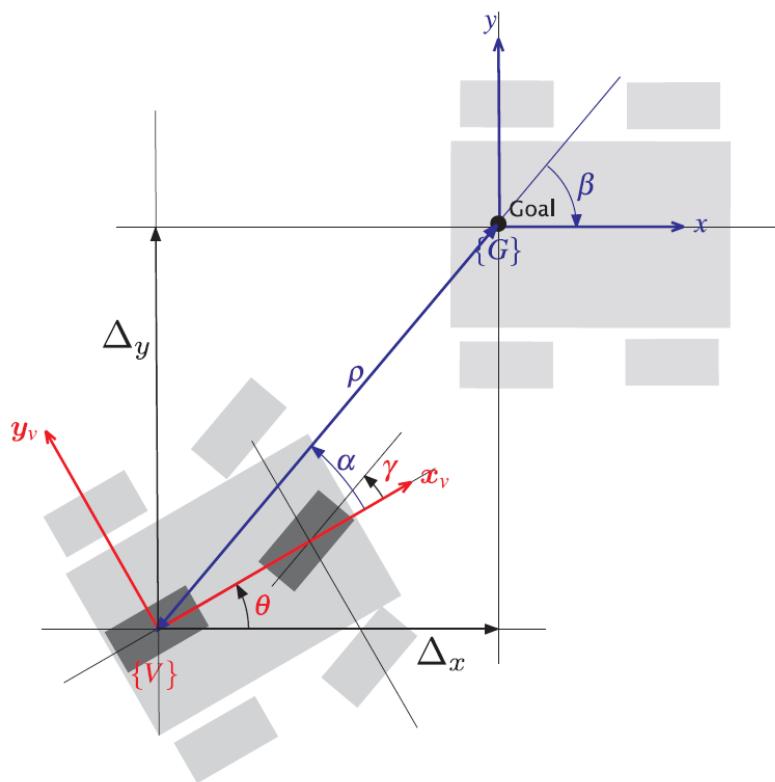


Figura 8-1: Diagrama origen -objetivo

El vehículo deberá girar sus ruedas un ángulo γ tal que pueda alcanzar ese objetivo, y este se puede definir como la diferencia entre el ángulo θ y el ángulo β .

$$\alpha = \beta - \theta$$

Este ángulo se define como ángulo objetivo, o dicho de otra manera es el ángulo de set-point al cual el vehículo debe llegar. Utilizando el modelo de bicicleta se puede plantear un sistema de control de lazo cerrado tal que el error se minimice.

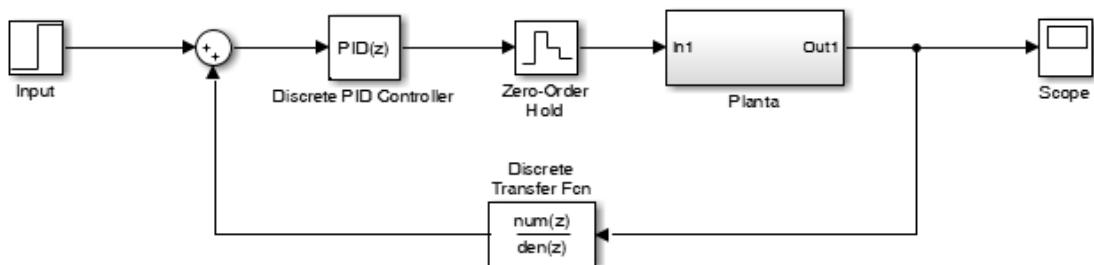


Figura 8-2: Modelo de control de lazo cerrado de orientación

Utilizando el mismo modelo planteado para el sistema de control de velocidad, se ajustaron los parámetros para que el controlador sea de tipo proporción para lograr una alineación sin sobre pico lo más veloz posible.

Por otro lado, para alcanzar un objetivo es necesario que el vehículo reduzca la velocidad inversamente proporcional a la distancia al objetivo. Por este motivo se estableció la siguiente relación:

$$v = Kv * \sqrt{(x1 - x0)^2 + (y1 - y0)^2}$$

Siendo Kv una constante de ajuste.

Este tipo de control no contempla todos los estados, solamente busca que la plataforma alcance el destino, pero sin especificar con que orientación.

8.2 DETECCIÓN DE OBSTÁCULOS

Con el control previo es posible hacer que el vehículo alcance un punto especificado, pero no cuenta con la capacidad de cambiar el recorrido con tal de evitar colisiones.

Para solucionar este problema, existen soluciones bien establecidas las cuales proponen técnicas de todo tipo pero ninguna de esas técnicas es apta para sistemas con poco poder de procesamiento.

La más útil, y con mejores resultados para plataformas en entornos indoor es la llamada “*probability grid map*” la cual se basa en la generación de un mapa del entorno dividido en celdas, a las cuales se le asigna un valor que corresponde a la probabilidad de que haya algo en ese lugar. Esta probabilidad es calculada mediante una función de densidad de probabilidad del sensor del robot y actualizada en cada delta temporal. Ademas se utilizan técnicas de filtrado de partículas para corregir la estimación mediante el método de Montecarlo.

Con respecto al poder de computo necesario. A medida que el espacio físico crece, la matriz que representa a ese espacio también lo hace. Teniendo en cuenta que cada celda tiene guardado un numero de tipo float o similar de 32 bits, un mapa de 10mx10m con una discretización de 10 cm equivale a 4Mb. En el Arduino solo se dispone de una memoria de 1Kb.

Teniendo en cuenta la potencia limitada se propuso una nueva técnica de detección y esquivado de obstáculos.

8.2.1.1 Sensores

El arreglo de sensores está ubicado aproximadamente a la distancia media entre los ejes. Los sensores están dispuestos a 45 grados y tienen la capacidad de girar libremente en conjunto. En la siguiente imagen se pueden ver las características del módulo de sensado.

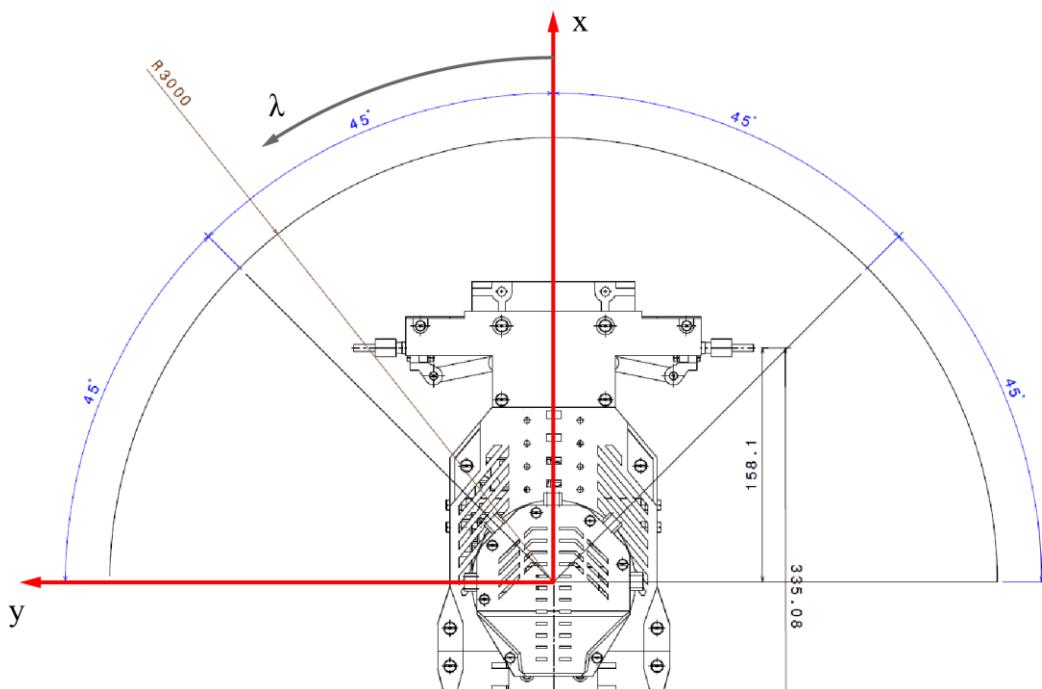


Figura 8-3: Deposición de sensores

Cada uno de los sensores mide distancia de acuerdo con la siguiente curva. La salida de sensor es una tensión inversamente proporcional a la distancia.

La curva se sigue extendiendo más allá de los 150 cm, pero como el error de la discretización del conversor aumenta, el fabricante limitó su distancia.

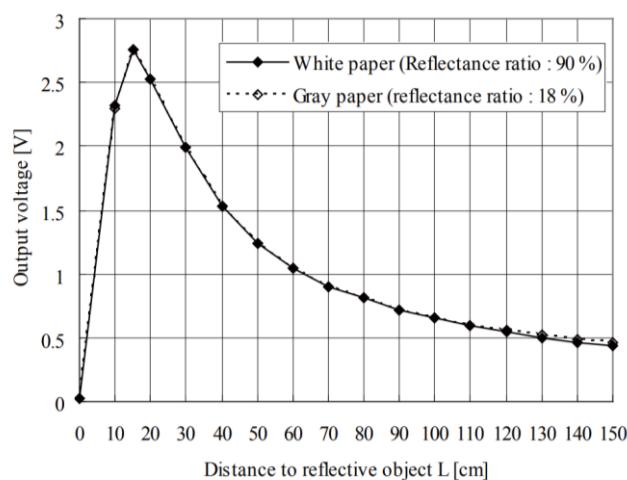


Figura 8-4: Puntos de medición

A partir de esta serie de punto se determinó que la función sera ajustada con una de potencias a partir de los 15 cm donde es decreciente. Luego realizando una serie de mediciones se obtuvo la función que modela al sensor:

$$L = 64.21 * V^{-0.9464}$$

Siendo V la tensión medida por el conversor analógico digital del Arduino. Este conversor se configuró con 16 bits dando como resultado una discretización de:

$$\Delta V_{min} = \frac{5V}{2^{16}} = 4.88E - 3 V$$

Esta discretización permite una medición a 300 cm con un error de 2% de discretización lo cual es aceptable debido a que representan 6 cm en 3 metros. Para verificar el rango de validez se realizó una serie de mediciones a intervalos de 25 cm, para obtener el desvío a partir de la medición de la curva ajustada.

La siguiente figura muestra que el desvío aumenta considerablemente a partir de los 150 cm. Se considerará que la función de probabilidad es de tipo normal.

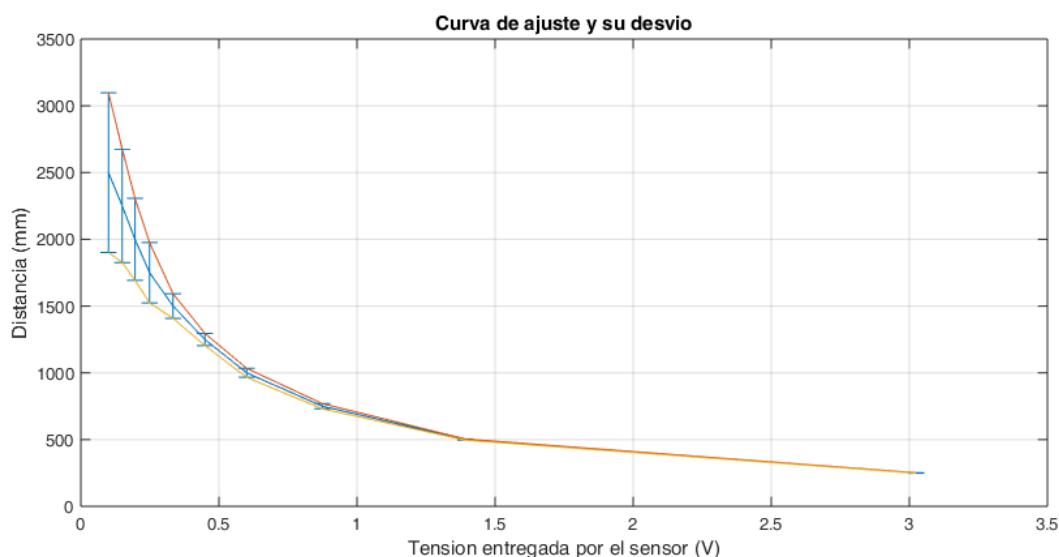


Figura 8-5: Desvío estándar de curva de ajuste

Si bien la incertidumbre es alta a partir de 150 cm es posible utilizar el sensor hasta los 300 cm debido a que la plataforma tiene que reaccionar a distancias menores, las cuales tienen un menor error.

8.2.2 Visual weight vector Estimation

Como la capacidad de computo está limitada se planteó resolver el sistema para evitar obstáculos a partir de la generación de un vector de pesos.

Como los sensores están dispuestos cada 45 grados, se generó un vector de 25 elementos, cada uno correspondiente a un ángulo del plano x-3 de la Figura 8-3.

Tabla 8-1: Vector de pesos

Angulo	-180	-165	-150	-135	-120	-105	-90	-75	-60	-45	-30	-15	0	15	30	45	60	75	90	105	120	135	150	165	180
Peso	3000	3000	3000	3000	3000	3000	3000	3000	3000	3000	3000	3000	3000	3000	3000	3000	3000	3000	3000	3000	3000	3000	3000	3000	

Como se puede ver en la tabla los marcados en gris son los ángulos correspondientes a la ubicación de cada uno de los sensores.

Si ahora, cada vez que existe una medición nueva por cada sensor se actualizan los pesos que corresponden a cada ángulo con la inversa de la medición con respecto al rango, es decir:

Peso = Rango maximo – Distancia medida

Si la plataforma se encuentra en un pasillo de 1 metro de ancho y 3 metros de largo el valor de los sensores sera el de la Tabla 8-2.

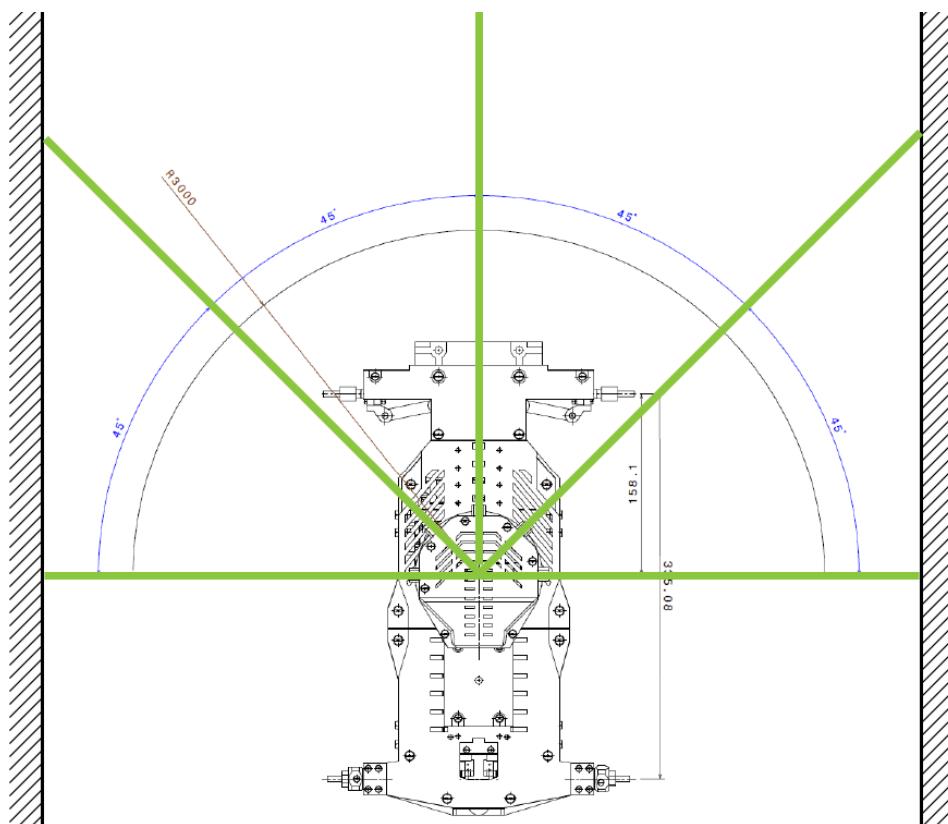


Figura 8-6: Plataforma en un pasillo

En este caso las mediciones de los sensores a -90 y 90 grados indicarán una medición de alrededor de 50 cm, los que están a -45 a 45 grados corresponderán a aproximadamente 70 cm mientras que el que está a 0 grados indicará 3 metros.

Tabla 8-2: Tabla de pesos cargada

Y luego se interpola linealmente entre cada sensor:

Tabla 8-3: Tabla de pesos interpolada

Con el vector de esta manera es posible tomar una decisión para evitar obstáculos, si se determina un valor de threshold por ejemplo de 1000 es posible definir que el ángulo para evitar obstáculos es 0. Para ver con más claridad en la siguiente imagen se grafica el ángulo en función del peso y el threshold.

