DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE GESTIÓN DE LA ENERGÍA ELÉCTRICA EN UN PROTOTIPO DE VEHÍCULO ELÉCTRICO

DANIEL GILBERTO CORTES BURBANO
LUIS ALBERTO NOGUERA ROSERO

UNIVERSIDAD DE NARIÑO
FACULTAD DE INGENIERÍA
PROGRAMA DE INGENIERÍA ELECTRÓNICA
SAN JUAN DE PASTO
2016

DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE GESTIÓN DE LA ENERGÍA ELÉCTRICA EN UN PROTOTIPO DE VEHÍCULO ELÉCTRICO

DANIEL GILBERTO CORTES BURBANO
LUIS ALBERTO NOGUERA ROSERO

Trabajo de grado para optar por el título de Ingeniero Electrónico

ASESOR
WILSON OLMEDO ACHICANOY MARTINEZ
INGENIERO ELECTRÓNICO

UNIVERSIDAD DE NARIÑO
FACULTAD DE INGENIERÍA
PROGRAMA DE INGENIERÍA ELECTRÓNICA
SAN JUAN DE PASTO
2016

NOTA DE RESPONSABILIDAD

Las ideas y conclusiones aportadas en este Trabajo de Grado son Responsabilidad de los autores.

Artículo 1 del Acuerdo No. 324 de octubre 11 de 1966, emanado por el Honorable Concejo Directivo de la Universidad de Nariño.

Nota de Aceptación

_____

_____

_____

_____

Jurado

_____

Jurado

San Juan de Pasto, 3 de febrero de 2016

DEDICATORIA

A Dios por permitirnos conocer más de Él, a nuestros familiares por el apoyo incondicional.

AGRADECIMIENTOS

A Dios por la capacidad y el talento regalado para cumplir esta meta y seguir adelante en este maravilloso mundo del saber.

A nuestro asesor Wilson Olmedo Achicanoy Martinez, por su dedicación y tiempo duranto todo el desarrollo del trabajo.

TABLA DE CONTENIDO

Pág.

LISTA DE TABLAS

Pág

LISTA DE FIGURAS

LISTA DE ANEXOS

Pág

RESUMEN

En este documento se plantea un método de control para gestionar la energía de un prototipo de vehiculo eléctrico, optimizando la energía consumida por los motores en un recorrido predeterminado del que se conocen sus características mas importantes. De igual forma se establece un método de medición, adecuación y procesamiento de señal que determinan el estado en tiempo real del prototipo a través de un recorrido, las variables medidas: son posición, orientación y estado de las baterías.

Se realiza un modelamiento matemático del prototipo, buscando disminuir la complejidad de este sin perder la fiabilidad en el comportamiento de sus variables de estado. Con el modelo matemático se plantea un método de control. El método usado para optimizar la energía es el principio de mínimo de Pontryagin. Con la implementación de este método podemos calcular las señales de alimentación óptima de los motores para recorrer el trayecto programado, cumpliendo las condiciones preestablesidas.

Con el fin de conocer los estados del prototipo se establecen métodos numéricos de estimación de datos como filtros Kalman. Para la medición de voltajes se usó un circuito de muestreo y retención, el cual envía el voltaje medido a un conversor análogo-digital en el microcontrolador. En la orientación, se utilizó un giroscopio fusionado con un magnetómetro. Para medir la posición se implementa una simulación en tiempo real, que utiliza el modelo del prototipo en espacio de estados y los voltajes entregados a los motores en cada instante de tiempo, junto con la medida del ángulo recorrido.

También se describe el hardware implementado para: medir las variables, gestionar la energía y mantener la comunicación inalámbrica entre el computador y el prototipo. El hardware principalmente se compone de un microcontrolador 18F2550 que gestiona la información, procesa los datos enviados desde la IMU y controla los motores a través del PWM disponible en su diseño de fábrica; una IMU GY85 que mide las variables requeridas para calcular el desplazamiento angular y modulos XBee para intercambiar información entre el computador y el prototipo de manera inalámbrica.

Por la versatilidad y programación sencilla en Matlab, se desarrolla una interfaz de usuario donde se controla la configuración del hardware. Además, se programa las características del recorrido y se muestran datos de las variables medidas como: las distancias recorridas en cada eje, el voltaje de la batería y el voltaje entregado a cada uno de los motores, el ángulo de orientación del prototipo, un gráfico de la posición del prototipo en el plano y otras opciones como: calibración dinámica y estática del vehículo, número de muestras para la calibración, velocidad de transmisión, gráficos de voltajes y corrientes en los motores y el botón para acceder a la ventana para cear la trayectoria deseada.

ABSTRACT

This document provides a control method arises to manage the energy of a prototype electric vehicle, optimizing the energy consumed by the motors in a predetermined its most important characteristics are known route. Likewise a measurement method, adaptation and signal processing that determine the real-time status of the prototype through a route is established, the measured variables: are position, orientation and battery status.

A mathematical modeling of the prototype is made, seeking to reduce the complexity of this without losing reliability in the behavior of their state variables. With the mathematical model a control method is proposed. The method used to optimize energy is the minimum principle of Pontryagin. With the implementation of this method we can calculate the optimal signal power engines scheduled to tour the journey, fulfilling the conditions preestablesidas.

In order to know the state of the prototype numerical data estimation methods such as Kalman filters are set. For measuring circuit voltages sample and hold, which sends the measured analog-digital converter in the microcontroller voltage it was used. a gyroscope fused with a magnetometer was used in the guidance. To measure the position of a simulation in real time, using the prototype model state space and voltages delivered to the motors at each instant of time, along with the travel angle measurement is implemented.

the hardware is also described implemented to: measure the variables, manage energy and maintain wireless communication between the computer and the prototype. The hardware consists mainly of a 18F2550 microcontroller that manages information, processes the data sent from the IMU and controls the motors through the PWM available in factory design; one GY85 IMU measuring variables required to calculate the angular displacement and XBee modules to exchange information between the computer and the prototype wirelessly.

For versatility and simple programming in Matlab, a user interface where the hardware configuration is controlled develops. Furthermore, the characteristics of the route program and data of the measured variables are shown as the distances covered in each axis, the battery voltage and the voltage delivered to each of the motors, the orientation angle of the prototype, a graph the position of the prototype in the plane and other options such as: dynamic and static calibration of the vehicle, number of samples for calibration, transmission speed, graphics voltages and currents in the motors and the button to access the window for cear the desired

INTRODUCCIÓN

En el contexto actual y teniendo en cuenta la creciente demanda energética del entorno que nos rodea, los nuevos avances científicos respecto a movilidad y transporte se han direccionado a disminuir el consumo energético y el impacto ambiental negativo de los vehículos destinados para tal fin. El desarrollo de nuevas tecnologías como los vehículos eléctricos es en sí mismo una alternativa para disminuir el impacto ambiental y energético efectuado por los vehículos de combustión interna [1]; sin embargo, dicha tecnología también representa un riesgo ambiental aún no dimensionado con la disposición de las baterías al cumplir su ciclo de vida útil; lo cual, hace de dicho sistema una buena plataforma para mejorar y/o innovar en el desarrollo de tecnologías que permitan el mayor aprovechamiento durante su vida útil.

La migración de las tecnologías de combustión interna a las alternativas eléctricas para los vehículos de desplazamiento personal o público, genera un gran impacto socio – ambiental donde las emisiones se reducen y la calidad del aire mejora. En el contexto económico regional, aún por ser una tecnología no implementada de forma masiva, resulta poco accesible para las personas de clase media y baja, lo cual ha levantado una gran barrera entre los usuarios y los vehículos eléctricos.

Teniendo en cuenta lo anterior, el uso eficiente de la energía eléctrica almacenada en los acumuladores o baterías representa un desafío para las fábricas de los mismos. Si bien, el desarrollo de nuevos y mejores acumuladores cada día mejora e incrementa también paralelamente se han desarrollado métodos de gestión energética para disminuir el consumo del vehículo y aumentar la autonomía. Dentro de los métodos para tal fin, se han planteado diversas estrategias como los frenos regenerativos, donde a través de un sistema de frenado híbrido (hidráulico y regenerativo) se cargan las baterías cada vez que el vehículo frena, realimentando la energía cinética hacia las baterías en forma de un voltaje de carga.

Este trabajo está enfocado en plantear un sistema de gestión energética para un prototipo de vehículo eléctrico logrado con control óptimo, principalmente basado en el principio de Pontryagin. Dentro de dicho sistema, entran etapas como la adquisición y procesamiento de señales, con un acelerómetro y giroscopio por eje, adecuación de señales a través de filtros análogos y digitales y un sistema de transmisión inalámbrica implementado con un módulo Xbee que se usa por su versatilidad en varios proyectos de tecnología inalámbrica de bajo costo. Para la corrección de la trayectoria, se establece un algoritmo basado en fusión de sensores con filtros Kalman, donde de forma recursiva y no iterativa se corrigen los datos sensados y transmitidos, efectuando un promedio entre la trayectoria pronosticada y la medida arrojada, calculando un peso de multiplicación según la desviación estándar para el dato adquirido.

1 PLANTEAMIENTO DEL PROBLEMA

1.1 DEFINICIÓN DEL PROBLEMA

Una de las problemáticas actuales con las que se enfrenta el mundo es la creciente necesidad de la oferta y la demanda energética requerida por la población mundial, y que originada también por el creciente desarrollo tecnológico en los diferentes campos de la industria y que se aplican en las actividades de la vida cotidiana. Para lidiar con esta problemática se han planteado múltiples soluciones, con una tendencia cada vez más clara, hacia el uso eficiente de energías renovables y no perjudiciales para el ambiente. Uno de los campos de interés, desde el punto de vista de la ingeniería, es el relacionado con el uso eficiente de las fuentes de energía que se utilizan en el transporte terrestre, tanto de pasajeros como de carga.

Actualmente, los motores de combustión interna de la mayoría de los vehículos que circulan por las ciudades del mundo utilizan derivados del petróleo como fuente principal de energía, y existen en el mundo cerca de 800 millones de vehículos de este tipo, que no solo requieren una alta demanda de combustible de origen fósil, sino que también generan consecuencias perjudiciales para el ambiente. Se proyecta un crecimiento de 1.500 millones de vehículos para el 2030 y de hecho, si se mantiene este índice de crecimiento, para el 2050 se esperarían unos 3.000 millones de autos en circulación en todo el mundo [2]. Por esta razón, existe un creciente interés de la población y la industria en fomentar el uso de fuentes renovables de energía y tecnologías alternas para el impulso de vehículos, por ejemplo, por medio de energía eléctrica almacenada en celdas o baterías y combinadas con motores de tipo eléctrico. En vehículos eléctricos (VE), el uso eficiente de esta energía almacenada se convierte en el principal requerimiento a tratar y la proposición de gestores de energía sobresale como uno de las principales líneas de investigación en ésta área del conocimiento.

La inclusión de los VE en el mercado mundial es cada día más sobresaliente; sin embargo, son varios los factores que aún contribuyen a su lenta expansión y aceptación por parte de los potenciales usuarios. Por ejemplo, para un VE su autonomía, el tiempo de recarga y su costo, determinado principalmente por el sistema de almacenamiento de la energía eléctrica, se convierten en las principales restricciones del sistema. Actualmente, la autonomía es el factor de mayor limitación, puesto que la energía máxima que se puede almacenar en las celdas no es muy grande y los rangos de aplicación de los VE aún son muy limitados. En comparación con los vehículos que utilizan combustibles fósiles, o los vehículos híbridos (VH), los VE se encuentran en desventaja; las densidades energéticas del diésel y de la gasolina son respectivamente 13 KWh/kg y 12,7 kWh/kg, mientras que la densidad energética de la última generación de baterías de iones de litio es de 0.16kWh/kg. Esto garantiza una mayor autonomía en los

autos tradicionales a pesar de su ineficiencia para convertir la energía química almacenada en kilómetros recorridos; además, existe una muy bien lograda infraestructura para el suministro de combustibles de origen fósil frente a una muy precaria red de distribución de energía eléctrica, sobre todo en países en vía de desarrollo [2].

Dado que el requerimiento de autonomía en un VE es fundamental, se plantea y desarrolla en este proyecto un sistema de gestión de energía para un prototipo de VE. Como la carga de las baterías debe extenderse el mayor tiempo posible, para que aumente el lapso entre recargas y se logre la mayor eficiencia en el uso de la energía eléctrica, se planteará un problema de optimización, que incluirá el recorrido de una distancia objetivo en el menor tiempo posible y con el menor gasto de energía eléctrica. También se propone el monitoreo de las variables internas y externas del sistema, como el registro de la aceleración y la posición del VE dentro de un circuito de prueba, así como también del registro de los perfiles de conducción desarrollados por un conductor de prueba. Al final, el gestor de energía eléctrica debe ser capaz de guiar al conductor del VE, por medio de una central de procesamiento externa al vehículo, hacia un uso eficiente de la energía eléctrica y a un mejor rendimiento en un circuito de prueba, a través de una conducción óptima. Basándonos en estos planteamientos, se podría plantear la siguiente pregunta de investigación: ¿Cómo desarrollar e implementar un sistema de gestión del recurso energético de un prototipo de VE, que optimice la distancia recorrida y el estado de carga de las celdas energéticas?

1.2 JUSTIFICACIÓN

Como se mencionó anteriormente, los vehículos convencionales con diseño de combustión interna (DCI) proveen buen rendimiento y un rango amplio de operación debido a su la alta densidad energética de los combustibles derivados del petróleo. Sin embargo, y paradójicamente, su principal desventaja radica en los bajos índices de economía y en la contaminación derivada por el uso de este tipo de combustibles. La principal causa del bajo índice de economía es la desarticulación del diseño de eficiencia de combustible con los requerimientos de operación real, sin considerar tanto la disipación de la energía cinética del vehículo durante el frenado, especialmente cuando se opera en áreas urbanas, como la baja eficiencia de la transmisión hidráulica en patrones de parada y arranque.

Por su parte los VE impulsados por batería poseen algunas ventajas sobre los vehículos convencionales DCI, tales como la alta eficiencia energética y el porcentaje de cero emisiones ambientales. Aunque en el desarrollo de los VE no se conoce muy a fondo su impacto a largo plazo, los estudios recientes prometen un gran impacto económico, social y ambiental positivo a diferencia del generado por los vehículos convencionales DCI. Por ejemplo, en [2] Shai Agassi, fundador de Better Place, señala que el automóvil medio europeo tiene un valor de 12.000

€, y que en sus 12 años de vida consume aproximadamente unos 30.000 l de combustible, lo que representa un costo de 30.000 € a 35.000 €; por lo tanto, si se mantiene esta tendencia creciente, y dependiendo claro está del país, solo el combustible llegaría a costar el triple del valor del vehículo. Por otro lado, la batería de un automóvil eléctrico cuesta 7.000 € y la electricidad consumida en toda su vida ascenderá a sólo 2.000 €; la suma de éstos valores solo representa un tercio del combustible consumido por un automóvil a gasolina o diésel. Además, el costo de las baterías y la electricidad, de fuentes renovables como la eólica, tienden a reducirse a lo largo de los años; lo que no sucede con los hidrocarburos, cuya tendencia es a subir, independientemente de bajas transitorias, como las provocadas por las crisis económicas mundiales.

Por otro lado, en la misma referencia, se estima que la suma del costo de las baterías y la electricidad se reducirá a unos 5.000 € para el año 2015 y unos 3.000 € para el 2020, mientras que para los vehículos tradicionales los costos de los combustibles derivados del petróleo superarán los 30.000 €. Esto, junto con una futura mejora en la eficiencia de los VE, se traducirá en una mejora económica para los propietarios de VE. Por ejemplo, en Chattanooga, Tennessee, EU, y desde hace casi dos décadas, ya funcionan autobuses eléctricos urbanos con excelentes resultados, puesto que consumen cuatro veces menos en combustible y la mitad en mantenimiento. Éstos tienen menos partes móviles, ya que en el diseño del VE sólo se mantiene la transmisión trasera; el cilindro de freno y depósito; el condensador y evaporador, si el aire acondicionado es requerido en el diseño final; la dirección y articulación frontal y el cableado eléctrico existente. Partes que se incluyen en el diseño de combustión interna, tales como el iniciador, el alterador, la bomba de dirección, el radiador, el núcleo calefactor, las líneas de combustible, el tanque de combustible, el escape y el silenciador, son retiradas en el diseño final de un VE [3]. Otro ejemplo lo constituyen los servicios postales en Francia, que han adquirido 10.000 VE. Tras el éxito de la prueba inicial, con gastos de operación seis veces inferiores a los de los vehículos diésel, otros países, como España, se han sumado a las empresas que han dado un primer paso hacia la electrificación de la flota de reparto, y no solo en el servicio postal; en ciudades como Madrid ya funcionan microbuses eléctricos.

Desde el punto de vista ambiental, los VE no emiten gases contaminantes; sin embargo, la electricidad necesaria para impulsar el vehículo necesita ser obtenida de alguna fuente, y esta producción de energía, dependiendo del tipo de generación, contribuirá al impacto total no solo ambiental sino también al económico y social que rodea el uso de los VE. El uso de los VE en ambientes urbanos, especialmente en ciudades con alto índice de contaminación, contribuirá drásticamente no solo a la reducción de los gases contaminantes, sino también a la reducción de los niveles de ruido. Teniendo en cuenta lo anterior, se evidencia la importancia de la investigación y desarrollo en sistemas de gestión y técnicas de optimización de los recursos energéticos utilizados para el impulso de los VE. Desde el punto de vista académico, la investigación y la aplicación de éstos sistemas de gestión y técnicas de optimización, en el ámbito regional y nacional, y

acordes con el contexto internacional, motivan el interés por el estudio y el uso de fuentes alternativas de energización, y abren las puertas para su aplicación en varios campos de la ingeniería.

1.3 ANTECEDENTES

En el ámbito internacional, la preocupación por el desarrollo de tecnologías amigables con el medio ambiente, y que reduzcan los niveles de contaminación en las grandes ciudades, han conducido al fomento de investigaciones para la reducción de uno de los factores que más contribuyen a la generación de gases de efecto invernadero, como lo es la propulsión de vehículos a base de hidrocarburos. En éstas investigaciones se proponen nuevos sistemas eléctricos de propulsión en conjunto con nuevas formas de optimización del recurso energético, beneficiando de manera directa al usuario final, que se ha decidido por utilizar vehículo eléctrico.

En el contexto nacional, después de años de pruebas e investigaciones, Ecocity, ofrece una alternativa práctica y ecológica de VE para el transporte en la capital colombiana. Estos vehículos se caracterizan por tener un peso aproximado de 240 kg, las baterías son de ciclo profundo y un pequeño motor en cada una de las ruedas traseras; la carga se realiza durante toda la noche y ofrecen una autonomía de 40 Km, sin consumo de batería durante los embotellamientos típicos de Bogotá. Además de esta innovadora idea de fabricación de VE, 100% colombianos, Bogotá, y otras ciudades como Medellín y Cali, son pioneras en la adquisición de este tipo de tecnologías a través de empresas como Codensa, la cual adquirió en el último año15 modelos de VE, EPM y EPSA, con 10 y 5 modelos respectivamente, y diseñados por Mitsubishi Motors. Ésta última cuenta con 40 años de experiencia en el campo de los VE y hace 3 años lanzó en el mercado colombiano el i-MiEV [5].

Bogotá se proyecta como la segunda ciudad latinoamericana con la mayor flota de taxis eléctricos, después de Ciudad de México. El proyecto piloto contará con 35 puntos de recarga en tres puntos cardinales de la ciudad, gracias a los convenios realizados entre el presidente mundial de BYD [6], Wang Chuanfu, el gerente general de Codensa, David Felipe Acosta, y el gerente general de Praco Didacol, Gustavo Pradilla. El costo de este vehículo será de alrededor de 86 millones COP, pero a diferencia de los taxis convencionales no habrá que pagar por la matrícula, dado que es un proyecto piloto, durante 5 años. La autonomía de estos vehículos está por encima de los 250 Km diarios y será suficiente para cubrir los recorridos que se realizan en promedio en la ciudad, que están entre los 300 Km y 350 Km, ya que el faltante de autonomía se completaría con la recarga suplementaria al medio día. Con respecto a las emisiones de CO_2, se evitarán 1.088 t al año con un recorrido total de 109 mil kilómetros en el mismo tiempo [7].

Tanto a nivel académico como industrial, se investiga sobre estrategias para el mejoramiento de la autonomía de los VE a través del uso eficiente de la energía eléctrica, que se puede utilizar sola o también en conjunto con otro tipo de combustible derivado del petróleo, y la infraestructura de alimentación, que en algunos países ya se encuentra muy avanzada. Estas estrategias pretenden dar solución a distintos requerimientos, como la selección de la trayectoria óptima, el logro del menor tiempo de recorrido o la emisión mínima de gases, cuando se trata de VH. Por ejemplo, en [1, 2011] se propone una estrategia de gestión de energía supervisada, cuyo objetivo es la optimización del uso de la energía de un VH que se conecta a una red de alimentación. Esta estrategia, también conocida como control basado en modelo se basa principalmente en la minimización de las emisiones de CO2. Aquí se formula un gestor de energía como un problema de control óptimo global que luego es cambiado a un problema local aplicando el principio mínimo de Pontryagin [8]. El gestor se prueba mediante simulaciones y datos experimentales, que incluyen las condiciones de uso del vehículo, los factores del entorno y algunos escenarios geográficos y perfiles de conducción.

Stockar asume la similitud que se puede asumir entre la gestión del consumo del combustible y el consumo de energía eléctrica en un VE; esta estrategia se conoce como Estrategia de Minimización de Consumo Equivalente (EMCE) y es ampliamente utilizada en VH con acceso a la red de alimentación. Esta aproximación asume que la energía consumida por el vehículo puede ser convertida en un equivalente del consumo de combustible, y se concluye que el gasto óptimo del combustible puede ser alcanzado si el algoritmo de control agota la batería proporcionalmente a la distancia conducida. Esto implica que la velocidad del vehículo se debe conocer a priori y esto evita que la EMCE se pueda generalizar para tipo de perfil de conducción.

Al igual que en [1], en [9, 2011], se plantea una estrategia de optimización del uso de la energía de un VH, pero cercana a la óptima. El objetivo de esta propuesta es reducir la complejidad de la solución, que como se sabe, puede ser muy demandante en recurso computacional y necesita además del conocimiento a priori de algunas de las variables que se incluyen dentro de la función de costo. Esta estrategia se traduce en una aproximación del principio de Pontryagin en tiempo real. Esta estrategia encuentra una solución global óptima basada en el hecho de que la resistencia interna de la batería y el voltaje en circuito abierto de la misma se pueden considerar independientes del estado de carga. Se presentan simulaciones y se comparan con los resultados obtenidos con programación dinámica.

Por otra parte, y a diferencia de [1] y [9], en [10, 2010], se propone una estrategia de tipo causal para el control del consumo de combustible en VH. Esta estrategia está basada en el Control Predictivo basado en Modelo (CPM), que utiliza un horizonte finito de combustible y un aproximado del cost-to-go. Con esta estrategia se trata de solucionar el problema de la no causalidad y el costo computacional del control óptimo; así como también de aprovechar las ventajas de

la causalidad y la no dependencia del conocimiento a priori de los ciclos de conducción. Como método de solución del problema de optimización se usa la relación entre la ecuación de Hamilton-Jacobi-Bellman y el principio mínimo de Pontryagin [8].

También se han planteado sistemas de optimización energética a través de la energía regenerativa obtenida por medio de frenos regenerativos. En [11, 2010], se hace una extensión del rango de conducción con frenos regenerativos. Se mitiga así el problema del incremento del número de baterías para mejorar la autonomía del vehículo. Sin embargo, las operaciones con frenos regenerativos deben estar coordinadas con los frenos hidráulicos para constituir el llamado sistema de freno híbrido (FH), el cual puede ser dividido en tipo serie y tipo paralelo. El primero es aplicado independientemente a cada rueda por modulación hidráulica, lo que permite la transmisión de tanta potencia como sea posible dentro de los límites aceptables del neumático sobre la carretera; los frenos tipo serie sólo son factibles para VE equipados con frenos electrohidráulicos. Por otro lado, los frenos tipo paralelo, puede aplicar el freno regenerativo y paralelamente el freno hidráulico para alcanzar desaceleración; sin embargo, una característica crítica con pestos sistemas es que mientras se maximiza la energía regenerativa se puede llegar a bloquear la llanta; por lo tanto, el controlador de este último freno debe determinar el torque regenerativo sujeto a ciertos requerimientos de seguridad en la conducción.

Aunque en la actualidad proliferan las aplicaciones para vehículos híbridos, en este proyecto nos enfocaremos solo al diseño y validación experimental de un sistema de gestión para la energía eléctrica que impulsará un modelo de VE. Este gestor tendrá en cuenta las variables internas y externas del vehículo en un circuito de prueba.

2 OBJETIVOS

2.1 OBJETIVO GENERAL

Diseñar e implementar un sistema de gestión de energía eléctrica para un prototipo de vehículo eléctrico VE, para optimizar el consumo de energía y la distancia recorrida en un circuito de prueba.

2.2 OBJETIVOS ESPECÍFICOS

- Construir un modelo de VE en el plano.
- Revisar algunos modelos de gestión de energía eléctrica y estrategias de control.
- Diseñar e implementar un gestor de energía eléctrica, en computador, con base en los modelos y estrategias ya analizadas.
- Diseñar e implementar los sistemas de adquisición y procesamiento de datos para las variables más relevantes del prototipo de VE.
- Diseñar e implementar el sistema de telemetría entre el prototipo de VE y una estación base estática.
- Calibrar el modelo del vehículo.
- Validar el sistema de gestión de energía eléctrica con datos experimentales en un circuito de prueba.

3 ADQUISICIÓN Y PROCESAMIENTO DE DATOS

3.1 CONCEPTOS ESPECÍFICOS SOBRE ADQUISICIÓN Y PROCESAMIENTO DE DATOS

3.1.1 REGULACIÓN DE TENSIÓN

Generalmente, el voltaje proporcionado por las fuentes de alimentación es un voltaje que sobrepasa los límites de algunos compontes; por ejemplo, el voltaje de alimentación de la tarjeta GY85 está en 3.3v, requiriendo de una etapa de regulación adecuada.

Los reguladores de voltaje están diseñados para un amplio rango de aplicaciones, donde además de incluir la regulación de voltaje aportan a la supresión del ruido generado por la fuente de alimentación.

En este trabajo se implementa la referencia LM7805 y LM317T, los cuales pueden soportar hasta 1.5 A en la salida de corriente, proporcionando características ideales para la implementación en la tarjeta de adquisición, alimentando con voltajes de 5v y 3.3v, respectivamente.

3.1.2 SENSOR

Son dispositivos capaces de convertir una magnitud física, como puede ser la temperatura, la presión, el valor de pH, etc., en una diferencia de potencial o una variación de intensidad. Es decir, realizan una conversión de energías y suministran información sobre el estado y tamaño de la magnitud. Los sensores informan de su entorno y además esa información es cuantificable, es decir, medible por algún instrumento [12].

Los dispositivos usados para realizar el sensado de las variables de voltaje, aceleración y giro, son: un conversor ADC, tarjetas de bajo coste como: IMU GY 85; donde se incorpora acelerómetro ADXL345, magnetómetro HMC5883L y giróscopo ITG3205 cuyas variables a medir son: aceleración en los 3 ejes de coordenadas, campo magnético y velocidad angular, respectivamente.

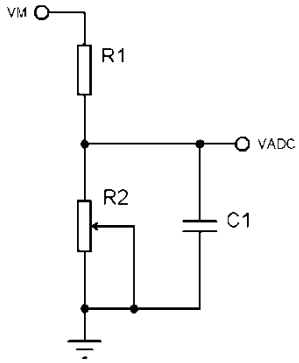
Cada uno es configurado según los requerimientos necesarios en su implementación, teniendo en cuenta las características físicas del vehículo. En este caso se experimentará una aceleración no mayor a 1g. Respecto a los ángulos de orientación del vehículo (pitch, yaw y roll), el giroscopio proporciona según su configuración una velocidad angular de ±2000°/s, lo cual es suficiente para un vehículo en condiciones normales de funcionamiento, es decir, sin que se genere volcamiento del mismo.

3.1.2.1 VOLTÍMETRO

Es un instrumento que sirve para medir la diferencia de potencial entre dos puntos de un circuito eléctrico.

La medición del voltaje se realiza a través de un sensor diseñado con un divisor de tensión cuyo rango de medición está entre 0 V a 15 V, con un factor de conversión de 0.3368 LSB/V. Se usa un capacitor de 10 µF que disminuye el rizado de la señal al momento de tomar la muestra en cada ciclo por parte del conversor análogo digital.

Figura 1. Circuito medidor de tensión.



Fuente propia.

El factor de conversión se determina haciendo uso de la fórmula para calcular divisores de tensión:

$$V_M = \frac{V_{ADC} * (R_1 + R_2)}{R_2} \quad (1)$$

donde:

V_M: Voltaje de la fuente de alimentación

R_1 y R_2 : son resistencias divisoras de tensión

C1: es el capacitor supresor de rizado.

Dicha configuración arroja en V_{ADC} un voltaje proporcional y lineal a V_M medido por el conversor análogo digital.

3.1.2.2 ACELERÓMETRO

El ADXL345 es un acelerómetro integrado de tres ejes micro maquinizado, superficial de alta resolución con un rango máximo de medida de $\pm 16g$. El dispositivo consiste de una celda de sensado capacitiva o celda g y un acondicionador de señal contenido en un solo paquete. La celda g es una estructura mecánica formada por materiales semiconductores de polisilicio acoplados con procedimientos de semiconductores como enmascaramiento y grabado.

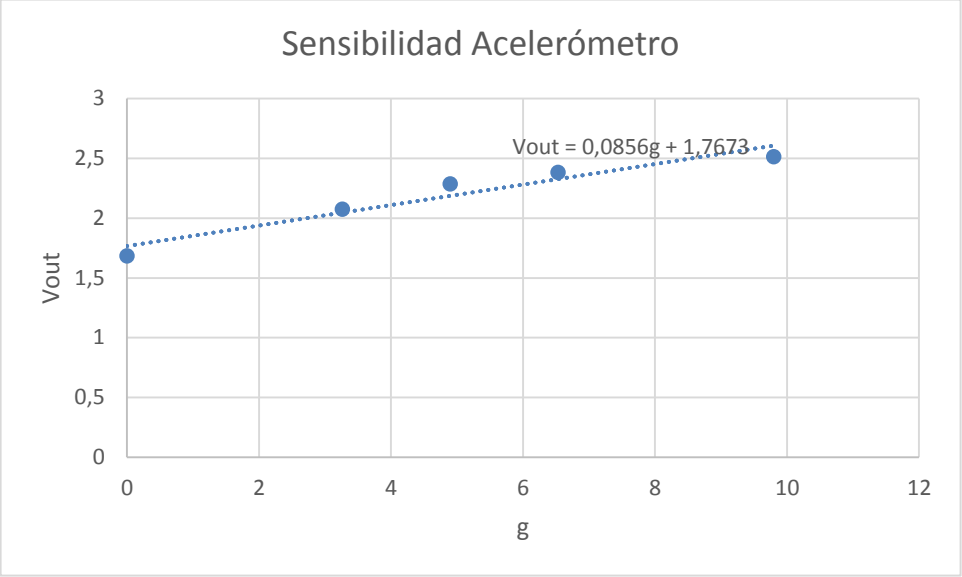
Los datos digitales obtenidos son de 16 bits en complemento a dos y una interfaz digital I^2C. La medida digitalizada corresponde a la aceleración estática de la gravedad, en el caso de aplicaciones donde se requiera el censado de inclinación. En el caso de aceleración dinámica, resulta del movimiento o choque. En la configuración de alta resolución (3.9 mg/LSB) se pueden efectuar medidas de variación menores de 1°.

El dispositivo cuenta con varias funciones como: la actividad o inactividad en el sensado, la presencia o ausencia de movimiento comparando la aceleración de algún eje con el umbral establecido al ser configurado; la detección de un “tap” o doble “tap” en alguna dirección y detección de caída libre si el dispositivo está cayendo.

La configuración de la sensibilidad se realiza a través de una palabra de 8 bits ubicada en la posición de memoria 0x31 dando la opción de medir entre: ±2, ±4, ±8 y ±16g que corresponden a los números en binario 0, 1, 2 y 3 respectivamente.

Si bien, en la hoja de características del acelerómetro se presentan valores para la sensibilidad, también se puede determinar su valor inclinando el acelerómetro un ángulo conocido y promediar los resultados generando una recta de relación voltaje-gravedad. Este proceso estadístico se llama método de regresión lineal, dicho método se emplea haciendo mediciones reales en el sistema a partir de las cuales se determinan los parámetros requeridos. En este caso, se asume la relación voltaje–gravedad como una relación lineal con lo cual se puede concluir en una recta cuya pendiente es la sensibilidad y cuyo punto de corte con el eje y es el valor de offset en el sensor. En dicho proceso se obtuvo el resultado mostrado en la figura 2.

Figura 2. Sensibilidad medida



Fuente propia

Se obtiene una sensibilidad de 85,6 mV/g y un V_{off} = 1,7673 V. En el caso del valor de offset, se hace uso del promedio de valores en un rango de muestras, 30 muestras. Este procedimiento arroja datos más cercanos a la que es la referencia cero en el sensor.

3.1.2.3 GIROSCOPIO ELECTRÓNICO

Es una micromáquina giroscópica capaz de medir la variación angular sobre los ejes de “pitch”, “roll” y “yaw”.

El giroscopio es la combinación de un actuador y un acelerómetro integrados en una sola estructura de micro máquina. Se incluye un elemento de censado conformado por una sola masa de conducción, manteniendo un movimiento de oscilación continuo y capaz de reaccionar cuando una variación angular es aplicada según el principio de Coriolis.

El sensor ITG3205 es un dispositivo de salida digital de censado en los tres ejes, optimizado para juegos, ratones y aplicaciones remotas en 3D. La parte característica de este sensor es el mantenimiento de los valores de desviación durante el sensado y la sensibilidad con la estabilización de la temperatura, reduciendo la necesidad de usar calibración. El ruido de baja frecuencia es menor que en los dispositivos de las generaciones previas, simplificando la implementación en desarrollos de dispositivos y obteniendo una mejor respuesta en controles remotos.

3.1.3 OBTENCIÓN DE MEDIDAS

Según los sensores implementados, los datos obtenidos para el acelerómetro, giroscopio y magnetómetro son datos de medida de gravedad, velocidad angular y gauss, respectivamente. Sin embargo, cada medición realizada debe ser procesada matemáticamente para conseguir el resultado de la medida deseada; en este caso, se implementa el algoritmo del método trapezoidal para la integración numérica.

3.1.3.1 MÉTODO TRAPEZOIDAL

Para obtener la posición a partir de la medida arrojada por el acelerómetro, se debe tener en cuenta que la velocidad es la razón de cambio de la posición de un objeto. En otras palabras, la velocidad es la derivada de la posición y la aceleración es la derivada de la velocidad, así:

$$\vec{a} = \frac{d\vec{v}}{dt} \qquad (2)$$

$$\vec{v} = \frac{d\vec{s}}{dt} \qquad (3)$$

Por lo tanto:

$$\vec{a} = \frac{d^2\vec{s}}{dt^2} \qquad (4)$$

Si la aceleración de un objeto es conocida, se puede obtener los datos de posición aplicando una doble integración, asumiendo que las condiciones iniciales son cero; es decir para se define el tiempo inicial como $t_0 = 0$.

$$\vec{s} = \int_0^t \left(\int_0^t (\vec{a}\,(t))dt \right) dt \qquad (5)$$

donde:

t : Tiempo final

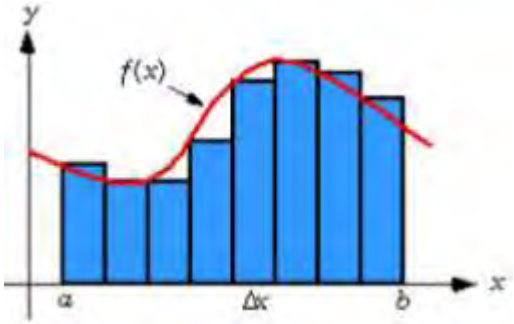
Dada la definición de la integral como el área bajo la curva de una señal, donde la integración es la suma de áreas suficientemente pequeñas donde su ancho es cercano a cero. Es decir, la suma de las pequeñas áreas bajo la curva, representa la magnitud de una variable física.

Las aproximación realizada entre dos puntos de una función *f(x)* como una recta, junto con la eficiencia computacional implicada que proporciona el método trapezoidal, hacen de la integración de la señal original un proceso donde se

obtiene el área bajo la curva del segmento [a, b], entre un tiempo t_{k-1} y t_k identificado como Δx, obteniendo un buen rendimiento y mejores resultados.

En la figura 3, se tiene el ejemplo de una función *f(x)* cuya aproximación de la integral se inicia realizando el muestreo de la señal, reteniendo valores instantáneos de su magnitud y dando lugar al cálculo de pequeñas áreas entre dos muestras consecutivas.

