ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE MECÁNICA

CARRERA INGENIERÍA DE MANTENIMIENTO

“IMPLEMENTACIÓN DE UN ROBOT SEGUIDOR DE LÍNEA PARA LA COMPARACIÓN DE UN CONTROLADOR DE LÓGICADIFUSA Y UN CONTROLADOR PID”

Trabajo de titulación:

Tipo: Propuesta tecnológica

Presentado para optar al grado académico de:

INGENIERO DE MANTENIMIENTO

AUTORES: ALEX FABRICIO CAIZA MULLO

ANGEL ADRIÁN MORALES RONQUILLO

DIRECTOR: Ing. GABRIEL VINICIO MOREANO SÁNCHEZ

Riobamba –

Ecuador2021

Nosotros, Alex Fabricio Caiza Mullo y Angel Adrián Morales Ronquillo declaramos que el presente trabajo de titulación es de nuestra autoría y los resultados del mismo son auténticos. Los textos en el documento que provienen de otras fuentes están debidamente citados y referenciados.

Como autores asumimos la responsabilidad legal y académica de los contenidos de este trabajo de titulación; El patrimonio intelectual pertenece a la Escuela Superior Politécnica de Chimborazo.

.

Riobamba, 04 de marzo de 2021

Alex Fabricio Caiza Mullo
C.I. 050377564-5

Angel Adrián Morales Ronquillo
C.I. 010550283-5

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE MECÁNICA

CARRERA DE INGENIERÍA DE MANTENIMIENTO

El Tribunal del trabajo de titulación certifica que: El trabajo de titulación: Tipo: Propuesta Tecnológica, **IMPLEMENTACIÓN DE UN ROBOT SEGUIDOR DE LÍNEA PARA LA COMPARACIÓN DE UN CONTROLADOR DE LÓGICA DIFUSA Y UN CONTROLADOR PID**, realizado por los señores: **ALEX FABRICIO CAIZA MULLO Y ANGEL ADRIÁN MORALES RONQUILLO**, ha sido minuciosamente revisado por los Miembros del Tribunal del trabajo de titulación, el mismo que cumple con los requisitos científicos, técnicos, legales, en tal virtud el Tribunal Autoriza su presentación.

	FIRMA	**FECHA**
Dr. José Antonio Granizo **PRESIDENTE DEL TRIBUNAL**	JOSE ANTONIO GRANIZO Firmado digitalmente por JOSE ANTONIO GRANIZO Fecha: 2021.04.09 16:32:29 -05'00'	2021-03-04
Ing. Gabriel Vinicio Moreano Sánchez **DIRECTOR DEL TRABAJO DE TITULACION**	Firmado digitalmente por GABRIEL VINICIO MOREANO SANCHEZ Fecha: 2021.04.08 22:45:54 -05'00'	2021-03-04
Ing. Julio Eduardo Cajamarca Villa **MIEMBRO DE TRIBUNAL**	JULIO EDUARDO CAJAMARCA VILLA Firmado digitalmente por JULIO EDUARDO CAJAMARCA VILLA Fecha: 2021.04.08 21:40:35 -05'00'	2021-03-04

DEDICATORIA

Este trabajo va dedicado a mi madre Ana, quien con su amor, paciencia y esfuerzo me ha permitido llegar a cumplir hoy un sueño más, por inculcar en mí el ejemplo de esfuerzo y valentía, de no temer a las adversidades.

A mis hermanos por su cariño y apoyo incondicional durante todo este proceso, a toda mi familia por sus consejos, sabiduría, apoyo y esfuerzo que hacen de mí una mejor persona y de una u otra forma me acompañan en mis sueños y metas.

Alex

Este trabajo está dedicado a mis padres Angel y Ana, quienes con su esfuerzo, paciencia y amor me brindan el apoyo necesario e incondicional para hoy alcanzar el objetivo propuesto años atrás; inculcándome valores de perseverancia y sacrificio antes las adversidades de la vida estudiantil y cotidiana.

A mis hijos Josué Nicolás y Angel Benjamín pilares fundamentales en mi vida para seguir adelante, quienes conjuntamente con mis hermanas y sobrinos fueron los motores para no rendirme en este caminar estudiantil demostrándome su amor a través del apoyo moral y psicológico.

Angel

AGRADECIMIENTOS

Quiero agradecer a dios por permitirme concluir una etapa más en mi vida, a mi familia por estar siempre presentes ante toda dificultad.

Mi profundo agradecimiento a la Escuela Superior Politécnica de Chimborazo, a la escuela de Ingeniería de Mantenimiento, a sus docentes quienes con sus conocimientos hacen que crezca día a día como profesional, por su paciencia y dedicación.

Finalmente, expreso mis agradecimientos, al director del presente trabajo de titulación Ing. Gabriel Moreano y a mi asesor Ing. Julio Cajamarca quienes con su conocimiento, paciencia, orientación y experiencia han hecho posible el desarrollo de este trabajo, a mis compañeros por compartir momentos gratos en esta travesía.

Alex

En primer lugar, quiero agradecer a Dios por llenarme de fuerzas, salud y vida para culminar un ciclo más en mi vida, a mi familia, amigos y compañeros por sus palabras de aliento y consejos en cada momento de alegría y tristeza. Mi sincero agradecimiento a la Escuela Superior Politécnica de Chimborazo por acogerme en su seno, en especial a la escuela de Ingeniería de Mantenimiento, a su personal docente quienes compartieron sus conocimientos y experiencia para formarme como profesional. Finalmente, extiendo mis agradecimientos, al director del presente trabajo de integración curricular Ing. Gabriel Moreano y a mi asesor Ing. Julio Cajamarca por su predisposición, paciencia y colaboración para el desarrollo de este trabajo, demostrando interés al momento de impartirnos sus conocimientos y experiencias.

Angel

TABLA DE CONTENIDO

Pág.

ÍNDICE DE TABLAS

Pág.

ÍNDICE DE FIGURAS

Pág.

ÍNDICE DE GRÁFICOS

Pág.

ÍNDICE DE ANEXOS

RESUMEN

El objetivo de este trabajo fue implementar un robot seguidor de línea para la comparación de un controlador PID y un controlador de lógica difusa, en su ensamblaje se utilizó elementos mecánicos y electrónicos. Su funcionamiento consiste en la programación y sintonización en el editor de código Arduino IDE, la sintonización se realizó con el método de prueba y error para un desplazamiento preciso en la pista y se obtiene valores de los parámetros: error, posición, y velocidad de control. Para el controlador PID el método de prueba y error consiste en dar valores aleatorios a las constantes proporcional y derivada, en cambio el controlador Fuzzy utiliza este método con etiquetas lingüísticas para establecer intervalos a los conjuntos difusos pertenecientes a las variables de entrada y salida, funciones de membresía y reglas de inferencia para su funcionamiento. Para la comparación se realizó pruebas de tiempo de recorrido y gráficas lineales para los parámetros, teniendo mejor resultado en tiempo, posición y error en el controlador PID y la velocidad de control en el controlador Fuzzy. De acuerdo a los resultados se identifica que el controlador PID tiene mayor estabilidad de error en el recorrido y el controlador Fuzzy envía señales de control menos bruscas a los motores deteriora y desgasta las baterías y escobillas de los motores en menor proporción al producir cambios de voltaje menos bruscos. Al realizar cada prueba de funcionamiento se debe limpiar la pista, las ruedas y poseer un ambiente adecuado de temperatura e iluminación.

Palabras clave: <ROBOT SEGUIDOR DE LÍNEA>, <PROPORCIONAL INTEGRAL Y DERIVATIVO (PID)>, <LOGICA DIFUSA> <ROBOTICA MOVIL>, <ARDUINO IDE (SOFTWARE)>, <CONTROLADOR DUAL DE MOTORES (DRIVER)>, <ENSAMBLAJE Y CONEXIÓN>.

ABSTRACT

The objective established for this work was to implement a line follower robot for the comparison between a PID controller and a fuzzy logic controller, using mechanical and electronic elements for its assembly. Its operation involves programming and tuning in the Arduino IDE code editor. The second one was completed with the trial and error method for having a precise displacement in the track and getting the parameters values: error, position, and speed of the control, while for PID controller, this method gives random values for the proportional and derivative constants. On the other hand, the Fuzzy controller uses this method with linguistic labels in order to establish intervals to the input and output variables, membership functions, and inference rules that conforms the fuzzy sets, so this work correctly. For comparison, journey time and linear graphs tests were carried out for the parameters, obtaining better results in terms of time, position and error of the PID controller and control speed on the fuzzy controller. According to the results, it is identified that the PID controller has higher error stability in its journey and Fuzzy controller sends less abrupt control signals to the motors, also deteriorates and wears down in minor proportion the batteries and brushes when the voltage changes are less abrupt. When performing each test run, the track and the wheels must be cleaned, also is necessary having a suitable environment for temperature and lighting.

Keywords: <LINE FOLLOWER ROBOT>, < PROPORTIONAL INTEGRAL AND DERIVATIVE (PID)>, <FUZZY LOGIC> <MOBILE ROBOTICS>, <ARDUINO IDE (SOFTWARE)>, <DUAL MOTOR CONTROLLER (DRIVER)>, <ASSEMBLY AND CONNECTION>.

CAPÍTULO I

1. INTRODUCCIÓN

1.1. Antecedentes

En el entorno industrial, se utilizan robots seguidores de línea, que tienen diseños, reprogramables y dispuestos a realizar tareas, de acuerdo con los requerimientos de un proceso para controlar distintas variables.

Un ejemplo de este tipo de robots es el llamado "Vehículo Guiado Automatizado" (Automatic Guided Vehicule o en siglas AGV), este robot además de seguir una línea, suele añadir sensores de distancia para evitar chocar con gente u otros vehículos, radares, visión artificial, etc. Las aplicaciones de estos robots suelen estar relacionadas con el transporte de mercancías dentro de la misma empresa o almacenaje. Se suele crear una red de robots que mueven la producción de un sitio hacia un almacén, o hacia otro punto de la cadena de fabricación (Ortiz, 2016).

En la cita (Carrillo, Cardona, Arvizo, & Rodríguez, 2017) se define que "al construir un robot seguidor de línea y ser programado con diferentes algoritmos de control como: un controlador PID, un controlador proporcional y un controlador ON-OFF", los mismos que permitieron recabar la información sobre el tiempo que duró el robot en recorrer la pista, se determinó que el algoritmo PID presenta un mejor rendimiento que los otros dos algoritmos de control en el robot en base a tiempo y estabilidad al recorrer el circuito, debido a que es un algoritmo que incorpora más elementos, proporcionando más información para la generación de la señal de control.

En (Villalobos, 2014) se establece como ventaja que la lógica difusa frente a otras técnicas de control ya que no es preciso conocer la función característica del sistema. Por medio de este trabajo se pudo demostrar que el modelamiento de sistemas físicos por medio de lógica difusa presenta resultados satisfactorios al momento de simular su comportamiento con entradas determinadas, lo cual es una gran herramienta en casos en los cuales no se tiene conocimiento de la curva característica del sistema. El control por medio de lógica difusa tuvo un mejor desempeño que la técnica de control ON-OFF utilizada en plataformas móviles, obteniendo una trayectoria con menos oscilaciones, lo cual disminuye el tiempo del recorrido, y transitando la trayectoria por completo.

1.2. Planteamiento del problema

La presente propuesta tecnológica tiene como finalidad implementar un robot seguidor de línea para realizar la comparación de un controlador de lógica difusa y un controlador PID, de tal manera determinar qué controlador tiene mayor rendimiento en cuanto a tiempo de recorrido de

trayectoria, estabilidad, etc, o forma de ejecutar su función para que pueda ser implementado en un proceso industrial para aumentar la productividad de un determinado proceso.

1.3. Justificación

El presente trabajo de integración curricular tiene como fin comparar las respuestas de funcionamiento entre los controladores de lógica difusa y PID, ya que estos controladores se implementan actualmente en varios procesos industriales; los controladores antes mencionados serán implementados en un seguidor de línea, el cual se construirá para la sintonización de cada uno de los controladores, el estudio de estos controladores permitirá conocer: la programación de los dos controladores utilizando Arduino, previo al ensamblaje del prototipo, además de identificar las características que distingan a cada controlador, para lo cual se desarrollará el diseño, funcionamiento e implementación de estos dos tipos de controladores, los mismos que serán programados en un robot móvil como es el seguidor de línea.

1.4. Objetivos

1.4.1. Objetivo general

Implementar un robot seguidor de línea para la comparación entre un controlador de lógica difusa y un controlador PID.

1.4.2. Objetivos específicos

- Sintetizar el conocimiento existente sobre los robots seguidores de línea y los tipos de controladores que se emplean en los mismos.
- Implementar un robot seguidor de línea a manera de plataforma experimental
- Sintonizar un controlador PID y un controlador borroso en el prototipo y realizar pruebas de funcionamiento de cada uno.
- Comparar las ventajas y desventajas de los dos controladores al implementarlos en el seguidor de línea, en cuanto a costo computacional, tiempos de respuesta, operación reconociendo su funcionamiento y posibles aplicaciones en el control de procesos industriales.

CAPÍTULO II

2. MARCO TEÓRICO

2.1. Robótica móvil

La robótica móvil en la actualidad ha creado múltiples aplicaciones para ser utilizadas en diferentes áreas de trabajo. En esta rama la evolución de los robots llegaría a reemplazar a las personas en ciertas actividades hasta ser un elemento más en el hogar, en las empresas, y en la sociedad(Ramírez & Reyes, 2015).

Hoy en día es común hablar de los robots que funcionan con movimiento autónomo, debido al creciente interés que se ha generado en esta área de la robótica. El movimiento autónomo se ha implementado a través de técnicas que permiten el control por medio de un software y un hardware moderno. La autonomía en un robot se consigue con la implementación de patas, ruedas o rieles; la elección de uno de ellos dependerá del campo y el trabajo que se va a desenvolver(Aguilera, Bautista, & Iruegas, 2007).

Un robot móvil (Aguilera et al., 2007, p. 1) lo define como "un vehículo de propulsión autónoma y movimiento (re) programable por medio del control automático para realizar una tarea específica"

Los robots móviles se clasifican en guiados y no guiados. El vehículo guiado está restringido a un conjunto de trayectorias predefinidas en su área de trabajo, en cambio; los vehículos no-guiados no están restringidos a una trayectoria predefinida(Aguilera et al., 2007).

2.2. Robot seguidor de línea

Un robot seguidor de línea es considerado dentro de la robótica como una plataforma móvil como se muestra en la figura 1-2, el mismo que se mueve en una pista de color negro sobre un fondo blanco o viceversa mediante el reconocimiento de uno de estos dos colores.

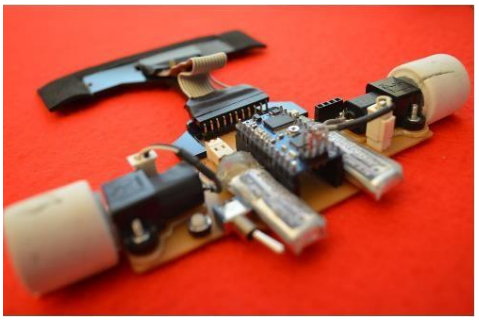


Figura 1-2. Robot seguidor de línea

Fuente: (Tdrobotica, 2019)

Cuando se habla de robots seguidores de línea, se están citando robots cuyo objetivo consiste en recorrer con la mayor precisión posible una línea marcada en el suelo, sin embargo, la precisión depende de varios factores, entre ellos, de la distinción de colores que realice el robot del ambiente a través de sus sensores(Soto & Gómez, 2013).

Los robots seguidores de línea son máquinas móviles capaces de detectar y seguir una línea, la cual se encuentra ubicada en el suelo de una superficie. Normalmente, el camino donde el robot se desplaza debe obtener un enorme contraste entre los dos colores, en la figura 2-2 se muestra un ejemplo de la pista mencionada que se utiliza para competencias de un robot seguidor de línea. En algunos u otros casos se pueden llegar a utilizar planos negros con líneas blancas, para que de igual manera se pueda detectar mediante la diferencia del contraste la trayectoria que el robot debe seguir (Carrillo et al., 2017).

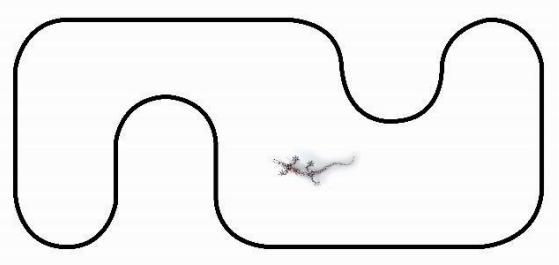


Figura 2-2. Pista de seguidor de línea

Fuente: (UEPRIM, 2016)

2.3. Funcionamiento

Los seguidores de línea capturan la posición en la que se encuentran mediante sensores ópticos, para poder seguir la línea correctamente se utilizan diferentes tipos de algoritmos de control, que logran estabilizar al robot sobre la línea para recorrer la trayectoria completa en el menor tiempo posible (Carrillo et al., 2017).

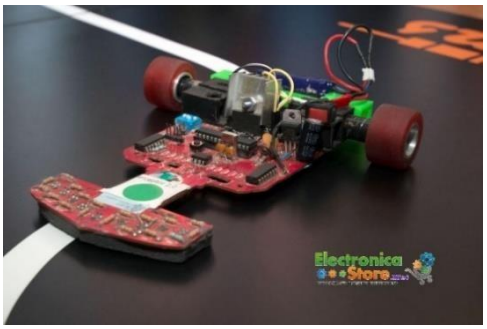


Figura 3-2. Seguidor de línea

Fuente: (ElectronicaStore, 2018)

El algoritmo de control para el funcionamiento necesita de un microcontrolador o circuitos integrados que permitan la movilidad en el robot seguidor de línea; que conjuntamente con los sensores ópticos permiten que él mismo siga su trayectoria de manera correcta reconociendo el

color para el cual ha sido programado, en la figura 3-2 se observa un robot seguidor de línea con sus componentes mecánicos y electrónicos, dentro una pista de color blanco sobre fondo negro.

2.4. Tipos de robots seguidores de línea

Existen diferentes tipos y modelos de robots, estos son construidos de diferente diseño y características, particularmente son creados para distintos fines y objetivos, ya sea para el desarrollo de competencias de robots de estilo de investigación de estudiantes y aplicaciones industriales, entre los diferentes tipos de podemos nombrar los siguientes: Seguidor de línea Turing, Seguidor de línea velocista y seguidor de línea industrial como el robot Weasel (Abrajan, 2020).

2.4.1. Seguidor de línea Turing

Este robot cuenta con todos sus elementos para poder armarlo, muy prácticos para estudiantes y maestros interesados en el desarrollo de conocimientos y destrezas que estén dentro de la rama de la robótica, en la figura 4-2 se muestra un robot de este tipo, el cual está constituido por los siguientes elementos: motores con caja de reducción, barra de sensores , ruedas del estilo Pololu, batería, chasis PCB elaborado de forma exclusiva, puente H Dual, Arduino Nano (ElectronicaStore, 2018,p1).

Figura 4-2. Robot seguidor de línea Turing
Fuente: (ElectronicaStore, 2018)

2.4.2. Seguidor de línea velocista

El robot velocista tiene como característica principal el uso de motores de alta velocidad y ruedas que tengan adherencia al suelo y permite obtener mayor velocidad. Según, (Tdrobotica, 2014) "Son indispensables este tipo de ruedas ya que el robot es un seguidor de línea, al alcanzar velocidades de más de 2 metros por segundo necesita un agarre óptimo en cada curva."

El chasis debe ser elaborado de PCB para que sea ligero y pueda soportar su propio peso, y llegue a imprimir la velocidad deseada (Tdrobotica, 2014).

2.4.3. Seguidor de línea Robot Weasel

Este robot es diseñado para uso en la industria, para el transporte de cargas de un lugar a bodegas de almacenamiento de productos, (SCHAEFER, 2016) afirma que: El robot WEASEL® no requiere de costosos sensores ni sistemas de control complejos. Circula y se desplaza a lo largo de un carril óptico que puede instalarse de forma rápida y sencilla, y también adaptarse a los cambios de recorrido. El WEASEL alcanza velocidades de hasta 1 metro por segundo (1m/s) en trayectos con hasta un 20% de pendiente. Gracias a su flexibilidad y eficiencia, WEASEL le garantiza el transporte seguro de contenedores, cajas y artículos de prenda colgada, en áreas de producción como se muestra en la figura 5-2, ya sea para abastecer la propia producción, como para conectar las áreas de producción y distribución.

Figura 5-2. Robot Weasel

Fuente: (SCHAEFER, 2016)

2.5. Tipos de robots de competencia

Se nombra robots seguidores de línea aquellos que se construyen para las competencias de robótica y mecatrónica entre lo más utilizados, pero existen también otros tipos que son utilizados en diferentes ramas de trabajo dándole distintas aplicaciones, entre estos ejemplares tenemos:

- Robots de sumo para las categorías pesado y liviano.
- Robot seguidor de línea modalidad velocista.
- Robot seguidor de línea modalidad destreza.
- Robot seguidor de línea con controlador PID.
- Robot seguidor de línea con controlador Fuzzy.

2.6. Aplicaciones

Los robots seguidores de línea son muy utilizados en competencias de robótica, dependiendo la categoría o las especificaciones se construyen diferentes tipos. También se puede encontrar en las industrias para el transporte de cargas, mercancías.

2.6.1. Industriales

En los entornos industriales se utilizan robots seguidores de línea, pero versiones mucho más optimizadas y preparadas, el ejemplo que se muestra en la figura 6-2 es un vehículo guiado automatizado (Automatic Guided Vehicule o en siglas AGV). Sigue una línea y tienen sensores de distancia para evitar chocar con gente u otros vehículos, radares, visión artificial, etc.

Las aplicaciones de estos robots suelen estar relacionadas con el transporte de mercancías dentro de la misma empresa o almacenaje. Se suele crear una red de robots que mueven la producción de un sitio hacia un almacén, o hacia otro punto de la cadena de fabricación (Ortiz, 2016).

Figura 6-2. Robot Weasel

Fuente: (SCHAEFER, 2016)

2.7. Arquitectura

Un robot seguidor de línea puede tener diferentes tipos de arquitecturas dependiendo las especificaciones para las que vaya a ser utilizado, el tipo de algoritmo, la programación o elementos que se tenga en cuenta para su construcción.

Las partes principales que conforman a un robot seguidor de línea son el chasis o carcaza, sensores, microcontrolador, motores, llantas y componentes electrónicos, buscando tener como resultado un modelo que sea ligero, compacto y de bajo consumo de energía (Carrillo et al., 2017).

2.7.1. Microcontrolador

Es el cerebro del robot, ya que lo controla y le brinda direccionalidad, a través de la lectura de datos recibidos a través los sensores (Ocampo, Maya, Rossette, Martinez, & Barrios, 2017).

Los microcontroladores están concebidos fundamentalmente para ser utilizados en aplicaciones puntuales, es decir, donde este elemento debe realizar un pequeño número de tareas, al menor costo posible. Este elemento ejecuta un programa de almacenamiento permanentemente en su memoria, el cual trabaja con algunos datos almacenados temporalmente e interactúa con el exterior a través de las líneas de entrada y salida que dispone (Lindao & Quilambaqui, 2014).

Dentro de estos microcontroladores encontramos el Arduino, que por su arquitectura y código libre permite cierta versatilidad y facilidad de programación para un robot seguidor de línea, permitiendo la lectura de entradas y salidas como interruptores, pulsadores y leds respectivamente. Además, puede controlar los motores, tomar decisiones o leer los datos emitidos por los sensores utilizados, en la figura 7-2 que se muestra a continuación se observa algunos ejemplos de microcontroladores de la marca A-Star.

Figura 7-2. Controladores programables A-Star

Fuente: (Pololu, 2020)

2.7.2. Motores

Se encargan de proporcionar movilidad y velocidad requerida para que el robot termine un recorrido específico durante un tiempo establecido. Por ejemplo (Ocampo et al., 2017) emplea dos motores marca Pololu, los cuales tiene una velocidad de 6000 RPM y un torque de 0.5 kg.cm.

Dependiendo del tamaño, el peso, la precisión del motor, entre otros factores, éstos pueden ser de varias clases: motores de corriente continua, motores paso a paso o servomotores (UNAM, 2016).

Figura 8-2. Motorreductor de metal micro

Fuente: (Pololu, 2020)

Estos motores constan también de 2 soportes que permiten fijar el micromotor al chasis o base del robot con mayor seguridad, en la figura 8-2 se ilustra un motorreductor muy utilizado en un robot seguidor de línea.

2.7.3. Controlador de motores (drivers)

El robot requiere de un circuito capaz de controlar y suministrar la potencia necesaria a los motores del robot, esa es la función principal de los Drivers, se encargan de recibir las señales de control provenientes del microcontrolador, entregando a su salida una señal equivalente de mayor potencia capaz de controlar al motor (Carrillo et al., 2017).

Estos incluyen una función de protección contra la inversión de voltaje, bajo voltaje, sobre corriente, y el exceso de temperatura, en la figura 9-2 se observa un controlador que es de gran solución para la alimentación de motores pequeños, de bajo voltaje (Tapiero, 2019).

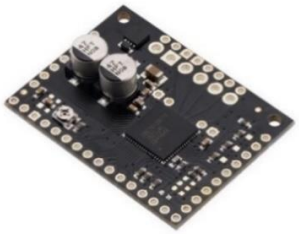


Figura 9-2. Portador de control
de motor paso a paso

Fuente: (Pololu, 2020)

2.7.4. Batería

Es la encargada de suministrar la energía al sistema y la circuitería del robot para permitir su funcionamiento, la batería tipo lipo es ideal por su capacidad de descarga, tamaño y peso en relación con su eficiencia que es muy buena; en la figura 10-2 se muestran baterías de diferentes tamaños y voltajes.

Figura 10-2. Baterías

Fuente: (Pololu, 2020)

2.7.5. Sensor de Reflexión de Luz

Es un componente electrónico que permite a un robot seguidor de línea detectar la trayectoria que debe seguir, la misma que está pintada en el piso y que puede ser recta, curva o combinada. Uno de los sensores por reflexión es el QTR-8A que se ilustra en la figura 11-2, consta de 8 emisores de infrarrojos y 8 receptores que captan la línea de manera óptima, dependiendo la cantidad de sensores que contenga un robot seguidor de línea su velocidad de respuesta es más precisa, es decir encuentra la trayectoria con más facilidad.

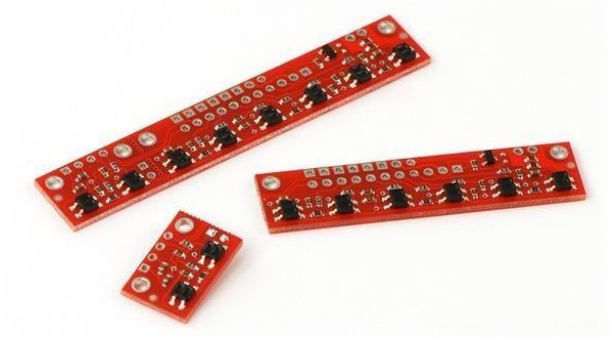


Figura 11-2. Sensores de proximidad

Fuente: (Pololu, 2020)

2.7.6. Ruedas

Son elementos mecánicos circulares que consta de una llanta (parte de goma) y un neumático (parte de aluminio), las cuales a través de los motores eléctricos ejercen movimiento y tracción sobre la superficie en contacto (Ocampo et al., 2017).

Según, (Ortiz, 2016) se establecen ciertas condiciones para elección de ruedas: las ruedas pequeñas proporcionarán una velocidad inicial muy elevada, y ruedas de mayor diámetro garantizarán una velocidad final muy elevada. Ruedas anchas significa que tendrán mucho agarre en curvas, pero ruedas estrechas garantizarán menos superficie de rozamiento, por tanto, mayor velocidad. En la figura 12-2 se observa una rueda que se utiliza con frecuencia para un robot seguidor de línea.