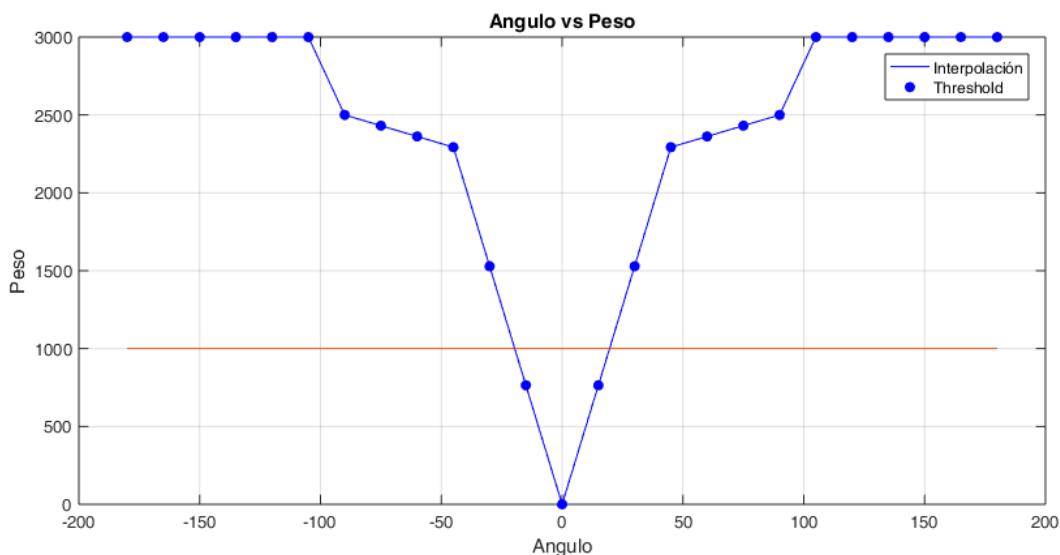


Figura 8-7: Visual weight vector

Como se puede ver en la Figura 8-7: Visual weight vector se puede discernir el grafico en dos zonas, sobre el threshold, llamado montañas, y por debajo, el valle.

Este grafico permite determinar el espacio disponible para que el vehículo siga y que ángulo debe tomar. Los ángulos que se encuentran por debajo del threshold en este caso son -15 y 15 grados, dando como resultado un ancho de 30 grados, con un promedio de 0 grados.

$$\frac{\lambda_{max} + \lambda_{min}}{2} = 0$$

Este es el ángulo que el vehículo debe tomar para que no colisione contra las paredes.

8.3 CONTROL DE OBJETIVOS + DETECCIÓN DE OBSTÁCULOS

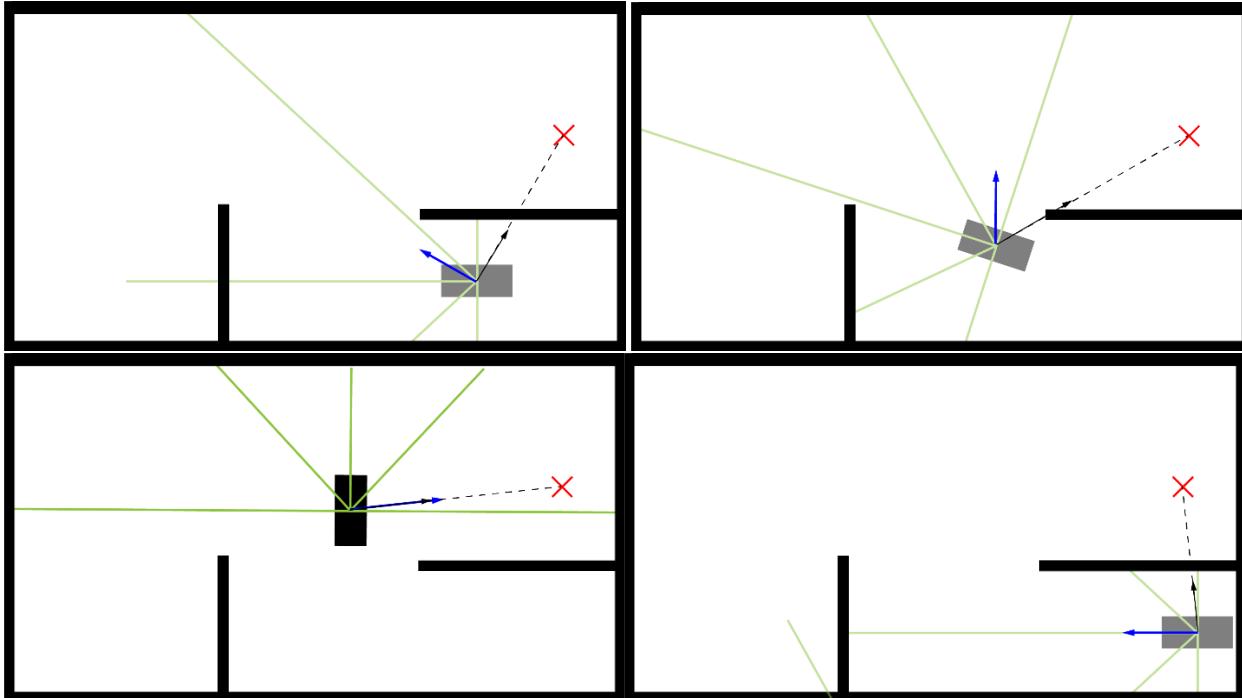
Ambos sistemas de control se pueden unir para lograr el objetivo mediante el análisis entre el ángulo α .

Si el ángulo α se encuentra dentro de la ventana generada por el valle el vehículo girará con el control de objetivos, si el ángulo esta fuera del valle y existe más de un valle el ángulo de giro del auto sera el promedio entre los ángulos extremos del valle más cercano al Angulo objetivo α .

Si solo existe un valle y el objetivo esta fuera de este, se ponderará el promedio dándole más peso al ángulo más cercano al objetivo.

Por ejemplo, en la situación de la Figura siguiente, el vehículo deberá doblar según el mismo criterio. En la siguiente serie de imágenes se pueden ver distintos casos.

La flecha en azul indica el giro calculado utilizando la combinación de los dos sistemas y la flecha negra indica el vector α .



En conclusión, el sistema deberá girar de acuerdo con las siguientes reglas:

$$\text{Si } \lambda_{min} < \alpha < \lambda_{max} \Rightarrow \gamma = \alpha$$

$$\text{Si } \alpha < \lambda_{min} < \lambda_{max} \Rightarrow \gamma = 0.7 * \lambda_{min} + 0.3 * \lambda_{max}$$

$$\text{Si } \lambda_{min} < \lambda_{max} < \alpha \Rightarrow \gamma = 0.3 * \lambda_{min} + 0.7 * \lambda_{max}$$

Siendo λ los ángulos debajo del threshold, α el ángulo del objetivo y γ el ángulo de giro efectivo.

8.4 ARQUITECTURA DE SOFTWARE

La arquitectura de software se definió de tal manera que funcione modularmente. Esto quiere decir que cada módulo tiene la capacidad de funcionar autónomamente y que cada uno puede recibir comandos y ser programado individualmente.

Estos comandos trascienden el tipo de comunicación usado, es decir, el mismo comando enviado por i2c, por serial o por RF puede ser interpretado de la misma manera. En el anexo se encuentra la lista de comandos que acepta cada módulo, a continuación, se explicara el funcionamiento básico de cada uno y como es el manejo de datos.

8.4.1 Módulo central

El módulo central está compuesto por 5 instancias que se encargan de controlar cada parte de sistema. Este se compone de 2 managers de comunicación, el de RF y el de comunicación por I2C.

El módulo central funciona de la siguiente manera:

- Setup
 - Configuración RF
 - Configuración I2C
 - Configuración de callbacks
- Loop
 - Request IMU Data
 - Cálculo de feedforward de posición y heading
 - Request I2C Sensor data del módulo radar
 - Envío por I2C de datos procesados al módulo de control
 - Request I2C de las órdenes de los módulos de control de velocidad y dirección
 - Seteo de velocidad y dirección del motor y el servo
 - Actualización del módulo de control de velocidad
 - Actualización de los módulos de comunicación
 - Respuesta ante callbacks

Este módulo es el principal y el irremplazable. Se encarga de las comunicaciones y su principal trabajo es enviar y recibir datos y comandos de acción respectivamente.

Cuando se enciende este módulo, comienza a enviar con el siguiente formato los datos del vehículo.

```

typedef struct I2C_command_controller_receive{
    unsigned char command;
    int16_t distance[5]; // mm
    double sensor_angle;
    double speed;
    double heading;
    double shaft_turns;
    double steering;
};

typedef struct I2C_command_controller_command_receive{
    unsigned char command;
    float data;
};

typedef union I2C_command_controller_packet_receive{
    I2C_command_controller_receive message;
    I2C_command_controller_command_receive com_message;
    uint8_t I2C_command_controller_bytes[sizeof(I2C_command_controller_receive)];
};

```

Y recibe los comandos con este formato:

```
typedef struct controller_data{
    float speed;
    float steering;
    int turret_angle;
};

typedef struct controller_data_bytes{
    uint8_t perbyte[sizeof(controller_data)];
};

typedef union I2C_controller_request{
    controller_data data_formatted;
    controller_data_bytes data_splitted;
};
```

Ademas mediante el módulo remoto se puede activar o desactivar mediante el comando correspondiente.

8.4.2 Módulo de radar

El módulo de radar utiliza los mismos controladores y funciona básicamente de la siguiente manera.

- Setup
 - Configuración I2C
 - Configuración de sensores
 - Configuración de feedback de comunicación
- Loop
 - Actualización del controlador de radar
 - Actualización de los módulos de comunicación
 - Lectura de sensores
 - Respuesta ante callbacks

El vehículo recibe la información del radar en el siguiente formato:

```
typedef struct sensor_data{
    int16_t distance[5]; // mm
    double time_stamp; // Relative to the center of the radar
    double sensor_angle;
};

typedef struct sensor_data_bytes{
    uint8_t perbyte[sizeof(sensor_data)];
};

typedef union I2C_sensor_request{
    sensor_data data_formatted;
    sensor_data_bytes data_splitted;
};
```

8.4.3 Módulo de control

El módulo de control posee solamente un parser de I2C y el sistema de control previamente explicado. Se enviarán los comandos de la forma mencionada previamente.

- Setup
 - Configuración I2C
 - Configuración de feedback de comunicación

- Loop
 - Actualización del controlador
 - Actualización de avoider
 - Respuesta ante callbacks

8.4.4 Módulo remoto

El módulo remoto simplemente se encarga de recibir la información para ser mostrada en una PC y poder ordenar al vehículo que active o desactive el control y un set de comandos explicados a continuación.

- Setup
 - Configuración SERIAL
 - Configuración I2C
 - Configuración de callbacks
 - Configuración de botón+joystick
- Loop
 - Lectura de joystick
 - Actualización de comunicación serial
 - Actualización del módulo RF
 - Request de mediciones de radar
 - Respuesta ante callbacks

Set de comandos:

```
// Set de comandos. Estos pueden ser utilizados en cualquier comunicación: I2C RF o Serial.
#define BATTERY A7
#define SET_MOTOR_SPEED 's'
#define GET_MOTOR_SPEED 'w'
#define TURN_STEERING 't'
#define GET_TURNS 'y'
#define BATTERY_STATUS 'b'
#define SET_TURRET_AUTO_MODE 'p'
#define SET_TURRET_MANUAL_MODE 'm'
#define GET_RADAR_MEASURES 'r'
#define RESET_CONTROLLER 'R'
#define SEND_CAR_NEW_X 'X' // para mandar los settings hay que armar el siguiente mensaje
q1234;1234;1234;1234;1234;1234;1234;1234;1234
#define SEND_CAR_NEW_Y 'Y'
#define SEND_CAR_NEW_THETA 'T'
#define SEND_CAR_X_FILTERED 'D'
#define SEND_CAR_Y_FILTERED 'E'
#define SEND_CAR_THETA_FILTERED 'F'
#define CONTROL_MODE 'P'
#define NEW_SET_POINT 'S'
#define CLEAR_GOALS 'C'
#define START_CONTROL 'G'
#define STOP_CONTROL 'K'
#define SEND_CAR_DATA 'L'
```

9 FABRICACIÓN

Gran parte de la plataforma se fabricó en PLA impreso en 3D con tecnología de deposición. Todas las partes fueron especialmente diseñadas para que puedan ser fabricadas con esta tecnología.

El proceso de diseño fue iterativo y sobre todo debido a las correcciones necesarias para lograr un mejor desempeño de la pieza impresa.

Uno de los puntos importantes en el momento de diseño de las partes destinadas a fabricarse en esta tecnología es comprender las características de la impresión 3D.

A continuación, se mencionarán algunas partes del proceso de fabricación y luego en el Anexo estará explicado el manual de reproducción.

9.1 IMPRESIÓN 3D

Durante los últimos años la tecnología de impresión 3D se popularizó a tal punto que es factible tener una impresora de escritorio en cualquier oficina.

Existen distintas calidades y distintos tipos de tecnología de impresión y no todas se proliferaron a tal punto.

Las más utilizadas en el mercado retail y la utilizada en este proyecto son las de deposición de polimérico termoplástico. Este se calienta hasta que fluya y se empuja a través de un extrusor de menos de 1mm de diámetro.

La impresora, imprime por capas, las cuales son cortes transversales de las piezas 3D y son unidas por el mismo calor provisto por el polimérico caliente.

9.1.1 Consideraciones de diseño

Es importante tener en cuenta que la calidad de la impresión 3D es inferior a un mecanizado, la tolerancia depende de cada máquina, pero está directamente ligado al diámetro del extrusor, la velocidad de impresión y la calidad de la máquina.

- Si se quiere hacer un agujero con un ajuste de holgura con un eje de 5mm hay que tener en cuenta que el diámetro tiene que ser entre 0.2 y 0.5 mm más grande.
- El ancho de las paredes no se recomienda que sean menores a 2 mm.
- Idealmente se tiene que diseñar para que las piezas tengan el menor vuelo posible y que los ángulos entre el plano de impresión y los planos de la pieza no sean menores a 45 grados.
- Para piezas que tienen que soportar solicitudes, se recomienda que la cantidad de capas sólidas en las paredes, techo y piso sea de al menos 3 y que el relleno no sea inferior al 30%.
- Si es necesario roscar un bulón o un tornillo en la pieza, el agujero debe tener un diámetro aproximadamente 0.2 mm menor que la fijación en cuestión. Igualmente debe tratar de evitarse este tipo de fijaciones.

Estos valores dependen de cada máquina y es indispensable realizar impresiones de calibración con el fin de obtener estos valores.

Por último, hay que considerar que el tiempo es un factor clave en el momento de impresión, las piezas grandes pueden alcanzar tiempos de 30 horas de impresión sin problemas, por lo tanto, es indispensable diseñar efectivamente con el objetivo de reducir el volumen al mínimo.

En conclusión, lo ideal es pensar que la pieza deberá ser fabricada en una matriz de inyección de plástico y aplicar todas las consideraciones para ese tipo de fabricación.

9.1.2 Configuración de las impresiones

La configuración de las piezas fabricadas en 3D del vehículo se vio modificada a lo largo del proyecto, en búsqueda de la mejor terminación y utilidad.

Luego de las impresiones realizadas se pudo concluir que para este tipo de aplicación la configuración de la impresión puede ser la siguiente:

- Material: PLA
- Ancho de capa: 0.4mm
- Temperatura de impresión: 205 °C
- Temperatura de cama: 70 °C
- Velocidad de impresión: 40m/s
- Soportes: NO
- Ancho de pared: 1.2 a 2 mm
- Infill: 30% para piezas genéricas y 50% para piezas de soporte.

9.2 ELECTRÓNICA

9.2.1 Módulo central

El módulo central está compuesto por un arduino nano, un módulo RF y un giroscopio MPU9250. El resto de los componentes están conectados al Arduino.

Como se buscó que la reproducción sea fácil, se evitó la fabricación tradicional de placas electrónicas y se utilizó una placa multiperforadora de 100 x 50 mm.

En la siguiente figura se muestra la conexión de los componentes y los conectores disponibles para el resto de los componentes.

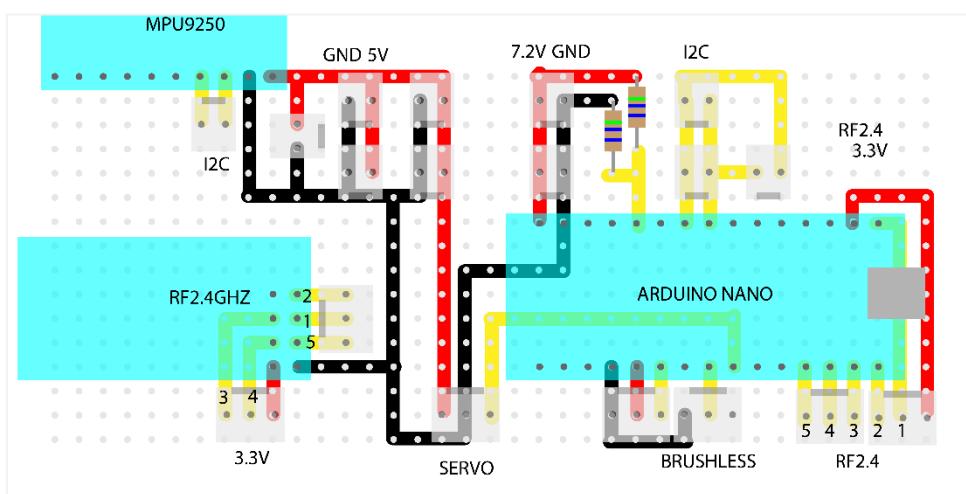


Figura 9-1: Modulo central

En negro están las pistas de masa, en rojo Vcc y en amarillo los pines de datos.