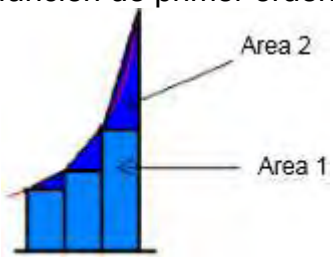
Figura 3. Señal muestreada.



Fuente: Seifert y Camacho. 2007. Implementing positioning algoritms using accelerometers.

Dado que existe un error en la aproximación de la señal en cuestión, el método propuesto asume el área total entre dos muestras consecutivas como la suma de dos áreas, figura 4; área total de un rectángulo y área de una aproximación a una función de primer orden.

Figura 4. Aproximación a función de primer orden (recta) en área 2.



Fuente: Seifert y Camacho. 2007. Implementing positioning algoritms using accelerometers.

Con las definiciones anteriores, se puede llegar a (6) cuya expresión representa el área total bajo la curva de una función *f(x)* en el tiempo *k*.

$$a_k = f(x)_k * T + \frac{|f(x)_k - f(x)_{k-1}|}{2} * T \quad (6)$$

donde:

T: tiempo de muestreo de la señal.

a_k: área bajo la curva en el instante *k.*

El método implementado en este trabajo para obtener la integración de las señales se aplica tanto en el acelerómetro como del giroscopio. En el primero, la aplicación de (6) se hace dos veces consecutivas para obtener el valor de la posición del objeto; y en el segundo, se aplica una sola vez para obtener la posición en grados a partir de la velocidad angular. Cabe resaltar que el procedimiento se realiza para cada eje lo que implica un consumo de recursos de procesamiento de datos.

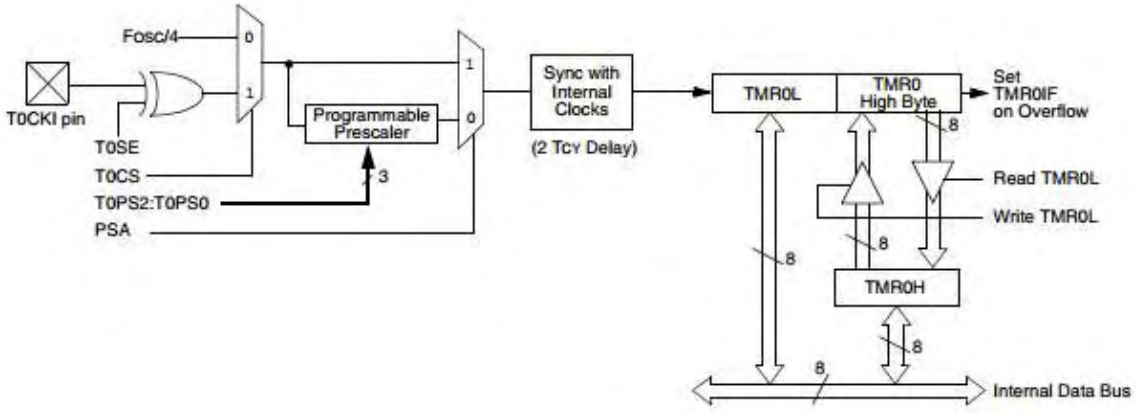
Sin embargo, cada proceso de integración de la variable original agrega un error acumulativo en cada ciclo de toma de muestras por lo cual se hace necesario un algoritmo iterativo para la corrección del error en el desplazamiento, sea angular o lineal.

3.1.3.2 MEDIDA DEL TIEMPO DE MUESTREO DE LAS SEÑALES

Según la velocidad de procesamiento del microcontrolador 18F2550, el tiempo de muestreo de las señales es del orden de los milisegundos y no es posible medirlo con métodos convencionales y/o empíricos. Por lo tanto, se configura un contador en el mismo chip, el cual proporciona la velocidad y precisión de cuantificación requerida para realizar las mediciones de la diferencia de tiempo entre muestra y muestra.

Analizando la estructura interna del microcontrolador, como se muestra en la figura 5, se observa que la frecuencia del oscilador de cristal de 20Mhz es divida entre 4 y afectada por el preescaler que divide la frecuencia, aumentando el tiempo de duración de un ciclo máquina y se puede calcular según (7). Tomando un preescaler de 1; es decir, que la división de la frecuencia se realiza en una unidad, se puede efectuar el cálculo directo.

Figura 5. Diagrama de bloques del timer 0 en modo 16 bits.



Fuente: ww1.microchip.com/downloads/en/devicedoc/39632c.pdf

$$CM = \frac{4}{F_{OSC}} \quad (7)$$

donde:

CM : Ciclos máquina

F_{OSC} : Frecuencia de oscilación del cristal.

3.1.3.3 MEDIDA DE INCLINACIÓN USANDO GIROSCOPIO

Teniendo en cuenta lo anterior, el giroscopio mide la velocidad angular a la cual está rotando el objeto en cada uno de sus ejes. Para obtener el desplazamiento angular, se procede a integrar la medida según el método trapezoidal que consiste en multiplicar la medida realizada por la diferencia de tiempo respecto a la medida anterior [17].

$$\frac{d(\alpha)}{dt} = \omega \quad (8)$$

$$d(\alpha) = \omega * dt \quad (9)$$

$$\int_0^t \omega * dt = \alpha \quad (10)$$

donde:

α: ángulo recorrido.

ω: velocidad angular.

dt: variación de tiempo entre muestra y muestra.

Mediante este modelo, se puede determinar el ángulo de desplazamiento en un instante de tiempo, siendo necesario establecer la inclinación como el resultado de un algoritmo acumulativo, de donde se puede obtener la siguiente expresión:

$$\propto_k = \propto_{k-1} + u_k * dt \qquad (11)$$

donde:

u_k: Señal acondicionada del giroscopio $[^{\omega}/_{S}]$.

dt: Tiempo de muestreo $[s]$.

$\propto_k$: Estado presente [°].

$\propto_{k-1}$: Estado anterior [°].

Al tratar de utilizar solamente el giroscopio en la medida del ángulo, se debe tener en cuenta la deriva del sensor que se produce a lo largo del tiempo debido al algoritmo acumulativo, por lo tanto se tendrá una medida confiable en cortos intervalos de tiempo solamente.

Según lo anterior, se puede concluir que: es casi imposible hacer medidas de desplazamiento angular solamente con giroscopios; sin embargo, se puede incluir en la medición otro dispositivo que contrarreste en cierto modo los problemas que acarrea el uso único del giroscopio y para ello, se hace uso del acelerómetro y el magnetómetro.

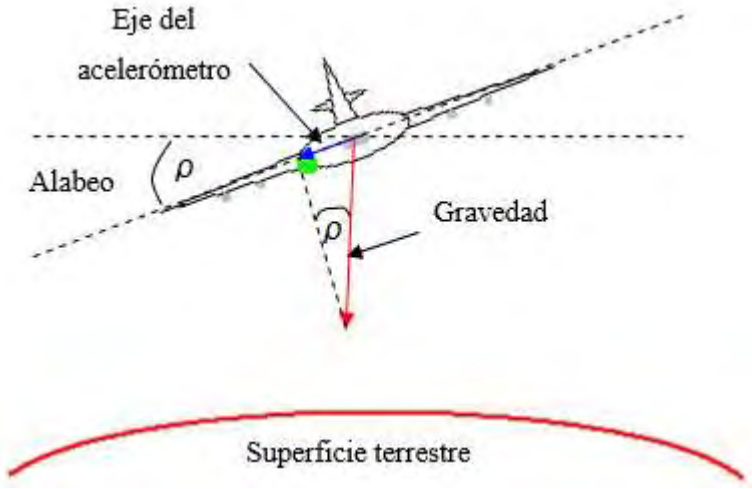
3.1.3.4 MEDIDA DE INCLINACIÓN USANDO ACELERÓMETRO

Dentro de las aplicaciones del acelerómetro, se puede utilizar dicho dispositivo como sensor de la inclinación según la aceleración a la cual sea sometido. La sensibilidad de los acelerómetros aumenta cuando sus ejes se encuentran perpendiculares a la aceleración de la gravedad, es decir, paralelos a la superficie terrestre.

Como condiciones para mejorar la exactitud de la medida, el acelerómetro se debe encontrar fijo a la superficie del objeto donde se van a realizar las mediciones de tal forma que si la plataforma se encuentra perpendicular a la gravedad, los dos ejes “x” e “y” se encuentren paralelos con respecto a tierra; es decir, perpendiculares a la gravedad.

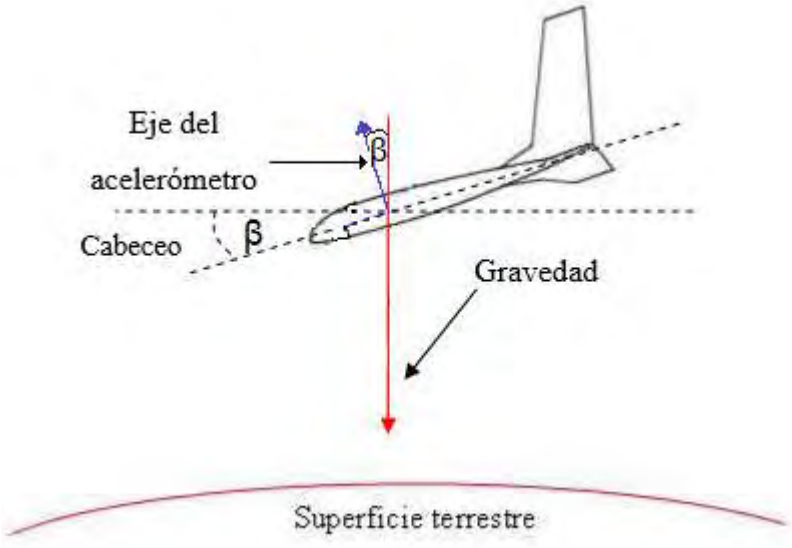
En las figuras 6 y 7, se representa lo que el acelerómetro está midiendo a causa de la gravedad y la aceleración de la gravedad con las flechas azul y roja respectivamente. Con fines prácticos, en las gráficas se representan los cambios angulares con un aeroplano.

Figura 6. Alabeo con un ángulo ρ en sentido anti – horario



Fuente: Pozo, David. 2010. Diseño y construcción de una plataforma didáctica para medir ángulos de inclinación usando sensores inerciales como acelerómetro y giroscopio.

Figura 7. Cabeceo con un ángulo β en sentido anti – horario.



Fuente: Pozo, David. 2010. Diseño y construcción de una plataforma didáctica para medir ángulos de inclinación usando sensores inerciales como acelerómetro y giroscopio.

Para obtener el valor en grados de la señal entregada por el sensor, se realizan cálculos geométricos que no veremos con detalle en este trabajo, desde los cuales se obtienen las siguientes fórmulas.

$$\rho = \operatorname{asin}\left(\frac{ax}{1g}\right) \quad (12)$$

$$\beta = \operatorname{acos}\left(-\frac{1g}{az}\right) \quad (13)$$

donde:

ρ: ángulo de alabeo.

β: ángulo de cabeceo.

La condición para poder aplicar (12) y (13) como medición indirecta de *ρ* y β en la adquisición, es que no debe existir una aceleración lineal sobre los ejes en cuestión dado que el acelerómetro experimentaría una fuerza extra y la medida sería imposible.

3.1.3.5 MEDIDA DE ÁNGULO DE DERRAPE USANDO MAGNETÓMETRO

La medida de derrape o su equivalente en inglés yaw, puede ser determinada usando solamente las componentes en *x* e *y* (14) del campo magnético de la tierra, es decir, la dirección planar respecto a la superficie de la tierra o del campo electromagnético predominante lo cual genera también inconvenientes al no tener un campo magnético estable. Las lecturas tomadas en cada dirección, varían según el giro del magnetómetro sobre su propio eje; es decir, cambiando la intensidad del campo de la tierra en este punto para ambos datos obtenidos. La orientación puede ser determinada en grados desde las lecturas de los ejes *x* e *y*; y el desplazamiento se calcula sustrayendo de la dirección actual la medida anterior, además de un algoritmo acumulativo el cual determina el desplazamiento angular total respecto de una posición inicial.

$$\gamma = atan2(H_y, H_x) \quad (14)$$

donde:

γ : dirección actual respecto al norte magnético de la tierra [°].

H_y: Componente en el eje *y* de la incidencia del campo magnético de la tierra sobre el sensor [G].

H_x: Componente en el eje *x* de la incidencia del campo magnético de la tierra sobre el sensor [G].

Con (14), obtenemos la orientación del eje *x* respecto del norte magnético de la tierra con variaciones de 0 a π y 0 a –π; por lo tanto, para poder medir ángulos, se

debe determinar primero cada orientación con un valor entre 0 a 2π aplicando (15).

$$\varphi = \begin{cases} \varphi\ ;\ 0 < \varphi < \pi \\ \varphi + 360;\ 0 < \varphi < -\pi \end{cases} \quad (15)$$

donde:

φ: dirección en grados en un rango de 0°-360° [°].

El grupo de ecuaciones anteriores expresan la dirección entre 0° y 360° y (16) expresa la medida del ángulo recorrido en un instante de tiempo. Se debe tener en cuenta que en la medida del ángulo resultante cambia según (16) si varía del cuadrante I al cuadrante IV del plano cartesiano o viceversa.

$$\zeta_{l-1} = \begin{cases} 360 + \zeta_{l-1};\ 270 < \zeta_l < 360 \wedge 0 \leq \zeta_{l-1} < 90 \\ \zeta_{l-1} - 360;\ 0 \leq \zeta_l < 90 \wedge 270 < \zeta_{l-1} < 360 \end{cases} \quad (16)$$

donde:

l: número de muestra.

ζ: ángulo medido entre muestra y muestra [°].

Se cambia el valor del ángulo a sustraer de la posición actual según (17).

$$\theta = \sum_{l=1}^{n} \theta_l - \theta_{l-1} \quad (17)$$

donde:

θ: desplazamiento angular [°].

n: número de muestras.

3.1.3.6 MEDIDA DE INCLINACIÓN Y GIRO USANDO ACELERÓMETRO, GIROSCOPIO Y MAGNETÓMETRO

Tomando como referencia los tres métodos anteriores, se aprecia claramente que al usar tanto el giroscopio como el acelerómetro de manera individual, se tienen varios inconvenientes que producen que el resultado deseado tienda a tener un error aditivo en su magnitud.

Debido a estos inconvenientes, surge la necesidad de trabajar con los tres sensores simultáneamente, con el fin de poder tomar las mejores características de cada uno y que a la vez se contrarresten en cierto grado los inconvenientes.

Existe un método capaz de lograr esta fusión de características positivas y atenuación de los problemas individuales, este método es llamado Filtro de Kalman.

3.1.4 ACONDICIONADOR DE SEÑAL

El acondicionador de señal genera a partir de lo obtenido de los sensores, una señal que sea aceptable por las tarjetas de adquisición de datos. Dichas tarjetas suelen medir tensiones que van entre unos márgenes determinados por ejemplo,

-10v a 10v, 0v a 10v, 0v a 5v.

Las funciones principales que realizará el acondicionador de señal son las siguientes [14]:

3.1.4.1 TRANSFORMACIÓN

Los sensores pueden proporcionar una diferencia de potencial, o variable de intensidad. Normalmente las tarjetas de adquisición de datos admiten diferencias de potencial, por lo que si el sensor proporciona una variación de intensidad, esta debe ser convertida en una diferencia de potencial proporcional. Los diferentes sensores y su correspondiente factor de conversión se presentan en la siguiente tabla:

Tabla 1. Sensor y sensibilidad respectiva

SENSOR	SENSIBILIDAD
Acelerómetro	3.9 g/LSB
Giroscopio	14.375 LSB per °/sec
ADC	4.88mV/LSB

Fuente propia.

Para el acelerómetro se utiliza una sensibilidad de 3.9 g/LSB dado que el vehículo presenta aceleraciones muy pequeñas y sería inaplicable en este trabajo.

3.1.4.2 CONVERSIÓN ANÁLOGA DIGITAL

La conversión análoga digital se realiza para relacionar las variables a medir, sea aceleración o velocidad angular, y el hardware conjuntamente con los sistemas de cómputo para el correcto procesamiento de la información recibida desde los sensores hacia la computadora.

El sistema de conversión depende de varios parámetros que se determinan según los requerimientos del sistema a medir. Dichos parámetros establecen la precisión del sistema de adquisición, resolución, error de conversión y velocidad de procesamiento de los datos, comunicación, envío y recepción de variables directamente involucradas en proceso de conversión análoga–digital.

Para realizar el proceso de adquisición, se utiliza el microcontrolador 18f2550 el cual consta de un conversor de 10 bits; parámetro que determina la resolución del sistema en 4,88 mV estableciendo una referencia de 5v según [16].

$$1LSB = V_{ref-} + \frac{(V_{ref+} - V_{ref-})}{2^n} \quad (18)$$

Donde:

$1LSB$: es el bit menos significativo

V_{ref-}: es el voltaje mínimo

V_{ref+}: voltaje de referencia máximo.

Si se determina que $V_{ref-} = 0$, entonces la ecuación a implementar sería:

$$1LSB = \frac{V_{ref+}}{2^n} \quad (19)$$

3.1.4.3 CALIBRACIÓN ESTÁTICA DE GIROSCOPIO Y ACELERÓMETRO

Para obtener una correcta medición de las variables arrojadas por los dispositivos como el acelerómetro y el giroscopio, se debe tener en cuenta que el conversor análogo digital está limitado para medir voltajes negativos; por lo que el fabricante, realiza la medición sobre un “offset” o valor de DC, a partir del cual debe asumir el punto cero de todas las mediciones.

El proceso de eliminación del valor de DC, se efectúa calculando el promedio de la señal con un número de muestras determinado y en total reposo para asumir el estado inicial del vehículo. Una vez obtenido el número de muestras suficientes, y calculado el promedio de la señal, se sustrae dicho valor de todas y cada una de

las mediciones realizadas a partir de la finalización del proceso de calibración según (20).

$$\mu = \frac{1}{N}\sum_{k=1}^{N} x_k \qquad (20)$$

Donde

μ: promedio de la señal.

N: número de muestras.

x_k: valor de la muestra.

k: muestra actual.

3.1.4.4 CALIBRACIÓN DINÁMICA DEL MAGNETÓMETRO

Al realizar la medida de la variación del campo magnético en cada uno de los ejes, se debe realizar una rotación de 360° sobre el plano que se requiera la medición. Dicha rotación genera una variación normalizada entre -1 y 1 de valores en la medida del campo magnético tanto del eje *x* como del eje *y* donde la combinación de medidas genera una circunferencia en el plano perpendicular al eje de rotación. El circulo generado debe estar centrado en los valores 0,0 si el magnetómetro está perfectamente calibrado.

En la primera prueba, se generan los siguientes valores de offset:

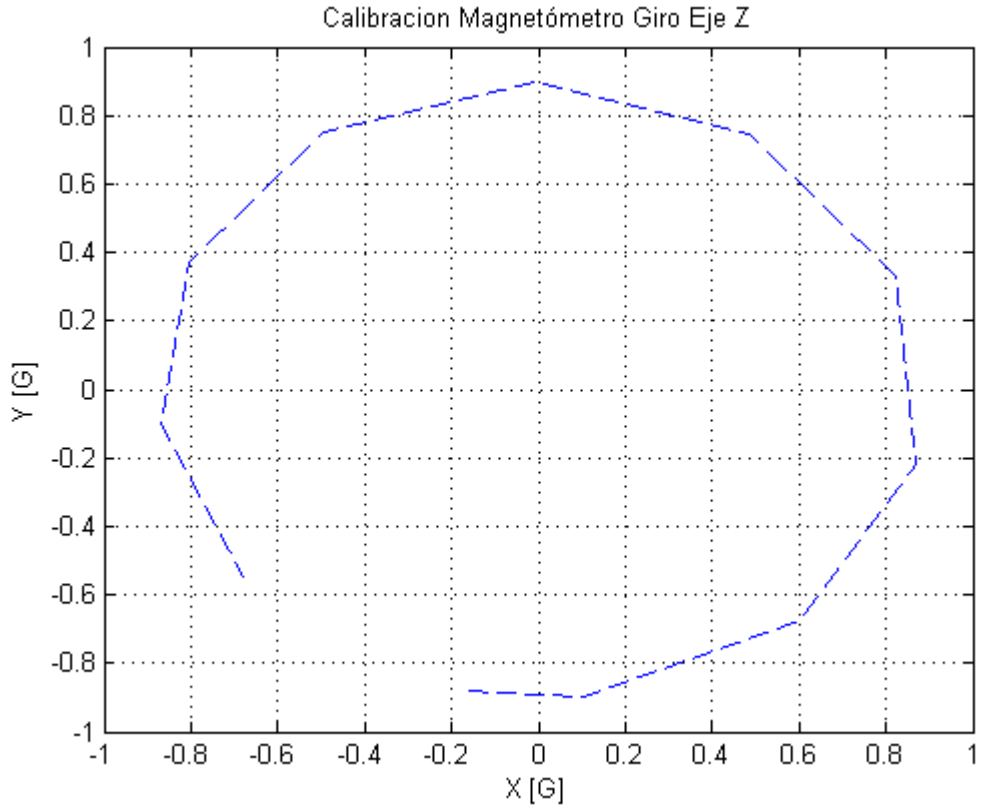
Tabla 2. Offset ejes magnetómetro

Eje	Offset
X	-0.5829
Y	-0.1174
Z	0.2044

Fuente propia.

Los valores en cuestión son sumados o restados si es negativo o positivo respectivamente al realizar el paso de calibración. Los resultados fueron los siguientes:

Figura 8. Calibración magnetómetro



Fuente propia.

Cabe resaltar que si se cambia el ambiente en el cual se hace la calibración, se debe proceder a recalibrar el circuito procurando mantener las condiciones iniciales de calibración.

3.1.5 FILTRO KALMAN

El filtro de Kalman es un algoritmo que fue desarrollado por Rudolf E. Kalman en 1960 y describe una solución recursiva para problemas de filtrado de datos discretos [17]. Desde su publicación, este algoritmo ha sido objeto de un exhaustivo estudio debido a las enormes aplicaciones que puede ofrecer especialmente en sistemas de navegación autónomos o asistidos.

El filtro de Kalman es un estimador óptimo que puede implementarse de manera sencilla en sistemas de carácter recursivo. Se denomina óptimo por que minimiza un criterio determinado y porque incorpora toda la información que se le suministra para determinar el filtrado; es decir, logra estimar el valor actual de las variables de interés. Con el fin de llevar a cabo este algoritmo es necesario:

a) Conocimiento del sistema en el cual va a ser implementado el filtro, así como también mediciones dinámicas provenientes de los dispositivos a utilizarse.
b) La descripción estática del ruido en el sistema, la información acerca del error, la incertidumbre en el modelo.
c) Conocer las condiciones iniciales de las variables más importantes presentes en el modelo

La palabra recursivo está relacionada con un algoritmo en el cual el filtro no tiene la necesidad de mantener almacenados todos los datos anteriores de las variables, por lo tanto no tiene que reprocesarlos cada vez que una nueva medida es tomada, en vez de esto hace uso de estados anteriores, es decir del último valor calculado, a partir del cual y junto con las nuevas medidas tomadas, el algoritmo entrega nuevos resultados que posteriormente serán considerados como estados anteriores. Con lo cual la implementación de este filtro resulta muy útil en sistemas de tiempo real.

Estas características hacen del filtro Kalman una poderosa herramienta para la estimación de estados pasados, presentes y futuros, e inclusive puede trabajar en circunstancias en las que la naturaleza precisa del modelo del sistema resulte desconocida.

3.1.5.1 FILTRO DISCRETO DE KALMAN

El filtro discreto de Kalman es un algoritmo mediante el cual se realiza un proceso de predicción y otro de corrección mediante la medición y observación de un grupo de variables presente en el sistema a tratar, dicho conjunto de variables forman el vector de estados y el observador. Estas variables se encuentran representadas en las ecuaciones del sistema dinámico del fenómeno físico en estudio, que incluyen el efecto del ruido presente en las observaciones así como en la incertidumbre de la dinámica del sistema.

Con el fin de implementar el filtro de Kalman para remover el ruido presente en la señal que se está midiendo en un proceso, se debe tomar en cuenta que la dinámica del sistema se tiene que presentar como un modelo de tipo lineal. Si el sistema no es lineal se utilizaría una versión del filtro Kalman conocida como FKE (Filtro de Kalman Extendido) o FKU.

Como se mencionó el filtro de Kalman usa un algoritmo de carácter recursivo con el fin de obtener una variable estimada que se acerque lo más posible a la realidad.

3.1.5.2 MODELO DEL SISTEMA ASUMIENTO UN MOVIMIENTO QUE ORIGINA UN SISTEMA LINEAL

El sistema dinámico en el cual se desea aplicar el filtro de Kalman debe expresarse como un modelo de carácter lineal en el cual se trata de estimar el estado $\boldsymbol{x} \in R^n$, dicho sistema se lo puede describir como:

$$x_k = Ax_{k-1} + Bu_k + w_{k-1} \quad (21)$$

Con una medición $z \in R^m$

$$z_k = Hx_k + v_k \quad (22)$$

donde:

x: estado del sistema

k*:* índice que indica un instante de tiempo, $k \in Z$.

z: valor observado

u: entrada del sistema

v: ruido en la medición

w: ruido en el proceso

A, B, H: matrices determinísticas que definen la dinámica del sistema.

La matriz *A* de dimensiones *nxn* en la ecuación (21) describe la relación que existe entre el estado en el momento *k-1* con el estado actual del instante *k*. la matriz *B* de dimensiones *m x l* relaciona la entrada de control $u \in R^l$con el estado *x.* La matriz *H* de dimensiones *mxn* presente en la ecuación de medición (11) relaciona el estado *x* con la observación z_k.

Los vectores *w* y *v* son independientes el uno del otro y representan el ruido gaussiano blanco con media cero presente en el proceso y en las observaciones respectivamente.

Los vectores *w* y *v* además traen consigo asociadas las matrices de covarianza *Q* y *R* que en general son diagonales, pudiendo también no serlo.

$$p(w) \sim N(0, Q) \quad (23)$$

$$p(v) \sim N(0, R) \quad (24)$$

En la práctica la matriz de covarianza de la perturbación del proceso $Q \in \mathbb{R}^{mxn}, R \in \mathbb{R}^{mxn}$ y la matriz de covarianza de la perturbación de la observación *R*, pueden

variar en el tiempo pero en general y por facilidad también pueden ser consideradas como constantes.

3.1.5.3 ALGORITMO

Son dos grupos de ecuaciones las que llevan al filtro de Kalman a converger a una solución óptima del proceso en observación: las primeras son las ecuaciones que se actualizan en el tiempo o también llamadas de predicción y el segundo grupo se refiere a las ecuaciones de actualización mediante observaciones, conocidas también de corrección.

Las ecuaciones de predicción hacen la estimación a priori de la matriz de la covarianza del error, así como del estado actual en el tiempo *k*, teniendo en cuenta el estado anterior en el tiempo *k-1*, como se observa en (25) y (26). Las ecuaciones de corrección en cambio tienen como objetivo realizar una retroalimentación, es decir, la incorporación de nuevas mediciones en la estimación a priori del estado, con el fin de corregir el error en una estimación posteriori.

$$\hat{x}_k^- = A\hat{x}_{k-1}^- + Bu_k \quad (25)$$

$$P_k^- = AP_{k-1}^- A^T + Q \quad (26)$$

El procedimiento para la etapa de corrección es realizar el cálculo de la ganancia de Kalman, K_k. Dicho factor de ponderación cumple el objetivo de minimizar la covarianza del error de la nueva estimación del estado. La siguiente tarea es tomar una medición del proceso en el momento *k* y con esto se obtiene el valor de la variable z_k, esta medida actualizada del proceso permite obtener una estimación a posteriori del estado, es decir, se consigue mejorar la estimación a priori conseguida en la etapa de predicción. Como tarea final en las ecuaciones de corrección se procede al cálculo de la estimación a posteriori de la matriz de covarianza del error mediante (29).

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1} \quad (27)$$

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-) \quad (28)$$

$$P_k = (I - K_k H)P_k^- \quad (29)$$

Después de cada par de actualizaciones, se puede apreciar claramente como una variable estimada con anterioridad puede ser mejorada por medio de una observación, y al mismo tiempo este nuevo estado estimado es usado para dar inicio una vez más a la etapa de predicción, y de esta manera el filtro de Kalman cumple con su característica claramente identificada de recursividad.

Se presenta a continuación el algoritmo del filtro de Kalman:

Predicción:

Paso 1: estimaciones iniciales para $\hat{x}_{k-1}\ y\ P_{k-1}$.

Paso 2: predicción de la covarianza del error. $P_k^- = AP_{k-1}A^T + Q$.

Paso 3: predicción del estado. $\hat{x}_k^- = A\hat{x}_{k-1}^- + Bu_k$.

Corrección:

Paso 4: Cálculo de la ganancia de Kalman. $K_k = P_k^- H^T (HP_k^- H^T + R)^{-1}$.

Paso 5: Actualiza la estimación con medida z_k. $\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-)$.

Paso 6: Actualiza la covarianza del error. $P_k = (I - K_k H)P_k^-$.

Paso 7: regresa al paso 2.

En el caso de la matriz de la covarianza, la estimación inicial con la cual dará inicio el filtro de Kalman puede ser la matriz identidad, mientras que en el caso de la estimación inicial para el estado, podría asignarse un valor inicial para disminuir el error inicial.

3.1.5.4 PARÁMETROS DEL FILTRO Y SINTONIZACIÓN

Como se observó en el grupo de ecuaciones que forman el filtro de Kalman existen dos parámetros que son de mucha importancia, ya que representan las matrices de covarianza de las perturbaciones en el proceso y en las observaciones, *Q* y *R,* respectivamente.

En el caso del parámetro *R*, este puede ser obtenido vía off-line (fuera del sistema), es decir, antes de ser implementado en el filtro, con lo cual se podría encontrar su valor de manera práctica mediante toma de muestras de mediciones y así hallar la varianza en el ruido o perturbación presente en las observaciones. Cabe indicar también que si bien el parámetro *R* puede ser calculado experimentalmente, también podría asignarse por ensayo y error dependiendo de cómo responda el sistema.

Sea cual fuese la manera como se decida establecer el parámetro *R* en el filtro de Kalman, siempre hay la posibilidad de manipularlo para obtener un valor ideal para el proceso, para esto se siguen ciertas pautas:

- Si las perturbaciones en las observaciones son grandes, entonces *R* será grande, por tanto si se observa las ecuaciones de la ganancia de Kalman se tiene que *K* será pequeña, lo cual significa que no se dará mucha credibilidad a la medición (observación) en el momento del cálculo del siguiente $\hat{x}_k$.

- Por otra parte, si en el momento de manipular la variable *R,* a esta se le da un valor pequeño, es decir, se considera que las perturbaciones en la observación son pequeñas, la ganancia de Kalman *K* será grande y se estará dando mayor peso o credibilidad a las mediciones (observaciones) cuando se calcule el siguiente $\hat{x}_k$.

La determinación del parámetro *Q* es un poco más complicada, esto debido a que es la representación de ruido en el proceso y por lo general esto es muy difícil de conocer, ya que se necesitará observar de manera directa el proceso que se está estimando.

En ciertas ocasiones, procesos relativamente sencillos pueden dar buenos resultados tan solo con la inyección de suficiente incertidumbre en su modelo por medio de la selección del parámetro *Q*, y de esta manera se esperaría que los datos arrojados por el proceso sean confiables.

Por otra parte si los parámetros *Q* y *R* son considerados constantes y los valores que se les fueron asignados son los correctos, lo que se obtendrá finalmente como resultado es que tanto la matriz de covarianza del error P_k así como la ganancia de Kalman K_k se estabilizarán rápidamente y llegarán incluso a permanecer constantes en el proceso.

3.1.5.5 FUSIÓN DE SENSORES MEDIANTE EL FILTRO DE KALMAN

La gran ventaja de aplicar el filtro de Kalman en la actualidad es en sistemas de navegación disponiendo de dispositivos como acelerómetros, giroscopios, magnetómetros, etc., donde cada uno agrega una incertidumbre al sistema de adquisición siendo o no confiable en ciertos aspectos.

El filtro de Kalman lo que hace es utilizar las mejores características de cada uno de estos sensores para obtener una información veraz y utilizable para los sistemas de navegación.

En el caso de los sensores mencionados anteriormente, se puede fusionar los datos de desplazamiento angular del giroscopio en el eje Z junto con la medición del desplazamiento angular arrojado por el magnetómetro. Si éste último se encuentra en un entorno donde el campo magnético es constante su medida será mucho más confiable que la arrojada por el giroscopio; por lo tanto, para el filtro Kalman tendrá mucha más relevancia la medida arrojada por el magnetómetro.

En primer lugar, se determina tomar las medidas realizadas por el giroscopio durante cortos instantes de tiempo mientras sea confiable; posteriormente, se implementaría la corrección en la medida usando el magnetómetro.

En la tabla 3, se hace una comparativa de los diferentes resultados obtenidos implementando el filtro Kalman tomando un tamaño de 10 muestras por cada ángulo programado.

Tabla 3. Comparación de medidas realizadas

Tiempo [s]	Desp. angular ingresado [°]	Desp. angular real [°]	Desp. angular medido con giroscopio [°]	Desp. angular medido con magnetómetro[°]	Desp. angular con Kalman [°]
4	90.00	93.00	87.10	91.20	91.10
4	100.00	99.00	97.30	98.20	97.75
8	180.00	180.00	172.00	177.00	177.00
16	360.00	365.00	370.00	357.00	363.50

Fuente propia.

3.1.6 MEMORIA EEPROM

Para disminuir el costo computacional del control y envío del valor de voltaje a los motores se añade una memoria EEPROM al sistema, que es un tipo de memoria ROM no volátil, o sea puede ser programada, borrada y reprogramada eléctricamente. En esta memoria se almacena el valor de voltaje previamente calculado por los algoritmos de optimización, para luego ser leído por el microcontrolador el cual modifica el valor enviado al PWM en cada instante de tiempo. Además de la reducción en costo computacional, guardar los datos en la memoria nos brinda la ventaja de poder controlar el tiempo exacto en el que se modifican los valores del PWM haciendo que la señal enviada a los motores sea lo más parecida a la señal idealmente calculada.

El chip usado es el 24LC256 de Microchip, con capacidad de 32Kbytes de memoria suficiente para almacenar los datos para el recorrido completo.

3.1.7 COMUNICACIÓN I^2C

I^2C es un bus de comunicación en serie, usado principalmente para comunicar circuitos integrados entre sí, que serían los microcontroladores y sus periféricos que residen en un mismo circuito impreso con diferente direccionamiento. Este protocolo de comunicación tiene varias ventajas, entre las que está el uso de pocas líneas de transmisión, la línea de reloj (SCL) y la de datos (SDK), para varios dispositivos y comunicándose con estos a través de códigos de dirección

programados por software o hardware. En este proyecto se usan dos periféricos que se comunican a través de I^2C, la IMU y la memoria EEPROM.

La tarjeta de medida inercial cuenta con 3 dispositivos que cuentan cada uno con su propia dirección como se indica en la tabla III.

Tabla 4. Direccionamiento I^2C

Dispositivo	Dirección Hexadecimal	
	Escritura	Lectura
Acelerómetro	0XA6	0XA7
Giroscopio	0XD0	0XD1
Magnetómetro	0X3C	0X3D
Memoria ROM	0X50	0X50

Fuente propia.

3.1.8 TRANSMISIÓN INALÁMBRICA

Los módulos Zigbee ofrecen una gran facilidad para la medida y control de variables de manera remota. Su tamaño y rápida configuración hacen de este dispositivo una herramienta efectiva en la transmisión de datos, pudiendo configurar parámetros como: velocidad de transmisión e identificador de red personal para el correcto enlace con la estación base.

El módulo Zigbee cuenta también con conversores análogo – digital de 10 bits, a través de los cuales se puede efectuar la adquisición de variables; sin embargo, en este trabajo se requiere además de la transmisión de datos de los sensores, la transmisión de datos desde un tarjeta integrada GY85 para poder realizar la fusión de sensores con filtros Kalman; por tal razón, se utiliza un microcontrolador que efectúe tanto la conversión análoga-digital como la gestión de datos para el correcto funcionamiento de la tarjeta de adquisición.

En el sistema se determina una velocidad de 9600 baudios, suficiente para los requisitos de comunicación definiendo una red punto a punto de comunicación bidireccional.

4 DISEÑO ELECTRÓNICO

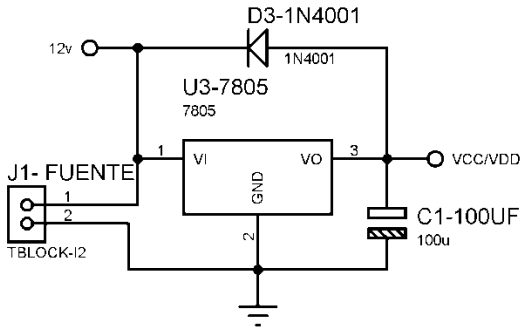
4.1 FUENTE DE ALIMENTACIÓN

Dado que el valor de referencia del conversor análogo digital debe ser una constante mientras se efectúa la medición del voltaje obtenido en los motores y la misma fuente, para la energización de la tarjeta de adquisición, se utiliza una batería de tecnología tipo LiPO con capacidad de 2650mAh y voltaje nominal de 11.1v distribuido en 3 celdas; donde a pesar del consumo, su capacidad se mantiene por un tiempo más prolongado que el tiempo establecido por una batería de uso comercial.