Figura 12-2. Ruedas seguidor de línea velocista aluminio

Fuente: (Tdrobotica, 2019)

En un robot seguidor de línea se considera la utilización de dos llantas posteriores para la tracción y en la parte delantera una rueda omnidireccional que se observa en la figura 13-2, la misma que permite atravesar mayor cantidad de obstáculos porque tiene un movimiento libre de 360°, es conocida también como “rueda loca”.

Figura 13-2. Pololu ball Casters

Fuente: (Pololu, 2020)

2.7.7. Chasis

Es la base del robot, que se puede construir de diferentes formas, materiales o colores dependiendo las especificaciones de uso. Además, es la parte donde se acoplan los elementos y que se transportan a través de la trayectoria.

Este chasis puede ser móvil, modular, de madera, de acero, con niveles, etc. Las características de los materiales y diseños pueden influir en el funcionamiento del prototipo. Un robot seguidor de línea debe contar con una estructura que le facilite el desplazamiento, que lo mantenga en equilibrio en todo momento y le permite cambiar de dirección fácilmente (Ortiz, 2016).

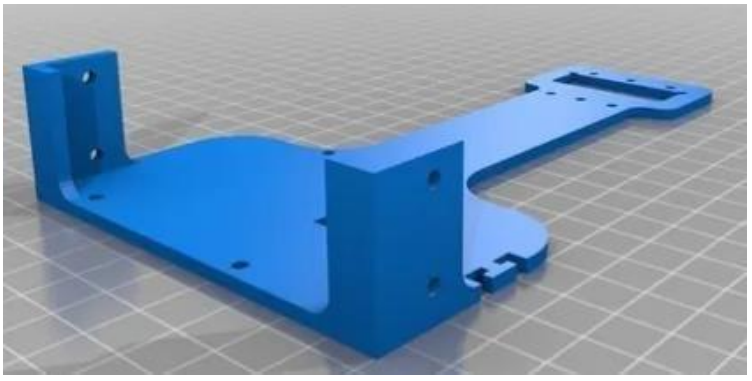


Figura 14-2. Seguidor de línea chasis impreso 3D

Fuente: (Tdrobotica, 2020)

2.8. Controlador PID

Un controlador PID es un mecanismo de control sobre la realimentación de bucle cerrado ampliamente utilizado en sistemas de control industrial. Este controlador calcula el error como la diferencia entre el valor actual del sistema y el valor al que se desea llegar e intenta minimizar el error ajustando la entrada del sistema (Ortiz, 2016).

(Tapiero, 2019) cita que este controlador es ampliamente utilizado en diversos sistemas donde se incluya control automático, es un controlador relativamente robusto que tiene como fundamento calcular la desviación entre valores medidos y un valor deseado, y a partir de estos generar una acción de control utilizando sus tres componentes.

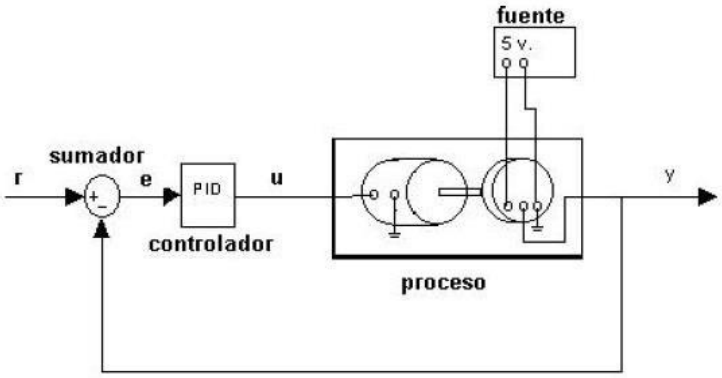


Figura 15-2. Diagrama de bloques del sistema PID en lazo cerrado

Fuente: (Tapiero, 2019)

El controlador PID toma su nombre porque está compuesto por tres parámetros que son: el proporcional (P), el integral (I) y el derivativo (D); cada uno de estos controla diferentes variables que permiten ajustar un sistema a través de un elemento de control. Así, se tiene por ejemplo controlar la velocidad de las ruedas de un robot seguidor de línea.

La parte proporcional P depende del error actual, la integral I depende de la suma de todos los errores pasados, y la derivativa D es la velocidad con que cambia el error (Ortiz, 2016).

En consideración a lo expuesto anteriormente en la figura 15-2, se muestra un diagrama de bloques de un controlador PID en la figura 16-2, el cual nos va a ayudar a una mejor comprensión del funcionamiento.

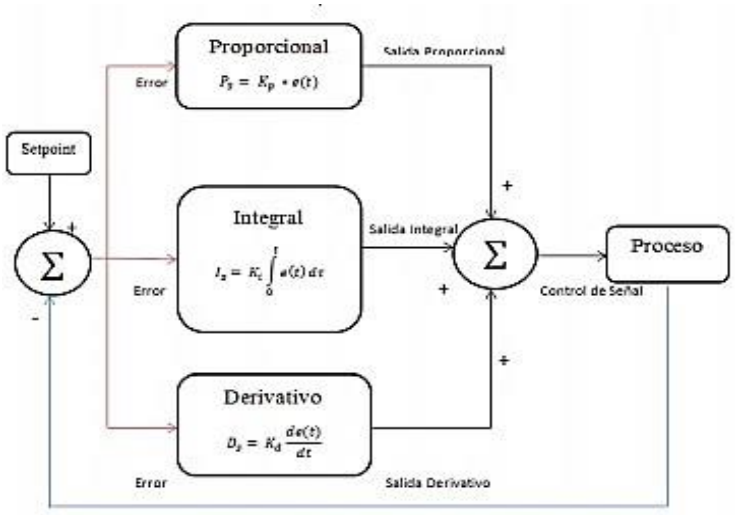


Figura 16-2. Diagrama de bloques control PID

Fuente: (CULCyT,2016)

Ajustando estas constantes K_P, K_I, K_D en el algoritmo de control del PID, el regulador es capaz de proporcionar acciones de control específicas a los requerimientos de un sistema. Los valores para ajustar el controlador se obtienen analizando el sistema que se quiere regular, y los ajustes tienen

que ver con el tiempo de respuesta, los errores que puede tener hasta ajustar la salida y las sobre oscilaciones del sistema hasta llegar al valor deseado (Ortiz, 2016).

En los controladores PID, existen controladores que no utilizan todos los parámetros, debido a que un sistema se estabiliza usando uno de estos parámetros, por tanto, dependiendo de las combinaciones que se realicen existen controladores: proporcionales (P), Proporcionales-Derivativos (PD), Proporcionales-Integrativo, Integrativo, etc.

2.8.1. Parámetros del control PID

2.8.1.1. Proporcional

Este término modifica la salida proporcionalmente con el error actual. La respuesta proporcional se puede ajustar multiplicando el error por una constante, típicamente K_P, conocida como ganancia proporcional, este término crece cuando el error es muy amplio, y se reduce cuando el error es muy pequeño (Ortiz, 2016).

La ganancia proporcional es representada por la siguiente ecuación:

$$P = K_P * e(t) \quad \text{(Ec. 1)}$$

- $e(t)$ error en un instante de tiempo
- K_P constante proporcional
- P ganancia proporcional

2.8.1.2. Integral

El término integral es la suma de todos los errores en cada instante de tiempo, o lo que es lo mismo, la integración de los errores, el error acumulado se multiplica por la ganancia integral K_I, que es la cantidad de acción integral que se debe sumar al control (Ortiz, 2016).

La ecuación de la ganancia integral es:

$$I = K_I * \int_0^t e(t)\, dt \quad \text{(Ec.2)}$$

- $\int_0^t e(t)\, dt$ integral de los errores
- K_I constante integral
- I cantidad de acción integral

2.8.1.3. Derivativo

El término derivativo calcula la variación del error mediante su pendiente en cada instante de tiempo, es la primera derivada con respecto al tiempo. Este error se multiplica con la ganancia derivativa, K_P (Ortiz, 2016).

Este parámetro ayuda a que el controlador se estabilice de manera suave y de esta manera disminuir las oscilaciones producidas por la parte integral, está representada por la fórmula:

$$D = K_D * \frac{d}{dt} e(t) \quad \text{(Ec. 3)}$$

- $\frac{d}{dt} e(t)$ variación del error
- K_D constante derivativa
- D cantidad de acción derivativa

De esta manera la ecuación que incluye a los tres parámetros es:

$$u(t) = K_p * e(t) + K_I * \int_0^t e(t)\, dt + K_D * \frac{d}{dt} e(t) \quad \text{(Ec. 4)}$$

La ecuación también se puede escribir de la siguiente forma:

$$u(t) = K_p * e(t) + \frac{K_P}{T_I} * \int_0^t e(t)\, dt + K_P * T_D * \frac{d}{dt} e(t) \quad \text{(Ec. 5)}$$

Donde T_I, T_D representa el tiempo integral y derivativo respectivamente, de esta manera podemos considerar:

- Si K_P es más grande, la respuesta es más rápida debido a que el error es mayor, pero si es demasiado grande puede presentar inestabilidad y oscilaciones.
- Si K_I es grande elimina los errores estacionarios con mayor rapidez, pero a su vez trae consecuencias de mayor oscilación.
- Si K_D aumenta demasiado, la sobreoscilación disminuye, pero a su vez aumenta el tiempo de respuesta, por tanto, se amplifica el ruido en el cálculo diferencial y el aumento del error puede provocar un sistema inestable.

2.8.2. Métodos de sintonía para un controlador PID

Dentro de los procesos, plantas o sistemas industriales si se tiene un modelo matemático es posible obtener los parámetros del controlador que cumplan las especificaciones del estado estacionario de un sistema de lazo cerrado. Por otro lado, si una planta no cuenta con un modelo matemático y es difícil de obtenerlo para encontrar los parámetros del controlador PID, e incluso no se puede utilizar un método analítico para el diseño del controlador, se recurre a procedimientos experimentales para su sintonía.

El proceso de encontrar o seleccionar los parámetros del controlador que satisfaga las necesidades y especificaciones de una planta se conoce como sintonía del controlador, entre estos podemos tener:

- Reglas de Ziegler-Nichols

- Método de respuesta de frecuencia
- Método de optimización computacional
- Método de asignación de ceros para mejorar las características de respuesta

2.8.2.1. Reglas de Ziegler- Nichols

Según (Ogata et al., 2010), Ziegler y Nichols propusieron reglas para determinar los valores de la ganancia proporcional K_P, del tiempo integral T_I y del tiempo derivativo T_D, basándose en las características de respuesta transitoria de una planta dada. Tal determinación de los parámetros de los controladores PID o sintonía de controladores PID la pueden realizar los ingenieros mediante experimentos sobre la planta y existen dos métodos para ser realizado:

1. En el primer método, la respuesta de la planta a una entrada escalón unitario se obtiene de manera experimental, tal como se muestra en la figura 17-2.

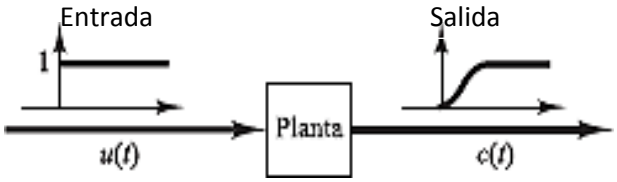


Figura 17-2. Respuesta a un escalón unitario de una planta

Fuente: (OGATA, 2010)

En este método la curva característica es de tipo S en caso de no tener integradores ni polos dominantes, esta curva se caracteriza por el tiempo de retardo L la constante de tiempo T, las mismas que se determinan trazando una tangente en el punto de inflexión de la curva y determinando las intersecciones de esta tangente con el eje del tiempo y con la línea $c(t) = k$, como se ve a continuación en la figura 18-2 (Ogata et al., 2010).

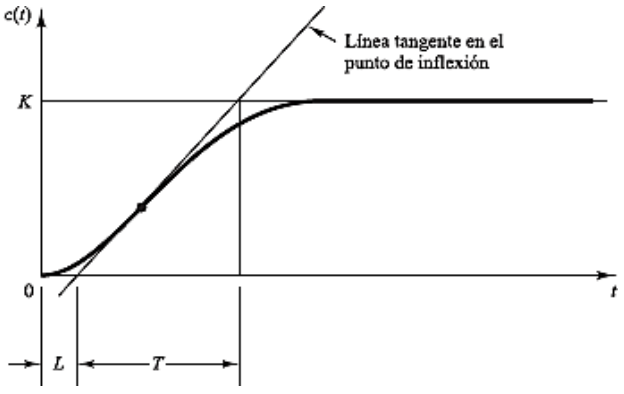


Figura 18-2. Curva de respuesta en forma de S

Fuente: (OGATA, 2010)

De esta forma estos dos autores sugieren establecer valores de K_P, T_I, T_D de acuerdo con lo que se indica en la tabla 1-2.

Por tanto, según la tabla y las ecuaciones, el controlador PID tiene un polo en el origen y un cero doble en $S = -1/L$.

Tabla 1-2 Regla de sintonía de Ziegler-Nichols (primer método)

Tipo de controlador	K_P	T_I	T_D
P	T/L	Infinito	0
PI	0,9T/L	L/0,3	0
PID	1,2T/L	2L	0.5L

Fuente: (OGATA, 2010)

2. En este método se fija T_I a infinito y T_D a 0, es decir de esta manera utilizamos solo el parámetro proporcional, se incrementa K_P desde 0 hasta un valor crítico K_{cr}, en donde la salida presenta oscilaciones sostenidas para cualquier valor que pueda tomar K_P, entonces Ziegler-Nichols sugirieron que se establecieran los valores K_P, T_I, T_D de acuerdo con la fórmula que se muestra en la tabla 2-2 (Ogata et al., 2010). Por lo tanto, el controlador PID tiene un polo en el origen y un cero doble en $S = -4/K_{cr}$.

Tabla 2-2: Regla de sintonía de Ziegler-Nichols (segundo método)

Tipo de controlador	K_P	T_I	T_D
P	0,5 K_{cr}	Infinito	0
PI	0,45 K_{cr}	P_{cr}/1,2	0
PID	0,6 K_{cr}	0,5 P_{cr}	0,125 P_{cr}

Fuente: (OGATA, 2010)

2.8.3. Control PID en un robot seguidor de línea

En la configuración de un robot seguidor línea un controlador PID en primer lugar calcula su posición actual, posteriormente calcula su error, y de esta manera da una orden a los micromotores de realizar un giro dependiendo de su error; en consecuencia, si su error es alto dará un giro grande y si su error es bajo dará un giro pequeño**.**

La magnitud del giro tomado será proporcional al error. Después de esto, si el error no disminuye o lo hace, pero lentamente, el controlador aumenta la magnitud del giro conforme pase el tiempo hasta que el robot se centre sobre la línea. En el proceso de centrarse, el robot puede pasar su posición objetivo y posicionarse al otro lado de la línea, en este punto el proceso anterior es efectuado nuevamente. Entonces el robot se mantendrá oscilando a lo largo de la línea con el fin de centrarse en la misma (Vera & Alejandro, 2016).

A continuación, para tener una mejor comprensión y realizar una comparación se muestra en la figura 20-2, la forma de movilizarse de un robot sin un controlador PID.

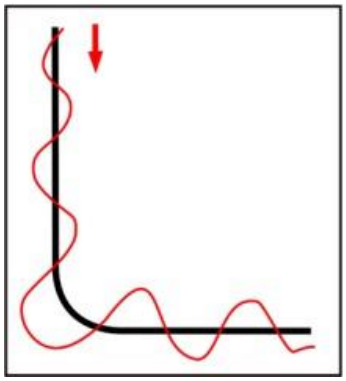


Figura 19-2. Patrón de seguimiento
de línea sin controlador PID

Fuente: (Palmisano, 2016)

Observando la figura 19-2, notamos que un robot seguidor de línea se moviliza mucho por fuera de línea utilizando tiempo y batería valiosa, entonces un seguidor de línea con un controlador PID seguirá su trayectoria como en la figura 20-2.

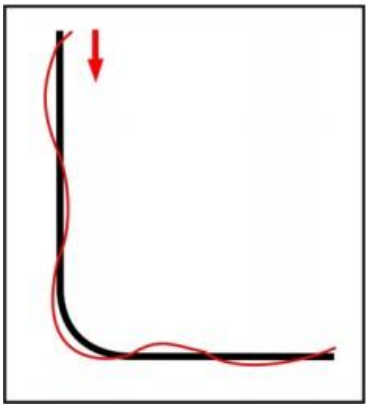


Figura 20-2. Patrón de seguimiento
con controlador PID

Fuente: (OGATA, 2010)

2.9. Controlador Fuzzy

2.9.1. Control con lógica difusa

(Meza, 2003) establece que el concepto de la lógica difusa está asociado con la manera en que las personas perciben el medio, por ejemplo, la velocidad con que se mueve un objeto, la temperatura de un lugar, etc. Las mismas que se dan cotidianamente de manera ambigua y depende de quién percibe el efecto físico o químico, para poder describir este fenómeno. Los conjuntos difusos definen estas ambigüedades y son una extensión de la teoría clásica de conjuntos, donde un elemento pertenece o no a un conjunto.

Dentro de estos sistemas se encuentra dos áreas, el modelado o identificación y el control propiamente dicho, y se basa en una idea simple que consiste en determinar de manera lógica que se debe hacer para lograr los objetivos de control de mejor manera.

Entre las principales aplicaciones de la lógica difusa tenemos el diseño de sistemas de control que a partir de entradas en un sistema se generan salidas, para poder actuar sobre determinados mecanismos. Un ejemplo sería un sistema de control para regular la velocidad de un ventilador, en función de la temperatura de un horno; en este caso tenemos como entrada el valor de temperatura en grados centígrados (°C) y como salida la velocidad del ventilador en RPM.

Los sistemas difusos son sistemas basados en el conocimiento o en reglas, tienen como característica principal reglas que consiste en un conjunto de proposiciones SI-ENTONCES (IF-THEN rules). Este conjunto de reglas es una declaración en la cual ciertas palabras son caracterizadas por funciones de transferencia continuas, por ejemplo, se describe una regla de esta forma: "SI la velocidad de un carro es alta ENTONCES se aplica menor fuerza en el acelerador", en la figura 21-2 se ilustra la variable de entrada y en la figura 22-2 la variable de salida.

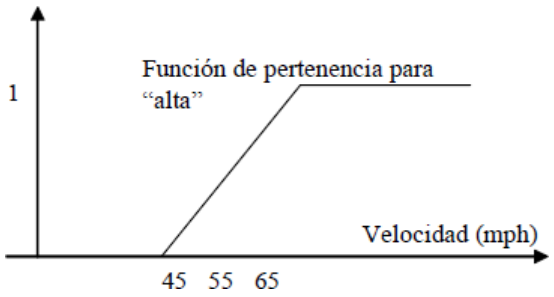


Figura 21-2. Función de pertenencia para la variable alta

Fuente: (Sistemas con lógica difusa, 2009)

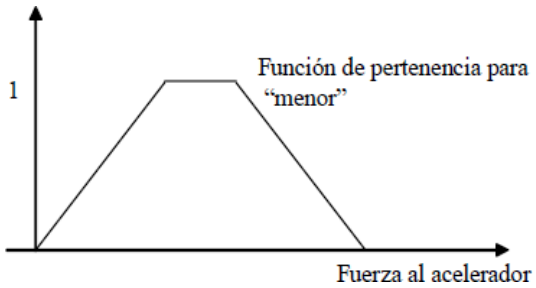


Figura 22-2. Función de pertenencia para la variable menor

Fuente: (Sistema con lógica difusa, 2010)

El punto de partida para construir un sistema difuso es una colección de reglas difusas sustentadas en el conocimiento humano de expertos, o bien reglas con base en la función de distribución del sistema de referencia.

Los sistemas difusos pueden ser usados en controladores de lazo abierto o controladores lazo cerrado como se muestra en la figura 23-2 y la figura 24-2 respectivamente.

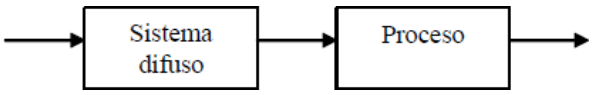


Figura 23-2. Sistema difuso como controlador en lazo abierto

Fuente: (Sistema con lógica difusa, 2009)

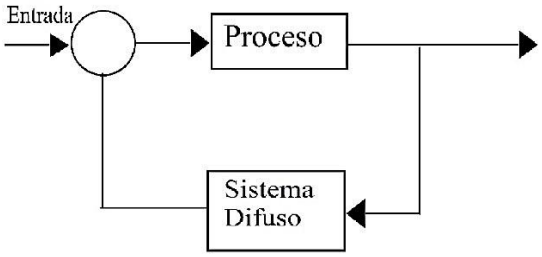


Figura 24-2. Sistema difuso como controlador en lazo cerrado

Fuente: (Sistema con lógica difusa, 2009)

2.9.2. Controladores difusos

Según (Meza, 2003), para el diseño e implementación de un controlador debemos tener en cuenta algunos aspectos, entre estas tenemos las siguientes leyes que ayudan en la etapa de diseño:

- Primera ley: "el mejor sistema de control es aquel más simple que hará el trabajo".
- Segunda ley: "se debe entender el proceso antes de poder controlarlo".
- Tercera ley: "el ejemplo típico de nivel de líquido siempre debe ser controlado".

Los sistemas expertos de control difuso están basados en reglas y son conocidos como controladores difusos o FLC (Fuzzy Logic Controllers), son la aplicación más extendida de la lógica difusa, en la figura 25-2, podemos observar un controlador de este tipo.

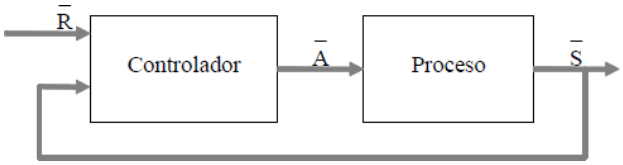


Figura 25-2. Control directo de un proceso o sistema

Fuente: (Sistema con lógica difusa, 2009)

(Santos, 2005), establece que el diseño de un controlador basado en lógica borrosa es una descripción lingüística de la estrategia de control utilizado por un humano en el control manual de un proceso, las reglas que se describen son estrategias de control más que el proceso en sí, ya que a partir de esas reglas se van a generar acciones de control.

En cambio (Alzate, López, & Restrepo, 2007) describen que los controladores difusos son la aplicación más importante de la teoría borrosa, los mismos que trabajan de una forma bastante diferente a los controladores convencionales, ya que el conocimiento experto se usa en vez de ecuaciones diferenciales para describir el comportamiento del sistema, dicho conocimiento se expresa de una manera muy natural, empleando variables lingüísticas que son descritas mediante conjunto difusos.

Un regulador borroso se compone internamente de un conjunto de reglas lingüísticas de control, que tienen como antecedentes los valores posibles de las variables de entrada, y que concluyen la acción de control a efectuar en términos también lingüísticos (Santos, 2005).

Según (García, Medel, Sánchez, & Tequianez, 2005), para poder controlar un proceso o sistema se emplea un módulo controlador, que recibe como entrada una o más variables de control denominadas referencia o señal deseada (R) como se observa en la figura 26-2, y una o más variables de salida del propio proceso (S), produciendo de esta forma una o más variables conocidas como actuadores (A). Normalmente el objetivo de este controlador es mantener R=S, en la figura 26-2, se puede ilustrar un controlador basado en un sistema difuso.

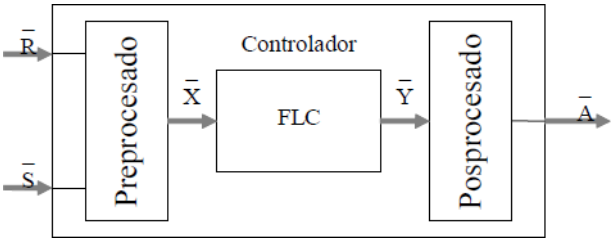


Figura 26-2. Estructura de un controlador FLC

Fuente: (Sistema con lógica difusa, 2009)

2.9.2.1. Reglas borrosas

Las reglas de los controladores borrosos combinan uno o más conjuntos borrosos de entrada llamados premisas (antecedentes) y se asocian un conjunto borroso de salida llamado consecuente (consecuencia), involucrando a conjuntos difusos, lógica e inferencia difusa.

La base de reglas se representa por tablas, es clara al momento de tener dos variables de entrada y una de salida, a medida que aumenta las variables crece la tabla y será más difícil su edición, dentro de estos tipos de reglas existe una gran variedad, pero (Meza, 2003) describe dos grupos que generalmente se emplean, que son:

1. Reglas difusas de Mamdani: estas reglas están basadas en la siguiente expresión:

$$\text{IF } \mathbf{x_1} \text{ is } \mathbf{A} \text{ AND } \mathbf{x_2} \text{ is } \mathbf{B} \text{ AND } \mathbf{x_3} \text{ is } \mathbf{C} \text{ THEN } \mathbf{u_1} \text{ is } \mathbf{D}, \mathbf{u_2} \text{ is } \mathbf{E} \qquad \text{(Ec. 6)}$$

Donde $\mathbf{x_1}$, $\mathbf{x_2}$ y $\mathbf{x_3}$ son las variables de entrada (por ejemplo: error, derivada del error, segunda derivada del error), **A**, **B** y **C** son funciones de membresía o pertenencia de entrada (alto, medio, bajo), $\mathbf{u_1}$ y $\mathbf{u_2}$ son las acciones de control (apertura de válvulas), **D** y **E** son las funciones de pertenencia de la salida, en la cual la sentencia IF es el antecedente y la sentencia THEN el consecuente. Estas reglas presentan ventajas como:

- Son intuitivas.
- Tienen una amplia aceptación.
- Están bien adaptadas a la incorporación de conocimiento y experiencia.

2. Reglas difusas de Takagi-Sugeno: estas reglas tienen la siguiente expresión:

$$\text{IF } \mathbf{x_1} \text{ is } \mathbf{A} \text{ AND } \mathbf{x_2} \text{ is } \mathbf{B} \text{ AND } \mathbf{x_3} \text{ is } \mathbf{C} \text{ THEN } \mathbf{u_1 = f\,(x_1, x_2, x_3)}, \mathbf{u_2 = g\,(x_1, x_2, x_3)} \qquad \text{(Ec. 7)}$$

Estas funciones f () y g () se pueden emplear como funciones no lineales, pero la elección de tal función puede ser muy compleja, por tal razón de manera general se utilizan funciones lineales. De igual manera presenta ciertas ventajas:

- Es computacionalmente eficiente.
- Trabaja bien con técnicas lineales.
- Trabaja bien con técnicas de optimización y control adaptable.
- Tiene garantizada una superficie de control continua.
- Está bien adaptado al análisis matemático.

2.9.2.2. Inferencia borrosa

Las reglas difusas representan el conocimiento y la estrategia de control, pero cuando se asigna información específica a las variables de entrada en el antecedente, es necesario la inferencia difusa para calcular el resultado de las variables de salida del consecuente. Existe una gran cantidad de métodos de inferencia difusa, pero existen cuatro que presenta mejores resultados en el campo de control, estos son:

- Inferencia de Mamdani por mínimos cuadrados.
- Inferencia del producto de Larsen.
- Inferencia del producto drástico.
- Inferencia del producto limitado.

Cuando el conjunto difuso de salida del consecuente es singleton, todos los métodos de inferencia tienen el mismo resultado (Meza, 2003).

2.9.3. Etapas de un sistema de control difuso

(Santos, 2005), ilustra en la figura 27-2, la configuración básica de un controlador borroso directo, en la cual puede haber algunas variantes, pero se las puede organizar en cuatro componentes principales, que son las siguientes:

- Tratamiento de la información de entrada: consiste en la determinación de las variables de entrada como su medida, posterior escalado y conversión de información concreta en borrosa (borrosificación).
- Base de conocimiento: almacena la caracterización de los términos lingüísticos utilizados en las reglas y la base de reglas que especifica los objetivos de control.
- Método de inferencia: evalúa la base de reglas en función del valor de las variables de entrada al controlador y determina el valor de salida.
- Tratamiento de la información de salida: convierte el valor de salida en un dato no difuso y el método empleado para obtener el valor determinista de salida del controlador borroso se conoce como defusificación.