9.2.1.1 MPU9250

El módulo MPU-9250 en un solo paquete incluye un giroscopio a 3 ejes, un acelerómetro a 3 ejes y un magnetómetro a 3 ejes). Lo que le da 9 grados de libertad, cuenta con comunicación serial I2C y SPI.

En esta plataforma se utilizará solo el giroscopio y la comunicación I2C.

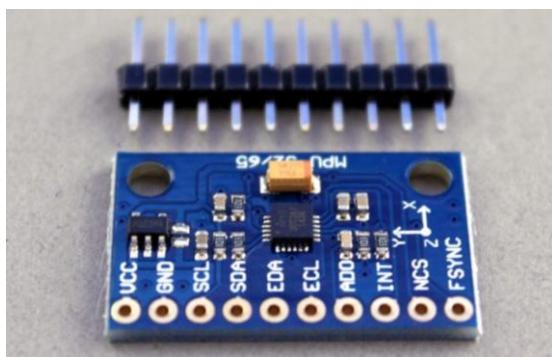


Figura 9-2: MPU9250

9.2.1.2 RF24

El NRF24L01 integra un transceptor RF (transmisor + receptor) con una frecuencia entre 2.4GHz y 2.5GHz. La velocidad de transmisión es configurable a 250 Kbps, 1Mbps, y 2 Mbps y permite la conexión simultánea hasta con 6 dispositivos.

El control del módulo se realiza a través de SPI, por lo que es sencillo controlarlo desde un procesador como Arduino.

La tensión de alimentación del NRF24L01 es de 1.9 a 3.6V, pero los pines de datos son tolerantes a 5V. Existen dos versiones de módulos, uno con antena integrada en forma de zigzag (el utilizado) y un alcance máximo de 20-30 metros, y el otro con antena externa con mayor alcance.



Figura 9-3: RF 24

9.2.1.3 Arduino nano

El Arduino nano está basado en un ATmega328P, tiene la capacidad de programarse vía USB con la interfaz gráfica desarrollada por Arduino.

Microcontroller	ATmega328
Architecture	AVR
Operating Voltage	5 V
Flash Memory	32 KB of which 2 KB used by bootloader
SRAM	2 KB
Clock Speed	16 MHz
Analog IN Pins	8
EEPROM	1 KB
DC Current per I/O Pins	40 mA (I/O Pins)
Input Voltage	7-12 V
Digital I/O Pins	22 (6 of which are PWM)
PWM Output	6
Power Consumption	19 mA
PCB Size	18 x 45 mm
Weight	7 g
Product Code	A000005

Figura 9-4: Características técnicas Arduino nano

9.2.1.4 Batería

La batería tiene que ser compatible con el módulo esc y el motor y poder proveer la corriente necesaria sin dañarse, por eso se eligió esta batería de dos celdas lipo que permite el funcionamiento correcto de todos los módulos.

- Tipo: Reaction 7.4V 5000mAh 2S 30C LiPo Hardcase: EC3
- Marca: Reaction
- Características:

Battery Balance Connector:	JST-XH
Battery Type:	LiPo
Battery Voltage:	7.4V
Capacity:	5000mAh
Cell Configuration:	2S1P
Connector Type:	EC3
Hard Case:	Yes
Height:	0.98 in (25mm)
Length:	5.45 in (138.5mm)
Maximum Burst Discharge Current:	300A
Maximum Burst Discharge Rate:	60C
Maximum Charge Current:	15.0A
Maximum Charge Rate:	3C
Maximum Continuous Discharge Current:	150A
Maximum Continuous Discharge Rate:	30C
Watt Hours:	37.0Wh
Width:	1.81 in (46mm)
Wire Gauge:	12 AWG

Tabla 9-1: Característica de la batería

9.2.1.5 Esc+Motor

El conjunto de controlador ESC y motor Brushless son los componentes más caros de toda la plataforma. Para seleccionar el combo se tuvo en cuenta el precio, la disponibilidad en el mercado nacional y las prestaciones necesarias para el tamaño de la plataforma.

Por este motivo se eligió el combo EZRUN MAX 10 de Hobbywing el cual está compuesto por un esc de hasta 60 A y un Motor busheles de 4000 KV.

Características:

	MAX5	MAX6	MAX8	MAX10 SCT	MAX10
					
Scale Vehicles	1/5th	1/6th, 1/7th	1/8th	1/10th	1/10th
Cont./Peak Current	200A/1300A	160A/1050A	150A/950A	120A/830A	60A / 450A
Motor Mode	Sensorless	Sensorless	Sensorless	Sensorless	Sensorless
Input	9-24 Cells NiMH , 3-8S Lipo	9-24 Cells NiMH , 3-8S Lipo	9-18 Cells NiMH 3-6S Lipo	6-12 Cells NiMH 2-4S Lipo	6-9 Cell NiMH 2-3S LiPo
Motor Limit	8S:KV≤1000 (Motor Size:58110)	6S:KV≤1500 8S:KV≤1200 (Motor Size:5892)	4S:KV≤3000 6S:KV≤2400 (Motor Size:4274)	2S LiPo:KV≤6000; 3S LiPo:KV≤4000; 4S LiPo:KV≤3000 (3656 Series of Motors)	2S LiPo/ 6S NiMH: KV≤6000 3S LiPo/ 9S NiMH: KV≤4000 (3652 size motor)
BEC Output	6V/7.2V adjustable, 6A	6V/7.2V adjustable, 6A	6V/7.2V adjustable, 6A	6V/7.4V Switchable, Continuous Current of 4A (Switch-mode)	6V/7.4V Switchable, Continuous Current of 3A (Switch- mode)
Programming Port	FAN/PRG Port	FAN/PRG Port	FAN/PRG Port	FAN/PRG Port	FAN/PRG Port
ESC Programming	Via the SET button/LED Program card /LCD Program Box/ WiFi Module	Same	Same	Same	Same
Switch Mode	Electronic Switch	Same	Same	Same	Same
Capacitor protection	Yes	Yes	Yes	Yes	Yes
Waterproof	Yes	Yes	Yes	Yes	Yes
Size (L.W.H) mm	93.35/58.12/47.81	70.0/56.0/46.5	59.8/48/36.8	49.0/39.5/34.7	39.4/32.8/23
Weight (g)	342g	240g	173.5g	105g	67.8g

Tabla 9-2:Datos técnicos ESC MAX10

EZRUN 3652 SL Brushless G2 motor			
Ezrun Series	3652 SL G2	3652 SL G2	3652 SL G2
KV	3300	4000	5400
P/n	30402600	30402601	30402602
LiPo	2-3S	2-3S	2S
R. (Ω)	0.0083	0.0060	0.0053
No-load Current	4.3A	5.2A	7.2A
Timing (deg.)	0	0	0
Motor Diameter & Length	$\Phi=36 / L=52$	$\Phi=36 / L=52$	$\Phi=36 / L=52$
Shaft Diameter & Length	$\Phi=5 / L=18.5$	$\Phi=5 / L=18.5$	$\Phi=5 / L=18.5$
Pole	4	4	4
Weight (g)	218	216	217

Tabla 9-3:Datos técnicos Motor 4000 KV

9.2.1.6 Fuente step down

La batería en este sistema es una fuente no regulada la cual necesita ser estabilizada a una tensión apropiada para poder alimentar a los distintos módulos. Para realizar esto era posible diseñar una placa custom con un regulador lineal, pero es necesario reducir el consumo y tener la mayor corriente disponible para poder alimentar los módulos, por eso se optó por la alternativa de adquirir un módulo step down tipo switching de 3A comercial y ajustarlo a 5V.



Figura 9-5:Fuente step down

Es una fuente basada en el regulador step-down DC-DC LM2596. Posee un preset multivuelta de alta precisión y es capaz de alimentar una carga de hasta 3A con una alta eficiencia.

Características:

- Tensión de entrada: 3 a 40V (DC)
- Salida de tensión: 1.25V a 35V (DC) ajustable
- Corriente Máxima de salida: 3A (por arriba de 2A se requiere disipador de calor adicional)
- Eficiencia de conversión: hasta el 92% (tensión de salida más alta, mayor será la eficiencia)
- Frecuencia de conmutación: 150 kHz
- Rectificador: no síncrono
- Propiedades del módulo: no aislado módulo reductor (buck)
Protección del cortocircuito: limitación de corriente por temperatura
- Temperatura de funcionamiento: grado industrial (-40 a 85) (potencia de salida de 10W)
- Temperatura con carga: 40
- Regulación de la carga: $\pm 0.5\%$
- Regulación de voltaje: $\pm 2.5\%$
- Velocidad de respuesta dinámica: 5% 200us
- Dimensión: 43 * 20 * 14 mm

9.2.1.7 Encoder

Para controlar la velocidad del vehículo es necesario medir la velocidad del eje del motor, para realizar esto se utilizó un sensor infrarrojo de barrera el cual sensa el paso de una pieza adosada al eje. El módulo utilizado es uno comercial el cual amplifica la señal del sensor de barrera y mediante un operacional provee una señal digital.

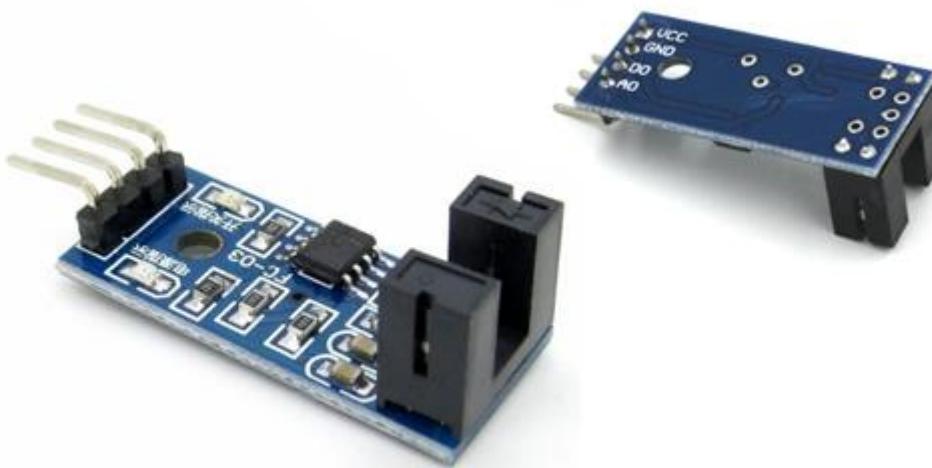


Figura 9-6: Encoder Óptico

9.2.1.8 Servo de dirección

El servo utilizado para la dirección fue seleccionado de acuerdo con el análisis del mecanismo y del torque necesario para que el auto pueda girar.

El servo que mejor se adapta a las necesidades y que se encuentra disponible en el mercado a un precio razonable es el “MG996R High Torque Metal Gear Dual Ball Bearing Servo” con las siguientes características:

- Weight: 55 g
- Dimension: 40.7 x 19.7 x 42.9 mm approx.
- Stall torque: 9.4 kgf·cm (4.8 V), 11 kgf·cm (6 V)
- Operating speed: 0.17 s/60° (4.8 V), 0.14 s/60° (6 V)
- Operating voltage: 4.8 V a 7.2 V
- Running Current 500 mA — 900 mA (6V)
- Stall Current 2.5 A (6V)
- Dead band width: 5 µs
- Stable and shock proof double ball bearing design
- Temperature range: 0 °C – 55 °C

9.2.2 Módulo de radar

El radar este compuesto por un servo idéntico al de dirección, MG996R, 5 sensores infrarrojos SHARP y un Arduino nano.

Como se buscó que la reproducción sea fácil, se evitó la fabricación tradicional de placas electrónicas y se utilizó una placa multiporadora de 50 x 50 mm.

En la siguiente figura se muestra la conexión de los componentes y los conectores disponibles para el resto de los componentes.

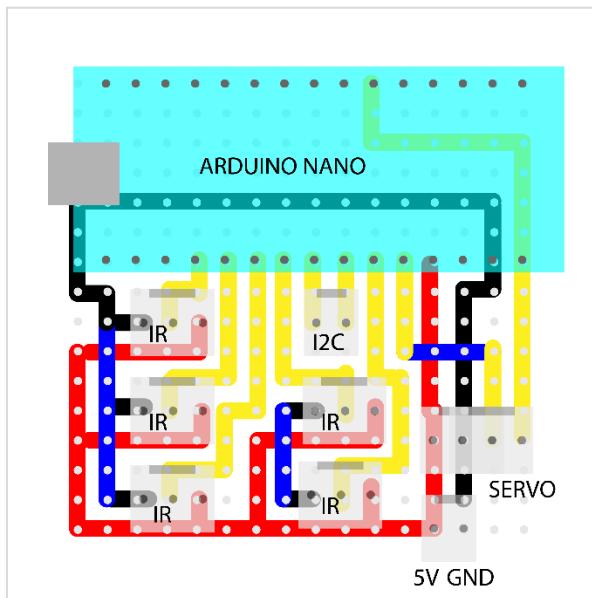


Figura 9-7: Modulo de radar

9.2.2.1 Sensor Sharp GP2Y0A02YK0F

Un sensor SHARP es un sensor óptico capaz de medir distancia, para esto el sensor mide la distancia usando triangulación.

Consiste en medir uno de los ángulos que forma el triángulo emisor-objeto-receptor. El Receptor es un PSD (Position Sensitive Detector) que detecta el punto de incidencia el cual depende del ángulo y a su vez de la distancia del objeto.

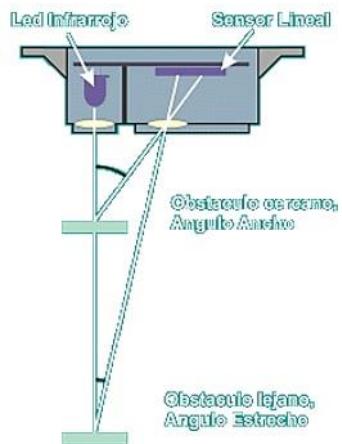


Figura 9-8: Método de triangulación SHARP

La luz que emite es puntual, lo que permite usar el sensor para escanear o mapear áreas, pero teniendo en cuenta que objetos pequeños serán difíciles de detectar. No son sensibles a la luz ambiental o el Sol, SHARP usa una luz infrarroja intermitente con una frecuencia determinada, que en el receptor es filtrada y elimina cualquier otra fuente de luz diferente a la frecuencia emitida.

Para utilizarlo basta con alimentarlo con 5V y leer la tensión en el tercer cable.

9.2.3 Módulo de control

El módulo de control es solamente un Arduino nano conectado al módulo central mediante I2C, montado sobre una placa multiporforadora de 50x50mm.

10 PRUEBAS REALIZADAS

Si bien el objetivo del proyecto es la plataforma en sí, los resultados presentados son las respuestas de control ante ensayos de la plataforma en entornos controlados.

Para realizar el relevo de datos se desarrolló una librería de interfaz para Matlab, la cual se encuentra adjunta en el Anexo.

Se mencionará el resultado en base al ensayo realizado más representativo. La plataforma se ensayó en un entorno controlado con la geometría y las dimensiones mostradas en la figura siguiente.

10.1 CONTROL

El vehículo inicia en la posición [0,0] y tiene como objetivo alcanzar el punto [0,-3] en metros. El vehículo fue colocado orientado como el eje x del plano.

Luego se midió la posición final del vehículo y se comparó con el objetivo propuesto la posición y con la posición final estimada con dead Reckoning.

El error entre los tres es de aproximadamente 50 cm, lo cual en principio no es aceptable. Pero considerando que el dead reckoning acumula error por cada medición y tiende a divergir el error es aceptable.

Como se puede ver en la imagen siguiente el recorrido del vehículo penetra las paredes, esto en la realidad no sucede y esquiva efectivamente cada obstáculo. Lo que se refleja es el error de la técnica de Dead Reckoning para obtener la posición.