4.1.1 FUENTE DE ALIMENTACIÓN 5V

Para proporcionar un voltaje de alimentación de 5v requerido por dispositivos como el microcontrolador, la tarjeta GY85, etc. Se implementa un regulador LM 7805, el cual cuenta con una salida máxima de corriente de 1.5A y es de fácil implementación. En la figura 3 se muestra el esquema implementado, el cual es recomendado por el fabricante.

Figura 9. Fuente de alimentación 5v.



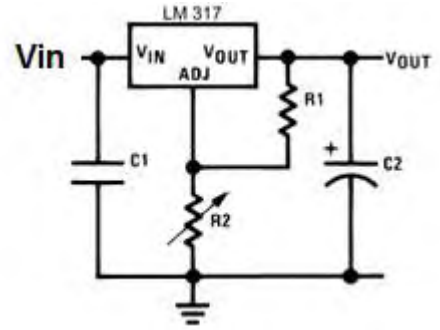
Fuente propia.

4.1.2 FUENTE DE ALIMENTACIÓN 3.3V

Dentro de los dispositivos utilizados en el diseño de la parte electrónica, existen elementos con voltajes de alimentación requerido a 3.3v por el fabricante. Para tal fin, se implementa un regulador LM317 que puede ser configurable a cualquier

voltaje requerido, en este caso, entre 0 a 12v dado el voltaje nominal máximo de la batería. Dicho dispositivo, se configura a través de un divisor de tensión realizado por dos resistencias externas calculadas según (30) donde R_1 y R_2 son resistencias programables a través de las cuales se configura el voltaje de salida.

Figura 10. Fuente de alimentación de 3.3v



Fuente propia

$$V_{out} = V_{ref}\left(1 + \frac{R_2}{R_1}\right) + I_{adj} * R_2 \qquad \textbf{(30)}$$

Donde:

$V_{ref} = 1.25$v.

I_{adj} = 100uA.

Se determina un valor comercial para una de las resistencias y se procede a obtener el valor ideal de la otra. En este caso, se toma el valor de $R_1 = 240\Omega$ y R_2 = 390Ω obteniendo un voltaje de salida de 3.3 V requerido para el correcto funcionamiento del módulo XBee.

Se utiliza la misma configuración del LM317 para enviar los datos desde el microcontrolador hacia el módulo inalámbrico y evitar superar el límite máximo de voltaje en el XBee.

4.2 MICROCONTROLADOR 18F2550

En este proyecto se utilizará el microcontrolador 18F2550, el cual se encarga de realizar el censado de los voltajes de los motores y la fuente de alimentación a través de un conversor análogo digital de 10 bits. Recibe la información proporcionada a través de la interfaz I^2C proveniente desde el acelerómetro, giroscopio, magnetómetro y la memoria EEPROM para ser reenviados a la PC y debidamente procesados.

Con la información de los voltajes óptimos, el microcontrolador envía la señal a cada motor a través del módulo PWM, el cual junto con el driver L293B, proporciona la diferencia de potencial correcta a aplicar en cada motor para trazar la trayectoria establecida en el software.

4.2.1 BREVE DESCRIPCIÓN

Este microcontrolador ofrece las ventajas de todos los dispositivos de la familia PIC18 como: alto rendimiento computacional y un precio económico con larga durabilidad, memoria flash asegurada, 16384 instrucciones y una memoria de datos de 2048 bytes. Esta familia incluye características que permiten un muy alto rendimiento y bajo consumo de potencia.

4.2.2 CONVERSOR A/D

Siendo el módulo utilizado de 10 bits, proporciona una resolución que equivale a (31)

$$r = \frac{V_{in}}{2^N - 1} \qquad (31)$$

Donde:

r: Resolución.

V_{in}: tensión de entrada [v].

N: número de bits del conversor.

Según las características mencionadas del microcontrolador, obtenemos una resolución equivalente a 4,8 mV/LSB.

La resolución cambia si se modifica la tensión de fondo de escala, es decir, la tensión de referencia. Los microcontroladores permiten cambiar la tensión de referencia en un valor absoluto de 0 a V_{ref} o en un margen de $-V_{ref}$ a $+V_{ref}$ donde las tensiones a convertir siempre son positivas.

4.2.3 MÓDULO CCP

Se cuenta con dos módulos CCP (Capture/Compare/PWM). Cada módulo contiene un registro de 16 bits, que puede operar como un registro de captura de 16 bits, y un registro de comparación, también de 16 bits o un registro de ciclo de trabajo maestro esclavo PWM o modulación por ancho de pulso.

En el modo PWM, se puede obtener en los pines CCPx una señal periódica en la que se puede modificar su ciclo de trabajo. Es decir, puede variarse el tiempo en el cual la señal está a nivel alto frente al tiempo que está a nivel bajo. De esta forma, la tensión media aplicada a la carga es proporcional al tiempo en alto, controlando, por ejemplo, la velocidad de motores, luminosidad de lámparas, etc.

El microcontrolador cuenta con 3 timer para las diferentes aplicaciones; para la implementación en cuestión se utiliza el timer 2, el cual corresponde al correcto funcionamiento del PWM, configurando dichos pines como salida. La configuración de dicho timer, proporciona el periodo de la señal requerido según (21) en [16].

$$PWMT = (PR2 + 1) * 4 * T_{osc} * T2DIV \quad \textbf{(32)}$$

Para obtener la frecuencia:

$$F_{PWM} = \frac{F_{OSC}}{(PR2 + 1) * 4 * T2DIV} \quad \textbf{(33)}$$

Despejando *PR2*

$$PR2 = \frac{F_{OSC}}{F_{PWM} * 4 * T2DIV} - 1 \quad \textbf{(34)}$$

donde:

PR2 : contenido del registro *PR2* que determina el periodo de la señal en un entero de 0 - 255.

T2DIV: divisor de frecuencia del timer 2.

F_{OSC} : frecuencia del oscilador.

Se establece una frecuencia aleatoria para el funcionamiento de los motores. Tomando una señal de 2khz en la salida de cada módulo, un oscilador de 20Mhz y el divisor *T2DIV* igual a 16, para el PWM se debe configurar el registro *PR2* en 156.

4.2.4 VALOR PROMEDIO

Para una función periódica de periodo *T*, es la media algebraica de los valores instantáneos durante ese mismo intervalo de tiempo y se define matemáticamente así:

$$V_{prom} = \frac{1}{T}\int_{<T>} f(t)dt \quad \textbf{(35)}$$

donde:

T : es el periodo de la señal.

f(t): la función aplicada a la carga; en este caso, una onda cuadrada.

4.2.5 CICLO DE TRABAJO

Es el porcentaje de una señal durante un ciclo de periodo que permanece en alto o un valor diferente de cero y positivo en términos de este trabajo. Para el microcontrolador se define como:

$$\tau = CT * T_{OSC} * T2DIV \quad (36)$$

donde:

CT : ciclo de trabajo contemplado en el rango de 0 a 1024 para 10 bits.

T_{OSC}: periodo de oscilación.

Para obtener el valor *CT* de cada uno de los valores obtenidos a aplicar en cada motor, se resuelve (35) para una señal de onda cuadrada con amplitud de V_p obteniendo (37).

$$V_{prom} = \frac{V_p * \tau}{T} \quad (37)$$

donde:

V_p: voltaje pico de la señal aplicada a los motores.

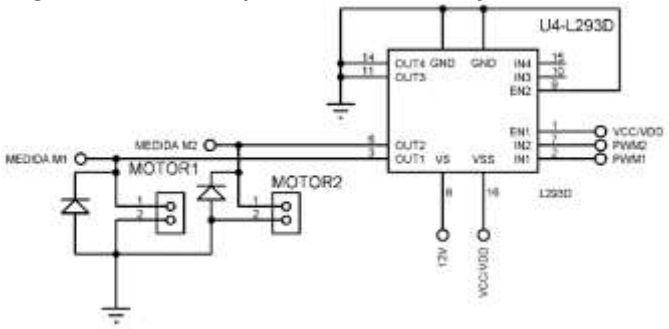
Despejando *CT* de (36) y τ de (37) se obtiene finalmente la expresión para el valor entero del ciclo de trabajo.

$$CT = \frac{V_{prom} * F_{OSC}}{V_p * T2DIV * F_{señal}} \quad (38)$$

4.3 DRIVER L293B

Para controlar la velocidad de los motores se requiere de una etapa de potencia entre las salidas de los PWM y la activación de los motores. Para ello, el L293B cuenta con cuatro drivers de pull-up capaces de manejar una corriente de 1A. Cada canal es controlado por una entrada lógica TTL compatible con las salidas del microcontrolador. Cuenta con dos fuentes de voltaje una para la lógica y otra para la puesta en marcha de los motores evitando la caída de tensión en el accionamiento. El montaje se realiza como se muestra en la figura 11.

Figura 11. Esquema del montaje del L293B con los motores.

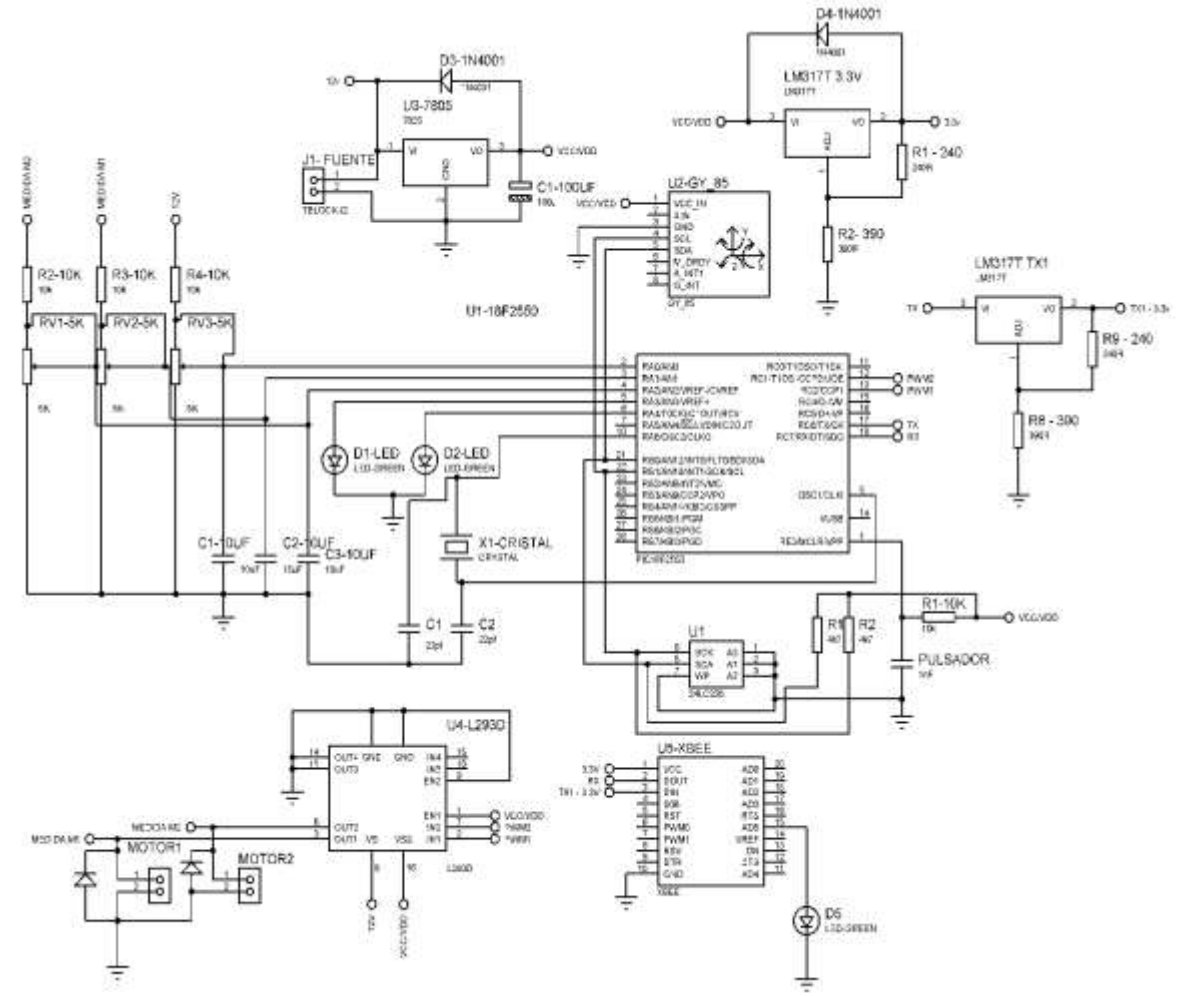


Fuente propia

4.4 DISEÑO DE LA TARJETA DE ADQUISICIÓN

El diseño de la tarjeta de adquisición se realizó en el programa ARES, para el diagrama de conexiones del impreso; y PROTEUS, para la representación esquemática de cada uno de los dispositivos, como se muestra en las Figuras 11, y 12.

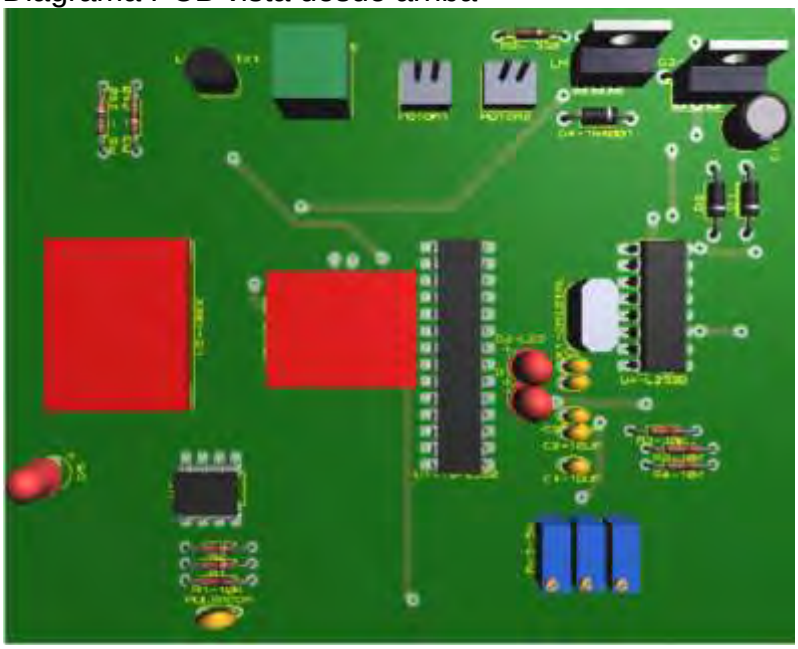
Figura 12. Diagrama de conexiones de la tarjeta de adquisición



Fuente propia

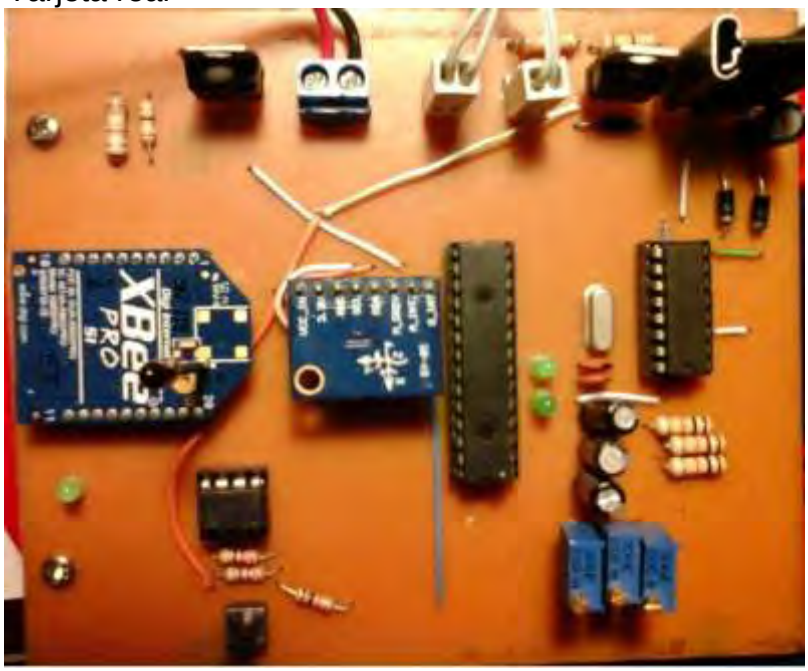
.

Figura 13. Diagrama PCB vista desde arriba



Fuente propia

Figura 14. Tarjeta real



Fuente propia

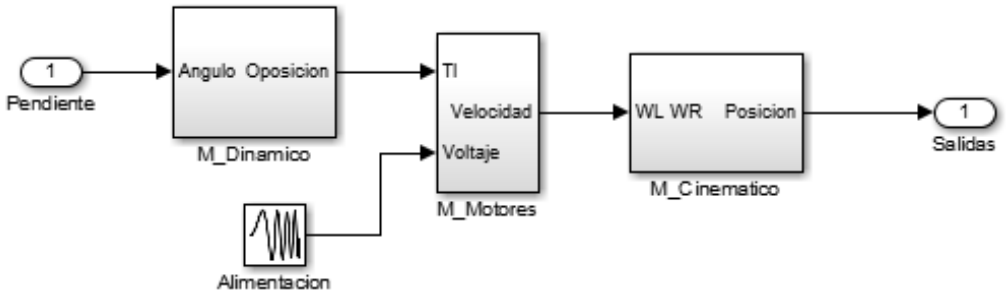
5 MODELAMIENTO MATEMÁTICO

El prototipo usado para el desarrollo de este proyecto es un robot móvil de dos ruedas (wheeled mobile robot WMR) con tracción diferencial. La posición, velocidad y orientación en cada momento está determinado por la velocidad desarrollada por cada una de las ruedas del robot.

Para el análisis del comportamiento del prototipo se desarrolló un modelo matemático que lo describe de manera aproximada, teniendo en cuenta los factores más importantes que influencian el comportamiento de cada una de las variables del sistema, priorizando la precisión en los resultados y la simplicidad del mismo. De esta forma tenemos que el modelo matemático del WMR consta principalmente de 3 partes:

- Modelo dinámico
- Modelo cinemático
- Modelo de los motores y transmisión

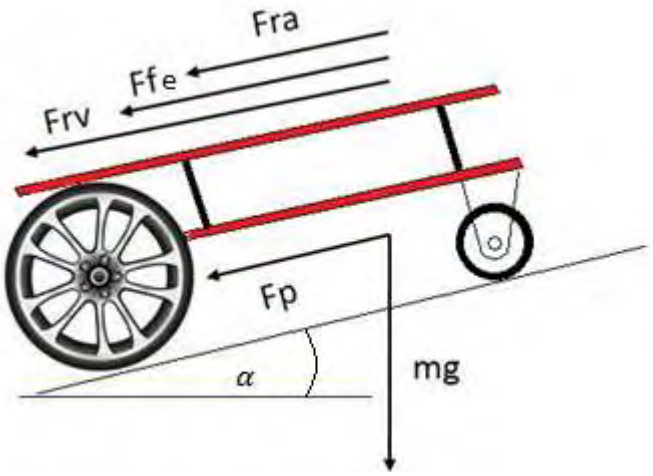
Figura 15. Diagrama de bloques general del prototipo



Fuente propia.

5.1 MODELAMIENTO DINÁMICO

Figura 16. Fuerzas involucradas en la dinámica del VE



Fuente propia

En un vehículo en movimiento principalmente se estudian dos tipos de dinámica que son: dinámica longitudinal y dinámica lateral. La primera hace referencia al estudio de las fuerzas que actúan sobre un vehículo al tomar una curva y la segunda al estudio de las fuerzas que actúan en el vehículo en una recta o curva de gran radio.

Este proyecto estudia únicamente la dinámica longitudinal, pues al ser un pequeño prototipo las velocidades desarrolladas son bajas y bajo esta condición la dinámica lateral tiene poca influencia en el comportamiento dinámico del prototipo.

Para el análisis dinámico longitudinal se aplican leyes de Newton sobre el prototipo, y seleccionamos cuales son las fuerzas que más influencia tienen en el comportamiento dinámico.

- Fuerza de fricción estática (F_{fe})
- Fuerza de rozamiento viscoso (F_{rv})
- Fuerza de rozamiento aerodinámico (F_{ra})
- Peso dependiente de la pendiente (F_p)

Estas son las fuerzas que actúan en el vehículo en sentido opuesto al del movimiento. Las ecuaciones que describen a cada una de estas fuerzas son las siguientes:

$$F_{fe} = m * g * \mu_e$$
$$F_{rv} = k_s * v$$
$$F_{ra} = \frac{\rho * c_d * A_f * v^2}{2} \quad (39)$$
$$F_p = m * g * \sin(\alpha)$$

donde:

m : masa del vehículo [kg]

g : aceleración de la gravedad [m/s^2]

μ_e : coeficiente de fricción estática

k_s : coeficiente de Stokes [m^2/s]

ρ : densidad de fluido [kg/m^3]

A_f : área frontal del vehículo [m^2]

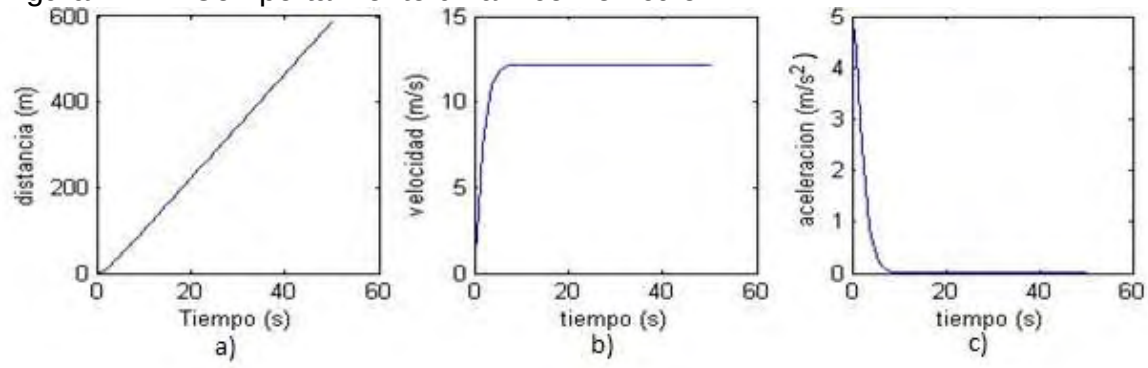
v : velocidad del vehículo [m/s]

c_d : coeficiente de resistencia especifica del vehículo

α :angulo de la pendiente [rad]

Una vez identificadas estas fuerzas, se realiza la suma de estas y se simulan el sistema previamente descrito, en Simulink usando como entrada una fuerza de tracción constante m*a, donde la aceleración suministrada es 5 $\mathrm{m/s^2}$

Figura 17. Comportamiento dinámico Vehículo



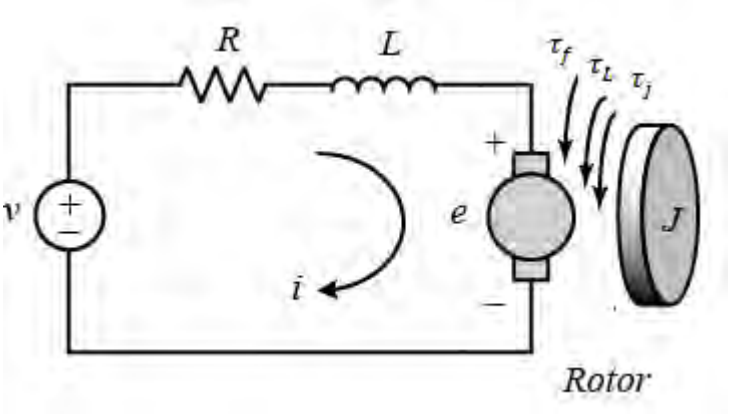
Fuente propia

Se observa que el objeto parte con la aceleración inicial de 5 $\mathrm{m/s^2}$ y está decrece con el tiempo hasta que se estabiliza en 0, la velocidad por el contrario aumenta hasta alcanzar una velocidad limite que depende tanto de las fuerzas de oposición como de la de empuje.

5.2 MODELAMIENTO MOTOR DC

5.2.1 MODELADO Y ANÁLISIS DEL MOTOR

Figura 18. Diagrama eléctrico y mecánico del motor



Fuente: http://ctms.engin.umich.edu

Para generar el empuje necesario para mover el prototipo, se usan dos motores conectados a las ruedas del WRM a través de una caja de engranajes, que reduce la velocidad angular y aumenta el torque. El motor usado es alimentado por una fuente de corriente directa (DC) y es una máquina que convierte la energía eléctrica en energía mecánica gracias a la interacción de campos magnéticos, provocando un movimiento rotario. Entonces para su modelamiento se tiene en cuenta el análisis de las dos partes: la mecánica y la eléctrica.

5.2.1.1 ECUACION ELÉCTRICA:

Al analizar con leyes de Kirchhoff el circuito electrico presentado en la figura 18 obtenemos la siguiente ecuación:

$$v(t) = V_L(t) + V_R(t) + e_b(t)$$
$$v(t) = L\frac{di(t)}{dt} + Ri(t) + K\omega(t) \quad (40)$$

En donde:

L : inductancia del embobinado del motor [H]
R : resistencia del embobinado del motor [Ω]
K : constante de fuerza contra electromotriz [$V * \mathrm{s/rad}$]
ω : velocidad angular [$\mathrm{rad/s}$]
Con: $\omega(t) = \frac{d\theta(t)}{dt}$

5.2.1.2 ECUACION MECÁNICA:

Haciendo una análisis de fuerzas con leyes de Newton en la parte mecánica del motor de la figura 18 obtenemos la siguiente ecuación:

$$\tau_m(t) = \tau_j + \tau_f(t) + \tau_L(t)$$
$$Ki(t) = J\frac{d\omega(t)}{dt} + B\omega(t) + \tau_L \quad (41)$$

donde:

J : momento de inercia del rotor

B : constante de Friccion viscosa

τ_L : torque de oposicion visto desde el eje del rotor

Usando transformada de Laplace y operando las ecuaciones eléctrica y mecánica se obtiene la ecuación de tranferencia de este modelo matemático

$$G(s) = \frac{W(s)}{U(s)} = \frac{K}{(Ls+R)(Js+B)+K^2} \quad (42)$$

Se reorganiza la ecuación de transferencia:

$$G(s) = \frac{\left(\frac{1}{K}\right)}{t_m t_e s^2 + t_m s + \frac{LB}{K^2}s + \frac{RB}{K^2} + 1} \quad (43)$$

Con la ecuación reescrita de esta forma obtenemos una idea del comportamiento que se tiene el sistema dinamico.

Donde:

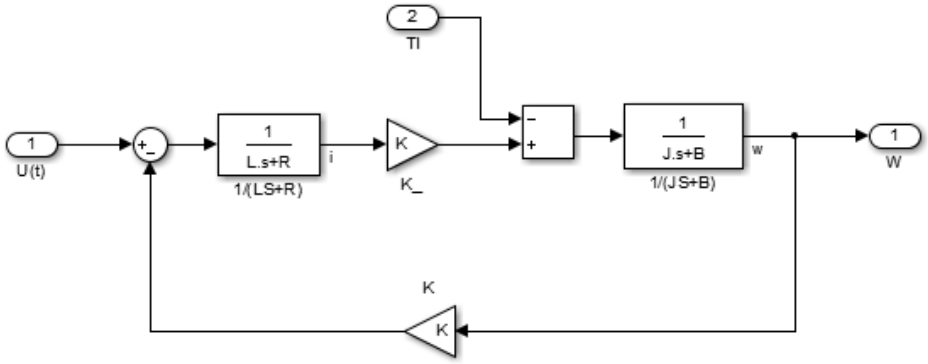
t_m: constante mecánica de tiempo [s]

t_e: constante eléctrica de tiempo [s]

Con: $t_m = \frac{RJ}{K^2}$ y $t_e = \frac{L}{R}$

Se usan las ecuaciones previamente planteadas y se implementan en simulink en forma de un modelo en diagrama de bloques, con el que se observa la respuesta y el comportamiento aproximado del motor DC.

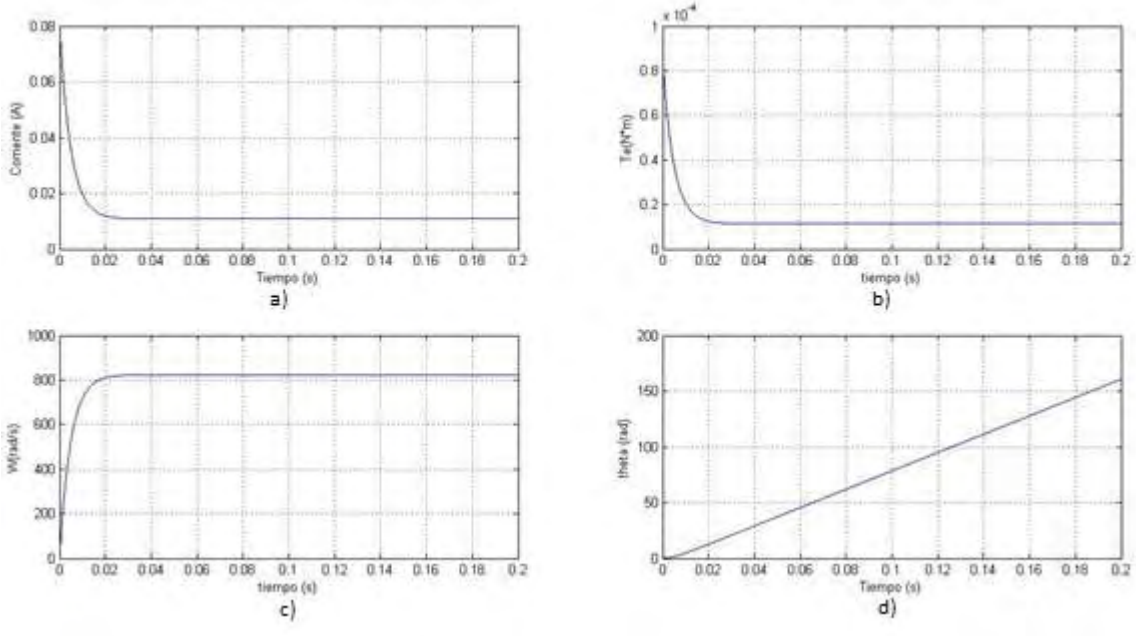
Figura 19. Diagrama de bloques Motor DC



Fuente propia.

A continuación se simula el modelo y se muestra la respuesta a una señal de entrada de tipo escalon con valor de 5V con el valor de las constantes obtenidas posteriormente. De esta forma se observa el comportamiento característico de las variables de estado del motor: corriente (i), torque eléctrico (T_e), velocidad angular (ω) y ángulo de giro theta(θ)

Figura 20. comportamiento modelo del motor DC

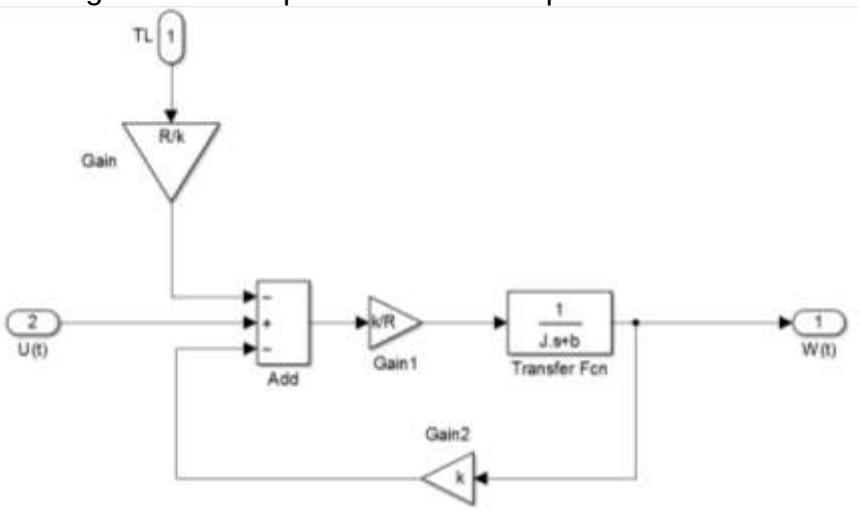


Fuente propia.

5.2.2 MODELO MOTOR DC SIMPLIFICADO

Se realiza una simplificación del modelo del motor, reduciendo su orden de segundo a primero.

Figura 21. Diagrama de bloques Motor DC simplificado



Fuente propia.

Para obtener la ecuación del modelo simplificado se expresa el diagrama de bloques en forma de ecuación y se escribe en forma de variables de estado:

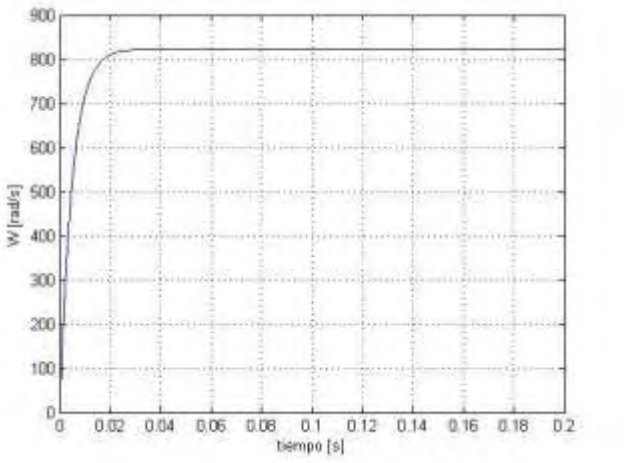
$$W(s) = \frac{K}{R(Js+B)}\left(U(s) - KW(s) - R\frac{\tau_l}{K}\right) \quad (44)$$

Despejando sW

$$sW = \frac{K}{RJ}U(s) - \frac{(K^2+RB)}{RJ}W(s) - \frac{1}{J}\tau_l \quad (45)$$

Se usa simulink para observar la respuesta al escalon unitario del modelo simplificado.

Figura 22. Respuesta al escalón modelo simplificado



Fuente propia.

Al comparar la respuesta del modelo simplificado en la variable de velocidad angular figura 22 con el modelo completo figura 20 c), se observa que la diferencia entre estos es mínima, demostrando que el modelo simplificado es una buena aproximación al modelo completo y que la respuesta final de este no va a diferir mucho de la del modelo completo, dándonos además la ventaja de simplicidad y una ganancia en el tiempo de ejecución al minimizar el coste computacional de las simulaciones.

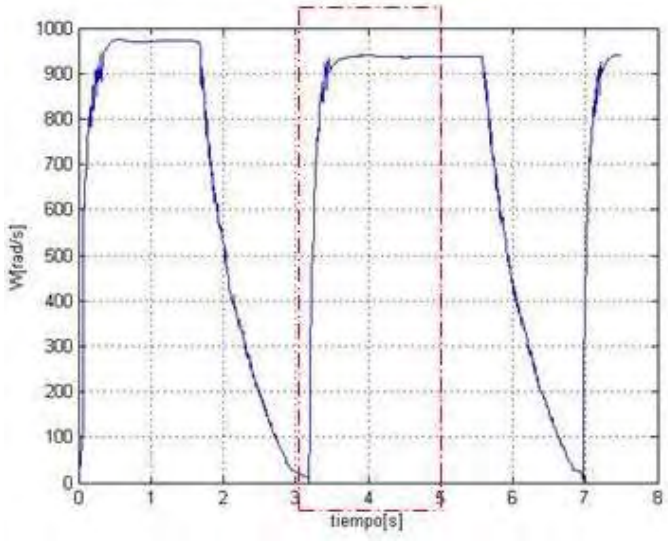
5.2.3 OBTENCIÓN EXPERIMENTAR DE LOS PARÁMETROS DEL MOTOR DC

Para obtener los parámetros del motor de una forma aproximada, se utilizan diferentes instrumentos de laboratorio, se tienen en cuenta las ecuaciones del motor y se diseñó e implemento un tacómetro de alta resolución.

El tacómetro fue diseñado a partir de un microcontrolador PIC 16f877A, un sensor de barrera y un encoder ranurado de fabricación manual. El código de funcionamiento se implementó en lenguaje C usando el compilador CCS y se detalla su funcionamiento en el anexo H, se estableció comunicación entre el computador y el PIC a través de comunicación serial, con un cable conversor de rs232 a usb. Los datos fueron observados y procesados a través de Matlab.

A continuación se muestra la respuesta del motor a la conexión y desconexión de una alimentación con valor de 7,5V.

Figura 23. Respuesta de conexión y desconexión del motor



Fuente propia.

