UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE QUITO

CARRERA:
INGENIERÍA ELECTRÓNICA

Trabajo de titulación previo a la obtención del título de:
INGENIERO ELECTRÓNICO

TEMA:
DESARROLLO DE UN CONTROLADOR ROBUSTO PARA ORIENTACIÓN Y DESPLAZAMIENTO DE UN VEHÍCULO TERRESTRE NO TRIPULADO REALIMENTADO POR VISIÓN ARTIFICIAL.

AUTOR:
NADIM JACOB TITO SALMERÓN

TUTOR:
JOSÉ ALCIDES RUMIPAMBA LÓPEZ

Quito, agosto del 2018

CESIÓN DE DERECHOS DE AUTOR

Yo, Nadim Jacob Tito Salmerón, con documento de identificación N° 220007497-5, manifiesto mi voluntad y cedo a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que soy el autor del trabajo de titulación íntitulado: DESARROLLO DE UN CONTROLADOR ROBUSTO PARA LA ORIENTACIÓN Y DESPLAZAMIENTO DE UN VEHÍCULO TERRESTRE NO TRIPULADO REALIMENTADO POR VISIÓN ARTIFICIAL. , mismo que ha sido desarrollado para optar por el título de Ingeniero Electrónico, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En aplicación a lo determinado en la Ley de Propiedad Intelectual, en mi condición de autor me reservo los derechos morales de la obra antes citada. En concordancia, se suscribo este documento en el momento que hago la entrega del trabajo final en formato impreso y digital a la Biblioteca de la Universidad Politécnica Salesiana.

Nadim Jacob Tito Salmerón
C. I. 2200074975

Quito, agosto 2018.

DECLARATORIA DE COAUTORÍA DEL DOCENTE TUTOR

Yo, declaro que bajo mi dirección y asesoría fue desarrollado el proyecto técnico, "DESARROLLO DE UN CONTROLADOR ROBUSTO PARA LA ORIENTACIÓN Y DESPLAZAMIENTO DE UN VEHÍCULO TERRESTRE NO TRIPULADO REALIMENTADO POR VISIÓN ARTIFICIAL", realizado por Nadim Jacob Tito Salmerón, obteniendo un producto que cumple con todos los requisitos estipulados por la Universidad Politécnica Salesiana para ser considerado como trabajo final de titulación.

José Alcides Rumipamba López
C.I.1803854353

Quito, agosto 2018.

DEDICATORIA

Este proyecto está dedicado para toda mi familia y como una muestra de cariño que se puede tener hacia mis padres Miguel Homero Tito Calle y Carmita Antonia Salmerón Alvarado personas nobles y sencillas, las cuales me ayudaron para que todo esto se pueda cumplir de la mejor manera y por su incondicional apoyo en todo momento. Por todo esto, hoy, me encuentro culminando mi carrera. También quiero dedicar este trabajo a la familia SALESIANA**,** para todos ellos tengo una enorme gratitud en el corazón que me impulsa a seguir hacia adelante; también dedico este trabajo a todos los que me conocen como una muestra que con perseverancia se alcanza los objetivos y que las cosas buenas no son fáciles y no llegan a la brevedad.

AGRADECIMIENTO

El agradecimiento que tengo es para muchas personas que conozco, principalmente a las que siempre estuvieron cuando más lo necesitaba, como son mis padres. Sin duda también quiero agradecer a la vida por permitirme conocer lo que es una institución universitaria a fondo, ya que en mi forma de pensar la universidad es algo que le debería pasar a todos, ya que es una experiencia que se la lleva por el resto de la vida. No me quiero olvidar de agradecer a todos los que conocí ya que al finalizar este proceso de titulación de pregrado solo se resume lo que se comparte con la gente que se conoció; y, es precisamente esto lo que nos hace una gran persona.

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

El presente trabajo expone el desarrollo que se realizó para diseñar un controlador robusto para la orientación y desplazamiento de un vehículo terrestre no tripulado realimentado por visión artificial. Para esto analizó el movimiento del robot diferencial en base a las ecuaciones cinemáticas de este y se modeló el motor de corriente directa (CD) con la ayuda del sistema de identificación de MATLAB. En base a lo anterior, se encontró las variables para el tratamiento matemático de la orientación y guiado del robot relacionadas con la velocidad angular de las ruedas y la orientación que describe el ángulo de desviación de la línea con respecto al robot móvil.

Para realizar este trabajo se dividió al mismo en las etapas que se mencionan a continuación. Como primera etapa, con la ayuda de MATLAB se identificó la planta, las variables involucradas y el modelo matemático del sistema, esto con la finalidad de caracterizar el sistema a ser controlado. Como segunda etapa se desarrolló un algoritmo para identificar una trayectoria marcada por una línea utilizando una cámara Web y una computadora programada en MATLAB con visión artificial. En la tercera etapa se diseñó el algoritmo del controlador robusto (que en este trabajo es un PID robusto) en base al ángulo de desviación entre la línea y el robot en base los parámetros característicos ITAE de (Dorf & Bishop, 2007). Como cuarta etapa se tiene la implementación del control PID robusto diseñado anteriormente para poner a prueba el vehículo no tripulado en la orientación y desplazamiento de este. En la quinta etapa se realizó un contraste del desempeño del controlador PID robusto con un PID convencional. Por último, en la sexta etapa se concluye el trabajo mencionando algunas conclusiones y recomendaciones.

ABSTRACT

The present work exposes the development that was made to design a robust controller that used in the orientation and displacement of an unmanned land vehicle fed by artificial vision. For this, he analyzed the movement of the differential robot based on its kinematic equations and the direct current (DC) motor was modeled with the help of the MATLAB identification system. Based on the above, the variables were found for the mathematical treatment of the guidance and guidance of the robot related to the angular velocity of the wheels and the orientation that describes the angle of deviation of the line with respect to the mobile robot.

To carry out this work, it was divided into the stages mentioned below. As a first step, with the help of MATLAB, the plant, the variables involved and the mathematical model of the system were identified, in order to characterize the system to be controlled. As a second stage, an algorithm was developed to identify a trajectory marked by a line using a Web camera and a computer programmed in MATLAB with artificial vision. In the third stage the algorithm of the robust controller (which in this work is a robust PID) was designed based on the angle of deviation between the line and the robot based on the characteristic ITAE parameters of (Dorf & Bishop, 2007). The fourth stage is the implementation of the robust PID control previously designed to test the unmanned vehicle in its orientation and displacement. In the fifth stage, a contrast was made on the performance of the robust PID controller with a conventional PID. Finally, in the sixth stage the work concludes mentioning some conclusions and recommendations.

INTRODUCCIÓN

Los robots móviles en la actualidad se han convertido en herramientas de gran utilidad para la vida cotidiana y para la industria. La arquitectura típica que se utiliza en varios prototipos es la de mecánica diferencial. Existen varias investigaciones que analizan este tipo de configuración de dirección desde el punto de vista matemático, con la finalidad de proporcionar métodos y algoritmos que ayuden a la orientación y desplazamiento del robot a través de entornos controlados.

De entre algunos trabajos se revisó el de (Kavuran, 2017) el cual presenta un análisis para seguir una trayectoria utilizando curvas de Bézier, para lo cual modela el robot diferencial en base a las ecuaciones cinemáticas de este. Por otro lado, el trabajo de (Dmitrii, Aleksandr, & Nikolay, 2017) expone la forma de lograr una navegación con un robot diferencial al conocer la cinemática del robot y el modelo matemático del motor CD. Así también, el trabajo de (Jitendra & Prashant, 2017) trata sobre un nuevo método para aprovechar el ángulo de desviación encontrado entre el camino y la dirección del robot y usarlo como variable de control. Por último, el estudio de (Vinod & Abraham, 2015) presenta una topología de control en cascada para seguir una trayectoria determinada con un robot diferencial. En estos últimos tres trabajos intervienen también el modelo cinemático del robot y el modelo matemático del motor CD.

El presente trabajo se desarrolló empezando por definir la planta, las variables involucradas y el modelo matemático del sistema. Luego, se programó un algoritmo para seguir una trayectoria definida por una línea, esto al visualizarla con una cámara Web y una computadora programada con visión artificial en MATLAB. Una vez que se definió la metodología para reconocer la trayectoria, se diseñó el controlador PID robusto en base al ángulo de desviación entre la trayectoria marcada por la línea y la orientación del robot móvil. Para diseñar el controlador PID robusto se utilizó los parámetros característicos ITAE de (Dorf & Bishop, 2007). A continuación, se implementó el control robusto diseñado anteriormente y se probó este sobre el vehículo no tripulado en la orientación y desplazamiento. Por último, se realizó un contraste del desempeño del controlador PID robusto con un PID convencional.

CAPÍTULO 1

ANTECEDENTES

1.1. Tema

Desarrollo de un controlador robusto para orientación y desplazamiento de un vehículo terrestre no tripulado realimentado por visión artificial

1.2. Planteamiento del Problema

El proyecto se basa en la necesidad de desarrollar un control robusto para el desplazamiento de un prototipo de vehículo móvil autónomo retroalimentado por visión artificial. Se busca identificar la planta, las variables involucradas y el modelo matemático. El desplazamiento terrestre no tripulado debe utilizar técnicas de procesamiento digital de imágenes para identificar un camino señalado por una línea. Además, debe utilizar un control robusto para la orientación y desplazamiento del sistema en un entorno controlado. Las técnicas de visión artificial deben ser desarrolladas utilizando MATLAB para el tratamiento de imágenes, donde la variable de entrada será dada por un periférico de adquisición de vídeo (cámara Web). Todo esto con el propósito final de guiar el robot dentro de su entorno, brindándole una autonomía en terreno controlado y marcado con líneas.

1.3. Justificación

En la actualidad son muy utilizados los robots de servicio que se guían ayudados por el reconocimiento de patrones y con tracción diferencial, por ello la necesidad de investigar en este campo de aplicación se vuelve interesante. La robótica móvil concerniente a vehículos terrestres se considera actualmente como tecnología avanzada que implica problemas de alta complejidad. Sus aplicaciones pueden estar dentro de las áreas de control, programación, inteligencia artificial, percepción e instrumentación. Además, sirven de base para el análisis de control en diversos campos de la industria y educación de tercer nivel. Esto último aporta con soluciones tecnológicas e innovadoras orientadas al desarrollo de mejores robots de servicio con tracción diferencial, que buscan mejorar el estilo de vida de las personas.

1.4. Objetivo General

Diseñar e implementar un controlador robusto para orientación y desplazamiento de un vehículo terrestre no tripulado realimentado por visión artificial

1.5. Objetivos Específicos

- Identificar la planta, las variables involucradas y el modelo matemático en el proceso del vehículo terrestre no tripulado.

- Utilizar la técnica del procesamiento digital de imágenes para identificar un camino segmentado por una línea.

- Diseñar e implementar un controlador robusto para orientación y desplazamiento del vehículo terrestre no tripulado.

- Contrastar el desempeño del controlador robusto con un PID convencional.

CAPÍTULO 2
MARCO TEÓRICO

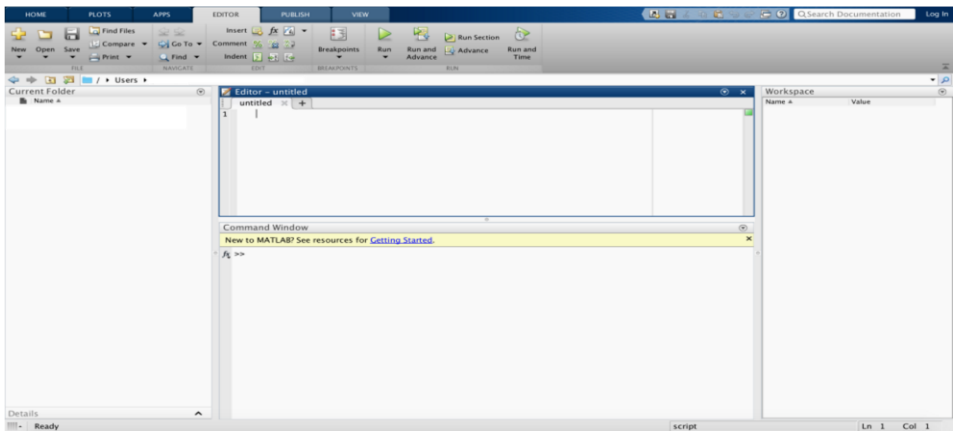
Con el objetivo de desarrollar el presente trabajo se revisaron algunos conceptos generales del funcionamiento y reconocimiento de imágenes con MATLAB. Además, fue necesario conocer acerca del modelado matemático de un robot diferencial y de un motor CD para diseñar un controlador PID convencional y un controlador PID robusto. Estos conceptos se detallan en los siguientes apartados.

