UNIVERSIDAD MICHOACANA DE
SAN NICOLÁS DE HIDALGO

FACULTAD DE INGENIERÍA ELÉCTRICA
DIVISIÓN DE ESTUDIOS DE POSGRADO

"CONSTRUCCIÓN Y CONTROL DE UN VEHÍCULO ELÉCTRICO AUTOBALANCEADO DE DOS RUEDAS PARA TRANSPORTE UNIPERSONAL"

TESIS

QUE PARA OBTENER EL GRADO DE
MAESTRO EN CIENCIAS EN INGENIERÍA ELÉCTRICA

PRESENTA
ING. GABRIEL CASARRUBIAS GUERRERO

DIRECTOR DE TESIS
DR. JOSÉ JUAN RINCÓN PASAYE
Doctor en Ciencias en la Especialidad de Control Automático

MORELIA, MICHOACÁN JULIO DE 2014

CONSTRUCCIÓN Y CONTROL DE UN VEHÍCULO AUTOBALANCEADO SOBRE DOS RUEDAS PARA TRANSPORTE UNIPERSONAL

Los Miembros del Jurado de Examen de Grado aprueban
la **Tesis de Maestría en Ciencias en Ingeniería Eléctrica** de ***Gabriel Casarrubias Guerrero***

Dr. Fernando Ornelas Téllez
Presidente del Jurado

Dr. José Juan Rincón Pasaye
Director de Tesis

Dr. Gilberto González Avalos
Vocal

Gilberto González

Dr. Juan Anzurez Marín
Vocal

Dr. Ignácio Juárez Campos
Revisor Externo

Dr. J. Aurelio Medina Rios
Jefe de la División de Estudios de Posgrado
de la Facultad de Ingeniería Eléctrica. UMSNH
(Por reconocimiento de firmas).

UNIVERSIDAD MICHOACANA DE SAN NICOLÁS DE HIDALGO
Mayo 2014

Agradecimientos

A mi madre Victoria Guerrero, que con su amor, comprensión y ejemplo, me da la fortaleza para superarme día a día personal y profesionalmente.

A mi compañera de la vida, mi esposa Ivón por su paciencia y cariño durante todo momento, pilar fundamental para culminar este gran logro en mi vida.

A mi asesor el Dr. José Juan Rincón Pasaye, con mucho respeto y admiración, por confiarme la responsabilidad de sacar adelante este trabajo y por el apoyo y paciencia brindados en el desarrollo del mismo, ya que sin su esfuerzo no hubiera sido posible la culminación de este trabajo.

A mi hermano Felipe, por su cariño, comprensión y apoyo en mis proyectos, tanto personales como profesionales.

A toda mi familia, por que siempre me reciben con una sonrisa, además de brindarme siempre todo su apoyo.

A CONACYT por haberme apoyado económicamente, motivándome para seguir con mis estudios.

A los técnicos del laboratorio por todo el apoyo recibido durante mi estadía en la Universidad Michoacana.

A mis compañeros de generación que su amistad y ayuda fue fundamental para la culminación de mis estudios.

Para Ivón y Mamá

Resumen

En este trabajo se presenta el diseño, construcción y control de un vehículo de dos ruedas autobalanceado para transporte unipersonal. El principal objetivo de este trabajo fue la construcción de un vehículo de dos ruedas autobalanceado con la capacidad para transportar a una persona de un lugar a otro. Uno de los principales retos para lograr esto es la implementación de controladores que permitan lograr el balanceo del vehículo. El vehículo construido estructuralmente cuenta con tres partes fundamentales, una base construida de aluminio, la cual sirve de apoyo para el piloto, además de albergar los motores, drivers, baterías y la tarjeta de control del sistema, las llantas sobre las cuales se desplaza el vehículo y el manubrio que facilita el manejo al piloto y que lleva montado un display donde se pueden observar las variables del sistema. En la parte matemática, se consideraron términos de fricción viscosa y el modelo de un motor de DC y se agregaron a un modelo ya existente de la parte mecánica, el cual fue obtenido por el método de Euler-Lagrange. En la parte electrónica, el vehículo cuenta con una tarjeta principal donde por medio de dos microcontroladores se obtienen las lecturas de los sensores, se evalúa el algoritmo de control y se envían las señales de control a los drivers de los motores de CD los cuales mueven las llantas del vehículo. En la parte del sensado, es usado un par acelerómetro-giroscopio para medir el ángulo de inclinación del vehículo y un par de encoders acoplados a las flechas de cada uno de los motores para obtener la medición de la velocidad de giro de las llantas. Toda la parte electrónica se alimenta con dos baterías de Gel de 12 Volts conectadas en serie, las baterías pueden ser cargadas sin retirarlas del vehículo lo cual facilita su manejo. En cuanto a los controladores, se implementan y comparan en rendimiento un controlador PID clásico con compensación de zona muerta y anti windup, y un controlador PD difuso mínimo. La programación de los microcontroladores se lleva a cabo en lenguaje C, en dos compiladores diferentes XC16 de microchip y en el copilador CCS. Al final del documento se presentan los resultados obtenidos de las pruebas en simulación y en tiempo real en el vehículo construido.

Palabras clave: Control, Vehículo eléctrico, Sistemas no lineales, PID, Lógica difusa.

Abstract

The design, construction and control of a unipersonal two-wheeled autobalance vehicle is presented in this thesis. The transportation of one person from one place to another on the two-wheeled autobalanced vehicle is the main aim of this work. The mechanical structure consists in three parts: an aluminium base which supports the pilot and houses the batteries, the drivers, motors and the main board, wheels for moving the vehicle and the handlebar which helps to the pilot to drive the vehicle, also the handlebar has been equipped with the control panel and the direction commands (joysticks) of the vehicle. The DC motor model and the viscous friction terms are added to an existing mathematical model for the mechanical subsystem which was obtained by the Euler-Lagrange method. A main board with two microcontrollers, an accelerometer-gyro pair and a pair of encoders form the electronic part of the vehicle. The main board reads the four sensors, evaluates the control algorithm and sends the control signals to the motor drivers to move the wheels, the accelerometer-gyro pair measures the tilt angle, while the encoders mounted in the wheels measure the angular velocity of the wheels for balancing the vehicle. Two 12 volt gel batteries feed all the electronics devices. The Recharge of the batteries is easy because the batteries can be recharged within the vehicle. A clasic PID controller with antiwindup and death zone compensation and a minimun fuzzy PD controller are implement and compared in this work. XC16 and CCS are the compilers used for programming the microcontrollers in the main board. The real time and numerical simulation results are presented at the end of this thesis.

Palabras clave: Control, Electric vehicle,Nonlinear systems, PID, Fuzzy logic.

Índice general

Índice de figuras

Índice de tablas

Lista de Abreviaturas

CD	Corriente directa
DOF	Degrees of Freedom (Grados de libretad)
HOT	Honda Omni-Traction (Omni-Tracción Honda)
IIC	Inter-Integrated Circuit (Inter-Circuitos Integrados)
LCD	Liquit Crystal Display (Display de Cristal Liquido)
LED	Light Emiter Diode (Diodo emisor de Luz)
MEMS	Micro-Electro-Mechanical Systems (Sistemas micro electromecánicos)
MIPS	Millones de instrucciones por segundo
OSMC	Open Source Motor Controller (Controlador de Motor Open Source)
PID	Proporcional, Integral, Derivativo
USART	Universal Synchronous/Asynchronous Receiver/Transmitter (Receptor-Transmisor Síncrono-Asíncrono Universal)

CAPÍTULO 1

Introducción

Los vehículos tipo péndulo invertido han atraído un considerable interés para los investigadores, desde la aparición de los primeros y pequeños prototipos desarrollados a lo largo de la década de los 90's [Yamafuji 1989, Ha 1996] y los comienzos del siglo XXI [Grasser 2002], hasta la aparición en las últimas décadas de vehículos comerciales montables [Segway 2001, Honda 2009]. El interés en este tipo de sistemas ha tenido un importante crecimiento, resultando en un auge de prototipos desarrollados alrededor del mundo, algunos de estos son pequeños prototipos didácticos o académicos [Bonales 2012, Kim 2005, Hamza 2011, Nawawi 2008, Nawawi 2006] y algunos otros, prototipos montables con diversos grados de funcionalidad [Deal 2008, DIY 2008, OSPV 2011, Maureira 2010, Moreno 2009].

Estos vehículos son sistemas simples pero interesantes y representan un reto para el control, ya que son sistemas no lineales y subactuados que se tienen que operar en su punto de operación inestable. La zona muerta y los efectos de saturación de los actuadores añaden dificultades para lograr una buena estabilización de la posición vertical. Las estrategias de control para este tipo de vehículos han sido motivación de mucho trabajo de investigación. Un enfoque que se puede tomar para tratar de controlar este tipo de vehículos es considerarlos como una variante del problema clásico de control del péndulo invertido sobre un carro, con este enfoque en mente existen varios trabajos ya publicados. Algunos esquemas de control lineal se han presentado desde el clásico PID, hasta esquemas LQR [Hamza 2011] y estrategias de modos deslizantes de orden completo basados en un modelo lineal [Nawawi 2006]. Recientemente se han presentado investigaciones que toman en cuenta la naturaleza no lineal del sistema, logrando así implementar técnicas de control no lineales [Shui 2009].

En este contexto, este documento presenta la construcción de un vehículo autobalanceado funcional de bajo costo hecho con componentes económicos y reciclados, se presenta también

una descripción del diseño del hardware involucrado en la construcción del prototipo, se presentan en detalle los aspectos electrónicos y mecánicos, así como las estrategias de control que son implementadas en tiempo real.

1.1. Motivación

Algunos aspectos que motivan la investigación sobre los vehículos de dos ruedas autobalanceados son:

- Como vehículo funcional: sus muchas ventajas comparados con otros tipos de vehículos: ocupan poco espacio, son ligeros, no usan combustibles fósiles, pueden usarse tanto en interiores como en exteriores.
- Como prototipo didáctico: son excelentes prototipos para ensayar esquemas de control lineales y no lineales ya que su punto de operación es inestable, además de ser sistemas subactuados y de fase no mínima.

1.2. Estado del Arte

La evolución de estos sistemas se ha dado en dos aspectos, uno de ellos es la incorporación de avances tecnológicos tales como mejores sensores y actuadores, la incorporación de procesadores más potentes y el diseño de estructuras más ligeras y esbeltas que facilitan la operación del vehículo.

Por otro lado, la parte de control del autobalanceo vertical del vehículo es un problema que se sigue investigando y que ha dado lugar a un sinnúmero de técnicas de control que consideran diferentes enfoques para el modelado y que producen resultados diversos.

A continuación se presenta un resumen de algunos de estos trabajos.

1.2.1. Modelado y control

La dinámica del vehículo de dos ruedas autobalanceado tiene un comportamiento muy similar a un problema clásico del control, el péndulo invertido [Shui 2009]. En la actualidad existen diversos estudios sobre el comportamiento del péndulo invertido sobre dos ruedas, y sobre robots móviles tipo péndulo invertido. Cabe mencionar que este problema es de naturaleza no lineal y que el punto de equilibrio donde se requiere la operación es inestable. El vehículo construido tiene 3 grados de libertad (DOF), el movimiento en las 2 llantas y el

giro de inclinación de la base, sin embargo únicamente cuenta con dos actuadores que son los dos motores que hacen girar las llantas, por esta razón se dice que este sistema es subactuado ya que tiene más grados de libertad que actuadores.

El primer estudio reportado sobre este tipo de dispositivos es el realizado por Yamafuji y Kawamura en 1989, ellos construyeron un modelo donde controlaron el ángulo respecto a la vertical y realizaron diferentes pruebas basados en el criterio de estabilidad de Routh-Hurwitz [Yamafuji 1989].

Ha y Yuta construyeron un pequeño prototipo de péndulo invertido montado en un robot móvil llamado Yamabico Kurara en 1996 [Ha 1996], éste usaba un giroscopio para medir la velocidad de inclinación y encoders para la velocidad de desplazamiento del robot, el prototipo podía conducirse de manera autónoma sobre un plano mientras mantenía su propio equilibrio, ellos pudieron dividir el control en 3 partes, control de balanceo, control de giro y control de seguimiento, utilizando para el equilibrio un controlador lineal por retroalimentación de estado.

En 2002, Gresser, D'Arrigo y Colombi presentan a JOE un péndulo invertido móvil [Grasser 2002], el cual tiene 3-DOF, en su trabajo presentan un modelado linealizado para describir la dinámica del prototipo, basados en la caracterización de los parámetros del prototipo. El controlador presentado es una retroalimentación lineal de estados más un bloque de desacoplamiento de los pares aplicados a cada motor con respecto a los pares producidos en cada variable a controlar (ángulo de inclinación y posición horizontal del robot).

En 2005, Kim, Kim y Kwak construyen un robot de 3-DOF tipo péndulo invertido [Kim 2006], el cual modelaron tomando en cuenta restricciones no holonómicas, pensando que no hay derrapes de las llantas, considerando superficies inclinadas y el efecto del giro del robot mediante el método de Kane. La parte del controlador fue basada en la linealización del modelo y la aplicación de una retroalimentación de estados óptima tipo LQR23.

En el mismo año Aracil y Gordillo presentan una revisión de varios métodos de diseño y análisis de sistemas de control no lineales, empleado el péndulo invertido como sistema de referencia [Aracil 2005]. En su trabajo abordan la doble problemática local y global del comportamiento del péndulo, revisan algunos controladores no lineales propuestos por otros autores y proponen una solución al problema del swing-up con estabilización, las técnicas revisadas son probadas en simulación.

Un año después en 2006, Nawawi, Ahmad y Osman presentan el modelo matemático de un péndulo invertido de dos ruedas, en cuanto al control, utilizan un controlador robusto basado en control por modos deslizantes para una estabilización robusta y rechazo de perturbaciones en base al modelo linealizado del sistema[Nawawi 2006]. La técnica es probada

sólo en simulación.

En 2009 Shui-Chun, Ching-Chih y Hsu-Chih presentan un método de control de modo deslizante adaptivo no lineal para el control de autobalanceo y velocidad de giro de un vehículo de dos ruedas [Shui 2009] , tomando en cuenta variaciones en la masa e incertidumbres del sistema debido a la masa del conductor. Los autores utilizaron un modelo no lineal modificado a partir del modelo presentado en [Grasser 2002] al cual se le añaden términos de fricción. En su vehículo ellos utilizaron un sensor de inclinación junto a giroscopio para obtener las variables del sistema. En su trabajo mostraron resultados tanto en simulación como en tiempo real implementado el controlador en un vehículo funcional, sin embargo, los resultados en tiempo real no son ilustrados mediante gráficas sino solamente son comentados.

En el año 2010, Sánchez, Aguirre y Patete modelan por medio del método de Lagrange, un pequeño robot construido en base a la plataforma Lego Mindstorm NXT [LEGO M 2010] al cual le implementan un controlador PID para el balanceo.

En 2010, Nasir, Raja Ismail y Ahmad presentan un estudio de comparación del rendimiento entre un control por modos deslizantes basado en un modelo lineal y un PID aplicados a sistemas de péndulo invertido [Nasir 2010], logrando exitosamente el equilibrio en ambos pero mostrando mejor rendimiento con respecto a especificaciones de tiempo al usar el controlador con modos deslizantes, las técnicas son probadas en simulación.

También en 2010 Khaled MK Goher y M O Tokhi llevan a cabo el diseño e implementación de un algoritmo de control PID utilizando algoritmos genéticos como técnica de optimización para un diseño nuevo de un vehículo de dos ruedas con la barra del manubrio extensible [Goher 2010]. El diseño del vehículo tiene una característica adicional a la del péndulo invertido sobre dos ruedas. La barra del manubrio cuenta con un actuador lineal en la parte media el cual permite alargar o encoger la barra del manubrio la cual tiene una carga en la parte superior que ayuda a la estabilización del sistema. Los resultados presentados son en simulación

Durante el mismo año 2010, Hu, Tsai, Hu y Hori presentan una investigación sobre el uso de un esquema de control robusto H_∞ para pilotear de manera remota un vehículo eléctrico de dos ruedas basándose únicamente en el ángulo de inclinación del vehículo [**?**], demostrando con sus resultados la factibilidad y flexibilidad del control remoto para el vehículo.

Ching-Chih, Hsu-Chih y Shui-Chun presentan un control adaptivo usando redes neuronales de base radial [Ching-Chih 2010], descomponen el sistema en dos subsistemas (péndulo invertido móvil y giro del vehículo) y proponen dos controladores adaptivos basados en redes neuronales de base radial para lograr el autobalanceo del vehículo de dos ruedas y el control en el giro del vehículo. El rendimiento de los controladores desarrollados se ejemplifican

mediante la realización de varias simulaciones y experimentos en el vehículo de dos ruedas construido. Los resultados experimentales son ilustrativos mediante algunas secuencias fotográficas que son parte de un video.

En el año 2011 Jen-Yang, Chuan-Hsi, Chien-Chun y Kuei-Chih proponen un controlador difuso adaptivo para balancear un vehículo de dos ruedas en la posición vertical [Chen 2011]. Las leyes de adaptación son derivadas del análisis de estabilidad de Lyapunov de esta manera el rendimiento del sistema de seguimiento y la convergencia del error pueden ser asegurados en el sistema de control en lazo cerrado. Ellos también implementan un controlador PD para comparar el rendimiento del sistema con los dos controladores. Para las implementaciones ellos utilizan la plataforma LEGO mindstorm.

En el mismo año, Jiménez muestra otro pequeño prototipo tipo péndulo invertido sobre dos ruedas, el cual se enfoca en la construcción, instrumentación, control y consideraciones para la implementación de este tipo de vehículos [**?**]. El prototipo es controlado por medio de un PID además muestra algunas técnicas de filtrado para la lectura de los sensores de posición (Acelerómetro y Giroscopio).

En 2013, Bonales, construye un prototipo a escala del vehículo de dos ruedas autobalanceado [Bonales 2012], donde hace un modelado por el método de Euler-Lagrange donde únicamente considera la dinámica de la estructura mecánica, sin tomar en cuenta la fricción y el modelo del motor de CD, los controladores diseñados en el prototipo de Bonales fueron un PID (simulación e implementación) y una retroalimentación de estado lineal (simulación), también presenta en su trabajo la implementación de un filtro de Kalman para mejorar la medición del ángulo de inclinación del prototipo.

En este mismo año Sun Lu y Yuan proponen un nuevo método de control de balance basado en un controlador LQR y redes neuronales [**?**]. En este método el controlador de balance es diseñado como un controlador LQR que contiene una red neuronal dentro. Los parámetros óptimos del controlador LQR son usados para inicializar la red neuronal, los cuales harán que la red tenga valores iniciales óptimos y rápida convergencia. Las pruebas del controlador son realizadas en un pequeño robot a escala construido por los autores.

1.2.2. Prototipos y vehículos comerciales

Existen en la actualidad un gran número de prototipos de vehículos autobalanceados montables construidos alrededor del mundo, algunos ya se fabrican a nivel comercial. También existen otros prototipos que se usan con fines didácticos, a continuación se describen algunos de estos prototipos didácticos y montables.

Robot móvil autobalanceado de Yamabico Kurara [Ha 1996]

Uno de los primeros prototipos de este tipo fue construido en 1996 en la Universidad de Tsukuba, Japón por Ha y Yuta. El robot es un péndulo invertido sobre ruedas el cual persigue mantener el equilibrio respecto a la vertical. El modelado matemático es basado en las ecuaciones de movimiento de Lagrange. Su esquema de control implementado es un controlador de retroalimentación de estado lineal junto a un controlador de prealimentación.

El JOE: Robot móvil autobalanceado [Grasser 2002]

Construido en 2002 por el Laboratorio de Electrónica Industrial en el Instituto de Tecnología Federal de Lausanne, Suiza. Este robot móvil es una representación a escala del vehículo montable personal el cual lleva una varilla con pesas en el centro que emula el peso y cuerpo de un conductor como se muestra en la figura 1.1. EL robot es controlado por medio de una retroalimentación de estado lineal, además éste cuenta con un joystick para el control del movimiento del robot. JOE alcanza velocidades de hasta 5.4 Km/h, pesa 15 Kg y se puede controlar con inclinaciones de hasta 30^0. Para la medición del ángulo de inclinación el JOE únicamente cuenta con un giroscopio y para el avance y velocidad cuenta con dos encoders incrementales instalados uno en cada motor.

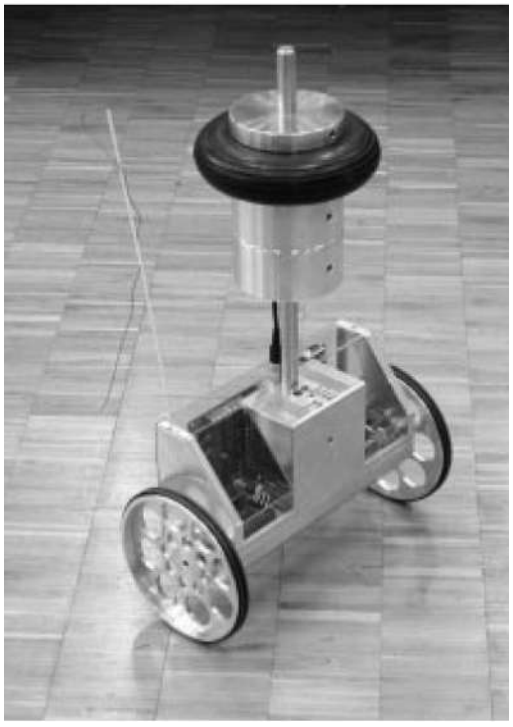


Figura 1.1: Robot movíl JOE [Grasser 2002]

Prototipo de Sánchez, Aguirre y Patete [LEGO M 2010]

Este prototipo fabricado por Héctor Sánchez, Iñaki Aguirre y Anna Patete en el Departamento de Sistemas de Control de la Universidad de los Andes, Venezuela en el año 2010, toma como base la plataforma Lego MINDSTORMS NXT, es un péndulo invertido sobre ruedas el cual presenta un comportamiento similar al de un vehículo autobalanceado. El prototipo cuenta con un controlador PID para mantener el equilibrio. Este fue creado como plataforma de prueba para algoritmos de control. Este prototipo usa como sensor de inclinación un giroscopio el cual mide la velocidad y dirección de giro.

Figura 1.2: Péndulo invertido construido con Lego Mindstorm [LEGO M 2010]

Prototipo de Bonales [Bonales 2012]

Este es un prototipo a escala donde la idea concebida en primera instancia fue la construcción de un vehículo montable, el modelado se realizó por medio de las ecuaciones de Lagrange, el esquema de control propuesto para controlar el prototipo es un controlador PID digital implementado en un microcontrolador de 16 bits. La principal aportación en este trabajo es la aplicación de un filtro de Kalman para compensar la lectura del giroscopio mediante el uso de un acelerómetro obteniendo como resultado una medición del ángulo mejorada.

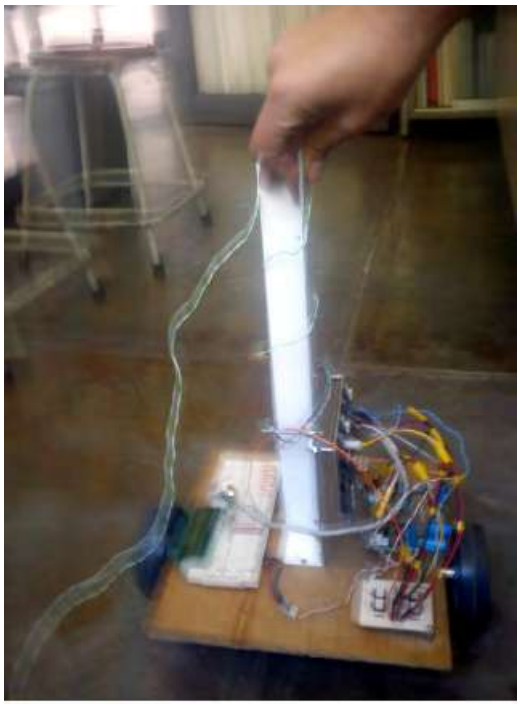


Figura 1.3: Prototipo a escala de Bonales [Bonales 2012]

Segway [Segway 2001]

Fue creado por Dean Kamen en 2001, fue el primer prototipo comercial de un vehículo de dos ruedas auto-balanceado para un tripulante. El controlador y motores fueron situados en la base. Segway Inc. presenta varios modelos de los cuales los dos modelos base son los llamados i2 y x2. La versión i2 es para zonas urbanas y el x2 es una versión todo terreno, como se muestra en la Figura 1.4

Este es el principal exponente de este tipo de prototipos montables y comerciales que existen en la actualidad, tienen como desventaja el ser excesivamente costoso alrededor de MX$100,000 por unidad. Su costo los hace poco accesibles , de tal manera que han pasado a usarse en aplicaciones muy específicas, como remplazo de caddies en canchas de golf, como vehículo de seguridad, y en supermercados. Algunas de sus características son mostradas en la tabla 1.1

Figura 1.4: Prototipos Segway. Izquierda modelo x2, derecha modelo i2 [Segway 2001]

Tabla 1.1: Características del Segway

	Segway X2	Segway i2
Peso	54.4 Kg	47.7 Kg
Vel. Máxima	20 Km/hr	20 Km/hr
Autonomía	19 Km	38 Km
Diametro de la llanta	53 cm	48 cm
Microcontrolador	DSP de TI	DSP de TI
Tiempor de aprendizaje	5 min	5 min
Motores	Brushless	Brushless

El segway es un vehículo comercial de Marca registrada y no hace públicos los detalles técnicos tales como la estrategia de control utilizada, ni los detalles de sus sensores y actuadores.

Freego [Freego 2008]

Otro prototipo comercial montable, muy parecido al Segway de origen chino diseñado en 2008 y puesto en el mercado en 2010, es uno de los varios competidores de Segway en Asia y Europa, con 45 Kg de peso este vehículo puede alcanzar los 18Km/h.

Figura 1.5: Vehículo comercial Freego [Freego 2008]

Prototipo montable de Blackwell [Blackwell 2009]

Este vehículo montable fue construido por Trevor Blackwell en el año 2002 con el fin de recrear el funcionamiento del Segway, éste consiste de un manubrio y una placa base de aluminio donde va ensamblado motores, tarjetas y baterías, Blackwell trató de economizar y utilizó piezas de fácil acceso para su construcción, en el año 2005 se construyó una versión mejorada del prototipo añadiendo mejoras en las llantas, sensores, baterías, giro y chasis.

A continuación en la tabla 1.2 se presentan las características de los dos prototipos probados la versión V1 (2002) y V2 (2005).

Tabla 1.2: Características Técnicas del Balancing Scooter [Blackwell 2009]

	V1	V2
Peso	40.8 Kg	31.75 Kg
Tamaño de ruedas	14”	20”
Vel. Máxima	14.5 Km/Hr	24 Km/Hr
Tipo de Batería	NiMH	60 HHR-6500D celdas de Panasonic
Motores	CD	CD
Microcontrolador	ATMEL	ATMEL
Tipo de control	PD	PD

Figura 1.6: Prototipo balancing scooter de Blanckwell [Blackwell 2009]

El Segway DIY [DIY 2008]

Diseñado y desarrollado por un grupo de estudiantes de diferentes escuelas (Wayland hight School, John D. O'Bryant School, Cambride Rindge, Latin School y el MIT), este prototipo tiene únicamente propósito académico y fue construido con un presupuesto de menos de USD $1000, el DIY segway (figura 1.8) implementa un controlador PD digital como esquema de control por medio de un microcontrolador PIC16F877 de Microchip logrando así la tarea del equilibrio del vehículo, la velocidad teórica de este diseño es de 17.5 Km/h.

El Two Wheel Deal [Deal 2008]

El Two Wheel Deal (Figura 1.7) prototipo totalmente funcional y montable creado en 2008 por Greg Eakins, Jeremy Gries, Pete Dudahs y Eric Geier estudiantes de la Universidad de Purdue, West Lafayette, Indiana, Estado Unidos. El control de balance de este vehículo fue hecho en base al pseudocódigo open source proporcionado por [Blackwell 2009] y el código open source provisto por [DIY 2008] los cuales tienen como base un controlador PD digital implementado en microcontrolador Atmel ATmega32 de 8 bits, el vehículo utiliza motores NPC T74 de 24 V para su movimiento . Este vehículo fue patentado pero no puesto en el mercado ni en venta.

Figura 1.7: The Two Wheel Deal [Deal 2008]

Figura 1.8: Prototipo balancing scooter DIY [DIY 2008]

Elektor Wheelie

Prototipo didáctico montable comercial fabricado por Elektor en el año 2010, esta plataforma es meramente didáctica, open source y open hardware. Este es la mejora de el modelo OSPV [OSPV 2011] que inicialmente era para interiores.

Figura 1.9: Prototipo Wheelie de Elektor[Wheelie 2009]

Algunas características de este modelo se encuentran en la tabla 1.3.

Tabla 1.3: Características de los prototipos Elektor

	Wheelie
Peso	35 Kg
Velocidad	18 Km/Hr
Batería	Ácido-Plomo de 12V
Autonomía	8 Km
Microcontrolador	ATmega16
Llantas	14”

El Toyota Winglet P.T. [Winglet 2008]

En 2008 la compañía Toyota lanzó al mercado un vehículo autobalanceado para interiores muy parecido al Segway pero con un manubrio corto operado con las piernas (ver figura 1.10), el cual puede alcanzar una velocidad de hasta 6 Km/hr, cuenta con tres versiones de diferentes tamaños con un peso desde los 10 Kg, mango de altura ajustable y con sensores para prevención de choque.

Figura 1.10: Toyota Winglet P.T. [Winglet 2008]

Prototipo de Sun, Zhou, Li, Wei y Li

Estos investigadores presentan en 2009 un vehículo de dos ruedas autobalanceado cuyo funcionamiento está basado en el uso de un par de acelerómetro y giroscopio MEMS [Sun 2009], el prototipo es diseñado basado en el modelo de un sistema de péndulo invertido. El vehículo usa dos ruedas en paralelo para mantener el balance (ver figura 1.11) y usa el acelerómetro y giroscopio como sensores de inclinación.

Figura 1.11: Prototipo de Sun, Zhou, Li, Wei y Li[Sun 2009]

Prototipo montable de Moreno [Moreno 2009] y Maureira [Maureira 2010]

Vehículo montable construido por Leonardo Moreno Bustamante en 2009 en La universidad de Chile, prototipo totalmente funcional, utiliza un controlador PD digital para mantener el equilibrio implementado mediante un DSP de Texas Instruments, fue el primer vehículo de este tipo construido totalmente en Chile, un año después fue mejorado por Rodrigo Andrés Maureira Tenorio quien añadió mejoras en la parte de dirección, electrónica y el esquema de control obteniendo un mejor equilibrio del vehículo y estableciendo un menor tiempo de aprendizaje para los pilotos inexpertos. Las características del primer prototipo y las mejoras se muestran en la tabla 1.4.

Figura 1.12: Prototipo mejorado de Maureira[Maureira 2010]

Tabla 1.4: Comparación entre el prototipo de Moreno y la mejora de Maureira

	Primer prototipo	Prototipo Mejorado
Peso	40 Kg	52 Kg
Vel. Máxima	6 Km/hr	4.7 Km/hr
Autonomía	>5 hrs.	2 hrs.
Tiempo de Aprendizaje	30 min	9.8 min
Motores	CD	CD

1.2.3. Variantes en este tipo de vehículos

Además de estos existen vehículos con el mismo principio de funcionamiento pero con ciertas variantes en la estructura mecánica y electrónica.

El Honda U3-X [Honda 2009]

Presentado como prototipo en New York en el año 2010, la principal característica de este vehículo experimental compacto es que permite el movimiento en cualquier dirección mediante su llanta omnidireccional (ver figura 1.13).

Figura 1.13: Honda U3-X [Honda 2009]

El U3-X pesa alrededor de 10 Kg y viaja hasta 6 Km/Hr, además cabe resaltar que en este vehículo el conductor va sentado sobre él. Usa un sistema llamado HOT que le permite moverse en cualquier dirección lateral. Otras características técnicas se muestran en la Tabla 1.5.

Tabla 1.5: Características Técnicas U3-X [Honda 2009]

	Honda U3-X
Largo	31.3 cm
Ancho	16 cm
Alto	64.7 cm
Sistema de Manejo	Omni Traction Drive System
Batería	Lithium-ion
Tiempo de Operación	1 Hr

Ryno Byke [RynoMotors 2013]

Esta motocicleta de una rueda autobalanceada fue creada en 2009 por la compañía Rino Motors fundada por Chris Hoffmannen en el 2006 en Portland, Oregon, Estados Unidos. Este prototipo funcional y comercial de baja velocidad es una variante construida sobre una única rueda, la estructura flexible permite al conductor poder dar vueltas, y el principio de balanceo es muy similar al de los prototipos anteriores, puede ser utilizado tanto en exteriores como interiores (ver figura 1.14).

Figura 1.14: Vehículo Ryno Byke [RynoMotors 2013]

En la actualidad existen además un gran número de diferentes modelos tanto comerciales (principalmente chinos), como didácticos fabricados por centros de investigación de escuelas, como bancos de prueba para la enseñanza del control automático.

1.3. Objetivos

1.3.1. Objetivo General

El objetivo de este trabajo es diseñar y construir el primer vehículo de dos ruedas autobalanceado fabricado en México, totalmente funcional capaz de transportar a una persona. Por otro lado, se quiere establecer este prototipo como un banco de pruebas para diferentes esquemas de control y como material didáctico para la enseñanza de la ingeniería de control.

1.3.2. Objetivos Particulares

1. Diseñar un prototipo montable totalmente funcional para transportar una persona abordo.
2. Verificar el modelo matemático añadiendo términos de fricción viscosa y el modelo de los actuadores.
3. Probar distintas técnicas de control para el balanceo y conducción del vehículo.
4. Utilizar componentes accesibles y de bajo costo.
5. Proveer al prototipo de un esquema de monitoreo de sus variables internas.

1.4. Aportaciones

Las aportaciones principales de este trabajo se enlistan a continuación:

- Construcción de un vehículo de dos ruedas autobalanceado tipo péndulo invertido, su instrumentación y control.
- Adición de términos de fricción viscosa al modelo matemático para una mejor aproximación al comportamiento real, respecto al presentado en [Bonales 2012].
- Adición del modelo de los actuadores a la parte de simulación para la obtención de resultados más acordes a la realidad.
- Creación de plantilla de programación en lenguaje C para implementar diversos controladores.
- Comparación del comportamiento del sistema al aplicar diferentes controladores.

- Implementación de un enlace de comunicación serial para monitoreo de las variables internas.

1.5. Justificación

De acuerdo con la investigación realizada, en el país no hay registro del desarrollo de este tipo de vehículos, lo cual hace a la universidad ser pionera en el campo de desarrollo de este tipo de vehículos eléctricos autobalanceados. Otras razones para el desarrollo de este vehículo son:

- El generar un banco de pruebas para la enseñanza de la teoría de control.
- Representa un gran ahorro económico ya que el construirlo de manera local reduce los costos teniendo el cuenta el costo de una unidad didáctica comercial.
- El prototipo puede ser base para nuevas investigaciones.
- Genera una nueva forma de transporte personal en la región para diferentes aplicaciones en distancias cortas.
- Es amigable con el ambiente ya que es un vehículo totalmente eléctrico.
- En un futuro el prototipo puede ser comercializable.

1.6. Contenido del Trabajo

La tesis está dividida en seis capítulos :

Capítulo 1: Introducción

Se presenta una visión general de los dispositivos tipo péndulo invertido, el estado del arte de lo que existe en cuanto a prototipos didácticos, prototipos montables, modelado y esquemas de control, los objetivos del trabajo realizado así como la justificación de éste.