Se realizan varias pruebas modificando el valor de voltaje de señal de entrada escalón. En cada prueba se observa de forma general el comportamiento típico del motor DC con un ruido introducido por el encoder, ruido que se presentó por que el encoder es de fabricación manual, donde la separación de los dientes no es perfecta. De las respuestas obtenidas se selecciona la respuesta al escalón con valor de 7,5 V que es donde menor ruido se observó; se toma una porción de los datos y se hace una regresión para encontrar la ecuación de la curva de tendencia usando la herramienta curve fitting de Matlab, aproximando el comportamiento a una ecuación exponencial de segundo grado de la siguiente forma.

$$f(t) = a * \exp(b * t) + c * \exp(d * t) \qquad (46)$$

donde.

a : 979.8,

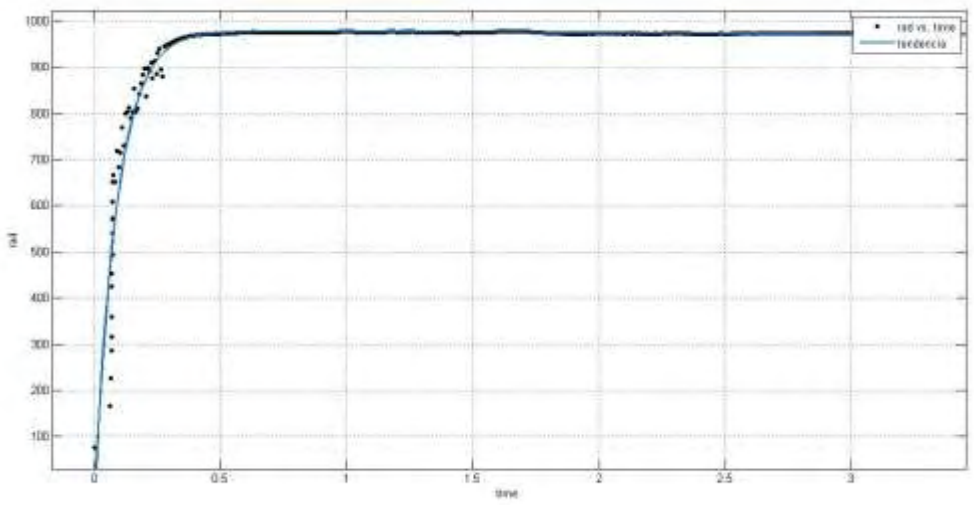
b : -0.002581,

c : -1053,

d : -11.25

La gráfica de los datos y su tendencia se muestran en la figura 24.

Figura 24. Tendencia de la respuesta al escalón



Fuente propia.

5.2.3.1 ECUACIÓN DE TRANSFERENCIA DE MODELO SIMPLIFICADO

Para encontrar el modelo simplificado, tenemos en cuenta la ecuación característica de este:

$$G(s) = \frac{k_p}{t_m * s + 1} \quad (47)$$

Donde: $k_p = \frac{\Delta W}{\Delta U}$ en estado estable y t_m es el tiempo de establesimiento del sistema.

Entonces $k_p = \frac{977,6}{7,5} = 130,3$

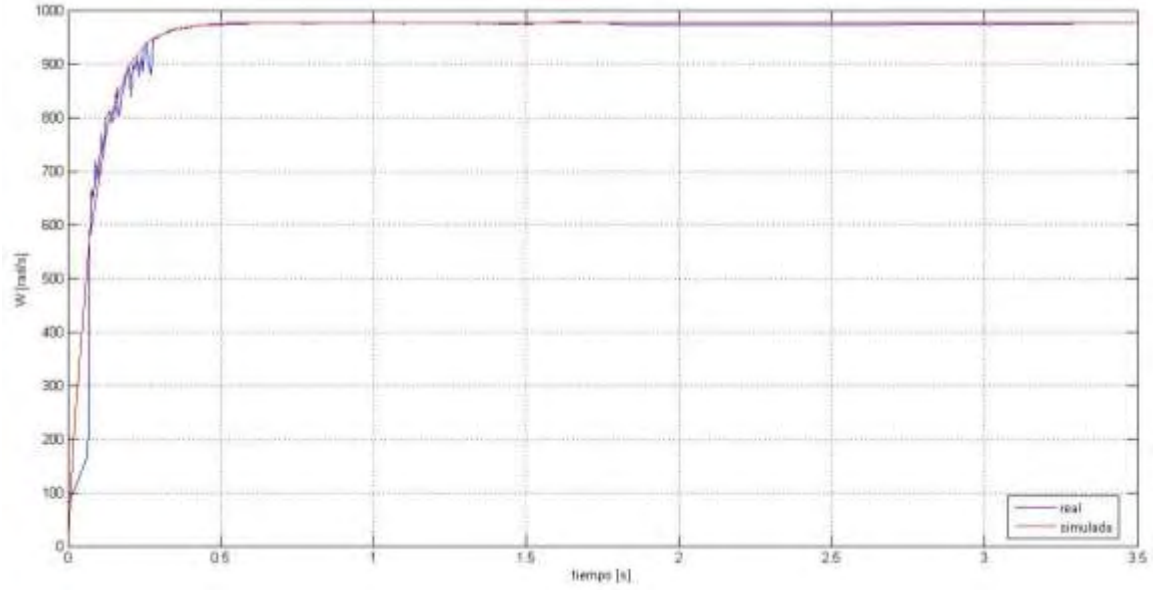
El 63,2% de la velocidad máxima es 617,8 y demora t_m =0,08 segundos en alcanzar este valor

Por tanto la ecuación de transferencia aproximada del modelo simplificado es:

$$\frac{W(s)}{U(s)} = G(s) = \frac{130.3}{0,08 * s + 1} \quad (48)$$

Se simula la ecuación de transferencia obtenida previamente y se la compara con los datos reales. La comparación se muestra en la Figura 25.

Figura 25. Comparación modelo simplificado vs real



Fuente propia

Escribiendo la ecuación de transferencia aproximada en forma de variables de estados tenemos que:

$$SW = \frac{130{,}3}{0{,}08}U(s) + \frac{1}{0{,}08}W(s) \qquad (49)$$

Comparando con la ecuación (45) $a = \frac{K}{R*J} = \frac{130.3}{0.08}$ y $b = \frac{(K^2+R*B)}{\mathrm{RJ}} = \frac{1}{0.08}$

Para encontrar los parámetros aproximados del motor, nos basamos en la respuesta al impulso y en las mediciones físicas del sistema.

5.2.3.2 OBTENCIÓN DE LA RESISTENCIA

El primer parámetro encontrado es R, el cual se mide poniendo el RLC entre los dos bornes del motor arrojando un resultado de R=8,85 Ω.

5.2.3.3 CALCULO DE CONSTANTE ELÉCTRICA (k_e) Y MECÁNICA (k_m)

Se usa el tacómetro y un amperímetro para medir la velocidad y la corriente de armadura del motor a diferentes voltajes. k_e se calcula teniendo en cuenta que $E_b = k_m w$. entonces, con el motor en estado estable, $k_e = \frac{v_{in} - R*i}{w} = k_m = K$ que anteriormente fue denominada únicamente K.

Tabla 5. Constante eléctrica y mecánica

Vin [v]	Eb [v]	I[A]	W[rad/s]	N[rpm]	K
3	2,6634	0,044	397	3791	0,00670882
4,5	4,00275	0,065	589	5624	0,00679584
6	5,2656	0,096	779	7438	0,00675944
7,5	6,5055	0,13	985	9406	0,00660457
9	7,6995	0,17	1122	10714	0,0068623
				Promedio	0,00674619

Fuente propia.

5.2.3.4 OBTENCIÓN DE LA INDUCTANCIA

Se mide la inductancia del motor con un dispositivo llamado LCR, el resultado fue de 3,03 mH

5.2.3.5 CÁLCULO DEL MOMENTO DE INERCIA

Para calcular la inercia J se usa el metodo paramétrico, que usa el conocimiento de los datos previamente obtenidos. Haciendo uso de la formula $t_m = \frac{J*R}{k_e * k_t}$, despejando y reemplazando las constantes obtenemos que $J = 4.76 * 10^{-7}\ \mathrm{kg.m^2}$

5.2.3.6 CÁLCULO DE LA FRICCIÓN VISCOSA

Se calcula el torque que produce el motor, por medio de la ecuación $T_m = K * i(t)$ y analizando la ecuación (41) en estado estable, de tal forma que tenemos que W(t) es constante por tanto la derivada de la velocidad angular se hace cero, y teniendo en cuenta que estamos analizando el motor sin carga $\tau_l = 0$ por lo que en estado estable:

$$\tau_m = Bw(t) \qquad (50)$$

De donde se despeja B, y sustituimos los valores por los medidos en cada uno de los casos mostrados a continuación.

Tabla 6. Calculo de fricción viscosa

Vin[v]	W[rad/s]	I[mA]	T_m[m·N]	B [x10^{-7}]
3	418,9	0,043	0,00029008	6,92476
4	558,8	0,0545	0,00036766	6,5794
5	678	0,0689	0,0004648	6,85545
6	819	0,0907	0,00061186	7,47084
7	941	0,1041	0,00070226	7,4629
9	1191	0,162	0,00109285	9,17592
			promedio	7,41154

Fuente propia.

5.2.3.7 CÁLCULO TORQUE DE FRICCIÓN ESTÁTICA

Se pone el motor en reposo con un voltaje de armadura en cero y se sube el voltaje hasta encontrar el punto donde el motor empieza a moverse; gracias a esto se obtiene el voltaje y corriente de arranque. El proceso se repite varias veces. El resultado se ve en la tabla V.

Tabla 7. Corriente y voltaje de arranque

Prueba	Varr [v]	Iarr[mA]
1	0.65	75
2	0.66	72
3	0.7	80
4	0.8	82
5	0.8	72
6	0.8	75
Promedio	0.735	76

Fuente propia.

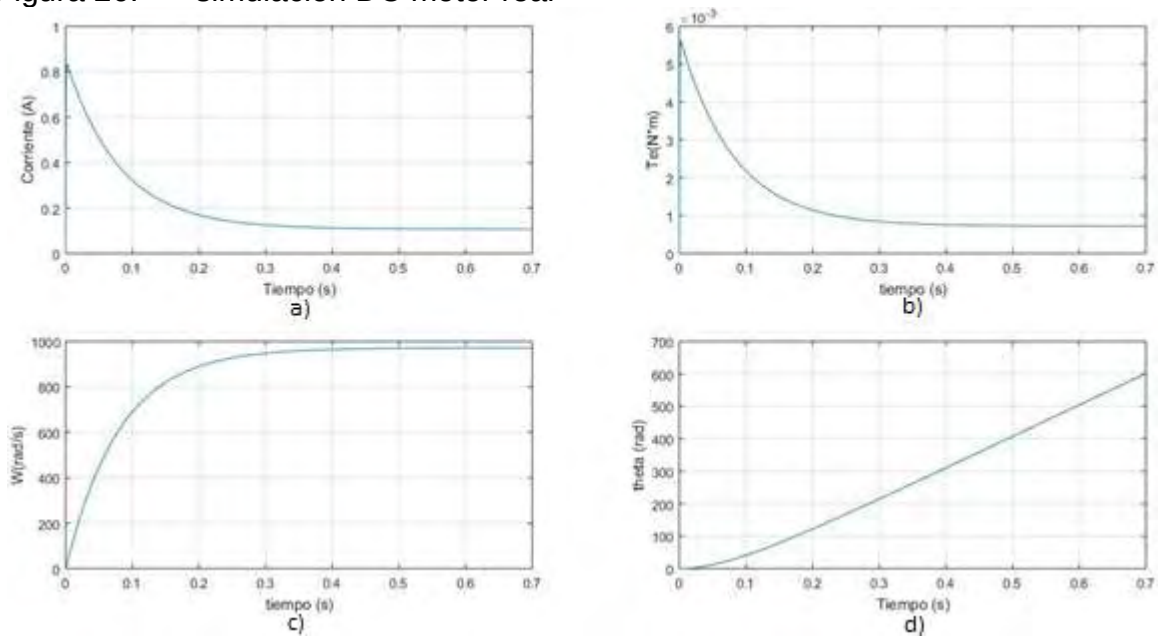
Gracias a la corriente de arranque se obtiene el torque de fricción Tf que es la fricción estática que debe vencer el motor para iniciar el movimiento.

$$T_f = K * I_{arr} = 512.7 * 10^{-6}[N.m] \quad (51)$$

También se observa que que el voltaje mínimo para que el motor se mueva sin carga es 0,735v

Con los datos previamente obtenidos se simula el comportamiento del motor DC a la respuesta de un escalón de 7,5 V.

Figura 26. simulacion DC motor real



Fuente propia.

En la de la figura 23 c) ilustra el comportamiento de la velocidad angular ω se observa que la respuesta es muy similar a la obtenida experimentalmente con el tacómetro, tanto en tiempos transitorios como en el valor de estado estable demostrando que el valor de las constantes encontradas es adecuado para aproximar el comportamiento matemático del motor a la realidad.

5.2.3.8 RELACIÓN DE REDUCCIÓN DE LA TRANSMISIÓN

Como se observa en las figuras anteriores, los motores desarrollan velocidades angulares muy altas y por el contrario, el torque que producen es muy pequeño,

por tanto se les añade una transmisión para reducir velocidad y aumentar el torque, de tal forma que ellos puedan mover el conjunto de carga del prototipo.

Se necesita determinar el relacion de reduccion que tiene la transmisión implementada, entonces se mide las velocidades con transmisión y sin transmisión, para diferentes casos.

Tabla 8. Medida de relación de reducción

Win[rad/s]	Wout[rad/s]	G
907	6,26	144,88
1026	7,14	143,69
1508	10,46	144,16
Promedio		144,24

La relacion de reducción de la transmisión es G=144,24

5.2.4 MODELO MOTOR BLDC

En aplicaciones de uso intensivo de los motores, como es el caso de los vehículos eléctricos, los motores DC convencionales son poco prácticos, pues estos usan escobillas para conectar el rotor y el estator y estas generan muchos inconvenientes tanto en pérdidas de potencia como en requerimientos de mantenimiento. Por tal razón, para este tipo de aplicaciones se usa un tipo de motor diferente, el motor o brushless DC (BLDC) que están diseñados para conmutar la tensión en los devanados sin usar escobillas, entonces no sufren desgaste mecánico. Esto se logra a través de controladores digitales y sensores de posición para energizan cada una de las bobinas del motor en el momento idóneo generando así el mayor torque posible.

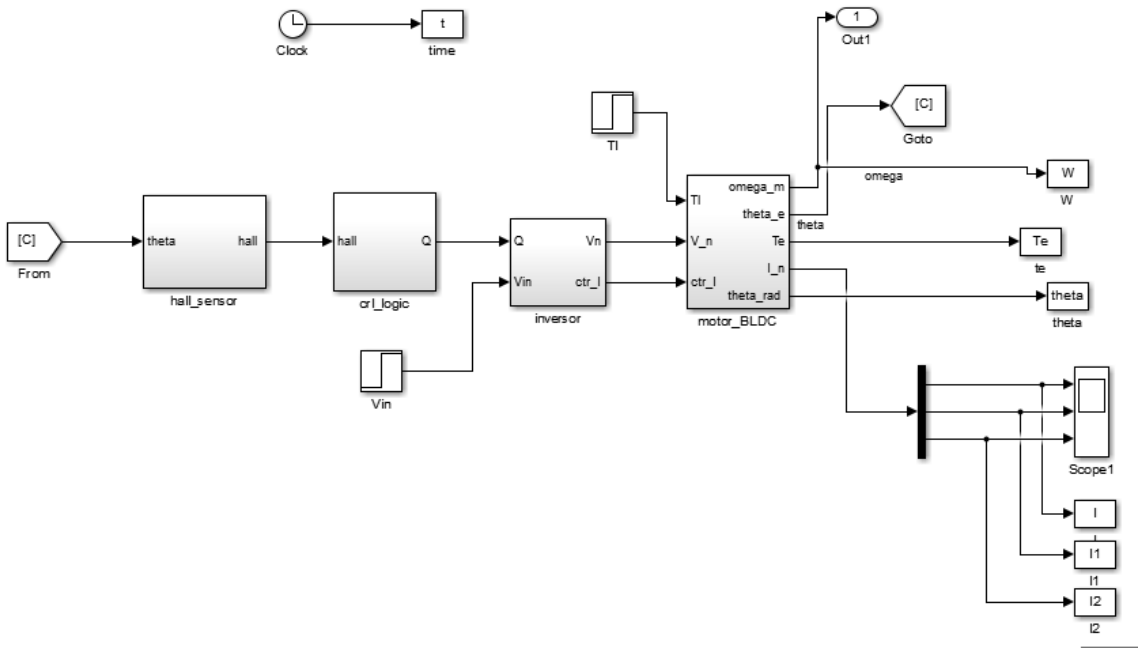
El motor BLDC se diferencia de uno convencional físicamente por la ausencia de las escobillas y en su reemplazo usa tres fases que se energizan en diferentes momentos y generar el campo necesario para el movimiento. En la selección del momento adecuado para energizar las bobinas se usa un controlador digital que con ayuda de sensores de efecto hall observa la posición del rotor en cada momento y según esto energiza las fases correctas.

Entonces el modelado del motor BLDC tiene 4 partes importantes mostradas en la figura 27

- Modelo de sensores de efecto hall
- Modelo del puente inversor
- Modelo del controlador

- Modelo de las ecuaciones del motor

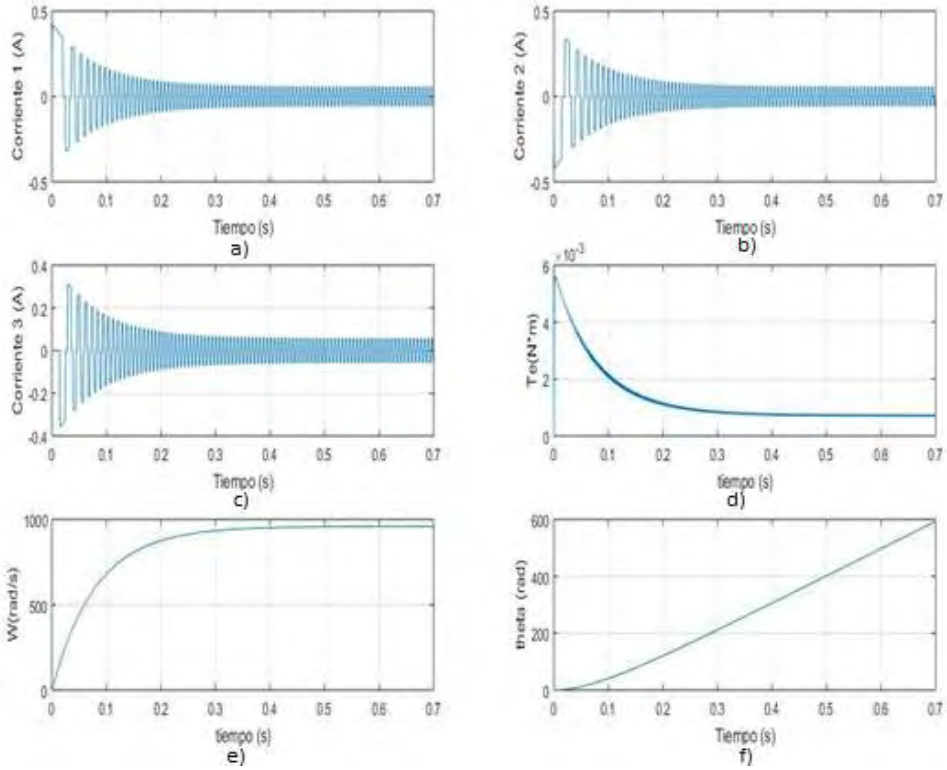
Figura 27. Diagrama de bloques motor BLDC



Fuente propia.

Se simula el modelo del motor BLDC previamente diseñado. La figura 28 muestra el comportamiento de las variables de estado en un motor BLDC, al observar y comparar estos resultados con los de la simulación del motor DC de la figura 26 observamos que el comportamiento de las variables de salida: torque y velocidad angular es muy similar para los dos modelos, por tanto se puede afirmar que al diseñar el control óptimo para el motor DC de nuestro prototipo, la señal de control generada funcionara de forma muy similar en el motor BLDC de un vehículo eléctrico comercial.

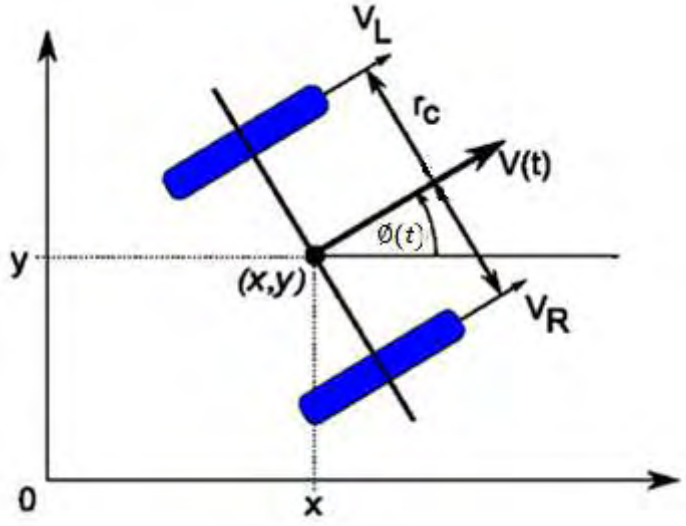
Figura 28. Simulación motor BLDC con parámetros reales



Fuente propia.

5.2.5 MODELAMIENTO CINEMÁTICO

Figura 29. Representación prototipo



Fuente: Marchewka y Piątek, Wheeled mobile robot modeling aspects.

Como ya se ha mencionado, el comportamiento cinemático del motor depende da la velocidad de cada una de las llantas (tracción diferencial). Así, si se modifica la velocidad de éstas, el robot variará su velocidad lineal o velocidad angular.

La velocidad lineal de cada llanta se expresa como:

$$v_L = \omega_L(t) * \frac{r_w}{G} = G_r \omega_L(t)$$
$$v_R = \omega_R(t) * \frac{r_w}{G} = G_r \omega_R(t) \quad (52)$$

Para motor izquierdo y derecho respectivamente.

Donde:

r_w : radio de la llanta [m]

G : relación de reducción

v_L : velocidad lineal llanta izquierda [m/s]

v_R : velocidad lineal llanta derecha [m/s]

ω_R : velocidad angular llanta izquierda [rad/s]

ω_L : velocidad angular llanta derecha [rad/s]

$$\mathrm{G_r} = \frac{\mathrm{r}_w}{\mathrm{G}} \quad (53)$$

Las velocidades lineal y angular del robot son:

$$V_T(t) = \frac{v_L(t) + v_R(t)}{2} = \dot{D_T}(t) \quad (54)$$

$$W_\emptyset(t) = \frac{v_L(t) - v_R(t)}{r_c} = \dot{\emptyset}(t) \quad (55)$$

donde:

V_T : velocidad lineal del WRM [m/s]

$W_\emptyset$: velocidad de giro del WRM [rad/s]

r_c : distancia de separación de las llantas

D_T : distancia recorrida por el WRM

$\emptyset$: ángulo de orientación del WMR

El ángulo de orientación y la posición (*x*,*y*) se encuentran a partir de:

$$\begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{\emptyset}(t) \end{bmatrix} = \begin{bmatrix} \mathrm{V_T(t)cos}\big(\emptyset(t)\big) \\ V_T(t)sen(\emptyset(t)) \\ W_\emptyset(t) \end{bmatrix} \quad (56)$$

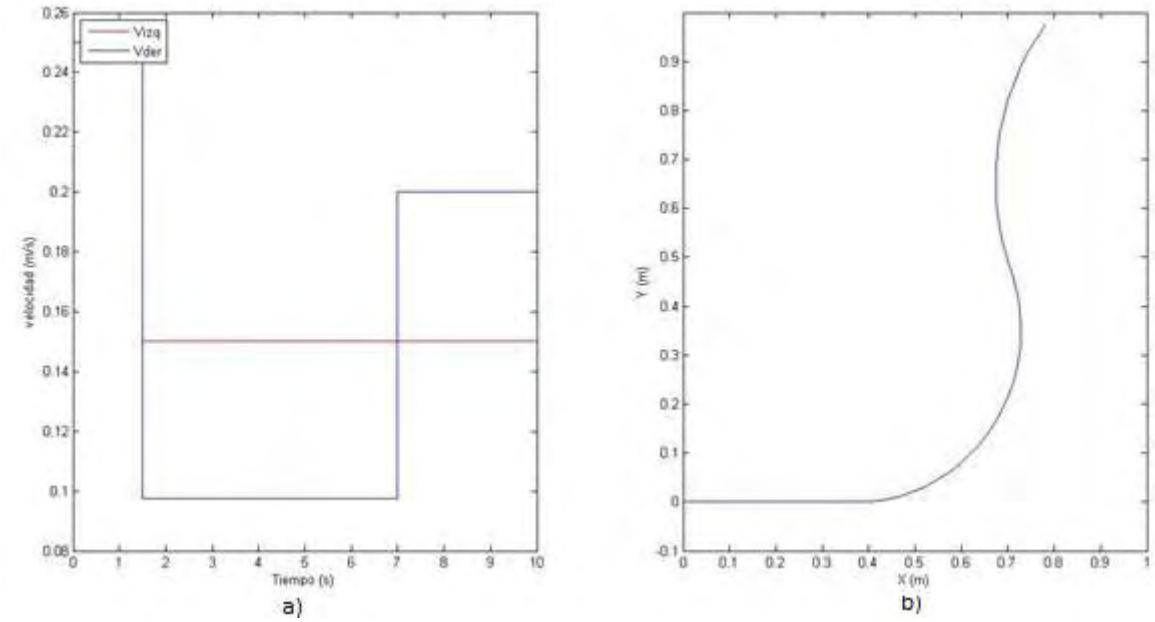
donde

x : posición del WMR en el eje x [m]

y : posición del WMR en el eje y [m]

Se simula el comportamiento cinemático del prototipo con diferentes condiciones en las entradas del sistema el resultado se muestra en la figura 30. La figura 30 a) muestra las entradas y la figura 30 b) muestra la posición del WRM en el plano (*x*,*y*).

Figura 30. Comportamiento cinemático del prototipo



Fuente propia.

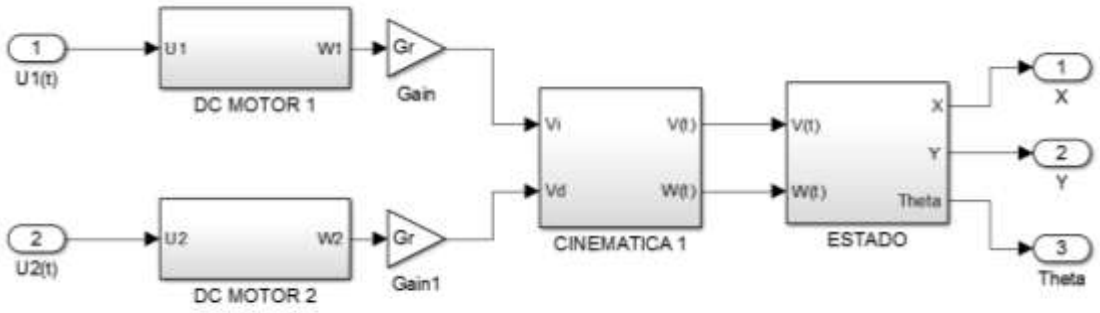
En la primera parte de la simulación, se plantea la condición v_R=v_L y se observa que el prototipo varía su posición mas no su dirección y se mueve en línea recta; luego se varía las velocidades de tal forma que v_R>v_L y el prototipo varía tanto posición como dirección, haciendo un recorrido en sentido opuesto a las manecillas del reloj; se hace nuevamente una variación en las velocidades de tal forma que v_R<v_L; en este caso el prototipo cambia su dirección de giro y se mueve

en el sentido de las manecillas del reloj. Así, para describir el comportamiento es notable que el prototipo se mueve en la dirección de la velocidad menor que se imprime. Si la velocidad menor es la izquierda, él girará hacia la izquierda y si la menor es la velocidad de la derecha girará hacia la derecha.

5.3 MODELO COMPLETO DEL SISTEMA

Usando las ecuaciones de las secciones anteriores se formula el siguiente diagrama de bloques, que describe la planta completa

Figura 31. Modelo completo prototipo



Fuente propia.

A continuación se muestran las ecuaciones que describen el modelo completo del prototipo. Para reducir la complejidad del problema de control se utiliza el modelo del motor simplificado en lugar del completo.

Usando la ecuación (45) obtenemos las ecuaciones que describen cada uno de los motores DC.

$$\begin{aligned} \dot{\omega_L}(t) &= \frac{K}{RJ}u_L(t) - \frac{(K^2+RB)}{RJ}\omega_L(t) - \frac{1}{J}\tau_l \\ \dot{\omega_R}(t) &= \frac{K}{RJ}u_R(t) - \frac{(K^2+RB)}{RJ}\omega_R(t) - \frac{1}{J}\tau_l \end{aligned} \quad (57)$$

Con la ecuación (53) añadimos el efecto de la transición a cada uno de los motores.

$$\begin{aligned} v_L &= G_r\omega_L(t) \\ v_R &= G_r\omega_R(t) \end{aligned} \quad (58)$$

Con las ecuaciones (54) y (55) se aplica la cinemática del WRM

$$V_T(t) = \frac{v_L(t) + v_R(t)}{2}$$
$$W_\emptyset(t) = \frac{v_L(t) - v_R(t)}{r_c} \quad (59)$$

De (56) obtenemos la posición coordenada y el ángulo de giro para cada instante.

$$\begin{bmatrix} \dot{x(t)} \\ \dot{y(t)} \\ \dot{\emptyset(t)} \end{bmatrix} = \begin{bmatrix} \mathrm{V_T(t)cos}\big(\emptyset(t)\big) \\ V_T(t)sen(\emptyset(t)) \\ W_\emptyset(t) \end{bmatrix} \quad (60)$$

En la tabla 9 se resumen los valores de las constantes de los modelos, medidos experimentalmente.

Tabla 9. Valores reales de constantes

Variable	**Valor**	**Unidades**
Resistencia (R)	8,85	Ω
Inductancia (L)	3,03	mH
Constante de motor (K)	0,00674619	-
Constante de fricción (B)	$7{,}41154 \cdot 10^{-7}$	-
Momento de inercia (J)	$4{,}76 \cdot 10^{-7}$	$Kg \cdot m^2$
Reducción de la transmisión (G)	144,24	-
Distancia entre llantas (r_c)	0,1	m
Radio de las ruedas (r_w)	0,023	m
Masa del prototipo (m)	0,67	kg

Fuente propia.

6 MÉTODO DE MÍNIMO DE PONTRYAGIN

El principio de mínimo de Pontryagin es un método que se deriva del cálculo variaciones y es usado en control óptimo para encontrar la mejor forma de llevar a un sistema dinámico de un estado a otro. El principio establece de manera informal que el hamiltoniano debe minimizarse sobre el conjunto de todos los controles permisibles, donde el hamiltoniano es una expresión conformada a partir del lagraniano y la dinámica del sistema siendo una función que describe el estado del sistema mecánico en términos de posición y de momento.

El principio de Pontryagin requiere de la solución de ecuaciones diferenciales, razón por la que se utiliza el Symbolic Math Toolbox de Matlab para facilitar los calculos, la solucion de estas ecuaciones de forma simbilica requiere un tiempo de procesamiento que aumenta drasticamente dependiendo de la complejidad del problema, por esta razon se usara el modelo simplificado del motor en lugar del completo y asi el tiempo de calculo de las soluciones se reduce de varios minutos a pocos segundos. El modelo completo del prototipo de VE es un sistema no lineal, y para disminuir el grado de dificultad en la solución que plantea el mínimo de Pontryagin se buscará la solución óptima bajo ciertas circunstancias que hacen que el sistema sea lineal e invariante en el tiempo.

6.1 PONTRYAGIN CON MODELO SIMPLIFICADO

6.1.1 MODELO DEL MOTOR EN VARIABLES DE ESTADO

Se toma las ecuaciones del motor (57) y hace un cambio de variable usando (58), así la variable de estado cambia de velocidad angular a velocidad lineal y además se añade el efecto de reducción de la transmisión.

$$\begin{aligned} \dot{\omega_L}(t) &= \frac{K}{RJ}u_L(t) - (K^2 + RB)\omega_L(t) - \frac{1}{J}\tau_l \\ \dot{\omega_R}(t) &= \frac{K}{RJ}u_R(t) - (K^2 + RB)\omega_R(t) - \frac{1}{J}\tau_l \end{aligned} \quad (61)$$

Multiplicando (61) por G_r se obtienen las ecuaciones en términos de velocidad lineal así:

$$\begin{aligned} \dot{v_L}(t) &= aG_r u_L(t) - bv_L(t) - cG_r \\ \dot{v_R}(t) &= aG_r u_R(t) - bv_R(t) - cG_r \end{aligned} \quad (62)$$

Se define: $a = \frac{K}{RJ}, b = \frac{k^2+RB}{RJ}, c = \frac{\tau_l}{J}G_r, G_r = \frac{r_w}{G}$

El efecto producido por la transmisión, se observa en la velocidad angular y en el torque. La velocidad angular producida por el conjunto motor-transmisión se reduce y el torque aumenta. Un factor a destacar en la ecuación resultante es el factor $-\frac{\tau_L}{J}G_r$, que describe el torque de fuerzas de oposición observado desde el eje del rotor.

Se definen las variables de estado del modelo.

Para motor 1(izquierda): $x_1 = \mathrm{V_L}$

Para motor 2(derecha): $x_2 = \mathrm{V_R}$

y del modelo cinemático (59): $x_3 = D_T$

$x_4 = \emptyset$

Expresando las ecuaciones en forma de estados:

Motor 1: $$\dot{x_1} = aG_r u_1 - bx_1 - cG_r \quad (63)$$

Motor 2:

$$\dot{x_2} = aG_r u_2 - bx_2 - cG_r \quad (64)$$

De las ecuaciones de cinemática (59):

$$\dot{x_3} = \frac{(x_1 + x_2)}{2} \quad (65)$$

$$\dot{x_4} = \frac{(x_1 - x_2)}{r_c} \quad (66)$$

Las ecuaciones (63) a (66) definen el modelo en variables de estado del sistema

6.1.2 CASO 1: LA TRAYECTORIA ES UNA RECTA

Cuando el robot se está moviendo en línea recta, al asumir que los motores izquierdo y derecho son idénticos, se debe poner la misma señal de entrada para que se produzcan velocidades iguales en las dos ruedas y el motor se mueva en línea recta.

Entonces las condiciones del caso serían: $u_1 = u_2$ y al suponer motores exactamente iguales también tendremos que: $x_1 = x_2$ y $\dot{x_1} = \dot{x_2}$

Usamos el siguiente índice de rendimiento

$$j = \int_{t_0}^{t_f} \frac{U^2}{2}\,dt = \int_{t_0}^{t_f} \frac{(u_1 + u_2)^2}{2}\,dt = \int_{t_0}^{t_f} 2u_1^2\,dt \quad (67)$$

Usando las condiciones del caso y reemplazando en (60) y (61) tenemos que:

$$\dot{x_3} = x_1$$
$$\dot{x_4} = 0 \qquad (68)$$

Al reemplazar las condiciones de este caso en las ecuaciones de estado de los motores, se elimina unas ecuaciones de estado. Entonces para este caso las ecuaciones de estado son: (63), (64) y (68).

El hamiltoniano para este caso sería:

$$H = j + l_1\dot{x_1} + \mathrm{l}_2\dot{x_2} + \mathrm{L}_3\dot{x_3} \qquad (69)$$

donde:

l_1, l_2, L_3 son los multiplicadores de Lagrange para este caso.

Por las condiciones impuestas al caso podemos reemplazar $\dot{x_1} = \dot{x_2}$ y operar para reducir la complejidad del problema obteniendo:

$$H = j + L_1\dot{x_1} + \mathrm{L}_3\dot{x_3}$$
$$H = 2u^2 + L_1(aG_r u_1 - bx_1 - cG_r) + \mathrm{L}_3 x_1 \qquad (70)$$

Donde por simplicidad del problema se define: $L_1 = l_1 + l_2$

Aplicando $\left(\frac{\partial H}{\partial u}\right)_* = 0$ y despejando u^* obtenemos la señal entrada de óptima:

$$u^* = -\frac{\mathrm{aG_r L_1}}{4} \qquad (71)$$

Y el hamiltoniano optimo:

$$H^* = L_3 x_1 - L_1(cG_r + bx_1) + \frac{L_1^2 G_r^2 a^2}{8} \qquad (72)$$

Entonces las ecuaciones diferenciales que describen el sistema son:

$$\dot{x_1^*} = \left(\frac{\partial H}{\partial L_1}\right)_* = -\,cG_r\; -\; bx_1\; - \frac{{G_r}^2 L_1 a^2}{4}$$
$$\dot{x_3^*} = \left(\frac{\partial H}{\partial L_3}\right)_* = x1$$
$$\dot{L_1^*} = \left(\frac{\partial H}{\partial x_1}\right)_* = L_1 b - L_3 \qquad (73)$$
$$\dot{L_3^*} = \left(\frac{\partial H}{\partial x_3}\right)_* = 0$$

Para resolver el problema de optimizacion por el metodo de pontryagin se hace uso del Symbolic Math Toolbox de Matlab, donde se introduce el sistema de ecuaciones diferenciales (73) y se plantean diferentes condiciones de contorno para las variables del sistema. En general el algoritmo resuelve el problema dejando las ecuaciones diferenciales en función de las constantes de integración, que se determinan aplicando las condiciones de contorno de las variables de estado especificas de cada problema especifico a implementar. El algoritmo implementado se muestra en el anexo E.