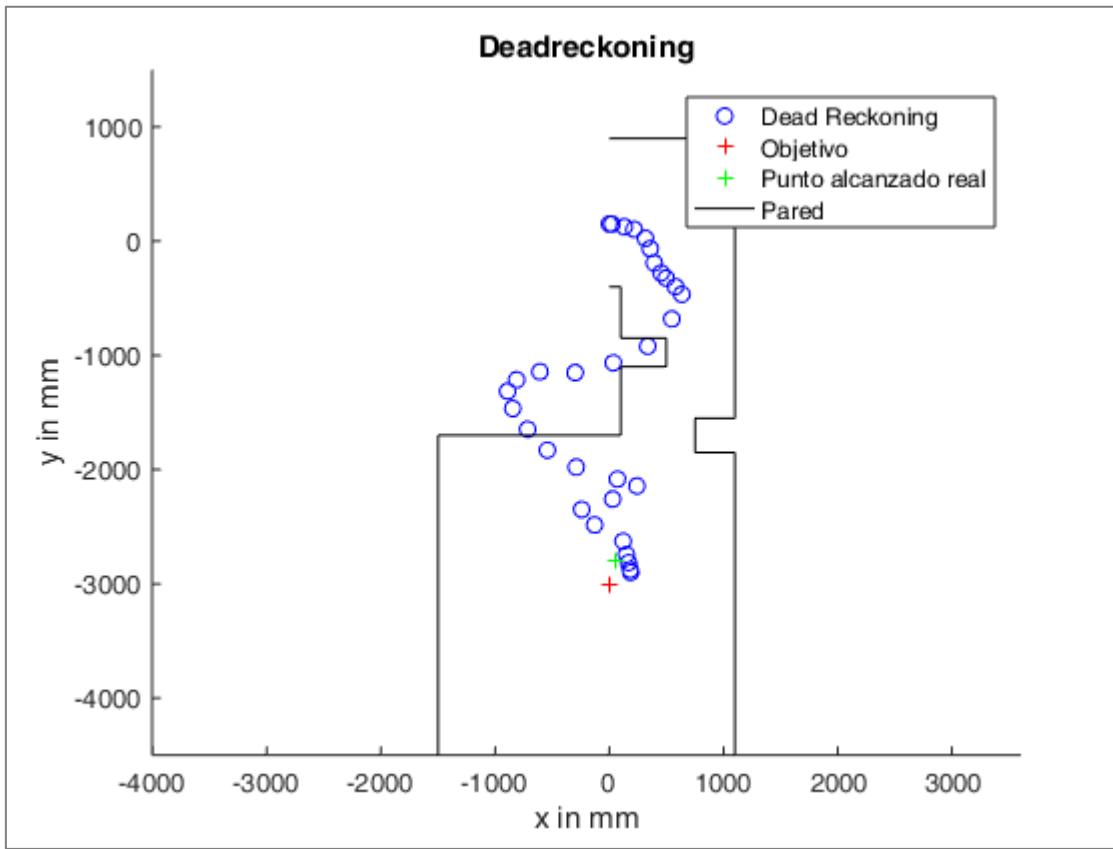


Figura 10-1: Ensayo del sistema

Este vehículo con el control que tiene implementado es capaz de moverse por un entorno evitando obstáculos aun si la localización no es correcta. Este problema de localización genera un error en el punto alcanzado debido a la acumulación de errores y es evitable si se disponen de herramientas de localización absoluta como el GPS o similar y *sensor fusión*.

10.2 COMUNICACIÓN

Por otro lado, otro de los puntos relevantes del ensayo es el problema de la falla de comunicación. Los módulos utilizados tienen un alcance mucho menor al indicado en la hoja de datos y provoca fallos en el envío y recepción de paquetes de datos limitando la frecuencia de transferencia a menos de 10Hz.

Esto provoca que todo el sistema se comporte lento y se pierda calidad de control

10.3 TRACCIÓN

Como fue comentado el motor elegido no fue la mejor opción, debido a la falta de control que tiene el ESC (Driver) el torque no es continuo y el sistema de control de velocidad no puede contrarrestar esta característica. Ademas, el arranque las bajas velocidades generan problemas mecánicos a los componentes que componen el sistema de dirección, sobre todo al diferencial.

Para poder medir la velocidad y el desplazamiento utilizado en el dead Reckoning se diseñó una pieza que en conjunto con el encoder óptico se transforma en un medidor de velocidad. En la siguiente imagen se puede ver el montaje en el acople flexible del sistema de tracción.

Este encoder es detectado utilizando una interrupción interna, lo que limita fuertemente las capacidades de velocidad del Arduino

10.4 RADAR

Se testeó el radar haciendo girar la plataforma con el objetivo de verificar que el vehículo doblaría en la dirección que menos obstáculos tiene. Como la velocidad del Servo que lo hace girar es baja en relación con la dinámica del sistema, una rotación de 90 grados puede significar un desplazamiento del vehículo tan grande que hace inútiles las mediciones.

10.5 DIRECCIÓN

El sistema de dirección se comportó correctamente desde el punto de vista electrónica, pero, los bujes plástico-utilizados en el mecanismo no son lo suficiente para anular los juegos en los acoplos amplificando el juego en las ruedas.

Esto genera un problema de conducción debido a que la alineación puede generar comportamientos indeseados. Por otro lado, el aumentar el rango de movimiento haciendo asimétrico el control de dirección no genera ninguna ventaja.

11 RESULTADOS

La determinación de las especificaciones de la plataforma fueron parte del proyecto. Esto hizo que el proceso fuera iterativo a partir de los requerimientos iniciales que fueron una simple guía para lograr el objetivo.

Esta sección se centra en los resultados obtenidos a partir de los objetivos planteados:

- Facilidad de reproducción y fabricación
- Modularización
- Capacidad de control para alcanzar un objetivo y evitar obstáculos

11.1 COMANDOS

El control remoto de la plataforma es una herramienta clave en el proceso de testeo y es la única forma de enviarle los puntos objetivo que debe alcanzar. Por eso, además de la plataforma se realizó un sistema de control remoto y una librería para Matlab con la capacidad de comunicarse. En el Anexo se muestran los distintos componentes y como se utiliza.

Esta librería se basa en comando serial enviados a una base que se conecta remotamente al vehículo mediante un módulo transporte de 2.4 GHz. La librería de Matlab simplemente envía por serial esos comandos.

Resulta útil realizar los comandos en una consola externa a Matlab debido a los retrasos temporales que Matlab impone, pero Matlab dispone de las herramientas de análisis de comportamiento y sobre todo la facilidad de uso.

11.2 TRACCIÓN

La tracción fue uno de los puntos más complejos de resolver. El motor tiene características no favorables para rotaciones a bajas velocidades, por lo que creó un desafío el realizar una medición correcta de la velocidad y construir un control.

Fue necesario determinar una secuencia de arranque y utilizar un sistema de control independiente en cada una de las etapas de arranque debido a que la transferencia de PWM a velocidad no es lineal debido al controlador del motor.

Estas complicaciones se suman a que el ESC no reacciona correctamente ante cambios repentinos de comandos debido a que fue diseñado para operar en conjunto con un sistema de mando manual.

El control del ESC requiere una señal de PWM para la velocidad máxima de giro en un sentido, una neutra para velocidad de giro nula y una máxima para velocidad de giro máximo en el otro sentido.

El controlador está diseñado para que antes de hacer un cambio de sentido, se tenga que enviar un comando de frenado. El comando de frenado implica un descenso repentino del PWM hasta su extremo contrario y luego un ascenso repentino a el punto muerto. Luego debe realizarse un descenso suave para lograr una velocidad inversa.

Esto, sumado a la alinealidad de sistema genera inestabilidades en el sistema de control de velocidad siendo posible un comportamiento erróneo del Driver y un aumento indiscriminado de velocidad del motor. Por este motivo se limitó el giro del motor a solamente una dirección. Aun así, existen condiciones por software que buscan limitar la velocidad de giro del motor en caso de falla del Driver.

11.3 DIRECCIÓN

La dirección Ackermann resultó fue el desafío más grande de la plataforma debido a la gran cantidad de partes que la componen y la dificultad para determinar cada una de sus dimensiones. Previo al diseño se realizaron investigaciones para implementar un método de diseño del mecanismo. Este método resultó iterativo y no contempla las limitaciones geométricas del mecanismo. Por este motivo gran parte del tiempo dedicado al proyecto fue en el desarrollo del módulo de dirección.

Es importante desatacar que el diseño de este mecanismo es una conjunción entre las propiedades mecánicas del material (en este caso PLA), el objetivo de diseño (Radio de giro mínimo), y la facilidad de fabricación y ensamble. Este proceso resultó iterativo y se realizaron diferentes evaluaciones de las partes hasta alcanzar el diseño implementado.

Uno de los puntos críticos es el desgaste de la pieza en los nodos, el PLA no es un material con propiedades superficiales aptas para soportar altos ciclos, por este motivo se diseñó para alojar un buje plástico por interferencia que sea capaz de ser de interfaz entre el eje y la pieza.

La suma entre la gran cantidad de piezas y la baja precisión (aproximadamente de 0.2 mm) de las impresiones hace que se acumule el error y consecuentemente se produzcan pequeños desvíos en las ruedas. Eso genera que a nivel de control de la plataforma se dificulte establecer con precisión un ángulo de giro de las ruedas. Depende del ángulo en el que se encuentre el “juego” puede llegar a ser de varios grados (de 2 a 7 grados).

11.4 CHASIS

Este es el único de los elementos impresos que no dispone de partes móviles, pero cumple con el rol de soporte y unión de los módulos.

Este diseño no contempla las fijaciones de la placa electrónica central debido a que en el momento de diseño no se había determinado las dimensiones de esta. Por este motivo se buscó generar un espacio libre suficientemente grande para albergar placas de tamaño estándar.

Cabe destacar que esta pieza es la responsable de brindarle rigidez al vehículo por lo tanto si diseño estilo “cajón” fue especialmente estudiado. Según los estudios realizados por Makerbot (Empresa de máquinas impresoras 3D) la tensión de rotura del PLA depende de la orientación de las capas, el relleno, el alto de capa entre otras cosas, pero se puede estimar entre 18 MPa y 95 MPa. Teniendo esto en cuenta, se analizó una placa de 3 x 100 mm de espesor y de 100 mm de largo empotrada en un extremo y una carga puntual en el otro y se obtuvo como conclusión que es capaz de soportar con 18 MPa de tensión bajo esas condiciones una carga de 2.7 Kg. Esto implica que un diseño estilo cajón que amplía el momento de inercia considerablemente es más que suficiente para soportar las cargas de la plataforma.

11.5 RADAR

El radar fue diseñado para que tenga la capacidad de rotar y escanear el entorno llenando los espacios entre los ángulos de cada sensor, aumentando virtualmente la resolución de este.

Luego de ensayos prácticos se observó que la capacidad de giro por el contrario de lo pensado empeora la navegación. Esto se debe principalmente a la velocidad de giro del radar. Cuando el vehículo se está moviendo a una determinada velocidad, la medición de los sensores en ese ángulo determinado indica la distancia de un objeto al vehículo en ese instante, y utilizando el método explicado se interpolan las mediciones entre cada sensor.

En el momento que el sistema de control decide cambiar la dirección, el radar se mueve en conjunto con la dirección debido a que las velocidades máximas son similares.

Como el sistema de control se basa en valles generados a partir de mediciones puntuales es posible que en esos valles existan obstáculos que no fueron vistos o que existan espacios donde se estimó que había un obstáculo. En este caso estos objetos serán detectados mientras el radar mientras girando y el sistema de control generara una nueva decisión en el camino. Estas mediciones intermedias entre la posición original y la final generan un ruido al que el sistema de control translada a la dirección provocando giros innecesarios.

Para tener un sistema de control con un radar móvil es indispensable incrementar la velocidad de giro del radar lo suficiente para que en el transcurso de tiempo entre una muestra y la siguiente sea capaz de verificar la medición antes de enviar el comando de giro a la dirección.

Con el servo implementado en el radar esto no es posible realizarlo, seria nefario utilizar un servo motor de mayor velocidad angular o un motor paso a paso con el torque suficiente.

11.6 CONTROL

El sistema de control desarrollado para esta plataforma fue pensado con el objetivo de utilizar la menor capacidad de computo posible. Esto genera una limitación del comportamiento que es difícil de superar utilizando el mismo hardware. Por otro lado, el comportamiento es más que satisfactorio considerando estas limitaciones ya es posible seguir mejorándolo.

Uno de los puntos críticos del control es el sistema de estimación de posición de la plataforma. Actualmente el vehículo integra las variaciones de desplazamiento para devolver una posición relativa al origen. Esto genera un acumulo de errores que no es sostenible en el tiempo. Como se pudo ver en el ensayo, el error es muy grande y es necesario reducirlo para lograr alcanzar un objetivo.

El diseño de software de la plataforma permite que se pueda remover el sistema de control y reemplazarlo simplemente respetando el protocolo de comunicación establecido.

12 CONCLUSIONES GENERALES

El objetivo principal de este proyecto es el desarrollo de una plataforma autónoma que posea facilidad de reproducción, modularización y capacidad de control para alcanzar un objetivo y evitar obstáculos en forma autonómica.

Para alcanzar este objetivo se analizaron distintas alternativas de solución. Los criterios guía seguidos incluyeron bajo costo, disponibilidad de herramientas en el ITBA, facilidad de fabricación y disponibilidad de adquisición local de partes constitutivas.

La plataforma diseñada e implementada cumple con los objetivos planteados, pero la aplicación de estos criterios para su diseño mecánico y electrónico, aún a pesar de no utilizar la última tecnología en sensores y actuadores por razones de costo. Como contrapartida, la facilidad de fabricación potencia la eventual proliferación de la plataforma y su uso práctico en entornos académicos.

El objetivo de modularizarían se alcanzó en los tres ámbitos: Mecánico, electrónico y software.

Desde el punto de vista mecánico existe la capacidad de remover completamente una sección, sea Tracción, Chasis o Dirección y reemplazarla con otro diseño, simplemente respetando las fijaciones. Las tapas tendrán que ser rediseñadas en caso de cambio de módulos. Además, es posible fabricar los módulos en cualquier impresora 3D y las partes que no pueden imprimirse son sencillas de obtener buscando en locales especializados de automodelismo.

Con respecto a la electrónica, se diseñó para que cada módulo sea lo más independiente posible, pero se limitó a realizar una conjunción entre los 3 módulos mecánicos centrales.

La tracción, dirección y el chasis están unidos por la misma electrónica. Esto no es una limitación en el momento de reemplazar los módulos mecánicos por otros, dado que los sistemas de tracción y dirección son comandados por señales PWM. La electrónica central se comunica con estos dos módulos mediante un conector de tres polos, dos de alimentación y uno de PWM. Además, para el caso del módulo de tracción se dispone un tercer conector de tres polos, conectado a un pin del módulo Arduino, que tiene la capacidad de funcionar como interrupción externa con el objetivo de medir la velocidad de giro. En el caso de que no sea suficiente, también se disponen de pares de pines libres para comunicación I2C.

El poder de cómputo del hardware es limitado, pero gracias a la modularización basada en comunicación I2C, es posible aumentar la capacidad de control solamente reemplazando el módulo de control. Esta limitación de computo es compensada por la facilidad de programación y testeo de código. El diseño basado en una placa de desarrollo como Arduino, de fácil y generalizado conocimiento, permite que muchas personas puedan hacer desarrollos para esta plataforma.

Desde el punto de vista académico esto tiene un valor inmenso ya que es posible testear y realizar investigaciones de algoritmos de control y navegación a un bajo costo y una curva de aprendizaje no restrictiva.

El software de la plataforma es completamente modular. Se desarrollaron bibliotecas independientes para cada función. Se puso especial énfasis en el desarrollo de bibliotecas de comunicación entre los módulos e interfaces (en el anexo se encuentran explicadas cada una de ellas).

El código fue desarrollado para funcionar sobre Arduino. Esta plataforma es un ejemplo de desarrollo cooperativo, por la variedad de herramientas que se ofrecen para el desarrollo de software. El código desarrollado se encuentra en su mayoría en C++ y hace uso de clases y objetos.

13 POSIBLES MEJORAS

La plataforma desarrollada podría considerarse una primera versión, al no tener definido objetivos cuantificables es posible desarrollar sin limitaciones.

Aun así, basado en la experiencia de esta plataforma es posible identificar una gran cantidad de modificaciones para que cumpla de mejor manera los mismos objetivos.

13.1 MECÁNICA

Desde el punto de vista mecánico se pueden atacar tres puntos importantes que siguen formando parte del mismo objetivo.

- Modularizarían y ensamble
- Fabricación
- Calidad

Con respecto a la modularización y ensamble sería necesario rediseñar tanto el módulo de tracción como el de dirección para simplificar el ensamble.

1. El sistema de tracción tiene problemas de ensamble en el momento de acoplar el motor con el diferencial. Es necesario desarmar el diferencial para poder ensamblarlos.
2. Las masas de la tracción impresas en 3D son funcionales, pero sería ideal fabricarlo en un material con mejores propiedades mecánicas debido a las cargas que tiene que soportar.
3. El sistema de dirección tiene mucho juego debido a la gran cantidad de piezas que la componen y a la baja precisión de los rodamientos utilizados.
4. La masa de la dirección necesita un rediseño para evitar juego en dirección radial.
5. Para simplificar el sistema de control, rediseñar la primera parte del mecanismo de dirección para que sea simétrico.
6. El radar necesita un reemplazo de actuador, debe colocarse un servo más rápido o cambiar el sistema de detección

13.2 ELECTRÓNICA

El primer paso para mejorar la electrónica sería diseñar un PCB específico para el módulo central, utilizando el mismo microcontrolador que utiliza el Arduino, pero en una placa diseñada especialmente para el vehículo. Por otro lado:

- Agregar un sistema de medición de posición absoluta como el GPS o similar. Durante el desarrollo de este proyecto se comenzó en paralelo un desarrollo de sistema de ubicación indoor utilizando ultrasonidos. El anexo se encuentra la explicación del sistema y una primera versión del sistema.
- La comunicación RF del módulo central tiene muy corto alcance, es necesario reemplazarlo por un sistema de mayor alcance.
- El módulo de control debería ser remplazado por una placa de desarrollo con mayor potencia de cómputo y que tenga capacidad de comunicación vía wifi, como por ejemplo una Raspberry PI 3
- Idealmente el radar debería ser reemplazado por un LIDAR. Pero en caso de que sea prohibitivo por el presupuesto, se podrían reemplazar los sensores por otros de mejor alcance y agregar mayor cantidad para de cubrir más puntos de medición y evitar puntos ciegos.