2.1. Conceptos generales sobre MATLAB

2.1.1. Definición de MATLAB

MATLAB es una herramienta diseñada para personas que realizan investigación o se encuentran estudiando algún tipo de ingeniería que requiere de cálculo. Esta herramienta combina una interfaz gráfica de fácil manejo con un lenguaje propio de esta. Además, este entorno cuenta con herramientas de ayuda documentadas llamadas *toolboxes* la cual se puede apreciar en la Figura 2.1. (MATLAB, 2018)

Figura 2.1. Entorno de MATLAB



Ventana principal de MATLAB. Elaborado por: Nadim Tito

2.1.2. Procesamiento de imágenes con MATLAB

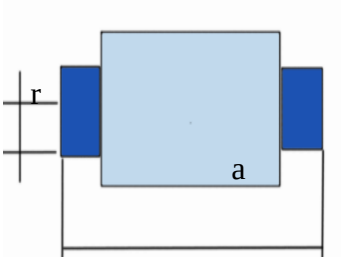
Una de las *toolboxes* especialmente permite desarrollar proyectos que tienen que ver con procesamiento de imágenes. Es decir, es necesario instalar un paquete de soporte para que permita abrir una cámara web, este paquete se denomina *Image Acquisition Toolbox Support Package for OS Generic*. (Image Acquisition Toolbox, 2018)

La herramienta *Image Processing Toolbox* permite al usuario segmentar imágenes para procesarlas de manera matemática, es decir, se las puede filtrar, transformar o quitarles parte de ellas. Esta herramienta es muy útil a la hora de trabajar con imágenes almacenadas o capturadas a partir de una cámara web. (Toolbox Image Processing, 2018)

2.2. Modelado matemático de un robot móvil diferencial

Un robot de dirección diferencial se caracteriza por usar sus ruedas de giro independientes para orientar y conducir el robot por una trayectoria determinada. En la Figura 2.2 se puede apreciar la representación gráfica de este robot, en la cual se acota el ancho de este mediante la variable "a" y el radio de las ruedas mediante la variable "r".

Figura 2.2. Robot con dirección diferencial

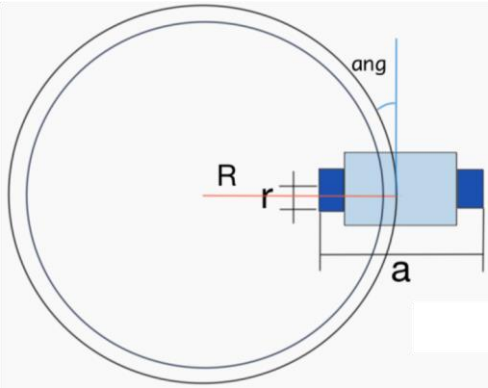


Representación gráfica de un robot diferencial. Elaborado por: Nadim Tito S

2.2.1. Cinemática de un robot diferencial

Utilizando la representación gráfica de la Figura 2.2 (Jitendra & Prashant , 2017) modelan el movimiento curvilíneo del robot diferencial en base a un radio de giro “R” proporcionado por la pista a recorrer, ver Figura 2.3.

Figura 2.3. Modelado de curvatura de un robot diferencial



Representación del movimiento de un robot diferencial. **Fuente:** (Jitendra & Prashant , 2017)

De la Figura 2.3 se puede obtener la velocidad tangencial del robot “V”, y las velocidades tangenciales de la rueda izquierda “VL” y derecha “VR”. Se observa que las velocidades antes mencionadas dependen del radio “R” de la trayectoria. Este hecho se representa en las ecuaciones (2.1), (2.2) y (2.3) según (Jitendra & Prashant , 2017).

$$V = \omega R \qquad \text{Ec. (2.1)}$$

$$VL = \omega(R - \frac{a}{2}) \qquad \text{Ec. (2.2)}$$

$$VR = \omega(R + \frac{a}{2}) \qquad \text{Ec. (2.3)}$$

Al restar (2.3) menos (2.2) para obtener la velocidad del móvil (Jitendra & Prashant, 2017) obtienen como resultado la ecuación (2.4). Así mismo, los autores consiguen la

ecuación (2.5) al interpretar que el radio de ambas ruedas es igual. Con las ecuaciones (2.4) y (2.5) los autores obtienen la ecuación (2.6), en la cual se observa la relación que existe entre la velocidad angular del robot con respecto a la velocidad angular de giro de cada una de las ruedas.

$$VR - VL = \omega\, a \qquad \text{Ec. (2.4)}$$

$$VR - VL = r\,(\omega R - \omega L) \qquad \text{Ec. (2.5)}$$

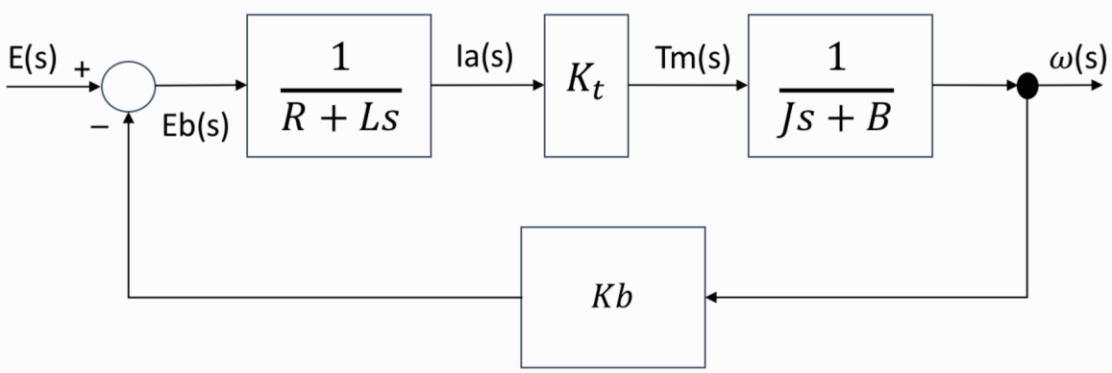
$$\omega\, a = r\,(\omega R - \omega L) \qquad \text{Ec. (2.6)}$$

2.3. Modelado matemático de un motor CD

Según (Vinod & Abraham, 2015) un motor de corriente directa (CD) se puede modelar como se muestra en el diagrama de bloques de la Figura 2.4. La función de transferencia que representa matemáticamente el diagrama d bloques mencionado se muestra en la ecuación (2.7), donde se observa la relación entre salida y entrada. Las variables de la ecuación (2.7) representan el momento de inercia del motor (J), coeficiente de fricción viscosa (B), constante de torque (Kt), constante de fuerza electromotriz (Kb), resistencia de la armadura (R) y la inductancia de la armadura (L).

$$\frac{T_m(s)}{E(s)} = \frac{K_t(Js+B)}{(R+Ls)(Js+B)+K_tK_b} \qquad \text{Ec. (2.7)}$$

Figura 2.4. Diagrama de bloques de un motor CD



Representación de un motor CD. Fuente: (Vinod & Abraham, 2015)

2.4. Controlador PID convencional y PID Robusto

Un controlador ayuda a compensar la respuesta de la planta para que no se aleje de la deseada en presencia de perturbaciones. Según (Ogata, 2010), se puede diseñar controladores P, PI, PD o PID al encontrar los parámetros de su función característica como son kp, ki y kd. A continuación, se describe el controlador PID y el controlador PID robusto utilizados en este trabajo.

2.4.1. Controlador PID

Este tipo de controlador es muy popular por su comportamiento ante la presencia de perturbaciones y por su simplicidad funcional. Un controlador PID se representa mediante la ecuación (2.8) que es característica de este tipo de controlador. En esta se puede apreciar la presencia de las constantes K1, K2 y K3 que son conocidas como los parámetros de dicho controlador. (Dorf & Bishop, 2007)

Para diseñar este tipo controladores es necesario encontrar los valores de la constante proporcional, integral y derivativa utilizando métodos de sintonización. Existes varios métodos para ajustar los valores de los parámetros de dicho controlador, entre los que se puede mencionar está el de Ziegler Nichols que se analiza en (Ogata, 2010), o el del lugar de las raíces que se analiza en (Dorf & Bishop, 2007).

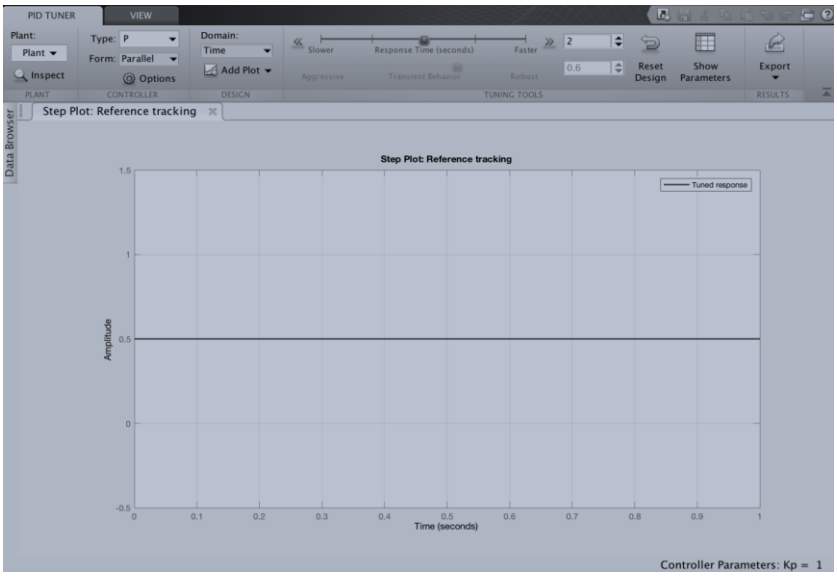
Todos los métodos al final persiguen el mismo objetivo que es determinar el valor de los parámetros K1, K2 y K3 que se muestran en la ecuación (2.8). Dichos parámetros son ajustados en base a requerimientos que se proponen en la etapa de diseño como son, por ejemplo el tiempo de asentamiento o el porcentaje de sobrelongación máxima. También se puede determinar el valor de estos parámetros utilizando la herramienta de MATLAB denominada *PID tuner*. Esta herramienta permite ajustar los valores de los parámetros de la ecuación (2.8) al observar el comportamiento que presenta la curva de salida al deslizar los controles en la interfaz gráfica de diseño. La ventana de dicha herramienta de ajuste se puede observar en la Figura 2.5.

$$G_c(s) = K_1 + \frac{K_2}{s} + K_3 s \qquad \text{Ec. (2.8)}$$

Donde,

- K1 es la ganancia proporcional
- K2 es la ganancia integral
- K3 es la ganancia derivativa.

Figura 2.5. Interfaz gráfica de la herramienta PID Tuner de MATLAB



Ventana principal de la herramienta PID Tuner de MATLAB. Elaborado por: Nadim Tito

2.4.2. Controlador PID robusto

Un controlador robusto es un compensador mejorado que intenta asegurar el comportamiento de un sistema ante la presencia de posibles perturbaciones significativas. En otras palabras, el objetivo es asegurar que un sistema sea robusto, y esto se considera de esta forma si sólo si el comportamiento en su salida es duradero, resistente y elástico.

El modelo de bloques que se considera para el diseño de este tipo de controladores se presenta en la Figura 2.6. Las perturbaciones pueden ser provocadas por cambios de parámetros, dinámicas no modeladas, retardos de tiempos no considerados en el modelado, desestabilización del punto de equilibrio, ruido filtrado en el sensor, entre otras. (Dorf & Bishop, 2007)

Figura 2.6. Diagrama de lazo cerrado para un sistema de control con PID robusto

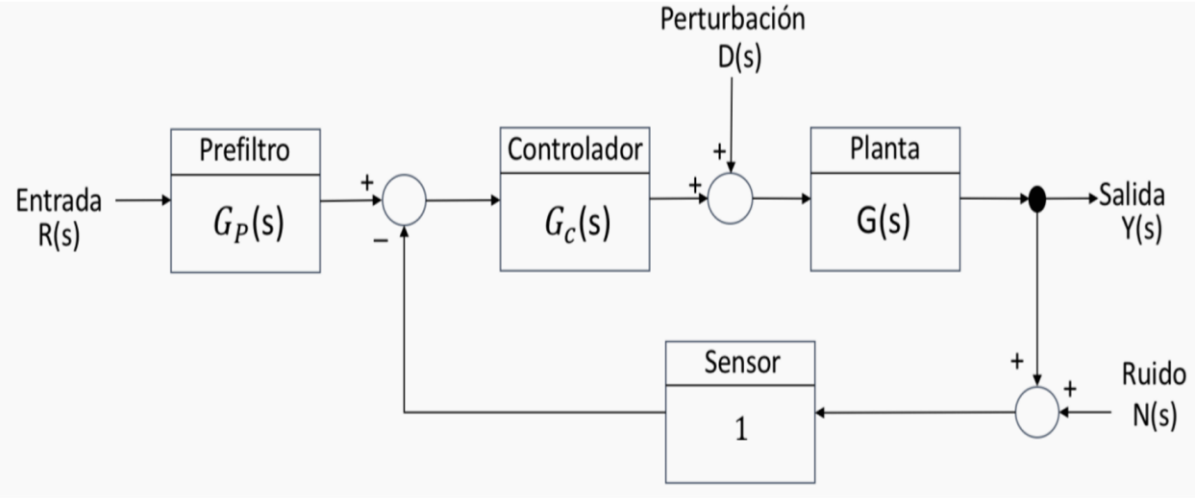


Diagrama estructural del sistema de control PID robusto. **Fuente:** (Dorf & Bishop, 2007)

Para seleccionar los valores de los parámetros de un controlador PID robusto (Dorf & Bishop, 2007) utilizan el índice de comportamiento ITAE y los coeficientes óptimos que se exponen en la Tabla 1.1.