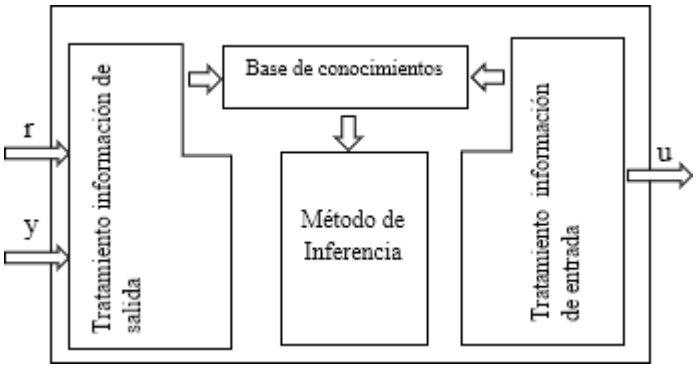


Figura 27-2. Elementos de un controlador borroso

Fuente: (Contribución a los métodos de sintonía, 2005)

Tanto el primer como el cuarto componente se pueden agrupar en conversores de información concreta-borrosa, en cualquier sentido; pero cómo se emplean algoritmos diferentes para estos, se exponen por separado.

Por ejemplo, (García et al., 2005), las define como etapas de un sistema de control difuso y estas son las siguientes:

- Interferencia de entrada: toma valores de la planta y los convierte en valores lingüísticos.

- Inferencia de salida: en esta etapa se convierten los datos lingüísticos en datos numéricos, mediante una ponderación y normalización de las sentencias lógicas antecedentes.
- Base de conocimiento: incluye los parámetros necesarios para las dos anteriores, los cuales pueden ser de naturaleza heurística u optimizados mediante alguna técnica particular.

Entonces se puede observar que el controlador difuso es como un sistema experto que toma decisiones y que opera en un sistema de lazo cerrado, compara la salida y'(t) del sistema difuso con la señal deseada y(t) y entonces decide cual es la entrada a la planta u(t) y asegura la realización de los objetivos (García et al., 2005).

2.9.4. Diseño de un controlador fuzzy

Para el diseño de un controlador se debe empezar por conocer el comportamiento del proceso a controlar, luego hay que definir qué tipo de regla se utiliza, lo más recomendable es Mamdani ya que es intuitivo. Normalmente, para problemas de seguimiento o regulación con referencia distinta de cero, las variables que se controlan son el error y su derivada.

En la dimensión del controlador se usa el rango de las variables de entrada y de salida, para la elección de las funciones de membresía no es tan crítico como su rango de representación, con esto se representa el conocimiento de las variables. Para la edición de la base de reglas se representa la estrategia de control, en cuanto a los métodos de inferencia, agregado, defusificación y las definiciones en las operaciones entre conjuntos son un procedimiento de prueba y error evaluando el desempeño del controlador, los procedimientos de diseño y análisis se los puede hacer de manera iterativa para lograr un desempeño más aceptable (Meza, 2003).

2.9.5. Aplicaciones

Los sistemas difusos están siendo empleados en una gran variedad de campos, entre estos tenemos: procesamiento de señales, control de sistemas, reconocimiento de patrones, comunicaciones y sistemas de información, medicina, sistemas expertos, manufactura de circuitos integrados. Sin embargo, las aplicaciones más significativas están siendo empleadas en los problemas de control, con el diseño e implementación de controladores difusos (García et al., 2005).

(Alzate et al., 2007) describe que la lógica difusa se ha convertido en una herramienta muy útil para el desarrollo de técnicas de control en robótica ya que es capaz de tratar la incertidumbre existente en la medida realizada por los sensores. Además, cita que las aplicaciones con lógica difusa para agentes autónomos han crecido significativamente debido a sus características que posee, entre estas tenemos: facilidad para interpolar las medidas de los sensores, tratamiento robusto de la información imprecisa y la flexibilidad en la definición de las reglas de control no lineal.

Por su parte (Villalobos, 2014) menciona su aplicación en el control de una plataforma móvil sobre una línea marcada en el suelo, estos robots pueden variar desde los más básicos hasta los que recorren laberintos, su prototipo consiste en un pequeño robot previamente armado dando la experiencia de explorador, el mismo que recorrerá diferentes líneas.

2.9.6. Controlador Fuzzy en un robot seguidor de línea

(Alzate et al., 2007) realiza un modelo de un robot seguidor de línea utilizando un controlador Fuzzy para el movimiento de la plataforma móvil para recorrer la trayectoria. Presenta el modelo difuso basado en la lectura de los sensores los cuales tienen un novedoso modelo estadístico basado en las reglas de Mamdani, que relacionan las entradas con las salidas de un sistema.

El modelo difuso para el control de la plataforma móvil utiliza un modelo para el posicionamiento del prototipo y otro para la trayectoria. La trayectoria tiene un recorrido complejo con características parecidas a las que se presenta en las competencias de robótica, es decir diferentes tipos de curvas y ángulos de quiebre, lo más relevante de la trayectoria es que el ancho es constante en todo su recorrido (Alzate et al., 2007).

Para el movimiento del robot se aplica un sistema de referencia fijo como es el sistema de coordenadas cartesianas, además de un sistema de referencia móvil para la plataforma. En la figura 28-2, se presenta la plataforma que es controlada por tres sensores que detectan la línea y dos servomotores para su movimiento.

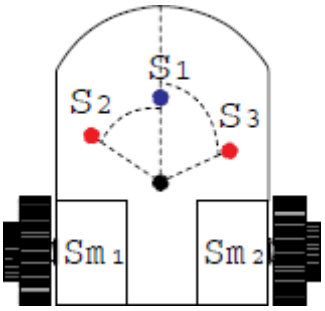


Figura 28-2. Configuración de la plataforma

Fuente: (Control difuso de una plataforma, 2007)

Los sensores entregan un voltaje de 5 voltios al detectar la línea de color negro y de 0 voltios cuando no detectan, en cambio los servomotores trabajan con señales PWM (Modulador por ancho de pulso), las cual tienen un rango de trabajo de 1ms a 2 ms y se detienen con un tren de pulso de 1,5 ms.

CAPÍTULO III

3. DISEÑO, SELECCIÓN Y ENSAMBLAJE DE ELEMENTOS PARA EL ROBOT SEGUIDOR DE LÍNEA

El robot seguidor de línea está basado en un diseño didáctico considerando parámetros que permitan un funcionamiento óptimo en cuanto a velocidad, resistencia, sintonización, tomando una estructura básica para su acoplamiento.

Para la conformación del robot seguidor de línea se consideran elementos electrónicos y mecánicos, los mismos que permitirán el funcionamiento del prototipo mediante la programación de un controlador PID y un controlador Fuzzy, entre los componentes tenemos: el microcontrolador, las ruedas, los sensores, el chasis, las baterías y una placa PCB para el accionamiento del robot seguidor de línea.

Los componentes antes mencionados se colocarán en el chasis de tal forma que tenga un espacio distribuido en la plataforma y de esta manera realizar la conexión de los distintos elementos para ejecutar las respectivas pruebas de funcionamiento del robot seguidor de línea.

3.1. Elementos de suministro de energía

Los dispositivos a utilizar en el suministro de energía son dos baterías de diferente voltaje; una batería de 7,4V suministra corriente a los elementos como: el Arduino, placa PCB, controlador de motores y sensor.

Por otra parte, la batería de 11,1V suministra corriente a los micromotores por medio de la entrada de 12V del controlador de motores (driver) permitiendo el movimiento de las ruedas para la traslación del robot seguidor de línea en la pista.

3.1.1. Batería lipo 3S 11,1V TURNIGY

Este tipo de batería de descarga alta es ideal para alimentar algún proyecto R/C, robótico o portátil, además es una excelente elección para prototipos que requieren de una batería pequeña pero que contenga mucha fuerza.

La batería está compuesta por tres celdas de 3,7V c/u, la mismas que entregan a la salida un total de 11,1 V y 1000mA de almacenamiento de carga, en la figura 29-3 se ilustra este elemento y presentando las siguientes características:

- 25C velocidad de descarga continua
- Conector de carga JST-XH
- Conector de descarga XT60

- Dimensiones: 76mm x 34mm x 15mm
- Peso: 82gr

Figura 29-3. Batería lipo 3S 11,1V TURNIGY

Realizado por: Caiza, A; Morales, A. 2020

3.1.2. Batería lipo 2S 7,4V TURNIGY NANO TEC

Esta batería está compuesta de 2 celdas de 3,7V c/u con un voltaje de salida igual a 7,4V y 300mA de almacenamiento como se observa en la figura 30-3. Para su carga se necesita de un cargador especial, está batería también puede alimentar varios componentes electrónicos y contiene las siguientes características:

- 35C velocidad de descarga continua/70C velocidad de descarga pico
- Conector de carga JST-XH
- Conector descarga JST
- Dimensiones: 43mm x 17mm x 12mm
- Peso: 17gr

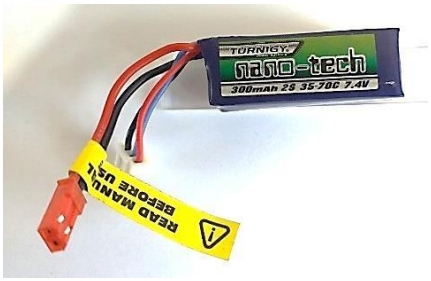


Figura 30-3. Batería lipo 2S 7,4V TURNIGY NANO-TECH

Realizado por: Caiza, A; Morales, A. 2020

3.2. Elementos para el desplazamiento

Para la traslación del prototipo en la pista se eligen los siguientes elementos mecánicos: un par de llantas para robot seguidor de línea tipo velocista, una rueda omnidireccional (rueda loca) y dos

micromotores. Como elemento electrónico se selecciona un controlador dual de motores (driver), él mismo controla la velocidad y sentido de giro de los micromotores.

3.2.1. Par de llantas para robot seguidor de línea velocista

Este tipo de llantas poseen alta adherencia a la superficie de la pista, soportan el peso del robot y son ideales para acoplar a los micromotores, estos elementos permiten el movimiento del robot seguidor de línea y se ilustran en la figura 31-3; de acuerdo a los requerimientos poseen las siguientes especificaciones:

- Diámetro exterior del rin:16mm
- Diámetro interior del rin: 13mm
- Diámetro total del rin + llanta: 23mm
- Ancho: 40mm
- Diámetro del eje: 3mm (+/- 0,02mm)
- Material del rin: Aluminio
- Material de la llanta: caucho de silicona de dureza 20A
- Peso: 34,5 gr
- Color: rojo

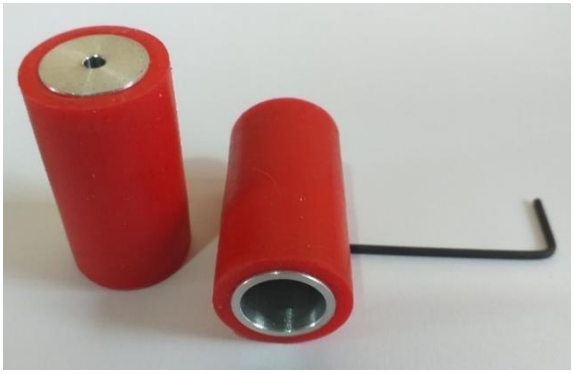


Figura 31-3. Ruedas seguidor de línea tipo velocista

Realizado por: Caiza, A; Morales, A. 2020

3.2.2. Rueda loca metálica de 3/8”

Esta rueda se utiliza como tercer punto de contacto para el robot seguidor de línea, permite al robot superar curvas y seguir la trayectoria debido a que puede moverse en todas las direcciones. Está compuesta por una esfera metálica, carcasa negra de ABS, dos espaciadores y dos tornillos para su sujeción en el chasis como se muestra en la figura 32-3; su altura puede variar de 0,4” a 0,6” dependiendo de la combinación de espaciadores posee las siguientes características:

- Diámetro de bola: 9,5mm (3/8¨)
- Soporte en ABS

- Distancia entre orificios de los tornillos: 13,4mm

Figura 32-3. Rueda loca metálica de 3/8¨

Realizado por: Caiza, A; Morales, A. 2020

3.2.3. Micromotor 10:1 Pololu HPCB 12V

Es un motorreductor de 12Vdc contiene alta potencia con escobillas de carbón de alta duración, además posee una caja de engranajes de metal de 9,96:1 como se muestra en la figura 33-3. Esta versión HPCB se diferencia de otros modelos por sus terminales de cobre que son 0,5 mm más anchos que otras versiones, el eje de salida en forma de D proporciona el giro a las ruedas y tiene las siguientes características:

- Dimensiones: 10mm x 12mm x 26mm
- Peso: 9,5gr
- Diámetro del eje: 9mm x 3mm
- Velocidad sin carga: 3400rpm
- Corriente sin carga: 0,06A
- Corriente de bloqueo: 0,75A
- Par de bloqueo: 0,17 kg*cm
- Potencia máxima: 1,5W

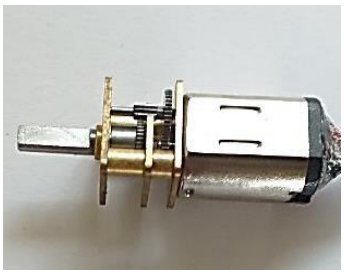


Figura 33-3. Micromotor metálico 10:1 Polulu HPCB 12V

Realizado por: Caiza, A; Morales, A. 2020

Este micromotor tiene elementos adicionales para el acople al chasis y su conexión como: soportes y cables de pines hembra respectivamente.

3.2.3.1. Soporte para micromotor

Este elemento es de material ABS de color negro diseñado para acoplarse al micromotor, se sujeta al chasis por medio de tornillos y tuercas; para el prototipo se utiliza un soporte para cada micromotor y se ilustra en la figura 34-3.

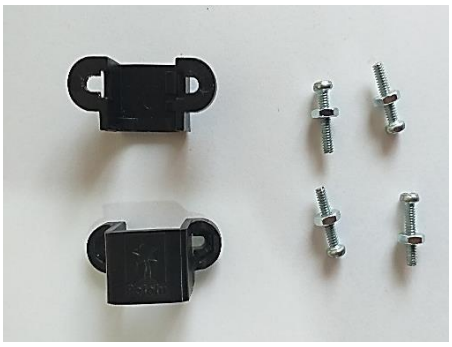


Figura 34-3. Soporte para micromotor

Realizado por: Caiza, A; Morales, A. 2020

3.2.3.2. Cable con plug JST de pines hembra

Este elemento de 10 cm de longitud contiene dos cables independientes de diferente color para identificar su polaridad, un cable de color negro para el polo negativo y otro de color rojo para el polo positivo. Los cables se sueldan a los terminales de cobre del motorreductor para su alimentación; de igual forma se utiliza un cable para cada micromotor, como se observa en la figura 35-3.

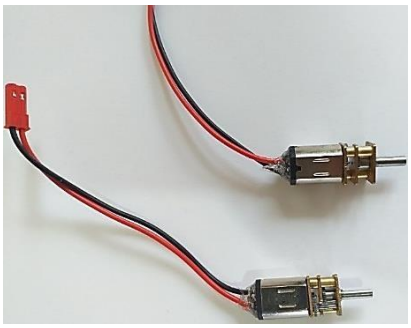


Figura 35-3. Cable plug JST 2 pines hembra

Realizado por: Caiza, A; Morales, A. 2020

3.3. Elementos de puesta de marcha y control

Para el accionamiento se diseña y construye una placa PCB con elementos mecánicos y electrónicos; para el control se utilizan un módulo dual de motores TB6612FNG y un Arduino NANO que es el cerebro de control de todos los elementos del prototipo.

3.3.1. Placa PCB de accionamiento

Esta placa es un conjunto formado por 20 pines machos, un pulsador, un interruptor, un led, 2 capacitores cerámicos de 10nF y 3 resistencias (dos resistencias de 10kΩ y una de 330Ω), conectados como se muestra en la figura 36-3.

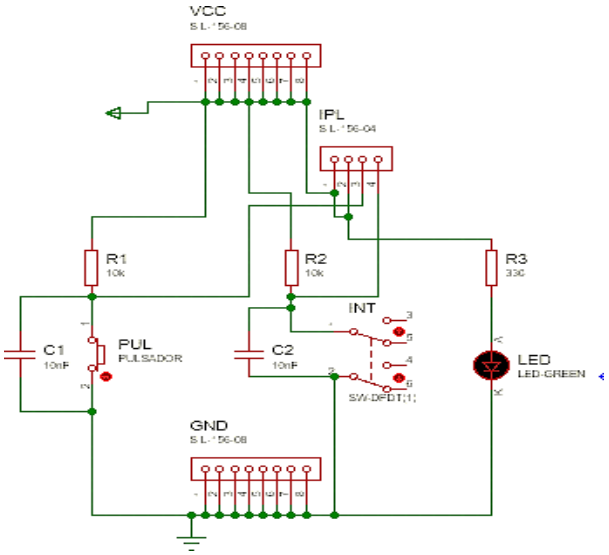


Figura 36-3. Conexión electrónica de la placa PCB de accionamiento

Realizado por: Caiza, A; Morales, A. 2020

Los 20 pines están distribuidos de la siguiente forma: 8 pines de conexión a Vcc ubicados horizontalmente en la parte superior de la placa, 8 pines de tierra (GND) colocados de manera horizontal en la parte inferior y 4 pines colocados de forma vertical a la derecha de la placa.

La barra de espadines de la parte derecha contiene salidas y están colocados en el siguiente orden desde arriba hacia abajo: el primer pin es un adicional de Vcc, el segundo pin del led, el tercer pin del pulsador y el último pin para el interruptor, así lo muestra la figura 37-3.

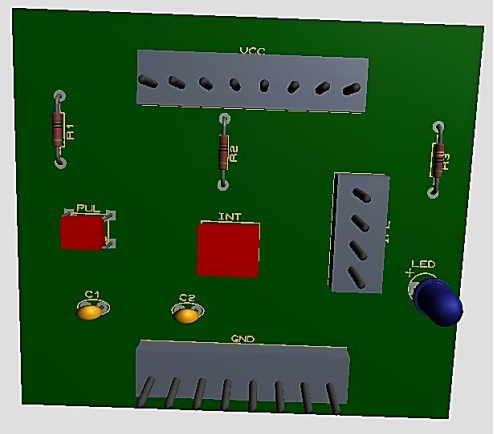


Figura 37-3. Placa PCB de accionamiento, Proteus

Realizado por: Caiza, A; Morales, A. 2020

Para la elaboración e impresión de la placa PCB se digitaliza el esquema mostrado en la figura 38-3, luego se coloca y se suelda los elementos descritos anteriormente.

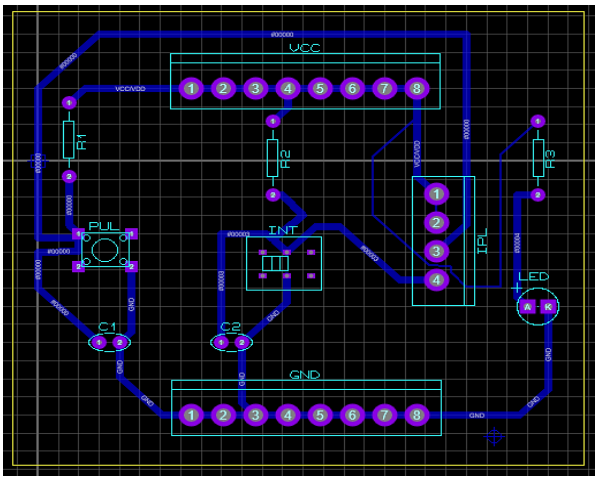


Figura 38-3. Placa PCB para impresión

Realizado por: Caiza, A; Morales, A. 2020

El resultado final de la placa se ilustra en la figura 39-3, misma que tiene forma rectangular y será sujetada al chasis mediante dos tornillos con tuercas, colocados en las esquinas opuestas de una de sus diagonales.

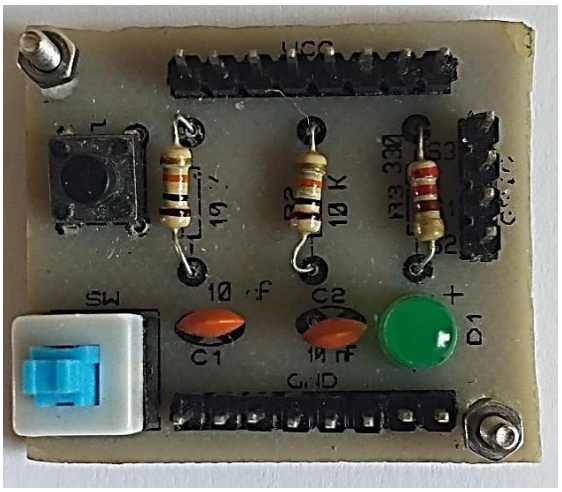


Figura 39-3. Placa PCB de accionamiento

Realizado por: Caiza, A; Morales, A. 2020

3.3.2. Controlador dual de motores TB6612FNG

Es un dispositivo conocido también como driver es encargado de proporcionar corriente, controlar la velocidad y el sentido de giro a los micromotores, su configuración interna basada en puentes H de MOSFET controla los motores bidireccionales de CC de forma independiente siendo más eficientes que los puentes H basados en BJT.

El driver tiene pines de entrada para el control de dirección (AIN y BIN), pines de velocidad (PWMA y PWMB) y pines de salida de alimentación (AOUT y BOUT) para cada motor, este componente se observa en la figura 40-3 e incluye una tira de 16 espadines tipo macho que se suelda a los pines de la placa; sus características son:

- Tensión recomendada del motor (VMOT): 4,5V a 13,5V
- Tensión lógica de (Vcc): 2,7V a 5,5V
- Corriente de salida máxima: 3A
- Circuito de apagado térmico incorporado
- Condensadores de filtrado

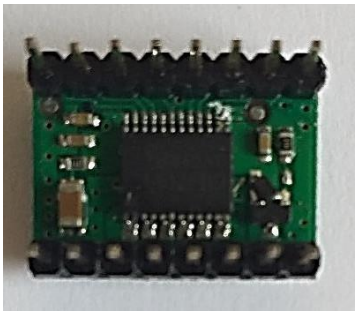


Figura 40-3. Controlador dual de motores TB6612FNG

Realizado por: Caiza, A; Morales, A. 2020

3.3.3. Arduino nano V3.0 ATMEGA 328

Es un dispositivo electrónico basado en el microcontrolador ATMega328, es más funcional que otro Arduino por su tamaño reducido, este dispositivo contribuye en el funcionamiento del robot; ya que, a través de él se carga el programa realizado en el software Arduino IDE utilizando un lenguaje de programación adecuado para cada elemento que compone el robot seguidor de línea.

Consta de una entrada USB que carga el código de programación para su ejecución, posee una memoria de 16KB, 2KB de SRAM y 1KB de EPROM, este microcontrolador se ilustra en la figura 41-3 y funciona en un rango de voltaje de 7 a 12 V. Entre sus desventajas se encuentra el menor número de entradas/salidas, menor espacio de memoria presentando las siguientes características:

- Tensión de operación: 5V
- Arquitectura: AVR
- Pines E/S digitales: 22(6 proveen salida PWM)
- Entradas analógicas: 8
- Memoria Flash: 32KB
- Velocidad de reloj: 16MHz
- Corriente continua de E/S: 40mA
- Tamaño de PCB: 18 x 45mm
- Peso: 7gr.
- Consumo energía: 19mA

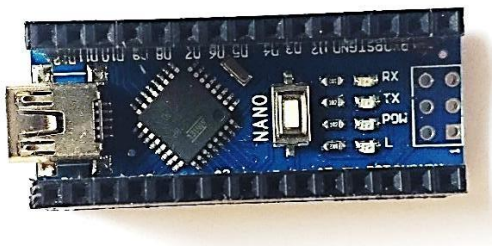


Figura 41-3. Arduino Nano ATMega328

Realizado por: Caiza, A; Morales, A. 2020

3.4. Elementos para detección de pista

Para el reconocimiento de la línea negra sobre fondo blanco de la pista se utiliza un sensor de reflectancia QRT-8RC, este componente electrónico envía una señal de posición al Arduino mediante su conjunto de 8 sensores.

3.4.1. Sensor QTR-8RC Pololu

Este sensor posee 8 sensores de led IR/fototransistor que están colocados en un paso de 0,375", por lo que es un excelente detector de pista para un robot seguidor de línea, estos sensores están conectado en serie para reducir a la mitad el consumo de corriente y cada sensor emite una salida digital que se mide por separado.

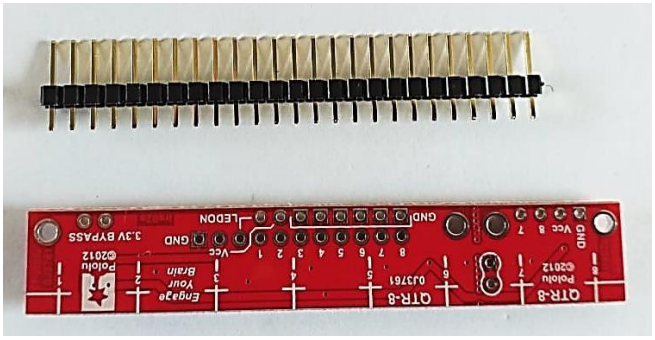


Figura 42-3. Sensor QTR-8RC

Realizado por: Caiza, A; Morales, A. 2020

Este sensor es diseñado para usarse de manera paralela a la superficie que se desea detectar a una altura de 3mm-9mm para detección óptima, incluye una tira de espadines como se muestra en la figura 42-3 y entre sus características encontramos:

- Tensión de funcionamiento:3,3-5V
- Corriente de suministro: 100mA
- Formato de salida: 8 señales digitales con E/S
- Dimensiones: 76mm x 12,7mm x 3mm

3.5. Base para componentes

Este elemento resiste el peso de todos los componentes y los cambios de velocidad durante el movimiento del prototipo, se puede realizar en diferentes formas, materiales, tamaños y colores. Para la distribución de los componentes electrónicos y mecánicos se construyó un chasis tomando consideraciones de tamaño, forma, peso, etc; obteniendo así un modelo didáctico y accesible.

3.5.1. Forma del chasis

Para construir la forma del chasis se ubica cada elemento mecánico y electrónico sobre el mismo a excepción de las ruedas posteriores, colocando los elementos de mayor peso en el eje de las ruedas para conseguir una mejor tracción en el robot. Para la unión entre la parte delantera y trasera del chasis se diseña una curvatura para evitar la fractura de la plataforma debido a que el robot adquiere velocidades altas en funcionamiento.

En la figura 43-3 se ilustra la vista superior de la distribución de elementos en el chasis y su forma; mismo que fue creado con la ayuda del software AutoCAD 2021.

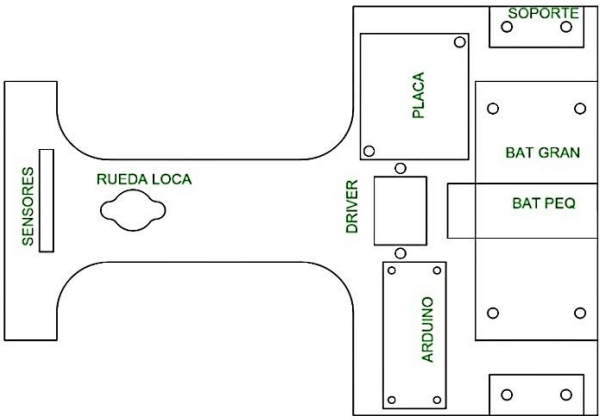


Figura 43-3. Vista superior chasis, AutoCAD 2021

Realizado por: Caiza, A; Morales, A. 2020

3.5.2. Dimensionamiento del chasis

Para el dimensionamiento del chasis se rige a parámetros de dimensiones establecidos en los reglamentos de competencias, mismo que es diferente dependiendo el organizador y en su mayoría establece no superar un rectángulo de 30cm de largo x 25cm de ancho con todos sus accesorios en su máxima extensión. Partiendo de esta consideración se mide las dimensiones de cada elemento a excepción de las ruedas.