- Reemplazar el controlador ESC del módulo de tracción por uno que sea capaz de controlar la potencia entregada y garantizar el mayor toque para cada situación.

13.3 SOFTWARE

El software del vehículo sera dependiente de las modificaciones de hardware que se realicen, actualmente el módulo central está pensado para funcionar independientemente del resto de los módulos y tiene la capacidad de actuar como Máster en una red I2C enviando al módulo de control los datos de los sensores y la posición del vehículo.

Si se dispone de la potencia de hardware necesaria sería interesante implementar sistemas de localización utilizando sensor fusión con filtros de Kalman modificados o mediante filtro de partículas basado en el método de Monte Carlo. Ademas, sería interesante realizar un sistema de control basado en procesamiento de imágenes y/o redes neuronales.

Estos algoritmos son los utilizados actualmente en vehículos reales de conducción autónoma y son totalmente aplicables a la plataforma desarrollada simplemente modificando el módulo de control.

14 BIBLIOGRAFÍA

DARGAY, Joyce, GATELY, Dermot y SOMMER, Martin. Vehicle Ownership and Income Growth, Worldwide: 1960-2030. *The Energy Journal*. Vol. 28, No. 4 (2007), pp. 143-170

ABDUL LATIP, Nor Badariyah and OMAR, Rosli. Feasible Path Generation Using Bezier Curves for Car-Like Vehicle. En: IOP Conference Series: Materials Science and Engineering, Volume 226, conference 1.

WEINSTEIN, Alejandro J. y MOORE, Kevin L. Pose estimation of Ackerman steering vehicles for outdoors autonomous navigation. En: 2010 IEEE International Conference on Industrial Technology. 14-17 de Marzo 2010: Vina del Mar, Chile.

MAKERBOT. *PLA and ABS Strength Data*. MAKERBOT. [consulta 14 Julio 2018]. Disponible en: <https://downloads.makerbot.com/legal/MakerBot_R__PLA_and_ABS_Strength_Data.pdf>

KONG, Jason Kong, PFEIFFER Mark, SCHILDBACH Georg, BORRELLI Francesco. Kinematic and Dynamic Vehicle Models for Autonomous Driving Control Design. En: 2015 IEEE Intelligent Vehicles Symposium (IV). 28 de junio y 1 de July 2015: Seoul, South Korea.

ANKUR Naik. *Arc Path Collision Avoidance Algorithm for Autonomous Ground Vehicles*. Tesis de Master. Virginia Tech. 15 de Diciembre 2005.

ELMENREICH, Wilfried, SCHNEIDER Lukas, KIRNER Raimund. A Robust Certainty Grid Algorithm for Robotic Vision. En: Proc. 6th IEEE International Conference on Intelligent Engineering Systems (INES'02), Opatija, Croatia, Mayo 2002.

KIM, Do-Eun, HWANG Kyung-Hun, LEE Dong-Hun, KUC Tae-Young. A Simple Ultrasonic GPS System for Indoor Mobile Robot System using Kalman Filtering. En: 2006 SICE-ICASE International Joint Conference. 18-21 octubre 2006. Busan, South Korea.

ALCÁCER Vítor, ÁVILA Francisco, MARAT-MENDES Rosa. Design, Manufacturing and analysis of an ABS differential gear system model. En: 6th International Conference on Mechanics and Materials in Design. Julio 2015. Ponta Delgada, Açores.

CORKE, Peter. Vision and Control Fundamental Algorithms in MATLAB. Berlin. Springer Tracts in Advanced Robotics 73. 2011.

OGATA, Katsuhiko. Modern Control Engineering. Pearson; 5 edición. Septiembre 4, 2009.

15 ANEXOS

15.1 REPRODUCCIÓN DE LA PLATAFORMA

Para reproducir la plataforma no es necesario de planos de cada piza ya que las impresoras 3D trabajan directamente con archivos en formato 3D, pero si se generaron planos de ensamble y de medidas básicas.

Para mayor facilidad y acceso al manual de reproducción se generó un repositorio con el manual y los archivos. Todos los planos presentados en el anexo estarán disponibles en su formato original en el repositorio.

Cada plano tendrá el listado de componentes, la cantidad y la codificación de cada uno para su impresión o compra.

El objetivo del manual es lograr armar la plataforma tal que se vea como la figura siguiente.



Figura 15-1: Imagen de Plataforma ensamblada

El proceso de fabricación se dividirá en dos: Mecánica y Electrónica.

15.2 MECÁNICA

La mecánica consta de 5 partes principales.

- Dirección
- Tracción
- Chasis
- Radar
- Tapas

Todas las piezas que tienen que imprimirse tienen el siguiente código:

XXX-XXX-I3D-L04-W20-IF50

Este código tiene indicado que la pieza es imprimible y que se recomienda una capa de 0.4 mm, pared de 2 mm y relleno de 50 %.

Cada una ensamblada como se muestra en la Figura 15-2.

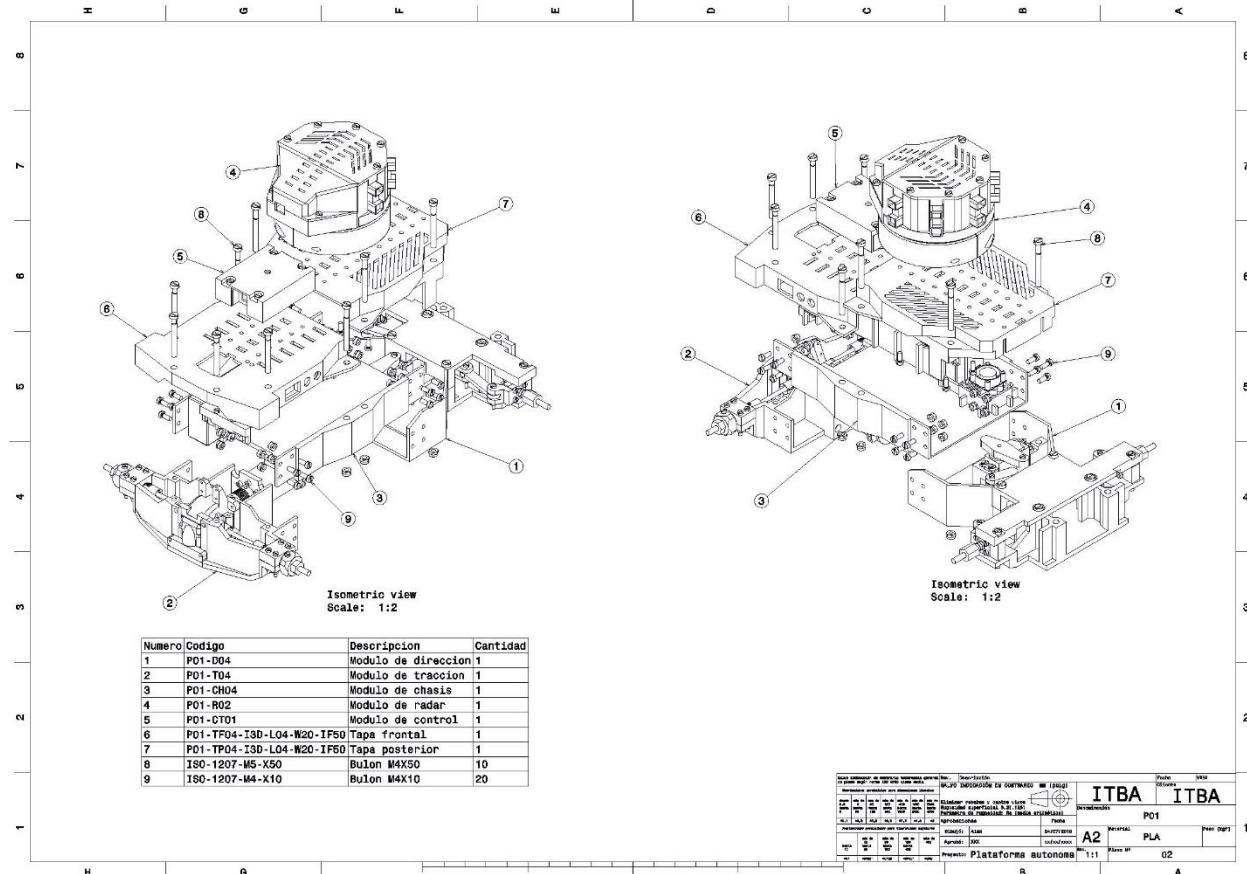


Figura 15-2: Ensamble general

Para lograr este ensamble general en primer lugar es necesario adquirir los componentes listados en cada plano y ensamblar cada módulo.

15.2.1 Dirección

El módulo de dirección es el más complejo de ensamblar debido a la cantidad de piezas que lo componen. Hay que tener especial cuidado en que las fijaciones entre las barras utilizando los bulones con las tuercas auto bloqueantes no sean ajustadas hasta apretar los componentes. Se diseño con el objetivo de hacer posible el giro de cada bulón en el buje plástico.

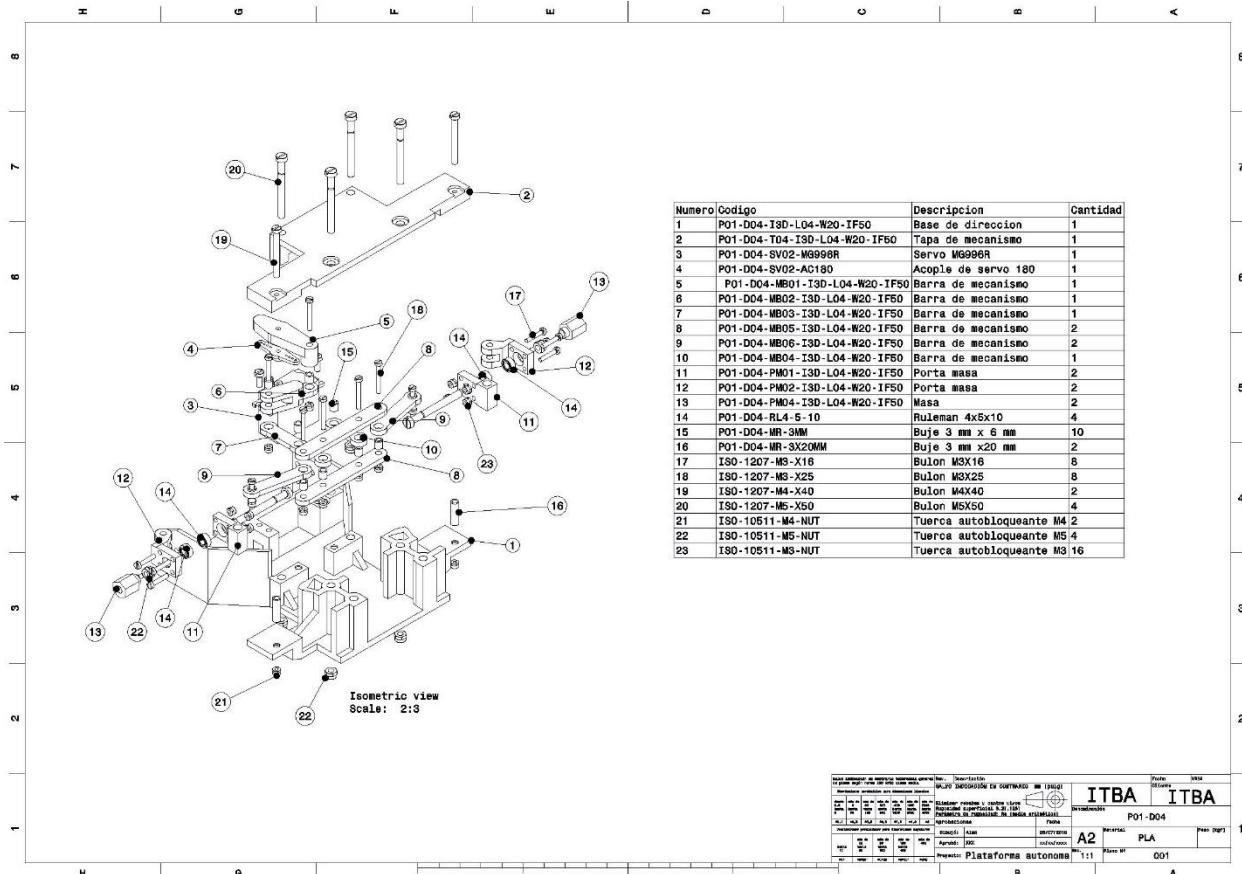


Figura 15-3: Modulo de dirección

Una vez ensamblado el módulo de dirección se verá como el de la Figura 15-2.

15.2.2 Tracción

El módulo de tracción tiene menor cantidad de piezas, pero es complejo el ensamblaje debido al poco espacio que existe entre el diferencial y el motor.

Para lograr un ensamblaje satisfactorio es necesario colocar primero el motor, luego desarmar el diferencial y ensamblar en conjunto el acople y el eje de entrada del diferencial. Además, para sensar la velocidad del eje se diseñó y se colocó una rueda dentada en el acople como se ve en la imagen.

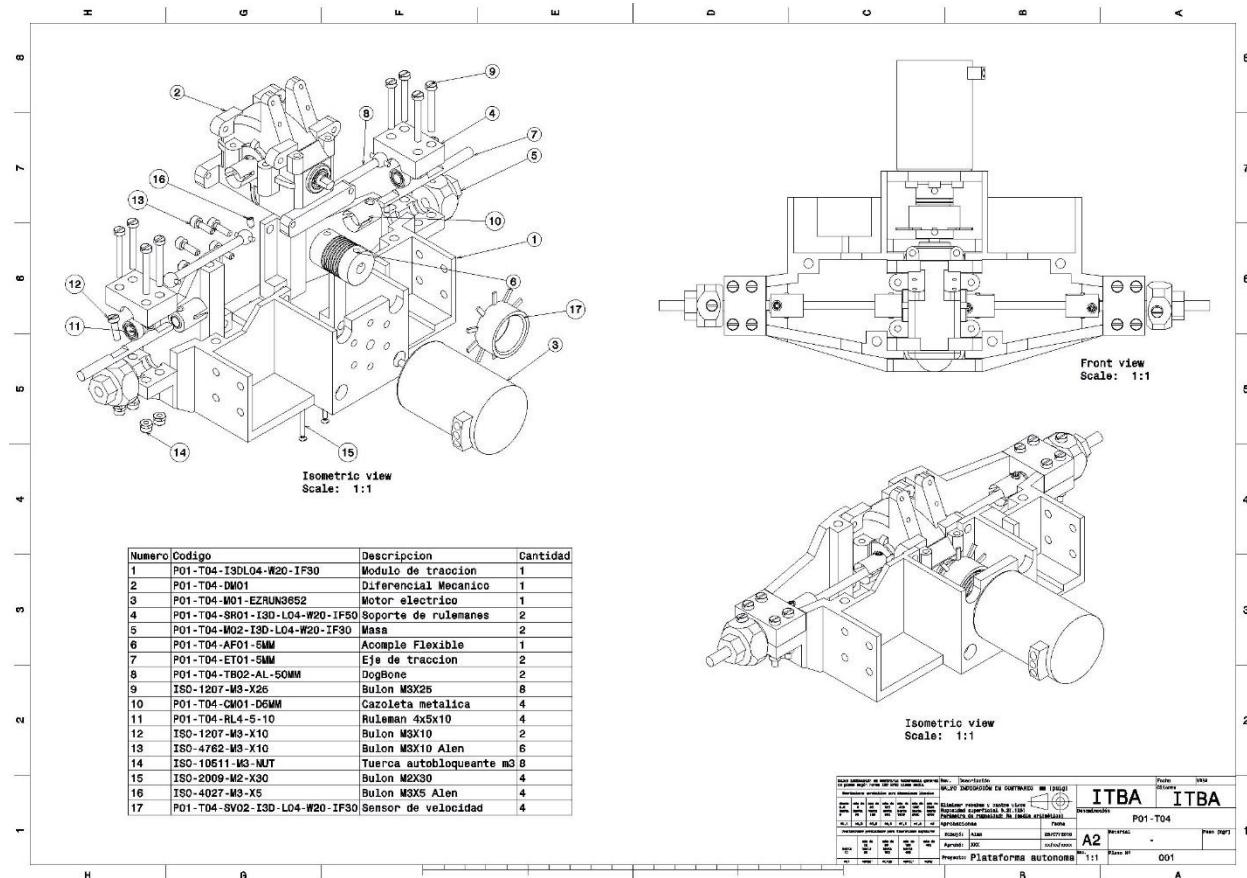


Figura 15-4: Modulo de tracción

15.2.3 Módulo de chasis

El chasis es el módulo más simple del ensamblaje, está diseñado para soportar la batería y el ESC. No están fijos los componentes, sino que están apoyados.