Capítulo 2: Descripción del Prototipo Construido

En este capítulo se presentan los detalles del modelo del sistema con los términos de fricción incluidos y el modelo del actuador, así como los detalles de la estructura mecánica y del sistema electrónico que gobiernan el funcionamiento del vehículo.

Capítulo 3: Control del Prototipo

Se describen las diferentes técnicas de control implementadas en el vehículo montable.

Capítulo 4: Pruebas y Resultados

En esta sección se describen las pruebas de funcionamiento de las diferentes partes del sistema así como todo en conjunto, las consideraciones para las pruebas de simulación, los resultados de las pruebas de simulación y la implementación en tiempo real y sus resultados.

Capítulo 5: Manual de Operación

En este capítulo se describe el funcionamiento general del vehículo y el procedimiento para poder conducirlo adecuadamente.

Capítulo 6: Conclusiones

Aquí se presentan las conclusiones del trabajo así como sugerencias para las posibles mejoras en el vehículo.

CAPÍTULO 2

Descripción del Prototipo Construido

Este capítulo menciona los detalles de la construcción del prototipo fabricado consistente en un vehículo autobalanceado sobre dos ruedas tipo péndulo invertido para transportar un pasajero. Se describe con detalle cada parte del modelado matemático y del diseño mecánico y electrónico. El diseño del prototipo fue planeado para ser usado por una persona promedio, desde 1.5 a 1.9 m de alto y desde 50 a 90 Kilogramos. El diseño contempla los siguientes aspectos principales:

- Modelado Matemático
- Estructura Mecánica
- Sensores y Electrónica de Control
- Actuadores y Electrónica de Potencia
- Esquemas de Control

2.1. Descripción general

A grandes rasgos, el sistema del vehículo de dos ruedas autobalanceado se puede representar por el diagrama de bloques mostrado en la figura 2.1, en la cual se pueden observar como principales elementos los bloques de control, los actuadores, la estructura mecánica y los bloques de medición de las señales.

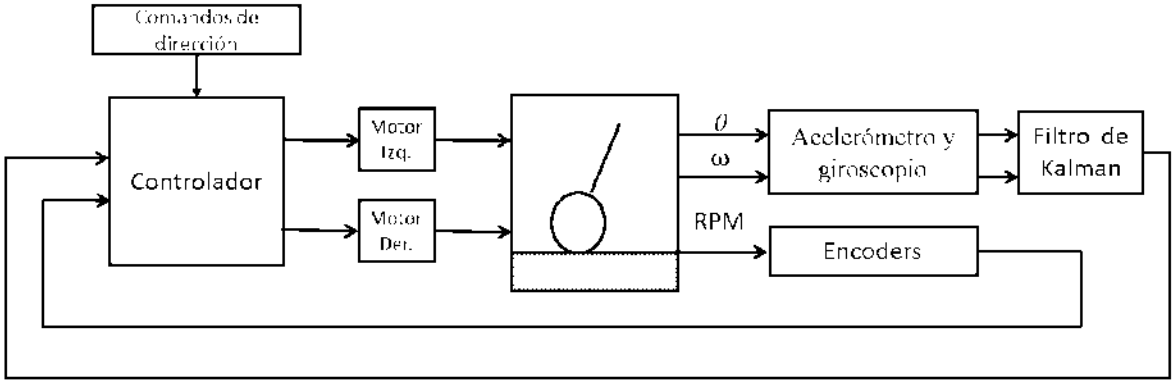


Figura 2.1: Diagrama de bloques del sistema.

El vehículo consta de una estructura mecánica de aluminio en la cual van montados los motores que mueven las dos llantas sobre las cuales se desplaza el vehículo, cuenta también con una tarjeta electrónica la cual adquiere el ángulo de inclinación θ y la velocidad angular de la inclinación ω por medio de un par Acelerómetro-Giroscopio y la velocidad de rotación de las llantas por medio de un par de encoders montados en los ejes de los motores. Al bloque de filtro de Kalman entran las señales correspondientes a la inclinación con las cuales a la salida se obtiene una medida del ángulo de inclinación θ mejorada. Para lograr la tarea del autobalanceo el vehículo consta de un bloque de control el cual se encarga por medio de las medidas obtenidas de los sensores de balancear el vehículo sobre su posición vertical.

En la siguiente sección se describen algunas adiciones al modelo matemático obtenido en [Bonales 2012] y posteriormente se explica con detalle cada uno de los aspectos antes mencionados.

2.2. Modelo Matemático

El modelo matemático del prototipo presentado en [Bonales 2012] fue obtenido por los métodos de las Leyes de Newton y Euler-Lagrange, tomando el sistema como un péndulo invertido sobre dos ruedas con la diferencia de que está montado sobre una plataforma móvil .

Las ecuaciones dinámicas del sistema se obtienen a partir del diagrama de la figura 2.2 en donde se explica gráficamente parte de los símbolos utilizados en las ecuaciones dinámicas, las cuales están dadas como

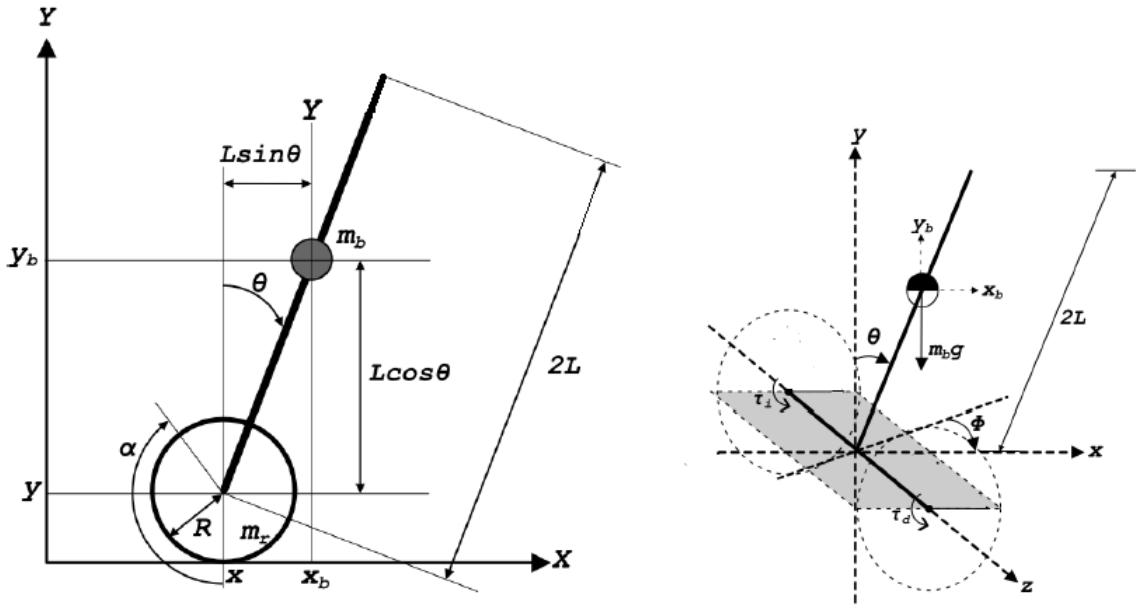


Figura 2.2: Diagrama esquemático del sistema [Bonales 2012]

$$\begin{aligned} (m_b L^2 + I_\theta)\ddot{\theta} + m_b RL\ddot{\alpha}cos\theta - m_{bgLsin\theta} &= -\tau_i - \tau_d \\ 2(m_r R^2 + \frac{1}{2}m_b R^2 + I_r)\ddot{\alpha} + m_b RL\ddot{\theta}cos\theta - m_b RL\dot{\theta}^2 sin\theta &= \tau_i + \tau_d \\ I_\phi \ddot{\phi} &= \frac{\tau_i - \tau_d}{R}D \end{aligned} \tag{2.1}$$

En donde θ es el ángulo de cabeceo (pitch) o inclinación del vehículo, α es el ángulo de giro de las llantas, φ es el ángulo de guiñada (yaw) o dirección de avance longitudinal, τ_i y τ_d son los pares aplicados a la rueda izquierda y derecha respectivamente. El modelo dado por las ecuaciones 2.1 y sus parámetros se describen en la Tabla 2.1.

2.2.1. Términos de Fricción

El modelo (2.1) fue obtenido sin considerar el efecto de la fricción. Para obtener un modelo más completo se introduce el efecto de la fricción mediante una función de disipación

de Rayleigh [Yepez 2007], quedando el modelo como sigue

$$\begin{aligned} (m_b L^2 + I_\theta)\ddot{\theta} + m_b RL\ddot{\alpha}cos\theta - m_b gLsin\theta &= -\tau_i - \tau_d + F_\theta \\ 2(m_r R^2 + \frac{1}{2}m_b R^2 + I_r)\ddot{\alpha} + m_b RL\ddot{\theta}cos\theta - m_b RL\dot{\theta}^2 sin\theta &= \tau_i + \tau_d + F_\alpha \\ I_\phi \ddot{\phi} &= \frac{\tau_i - \tau_d}{R}D \end{aligned} \tag{2.2}$$

donde F_θ es el par debido a la fricción implicada en el ángulo de inclinación θ y F_α es el par debido a la fricción en los ejes de las llantas y las cajas de engranes.

Tabla 2.1: Descripción de los parámetros del sistema

Símbolo	Descripción
g	Fuerza de gravedad
m_r	Masa de la llanta
m_b	Masa del cuerpo del vehículo más el tripulante
R	Radio de la llanta
L	Distancia del centro de masa m_b con respecto del origen
D	Distancia del centro del vehículo al centro de una llanta
I_R	Momento de inercia de la llanta
I_θ	Momento de inercia del cuerpo del vehículo respecto al eje z
I_ϕ	Momento de inercia del cuerpo del vehículo respecto al eje y

El efecto de la fricción es uno de los componentes más difíciles de modelar, ya que es el resultado de una gran variedad de fenómenos físicos que se producen cuando dos superficies en contacto se ponen en movimiento relativo. Desde el trabajo de Coulomb en 1785 se han desarrollado una gran variedad de enfoques sofisticados para modelar la fricción [Berger 2002].

Aunque se han propuesto una gran variedad de modelos para los términos de fricción [Olsson 1997] en este trabajo se usó uno de los más simples que es el de la fricción viscosa y que supone que las superficies en rozamiento están bien lubricadas. En el caso de la fricción viscosa los términos se pueden expresar como

$$\begin{aligned} F_\theta &= -b_0\dot{\theta} \\ F_\alpha &= -b_1\dot{\alpha} \end{aligned} \tag{2.3}$$

donde b_0, b_1 son los coeficientes de fricción viscosa correspondientes a las superficies en rozamiento y en este caso se suponen constantes positivas desconocidas.

2.2.2. Ecuaciones de estado

Basándonos en el modelo con fricción obtenido (2.2), se procede a obtener el modelo en ecuaciones de estado. A continuación se despeja $\ddot{\theta}$, $\ddot{\alpha}$ y $\ddot{\phi}$. De la segunda ecuación en (2.2), se obtiene

$$\ddot{\alpha} = \frac{-m_b RL\ddot{\theta}cos\theta + m_b RL\dot{\theta}^2 sin\theta + \tau_i + \tau_d + F_\alpha}{2(m_r R^2 + \frac{1}{2}m_b R^2 + I_r)} \tag{2.4}$$

sustituyendo (2.4) en la primera ecuación en (2.2)

$$\begin{aligned}(m_b L^2 + I_\theta)\ddot{\theta} &+ m_b RLcos\theta \left(\frac{-m_b RL\ddot{\theta}cos\theta + m_b RL\dot{\theta}^2 sin\theta + \tau_i + \tau_d + F_\alpha}{2(m_r R^2 + \frac{1}{2}m_b R^2 + I_r)} \right) \\ -m_b gLsin\theta &- F_\theta = -\tau_i - \tau_d\end{aligned}$$

factorizando $\ddot{\theta}$

$$\begin{aligned}&\frac{\left[2(m_b L^2 + I_\theta)\left(m_r R^2 + \frac{1}{2}m_b R^2 + I_r\right) - m_b^2 R^2 L^2 cos^2\theta\right]}{2(m_r R^2 + \frac{1}{2}m_b R^2 + I_r)}\ddot{\theta} + \\ &\frac{m_b RLcos\theta(m_b RL\dot{\theta}^2 sin\theta + F_\alpha + \tau_i + \tau_d)}{2(m_r R^2 + \frac{1}{2}m_b R^2 + I_r)} - m_b gLsin\theta - F_\theta = -\tau_i - \tau_d\end{aligned}$$

simplificando

$$\begin{aligned}&\frac{\left[2(m_b L^2 + I_\theta)\left(m_r R^2 + \frac{1}{2}m_b R^2 + I_r\right) - m_b^2 R^2 L^2 cos^2\theta\right]}{2(m_r R^2 + \frac{1}{2}m_b R^2 + I_r)}\ddot{\theta} = \\ &-\frac{m_b RLcos\theta(m_b RL\dot{\theta}^2 sin\theta + F_\alpha + \tau_i + \tau_d)}{2(m_r R^2 + \frac{1}{2}m_b R^2 + I_r)} + m_b gLsin\theta + F_\theta - \tau_i - \tau_d\end{aligned}$$

despejando $\ddot{\theta}$

$$\ddot{\theta} = \frac{-m_b RLcos\theta(m_b RL\dot{\theta}^2 sin\theta + F_\alpha + \tau_i + \tau_d) + (m_b gLsin\theta + F_\theta - \tau_i - \tau_d)\left[2\left(m_r R^2 + \frac{1}{2}m_b R^2 + I_r\right)\right]}{2(m_b L^2 + I_\theta)\left(m_r R^2 + \frac{1}{2}m_b R^2 + I_r\right) - (m_b RLcos\theta)^2} \tag{2.5}$$

Ahora sustituimos (2.5) en (2.4)

$$\ddot{\alpha} = \frac{-m_b RLcos\theta \left[\frac{-m_b RLcos\theta(m_b RL\dot{\theta}^2 sin\theta + F_\alpha + \tau_i + \tau_d) + (m_b gLsin\theta + F_\theta - \tau_i - \tau_d)\left[2\left(m_r R^2 + \frac{1}{2}m_b R^2 + I_r\right)\right]}{2(m_b L^2 + I_\theta)\left(m_r R^2 + \frac{1}{2}m_b R^2 + I_r\right) - (m_b RLcos\theta)^2} \right]}{2(m_r R^2 + \frac{1}{2}m_b R^2 + I_r)} + \frac{\tau_i + \tau_d + m_b RL\dot{\theta}^2 sin\theta + F_\alpha}{2(m_r R^2 + \frac{1}{2}m_b R^2 + I_r)}$$

simplificando $\ddot{\alpha}$ tenemos

$$\ddot{\alpha} = \frac{(m_b L^2 + I_\theta)\left(\tau_i + \tau_d + F_\alpha + m_b RL\dot{\theta}^2 sin\theta\right) - m_b RLcos\theta(m_b gLsin\theta - \tau_i - \tau_d + F_\theta)}{2\left(m_b L^2 + I_\theta\right)\left(m_r R^2 + \frac{1}{2}m_b R^2 + I_r\right) - (m_b RLcos\theta)^2} \quad (2.6)$$

Ahora se despeja $\ddot{\phi}$ de la tercera ecuación en (2.2)

$$\ddot{\phi} = \frac{\tau_i - \tau_d}{RI_\theta} \quad D \quad (2.7)$$

Definiendo las variables de estado como:

$$\begin{aligned} x_1 = \theta \quad x_3 = \alpha \quad x_5 = \phi \\ x_2 = \dot{\theta} \quad x_4 = \dot{\alpha} \quad x_6 = \dot{\phi} \end{aligned} \quad (2.8)$$

y se obtiene el modelo de estado utilizando (2.8) y acomodándolo de la forma

$$\dot{x} = f(x) + g(x)u$$

como se muestra a continuación

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \\ \dot{x}_5 \\ \dot{x}_6 \end{bmatrix} = \begin{bmatrix} x_2 \\ \frac{2\zeta m_b g L sinx_1 - \gamma^2 cosx_1 (x_2^2 sinx_1)}{\beta(x_1)} \\ x_4 \\ \frac{\gamma cosx_1 m_b g L sinx_1 - \vartheta\gamma \dot{x_2}^2 sinx_1}{\beta(x_1)} \\ x_6 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ \frac{2\zeta}{\beta(x_1)} & \frac{\gamma cos\theta}{\beta(x_1)} \\ 0 & 0 \\ \frac{\gamma cosx_1}{\beta(x_1)} & \frac{\vartheta}{\beta(x_1)} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} F_\theta(x_2) \\ F_\alpha(x_4) \end{bmatrix} + \tag{2.9}$$

$$\begin{bmatrix} 0 & 0 \\ \frac{\gamma cosx_1 - 2\zeta}{\beta(x_1)} & \frac{\gamma cosx_1 - 2\zeta}{\beta(x_1)} \\ 0 & 0 \\ \frac{\vartheta - \gamma cosx_1}{\beta(x_1)} & \frac{\vartheta - \gamma cosx_1}{\beta(x_1)} \\ 0 & 0 \\ \frac{D}{RI_\phi} & \frac{D}{RI_\phi} \end{bmatrix} \begin{bmatrix} \tau_i \\ \tau_d \end{bmatrix}$$

donde

$$\begin{aligned} \gamma &= -m_b RL \\ \zeta &= m_r R^2 + \frac{1}{2} m_b R^2 + I_r \\ \vartheta &= m_b L^2 + I_\theta \\ \beta(x_1) &= 2\left(m_b L^2 + I_\theta\right)\left(m_r R^2 + \frac{1}{2} m_b R^2 + I_r\right) - \left(m_b RL cosx_1\right)^2 \\ F_\theta(x_2) &= -b_0 x_2 \\ F_\alpha(x_4) &= -b_1 x_4 \end{aligned}$$

El modelo en espacio de estado resultante (2.9) consta de 6 variables de estado donde las últimas dos tienen una dinámica desacoplada que nos permite considerar el sistema como dos subsistemas separados.

2.2.3. Modelo de los actuadores

En esta sección se introduce el modelo del motor de corriente directa en el modelo (2.9) el cual tiene como entrada de control los pares generados por los motores de cada llanta,τ_i y τ_d, en el diagrama de bloques de la figura 2.3 se ilustra la manera en la que el modelo de los motores de cd que impulsan las llantas, se incorporan al modelo del vehículo.

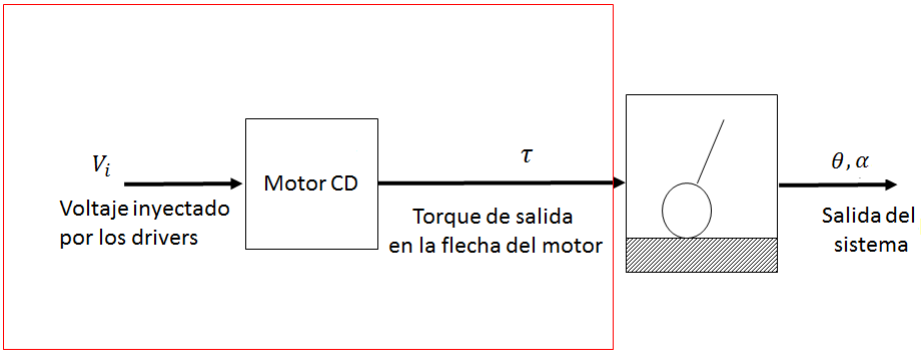


Figura 2.3: Diagrama a bloques de la entrada de control del sistema

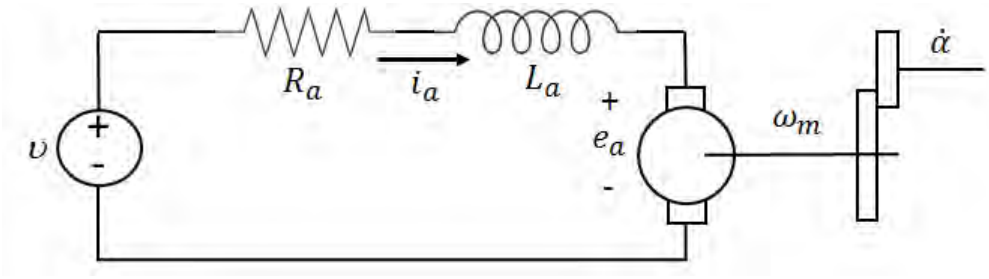


Figura 2.4: Diagrama equivalente de un motor de cd.

El par τ generado por el motor de corriente directa el cual es un motor de imán permanente, se puede obtener a partir del diagrama de la figura 2.4 despreciando la inductancia del devanado de armadura

$$\upsilon = R_a i_a + e_a \tag{2.10}$$

o bien,

$$\upsilon = R_a i_a + K_e \omega_m$$

donde υ es el voltaje aplicado en las terminales del motor, R_a es la resistencia del devanado de armadura, i_a es la corriente de armadura, ω_m es la velocidad angular del eje del motor que multiplicada por la ganancia K_e es la fuerza contra-electromotriz e_a. Como se tiene una caja de reducción de engranes, la velocidad angular $\dot{\alpha}$ de la rueda es igual a la velocidad del eje del motor multiplicada por el factor de reducción $b < 1$, es decir,

$$\dot{\alpha} = \omega_m b \tag{2.11}$$

por lo tanto

$$\upsilon = R_a i_a + K_e \frac{\dot{\alpha}}{b} \tag{2.12}$$

despejando la corriente de armadura

$$i_a = \frac{\upsilon}{R_a} - \frac{K_e \dot{\alpha}}{R_a b} \tag{2.13}$$

y como el campo magnético es constante, el par es proporcional a la corriente

$$\tau = K_\tau i_a \tag{2.14}$$

donde K_τ es la constante de proporcionalidad del par del motor, sustituyendo (2.13) en (2.14) obtenemos la siguiente ecuación

$$\tau = \frac{K_\tau \upsilon}{R_a} - \frac{K_\tau K_e \dot{\alpha}}{R_a b} \tag{2.15}$$

de acuerdo a 2.8, sustituyendo en (2.15) se obtiene

$$\tau = \frac{K_\tau}{R_a}\upsilon - \frac{K_\tau K_e}{R_a b} x_4 \tag{2.16}$$

sustituyendo la ecuación (2.16) en el modelo (2.9) obtenemos un sistema cuya entrada de control es el voltaje en las terminales de los motores.

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \\ \dot{x_3} \\ \dot{x_4} \\ \dot{x_5} \\ \dot{x_6} \end{bmatrix} = \begin{bmatrix} x_2 \\ \frac{2\zeta m_b g L sinx_1 - \gamma^2 cosx_1(x_2^2 sinx_1)}{\beta(x_1)} - 2\frac{(\gamma cosx_1 - 2\zeta)K_\tau K_e}{\beta(x_1)R_a b}x_4 \\ x_4 \\ \frac{\gamma cosx_1 m_b g L sinx_1 - \vartheta\gamma\dot{x_2}^2 sinx_1}{\beta(x_{1)}} - 2\frac{(\upsilon - cosx_1)K_\tau K_e}{\beta(x_1)R_a b}x_4 \\ x_6 \\ -2\frac{DK_\tau K_e}{R^2 I_\theta R_m}x_4 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ \frac{2\zeta}{\beta(x_1)} & \frac{\gamma cos\theta}{\beta(x_1)} \\ 0 & 0 \\ \frac{\gamma cosx_1}{\beta(x_1)} & \frac{\vartheta}{\beta(x_1)} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} F_\theta(x_2) \\ F_\alpha(x_4) \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ \frac{(\gamma cosx_1 - 2\zeta)K_\tau}{\beta(x_1)\cdot R_a} & \frac{(\gamma cosx_1 - 2\zeta)K_\tau}{\beta(x_1)\cdot R_a} \\ 0 & 0 \\ \frac{(\vartheta - \gamma cosx_1)K\tau}{\beta(x_1)R_a R} & \frac{(\vartheta - \gamma cosx_1)K_\tau}{\beta(x_1)R_a R} \\ 0 & 0 \\ \frac{DK_\tau}{RI_\phi R_a} & \frac{DK_\tau}{RI_\phi R_a} \end{bmatrix} \begin{bmatrix} \upsilon_i \\ \upsilon_d \end{bmatrix} \tag{2.17}$$

donde υ_i es el voltaje aplicado en las terminales del motor que mueve la llanta del lado izquierdo y υ_d es el voltaje aplicado en las terminales del motor que mueve la llanta del lado derecho. El elemento L_a se desprecia en el modelado debido a que este tiene una dinámica muy rápida la cual no afecta el funcionamiento del vehículo, además que al agregarlo se estaría aumentando el orden del modelo del actuador.

2.2.4. Análisis de los Puntos de equilibrio

Los sistemas no lineales pueden tener varios puntos de equilibrio, estamos interesados en analizar los puntos de equilibrio del sistema cuando la entrada de control es cero, entonces suponemos que $v_i = 0$ y $v_d = 0$. Para obtener los puntos de equilibrio del sistema se hace $\dot{x}$=0 y se reescriben las ecuaciones de estado (2.17) para esta condición, como sigue.

$$x_2 = 0 \tag{2.18}$$

$$2\zeta m_b g L sinx_1 - \gamma^2 cosx_1(x_2^2 sinx_1) + 2\zeta F_\theta + \gamma cos\theta F_\alpha = 0 \tag{2.19}$$

$$x_4 = 0 \tag{2.20}$$

$$\gamma cosx_1 m_b g L sinx_1 - \vartheta\gamma \dot{x_2}^2 sinx_1 + \quad \gamma cosx_1 F_\theta + \quad \vartheta F_\alpha = 0 \tag{2.21}$$

$$x_6 \quad = \quad 0 \tag{2.22}$$

en el caso de fricción viscosa, $F_\theta = -b_0 x_2 = 0$, $F_\alpha = -b_1 x_4 = 0$ debido a (2.18) y (2.20), por lo tanto (2.19) y (2.21) quedan de la siguiente manera

$$2\zeta m_b g L sinx_1 \quad = \quad 0 \tag{2.23}$$

$$\gamma cosx_1 m_b g L sinx_1 \quad = \quad 0 \tag{2.24}$$

de las ecuaciones (2.23) y (2.24) se obtiene que

$$sinx_1 \quad = \quad 0$$

de donde se tiene que

$$x_1 \quad = \quad 0, \pm\pi, \pm 2\pi, ...$$

Por lo tanto, el conjunto de puntos de equilibrio del sistema está dado por

$$\begin{aligned} x_1 &= 0, \pm\pi, \pm 2\pi, ... \\ x_2 &= x_4 = x_6 = 0 \\ x_3 &\quad arbitrario \\ x_5 &\quad arbitrario \end{aligned}$$

Sin embargo, de todos estos puntos de equilibrio el que nos interesa como punto de operación para aplicar los esquemas de control es el que corresponde a la posición vertical del vehículo: $x_1 = x_2 = x_4 = x_6 = 0$, x_3, x_5 arbitrarios.

2.2.5. Análisis de estabilidad

En el trabajo [Bonales 2012] se hace el análisis de estabilidad del sistema, en el cual se realiza una linealización en el punto de operación $x_1 = x_2 = x_3 = x_4 = x_5 = x_6 = 0$ considerando que $sinx_1 \approx x_1$ y $cosx_1 \approx 1$. De la misma manera, el sistema (2.9) linealizado usando estas condiciones queda como (2.25)

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \\ \dot{x_3} \\ \dot{x_4} \\ \dot{x_5} \\ \dot{x_6} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ \frac{2\zeta(m_b g L)}{\beta} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ \frac{\gamma(m_b g L)}{\beta} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ \frac{\gamma-2\zeta}{\beta} & \frac{\gamma-2\zeta}{\beta} \\ 0 & 0 \\ \frac{-\gamma+\vartheta}{\beta} & \frac{-\gamma+\vartheta}{\beta} \\ 0 & 0 \\ \frac{D}{RI_\phi} & -\frac{D}{RI_\phi} \end{bmatrix} \begin{bmatrix} \tau_i \\ \tau_d \end{bmatrix} \quad (2.25)$$

el cual tiene exactamente la misma forma que el modelo en [Bonales 2012] en el cual se concluye que el sistema es inestable en el punto de operación debido a que los eigenvalores de la matriz A de la ecuación de estados lineal tienen parte real positiva.

$$\begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix} = \begin{bmatrix} 4.005 \\ -4.005 \\ 0 \\ 0 \end{bmatrix}$$

2.2.6. Grado relativo del sistema

El grado relativo del sistema se calcula derivando la salida del sistema hasta que aparezca explícitamente en ella la entrada de control. La salida de interés en el problema del autobalanceo es el ángulo de inclinación del vehículo, es decir, la salida está dada por

$$y = \theta = x_1$$

derivando una vez se obtiene

$$\dot{y} = \dot{x}_1 = x_2$$

derivando por segunda vez

$$\dot{x}_2 = \frac{\gamma cos x_1 m_b g L sin x_1 - \vartheta\gamma \dot{x_2}^2 sin x_1 + (\gamma cos x_1 - 2\zeta)(\tau_i + \tau_d) + 2\zeta F_\theta + \gamma cos\theta F_\alpha}{\beta(x_1)} \quad (2.26)$$

En la ecuación (2.26) podemos ver que aparece la entrada que son los términos τ_i y τ_d al derivar dos veces la salida, por esta razón el grado relativo del sistema es 2.

El grado relativo del sistema está involucrado en el diseño y la aplicabilidad de algunas técnicas de control, en particular, define el orden de la dinámica cero del sistema.

En el capítulo (4) subsección (4.1.4) se analiza por medio de una gráfica el comportamiento de la dinámica interna del sistema.

2.2.7. Efectos no lineales

Zona Muerta

La zona muerta de un actuador está dada por el nivel de energía de entrada requerida para que el actuador supere su propia inercia y se comience a mover, en el caso de los motores la zona muerta es el rango de voltaje en el que los motores no se mueven, este efecto depende de diferentes factores para cada motor como son la temperatura, la lubricación de la transmisión, el sentido de giro, la frecuencia del PWM, la carga mecánica acoplada al eje del motor, el juego mecánico entre engranes, entre otros. En la figura 2.5 se muestra el efecto de la zona muerta al aplicar voltaje a un motor

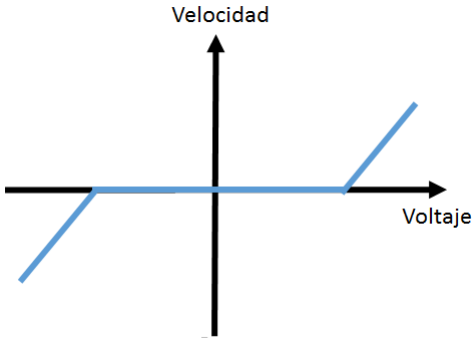


Figura 2.5: Efecto de la zona muerta en el motor de CD

Saturación

La saturación en los actuadores, es el efecto producido debido a los niveles máximos y mínimos de entrada que pueden manejar los actuadores, debido a esto el funcionamiento del actuador está limitado a una región de operación. En la figura 2.6 se muestra el efecto de la saturación al inyectar corriente, se observa que el torque es limitado por un máximo y mínimo debido a la saturación

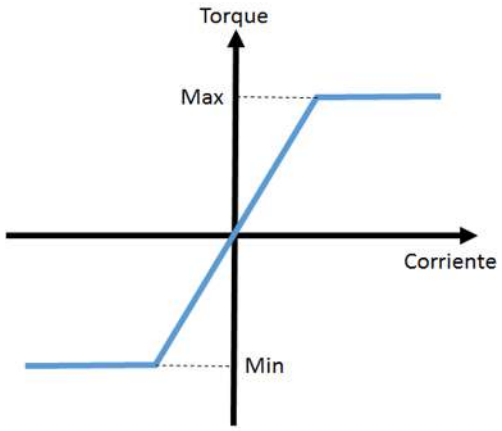


Figura 2.6: Efecto de la saturación en los motores de CD

El uso de la técnica PWM para manejar el voltaje del motor de CD implica necesariamente un efecto de saturación, ya que el ciclo de trabajo del PWM solo puede variarse en el rango de 0 al 100 %.

2.3. Estructura Mecánica

La estructura principal del prototipo fue construida en aluminio porque se requiere una estructura firme y ligera, además de que le da una vista estética al vehículo. Esta parte se compone básicamente de tres elementos: las llantas que harán posible el movimiento del vehículo, la base donde el pasajero se apoya para poder conducir el vehículo y que además sirve de alojamiento para la tarjeta de control, drivers y baterías del prototipo, y el manubrio que le ayudará al conductor al manejo del vehículo. En la figura 2.7 se muestra el prototipo construido. En la siguiente sección se describe la estructura mecánica del vehículo autobalanceado y sus principales componentes.

Figura 2.7: Prototipo construido

2.3.1. La Base del prototipo

La base es la caja que sirve de plataforma donde el usuario deberá ir parado para conducir el prototipo, está fabricada en aleación de aluminio 6061, dureza brinell "95" con peso específico $2.7g/cm^3$ con tornilleria 3/16 de pulgada estándar tipo allen terminado pavonado y 5/16 de pulgada estándar hexagonal galvanizada. El piso y las guardas están construidas en mdf de 9mm con terminado en laca industrial negra. Además la base también sirve de alojamiento para los motores, baterías, tarjetas de control y drivers. En la figura 2.8 se muestra la estructura de la base y los componentes que lleva dentro.

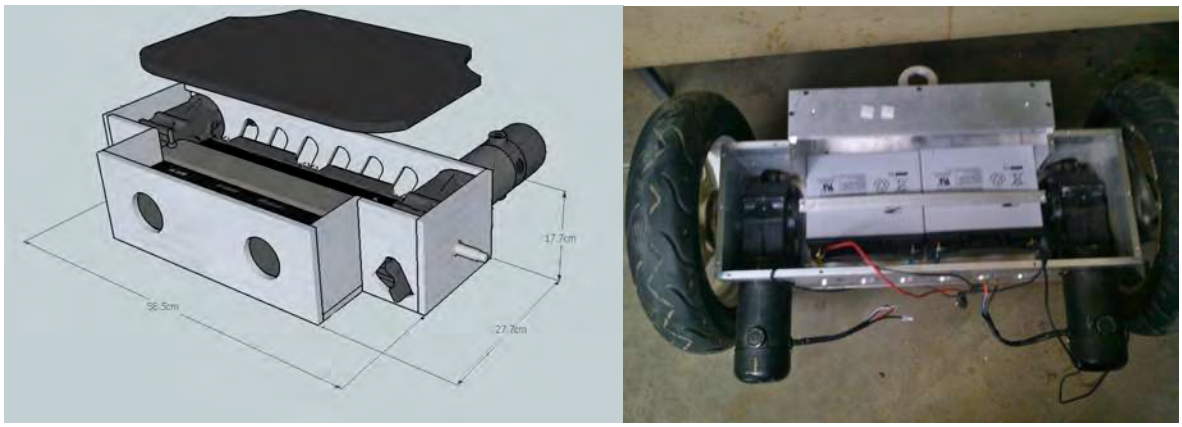


Figura 2.8: Base del prototipo

2.3.2. Llantas

El vehículo usa llantas con un radio exterior de 15" que se encuentran montadas sobre rines de 11". Estos dos elementos son partes usadas de motocicletas, de fácil obtención. Las llantas son montadas directamente al eje del motor por medio de un birlo y de una adaptación de la llanta que se hizo para no modificar el eje del motor. En la figura 2.9 se muestra una de las llantas montadas al vehículo de dos ruedas autobalanceado.