Un índice de comportamiento según (Dorf & Bishop, 2007) es una medida cuantitativa del comportamiento de la salida de un sistema y se considera óptimo cuando alcanza un valor mínimo.

El índice de comportamiento ITAE se representa en la ecuación (2.9) donde se interpreta que es igual a la integral del tiempo multiplicada por el error absoluto.

$$ITAE = \int_0^T t|e(t)|\,dt \qquad \text{Ec. (2.9)}$$

Tabla 2.1. Coeficientes óptimos de T(s) para una entrada escalón

$s1$	$s + \omega_n$
$s2$	$s^2 + 1.4\omega_n s + {\omega_n}^2$
$s3$	$s^3 + 1.75\omega_n s^2 + 2.15{\omega_n}^2 s + {\omega_n}^3$
$s4$	$s^4 + 2.1\omega_n s^3 + 3.4{\omega_n}^2 s^2 + 2.7{\omega_n}^3 s + {\omega_n}^4$

Coeficientes óptimos basados en el criterio ITAE para una entrada escalón. Fuente: (Dorf & Bishop, 2007)

El procedimiento de diseño para encontrar los valores de los parámetros de un controlador PID robusto de (Dorf & Bishop, 2007) consiste en seguir tres pasos detallados a continuación:

- Seleccionar la frecuencia natural (ω_n) del sistema en lazo cerrado especificando el tiempo de asentamiento.
- Determinar los tres coeficientes utilizando la ecuación óptima apropiada de la Tabla 1.1 y el valor de ω_n para determinar (s).

- Determinar un prefiltro (s) de forma que la función de transferencia del sistema en lazo cerrado, T(s), no tenga ningún cero, de acuerdo como se muestra en la ecuación (2.10).

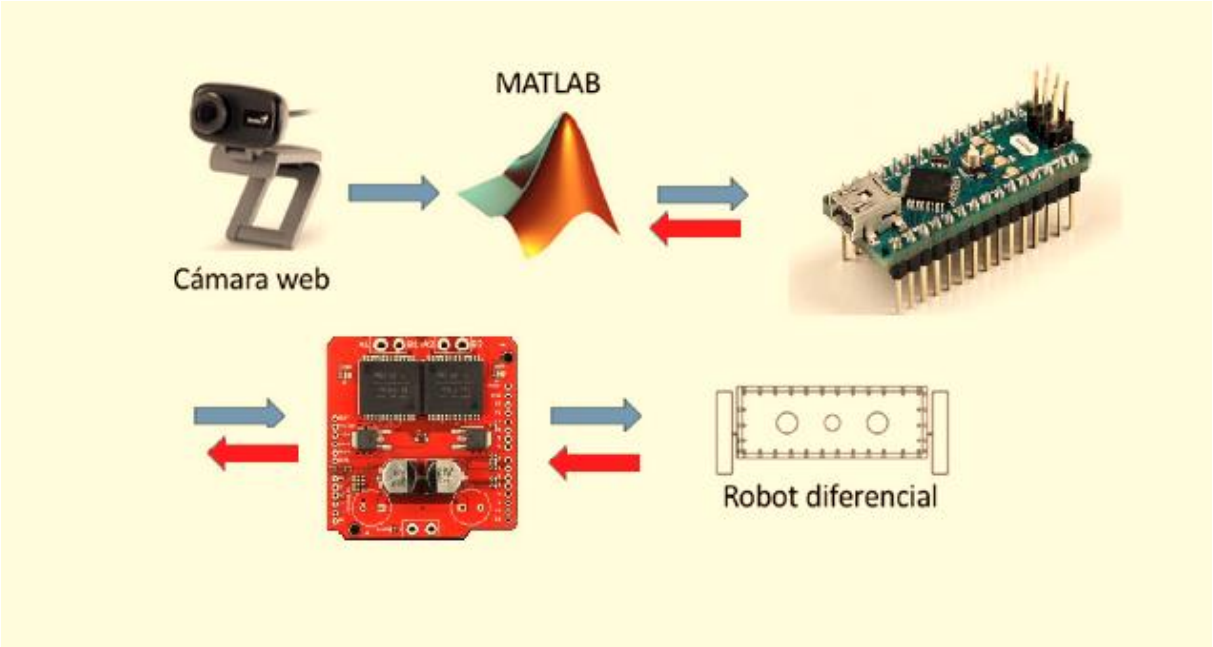
$$T(s) = \frac{Y(s)}{R(s)} = \frac{b_0}{s^n + b_{n-1}s^{n-1} + \cdots + b_1 s + b_0} \qquad \text{Ec. (2.10)}$$

CAPÍTULO 3

DISEÑO E IMPLEMENTACIÓN DEL SISTEMA

Para el diseño del sistema de control se consideró la arquitectura que se muestra en la Figura 3.1. En ella se puede observar que los componentes principales son la cámara web, la herramienta de MATLAB y el robot diferencial. Para conseguir el prototipo del robot diferencial se analizaron los requerimientos que éste deberá tener, los mismos que se muestran en los siguientes apartados.

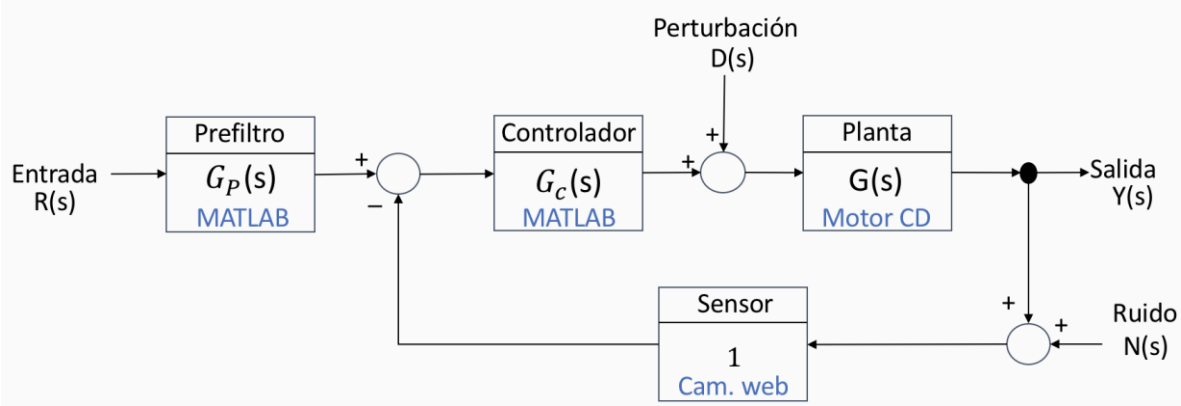
Figura 3.1. Arquitectura propuesta en el presente trabajo



Arquitectura del sistema de control propuesto. **Autor:** Nadim Tito S.

En base a la arquitectura del sistema propuesto en la Figura 3.1 se ha representado el sistema de control que se analizó en este trabajo y que se expone en la Figura 3.2. En la figura se observa que la planta en este caso son los motores del robot diferencial, el controlador y el prefiltro están diseñados en MATLAB y el sensor de realimentación del sistema es la cámara web.

Figura 3.2. Arquitectura propuesta en el presente trabajo



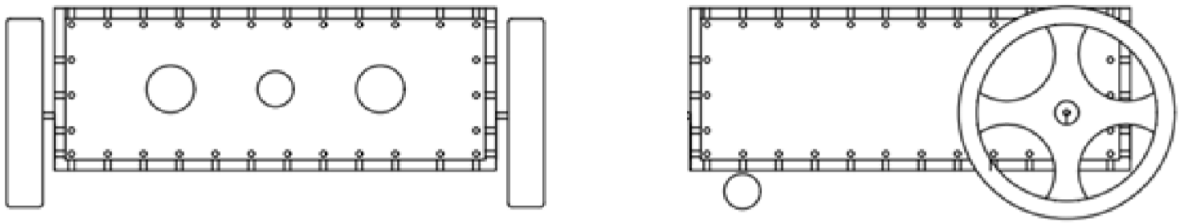
Arquitectura del sistema de control con Prefiltro (Gp) y Controlador PID (Gc). Elaborado por: Nadim Tito

3.1. Diseño e implementación de la estructura del Robot

La estructura del robot diferencial fue uno de los principales retos al momento de diseñar en base a las siguientes especificaciones. Se buscaba una estructura resistente pero liviana para montar sobre ella todos los equipos electrónicos y mecánicos.

La estructura debía soportar una computadora portátil y alojar una cámara web en la parte frontal. Además, debía proporcionar una resistencia a los golpes que pudiera tener el robot contra objetos en la fase de pruebas de funcionamiento de este. La Figura 3.3 muestra el diseño del robot que fue realizado en AUTOCAD antes de desarrollarlo.

Figura 3.3. Estructura del robot diferencial



Estructura del Robot en AUTOCAD. Elaborado por: Nadim Tito

Para conseguir el robot diferencial diseñado en la sección anterior se investigó varios componentes junto con sus características. En base a los requerimientos para cumplir

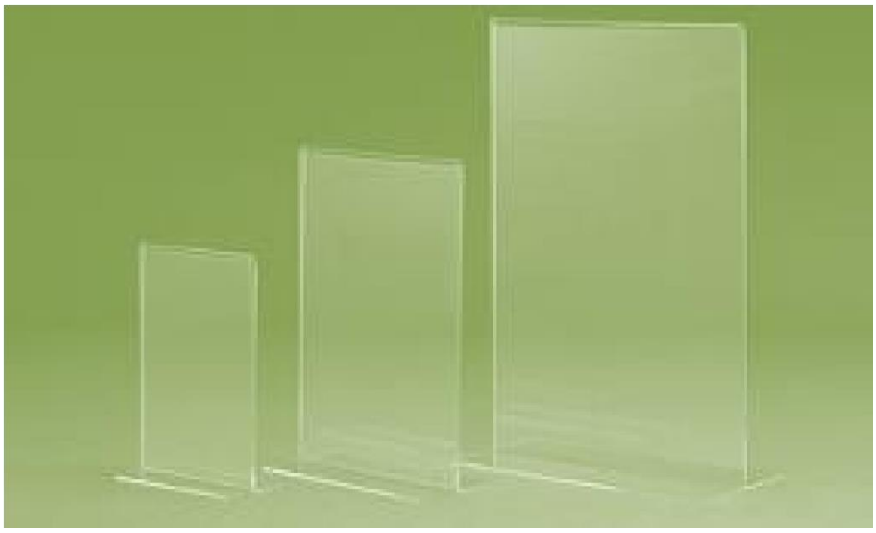
con los objetivos de peso y potencia, se seleccionó de entre ellos los elementos que se describen a continuación.

3.1.1. Acrílico

Se escogió el acrílico como material idóneo para construir la estructura del robot diferencial del presente trabajo. Esto debido a que es un termoplástico rígido, transparente, y se lo puede utilizar sin color o pigmentado en diferentes colores.

Este material fue utilizado para construir el robot ya que es de fácil trazado y de peso liviano. La densidad del acrílico está en el orden de 1190 kg/m^3, es decir, 1.19 gm/cm^3. Esto es algo menos de la mitad de la densidad del vidrio, el cual cae dentro del rango de 2400 a 2800 kg/m^3.

Figura 3.4. Acrílico Industrial



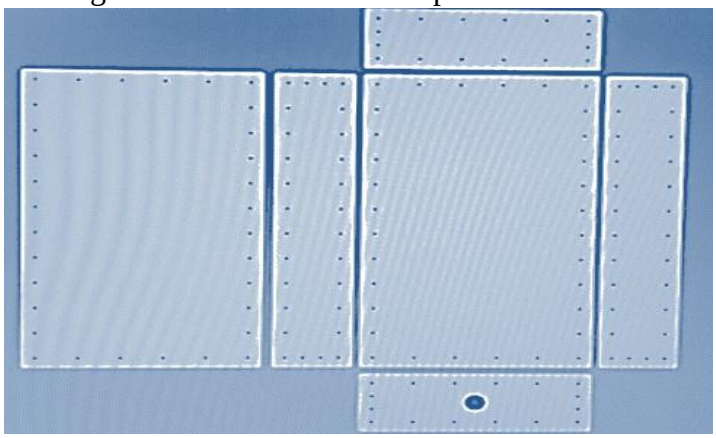
Lamina de acrílico **Fuente:**(Plastiseal. n.d.)

El acrílico empleado para el presente proyecto es industrial y de 9 mm de espesor. Para utilizarlo fue necesario cortarlo en laser con medidas establecidas. Las planchas que se obtuvieron favorecieron al peso liviano del robot lo cual benefició su desplazamiento.

En la Figura 3.5. Se muestra el diseño que se realizó con el software “Ilustration” el cual es el software que utiliza la cortadora laser para dividir las piezas diseñadas en dicho programa antes de cortar la plancha acrílica.