En la figura 44-3 se indica las dimensiones del chasis en milímetros el cual está construido en un rectángulo de 170mm x 120mm, también se puede observar las distancias de sujeción de cada elemento de acuerdo a su ubicación. Para obtener una distribución uniforme de los elementos se consideró un eje paralelo en la mitad de la posición de los soportes de los micromotores tomado de izquierda a derecha y un eje perpendicular tomado en el punto medio de la distancia total del ancho del chasis.

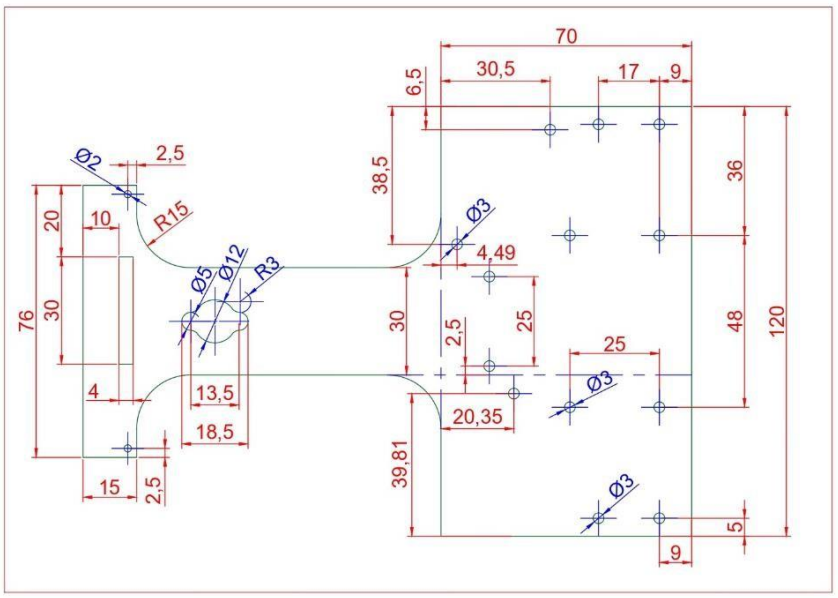


Figura 44-3. Dimensiones del chasis, AutoCAD 2021

Realizado por: Caiza, A; Morales, A. 2020

3.5.3. Material del chasis

Tomando en consideración que el chasis debe ser de un material liviano para alcanzar velocidades altas y a la vez resistente para soportar el peso de todos los elementos cuando se encuentre en funcionamiento se elige como material el vidrio acrílico. Este material es utilizado con mayor frecuencia en un robot seguidor de línea por su accesibilidad y facilidad de manipulación para obtener la forma deseada por medio de corte láser.

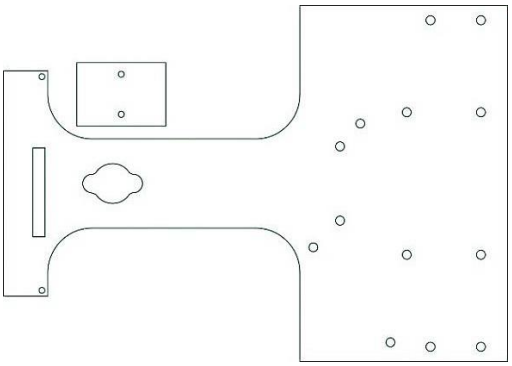


Figura 45-3. Chasis corte, AutoCAD 2021

Realizado por: Caiza, A; Morales, A. 2020

En la figura 45-3 se muestra la forma del chasis para su corte, además se observa una pequeña placa rectangular que permite sujetar la rueda loca para evitar la fricción del prototipo con la superficie de la pista, obteniendo la altura requerida para los sensores.

En la figura 46-3, se ilustra el resultado final del chasis en el material acrílico con un espesor de 3mm y color negro, este material presenta como desventajas: rayado con facilidad y fractura al aplicar fuerzas de alto impacto.

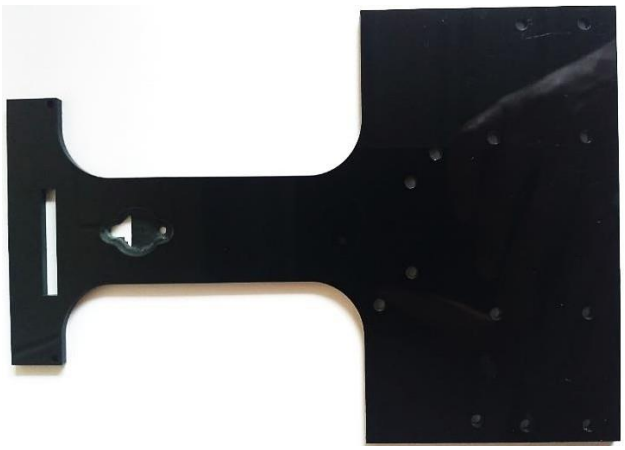


Figura 46-3. Chasis final

Realizado por: Caiza, A; Morales, A. 2020

3.6. Ensamblaje y conexión del robot seguidor de línea

Para el ensamblaje de los elementos con el chasis se utilizan correas plásticas para la sujeción del Arduino, el driver y la batería 11,1V, para los elementos restantes se usan tornillos y tuercas a excepción de la batería de 7,4V que es fijada con cinta doble faz. Para su conexión se elige cables de tipo hembra-hembra de 5cm y tipo macho-hembra de 10cm.

3.6.1. Ensamblaje de elementos electrónicos y mecánicos

Para comenzar el ensamblaje del robot se coloca las baterías en el chasis centradas en el eje de los micromotores. Primero se ubica la batería de 7,4V perpendicular al eje de los motores y la batería de 11,1V paralela al eje y sobre la batería pequeña, como se muestra en la figura 47-3.

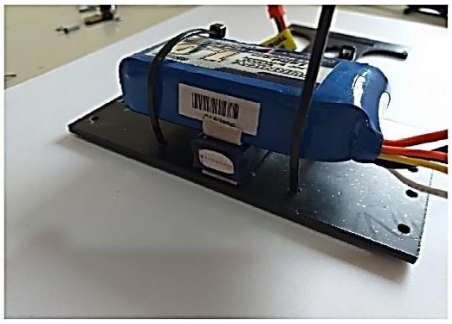


Figura 47-3. Colocación de baterías

Realizado por: Caiza, A; Morales, A. 2020

Para fijar los micromotores se soldó en primera instancia los cables jack tipo hembra a los terminales de cada motor como se observa en la figura 48-3a, se acoplan las ruedas al eje de cada micromotor mediante un prisionero y el uso de una llave hexagonal ilustrado en la figura 48-3b y 48-3c. Finalmente se sujeta el conjunto soporte-motor al chasis por medio de tornillos y tuercas como se muestra en la figura 48-3d.

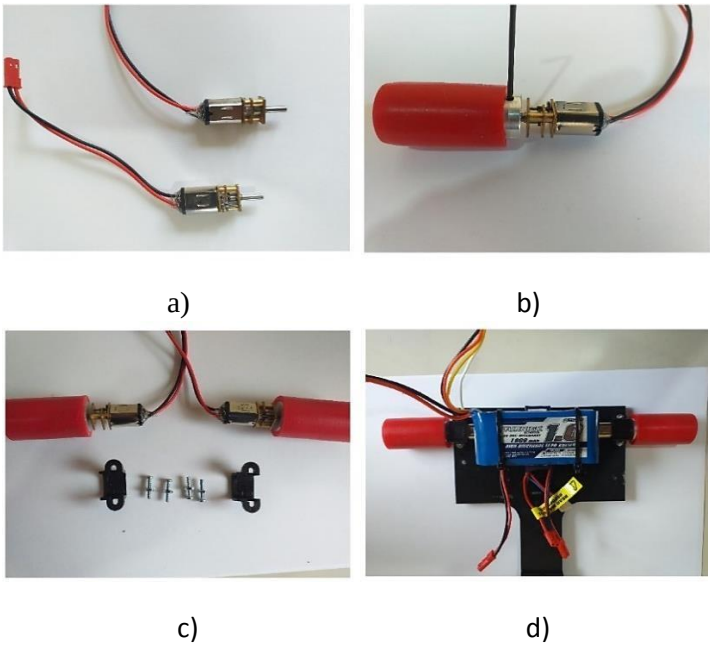


Figura 48-3. Sujeción micromotores y ruedas

Realizado por: Caiza, A; Morales, A. 2020

El driver y el Arduino se ubican en sus áreas designadas como se muestra en la figura 49-3a y 49-3b respectivamente.

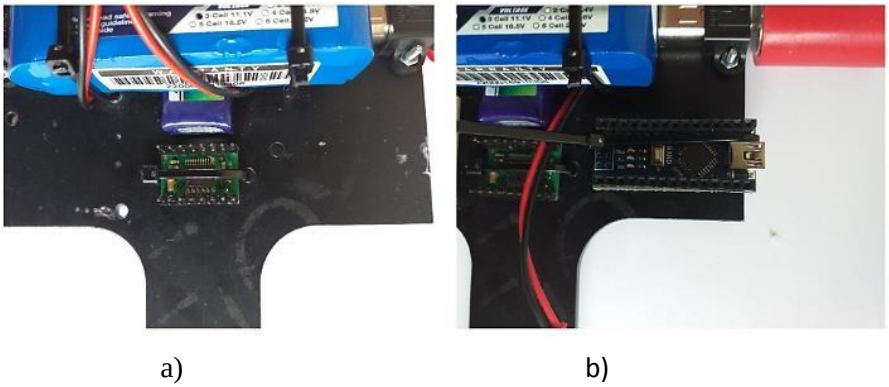


Figura 49-3. Ubicación de microcontroladores

Realizado por: Caiza, A; Morales, A. 2020

En la figura 50-3 se observa la placa PCB acoplada al chasis, en la siguiente figura 51-3 se ilustra el área donde se ubica la rueda loca ensamblada a su base adicional (adherida con silicona).

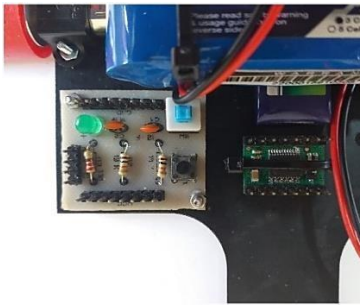


Figura 50-3. Sujeción de placa de accionamiento

Realizado por: Caiza, A; Morales, A. 2020

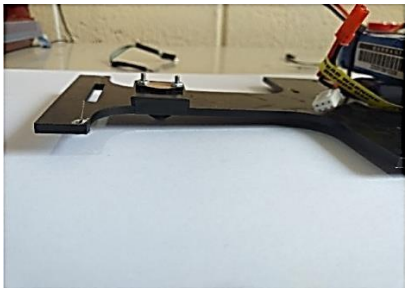
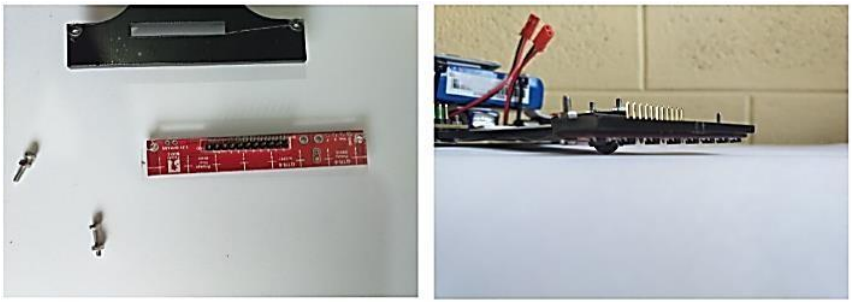


Figura 51-3. Acople rueda loca

Realizado por: Caiza, A; Morales, A. 2020

El último componente en acoplar al chasis es el sensor QTR-8RC, para colocar este elemento el chasis tiene una perforación en la parte delantera para encajar los espadines y facilitar su conexión con el Arduino como se observa en la figura 52-3a; además es ubicado debajo del chasis como se visualiza la figura 52-3b.



a) b)

Figura 52-3. Ensamble del sensor

Realizado por: Caiza, A; Morales, A. 2020

En la figura 53-3 se muestra el robot seguidor de línea con todos sus componentes electrónicos y mecánicos a utilizar para la programación de un controlador PID y un controlador Fuzzy.

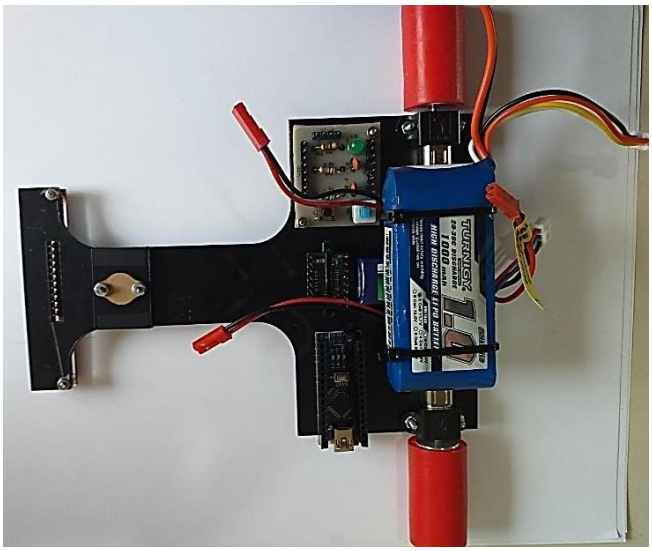


Figura 53-3. Robot seguidor de línea

Realizado por: Caiza, A; Morales, A. 2020

3.6.2. Conexión del robot seguidor de línea

Para conectar los elementos entre sí utilizamos cables de tipo hembra-hembra y tipo macho-hembra, la utilización de ellos depende de la distancia entre componentes y de sus pines. En la figura 54-3 se ilustra el esquema de conexión que se tiene para cada elemento, la conexión es simulada con la ayuda del programa Proteus 8.1.

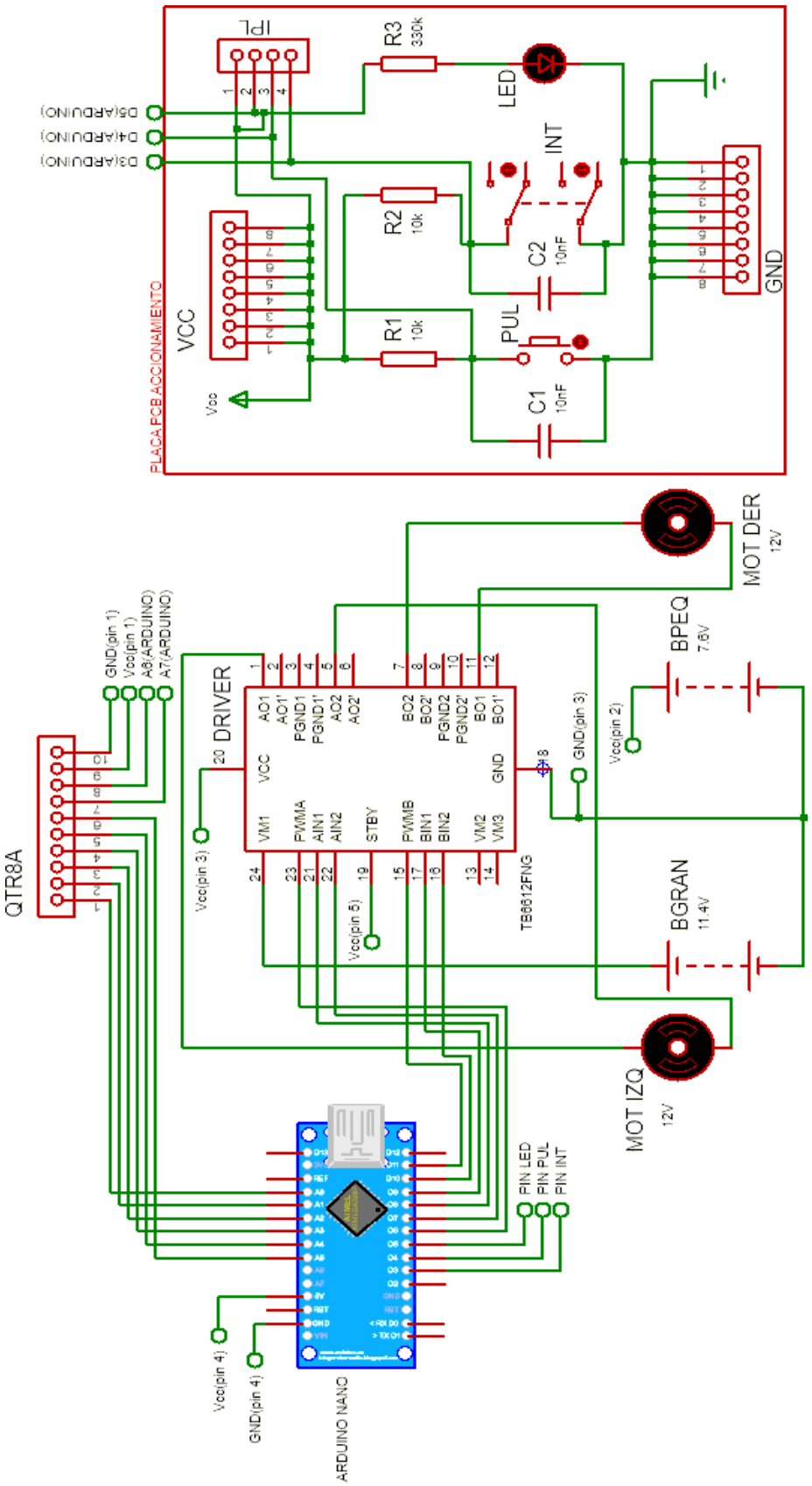


Figura 54-3. Simulación de conexión

Realizado por: Caiza, A; Morales, A. 2020

CAPÍTULO IV

4. PROGRAMACIÓN Y SINTONIZACIÓN DE UN CONTROLADOR PID Y UN CONTROLADOR FUZZY

Para el desarrollo de la programación de los controladores se requiere del software Arduino IDE para crear un código funcional, secuencial y con restricciones. Este software incluye en su lenguaje librerías, variables, funciones, estructura de programa y control para facilitar la edición del código.

El código de programación para cada controlador es descrito por algoritmos de control que permiten la funcionalidad independiente de cada elemento y en conjunto conseguir el desenvolvimiento preciso del robot seguidor de línea en la pista.

La sintonización va a depender de la teoría fundamentada para cada controlador y se logra su precisión por medio de la técnica de prueba y error.

4.1. Programación y sintonización del controlador PID

La programación del controlador PID se fundamenta en la acción de control de los parámetros: proporcional (P), integral (I) y derivativo (D); mientras la sintonización se consigue variando las constantes $K_P,\ K_I,\ K_D$ con el método heurístico hasta obtener una respuesta precisa del robot seguidor de línea en las pruebas de funcionamiento.

4.1.1. Programación

El robot seguidor de línea funciona con la programación del editor de código de Arduino que se observa en el anexo (A) que consta de 3 etapas, en la primera se incluye librerías y se definen variables y constantes, la segunda y tercera son dos funciones que Arduino coloca por defecto como son: el void setup y el void loop respectivamente. Estas funciones mencionadas son las que controlan el programa y la secuencia a través del lenguaje especificado dentro de ellas.

En la programación se incluye la librería "**QTRSensores.h"** al inicio de la programación, librería que permite mediante código la calibración del sensor de reflectancia y emisión de posición del robot seguidor de línea.

También se realiza la declaración de pines de acuerdo con la conexión entre elementos electrónicos y mecánicos hacia el Arduino, además se define las variables y constantes a utilizar. Cabe recalcar que a cada variable o constante se le inscribe un nombre único y así se identifica dentro del código.

Las variables y constantes van antecedidas con una variable del lenguaje Arduino IDE de acuerdo a los valores que tomen y tengan dentro del código de programación, en el gráfico 1-4 se ilustra el diagrama de flujo de dicho código.

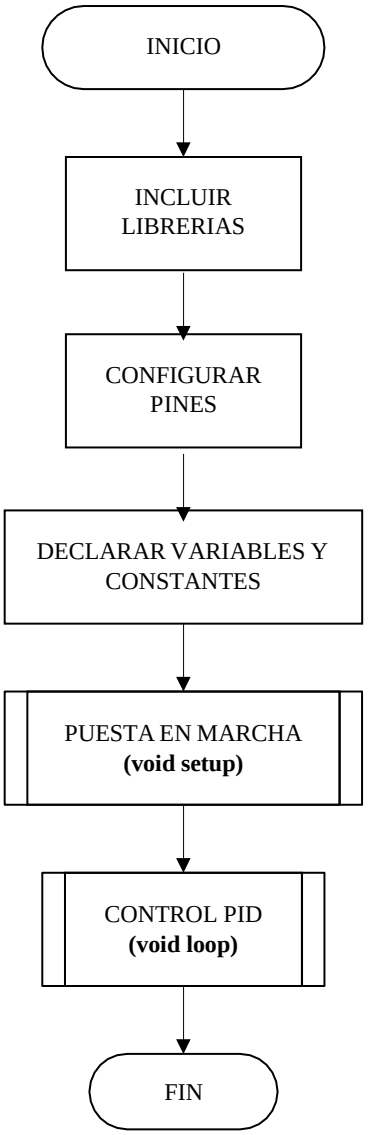


Gráfico 1-4. Diagrama de flujo para la programación de un controlador PID

Realizado por: Caiza, A; Morales, A. 2020

La función **void setup** es la encargada de inicializar la programación y es una función que se ejecuta una sola vez, en este subproceso se definen los pines del Arduino que funcionan como entradas (INPUT) y salidas (OUTPUT).

A continuación, se indica el direccionamiento de arranque de los motores utilizando la función digitalWrite y el estado HIGH (encendido) en los pines digitales que controlan la dirección de avance y el estado LOW (apagado) en la dirección de reversa de nuestro prototipo. En cambio, para establecer la velocidad de arranque de los motores se utiliza los pines PWM del Arduino

manejando valores desde 0 (apagado) hasta 255 (encendido) acompañado con la función analogWrite.

Finalmente se programa el encendido del robot seguidor de línea, la calibración de los sensores y la puesta en marcha para empezar su recorrido, en el gráfico 2-4 se detalla el algoritmo de la segunda etapa.

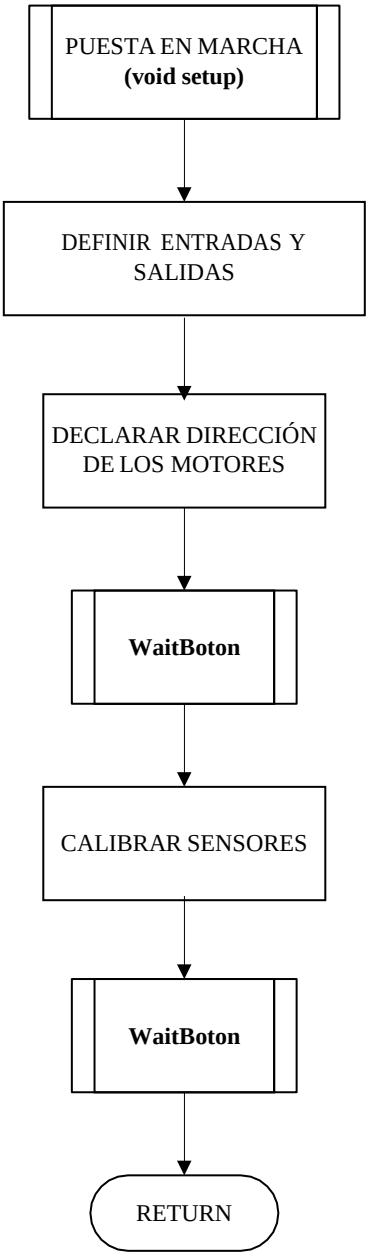


Gráfico 2-4. Función void setup

Realizado por: Caiza, A; Morales, A. 2020

Como se observa en la figura anterior existe un subproceso que va a permitir leer el estado del pulsador y está descrito como la función **WaitBoton,** esta función comprueba que el pulsador se presione para continuar procesando la programación, así lo ilustra el diagrama en el gráfico 3-4.

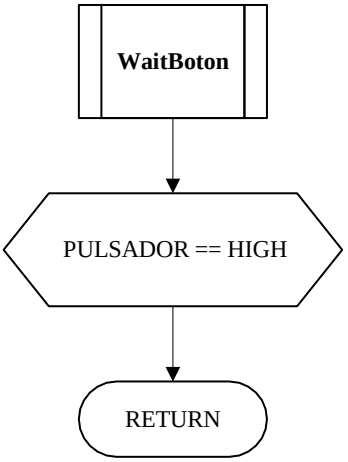


Gráfico 3-4. Función WaitBoton

Realizado por: Caiza, A; Morales, A. 2020

En la tercera etapa se desarrolla el proceso para el control PID, esta etapa controla los parámetros: proporcional (P), derivativo (D) e integral (I) del controlador para conseguir que el robot seguidor de línea siga su trayectoria sin oscilaciones y con una velocidad adecuada programando subprocesos como: la lectura de posición, accionamiento de frenos, cálculo del error y el control PID.

Para programar el modelo del control PID se emplea en base a código un procedimiento de fórmulas con el uso de las siguientes variables: error acumulado, error y error anterior; el resultado final de la suma de sus parámetros es igual a la velocidad de control o equivalente al control PID.

Se escribe en el código las acciones de control para que el motor derecho e izquierdo frenen o aceleren adecuadamente y se establece límites de velocidad para la acción de control PWM (0 = mínimo, 255 = máximo) para evitar desbordamiento; este algoritmo finaliza con la impresión de valores de set point, posición, error, velocidad del motor derecho y velocidad del motor izquierdo para su posterior análisis como se observa en el diagrama del gráfico 4-4.