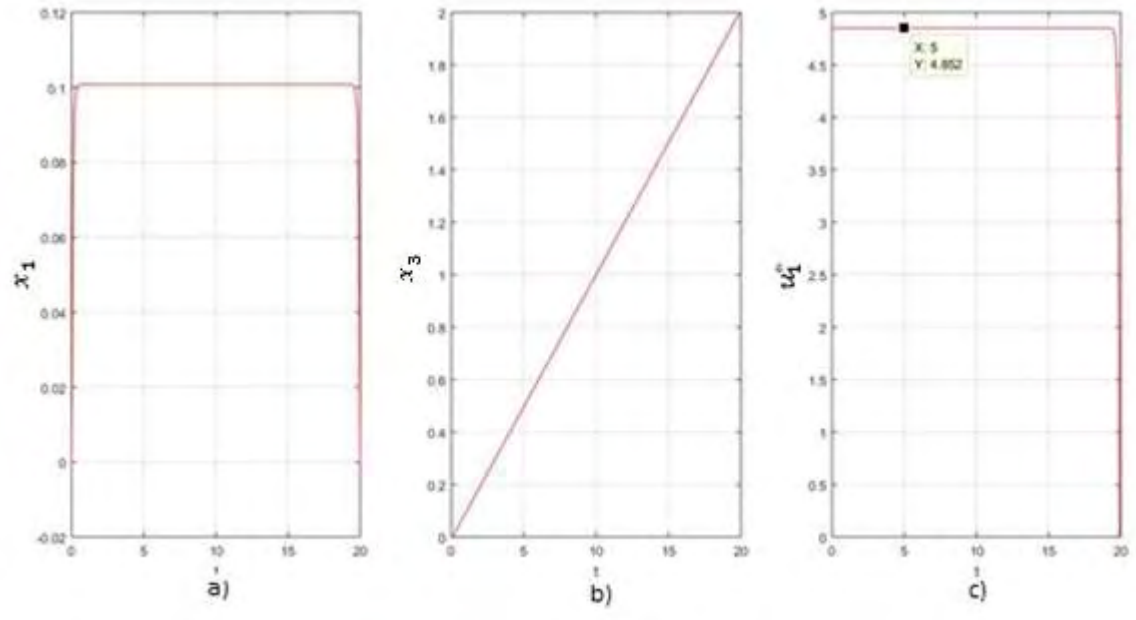
Las soluciones generales de cada caso de optimizacion son demasiado extensas para mostrarlas en forma de ecuacion, por tanto unicamente se muestra la forma de la señal que ellas representan para cada caso especifico.

Para la solucion de este caso, usamos los valores de las constantes previamente encontradas $a = \frac{130.3}{0.08}$ y $b = \frac{1}{0.08}$ y se simula el caso donde se recorre una recta sin pendiente, por tanto se tiene que el torque debido a la carga $\tau_l = 0$ por tanto: $c = 0$

A continuacion se presenta la solucion del problema variando las condiciones de contorno:

a. Se plantea que el vehiculo inicie desde el reposo, recorra una distancia en un tiempo preestablecido y se detiene. Por tanto las condiciones de contorno son: $x_1(0) = 0, x_1(20) = 0,\ x_3(0) = 0, x_3(20) = 2.$

Figura 32. Control óptimo con condiciones de contorno de inicio y parada

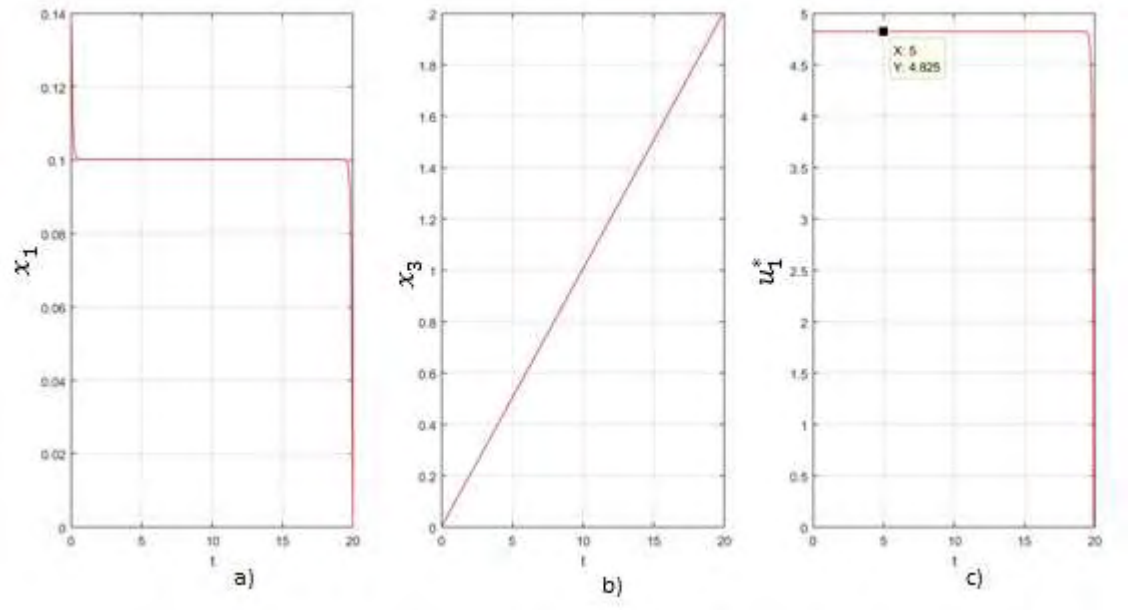


Fuente propia

La figura 32 a) representa la entrada óptima calculada por el algoritmo donde se observa que el voltaje de entrada se mantiene constante durante la mayor parte de tiempo y al final reduce su valor hasta llegar a cero, este comportamiento se debe a las condición final de velocidad $x_1(tf) = 0$ que hace que se reduzca el valor de la alimentación para que los motores se detengan. La figura 32 a) muestra que la velocidad del vehículo aumenta rápidamente, se mantiene constante durante la mayor parte del recorrido y al final se reduzca hasta cero que son las condiciones que se habían definido cabe resaltar que durante la mayor parte del recorrido la velocidad de mantiene constante, esto con el fin de ahorrar energía pues la velocidad encontrada es la mínima que el vehículo debe desarrollar para alcanzar la meta final de distancia en ese intervalo de tiempo. La figura 32 b) nos muestra que la meta deseada llego a cumplirse pues al llegar al cabo de 20 segundos el vehículo alcanzo la distancia de dos metros y se detuvo.

b. Se plantea que el vehiculo inicie con una velocidad predefinida mayor, recorra una distancia en un tiempo preestablecido y termine su movimiento con una velocidad constante. Por tanto las condiciones de contorno planteadas son: $x_1(0) = 0.14, x_1(20) = 0,\ x_3(0) = 0, x_3(20) = 2.$

Figura 33. Control óptimo con condiciones de contorno diferentes de cero

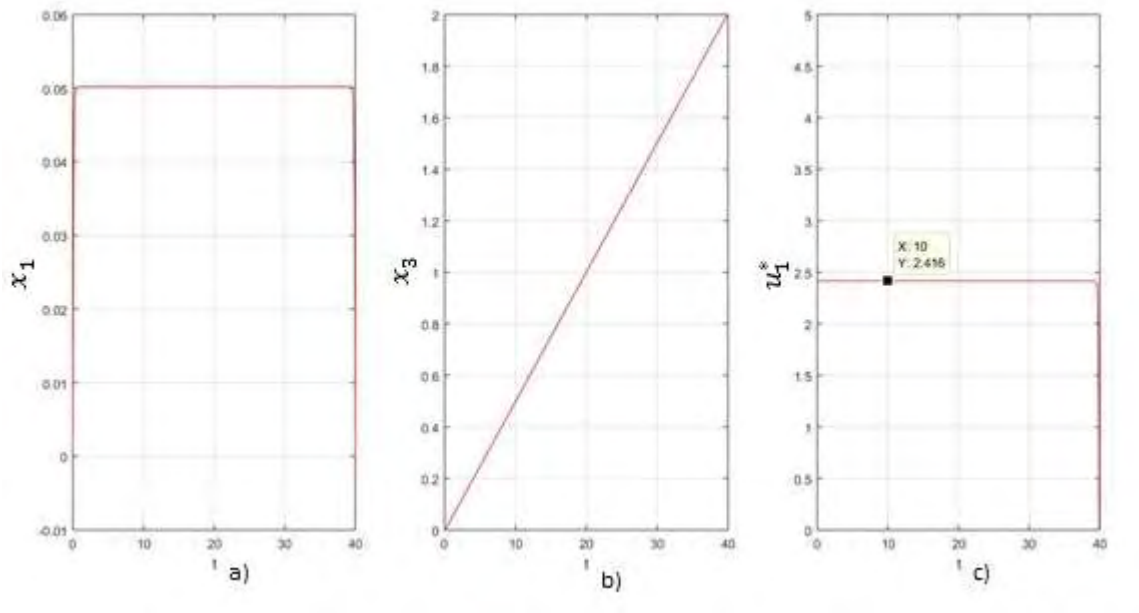


Fuente propia.

En la figura 33 c) se observa un comportamiento similar al anterior pero el voltaje en la zona constante es ligeramente menor debido a que se necesita menos energía por que el vehículo no debe arrancar. La figura 33 a) muestra que la velocidad decrece rápidamente, se establece en un valor constante y al final decrece hasta detenerse cumpliendo la condición final de parada. La figura 33 b) muestra que para este caso la meta deseada también se cumplió en el tiempo final predefinido. En este caso al no tener que gastar energía en el arranque hay un pequeño ahorro de energía que se observa voltaje estable de la alimentación que es menor al del caso 1.

c. Se plantea que el vehiculo recorra la distancia en el doble de tiempo. Por tanto las condiciones de contorno planteadas son las mismas del caso 1, pero duplicando el tiempo final: $x_1(0) = 0, x_1(40) = 0,\ x_3(0) = 0, x_3(40) = 2$

Figura 34. Control óptimo modificando el tiempo límite



Fuente propia.

En la figura 34 c) se observa que al aumentar el tiempo requerido por el algoritmo para cumplir la meta, se disminuye la magnitud de la señal de entrada, que pasa de ser 4,852 a 2,416 en la región estable. En la comportamiento de la velocidad y la distancia recorrida mostrados en la figura 34 a) y la figura 34 b) es similar al del caso 1.

Comparando el caso 1 con el caso 2 se observa que el valor de voltaje se disminuye un poco más de la mitad por lo que hay un leve ahorro en la energía

que se suministra al vehículo. El ahorro de esta energía es poco apreciable pues al coeficiente de fricción viscosa B es muy pequeño para el motor usado, por tanto hay pocas perdías de energía debido a ella; este ahorro resultara mucho más notorio en un vehículo real, con motores de gran tamaño y que permitan alcanzar velocidades más elevadas pues los efectos de fricción viscosa y además de fricción aerodinámica van a ser mucho más notorios dependiendo que tan elevada sea la velocidad que estos alcancen.

En los 3 casos presentados anteriormente siempre se observa que sin importar las condiciones de contorno que se programen el algoritmo siempre tiende a estabilizar la alimentación buscando el voltaje más bajo posible para realizar el recorrido y al final aumentar o disminuir rápidamente su valor para alcanzar las condiciones de contorno previamente programadas. Para nuestro caso de estudio el cambio que se produce en la velocidad y en la alimentación es muy rápido, debido a que el motor tiene transitorios muy bajos y aumenta o disminuye su velocidad en menos de 0,1 segundos.

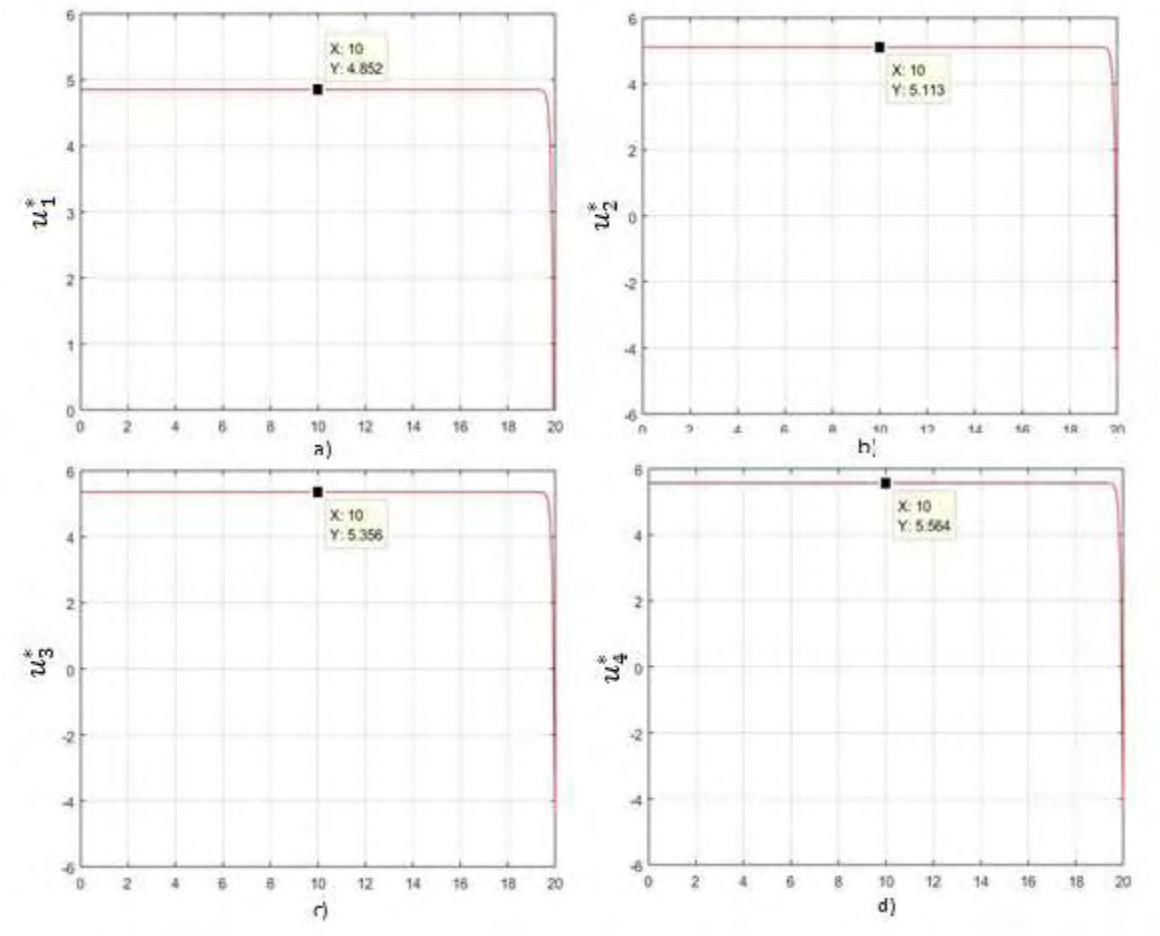
6.1.3 CASO 2: LA TRAYECTORIA ES UNA RECTA CON PENDIENTE

En este caso se tiene en cuenta la fuerza de oposición que genera el conjunto total de la carga que tienen que mover los motores expresada como τ_L. Las principales fuerzas que se oponen al movimiento del prototipo se describieron en la sección 5.1; sin embargo, de estas fuerzas solo se tiene en cuenta la que es debida al peso, pues el prototipo no alcanza velocidades demasiado grandes; así, el efecto de las otras fuerzas es despreciable.

El sistema de ecuaciones de este caso es el mismo utilizado en el caso 1, con la diferencia única de que aquí se analizara la trayectoria como una recta con una pendiente constante, por tanto $\tau_L \neq 0$.

Se plantea hacer una comparativa entre las alimentaciones óptimas obtenidas manteniendo las condiciones de contorno en todas las pruebas y solo variando el ángulo de inclinación de la pendiente. Las condiciones de contorno programadas son: $x_1(0) = 0, x_1(20) = 0,\ x_3(0) = 0, x_3(20) = 2$ y los ángulos para cada prueba son: $\alpha_1 = 0, \alpha_2 = 15, \alpha_3 = 30,\ \alpha_4 = 45$

Figura 35. Señal de alimentación óptima aplicando diferentes pendientes.



Fuente propia.

En la figura 33 se observa los resultados de las pruebas realizadas en este caso. La figura a) corresponde a una pendiente de 0°, b) una pendiente de 15°, c) una pendiente de 30° y d) una pendiente de 45°. Es apreciable que el valor de voltaje de la alimentación óptima varía de acuerdo al modificar pendiente de la recta, se observa que al aumentar la pendiente aumenta el valor medio de la alimentación, aumentando así el consumo de energía requerido para cumplir la meta planteada pues se necesita un mayor esfuerzo para mover el peso total del prototipo.

6.1.4 CASO 3: LA TRAYECTORIA ES UNA CURVA DE RADIO CONSTANTE

En este caso se propone que el prototipo siga la trayectoria de una curva con radio constante, o sea que describa una semicircunferencia por una determinada

distancia. Condición que asegura que el sistema siga siendo lineal e invariante en el tiempo.

De las ecuaciones de movimiento circular uniforme:

$$V_T(t) = C_R * W_\emptyset(t) \quad (74)$$

donde:

C_R : radio curva [m]

$V(t)$: velocidad lineal del WMR [m/s^2]

$W_\emptyset(t)$: velocidad angular del WMR [rad/s^2]

Se reemplaza fórmulas del modelo cinemático del robot (59) en la ecuación (74), se opera y reorganiza la ecuación, obteniendo:

$$V_L = V_R * \frac{2C_R - r_c}{2C_R + r_c} \quad (75)$$

En la ecuación (75) se observa que C_R y r_c son constantes entonces para que el prototipo se mueva con una trayectoria de radio constante V_l debe ser proporcional a V_r. Con un factor de proporción, ρ, que se calcula con:

$$\rho = \frac{2C_R - r_c}{2C_R + r_c} \quad (76)$$

Los motores son sistemas lineales e invariantes en el tiempo, por tanto se puede afirmar que:

Si $u_2 = \rho u_1$, entonces $x_2 = \rho x_1$

Con lo que se modifica las ecuaciones de estado cinemáticas del sistema (65) y (66):

$$\dot{x_3} = \frac{x_1(\rho + 1)}{2} \quad (77)$$

$$\dot{x_4} = \frac{x_1(\rho + 1)}{r_c} \quad (78)$$

Donde para este caso x_3 es el perímetro de la circunferencia recorrido y x_4 es el angulo de la circunferencia barrido por el WRM.

Al cumplirse la condición de que $V_l = \rho V_r$, el prototipo siempre se va a mover en una curva de radio constante añadiendo la ventaja de tener que calcular solo una señal de alimentación optima ya que gracias a las condiciones impuestas, al calcular la variable para uno de las llantas del WRM obtendremos la otra multiplicándola por el factor ρ. Para este caso no se tiene en cuenta la ecuación de estado (78) pues se va a analizar como condición de salida la variable de estado de distancia recorrida ósea el perímetro de la circunferencia recorrido

variable que es suficiente para calcular el control optimo en este caso haciendo que (78) sea innecesaria.

Teniendo en cuenta lo anterior, se plantea el problema de optimización.

Se escoge el índice de rendimiento:

$$j = \int_{t_0}^{t_f} \frac{U^2}{2} dt = \int_{t_0}^{t_f} \frac{(u_1 + \rho u_2)^2}{2} dt = \int_{t_0}^{t_f} \frac{(1+\rho)^2 u_1^2}{2} dt \quad (79)$$

El Hamiltoniano de este caso sería:

$$H = \frac{u_1^2(1+\rho)^2}{2} + l_1 \dot{x_1} + l_2 \dot{x_2} + \mathrm{L}_3 \dot{x_3} \quad (80)$$

Donde

l_1, l_2, L_3 son los multiplicadores de Lagrange para este caso.

Por las condiciones impuestas al caso podemos reemplazar $\dot{x_1} = \rho \dot{x_2}$ y operar para reducir la complejidad del problema obteniendo:

$$H = \frac{u_1^2(1+\rho)^2}{2} + l_1 \dot{x_1} + l_2 \dot{x_1} \rho + L_3 \dot{x_3} \quad (81)$$

$$H = \frac{u_1^2(1+\rho)^2}{2} + L_1 \dot{x_1} + L_3 \dot{x_3}$$

Donde $L_1 = l_1 + \rho l_2$

Reemplazando las ecuaciones de estado en el hamiltoniano:

$$H = \frac{\mathrm{u}^2(1+\rho)^2}{2} + \mathrm{L}_1(aG_r u_1 - bx_1 - cG_r \tau_l) + \mathrm{L}_3 \frac{x_1(\dot{\rho} + 1)}{2} \quad (82)$$

Aplicando $\left(\frac{\partial H}{\partial u}\right)_* = 0$ y despejando u^* obtenemos la señal de entrada óptima:

$$u^* = -\frac{aGrL_1}{(\rho + 1)^2} \quad (83)$$

Y el hamiltoniano optimo

$$H^* = \frac{L_3 x_1(1+\rho)}{2} - L_1(cG_r - bx_1) - \frac{L_1^2 G_r^2 a^2}{2(\rho + 1)^2} \quad (84)$$

Las ecuaciones diferenciales generales son:

$$
\begin{aligned}
\dot{x_1} &= \left(\frac{\partial H}{\partial L_1}\right)_* = -cG_r - bx_1 - \frac{L_1 G_r^2 a^2}{(\rho + 1)^2} \\
\dot{x_3} &= \left(\frac{\partial H}{\partial L_2}\right)_* = \frac{x_1(\rho + 1)}{2} \\
\dot{L_1} &= \left(\frac{\partial H}{\partial x_1}\right)_* = L_1 b - \frac{L_3(\rho + 1)}{2} \\
\dot{L_3} &= \left(\frac{\partial H}{\partial x_3}\right)_* = 0
\end{aligned}
\qquad (85)
$$

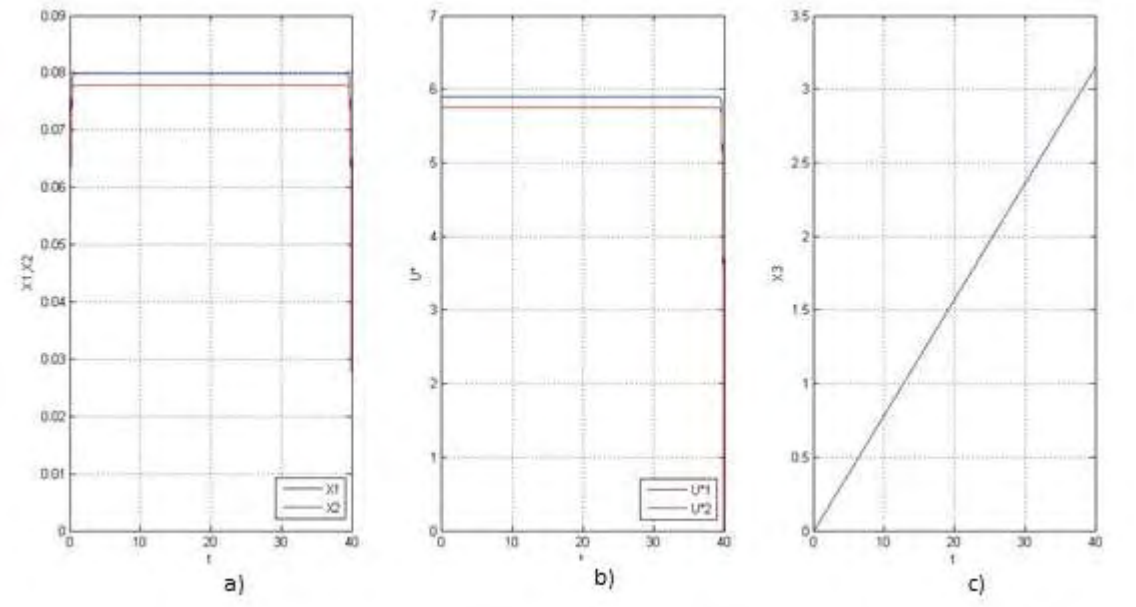
Al resolver el sistema de ecuaciones planteado en (85) se encontrara la señal óptima del problema para una curva de radio constante. Para la solución de este sistema de ecuaciones se hace uso del Symbolic Math Toolbox de Matlab que en general resuelve el problema dejando las ecuaciones en funcion de las variables de integracion para luego con las condiciones de contorno de cada problema en especifico encontrar el valor real de estas constantes. Hay que resaltar que de la forma que se planteo el problema se obtendra la señal de alimentacion optima para uno de los motores y la otra señal se calcula al final del algoritmo multiplicando la señal calculada por el factor ρ. El algoritmo implementado se muestra en el anexo F

Las soluciones generales de cada caso de optimizacion son demasiado extensas para mostrarlas en forma de ecuacion, por tanto unicamente se muestra la forma de la señal que ellas representan para cada caso especifico.

Para la solucion de este caso, usamos los valores de las constantes previamente encontradas $a = \frac{130.3}{0.08}$ y $b = \frac{1}{0.08}$ y se simula el caso donde se recorre una curva sin pendiente, por tanto se tiene que el torque debido a la carga $\tau_l = 0$ por tanto: $c = 0$. Se plantea que el vehículo inicie desde el reposo y haga un recorrido sobre una curva de radio constante $C_R = 2m$ y que recorra una cuarta parte de la circunferencia en un tiempo de 40 segundos y se detenga. Para este caso las condiciones de contorno son:

$x_1(0) = 0, x_1(40) = 0,\ x_3(0) = 0, x_3(40) = C_R \frac{\pi}{4} = \frac{\pi}{2}[m]$

Figura 36. Control óptimo curva

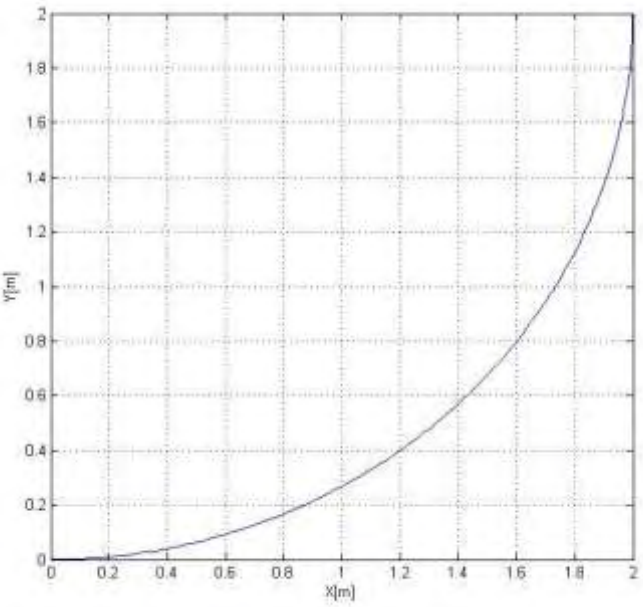


Fuente propia.

Una vez que se encontró la señal óptima para u_1 se la multiplico por el factor ρ para encontrar u_2 teniendo así las señales de control para los dos motores. En la figura 36 b) se muestran las señales óptimas de alimentación para cada motor. En la figura 36 a) se observa el comportamiento de la velocidad para cada motor y en la figura 36 c) se observa la distancia total recorrida por el WRM.

Se usa las señales optimas de alimentación obtenidas y se simulan como entradas en el modelo del prototipo donde se observa la respuesta de recorrido en el plano coordenado (x,y) al cabo de los 40 segundos de recorrido. El resultado se observa en la figura 37 demostrando que el WMR realiza una trayectoria en forma de circunferencia con un radio constante de 2 metros y que recorre un cuarto de circunferencia antes de detenerse al llegar al tiempo límite propuesto.

Figura 37. Trayectoria óptima recorrida en 40 segundos



Fuente propia.

En la anterior solución, se observa que al hacer las simplificaciones de las suposiciones planteadas, el problema de optimización en la curva se asemeja mucho al problema de la recta, variando únicamente algunas constantes. Y por tanto, la forma de su respuesta es similar también. La diferencia radica en que la señal que alimenta uno de los motores está multiplicada por el factor de proporcionalidad "ρ".

7 IMPLEMENTACIÓN DE LAS SOLUCIONES ENCONTRADAS

7.1 ENVIÓ DE LAS SEÑALES ÓPTIMAS

Al implementar el método de Pontryagin, obtenemos como resultado la entrada optima que se debe introducir al prototipo para que este logre la meta propuesta y consuma la menor energía posible; además muestra la forma que tendrán las variables de estado del sistema como respuesta a la entrada optima; por tanto, el método de Pontryagin proporciona una señal analógica que se debe introducir al prototipo.

Para alimentar el prototipo con esta señal existen dos soluciones: la primera seria implementar un conversor digital–análogo (DAC) e implementar una etapa de potencia para controlar el voltaje enviado a través de un PIC e implementar una etapa de potencia para amplificar la corriente que necesita el prototipo; la segunda opción es controlar el promedio de voltaje que a entregar por medio de un control por ancho de pulso (PWM).

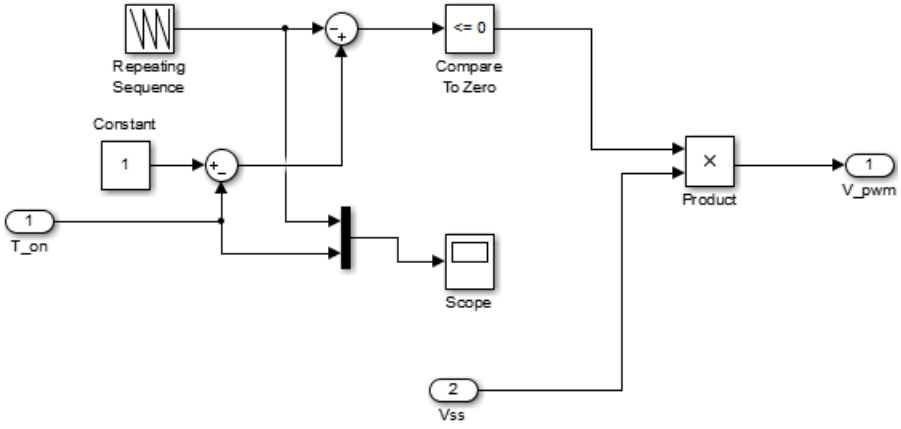
Se implementó la primera solución usando un DAC referencia TLC5615C y se implementó un driver con dos transistores en configuración Darlingtong en la etapa de potencia, surgiendo algunos inconvenientes pues el voltaje que el DAC debía vencer para iniciar el movimiento del prototipo aumentaba mucho por la influencia de los transistores y la resolución del conversor resultó siendo muy pequeña y fluctuante no pudiendo obtener un voltaje estable.

Por lo anterior, se decide implementar un control por PWM, que presenta menos inconvenientes.

7.1.1 IMPLEMENTACIÓN DEL PWM

Con la ayuda de Simulink se diseñó un modelo de generador PWM que se aproxima al que se plantea usar figura 38. Una vez implementado el modelo el generador PWM, se une con el modelo del motor previamente diseñado y se observa la salida generada por esta señal de entrada

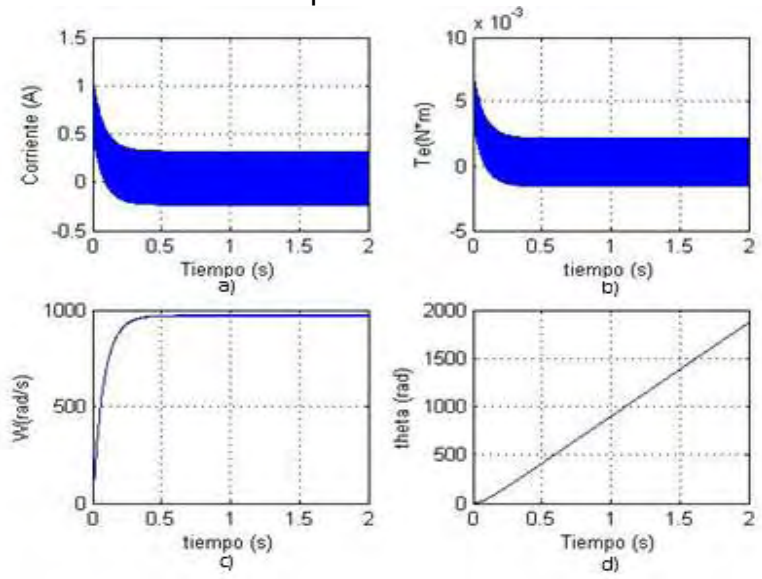
Figura 38. Modelamiento generador de PWM



Fuente propia.

En la figura 39 se observan las salidas del modelo del motor al alimentarlo con un generador PWM. En la figura 39 a) y b) se observa que hay un cambio en la corriente y el torque, pues el swicheo del PWM produce una oscilación en estas dos señales sin embargo en la figura c) y d) se observa que la salida de velocidad angular y la de posición son muy aproximadas a las generadas al alimentar el modelo con una señal analógica constante. Razón por la que se determina que implementar la solución a través de un PWM es válida para enviar nuestras señales optimas al prototipo.

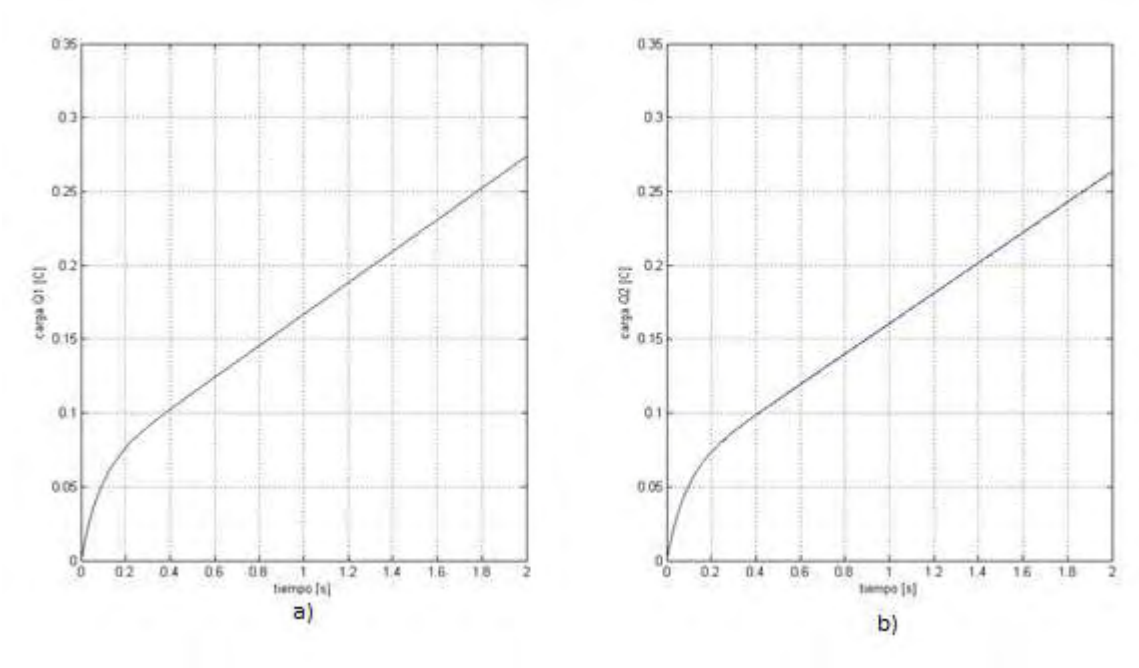
Figura 39. Motor DC alimentado por WMR



Fuente propia.

La figura 40 muestra una comparación entre el consumo presentado alimentando el motor con una señal analógica y alimentándolo con una señal PWM. Se observa que la diferencia que se presenta en las dos señales se puede despreciar, pues al integrar la corriente con respecto al tiempo se obtiene la carga Q en Coulombs que consume el motor a través del tiempo. En la figura 40 se muestra el consumo de carga vs tiempo simulado para uso de la entrada analógica a) y uso de alimentación con PWM b). Se observa que el consumo de carga también es muy similar en los dos casos.

Figura 40. Comparación consumo



7.2 MEDIDA DE DESPLAZAMIENTO

En primera instancia se trata de medir las señales a través de una IMU análoga que estaba disponible en los laboratorios de la universidad. Se hace una gran cantidad de filtrado tanto físico como digital y al final se determinó que los resultados de medición obtenidos eran muy erróneos; luego se cambia a una IMU digital de bajo costo, a la que se somete a un proceso de filtrado digital y al ponerlo a prueba se observó que la calidad de las señales obtenidas de los acelerómetros era muy baja, además sumando al error producido por la doble integración necesaria para convertir aceleración a distancia los errores acumulativos encontrados eran muy grandes. Por estas razones se implementó una solución alternativa: ya que conocemos el modelo del sistema, se implementó

una simulación en tiempo real usando el modelo dinámico en espacio de estados del WMR (86) y las entradas de voltaje medidas desde los motores.

Al hacer esta simulación en tiempo real, y usando los valores reales entregados a los motores se obtuvo una medida aproximada de la posición del WMR en el plano (x,y) de cerca de $\pm 2\,cm$ por cada metro recorrido. Hay que resaltar que este método funciona bajo la condición de idealidad en el recorrido, sin tener en cuenta los imprevistos u obstáculos que puedan alterar su movimiento ideal.

$$\begin{bmatrix} \dot{v}_1 \\ \dot{v}_2 \\ \dot{\delta} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} -b & 0 & 0 & 0 \\ 0 & -b & 0 & 0 \\ 0.5 & 0.5 & 0 & 0 \\ \frac{1}{r_c} & -\frac{1}{r_c} & 0 & 0 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ \delta \\ \theta \end{bmatrix} + \begin{bmatrix} a * G_r & 0 & 1 \\ 0 & a * G_r & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} U_1 \\ U_2 \\ F_l \end{bmatrix} \quad (86)$$

$$\begin{bmatrix} v_1 \\ v_2 \\ \delta \\ \theta \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ \delta \\ \theta \end{bmatrix}$$

Los resultados de la medición de distancia por este método se presentan más detalladamente en el capítulo 8.

8 DESCRIPCIÓN DEL PROTOTIPO DEL VEHÍCULO

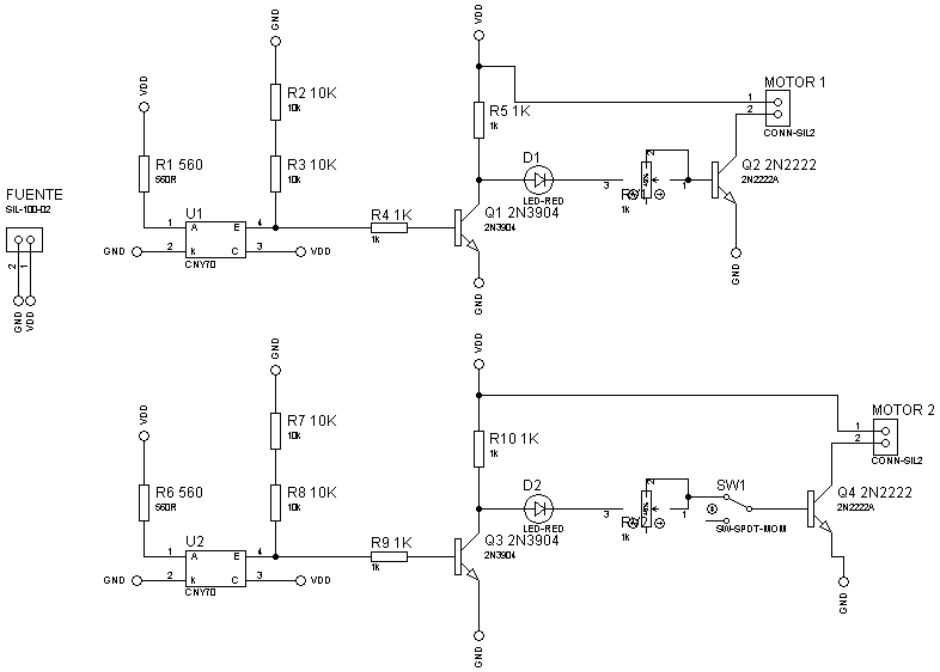
El prototipo de vehículo eléctrico cuenta con un chasis desarmable de SparkFun sobre el cual se hace el montaje de las abrazaderas para los motores, la batería, el circuito seguidor de línea, una rueda loca y la tarjeta de adquisición. Además, cuenta con tres placas (superior, media e inferior), a las cuales se fijan los distintos dispositivos.

Para determinar una trayectoria cerrada se debe contar con un piloto, el cual se simula con un circuito seguidor de línea utilizando los sensores CNY70 cuyo esquema es el de la figura 41, los cuales corrigen la trayectoria del vehículo y lo mantienen dentro de la pista de prueba. La batería utilizada es de 2650 mAh, dada la necesidad de mantener un voltaje estable durante todo el recorrido del circuito, además de alimentar las distintas partes del circuito.

En el montaje de la tarjeta de adquisición se cuenta con cuatro soportes, como se muestra en la figura 42, los cuales adhieren con tornillos la tarjeta al chasis, manteniendo un correcto movimiento en conjunto vehículo-tarjeta.

Los motores pueden impulsar un peso máximo de 2 kg y gira a un máximo de 200 rpm a 12 V. Se escogieron dichos parámetros según el peso total del prototipo, medido en 670 g.

Figura 41. Esquema del circuito seguidor de línea.



Fuente propia.

Figura 42. A, B, C y D son los soportes para la tarjeta de adquisición. E, llanta. F, placa inferior; G, placa media.

Fuente propia.

Cada motor se introduce en una abrazadera marca pololu la cual se atornilla a la placa inferior como se muestra en la figura 43.

Figura 43. Motores en abrazadera y fijados a la placa inferior.

Fuente propia.

Los sensores CNY70, se mantienen firmes a través de soportes con tornillos y una placa de acrílico que los mantiene a una distancia de 3cm, determinada por el ancho de la línea en la pista de prueba final de 3,5cm. Figura 44.

Figura 44. A y B; Sensores CNY70 instalados.

Fuente propia.

9 PRUEBAS Y RESULTADOS

En esta sección se presentan los resultados de la medida de las variables por telemetría y de la respuesta obtenida en el prototipo al implementar el control óptimo. Todas las mediciones se tomaron utilizando el software descrito en el anexo A.

Como se había planteado, los resultados se analizarán para los dos casos propuestos: recta y curva de radio constante.

La medida del error relativo porcentual se calcula con (79)

$$E_r = \left|\frac{V_r - V_t}{V_r}\right| \times 100 \qquad (87)$$

donde:

E_r: error relativo porcentual.

V_r: valor real medido.

V_r: valor teórico.

9.1 PRUEBAS EN RECTA

Se somete el sistema a diferentes condiciones de entorno, para observar la variación en los resultados adquiridos, cabe resaltar que las pruebas se realizan sin el seguidor línea para corroborar el funcionamiento del sistema sin correcciones en la trayectoria. En el caso de la tabla 10 se propone como meta el recorrer un metro y se usan diferentes tiempos como condición para lograr este recorrido.

Tabla 10. Medida de distancia con diferentes tiempos

Tiempo programado [s]	Distancia Real recorrida [m]	Error relativo distancia real [%]	Distancia medida con software [m]	Error relativo software [%]
6	0.997	0.3	0.987064	0.997
7	0.994	0.6	0.982533	1.154
8	0.986	1.4	0.979069	0.703
9	0.992	0.8	0.997644	0.569
10	0.992	0.8	0.997771	0.582
11	0.987	1.3	0.994328	0.742
Promedio		0.87		0.791

Fuente propia.

El error relativo de la distancia real se calcula entre la distancia programada y la distancia recorrida por el prototipo y medida con un metro. Se observan variaciones tanto en la medica como en la distancia real recorrida pues las señales enviadas a los motores no son exactamente iguales a las calculadas y existe un leve desface entre el tiempo de la simulación y el tiempo real como se muestra en la Figura 45.

La Tabla 11 muestra los resultados de pruebas programando el software para diferente distancia y tiempo límite. Es observable que en todo momento el error del recorrido y de la medida son inferiores al 2%, mostrando que bajo las condiciones planteadas el sistema de control y medida de posición funcionan correctamente.

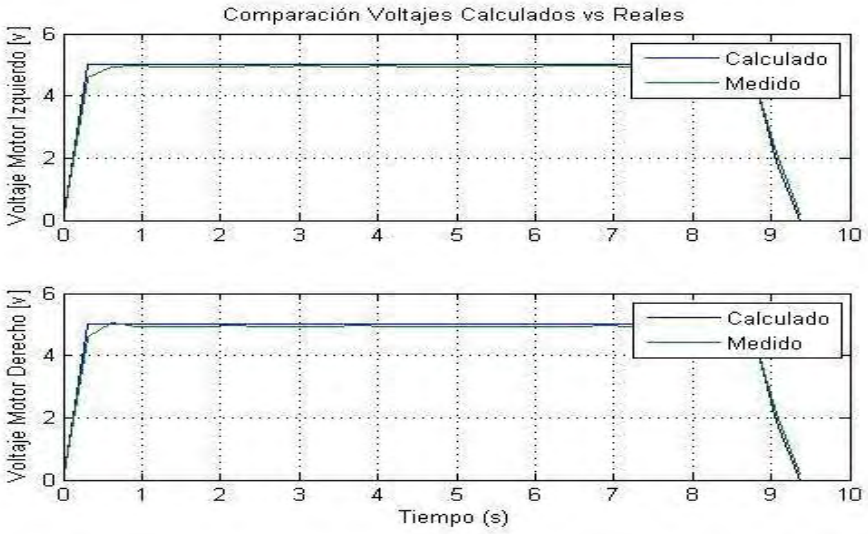
Tabla 11. Comparación distancia recorrida vs programada

Distancia programada [m]	Tiempo [s]	Distancia real recorrida [m]	Error relativo distancia real [%]	Distancia medida con software [m]	Error relativo en la medida del software [%]
0,3	4	0,305	1,667	0,30971	1,544
0,6	5	0,6	0,000	0,60174	0,290
1	10	0,99	1,000	0,99774	0,782
1,2	11	1	0,750	1,19325	0,189
1,5	11	1	1,000	1,48363	0,092
1,8	18	1,78	1,111	1,77902	0,055
2,2	15	2	1,182	2,14902	1,149
Promedio			0,959		0,586

Fuente propia.

En la Figura 45 se muestra el voltaje de alimentación ideal calculado con el método de Pontryagin y el voltaje real que envía el PWM del PIC. Se observan pequeñas diferencias que son las causantes de las imprecisiones tanto en la medida como en el desplazamiento real.

Figura 45. Voltajes reales vs. ideales

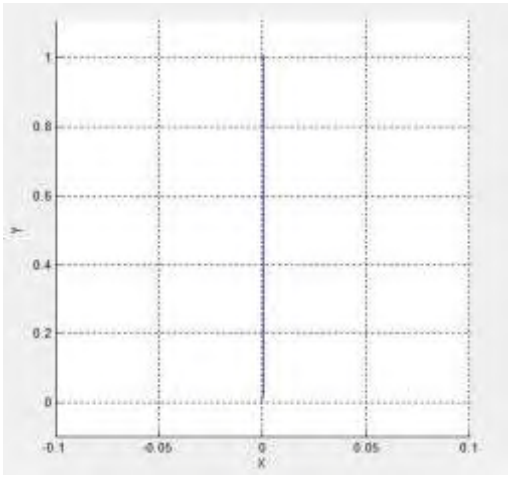


Fuente propia.

La Figura 46 muestra la gráfica de recorrido entregada por el software, calculada a partir de los datos obtenidos desde la IMU y los valores de los voltajes aplicados al

prototipo y simulados en tiempo real con el sistema de espacio de estados previamente diseñado.

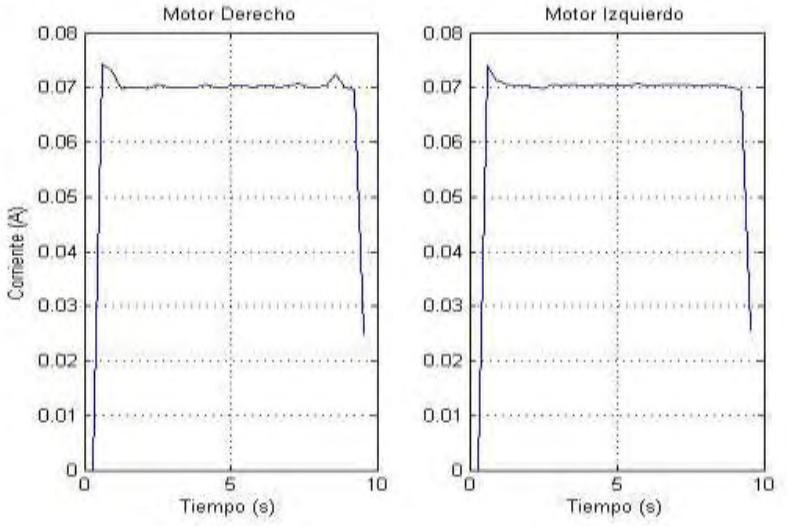
Figura 46. Trayectoria medida de una recta en metros.



Fuente propia.

De forma similar a la gráfica de posición, se calcula un aproximado de la corriente que consumen los motores, usando el modelo en espacio de estados de segundo orden del prototipo y simulando la alimentación del mismo en tiempo real, se tiene un aproximado de la corriente que está siendo consumida por el prototipo.

Figura 47. Corriente calculada en los motores



Fuente propia.

9.2 PRUEBAS EN CURVA

Se usan diferentes condiciones de contorno con el algoritmo de curva, para observar el comportamiento del prototipo en cada caso. Para lograr las condiciones de meta en estas pruebas se programa realizar un giro de 180°, con un radio constante de 0.3 metros a diferentes tiempos. Los resultados se muestran en la tabla 12.

Tabla 12. Medida del Angulo de giro en diferentes tiempos

Tiempo programado [s]	**Angulo Real[°]**	**Error relativo Angulo real [%]**	**Angulo medido en software [°]**	**Error relativo En la medida del software [%]**
6	178	1,111	175,367	1,479
7	178	1,111	174,686	1,862
8	179	0,556	177,183	1,015
9	180	0,000	175,993	2,226
10	180	0,000	176,249	2,084
11	182	1,111	178,223	2,075
Promedio		0,648		1,790

Fuente propia.

La segunda prueba que se realizo para observar el comportamiento del prototipo se hiso bajo las condiciones de un radio constante de 0,5, un ángulo de giro variable entre 30° y 180° con pasos de 5° y un tiempo límite variable. Los resultados se muestran en la tabla 13.

Tabla 13. Comparación Ángulo Barrido Vs. Programado

Desplazamiento angular [°]	Tiempo programado	Angulo real del giro [°]	Error relativo ángulo real [%]	Angulo medido con software [m]	Error relativo software [%]
30	3	30	0,00	29,3783	2,072
45	3	45	0,00	43,2573	3,873
60	4	59	1,67	57,0152	3,364
75	4	74	1,33	72,9745	1,386
90	5	88	2,22	87,8647	0,154
135	10	134	0,74	131,8463	1,607
180	15	180	0,00	177,2937	1,504
promedio			0,85		1,994

Fuente propia.

En las tablas 12 y 13 se observa que bajo las condiciones planteadas las soluciones implementadas funcionan correctamente al obtener errores inferiores al 2% tanto en el angulo real barrido como en el angulo que miden los sensores.

9.3 PRUEBAS EN CIRCUITO CERRADO

La prueba en circuito cerrado se realizó en una pista previamente diseñada la cual posee las características para poner a prueba todos los casos para los que fue diseñado el control óptimo: recta, recta con pendiente positiva, recta con pendiente negativa y curva de radio constante.

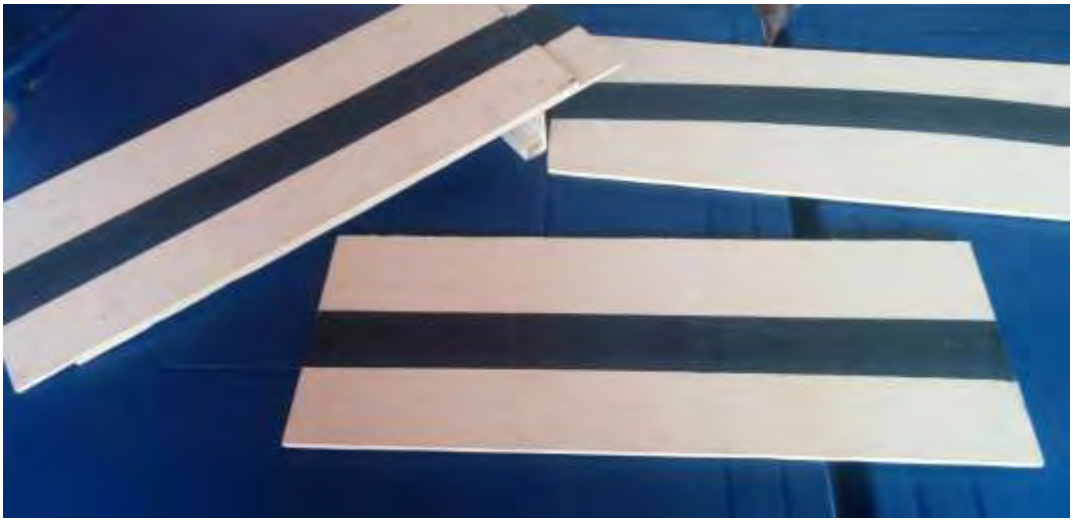
Las características del circuito diseñado son:

- una recta de 6 metros
- dos curvas de radio constate y giro de 180°
- una recta con pendiente de 6° y distancia 3.02 metros
- una recta con pendiente de -6° y distancia 3.02 metros

Las cuales conforman un circuito cerrado.

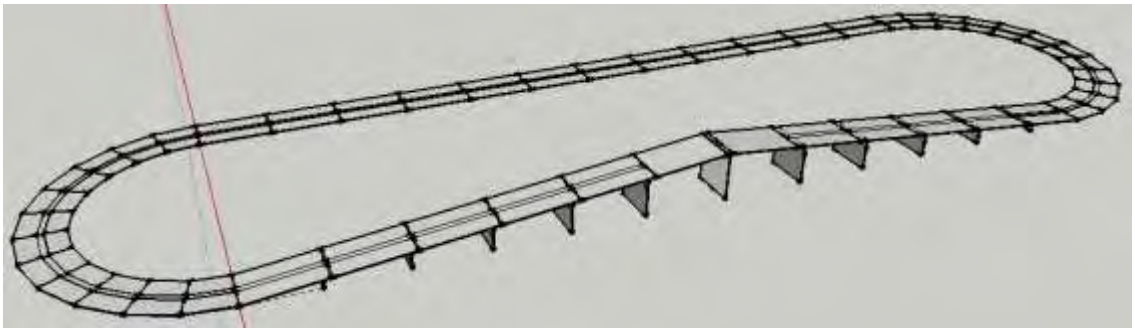
Se diseñó el circuito cerrado con las características citadas anteriormente, en pequeñas piezas de madeflex que se pueden ensamblar, para facilitar el transporte. La figura 48 muestra algunas piezas y la figura 49 muestra la pista completamente armada y el diseño en sketchup.

Figura 48. Piezas de la pista



Fuente propia.

Figura 49. a) Diseño en CAD y b) Pista completa ensamblada



a)

b)

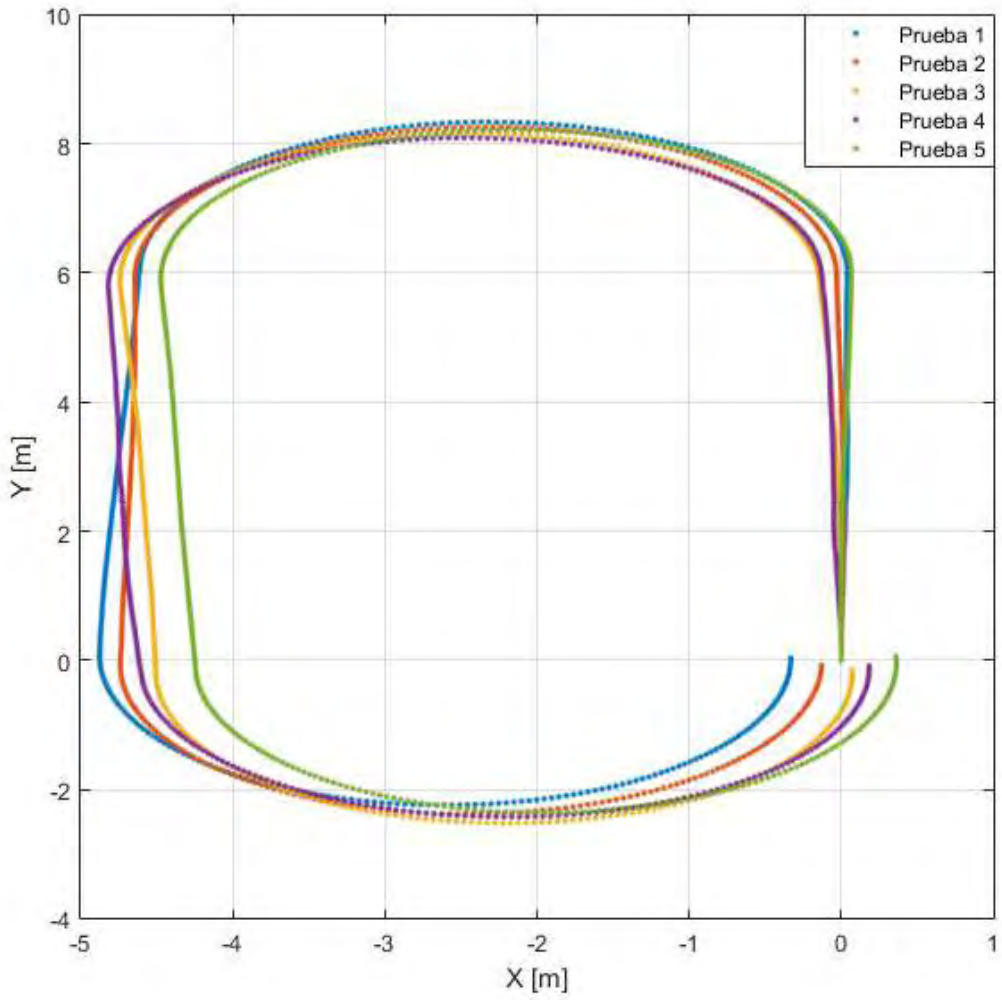
Fuente propia.

El diseño del control se realizó asumiendo algunas condiciones ideales para simplificar su complejidad, pero dichas condiciones no se cumplen en la realidad, para lo cual se implementa un circuito seguidor de línea que proporciona una corrección en pequeñas desviaciones existentes en la trayectoria del prototipo, simulando de esta forma la corrección de la dirección que un conductor controla con el volante de un vehículo real.

Se programan las características del circuito usando el software y se calibran los sensores en el punto de partida de la pista; una vez se ha terminado el cálculo de los voltajes de alimentación óptimos y el almacenamiento de estos datos en la memoria, proceso que dura aproximadamente 10 minutos; enseguida se da la orden de inicio. Se observa que existe una leve desviación hacia la derecha que es corregida con el seguidor de línea; el prototipo recorre cada trayecto según lo esperado con pequeños errores en la distancia final en cada segmento; al llegar al final de la pista el prototipo se detiene a 38 cm tras la meta programada. Se observa un error bastante pequeño comparado con el perímetro de la pista diseñada que es de 18.6 m. La experiencia completa se muestra en un video adjunto.

En la Figura 50, se muestra la dispersión en un circuito cerrado recorrido 5 veces visto desde arriba y graficada por el software; se observa que es muy similar a la trayetoria real, demostrando que bajo las condiciones implementadas, la fusión entre sensores de la IMU y la simulación en tiempo real del sistema obtenemos una buena aproximación del comportamiento que tuvo el prototipo a través del recorrido.

Figura 50. Gráfica trayectoria recorrida por el prototipo



Fuente propia.

10 CONCLUSIONES

En un trayecto a recorrer con condiciones cercanas a las ideales, se puede implementar una simulación del vehículo, que junto con los datos de una IMU de baja gama y filtros Kalman, estimaría el estado actual del vehículo donde se observa: posición, orientación y estado de carga de la batería del mismo en todo momento del trayecto de una forma fiable y a un bajo costo. La medida de la distancia recorrida tiene un error de 2.2% o aproximadamente 2cm por cada metro recorrido. Además, se gestiona la información sin presentar problemas de sincronización o retardos excesivos en el tiempo de muestreo de las señales, almacenando la información proveniente tanto de la tarjeta GY85 como de los voltímetros implementados para medir la tensión en los motores.

Implementando el método de mínimo de Pontryagin se diseñaron algoritmos capaces de calcular una señal de alimentación del sistema para cumplir con los requerimientos previamente especificados de recorrer una trayectoria en un tiempo predeterminado con un gasto de energía óptimo. En la implementación se demostró que los modelos y los métodos de control diseñados funcionan correctamente al observar que el comportamiento del prototipo se aproxima al deseado con la solución implementada. La exactitud con la que el prototipo realiza las trayectorias diseñadas se puede mejorar dependiendo de la precisión al recorrer la trayectoria calculada y la similitud entre los motores del prototipo.

El proyecto se realizó teniendo en cuenta varias aproximaciones y condiciones ideales, que se lograron replicar en las pruebas reales y funcionaron correctamente en el prototipo pequeño de VE usado; sin embargo, al implementar esta solución en un vehiculo real se deben tener en cuenta varias variables que aumentarán la complejidad de los algoritmos de optimización como son: las oposiciones al movimiento por el rozamiento, una trayectoria mas compleja, la dinámica lateral del vehiculo, el uso de un motor BLDC de un VE real o la implementación de un control en tiempo real teniendo en cuenta eventos no contemplados y de carácter no predecible. Todas estas y muchas otras condiciones aumentarán la dificultad de la solución del problema y se pueden proyectar para trabajos futuros. Con respecto a la medición de las señales del vehiculo se pueden implementar otro tipo de sensores como GPS y sensores de mayor gama como acelerómetros marca MEMS, que junto a las técnicas de fusión de sensores aumentarán considerablemente la precisión en las medidas.

11 REFERENCIAS

[1] S. Stockar, V. Marano, M. Canova, G. Rizzoni y L. Guzzella, «Energy-Optimal Control of Plug-in Hybrid Electric Vehicles for Real-World Driving Cycles,» *Vehicular Technology, IEEE Transactions on,* vol. 60, nº 7, pp. 2949-2962, Sept. 2011.

[2] A. Ceña y J. Santamarta, «Reve: Revista eólica y del vehículo eléctrico,» 23 Marzo 2009. [En línea]. Available: http://www.evwind.com/2009/03/23/el-coche-electrico-el-futuro-del-transporte-la-energia-y-el-medio-ambiente-por-alberto-cena-y-jose-santamarta/. [Último acceso: 15 04 2013].

[3] B. Keoun, «Designing an electric vehicle conversion,» de *Southcon/95. Conference Record*, Mar. 1995.

[4] «El Espectador,» 2 Febrero 2011. [En línea]. Available: http://www.elespectador.com/impreso/vivir/articulo-248645-dos-ingenieros-bogotanos-disenan-y-fabrican-carros-electricos. [Último acceso: 25 Abril 2013].

[5] «i-Miev: Mitsubishi Innovative Electric Vehicle,» 2012. [En línea]. Available: http://www.imiev.es/. [Último acceso: 25 Abril 2013].

[6] «BYD Build Your Dreams,» Febrero 2013. [En línea]. Available: http://www.byd.com/sa/index.html. [Último acceso: 20 Abril 2013].

[7] «El Tiempo,» 12 Diciembre 2012. [En línea]. Available: http://www.eltiempo.com/archivo/documento/CMS-12445526. [Último acceso: 10 Abril 2013].

[8] D. E. Kirk, Optimal Control Theory: An Introduction, Mineola, New York: Dover Publications, 2004.

[9] N. Kim, S. Cha y H. Peng, «Optimal Control of Hybrid Electric Vehicles Based on Pontryagin's Minimum Principle,» *Control Systems Technology, IEEE Transactions on,* vol. 19, nº 5, pp. 1279-1287, Sept. 2011.

[10] H. Borhan, A. Vahidi, A. Phillips, M. Kuang, I. Kolmanovsky y S. Di Cairano, «MPC-Based Energy Management of a Power-Split Hybrid Electric Vehicle,» *Control Systems Technology, IEEE Transactions on,* vol. 20, nº 3, pp. 593 - 603, May 2012.

[11] B.-R. Liang y W.-S. Lin, «Optimal regenerative torque control to maximize energy recapture of electric vehicles,» de *World Automation Congress (WAC), 2010*, 2010.

[12] ALLEGRO MICROSYSTEM, LLC, Automotive Grade, Fully Integrated, Hall Effect-Based Linear Current Sensor IC with 2.1 kVRMS Voltage Isolation and a Low-Resistance Current Conductor. p2.

[13] ANALOG DEVICES, *±2 g/±4 g/±8 g/±16 g* Three Axis Digital Accelerometer.

[14] Sistemas de Adquisición Y Procesamiento De Datos. [En línea]. http://rua.ua.es/dspace/bitstream/10045/19119/1/Sistemas%20de%20adquisici%C3%B3n%20y%20Procesamiento%20de%20datos.pdf. [Citado el 17 de septiembre de 2014].

[15] FRANCO, Sergio. Diseño Con Amplificadores Operacionales y Circuitos Integrados Analógicos. 3 ed. México D. F. Mc Graw Hill.

[16] GARCÍA FREIJO, Eduardo. Compilador C CCS y Simulador Proteus Para Microcontroladores PIC.1 ed. México D. F. Alfaomega Grupo Editor.

[17] POZO ESPÍN, David Fernando. et al. Diseño y construcción de una plataforma didáctica para medir ángulos de inclinación usando sensores inerciales como acelerómetro y giroscopio. Quito, febrero 2010. 132p. Trabajo de grado (Ingeniero en electrónica y control). Escuela Politécnica Nacional. Facultad de Ingeniería Electrica y Electrónica.

[18] VALENZUELA SALAMANCA, Francisco Jose. Modelado, Simulación Y Puesta En Marcha De Una Bancada De Máquinas De Imanes Permanentes. Trabajo de grado (Ingeniero Industrial) Universidad de Sevilla.

[19] OGATA, Katsuhiko. Ingenieria de Control Moderna, 2010

[20] M.-H. Felix y G. Albaro. Modelo Lineal De Un Motor De Corriente Continua. Abril 2012.

[21] D. Subbaram Naidu. Optimal Control Systems. 2003.

[22] T. Jose, L. Javier, G. Antonio y R. Francisco. Estudio Dinámico De Un Vehículo Eléctrico Mediante Simmechanics. Universidad de Almería, España.

[23] S.-G. Leonardo, M.-V. Manuel y R.-V. Edgar. Seguimiento De Trayectorias Con Un Robot Móvil De Configuración Diferencial. Universidad Militar Nueva Granada Junio 2014.

[24] Q.-O. Manuel y H.-C. Carlos. Obtención Experimental De Los Parámetros Del Motor Que Se Utilizará En El Sistema De Locomoción De Una Esfera Rodante. Facultad Ingeniería Electrónica. Universidad Pontificada Bolivariana. 2009.

[25] S.- S. Salvador, C.- Q. Miguel y G.- B. Rogelio. Determinación de los parámetros de un motor de cd por medición física directa. Instituto De Electrónica Y Mecatrónica. Universidad Tecnológica De La Mixteca.2014.

[26] P.-S. Kim, «Cost modeling of battery electric vehicle and hybrid electric vehicle based on major parts cost,» de *Power Electronics and Drive Systems, 2003. PEDS 2003. The Fifth International Conference on*, 2003.

[27] M. Marchesoni y C. Vacca, «New DC–DC Converter for Energy Storage System Interfacing in Fuel Cell Hybrid Electric Vehicles,» *Power Electronics, IEEE Transactions on,* vol. 22, nº 1, pp. 301-308, Jan. 2007.

[28] J. Traube, F. Lu y D. Maksimovic, «Electric vehicle DC charger integrated within a photovoltaic power system,» de *Applied Power Electronics Conference and Exposition (APEC), 2012 Twenty-Seventh Annual IEEE*, 2012.

[29] D. Bellur y M. Kazimierczuk, «DC-DC converters for electric vehicle applications,» de *Electrical Insulation Conference and Electrical Manufacturing Expo, 2007*, 2007.

[30] «MathWorks Accelerating the pace of engineering and science,» 2013. [En línea]. Available: http://www.mathworks.es/. [Último acceso: 2013].

[31] T. Sweda y D. Klabjan, «Finding minimum-cost paths for electric vehicles,» de *Electric Vehicle Conference (IEVC), 2012 IEEE International*, 2012.

[32] A. Bedir y A. Alouani, «A simple power based control strategy for hybrid electric vehicles,» de *Vehicle Power and Propulsion Conference, 2009. VPPC '09. IEEE*, 2009.

[33] N. Kim, S. W. Cha y H. Peng, «Optimal Equivalent Fuel Consumption for Hybrid Electric Vehicles,» *Control Systems Technology, IEEE Transactions on,* vol. 20, nº 3, pp. 817-825, May 2012.

[34] «Digi Your M2M Expert,» 2013. [En línea]. Available: http://www.digi.com. [Último acceso: 2013].

[35] W. Zhou, C. Zhao y J. Guo, «The Study of Improving Kalman Filters Family for Nonlinear SLAM,» *Journal of Intelligent and Robotic Systems,* vol. 56, nº 5, pp. 543-564, 2009.

[36] L. Meng, L. Li y S. Veres, «Aerodynamic parameter estimation of an Unmanned Aerial Vehicle based on extended kalman filter and its higher order approach,» de *Advanced Computer Control (ICACC), 2010 2nd International Conference on*, 2010.

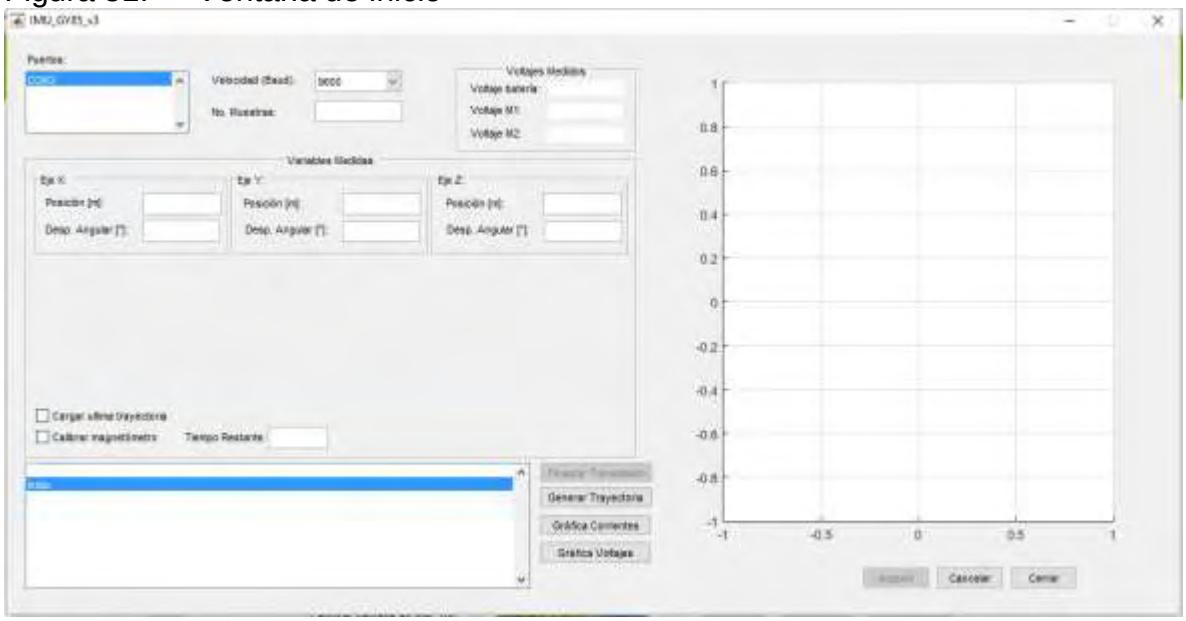
[37] Universidad de Nariño, *Acuerdo 005,* San Juan de Pasto, Nariño, 2010.

ANEXO A. DESCRIPCIÓN DEL SOFTWARE

El software es diseñado con la herramienta de creación de interfaz gráfica de Matlab generando un archivo con el código fuente de ejecución que se muestra en el anexo B.

En la figura 51 se presenta la ventana principal de la interfaz, que es observada al ejecutar el código fuente. En ella se muestran los datos de los sensores y las diferentes opciones de configuración y visualización.

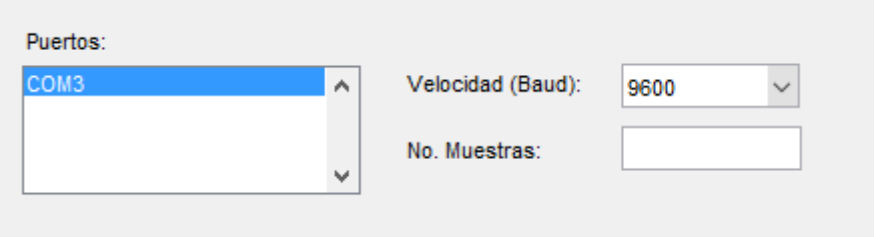
Figura 51. Ventana de inicio



Fuente propia.

El primer paso a seguir es la selección y configuración del puerto a usar. El software diseñado detecta todos los puertos RS232 disponibles en el computador para ser seleccionados por el usuario y las configuraciones básicas de velocidad de trasmisión. Estas configuraciones son manipuladas por el usuario en la ventana mostrada en la Figura 52. Se selecciona el puerto al que está enlazado el Xbee y se configura a la velocidad de transmisión con el mismo valor de la programada en el microcontrolador. En esta sección también se encuentra un espacio para rellenar en el que se configura el número de muestras que se tomarán al calibrar los giroscopios.

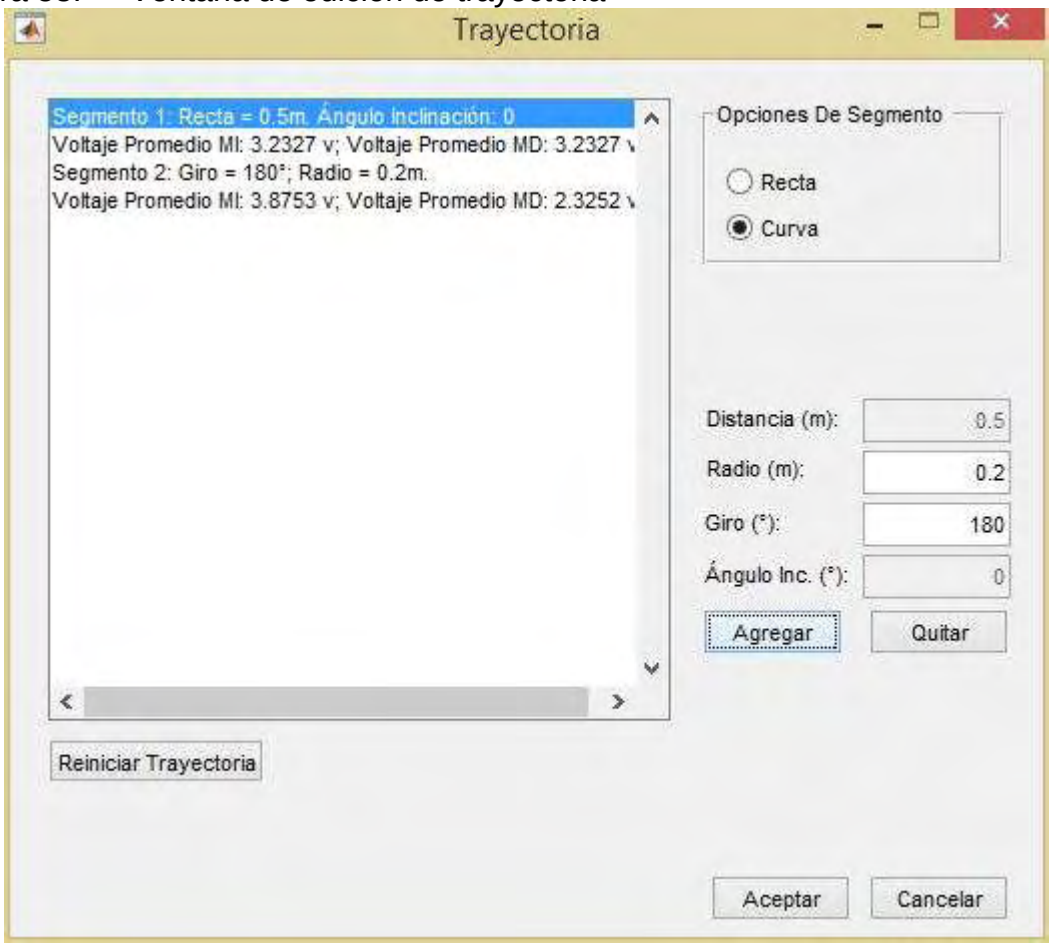
Figura 52. Sección de Puertos



Fuente propia.

Una vez configurados los puertos, se presiona el botón “generar trayectoria”; al hacerlo emerge una nueva ventana. Esta ventana es donde el usuario crea la trayectoria; para el proyecto se elige entre dos tipos de trayectoria: recta y curva de radio constante.

Figura 53. Ventana de edición de trayectoria

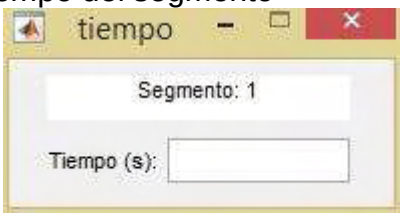


Fuente propia.

Se selecciona una opción, se definen las características de ese segmento y se da agregar; al hacerlo aparece una pequeña ventana que implementa la última característica necesaria, el tiempo que debe durar el trayecto. Con los datos introducidos los algoritmos de optimización realzan los cálculos necesarios para generar las entradas óptimas del sistema. Se repite esta acción para cada segmento del recorrido hasta tener diseñada la trayectoria total, pudiendo introducir varios segmentos de diferentes características para obtener un recorrido final con muchas formas posibles. En la parte izquierda de la ventana se mostrará un resumen de las características de la sección y los voltajes promedio que tiene para cada uno de los motores.

El botón reiniciar trayectoria, borra los segmentos previamente introducidos lo que permite diseñar una trayectoria nueva desde cero.

Figura 54. Ventana de tiempo del segmento



Fuente propia.

Una vez finalizada la creación de la trayectoria se oprime el botón “aceptar”, acción que enlaza el computador con el microcontrolador enviando los datos de las entradas óptimas de voltaje para cada instante de tiempo hacia la memoria EEPROM, donde se almacenan esperando la orden de inicio.

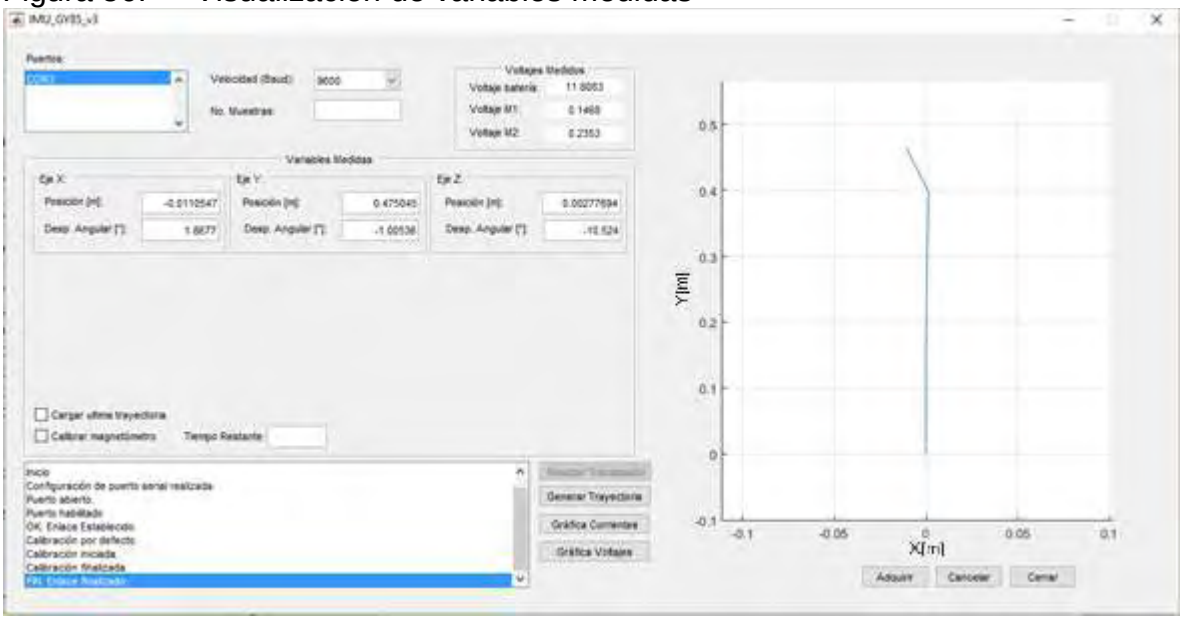
La sección de estado muestra cada uno de los pasos que se sigue en los procesos ejecutados. Esta sección está ubicada en la parte inferior izquierda de la ventana principal de la interfaz.

Figura 55. Sección de estado

Fuente propia.

El último paso, es presionar el botón “adquirir”. Haciendo esto, el algoritmo envía la orden de inicio al microcontrolador, con lo que éste empieza a enviar el voltaje en los motores según los que tiene almacenados en la memoria EEPROM y a transmitir al computador los datos de las variables medidas por los diferentes sensores implementados en la tarjeta diseñada. Estos datos se muestran en las figuras 56 y 57. La figura 56 presenta principalmente los datos de posicionamiento del prototipo mientras la 57, los datos de los voltajes de la batería y voltajes que llegan a los motores.

Figura 56. Visualización de variables medidas



Fuente propia.

En la sección de variables medidas se puede habilitar la calibración del magnetómetro con un check, acción que se debe realizar cada vez que se varíe el lugar donde se realizan las pruebas.

Figura 57. Visualización de voltajes medidos

Fuente propia.

Los datos estimados de posición y orientación se muestran en la ventana principal del software como una trayectoria en tiempo real en la sección derecha de la Figura 51.

ANEXO B. CÓDIGO EN MATLAB DE LA VENTANA PRINCIPAL

```
function varargout = IMU_GY85_v3(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @IMU_GY85_v3_OpeningFcn, ...
                   'gui_OutputFcn',  @IMU_GY85_v3_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
function IMU_GY85_v3_OpeningFcn(hObject, eventdata, handles, varargin)
global vectorTiempoSimulacion voltajesMotores

voltajesMotores=[0 0]';
vectorTiempoSimulacion=[0 0];

instrreset
puertosDisponibles = instrhwinfo('serial');


puertosDisponibles = puertosDisponibles.AvailableSerialPorts;
set(handles.puertos,'String',puertosDisponibles);
set(handles.graficaPista,'XGrid','on','YGrid','on','XLim',[-1 1]...
    ,'YLim',[-1 1],'Units','centimeters');
cla(handles.graficaPista)
datacursormode on
clearvars -global
dato_num=1;
set(handles.adquirir,'Enable','off');
% ***CODIGO DE INICIO DE INFO DEL SISTEMA***
[infoSistema,dato_num]=infoSys('','Inicio',dato_num);
set(handles.sistema,'String',infoSistema);
set(handles.sistema,'Value',dato_num);
handles.infoSistema=infoSistema;
handles.dato_num=dato_num;
handles.output = hObject;
guidata(hObject, handles);
% CÁLCULO FILTROS IMPLEMENTADOS
function varargout = IMU_GY85_v3_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;
function cancelar_Callback(hObject, eventdata, handles)
instrreset
puertoCom=get(handles.puertos,'String');
puertoCom=char(puertoCom(1));
```

```
velocidad=get(handles.baudios,'String');
velocidad=str2double(char(velocidad(get(handles.baudios,'Value'))));
puerto =
serial(puertoCom,'baudrate',velocidad,'Timeout',5,'Timerperiod',3,'StopBi
ts',1);
fopen(puerto);
fprintf(puerto,'%s',['4' 13]);
instrreset
close(gcbf)
IMU_GY85_v3
function baudios_Callback(hObject, eventdata, handles)
function puertos_Callback(hObject, eventdata, handles)
function sistema_Callback(hObject, eventdata, handles)
function adquirir_Callback(hObject, eventdata, handles)
%     Configuracion variables de reconocimiento y finalizacion de
transmision
    global vs T sisc anguloInclinacion c Gr M %voltajesMotores

if get(handles.ultimaTrayectoria,'Value')
    load T
    load sisc
    load anguloInclinacion
    load c
    load Gr
    load M
    load voltajesSimulacion
    load nVoltajes
end
% CONFIGURACIÓN DE PUERTO Y APERTURA
com = get(handles.puertos,'String');
item_com = get(handles.puertos,'Value');
baud = get(handles.baudios,'String');
item_baud = get(handles.baudios,'Value');
com = com(item_com);
baud = str2double(baud(item_baud));
pause(0.5);
puerto = serial(com,...
                'baudrate',baud,...
                'Timeout',7,...
                'Timerperiod',7,...
                'StopBits',1);
[handles.infoSistema,handles.dato_num]=infoSys(handles.infoSistema,'Confi
guración de puerto serial realizada',handles.dato_num);
set(handles.sistema,'String',handles.infoSistema);
set(handles.sistema,'Value',handles.dato_num);
pause(0.5);

fopen(puerto);

[handles.infoSistema,handles.dato_num]=infoSys(handles.infoSistema,'Puert
o abierto.',handles.dato_num);
set(handles.sistema,'String',handles.infoSistema);
set(handles.sistema,'Value',handles.dato_num);
pause(0.5);
```

```
[handles.infoSistema,handles.dato_num] =
infoSys(handles.infoSistema,'Puerto habilitado',handles.dato_num);
set(handles.sistema,'String',handles.infoSistema);
set(handles.sistema,'Value',handles.dato_num);
pause(0.5);

handles.puerto=puerto;
% *********************************
if get(handles.ultimaTrayectoria,'Value')
    nVoltajes=2*(nVoltajes-1);
    fprintf(puerto,'%s',['C ' num2str(nVoltajes) 13]);
    pause(0.1)
end
    ack = ' ';
    while strcmp(ack,'OK')~=1
        fprintf(puerto,'%s',['6' 13]);
        ack = fscanf(puerto,'%s');
    end
    [handles.infoSistema,handles.dato_num] =
infoSys(handles.infoSistema,[ack ': Enlace
Establecido.'],handles.dato_num);
    set(handles.sistema,'String',handles.infoSistema);
    set(handles.sistema,'Value',handles.dato_num);
    pause(0.5);

    set(handles.finalizar,'Enable','on','Value',0);
    set(handles.adquirir,'Enable','off');
    set(handles.cancelar,'Enable','off');
    set(handles.cerrar,'Enable','off');

    muestra = 1;

    ax=[0 0];
    ay=[0 0];
    az=[0 0];

    vx=[0 0];
    vy=[0 0];
    vz=[0 0];

    wx=[0 0];
    wy=[0 0];
    wz=[0 0];

    x=[0 0];
    y=[0 0];
    z=[0 0];

    gx=[0 0];
    gy=[0 0];
    gz=[0 0];
```

```
    cx=[0 0];
    cy=[0 0];
    cz=[0 0];

    tiempoReal=[0 0];

    media_ax=[0 0];
    media_ay=[0 0];
    media_az=[0 0];
    media_wx=[0 0];
    media_wy=[0 0];
    media_wz=[0 0];

    ax_sinOffset=[0 0];
    ay_sinOffset=[0 0];
    az_sinOffset=[0 0];
    wx_sinOffset=[0 0];
    wy_sinOffset=[0 0];
    wz_sinOffset=[0 0];

    vs=[0 0];
    vMotor1=[0 0];
    vMotor2=[0 0];
    voltajeEntero1=[0 0];
    voltajeEntero2=[0 0];

    voltajesMedidos=[0 0;0 0;0 0];
    tiempoSim=[0 0];
    estados_adquisicion=0;

    acMotor1=[0 0];
    acMotor2=[0 0];

    anguloXMag=[0 0];
    anguloYMag=[0 0];
    anguloZMag=[0 0];
    offsetX=0;
    offsetY=0;
    offsetZ=0;
    desplazamiento_MagZ=0;
    XmasGuardada=[0 0 0 0 0 0]';
% MATRICES GIROSCOPIO
    A=[1    -T  0   0   0   0;
        0   1   0   0   0   0;
        0   0   1   -T  0   0;
        0   0   0   1   0   0;
        0   0   0   0   1   -T;
        0   0   0   0   0   1];
    B=[T 0 0;
        0 0 0;
        0 T 0;
        0 0 0;
```

```
        0 0 T;
        0 0 0];
    Xmas=[0 0 0 0 0 0]';
    Rwz=0.0001;
    Rwy=0.0001;
    Rwx=0.0002;
% ******************
Qwz=[0.001 0 0  0   0   0;
    0 0.003 0   0   0   0;
    0   0   0.001 0 0   0;
    0   0   0 0.003 0   0;
    0   0   0   0   0.001   0;
    0   0   0   0   0   0.003];
I=eye(size(A));
Pmas=[2 0 0 0 0 0;
    0 2 0 0 0 0;
    0 0 2 0 0 0;
    0 0 0 2 0 0;
    0 0 0 0 2 0;
    0 0 0 0 0 2];
H=[1 0 1 0 1 0];
roll=[0 0];
pitch=[0 0];
xSim=[0 0];
ySim=[0 0];
zSim=[0 0];
try
    while ~get(handles.finalizar,'Value')
    if estados_adquisicion==2&&muestra==1
        fprintf(handles.puerto,'%s',13);
    end
        muestraPic=str2double(fscanf(handles.puerto,'%s'));

        if isnan(muestraPic)
            break
        end
        voltajeEntero1(muestra)=fscanf(handles.puerto,'%li');
        voltajeEntero2(muestra)=fscanf(handles.puerto,'%li');
        vs(muestra)=fscanf(handles.puerto,'%f');
        vMotor1(muestra)=fscanf(handles.puerto,'%f');
        vMotor2(muestra)=fscanf(handles.puerto,'%f');
        ax(muestra)=fscanf(handles.puerto,'%f');
        ay(muestra)=fscanf(handles.puerto,'%f');
        az(muestra)=fscanf(handles.puerto,'%f');
        wx(muestra)=fscanf(handles.puerto,'%f');
        wy(muestra)=fscanf(handles.puerto,'%f');
        wz(muestra)=fscanf(handles.puerto,'%f');
        cx(muestra)=fscanf(handles.puerto,'%f')-offsetX;
        cy(muestra)=fscanf(handles.puerto,'%f')-offsetY;
        cz(muestra)=fscanf(handles.puerto,'%f')-offsetZ;
        tiempoMuestreo=fscanf(handles.puerto,'%f')*1e-6;
        sobrePasos=fscanf(handles.puerto,'%u');
        ACK=fscanf(handles.puerto,'%s');
        anguloZMag(muestra)=rad2deg(atan2(cy(muestra),cx(muestra)));
```

```
        anguloXMag(muestra)=rad2deg(atan2(cz(muestra),cy(muestra)));
        tiempoMuestreo=tiempoMuestreo+sobrePasos*65536*0.2e-6;

        if anguloZMag(muestra)<0
            anguloZMag(muestra)=anguloZMag(muestra)+360;
        end
        if anguloXMag(muestra)<0
            anguloXMag(muestra)=anguloXMag(muestra)+360;
        end
%         ACTUALIZAMOS LOS VALORES DE OFFSET
        if estados_adquisicion==0
            media_ax(muestra)=median(ax);
            media_ay(muestra)=median(ay);
            media_az(muestra)=median(az);
            media_wx(muestra)=median(wx);
            media_wy(muestra)=median(wy);
            media_wz(muestra)=median(wz);
        end
% ***************************************
        ax_sinOffset(muestra)=ax(muestra)-media_ax(end);
        ay_sinOffset(muestra)=ay(muestra)-media_ay(end);
        az_sinOffset(muestra)=az(muestra);%-media_az(end);
        wx_sinOffset(muestra)=wx(muestra)-media_wx(end);
        wy_sinOffset(muestra)=wy(muestra)-media_wy(end);
        wz_sinOffset(muestra)=wz(muestra)-media_wz(end);


        set(handles.voltajeBateria,'String',num2str(mean(vs(end))));
        set(handles.voltajeMotor1,'String',num2str(vMotor1(end)));
        set(handles.voltajeMotor2,'String',num2str(vMotor2(end)));
        if ax_sinOffset(end)>=-0.05 && ax_sinOffset(end)<=0.05
            ax_sinOffset(end)=0;
        end
        if ay_sinOffset(end)>=-0.008 && ay_sinOffset(end)<=0.008
            ay_sinOffset(end)=0;
        end
    %     VENTANAS DE DISCRIMINACIÓN
        if wx_sinOffset(end)>=-0.075 && wx_sinOffset(end)<=0.075
            wx_sinOffset(end)=0;
        end
        if wy_sinOffset(end)>=-0.075 && wy_sinOffset(end)<=0.075
            wy_sinOffset(end)=0;
        end
    %     *************************
        if tiempoReal(end)<=2
            ax_sinOffset(end)=0;
            az_sinOffset(end)=0;
            wx_sinOffset(end)=0;
            wy_sinOffset(end)=0;
        end
%     *************************
        switch estados_adquisicion
            case 0
                if muestra == 1
```

```
                    muestras =
str2double(get(handles.muestras,'String'));
                    if isnan(muestras)
                        [handles.infoSistema,handles.dato_num] =
infoSys(handles.infoSistema,'Calibración por defecto.',handles.dato_num);

set(handles.sistema,'String',handles.infoSistema);
                        set(handles.sistema,'Value',handles.dato_num);
                        pause(0.5);
                        muestras=30;
                    end
                    barraDeProgreso=waitbar(1/muestras,'Calibrando...');
                    [handles.infoSistema,handles.dato_num] =
infoSys(handles.infoSistema,'Calibración iniciada.',handles.dato_num);
                    set(handles.sistema,'String',handles.infoSistema);
                    set(handles.sistema,'Value',handles.dato_num);
                    pause(0.5);
                end
                waitbar(muestra/muestras,barraDeProgreso);
                if muestra >= muestras

                    muestra=0;
                    if get(handles.calibrarMag,'Value')
                        estados_adquisicion = estados_adquisicion + 1;
                    else
                        estados_adquisicion = estados_adquisicion + 2;
                        fprintf(handles.puerto,'%s','1');
                        load offsetX
                        load offsetY
                        load offsetZ
                    end
                    [handles.infoSistema,handles.dato_num] =
infoSys(handles.infoSistema,'Calibración finalizada.',handles.dato_num);
                    set(handles.sistema,'String',handles.infoSistema);
                    set(handles.sistema,'Value',handles.dato_num);
                    close(barraDeProgreso);
                    pause(0.5);

                ax=[0 0];
                ay=[0 0];
                az=[0 0];

                vx=[0 0];
                vy=[0 0];
                vz=[0 0];

                wx=[0 0];
                wy=[0 0];
                wz=[0 0];

                x=[0 0];
                y=[0 0];
                z=[0 0];
```

```
                gx=[0 0];
                gy=[0 0];
                gz=[0 0];

                tiempoReal=0;

                ax_sinOffset=[0 0];
                ay_sinOffset=[0 0];
                az_sinOffset=[0 0];
                wx_sinOffset=[0 0];
                wy_sinOffset=[0 0];
                wz_sinOffset=[0 0];

                vs=[0 0];
                vMotor1=[0 0];
                vMotor2=[0 0];

                anguloZMag=[];
                end
            case 1
                if muestra==1
                [handles.infoSistema,handles.dato_num] =
infoSys(handles.infoSistema,'Calibración de magnetómetro
iniciada.',handles.dato_num);
                set(handles.sistema,'String',handles.infoSistema);
                set(handles.sistema,'Value',handles.dato_num);
                tic
                end
                tiempoCalMagnetometro=44-toc;
                plot(cx,cy);grid on;

set(handles.tiempoMag,'String',num2str(tiempoCalMagnetometro));
                if muestra>=muestras+100
                    estados_adquisicion=estados_adquisicion+1;
                    offsetX=max(cx)-(max(cx)-min(cx))/2;
                    offsetY=max(cy)-(max(cy)-min(cy))/2;
                    offsetZ=max(cz)-(max(cz)-min(cz))/2;
                    ax=[0 0];
                    ay=[0 0];
                    az=[0 0];

                    vx=[0 0];
                    vy=[0 0];
                    vz=[0 0];

                    wx=[0 0];
                    wy=[0 0];
                    wz=[0 0];

                    x=[0 0];
                    y=[0 0];
                    z=[0 0];
```

```
                    gx=[0 0];
                    gy=[0 0];
                    gz=[0 0];

                    tiempoReal=0;

                    ax_sinOffset=[0 0];
                    ay_sinOffset=[0 0];
                    az_sinOffset=[0 0];
                    wx_sinOffset=[0 0];
                    wy_sinOffset=[0 0];
                    wz_sinOffset=[0 0];

                    vs=[0 0];
                    vMotor1=[0 0];
                    vMotor2=[0 0];

                    anguloZMag=[];
                    muestra=0;
                    [handles.infoSistema,handles.dato_num] =
infoSys(handles.infoSistema,'Calibración de magnetómetro
finalizada.',handles.dato_num);
                    set(handles.sistema,'String',handles.infoSistema);
                    set(handles.sistema,'Value',handles.dato_num);
                    set(handles.tiempoMag,'String',0);
                    fprintf(handles.puerto,'%s','1');
                end
            case 2
            if muestra>=2
%                 ÁNGULOS EN X MAGNETÓMETRO
            if anguloXMag(muestra)>270 && anguloXMag(muestra)<360 && ...
                    anguloXMag(muestra-1)>=0 && anguloXMag(muestra-1)<90
                anguloXMag(muestra-1)=360+anguloXMag(muestra-1);
            end
            if anguloXMag(muestra)>=0 && anguloXMag(muestra)<90 && ...
                   anguloXMag(muestra-1)>270 && anguloXMag(muestra-1)<360
                anguloXMag(muestra-1)=anguloXMag(muestra-1)-360;
            end
%               ****************************
%                 ÁNGULOS EN Z MAGNETÓMETRO
            if anguloZMag(muestra)>270 && anguloZMag(muestra)<360 && ...
                    anguloZMag(muestra-1)>=0 && anguloZMag(muestra-1)<90
                anguloZMag(muestra-1)=360+anguloZMag(muestra-1);
            end
            if anguloZMag(muestra)>=0 && anguloZMag(muestra)<90 && ...
                   anguloZMag(muestra-1)>270 && anguloZMag(muestra-1)<360
                anguloZMag(muestra-1)=anguloZMag(muestra-1)-360;
            end
%               ****************************
            desplazamiento_MagZ=desplazamiento_MagZ-(anguloZMag(muestra)-
            anguloZMag(muestra-1));
            tiempoReal(muestra)=tiempoReal(end)+tiempoMuestreo;
            tiempoSim(muestra)=tiempoSim(end)+T;
            if isempty(anguloInclinacion)
```

```
                anguloInclinacion=0;
            end
            Fl=9.8*M*sin(deg2rad(anguloInclinacion));
            vectorPeso=1*Fl*M*ones(1,length(vMotor1));
            voltajesMedidos(:,muestra)=[(vMotor1(muestra))
(vMotor2(muestra)) c*Gr*vectorPeso(muestra)]';
            [ys,~,xs]=lsim(sisc,voltajesMedidos,tiempoSim);
            acMotor1(muestra)=(ys(muestra,1)-ys(muestra-
1,1))/tiempoMuestreo;
            acMotor2(muestra)=(ys(muestra,2)-ys(muestra-
1,2))/tiempoMuestreo;

            %   INTEGRACIÓN ACELERÓMETRO
            vx(muestra)=vx(muestra-
1)+(ax_sinOffset(muestra)+abs(ax_sinOffset(muestra)-ax_sinOffset(muestra-
1))/2)*tiempoMuestreo;
            vz(muestra)=vz(muestra-
1)+(az_sinOffset(muestra)+abs(az_sinOffset(muestra)-az_sinOffset(muestra-
1))/2)*tiempoMuestreo;
            x(muestra)=x(muestra-1)+(vx(muestra)+abs(vx(muestra)-
vx(muestra-1))/2)*tiempoMuestreo;
            z(muestra)=z(muestra-1)+(vz(muestra)+abs(vz(muestra)-
vz(muestra-1))/2)*tiempoMuestreo;
            %     INTEGRACIÓN GIROSCOPIO
                if wx_sinOffset(muestra)>wx_sinOffset(muestra-1)
                    gx(muestra)=gx(muestra-1)+(wx_sinOffset(muestra)-
abs(wx_sinOffset(muestra)-wx_sinOffset(muestra-1))/2)*tiempoMuestreo;
                elseif wx_sinOffset(muestra)==wx_sinOffset(muestra-1)
                    gx(muestra)=gx(muestra-
1)+wx_sinOffset(muestra)*tiempoMuestreo;
                elseif wx_sinOffset(muestra)<wx_sinOffset(muestra-1)
                    gx(muestra)=gx(muestra-
1)+(wx_sinOffset(muestra)+abs(wx_sinOffset(muestra)-wx_sinOffset(muestra-
1))/2)*tiempoMuestreo;
                end

                if wy_sinOffset(muestra)>wy_sinOffset(muestra-1)
                    gy(muestra)=gy(muestra-1)+(wy_sinOffset(muestra)-
abs(wy_sinOffset(muestra)-wy_sinOffset(muestra-1))/2)*tiempoMuestreo;
                elseif wy_sinOffset(muestra)==wy_sinOffset(muestra-1)
                    gy(muestra)=gy(muestra-
1)+wy_sinOffset(muestra)*tiempoMuestreo;
                elseif wy_sinOffset(muestra)<wy_sinOffset(muestra-1)
                    gy(muestra)=gy(muestra-
1)+(wy_sinOffset(muestra)+abs(wy_sinOffset(muestra)-wy_sinOffset(muestra-
1))/2)*tiempoMuestreo;
                end

                if wz_sinOffset(muestra)>wz_sinOffset(muestra-1)
                    gz(muestra)=gz(muestra-1)+(wz_sinOffset(muestra)-
abs(wz_sinOffset(muestra)-wz_sinOffset(muestra-1))/2)*tiempoMuestreo;
                elseif wz_sinOffset(muestra)==wz_sinOffset(muestra-1)
                    gz(muestra)=gz(muestra-
1)+wz_sinOffset(muestra)*tiempoMuestreo;
```

```
                elseif wz_sinOffset(muestra)<wz_sinOffset(muestra-1)
                    gz(muestra)=gz(muestra-
1)+(wz_sinOffset(muestra)+abs(wz_sinOffset(muestra)-wz_sinOffset(muestra-
1))/2)*tiempoMuestreo;
                end
                if gz(muestra)>360||gz(muestra)<-360
                    wz_sinOffset(:)=0;
                    desplazamiento_MagZ=0;
                    gz(:)=0;
                end

%             ***KALMAN***
    Xmenos=A*Xmas+B*[wz_sinOffset(muestra) wy_sinOffset(muestra)
wx_sinOffset(muestra)]';
    Pmenos=A*Pmas*A'+Qwz;

K(1:2)=Pmenos(1:2,1:2)*H(1:2)'*inv(H(1:2)*Pmenos(1:2,1:2)*H(1:2)'+Rwz);

K(3:4)=Pmenos(3:4,3:4)*H(3:4)'*inv(H(3:4)*Pmenos(3:4,3:4)*H(3:4)'+Rwy);

K(5:6)=Pmenos(5:6,5:6)*H(5:6)'*inv(H(5:6)*Pmenos(5:6,5:6)*H(5:6)'+Rwx);
    roll(muestra)=rad2deg(asin(ax_sinOffset(muestra)/9.8));
    pitch(muestra)=rad2deg(abs(acos(9.8/az_sinOffset(muestra))))-
155.8523;
    if isinf(pitch(muestra))
        pitch(muestra)=0;
    end
    if gx(muestra)<0
        pitch(muestra)=-pitch(muestra);
    end
    Xmas(1:2)=Xmenos(1:2)+K(1:2)'*(desplazamiento_MagZ-
H(1:2)*Xmenos(1:2));
    Xmas(3:4)=Xmenos(3:4)+K(3:4)'*(roll(muestra)-H(3:4)*Xmenos(3:4));
    Xmas(5:6)=Xmenos(5:6)+K(5:6)'*(pitch(muestra)*2-H(5:6)*Xmenos(5:6));
    XmasGuardada(:,muestra)=Xmas;
    Pmas=(I-K'*H)*Pmenos;
%             %   ****************************************
                xSim(muestra)=xSim(end)+(ys(end,3)-ys(end-
1,3))*sin(deg2rad(Xmas(1)));
                ySim(muestra)=ySim(end)+(ys(end,3)-ys(end-
1,3))*cos(deg2rad(Xmas(1)));
                disp([gz(muestra) Xmas(1) desplazamiento_MagZ])
                zSim(muestra)=zSim(end)+(ys(end,3)-ys(end-
1,3))*sin(deg2rad(Xmas(5)));
                axes(handles.graficaPista)
                plot3(xSim,ySim,zSim);grid on;
                xlim([min(xSim)-0.1 max(xSim)+0.1]);
                ylim([min(ySim)-0.1 max(ySim)+0.1]);
                xlabel('X')
                ylabel('Y')
                zlabel('Z')
                view(0,90)
                set(handles.x,'String',xSim(muestra));
                set(handles.y,'String',ys(muestra,3));
```

```
                set(handles.z,'String',zSim(muestra));
                set(handles.gx,'String',Xmas(5));
                set(handles.gy,'String',Xmas(3));
                set(handles.grados_z,'String',Xmas(1));
%                 disp([gz(end) desplazamiento_MagZ Xmas(1)])
            end
            otherwise
        end
        muestra=muestra+1;
        pause(0.005);
    end
catch err
    beep;
    identificador=cellstr(char(err.identifier));
    mensaje=cellstr(char(err.message));
    ruta=cellstr(char(err.stack.file));
    nombre=char(err.stack.name);
    nombreProceso=cellstr(['Proceso: ' nombre(2,:)]);
    [l1,l2]=err.stack.line;

    errordlg([cellstr('Ha Ocurrido Un Error, Se Cancelará El Proceso')
        cellstr('')
        identificador
        mensaje
        cellstr('')
        ruta{2}
        nombreProceso
        cellstr(['Linea: ' num2str(l2)])],'ERROR');
    uiwait
    delete(gcf);
end
    EOT=' ';
    while strcmp(EOT,'FIN')~=1
        EOT = fscanf(handles.puerto,'%s');
    end
    save('voltajeEntero1.mat','voltajeEntero1')
    save('voltajeEntero2.mat','voltajeEntero2')
    save('anguloInclinacion.mat','anguloInclinacion')
    save('M.mat','M')
    save('T.mat','T')
    save('sisc.mat','sisc')
    save('c.mat','c')
    save('Gr.mat','Gr')
    save('XmasGuardada.mat','XmasGuardada')
    save('tiempoSim.mat','tiempoSim')
    save('ys.mat','ys')
    save('Xmas.mat','Xmas')
    save('vs.mat','vs')
    save('offsetX.mat','offsetX')
    save('offsetY.mat','offsetY')
    save('offsetZ.mat','offsetZ')
    save('anguloXMag.mat','anguloXMag')
    save('anguloYMag.mat','anguloYMag')
    save('anguloZMag.mat','anguloZMag')
```

```
    save('cx.mat','cx')
    save('cy.mat','cy')
    save('cz.mat','cz')
    save('tiempoReal.mat','tiempoReal')
    save('voltajesSimulacion.mat','voltajesMotores')
    save('voltajesMedidos.mat','voltajesMedidos')
    save('vMotor1.mat','vMotor1')
    save('vMotor2.mat','vMotor2')
    save('ax.mat','ax_sinOffset')
    save('ay.mat','ay_sinOffset')
    save('az.mat','az_sinOffset')
    save('wx.mat','wx_sinOffset')
    save('wy.mat','wy_sinOffset')
    save('wz.mat','wz_sinOffset')
    save('gx.mat','gx')
    save('gy.mat','gy')
    save('gz.mat','gz')
    instrreset
    set(handles.adquirir,'Enable','on');
    set(handles.cancelar,'Enable','on');
    set(handles.finalizar,'Value',0,'Enable','off');
    pause(0.5);
    [handles.infoSistema,handles.dato_num] =
infoSys(handles.infoSistema,[EOT ': Enlace
finalizado.'],handles.dato_num);
    set(handles.sistema,'String',handles.infoSistema);
    set(handles.sistema,'Value',handles.dato_num);
    pause(0.5);
    set(handles.cerrar,'Enable','on');
    guidata(hObject,handles);
function z_Callback(hObject, eventdata, handles)
function y_Callback(hObject, eventdata, handles)
function x_Callback(hObject, eventdata, handles)
function grados_z_Callback(hObject, eventdata, handles)
function gy_Callback(hObject, eventdata, handles)
function gx_Callback(hObject, eventdata, handles)
function finalizar_Callback(~, eventdata, handles)
function muestras_Callback(hObject, eventdata, handles)
function [infoSistema,dato_num]=infoSys(cadena1,cadena2,dato_num)
dato_num = dato_num+1;
infoSistema = [cellstr(cadena1);cellstr(cadena2)];
function cerrar_Callback(hObject, eventdata, handles)
instrreset
close all;
function trayectoria_Callback(hObject, eventdata, handles)
global segmento valoresTrayectoria indicadoresTrayectoria
numeroTrayectoria...
    enviarOrden voltajesMotores sisc T c M Gr nVoltajes
instrreset
puertoCom=get(handles.puertos,'String');
puertoCom=char(puertoCom(1));
velocidad=get(handles.baudios,'String');
velocidad=str2double(char(velocidad(get(handles.baudios,'Value'))));
```

```
puerto =
serial(puertoCom,'baudrate',velocidad,'Timeout',5,'Timerperiod',3,'StopBi
ts',1);
fopen(puerto);
%constantes sistema
rc=0.085; %separacion centro de masa llanta
G=144.24;    %ratio de convercion transmicion
rw=0.026;   %radio llanta
M=0.67;
Gr=rw/G;

a=125.3/0.08;
b=1/0.08;
c=1/4.76e-7;
%matriz
A=[ -b 0 0 0
    0 -b 0 0
    .5 .5 0 0
    1/rc -1/rc 0 0];
B=[ a*Gr 0      0
    0   a*Gr    0
    0 0  0
    0 0  0         ];
C=[ 1 0 0 0
    0 1 0 0
    0 0 1 0
    0 0 0 1];
D=zeros(4,3);
sisc=ss(A,B,C,D);
% ************************************************
T=0.3561;
valoresTrayectoria=[];
indicadoresTrayectoria='';
numeroTrayectoria=1;
segmento=1;
vs=[];
Trayectoria
uiwait();

ack = ' ';
    while strcmp(ack,'VOK')~=1
        fprintf(puerto,'%s',['6' 13]);
        ack = fscanf(puerto,'%s');
    end
[handles.infoSistema,handles.dato_num] =
infoSys(handles.infoSistema,'Guardando datos en memoria
eeprom...',handles.dato_num);
set(handles.sistema,'String',handles.infoSistema);
set(handles.sistema,'Value',handles.dato_num);
pause(0.1);
muestraFuente=0;
while muestraFuente<=50
    vs=[vs fscanf(puerto,'%f')];
    muestraFuente=muestraFuente+1;
```

```
end
fprintf(puerto,'%s',['2' 13]);
msjPic=fscanf(puerto,'%s');
if strcmp(msjPic,'FINMUESTREOFUENTE')
    [handles.infoSistema,handles.dato_num] =
infoSys(handles.infoSistema,'Fuente de voltaje
medida...',handles.dato_num);
    set(handles.sistema,'String',handles.infoSistema);
    set(handles.sistema,'Value',handles.dato_num);
    pause(0.1);
else
    [handles.infoSistema,handles.dato_num] =
infoSys(handles.infoSistema,'MEDICIÓN DE FUENTE SIN
COMPLETAR...',handles.dato_num);
    set(handles.sistema,'String',handles.infoSistema);
    set(handles.sistema,'Value',handles.dato_num);
    pause(0.1);
end
for punteroVectorVoltajes=2:1:length(voltajesMotores(2,:))

voltajesEnterosD=dec2hex(ceil(voltajesMotores(2,punteroVectorVoltajes)*62
5/(mean(vs)-0.1)),4);
    MSB=voltajesEnterosD(1:2);
    LSB=voltajesEnterosD(3:4);
    MSB=hex2dec(num2str(MSB));
    LSB=hex2dec(num2str(LSB));
    fprintf(puerto,'%s',['M1 ' num2str(MSB) 13]);
    fscanf(puerto,'%i');
    pause(0.1)
    fprintf(puerto,'%s',['M1 ' num2str(LSB) 13]);
    disp(fscanf(puerto,'%i'))
    pause(0.1)
end
fprintf(puerto,'%s','10');
[nMotores,nVoltajes]=size(voltajesMotores);
save('nVoltajes.mat','nVoltajes')
[handles.infoSistema,handles.dato_num] =
infoSys(handles.infoSistema,'Datos motor 1
guardados...',handles.dato_num);
set(handles.sistema,'String',handles.infoSistema);
set(handles.sistema,'Value',handles.dato_num);
pause(0.1);
fprintf(puerto,'%s',13);
for punteroVectorVoltajes=2:1:length(voltajesMotores(1,:))

voltajesEnterosI=dec2hex(ceil(voltajesMotores(1,punteroVectorVoltajes)*62
5/(mean(vs(end))-0.1)),4);
    MSB=voltajesEnterosI(1:2);
    LSB=voltajesEnterosI(3:4);
    MSB=hex2dec(num2str(MSB));
    LSB=hex2dec(num2str(LSB));
    fprintf(puerto,'%s',['M2 ' num2str(MSB) 13]);
    fscanf(puerto,'%i');
    pause(0.1)
```

```
    fprintf(puerto,'%s',['M2 ' num2str(LSB) 13]);
    disp(fscanf(puerto,'%i'))
    pause(0.1)
end
[handles.infoSistema,handles.dato_num] =
infoSys(handles.infoSistema,'Datos motor 2
guardados...',handles.dato_num);
set(handles.sistema,'String',handles.infoSistema);
set(handles.sistema,'Value',handles.dato_num);
pause(0.5);
fprintf(puerto,'%s',['23' 13]);
[handles.infoSistema,handles.dato_num] =
infoSys(handles.infoSistema,'Datos guardados
correctamente.',handles.dato_num);
set(handles.sistema,'String',handles.infoSistema);
set(handles.sistema,'Value',handles.dato_num);
set(handles.adquirir,'Enable','on');
pause(0.5);
enviarOrden=1;
fclose(puerto);
function graficaPista_CreateFcn(hObject, eventdata, handles)
function voltajeMotor1_CreateFcn(hObject, eventdata, handles)
function voltajeMotor2_CreateFcn(hObject, eventdata, handles)
function calibrarMag_Callback(hObject, eventdata, handles)
function graficaCorrientes_Callback(hObject, eventdata, handles)
%%%constantes motor
global M anguloInclinacion
load voltajesMedidos
load tiempoSim
%%%constantes motor
K=0.00674619;
R=8.85;
L=0.00303;
J=4.76e-7;
b=7.41e-7;

rw=0.026;       %radio en metros
G=144.24;       %ratio medido
Gr=rw/G;
anguloInclinacion=0;
M=0.67;
Fl=M*9.8*sin(anguloInclinacion);
vectorPeso=Fl*ones(1,length(tiempoSim));
u1=[voltajesMedidos(1,:);vectorPeso];
u2=[voltajesMedidos(2,:);vectorPeso];
A=[-R/L -K/L 0;
    K/J -b/J 0;
    0    1   0];

B=[1/L 0;
    0  -1/J;
    0  0];
C=[1 0 0;
   0 1 0;
```

```
   0 0 1];
D=zeros(3,2);
sisc=ss(A,B,C,D);
[ysMotor1,ts]=lsim(sisc,u1,tiempoSim);
[ysMotor2,ts]=lsim(sisc,u2,tiempoSim);
figure
subplot(121);plot(tiempoSim,ysMotor1(:,1)');xlabel('Tiempo
(s)');ylabel('Corriente (A)');
title('Motor Derecho');grid on;
subplot(122);plot(tiempoSim,ysMotor2(:,1)');xlabel('Tiempo (s)');
title('Motor Izquierdo');grid on;
function graficaVoltajes_Callback(hObject, eventdata, handles)
load voltajesMedidos
load voltajesSimulacion
load tiempoReal
figure
subplot(211);plot(tiempoReal,voltajesMotores(1,:),tiempoReal,voltajesMedi
dos(2,:));legend('Calculado','Medido');
title('Comparación Voltajes Calculados vs Reales');
ylabel('Voltaje Motor Izquierdo [v]');
grid on;
subplot(212);plot(tiempoReal,voltajesMotores(2,:),tiempoReal,voltajesMedi
dos(1,:));legend('Calculado','Medido')
ylabel('Voltaje Motor Derecho [v]')
xlabel('Tiempo (s)');
grid on;
function tiempoMag_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function ultimaTrayectoria_Callback(hObject, eventdata, handles)
set(handles.adquirir,'Enable','on');
```