En el diseño se puede observar las perforaciones que se realizaron en las planchas utilizadas como soporte para el ensamble del robot con cinta perforada, tornillos milimétricos, tuercas y arandelas.

Figura 3.5. Piezas diseñadas para el robot



Diseño de la Piezas en ILUSTRATION. Elaborado por: Nadim Tito

3.1.2. Motor con caja reductora

Los motores utilizados para impulsar el robot tienen una caja reductora con corona y tornillo sin fin. La alimentación es de 12v, su marca es **KB1** y es de fabricación japonesa**.**

Este motor tiene la capacidad de desarrollar una velocidad de 20 metros por minuto en superficies planas. Cuenta con un torque de 21 Kg/m^2 proporcionado por un eje de acero con rosca en el extremo para asegurar la rueda en su posición. Dicho motor se puede observar en

Figura 3.6. Motor con caja reductora de 12V en CD

Representación del motor con caja reductora. **Fuente:** (Car multimedia zone, n.d.)

Tabla 3.2 Características del motor con caja reductora

Características:
12 V-DC, 4 Amperios con carga
Velocidad 20 m/min
Torque 21 kg/m2
Tornillo sin fin de acero
Peso 500 gr
Carcasa Plastica

Detalle de las características principales del motor **Fuente:** (Car multimedia zone, n.d.)

3.1.3. Batería Recargable de 12 V DC.

Las baterías utilizadas para la alimentación de los motorreductores son recargables de la serie BA11 y no necesitan ningún mantenimiento adicional el medio por el cual se cargan es un cargador de laptops convencional con dos borneras en su extremos.

Tienen gran capacidad de carga y una corriente nominal de 7 amperios que es adecuada para mover el robot diferencial.

Figura 3.7. Batería recargable seca

Batería recargable utilizada para impulsar el robot diferencial. **Fuente: Kapton**

3.1.4. Tarjeta Arduino Nano

La tarjeta electrónica Arduino Nano fue escogida como periférico de conexión entre la computadora portátil y el puente H que controla el giro de los motores. La comunicación con la computadora se realizó mediante conexión serial y utilizando tramas de datos tipo *String*.

Se utilizó los pines digitales para leer los datos de los encoders de cada rueda y controlar la velocidad angular de las mismas con señales de modulación de ancho pulso (PWM).

Figura 3.8. Arduino Nano

Representación de un Arduino Nano. Fuente: (Arduino Nano, 2018)

Tabla 3.4. Características del Arduino Nano

Características.
Microcontrolador Atmega328.
Voltaje de entrada 7-12 VDC.
14 pines digitales de I/O
6 entradas análogas
32K de memoria flash
Reloj de 16MHz de Velocidad

Características de un Arduino Nano. Fuente: (Arduino Nano, 2018)

3.1.5. Ruedas

Para el desplazamiento del robot se utilizaron ruedas que se ajustaron a las exigencias del proyecto. Entre las exigencias se estableció que debían ser ruedas livianas con unas características de resistencia apreciables. En el mercado se encontraron varias opciones, entre ellas varios tipos de ruedas metálicas que cubrían la necesidad de ser resistentes pero no la necesidad de ser livianas. Se encontraron otros tipos de ruedas más livianas que las metálicas en base a una construcción de plástico y espuma antideslizante es los filos. La espuma antideslizante permitió una mayor adherencia con el piso.

Figura 3.9. Ruedas de plástico con contornos antideslizantes

Fotografía de una de las ruedas implementadas en el robot diferencial. Fuente: (Finehope, n.d.)

Tabla 3.5. Características de las ruedas

Características:
Diámetro: 18 cm
Peso: 200 gr
Material Plástico
Dureza: 55
Ancho: 35-37 mm
Neumática EVA y llanta PP

Detalle de las características de las ruedas del robot diferencial. Fuente: (Finehope, n.d.)

3.1.6. Cámara Web

La cámara WEB utilizada en el robot es con una definición estándar. Se decidió en base a pruebas realizadas que un tipo común de cámara web era suficiente para reconocer una línea en el camino utilizando técnicas de procesamiento de imágenes con MATLAB. Fue ubicada en la parte frontal del robot apuntando el lente de esta hacia el camino para capturar en video la trayectoria que se recorría. Se tubo presente las posibles incidencias de la luz externa y se ubicó la cámara con un ángulo fijo de apuntando hacia el piso. La transmisión de vídeo hacia el computador portátil se la realizo mediante el cable USB incorporado en la cámara web. Ya que la cámara es estándar, el controlador de ésta se instaló automáticamente al conectarla en la PC.

Figura 3.10. Cámara web

Cámara web Genius. **Fuente:** (Genius, 2018)

Tabla 3.6. Características de cámara

Características:
Resolución MAX 720 p.
Alimentación 5 VDC.
Imagen fija
Sensibilidad a la Luz
Salida para Audio
Micrófono

Detalle de las características de cámara web Genius. Fuente: (Genius, 2018)

3.1.7. Puente H

Para manejar los motores se escogió el driver Sparkfun, ya que soporta una corriente máxima de 30 amperios que cubre la corriente nominal de los motores mencionados anteriormente. Con esta capacidad, esta tarjeta controladora soporta las corrientes picos generadas por el robot al momento de romper la inercia. El voltaje que alimenta a esta tarjeta es suplido por la batería antes mencionada.

Figura 3.11. Puente H

Driver de Motores Sparkfun Monster Moto Shield. **Fuente:** (Sparkfun, 2003)

Tabla 3.6. Características del puente H

Características
Consumo de energía max. 20W
Tamaño 53 x 43 mm
Peso 35 gr
Pico de corriente 30 Amp
L298N DC motor
Alimentación 5 – 7 v

Características del Driver de Motores Sparkfun Monster Moto Shield. Fuente: (Sparkfun, 2003)

3.1.8. Encoders

El encoder KY-040 es muy parecido a un potenciómetro en su forma física pero muy diferente en cuanto a su funcionalidad. Este elemento viene montado sobre una placa que permite conectarlo directamente con el Arduino y permite medir la posición y la velocidad angular. La Figura 3.11 muestra la apariencia física del encoder KY-040.

Figura 3.12. Sensor KY-040

Forma física del encoder rotativo. **Fuente:** (LLamas, 2016)

3.1.9. Diseño e implementación de la placa de potencia

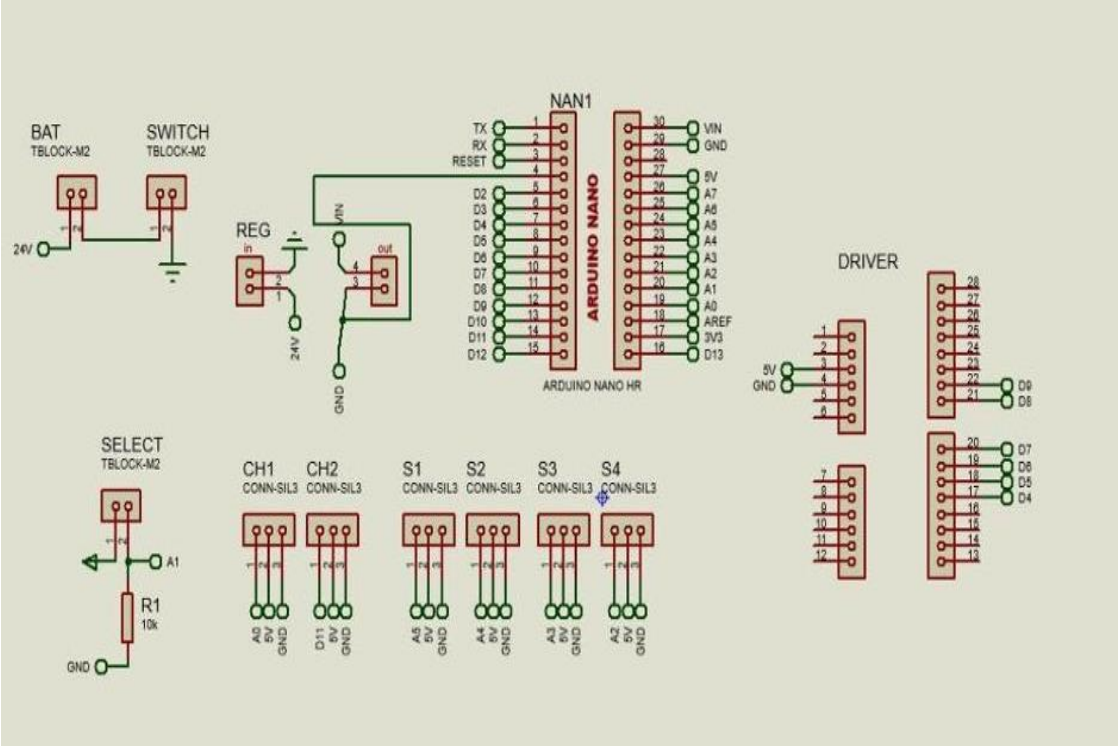
En la Figura 3.13 se muestra la simulación que se realizó de la conexión de los elementos que conforman la placa de potencia del robot. Entre ellos, se encuentran los pines de conexión del Arduino Nano, Driver de motores, batería, interruptor general, motores y encoders expuestos en los párrafos anteriores.

Una vez que se observó que las conexiones realizadas eran suficientes se obtuvo el ruteado de las pistas con la ayuda de la herramienta ARES de PROTEUS**.** La cual, usa algoritmos de ruteado que buscan la mejor ruta para cada una de las pistas evitando un solapamiento de estas, es decir, evitando que crucen unas con otras.

Al final, se obtuvo el diseño de la placa de potencia la cual fue impresa, soldada e implementada en el robot sobre la estructura del robot diferencial. Las dimensiones de la placa de acuerdo con el modelado en PROTEUS y a la ubicación de los elementos quedaron delimitadas entre 12cm x 10 cm de superficie.

Las Figuras 3.13, 3.14, 3.15 y 3.16 representan las etapas de modelado antes mencionadas.

Figura 3.13 Diseño de la placa en PROTEUS



Elaborado por: Nadim Tito

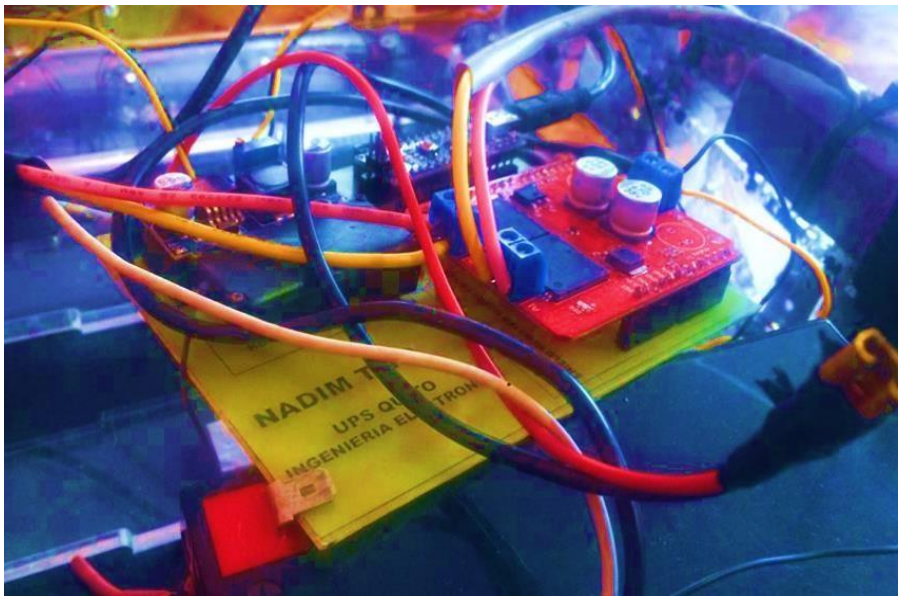
Figura 3.14. Diseño de las pistas de la placa de potencia en PROTEUS

Representación simulada de las pistas creadas con ARES **Autor:** Nadim Tito S.

Figura 3.15. Modelado 3d de la placa de potencia

Representación simulada de la placa de potencia en 3D creada con PROTEUS. Elaborado por: Nadim Tito S.

Figura 3.16. Implementación de la placa de potencia



Fotografía la placa de potencia. Elaborado por: Nadim Tito S.

3.2. Detección de una línea utilizando MATLAB

Para reconocer una línea en el piso se utilizó la herramienta de MATLAB con técnicas de procesamiento de imágenes. El objetivo se centró en reconocer una línea de color azul y descartar otros objetos a parte de este color. Además de reconocer la línea, se propuso también como objetivo encontrar el ángulo de desviación de esta con respecto a la orientación del robot diferencial. El procedimiento que se llevó a cabo para resolver estos dos objetivos secundarios se detalla a continuación en los siguientes apartados.

3.2.1. Detección de objetos de color azul

Para reconocer objetos de color azul con MATLAB se realizó los siguientes pasos. Primero, se utilizó el comando *videoinput* para crear un objeto contenedor del vídeo que captura la cámara web.