2.3.3. Manubrio

El manubrio del vehículo sirve para dar apoyo al conductor y facilitar la conducción al pasajero y le permite avanzar o frenar el vehículo empujando el manubrio hacia adelante o jalando hacia atrás respectivamente, por otra parte también lleva montado sobre él los mandos de dirección. El manubrio consta de dos piezas de tubo de aluminio de 3.175 cm de radio exterior, un tramo largo de 115 cm unido con uno de 35 cm en la parte superior donde el pasajero se sujeta y donde se encuentran los mandos de dirección. En la figura 2.10 se puede observar el manubrio montado sobre el vehículo de dos ruedas autobalanceado con sus medidas reales.

Figura 2.9: Una de las llantas del prototipo

Figura 2.10: Medidas del manubrio del vehículo

2.3.4. Los motores

Para seleccionar los motores se utilizó el razonamiento simplificado, reportado en [Bonales 2012], donde se hace una estimación del torque máximo que tendrán que desarrollar los motores al moverse sobre una superficie horizontal, para lo cual se considera el diagrama de la figura 2.11 el cual representa el vehículo desplazándose sobre una superficie horizontal, éste es impulsado

por las fuerzas ejercidas por los motores F a una aceleración lineal a. Entonces se tiene que

$$F = M \cdot a + F_M \tag{2.27}$$

donde M es la masa total del prototipo más la del tripulante y F_M es la fuerza equivalente de fricción. Si se requiere una aceleración de $0.5m/s^2$ y despreciando la fricción, el par requerido será $\tau = FR$ donde R es el radio de la llanta. Midiendo los parámetros que se muestran en la Tabla 2.1 como $g = 9.81\text{m/s}^2$, $L = 0.85\text{m}$, $R = 0.2\text{m}$, $D = 0.39\text{m}$, $m_r = 3\text{Kg}$ y $m_b = 130\text{Kg}$, tenemos entonces que

$$\tau = (130)(0.5)(0.2) + F_M = 13Kgf\ m + F_M \quad = 127,53N\ m + F_M$$

y para cada llanta

$$\tau_i = \tau_d = 63.765N\ m + \frac{F_m}{2}$$

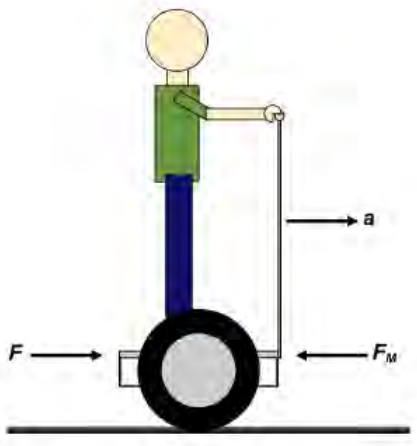


Figura 2.11: Desplazamiento sobre el plano horizontal[Bonales 2012]

Conociendo este resultado y considerando que F_M es desconocida, se toma en cuenta para la selección de los motores que superen un par máximo de $63.765Nm$, en este caso, los motores se compraron usados, son de la marca FRACMO serie X01429/24 (ver figura 2.12) los cuales tienen las características mostradas en la Tabla 2.2.

Tabla 2.2: Características del motor FRACMO serie X01429/24

Tipo transmisión	De gusano
Caja reductora	35:1, $b = 1/35$
Velocidad sin carga	114 RPM
Corriente Nominal	8.5 A
Torque sin carga	8.7 Nm
Peso	6.2 Kg
Medidas	34 cm ×12 cm × 12 cm
Material de la transmisión	Acero
Torque Máximo	80 Nm
Voltaje Nominal	24 V

Con esto se cubre las especificaciones requeridas para mover el vehículo junto con un pasajero encima, a lo largo de un terreno horizontal.

Figura 2.12: Motor FRACMO serie X01429/24 con su caja reductora

2.4. Sensores y Electrónica de Control

Las partes que constituyen el sistema de control del vehículo autobalanceado de dos ruedas consisten en varios elementos electrónicos y electromecánicos que le permitirán al prototipo realizar las tareas necesarias para moverse en la forma deseada por el tripulante, tales como: mantener el equilibrio, avanzar, girar, acelerar o frenar. En la figura 2.13 se muestra un diagrama de bloques de los elementos que forman el sistema de control del vehículo.

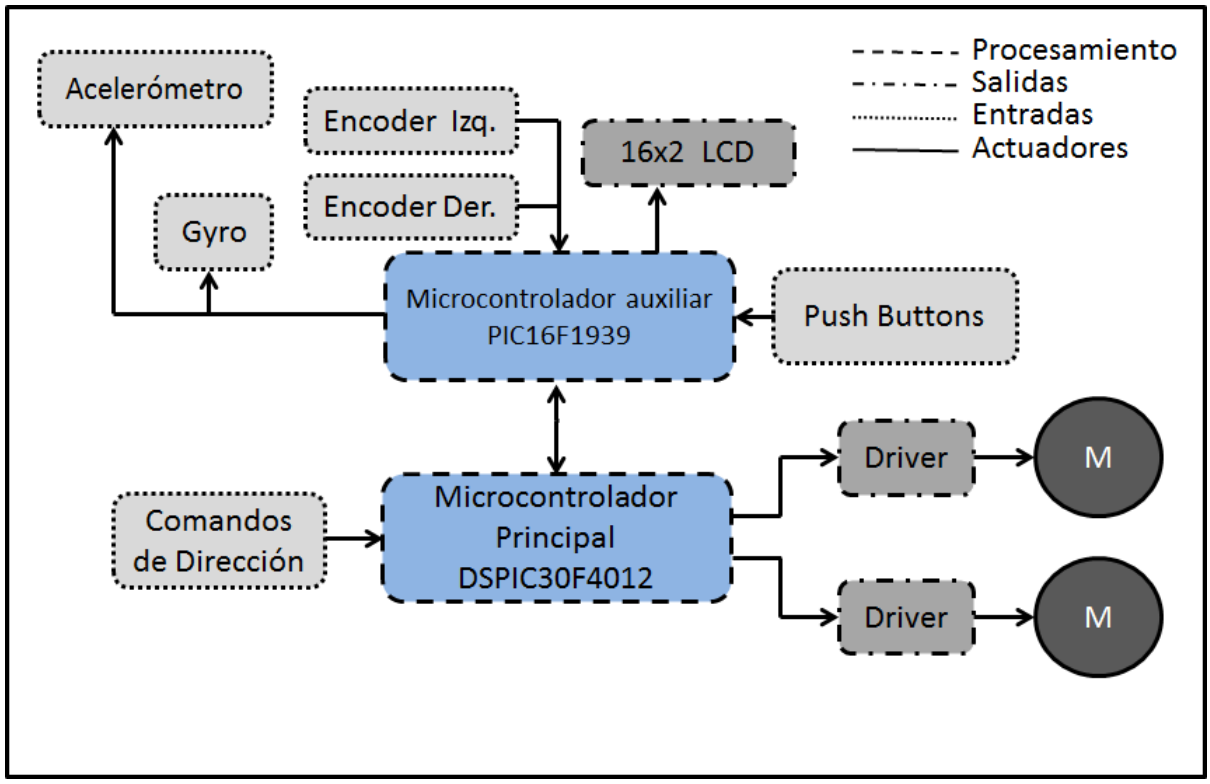


Figura 2.13: Diagrama a Bloques del sistema electrónico del prototipo

donde podemos observar los recuadros con lineas a puntos son las entradas ó sensores del sistema, los recuadros con linea a trazos y puntos son las salidas del sistema , los bloques con lineas a trazos son de procesamiento y los de linea continua son los actuadores.

2.4.1. Módulo de Control y Sensores

Para poder hacer el control del balanceo del vehículo es necesario medir el ángulo de inclinación θ con respecto a la vertical para controlar el equilibrio del vehículo, para poder medir el ángulo θ se utilizó un par acelerómetro-giroscopio de tecnología MEMS. La medición de ángulo y velocidad angular proporcionada por estos dos sensores es procesada por un filtro de Kalman digital para obtener una medición mejorada del ángulo de inclinación θ de acuerdo a la figura 2.14.

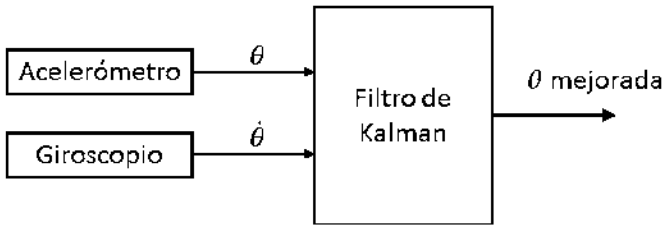


Figura 2.14: Diagrama de bloques del filtro de Kalman

El algoritmo del filtro de Kalman utilizado, así como su sintonización son descritos a detalle en [Bonales 2012], para obtener un mejor control se agregan al prototipo un par de encoders montados a cada eje de los motores para poder medir y controlar la velocidad de avance lineal del vehículo, para la visualización de variables el operador cuenta con un display de 16x2 caracteres y un panel de varios botones para las diferentes opciones del despliegue de variables, y para que el usuario manipule la dirección de avance se incorporan dos palancas de mando tipo joystick. En la siguiente sección se describen los diferentes tipos de acelerómetros y giroscopios y la manera de medir el ángulo θ a partir del conjunto de los dos dispositivos.

Para poder procesar la información obtenida de los sensores se utilizaron dos microcontroladores, el PIC16F1939 para la adquisición de datos y el DSPIC30F4012 para la implementación del esquema de control que mantendrá el vehículo balanceado y en la dirección deseada por el tripulante. Para todo esto se construyó una tarjeta que incluye tanto los microcontroladores como el par de sensores acelerómetro-giroscopio, la tarjeta construida se muestra en la figura 2.15.

2.4.2. Sensores

La medición del ángulo θ se realiza mediante un par acelerómetro-giroscopio con el cual se obtiene una medida mejorada del ángulo comparada con la que se obtendría si se usara el acelerómetro o el giroscopio solos. Los dos dispositivos seleccionados tienen interfaz de comunicación I^2C lo cual disminuye el ruido de medición debido a que toda la instrumentación está dentro del mismo chip, a continuación veremos una descripción de estos dispositivos y su funcionamiento, por otro lado la medición de la velocidad de las ruedas $\dot{\alpha}$ permite mejorar el movimiento deseado, ésta es medida por medio de un encoder óptico incremental con salidas en cuadratura montados uno sobre cada motor. A continuación se describen los elementos usados para el sensado.

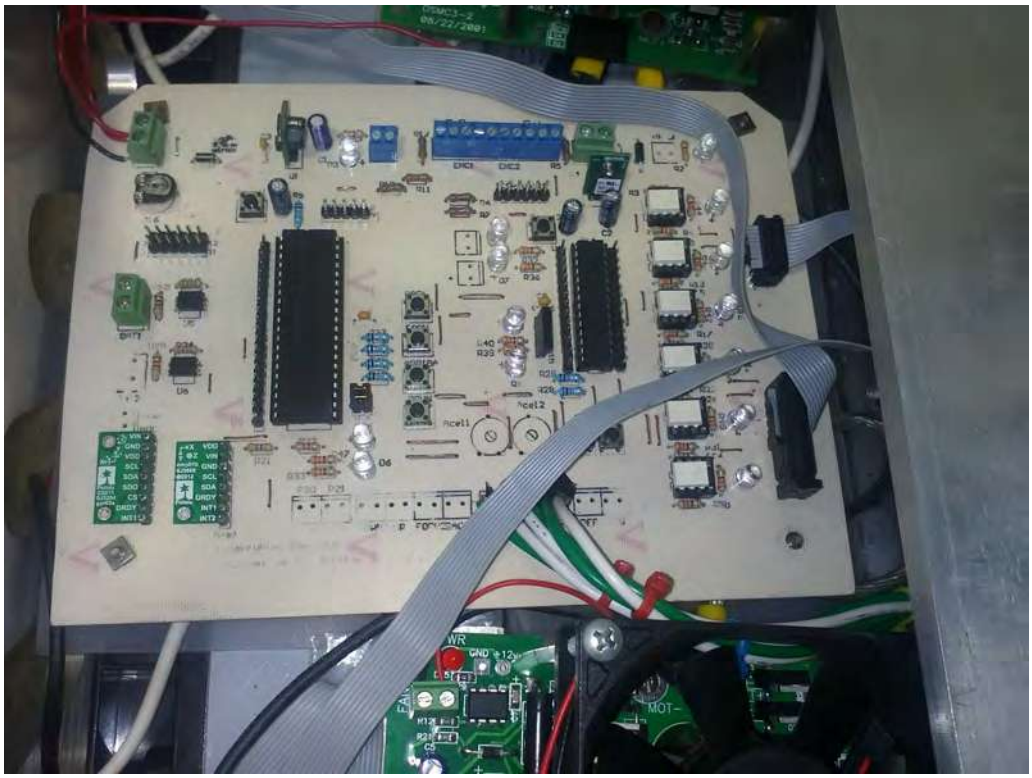


Figura 2.15: Tarjeta de Control construida

2.4.3. Acelerómetro

Los acelerómetros son instrumentos que ayudan a medir las aceleraciones del objeto sobre el cual se montan, en la actualidad se usan acelerómetros con tecnología MEMS la cual permite tener todo el sistema mecánico y electrónico dentro del mismo chip. Para poder medir el ángulo de inclinación mediante la aceleración que sufre el vehículo, tomamos en consideración dos cosas: que la aceleración de la gravedad terrestre es $1g = 9.81m/s^2$ y que al inclinar el acelerómetro con respecto a la vertical esta aceleración se descompone en dos componentes, como se muestra en la figura 2.16. Se puede observar que mediante cualquiera de las mediciones en los ejes Y y X se puede obtener el ángulo, para el caso de usar el eje X se puede deducir el ángulo θ de la siguiente ecuación $g \cdot sin(\theta) = g_x$, o bien si usamos el eje Y, sería $g \cdot cos(\theta) = g_y$, todo esto depende de la posición en que este montado el acelerómetro dentro del vehículo. El poder de cálculo de los microcontroladores permite usar funciones trigonométricas para obtener un valor muy aproximado de θ, una consideración importante para esta medición es que esto se cumple únicamente si la única aceleración ejercida sobre el acelerómetro es la fuerza de gravedad, lo cual no es posible en general, ya que el movimiento

del vehículo introduce otras aceleraciones que contaminan la medición. Por este motivo y para no tener lecturas erróneas de θ se utiliza un giroscopio para corregir el error por aceleraciones parásitas y poder obtener una medición más exacta de θ. Para la implementación en el vehículo autobalanceado se utilizó el acelerómetro LSM303DHLHC, el cual tiene lectura de tres ejes, con comunicación I^2C, además de tener incorporado un magnetómetro dentro del mismo chip que puede servir para ayudar a la dirección del vehículo.

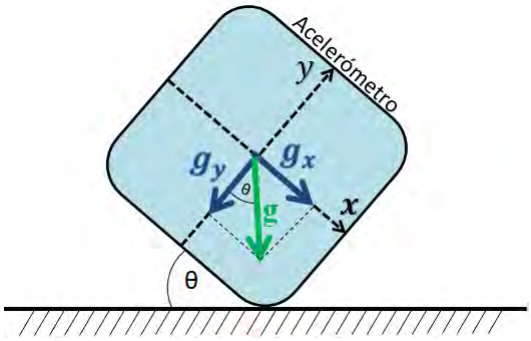


Figura 2.16: Mediciones del acelerómetro en los ejes X y Y

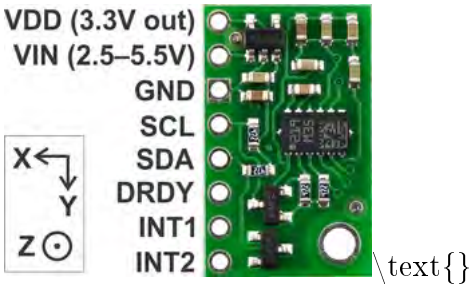


Figura 2.17: Diagrama de conexión del acelerómetro LSM303DHLHC

2.4.4. Giroscopio

El giroscopio es un instrumento que permite medir la velocidad angular del objeto sobre el cual se monta el sensor. Debido al error de la medición del acelerómetro producido por las aceleraciones externas del vehículo se opta por introducir un giroscopio para poder obtener una medida mejorada de θ. El giroscopio en el vehículo da la medición de la velocidad angular $\dot{\theta}$ con la cual mediante el filtro de Kalman o algún otro filtro podemos estimar el valor del ángulo θ. En el prototipo realizado se usa un giroscopio L3DG20 de 3 ejes con comunicación I^2C.

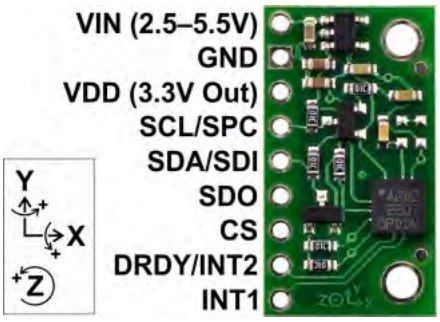


Figura 2.18: Diagrama de conexión del Giroscópio de 3 ejes L3DG20

2.4.5. Encoders Ópticos

Los encoders son codificadores rotatorios, también conocidos como codificadores de eje o generadores de pulsos, suelen ser dispositivos electromecánicos usados para convertir la posición angular de un eje a un código digital. Los encoder ópticos se componen básicamente de dos elementos, un sensor óptico y un disco con ranuras por las cuales pasa la señal para los sensores ópticos, el disco se monta sobre el eje del dispositivo del que se quiere medir la velocidad angular, de esta manera si el disco tiene 360 marcas se obtendrá un pulso por cada grado que gire el eje, como se muestra en la figura 2.19.

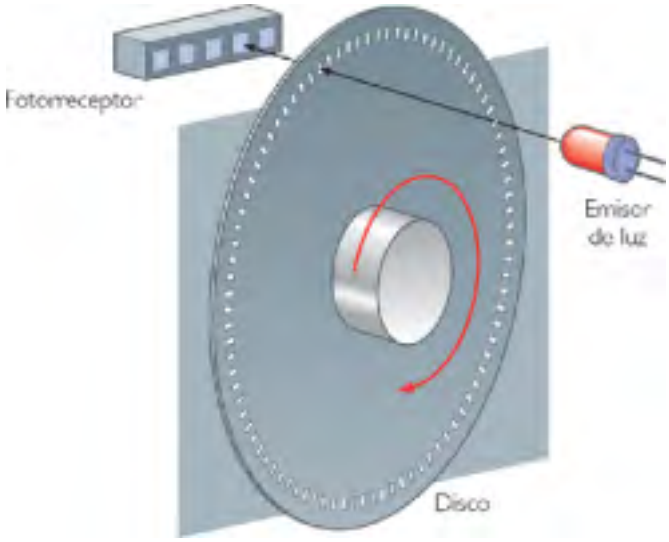


Figura 2.19: Estructura física de un encoder

Una clasificación de los encoders ópticos según el tipo de información sobre la posición que genera sería:

- Encoder Incremental. Es un encoder que indica por medio de dos canales en cuadratura o desfasadas 90^{0}, la velocidad y dirección de giro de la flecha del motor, algunos cuentan con un canal extra para indicar que se ha girado una vuelta completa, las señales generadas por este encoder son las de la figura 2.20.

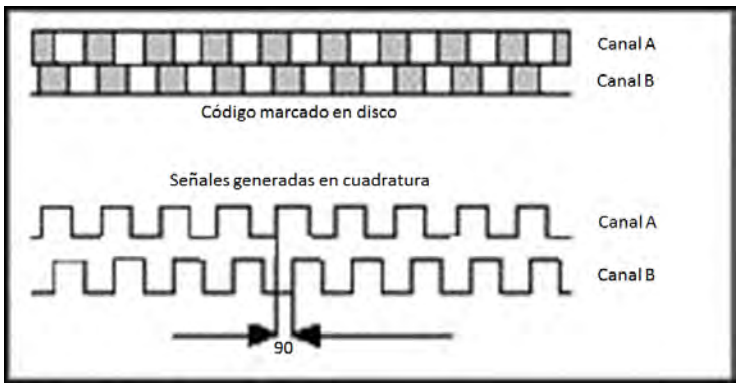


Figura 2.20: Señales en cuadratura del encoder incremental

- Encoder Absoluto. A diferencia del encoder incremental el encoder absoluto indica en que posición esta exactamente la flecha acoplada al disco, esto lo hace por medio de un código único grabado en el disco para cada ángulo en el disco acoplado como se muestra en la siguiente figura.

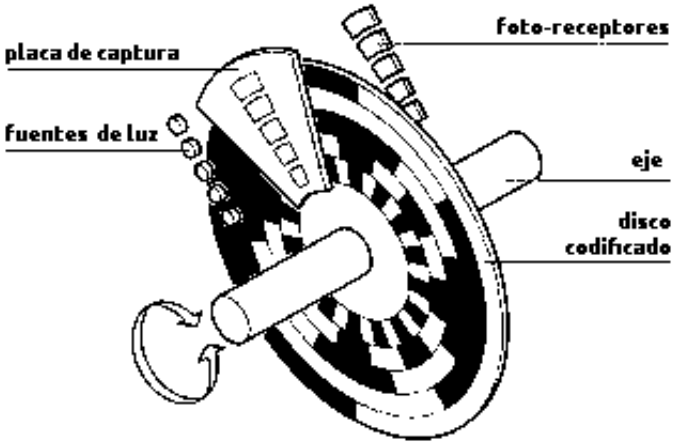


Figura 2.21: Encoder absoluto

Para la aplicación en el vehículo se utilizaron encoders incrementales de la marca Avago, modelo HEDS-5645 G13 que cuentan con dos canales con salida en cuadratura y un

canal extra que da un pulso por vuelta, estos se alimentan con 5 V. En la figura 2.22 se muestra el encoder utilizado en el prototipo.

Figura 2.22: Encoder de la marca Avago

2.4.6. Microcontroladores

Los dos microcontroladores son la parte más importante de todo el diseño, ya que se encargan de obtener la información de los sensores y procesarla para poder generar una entrada que permita controlar el balanceo del vehículo. Además se encargan de desplegar las variables del proceso por medio de un LCD de 16x2 caracteres y de monitorear el estado de la batería. Se utilizaron dos microcontroladores para repartir los procesos y tareas que tienen que llevar a cabo para mantener el equilibrio del vehículo. A uno de ellos se le asignó la tarea de sensado, y al otro la tarea de control. Para la parte de sensado los requerimientos mínimos para el microcontrolador son:

- 2 Canales de ADC para sensado del nivel de las baterías.
- 17 pines GPIO para el manejo del LCD, leds indicadores y entradas de los push button instalados.
- Puerto de comunicación USART para la comunicación entre microcontroladores.
- Puerto de comunicación I^2C para poder comunicarse con los sensores.
- Ser económico de fácil acceso y programación.
- Velocidad suficiente para hacer el sensado.

Todas estas características deseadas nos llevan a la elección del microcontrolador PIC16F1939 con un precio de USD $3.64 y las características que se muestran en la figura 2.23.

Device	Program Memory Flash (words)	Data EEPROM (bytes)	SRAM (bytes)	I/O's	10-bit A/D (ch)	CapSense (ch)	Comparators	Timers 8/16-bit	EUSART	I²C™/SPI	ECCP	CCP	LCD
PIC16F1933 PIC16LF1933	4096	256	256	25	11	8	2	4/1	Yes	Yes	3	2	16(1)/4
PIC16F1934 PIC16LF1934	4096	256	256	36	14	16	2	4/1	Yes	Yes	3	2	24/4
PIC16F1936 PIC16LF1936	8192	256	512	25	11	8	2	4/1	Yes	Yes	3	2	16(1)/4
PIC16F1937 PIC16LF1937	8192	256	512	36	14	16	2	4/1	Yes	Yes	3	2	24/4
PIC16F1938 PIC16LF1938	16384	256	1024	25	11	8	2	4/1	Yes	Yes	3	2	16(1)/4
PIC16F1939 PIC16LF1939	16384	256	1024	36	14	16	2	4/1	Yes	Yes	3	2	24/4

Figura 2.23: Característcas principales del microcontrolador PIC16F1939

Para la parte de control se seleccionó en base a los requerimientos mínimos del diseño:

- Buena velocidad para procesamiento de los datos y ejecución del esquema de control.
- 2 Canales ADC para la lectura de los Joysticks de dirección del vehículo.
- 2 Canales de control de motores
- 6 pines GPIO para entradas de push button y salida de leds indicadores.
- Puerto de comunicación USART para comunicación entre microcontroladores y para el monitoreo de variables en la PC.

Device	Pins	Program Mem. Bytes/ Instructions	SRAM Bytes	EEPROM Bytes	Timer 16-bit	Input Cap	Output Comp/Std PWM	Motor Control PWM	10-bit A/D 1 Msps	Quad Enc	UART	SPI	I²C™	CAN
dsPIC30F4012	28	48K/16K	2048	1024	5	4	2	6 ch	6 ch	Yes	1	1	1	1
dsPIC30F4011	40/44	48K/16K	2048	1024	5	4	4	6 ch	9 ch	Yes	2	1	1	1

Figura 2.24: Características principales del microcontrolador DSPIC30F4012

Con una velocidad de oscilador de 120 MHz (30 MIPS). El microcontrolador DSPIC30F4012 es lo suficientemente rápido para llevar a cabo las operaciones requeridas para implementar

los controladores además se puede observar en la figura 2.24 que cumple con las otras características deseadas.

2.4.7. Drivers

Para el control de los motores se utilizaron dos OSMC (Open Source Motor Controller), este es un circuito controlador con un Puente H de alto poder diseñado para controlar motores de DC de imán permanente de 13 a 50 volts [OSMC 2014]. Las características principales se muestran en la Tabla 2.3. En la figura 2.25 se muestra una fotografía del driver utilizado

Tabla 2.3: Características del Driver OSMC

Voltaje de alimentación	13V a 50V
Corriente de salida(Continua)	160A
Peso	0.27Kg
Mosfets	16 IRFB3207
Resistencia	0.0026 ohm
Controlador de conmutación	Intersil HIP4081A
Fuente de alimentación de salida	12V 0.5A

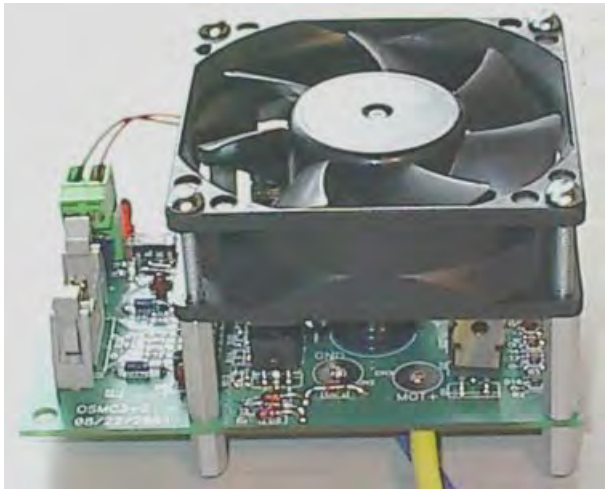


Figura 2.25: Driver OSMC (Open source motor controller)

2.4.8. Acoplamientos Ópticos

Entre las salidas de control y los drivers y en la entrada de los canales analógicos que se usan para leer el nivel de voltaje de las baterías existe un aislamiento para separar la

parte lógica de la parte de potencia. Este aislamiento se hace por medio de optoacopladores, para la entrada de las baterías se usa un optoacoplador con un optotransistor como receptor (PC817) para poder bajar el voltaje a un nivel adecuado para el microcontrolador, y para las salidas a los drivers se usó un optoacoplador con salida lógica de alta velocidad (H11L1) para aislar las señales de PWM que son enviadas al driver. En la figura 2.26 se muestran los optoacopladores usados en la construcción del vehículo.

(a) H11L1 (b) PC817

Figura 2.26: Optoacopladores usados en la construcción del vehículo.

2.4.9. Comunicaciones

Como se menciona al inicio de este capítulo la tarjeta principal cuenta con dos microcontroladores, uno para sensado y otro para control, el microcontrolador que se encarga de sensar el ángulo de inclinación θ obtiene las lecturas del acelerómetro y giroscopio por medio de una red serial I^2C, el microcontrolador encargado del control le pide los datos de los sensores al microcontrolador de sensado por medio de comunicación USART por lo cual se mencionan a continuación los aspectos más importantes de cada uno de estos buses de comunicación.

2.4.9.1. Bus I^2C

El bus I^2C fue desarrollado por Philips semiconductors (ahora NXP semiconductors) en los 80's y hoy en día esta implementado en más de 1000 diferentes tipos de circuitos integrados y fabricado por más de 50 compañías en el mundo. Este bus es de comunicaciones serie asíncrono de dos líneas (SDA y SCL) que permite la comunicación entre múltiples dispositivos (en teoría 1000), todos conectados paralelamente a las dos líneas. Las transferencias de datos siempre se realizan entre dos dispositivos a la vez y en una relación maestro-esclavo. Las características de este bus son:

- La comunicación es a través de sólo dos líneas; una línea de datos serial (SDA) y una línea de reloj (SCL).

- Cada dispositivo conectado a el bus es direccionable por software con una dirección única y una relación maestro-esclavo existe en todo tiempo.

- El bus puede ser multi-maestro incluyendo detección de colisión para prevenir corrupción de información si dos maestros inician transferencia simultáneamente.

- La transmisión puede ser bidireccional de 8 bits alcanzando transferencias de datos hasta de 100Kbit/ss en el modo Standard, hasta 400kbit/s en el modo Fast, hasta 1 Mbit/s en el modo Fast Plus, y hasta 3.4 Mbit/s en el modo High-Speed.

- La transmisión puede ser unidireccional de 8 bits con tasa de transferencia de hasta 5 Mbit/s en modo Ultra Fast.

- El número de dispositivos que pueden ser conectados es limitado únicamente por la capacitancia del bus.

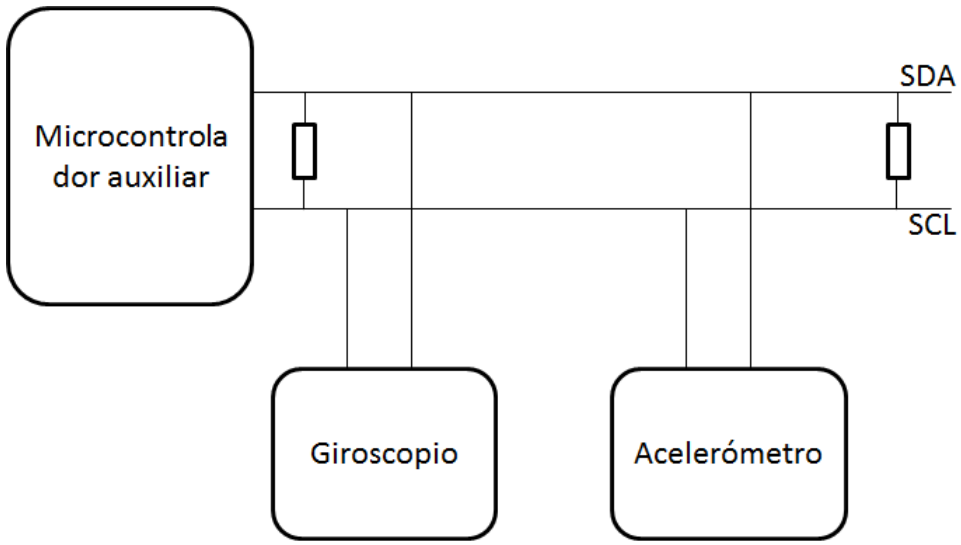


Figura 2.27: Diagrama de conexión del bus I^2C en el vehículo.

La estructura de la red utilizada en el vehículo es la mostrada en la figura 2.27. La transferencia de datos por el bus es en forma de paquetes, como se ve en la figura 2.28 una transferencia empieza con una señal de START y termina con una señal de STOP. Entre estas señales van los datos a mandar. Cada dato debe ser de 8 bits y debe ir seguido un noveno bit, llamado bit de reconocimiento donde un 0 es un ACK y un 1 es un NACK.

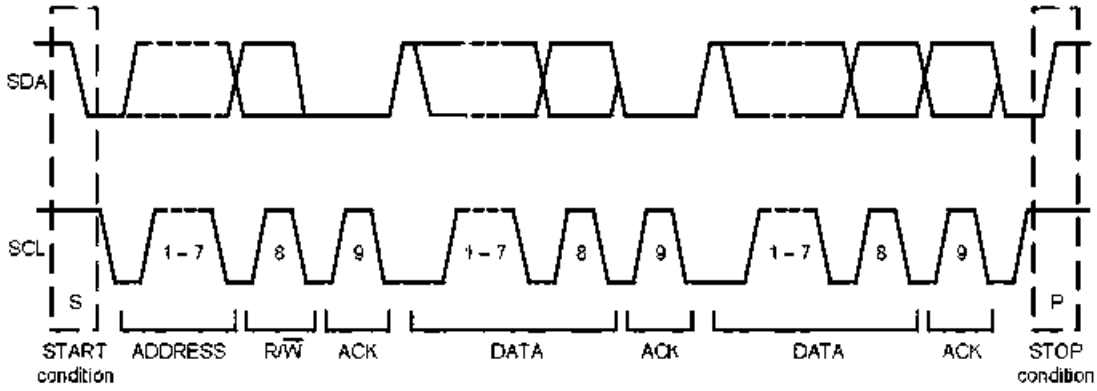


Figura 2.28: Estructura básica de una trama de comunicación por el bus I^2C

2.4.9.2. La USART

La USART es un puerto serie de transmisión o recepción síncrona (half duplex) o asíncrona (full duplex) punto a punto, que viene incorporado en muchos microcontroladores para comunicarse usando el estándar RS-232. En el modo full-duplex se utilizan los pines TX y RX del dispositivo para transmisión y recepción respectivamente como se muestra en la figura 2.29, los datos enviados son de 8 bits, al formato de la trama se le añade un bit de START al inicio y un bit de STOP al final (figura 2.30) pudiendo añadir un noveno bit de datos que puede ser utilizado paridad.

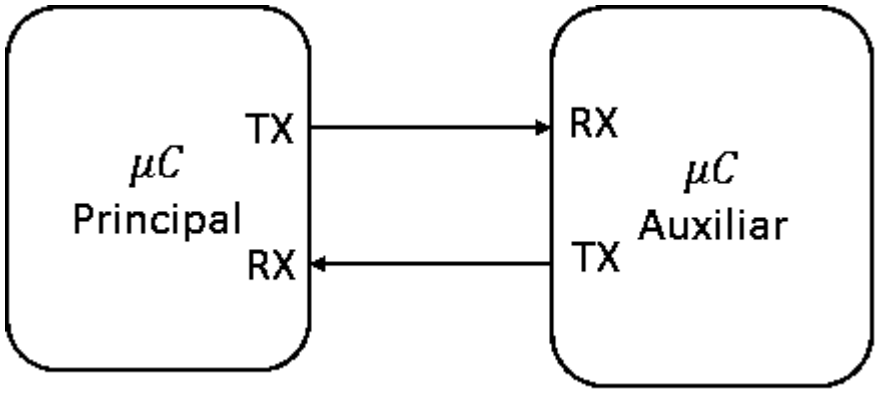


Figura 2.29: Diagrama de conexión de la USART

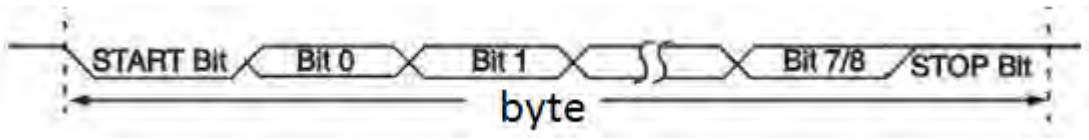


Figura 2.30: Ejemplo de trama de envío de datos por la USART.

2.4.10. Panel de visualización y control

El vehículo de dos ruedas autobalanceado cuenta con un panel para poder visualizar las variables principales del sistema, niveles de batería, y modo de operación, el panel esta formado por un LCD de 16x2 caracteres tres botones para poder moverse entre mensajes, 4 LEDs indicadores y un botón sostenible para indicar el modo de operación con piloto o sin piloto (ver figura 2.31).

Figura 2.31: Panel de visualización y control del vehículo

La operación del panel de visualización es descrita en el capítulo 5.

2.4.11. Comandos de dirección

Para poder girar, el vehículo tiene incorporados dos palancas de mando tipo joystick uno para girar a la derecha y otro para girar a la izquierda, los joystick (figura 2.32) son arreglos de potenciómetros que permiten comandar fácilmente un cambio de dirección por medio del dedo pulgar del conductor.

Figura 2.32: Joystick utilizado para el giro del vehículo.

2.4.12. Alimentación del vehículo

Toda la circuitería del vehículo autobalanceado depende de dos baterías de gel de 12 Volts, las cuales son conectadas en serie para proporcionar los 24 volts requeridos para alimentar los motores de CD a través de los drivers OSMC, las dos son de la marca Ritar modelo RT12200 (ver figura 2.33). Las características principales de la batería se pueden ver en la Tabla 2.4

Figura 2.33: Batería Ritar RT12200

Tabla 2.4: Características principales de las baterías RT12200

Característica	RT12200
Marca	Ritar
Tipo	Gel
Capacidad	20 AH
Ciclo de carga	6 Hrs
Peso	5.9 Kg
Dimensiones	18.1 cm×7.7 cm ×16.7 cm

La tarjeta principal se alimenta de 12 volts los cuales son tomados de una fuente incluida en los drivers OSMC ver la tabla 2.3, los cuales son convertidos al nivel de 5 volts mediante un regulador conmutado de 5 volts OKI-78SR05 que es un convertidor DC-DC de 1.5 A (ver figura 2.34).

Figura 2.34: Regulador de 5 volts OKI-SR05

2.5. Costos

La tabla muestra los costos requeridos para la construcción del vehículo autobalanceado.

Tabla 2.5: Costos construcción del vehículo (pesos mexicanos)

Elemento	Precio por unidad	Cantidad	Precio total
Estructura de Aluminio	$10,500	1	$10,500
Baterías	$600.00	2	$1,200
OSMC	$2,970	2	$5,940
Llantas	$150	2	$300
Motores	$2,000	2	$4,000
Acelerómetro	$300	1	$300
Giroscopio	$300	1	$300
Cargador de batería	$500	1	$500
Microcontroladores	$300	1	$300
Miscelaneos	$1,000	1	$1,000
Total			$24,340

Se puede ver que el costo es mucho menor comparado con el de un Segway, si se produce en linea pueden reducirse bastante los costos de materiales.

CAPÍTULO 3

Esquemas de Control del Prototipo

En este capítulo se presentan las técnicas de control probadas en simulación y en el vehículo real. La parte del esquema de control es la de mayor relevancia en la operación del vehículo, su objetivo es que el controlador mantenga estable la posición vertical del vehículo, permitiendo así que el piloto se desplace con comodidad, sin que el vehículo se caiga. Para ello, éste se sustenta en todos los sistemas descritos en el capítulo anterior.

En esencia las técnicas de control probadas en el vehículo fueron un PID discreto con anti-windup y compensación de zona muerta, un modo deslizante supertwisting y un controlador PD difuso mínimo, los cuales tienen el ángulo de inclinación θ como variable a controlar y el voltaje aplicado en las terminales de los motores v_i y v_d como acción de control.

3.1. Controlador PID

En la actualidad en la industria y en muchos otros procesos y sistemas el controlador más usado es el PID, el cual es un mecanismo de control por retroalimentación que calcula el error entre un valor medido por los sensores y el valor deseado, para aplicar una entrada que corrija y ajuste el proceso. El cálculo de esta entrada se da en tres componentes distintas: la proporcional, la integral, y la derivativa. En el diagrama de bloques de la figura 3.1 se muestra como está conformado el controlador PID con sus componentes.

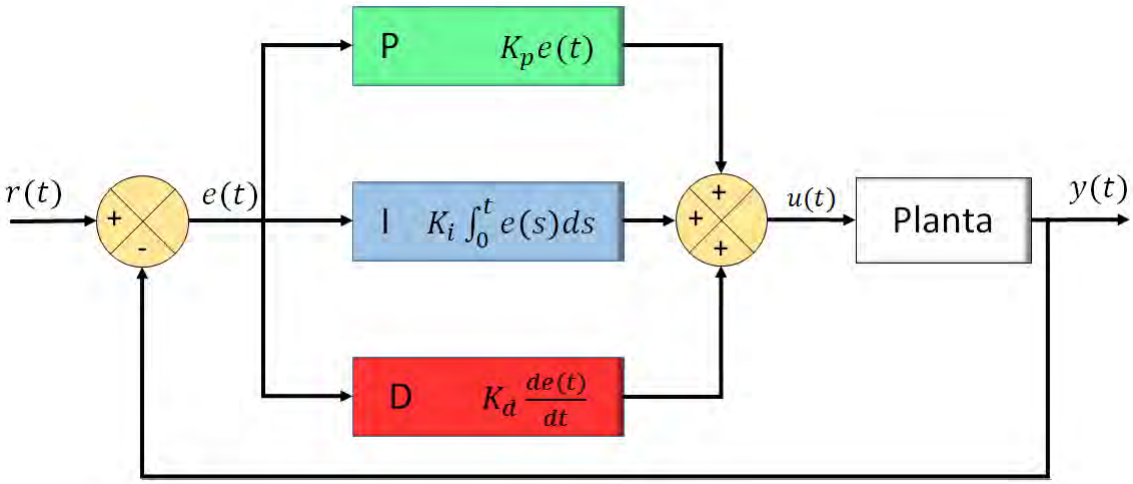


Figura 3.1: Diagrama a bloques de un controlador PID

La ecuación que representa el diagrama de la figura 3.1 es

$$u(t) = K_p \left(e(t) + \frac{1}{T_i} \int^t e(s)ds + T_d \frac{de(t)}{dt} \right) \tag{3.1}$$

donde el error e es la diferencia entre la referencia y la salida del proceso (variable medida). K_p es la ganancia proporcional del controlador, T_i el tiempo de integración, y T_d el tiempo derivativo. El controlador PID era originalmente implementado usando tecnología analógica, hoy en día los controladores PID son implementados digitalmente. En el dominio de Laplace, el controlador PID se puede escribir como:

$$U(s) = K_p \left[1 + \frac{1}{T_i s} + T_d s \right] E(s) \tag{3.2}$$

Un derivador puro no puede, y no debe ser implementado, porque esto ocasionará una muy grande amplificación del error medido, por lo cual la ganancia de la derivada del error debe ser limitada. Esto puede ser logrado aproximando la función de transferencia ideal de la parte derivativa como se muestra a continuación [Astrom 1997]

$$T_d s \approx \frac{T_d s}{1 + s T_d / N} \tag{3.3}$$

3.1.1. Discretización

Una posible realización de un controlador PID discreto viene dado por la aproximación de cada término de (3.2), por una versión discreta.

Aproximando la derivada por diferencias hacia adelante, la integral por una suma de rectángulos y tomando en cuenta (3.3) se obtiene la siguiente realización [Astrom 1997]

$$U(z) = K_p \left[1 + \frac{T}{T_i \left(1 - z^{-1}\right)} + \frac{T_d \left(1 - z^{-1}\right) N}{NT + \left(1 - z^{-1}\right)} \right] E(z) \tag{3.4}$$

Entonces la discretización en el dominio del tiempo está dada por

$$u(kT) = P(kT) + I(kT) + D(kT) \tag{3.5}$$

donde

$P(kT)$ es el término Proporcional

$I(kT)$ es el término Integral

$D(kT)$ es el término Derivativo

cada uno de estos términos son

$$P(kT) = K_p e(kT) \tag{3.6}$$

$$I(kT) = I(kT - T) + \frac{K_p T}{T_i} e(kT - T) \tag{3.7}$$

$$D(kT) = \frac{T_d}{T_d + NT} D(kT - T) - \frac{K_p T_d N}{T_d + NT} (y(kT) - y(kT - T)) \tag{3.8}$$

los parámetros del PID se muestran en la tabla 3.1

Tabla 3.1: Parámetros del controlador PID discreto

T	Periodo de muestreo
K_p	Ganancia proporcional
$e(kh)$	Error actual
$e(kT-T)$	Error en un instante anterior
T_i	Tiempo integral
$I(kT-T)$	Integral del error en un instante anterior
T_d	Tiempo derivativo
N	Constante para limitar termino derivativo a altas frecuencias
$D(kT-T)$	Acción derivativa en un instante anterior
$y(kT)$	Salida actual
$y(kT-T)$	Salida en un instante anterior

Para poder implementar la ley de control en algún microcontrolador ó computadora digital se puede seguir el algoritmo mostrado en la figura (3.2). El ciclo mostrado en la figura 3.2 se ejecuta una vez cada periodo de muestreo.

3.1.2. Selección e implementación del periodo de muestreo

La selección del periodo de muestro se hace de acuerdo al experimento realizado sobre el microcontrolador que es detallado en el capítulo 4

3.1.3. Antiwindup

El antiwindup se implementó usando una condición para calcular el valor del término integral del controlador, usando como referencia el valor de saturación de los actuadores o en este caso el valor máximo y mínimo que puede alcanzar la entrada de control $u(t)$.

- Si mínimo valor de entrada$<u(t)<$ máximo valor de entrada, entonces

$$I(kT) = I(kT-T) + \frac{K_p T}{T_i} e(kT-T)$$

de lo contrario

$$I(kT) = I(kT-T)$$

De esta manera se asegura que el término integral no aumente más, cuando los actuadores ya han sido saturados.

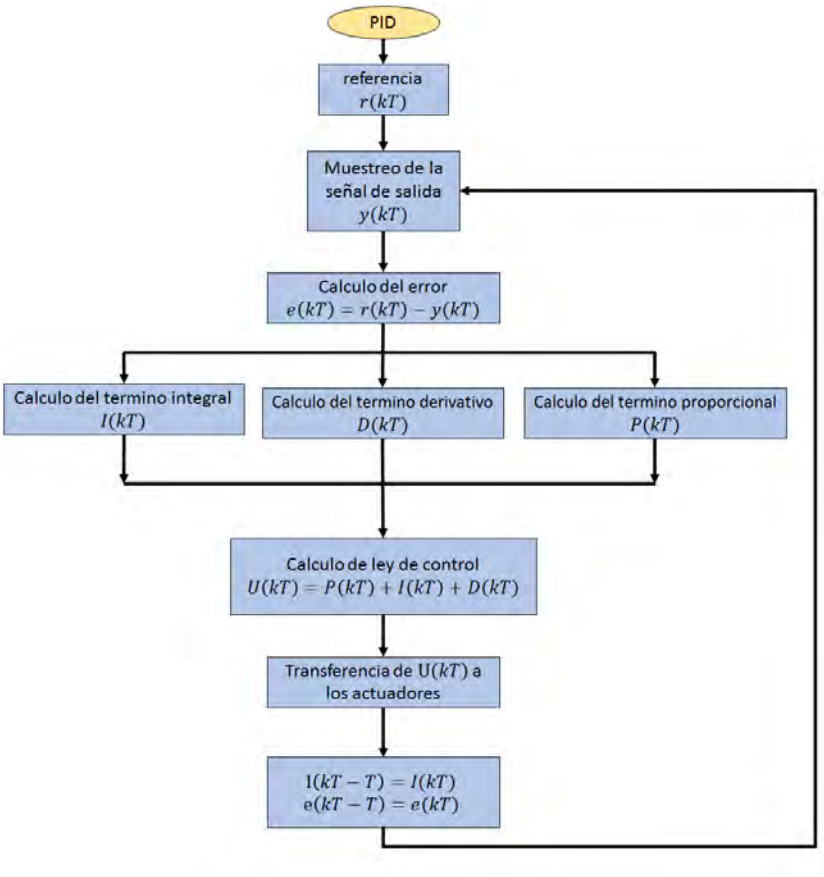


Figura 3.2: Algoritmo de programación del PID

3.1.4. Compensación de la zona muerta

Para operar, los motores requieren un nivel de voltaje para poder romper su estado de reposo y comenzar a moverse, esta zona de operación es conocida como la zona muerta del motor, esta zona puede variar dependiendo el tipo de motor, el sentido de giro y la frecuencia de la señal PWM con la que se esta variando el voltaje en los motores, esto ocasiona que se tenga que medir el voltaje en el cual los motores comienzan a moverse para poder ser considerados en la parte del control del vehículo de dos ruedas autobalanceado y así obtener mejores resultados.

3.2. El Controlador PD Difuso mínimo

Este controlador se basa en el uso de lógica difusa para lo cual se explican algunos conceptos base

3.2.1. Conceptos

3.2.1.1. Conjunto difuso

Es un conjunto que puede contener elementos con grados parciales de pertenencia, a diferencia de los conjuntos clásicos en los que los elementos pueden "pertenecer" o "no pertenecer" a dichos conjuntos. Desde el punto de vista de que se aplican palabras a la definición de cualquier propiedad por ejemplo: mujeres altas, edificios viejos, hombres bajos, etc.

3.2.1.2. Función de membresía

Es una curva que determina el grado de pertenencia de los elementos de un conjunto. Se denota generalmente por μ y se puede adoptar valores entre 0 y 1.

3.2.1.3. Inferencia difusa

Se llama reglas difusas al conjunto de proposiciones **IF-THEN** que modelan al problema que se quiere resolver. Una regla difusa tiene la forma:

"Si u es A entonces v es B"

donde A y B son conjuntos difusos definidos en los rangos "u" y "v" respectivamente. Una regla expresa un tipo de relación entre los conjuntos A y B.

3.2.1.4. Sistema basado en técnicas de lógica difusa

El esquema de un sistema basado en técnicas de lógica difusa se presenta en la figura 3.3.

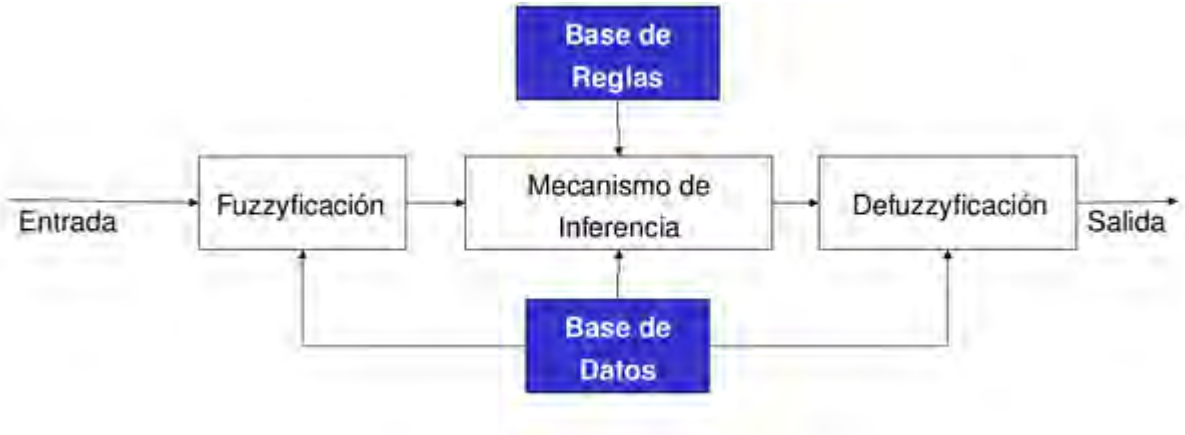


Figura 3.3: Diagrama de bloques de un sistema que usa lógica difusa.

3.2.1.5. Fuzzyficación

Bloque en el que a cada variable de entrada se le asigna un grado de pertenencia a cada uno de los conjuntos difusos que se han considerado, mediante las funciones características asociadas a estos conjuntos difusos. La entrada a este bloque son valores concretos de las variables de entrada y las salidas son grados de pertenencia a los conjuntos difusos considerados.

3.2.1.6. Bloque de inferencia

Bloque que, mediante los mecanismos de inferencia, relaciona conjuntos difusos de entrada y salida y que representan a las reglas que definen el sistema. La entrada a este bloque son conjuntos difusos (grados de pertenencia) y las salidas son también conjuntos difusos, asociados a la variable de salida.

3.2.1.7. Defuzificación

Bloque en el cual a partir del conjunto difuso obtenido en el mecanismo de inferencia y mediante los métodos matemáticos de defuzzificación, se obtiene un valor concreto de la variable de respuesta, es decir, la salida.

3.2.2. Estructura de un controlador incremental

La estructura de un controlador difuso incremental se basa principalmente en los siguientes elementos: el error e que es la diferencia entre la referencia r y la salida y, el bloque de fuzzificación donde se pasa de variables numéricas a variables lingüísticas por medio de las funciones de membresía, las reglas de inferencia donde se decide que acción tomar dependiendo la pertenencia que tenga a cada función tanto el error como el cambio del error, y la defuzzificación donde el resultado de estas inferencias se pasa a variables numéricas para poder aplicarse a la planta o sistema (ver figura 3.4) [Pham 2001].

En la figura (3.4) se incluye un integrador a la salida del bloque de defuzzificación, este integrador es necesario sólo sí la planta no lo tiene de forma natural, en este caso la planta tiene un integrador natural por lo cual no se utiliza el integrador.

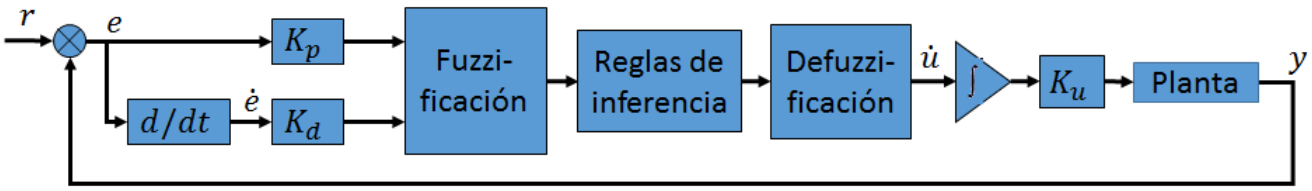


Figura 3.4: Estructura controlador PD difuso mínimo.

3.2.3. Definición de las funciones de membresía

Como el controlador es mínimo el error sólo puede tomar dos valores: error negativo (e_n) y error positivo (e_p), de igual manera el cambio del error puede tomar sólo dos valores: cambio de error negativo (r_n) y cambio de error positivo (r_p), para la acción de control se establecen tres valores, positivo (p), negativo (n) y cero (0) [Pham 2001]. Así que para el error y el cambio del error se definen, para cada una, sólo dos subconjuntos difusos: n y p y para la acción de control tres: n, 0 y p. Por lo cual las respectivas funciones de membresía son las mostradas en las siguientes figuras 3.5, 3.6 y 3.7.

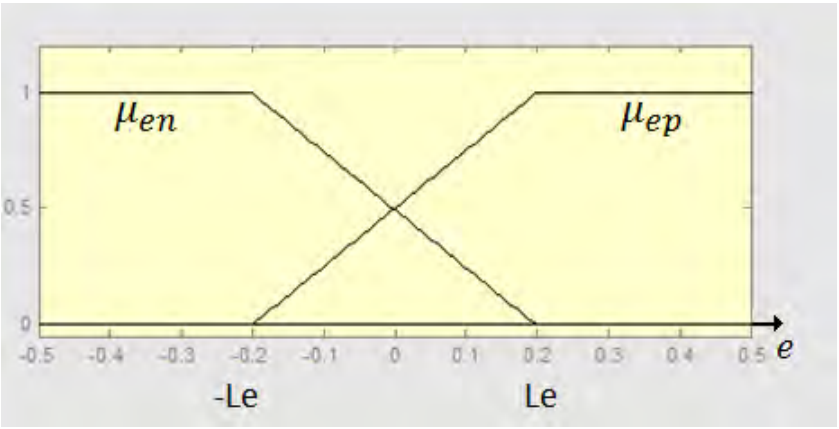


Figura 3.5: Funciones de membresía para el error.

Los rangos horizontales de cada función de membresía Le, Lde y H son determinados a partir del conocimiento experto, en base a una serie de pruebas para determinar los valores adecuados.

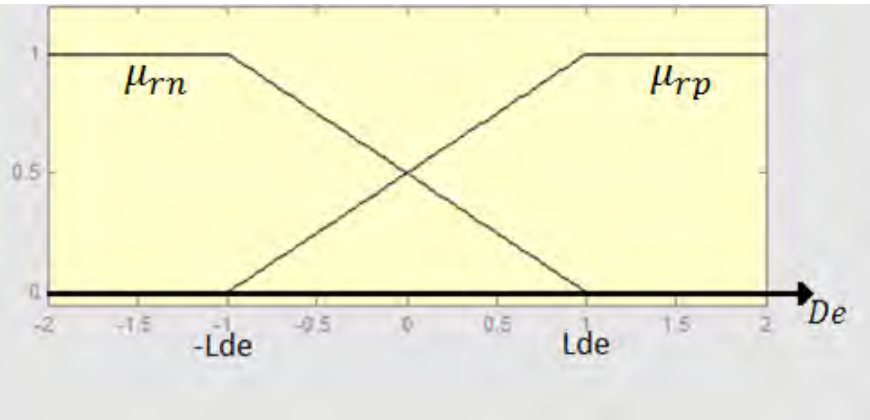


Figura 3.6: Funciones de membresía para el cambio del error.

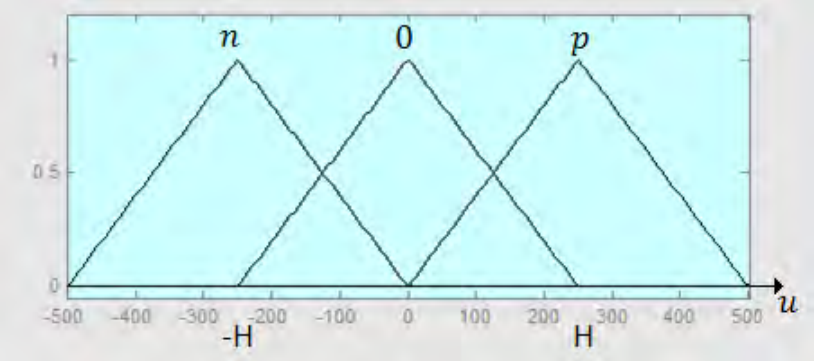


Figura 3.7: Funciones de membresía para la acción de control.

3.2.4. Base de reglas

Si se define el error como

$$e = y - r$$

donde y es la salida del sistema y r es la referencia.

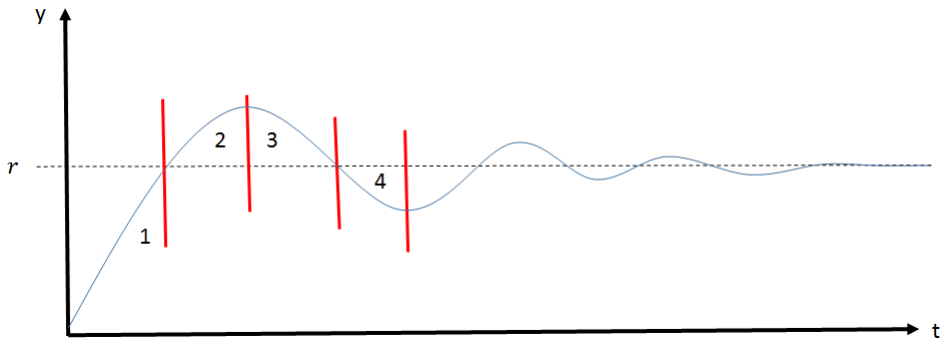


Figura 3.8: Respuesta deseada usando el PD difuso.

Si el comportamiento deseado es el mostrado en la figura 3.8, se pueden definir las cuatro regiones mostradas en la figura en relación al error y el cambio del error [Pham 2001].

1. Error negativo (e_n) y cambio de error positivo (r_p)
2. Error positivo (e_p) y cambio de error positivo (r_p)
3. Error positivo (e_p) y cambio de error negativo (r_n)
4. Error negativo (e_n) y cambio de error negativo (r_n)

A partir de estos bloques se puede definir una base de reglas mínimas para el diseño de un controlador.

Si el sistema tiene un integrador natural como es el caso del control de velocidad del motor eléctrico entonces se puede utilizar la estructura PD para definir las siguientes reglas

r_1 : Si error $= e_p$ y cambio de error $= r_p \longrightarrow u = n$

r_2 : Si error $= e_p$ y cambio de error $= r_n \longrightarrow u = 0$

r_3 : Si error $= e_n$ y cambio de error $= r_p \longrightarrow u = 0$

r_4: Si error $= e_n$ y cambio de error $= r_n \longrightarrow u = p$

Esta base de reglas es la mínima que se puede sintetizar para la implementación de un controlador PD difuso [Pham 2001]. Este controlador se puede implementar en versión continua o discreta. Para la versión discreta se utiliza $\Delta u = u(k) - u(k-1)$ y para la versión continua $\Delta u = \dot{u}$. Donde el incremento de la entrada es Δu y $u(k)$ y $u(k-1)$ son la entrada actual y la entrada en un instante anterior respectivamente. EL antecedente de las reglas activas se calculan con la función mínimo.

$$\begin{aligned} r_1 &: min(\mu_{ep}, \mu_{rp}) \\ r_2 &: min(\mu_{ep}, \mu_{rn}) \\ r_3 &: min(\mu_{en}, \mu_{rp}) \\ r_4 &: min(\mu_{en}, \mu_{rn}) \end{aligned}$$

3.2.5. Defuzzificación

La defuzzificación se lleva a cabo mediante el método de centro de gravedad. Puesto que los conjuntos de salida son todos simétricos, el centro de gravedad o centroide se puede calcular de los centroides locales como:

$$\Delta u = H \cdot \frac{S(\mu_{r4}) - S(\mu_{r1})}{S(\mu_{r1}) + S(\mu_{r2 \vee r3}) + S(\mu_{r4})}$$

donde $S(\cdot)$ es el área respectiva de cada regla disparada en la salida, y H es un parámetro que se considera de acuerdo al conocimiento experto del sistema, y por último la acción de control queda como

$$u(k) = Ku \cdot \Delta u(k)$$

3.3. El Controlador Supertwisting

Este algoritmo fue desarrollado para controlar sistemas con grado relativo uno principalmente para evitar el chattering [Perruquetti 2002]. En la figura (3.9) se puede observar como se espera que converja la trayectoria en el plano de fase en tiempo finito con el algoritmo supertwisting.

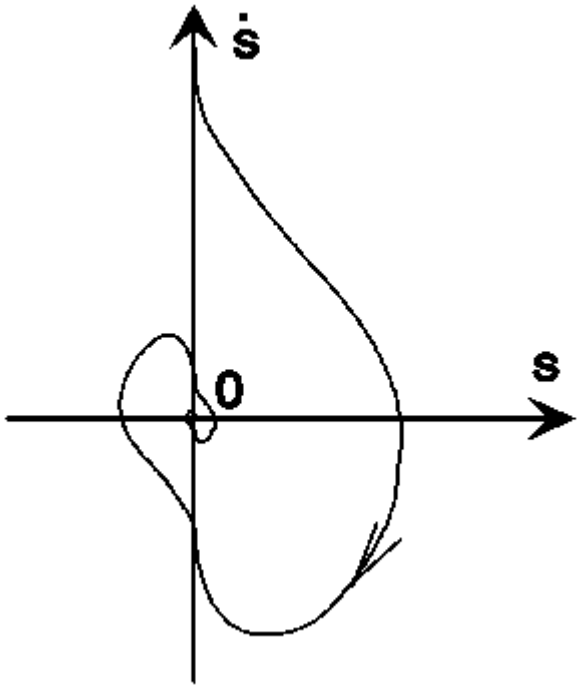


Figura 3.9: Trayectoria esperada de fase del algoritmo supertwisting [Perruquetti 2002]

El algoritmo de control está definido por la siguiente ley de control (3.9)

$$u(t) = u_1(t) + u_2(t) \tag{3.9}$$

$$\dot{u}_1(t) = \begin{cases} -u & si\ |u| > 1 \\ -K_1 sign(y) & si\ |u| \leq 1 \end{cases}$$

$$u_2(t) = \begin{cases} -K_2 |s_0|\, sign(y) & si\ |y| > s_0 \\ -K_2 |y|^{\rho}\, sign(y) & si\ |y| \leq s_0 \end{cases}$$

donde y es la salida del sistema, u_1 y u_2 son las componentes de la entrada de control del algoritmo, u(t) es la entrada de control, K_1 y K_2 son constantes, el controlador puede ser simplificado cuando los sistemas a controlar son linealmente independientes en control, u no necesita ser acotada y $s_0 = \infty$: [Perruquetti 2002]

$$\begin{aligned} u &= -K_2 |s|^{\rho}\, sign(y) + u_1, \\ \dot{u}_1 &= -K_1 sign(y). \end{aligned}$$

donde

$$s = y - y_{deseada}$$

para la implementación del algoritmo supertwisting en el vehículo, las ecuaciones a utilizar son

$$\begin{aligned} \dot{u_1} &= -K_1 sign(\theta) \\ u &= -K_2 |\theta|^{1/2} sign(\theta) + u_1 \end{aligned} \tag{3.10}$$

3.3.1. Discretización

Para la implementación del algoritmo supertwisting en un microcontrolador es necesaria una discretización del controlador, tras esta discretización se obtiene

$$\begin{aligned} u_{1(k)} &= T(-K_1 sign(\theta)) + u_{1(k-1)} \\ u_{(k)} &= -K_2 \sqrt{|\theta|} sign(\theta) + u_{1(k)} \end{aligned} \tag{3.11}$$

donde T es el periodo de muestreo.

CAPÍTULO 4

Pruebas y Resultados

El vehículo construido fue sometido a una serie de pruebas para probar su funcionamiento desde la exposición a movimientos violentos, largos tiempos de operación hasta algunos golpes que pusieron a prueba la robustez de la estructura mecánica.

En este capítulo se abordan en detalle las pruebas realizadas a cada una de las partes que conforman el vehículo autobalanceado de dos ruedas así como la verificación del funcionamiento en conjunto de todos los componentes para el correcto funcionamiento del vehículo.

4.1. Pruebas del equipo.

4.1.1. Pruebas en los actuadores

A los actuadores compuestos por el driver y el motor se les aplicaron las siguientes pruebas

4.1.1.1. Prueba en vacío

Esta prueba consistió en conectar los motores acoplados a la caja de engranes sin carga mecánica adicional para saber cuál es la potencia en vacío de éstos, el voltaje aplicado fue de 24 volts obteniendo para el motor derecho una corriente de 8.36 A y para el motor izquierdo de 8.43 A que son valores muy aproximados al valor marcado por fabricante que es de 8.5 A con la caja de engranes acoplada.

4.1.1.2. Prueba de potencia

Esta prueba consistió en subir a un piloto y mover el vehículo para saber si los motores eran capaces de mover todo en conjunto, se realizaron diversas pruebas con pilotos desde los

55Kg hasta los 89 Kg y los resultados fueron satisfactorios ya que los actuadores movieron el vehículo sin ningún problema. En esta prueba no se obtuvieron mediciones de corriente, ya que no se tienen instalados los sensores correspondientes.

4.1.1.3. Zona muerta de los actuadores

El efecto de la zona muerta se pudo compensar mediante la inclusión en la entrada de control de un término el cual permite que la acción de control afecte instantáneamente el comportamiento del vehículo. Este término se determino mediante el experimento detallado en el capítulo Experimentalmente se determinó que los motores tienen una zona muerta de un valor aproximado de 80 (en unidades del dato a enviar al PWM) en cada sentido de giro. Así, para la compensación de zona muerta se observa que dependiendo el signo de $u(t)$ es el sentido de giro de los motores, entonces para $u(t) > 0$ se realizara la corrección siguiente

$$u(t) = u(t) + 80 \tag{4.1}$$

y para $u(t) < 0$, la corrección correspondiente es

$$u(t) = u(t) - 80 \tag{4.2}$$

de esta manera aseguramos una acción inmediata del controlador sobre los actuadores.

Para los motores FRACMO utilizados en la construcción del vehículo, el nivel de la zona muerta fue medida por medio de la variación del ciclo de trabajo de la señal PWM generada por el microcontrolador, en la tabla (4.1) se muestran los niveles de voltaje necesarios para romper el estado de reposo para cada motor en los dos sentidos de giro.

Motor	Sentido de giro	Voltaje de zona muerta
Derecho	Hacia adelante	0.804 volts
Derecho	Hacia atrás	0.78 volts
Izquierdo	hacia adelante	0.79 volts
Izquierdo	Hacia atrás	0.78 volts

Tabla 4.1: Zona muerta de los motores en ambos sentidos de giro

Dados los datos de la tabla (4.1) se compensó la zona muerta del motor utilizando como salida de control mínima una señal PWM con un ciclo de trabajo de 3.3 % equivalente a un

voltaje aplicado de 0.8 volts, estos resultados son obtenidos utilizando una señal PWM de 5 KHz con una alimentación de 24 volts. Esta zona muerta es compensada por software como se se observa en (4.1) y (4.2).

4.1.2. Prueba de sensores

Antes de probar cualquier esquema de control es necesario verificar las lecturas tanto del acelerómetro como del giroscopio ya que las señales obtenidas de este par de sensores son la parte mas importante del funcionamiento del vehículo ya que permiten obtener la medida del ángulo de inclinación θ.

4.1.2.1. Acelerómetro y giroscopio

Para poder trabajar con el acelerómetro LSM303DLHC y el giroscopio L3GD20 en la red I^2C es necesario primeramente configurar los dispositivos de acuerdo a las características del muestreo deseadas para el funcionamiento de este par de sensores. La configuración de funcionamiento de estos sensores se muestra en la figura 4.2.

Tabla 4.2: Configuración del acelerómetro

Configuración	Acelerómetro	Giroscopio
Tipo de comunicación	I^2C	I^2C
Modo	Normal	Normal
Taza de muestreo	200 muestras/seg	200 muestras/seg
Actualización	Continua	Continua
Justificado del resultado	Izquierda	Izquierda
Rango	$\pm$2g	250 grados por segundo
Resolución	1mg	8.75 grados por segundo

Una vez configurados los parámetros se establecen las pruebas de medición del acelerómetro manteniendo en reposo el vehículo (θ constante) las mediciones mostraron una lectura del ángulo de inclinación θ con una variación de hasta $\pm$ 1^{o}.

4.1.2.2. Filtrado

Para mejorar la medición del ángulo de inclinación θ, se incorpora la medición de la velocidad angular por medio de un giroscopio y mediante la aplicación de un filtro de Kalman se obtiene esta medida mejorada del ángulo θ, ver [Bonales 2012].

Los parámetros del filtro de Kalman fueron sintonizados a prueba y error donde los mejores resultados obtenidos en la prueba de conducción se dieron para los valores mostrados en (4.3)

$$\begin{aligned} Q &= \begin{bmatrix} 0.3 & 0 \\ 0 & 0.3 \end{bmatrix} \\ R &= 40 \\ P_{inicial} &= \begin{bmatrix} 100000 & 0 \\ 0 & 100000 \end{bmatrix} \\ T &= 0.017 \end{aligned} \tag{4.3}$$

donde T es el periodo de muestreo seleccionado para el sistema. Los resultados de este ajuste en los parámetros del filtro de Kalman tienen como resultado la eliminación de sensibilidad ante movimientos rápidos producidos por la vibración generada por la irregularidad de los terrenos donde se desplaza el vehículo. En la figura (4.1) podemos observar la comparación entre la señal obtenida por el acelerómetro y la medida de la salida del filtro de Kalman la cual sufre un pequeño retraso de aproximadamente una décima de segundo pero que permite despreciar las oscilaciones rápidas para evitar un accidente, además de que disminuye el ruido en la medición del ángulo de inclinación θ hasta llegar a un $\pm 0.3^{o}$ de error. El experimento mostrado en la figura (4.1) consistió en mantener en reposo el vehículo unos segundos y después aplicarle oscilaciones lentas y rápidas para poner a prueba el seguimiento de la señal filtrada a la inclinación real.

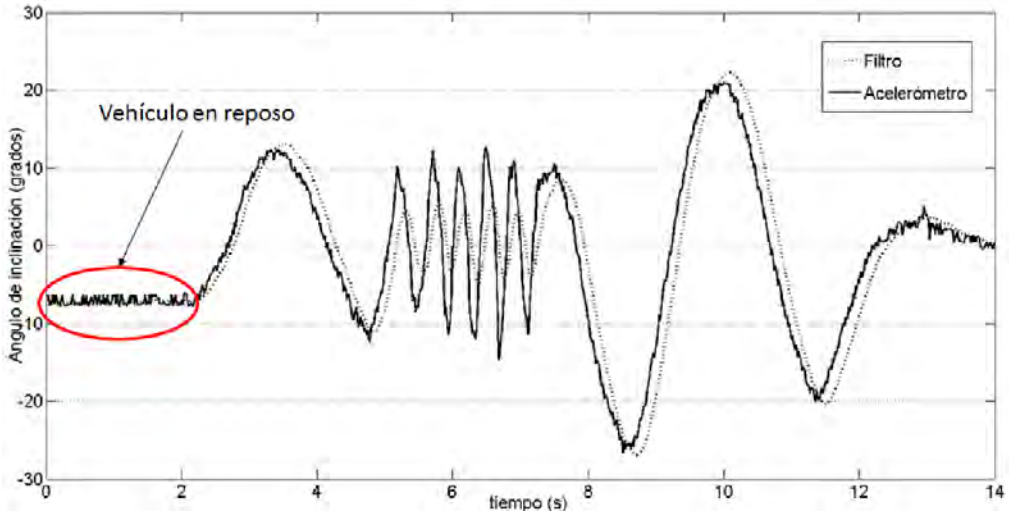


Figura 4.1: Comparación entre la medida del ángulo.

4.1.3. Selección e implementación del tiempo de muestreo

El periodo de muestreo utilizado en los controladores se seleccionó en base a el tiempo que tarda el microcontrolador en realizar las operaciones para llevar a cabo cada uno de los algoritmos de control, donde estos tiempos se midieron utilizando un Timer del mismo microcontrolador siguiendo los siguientes pasos

- Inicio del Timer.
- Ejecución de muestreo, operaciones de los algoritmos de control, actualización de la entrada de control y envío de datos a la PC.
- Lectura del Timer.
- Cálculo de tiempo.

El resultado de estas pruebas arrojaron para el controlador PID un tiempo aproximado de 15 ms, para el controlador PD difuso 16 ms y para el controlador supwertwisting 15ms, por lo cual se optó por la selección de un periodo de muestreo de 17 ms para la implementación de los tres controladores, el cual fue asegurado de igual manera por el uso del Timer del microcontrolador principal como se muestra en la figura 4.2.

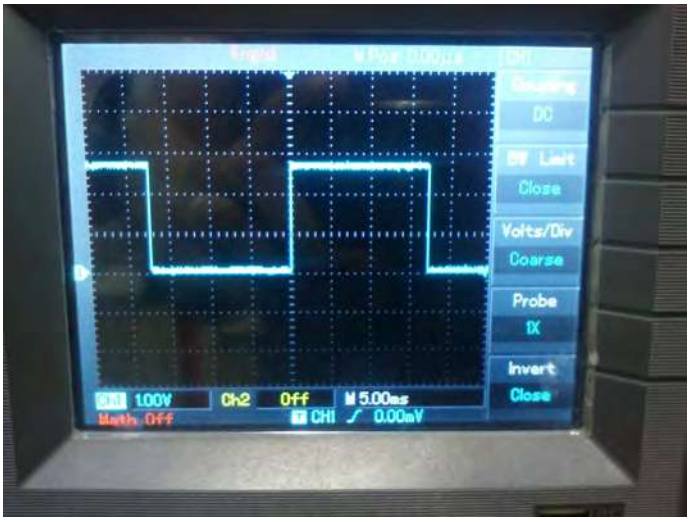


Figura 4.2: Implementación del periodo de muestreo.

4.1.4. Dinámica cero del sistema

Para saber que pasa con la dinámica cero del sistema se puede visualizar en simulación como se comportan los estados que no se están controlando, en este caso para ver si es de

fase mínima o no mínima se muestra en la figura (4.3) el comportamiento de x_3.

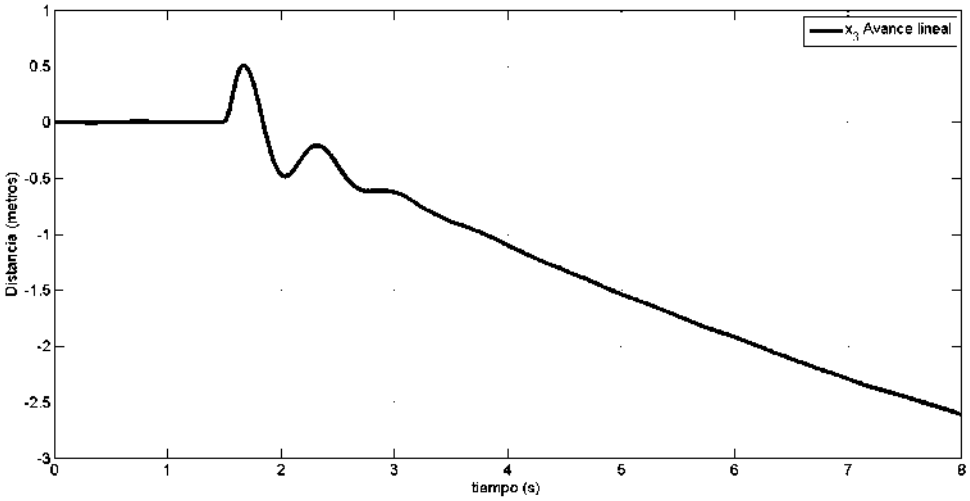


Figura 4.3: Comportamiento de x_3 cuando se aplica el controlador PD difuso mínimo

Como se puede observar la variable x_3 sigue incrementando mientras pasa el tiempo, con lo cual se deduce que el sistema es de fase no mínima. Para los otros controladores pasa exactamente lo mismo con la variable x_3.

4.2. Pruebas de control en Simulación

En la siguiente sección se establecen los parámetros utilizados para la realización de las pruebas en simulación los cuales fueron tomados directamente de las medidas del vehículo construido, las pruebas y resultados de cada una de las simulaciones que se realizaron para probar el desempeño de cada uno de los controladores implementados.

4.2.1. Consideraciones para la simulación

En el capítulo 2 se analizó el modelo matemático obtenido en [Bonales 2012] con la adición del modelo de los actuadores y los términos de fricción viscosa. El modelo matemático considerado para la implementación de los controladores en simulación es (2.17). En donde se tomarán únicamente en cuenta los estados x_1, x_2, x_3 y x_4 es decir, no se consideran cambios de dirección en el avance del vehículo para poder hacer una comparación entre simulación y la respuesta del sistema en tiempo real.

4.2.1.1. Consideraciones realistas

A continuación se enlistan las consideraciones realistas a tomar en cuenta en la simulación para que los resultados obtenidos en ésta sean mas fiables para poder compararlos con los resultados obtenidos en las pruebas realizadas al sistema en tiempo real.

- Características físicas del prototipo: El vehículo de dos ruedas autobalanceado tiene sus características físicas fijas por lo cual únicamente debemos tomar medida de sus dimensiones, masas y calcular los momentos de inercia establecidos en [Bonales 2012].

- Características del piloto: El vehículo será conducido por una persona, pero no se puede saber con exactitud cuáles son las características físicas de cada piloto, lo cual puede afectar el comportamiento de los controladores implementados. Para solucionar esto se realizaron pruebas considerando diferentes masas del piloto.

- Ruido de medición: El sistema real es afectado por un ruido de medición después del filtro de Kalman de $\pm 0.3°$lo cual se introduce en la simulación para darle parecido al sistema real.

- Saturación y zona muerta de los actuadores : Estos dos efectos aparecen en los motores en el sistema real y provocan un comportamiento diferente al previsto por las ecuaciones en la dinámica del sistema por lo cual estos son introducidos en la simulación. Se considera una saturación en la acción de control de ± 24 volts correspondiente a 100 % que son 6000 cuentas del PWM y la zona muerta descrita en la sección 4.1.1.3 de 0.8 volts

- Efecto de la discretización: En simulación se implementará un sistema continuo controlado por un controlador discreto, con el periodo de muestreo de $T = 0.017$s que es el mismo implementado en el vehículo real.

Los parámetros físicos utilizados en la simulación son mostrados en la tabla 4.3

Tabla 4.3: Parámetros de la simulación

Parámetro	Valor
Fuerza de gravedad	9.81 m/s^2
Masa de la rueda	3 Kg
Radio de la rueda	0.2 m
Distancia del centro de masa al origen	0.85 m
Distancia del centro del vehículo al centro de la llanta	0.4 m
Ancho de la base	0.58 m
Altura de la base	0.17 m
Altura del tripulante	1.8 m
Ancho del tripulante	0.25 m
Distancia del centro de la base al eje central	0.09 m
Masa de la base	39.5 Kg
Coeficiente de fricción b_0 respecto a θ	30
Coeficiente de fricción $b_1$respecto a α	3
Resistencia del devanado de armadura del motor R_a	1.5Ω
Coeficiente de la velocidad angular K_e	0.001
Constante de proporcionalidad del par del motor K_τ	0.2

con los cuales se calculan los momentos de inercia I_θ e I_ϕ como se indica en [Bonales 2012]. Obteniéndose

$$\begin{aligned} I_\theta &= 29.32\, Kg/m^2 \\ I_\varphi &= 0.06\, Kg/m^2 \end{aligned}$$

4.2.2. Pruebas de esquemas de control en simulación

Para poder tener una idea de lo que pasará al probar las técnicas de control en el sistema en tiempo real existen herramientas de simulación las cuales permitirán hacer pruebas cercanas a la realidad sin tener los gastos ocasionados por realizar las pruebas físicas. En el caso del vehículo autobalanceado de dos ruedas se realizan simulaciones para tratar de conocer el comportamiento que tendrá el vehículo al tratar de equilibrarse y así poder verificar el posible funcionamiento de éste en tiempo real. Estas simulaciones fueron realizadas en la aplicación Simulink del software MATLAB.

4.2.3. Controlador PID

Para la implementación del PID discreto se utilizaron los bloques de función de transferencia discreta para poder implementar la ecuación (3.4) como se muestra en los diagramas programados en simulink (ver figura B.1 del apéndiceB).

La sintonización de los parámetros del PID discreto se hizo a prueba y error, tras una serie de experimentos se lograron resultados adecuados en el equilibrio del vehículo con los siguientes valores de las ganancias

K_p	3000
T_i	10
T_d	0.1
T	0.017 s
N	1

El experimento parte de la posición de inicio y después de unos instantes se incluye una perturbación de tipo pulso de corta duración aplicada a la medición de la salida, el controlador calcula la entrada de control de tal manera que el vehículo regresa a la posición de equilibrio, los resultados obtenidos con esta sintonización se muestran en la figura 4.4, donde se muestra tanto el ángulo de inclinación como la entrada de control calculada por el microcontrolador principal.

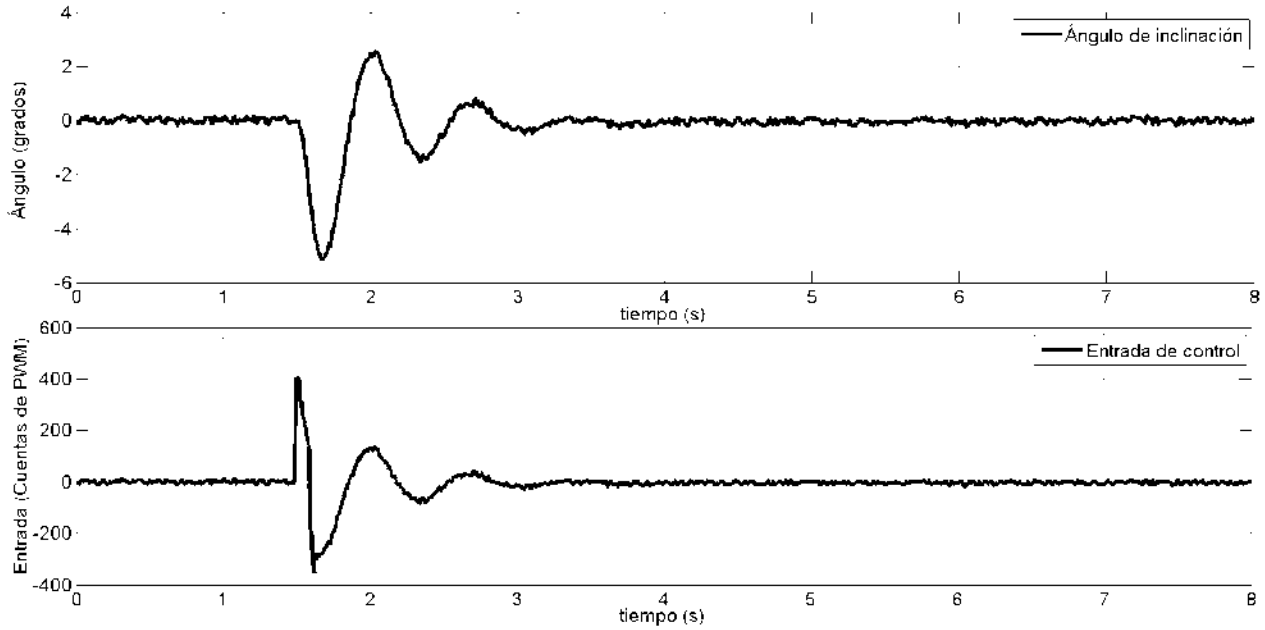


Figura 4.4: Respuesta del sistema sin tripulante usando el PID en simulación.

A pesar de que en el sistema en tiempo real no se puede hacer una prueba comparativa de los controladores debido a que cada piloto representa una dinámica diferente y muy difícil de modelar, se optó por realizar algunas pruebas con pilotos de diferentes masas en simulación obteniendo los resultados mostrados en las figuras 4.5, 4.6 y 4.7.

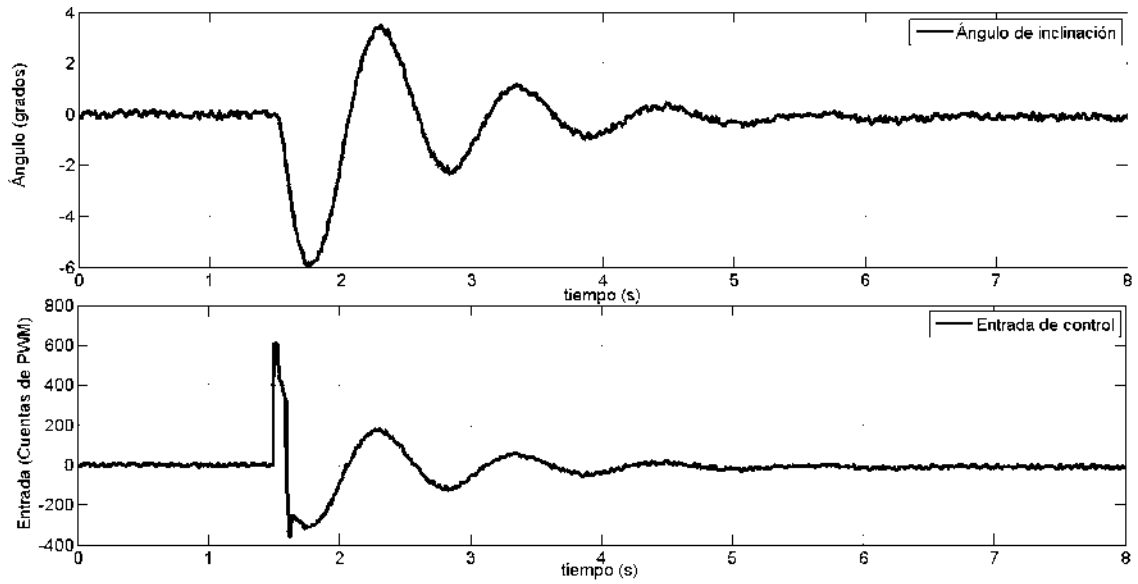


Figura 4.5: Respuesta del sistema con tripulante de 60 Kg usando el PID en simulación.

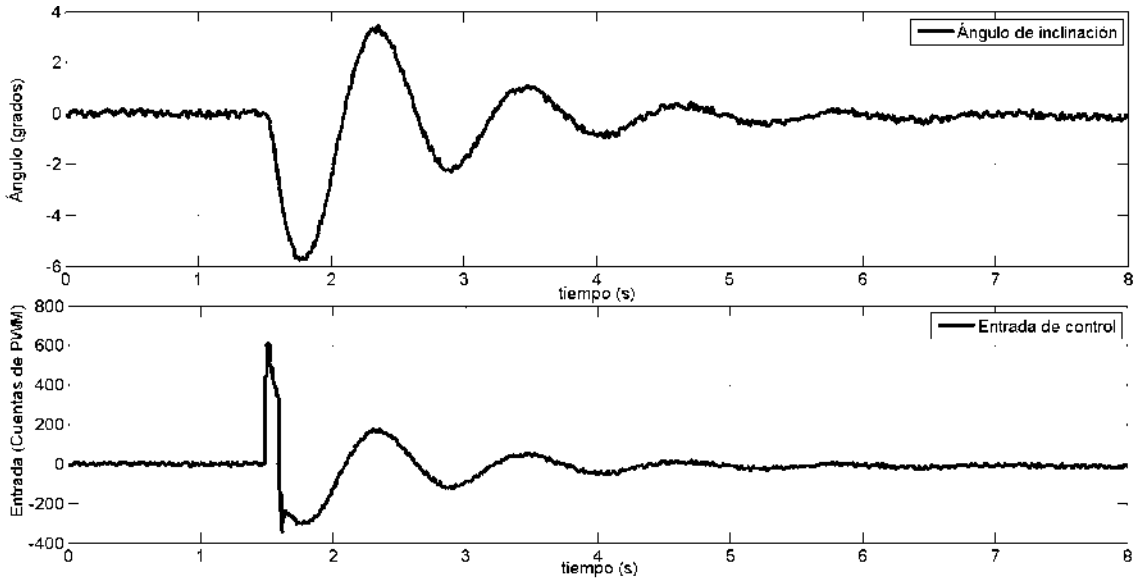


Figura 4.6: Respuesta del sistema con tripulante de 70 Kg usando el PID en simulación.

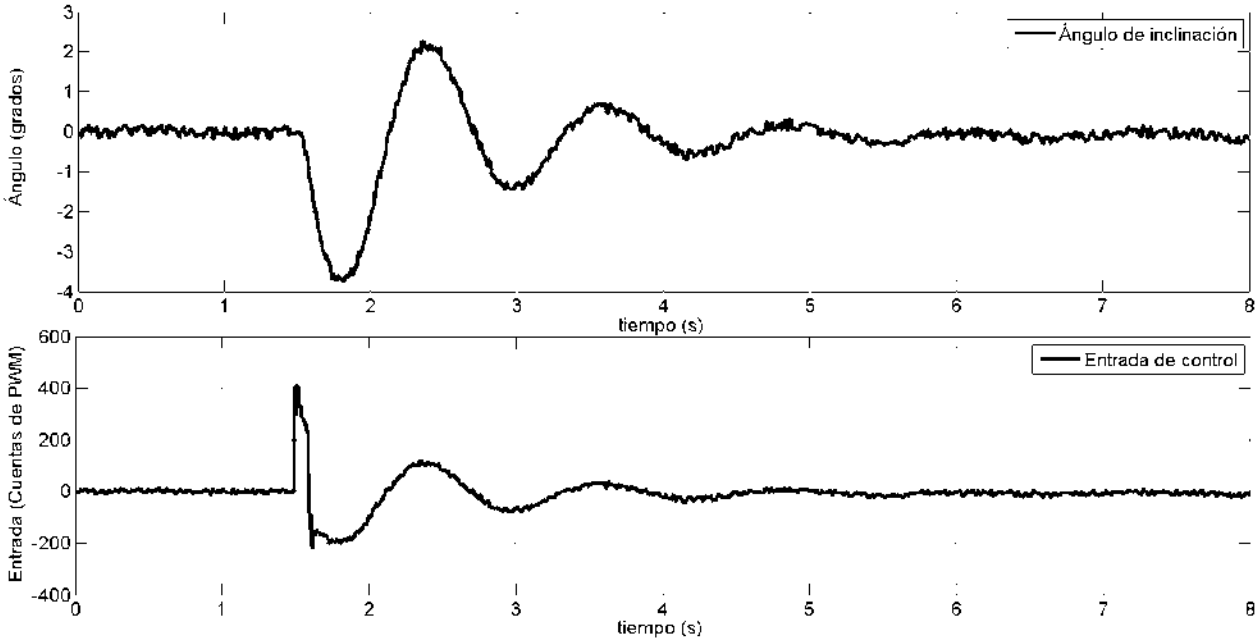


Figura 4.7: Respuesta del sistema con tripulante de 80 Kg usando el PID en simulación.

.

4.2.4. Controlador PD difuso mínimo

Para implementar el controlador PD difuso mínimo en la simulación se recurrió al toolbox FUZZY Logic de la aplicación simulink de MATLAB en específico a los bloques de función de membresía para poder realizar la fuzzificación la salida, y a los bloques MinMax para el cálculo de la defuzzificación (ver figura B.1 del apéndice B). Los parámetros ajustables de este controlador se sintonizaron a prueba y error hasta obtener los resultados deseados al equilibrar el vehículo de dos ruedas autobalanceado, los valores obtenidos para esta tarea fueron

Le	0.2
Lde	1
H	250
K_d	1
K_p	1
K_u	28

El experimento fue el mismo que el del PID discreto, obteniendo los resultados mostrados en la figura 4.8.

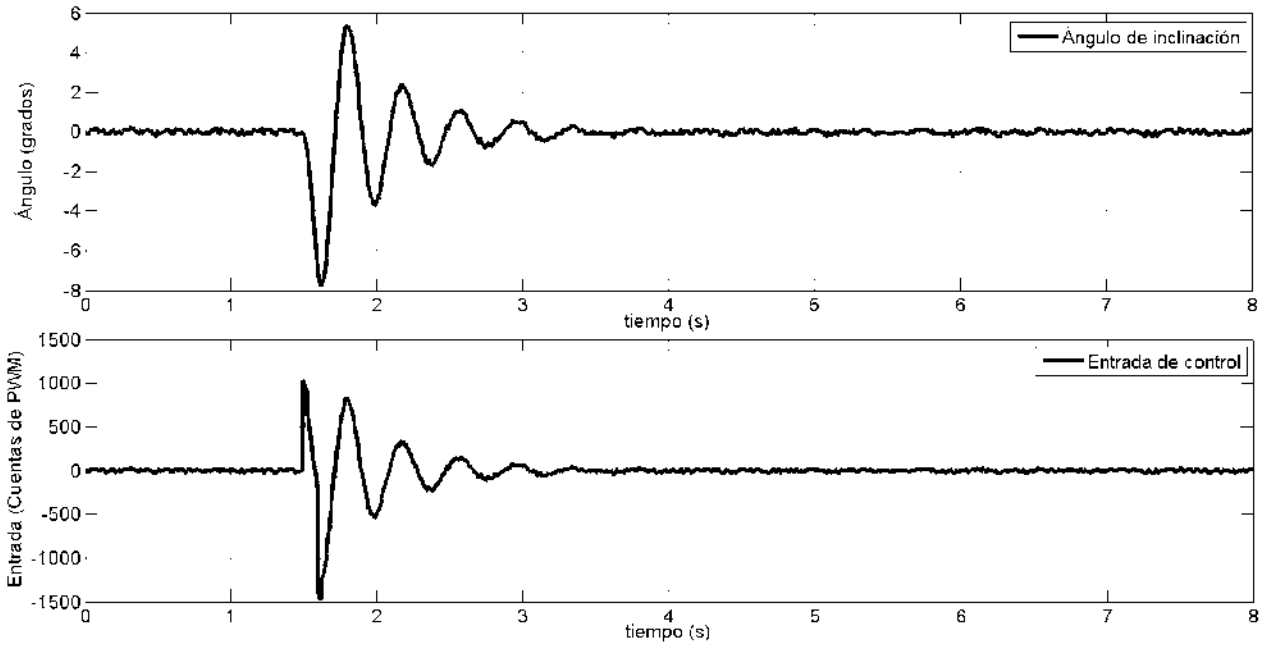


Figura 4.8: Respuesta del sistema sin tripulante usando PD difuso mínimo.

El controlador de igual manera que con el PID se prueba en simulación con la incorporación de pilotos de diferentes masas (60, 70 y 80 Kg) obteniendo los comportamientos mostrados en las figuras 4.9, 4.10 y 4.11.

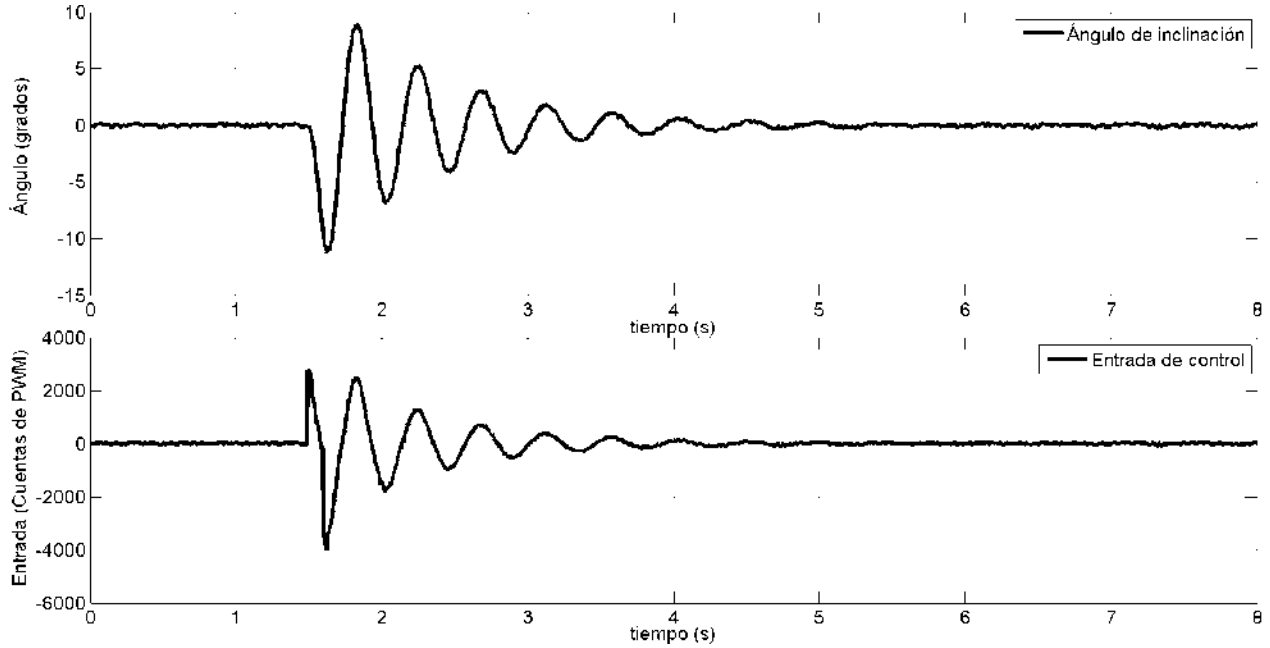


Figura 4.9: Respuesta del sistema con tripulante de 60 Kg usando el PD difuso en simulación.

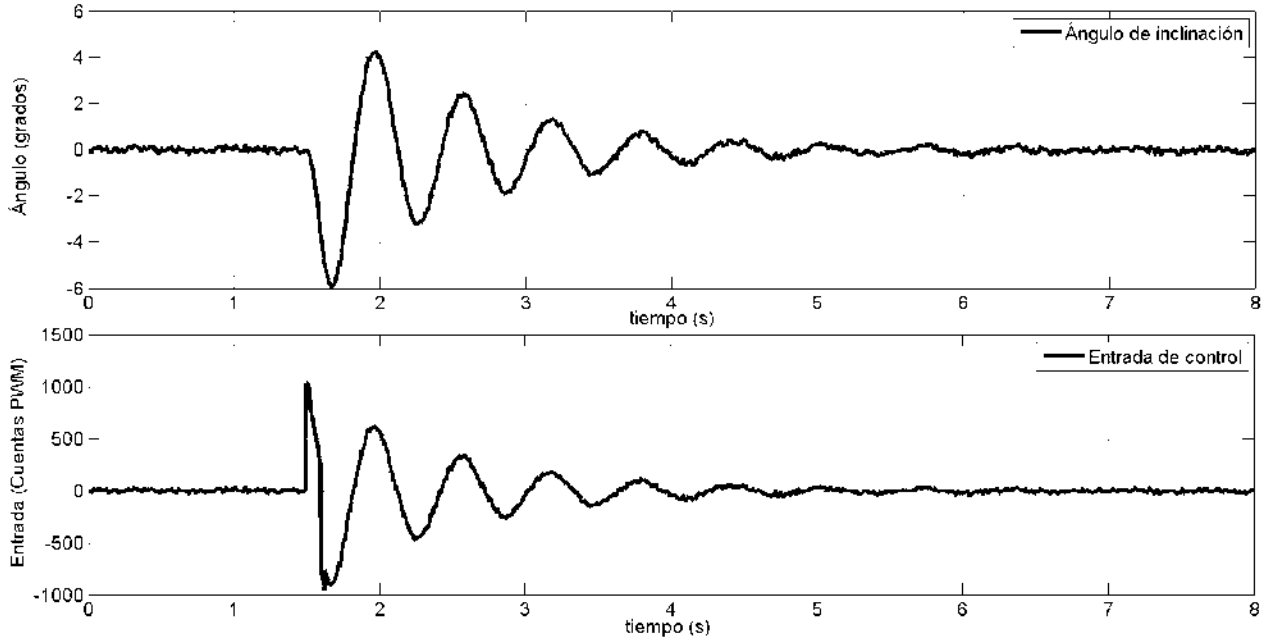


Figura 4.10: Respuesta del sistema con tripulante de 70 Kg usando el PD difuso en simulación.

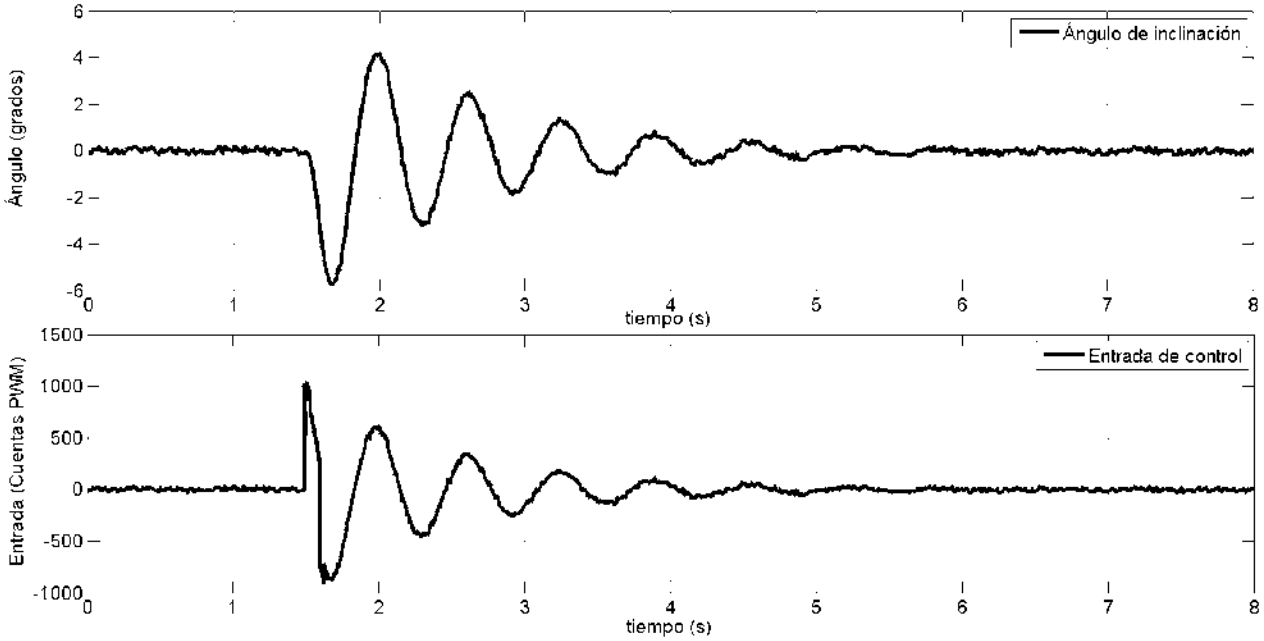


Figura 4.11: Respuesta del sistema con tripulante de 80 Kg usando el PD difuso en simulación.

4.2.5. Controlador Supertwisting

EL controlador supertwisting fue implementado a partir de las ecuaciones (3.11) programadas en MATLAB haciendo uso de Simulink y los bloques de valor absoluto y raíz cuadrada (ver figura B.1 del apéndice B). El experimento desarrollado fue el mismo que en los dos controladores anteriores teniendo que sintonizar los parámetros K_1 y K_2 de manera empírica y así obteniendo los resultados mostrados en la figura 4.12, de igual manera se hicieron experimentos con diferentes masas del piloto aunque no se apreció mucho cambio al cambiar la masa del tripulante.

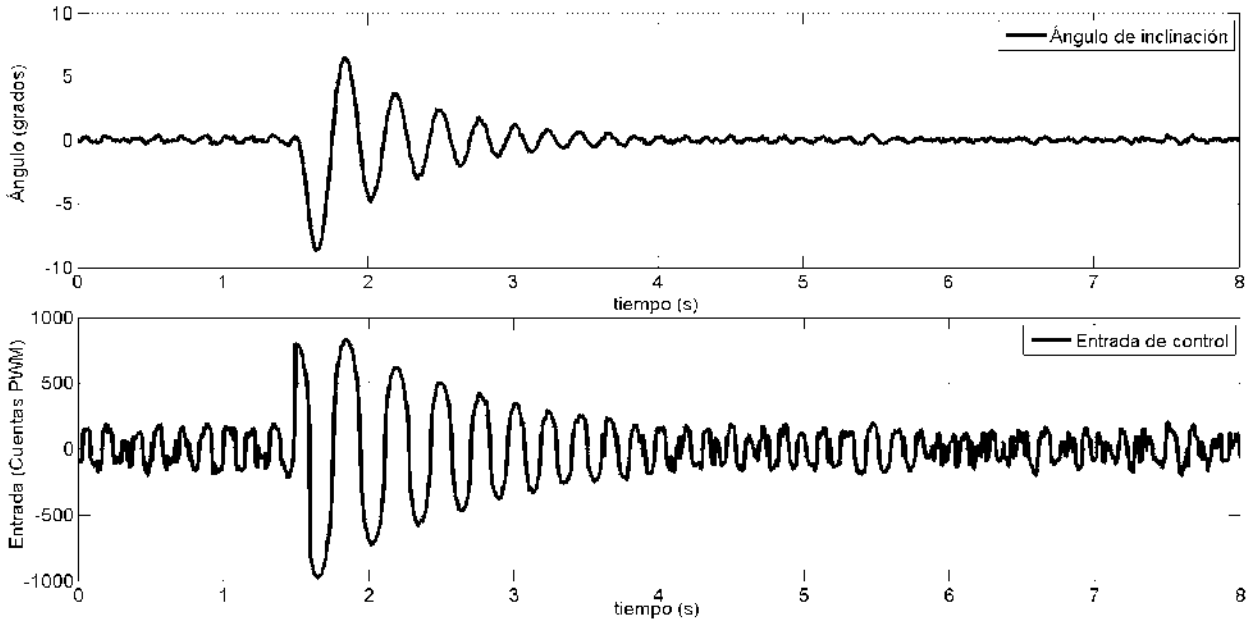


Figura 4.12: Respuesta del sistema sin tripulante usando el algoritmo Supertwisting en simulación.

4.3. Pruebas en el Sistema Real

4.3.1. Controlador PID

La implementación del controlador PID en el sistema en tiempo real se basa en la programación del algoritmo dado por las ecuaciones (3.2) a (3.8) dentro del microcontrolador principal a partir de las señales obtenidas por el microcontrolador auxiliar. La sintonización de los parámetros del PID al ser un sistema inestable en el punto de operación se realizó a prueba y error tomando como punto de partida los valores $K_p = 10$, $T_i = 0$ y T_d=0, has-

ta obtener un buen funcionamiento en el vehículo. Los valores de los parámetros una vez sintonizados quedaron como sigue

K_p	3000
T_i	10
T_d	10
N	1
T	0.017s

Con estos valores, el vehículo logra equilibrar su estructura sin el pasajero. El comportamiento del sistema en tiempo real con los parámetros obtenidos a través de las pruebas es el mostrado en la figura 4.13. En la figura 4.13 se muestra el resultado de un intervalo de tiempo en el cual se parte del reposo del vehículo y se genera una perturbación para que la acción de control actué sobre el vehículo y este se equilibre.

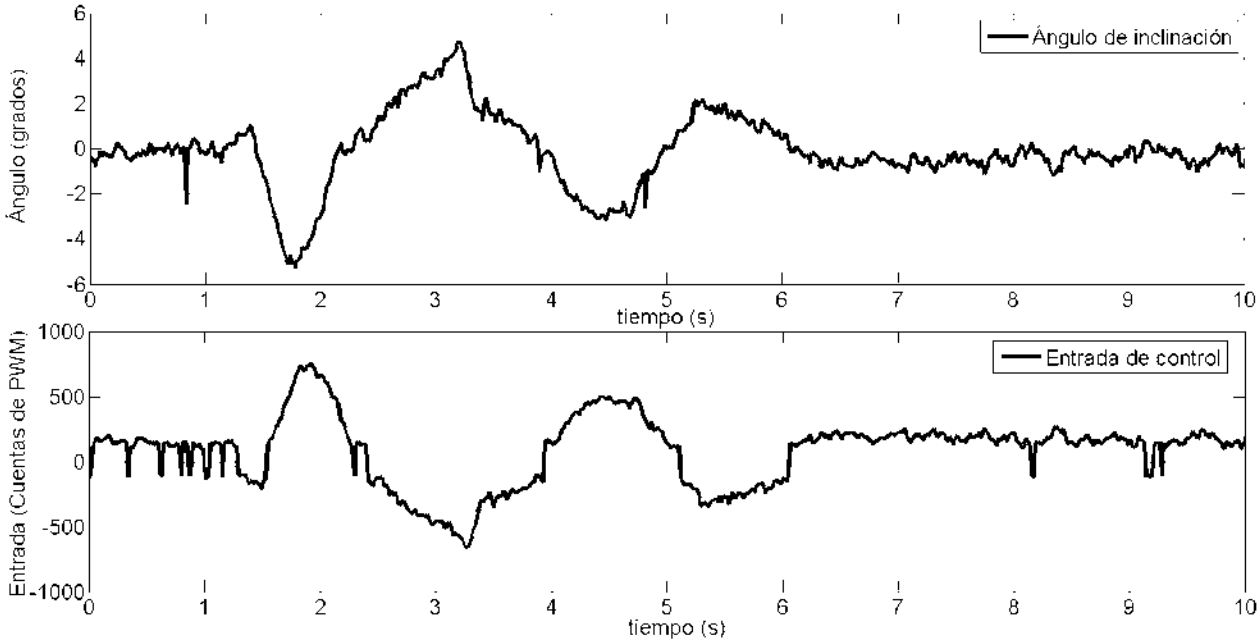


Figura 4.13: Respuesta del sistema en tiempo real controlador PID.

Podemos ver que existe una variación en los parámetros K_p, T_i y T_d entre los obtenidos en la sintonización de la simulación y los de la aplicación en tiempo real debido a los coeficientes reales de fricción son desconocidos y para poder realizar la simulación su valor se supuso en $b_0 = 30$ y $b_1 = 3$. Para la operación del vehículo con un pasajero se tuvo que sintonizar de nuevo el controlador PID obteniendo un buen funcionamiento con las ganancias mostradas en la siguiente tabla.

K_p	6000
T_i	0.5
T_d	2000
N	1
T	0.017s

4.3.2. Controlador PD difuso mínimo

Para la programación del controlador PD difuso mínimo dentro del microcontrolador principal se hizo necesario programar algunas funciones de mínimos y máximos para poder implementar las funciones de membresía como se muestra en la figura 4.14

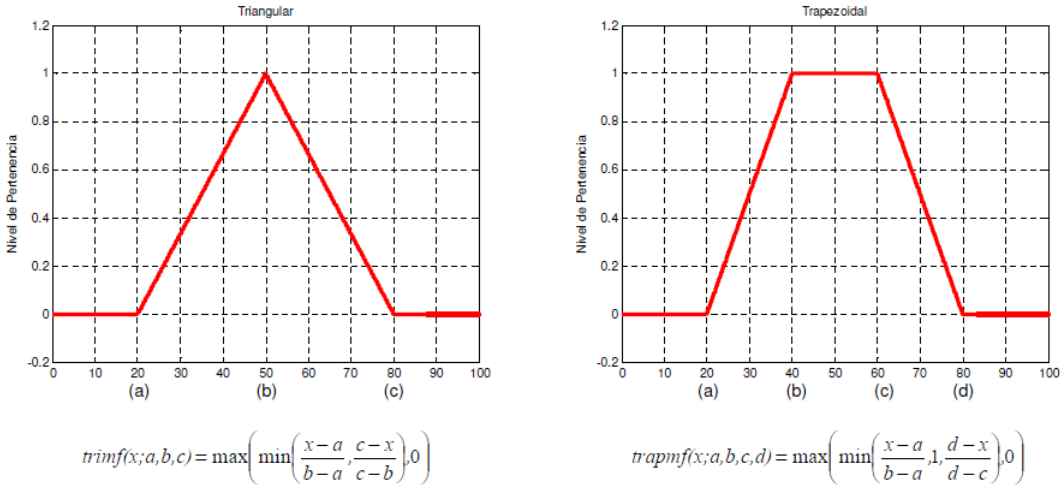


Figura 4.14: Equivalente de las funciones de membresía triangular y trapezoidal.

Una vez programada la ley de control, la sintonización de los parámetros ajustables se realizó a prueba y error hasta encontrar un funcionamiento deseado en el vehículo. Los parámetros obtenidos para un buen equilibrio del sistema en tiempo real son los siguientes

Le	0.2
Lde	1
H	250
K_p	1
K_d	1
K_u	28

con los cuales se obtuvieron los resultados mostrados en la figura 4.15.

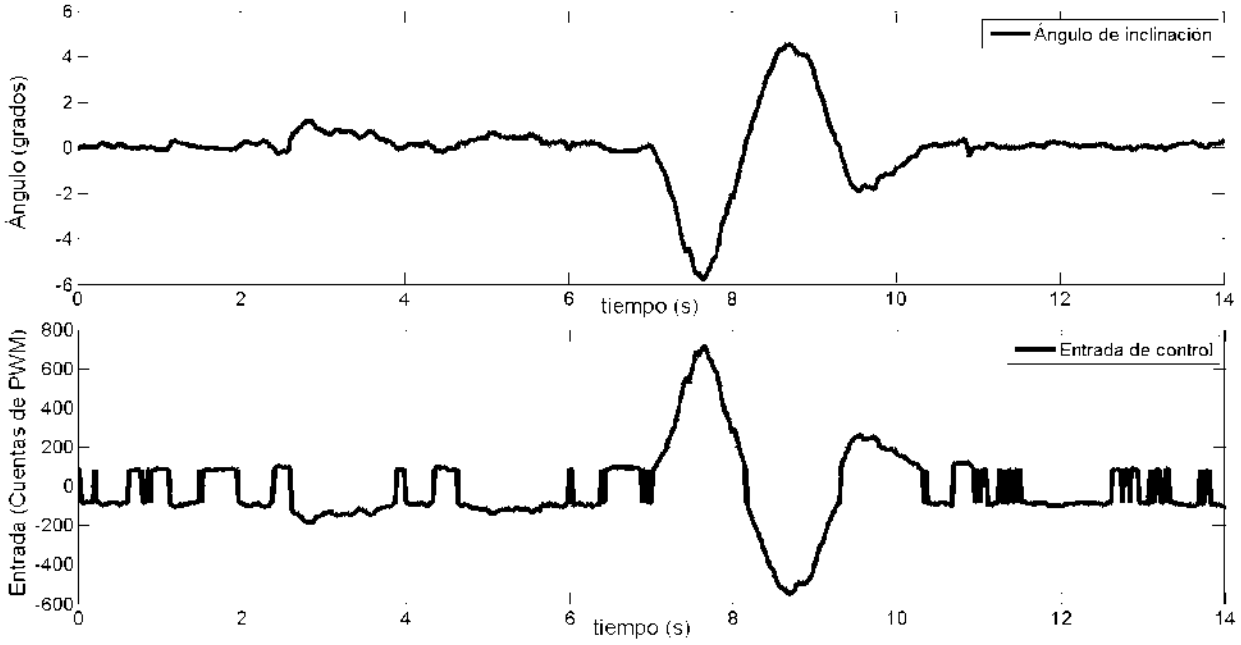


Figura 4.15: Resultado en tiempo real del controlador PD difuso.

4.3.3. Controlador Supertwisting

La sintonización es realizada a prueba y error obteniendo los mejores resultados con los siguientes valores

K_1	2500
K_2	100
T	0.017s

teniendo como resultados los mostrados en la figura 4.16, donde se puede ver una gran oscilación del sistema la cual se atribuye a que el sistema es de grado relativo dos, en vez de grado relativo uno como lo requiere esta técnica.

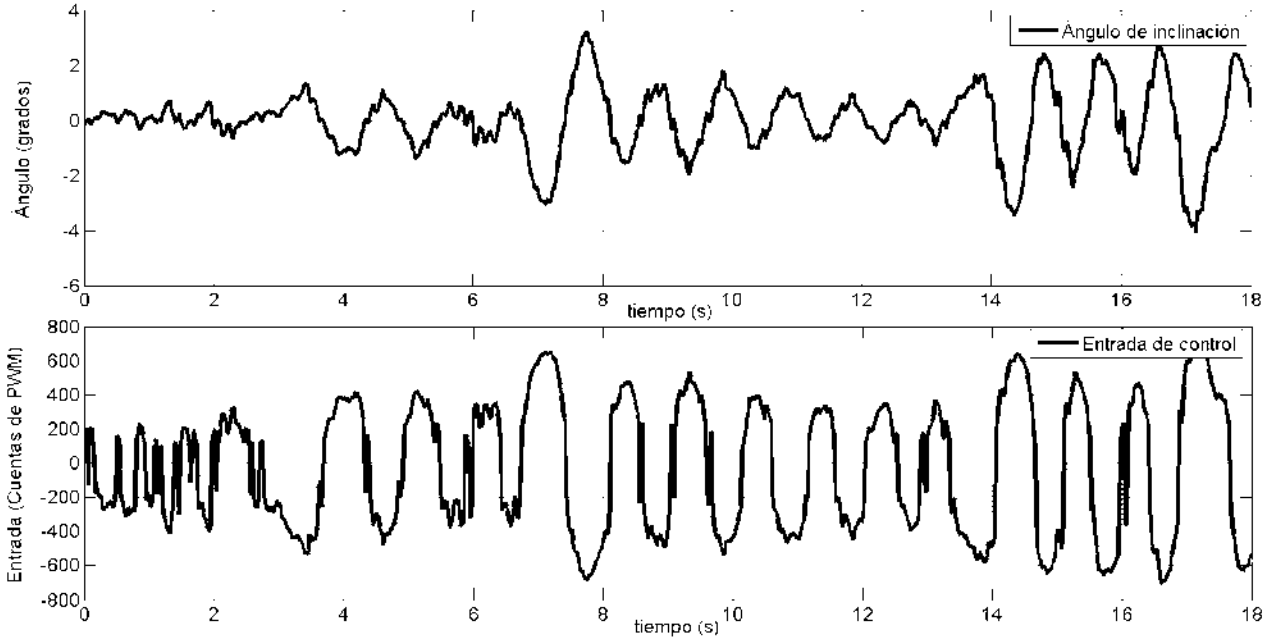


Figura 4.16: Resultados en tiempo real del controlador Supertwistng

Para hacer las pruebas con tripulante en el sistema real se requirió una nueva sintonización de las ganancias para una conducción más suave y segura.

Tabla 4.4: Ganancias para el controlador PID en el sistema real para conducción con piloto

K_p	3000
T_i	0.5
T_d	2000
N	1

Tabla 4.5: Ganancias para el controlador PD difuso en el sistema real para conducción con piloto

Le	0.2
Lde	1
H	250

Tabla 4.6: Ganancias para el controlador supertwisting en el sistema real para conducción con piloto

K_1	2500
K_2	100

Se observa una pequeña diferencia en cuanto a las ganancias del controlador PID debido a la inexactitud del modelado de los términos de fricción viscosa.

4.3.4. Pruebas de manejo en un plano inclinado

Las pruebas realizadas hasta ahora han sido sobre un plano horizontal, dado que las superficies en las instalaciones de la universidad y de las calles de la ciudad no son totalmente horizontales, es necesario investigar si el vehículo puede ser conducido a diferentes inclinaciones.

Las pruebas se realizaron mediante una base inclinada de madera midiendo la inclinación con un transportador y un nivel de gota para verificar el ángulo respecto a un plano totalmente horizontal. Los resultados de estas pruebas mostraron que el mayor ángulo posible de inclinación de la superficie en la que el vehículo responde de una manera adecuada es de 33^{o} como se muestra en la figura 4.17

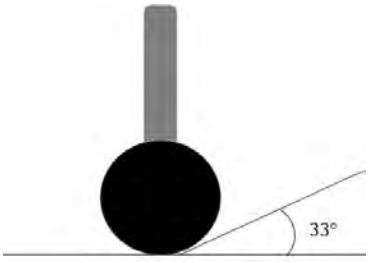


Figura 4.17: Mayor ángulo de inclinación para la conducción del vehículo.

Después de sobrepasar este ángulo de inclinación el vehículo muestra un comportamiento inestable que da inseguridad al conductor y podría provocar una caída.

4.4. Comparación de los controladores

Como se pudo ver durante las pruebas, los controladores PID y PD difuso fueron capaces de mantener el equilibrio del vehículo, mientras que el controlador supertwisting, debido a que el sistema es de grado relativo dos, no pudo mantener el equilibrio del vehículo presentando oscilaciones que pueden derribar al tripulante al conducir el vehículo. El PD difuso le da un control más suave al vehículo haciendo más fácil su manejo.

La sintonización de las ganancias y parámetros de cada controlador se realizó a prueba y error debido a que estos controladores son de caja negra es decir que no ocupan conocer

la planta para poder ser utilizado. Se opto por usar este tipo de controladores debido a que los términos de fricción viscosa son muy difíciles de modelar y las incertidumbres de estos elementos al usar controladores basados en el modelo de la planta puede contribuir a desestabilizar el sistema.

CAPÍTULO 5

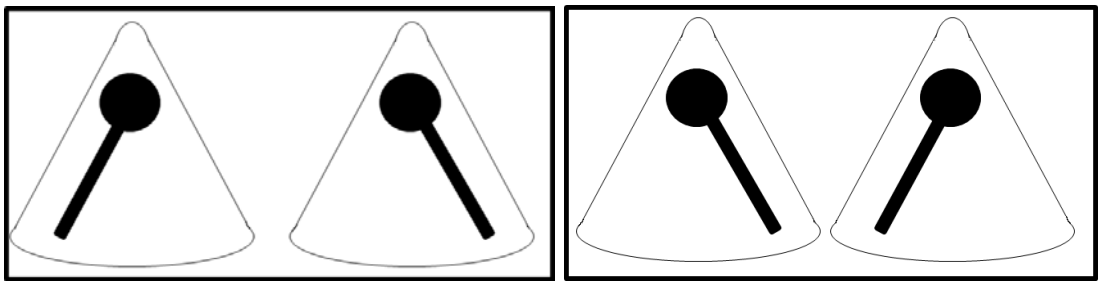
Manual de Operación

Este capítulo describe a detalle cómo conducir el vehículo autobalanceado de dos ruedas construido, la forma de operar la interfaz de datos y como modificar el funcionamiento del mismo para futuras adecuaciones o mejoras.

5.1. Conducción

Para conducir el vehículo de dos ruedas autobalanceado hay que tener en cuenta los siguientes aspectos:

1. Debemos asegurarnos de que los motores estén perfectamente acoplados a sus cajas de engranes, esto se puede realizar asegurando la posición de las palancas del embrague de los motores que se encuentran en la parte frontal del vehículo como se muestra en la figura 5.1a.



(a) Posición para acoplar motores (b) Desacoplo de los engranes

Figura 5.1: Posiciones del embrague de los motores

La posición mostrada en la figura 5.1b es para poder mover el vehículo sin encenderlo ya que suelta el eje del motor de la transmisión y permite el fácil arrastre del vehículo.

2. Una vez que aseguramos el acoplamiento de los motores, sujetamos por el manubrio el vehículo y activamos el Switch principal que es un switch rojo ubicado a la mitad del manubrio (ver figura 5.2)

Figura 5.2: Switch principal.

al encender el vehículo se prenderá automáticamente el LED de “Encendido” de la interfaz de usuario como se ve en la figura 5.3 .

Figura 5.3: Interfaz de usuario. Vehículo encendido.

3. Una vez encendido el vehículo para que el piloto suba a conducir el vehículo debe esperar hasta que el LED de "Control Activo" como se muestra en la figura 5.4 esté encendido ya que éste indica que el microcontrolador ya se encuentra aplicando control sobre el vehículo y sólo hasta entonces es seguro el funcionamiento del vehículo.

Figura 5.4: Interfaz de usuario y de control, Control activo.

4. Una vez listo el vehículo para su operación los comandos para poder conducirlo se enlistan a continuación

- Adelante.- El piloto deberá de inclinar el manubrio hacia adelante así la acción correctiva que hará el vehículo para mantener la posición vertical será avanzar hacia adelante.
- Atrás.- El piloto deberá de inclinar el vehículo hacia si mismo, así la acción correctiva que hará el vehículo para mantener la posición vertical sera retroceder. Avanzar hacia atrás es una acción peligrosa, sólo se recomienda inclinarse levemente para frenar el vehículo, para operar el vehículo se requiere tener experiencia en el manejo del mismo.
- Giro a la derecha.- El piloto podrá hacer un giro hacia la derecha accionando el joystick derecho hacia el lado derecho.
- Giro a la izquierda.- El piloto podrá hacer un giro hacia la izquierda accionando el joystick izquierdo hacia la izquierda.
- Giro sobre el mismo eje del vehículo. Para poder girar sobre el propio eje del vehículo se deberá estar totalmente detenido y accionar los dos joystick al mismo tiempo hacia el mismo lado.
- Aumentar la velocidad en línea recta. Para aumentar la velocidad en línea recta, el piloto deberá girar el joystick izquierdo hacia la izquierda y al mismo tiempo girar el joystick derecho hacia la derecha.
- Disminuir la velocidad en línea recta. Para disminuir la velocidad en línea recta, el piloto deberá girar el joystick izquierdo hacia la derecha y al mismo tiempo girar el joystick derecho hacia la izquierda.

Con estos pasos e instrucciones el piloto puede comenzar a conducir el vehículo lo cual le llevará alrededor de entre 15 a 20 minutos de aprendizaje dependiendo la agilidad de cada piloto.

5.2. Precauciones y recomendaciones

5.2.1. Precauciones

- Asegúrese de que el sistema está controlando el vehículo de manera correcta por medio de los leds del panel de control.
- Esté siempre alerta a los comportamientos del vehículo.

- No incline el vehículo a más de 30^{o} , podría perder el equilibrio fácilmente.
- Si pierde el equilibrio salte hacia su parte trasera para evitar un choque directo con el vehículo.
- No conduzca el vehículo en caminos con muchas irregularidades.

5.2.2. Recomendaciones

- Use casco y las protecciones necesarias para la conducción.
- Siempre cargue la batería del vehículo después de usarlo.
- No tenga miedo al conducir, eso lo puede sacar de balance.
- Asegúrese de que las baterías estén con carga llena antes de usar.

5.3. Mantenimiento y mejoras

En esta sección se mostrarán los pasos a seguir para poder modificar el funcionamiento del vehículo, mejorarlo y como cambiar la parte del control para implementación de nuevos controladores.

5.3.1. Estructura mecánica

Para poder darle mantenimiento el vehículo fue construido de aluminio y ensamblado con tornillería lo cual facilita el desmantelar el vehículo sin suponer mucha dificultad. Todas las piezas de la base y el manubrio a la base están unidos con tornilleria tipo allen, las llantas y los motores están fijados con tornillos estándar de media y dos y media pulgadas respectivamente, los comandos de dirección y el panel de control están fijados con tornillería estándar de media pulgada.

5.3.2. Funcionamiento

Para modificar el funcionamiento del vehículo autobalanceado de dos ruedas es necesario modificar el firmware de los microcontroladores para cambiar tanto señalamientos del panel de control así como el control del vehículo en sí. Para modificar dichos programas es necesario tener instalado el compilador CCS y el XC16 para programar el microcontrolador auxiliar y el principal respectivamente.

5.3.2.1. Microcontrolador de Monitoreo

Para modificar el funcionamiento del microcontrolador auxiliar o de monitoreo lo tendremos que hacer en el compilador en lenguaje C CCS lo cual nos permitirá modificar las diferentes librerías desarrolladas para el proyecto.

Para modificar la configuración del microcontrolador y la selección de pines de entrada para el microcontrolador sera necesario modificar el archivo **config.h.**

Para cambiar los parámetros configurados y leer los sensores conectados a la red I^2C es necesario modificar el archivo **IMU.h.**

La lectura de los encoders se lleva a cabo en la librería **encoders.h.**

La inicialización de los parámetros se modifica desde el archivo principal de este proyecto, el archivo **main.c**

Para el monitoreo de las baterías se utiliza el archivo **baterias.h** .

El despliegue de los mensajes en la LCD del panel de control se realiza en la librería **depliegue.h**.

La comunicación entre microcontroladores se lleva a cabo en la libreria **com.h**.

5.3.2.2. Microcontrolador de Control

Si se desea modificar el funcionamiento del microcontrolador que realiza la tarea de control es necesario tener instalado el programa MPLAB X y tener instalado el compilador XC16 los cuales nos permitirán modificar las librerías hechas para las principales tareas del funcionamiento del vehículo autobalanceado de dos ruedas.

Para modificar la configuración del microcontrolador principal tenemos que modificar el archivo **Config.h**, aquí podremos cambiar los bits de configuración del microcontrolador.

En el archivo **sensores.h** podemos cambiar la recepción de datos del microcontrolador auxiliar y la conversión de estos datos a los valores de aceleración, velocidad angular y velocidad lineal de los sensores.

La configuración de las salidas PWM es necesario cambiar el archivo **PWM.h.**

El archivo **Kalman.h** tiene la implementación del filtro de Kalman para la obtención de la medición del ángulo, esta librería fue desarrollada en [Bonales 2012].

La principal tarea que desarrolla el microcontrolador principal es la de control y ésta se encuentra en el archivo Control.h en el cual se encuentra comentado que es lo necesario que hay que modificar para poder implementar cualquier controlador en la función controlador contenida en este archivo.

Para inicializar las variables y configuración de periféricos es necesario acceder al archivo principal del proyecto **main.h**.

5.4. Monitoreo de variables

Para poder sintonizar y ajustar los parámetros del filtro de Kalman y de los controladores implementados en el vehículo de dos ruedas autobalanceado se diseñó la tarjeta principal de tal forma que puede existir una comunicación serial entre la PC y la tarjeta de control mediante una interfaz de comunicación utilizando la UART del microcontrolador principal para acceder a las variables del sistema desde la computadora y así poder graficar en tiempo real el comportamiento de éstas y así decidir como ajustar los parámetros requeridos. Para poder obtener los datos es necesario la aplicación PICKIT 2 o PICKIT 3 (dependiendo el programador con el que se cuenta ver figura 5.5) la cual además de permitirnos cargar los programas a los microcontroladores tiene una herramienta llamada **UART Tool** a la cual podemos acceder mediante el menú **Tools** como se muestra en la figura 5.6.

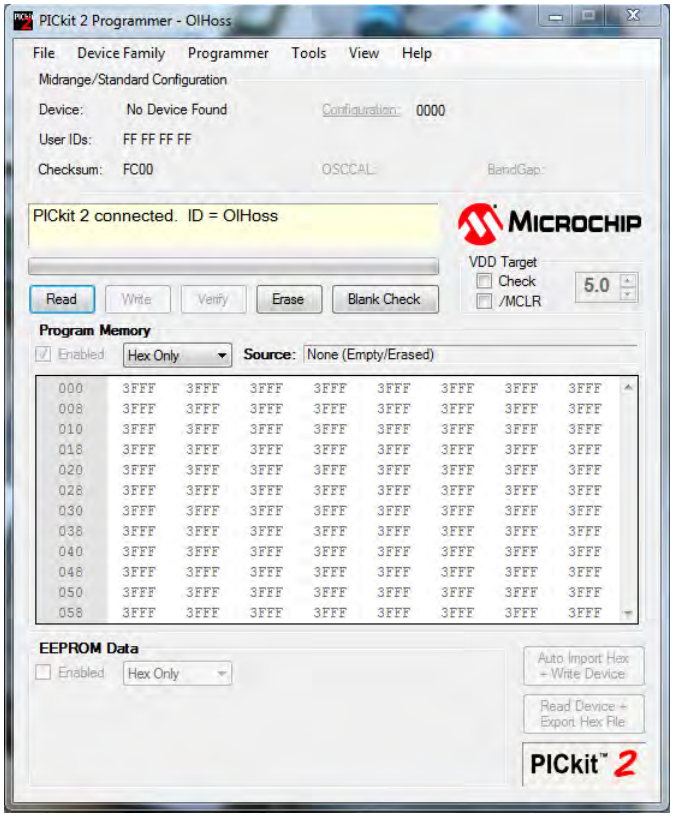


Figura 5.5: Interfaz de la apliación PICKIT 2

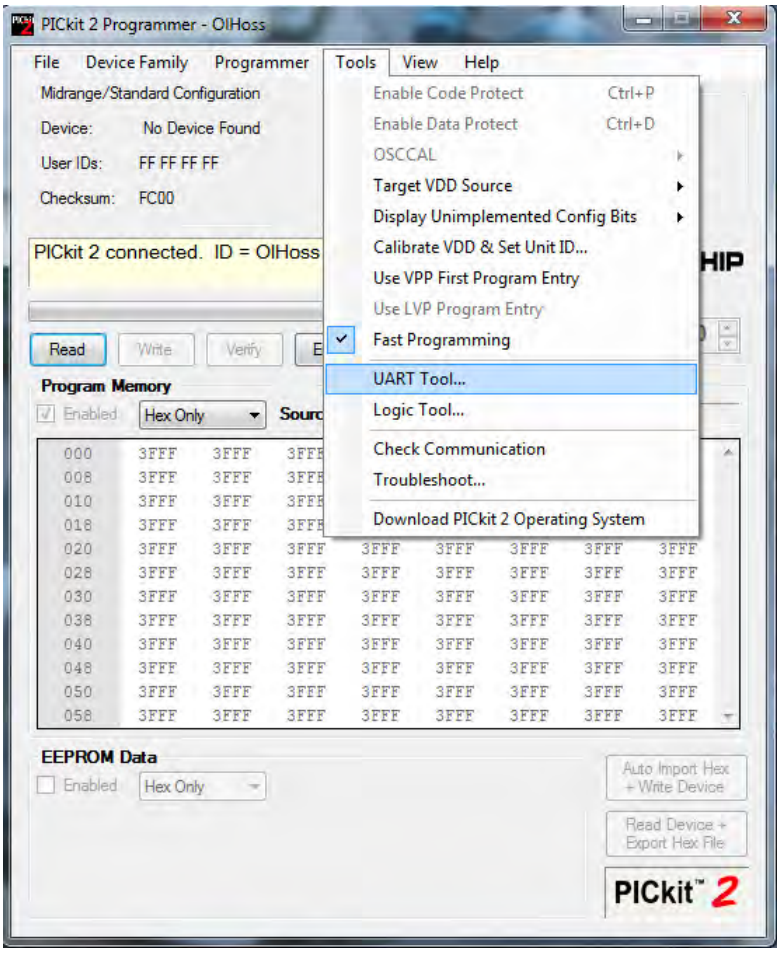


Figura 5.6: Acceso a la herramienta UART Tool de la aplicación PICKIT 2

Para poder hacer uso de esta aplicación es necesario contar en el hardware con la interfaz con las conexiones mostradas en la figura 5.7 y así podremos acceder a la herramienta de UART de la aplicación.

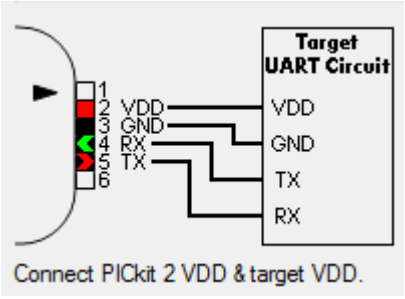


Figura 5.7: Interfaz de comunicación

Una vez dentro la aplicación permite configurar la velocidad de transición a la cual esta-

remos enviando la información desde el microcontrolador como se muestra en la figura 5.8 además de poder mandar datos, configurar un eco, y algunas opciones de configuración la herramienta UART nos permite generar un archivo **log** el cual tendrá guardado todos los datos que se transfieran mientras esté la interfaz en el estado **Connect**.

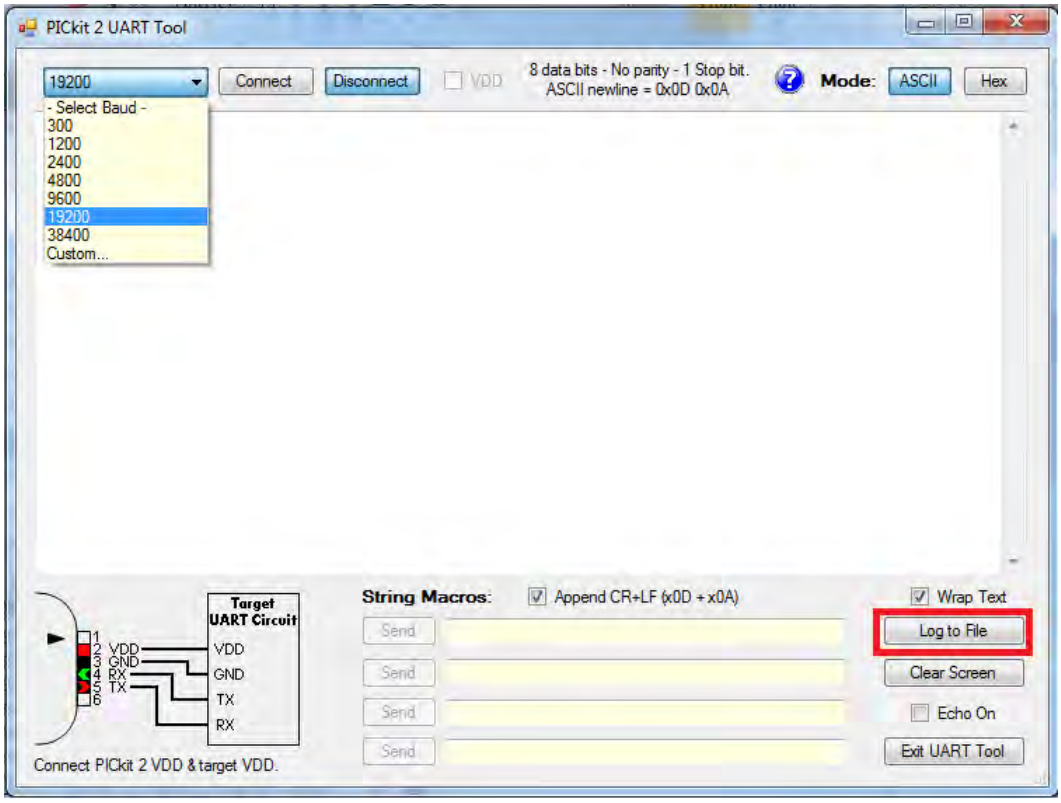


Figura 5.8: Interfaz UART Tool

Respecto al programa del microcontrolador, éste deberá ser modificado en la línea mostrada en la figura 5.9, donde únicamente podremos acceder a dos variables a la vez ya que si excedemos este número el tiempo de muestreo se incrementa y se deberá de hacer cambios en la programación de la ley de control.

```
87 |          printf("%f?%f",angulo,acel);
```

Figura 5.9: Línea para modificar variables a monitorear

Una vez adquiridas las mediciones en tiempo real en el archivo tipo log generado por la interfaz por medio de un programa como Excel se puede crear una tabla de dos columnas

usando como separador el símbolo **?** el cual está programado ya en el microcontrolador de control de acuerdo a la figura (5.9), una vez hecho esto se pueden graficar los datos obtenidos.

5.5. Mantenimiento mecánico

Las partes mecánicas que ocupan un mantenimiento continuo para el buen funcionamiento del vehículo autobalanceado de dos ruedas son:

- Los motores, revisar periódicamente el nivel de lubricación de la caja de engranes y en caso ser necesario cambiar el aceite lubricante.
- Los tornillos que sujetan las llantas a los ejes de los motores, esta parte es la más susceptible a fallas de acuerdo a las pruebas realizadas y es necesario cambiar este tornillo cada vez que se rompa. Esto se debe corregir mediante otro tipo de acoplo de la llanta en un prototipo futuro.
- Las baterías, se deben cargar y descargar completamente al menos una vez al mes, esta recomendación está dada por el fabricante para prolongar su vida útil.

CAPÍTULO 6

Conclusiones y Trabajo Futuro

6.1. Conclusión

En este trabajo se han presentado los detalles de construcción de un prototipo funcional del vehículo autobalanceado de dos ruedas unipersonal, así como algunos esquemas de control para lograr su funcionamiento. El vehículo fue construido en su totalidad, su estructura mecánica, sistemas electrónicos y software asociado, consolidando el correcto funcionamiento del vehículo cumpliendo así con los objetivos planteados.

El vehículo diseñado es totalmente funcional, permite transportar a una persona de manera eficaz de un punto a otro sin mayores dificultades, una vez que éste ha practicado lo suficiente. Sin embargo, aún son necesarias algunas mejoras para facilitar el aprendizaje de control básico del vehículo, el cual continúa siendo dependiente, en alguna medida, de las habilidades de cada piloto.

El controlador PID implementado se hizo en base a la teoría desarrollada en el trabajo de [Bonales 2012], con un ajuste de parámetros que se realizó de manera empírica a través de varias pruebas realizadas. El controlador PD difuso mínimo fue incorporado en esta investigación con el cual se pudo obtener un control más suave y estable que el PID clásico. En cuanto al algoritmo supertwisting no se logró el objetivo de mantener el equilibrio del vehículo mientras se conduce.

En cuanto a la parte electrónica, el diseño de sus componentes cumplió satisfactoriamente los objetivos planteados, haciendo del vehículo un equipo confiable con comportamiento predecible. Es posible operarlo suponiendo que los componentes funcionan de forma correcta y que no es necesario tener precauciones.

Mecánicamente, el vehículo tiene muy buenas características, es ligero, resistente a golpes

y fácil de desarmar para su mantenimiento. Sin embargo el tornillo que fija las llantas con el eje del motor, sufre grandes esfuerzos mecánicos y hacen que éste sea un punto potencialmente vulnerable. Por otra parte la base es lo suficientemente rígida lo cual ayuda a proteger a los componentes electrónicos de golpes directos, y al ser de aluminio brinda un aspecto estéticamente agradable.

El software realizado resultó ser totalmente funcional, con una estructura jerárquica que facilita su lectura y compresión así como su modificación. El sistema modular de implementación permite de manera fácil poder intervenir el software en alguna tarea específico sin modificar las otras tareas realizadas por el microcontrolador. La función de control es fácil de modificar teniendo únicamente como entrada el ángulo θ y una salida del controlador que tiene que estar entre -6000 y 6000, esta característica busca facilitar trabajos futuros en cuanto a la implementación de más técnicas de control.

Los encoders fueron instalados pero debido a un accidente sufrieron daño en su estructura por un mal diseño en el montaje, por tal razón se desmontaron y se decidió dejarlos fuera hasta encontrar una forma segura de montarlos en los motores del vehículo de dos ruedas autobalanceado.

Con algunas mejoras en el vehículo autobalanceado se podría pensar en la implementación a escala comercial.

6.2. Trabajo Futuro

Lo realizado en este trabajo da paso a tres posibles líneas de continuación para el proyecto: La determinación de un modelo más exacto del sistema, el estudio de nuevas técnicas de control para el prototipo y la mejora del diseño mecánico y electrónico.

En el diseño actual, aún queda pendiente el montaje de los encoders a los motores, ya que se tiene el software listo para la incorporación, pero debido a que no se encontró una forma correcta de montarlos sin poner en riesgo la integridad de éstos se optó por mantenerlos fuera del prototipo. La incorporación de los encoders permitirá explorar el efecto de un controlador de velocidad sobre los neumáticos. Unas mejoras electrónicas pueden ser el mejoramiento de la tarjeta principal optando por un microcontrolador más potente que pueda encargarse de todas las tareas simultáneamente sin necesidad de estar en comunicación con otro, ya que la comunicación provoca retardos muy grandes que afectan en el periodo de muestreo del sistema, también se puede hacer la instalación de una interfaz inalámbrica de comunicación para facilitar la captura de datos del sistema en movimiento.

Futuras investigaciones pueden enfocarse a un modelado más riguroso del sistema con el fin de poder implementar estrategias de control de mejor desempeño, y así poder controlar tal vez la corriente en los motores, la velocidad del vehículo además del ángulo de inclinación. Del mismo modo, se puede mejorar el algoritmo del filtro de Kalman para obtener mejores lecturas.

El vehículo puede ser rediseñado en base a la plataforma ya desarrollada para obtener un vehículo con mejores características que permita una fácil conducción para cualquier pasajero en condiciones normales. Las baterías de gel son un peso significativo que los motores deben mover, este peso puede ser reducido a una tercera parte, de reemplazarlas por tecnología de ion de Litio, cuya densidad de energía es alrededor de tres veces más que la de las baterías convencionales.

El montaje de las llantas directamente sobre el motor produce un gran problema el cual se puede corregir usando coples flexibles para el montaje de las llantas sobre la estructura eliminando el problema que hace que se rompan los tornillos de sujeción.

Un nuevo diseño del vehículo puede incluir materiales mas ligeros como algún polímero ligero y otras aleaciones para darle mas estética y menor peso.

Con estas adecuaciones se podría pasar a la implementación a escala local o comercial.

Apéndice A

Diagramas Eléctricos

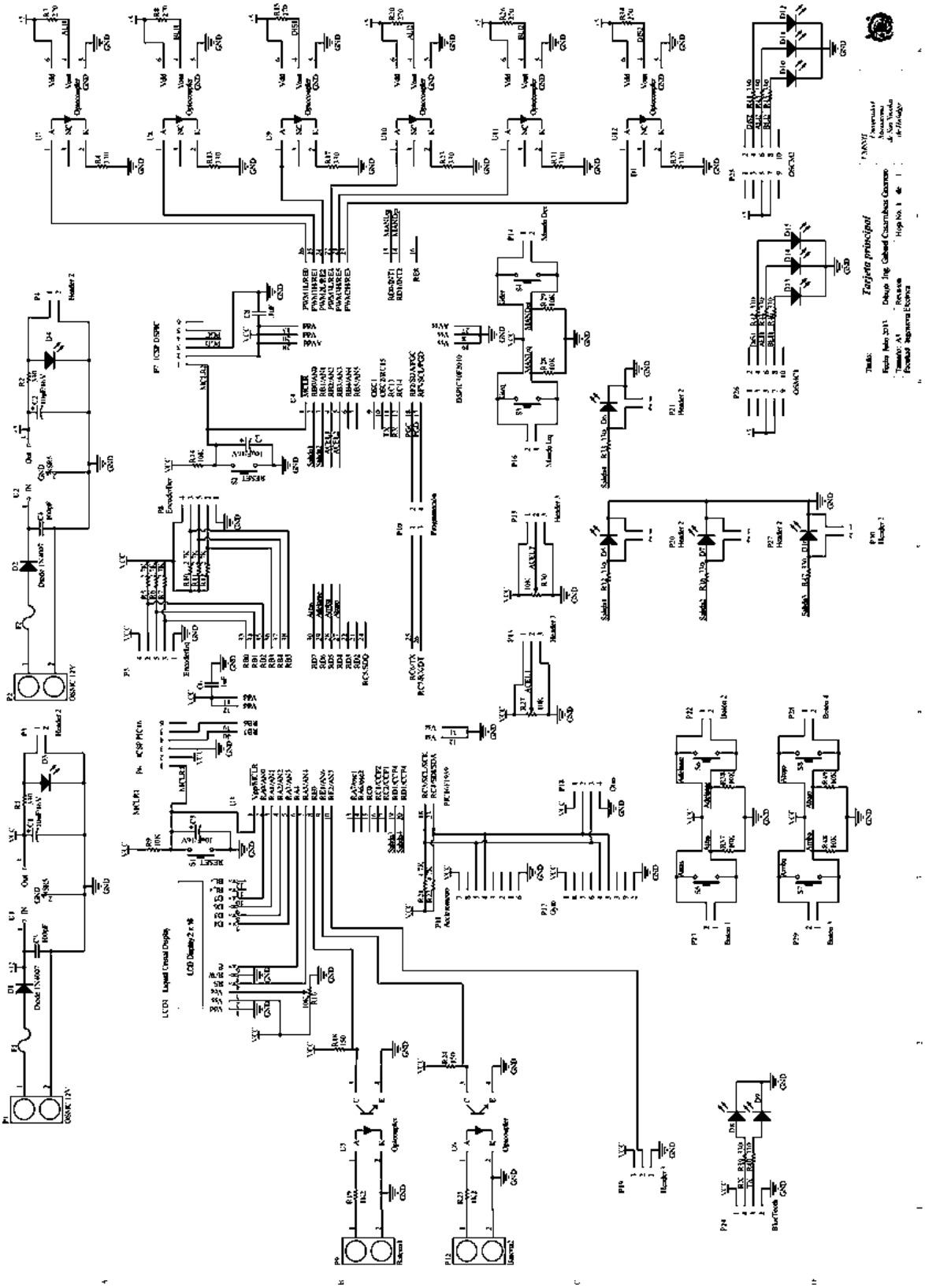


Figura A.1: Diagrama esquemático de tarjeta principal.

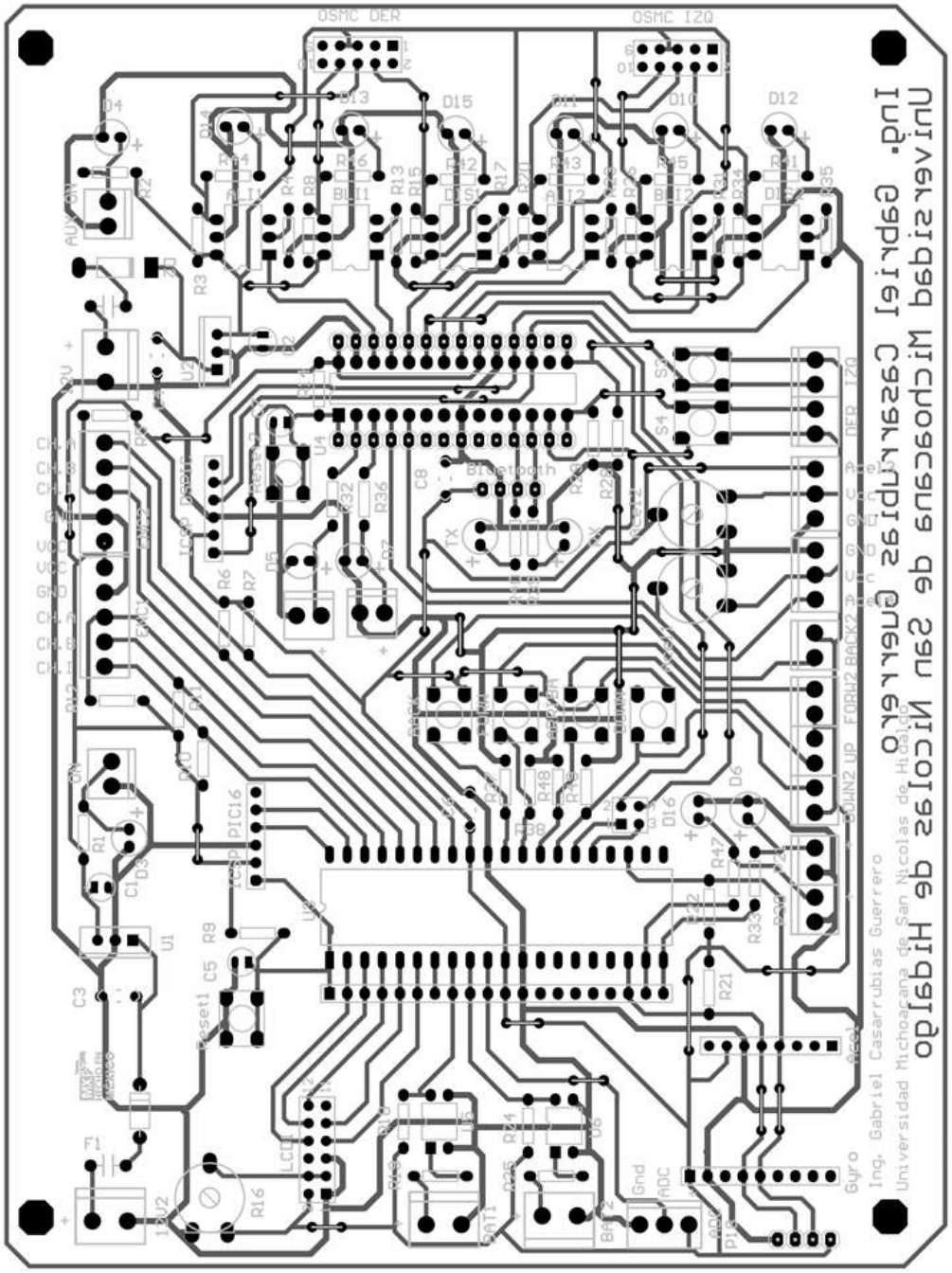


Figura A.2: PCB fabricado para el vehículo de dos ruedas autobalanceado.

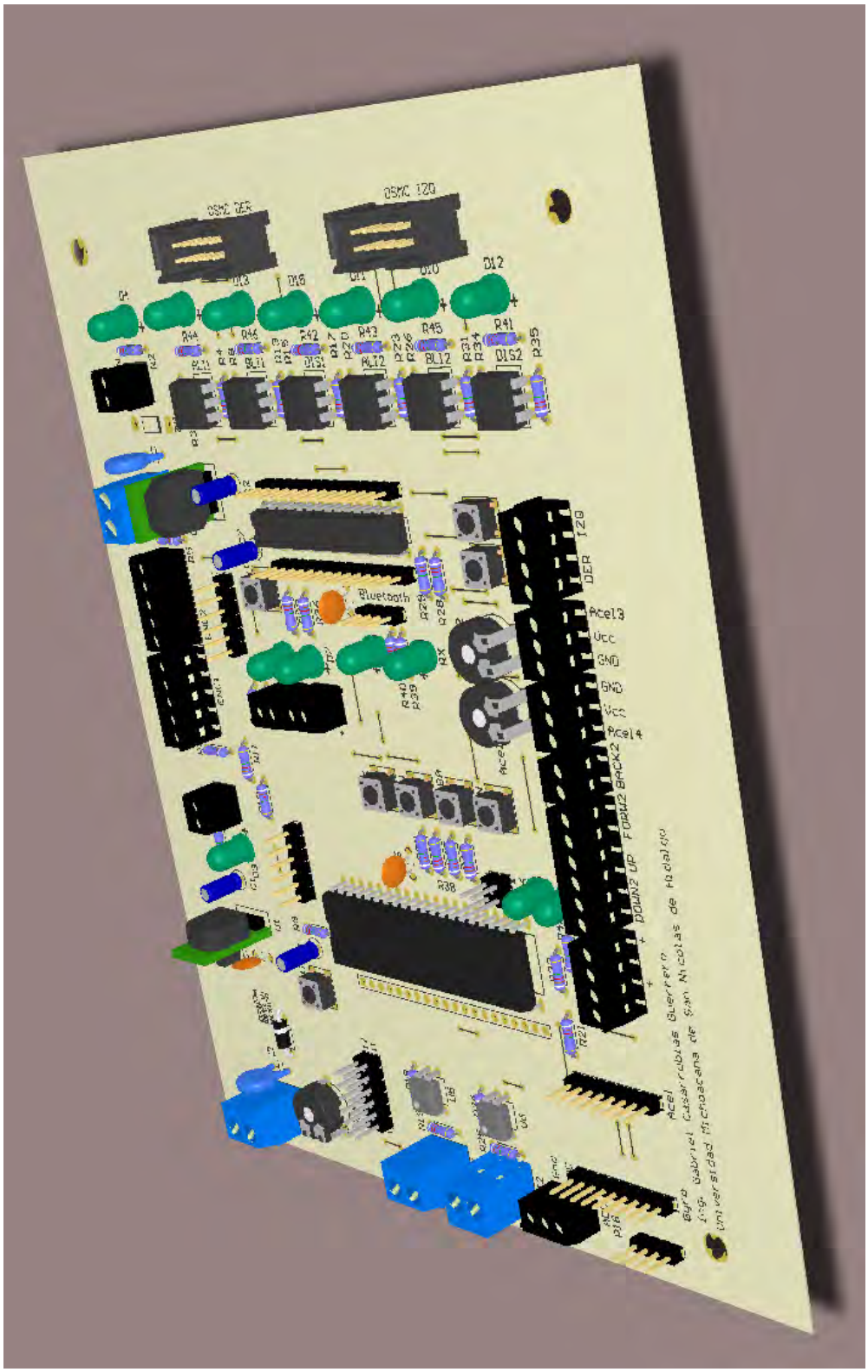


Figura A.3: Modelo en 3D del PCB construido.

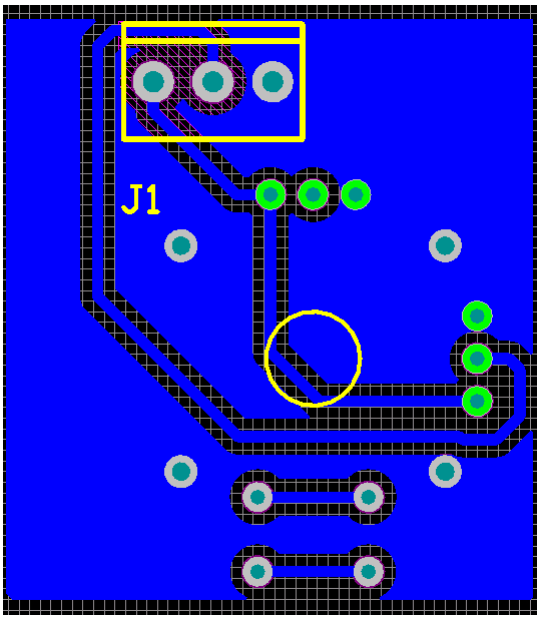


Figura A.4: PCB para joystick

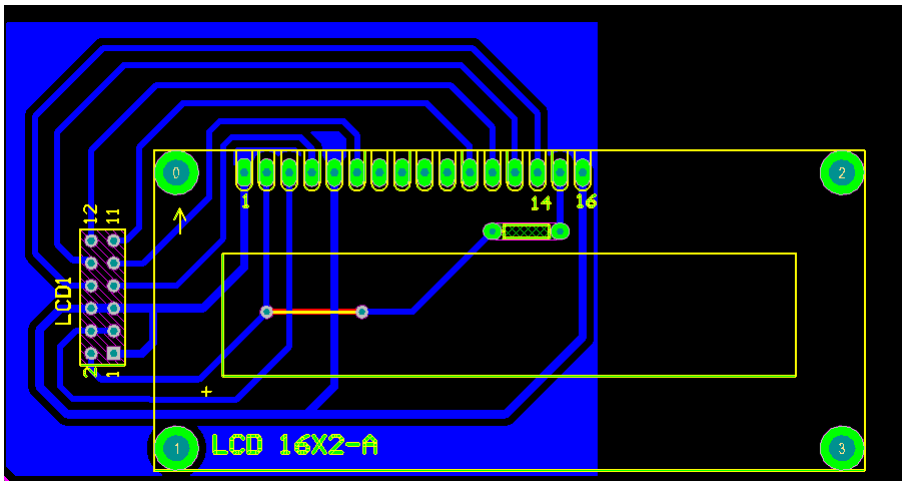


Figura A.5: PCB elaborado para LCD

Apéndice B

Código y Diagramas en Simulink

Para la generación de la simulación se utilizo código elaborado en MATLAB R2012A y la herramienta Simulink en la versión 7.9

B.1. Código a ejecutar para simulación

```
%%Simulación para vehiculo autobalanceado sobre dos ruedas
clc
clear all
%%Tiempo de simulación**********************************
simu=10;
%%*****************************************************
%parametros del vehiculo
g=9.81;     %fuerza de gravedad
mr=3;       %masa de la rueda
R=.2;       %Radio de la llanta
L=.85;      %distancia del centro de la masa mb con respecto al origen
D=.4;       %distancia del centro del vehículo al centro de la llanta
%%*****************************************************
%%
a=.8;       %Ancho del cuerpo inferior
b=.18;      %Altura del cuerpo inferior
c=1.7;      %Altura del cuerpo superior
d=.25;      %Ancho aproximado del cuerpo superior
```

```
d1=.85;    %Distancia del centro del cuerpo superior al eje central
d2=.09;    %Distancia del centro del cuerpo inferior al eje central
%mh=0;     %Masa del cuerpo humano
mB=39.5;   %Masa de la base del prototipo
%mb=mB+mh;%masa del cuerpo del vehiculo mas tripulante
Ir=1/2*mr*R^2
%Itheta1=mh*d1^2+d*c^3/12;
Itheta2=mB*d2^2+a*b^3/12;
%Itheta=Itheta1+Itheta2;
%gamma=-mb*R*L;
%zetta=mr*R^2+1/2*mb*R^2+Ir;
%v=mb*L^2+Itheta;
%%Parametros del actuador **********************************
Ra=1.5;
Ke=.0001;
Kt=.2;
%%**********************parametros de control*************
T=.017;        %tiempo de muestreo
mh0=120;        %masa inicial del sistema
mhd=0;         %masa adicional para la simulación
%(se puede simular la acción de subida del piloto)
%%Parametros del controlador PD difuso mínimo
Le=0.25;
%Lde=0.03;
Lde=1.2;
%Lde=1;
%H=250;
H=200;
%H=100;
%%Ganancias supertwisting
K1=2500;
K2=100;
%%Ganancias PID digital
Kp=3000;
```

```
Ti=10;
Td=.01;
%Kp=6000;
%Ti=.5;
%Td=.001;
%Td=2000;
%Td=10;
N=1;
%%Constantes para los terminos de fricción viscosa
btheta=30;
%btheta=300000; funciona
%btheta=5000000;
%btheta=30;
%balfa=10;
%balfa=50; funciona
balfa=3;
%%Condiciones iniciales del sistema
x01=0;
x02=0;
x03=0;
x04=0;
%%              Simulación
sim('esquema4');
%%              Grafcación
lon=length(angulo);
paso=simu/lon;
tiempo=paso:paso:simu;
figure(3);
subplot(2,1,1);
plot(tiempo,angulo(:,2));
legend('Ángulo de inclinación')
xlabel('tiempo (s)');
ylabel('Ángulo (grados)')
grid on
```

```
subplot(2,1,2);
plot(tiempo,entrada(:,2));
legend('Entrada de control')
xlabel('tiempo (s)');
ylabel('Entrada (Cuentas de PWM)')
grid on
%%
```

B.1.1. Diagramas de Simulink

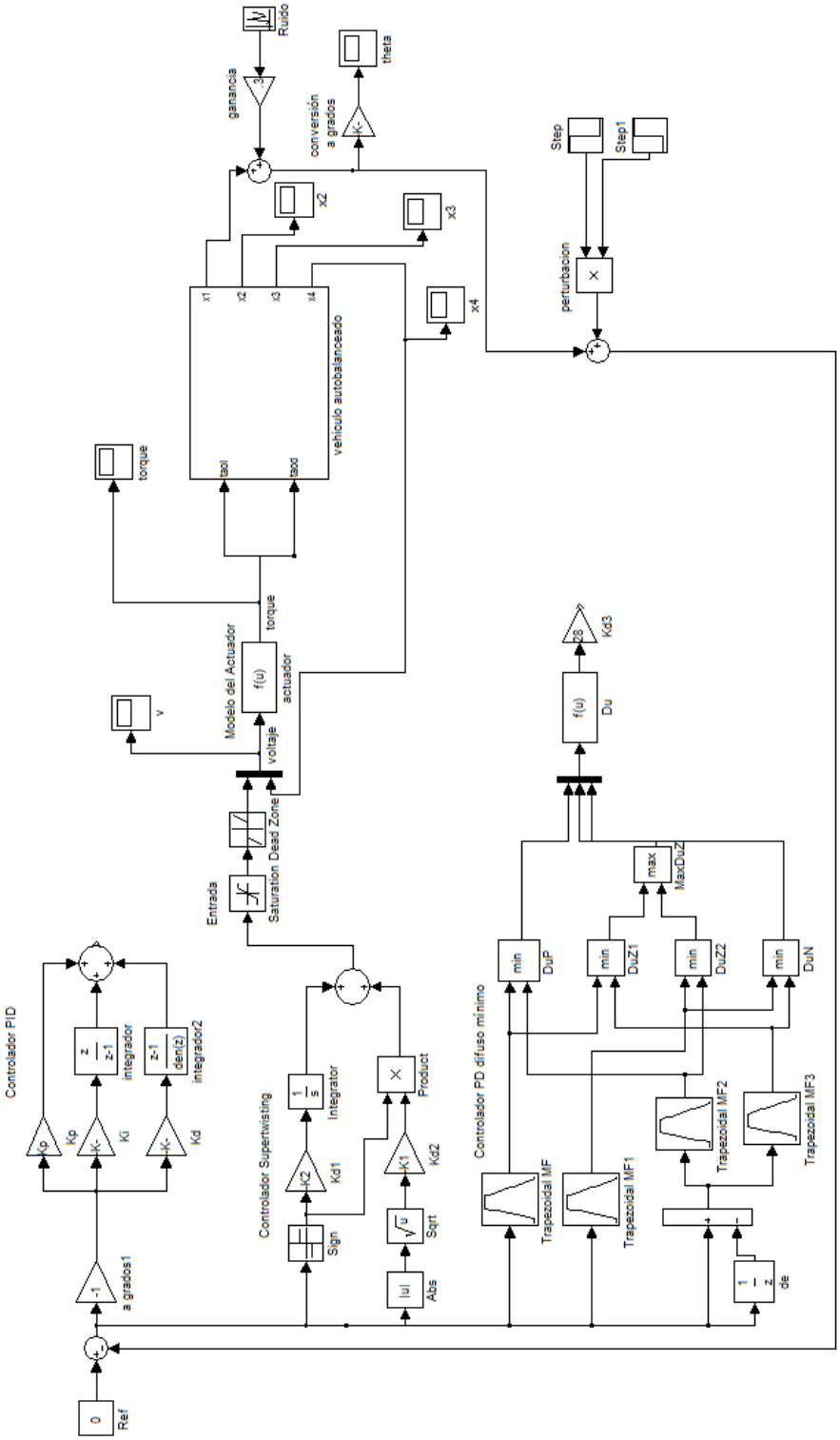


Figura B.1: Sistema con controladores

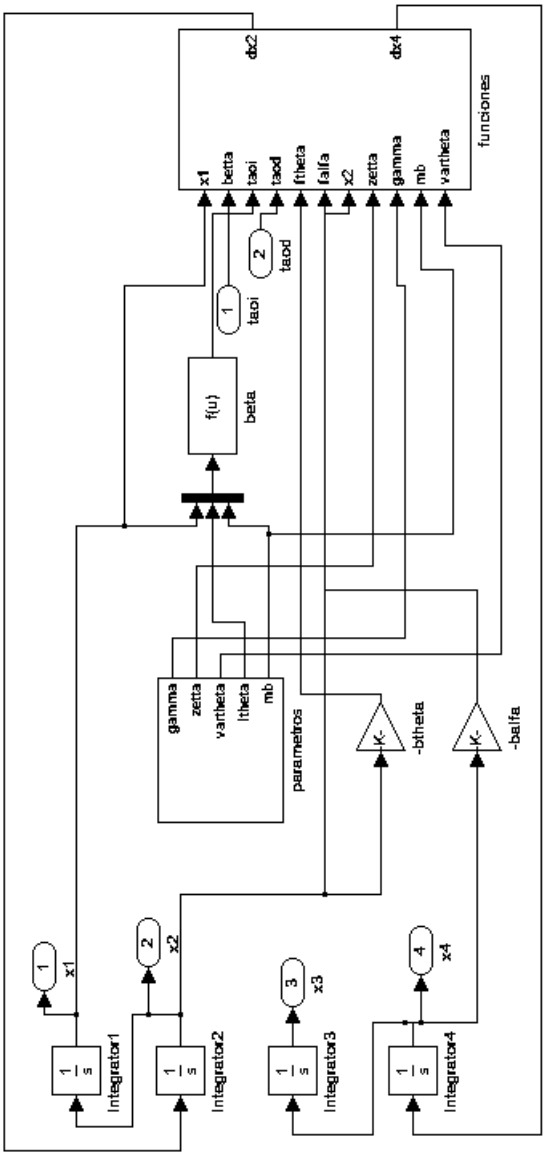


Figura B.2: Diagrama de la planta

Apéndice C

Programas

C.1. Microcontrolador Auxiliar

La programación del microcontrolador auxiliar se realizó por medio del compilador de c CCS por la facilidad de uso de librerías para la comunicación con los sensores inerciales.

C.1.1. Configuración archivo Config.h

```
#fuses INTRC_IO      //oscilador cristal
#fuses NOWDT         //No watchdog
#fuses mclr          //Master clear habilitado
#fuses noprotect     //Codigo sin proteccion de escritura y lectura
#fuses NOLVP,NODEBUG
//Programación en bajo voltaje desactivada y modo debug
#fuses PUT,PLL       //activado Power On Timer y Phase Locked Loop
#device adc=10       //ADC a 10 bits de resolución
#use delay(clock=32000000)
//se asigna la frecuencia del reloj a 32Mhz
#priority timer1,ext,rb     //prioridad de interrupción
        //          1.- Timer 1
        //          2.- Interrupción Externa
        //          3.- Interrupción por cambio de estado de puerto B
//declaración de    pines de LCD
#define LCD_DB4   PIN_A5
```

```
#define LCD_DB5   PIN_A2
#define LCD_DB6   PIN_A1
#define LCD_DB7   PIN_A0
#define LCD_RS    PIN_A4
#define LCD_E     PIN_A3
#include "flex_lcd.c"
#byte PORTD=0x00F    //se asigna dirección de Puerto D
//#byte PIR1 = 0x011
//#byte PIE1   =0x091
//*************************************************************
#use rs232(baud=19200, xmit=PIN_C6, rcv=PIN_C7, bits=8,ERRORS)
//definimos uso de USART y su configuración
```

C.1.2. Manejo de LCD, FLEX_LCD.h

```
// flex_lcd.c
// These pins are for the Microchip PicDem2-Plus board,
// which is what I used to test the driver.  Change these
// pins to fit your own board.

//#define LCD_DB4   PIN_B4
//#define LCD_DB5   PIN_B5
//#define LCD_DB6   PIN_B6
//#define LCD_DB7   PIN_B7
//
//#define LCD_RS    PIN_C0
//#define LCD_RW    PIN_C1
//#define LCD_E     PIN_C2

// If you only want a 6-pin interface to your LCD, then
// connect the R/W pin on the LCD to ground, and comment
// out the following line.

//#define USE_LCD_RW   1
```

```
//=====================================

#define lcd_type 2          // 0=5x7, 1=5x10, 2=2 lines
#define lcd_line_two 0x40 // LCD RAM address for the 2nd line


int8 const LCD_INIT_STRING[4] =
{
 0x20 | (lcd_type << 2), // Func set: 4-bit, 2 lines, 5x8 dots
 0xc,                    // Display on
 1,                      // Clear display
 6                       // Increment cursor
 };


//-------------------------------------
void lcd_send_nibble(int8 nibble)
{
// Note:  !! converts an integer expression
// to a boolean (1 or 0).
 output_bit(LCD_DB4, !!(nibble & 1));
 output_bit(LCD_DB5, !!(nibble & 2));
 output_bit(LCD_DB6, !!(nibble & 4));
 output_bit(LCD_DB7, !!(nibble & 8));

 delay_cycles(1);
 output_high(LCD_E);
 delay_us(2);
 output_low(LCD_E);
}

//-------------------------------------
// This sub-routine is only called by lcd_read_byte().
// It's not a stand-alone routine.  For example, the
```

```
// R/W signal is set high by lcd_read_byte() before
// this routine is called.

#ifdef USE_LCD_RW
int8 lcd_read_nibble(void)
{
int8 retval;
// Create bit variables so that we can easily set
// individual bits in the retval variable.
#bit retval_0 = retval.0
#bit retval_1 = retval.1
#bit retval_2 = retval.2
#bit retval_3 = retval.3

retval = 0;

output_high(LCD_E);
delay_cycles(1);

retval_0 = input(LCD_DB4);
retval_1 = input(LCD_DB5);
retval_2 = input(LCD_DB6);
retval_3 = input(LCD_DB7);

output_low(LCD_E);

return(retval);
}
#endif

//-----------------------------------------
// Read a byte from the LCD and return it.

#ifdef USE_LCD_RW
```

```
int8 lcd_read_byte(void)
{
int8 low;
int8 high;

output_high(LCD_RW);
delay_cycles(1);

high = lcd_read_nibble();

low = lcd_read_nibble();

return( (high<<4) | low);
}
#endif

//-----------------------------------------
// Send a byte to the LCD.
void lcd_send_byte(int8 address, int8 n)
{
output_low(LCD_RS);

#ifdef USE_LCD_RW
while(bit_test(lcd_read_byte(),7)) ;
#else
delay_us(60);
#endif

if(address)
   output_high(LCD_RS);
else
   output_low(LCD_RS);

 delay_cycles(1);
```

```
#ifdef USE_LCD_RW
output_low(LCD_RW);
delay_cycles(1);
#endif

output_low(LCD_E);

lcd_send_nibble(n >> 4);
lcd_send_nibble(n & 0xf);
}

//-----------------------------------
void lcd_init(void)
{
int8 i;

output_low(LCD_RS);

#ifdef USE_LCD_RW
output_low(LCD_RW);
#endif

output_low(LCD_E);

delay_ms(15);

for(i=0 ;i < 3; i++)
    {
     lcd_send_nibble(0x03);
     delay_ms(5);
    }

lcd_send_nibble(0x02);
```

```
for(i=0; i < sizeof(LCD_INIT_STRING); i++)
   {
    lcd_send_byte(0, LCD_INIT_STRING[i]);

    // If the R/W signal is not used, then
    // the busy bit can't be polled.  One of
    // the init commands takes longer than
    // the hard-coded delay of 60 us, so in
    // that case, lets just do a 5 ms delay
    // after all four of them.
    #ifndef USE_LCD_RW
    delay_ms(5);
    #endif
   }

}

//-----------------------------------

void lcd_gotoxy(int8 x, int8 y)
{
int8 address;

if(y != 1)
   address = lcd_line_two;
else
   address=0;

address += x-1;
lcd_send_byte(0, 0x80 | address);
}

//-----------------------------------
```

```
void lcd_putc(char c)
{
 switch(c)
   {
    case '\f':
      lcd_send_byte(0,1);
      delay_ms(2);
      break;

    case '\n':
       lcd_gotoxy(1,2);
       break;

    case '\b':
       lcd_send_byte(0,0x10);
       break;

    default:
       lcd_send_byte(1,c);
       break;
   }
}

//-----------------------------------
#ifdef USE_LCD_RW
char lcd_getc(int8 x, int8 y)
{
char value;

lcd_gotoxy(x,y);

// Wait until busy flag is low.
while(bit_test(lcd_read_byte(),7));
```

```
output_high(LCD_RS);
value = lcd_read_byte();
output_low(lcd_RS);

return(value);
}
#endif

void lcd_setcursor_vb(short visible, short blink) {
  lcd_send_byte(0, 0xC|(visible<<1)|blink);
}
```

C.2. Microcontrolador Principal

El microcontrolador principal se programo con el Software MPLAB X y el compilador XC16 de Microchip el cual esta optimizado para trabajar con todos los microcontroladores de la gama de 16 bits de Microchip.

C.2.1. Configuración archivo, Config.h

```
// FOSC
#pragma config FPR = FRC_PLL16
// Primary Oscillator Mode (FRC w/ PLL 16x)
#pragma config FOS = PRI
// Oscillator Source (Internal Fast RC)
#pragma config FCKSMEN = CSW_FSCM_OFF
// Clock Switching and Monitor (Sw Disabled, Mon Disabled)

// FWDT
#pragma config FWPSB = WDTPSB_16        // WDT Prescaler B (1:16)
#pragma config FWPSA = WDTPSA_512       // WDT Prescaler A (1:512)
#pragma config WDT = WDT_OFF            // Watchdog Timer (Disabled)

// FBORPOR
#pragma config FPWRT = PWRT_64          // POR Timer Value (64ms)
```

```
#pragma config BODENV = BORV20          // Brown Out Voltage (Reserved)
#pragma config BOREN = PBOR_ON          // PBOR Enable (Enabled)
#pragma config LPOL = PWMxL_ACT_HI
// Low-side PWM Output Polarity (Active High)
#pragma config HPOL = PWMxH_ACT_HI
// High-side PWM Output Polarity (Active High)
#pragma config PWMPIN = RST_IOPIN
// PWM Output Pin Reset (Control with PORT/TRIS regs)
#pragma config MCLRE = MCLR_EN          // Master Clear Enable (Enabled)

// FGS
#pragma config GWRP = GWRP_OFF
// General Code Segment Write Protect (Disabled)
#pragma config GCP = CODE_PROT_OFF
// General Segment Code Protection (Disabled)

// FICD
#pragma config ICS = ICS_PGD
// Comm Channel Select (Use PGC/EMUC and PGD/EMUD)
#define _XTAL_FREQ 120000000      //Frecuencia de clock 12MHz
#define FCY (_XTAL_FREQ/4)        //FCY
#include <uart.h>                 //Libreria para UART

//******************** Varibles de recepción de datos
char  con=0,con2,acelh,acell,gyroh,gyrol,vel1h,vel1l,vel2h,vel2l,rcvchar;
int izq,de
```

C.2.2. Configuración de PWM, PWM.h

```
/*
 * File:   PWM.h
 * Author: Gabriel
 *
 * Created on 15 de agosto de 2013, 03:03 PM
 */
```

```
//************* definiciones para la salida PWM
#define ALI1 LATEbits.LATE0
#define BLI1 LATEbits.LATE1
#define DIS1 LATEbits.LATE2

#define ALI2 LATEbits.LATE4
#define BLI2 LATEbits.LATE5
#define DIS2 LATEbits.LATE3
//*********************************************

void PWM_INIT(){
    // Configure PWM for free running mode
    //   PWM period = Tcy * prescale * PTPER = 0.33ns * 64 * PTPER
    //   PWM pulse width = (Tcy/2) * prescale * PDCx

    PWMCON1 = 0x00ff;// Enable all PWM pairs in complementary mode
    PWMCON1bits.PMOD1=1;
    PWMCON1bits.PMOD2=1;
    PWMCON1bits.PMOD3=1;

    OVDCONbits.POUT1H=1;
    OVDCONbits.POUT1L=0;
    OVDCONbits.POUT3H=1;
    OVDCONbits.POUT3L=0;



    PTCONbits.PTCKPS = 0;
        // prescale=1:64 (0=1:1, 1=1:4, 2=1:16, 3=1:64)
    PTPER = 3000;   // 5KHz PWM period (15-bit period value)
    PTMR = 0;       // Clear 15-bit PWM timer counter
    PTCONbits.PTEN = 1;    // Enable PWM time base
    PDC1 = 6000;   // channel 1 pulse width = 1.0ms
    PDC3 = 6000;
```

```
    PWMCON1bits.PEN1H=0;
    PWMCON1bits.PEN1L=0;
    PWMCON1bits.PEN2H=0;
    PWMCON1bits.PEN2L=0;
    PWMCON1bits.PEN3H=0;
    PWMCON1bits.PEN3L=0;
    PWMCON1bits.PEN2H=0;
    ALI1=1;
    BLI1=1;
    ALI2=1;
    BLI2=1;
    DIS1=0;
    DIS2=0;

}
```

C.2.3. Medida de sensores, Sensores.h

```
/* ******************************
 *           File:   Control.h
 *           Author: Gabriel Casarrubias Guerrero
 *           Created on 13 de agosto de 2013, 10:35 AM
 ********************************/
/* Interrupción por recepcion de uart1 */
 void __attribute__((interrupt,no_auto_psv)) _U1RXInterrupt(void)
 {
      con++;
    if(DataRdyUART1())          // Si hay algo pendiente de recibir ...
        rcvchar=ReadUART1();     // lo descargo y ...
      //asegura la llegada de todos los datos
     switch(con){
         case 2:{
             acelh=rcvchar;
             break;
         }
```

```
        case 3:{
            acell=rcvchar;
            break;
        }
        case 4:{
            gyroh=rcvchar;
            break;
        }
        case 5:{
            gyrol=rcvchar;
            break;
        }
        case 6:{
            vel1h=rcvchar;
            break;
        }
        case 7:{
            vel1l=rcvchar;
            break;
        }
        case 8:{
            vel2h=rcvchar;
            break;
        }
        case 9:{
            vel2l=rcvchar;
            con=0;
            break;
        }
        }
    IFS0bits.U1RXIF = 0;
    //Ponemos a 0 el flag de la interrupcion de recepcion
    if(U1STAbits.OERR) U1STAbits.OERR=0;
    //si hay error de overflow es borrado
```

```
 }

//calcula el ángulo segun la medida del acelérometro
 float anguloacel(){
     int dato;
     float g,anguloacel;
     dato=acelh;
     dato=dato<<8;
     dato=dato+acell;
     dato=dato>>4;
     g=(float)dato;
     g=g*.001;
         if(g>3.0)
             g=-(4.1-g);
    anguloacel=asin(g);
    return(anguloacel);
 }

//calcula la velocidad angular
 float gyrovel(){
     int dato;
     float vel;
      dato=gyroh;
     dato=dato<<8;
     dato=dato+gyrol;
     dato=dato>>4;
     vel=(float)dato;
     vel=vel*.00875;
     return(vel);
 }
```

C.2.4. Filtro de Kalman, Kalman.h

```
/*
***              Filtro de kalman
```

```
***                M.C. Jorge Bonales
************************************************************
*/
//********Variables que se actualizan**********************
  //Matrz de covarianza P
  float P[2][2]={{100000,0},{0,100000}};

  //matriz inicial de los estados
  float x[2][2]={{0,0},{0,0}};

  //Ganancia de Kalman
  float K[2][1];
  float Pn[2][2];
  float Acel_K;
  float gyro_ant;
  int w=0;



 //************************************************************
float k_filter(float acel_rad,float gyro_rad){

  //Tiempo de muestreo en ms
  const float Ts=.017;
  //Matriz de covarianza del ruido
  float Q[2][2]={{0.3,0},{0,0.3}};
    //Varianza R
  float R=40;
     //Matriz A de la forma A=[1,-Ts;0,1]
  float A[2][2]={{1,-Ts},{0,1}};
    //Matriz B=[Ts;0]
  float B[2][1]  ={{Ts},{0}};
 //********************Etapa de predicción
         if (w==0){
         gyro_ant=gyro_rad;
```

```
w=1;
}
else{
//x(:,k)=A*x(:,k-1)+B*gyro_ant
x[0][1]=x[0][0]+A[0][1]*x[1][0]+B[0][0]*gyro_ant;
x[1][1]=A[1][1]*x[1][0];

Pn[0][0]=P[0][0]+A[0][1]*P[1][0]+(P[0][1]+A[0][1]*P[1][1])
        *A[0][1]+Q[0][0];
Pn[0][1]=P[0][1]+A[0][1]*P[1][1];
Pn[1][0]=Pn[0][1];
Pn[1][1]=P[1][1]+Q[1][1];

P[0][0]=Pn[0][0];
P[0][1]=Pn[0][1];
P[1][0]=Pn[1][0];
P[1][1]=Pn[1][1];
//******************Etapa de corrección***********
// Actualización de la ganancia de Kalman
K[0][0]=P[0][0]/(P[0][0]+R);
K[1][0]=P[1][0]/(P[0][0]+R);

//Actualización del estado
x[0][0]=x[0][1]+K[0][0]*(acel_rad-x[0][1]);
x[1][0]=x[1][1]+K[1][0]*(acel_rad-x[0][1]);

//Actualización de la covarianza del error de estimación
Pn[0][0]=P[0][0]-K[0][0]*P[0][0];
Pn[0][1]=P[0][1]-K[0][0]*P[0][1];
Pn[1][0]=P[1][0]-K[1][0]*P[0][0];
Pn[1][1]=P[1][1]-K[1][0]*P[0][1];

P[0][0]=Pn[0][0];
P[0][1]=Pn[0][1];
```

```
            P[1][0]=Pn[1][0];
            P[1][1]=Pn[1][1];
            }
            //regresar el valor del acelerometro
            Acel_K=x[0][0];
            gyro_ant=gyro_rad;
            Acel_K=Acel_K;
            return(Acel_K);
}
```

C.2.5. Función del controlador PID discreto, ControlPID.h

```
const float Ts=0.017; //periodo de muestreo
float acel,gyro;      //variables
float angulo;
//ganancias del controlador
//float Kp=3000,Ti=10,Td=100;
float Kp=6000,Ti=.5,Td=2000;
//float Kp=5000,Ti=100000,Td=200;
//Constante N, salida y variables del error
float N=1,e=0,e_ant=0,e_ant_ant=0;
float y,y_ant=0,Uc=0,U=0;
float Prop,I,I_ant=0,D,D_ant=0;
//variables auxiliares
float a,b,c;
int Ucontrol;
int Ud,Ui;
//función para calculo de ganancias del PID
void control_init(){
    a=(Kp*Ts)/Ti;
    b=Td/(Td+N*Ts);
    c=(Kp*Td*N)/(Td+N*Ts);
}
//Función para avance del motor iquierdo
void forwardizq(){
```

```
            //PWMCON1bits.PEN1H=0;
            PWMCON1bits.PEN3L=0;
            //BLI1=1;
            BLI2=1;
            //PWMCON1bits.PEN1L=1;
            PWMCON1bits.PEN3H=1;
}
//Función para avance del motor derecho
void forwardder(){
            PWMCON1bits.PEN1H=0;
            //PWMCON1bits.PEN3L=0;
            BLI1=1;
            //BLI2=1;
            PWMCON1bits.PEN1L=1;
            //PWMCON1bits.PEN3H=1;
}
//Función para reversa del motor iquierdo
void reverseizq(){
            //PWMCON1bits.PEN1L=0;
            PWMCON1bits.PEN3H=0;
            //ALI1=1;
            ALI2=1;
            //PWMCON1bits.PEN1H=1;
            PWMCON1bits.PEN3L=1;
}
//Función para reversa del motor derecha
void reverseder(){
            PWMCON1bits.PEN1L=0;
            //PWMCON1bits.PEN3H=0;
            ALI1=1;
            //ALI2=1;
            PWMCON1bits.PEN1H=1;
            //PWMCON1bits.PEN3L=1;
}
```

```
//función del controlador
void controlador(){//float angulo, float vel1, float vel2){
    float a,b,c,Uc=0;
    int der1,izq1;
    //lectura de joysticks
    der1=(der-498)*3;
    izq1=(509-izq)*3;
    //calculo de ganancias del PID
    a=(Kp*Ts)/Ti;
    b=Td/(Td+N*Ts);
    c=(Kp*Td*N)/(Td+N*Ts);
    //Calculo del error
    e=Uc-angulo;
    y=angulo;
    //Cálculo de P(k)=K(b*Uc(k)-y(k))//
        Prop=Kp*e;
    //Cálculo de I(k)=I(k-1)+(K*Ts)/(Ti)*e(k)
      if(U<6000 && U>-6000)
            I=I_ant+a*e;
      else
          I=I_ant;
    //Cálculo de D(k)=(Td/(Td+N*Ts))*D(k-1)-((K*Td*N)/(Td+N*Ts))*(y(k)-y(k-
        D=b*D_ant-c*(y-y_ant);
    //Cálculo de acción de control
        U=Prop+I+D;
    //delimitación de la acción de control
        if(U>6000)
            U=6000;
        else{
            if(U<-6000)
                U=-6000;
        }
    //compensación por zona muerta
        if(U>0)
```

```
            U=U+80;
        else
            U=U-80;

        Ucontrol=U;
        Ucontrol=U;
      //Carga para cada motor
        Ud=Ucontrol+izq1;
        Ui=Ucontrol+der1-20;

        if(Ud>0){
            reverseder();
            PDC1=6000-Ud;
            //PDC3=6000-Ucontrol+20-der1;
        }
        else{
            forwardder();
            PDC1=6000-(-Ud);
        }
        if(Ui>0){
            reverseizq();
            //PDC1=6000-(-Ucontrol);
            //PDC3=6000-(-Ucontrol)+20;
            PDC3=6000-Ui;
        }
        else{
            forwardizq();
            PDC3=6000-(-Ui);
        }

    //Actualización de variables
        I_ant=I;
        D_ant=D;
        y_ant=y;
```

```
}
```

C.2.6. Función del controlador PD difuso, Control.h

```
const float Ts=0.017;
float acel,gyro;
float angulo;
//float Kp=3000,Ti=10,Td=100;
float Le=0.2;     //radianes———     0.2//.17
float Lde=1;          //1//.7
float de=0;           //cambio del error
float H=250;          //H=250//150
float DuN=0;          //incremento de u negativo
float DuZ1=0;         //incremento de u zero1
float DuZ2=0;         //incremento de u zero2
float DuP=0;          //incremento de u positivo
float e=0,e1=0,Kd=28,Du=0;     //Kd=28//17
float ue=0,ude=0;

float Uc=0,U=0;       //variable de salida
float a,b,c;          //parametros
int Ucontrol=0;       //carga para PWM
int Ud=0,Ui=0;        //Carga llanta derecha e izq.
//función para llanta izq. hacia el frente
void forwardizq(){
             //PWMCON1bits.PEN1H=0;
             PWMCON1bits.PEN3L=0;
             //BLI1=1;
             BLI2=1;
             //PWMCON1bits.PEN1L=1;
             PWMCON1bits.PEN3H=1;

}
//función para llanta der. hacia el frente
void forwardder(){
```

```
                PWMCON1bits.PEN1H=0;
                //PWMCON1bits.PEN3L=0;
                BLI1=1;
                //BLI2=1;
                PWMCON1bits.PEN1L=1;
                //PWMCON1bits.PEN3H=1;

}
//función reversa llanta izq.
void reverseizq(){
            //PWMCON1bits.PEN1L=0;
            PWMCON1bits.PEN3H=0;
            //ALI1=1;
            ALI2=1;
            //PWMCON1bits.PEN1H=1;
            PWMCON1bits.PEN3L=1;

}
//función reversa llanta der.
void reverseder(){
            PWMCON1bits.PEN1L=0;
            //PWMCON1bits.PEN3H=0;
            ALI1=1;
            //ALI2=1;
            PWMCON1bits.PEN1H=1;
            //PWMCON1bits.PEN3L=1;

}

//función min para dos elementos
float min(float uno, float dos){
    float minimo=uno;
    if(minimo>dos)
        minimo=dos;
```

```
    return(minimo);
}
//función mínimo para tres elementos
float min3(float uno, float dos){
   float minimo=1;
   if(minimo>uno)
       minimo=uno;
   if(minimo>dos)
       minimo=dos;
   return(minimo);
 }
//función máximo comparado con 0
float max (float uno){
    float max=0;
    if(uno>0)
        max=uno;
    return(max);
}
//función máximo para dos elementos
float max2(float uno, float dos){
    float max;
    if(uno>dos)
        max=uno;
    else
        max=dos;
    return (max);
}
//función de membresia trapesoidal
float trapezoidal(float x,float a,float b,float c, float d){
    float a1,b1,mini,res;
    a1=(x-a)/(b-a);
    b1=(d-x)/(d-c);
    mini=min3(a1,b1);
    res=max(mini);
```

```
    return(res);
}
//función del controlador
void controlador(){//float angulo, float vel1, float vel2){
    float Uc=0,MaxDuZ;
    int der1,izq1;
    der1=(der-498)*4;
    izq1=(509-izq)*4;
//*******Funcion de control modificar aquí************
//***************************************************
//**************        Entrada variable angulo
//*************             Salida variable U
    e=angulo;           //calculo del error
    de=e-e1;             //calculo de cambio del error
    if(e>-50 && e<Le){
        ue=trapezoidal(e,-50,-45,-Le,Le);
                //se calcula la pertenncia del error en negativo
        if(de>=-20 && de<Lde){
            ude=trapezoidal(de,-20,-15,-Lde,Lde);
                //pertenencia a negativo de el cambio de error
            DuP=min(ue,ude);
        }
        if(de>=-Lde && de<=20){
            ude=trapezoidal(de,-Lde,Lde,15,20);
            DuZ1=min(ue,ude);
        }
    }
    if(e>=-Le && e<50){
        ue=trapezoidal(e,-Le,Le,45,50);
        if(de>=-20 && de<=Lde){
            ude=trapezoidal(de,-20,-15,-Lde,Lde);
            DuZ2=min(ue,ude);
    }
        if(de>=-Lde && de<=20){
```

```
            ude=trapezoidal(de,-Lde,Lde,15,20);
            DuN=min(ue,ude);
    }
    }
    MaxDuZ=max2(DuZ1,DuZ2);
    Du=(-H*DuN+0.0*MaxDuZ+H*DuP)/(DuN+MaxDuZ+DuP);
    U=Kd*Du;
    e1=e;
    DuN=0;
    DuZ1=0;
    DuZ2=0;
    DuP=0;

//**         finaliza función de control ************
//*************************************************
    //se acota la salida
        if(U>6000)
            U=6000;
        else{
            if(U<-6000)
                U=-6000;
        }
    //compensación por zona muerta
        if(U>0)
            U=U+80;
        else
            U=U-80;
    //cambio a entero
        Ucontrol=U;
    //carga mas cambio en joystick
        Ud=Ucontrol+izq1;
        Ui=Ucontrol+der1-20;
    //condición derecha reversa
        if(Ud>0){
```

```
            reverseder();
            PDC1=6000-Ud;
            //PDC3=6000-Ucontrol+20-der1;
        }
        else{
            forwardder();
            PDC1=6000-(-Ud);
        }
    //condición izquierda reversa
        if(Ui>0){
            reverseizq();
            //PDC1=6000-(-Ucontrol);
            //PDC3=6000-(-Ucontrol)+20;
            PDC3=6000-Ui;
        }
        else{  //izq. hacia adelante
            forwardizq();
            PDC3=6000-(-Ui);
        }
  }
```

C.2.7. Función del controlador super-Twisting

```
#include <math.h> //se incluye la libreria
const float Ts=0.017;   //periodo de muestreo
float acel,gyro;
float angulo;
//float K1=2500,K2=100,signo,absoluto;
//float K1=2500,K2=50,signo,absoluto;
//float K1=2500,K2=10,signo,absoluto;
float K1=2500,K2=100,signo,absoluto;     //ganancias del controlador

float v,v_ant=0,Uc=0,U=0;
int Ucontrol;
int Ud,Ui;
```

```
void forwardizq(){
            //PWMCON1bits.PEN1H=0;
            PWMCON1bits.PEN3L=0;
            //BLI1=1;
            BLI2=1;
            //PWMCON1bits.PEN1L=1;
            PWMCON1bits.PEN3H=1;

}
void forwardder(){
            PWMCON1bits.PEN1H=0;
            //PWMCON1bits.PEN3L=0;
            BLI1=1;
            //BLI2=1;
            PWMCON1bits.PEN1L=1;
            //PWMCON1bits.PEN3H=1;

}

void reverseizq(){
            //PWMCON1bits.PEN1L=0;
            PWMCON1bits.PEN3H=0;
            //ALI1=1;
            ALI2=1;
            //PWMCON1bits.PEN1H=1;
            PWMCON1bits.PEN3L=1;

}
void reverseder(){
            PWMCON1bits.PEN1L=0;
            //PWMCON1bits.PEN3H=0;
            ALI1=1;
            //ALI2=1;
```

```
            PWMCON1bits.PEN1H=1;
            //PWMCON1bits.PEN3L=1;

}

void controlador(){

    int der1,izq1;


    der1=(der-498)*3;
    izq1=(509-izq)*3;

//*****************Programación del esquema de control****************
    //entradas=angulo
    //salida=U
 //*****************************************************************
//Algoritmo super-Twisting
    if(angulo>0)
        signo=1;
    if(angulo<0)
        signo=-1;
    if(angulo==0)
        signo=0;

    if (angulo>=0)
        absoluto=angulo;
    else
        absoluto=-angulo;

    //calculo de la integral de dv/dx
    if(U<6000)
        v=Ts*(-K2*signo)+v_ant;
    else
```

```
        v=v_ant;
    //calculo de la entrada de control u=-K1|sigma|^1/2*sign(sigma)+v
    U=-K1*sqrt(absoluto)*signo+v;

//*****************************************************************************



        if(U>6000)
            U=6000;
        else{
            if(U<-6000)
                U=-6000;
        }

        if(U>0)
            U=U+80;
        else
            U=U-80;

        Ucontrol=U;

        Ud=Ucontrol+izq1;
        Ui=Ucontrol+der1-20;

        if(Ud>0){
            reverseder();
            PDC1=6000-Ud;
            //PDC3=6000-Ucontrol+20-der1;
        }
        else{
            forwardder();
            PDC1=6000-(-Ud);
        }
```

```
        if(Ui>0){
            reverseizq();
            //PDC1=6000-(-Ucontrol);
            //PDC3=6000-(-Ucontrol)+20;
            PDC3=6000-Ui;
        }
        else{
            forwardizq();
            PDC3=6000-(-Ui);
        }

         v_ant=v;
  }
```

C.2.8. Programa main del microcontrolador principal, main.c

```
#include "xc.h"
#include "Config.h"
#include <MATH.h>
#include "Sensores.h"
#include "Kalman.h"
#include "PWM.h"
#include "Control.h"
#include "libpic30.h"
#include "stdio.h"
int count;
// función para obtener el ángulo de acelerómetro
float anguloacel();
//función para obtener vel. ang. de giro
float gyrovel();
//declaración función interrupcion RDA
void __attribute__((interrupt,no_auto_psv)) _U1RXInterrupt(void);
//declaración función interrupción Timer1
void __attribute__((interrupt,no_auto_psv)) _T1Interrupt(void);
//declaración función lectura adc
```

```
unsigned int read_analog_channel(int channel);
//función de interrupción timer 1
void __attribute__((interrupt,no_auto_psv)) _T1Interrupt(void)
{
    IFS0bits.T1IF=0;
    count=1;
   // if(U1STAbits.OERR) U1STAbits.OERR=0;//
}
//inicia el main
int main(void) {
    //configuración canales analogicos
    ADPCFG=0;
    //configuración I/O digitales
    TRISB=0b00001100;
    TRISE=0;
    TRISF=2;

    PORTE=0;
    CNEN1=0;
    //configuración ADC***************
    ADPCFG=0b1111111111110011;
    ADCON1=0x0000;
    ADCHS=0;
    //ADCSSLbits.CSSL3=2;
    ADCON3=0b0000001110010011;
    ADCON2=0x0000;
    ADCSSL=12;
    ADCON1bits.ADON=1;
    //*****************************
    __delay_ms(2000);
    PWM_INIT();
    //configuración UART
    IFS0bits.U1RXIF=0;
    IEC0bits.U1RXIE=1;
```

```
//U1BRG = 48;                          //38400 Baud @ 30 MIPS
//U1BRG = 23;                          //38400 Baud @ 15 MIPS
//U1BRG = 96;                          //9600 Baud @ 15 MIPS
//U1BRG = 49;                          //19200 Baud @ 15 MIPS
U1BRG=97;                              //19200 Baud @ 30 MIPS
U1MODEbits.UARTEN=1;
//***** Configuración Timer1
PR1 = 1992;             // Set the Timer 1 period (max 65535)
TMR1 = 0;                // Reset Timer 1 counter
IEC0bits.T1IE = 1;      // Enable Timer 1 interrupt
T1CONbits.TCKPS = 3;   // Prescaler (0=1:1, 1=1:8, 2=1:64, 3=1:256)
T1CONbits.TON = 1;      // Turn on Timer 1
//estado inical de motores apagado
DIS1=1;
DIS2=1;
//*******************************
while(1){

    while(!count);
    LATBbits.LATB0=1;
    count=0;
    putchar(255);
    __delay_us(10);
    con++;
    while(con);
    acel=anguloacel();
    gyro=gyrovel();
    angulo=k_filter(acel,gyro)+1.1*3.14159265358979323846/180;
    der=read_analog_channel(2);
    izq=read_analog_channel(3);
    if(angulo<40)
    controlador();
    angulo=angulo*180/3.14159265358979323846;
    acel=acel*180/3.14159265358979323846;
```

```
            printf("%f?%f",angulo,U);
            LATBbits.LATB0=0;
        }
        return 0;
}

unsigned int read_analog_channel(int channel)
{
    ADCHS = channel;                // Select the requested channel
    ADCON1bits.SAMP = 1;            // Start sampling
    __delay_us(8);                  // 1us delay @ 30 MIPS
    ADCON1bits.SAMP = 0;            // Start Converting
    while (!ADCON1bits.DONE); // Should take 12 * Tad = 3.2us
    return ADCBUF0;
}
```

Apéndice D

Hojas de Datos

En esta sección se muestran las hojas de datos de los componentes utilizados para la construcción del prototipo. Estos componentes se muestran en la Tabla D.1

Tabla D.1: Lista de hojas de datos

Matricula	Dispositivo	Liga de Internet
AV02-1046EN	Encoder Óptico	http://www.avagotech.com/docs/AV02-1046EN
DSPIC30F4012	Microcontrolador	http://ww1.microchip.com/downloads/en/DeviceDoc/70135G.pdf
H11L1D	Optoacoplador	http://pdf.datasheetcatalog.com/datasheet/QT/H11L1D.pdf
L3GD20	Giroscopio	http://www.st.com/st-web-ui/static/active/en/resource/technical/document/datasheet/DM00036465.pdf
LSM303DLHC	Acelerómetro	http://www.st.com/st-web-ui/static/active/en/resource/technical/document/datasheet/DM00027543.pdf
OKI-78SR	Regulador 5 volts	http://www.murata-ps.com/data/power/oki-78sr.pdf
OSMC	Driver de motor	http://www.robotpower.com/downloads/osmc3-22sch-clean.pdf
PC817	Optoacoplador	http://www.classiccmp.org/rtellason/chipdata/pc817.pdf
PIC16F1939	Microcontrolador	http://ww1.microchip.com/downloads/en/DeviceDoc/40001574C.pdf
RT12180	Batería	http://ritar.hu/doc/RT12180.pdf

Apéndice E

Publicaciones

Se ha publicado un articulo en la Reunión de otoño de Potencia, Electrónica y Computación, ROPEC 2013, Internacional, Titulado *"Real-Time control to hold upright equilibrium of a two-wheeled autobalancing vehicle"*, pp. 1-5, INSPEC 14026226.

Apéndice F

Material Adicional en Formato Digital

Se adjunta un DVD, el cual contiene el siguiente material.

- Copia digital de este documento.
- Presentación del Examen de Grado.
- Circuitos esquemáticos y diseño en CAD de las tarjetas electrónicas diseñadas.
- Códigos fuente de las diferentes librerías en formato de proyecto compilable.
- Hojas de datos de los componentes utilizados.
- Fotografías y videos del vehículo en diversas etapas de su construcción

Bibliografía

[Aracil 2005] Aracil J., Gordillo F., "*El Péndulo Invertido: Un Desafío para el Control No Lineal*", Revista Iberoamericana de Automática e Informática Industrial, Vol. 2 Num. 2, Abril 2005, pp. 8-19.

[Astrom 1997] Astrom K. y Wittenmark B. "*Computer Controlled Systems Theory and Design*", Prentice-Hall 3^a Ed. 1997.

[Berger 2002] Berger E. J. , *"Friction modeling for dynamic system simulation"*, American Society of Mechanical Engineers Appl Mech Rev vol 55, no 6, pp. 535-577

[Blackwell 2009] Balancing Scooter, [Web en línea].http://www.tlb.org/scooter.html. [Consulta: 01-8-2012].

[Bonales 2012] Bonales J., "*Construcción, Modelado y Control de un vehículo eléctrico sobre dos ruedas tipo péndulo invertido"*, Tesis de maestría, Universidad Michoacana de San Nicolás de Hidalgo, 2013.

[Chen 1999] Chen Chi-Tsong. "*Linear System Theory and Design*" Oxford University Press 3^a Ed, New York. 1999. ISBN 0-19-511777-8.

[Chen 2011] Chen J. Y., Liu C. H., Chen C. C., Lin K. C., "*Adaptive Fuzzy Control of Two-Wheeled Balancing Vehicle*", 2011 International Conference on Instrumentation, Measurement, Computer, Communication and Control, pp. 341-344, 2011.

[Chenxi 2013] Sun C, Lu T, and Yuan K., "*Balance Control of Two-wheeled Self-balancing Robot Based on Linear Quadratic Regulator and Neural Network*", 2013 Fourth International Conference on Intelligent Control and Information Processing (ICICIP) June 9 – 11, 2013, Beijing, China.

[Ching-Chih 2010] Ching-Chih T., Hsu-Chih H., Shui-Chun L., "*Adavtive Neural Network Control of a Self-Balancing Two-Wheeled Scooter*", IEEE Transactions on Industrial Electronics, Vol. 57, no. 4, Abril 2010

[Deal 2008] The Two Wheel Deal, [Web en línea].https://engineering.purdue.edu/ece477/Webs/S08-Grp12/documents.html. [Consulta: 10-10-2013].

[DIY 2008] The segway DIY,.[Web en línea]. http://web.mit.edu/first/segway/#TOC [Consulta: 10/10/2013]

[OSPV 2011] Elektor OSPV, [Web en línea].https://www.elektor.es/proyectos/elektor-ospv1.1812954.lynkx. [Consulta: 10-10-2013]

[Freego 2008] Freego., [Web en linea]. http://www.freegochina.com/.[Consulta: 08-10-2013]

[Garcia 2008] García E.,"*Compilador C CCS y simulador PROTEUS para microcontrolador PIC*", Marcombo 1^a Ed. 2008.

[Goher 2010] Goher, K.M.K, Tokhi, M.O, "*Genetic algorithm based modeling and control of a two wheeled vehicle with an extended rod, a Lagrangian based dynamic approach*", Cybernetic Intelligent Systems (CIS), 2010 IEEE 9th International Conference on, pp. 1-6, 2010.

[Grasser 2002] Jia-Sheng H., Mi-Ching T., Feng-Rung H., and Yoichi H., "*Robust Control for 200coaxial two-wheeled electric vehicle*", Journal of Marine Science and Technology, Vol. 19, No- 2, pp. 172-180, 2010.

[Ha 1996] Ha Y., Yuta S., "*Trajectory tracking control for navigation of the inverse pendulum type self-contained mobile robot*", Robot Auton. Syst. 17,1996, pp. 65-80.

[Hamza 2011] Hamza M. , Zaka-ur-Rehman, Zahid Q., Tahir F., and Khalid Z., "*Real-time control of an inverted pendulum: A comparative study*", IEEE Computer Society, vol. 41, 2011.

[Honda 2009] Honda U3-X, [Web en línea].http://world.honda.com/U3-X/. [Consulta: 10-10-2013].

[Jiménez 2011] Jiménez A., Ramirez S., Barriga O., Torres V., Argumedo P., "*Construcción y Control PID de un Péndulo Invertido en Dos Ruedas*", Reunión de Otoño

de Potencia, Electrónica y Computación (ROPEC) 2011 Internacional, ISBN: 978-607-95476-3-9.

[Grasser 2002] Grasser, F., D'Arrigo, A., Colombi, S. y Rufer, A. C, "*JOE: a mobile, Inverted Pendulum*" , IEEE Transactions on Industrial Electronics, Vol. 49, No 1, 2002.

[Kim 2005] Kim Y., Kim S. H., and Kwak Y. K., "*Dynamic analysis of a nonholonomic two-wheeled inverted pendulum robot*", Journal of Intelligent and Robotics Systems, vol. 44, pp. 25-46, 2005.

[Kim 2006] Kim Y., Kim S. H., Kwak Y. K, "*Dynamic Analysis of a Nonholonomic Two-Wheeled Inverted Pendulum Robot*", Journal of Intelligent and Robotics Systems, Springer 2006 vol. 44, p. 25-46.

[LEGO M 2010] Sánchez H., Aguirre I., Patete A., "*Construcción y Control de un Péndulo Invertido utilizando la Plataforma Lego MINDSTORMS NXT*", 4° Congreso Iberoamericano de estudiantes de Ingeniería Eléctrica (IV CIBELEC 2010), ISBN: 978-980-7185-1.

[Maureira 2010] Maureira R. A., "*Mejoramiento del diseño de control y electrónico de un Vehículo Autobalanceado*" , Tesis de licenciatura, Universidad de Chile, 2010.

[Moreno 2009] Moreno L. F., "*Diseño e implementacion de vehiculo auto-balanceado sobre dos ruedas*", Tesis de licenciatura, Universidad de Chile, 2009.

[Nasir 2010] Nasir A. N. K., Raja Ismail R. M. T., Ahmad M. A., "*Performance Comparison between Sliding Mode Control (SMC) and PD-PID Controllers for a Nonlinear Inverted Pendulum System*", World Academy of Science, Engineering and Technology, 46, 2010.

[Nawawi 2006] Nawawi S. W., Ahmad M. N., Osman J.H.S., "*Control of two-wheeled Inverted Pendulum Mobil Robot Using Ful Order sliding Mode Control*", Proceedings of International Conference on Man-Machine Systems, Malasya, 2006

[Nawawi 2008] Nawawi S. W., Ahmad M. N. and Osman J. H. S., "*Real-Time Control of a Two-wheeled Inverted Pendulum Mobile Robot*", World Axademy of Science, Engieneering and Thechnology, No. 15, pp. 214-220, 2008.

[Olsson 1997] Olsson H., Astrom K.J., Canudas de Wit C., Gafvert M., Lischinsky P., "*Friction Models and Friction Compensation*" European Jornal of Control, 1998, pp. 176-195, Vol. 4.

[OSMC 2014] Robot Power, "*Open sorce motor controller, Product information*" [Web en linea]. http://www.robotpower.com/products/osmc_info.html [Consulta: 27-01-2014].

[Perruquetti 2002] Perruquetti W., Barbot J-P., "*Sliding Mode Control In Engineering*", CRC Press, 432 páginas, 1^a Ed.. 2002.

[Pham 2001] Chen G, Pham T. T., "*Introduction to Fuzzy Sets, Fuzzy Logic, and Fuzzy Control Systems*", CRC Press, Florida, $1^{\text{`}a}$ Ed. 2009

[Rao 2004] Rao S., Brandtstädter H., Buss M., Utkin V., "*Sliding Mode Control in Mechanical Systems with Electric Actuators*"

[RynoMotors 2013] Rynomotors,.[Web en línea]. http://rynomotors.com/. [Consulta: 08/10/2013]

[Segway 2001] Segway Inc., [Web en línea].http://www.segway.com/. [Consulta: 08-10-2013]

[Shui 2009] Shui-Chun L., Ching-Chih T. and Hsu-Chih H., "*Nonlinear adaptive sliding-mode control design for two-wheeled human transportation vehicle*", SMC 09 Proceedings of the 2009 IEEE international conference on System, Man and Cybernetics, pp. 1965-1970, 2009.

[Sun 2009] Sun H., Zhou H., Li X., Wei Y., and Li X.,"*Design of Two-Wheel Self-Balanced Electric Vehicle based on MEMS*", Proceedings of the 2009 4th IEEE International Conference on Nano/Micro Engenieered and Molecular Systems January 5-8, 2009, Shenzhen, China.

[Winglet 2008] Toyota Winglet P.T., [Web en línea].http://www.toyota.co.jp/en/news/08/0801_1.html. [Consulta: 01-8-2012].

[Wenjun 2013] Wenjun L. V., Yu Z. y Ping Z., "*Speed and Orientation Control of a Two-Coaxial-Wheeled Iverted Pendulum*", Proceedings of the 32nd Chinese Control Conference, Julio 2013, pp.334-337, Xi'an China.

[Wheelie 2009] Krohne C, "*Elektor Wheelie - The electronics behind a rather special kind of vehicle*", elektor electronics & microelectronics. (Julio-Agosto 2009), pp. 66.

[Yamafuji 1989] Yamafuji K., Kawamura T., "*Postural Control of a monoaxial bicycle*", J. Robot. Soc. of Japan 7(4), 1989, pp. 74-79.

[Yepez 2007] Yépez E, Yépez M. Y., "*Mecánica Analítica*", Las prensas de ciencias 1^a Ed. 2007, pp. 171-220