ANEXO C. CÓDIGO EN MATLAB PARA DEFINIR TRAYECTORIA

```
function varargout = Trayectoria(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Trayectoria_OpeningFcn, ...
                   'gui_OutputFcn',  @Trayectoria_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
function Trayectoria_OpeningFcn(hObject, eventdata, handles, varargin)
global curvaNumero voltajesMotores
voltajesMotores=[0 0]';
curvaNumero=0;
set(handles.distancia,'Enable','off');
set(handles.radio,'Enable','off');
set(handles.giro,'Enable','off');
set(handles.anguloInclinacion,'Enable','off');
reiniciarTrayectoria_Callback(hObject, eventdata, handles)
handles.output = hObject;
guidata(hObject, handles);
function varargout = Trayectoria_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;
function infoUsuario_Callback(hObject, eventdata, handles)
function recta_Callback(hObject, eventdata, handles)
set(handles.distancia,'Enable','on');
set(handles.radio,'Enable','off');
set(handles.giro,'Enable','off');
set(handles.anguloInclinacion,'Enable','on');
set(handles.curva,'Value',0);
uicontrol(handles.distancia);
function curva_Callback(hObject, eventdata, handles)
set(handles.distancia,'Enable','off');
set(handles.radio,'Enable','on');
set(handles.giro,'Enable','on');
set(handles.anguloInclinacion,'Enable','off');

set(handles.recta,'Value',0);
uicontrol(handles.radio);
function aceptar_Callback(hObject, eventdata, handles)
global enviarOrden voltajesMotores tamanoVectorVoltajes
tamanoVectorVoltajes=size(voltajesMotores);
enviarOrden=1;
```

```
save('voltajesMotores.mat','voltajesMotores');
close Trayectoria;
function agregarSegmento_Callback(hObject, eventdata, handles)
global valoresTrayectoria indicadoresTrayectoria informacionUs
numeroTrayectoria...
    radios curvaNumero angulo valoresAnguloInclinacion
tiempo
if isempty(valoresTrayectoria)
    valoresTrayectoria=[];
    valoresAnguloInclinacion=[];
    indicadoresTrayectoria='';
    informacionUs='';
    numeroTrayectoria=1;
end
    if get(handles.recta,'Value')
        valoresTrayectoria=[valoresTrayectoria
str2double(get(handles.distancia,'String'))];
        valoresAnguloInclinacion=[valoresAnguloInclinacion
str2double(get(handles.anguloInclinacion,'String'))];
        indicadoresTrayectoria=[indicadoresTrayectoria 'R'];
        informacionUs=[informacionUs cellstr(['Segmento '
num2str(numeroTrayectoria) ...
            ': Recta = ' get(handles.distancia,'String') 'm. ' ...
            'Ángulo Inclinación: '
get(handles.anguloInclinacion,'String')])];
        uiwait();
        set(handles.infoUsuario,'String',informacionUs);
    end
    if get(handles.curva,'Value')
        valoresAnguloInclinacion=[valoresAnguloInclinacion 0];
        curvaNumero=curvaNumero+1;
        r=str2double(get(handles.radio,'String'));
        angulo=str2double(get(handles.giro,'String'));
        valoresTrayectoria=[valoresTrayectoria angulo];
        indicadoresTrayectoria=[indicadoresTrayectoria 'G'];
        radios=[radios r];

        informacionUs=[informacionUs cellstr(['Segmento '
num2str(numeroTrayectoria)...
            ': Giro = ' get(handles.giro,'String') '°; Radio = '
get(handles.radio,'String') 'm.'])];
        uiwait();
        set(handles.infoUsuario,'String',informacionUs);
    end
function distancia_Callback(hObject, eventdata, handles)
uicontrol(handles.anguloInclinacion)
function radio_Callback(hObject, eventdata, handles)
uicontrol(handles.giro);
function giro_Callback(hObject, eventdata, handles)
uicontrol(handles.agregarSegmento)
function anguloInclinacion_Callback(hObject, eventdata, handles)
global anguloInclinacion
anguloInclinacion = str2double(get(handles.anguloInclinacion,'String'));
uicontrol(handles.agregarSegmento)
```

```
function reiniciarTrayectoria_Callback(hObject, eventdata, handles)
global informacionUs valoresTrayectoria indicadoresTrayectoria
numeroTrayectoria...
    radios voltajesI voltajesD vectorTiempoSimulacion voltajesMotores

voltajesMotores=[0 0]';
vectorTiempoSimulacion=[];
valoresTrayectoria=[];
indicadoresTrayectoria='';
numeroTrayectoria=1;
voltajesI=[];
voltajesD=[];
radios=[];

set(handles.recta,'Value',0);
set(handles.curva,'Value',0);

set(handles.distancia,'String','','Enable','off');
set(handles.giro,'String','','Enable','off');
set(handles.radio,'String','','Enable','off');
set(handles.anguloInclinacion,'String','','Enable','off');
informacionUs='';
set(handles.infoUsuario,'String',informacionUs);
function quitar_Callback(hObject, eventdata, handles)
global valoresTrayectoria indicadoresTrayectoria informacionUs
numeroTrayectoria...
    vectorTiempoSimulacion
if ~isempty(valoresTrayectoria)
    if indicadoresTrayectoria(end)=='G'
        valoresTrayectoria(end)=[];
    end
indicadoresTrayectoria(end)=[];
informacionUs(end)=[];
informacionUs(end)=[];
vectorTiempoSimulacion(end)=[];
numeroTrayectoria = numeroTrayectoria-1;
set(handles.infoUsuario,'String',informacionUs);
else
    beep;
    msgbox('Los Valores Han Sido eliminados o No
Existen','Información','warn');
end
function cancelar_Callback(hObject, eventdata, handles)
global informacionUs valoresTrayectoria indicadoresTrayectoria
numeroTrayectoria...
    radios voltajesI voltajesD vectorTiempoSimulacion segmento

vectorTiempoSimulacion=[];
valoresTrayectoria=[];
indicadoresTrayectoria='';
numeroTrayectoria=1;
voltajesI=[];
voltajesD=[];
radios=[];
```

```
informacionUs='';
close Trayectoria
function agregarSegmento_KeyPressFcn(hObject, eventdata, handles)
agregarSegmento_Callback(hObject, eventdata, handles)
```

ANEXO D. CÓDIGO EN MATLAB PARA INGRESAR EL TIEMPO DE RECORRIDO

```
function varargout = tiempo(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @tiempo_OpeningFcn, ...
                   'gui_OutputFcn',  @tiempo_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
function tiempo_OpeningFcn(hObject, eventdata, handles, varargin)
global numeroTrayectoria
set(handles.indicadorTiempo,'String',['Segmento: '
num2str(numeroTrayectoria)]);
handles.output = hObject;
guidata(hObject, handles);

function varargout = tiempo_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;
function campoTiempo_Callback(hObject, eventdata, handles)
global voltajesMotores valoresTrayectoria indicadoresTrayectoria...
    numeroTrayectoria radios curvaNumero informacionUs angulo
vectorTiempoSimulacion...
    valoresAnguloInclinacion
    disp(valoresAnguloInclinacion)
switch indicadoresTrayectoria(numeroTrayectoria)
    case 'R'
        vectorTiempoSimulacion=[vectorTiempoSimulacion
str2double(get(handles.campoTiempo,'String'))];
        v=recta(vectorTiempoSimulacion(end),...
            valoresTrayectoria(numeroTrayectoria),...
            valoresAnguloInclinacion(numeroTrayectoria));
     voltajesMotores=[voltajesMotores(1,:) v 0;voltajesMotores(2,:) v 0];
        voltajesI=mean(v);
        voltajesD=mean(v);
    case 'G'
        if angulo<0
        vectorTiempoSimulacion=[vectorTiempoSimulacion
str2double(get(handles.campoTiempo,'String'))];
        [vD,vI]=curva(vectorTiempoSimulacion(end),...
            radios(curvaNumero),deg2rad(abs(angulo)));
  voltajesMotores=[voltajesMotores(1,:) vD 0 ;voltajesMotores(2,:) vI 0];
```

```
        else
        vectorTiempoSimulacion=[vectorTiempoSimulacion
str2double(get(handles.campoTiempo,'String'))];
        [vI,vD]=curva(vectorTiempoSimulacion(end),...
            radios(curvaNumero),deg2rad(angulo));
   voltajesMotores=[voltajesMotores(1,:) vD 0;voltajesMotores(2,:) vI 0];
        end
        voltajesI=mean(vI);
        voltajesD=mean(vD);
    otherwise
end
    informacionUs=[informacionUs cellstr(['Voltaje Promedio MI: '
num2str(voltajesI) ' v; '...
        'Voltaje Promedio MD: ' num2str(voltajesD) ' v.'])];
    numeroTrayectoria = numeroTrayectoria+1;
close tiempo
```

ANEXO E. CÓDIGO EN MATLAB PARA CALCULAR LOS VOLTAJES EN RECTA

```
function [ ur,alf ] = recta(tf,D,alf)
x1i=0;      x1f=0;
x2i=0;      x2f=D;
%constantes del sistema
rc=0.108;    %distancia entre llantas/2 [m]
M=0.5;

%descripcion del sistema
rw=0.026;       %radio en metros
G=144.24;       %ratio medido

Gr=rw/G;
Fl=9.8*M*sin(alf)*Gr;
%%constantes motor
J=4.76e-7;
a=130.3/0.08;
b=1/0.08;
c=Fl/J;

const=Gr*a/2;

%definicion de varibles como simbolicas
a=sym(a);
b=sym(b);
c=sym(c);
Gr=sym(Gr);
%solucion de EC diferenciales
S=dsolve('Dx1=-Gr*c-b*x1-(Gr^2*L1*a^2)/2,  Dx2=x1,  DL1=L1*b-L3,  DL3=0,
x1(0)=x1i,x2(0)=x2i,x1(tf)=x1f,x2(tf)=x2f');
ts=0.3181;
j=1;
for tp=0:ts:tf
    t=sym(tp);
     x1p(j)=double(subs(S.x1));
     x2p(j)=double(subs(S.x2));
    L1p(j)=-double(subs(S.L1));
   if L1p(j)>0
    ur(j)=L1p(j);
   end
    tr(j)=tp;
    j=j+1;
end
 ur=ur*const;
end
```

ANEXO F. CÓDIGO EN MATLAB PARA CALCULAR VOLTAJES EN CURVA

```
function [ uc1 uc2 ] = curva(tf,Cr,th)
x1i=0;      x1f=0;
x2i=0;      x2f=Cr*(th+pi/7);
%Constantes del sistema
rc=0.108;%distancia entre llantas [m]
Cr=Cr;
p=(2*Cr-rc)/(2*Cr+rc);
p2=p;

%Descripcion del sistema
rw=0.026; %radio en metros
G=144.24; %ratio de reducción
Gr=rw/G;
Fl=0;

%Constantes motor
J=4.76e-7;
a=130.3/0.08;
b=1/0.08;
c=Fl/J;
const=Gr*a/(p^2+1);

%definición de variables como simbólicas
a=sym(a);
b=sym(b);
c=sym(c);
Gr=sym(Gr);
p=sym(p);
%solución de ecuaciones diferenciales
S=dsolve('Dx1=-Gr*c-b*x1-(Gr^2*L1*a^2)/(p^2+1),  Dx2=(x1*(p+1))/2,
DL1=L1*b-L3*(p+1)/2,  DL3=0,
x1(0)=x1i,x2(0)=x2i,x1(tf)=x1f,x2(tf)=x2f');
%transformando a vectores
ts=0.3181;
j=1;
for tp=0:ts:tf
    t=sym(tp);
    L1p(j)=-double(subs(S.L1));
     if L1p(j)>0
     uc1(j)=L1p(j);

     end
    tc(j)=tp;
    j=j+1;
end
uc1=const*uc1;
uc2=uc1*p2;
end
```

ANEXO G. CÓDIGO DE ADQUISICIÓN DE DATOS PARA EL MICROCONTROLADOR

```
#INCLUDE <18F2550.h>
#DEVICE ADC=10
#FUSES HS,NOWDT,NOPROTECT,MCLR,NOBROWNOUT,NOLVP,NOVREGEN
#USE DELAY (CLOCK=20M)
#USE RS232(baud=9600,xmit=pin_c6,rcv=pin_c7)
#use i2c(MASTER, SDA=PIN_B0, SCL=PIN_B1,FORCE_HW)

#DEFINE SENSIBILIDAD ACC_SENS_1
#DEFINE g   9.8

#DEFINE GX_OFF 0X4C
#DEFINE GY_OFF 0X37
#DEFINE GZ_OFF 0X09
#DEFINE BUFFER_VOLTAJES    0x03
#DEFINE ANGULO_DECLINACION 4.1667


#define RECEPCION_DATOS PIN_A3
#define ENVIO_DATOS     PIN_A4//PIN DE INFORMACIÓN VISUAL DE ENVÍO
DE DATOS

#define OFFSETX_MAG   0.3197
#define OFFSETY_MAG   0.0261

#include <ADXL345.h>
#include <ITG3205.h>
#include <HMC5883L.h>
#include <string.h>
#include <stdlib.h
#include <math.h>
#include <24256.c>

int8 estado=0,desborde_contador=0;
int contador_palabras=0,calibracion=1,tomaMuestras=0;
INT8 ACK,STATUS_MAG;
```

```
int16 dato_num=5,tiempo,ADC,muestraM1=0,muestraM2=16000,contadorM1=0,
contadorM2=16000;
long valor;
iNT16 x,y,z,wx,wy,wz,cx,cy,cz;
FLOAT X1,Y1,Z1,wx1,wy1,wz1,cx1,cy1,cz1;
BYTE ACELERACION[6],GIROSCOPIO[8],MAGNETOMETRO[6];
BYTE voltajesM1[BUFFER_VOLTAJES],voltajesM2[BUFFER_VOLTAJES];
INT16 voltajeEntero1=0,voltajeEntero2=0;
char dato[8],*ptr;
char M1[]="M1",M2[]="M2",C[]="C";
char *ordenMotores,*palabra1,*palabra2,separador_espacio[2]=" ";
float vMotor1,vMotor2,vFuente;
float32 microsegundos=0;

#int_timer0
void timer0()
{
  desborde_contador++;
}
#int_rda
void adquirirCadenaPuerto1(){
     output_high(RECEPCION_DATOS);
     gets(dato);
     dato_num=strtoul(dato,&ptr,10);
  ordenMotores=strtok(dato,separador_espacio);

  while(ordenMotores!=0){
    if(contador_palabras==0){
      palabra1=ordenMotores;
      }
    else{
      palabra2=ordenMotores;
     }
    ordenMotores = strtok(0,separador_espacio);
    contador_palabras++;
  }
  contador_palabras=0;
  valor=strtoul(palabra2,&ptr,10);
  output_low(RECEPCION_DATOS);
```

```
   if(dato_num==1)
      calibracion=0;
   if(dato_num==4)
      estado=4;
}
VOID MAIN()
{
output_high(ENVIO_DATOS);
delay_ms(200);
output_low(ENVIO_DATOS);
setup_timer_0(RTCC_INTERNAL|RTCC_DIV_1);
setup_ccp1(CCP_PWM);
setup_ccp2(CCP_PWM);
setup_timer_2(T2_DIV_BY_16,156,1);

setup_adc(ADC_CLOCK_DIV_16);
setup_adc_portS(AN0_TO_AN2);

enable_interrupts(global);
enable_interrupts(int_timer0);
enable_interrupts(int_rda);

set_pwm1_duty(0);
set_pwm2_duty(0);
init_ext_eeprom();
DELAY_MS(100);
   I2C_START();
   I2C_WRITE(ACC_ESCRIBIR);
   I2C_WRITE(ACC_DATA_FORMAT);
   I2C_WRITE(SENSIBILIDAD);
   I2C_STOP();

   I2C_START();
   I2C_WRITE(ACC_ESCRIBIR);
   I2C_WRITE(ACC_OFSX);
   I2C_WRITE(0X01);
   I2C_STOP();

   I2C_START();
```

```
I2C_WRITE(ACC_ESCRIBIR);
I2C_WRITE(ACC_OFSY);
I2C_WRITE(0X06);
I2C_STOP();

I2C_START();
I2C_WRITE(ACC_ESCRIBIR);
I2C_WRITE(ACC_OFSZ);
I2C_WRITE(0X06);
I2C_STOP();

I2C_START();
I2C_WRITE(ACC_ESCRIBIR);
I2C_WRITE(ACC_TAP_AXES);
I2C_WRITE(0X0F);
I2C_STOP();

I2C_START();
I2C_WRITE(ACC_ESCRIBIR);
I2C_WRITE(ACC_THRESH_TAP);
I2C_WRITE(0X01);
I2C_STOP();

I2C_START();
I2C_WRITE(ACC_ESCRIBIR);
I2C_WRITE(ACC_DUR);
I2C_WRITE(0XFE);
I2C_STOP();

I2C_START();
I2C_WRITE(ACC_ESCRIBIR);
I2C_WRITE(ACC_BW_RATE);
I2C_WRITE(0X03);
I2C_STOP();

I2C_START();
I2C_WRITE(ACC_ESCRIBIR);
I2C_WRITE(ACC_POWER_CTL);
I2C_WRITE(0X08);
```

```
I2C_STOP();
//GIROSCOPIO
delay_ms(100);
I2C_START();
I2C_WRITE(GYRO_ESCRIBIR);
I2C_WRITE(GYRO_PWR_MGM);
I2C_WRITE(0X00);
I2C_STOP();

I2C_START();
I2C_WRITE(GYRO_ESCRIBIR);
I2C_WRITE(GYRO_SMPLRT_DIV);
I2C_WRITE(0X07);
I2C_STOP();

I2C_START();
I2C_WRITE(GYRO_ESCRIBIR);
I2C_WRITE(GYRO_DLPF_FS);
I2C_WRITE(0X1A);
I2C_STOP();

I2C_START();
I2C_WRITE(GYRO_ESCRIBIR);
I2C_WRITE(GYRO_INT_CFG);
I2C_WRITE(0X00);
I2C_STOP();
//MAGNETOMETRO
delay_ms(100);
I2C_START();
I2C_WRITE(MAG_ESCRIBIR);
I2C_WRITE(MAG_CFG_REG_A);
ACK=I2C_WRITE(00011000);
I2C_STOP();
I2C_START();
I2C_WRITE(MAG_ESCRIBIR);
I2C_WRITE(MAG_CFG_REG_B);
I2C_WRITE(11100000);
I2C_STOP();
I2C_START();
```

```
   I2C_WRITE(MAG_ESCRIBIR);
   I2C_WRITE(MAG_MODE_REG);
   ACK=I2C_WRITE(00000000);
   I2C_STOP();

DO
{
set_timer0(0);
if(strcmp(palabra1,C)==0)
{
   contadorM1=valor;
   contadorM2=valor+16000;
   estado=2;
   palabra1="NULL";
}
 switch(estado)
   {
   case 0:
   if(dato_num==6)
   {
      printf("VOK\n");
      dato_num=0;
      output_high(ENVIO_DATOS);
      output_high(RECEPCION_DATOS);
   }
   if(dato_num==0)
   {
      set_adc_channel(0);
      delay_us(40);
      ADC=read_adc();
      vFuente=ADC*5.0/(1024.0*0.3337);
      printf("%1.4f\n\r",vFuente);
   }
   if(dato_num==2)
   {
      printf("FIN MUESTREO FUENTE\n\r");
      output_low(ENVIO_DATOS);
      output_low(RECEPCION_DATOS);
      estado++;
```

```
}
delay_ms(100);
break;
case 1:
  output_toggle(ENVIO_DATOS);
  if(strcmp(palabra1,M1)==0)
  {
    write_ext_eeprom(contadorM1,valor);
    printf("%u\n\r",read_ext_eeprom(contadorM1));
    contadorM1++;
    palabra1="NULL";
  }
  delay_ms(100);
  if(strcmp(palabra1,M2)==0)
  {
    write_ext_eeprom(contadorM2,valor);
    printf("%u\n\r",read_ext_eeprom(contadorM2));
    delay_ms(50);
    contadorM2++;
    palabra1="NULL";
  }
  if (dato_num==23)
  {
    output_toggle(ENVIO_DATOS);
    delay_ms(300);
    output_toggle(ENVIO_DATOS);
    delay_ms(300);
    output_toggle(ENVIO_DATOS);
    delay_ms(300);
    output_high(ENVIO_DATOS);
    delay_ms(300);
    estado++;
  }
break;
case 2:
  if(dato_num==6){
    printf("OK\n\r");
    estado++;}
break;
```

```
case 3:
if(calibracion==0)
{
  voltajesM1[0]=read_ext_eeprom(muestraM1);
  voltajesM1[1]=read_ext_eeprom(muestraM1+1);
  voltajeEntero1=(voltajesM1[0]<<8);
  voltajeEntero1=voltajeEntero1+voltajesM1[1];
  set_pwm1_duty(voltajeEntero1);
  delay_ms(10);
  voltajesM2[0]=read_ext_eeprom(muestraM2);
  voltajesM2[1]=read_ext_eeprom(muestraM2+1);
  voltajeEntero2=(voltajesM2[0]<<8);
  voltajeEntero2=voltajeEntero2+voltajesM2[1];
  set_pwm2_duty(voltajeEntero2);
  delay_ms(10);
  muestraM1=muestraM1+2;
  muestraM2=muestraM2+2;}
if ((muestraM1)<=contadorM1||(muestraM2)<=contadorM2)
  {
I2C_START();
I2C_WRITE(ACC_ESCRIBIR);
I2C_WRITE(ACC_DATAX0);
I2C_START();
I2C_WRITE(ACC_LEER);
ACELERACION[0]=I2C_READ();
ACELERACION[1]=I2C_READ();
ACELERACION[2]=I2C_READ();
ACELERACION[3]=I2C_READ();
ACELERACION[4]=I2C_READ();
ACELERACION[5]=I2C_READ(0);
I2C_STOP();
I2C_START();
I2C_WRITE(GYRO_ESCRIBIR);
I2C_WRITE(GYRO_XOUT_H);
I2C_START();
I2C_WRITE(GYRO_LEER);
GIROSCOPIO[0]=I2C_READ();
GIROSCOPIO[1]=I2C_READ();
GIROSCOPIO[2]=I2C_READ();
```

```
GIROSCOPIO[3]=I2C_READ();
GIROSCOPIO[4]=I2C_READ();
GIROSCOPIO[5]=I2C_READ();
GIROSCOPIO[6]=I2C_READ();
GIROSCOPIO[7]=I2C_READ(0);
I2C_STOP();

I2C_START();
I2C_WRITE(MAG_ESCRIBIR);
ACK=I2C_WRITE(MAG_DATAX_H);
I2C_START();
I2C_WRITE(MAG_LEER);
MAGNETOMETRO[0]=I2C_READ();
MAGNETOMETRO[1]=I2C_READ();
MAGNETOMETRO[4]=I2C_READ();
MAGNETOMETRO[5]=I2C_READ();
MAGNETOMETRO[2]=I2C_READ();
MAGNETOMETRO[3]=I2C_READ();
STATUS_MAG=I2C_READ(0);
I2C_STOP();

x=(ACELERACION[1]<<8);
x=x+ACELERACION[0];

y=(ACELERACION[3]<<8);
y=y+ACELERACION[2];

z=(ACELERACION[5]<<8);
z=z+ACELERACION[4];

x1=(float)x*factor;
y1=(float)y*factor;
z1=(float)z*factor;
if (bit_test(x,15))
{
  x=~x+1;
  x1=(float)x*factor;
  x1=x1*(-1);
}
```

```
if (bit_test(y,15))
{
  y=~y+1;
  y1=(float)y*factor;
  y1=y1*(-1);
}
if (bit_test(z,15))
{
  z=~z+1;
  z1=(float)z*factor;
  z1=z1*(-1);
}

//CÁLCULOS GIROSCOPIO
wx=(GIROSCOPIO[0]<<8);
wx=wx+GIROSCOPIO[1]-GX_OFF;

wy=(GIROSCOPIO[2]<<8);
wy=wy+GIROSCOPIO[3]+GY_OFF;

wz=(GIROSCOPIO[4]<<8);
wz=wz+GIROSCOPIO[5]+GZ_OFF;

wx1=(float)wx/14.375;
wy1=(float)wy/14.375;
wz1=(float)wz/14.375;

if (bit_test(wx,15))
{
  wx=~wx+1;
  wx1=(float)wx/14.375;
  wx1=wx1*(-1);
}
if (bit_test(wy,15))
{
  wy=~wy+1;
  wy1=(float)wy/14.375;
  wy1=wy1*(-1);
}
```

```
if (bit_test(wz,15))
{
  wz=~wz+1;
  wz1=(float)wz/14.375;
  wz1=wz1*(-1);
}
//CÁLCULOS MAGNETOMETRO
cx=(MAGNETOMETRO[0]<<8);
cx=cx+MAGNETOMETRO[1];

cy=(MAGNETOMETRO[2]<<8);
cy=cy+MAGNETOMETRO[3];

cz=(MAGNETOMETRO[4]<<8);
cz=cz+MAGNETOMETRO[5];

cx1=(float)cx*0.00435;
cy1=(float)cy*0.00435;
cz1=(float)cz*0.00435;

if (bit_test(cx,15))
{
  cx=~cx+1;
  cx1=(float)cx*0.00435;
  cx1=cx1*(-1);
}
if (bit_test(cy,15))
{
  cy=~cy+1;
  cy1=(float)cy*0.00435;
  cy1=cy1*(-1);
}
if (bit_test(cz,15))
{
  cz=~cz+1;
  cz1=(float)cz*0.00435;
  cz1*=(-1);
}
//MEDICIÓN DE VOLTAJES
```

```
delay_ms(100);

set_adc_channel(0);
delay_us(40);

ADC=read_adc();
vFuente=ADC*5.0/(1024.0*0.3337);

set_adc_channel(1);
delay_us(40);

ADC=read_adc();
vMotor1=ADC*5.0/(1024.0*0.3326);

set_adc_channel(2);
delay_us(40);

ADC=read_adc();
vMotor2=ADC*5.0/(1024.0*0.3319);
printf("\f%Lu\n\r",(muestraM1/2)+1);
printf("%Lu\n\r",voltajeEntero1);
printf("%Lu\n\r",voltajeEntero2);
printf("%0.4f\n\r",vFuente);
printf("%0.4f\n\r",vMotor1);
printf("%0.4f\n\r",vMotor2);
printf("%0.4f\n\r",x1*g);
printf("%0.4f\n\r",y1*g);
printf("%0.4f\n\r",z1*g);
printf("%0.4f\n\r",wx1);
printf("%0.4f\n\r",wy1);
printf("%0.4f\n\r",wz1);
printf("%0.4f\n\r",cx1);
printf("%0.4f\n\r",cy1);
printf("%0.4f\n\r",cz1);
microsegundos=tiempo*0.2;
printf("%1.4f\n\r%u\n\r",microsegundos,desborde_contador);
desborde_contador=0;
printf("ACK\n\r");
}
```

```
else
{
  set_pwm1_duty(0);
  set_pwm2_duty(0);
  printf("EOT\n\r");
  estado++;
  calibracion=1;
}
voltajeEntero1=0;
voltajeEntero2=0;
delay_ms(100);
output_toggle(ENVIO_DATOS);
break;
case 4:
  tomaMuestras=0;
  printf("NAK\n\r");
  output_low(ENVIO_DATOS);
  set_pwm1_duty(0);
  delay_ms(100);
  set_pwm2_duty(0);
  delay_ms(100);
  printf("FIN\n");
  output_high(RECEPCION_DATOS);
  delay_ms(500);
  output_low(RECEPCION_DATOS);
  delay_ms(500);
  output_high(RECEPCION_DATOS);
  delay_ms(500);
  output_low(RECEPCION_DATOS);
  delay_ms(500);
  printf("FIN\n");
  estado=0;
  contadorM1=0;
  dato_num=5;
  contadorM2=16000;
  muestraM1=0;
  muestraM2=16000;
  valor=0;
  calibracion=1;
```

```
   break;
   default:
   break;
   }
tiempo=get_timer0();
}WHILE(1);
}
```

ANEXO H. CÓDIGO PARA TACÓMETRO EN MICROCONTROLADOR

```
#include <16f877a.h>
#fuses HS,NOWDT
#USE DELAY (CLOCK=20000000)
#use rs232 (BAUD=38400 , XMIT=PIN_C6 , RCV=PIN_C7,bits=8, parity=N)
//#INCLUDE <LCD.C>
#use fast_io(A)

int16 TFB;
int flanco=0,ini=0;
int32 desborde=0;
float32 Tseg,frec=0,Rps,Radps;

#INT_ext
void funcion_ext_INT()
{
  output_toggle (pin_A0) ;

  IF (flanco == 0)
  {
    set_timer1 (0);
    flanco = 1;
    desborde = 0;
  }

  ELSE
  {
    TFB = get_timer1 (); //tiempo entre flancos... us
    Tseg = 0.2*( (FLOAT) TFB + 65535.0 * (float)desborde) / 1000000.0; //periodo
de flancos en seg
    frec = 1.0 / (Tseg); //frecuencia entre flancos
    Rps = (frec / 8.0); //rps motor
    Radps = Rps * 2 * 3.141592; //radianes por segundo motor
    printf("\n%6.4f\r",frec);
    //printf (lcd_putc, "\n %6.4f ", frec);
    flanco = 0;
  }
}

#INT_timer1

void timer0()
```

```
{
   desborde++;
}

void main()
{
   //lcd_init ();

   set_tris_A (0x00) ;
   output_low (pin_A0) ;
   enable_interrupts (INT_ext) ;
   ext_INT_edge (0, L_TO_H) ;

   enable_interrupts (INT_timer1);
   setup_timer_1 (T1_INTERNAL|T1_DIV_BY_1);
   enable_interrupts (global);


   WHILE (true)
   {
      if(ini==0)
      {
      printf("\n%6.4f\r",frec);
      ini=1;
      }
      else{
      //printf("\n%6.4f\r",frec);
      output_toggle (pin_a1) ;
      delay_ms (500) ;
      }

   }
}
```