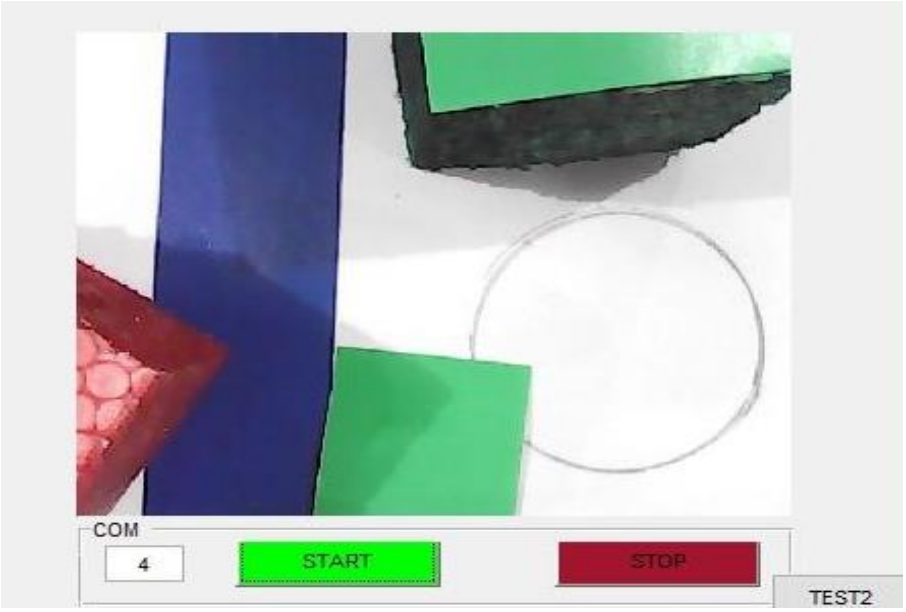
Después de esto, se almacenó una imagen con un intervalo de tiempo de medio segundo con el comando *getsnapshot* que fue colocado dentro de un bucle infinito *while*.

Una vez obtenida la imagen, dentro del bucle *while* se realiza el proceso de diferenciación de colores, y para esto se utiliza el comando *imsubtract* que detecta un color azul al colocar un valor *data(:, :, 3).* A continuación, se procesó la imagen para quitarle impurezas con un filtro de valor medio proporcionado por el comando *medfilt2*.

Seguido de esto, se binarizó la imagen utilizando el comando *im2bw,* para al final encontrarle contornos con el comando *regionprops* y dibujarlos mediante el comando *rectangle.*

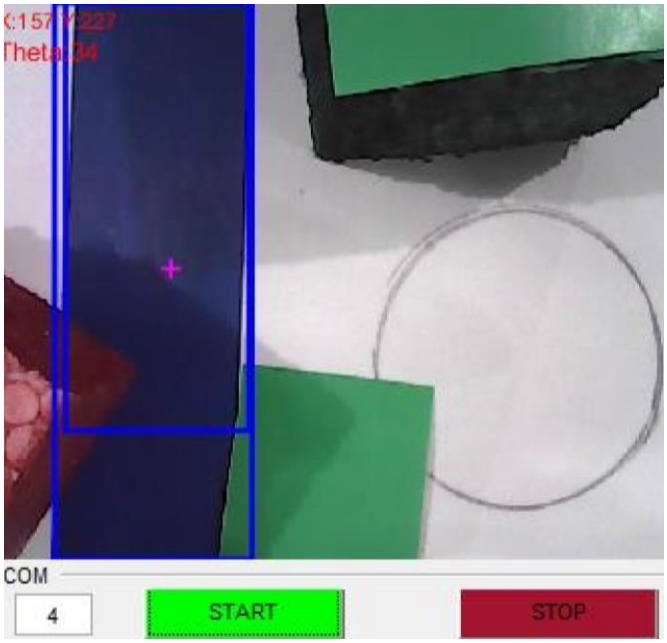
En la Figura 3.17 y 3.18 se muestran los resultados obtenidos con el algoritmo creado en MATLAB con el proceso antes mencionado. Por otro lado, la Tabla 3.8 detalla los comandos utilizados en el algoritmo de reconocimiento de imágenes de color azul.

Figura 3.17. Imagen capturada por la cámara web



Diferenciación de objetos de color azul en una imagen. Elaborado por: Nadim Tito

Figura 3.18. Reconocimiento de objetos de color azul



Contornos y centroide dibujados sobre los objetos de color azul. Elaborado por: Nadim Tito

Tabla 3.7. Comando utilizados en reconocimiento de objetos de color azul

Comando	**Descripción**
videoinput	Crea un objeto contenedor del video capturado por la cámara web
getsnashop	Captura una imagen desde el objeto de video antes mencionado
imsubtract	Diferencia colores en la imagen capturada
medfilt2	Filtra la imagen para quitarle impurezas
im2bw	Convierte la imagen a binario
regionprops	Encuentra regiones de una imagen y las características de estas
rectangle	Dibuja un rectángulo al contorno de la figura encontrada

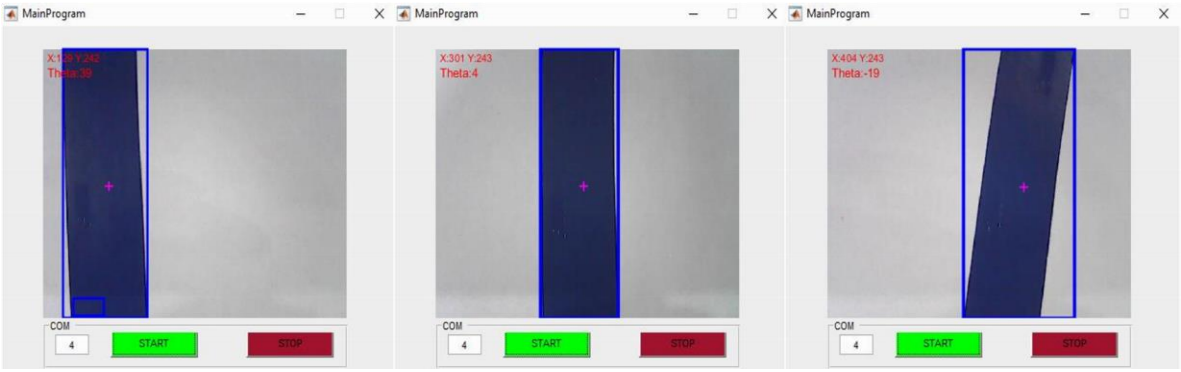
Descripción de los comandos de MATLAB utilizados. Elaborado por: Nadim Tito

En la Figura 3.18 se observa la diferenciación de objetos con el algoritmo comentado en esta sección. Se observa, además, que se identifica dos regiones a la vez y se pinta con una cruz de color purpura el centroide de la última región encontrada en la imagen.

3.2.2. Cálculo para encontrar el ángulo de desviación

Para calcular el ángulo de desviación del robot respecto de la línea de color azul se utilizó el centroide encontrado con el algoritmo detallado en el párrafo anterior. Como se observa en la Figura 3.19 el único objeto encontrado por la cámara es la línea de color azul. Se observa también a simple vista que si la línea se desvía del centro de la cámara hacia un costado el centroide de esta también se desvía. De acuerdo con este análisis, se determinó que si se puede encontrar la desviación del centroide con respecto al punto medio de la cámara se hallará mediante cálculo el ángulo de desviación del robot.

Figura 3.19. Reconocimiento de la línea azul



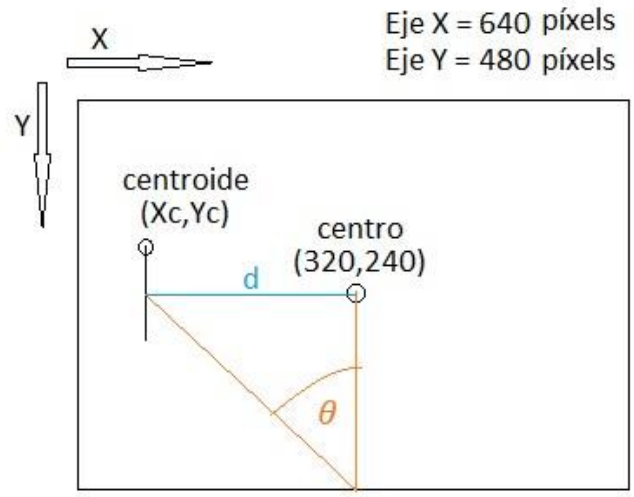
Centroide dibujado sobre la línea según la posición de esta. Elaborado por: Nadim Tito

Las dimensiones de la ventana (640x480) que fueron ajustadas al momento de abrir el vídeo definieron el punto medio en el eje “X” en 320 y en el eje “Y” es 240.

El centroide se dibuja de acuerdo con la posición actual del objeto con coordenadas cartesianas. La coordenada del eje “X” del centroide permitió encontrar la distancia “d”.

Esta distancia se utilizó para calcular el ángulo de desviación mediante geometría con la ecuación 3.1. La Figura 3.20 detalla de forma gráfica los parámetros valorados en este análisis.

Figura 3.20. Modelado matemático del ángulo de desvío



Geometría utilizada para calcular el ángulo de desvío. Elaborado por: Nadim Tito

$$\theta = \frac{(320-Xc)}{240} = \frac{d}{240} \qquad \text{Ec. (3.1)}$$

Donde, θ es el ángulo de desviación del robot respecto a la línea

- *Xc* es la coordenada en "X" del centroide

- 320 es la mitad del eje "X" de la ventana que muestra la imagen - 240 es la mitad del eje "Y" de la ventana que muestra la imagen - *d* es la distancia en el eje "X" desde el centro hasta el centroide

3.3. Modelado matemático del motor CD con caja reductora

El modelado matemático del motor CD utilizado en el presente trabajo se realizó mediante la ayuda del sistema de identificación MATLA (*System Identification*). Para ello se utilizó el encoder rotacional que se presentó en la sección 3.1.8 y el Arduino Nano explicado en la sección 3.1.4. Se le proporcionó al motor una velocidad del 45% de modulación de ancho de pulso (PWM) para realizar las mediciones de revoluciones

por segundo (RPS). En una estructura de control en lazo abierto se le proporcionó al motor una referencia de entrada de una revolución por segundo. La medida fue tomada en un período de un segundo. Los datos obtenidos con MATLAB se muestran en la Figura 3.21.

Figura 3.21. Comportamiento del motor CD en lazo abierto

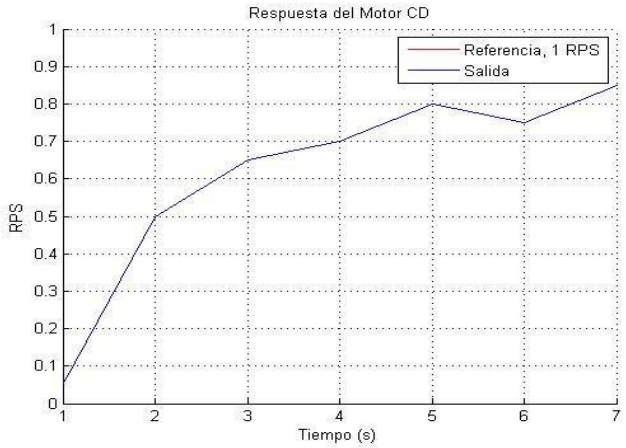
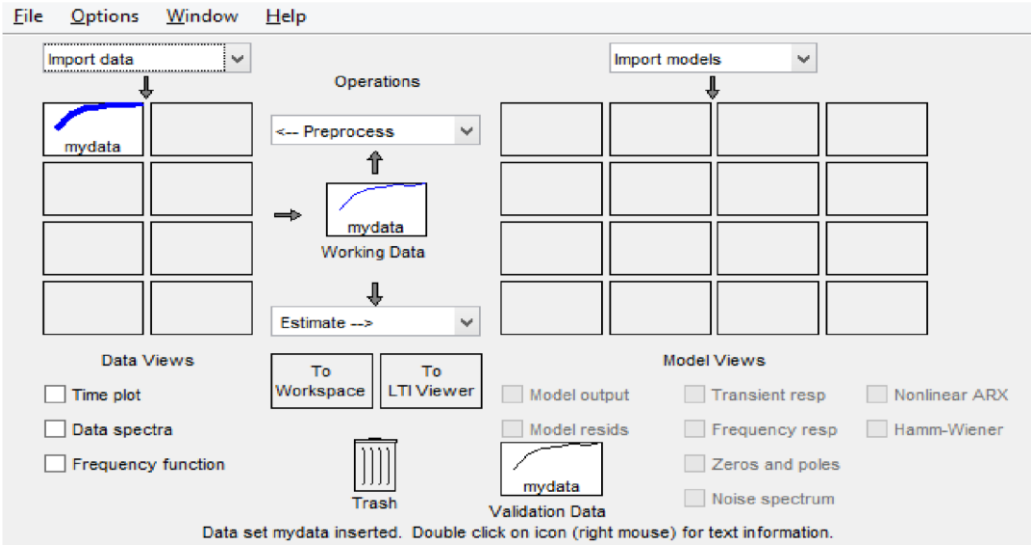


Figura obtenida con MATLAB de acuerdo con las medidas de RPS. Elaborado por: Nadim Tito

Los datos obtenidos de las medidas realizadas se ordenaron en dos vectores, uno de entrada y uno de salida. El vector de entrada se completó con la referencia de una revolución por segundo y el vector de salida se completó con los datos medidos en RPS. Luego, estos vectores se utilizaron para buscar la función de transferencia del motor con la ayuda de MATLAB, ver Figura 3.22. Mediante el sistema de identificación de MATLAB se obtuvo la función de transferencia que se muestra en la ecuación 3.2.