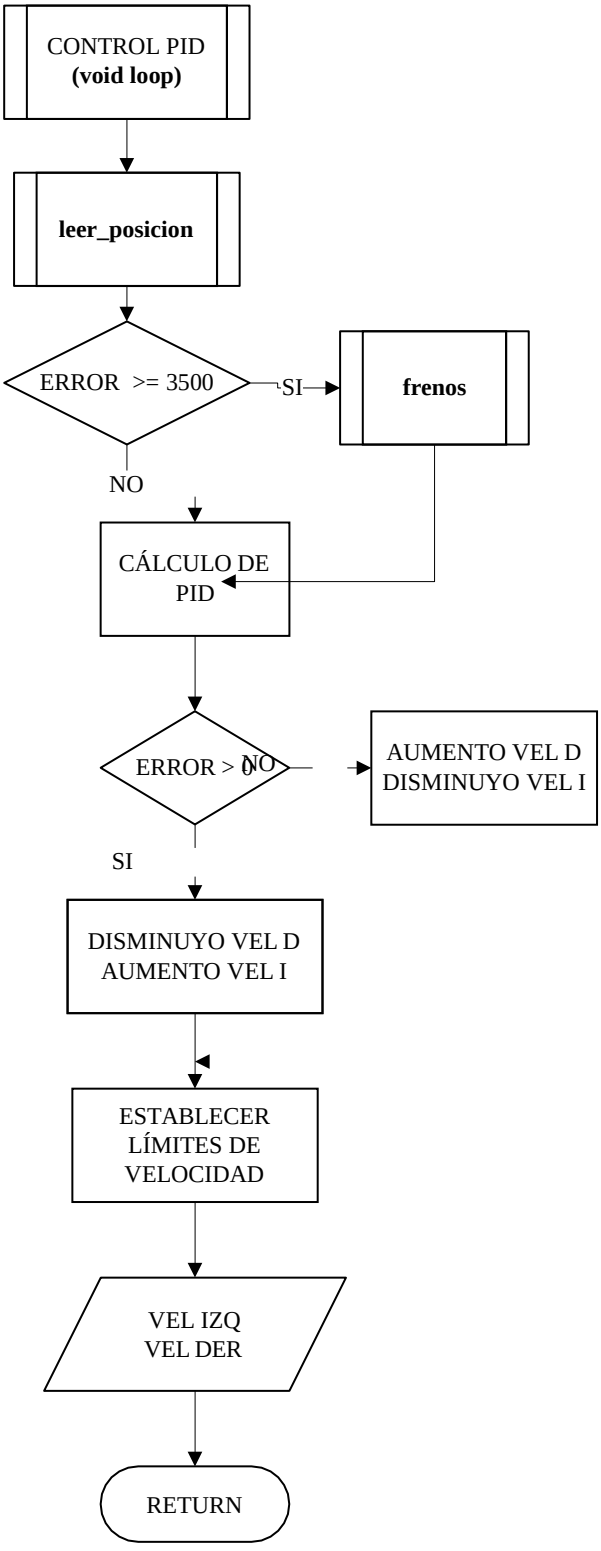


Gráfico 4-4. Función void loop

Realizado por: Caiza, A; Morales, A. 2020

Dentro de la función **void loop** se encuentran dos subprocesos: **leer_posicion** y frenos, la función frenos acciona la dirección de giro y la velocidad de control que deben tomar los motores, es decir se programa una acción cuando el robot seguidor de línea tenga una posición excesiva a la

derecha, otra cuando tenga posición excesiva a la izquierda como se ilustra en el algoritmo del gráfico 5-4.

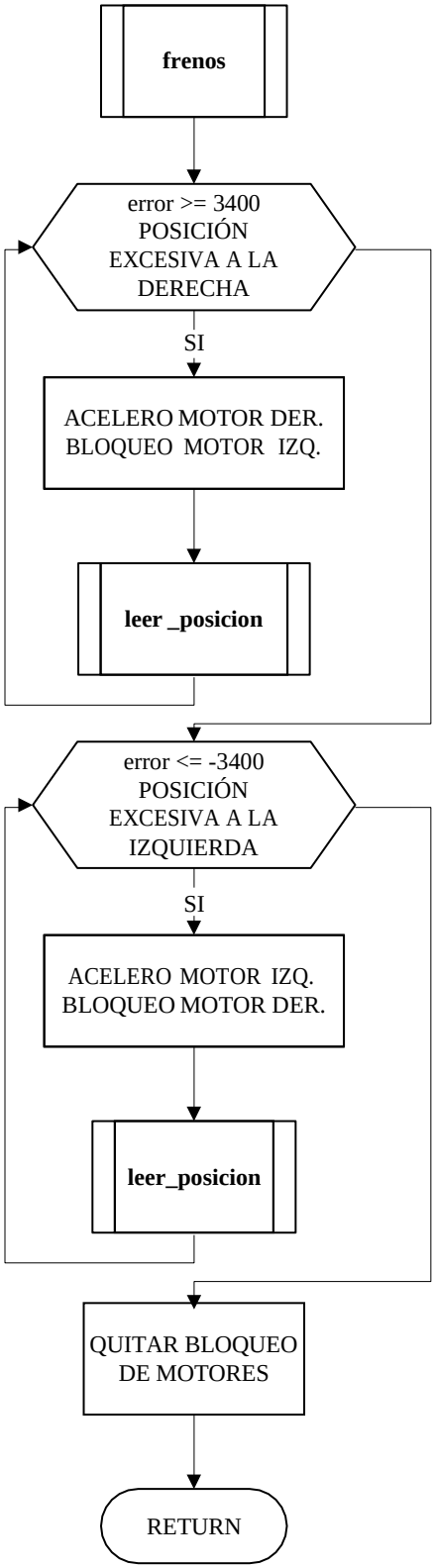


Gráfico 5-4. Función frenos

Realizado por: Caiza, A; Morales, A. 2020

Por otra parte, la función **leer_posicion** calcula la variable error de la diferencia entre el valor de set point (3500) y el valor de posición emitido por los sensores, el resultado de esta operación es tomado en valor absoluto así se describe este subproceso en el diagrama de flujo del gráfico 6-4.

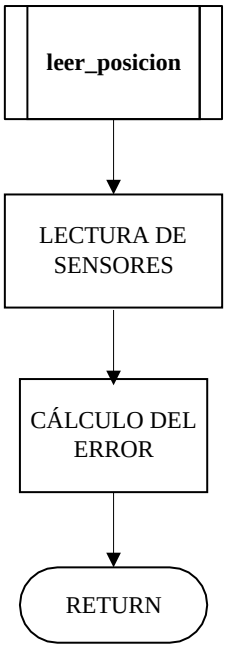


Gráfico 6-4. Función leer_posición

Realizado por: Caiza, A; Morales, A. 2020

4.1.2. Sintonización

La sintonización es un procedimiento para encontrar los valores de $K_P,\ K_I,\ K_D$, estos valores son declarados como constantes dentro de la programación y permiten que se ejecute el control PID en el código de Arduino IDE.

En el desarrollo de este proceso utilizamos un método heurístico o conocido en otras palabras como método de prueba y error, que consiste en dar valores a los parámetros $K_P,\ K_I,\ K_D$ hasta encontrar la estabilidad del robot seguidor de línea.

El primer valor en modificar y estabilizar será el parámetro K_P, valor que ayuda al prototipo seguir la trayectoria con oscilaciones, se comienza dando valores desde 0 hasta conseguir que el robot seguidor de línea cumpla lo descrito anteriormente manteniendo a $K_I = K_D = 0$, para encontrar este parámetro se aumenta o disminuye enteros, décimas, centésimas, etc, dependiendo de la respuesta que se tenga al realizar las pruebas de funcionamiento.

Luego de estabilizar K_P se comienza a variar el valor K_D para disminuir las oscilaciones del robot seguidor de línea, de igual manera se empieza a dar valores desde 0 manteniendo a $K_I = 0$ y K_P igual al valor encontrado en el procedimiento anterior. Para encontrar este valor se recomienda empezar desde la siguiente consideración $K_D = 2K_P$, y de igual manera se suma o resta enteros,

décimas, centésimas, etc; dependiendo de la respuesta que obtenga el robot seguidor de línea en sus respectivas pruebas.

Finalmente encontrados estos valores de K_P y K_D mediante una prueba de funcionamiento se analiza si es necesario cambiar el valor de K_I o mantenerlo igual a 0, esto se determina observando si el robot seguidor de línea consigue seguir su trayectoria sin oscilaciones o necesita de más precisión para alcanzar un recorrido preciso. De ser necesario se aumenta K_I y se disminuye K_D en igual proporción o viceversa hasta lograr estabilizar el prototipo a través de las pruebas de funcionamiento, cabe mencionar que en nuestro estudio se mantiene $K_I = 0$ porque al agregar valores no muestra un cambio significativo en el funcionamiento robot seguidor de línea.

Tabla 3-4: Valores de sintonización del controlador PID

PARAMETROS CONTROLADOR PID			CRITERIOS A CONSIDERAR		
Kp	**Kd**	**Ki**	**VEL NOM**	**SIGUE LA LÍNEA**	**OSCILACIONES**
1	0	0	60	NO	MUY ALTAS
0,0785	0,157	0	90	NO	ALTAS
0,0685	0,137	0	110	NO	ALTAS
0,0585	0,117	0	120	NO	BAJAS
0,0485	0,097	0	140	SI	MUY BAJAS
0, 052	0,171	0	150	SI	CASI NULAS

Fuente: (Autoría propia, 2020)

Realizado por: Caiza, A; Morales, A. 2020

La estabilización del controlador PID se logró variando los valores de las constantes K_P, K_I, K_D y velocidad nominal, estos valores fueron variando de manera aleatoria de acuerdo a la respuesta del robot seguidor de línea en las pruebas de funcionamiento. Obteniendo como resultado los siguientes valores: $K_P = 0{,}052, K_D = 0{,}171, K_I = 0$ y velocidad nominal = 150 RPM.

4.2. Programación y sintonización del controlador Fuzzy

Este controlador se basa en definir las variables de entrada y salida, su funcionamiento se rige en formar reglas IF-THEN (SI-ENTONCES) con la combinación de conjuntos de entrada y conjuntos de salida. La combinación de estos conjuntos logra que el robot seguidor de línea recorra su trayectoria a una velocidad adecuada.

Este código consta de dos variables de entrada que son: el error y la derivada del error y una variable de salida equivalente a la velocidad de control, cada variable de entrada y salida está formado por un número determinado de conjuntos.

La sintonización es el diseño y combinación entre etiquetas lingüísticas de entrada y salida para formular reglas que permitan el movimiento preciso del robot seguidor de línea sobre la pista.

4.2.1. Programación del controlador Fuzzy

El robot seguidor de línea funciona con la programación del controlador Fuzzy mostrado en el anexo (B), este código de programación va a controlar los siguientes parámetros: posición, velocidad de motores, el error y la derivada del error.

De igual forma está programación contiene 3 etapas; la primera etapa para incluir librerías, configurar pines de conexión al Arduino, declarar variables y constantes, la segunda etapa es la función void setup la cual inicializa la programación y la puesta en marcha del robot seguidor de línea y la última etapa ejecuta el control Fuzzy de este código.

Las librerías que se incluye en el código son: "**QTRSensores.h"** para el funcionamiento del sensor de posición y **"Fuzzy.h"** para realizar el sistema de fusificación. Las variables y constantes tienen un nombre de identificación dentro de la programación.

Cabe recalcar que tanto las variables y constantes son acompañadas de una variable del lenguaje Arduino dependiendo del valor que vayan a tomar de acuerdo con los requerimientos del robot seguidor de línea, en el gráfico 7-4 se ilustra el diagrama de flujo del controlador Fuzzy.

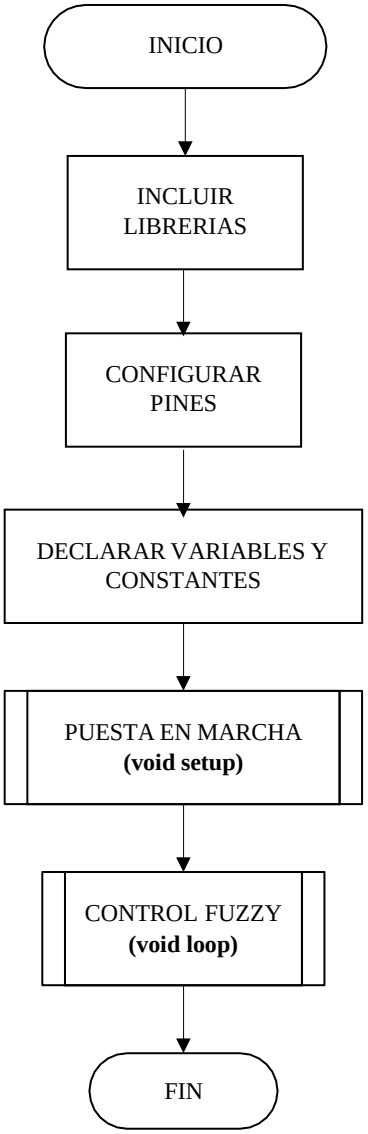


Gráfico 7-4. Diagrama de flujo del controlador Fuzzy

Realizado por: Caiza, A; Morales, A. 2020

Para la etapa de inicialización del robot seguidor de línea en la función **void setup** se declara la función del sistema de fusificación, a continuación, definimos los pines del Arduino que van a funcionar como entradas y como salidas, luego se programa la dirección de giro que van a tomar los micromotores usando la función digitalWrite y el estado HIGH (encendido) en los pines de avance, y LOW (apagado) para los pines de reversa que están conectados desde el driver al Arduino.

La velocidad de los motores en este ciclo se programa con la función analogWrite en los pines que controlan a los mismos, colocando el valor de 0 para mantenerlos apagados hasta que se ejecute toda la segunda etapa; para terminar el código en esta etapa se programa el encendido del

robot seguidor de línea, la calibración de los sensores y la puesta en marcha del mismo, en el algoritmo del gráfico 8-4 se observa el código de la segunda etapa.

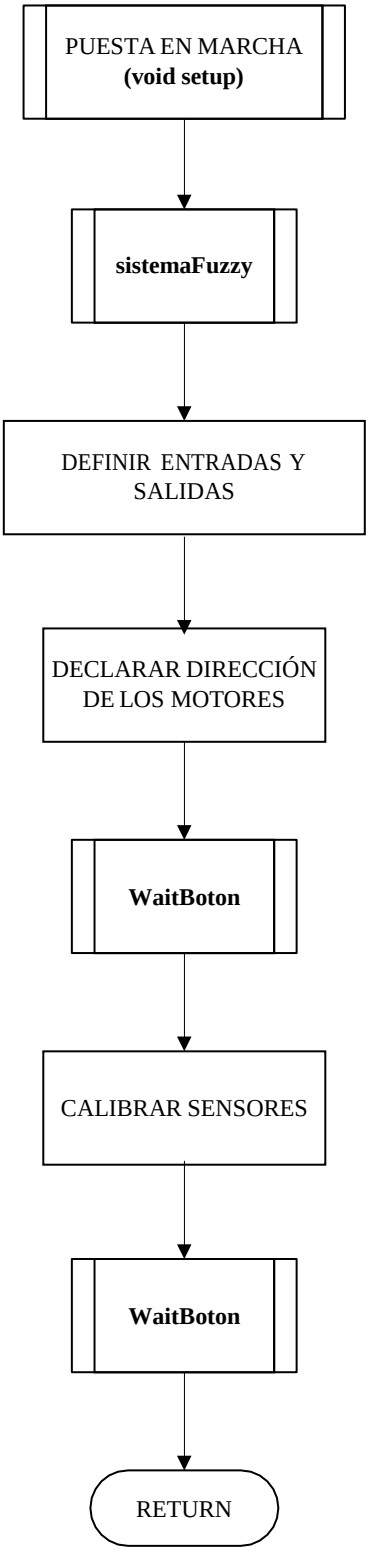


Gráfico 8-4. Diagrama de flujo de la función void setup

Realizado por: Caiza, A; Morales, A. 2020

En el diagrama anterior se observa que existe dos subprocesos: **sistemaFuzzy** y **WaitBoton,** la programación de estas funciones se detalla a través de una breve explicación y de un diagrama de flujo. La función **WaitBoton** mantiene el mismo funcionamiento de la programación del controlador PID, mostrado en el algoritmo del gráfico 3-4.

La función **sistemaFuzzy** se observa en el algoritmo del gráfico 9-4, en la cual mediante código se declara los conjuntos pertenecientes a las variables de entrada y salida. Tanto las variables de entrada como la de salida son programadas de acuerdo al intervalo que puedan tener: el error (variable de entrada 1), la derivada del error (variable de entrada 2) y la velocidad de control (variable de salida); el ciclo de esta función finaliza al describir la relación entre conjuntos a través de reglas.

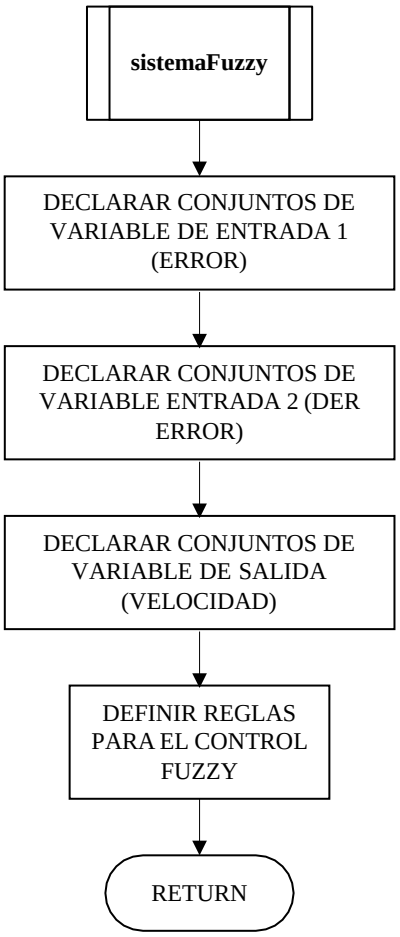


Gráfico 9-4. Función sistemaFuzzy

Realizado por: Caiza, A; Morales, A. 2020

En la tercera etapa del código de este controlador se describe el control fuzzy, las acciones de control, los límites de velocidad y los subprocesos: **leer_posicion** y frenos; el algoritmo de esta etapa es mostrado en el gráfico 10-4.

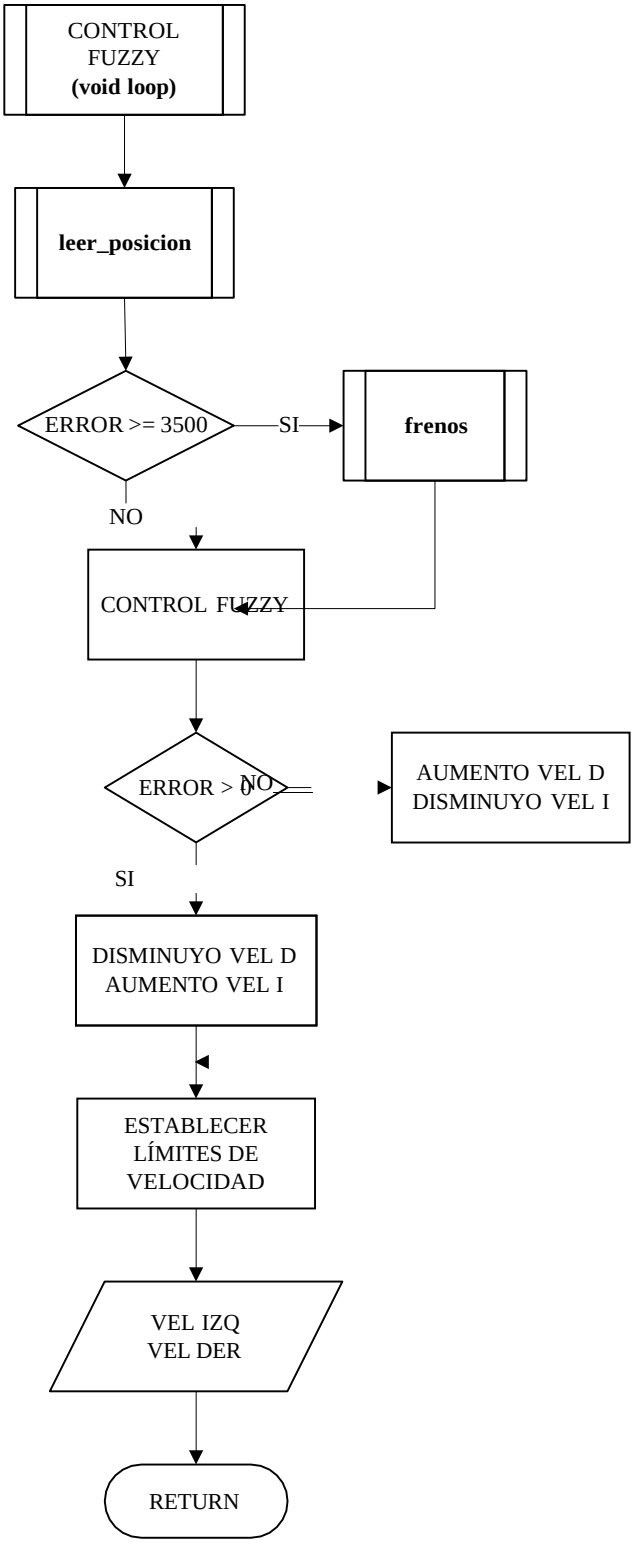


Gráfico 10-4. Función void loop

Realizado por: Caiza, A; Morales, A. 2020

El control fuzzy es el proceso de establecer las variables de entrada para ejecutar la fusificación y defusificación de los conjuntos obteniendo un resultado del sistema fuzzy, valor que representa la velocidad de control.

Las acciones de control se establecen para acelerar o desacelerar los motores dependiendo la lectura de posición del sensor de reflectancia y para evitar el desbordamiento de los motores se incluyen límites de velocidad.

Las funciones frenos y **leer_posicion** ejecutan el mismo proceso expuesto en el controlador PID, mostrado en los diagramas de flujo de los gráficos 5-4 y 6-4 respectivamente.

4.2.2. Sintonización

La sintonización es el proceso que va a determinar el número de conjuntos y el rango de cada conjunto para las variables de entrada y salida, además se realiza la combinación entre conjuntos para formular las reglas que forman parte la fusificación y defusificación.

La sintonización se realiza con la ayuda y diseño del fuzzy Toolkit de Matlab, procedimiento que consta en elaborar una relación entre dos variables de entrada y una salida con el método de Mamdani como se muestra en la figura 55-4.

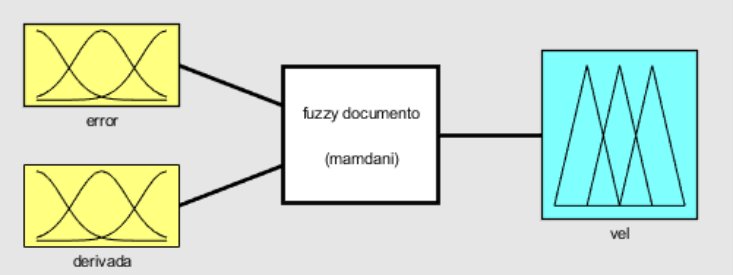


Figura 55-4. Esquema del control Fuzzy, Matlab

Realizado por: Caiza, A; Morales, A. 2020

Para la variable de entrada 1 (error) se definieron 3 conjuntos con las siguientes etiquetas lingüísticas: bajo (conjunto triangular), medio (conjunto triangular) y alto (conjunto trapezoidal) ilustrado en la figura 56-4. El error se calcula mediante la ecuación 8, el resultado esta operación puede tomar valores negativos o positivos por ello se toma su resultado como valor absoluto para definir un rango positivo de 0 a 3500 para este controlador.

$$error = sp - position \qquad \text{(Ec. 8)}$$

Donde:

$sp = 3500$

$position$ $(0\ a\ 7000)$, valor emitido por el sensor.

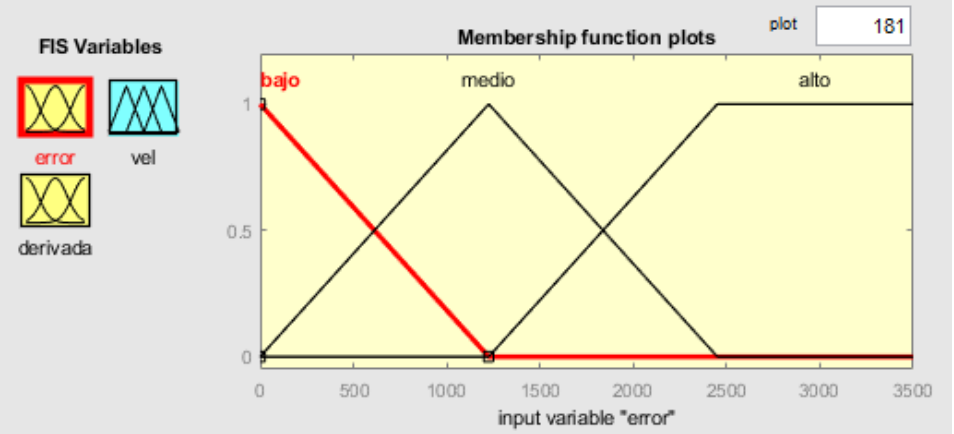


Figura 56-4. Etiquetas lingüísticas para el error

Realizado por: Caiza, A; Morales, A. 2020

La variable de entrada 2 (derivada del error) consta de 3 conjuntos con las siguientes etiquetas lingüísticas: Negativo (conjunto trapezoidal), zero (conjunto triangular) y Positivo (conjunto trapezoidal) como se observa en la figura 57-4. La derivada del error se encuentra con la ecuación 9, cabe mencionar que para encontrar el rango de funcionamiento se inicia con un rango de -350000 a 350000 y se va disminuyendo conforme el robot encuentre la estabilización requerida.

$$\Delta error = \frac{errorabs - error\ anterior}{t} \qquad \text{(Ec. 9)}$$

Donde:

$t = 0{,}15$: tiempo en seg.

$errorabs = abs(error)$.

$erroranterior = errorabs$.

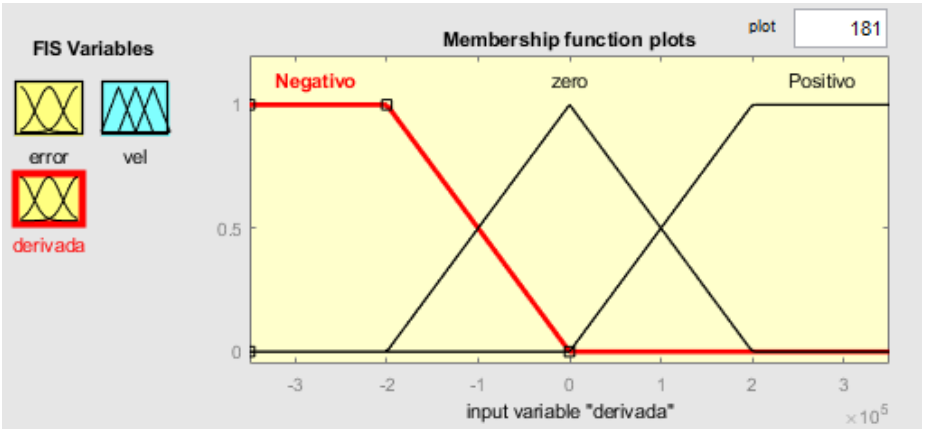


Figura 57-4. Etiquetas lingüísticas para la derivada del error

Realizado por: Caiza, A; Morales, A. 2020

A continuación, en la tabla 4-4 se muestra las funciones de membresía y el rango para cada conjunto de las dos variables de entrada.

Tabla 4-4: Funciones de membresía de las variables de entrada

VARIABLE DE ENTRADA 1 (error)			
Funciones de membresía	**Bajo**	**medio**	**alto**
Intervalo	(0, 0, 0, 1225)	(0, 1225, 1225, 2450)	(1225, 2450, 3500, 3500)
VARIABLE DE ENTRADA 2 (derivada del error)			
Funciones de membresía	**Negativo**	**zero**	**Positivo**
Intervalo	(-10000, -10000, -5000, 0)	(-5000, 0, 0, 5000)	(0, 5000, 10000, 10000)

Fuente: (Autoría propia, 2020)

Realizado por: Caiza, A; Morales, A. 2020

La variable de salida (velocidad) está formada por 5 conjuntos con las siguientes etiquetas lingüísticas: V_zero (conjunto triangular), V_bajo (conjunto triangular), V_medio (conjunto triangular), V_alta (conjunto triangular) y V_m_malta (conjunto trapezoidal) como se muestra en la figura 58-4. El rango de esta variable se encuentra con las pruebas de funcionamiento y para este controlador es de 0 a 150.