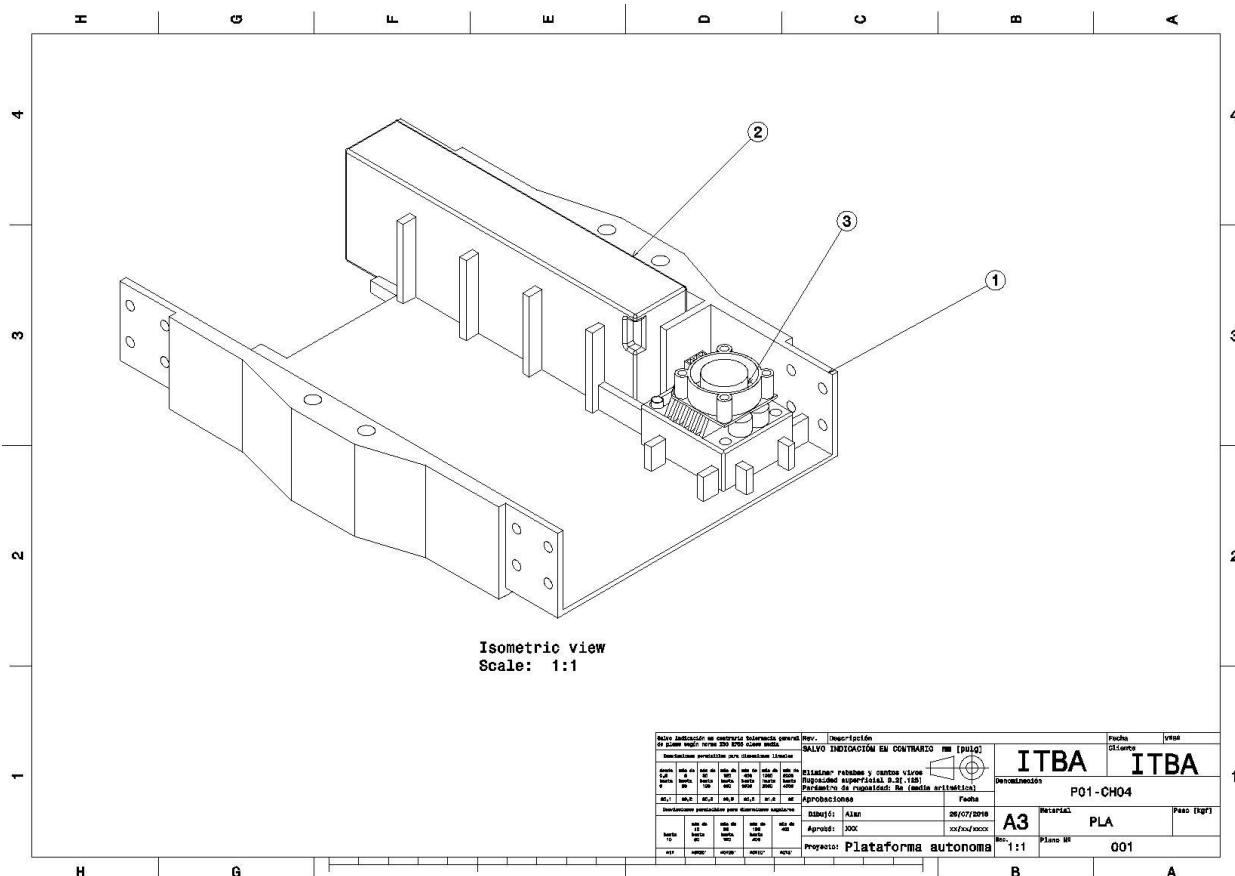


Figura 15-5: Modulo de chasis

15.2.4 Módulo de radar

El módulo de radar tiene un alojamiento de 50x50 mm para colocar la placa electrónica de este.

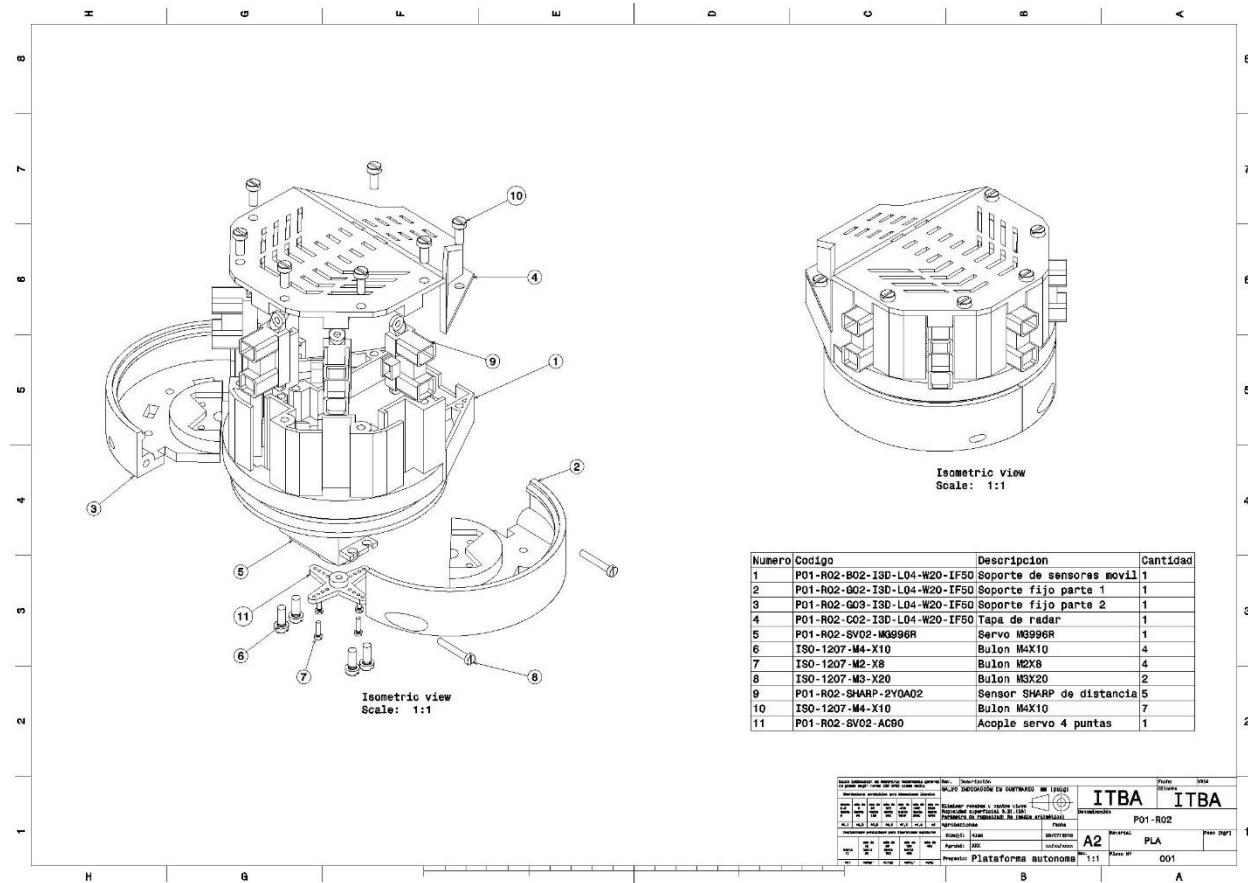


Figura 15-6: Modulo de radar

15.2.5 Módulo de control

El módulo de control consta de dos partes impresas que alojan un PCB.

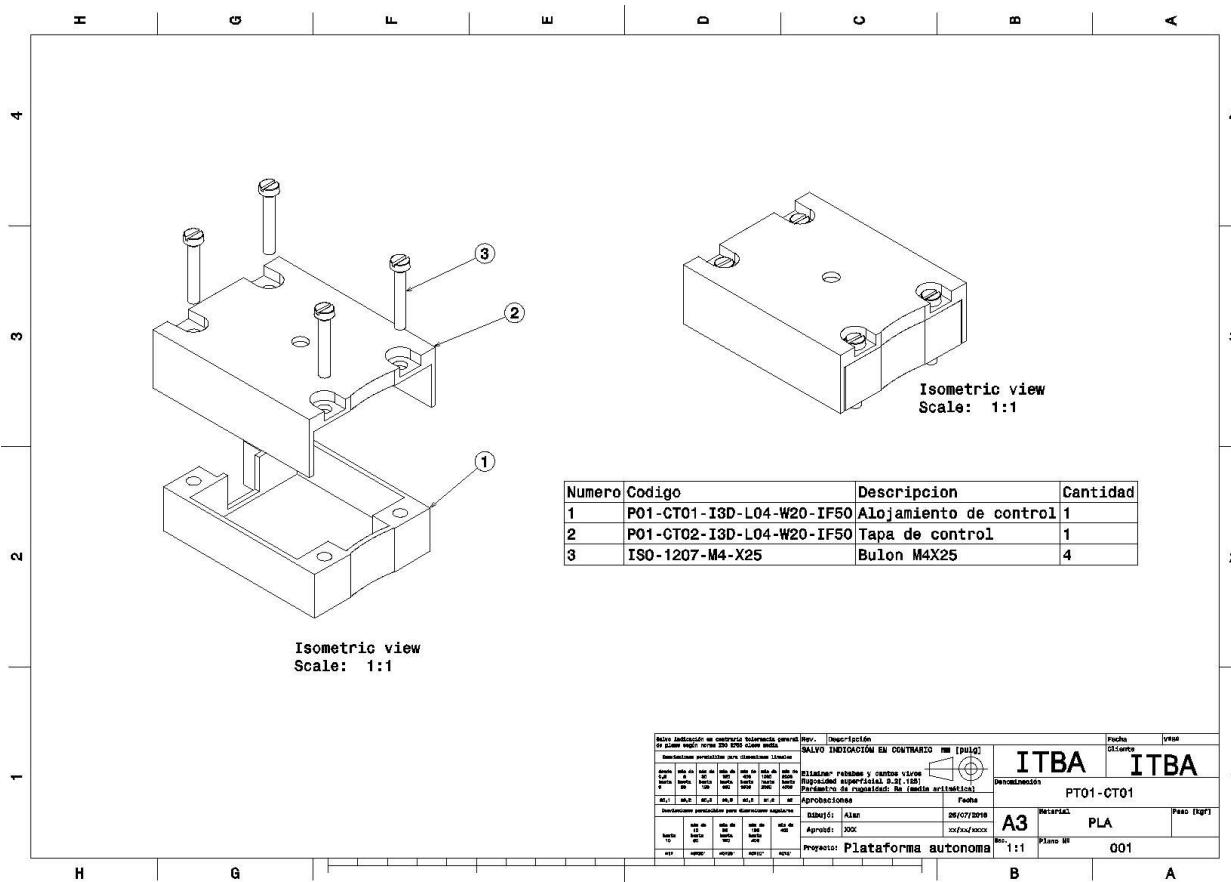


Figura 15-7: Modulo de control

15.2.6 Modulo remoto

Este módulo fue mencionado durante el informe, pero no fue explicado. Es el módulo a través del cual se le transfieren los comandos al módulo central y también recibe información sobre el estado del vehículo.

Es indispensable este módulo para controlar la plataforma, por lo tanto, se diseño una pieza de contención acorde para contener la electrónica que consta de un Arduino, un módulo transceptor y un joystick para controlar al auto sin conectar el módulo a una computadora.

Ademas se agregó un alojamiento para una batería externa tal que puede ser portable.

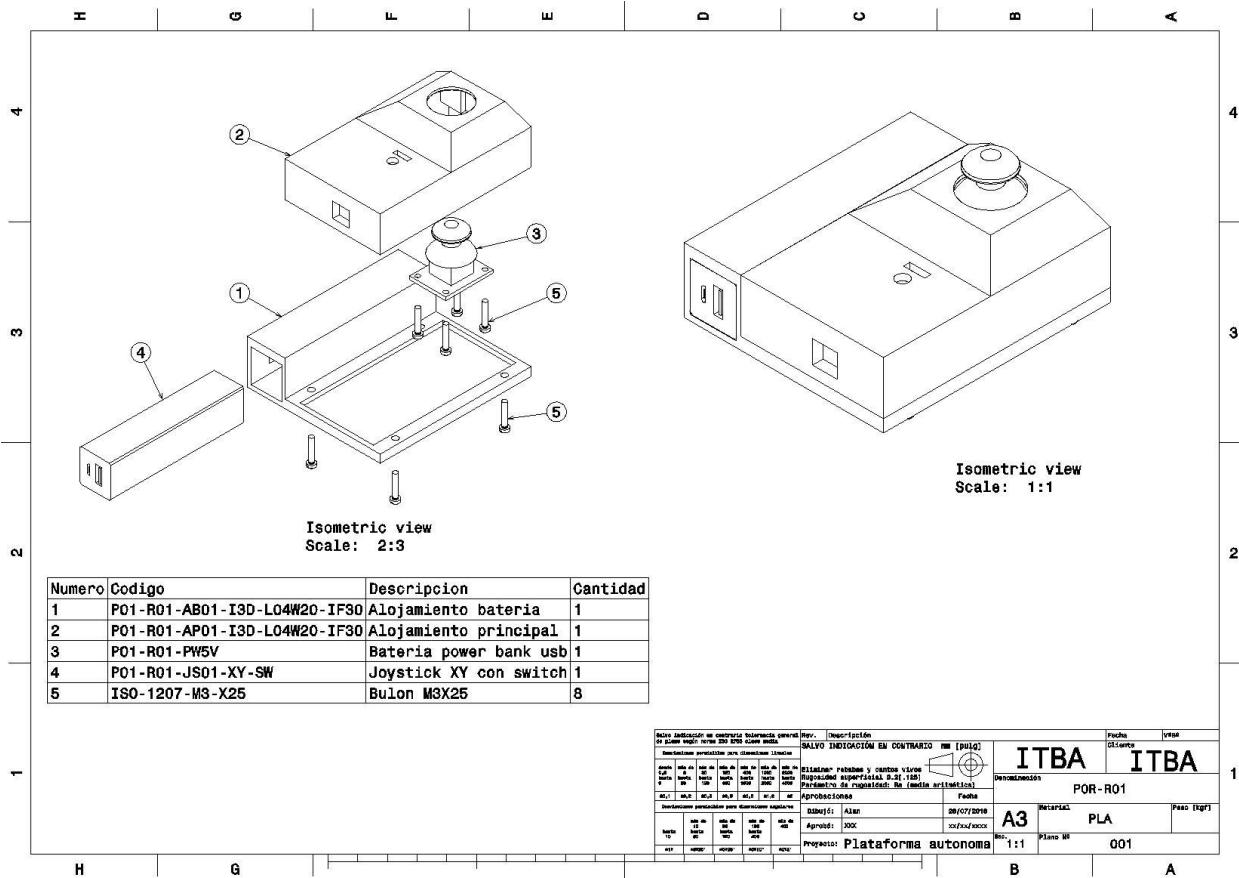
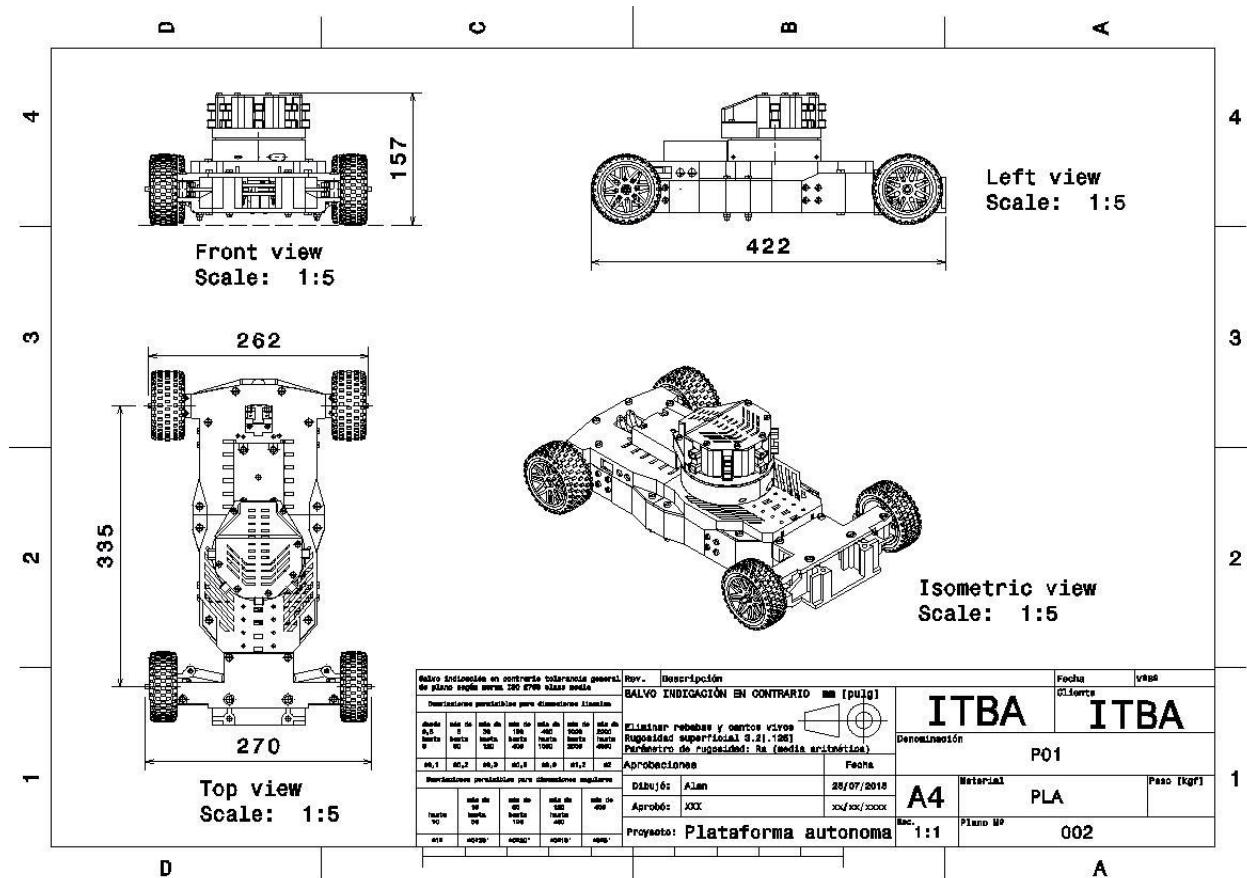


Figura 15-8: Modulo remoto.