Cabe recalcar que no se utilizó el modelado convencional del motor DC ya fue complicado conseguir la información de las características técnicas de este, como son el momento de inercia del motor (J), coeficiente de fricción viscosa (B), constante de torque (Kt), constante de fuerza electromotriz (Kb), resistencia de la armadura (R) y la inductancia de la armadura (L).

Figura 3.22. Sistema de identificación de MATLAB



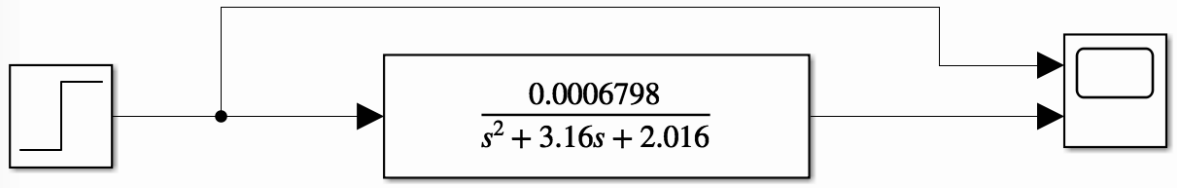
Ventana principal de MATLAB para calcular la función de transferencia. Elaborado por: Nadim Tito

$$G(s) = \frac{0.0006798}{s^2+3.16\,s+2.016} \qquad \text{Ec. (3.2)}$$

3.4. Diseño de un controlador PID

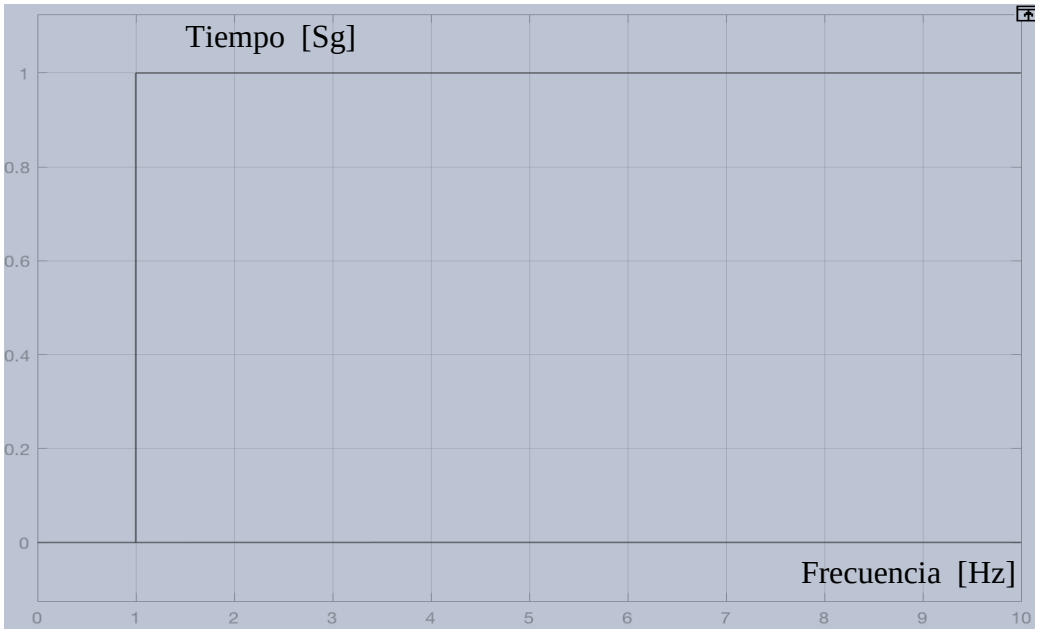
Para el diseño del controlador PID se utilizó la función de transferencia del robot que describe matemáticamente su comportamiento en la ecuación 3.2. Esta ecuación matemática permitió observar gráficamente la curva de salida al proporcionarle una señal de escalón unitario tal como se muestra en las Figuras 3.23 y 3.24.

Figura 3.23. Motor CD en lazo abierto



Función de transferencia del motor en lazo abierto en SIMULINK. **Autor:** Nadim Tito S.

Figura 3.24. Respuesta del motor CD a un escalón unitario



Respuesta del motor CD obtenida con SIMULINK. Elaborado por: Nadim Tito

Como se observa en las figuras anteriores el comportamiento del sistema en lazo abierto presenta un gran error al momento de seguir la señal de referencia.

Dicho comportamiento se corrigió con un controlador PID. Se establecieron restricciones en el tiempo de asentamiento (Ts) a 0.5 segundos y la una sobre elongación de 2% para diseñar un controlador PID. En base a estas restricciones se determinó los valores de la frecuencia natural y el factor de amortiguamiento con las ecuaciones 3.3 y 3.4 respectivamente.

$$T_s = \frac{4}{\zeta \omega_n} \qquad \text{Ec. (3.3)}$$

$$\zeta = \frac{1}{\sqrt{2}} \qquad \text{Ec. (3.4)}$$

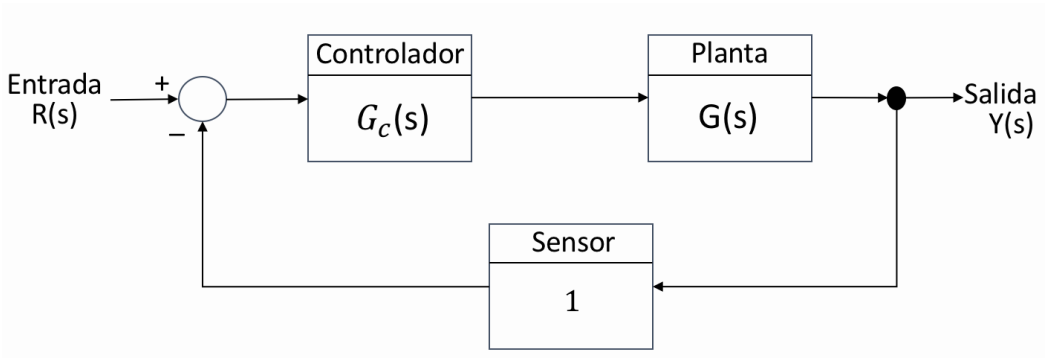
Con la ecuación 3.4 se determinó que el factor de amortiguamiento se aproximaba a 0.8. Por otro lado, con la ecuación 3.3 se encontró que el valor de la frecuencia natural es 10 Hz. Tomando como modelo el diagrama de bloques de la Figura 3.25 se determinó la

ecuación 3.6, la misma que resultó de multiplicar *G(s)* (ecuación 3.2) con *Gc(s)* (ecuación 3.5).

El controlador PID de la ecuación 3.5 se obtuvo al despejar la ecuación 2.8 de la sección 2.4.1. Al final, se realimentó el resultado de la multiplicación anterior con uno y se consiguió *Tc(s)* de la ecuación 3.6.

$$G_c(s) = \frac{K_3\, s^2 + K_1\, s + K_2}{s} \qquad \text{Ec. (3.5)}$$

Figura 3.25. Motor CD con controlador PID



Diseño del controlador PID para el motor CD. Elaborado por: Nadim Tito S.

$$T_c(s) = \frac{0.0006798\,(K_3\, s^2 + K_1\, s + K_2)}{s^3 + (3.16 + 0.0006798\,K_3)s^2 + (2.016 + 0.0006798\,K_1)s + 0.0006798\,K_2} \qquad \text{Ec. (3.6)}$$

Aplicando los criterios del índice de comportamiento ITAE de la tabla 2.1 de la sección 2.4.2 se emparejó el denominador de la ecuación 3.6 con la ecuación 3.7 de la forma como lo muestran las ecuaciones 3.8, 3.9 y 3.10. De la relación de las ecuaciones antes mencionadas se consigue los valores de los parámetros K1, K2 y K3 del controlador PID. Este proceso se observa a continuación:

$$s^3 + 1.75\omega_n s^2 + 2.15{\omega_n}^2 s + {\omega_n}^3 \qquad \text{Ec. (3.7)}$$

$$1.75\omega_n = 3.16 + 0.0006798\ K_3 \qquad \text{Ec. (3.8)}$$

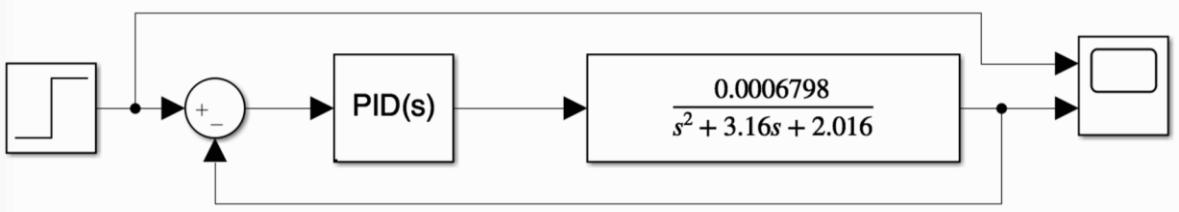
$$2.15\omega_n{}^2 = 2.016 + 0.0006798\ \mathrm{K}_1 \qquad \text{Ec. (3.9)}$$

$${\omega_n}^3 = 0.0006798\ K_2 \qquad \text{Ec. (3.10)}$$

La función de transferencia para el controlador PID de acuerdo con la ecuación 2.8 de la sección 2.4.1 quedó como se muestra en la ecuación 3.11. Este controlador se implementó en SIMULINK para analizar su funcionamiento. La Figura 3.26 y 3.27 muestran el comportamiento en la respuesta del motor CD en lazo cerrado con un controlador PID.

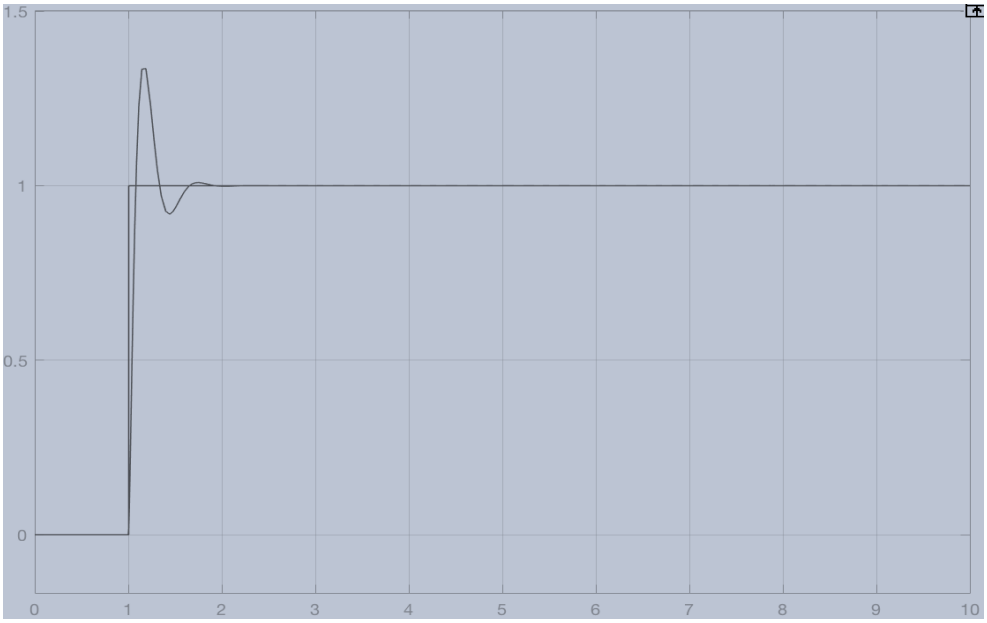
$$G_c(s) = 313303{,}9129 + \frac{1471020{,}888}{s} + 21094{,}4395\ s \qquad \text{Ec. (3.11)}$$

Figura 3.26. Motor CD con controlador PID en SIMULINK



Simulación del comportamiento del controlador PID para el motor CD. **Autor:** Nadim Tito S.

Figura 3.27. Respuesta de la salida del motor CD



Salida del motor CD controlada por un compensador PID. Elaborado por: Nadim Tito

Al colocar el controlador PID en SIMULINK para observar la respuesta de la salida se observó un comportamiento que no se acerca al deseado, es decir con un tiempo de asentamiento de 0.5 segundos y una sobreelongación de 2%.

3.5. Diseño de un controlador PID robusto

Para sintonizar los parámetros de un controlador PID robusto se aplicó el concepto explicado en la sección 2.4.2, donde se observa el diagrama de bloques que se aplica al sistema en la Figura 2.6. Para considerar el sistema completo según el diagrama de la Figura 2.6 se calculó los parámetros del prefiltro *Gp(s)*. Para calcular dicho prefiltro se evaluó la función de trasferencia *Tc(s)* (ecuación 3.6) que después de reemplazar los valores de K1, K2 y K3 quedaría como se muestra en la ecuación 3.12.

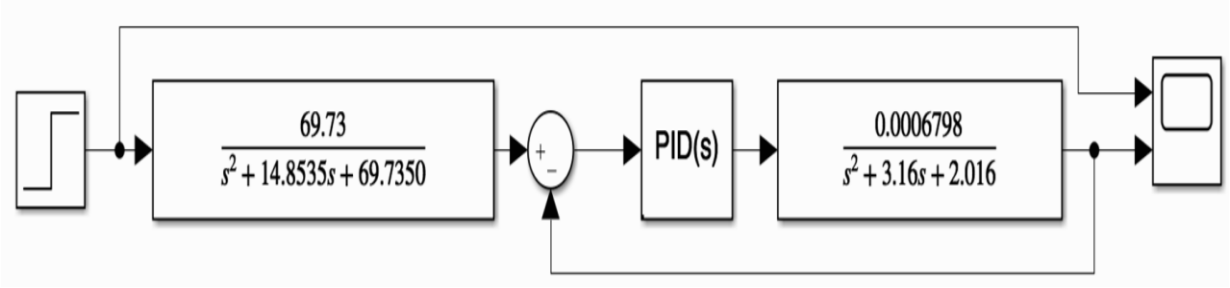
$$T_c(s) = \frac{14.34\,s^2+213\,s+1000}{s^3+17.5\,s^2+125\,s+1000} \qquad \text{Ec. (3.12)}$$

El proceso consistió en dividir todo el numerador para el valor que acompaña al termino s^2, es decir, se dividió para 14,34. La expresión encontrada al dividir dicho valor se observa en el denominador del prefiltro de la ecuación 3.13. Luego, se dividió 1000 para el valor 14,34 lo que proporcionó un resultado de 69.73 que se colocó en el numerador de dicho prefiltro.

$$G_p(s) = \frac{69.73}{s^2+14.8535\ s + 69.7350} \qquad \text{Ec. (3.13)}$$

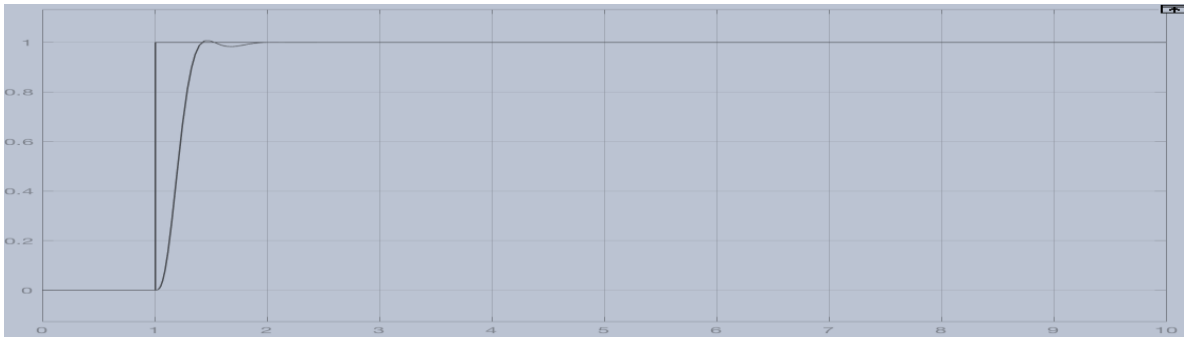
Al colocar el prefiltro en SIMULINK (Figuras 3.27 y 3.28) para observar la respuesta de la salida se observó un comportamiento que se acerca al deseado, es decir con un tiempo de asentamiento de 0.5 segundos y una sobreelongación de 2%.

Figura 3.27. Motor CD con controlador PID en SIMULINK



Simulación del comportamiento del controlador PID para el motor CD. Elaborado por: Nadim Tito

Figura 3.28. Respuesta de la salida del motor CD



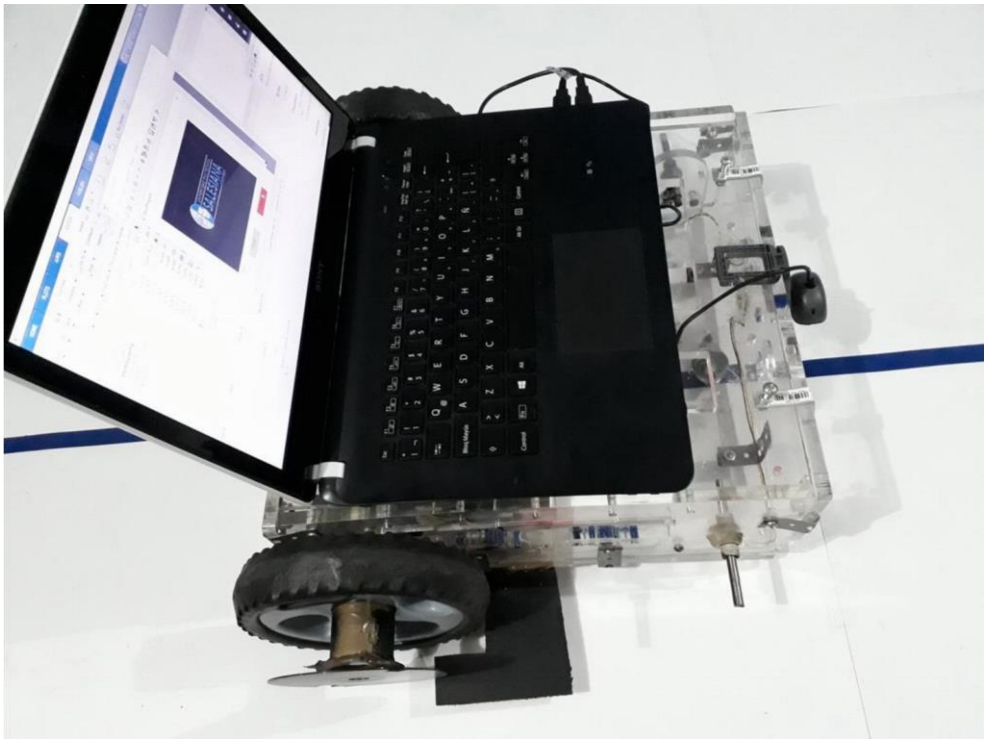
Salida del motor CD controlada por un compensador PID robusto. Elaborado por: Nadim Tito

CAPÍTULO 4

PRUEBAS Y RESULTADOS OBTENIDOS

Para realizar las pruebas del robot en el seguimiento de la línea se midió el tiempo que tardó el robot en desplazarse de un punto a otro. Se utilizó para esto el robot implementado como se muestra en la Figura 4.1. De acuerdo a las gráficas obtenidas en el diseño del controlador PID y PID robusto se esperó un mejor tiempo de respuesta del robot con los controladores diseñados de los motores DC.

Figura 4.1. Robot implementado para las pruebas



Estructura utilizada para las pruebas de funcionamiento del robot diferencial. Elaborado por: Nadim Tito

Para corregir la desviación del robot con respecto a la línea se realizaron pruebas previas para encontrar la relación entre ángulo de desviación y tiempo de activación del motor CD. Es decir, el ángulo de desviación del robot se corrige al activar una sola rueda por un tiempo calculado de acuerdo a la desviación y detener la otra para lograr el giro. No se utilizó encoders porque se pretendía obtener un sistema que dependiera

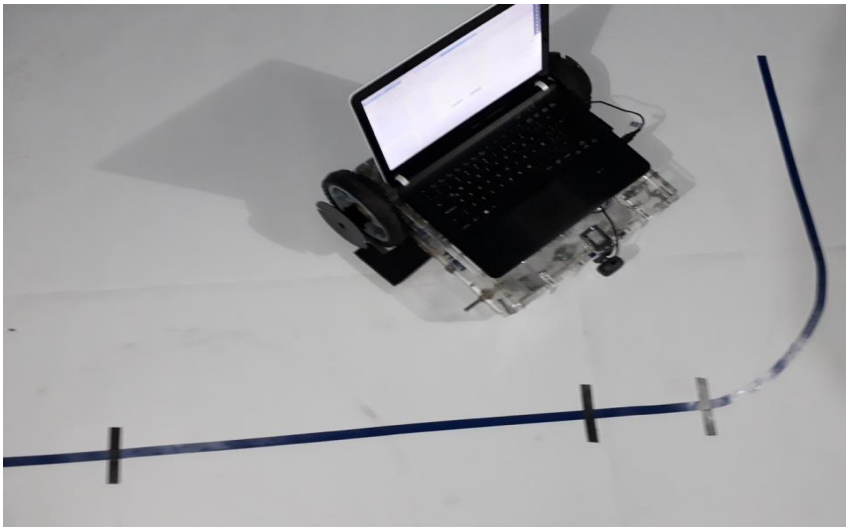
sólo de la visión artificial y el tiempo de respuesta de esta. En los siguientes apartados se hablan acerca de las pruebas realizadas con el sistema propuesto.

4.1. Pruebas de guiado sin control PID

Se utilizó marcas en la pista para medir el tiempo que tarda el robot en desplazarse de un punto a otro, ver Figura 4.2. Para guiar el robot se programó un algoritmo que calcula el ángulo de desviación de este con respecto a la línea. La desviación tiene un intervalo de error permisible entre 12 y menos 12 grados, es decir que mientras el ángulo sea menor a 12 grados de desviación el robot avanza hacia adelante.

De lo contrario, si el ángulo de desviación supera los 12 grados el robot corrige la desviación activando por un tiempo adecuado una sola rueda. El tiempo de activación de la rueda se calculó en base a pruebas realizadas con el robot y se encontró que por cada grado necesita un tiempo de 11 milisegundos para corregirlo. Es decir, si el ángulo de desviación es 20 grados el tiempo de activación de la rueda para corregirlo será 220 milisegundos. Cabe recalcar que, si la batería de las ruedas se descarga el tiempo de corrección varia, pero aun así se consigue guiar el robot de forma adecuada.

Figura 4.2. Pista marcada para las pruebas de tiempo



Marcas para medir el tiempo de desplazamiento. Elaborado por: Nadim Tito

El tiempo se midió con la ayuda de un cronómetro con una escala en segundos. Las pruebas ayudaron a dar una idea del funcionamiento del robot sin control PID. Se repitieron 10 pruebas de desplazamiento del robot del punto de inicio hacia el punto final (Figura 4.2) y se tomaron los datos tiempo que se exponen en la Tabla 4.1.

Tabla 4.1. Medidas de tiempo de desplazamiento

Prueba	**Tiempo medido (s)**
1	34,51
2	28,05
3	29,1
4	30,53
5	33,4
6	34,01
7	30,24
8	29,47
9	28,52
10	35,01

Medidas de tiempo tomadas en 10 pruebas de desplazamiento. Elaborado por**:** Nadim Tito

De los tiempos tomados de la Tabla 4.1 se encontró que el valor medio del tiempo que se demora el robot en desplazarse del punto inicial al punto final es 31,28 segundos. Lo que demuestra que el guiado del robot no tarda minutos en alcanzar una distancia de aproximadamente un metro de longitud.

4.2. Pruebas de guiado con control PID

Utilizando las mismas marcas en la pista se midió el tiempo que tarda el robot en desplazarse de un punto a otro, ver Figura 4.2. Para guiar el robot se añadió al algoritmo las líneas de código del controlador PID. Esta línea fue obtenida al aplicar la transformada Z de la función de transferencia de la ecuación 3.11 de la sección 3.4. A continuación se observan dicha línea de código del controlador PID:

$$U(k) = [K1 + K2\,T + K3/T]\,e(k) + K3\,T\,e(k-1) + K2\,u(k-1) \qquad \text{Ec.(4.1)}$$

Donde:

- U (k) es la señal manipulada actual
- u (k-1) es la señal manipulada anterior
- e (k) es el error actual
- e(k-1) es el error anterior
- T es el período de muestreo
- K1 es la constante de proporcionalidad
- K2 es la constante integral
- K3 es la constante derivativa

Los resultados obtenidos al medir el tiempo con el efecto del controlador PID se observan en la Tabla 4.2. La cual evidencia un valor medio de tiempo de 29 segundos, que comparados con el tiempo de 31,28 del funcionamiento sin PID no muestra gran diferencia. Se interpretó debido a estas pruebas que en este caso el controlador PID no es muy útil.

Tabla 4.2. Medidas de tiempo de desplazamiento

Prueba	**Tiempo medido (s)**
1	34,51
2	28,05
3	29,1
4	30,53
5	33,4
6	34,01
7	30,24
8	29,47
9	28,52
10	35,01

Medidas de tiempo tomadas en 10 pruebas de desplazamiento. Elaborado por: Nadim Tito

4.3. Pruebas de guiado con control PID robusto

De la misma forma se procedió a medir el tiempo de respuesta en desplazamiento del robot al programarlo con el controlador robusto. En estas pruebas se evidenció en base a los datos mostrados en la Tabla 4.3 que el uso de un controlador robusto tampoco hace mucha diferencia en este caso particular, ya que el tiempo promedio de desplazamiento es 29 segundos aproximadamente igual que con el controlador PID.