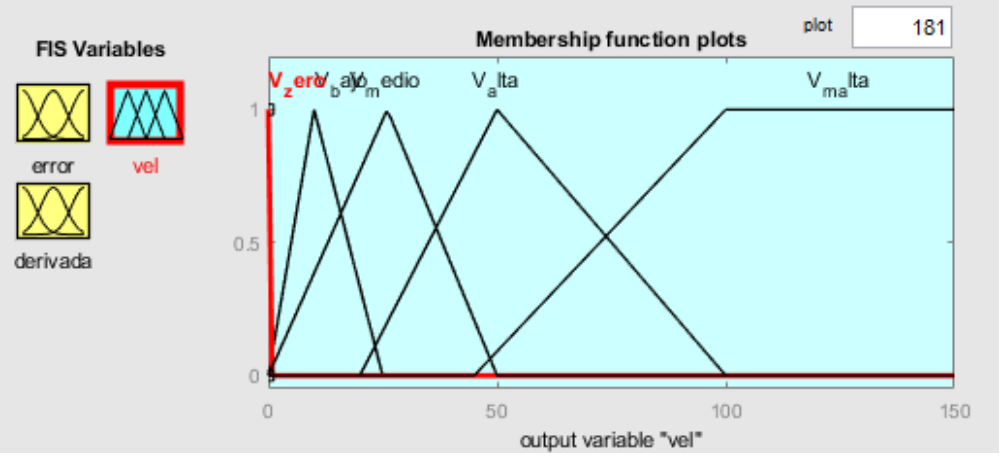


Figura 58-4. Etiquetas lingüísticas de la variable de salida

Realizado por: Caiza, A; Morales, A. 2020

En la tabla 5-4 se muestra las funciones de membresía y el rango de cada conjunto para la variable de salida.

Tabla 5-4: Funciones de membresía de la variable de salida

VARIABLE DE SALIDA (velocidad motores)					
Funciones de membresía	**V_zero**	**V_bajo**	**V_medio**	**V_alta**	**V_m_alta**
Intervalo	(0, 0, 0, 0)	(0, 10, 10, 25)	(0, 26, 26, 50)	(20, 50, 50, 100)	(45, 100, 150, 150)

Fuente: (Autoría propia, 2020)

Realizado por: Caiza, A; Morales, A. 2020

Luego de obtener los conjuntos pertenecientes a cada variable entrada se realiza la formación de reglas de inferencia, el número de reglas se encuentra con todas las posibles combinaciones entre los conjuntos de las dos variables de entrada y para este controlador se obtiene un total de nueve

reglas a las cuales se le asigna una función de membresía de la variable de salida como se muestra en la tabla 6-4.

Tabla 6-4: Formulación de reglas

		Derivada del error		
		Negativo	**Zero**	**Positivo**
Error	**bajo**	V_zero	V_bajo	V_medio
	medio	V_bajo	V_medio	V_medio
	alto	V_medio	V_alta	V_m_alta

Fuente: (Autoría propia, 2020)

Realizado por: Caiza, A; Morales, A. 2020

De acuerdo la tabla anterior se forman las siguientes reglas de inferencia:

- Regla 1: si el error es bajo y la derivada del error es negativo, entonces la velocidad es V_zero.
- Regla 2: si el error es bajo y la derivada del error es zero, entonces la velocidad es V_bajo.
- Regla 3: si el error es bajo y la derivada del error es positivo, entonces la velocidad es V_medio.
- Regla 4: si el error es medio y la derivada del error es negativo, entonces la velocidad es V_bajo.
- Regla 5: si el error es medio y la derivada del error es zero, entonces la velocidad es V_medio.
- Regla 6: si el error es medio y la derivada del error es positivo, entonces la velocidad es V_medio.
- Regla 7: si el error es alto y la derivada del error es negativo, entonces la velocidad es V_medio.
- Regla 8: si el error es alto y la derivada del error es zero, entonces la velocidad es V_alta.
- Regla 9: si el error es alto y la derivada del error es positivo, entonces la velocidad es V_m_alta.

El controlador Fuzzy se estabilizó variando velocidad nominal, tiempo de muestreo, reglas de inferencia, rango y número de conjuntos de las dos variables de entrada y una variable de salida, obteniendo lo siguiente: velocidad nominal = 150RPM, tiempo de muestreo = 0,15 seg, 9 reglas de inferencia, 3 conjuntos para cada variable de entrada (error y derivada del error) de rangos de (0 a 3500) y (-10000 a 10000) respectivamente y 5 conjuntos para la variable de salida de rango de (0 a 150).

En la figura 59-4 se muestra el método del centroide que utiliza Matlab para realizar la defusificación.

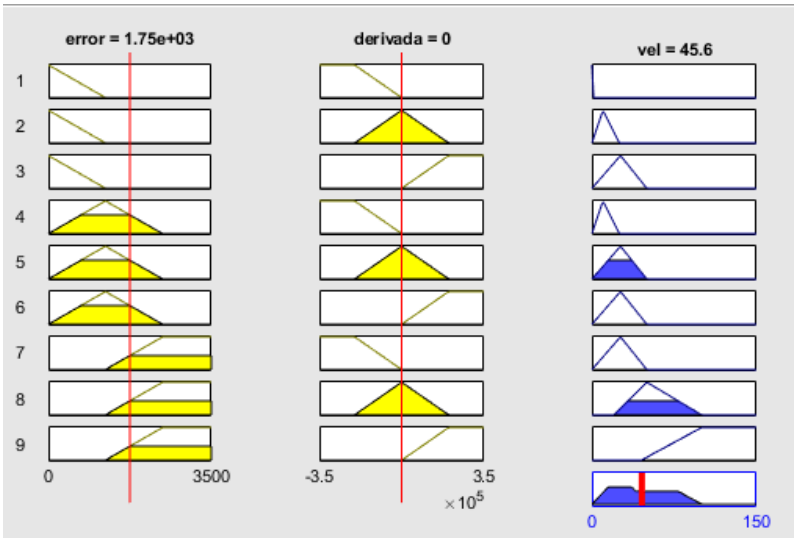


Figura 59-4. Método del centroide-defusificación, Matlab

Realizado por: Caiza, A; Morales, A. 2020

En la figura 60-4 se visualiza la superficie del controlador Fuzzy, misma que representa el resultado de la formación de las reglas de inferencia. Superficie formada en 3 dimensiones tomando como eje "x" al error, al eje "y" la derivada del error y al eje "z" la variable de salida.

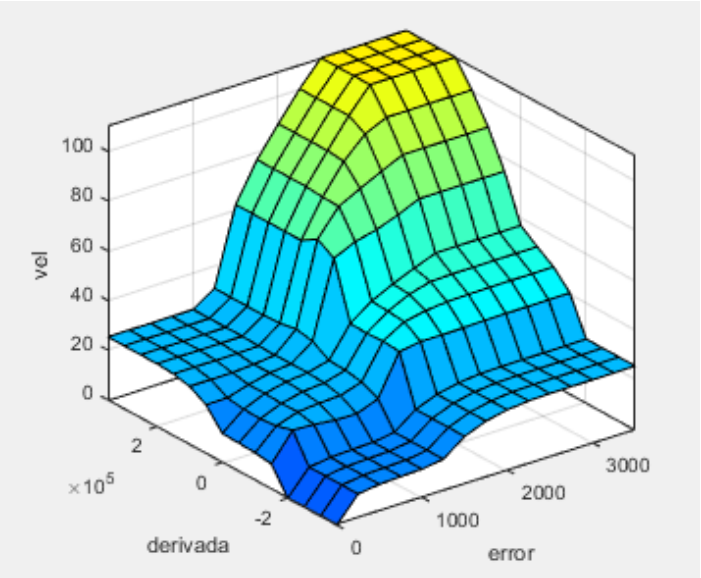


Figura 60-4. Superficie del controlador Fuzzy, Matlab

Realizado por: Caiza, A; Morales, A. 2020

Está superficie va a demostrar que la curva del error tenga una tendencia creciente aproximada a una función lineal como se muestra en la figura 61-4, representando que la velocidad de control cambia de manera proporcional con relación al error.

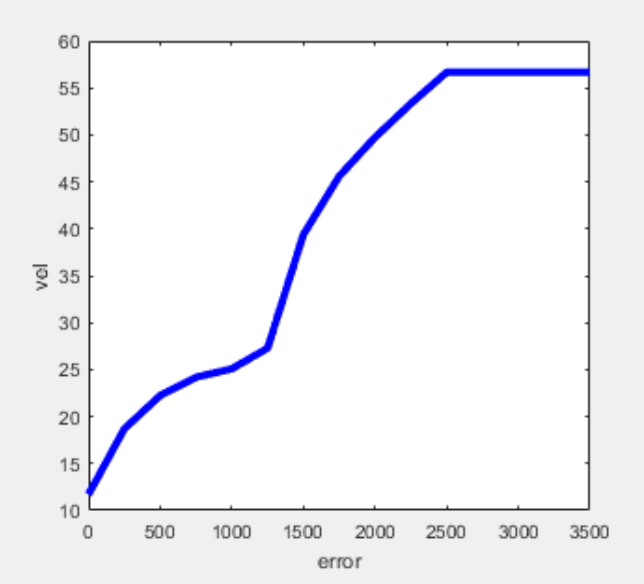


Figura 61-4. Gráfica error vs velocidad de salida, Matlab

Realizado por: Caiza, A; Morales, A. 2020

CAPÍTULO V

5. PRUEBAS Y ANÁLISIS DE RESULTADOS, CONCLUSIONES Y RECOMENDACIONES

En este capítulo se describió las pruebas de funcionamiento, se analizó los resultados obtenidos y se realizó la comparación entre el controlador PID y el controlador Fuzzy, mediante la descripción del comportamiento de las variables usando gráficas para posteriormente establecer conclusiones y redactar recomendaciones.

5.1. Pruebas y análisis de resultados

Una vez sintonizado el controlado PID y el controlador fuzzy se realizó pruebas de funcionamiento para cada controlador con la finalidad de medir el tiempo que tarda el robot seguidor de línea en recorrer 1,3 y 5 vueltas la trayectoria de la pista mostrada en la figura 62-5. Además, con la ayuda del software Matlab encontramos datos de los siguientes parámetros: set point, posición, error y velocidad de control de cada motor para un tiempo de 6 segundos con el propósito de realizar su análisis y comparación mediante una gráfica parámetro vs tiempo.

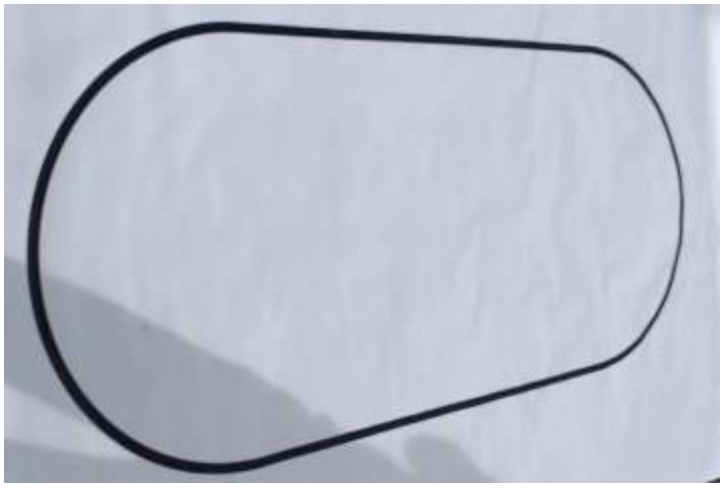


Figura 62-5. Pista

Realizado por: Caiza, A; Morales, A. 2020

En la tabla 7-5 se muestra los tiempos que tarda el robot seguidor de línea en dar un número de vueltas al compilar en el Arduino la programación de cada controlador.

Tabla 7-5: Tiempos de recorrido

		CONTROLADOR PID			CONTROLADOR FUZZY		
		1 vuelta	3 vueltas	5 vueltas	1 vuelta	3 vueltas	5 vueltas
TIEMPO (T en seg)	T1	4,81	13,86	22,23	5,22	13,84	22,47
	T2	4,88	13,63	22,10	5,13	13,84	22,73
	T3	4,85	13,64	22,07	5,04	13,64	22,85

Fuente: (Autoría propia, 2020)

Realizado por: Caiza, A; Morales, A. 2020

Los datos mostrados en la tabla 7-5 fueron medidos en una pista de 4,45 m de longitud con una velocidad nominal 150 RPM para ambos controladores, de acuerdo a los datos tomados se puede observar que el robot seguidor de línea se demora menor tiempo en recorrer la pista al estar programado con el controlador PID.

Para observar el comportamiento de la variable posición se realiza una representación gráfica lineal de los datos obtenidos en función del tiempo, como se ilustra el gráfico 1-5; donde se visualiza que la curva del controlador PID mantiene valores más cercanos a la línea de referencia mientras la curva del controlador Fuzzy muestra valores más lejanos, en cada controlador se toma como punto de referencia al set point (3500) y se visualiza de color rojo en el gráfico 11-5. Demostrando que el controlador PID tiene mejor estabilización en la posición con respecto al set point en comparación al controlador Fuzzy.

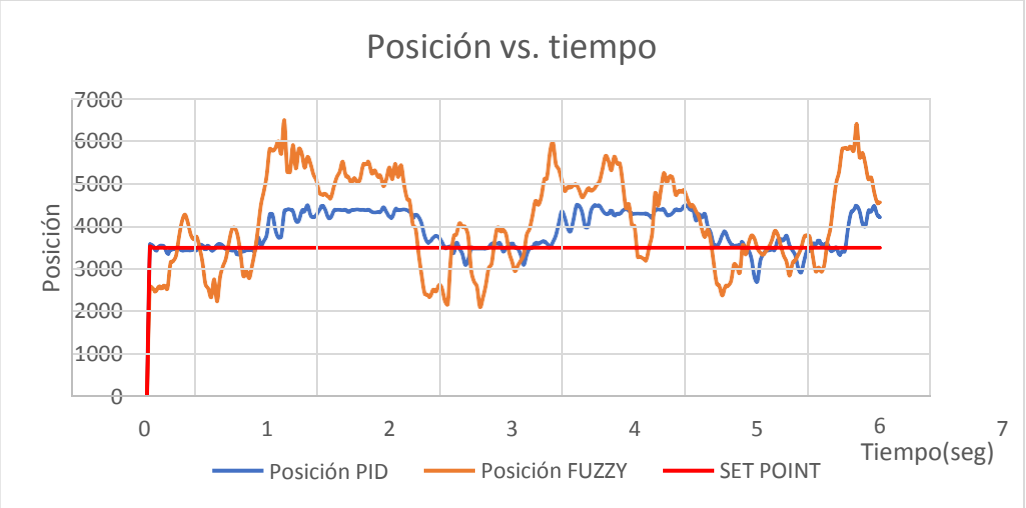


Gráfico 11-5. Gráfica posición en función del tiempo

Realizado por: Caiza, A; Morales, A. 2020

En el gráfico 12-5 se observa la representación de los datos obtenidos del error en función del tiempo para cada controlador, donde un error mínimo indica que el robot seguidor de línea se encuentra recorriendo una trayectoria en línea recta en la pista, mientras un error máximo revela que el prototipo recorre la curva dentro de la pista. Las curvas de los controladores muestran que el error del controlador PID tiene un rango menor que el controlador Fuzzy.

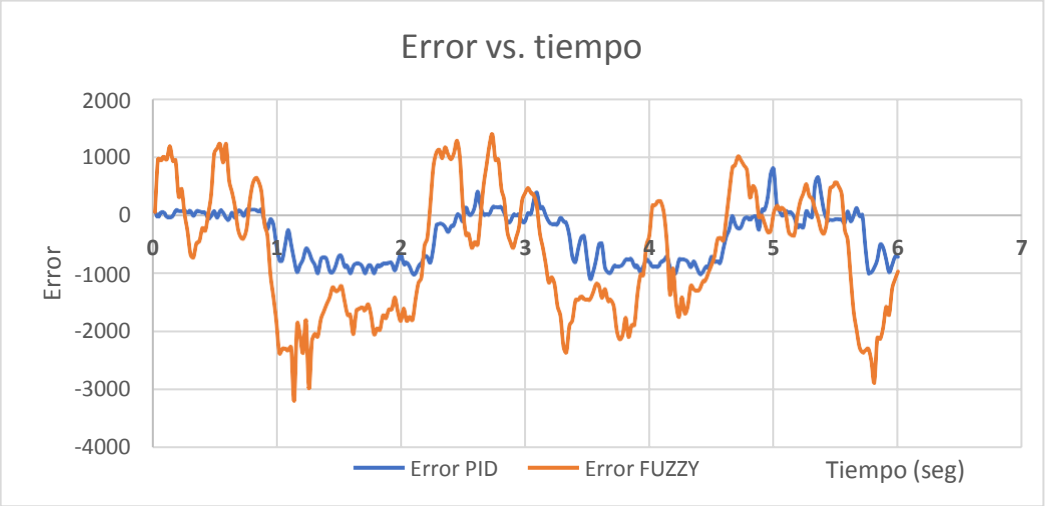


Gráfico 12-5. Gráfica del error en función del tiempo

Realizado por: Caiza, A; Morales, A. 2020

Los valores obtenidos para la velocidad de control del motor derecho están representados en el gráfico 13-5. En este gráfico se observa que al utilizar el controlador PID su velocidad de control tiene valores con picos más altos, en cambio el controlador Fuzzy adquiere valores con picos regulares tanto en una trayectoria recta como en una curva.

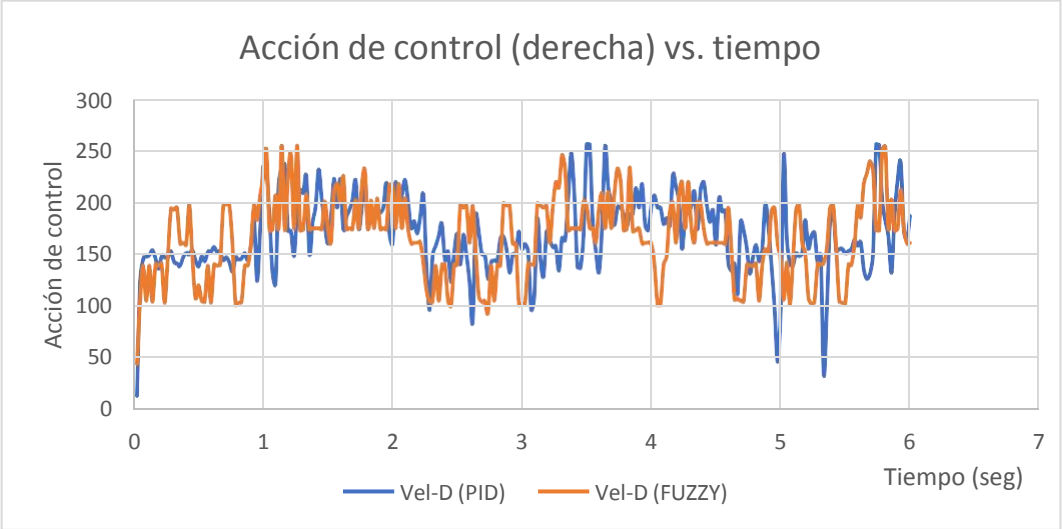


Gráfico 13-5. Gráfica de velocidad del motor derecho en función del tiempo

Realizado por: Caiza, A; Morales, A. 2020

En el gráfico 14-5 se observa la representación de los datos obtenidos para la velocidad del motor izquierdo al utilizar la programación de cada controlador, mostrando un comportamiento similar a la velocidad de control del motor derecho. Es decir, el controlador Fuzzy tiene un mejor desempeño por mantener valores de picos regulares en comparación al controlador PID que posee valores de picos más elevados.

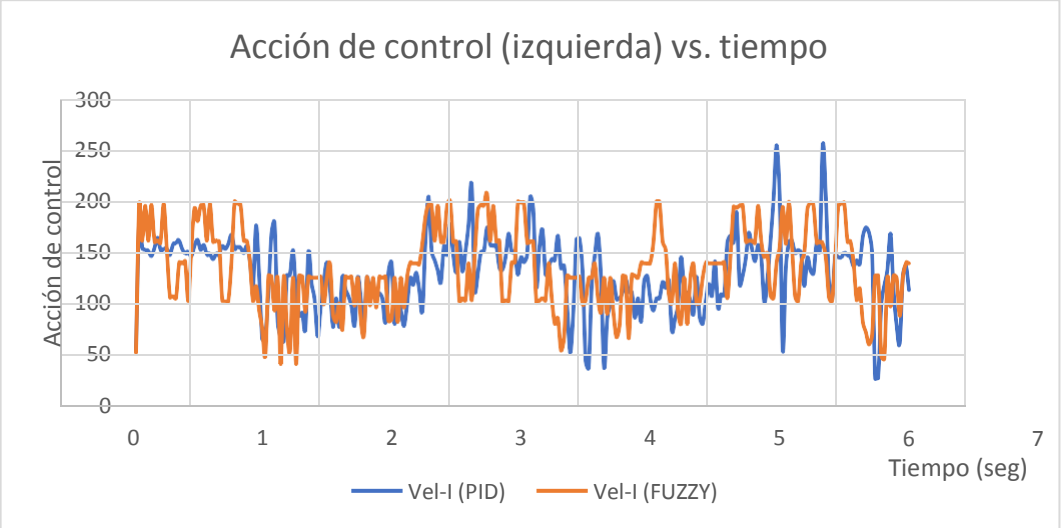


Gráfico 14-5. Gráfica de velocidad del motor izquierdo en función del tiempo

Realizado por: Caiza, A; Morales, A. 2020

5.2. Conclusiones

- El robot seguidor de línea fue construido con éxito en un modelo didáctico, manipulable y accesible utilizando un microcontrolador (Arduino) que en base a un lenguaje de programación universal permitan el funcionamiento del conjunto de los elementos mecánicos y electrónicos del prototipo; la creación y compilación de un código en el software Arduino IDE facilitaron la comparación de los parámetros usados en el controlador PID y en el controlador de lógica difusa.

- Los robots seguidores de línea son plataformas móviles que proporcionan una entrada al mundo de la robótica por su inclusión en la industria para el transportar objetos y por su alta aplicación en competencias de tipo velocista, lo cual despierta el interés a la investigación de los controladores descritos en esta propuesta mediante la aplicación de sus teorías, conceptos y métodos de sintonización con la finalidad de obtener una fiabilidad alta con movimientos más precisos a velocidades altas en el robot seguidor de línea al trasladarse sobre la trayectoria de una pista determinada en el menor tiempo posible.

- Los elementos mecánicos y electrónicos para armar el robot seguidor de línea fueron elegidos considerando características de funcionalidad, vida útil, coste y factibilidad. Para la elección del material del chasis se consideró las siguientes características: resistencia al impacto, peso, coste y maquinabilidad; en cambio para su construcción se tomó en cuenta la dimensión y ubicación de cada elemento.

- Para la sintonización de los controladores descritos en este trabajo se utilizó la técnica de prueba y error, evidenciando de forma general que el controlador Fuzzy es un control experimental a diferencia del controlador PID que se rige a un fundamento matemático. Esta prueba es un proceso sencillo para la sintonización de un robot seguidor de línea, prueba no

aplicable a la hora de lidiar con máquinas que requieren un control más minucioso. Estabilizar el controlador PID fue más sencillo porque se varía valores de sus constantes K_P, K_I, K_D, a diferencia del control Fuzzy que maneja una lógica lingüística entre sus conjuntos para formular reglas, estableciendo así que este controlador es muy parecido al controlador PID, pero menos efectivo al implementarlo en un robot seguidor de línea.

- La estabilización del controlador PID se logró variando los valores de las constantes K_P, K_I, K_D y velocidad nominal, estos valores fueron variando de manera aleatoria de acuerdo a la respuesta del robot seguidor de línea en las pruebas de funcionamiento. Obteniendo como resultado los siguientes valores: $K_P = 0{,}052$, $K_D = 0{,}171$, $K_I = 0$ y velocidad nominal = 150 RPM. Mientras el controlador Fuzzy se estabilizó variando velocidad nominal, tiempo de muestreo, reglas de inferencia, rango y número de conjuntos de las dos variables de entrada y una variable de salida, obteniendo lo siguiente: velocidad nominal = 150RPM, tiempo de muestreo = 0,15 seg, 9 reglas de inferencia, 3 conjuntos para cada variable de entrada (error y derivada del error) de rangos de (0 a 3500) y (-10000 a 10000) respectivamente y 5 conjuntos para la variable de salida de rango de (0 a 150).

- Las pruebas de funcionamiento se realizaron con el propósito de medir y almacenar datos de parámetros idénticos en cada controlador, lo que demostró que el controlador PID genera: menor tiempo de respuesta en sus algoritmos de control y menor tiempo en recorrer un número de vueltas sobre la pista; en cambio, el controlador Fuzzy presentó valores más altos, pero no tan significativos.

- El error y la posición en el controlador PID tuvieron un rango de variación pequeño, mientras al utilizar el controlador Fuzzy estos parámetros presentaron un rango de variación mayor; lo que demostró el robot seguidor de línea, presentó mayor estabilidad de control en la posición al estar programado con el controlador PID.

- La velocidad de control en los motores al utilizar el controlador PID muestra variaciones más bruscas en las señales enviadas a los motores, lo que produce mayor desgaste de los motores y del consumo de la batería; en cambio las señales de control enviadas por el controlador Fuzzy son menos bruscas al recorrer la pista percibiendo un desgaste menor de los elementos antes mencionados, optimizando el tiempo de vida útil.

5.3. Recomendaciones

- Tener un cuidado particular al momento de manipular los elementos electrónicos en el ensamblaje del robot seguidor de línea, porque se pueden averiar o dañar y esto disminuye el rendimiento total o parcial del prototipo.

- Crear el código de programación de los controladores por etapas para generar un orden y una secuencia adecuada, lo cual permite comprobar el funcionamiento de cada elemento del robot seguidor de línea e invertir menos tiempo en encontrar un error en el código.

- Verificar el voltaje de las baterías del prototipo para impedir su descarga total y evitar el fallo en sus celdas.

- Efectuar mantenimiento en los micromotores para probar que las cajas reductoras no presentan daños o averías y sus componentes se encuentren debidamente ajustados.

- Realizar las pruebas de funcionamiento a una temperatura ambiente para evitar el sobrecalentamiento en los componentes y la dilatación de los neumáticos, con iluminación incapaz de variar el contraste de los colores de la pista. También se debe limpiar las llantas y la pista después de cada prueba.

GLOSARIO

Algoritmo: Conjunto realizado de forma ordenada y explicita las operaciones sistemáticas que permiten hacer cálculos y hallar una solución para un tipo de problemas (Ortiz, 2016).

Controlador: Son sistemas que por medio de programación permiten automatizar procesos productivos y mejora los tiempos de ejecución y reduce las tasas de fallos y permite operar en ambientes peligrosos (Ortiz, 2016).

Defusificación: Es el proceso que adecua los valores difusos generados en la inferencia en valores reales que posteriormente se utilizara en el proceso de control, en la defusificacion se utilizan métodos matemáticos simples, como el método del centroide (Meza, 2003)

Emisor: Es un dispositivo que emite en este caso un led emisor de luz (Pololu, 2020).

Fusificación: Su función es convertir valores reales en valores difuso, es decir se asignan grados de pertenencia a cada una de las variables de entrada con relación a los conjuntos difusos que son previamente definidos utilizando funciones de pertenencia(Meza, 2003)

Fuzzy: Es un controlador cuyo funcionamiento se baja en la lógica difusa o también llamada lógica borrosa, esta se basa en tomar valores aleatorios, pero contextualizados y referidos entre sí (Santos, 2005)

Mamdani: es un sistema de inferencia borrosa, y es un método que transforma una variable de entrada, en una variable de salida por medio de la lógica borrosa (García et al., 2005).

PID: es un controlador cuyas siglas se establecen en parámetros proporcional, integral y derivativo su funcionamiento se basa en un mecanismo de control simultaneo por realimentación y es ampliamente usado en control industrial (Ogata et al., 2010).

Receptor: es un dispositivo que recepta o recibe una señal eléctrica y las convierte en sonidos o señales que se puedan ver y oir (Pololu, 2020).

Singletón: Es una forma de conjunto que tiene un solo valor de salida (Alzate et al., 2007).