15.2.7 Ensamble completo

La única pieza no rotulada son las ruedas, las cuales pueden ser cualquier rueda de modelismo para autos estala 1/10 con acople hexagonal de 12 mm.



15.3 ELECTRÓNICA

La electrónica esta dividida en 4 partes:

- Modulo central
- Módulo de radar
- Módulo de control
- Control remoto

Todas fueron fabricadas con placas multiporadas y conectores polarizados. A continuación, se muestra el circuito y las conexiones en las placas.

15.3.1.1 Modulo central

El módulo central tiene los siguientes componentes

Componentes	Descripción	Cantidad
Arduino nano 328P	Placa de desarrollo	1
NRF24L01	Modulo comunicación RF	1
MPU9250	Acelerómetro 9 ejes	1
Molex mini macho 3 vías	Conector polarizado 3 vías macho	7
Molex mini hembra 3 vías	Conector polarizado 3 vías hembra	7
Molex mini macho 2 vías	Conector polarizado 2 vías macho	11
Molex mini macho 2 vías	Conector polarizado 2 vías hembra	11
--	Resistencia 1k 1/8 W	2
--	Cable	3m
Fichas bananas	Fichas banana roja y negra	2
LM2596	Fuente step down	1
Switch	Interruptor de 2 estados	1
MG996R	Servo MG996R	1
MOCH22A	Modulo encoder óptico de barrera	1
EZRUN MAX10	Combo MAX10 – Brushless + ESC (driver)	1

Tabla 15-1: Componentes del módulo central

Las conexiones se pueden ver en la Figura 15-10 y en la Figura 15-1160 Se ven como está dispuesto en la placa multiporadora.

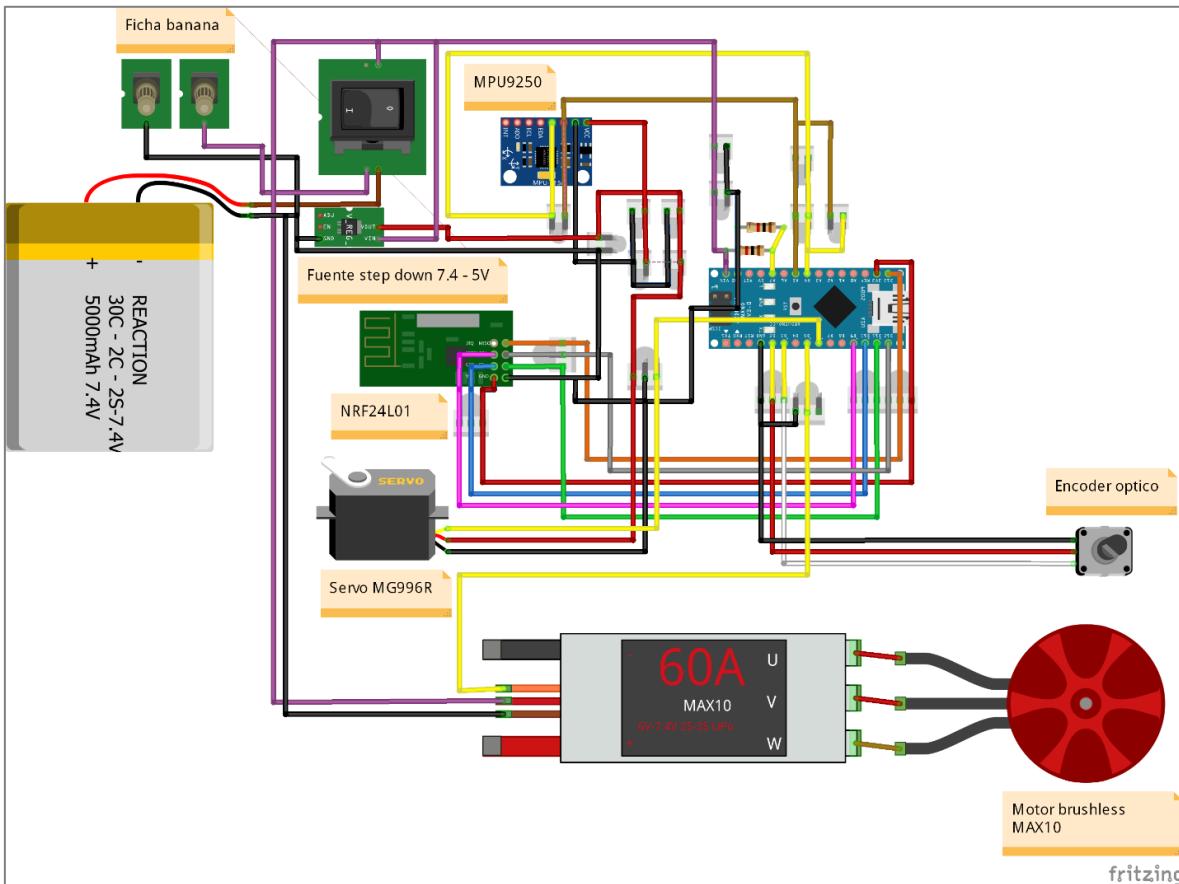


Figura 15-10: Conexiones

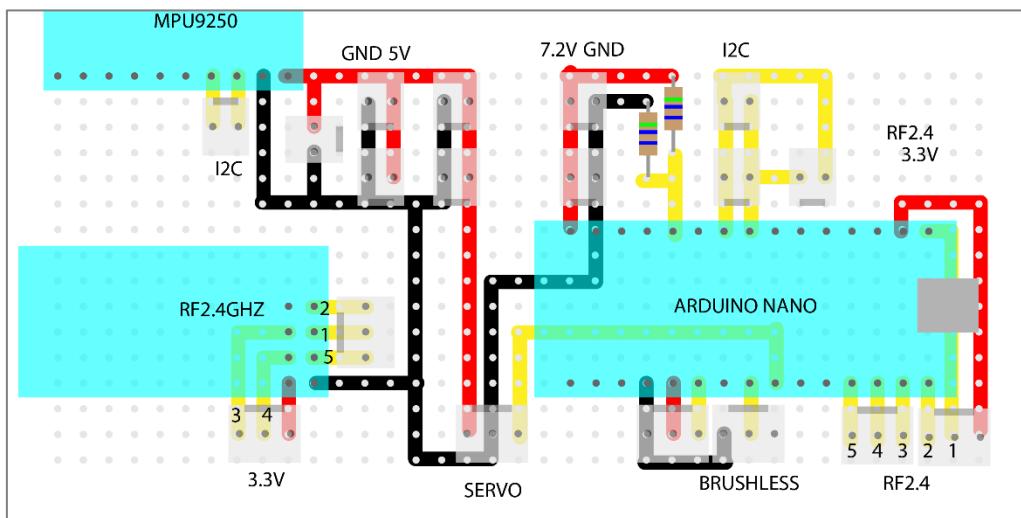


Figura 15-11: Disposición de componentes en la placa. Las líneas azules indican que la conexión es mediante un cable sobre el PCB.

15.3.1.2 Modulo radar

La electrónica del módulo de radar se basa en un Arduino montado en una placa de 50x50 mm y se conecta a la placa central a los conectores moles y la alimentación de 5V.

Componentes	Descripción	Cantidad
Arduino nano 328P	Placa de desarrollo	1
Molex mini macho 2 vías	Conector polarizado 2 vías macho	3
Molex mini macho 2 vías	Conector polarizado 2 vías hembra	3
Molex mini macho 3 vías	Conector polarizado 3 vías macho	5
Molex mini macho 3 vías	Conector polarizado 3 vías macho	5
--	Cable	60 cm
MG996R	Servo MG996R	1
Sharp 2y0a21	Sensor SHARP 1.5m	5

Tabla 15-2: Componentes del módulo de radar

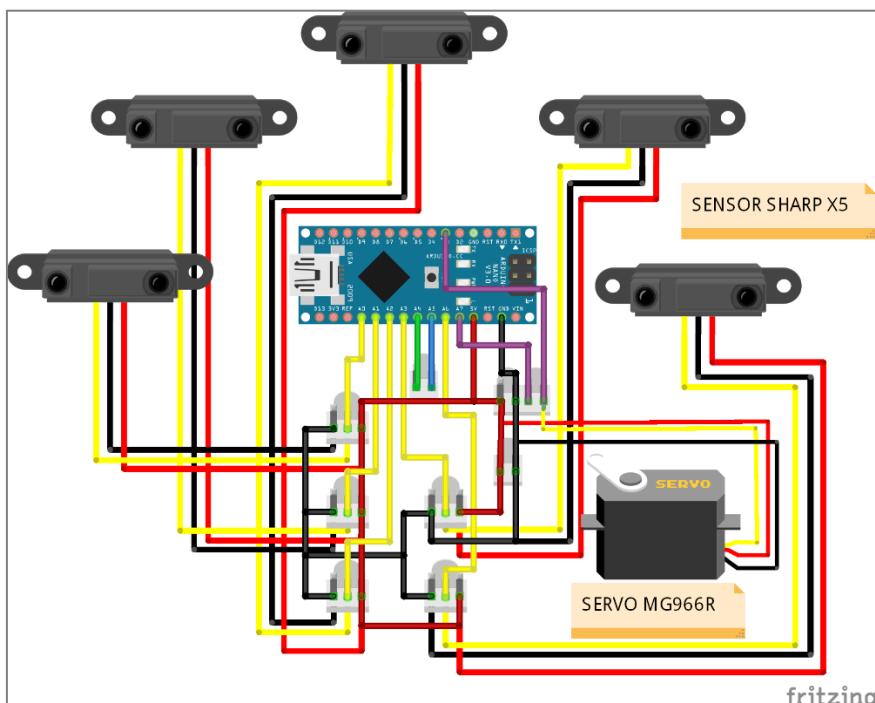


Figura 15-12: Conexiones del módulo de radar

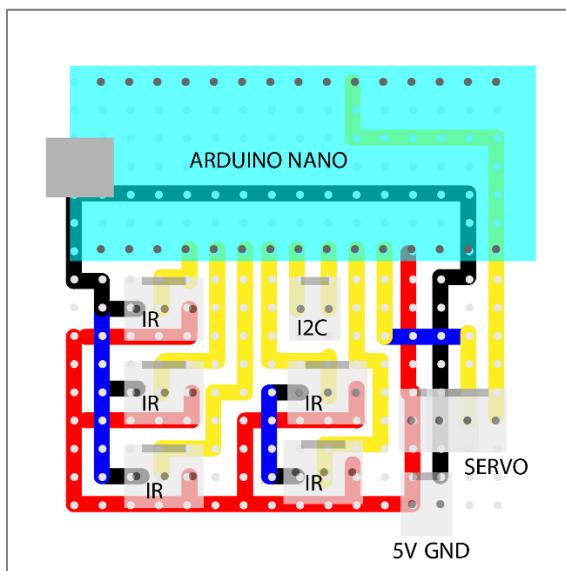


Figura 15-13: Disposición de componentes en la placa

15.3.1.3 Módulo de control

El módulo de control está compuesto solamente por un Arduino colocado en una placa e 50x50mm. Es importante la ubicación del Arduino en el centro de la placa, para que se puede acceder desde el exterior cuando la placa este colocada en el módulo de control impreso.

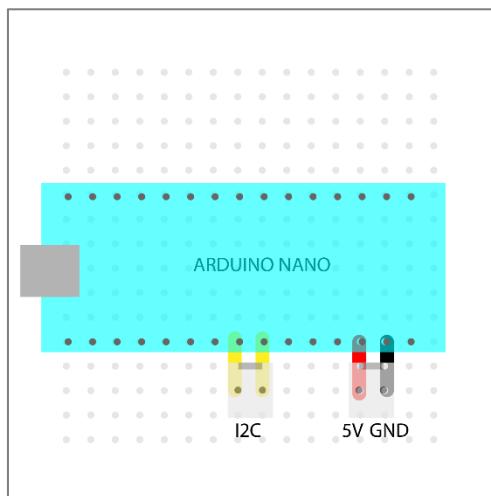


Figura 15-14: Modulo de control

15.3.1.4 Modulo remoto

Este módulo fue mencionado durante el informe, pero no fue explicado. Es el módulo a través del cual se le transfieren los comandos al módulo central y también recibe información sobre el estado del vehículo.

Es indispensable este módulo para controlar la plataforma, por lo tanto, se diseño una pieza de contención acorde para contener la electrónica que consta de un Arduino, un módulo transceptor y un joystick para controlar al auto sin conectar el módulo a una computadora.

Ademas se agregó un alojamiento para una batería externa tal que puede ser portable.

Componentes	Descripción	Cantidad
Arduino nano 328P	Placa de desarrollo	1
Molex mini macho 2 vías	Conector polarizado 2 vías macho	3
Molex mini macho 2 vías	Conector polarizado 2 vías hembra	3
--	Cable	60 cm
JOYSTICK	Modulo joystick	1
POWER BANK	Bateria power bank 22x22 mm USB 5V	1

Tabla 15-3: Componentes del módulo remoto

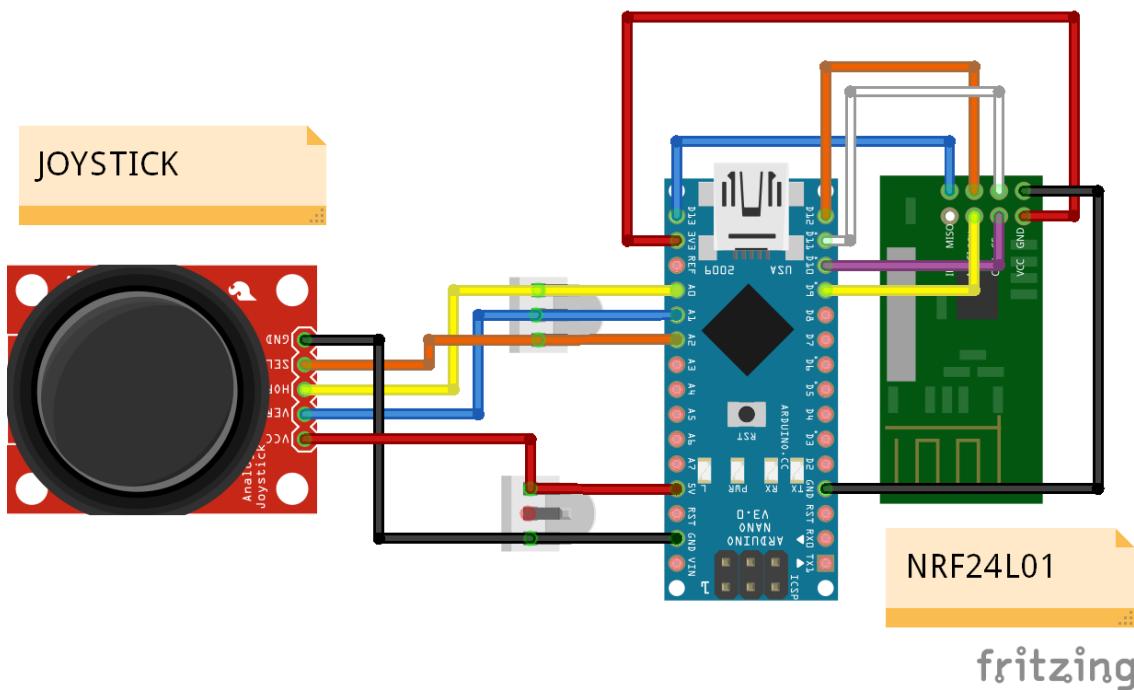


Figura 15-15: Conexiones del módulo remoto

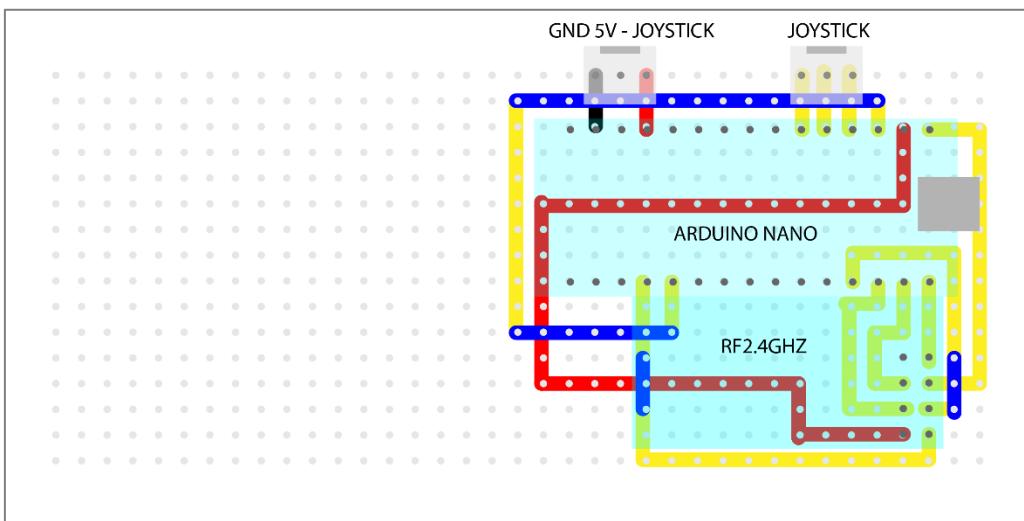


Figura 15-16:: PCB modulo remoto. Las líneas azules indican que la conexión es mediante un cable sobre el PCB.

15.4 SOFTWARE

Como se mencionó previamente el software esta modularizado. Cada módulo de hardware tiene su propia lógica y todos interconectados entre sí.

Especificamente se explicará el método de uso de la librería de comunicación que es la más compleja de comprender.

Ademas de los códigos de los módulos de hardware se desarrollaron funciones de Matlab para comandar al vehículo.

15.4.1 Comunicación

En esta plataforma hay 3 tipos de comunicación que se dan simultáneamente.

- RF mediante el módulo NRF24L01
- Serial mediante USB
- I2C entre los Arduinos

Las tres comunicaciones funcionan distintas y tienen capacidades diferentes, pero el mensaje que se tiene que transportar tiene que ser el mismo en los tres canales.

Para eso se desarrollaron 3 clases con un conjunto de métodos que liberan al usuario de ocuparse de discriminar entre tecnologías.

Para dejar claro el funcionamiento se explicada con el módulo de comunicación serial.

La clase está definida de la siguiente manera:

```
class serialManager{
    private:
        int baudrate;
    void (*trigger_functions[MAX_SUPPORTED_COMMANDS])(float);
    unsigned char trigger_commands[MAX_SUPPORTED_COMMANDS];
    unsigned char commands_configured=0;

    public:
        serialManager();
        bool add_callback (void (*triggerFunction)(float), unsigned char triggerCommand);
        void update();
        void sendCommand(unsigned char command, float data);
};
```

Básicamente la lógica de esta clase se centra en la ejecución de funciones cargadas en un vector de punteros a función según un comando preestablecido.

Por ejemplo:

Si se quiere que el Arduino realice una operación con un número y que devuelva el resultado por serial, bastaría con hacer el siguiente código.

```
// Se incluye la librería serialParser.h
#include "serialParser.h"

//Se genera un puntero de tipo serialManager
serialManager *manager_serial;

void setup() {
    Serial.begin(250000);
    // Se instancia un objeto de la clase serialManager y se lo asigna al puntero
    manager_serial = new serialManager();
    // Se agrega la función la cual tiene que ser llamada cuando llegue la letra 'X'.
    // Esta función tiene que tener la forma void nombre(float).
    // Es decir, recibe un dato float y no retorna nada.
    // Por otro lado, tiene que ser estática.
    manager_serial->add_callback(cuadrado,'X');
}

void loop(){
    // Se actualiza en cada ciclo la instancia del manager
    manager_serial->update();
}

// Función que eleva al cuadrado el valor
void cuadrado(float dato){
    // Imprime por serial el dato al cuadrado
    Serial.println(dato*dato);
}
```

Supongamos que ahora se quiere recibir por serial el dato a elevar al cuadrado, enviarlo por RF a otro Arduino, que este otro lo eleve al cuadro, retorne el dato al original y lo imprima.

La librería de RF funciona de manera similar, a continuación, se muestra el prototipo de la clase la a la cual se le agrego la capacidad de además de enviar float tiene la capacidad de enviar otro tipo de datos de tipo *car_data*

```

class rf_communication{
private:
    RF24 *radio_module;

    long pipes[7] = {0xF0F0F0F000,0xF0F0F0F001,0xF0F0F0F002,0xF0F0F0F003,0xF0F0F0F004,0xF0F0F0F005};

    RF_packet incoming;
    RF_packet outgoing;

    bool new_com = false;

    rf_mode module_mode;
    unsigned char myID;

    void (*trigger_functions[MAX_SUPPORTED_RF_COMMANDS])(float);
    void (*trigger_function_long)(car_data);
    unsigned char trigger_commands[MAX_SUPPORTED_RF_COMMANDS];
    unsigned char commands_configured=0;

public:
    rf_communication(unsigned char ID,rf_mode mode, unsigned char CE, unsigned char CSN); //para el master se usa ID 0
    bool add_callback (void (*triggerFunction)(float), unsigned char triggerCommand);
    bool add_car_data_callback (void (*triggerFunctionLong)(car_data));
    void rf_communication::start();
    void rf_communication::update(void);
    bool rf_communication::get_connection();
    void rf_communication::send_message(unsigned char command, float data);
    void rf_communication::send_car_data(sensor_data sensorData, float axisTurns, float motorSpeed, float anguloDireccion);
};


```

Volviendo al ejemplo.

1. El Arduino A recibe por serial un comando y un número.
2. El Arduino A envía el mismo comando y el mismo número por RF.
3. El Arduino B recibe el comando y el número
4. El Arduino B realiza la operación
5. El Arduino B envía el resultado por RF
6. El Arduino A recibe el resultado por RF
7. El Arduino A imprime el resultado

ARDUINO A

```
// Se incluye la librería serialParser.h
#include "serialParser.h"
#include "rf_parser.h"

//Se genera un puntero de tipo serialManager
serialManager *manager_serial;
//Se genera un puntero de tipo rf_communication
rf_communication *rf24;

void setup() {
    Serial.begin(250000);
    // Se instancia un objeto de la clase serialManager y se lo asigna al puntero
    manager_serial = new serialManager();
    // Se agrega la funcion la cual tiene que ser llamada cuando llegue la letra 'X'.
    // Esta funcion tiene que tener la forma void nombre(float).
    // Es decir, recibe un dato float y no retorna nada.
    // Por otro lado, tiene que ser estática.
    manager_serial->add_callback(cuadrado,'X');
    // Se instancia un objeto de la clase rf_communication y se lo asigna al puntero
    rf24=new rf_communication(0,MASTER,9,10);
    // Se agrega la funcion la cual tiene que ser llamada cuando llegue la letra 'R'.
    // Esta funcion tiene que tener la forma void nombre(float).
    rf24->add_callback(mostrar_resultado,'R');
}

void loop () {
    // Se actualiza en cada ciclo las dos instancias
    manager_serial->update();
    rf24->update();
}

// Funcion que eleva al cuadrado el valor
void cuadrado(float dato){
    rf24->send_message('X',dato);
}

// Funcion que muestra el resultado
void mostrar_resultado (float dato) {
    // Imprime por serial el dato al cuadrado
    Serial.println(dato*dato);
}
```

ARDUINO B

```
// Se incluye la librería rf_parser.h
#include "rf_parser.h"

//Se genera un puntero de tipo rf_communication
rf_communication *rf24;

void setup() {
    rf24=new rf_communication(0,MASTER,9,10);
    // Se agrega la función la cual tiene que ser llamada cuando llegue la letra 'X'.
    // Esta función tiene que tener la forma void nombre(float).
    rf24->add_callback(cuadrado,'X');
}

void loop(){
    // Se actualiza en cada ciclo las dos instancias
    rf24->update ();
}

// Función que eleva al cuadrado el valor
void cuadrado(float dato){
    rf24->send_message('R',dato*dato);
}
```

Como se puede ver la comunicación se simplificó de tal manera que no es necesario preocuparse por cómo es que funciona cada módulo. Esta librería es capaz de recibir tantos comandos como el usuario desee, para eso es necesario editar el valor de `MAX_SUPPORTED_COMMANDS` del header de cada clase.

Por último, el ejemplo más complejo y el que se da en la plataforma:

1. El Arduino A recibe por serial el comando de girar el radar y el ángulo.
2. El Arduino A envía el mismo comando y el mismo número por RF.
3. El Arduino B recibe el comando y el número.
4. El Arduino B envía el mismo comando y el mismo número por I2C al Arduino C.
5. El Arduino C mueve el servo.

Hay pequeñas diferencias entre la comunicación I2C y las anteriores.

1. Se tiene que definir un MÁSTER debido a que es posible que existan más de Arduino conectado a la red. Como el que inicia la comunicación sera Arduino B, este se elegirá máster, por otro lado, el Arduino C sera el Slave.
2. El envío de datos puede ser de dos maneras, mediante un request o un post.
 - a. El request es un mensaje que envía el máster al slave para que este último le devuelva un paquete con información.
 - b. El post lo envía el máster cuando quiere comunicar al slave sin requerir respuesta.
3. El protocolo I2C genera interrupciones en el micro haciendo innecesario la actualización del manager en el loop principal

ARDUINO A

```
// Se incluye la librería serialParser.h
#include "serialParser.h"
#include "rf_parser.h"

//Se genera un puntero de tipo serialManager
serialManager *manager_serial;
//Se genera un puntero de tipo rf_communication
rf_communication *rf24;

void setup() {
    Serial.begin(250000);
    // Se instancia un objeto de la clase serialManager y se lo asigna al puntero
    manager_serial = new serialManager();
    // Se agrega la función la cual tiene que ser llamada cuando llegue la letra 'X'.
    // Esta función tiene que tener la forma void nombre(float).
    // Es decir, recibe un dato float y no retorna nada.
    // Por otro lado, tiene que ser estática.
    manager_serial->add_callback(doblar,'D');
    // Se instancia un objeto de la clase rf_communication y se lo asigna al puntero
    rf24=new rf_communication(0,MASTER,9,10);

}

void loop () {
    // Se actualiza en cada ciclo las dos instancias
    manager_serial->update();
    rf24->update();
}

// Función que eleva al cuadrado el valor
void doblar(float angulo){
    rf24->send_message('D',angulo);
}
```

ARDUINO B

```
// Se incluye la librería rf_parser.h
#include "rf_parser.h"
#include "I2C_parser.h"

//Se genera un puntero de tipo rf_communication
rf_communication *rf24;
I2C_communication* radar_com; //(0,8,MASTER_I2C);

void setup () {
    rf24=new rf_communication (0, MASTER,9,10); // 9 y 10 corresponden a los pines de datos
        // Se agrega la función la cual tiene que ser llamada cuando llegue la letra 'X'.
        // Esta función tiene que tener la forma void nombre(float).
    rf24->add_callback(doblar,'D');
    // Se genera una nueva instancia y se inicia con el ID propio 0, el id del destino '2' y si
    // es master o slave.
    radar_com= new I2C_communication(0,RADAR_ID,MASTER_I2C);
}

void loop () {
    // Se actualiza en cada ciclo las dos instancias
    rf24->update ();
}

// Función que eleva al cuadrado el valor
void doblar(float angulo){
    radar_com->send_message(SET_TURRET_MANUAL_MODE,angulo);
}
```

ARDUINO C

```
#include "radar.h"
#include "Wire.h"
#include "CComands.h"

#define NSENSORS 5
#define ADDRESS 8
redgrey_radar *radar;

I2C_sensor_request sensor_measures;
I2C_communication *radar_com;

bool radar_mode=false; // false es manual
bool radar_new_mode=false;
float new_angle;

void setup() {
    Serial.begin(250000);
    Serial.println("Radar");
    radar = new redgrey_radar ();
    radar->configuration(60,90,60,MANUAL_RAD,300,-1);
    radar->move_to(0);
    Serial.println("New I2C manager");
    radar_com =new I2C_communication(0,RADAR_ID,SLAVE_I2C);
    radar_com->add_callback(manual_mode_setting,SET_TURRET_MANUAL_MODE);
    radar_com->add_callback(get_distances,GET_RADAR_MEASURES);
    radar_com->add_callback(automatic_mode_setting,SET_TURRET_AUTO_MODE);

    Wire.onRequest(send_sensor_data); // register event
    Wire.onReceive(get_commands);
    Serial.println("Ready i2c_com");

}

void loop() {
    // put your main code here, to run repeatedly:
    radar->update();
    get_measures();
    radar_com->update_I2C_sensordata(sensor_measures);

    if (radar_mode==false && radar_new_mode==true){
        automatic_mode(new_angle);
        radar_mode=true;
    }else if(radar_mode==true && radar_new_mode==false){
        radar_mode=false;
    }
}

void get_measures(){
    sensor_measures.data_formatted=radar->get_sensor_data();
}
void automatic_mode(float numero){
    Serial.println(numero);
    radar->configuration(60,90,(int)numero,AUTOMATIC_RAD,300,-1);
}
```

```
void manual_mode_setting(float angulo){  
    new_angle=angulo;  
    radar_new_mode=false;  
    manual(new_angle);  
}  
void manual (float numero){  
    radar->move_to(numero);  
    Serial.print("Moviendo torre a: ");  
    Serial.println(numero);  
}  
  
void send_sensor_data(){  
    radar_com->request_callback();  
}  
void get_commands(){  
    radar_com->read_callback();  
}  
  
void get_distances(){  
    Serial.println (radar->get_radar_angle());  
    Serial.println("\t");  
    for(int i=0;i<NSENSORS;i++){  
        Serial.print(sensor_measures.data_formatted.distance[i]);  
        Serial.print("\t");  
    }  
    Serial.println("\t");  
}
```

15.4.2 Uso de comandos – Matlab

Para poder comandar al vehículo se dispuso de una serie de comandos que permiten el manejo del vehículo. Todos los comandos comentados a continuación tienen que ser enviados al módulo remoto por serial con el siguiente formato:

LETRANUMERO

Por ejemplo, para setear una coordenada objetivo se tiene que enviar lo siguiente

X10

Y10

S1

El primer comando indica que el siguiente número sera la coordenada X en metros, el segundo sera la coordenada Y el tercero confirma esas dos coordenadas (Set).

A continuación, se muestra la lista de comandos.

Alias	Comando de un byte	Funcion
SET_MOTOR_SPEED	's'	Setea la velocidad del vehiculo y para el control en m/s
GET_MOTOR_SPEED	'w'	Devuelve la velocidad del vehiculo en m/s
TURN_STEERING	't'	Gira la direccion en grados
GET_TURNS	'y'	Devuelve la cantidad de vueltas del eje de las ruedas
BATTERY_STATUS	'b'	Devuelve el estado de la bateria
SET_TURRET_AUTO_MODE	'p'	Setea en modo automatico la torre girando el angulo pasado como parametro
SET_TURRET_MANUAL_MODE	'm'	Mueve el radar los grados pasados como parametro
GET_RADAR_MEASURES	'r'	Devuelve las medidas de los sensores en mm
RESET_CONTROLLER	'R'	Borra el estado del controlador
SEND_CAR_NEW_X	'X'	Agrega una coordenada X al proximo goal en metros
SEND_CAR_NEW_Y	'Y'	Agrega una coordenada Y al proximo goal en metros
NEW_SET_POINT	'S'	Agrega el goal a la lista de goals
CLEAR_GOALS	'C'	Borra los goals
START_CONTROL	'G'	Se activa el control
STOP_CONTROL	'K'	Se detiene el control

Tabla 15-4: Comandos de control

Para simplificar el uso se desarrollaron una serie de funciones en Matlab que permiten el control de la plataforma de manera más sencilla.

Funciones de interfaz de Matlab:

- *[s] = new_car (port)*: genera una instancia de comunicación del vehículo y retorna el objeto si la comunicación es exitosa.
- *[car_data, radar_data] = get_car_data(s)*: con la instancia de la placa creada, retorna la posición del vehículo (x, y, theta) en un vector de 3 elementos car_data y la medida obtenida con los 5 sensores en un el vector radar_data de 1x5.
- *set_goal (s, point)*: Envía el valor del nuevo goal en metros en un vector point de 1x2
- *start_car(s)*: Inicia el control del vehículo.
- *stop_car(s)*: Detiene el control, lo reinicia y borra los goals.
- *wait_connection (s)*: Espera a que el vehículo este listo para recibir comandos.

Ademas de estos comandos se provee del código Analyzer.m que utiliza estos comandos para enviar al vehiculo a un punto determinado mientras releva todos los datos del vehiculo, los almacena y la gráfica.