Tabla 4.3. Medidas de tiempo de desplazamiento

Prueba	**Tiempo medido (s)**
1	25,5
2	29,06
3	29,1
4	26,49
5	31,56
6	28,07
7	31,08
8	30,11
9	27,38
10	29,52

Medidas de tiempo tomadas en 10 pruebas de desplazamiento. Elaborado por: Nadim Tito

4.4. CONCLUSIONES

El tiempo de reconocimiento de la línea con técnicas de visión artificial no es el adecuado para realizar control por la demora en la respuesta de realimentación, esto debido al cálculo que realiza MATLAB para el procesamiento de imágenes. Si se suma a este hecho las características de una PC que no tenga buen procesador el cálculo se vuelve más lento por la cantidad de frames utilizados para realizar el procesamiento de las imágenes.

Utilizar la técnica del procesamiento digital de imágenes para identificar un camino segmentado por una línea la identificación de MATLAB debido a la dificultad presentada al momento de encontrar los valores característicos de este. Dicho valores son el momento de inercia del motor (J), coeficiente de fricción viscosa (B), constante de torque (Kt), constante de fuerza electromotriz (Kb), resistencia de la armadura (R) y la inductancia de la armadura (L).

Contrastar el desempeño del controlador robusto con un PID convencional. en la sección 3.6 no mostró una respuesta de acuerdo con las especificaciones dadas para el modelado, es decir, no presentó un tiempo de asentamiento de 0.5 segundos y una sobreelongación de 2%. Estas especificaciones se cumplen al completar el modelo del PID robusto mostrado en la sección 2.4.2.

El tiempo de 11 milisegundos calculado para corregir un grado de desviación del robot funcionó de manera correcta y sirvió para guiar el robot por la línea que trazaba el camino de desplazamiento. Cabe recalcar que, si la batería de las ruedas se descarga el tiempo de corrección varia, pero aun así, se consiguió guiar el robot de forma adecuada.

En este caso particular el controlador PID funciona de la misma forma que el controlador PID robusto, esto debido a que las condiciones de prueba se limitan a medir el tiempo de desplazamiento más no la reacción de la rueda directamente.

4.5. RECOMENDACIONES

Se recomienda cargar bien la batería del robot antes de realizar pruebas porque la velocidad del motor varía en función de la cantidad de corriente que entrega la fuente.

Se debe tener cuidado al momento de manipular los elementos electrónicos del robot cuando aún no han sido investigados sus manuales de funcionamiento. Ya que se debe saber que el robot presentaría reacciones no adecuadas por avería de alguno de sus elementos.

Se recomienda también revisar la teoría de control robusto de (Dorf & Bishop, 2007) para entender de forma más detallada el modelado matemático utilizado en el presente trabajo.

BIBLIOGRAFÍA

- Ogata, K. (2010). *Ingeniería de control moderna.* Madrid-España: PEARSON EDUCACIÓN, S.A.

- Dorf, R., & Bishop, R. (2007). *Sistemas de control moderno - Décima Edición.* MadridEspaña : Pearson Educación. S.A.

- Kavuran, G. (2017). Path Planning for Nonholonomic Mobile Robot. *IEEE*.

- D. D., A. K., & N. N. (2017). The basic of the identification, localization and navigation for mobile robots. *IEEE*.

- J. S., & P. S. (2017). A new approahc for line following robot using radius of path curvature and differential drive kinematics. *IEEE*.

- V. R., & A. M. (2015). Desing, simulation and implementation of cascaded path traking controller for a differential drive mobile robot. *IEEE*.

- MATLAB. (2018). *MATLAB.* Obtenido de Mathworks: https://es.mathworks.com/products/matlab.html

- Image Acquisition Toolbox . (2018). *Image Acquisition Toolbox Support Package for OS Generic Video Interface*. Obtenido de Mathworks: https://www.mathworks.com/matlabcentral/fileexchange/45183-image-acquisitiontoolbox-support-package-for-os-generic-video-interface

- Toolbox Image Processing. (2018). *Procesamiento, análisis y desarrollo de algoritmos de imágenes*. Obtenido de MathWorks: https://es.mathworks.com/products/image.html

- Plastiseal. (s.f.). *Plastiseal industrial*. Obtenido de http://www.plastisealindustrial.com/acrilico.html

- Car multimedia zone. (s.f.). *Car multimedia zone*. Obtenido de https://www.carmultimediazone.com/epages/accesorioscoche.sf/es_ES/?ObjectPath=/Shops/accesorioscoche/Products/SPYLA3032

- Arduino Nano. (2018). *Arduino Nano*. Obtenido de Arduino: https://store.arduino.cc/usa/arduino-nano

- Finehope. (s.f.). *Finehope*. Obtenido de http://www.polyurethanesupplierschina.com/es/products/Supplier-of-

polyurethaneself-skinning-wheelchair-tires-inflatable-tires-durable-high-quality-baby-c.html

- Genius. (2018). *720p HD webcam*. Obtenido de http://es.geniusnet.com/product/facecam1000x

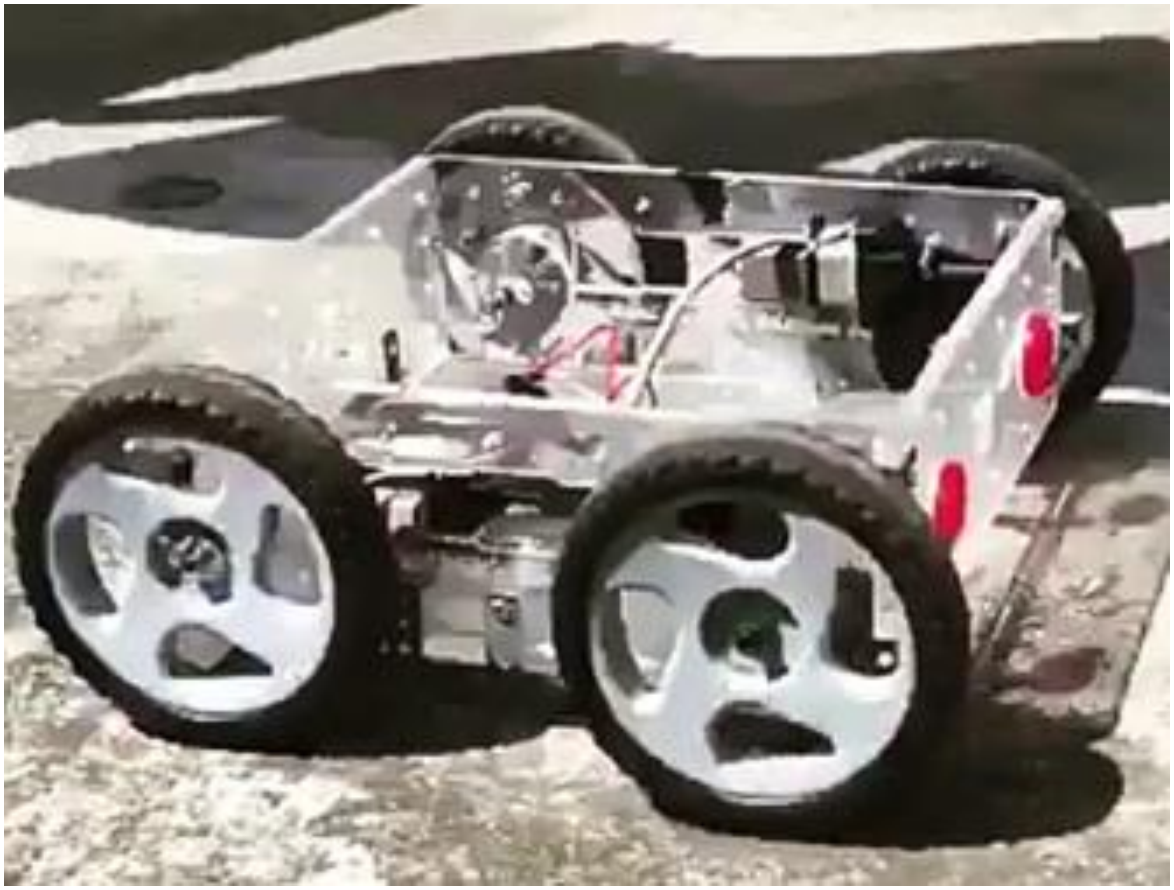
- Sparkfun. (2003). *SparkFun Monster Moto Shield*. Obtenido de https://www.sparkfun.com/products/10182

- ADDICARE. (2018). *Reflective Optical Sensor CNY70*. Obtenido de https://www.addicore.com/Reflective-Optical-Sensor-CNY70-p/224.htm

- LLamas, L. (29 de 09 de 2016). *Medir el ángulo y sentido de giro con Arduino y encoder rotativo*. Obtenido de ¿Qué es un encoder rotativo?: https://www.luisllamas.es/arduino-encoder-rotativo/

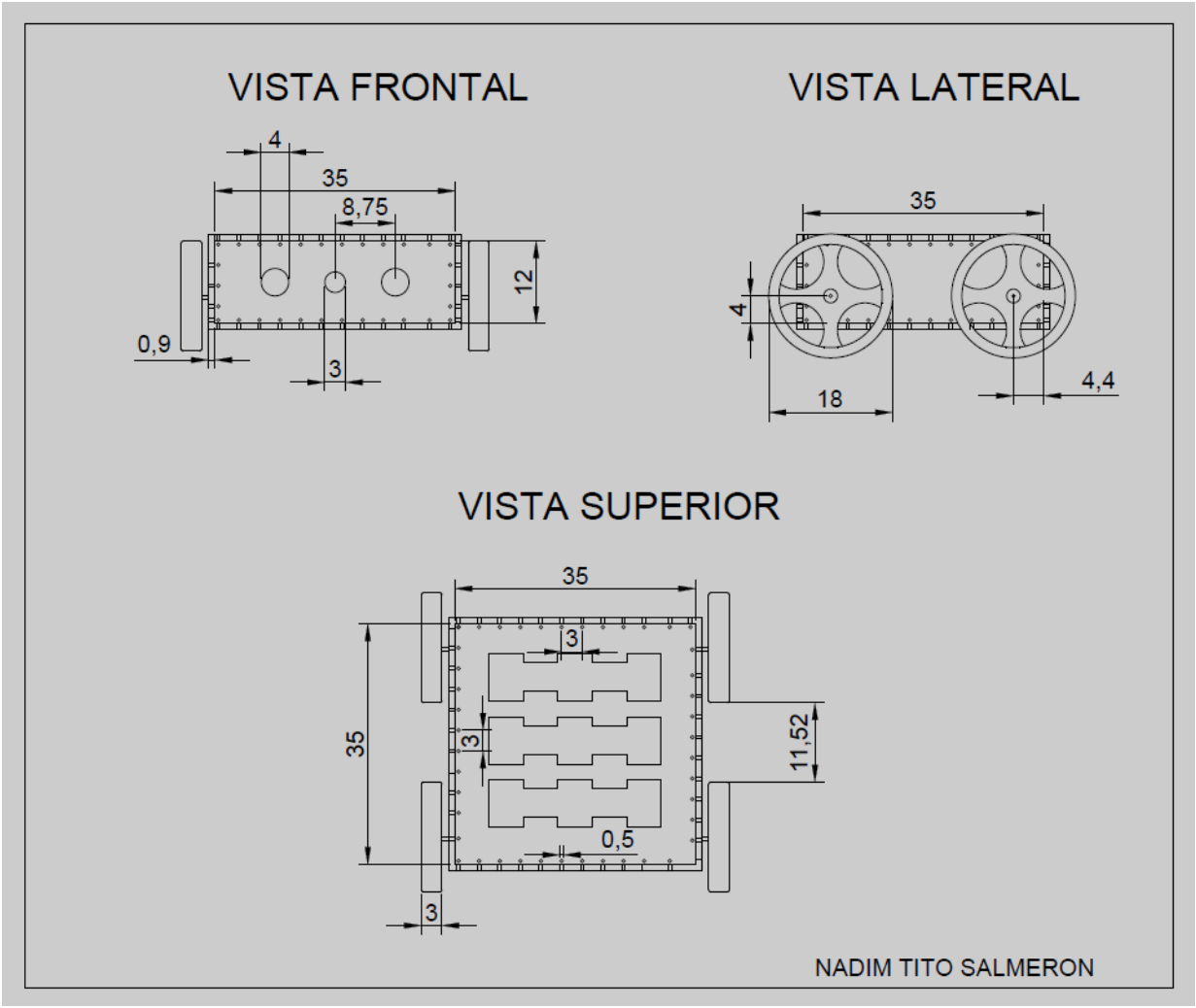
ANEXOS:

Fotografía del robot



Elaborado por: Nadim Tito

Diseño en AUTOCAD



Elaborado por: Nadim Tito