BIBLIOGRAFIA

ABRAJAN, Christian. Diseño y construcción de un robot seguidor de línea evasor de obstáculos empleando arduino nano [en línea] (Trabajo de titulación). (Tecnología) Instituto Superior Tecnológico Nueva Vida, Quito, Ecuador. 2020. pp. 4-22. [Consulta: 3 abril 2020]. Disponible en: http://dspace.istvidanueva.edu.ec/bitstream/123456789/96/3/43.1420-ABRAJAN-ARIAS-CRISTHIAN-SANTIAGO.pdf?fbclid=IwAR2dZR7jl_UL1_V7dMQyshaFOzpzIUw8kPx50xoDOu4iY-2VOhkSjyfq9Ew

AGUILERA, Martha; et al. "Diseño y Control de Robots Móviles". Instituto Tecnológico de Nuevo Laredo [en línea], (2007), México 88250, pp. 1-7.[Consulta: 31 marzo 2020]. Disponible en: https://bit.ly/2Qzj9I6%0Ahttp://www.mecamex.net/anterior/cong02/papers/art24.pdf

ALZATE, Alfonso; LÓPEZ, Andrés; & RESTREPO, Carlos. "Control difuso de una plataforma móvil para el seguimiento de trayectorias". Scientia et Technica [en línea], (2007), Colombia *3*(35), pp. 169–174. [Consulta: 15 abril 2020]. ISSN 0122-1702. Disponible en: https://doi.org/10.22517/23447214.5391

CARRILLO, Mariano; et al. "Sistema de control y arquitectura de un robot seguidor de línea". Cultura Científica y Tecnológica [en línea], (2017), México D.F.(59), pp. 115-128. [Consulta: 30 marzo 2020]. Disponible en: http://erevistas.uacj.mx/ojs/index.php/culcyt/article/view/1570/1390

ElectronicaStore. *Seguidor de Línea TURING: Perfecto para iniciar en el mundo de la robótica | ElectronicaStore* [blog]. (2018). [Consulta: 4 abril 2020]. Disponible en: https://electronicastore.mx/producto/seguidor-de-linea-turing-perfecto-para-iniciar-en-el-mundo-de-la-robotica/?fbclid=IwAR0eqRppbPiclcIw_-1r6ttgoxEFuHMKb-sCoUkDkF0pF9gZtMxzWv0k1hw

GARCÍA, Juan; & MEDEL, José; et al. *Sistemas con lógica difusa* [en línea]. Zacatenco-México D.F.: Instituto Politéncico Nacional, (2009). [Consulta: 4 mayo 2020]. Disponible en: http://ebookcentral.proquest.com Created

LINDAO, José; & QUILAMBAQUI, Erick. Diseño y construcción e 2 Robots Sumo para las categorías pesado y liviano y un robot seguidor de línea modalidad velocista [en línea] (Trabajo de titulación). (Ingeniería) Universidad Politécnica Salesiana, Ingenierías, Ingeniería Electrónica Mención en Sistemas Computacionales. Guayaquil, Ecuador. (2014). pp. 31-25. [Consulta: 2020-03-31]. Disponible en: https://dspace.ups.edu.ec/bitstream/123456789/6554/1/UPS-GT000604.pdf?fbclid=IwAR1ds9mUufWFsyEQslivvNQJ4SSqaKbQkpl1flVXwk-

GYzuLmTljafqWxNo

MEZA, Araceli. Observadores Difusos y Control Adaptable Difuso Basado en Observadores [en línea] (Trabajo de titulación). (Maestría) Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional, México D.F. (2003). pp 20-30. [Consulta: 2020-05-10]. Disponible en: https://ctrl.cinvestav.mx/~yuw/pdf/MaTesAG.pdf?fbclid=IwAR3yZkYSq0tWN9cTOhjm7o6dmhjZOdogoV08W-hLRrQjwIJQOLuSZg4F-LM

OCAMPO, José; & MAYA, Edgar; et al. "Diseño y construccion de un prototipo Robot seguidor de linea velocista". Pistas Educativas [en línea], (2017), México D.F. *38*, pp. 208-218. [Consulta: 3 abril 2030]. ISSN 2448-847X. Disponible en: http://www.itc.mx/ojs/index.php/pistas/article/view/747/731?fbclid=IwAR3vydXF64YLO6FZSOuvJqG31GSilsR_4KxlkgfHa5Rf4WRhh7bIeW_8n7k

OGATA, Katsuhiko; & PINTO BERMÚDEZ; et al. *Ingeniería de control moderna* [en línea]. Quinta edición. Madrid-España: PEARSON EDUCACIÓN, S.A, (2010). [Consulta: 13 abril 2020]. Disponible en: https://www.u-cursos.cl/usuario/78303fe04da8e4eb340eaee09f1840b2/mi_blog/r/Ingenieria_de_Control_Moderna_Ogata_5a_ed.pdf?fbclid=IwAR3yZkYSq0tWN9cTOhjm7o6dmhjZOdogoV08W-hLRrQjwIJQOLuSZg4F-LM

ORTIZ, David. Robótica para seguimiento de líneas [en línea] (Trabajo de titulación). (Ingeniería) Universidad Politénica de Cataluña, Escuela Técnica Superior de Ingeniería de Telecomunicaciones de Barcelona, Ingeniería Electrónica. Barcelona-España. (2016). pp. 22-70. [Consulta: 2020-03-31]. Disponible en: https://upcommons.upc.edu/bitstream/handle/2117/97322/PFC?fbclid=IwAR2QSUy-Sx0K_rrgqTBWo4gzzsuXfHDSd8x7XLdNOZ3XkCKAZfjm0WySHpc

RAMÍREZ, Roberto; & REYES, Roberto. Diseño e implementación de un robot autónomo móvil usando tecnología FPGA [en línea] Trabajo de titulación). (Ingeniería) Universidad Politécnica Salesiana, Ingenierías, Ingeniería Electrónica. Guayaquil-Ecuador. (2015). pp. 9-11. [Consulta: 2020-04-03]. Disponible en: https://dspace.ups.edu.ec/bitstream/123456789/10429/1/UPS-GT001506.pdf?fbclid=IwAR2QSUy-Sx0K_rrgqTBWo4gzzsuXfHDSd8x7XLdNOZ3XkCKAZfjm0WySHpc

SANTOS, Matilde. Contribución a los métodos de sintonía de los controladores basados en lógica borrosa [en línea] (Trabajo de titulación). (Doctorado) Universidad Complutence de Madrid. Madrid-España. (2005). pp. 44-46. [Consulta: 2020-05-04]. Disponible en:

https://eprints.ucm.es/1925/1/T19935.pdf

SCHAEFER, S. *AGV WEASEL® en Intralogística | SSI Schaefer | SSI SCHAEFER* [blog]. (2016). [Consulta : 6 abril 2020]. Disponible en: https://www.ssi-schaefer.com/es-br/productos/agv-weasel-371266?fbclid=IwAR2ijayEaSdLT-6wXbQJaA-qRhFL4IAGcUhC3NcrxzLDs_ATjPQg2SX52wo

SOTO, Jonathan; & GÓMEZ, Julio. "Uso de los Conceptos Básicos de NXT-G 2.0 en la Construcción y Desarrollo de un Robot Seguidor de Línea". Lámpsakos [en línea], (2013), Colombia (9), pp.51-58. [Consulta: 6 abril 2020]. ISSN 2145-4086. Disponible en: https://core.ac.uk/reader/268187780?fbclid=IwAR0CX0F8loZUnwZTMcU1xQ0BnqXMd5yaSU4xKL6rwWSiYbtPunDnO5EP8W8

TAPIERO, Yeison. Diseño e implementación de un robot seguidor de línea de competencia para la categoría velocista [en línea] (Trabajo de titulación). (Ingeniería) Universidad de Ibagué, Ingeniería, Ingeniería Electrónica. Ibagué-Colombia. (2019). pp. 33-60. [Consulta: 2020-04-21]. Disponible en: https://repositorio.unibague.edu.co/bitstream/20.500.12313/1297/1/Trabajo?fbclid=IwAR2OvrNjWWX1UNvAAaPgDTEpuJVExpPmvg3O5hDSqdZmuLnStNEhIGVt7qM

Tdrobotica.co. *Seguidor de línea(profesional)* [blog]. (2019). Disponible en: http://aprender.tdrobotica.co/seguidor-de-lineaprofesional/

UNAM. "Robot Seguidor de Línea". (2016). México D.F., pp. 1-24. [Consulta: 9 abril 2020]. Disponible en: http://www.feriadelasciencias.unam.mx/anteriores/feria23/feria198_01_robline_robot_seguidor_de_linea.pdf

VERA, Jorge; & ALEJANDRO, Edward. Diseño e Implementación de dos Robots Seguidores de línea modalidad velocista y destreza para participaciones en Concursos de Robótica [en línea] (Trabajo de titulación). (Ingeniería) Universidad Católica Santiago de Guayaquil), Educación Técnica para el Desarrollo, Ingeniería en Telecomunicaciones. Guayaquil-Ecuador. (2016). pp. 24-86. [Consulta: 2020-04-23]. Disponible en: http://repositorio.ucsg.edu.ec/bitstream/3317/5445/1/T-UCSG-PRE-TEC-ITEL-128.pdf?fbclid=IwAR3E5FxuyPgfq2dILikSnBBi3LbwK7hwn2wXh_ttxJSP89zqNyW_xQBMBHQ

VILLALOBOS, Jojhan. "Seguidor de línea con lógica difusa". Universidad Católica de Colombia [en línea], (2014), Colombia, pp. 1-4. [Consulta: 23 abril 2020]. Disponible en: https://www.academia.edu/9495462/Seguidor_de_l%C3%ADnea_con_l%C3%B3gica_difusa?fbclid=IwAR0S8EX5GB-W2IzFo1ETJxIAmxKtibS2QM-1bxCfUScCy9Gv_3QS5S7DcDQ

ANEXOS

ANEXO A: CÓDIGO DE PROGRAMACIÓN CONTROLADOR PID

```
//LIBRERIAS
#include <QTRSensors.h>
// DEFINIR PINES
#define avanceD 9
#define avanceI 7
#define reversaD 8
#define reversaI 6
#define controlD 10
#define controlI 5
#define boton 3            // PULSADOR DEL ROBOT
#define led 4                 // LED DEL ROBOT
#define interruptor 13
// VARIABLES
uint16_t position = 0;
int16_t error;
uint16_t errorabs;      //PI
int16_t error_acumulado;//PI
uint16_t error_anterior;//PD
uint16_t sp = 3500;     //SENSOR
uint16_t vel_N = 150;   //VEL MIN
int16_t vel_D;
int16_t vel_I;
uint16_t vel_control = 0;
float kp = 0.052;
float kd = 0.171 ;
float ki = 0;
char valor;
QTRSensors qtr;
```

```
const uint8_t SensorCount = 8;
uint16_t sensorValues[SensorCount];

void setup()
{
pinMode(controlI, OUTPUT);
pinMode(reversaI, OUTPUT);
pinMode(avanceI, OUTPUT);
pinMode(avanceD, OUTPUT);
pinMode(reversaD, OUTPUT);
pinMode(controlD, OUTPUT);
pinMode(led,OUTPUT);
pinMode(boton,INPUT);
pinMode(interruptor,INPUT);

// configurar sensores
qtr.setTypeRC();
qtr.setSensorPins((const uint8_t[]){12, 11, 14, 15, 16, 17, 18,
19}, SensorCount); //( 1, 2, 3, 4, 5, 6, 7, 8)Orden de sensores
qtr.setEmitterPin(2);
delay(500);
pinMode(LED_BUILTIN, OUTPUT);
//PUESTA EN MARCHA DE MOTORES
digitalWrite(avanceI, HIGH);
digitalWrite(avanceD, HIGH);
digitalWrite(reversaI, LOW);
digitalWrite(reversaD, LOW);
analogWrite(controlD,0);
analogWrite(controlI,0);
WaitBoton();
digitalWrite(led, HIGH);
for (uint16_t i = 0; i < 100; i++)
{
```

```
qtr.calibrate();
}
digitalWrite(led, LOW);
//ARRANQUE
WaitBoton();
digitalWrite(led,HIGH);
delay(1000);
Serial.begin(9600);
}
void loop()
{
// IMPRIMIR VALOR DE POSICION
uint16_t position = qtr.readLineBlack(sensorValues);

// ACCIONES DE CONTROL
leer_posicion();
if (errorabs >= 3500)
{
frenos();
}
// PROGRAMCION DEL CONTROLADOR PID
error_acumulado = abs(error_acumulado + error);
if (error_acumulado > 1000) error_acumulado = 1000;
int proporcional = kp*errorabs;
int derivada = errorabs - error_anterior;//dividir para el
tiempo de procesamiento, por ser una derivada;si es necesario
int derivativo = kd*derivada;
int integral = ki*error_acumulado;
vel_control = proporcional + derivativo + integral; //RESULTADO
DE PID
error_anterior = errorabs;
     // ACCIONES DE CONTROL
if (error > 0)
```

```
{
vel_D = vel_N - vel_control; //DISMINUYO VEL D
vel_I = vel_N + vel_control; //AUMENTO VEL I
}
else
{
vel_D = vel_N + vel_control; //AUMENTO VEL D
vel_I = vel_N - vel_control; //DISMINUYO VEL I
}
// límites  de  velocidad
if (vel_D <= 0) vel_D = 0;
if (vel_I <= 0) vel_I = 0;
if (vel_D >= 255) vel_D = 255; //EVITAR DESBORDAMIENTO DE
VELOCIDAD
if (vel_I >= 255) vel_I = 255;
analogWrite(controlD,vel_D);
analogWrite(controlI,vel_I);
// Imprimir variables
Serial.print(sp);
Serial.print(",");
Serial.print(position);
Serial.print(",");
Serial.print(error);
Serial.print(",");
Serial.print(vel_I);
Serial.print(",");
Serial.println(vel_D);
}
//*************************************************************
//                              FUNCIONES
//*************************************************************
void WaitBoton()
{
```

```
while (digitalRead(boton));
delay(500);
}
void leer_posicion()
{
position = qtr.readLineBlack(sensorValues);
error = sp - position ;
errorabs = abs(error);
}
void frenos()
{
while(error >= 3400 )
{
//POSICION EXCESIVA A LA DERECHA
analogWrite(controlD,200); //acelero rueda derecha
digitalWrite(avanceI,HIGH); //bloqueo rueda izquierda
digitalWrite(reversaI,HIGH);
analogWrite(controlI,250);
leer_posicion();
}
while(error <= -3400)
{
//POSICION EXCESIVA A LA IZQUIERSA
analogWrite(controlI,200); //acelero rueda izquierda
digitalWrite(avanceD,HIGH); //bloqueo rueda derecha
digitalWrite(reversaD,HIGH);
analogWrite(controlD,250);
leer_posicion();
}
// Quitar el bloqueo de los motores
digitalWrite(reversaD,LOW);
digitalWrite(reversaI,LOW);
}
```

ANEXO B: CÓDIGO DE PROGRAMACIÓN CONTROLADOR FUZZY

```
//LIBRERIAS
#include <QTRSensors.h>
#include <Fuzzy.h>

// Funcion fuzzy
Fuzzy*fuzzy = new Fuzzy();

// DEFINIR PINES
#define avanceD 9
#define avanceI 7
#define reversaD 8
#define reversaI 6
#define controlD 10
#define controlI 5
#define boton 3            // PULSADOR DEL ROBOT
#define led 4               // LED DEL ROBOT
#define interruptor 13

// VARIABLES
uint16_t position = 0;
int16_t error;
uint16_t errorabs;
int16_t error_acumulado;
uint16_t error_anterior;
uint16_t sp = 3500;//SENSOR
uint16_t vel_N = 150; //VEL MIN
int16_t vel_D;
int16_t vel_I;
uint16_t vel_control = 0;
QTRSensors qtr;
const uint8_t SensorCount = 8;
```

```
uint16_t sensorValues[SensorCount];

void setup()
{
// Conjuntos Fuzzy
sistemaFuzzy();

//Configuracion pines
pinMode(controlI, OUTPUT);
pinMode(reversaI, OUTPUT);
pinMode(avanceI, OUTPUT);
pinMode(avanceD, OUTPUT);
pinMode(reversaD, OUTPUT);
pinMode(controlD, OUTPUT);
pinMode(led,OUTPUT);
pinMode(boton,INPUT);
pinMode(interruptor,INPUT);

// configuracion sensores
qtr.setTypeRC();
qtr.setSensorPins((const uint8_t[]){12, 11, 14, 15, 16, 17, 18,
19 }, SensorCount); // ( 1, 2, 3, 4, 5, 6, 7, 8) ORDEN SENSORES
qtr.setEmitterPin(2);
delay(500);

//PUESTA EN MARCHA DE MOTORES
digitalWrite(avanceI, HIGH);
digitalWrite(avanceD, HIGH);
digitalWrite(reversaI, LOW);
digitalWrite(reversaD, LOW);
analogWrite(controlD,0);
analogWrite(controlI,0);
```

```
//Calibracion de sensores
WaitBoton();
digitalWrite(led, HIGH);
for (uint16_t i = 0; i < 100; i++)
{
qtr.calibrate();
}
digitalWrite(led, LOW);
//ARRANQUE
WaitBoton();
digitalWrite(led,HIGH);
delay(1000);
Serial.begin(9600);
}
void loop()
{
uint16_t position = qtr.readLineBlack(sensorValues);
// ACCIONES DE CONTROL
leer_posicion();
if (errorabs >= 3500)
{
frenos();
}
int derivada = ((errorabs - error_anterior)/0.15);
fuzzy->setInput(1,errorabs);
fuzzy->setInput(2,derivada);
fuzzy->fuzzify();
vel_control = fuzzy->defuzzify(1);
error_anterior = errorabs;

// ACCIONES DE CONTROL
if (error > 0)
{
```

```
vel_D = vel_N - vel_control;//DISMINUYO VEL D
vel_I = vel_N + vel_control;//AUMENTO VEL I
}
else
{
vel_D = vel_N + vel_control;//AUMENTO VEL D
vel_I = vel_N - vel_control; //DISMINUYO VEL I
}

// límites  de  velocidad
if (vel_D <= 0) vel_D = 0;
if (vel_I <= 0) vel_I = 0;
if (vel_D >= 255) vel_D = 255;//EVITAR DESBORDAMIENTO DE
VELOCIDAD
if (vel_I >= 255) vel_I = 255;
analogWrite(controlD,vel_D);
analogWrite(controlI,vel_I);
Serial.print(sp);
Serial.print(",");
Serial.print(position);
Serial.print(",");
Serial.print(error);
Serial.print(",");
Serial.print(vel_I);
Serial.print(",");
Serial.println(vel_D);
}
//*************************************************************
//                          FUNCIONES
//*************************************************************

void WaitBoton()
{
while (digitalRead(boton));
```

```
delay(500);
}
void leer_posicion()
{
position = qtr.readLineBlack(sensorValues);
error = sp - position ;
errorabs = abs(error);
}
void frenos()
{
while(error >= 3400 )
{
//POSICION EXCESIVA A LA DERECHA
analogWrite(controlD,200);//acelero rueda derecha
digitalWrite(avanceI,HIGH);//bloqueo rueda izquierda
digitalWrite(reversaI,HIGH);
analogWrite(controlI,255);
leer_posicion();
}
while(error <= -3400)
{
//POSICION EXCESIVA A LA IZQUIERSA
analogWrite(controlI,200);//acelero rueda izquierda
digitalWrite(avanceD,HIGH);//bloqueo rueda derecha
digitalWrite(reversaD,HIGH);
analogWrite(controlD,255);
leer_posicion();
}
// quitar el bloqueo de los motores
digitalWrite(reversaD,LOW);
digitalWrite(reversaI,LOW);
}
void sistemaFuzzy()
```

```
{

//*************************************************************
//                      VARIABLES DE ENTRADA
//*************************************************************

//Variable de entrada 1 *error*
FuzzyInput*errorFuzzy = new FuzzyInput(1);
FuzzySet*bajo = new FuzzySet(0, 0, 0, 1225);
errorFuzzy->addFuzzySet(bajo);
FuzzySet*medio = new FuzzySet(0, 1225, 1225, 2450);
errorFuzzy->addFuzzySet(medio);
FuzzySet*alto = new FuzzySet(1225, 2450, 3500, 3500);
errorFuzzy->addFuzzySet(alto);
fuzzy->addFuzzyInput(errorFuzzy);

//Variable de entrada 2 *cambio_error*
FuzzyInput*cambio_error = new FuzzyInput(2);
FuzzySet*N = new FuzzySet(-10000, -10000, -5000, 0);
cambio_error->addFuzzySet(N);
FuzzySet*zero = new FuzzySet(-5000, 0, 0, 5000);
cambio_error->addFuzzySet(zero);
FuzzySet*P = new FuzzySet(0, 5000, 10000, 10000);
cambio_error->addFuzzySet(P);
fuzzy->addFuzzyInput(cambio_error);

//*******************VARIABLES DE SALIDA************************
//Variable de salida *VEL*
FuzzyOutput*velFuzzy = new FuzzyOutput(1);
FuzzySet*V_zero = new FuzzySet(0, 0, 0, 0);
velFuzzy->addFuzzySet(V_zero);
FuzzySet*V_bajo = new FuzzySet(0, 10, 10, 25);
velFuzzy->addFuzzySet(V_bajo);
```

```
FuzzySet*V_medio = new FuzzySet(0, 26, 26, 50);

velFuzzy->addFuzzySet(V_medio);

FuzzySet*V_alta = new FuzzySet(20, 50, 50, 100);

velFuzzy->addFuzzySet(V_alta);

FuzzySet*V_m_alta = new FuzzySet(45, 100, 150, 150);

velFuzzy->addFuzzySet(V_m_alta);

fuzzy->addFuzzyOutput(velFuzzy);


//**************************REGLAS************************

// Regla ***1***

FuzzyRuleAntecedent*si_bajo_y_N = new FuzzyRuleAntecedent();

si_bajo_y_N->joinWithAND(bajo, N);

FuzzyRuleConsequent*entonces_V_zero_a = new
FuzzyRuleConsequent();

entonces_V_zero_a->addOutput(V_zero);

FuzzyRule*regla_01 = new FuzzyRule(1,
si_bajo_y_N,entonces_V_zero_a);

fuzzy->addFuzzyRule(regla_01);


// Regla ***2***

FuzzyRuleAntecedent*si_bajo_y_zero = new FuzzyRuleAntecedent();

si_bajo_y_zero->joinWithAND(bajo, zero);

FuzzyRuleConsequent*entonces_V_bajo_b = new
FuzzyRuleConsequent();

entonces_V_bajo_b->addOutput(V_bajo);

FuzzyRule*regla_02 = new
FuzzyRule(2,si_bajo_y_zero,entonces_V_bajo_b);

fuzzy->addFuzzyRule(regla_02);


// Regla ***3***

FuzzyRuleAntecedent*si_bajo_y_P = new FuzzyRuleAntecedent();

si_bajo_y_P->joinWithAND(bajo, P);

FuzzyRuleConsequent*entonces_V_medio_c = new
FuzzyRuleConsequent();
```

```
entonces_V_medio_c->addOutput(V_medio);

FuzzyRule*regla_03 = new FuzzyRule(3,
si_bajo_y_P,entonces_V_medio_c);

fuzzy->addFuzzyRule(regla_03);


// Regla ***4***

FuzzyRuleAntecedent*si_medio_y_N = new FuzzyRuleAntecedent();

si_medio_y_N->joinWithAND(medio, N);

FuzzyRuleConsequent*entonces_V_bajo_d = new
FuzzyRuleConsequent();

entonces_V_bajo_d->addOutput(V_bajo);

FuzzyRule*regla_04 = new FuzzyRule(4,
si_medio_y_N,entonces_V_bajo_d);

fuzzy->addFuzzyRule(regla_04);


// Regla ***5***

FuzzyRuleAntecedent*si_medio_y_zero = new FuzzyRuleAntecedent();

si_medio_y_zero->joinWithAND(medio, zero);

FuzzyRuleConsequent*entonces_V_medio_e = new
FuzzyRuleConsequent();

entonces_V_medio_e->addOutput(V_medio);

FuzzyRule*regla_05 = new FuzzyRule(5,
si_medio_y_zero,entonces_V_medio_e);

fuzzy->addFuzzyRule(regla_05);


// Regla ***6***

FuzzyRuleAntecedent*si_medio_y_P = new FuzzyRuleAntecedent();

si_medio_y_P->joinWithAND(medio, P);

FuzzyRuleConsequent*entonces_V_medio_f = new
FuzzyRuleConsequent();

entonces_V_medio_f->addOutput(V_medio);

FuzzyRule*regla_06 = new FuzzyRule(6,
si_medio_y_P,entonces_V_medio_f);

fuzzy->addFuzzyRule(regla_06);
```

```
// Regla ***7***

FuzzyRuleAntecedent*si_alto_y_N = new FuzzyRuleAntecedent();

si_alto_y_N->joinWithAND(alto, N);

FuzzyRuleConsequent*entonces_V_medio_g = new
FuzzyRuleConsequent();

entonces_V_medio_g->addOutput(V_medio);

FuzzyRule*regla_07 = new FuzzyRule(7,
si_alto_y_N,entonces_V_medio_g);

fuzzy->addFuzzyRule(regla_07);


// Regla ***8***

FuzzyRuleAntecedent*si_alto_y_zero = new FuzzyRuleAntecedent();

si_alto_y_zero ->joinWithAND(alto, zero);

FuzzyRuleConsequent*entonces_V_alta_h = new
FuzzyRuleConsequent();

entonces_V_alta_h->addOutput(V_alta);

FuzzyRule*regla_08 = new FuzzyRule(8,
si_medio_y_zero,entonces_V_alta_h);

fuzzy->addFuzzyRule(regla_08);


// Regla ***9***

FuzzyRuleAntecedent*si_alto_y_P = new FuzzyRuleAntecedent();

si_alto_y_P->joinWithAND(alto, P);

FuzzyRuleConsequent*entonces_V_m_alta_i = new
FuzzyRuleConsequent();

entonces_V_m_alta_i->addOutput(V_m_alta);

FuzzyRule*regla_09 = new FuzzyRule(9,
si_alto_y_P,entonces_V_m_alta_i);

fuzzy->addFuzzyRule(regla_09);

}
```

ANEXO C: CÓDIGO DE PROGRAMACIÓN PARA IMPRIMIR VALORES (MATLAB)

```
%borrar previos
delete(instrfind({'Port'},{'COM8'}));
%crear objeto serie
s = serial('COM8','BaudRate',9600,'Terminator','CR/LF');
warning('off','MATLAB:serial:fscanf:unsuccessfulRead');
%abrir puerto
fopen(s);

%% parámetros de medidas
tmax = 30; % tiempo de captura en s
rate = 42; % capturas por segundo

%% preparar la figura
f = figure('Name','Captura');
g = axes('XLim',[0 tmax],'YLim',[10 60]);
l1 = line(nan,nan,'Color','r','LineWidth',1);
l2 = line(nan,nan,'Color','b','LineWidth',1);

xlabel('Tiempo (s)')
ylabel('error')
title('Lectrua de temperatura en tiempo real con Arduino')
grid on
hold on

%% inicializar vectores
sp = zeros(1,tmax*rate); % sp set point
position = zeros(1,tmax*rate);
error = zeros(1,tmax*rate);
vel_I = zeros(1,tmax*rate);
vel_D = zeros(1,tmax*rate);
i = 1;
t = 0;

%% ejecutar bucle cronometrado
tic
while t<(tmax)
    t = toc;
    % leer del puerto serie
    a = fscanf(s,'%d,%d,%d,%d,%d')';
    sp(i)=a(1);
    position(i)=a(2);
    error(i)=a(3);
    vel_I(i)=a(4);
    vel_D(i)=a(5);
    % dibujar en la figura
    x = linspace(0,i/rate,i);
    set(l1,'YData',sp(1:i),'XData',x);
    set(l2,'YData',error(1:i),'XData',x);
    drawnow
    % seguir
    i = i+1;
end
% resultado del cronometro
clc;
fprintf('%g s de captura a %g cap/s \n',t,i/t);
```

```
%% Limpiar la escena del crimen
fclose(s);
delete(s);
clear s;
```

ANEXO D: DATOS IMPRESOS DEL CONTROLADOR PID

Tiempo	Error	Position	sp	Vel-D	Vel-I
0,02380952	55	49	44	44	55
0,04761905	440	2950	3500	205	95
0,07142857	101	3435	3500	202	98
0,0952381	198	3390	3500	124	176
0,11904762	322	3316	3500	113	187
0,14285714	390	3114	3500	119	181
0,16666667	386	3121	3500	130	170
0,19047619	207	3240	3500	170	130
0,21428571	200	3295	3500	141	159
0,23809524	160	3359	3500	148	152
0,26190476	158	3523	3500	142	158
0,28571429	86	3420	3500	158	142
0,30952381	92	3359	3500	145	155
0,33333333	153	3356	3500	133	167
0,35714286	198	3300	3500	133	167
0,38095238	203	3308	3500	140	160
0,4047619	204	3306	3500	140	160
0,42857143	155	3359	3500	150	150
0,45238095	160	3360	3500	142	158
0,47619048	158	3350	3500	142	158
0,5	155	3359	3500	142	158
0,52380952	160	3356	3500	142	158
0,54761905	153	3350	3500	144	156
0,57142857	80	3359	3500	158	142
0,5952381	158	3523	3500	129	171
0,61904762	158	3350	3500	142	158
0,64285714	92	3359	3500	157	143
0,66666667	87	3505	3500	146	154
0,69047619	150	3436	3500	133	167
0,71428571	92	3524	3500	155	145
0,73809524	158	3356	3500	131	169
0,76190476	155	3359	3500	142	158
0,78571429	160	3524	3500	142	158
0,80952381	158	3350	3500	142	158
0,83333333	80	3359	3500	159	141
0,85714286	92	3433	3500	144	156
0,88095238	82	3350	3500	147	153
0,9047619	80	3599	3500	146	154
0,92857143	92	3611	3500	144	156
0,95238095	81	3420	3500	147	153
0,97619048	38	3456	3500	156	144
1	207	3306	3500	112	188
1,02380952	830	2845	3500	1	255
1,04761905	874	2739	3500	98	202
1,07142857	882	2552	3500	104	196
1,0952381	887	2555	3500	104	196
1,11904762	957	2553	3500	90	210
1,14285714	1004	2469	3500	90	210
1,16666667	1023	2477	3500	94	206
1,19047619	1105	2374	3500	79	221
1,21428571	1197	2296	3500	73	227
1,23809524	1330	2250	3500	59	241
1,26190476	1105	2335	3500	131	169
1,28571429	1004	2492	3500	115	185
1,30952381	1056	2444	3500	88	212
1,33333333	1172	2424	3500	71	229
1,35714286	1149	2346	3500	94	206
1,38095238	1181	2296	3500	84	216
1,4047619	1160	2304	3500	93	207
1,42857143	1154	2311	3500	91	209
1,45238095	1217	2256	3500	77	223
1,47619048	1225	2259	3500	86	214
1,5	1197	2363	3500	92	208

1,52380952	1105	2332	3500	108	192
1,54761905	1032	2473	3500	109	191
1,57142857	1259	2250	3500	47	253
1,5952381	1567	1913	3500	17	255
1,61904762	1188	2226	3500	153	147
1,64285714	957	2486	3500	140	160
1,66666667	1171	2342	3500	54	246
1,69047619	1694	1925	3500	0	255
1,71428571	1876	1610	3500	22	255
1,73809524	1023	2316	3500	242	58
1,76190476	869	2526	3500	131	169
1,78571429	837	2710	3500	112	188
1,80952381	1737	1858	3500	0	255
1,83333333	1828	1660	3500	40	255
1,85714286	1914	1558	3500	37	255
1,88095238	1694	1705	3500	99	201
1,9047619	1004	2414	3500	215	85
1,92857143	938	2533	3500	113	187
1,95238095	964	2536	3500	96	204
1,97619048	1023	2444	3500	87	213
2	1225	2259	3500	53	247
2,02380952	1385	2115	3500	51	249
2,04761905	1177	2316	3500	124	176
2,07142857	884	2585	3500	155	145
2,0952381	960	2530	3500	89	211
2,11904762	873	2593	3500	119	181
2,14285714	992	2556	3500	79	221
2,16666667	1250	2281	3500	41	255
2,19047619	1349	2137	3500	64	236
2,21428571	1142	2314	3500	126	174
2,23809524	1078	2378	3500	104	196
2,26190476	837	2661	3500	148	152
2,28571429	484	3065	3500	185	115
2,30952381	386	3294	3500	146	154
2,33333333	141	3414	3500	184	116
2,35714286	365	3233	3500	94	206
2,38095238	513	2945	3500	99	201
2,4047619	523	2893	3500	122	178
2,42857143	319	3105	3500	168	132
2,45238095	149	3508	3500	172	128
2,47619048	156	3179	3500	141	159
2,5	273	3418	3500	116	184
2,52380952	448	3057	3500	98	202
2,54761905	146	3233	3500	194	106
2,57142857	68	3508	3500	160	140
2,5952381	322	3240	3500	91	209
2,61904762	181	3174	3500	165	135
2,64285714	149	3261	3500	148	152
2,66666667	329	3283	3500	103	197
2,69047619	228	3276	3500	156	144
2,71428571	207	3340	3500	143	157
2,73809524	73	3410	3500	169	131
2,76190476	198	3300	3500	119	181
2,78571429	207	3340	3500	139	161
2,80952381	196	3350	3500	141	159
2,83333333	67	3415	3500	169	131
2,85714286	203	3456	3500	117	183
2,88095238	204	3462	3500	140	160
2,9047619	201	3300	3500	140	160
2,92857143	209	3230	3500	139	161
2,95238095	-41	3548	3500	124	176
2,97619048	-58	3581	3500	155	145
3	-51	3607	3500	151	149

3,02380952	33	3462	3500	152	148
3,04761905	92	3359	3500	136	164
3,07142857	-111	3607	3500	158	142
3,0952381	86	3350	3500	150	150
3,11904762	92	3609	3500	145	155
3,14285714	-145	3607	3500	166	134
3,16666667	128	3381	3500	146	154
3,19047619	322	3435	3500	101	199
3,21428571	365	3124	3500	125	175
3,23809524	377	3105	3500	129	171
3,26190476	498	2998	3500	105	195
3,28571429	612	2953	3500	100	200
3,30952381	656	2841	3500	109	191
3,33333333	696	2786	3500	108	192
3,35714286	859	2635	3500	79	221
3,38095238	878	2553	3500	102	198
3,4047619	903	2584	3500	100	200
3,42857143	903	2599	3500	104	196
3,45238095	975	2486	3500	88	212
3,47619048	1171	2342	3500	57	243
3,5	1257	2240	3500	71	229
3,52380952	1217	2240	3500	93	207
3,54761905	1066	2437	3500	120	180
3,57142857	1062	2442	3500	95	205
3,5952381	1072	2532	3500	94	206
3,61904762	983	2518	3500	114	186
3,64285714	1001	2593	3500	95	205
3,66666667	1254	2380	3500	42	255
3,69047619	1512	2042	3500	28	255
3,71428571	1343	2115	3500	109	191
3,73809524	1065	2397	3500	142	158
3,76190476	1041	2393	3500	100	200
3,78571429	1002	2525	3500	104	196
3,80952381	1170	2330	3500	62	238
3,83333333	1606	1948	3500	0	255
3,85714286	1635	1846	3500	61	239
3,88095238	1129	2335	3500	178	122
3,9047619	1005	2493	3500	119	181
3,92857143	1051	2445	3500	89	211
3,95238095	1098	2402	3500	85	215
3,97619048	1126	2409	3500	88	212
4	1065	2448	3500	105	195
4,02380952	982	2533	3500	113	187
4,04761905	1171	2420	3500	58	242
4,07142857	1418	2076	3500	35	255
4,0952381	1330	2209	3500	96	204
4,11904762	973	2469	3500	161	139
4,14285714	887	2589	3500	118	182
4,16666667	1071	2461	3500	64	236
4,19047619	1453	2141	3500	10	255
4,21428571	1609	1923	3500	41	255
4,23809524	1042	2399	3500	192	108
4,26190476	667	2725	3500	180	120
4,28571429	960	2652	3500	51	249
4,30952381	1126	2486	3500	64	236
4,33333333	1118	2387	3500	93	207
4,35714286	1119	2332	3500	92	208
4,38095238	1211	2383	3500	73	227
4,4047619	1211	2242	3500	88	212
4,42857143	1079	2374	3500	116	184
4,45238095	1004	2489	3500	110	190
4,47619048	1008	2492	3500	98	202
4,5	957	2530	3500	109	191

4,52380952	882	2552	3500	117	183
4,54761905	500	2882	3500	189	111
4,57142857	207	3414	3500	190	110
4,5952381	204	3395	3500	140	160
4,61904762	266	3241	3500	127	173
4,64285714	322	3238	3500	125	175
4,66666667	329	3179	3500	132	168
4,69047619	322	3241	3500	135	165
4,71428571	204	3306	3500	160	140
4,73809524	204	3300	3500	140	160
4,76190476	203	3359	3500	140	160
4,78571429	87	3304	3500	165	135
4,80952381	132	3326	3500	137	163
4,83333333	203	3456	3500	128	172
4,85714286	204	3341	3500	140	160
4,88095238	198	3308	3500	141	159
4,9047619	203	3308	3500	140	160
4,92857143	204	3300	3500	140	160
4,95238095	198	3359	3500	141	159
4,97619048	160	3356	3500	148	152
5	158	3350	3500	142	158
5,02380952	155	3359	3500	142	158
5,04761905	158	3356	3500	142	158
5,07142857	82	3350	3500	158	142
5,0952381	160	3432	3500	129	171
5,11904762	158	3523	3500	142	158
5,14285714	80	3420	3500	159	141
5,16666667	61	3431	3500	150	150
5,19047619	141	3523	3500	130	170
5,21428571	86	3350	3500	155	145
5,23809524	160	3359	3500	130	170
5,26190476	158	3350	3500	142	158
5,28571429	150	3456	3500	144	156
5,30952381	160	3362	3500	141	159
5,33333333	158	3350	3500	142	158
5,35714286	80	3359	3500	159	141
5,38095238	92	3356	3500	144	156
5,4047619	158	3350	3500	131	169
5,42857143	160	3456	3500	142	158
5,45238095	157	3459	3500	142	158
5,47619048	158	3350	3500	142	158
5,5	386	3228	3500	92	208
5,52380952	858	2742	3500	26	255
5,54761905	944	2608	3500	87	213
5,57142857	878	2553	3500	116	184
5,5952381	940	2693	3500	92	208
5,61904762	944	2555	3500	101	199
5,64285714	938	2552	3500	103	197
5,66666667	882	2608	3500	114	186
5,69047619	1008	2565	3500	77	223
5,71428571	1330	2341	3500	26	255
5,73809524	1371	2042	3500	72	228
5,76190476	1188	2250	3500	120	180
5,78571429	1045	2447	3500	120	180
5,80952381	1084	2451	3500	88	212
5,83333333	1291	2332	3500	48	252
5,85714286	1421	2218	3500	55	245
5,88095238	1105	2346	3500	147	153
5,9047619	1001	2486	3500	115	185
5,92857143	1049	2492	3500	88	212
5,95238095	1119	2405	3500	81	219
5,97619048	1225	2406	3500	69	231
6	1453	2076	3500	37	255

ANEXO E: DATOS IMPRESOS DEL CONTROLADOR FUZZY

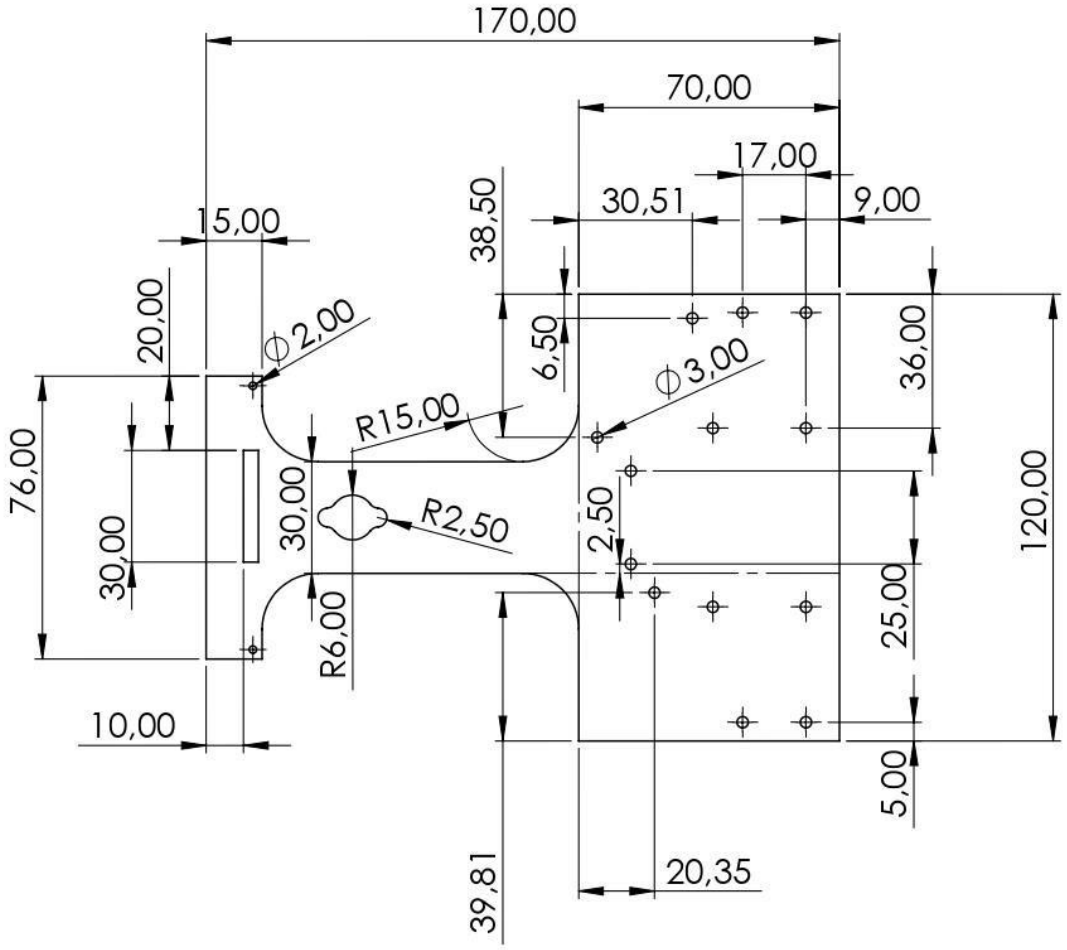
Tiempo	Error	Position	sp	Vel-D	Vel-I
0,02380952	53	56	44	44	52
0,04761905	967	2586	3500	105	195
0,07142857	942	2543	3500	139	161
0,0952381	1003	2476	3500	105	195
0,11904762	956	2586	3500	139	161
0,14285714	1187	2547	3500	104	196
0,16666667	938	2611	3500	139	161
0,19047619	926	2534	3500	139	161
0,21428571	311	3127	3500	140	160
0,23809524	454	3190	3500	103	197
0,26190476	19	3325	3500	147	153
0,28571429	-279	3773	3500	193	107
0,30952381	-680	4128	3500	193	107
0,33333333	-729	4282	3500	195	105
0,35714286	-483	4089	3500	161	139
0,38095238	-451	3806	3500	161	139
0,4047619	-219	3698	3500	160	140
0,42857143	-270	3762	3500	198	102
0,45238095	24	3503	3500	146	154
0,47619048	365	3160	3500	107	193
0,5	1059	2649	3500	120	180
0,52380952	1147	2539	3500	105	195
0,54761905	1223	2334	3500	105	195
0,57142857	903	2763	3500	139	161
0,5952381	1231	2238	3500	103	197
0,61904762	609	2804	3500	139	161
0,64285714	399	3063	3500	139	161
0,66666667	150	3243	3500	141	159
0,69047619	-228	3602	3500	196	104
0,71428571	-380	3915	3500	198	102
0,73809524	-390	3970	3500	197	103
0,76190476	-218	3800	3500	160	140
0,78571429	236	3383	3500	102	198
0,80952381	534	2831	3500	103	197
0,83333333	638	2965	3500	105	195
0,85714286	577	2782	3500	139	161
0,88095238	391	3085	3500	139	161
0,9047619	-158	3444	3500	159	141
0,92857143	-354	3944	3500	198	102
0,95238095	-1019	4500	3500	183	117
0,97619048	-1401	4822	3500	207	93
1	-1827	5199	3500	224	76
1,02380952	-2377	5821	3500	252	48
1,04761905	-2300	5777	3500	174	126
1,07142857	-2296	5845	3500	174	126
1,0952381	-2324	6003	3500	207	93
1,11904762	-2267	5702	3500	174	126
1,14285714	-3195	6500	3500	255	41
1,16666667	-1865	5311	3500	173	127
1,19047619	-2064	5288	3500	223	77
1,21428571	-2374	5914	3500	247	53
1,23809524	-1802	5359	3500	175	125
1,26190476	-2980	5824	3500	255	41
1,28571429	-2168	5721	3500	174	126
1,30952381	-2039	5377	3500	174	126
1,33333333	-2089	5637	3500	208	92
1,35714286	-1794	5473	3500	175	125
1,38095238	-1660	5221	3500	175	125
1,4047619	-1530	5080	3500	175	125
1,42857143	-1415	4810	3500	175	125
1,45238095	-1238	4745	3500	175	125
1,47619048	-1304	4765	3500	201	99
1,5	-1287	4704	3500	175	125

1,52380952	-1218	4680	3500	161	139
1,54761905	-1466	4938	3500	211	89
1,57142857	-1703	5158	3500	218	82
1,5952381	-1730	5286	3500	200	100
1,61904762	-2048	5526	3500	226	74
1,64285714	-1659	5211	3500	175	125
1,66666667	-1611	5157	3500	175	125
1,69047619	-1589	5025	3500	175	125
1,71428571	-1632	5139	3500	202	98
1,73809524	-1530	5036	3500	175	125
1,76190476	-1724	5120	3500	216	84
1,78571429	-2053	5442	3500	232	68
1,80952381	-1948	5462	3500	174	126
1,83333333	-1974	5496	3500	202	98
1,85714286	-1722	5239	3500	175	125
1,88095238	-1775	5321	3500	204	96
1,9047619	-1621	5175	3500	175	125
1,92857143	-1618	5179	3500	175	125
1,95238095	-1414	4943	3500	175	125
1,97619048	-1649	5122	3500	217	83
2	-1822	5389	3500	215	85
2,02380952	-1605	5107	3500	175	125
2,04761905	-1820	5473	3500	218	82
2,07142857	-1748	5161	3500	175	125
2,0952381	-1804	5442	3500	204	96
2,11904762	-1452	4996	3500	175	125
2,14285714	-1163	4670	3500	161	139
2,16666667	-1073	4567	3500	161	139
2,19047619	-533	4050	3500	161	139
2,21428571	-391	3947	3500	161	139
2,23809524	175	3369	3500	141	159
2,26190476	819	2876	3500	115	185
2,28571429	1068	2450	3500	104	196
2,30952381	1127	2395	3500	105	195
2,33333333	978	2359	3500	139	161
2,35714286	1169	2519	3500	105	195
2,38095238	1042	2472	3500	139	161
2,4047619	961	2625	3500	139	161
2,42857143	1067	2547	3500	105	195
2,45238095	1283	2255	3500	100	200
2,47619048	993	2171	3500	139	161
2,5	146	3240	3500	141	159
2,52380952	-323	3745	3500	198	102
2,54761905	-329	3830	3500	195	105
2,57142857	-565	4079	3500	197	103
2,5952381	-461	3992	3500	161	139
2,61904762	-511	3980	3500	197	103
2,64285714	-18	3777	3500	153	147
2,66666667	511	2973	3500	109	191
2,69047619	858	2706	3500	104	196
2,71428571	1212	2586	3500	104	196
2,73809524	1392	2108	3500	93	207
2,76190476	944	2317	3500	139	161
2,78571429	961	2616	3500	105	195
2,80952381	455	3023	3500	139	161
2,83333333	254	3122	3500	140	160
2,85714286	-280	3629	3500	198	102
2,88095238	-454	3979	3500	197	103
2,9047619	-565	3888	3500	197	103
2,92857143	-386	3923	3500	160	140
2,95238095	-195	3806	3500	160	140
2,97619048	251	3396	3500	102	198
3	375	3177	3500	102	198

3,02380952	467	2950	3500	103	197
3,04761905	383	3084	3500	140	160
3,07142857	337	3202	3500	140	160
3,0952381	152	3340	3500	141	159
3,11904762	-308	3800	3500	198	102
3,14285714	-552	3944	3500	197	103
3,16666667	-837	4251	3500	195	105
3,19047619	-1160	4603	3500	196	104
3,21428571	-1066	4506	3500	161	139
3,23809524	-1192	4690	3500	195	105
3,26190476	-1570	5087	3500	220	80
3,28571429	-1738	5165	3500	214	86
3,30952381	-2262	5748	3500	246	54
3,33333333	-2361	5981	3500	233	67
3,35714286	-1903	5461	3500	173	127
3,38095238	-1813	5344	3500	175	125
3,4047619	-1479	5088	3500	175	125
3,42857143	-1455	4832	3500	175	125
3,45238095	-1402	4921	3500	175	125
3,47619048	-1444	4915	3500	201	99
3,5	-1452	4963	3500	197	103
3,52380952	-1444	4980	3500	175	125
3,54761905	-1331	4855	3500	175	125
3,57142857	-1187	4683	3500	161	139
3,5952381	-1218	4770	3500	195	105
3,61904762	-1424	4892	3500	209	91
3,64285714	-1269	4841	3500	175	125
3,66666667	-1464	4876	3500	210	90
3,69047619	-1461	4974	3500	175	125
3,71428571	-1570	5071	3500	208	92
3,73809524	-1983	5356	3500	233	67
3,76190476	-2138	5659	3500	222	78
3,78571429	-2059	5547	3500	174	126
3,80952381	-1763	5317	3500	175	125
3,83333333	-2098	5643	3500	234	66
3,85714286	-1899	5480	3500	173	127
3,88095238	-1868	5452	3500	173	127
3,9047619	-1352	4961	3500	175	125
3,92857143	-1034	4430	3500	161	139
3,95238095	-1012	4529	3500	161	139
3,97619048	-523	4045	3500	161	139
4	-353	3975	3500	160	140
4,02380952	143	3310	3500	141	159
4,04761905	176	3285	3500	102	198
4,07142857	241	3246	3500	102	198
4,0952381	217	3224	3500	140	160
4,11904762	12	3536	3500	148	152
4,14285714	-523	3843	3500	191	109
4,16666667	-1372	4792	3500	200	100
4,19047619	-916	4488	3500	161	139
4,21428571	-1501	4872	3500	208	92
4,23809524	-1755	5258	3500	220	80
4,26190476	-1412	5036	3500	175	125
4,28571429	-1694	5179	3500	220	80
4,30952381	-1565	5125	3500	175	125
4,33333333	-1213	4744	3500	161	139
4,35714286	-1274	4831	3500	199	101
4,38095238	-1300	4814	3500	199	101
4,4047619	-1271	4832	3500	175	125
4,42857143	-1151	4651	3500	161	139
4,45238095	-1113	4494	3500	161	139
4,47619048	-985	4490	3500	161	139
4,5	-857	4357	3500	161	139

4,52380952	-711	4224	3500	161	139
4,54761905	-428	3944	3500	161	139
4,57142857	-389	3750	3500	161	139
4,5952381	-434	3929	3500	195	105
4,61904762	-47	3669	3500	156	144
4,64285714	385	3170	3500	107	193
4,66666667	794	2695	3500	107	193
4,69047619	867	2597	3500	105	195
4,71428571	1014	2376	3500	105	195
4,73809524	939	2586	3500	139	161
4,76190476	853	2609	3500	139	161
4,78571429	770	2720	3500	139	161
4,80952381	300	3114	3500	140	160
4,83333333	503	3056	3500	105	195
4,85714286	401	2913	3500	139	161
4,88095238	-33	3565	3500	155	145
4,9047619	-8	3340	3500	151	149
4,92857143	-151	3609	3500	192	108
4,95238095	-290	3792	3500	195	105
4,97619048	-264	3717	3500	160	140
5	36	3491	3500	145	155
5,02380952	151	3385	3500	106	194
5,04761905	98	3344	3500	142	158
5,07142857	121	3428	3500	101	199
5,0952381	42	3525	3500	145	155
5,11904762	-269	3683	3500	193	107
5,14285714	-344	3900	3500	198	102
5,16666667	-332	3805	3500	160	140
5,19047619	-8	3553	3500	151	149
5,21428571	239	3318	3500	108	192
5,23809524	371	3170	3500	102	198
5,26190476	535	2843	3500	105	195
5,28571429	333	3126	3500	140	160
5,30952381	275	3217	3500	140	160
5,33333333	101	3337	3500	142	158
5,35714286	-55	3495	3500	156	144
5,38095238	-258	3746	3500	194	106
5,4047619	-318	3794	3500	198	102
5,42857143	-105	3733	3500	158	142
5,45238095	437	3294	3500	106	194
5,47619048	492	2939	3500	103	197
5,5	563	3041	3500	103	197
5,52380952	490	2935	3500	139	161
5,54761905	355	3124	3500	140	160
5,57142857	-249	3650	3500	160	140
5,5952381	-427	3947	3500	197	103
5,61904762	-1058	4512	3500	185	115
5,64285714	-1648	5021	3500	218	82
5,66666667	-1971	5287	3500	228	72
5,69047619	-2272	5793	3500	240	60
5,71428571	-2357	5843	3500	231	69
5,73809524	-2326	5819	3500	174	126
5,76190476	-2296	5872	3500	174	126
5,78571429	-2510	5775	3500	251	49
5,80952381	-2887	6411	3500	254	46
5,83333333	-2107	5617	3500	174	126
5,85714286	-2129	5729	3500	203	97
5,88095238	-1928	5460	3500	173	127
5,9047619	-1574	5107	3500	175	125
5,92857143	-1720	5153	3500	212	88
5,95238095	-1269	4808	3500	175	125
5,97619048	-1106	4568	3500	161	139
6	-969	4567	3500	161	139

ANEXO F: DIMENSIONES DEL CHASIS



Nº. Lámina:	Nº. Hojas:	Sustitución:
1 de 1	1	

Email:angelmr92@hotmail.com/alexf.caiza@espoch.edu.ec

Teléfonos:0981449259/0999930461

Datos	Nombre	Firma	Fecha
Proyectó	Robot Seguidor de Línea	1425 1275	2020/10/20
Dibujó	Angel Morales Alex Caiza	1425 1275	2020/10/20
Revisó	Ing.MoreanoG.		2020/10/20
Aprobó	Ing. Moreano G.		2020/10/20

Codificación: FM-EIM-TIC-E-001-01-2020

Denominación:

CHASIS

Materiales: Vidrio acrílico

Nombre de archivo: ROBOT

ESPOCH

FACULTAD DE MECÁNICA

ESCUELA DE INGENIERÍA DE MANTIMIENTO INDUSTRIAL

Peso [*Kg*]	Tolerancia	Escala	Registro
S.D.	±0.3 [mm]	1:1	

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

DIRECCIÓN DE BIBLIOTECAS Y RECURSOS PARA EL APRENDIZAJE Y LA INVESTIGACIÓN

DBRAI

UNIDAD DE PROCESOS TÉCNICOS

REVISIÓN DE NORMAS TÉCNICAS, RESUMEN Y BIBLIOGRAFÍA

Fecha de entrega: Día / Mes / Año

INFORMACIÓN DEL AUTOR/A (S)
Nombres – Apellidos:
INFORMACIÓN INSTITUCIONAL
Facultad:
Carrera:
Título a optar:
f. Analista de Biblioteca responsable: