UNIVERSIDAD NACIONAL DEL ALTIPLANO

FACULTAD DE INGENIERÍA MECÁNICA ELÉCTRICA ELECTRÓNICA Y SISTEMAS

ESCUELA PROFESIONAL DE INGENIERÍA ELECTRÓNICA

DISEÑO DE UN CONTROLADOR PID CON REALIMENTACIÓN DE STREAMING DE VIDEO PARA GUIADO DE UN PROTOTIPO DE VEHÍCULO AUTÓNOMO

TESIS

PRESENTADA POR:

MARIELA LAQUI HUANCA

FLOR BRIGHITT PAMPA SULCA

PARA OPTAR EL TÍTULO PROFESIONAL DE:

INGENIERO ELECTRÓNICO

PUNO – PERÚ

2023

Reporte de similitud

NOMBRE DEL TRABAJO

DISEÑODEUNCONTROLADORPIDCONREALIMENTACIÓNDESTREAMINGDEVIDEOPARAGUIADODEUNPROTOTIPODEVEHÍCULOAUTÓNOMO

AUTOR

Flor Brighitt Pampa Sulca y Mariela Laqui Huanca

JAMES ROLANDO ARREDONDO MAMANI
CIP. 122464
SUB DIRECTOR DE INVESTIGACIÓN
E.P. INGENIERÍA ELECTRÓNICA

RECUENTO DE PALABRAS

16775 Words

RECUENTO DE CARACTERES

92168 Characters

RECUENTO DE PÁGINAS

102 Pages

TAMAÑO DEL ARCHIVO

2.4MB

FECHA DE ENTREGA

May 15, 2023 10:19 AM GMT-5

FECHA DEL INFORME

May 15, 2023 10:21 AM GMT-5

19% de similitud general

El total combinado de todas las coincidencias, incluidas las fuentes superpuestas, para cada base

- 17% Base de datos de Internet
- 3% Base de datos de publicaciones
- Base de datos de Crossref
- Base de datos de contenido publicado de Crossr
- 14% Base de datos de trabajos entregados

Excluir del Reporte de Similitud

- Material bibliográfico
- Material citado
- Material citado
- Coincidencia baja (menos de 12 palabras)

Maximo A. Montalvo Arco
ING. ELECTRÓNICO
CIP 100169

Resumen

DEDICATORIA

Dedico este logro a Dios y especialmente a mis padres por ser los pilares más importantes en mi vida y haber forjado e inculcado valores que han influido esencialmente en mi formación profesional y convertido en la persona que soy hoy en día. A mi hermana por estar siempre a mi lado en los momentos más difíciles, brindándome siempre su apoyo incondicional. A mis hermanos Luis Antonio y Leonel, en quien puedo contar están en los momentos más importantes y significativos en mi vida, por escucharme y ayudarme en cualquier momento y por la confianza que siempre nos tenemos. A mis familiares y amigos con quienes siempre puedo contar para cualquier eventualidad, que siempre estuvieron presente a mi lado para nunca rendirme. A la Universidad Nacional del Altiplano y en especial a la escuela profesional de Ingeniería Electrónica por ser parte de mi formación profesional y académica, brindarme la oportunidad de ser parte de su generación.

Mariela Laqui Huanca

DEDICATORIA

Dedicó esta tesis a mi madre y mi hermana quienes siempre estuvieron conmigo, por recibir su constante amor y apoyo, a mi tío por enseñarme a nunca rendirme ante los obstáculos.

Flor Brighitt Pampa Sulca

AGRADECIMIENTOS

A mis padres, Enrique Laqui Arce y Elena Huanca Quenaya, por su apoyo y comprensión incondicional, que fueron mi soporte y mayor admiración, por sus por sus sabios consejos, por haber demostrado que son los mejores padres que la vida me pudo dar.

A mis hermanos por apoyarme y estar siempre presentes en los buenos y malos momentos dándome fuerzas para superar los obstáculos que se me atraviesan en el camino.

A mis docentes por impartirme sus enseñanzas, orientación y conocimientos, a mis amigos y aquellas personas que me apoyaron directa e indirectamente en la realización de este proyecto, con sus conocimientos sabios y experiencias que fueron de gran aporte para la culminación de este proyecto.

Mariela Laqui Huanca

Agradezco principalmente a dios por guiarme y darme fortaleza para seguir adelante.

A mi madre y hermana por ser mis mejores guías, quienes siempre estuvieron a mi lado.

A mis docentes por su comprensión y paciencia, por compartir sus conocimientos durante los cinco años de mi desarrollo profesional.

A mí mejor amigo agradecerle por su apoyo y por compartir horas de estudio.

Flor Brighitt Pampa Sulca

ÍNDICE GENERAL

Área : Automatización e Instrumentación

Tema : Procesamiento de Señales

Fecha de Sustentación: 19 de mayo 2023

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE ACRÓNIMOS

CMOS **:** Semiconductor de Óxido Metálico Complementario

CCD : Dispositivo Acoplamiento de Carga

PID : Proporcional Integral Derivativo

PDI : Procesamiento Digital de Imágenes

PWM : Modulador de ancho de pulso

OpenCV : Real Time Computer Vision Library

TensorFlow : Software Library for Machine Learning

Keras : The Python Deep Learning API

API **:** Application Programming Interface

RESUMEN

Los avances tecnológicos en conducción autónoma aumentan lentamente y se integran poco a poco en el Perú, esto logra reducir la cantidad de accidentes, el desplazamiento de las personas discapacitadas son ventajas para los vehículos autónomos, para ello es necesario combinar una serie de elementos importantes en el vehículo como el circuito, un ordenador que procesa la información recibida y que le permita controlarlo. En este trabajo se ha desarrollado un diseño de un controlador PID implementado en Python3 con realimentación de transmisión de video desde Open CV para guiado de un prototipo de vehículo autónomo, para ello es necesario las imágenes y video, los cuales fueron procesados mediante procesamiento digital de imágenes para recoger los datos requeridos para el desplazamiento automático de un vehículo en un circuito de dos carriles paralelos. La fuente de video transmitido por una cámara web permite capturar y visualizar las líneas del carril. La cámara provee una salida de video que se direcciona a una base donde el procesamiento de video se realiza por los algoritmos de detección, el cual encarga de controlar la velocidad y dirección del vehículo, para que pueda circular por las líneas del carril, esto permite que el vehículo se pueda movilizar.

Palabras Clave: Vehículo autónomo, visión artificial, PID, controlador, motor DC, Raspberry Pi.

ABSTRACT

Technological advances in autonomous driving are slowly increasing and gradually integrating into Peru, which helps reduce the number of accidents. The mobility of disabled people is also a benefit of autonomous vehicles. To achieve this, it is necessary to combine a series of important elements in the vehicle, such as the circuit, a computer that processes the received information, and allows for control. In this work, a design of a PID controller implemented in Python3 with video feedback transmission from Open CV was developed to guide a prototype of an autonomous vehicle. Images and videos were processed through digital image processing to collect the required data for the automatic movement of a vehicle in a circuit of two parallel lanes. The video source transmitted by a webcam allows for the capture and visualization of the lane lines. The camera provides a video output that is directed to a base where video processing is performed by detection algorithms, which are responsible for controlling the speed and direction of the vehicle so that it can circulate through the lane lines, allowing the vehicle to move autonomously.

Keywords: Autonomous vehicle, Machine vision, PID, Controller, DC Motor, Raspberry Pi.

CAPITULO I

INTRODUCCIÓN

En los últimos años, ha habido un aumento en la búsqueda de un transporte más ecológico, seguro y eficiente debido a la creciente necesidad de vehículos y las demandas de la población. Como resultado, se genero una rápida evolución hacia la creación de vehículos autónomos, estos pueden operar por sí mismos y realizar las funciones necesarias sin la intervención humana.

En la primera parte de este proyecto se presenta las generalidades del proyecto, así como los objetivos, hipótesis y el planteamiento del problema y se exploran las diferentes, zonas de trabajo y opciones que existen para la para el desarrollo de sistemas autónomos y vehículos autónomos basados en sistemas tradicionales de control. Seguidamente en el segundo capítulo se presenta las metodologías que son empleadas para este tipo de proyecto desarrollan los cuales consiste en el procesamiento digital de imágenes como el uso de algoritmos de control como controlador PID todos estos son integrados dentro de un minicomputador que procesa las entradas y salidas del sistema. En el tercer capítulo se analizan los métodos de procesamiento digital de imágenes para detección de vía o línea negras mediante la conversión de imagen RGB a escala de grises. También se realiza los estudios para control de velocidad en motor DC esto junto al controlador PID.

En el cuatro capítulo se presentan los resultados en la implementación de estos dos procesos procesamiento digital de imagen y controlador PID para la movilidad del prototipo móvil en un carril. Por último, presentamos los resultados, junto a las conclusiones del trabajos y recomendaciones para trabajos futuros para la implementación y mejora del proyecto.

1.1. OBJETIVOS

1.1.1. Objetivo General

Implementar un controlador PID idóneo para un prototipo de un vehículo autónomo usando una transmisión de video como realimentación para mantener una trayectoria proyectada.

1.1.2. Objetivos Específicos

- Diseñar un controlador PID para el control de velocidad de motores.
- Conectar el controlador PID con un sistema de procesamiento digital de imágenes para realimentación de posicionamiento.
- Implementar un controlador PID con prototipo de vehículo autónomo para el control de motores.

2.1. HIPOTESIS

2.1.1. Hipótesis General

Se emplea mini-computadores con multiprocesadores como nuevas tecnologías e interfaces que son capaces de realizar procesamientos de información a una alta velocidad y mantener la trayectoria y el control de actuadores, lectura de sensores digitales y analógicos. Haciendo uso del enlace del controlador PID con el procesamiento de reconocimiento de imágenes, mediante la integración de sistema de transmisión de video se puede normalizar un proceso.

2.1.2. Hipótesis Específicas

- Se diseñará un prototipo de un vehículo autónomo que podrá mantener la trayectoria basada en la información procesada de los sensores implementados a bordo del vehículo.
- Se usará algoritmos en redes neuronales multivariable como de la posición en el espacio es enlazada a un controlador PID. La salida de un controlador PID es integrada en un microcontrolador para realizar las operaciones matemáticas dando como resultado salidas en PWM para accionar los motores.

CAPÍTULO II

REVISIÓN DE LITERATURA

2.1. ANTECEDENTES

2.1.1. A Nivel Local

En este nivel el desarrollo o investigación en la región de Puno no se han registrado desarrollado en investigaciones o implementados vehículos autónomos.

2.1.2. A Nivel Nacional

En el Perú se ha planteado realizar una propuesta de un vehículo autónomo para personas discapacitados en la región de Piura, la relevancia de iniciar y crear nuevas ideas en vehículos de transporte personal innovadores para lograr una mayor autonomía e independencia en cuanto a su movilidad, un aspecto es el incentivo la igualdad de oportunidades e inclusión en todos los aspectos cotidianos. Este vehículo presenta diferentes sistemas que podrían desarrollarse de maneras independientes, dentro de un equipo de investigación multidisciplinario. Para extender y perfeccionar la propuesta, con sistemas más confiables y económicos. (Gonzalez. F., 2019). Siendo este antecedente de aplicación de vehículos autónomos.

2.1.3. A Nivel Internacional

Una propuesta de un vehículo autónomo es realizada en Brasil donde la propuesta se basa en proponer un nuevo enfoque para la inferencia en tiempo real de mapas de ocupación para automóviles autónomos que utilizan redes neuronales profundas (DNN)

llamado NeuralMapper. NeuralMapper recibe los datos del sensor LiDAR como entrada y genera como salida el mapa de la cuadrícula de ocupación alrededor del automóvil. NeuralMapper infiere la probabilidad de cada celda del mapa de cuadrícula de una de las tres clases siguientes: Ocupada, Libre y Desconocida. El sistema se probó con dos conjuntos de datos y logró una precisión promedio de 76,48% y 73,81%. También se evalúa el enfoque con fines de localización en la conducción autónoma y la mayoría de los errores de posición de localización fueron inferiores a 0,20 m con un RMSE de 0,28, que se acercan a los resultados de la literatura para métodos que utilizan otros enfoques de mapeo de cuadrícula.

En la actualidad algunos robots móviles autónomos (AMR) son implementos con mini computadores para el empleo en industriales, esto de acuerdo a K. Piemngam(2019). Así como poder encontrar la fmejor ruta para llegar al área objetivo utilizando un mapa precargado. El AMR utiliza datos de cámaras y LIDAR o escáner láser para detectar los alrededores y crear un mapa de computadora para la planificación de la ruta. Este artículo muestra el desarrollo de una plataforma de robot móvil con ruedas Mecanum para navegación autónoma, incluido su diseño mecánico, diseño de sistema y construcción de robot. Además, el robot que utiliza NVIDIA Jetson TX2 se instaló con ROS como un control de alto nivel para realizar las tareas de navegación. Se utilizaron datos de RP LIDAR A2 e Intel Realsense Camera D415 para crear un mapa de robot. Mientras que un control de bajo nivel usa el microcontrolador Teensy 3.2 para enviar la señal PWM y obtener los datos del sensor de IMU, incluido el codificador. El diseño del robot con fines educativos y de investigación con el fin de mover el objeto desde el origen al área deseada automáticamente.

M. E. Abed, (2020) los modelos de control para la conducción con dirección automática fueron desarrollados y probados tanto en simulación como en hardware ambientes. Los resultados mostraron el éxito de la propuesta control PID sobre la clonación del comportamiento de aprendizaje automático modelo. Esto hace que el controlador PID sea el principal método de control en el esfuerzo de investigación adicional en conducción autónoma por el equipo y se utilizará en el vehículo experimental.

Siddartha Khastgir, (2021) presenta enfoques para el diseño de vehículos autónomos se basa en el aumento de la seguridad como uno de los principales beneficios de la introducción de los sistemas de conducción automatizada (ADS). Con un enfoque de pruebas basadas en peligros, este documento propone que la medida en que las millas de prueba son 'millas inteligentes' que reflejan escenarios basados en peligros relevantes para la forma en que un ADS falla o maneja los peligros es una consideración fundamental, si no fundamental, para garantía de seguridad de los ADS. Utilizando el método de Análisis Teórico de Procesos de Sistemas (STPA) como base, se ha desarrollado una extensión del método STPA para identificar escenarios de prueba. El enfoque se ha aplicado a un estudio de caso del mundo real de un sistema de conducción automatizada de baja velocidad SAE Nivel 4. Este documento analiza el análisis STPA y un método de creación de escenarios de prueba recientemente desarrollado derivado de STPA.

2.2. MARCO TEÓRICO

2.2.1. Vehículo Autónomo

Los automóviles autónomos utilizan tecnologías avanzadas de seguridad vehicular a través de computadoras, dispositivos electrónicos y sensores para conducirse solos (Sleight, 2022). También conocidos como vehículos autónomos, o AV para abreviar, los vehículos sin conductor luchan por volverse omnipresentes.

Un automóvil autónomo es un vehículo capaz de detectar su entorno y operar sin la necesidad de intervención humana. No se requiere que un pasajero humano tenga el control del vehículo en ningún momento, ni se requiere que un pasajero humano esté presente en el vehículo (SAE, 2022). Estos vehículos tienen una autonomía propia.

2.2.2. Visión Artificial

La visión artificial es un área de la inteligencia artificial (IA) que permite obtener datos importantes imágenes digitales, videos y otras entradas visuales para luego tomar decisiones o proporcionar recomendaciones basadas en dicha información, según IBM (2022). De manera similar a cómo la inteligencia artificial permite a las computadoras pensar, la visión artificial les permite ver, observar y comprender entornos y situaciones de manera autónoma.

La visión artificial según Contaval(2016) es una disciplina científica que incluye métodos para adquirir, procesar y analizar imágenes del mundo real para generar información que pueda ser procesada por máquinas.

La visión artificial por ordenador se enfoca en técnicas para capturar, procesar y analizar imágenes con el objetivo de obtener información procesable por computadoras. Existen diferentes librerías con herramientas útiles para la visión artificial, se proponen

las siguientes: OpenCV (Real Time Computer Vision Library), TensorFlow (Software Library for Machine Learning) y Keras (The Python Deep Learning API).

2.2.2.1 OpenCV

Para Gaudenz (2022), OpenCV es actualmente la biblioteca de visión por computadora de código abierto más popular que se enfoca en la visión en tiempo real. Es una biblioteca multiplataforma, compatible con Windows, Linux, Android y macOS, y se puede utilizar en diferentes lenguajes como Python, Java, y C++.

Ventajas:

- El uso es gratuito y es de código abierto.
- Ofrece acceso a más de 2500 algoritmos.
- Le permite modificar el código para que sirva a propósitos específicos.

Contras:

- No es tan fácil de usar como otras herramientas como MATLAB
- Curva de aprendizaje bastante empinada

2.2.2.2 TensorFlow

Gaudenz (2022), comenta que TensorFlow es una de las herramientas de visión por computadora más simples que permite a los usuarios desarrollar modelos de aprendizaje automático relacionados con la visión por computadora para tareas como reconocimiento facial, clasificación de imágenes, reconocimiento de objetos y más. Tensorflow, al igual que OpenCV, también admite múltiples lenguajes como Python, C, C++, Java o JavaScript.

Ventajas:

- Es una plataforma de código abierto.

- La plataforma es compatible con varios idiomas
- Proporciona actualizaciones constantes para más funciones y mejoras

Contras:

- Es un conjunto de herramientas muy intenso en recursos.

2.2.2.3 Keras

Keras es una biblioteca de software de código abierto basada en Python que es especialmente útil para principiantes porque le permite crear rápidamente modelos de redes neuronales y brinda soporte en back-end. (Gaudenz, 2022).

Ventajas:

- Biblioteca de Python fácil de usar.
- Proporciona soporte de back-end múltiple.

Contras:

- La funcionalidad se puede mejorar.
- La depuración puede ser difícil.

2.2.3. Controlador PID

Durante décadas, los controladores han sido ampliamente utilizados en la industria. En 1911 se creó el primer control PID (Proporcional - Integral - Derivado), el cual sigue siendo ampliamente utilizado en la automatización. El control PID combina acciones proporcional, integral y derivativa para generar una única señal de control, en la que cada acción tiene una característica especifica que ayuda a controlar la salida. La acción proporcional hace que el sistema reaccione al presente error, permitiendo una acción inmediata ante variaciones o perturbaciones; La acción integral elimina los errores de estado estacionario a largo plazo, y la acción Derivativa anticipa el comportamiento del proceso (BAZANELLA, 2005).

2.2.3.1 Control de Velocidad del Motor mediante la implementación del controlador PID.

SITI NUR WAHIDAH / BINTI ABDUL WAHAB. (2014). En el ámbito de la automatización y el control industrial, los controladores PID ha demostrado ser uno de los algoritmos de control más confiables para estabilizar la respuesta de salida de cualquier sistema. Las siglas PID significa Proporcional Derivada Integral, estos tres tipos de mecanismos de control se combinan para generar una señal de error que se utiliza como retroalimentación para controlar la aplicación final.

Justificación de su uso:

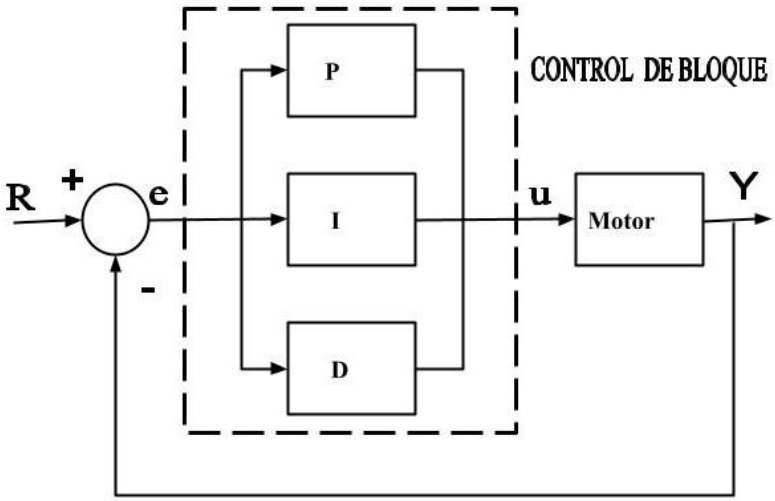
El controlador PID combina los componentes proporcional, integral y derivativo obteniendo la ecuación clásica que se puede observar a continuación:

$$u(t) = k_p + [\, e(t) + \frac{1}{T_i}\int e(t)dt + T_d \frac{de(t)}{dt}] \quad (1)$$

donde:

- k_p, es la ganancia proporcional.
- T_i, es la constante de tiempo de integración
- T_d, es la constante de tiempo de derivación
- $e(t)$, es la entrada del controlador
- $u(t)$, es la salida del controlador.

Figura 1: Diagrama de bloques que representa el controlador PID



Elaborado por el equipo de trabajo.

Acción Proporcional: K_p

Según WAHIDAH (2014) La parte proporcional es el resultado de la señal de error y la constante proporcional, lo que reduce el error de estado estacionario, pero en general estos valores solo serán óptimos para cierta parte del rango de control. El término proporcional genera una señal de salida que es proporcional al valor de error actual. La respuesta proporcional se puede expresar mediante la siguiente ecuación:

$$P_{out} = K_p . e(t) \quad (2)$$

donde:

- P_{out} , término proporcional de salida.
- K_p , ganancia proporcional.
- e, error = SP- PV
- t, tiempo o tiempo instantáneo

Acción Integral:

WAHIDAH (2014) El modo de control integrado está diseñado para reducir y eliminar los errores de estado estable causados por perturbaciones externas que no pueden eliminarse mediante el control proporcional.

$$I_{out} = K_i \int e(\tau) d\tau \quad (3)$$

donde:

- I_{out}, término integral de salida
- K_i, ganancia integral
- e, error = SP-PV
- τ, tiempo en el pasado que contribuye a la respuesta integral

Acción Derivativo:

WAHIDAH (2014) El término derivativo se usa para determinar la respuesta de un controlador a un cambio o perturbación del proceso. El término derivativo viene dado por:

$$D_{out} = K_d \frac{de}{dt} e(t) \quad (4)$$

donde:

- D_{out}, término derivado de la salida
- K_d, ganancia derivada
- e, error = SP - PV
- t, tiempo o tiempo instantáneo

2.2.3.2. Velocidad del motor de DC: Diseño del controlador PID

Desde el problema principal, las ecuaciones dinámicas expresadas en el dominio de Laplace y la función de transferencia de circuito abierto del motor DC son las siguientes.

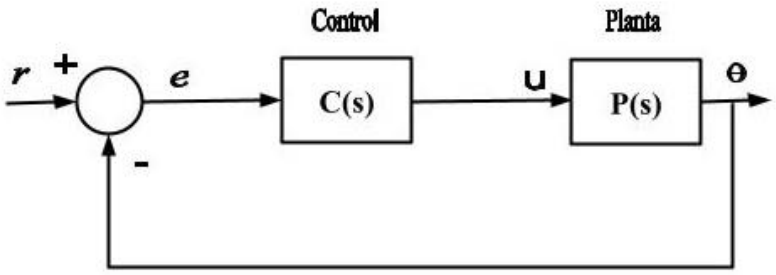
$$s(Js + b)\,\theta(s) = KI(s) \tag{5}$$

$$(Ls + R)\,I(s) = V(s) - Ks\,\theta(s) \tag{6}$$

$$P(s) = \frac{\boldsymbol{\theta}(s)}{V(s)} = \frac{K}{(Js+b)(Ls+R)+K^2} \quad [\frac{rad/sec}{V}] \tag{7}$$

La estructura del sistema de control tiene la forma que se muestra en la figura a continuación.

Figura 2: Diagrama de bloques del controlador PID.



https://ctms.engin.umich.edu/

Función de transferencia para un controlador PID es:

$$C(s) = K_p + \frac{K_i}{s} + K_d s = \frac{K_d s^2 + K_d s + K_I}{s} \tag{9}$$

Comandos en Matlab: “ARCHIVO Q”

J = 0.01;

b = 0.1;

K = 0.01;

R = 1;

L = 0.5;

s = tf ('s');

P_ Motor = K/((J*s+b)*(L*s+R)+K^2);

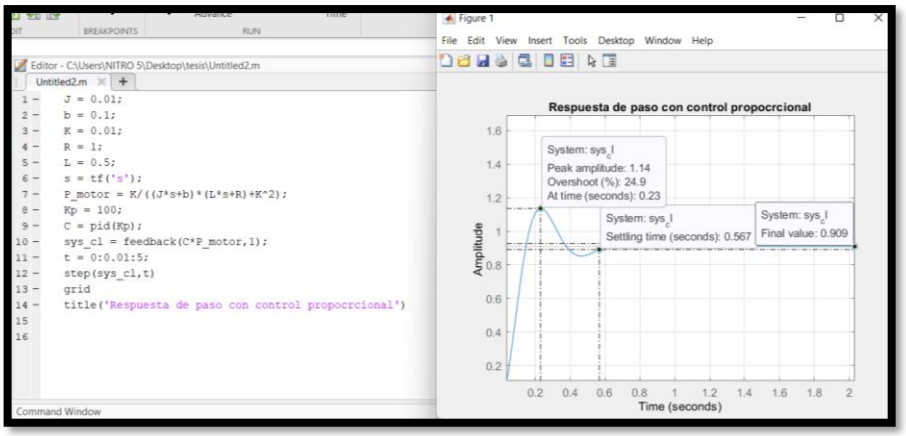
Control proporcional

Primero se intentará emplear un controlador proporcional con una ganancia de 100, es decir, C (s) = 100. Para determinar la función de transferencia de circuito cerrado, utilizamos el comando de retroalimentación.

Agregando el siguiente código al final del ARCHIVO Q.

```
Kp = 100;

C = pid(Kp);

sys_cl = feedback(C*P_motor,1);

t = 0:0.01:5;

step(sys_cl,t)

grid

title('Respuesta de paso con control proporcional')
```

Figura 3: Respuesta paso con control proporcional



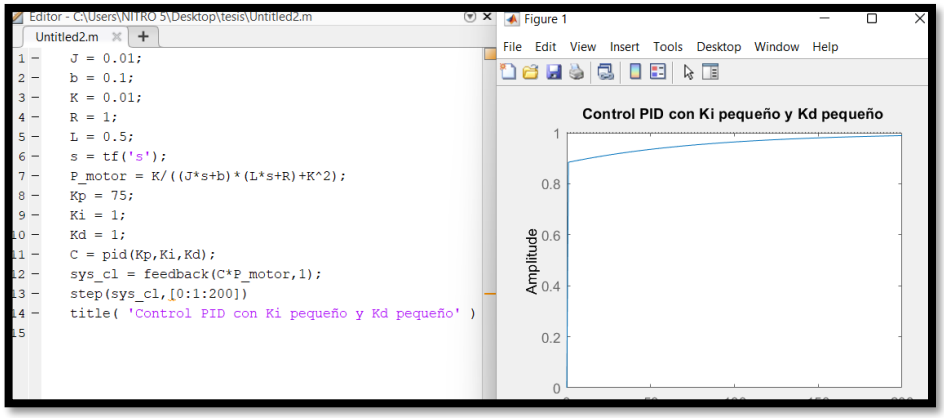
Elaborado por el equipo de trabajo.

Control PID

Se probará un controlador PID con Ki y Kd. Modifique su archivo Q para que las líneas que definan su control sean las siguientes. Ejecutar este nuevo archivo Q le da la trama que se muestra a continuación.

```
Kp = 75;
Ki = 1;
Kd = 1;
C = pid(Kp,Ki,Kd);
sys_cl = feedback(C*P_motor,1);
step(sys_cl,[0:1:200])
title('PID Control with Small Ki and Small Kd')
```

Figura 4: Control PID con Ki y KD



Elaborado por el equipo de trabajo.

La inspección de lo anterior indica que el error de estado estacionario realmente va a cero para una entrada de paso. Sin embargo, el tiempo que lleva alcanzar el estado estacionario es mucho más grande que el tiempo de asentamiento requerido de 2 segundos.

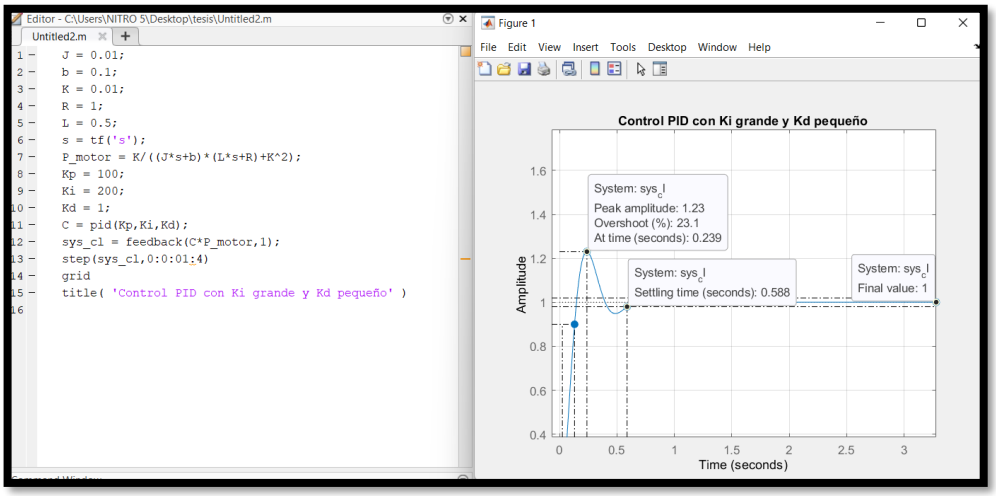
Ajustar las ganancias

En este caso, la cola larga en el gráfico de respuesta del paso se debe al hecho de que la ganancia integral es pequeña y, por lo tanto, tarda mucho tiempo en acumular la acción integral y eliminar el error de estado estacionario. Este proceso se puede acelerar aumentando el valor de Ki. Regrese a su archivo Q y cambie Ki a 200 como a continuación. Vuelva a ejecutar el archivo y debe obtener el diagrama que se muestra a

continuación. Nuevamente, las acciones se agregan haciendo clic derecho en la figura y eligiendo características del menú resultante.

```
Kp = 100;
Ki = 200;
Kd = 1;
C = pid(Kp,Ki,Kd);
sys_cl = feedback(C*P_motor,1);
step(sys_cl, 0:0.01:4)
grid
title('PID Control with Large Ki and Small Kd')
```

Figura 5: Control PID con Ki grande y Kd pequeño



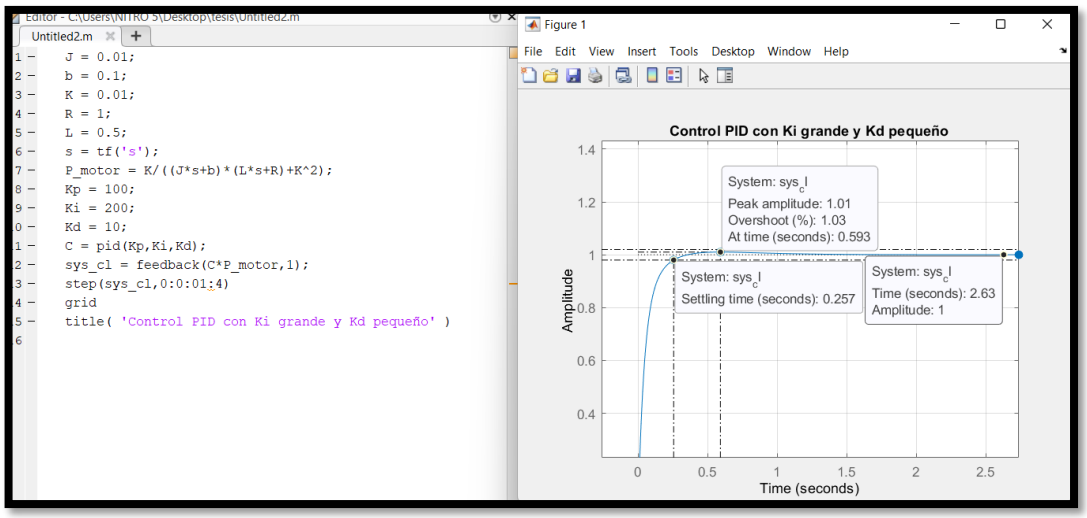
Elaborado por el equipo de trabajo.

Como se esperaba, el error de estado estacionario ahora se elimina mucho más rápido que antes. Sin embargo, el gran Ki ha aumentado enormemente el sobre impulso. Se aumentará Kd en un intento por reducir el sobre impulso. Regrese al archivo Q y cambie Kd a 10 como se muestra a continuación. Vuelva a emitir su archivo ".m" y la gráfica que se muestra a continuación debe generarse.

```
Kp = 100;
Ki = 200;
Kd = 10;
```

```
C = pid(Kp,Ki,Kd);
sys_cl = feedback(C*P_motor,1);
step(sys_cl, 0:0.01:4)
grid
title('PID Control with Large Ki and Large Kd')
```

Figura 6: Control PID con Ki grande y Kd grande



Elaborado por el equipo de trabajo.

Como se esperaba, el aumento de Kd redujo el sobre impulso resultante. Ahora sabemos que si usamos un controlador PID con Kp= 100, Ki= 200 y Kd= 10, todos nuestros requisitos de diseño se cumplirán.

2.2.4. Modelado del motor DC

Se demostrará cómo determinar de forma rápida estos parámetros de forma experimental y realizar ensayos técnicos para obtener los parámetros del motor de corriente continua.

Resistencia de Armadura

Existen dos maneras de determinar la resistencia de la armadura (R) de un motor, esto según Manuel (2009). El primer método consiste en medir directamente la resistencia con un multímetro conectado a los conductores de la armadura del motor (los cables rojo y negro). El segundo método consiste en ajustar un valor mínimo de voltaje para alimentar

el motor DC y medir la corriente de la armadura antes de que el eje del motor comience a moverse. Con este valor y utilizando un amperímetro y la ley de Ohm, se puede calcular la resistencia de la armadura. En este momento, el voltaje contraelectromotriz (Eb) es cero debido a que el eje no está en movimiento. Ambos métodos proporcionaran el mismo valor de resistencia.

$$R=\Omega \tag{10}$$

Inductancia de la Armadura

MANUEL (2009) Existen herramientas de medición, como el inductómetro, que permiten obtener el valor de la inductancia de armadura (L) a partir de los devanados del motor (Cables de alimentación del motor).

$$L=mH \tag{11}$$

Constante electromotriz y de torque

MANUEL (2009) Cuando un motor DC está en movimiento, la tensión en el componente llamado "inducido" aumenta en proporción al producto del flujo magnético y la velocidad angular. Si el flujo magnético es constante, como es el caso aquí, entonces la tensión inducida (Ea) aumenta directamente con la velocidad angular. De la ecuación podemos extraer la constante contraelectromotriz K_a.

$$K_a = \frac{Ea}{\omega\,(t)} \tag{12}$$

Aplicando Ley de Kirchoff: Ea = v - iR

$$K_a = \frac{v - iR}{\omega\,(t)} \tag{13}$$

- K_a: Constante electromotriz

- v : Voltaje aplicado
- i : corriente de la armadura consumida por el motor
- R: Resistencia de la armadura en Ohms
- $\omega\ (t)$: Velocidad en Radianes por Segundo

Con el siguiente circuito y algún tacómetro o microcontrolador que lea el encoder del motor podemos determinar el voltaje, la corriente y la velocidad para el cálculo de la constante K_a.

$$K_a = \frac{V\ - i\,R}{W\left(\frac{rad}{s}\right)} = \frac{v}{rads} \tag{14}$$

Dado que la constante electromotriz de voltaje es igual a la constante de torque tenemos:

$$K_a = K_m = \text{N. m/A} \tag{15}$$

Constante de tiempo mecánica

MANUEL (2009) Debido a la relación proporcional entre la tensión de armadura y la velocidad del eje del motor, al aplicar un cambio de voltaje al motor, podemos observar una respuesta transitoria en la velocidad para una carga fija. A partir de esta respuesta, se puede determinar la constante de tiempo, el cual es el 63.2% del crecimiento de la dinámica hasta llegar al estado estable.

Para realizar esta medición en el motor, se debe colocar un osciloscopio en los terminales del motor y aplicar escalones de voltaje entre +5V y 0V para poder observar la dinámica. Es posible utilizar un generador de señales o utilizar algún código para controlar motores con la plataforma Arduino, o simplemente conectar el motor

directamente a la fuente y capturar el momento exacto en que ocurre la transición de voltaje.

$$t_m = s \tag{16}$$

Momento de Inercia J

MANUEL (2009) Se puede utilizar la siguiente ecuación para calcular el momento de inercia J utilizando los parámetros que tenemos hasta este momento.

$$J = \frac{t_m \ K_a \ K_m}{R} = kg.m^2 \tag{17}$$

Corriente de Arranque y Torque de Fricción

MANUEL (2009) A través de un aumento gradual de voltaje DC sobre los terminales del motor, se debe incrementar la tensión gradualmente hasta que se observe que el eje del motor comienza a girar En este punto, se debe registrar la corriente del amperímetro, lo que proporciona la corriente necesaria del motor, para saber cuál es la corriente necesaria para el arranque del motor, i_{arr}.

Usando esta información, se puede calcular el torque de fricción T_f usando la siguiente ecuación:

$$T_m = K_m i_{arr} \tag{18}$$

Constante de Fricción de Coulomb

MANUEL (2009) La constante B se determina cuando el sistema alcanza una velocidad constante y, por lo tanto, se encuentra en un estado estable. En este momento, la derivada de la velocidad es igual a la aceleración, que es cero en muchos casos.

Utilizando esta información y la fórmula para el torque mecánico, se puede calcular el valor de B.

- B : Constante de fricción de Coulomb
- T_m : torque mecánico
- T_f : torque de fricción
- V : velocidad en radianes

$$B = \frac{T_m - T_f}{\omega(t)} \tag{19}$$

$$T_m = K_m\, i(t) \tag{20}$$

$$B = \frac{T_m}{\omega(t)} = N.m.s \tag{21}$$

2.2.4. Lenguaje de programación Python 3

Python es un lenguaje de programación poderoso y fácil de aprender que tiene estructuras de datos de alto nivel y eficientes y un sistema de programación orientado a objetos simple pero efectivo, Python.org (2022). Su elegante sintaxis y tipado dinámico, junto con su naturaleza interpretada, lo hacen ideal para crear scripts y desarrollar aplicaciones rápidamente en muchas áreas y en la mayoría de las plataformas. El intérprete de Python, así como su amplia librería estándar, se encuentran disponibles gratuitamente en código fuente y binario para la mayoría de las plataformas en el sitio web de Python.

Python es un lenguaje de programación que se puede usar en diferentes sistemas operativos y se ejecuta a través de un intérprete (AprendePython, 2020). Su enfoque se centra en tener una sintaxis clara y legible. Además, es un lenguaje multiparadigma, lo

que significa que admite diversos enfoques de programación, como la orientación a objetos, la programación imperativa y, en menor medida, programación funcional.

Características de Python

Python es un lenguaje:

- Interpretado
- Alto nivel
- Multiparadigma, ya que soporta orientación a objetos, programación imperativa y programación funcional.
- Multiplataforma
- Libre

Justificación de uso

- Porque es fácil de aprender
- Sintaxis muy limpia y sencilla
- Hay que escribir menos
- Obtienes resultados muy rápido
- Puedes programar con distintos paradigmas:
    - Programación imperativa
    - Orientación a objetos
    - Programación funcional
- Puedes programas distintos tipos de aplicaciones:
    - Aplicaciones de escritorio
    - Aplicaciones web
    - Scripts
- Muchos usan Python (Google, Nokia, IBM). Es demandado.

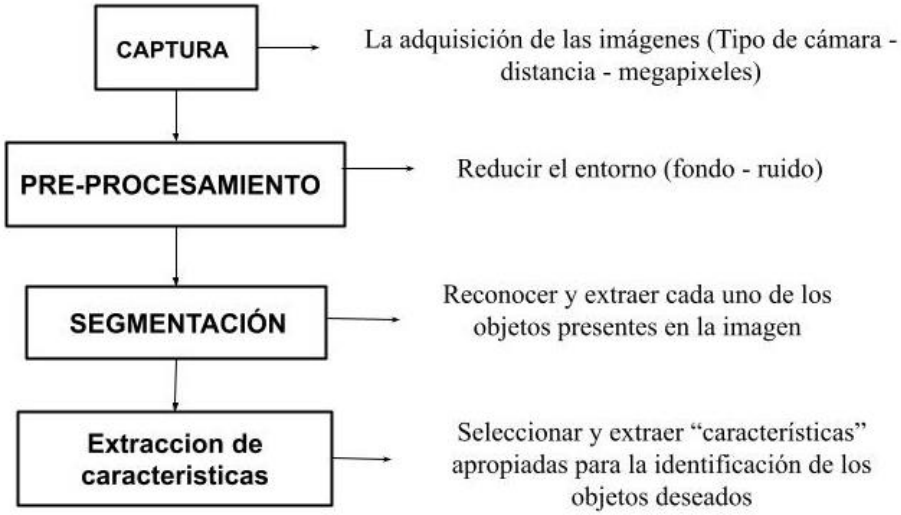
- Gran cantidad de módulos, muchísimas funcionalidades.
- Una gran comunidad que apoya el proyecto.
- Viene preinstalado en la mayoría de los sistemas.

2.2.6. Procesamiento Digital de Imágenes (PDI)

El procesado digital de imágenes es el procesamiento de imágenes digitales a través de una computadora digital, que incluye el almacenamiento, transmisión y representación de la información de la imagen. Una imagen es una función bidimensional que representa la intensidad de la luz en un punto (x, y) en la imagen, y puede ser representada en forma de matriz, donde cada elemento de la matriz (llamado pixel) corresponde a un punto en la imagen y tiene un valor que indica la intensidad de luz en ese punto. (Sihuacollo,2016).

Básicamente el procesamiento de imágenes se basa en capturar una imagen de la webcam y después modificarla utilizando funciones de la librería de OpenCV. Un filtrado en la imagen es una técnica muy útil cuando se trabaja con captura de imágenes desde una webcam. El propósito principal de este proceso es separar los colores y eliminar colores y/o secciones de la imagen que son consideradas inservibles. Los filtros más usados son los umbrales de escala de grises, RGB, YUV y otros. (Gallardo y Parga, 2011).

Figura 7: Etapas involucradas en el procesamiento de imágenes



Elaborado por el equipo de trabajo

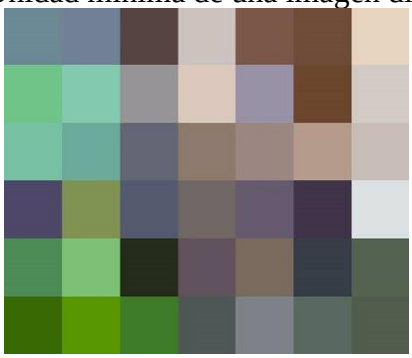
Según Alfaro (2003), Las cámaras digitales toman imágenes utilizando un sensor llamado CCD (dispositivo de acoplamiento de carga). La calidad de la imagen depende de la cantidad de píxeles que este sensor es capaz de capturar. Otros tipos de sensores también se utilizan en las cámaras digitales, como los llamados CMOS (semiconductor de óxido metálico complementario), que permiten una mayor velocidad de captura y convierten señales analógicas en imágenes digitales almacenadas en un formato binario como JPEG o RAW.

Figura 8: Sensor CCD y CMOS

https://www.calytel.com/

Se define una imagen digital como aquella en la que las variables x,y, y los valores de f son finitas, discretas y toman valores específicos. Estos elementos son llamados pels, o píxel.

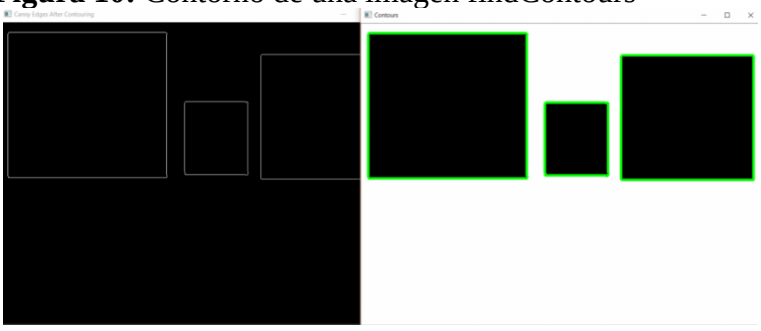
Figura 9: Unidad mínima de una imagen digital (pixel)



Elaborado por el equipo de trabajo.

La función findContours de OpenCV es una herramienta que permite identificar los contornos internos y externos de una imagen en blanco y negro. Estos contornos pueden ser clasificados de acuerdo con el método de detección seleccionado y pueden ser externos o internos. También se utiliza comúnmente en aplicaciones de visión artificial, como la detección de formas, colores y objetos. Rubén (2018).

Figura 10: Contorno de una imagen findContours



https://www.geeksforgeeks.org

La segmentación de imágenes es el proceso de dividir una imagen en distintas partes o regiones, cada una de las cuales pertenece a una clase o instancia específica. Esto se realiza mediante la aplicación de heurísticas o características relevantes, como el color, los bordes y los histogramas. Los algoritmos de segmentación de imágenes típicamente

utilizan estas características para clasificar los píxeles de la imagen en distintas categorías. Tyagi (2021)

2.2.5. Sistema Híbrido PID-OpenCV

Open CV

OpenCV (Open Source Computer Vision) es una librería multiplataforma de visión por computadora desarrollada por Intel en 1999. Se utiliza libremente con fines de investigación o comerciales bajo términos y condiciones establecidos. Incluye más de 500 funciones que cubren diversas áreas de procesamiento de imágenes, como reconocimiento facial, de objetos y estereoscópico, calibración de cámaras y visión robótica. OpenCV está programada en código C y C++ optimizado y aprovecha las capacidades de procesamiento de los procesadores multinúcleo. (Rodriguez, 2014).”

OpenCV (Open Source Computer Vision) es una librería multiplataforma que fui inicialmente desarrollada por Intel complementar a los primeros compiladores Intel C++ y Microsoft Visual C++ en x86. En la actualidad es ampliamente utilizada para el procesamiento de imágenes y la visión artificial en general, y ha sido implementada en una amplia gama de aplicaciones, como sistemas de control de procesos, seguridad con detección de movimiento, reconocimiento de objetos y robótica avanzada, entre otros.

Las librerías OpenCV son compatibles con los sistemas operativos Linux y Windows. ofrecen una gran eficiencia computacional y están diseñadas para aplicaciones en tiempo real. Están escritas en C y C++ optimizados y pueden ser ejecutadas en diversos sistemas operativos, como GNU/Linux, Windows, Android, iOS y Mac OS X. Además, cabe destacar que hay un continuo desarrollo en las interfaces de Python, Ruby, Matlab y otros lenguajes. OpenCV se divide en cinco componentes principales, según Bradski y Kaehler (2008):

- El componente de CV contiene los algoritmos de procesamiento de imágenes y de visión por ordenador en nivel básico y superior.
- ML es la biblioteca de aprendizaje automático, que contiene numerosos clasificadores estadísticos y herramientas de clustering.
- HighGUI incluye rutinas y funciones para la entrada y salida de datos, específicamente para el almacenamiento y carga de imágenes y videos.
- CXCore es el componente que contiene las estructuras básicas y algoritmos para el procesamiento de imágenes, además de ofrecer soporte para XML y funciones gráficas. También recibe información de CV, MIL y HighGUI.
- CvAux, que incluye algoritmos experimentales.

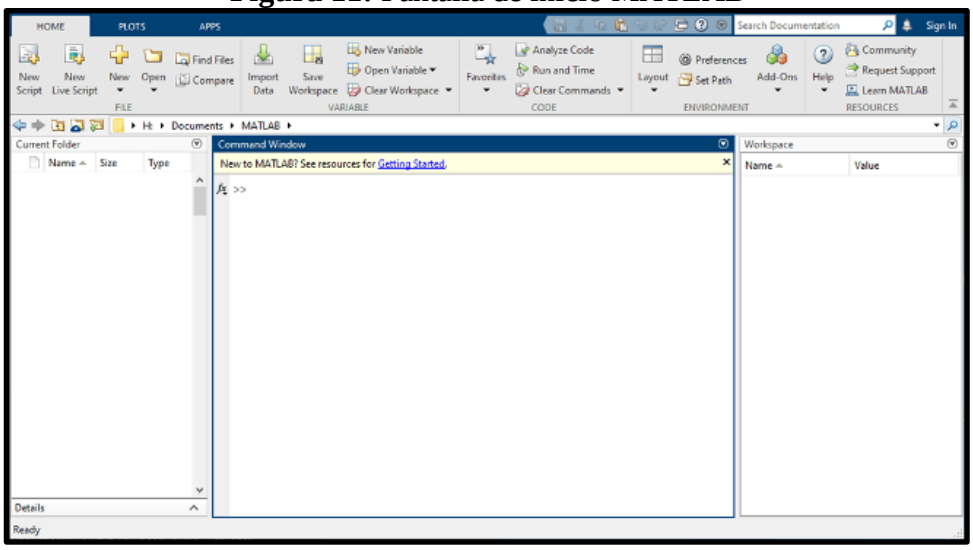
En lo que respecta al procesamiento de imágenes, un objetivo clave de OpenCV es ofrecer una estructura fácilmente accesible. La librería OpenCV cuenta con funciones que son empleadas en diferentes áreas de la visión artificial, como la inspección de productos, la identificación de objetos o personas en movimiento, el reconocimiento facial en imágenes, las imágenes médicas, la seguridad, las interfaces de usuario, la reconstrucción 3D, la robótica, entre otras aplicaciones.

2.2.8. Matlab Y Simulink

MATLAB:

Es un lenguaje de alto desempeño para computación técnica que fusiona cómputo, visualización y programación en una interfaz de usuario intuitiva, con problemas y soluciones mostrados en notación matemática común.

Figura 11: Pantalla de inicio MATLAB



www.mathworks.com

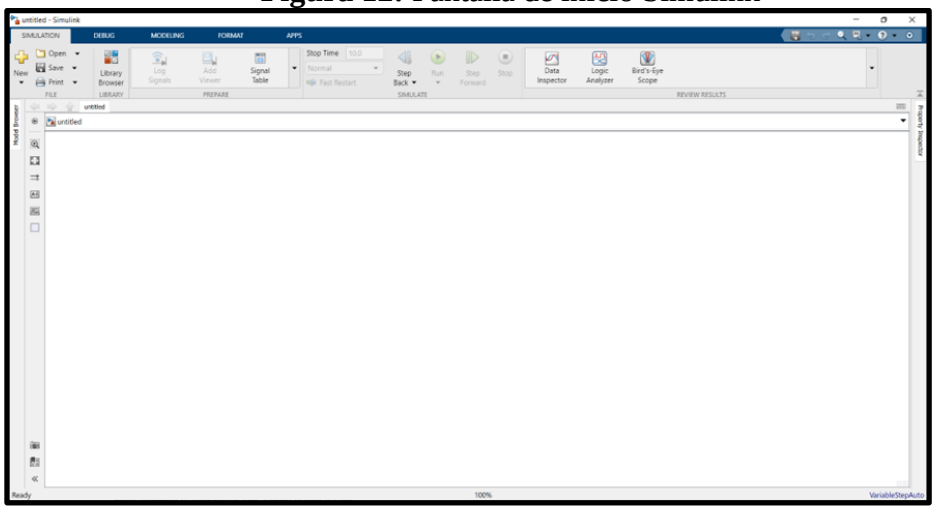
Justificación de su uso:

- Matemáticas y procesamiento de datos
- Desarrollo de algoritmos
- Recopilación de datos
- Creación de prototipos, modelado y simulación de sistemas
- Análisis, exploración y representación visual de datos técnicos y científicos
- Desarrollo de aplicaciones que incluyen la construcción de interfaces gráficas de usuario.

SIMULINK:

El software permite modelar, simular y analizar sistemas dinámicos a través de la construcción de diagramas de bloque gráficos, lo que permite evaluar su rendimiento y mejorar sus diseños.

Figura 12: Pantalla de inicio Simulink



Elaborado por el equipo de trabajo.

Justificación de su uso:

- Amplia colección de bloques predefinidos para crear modelos gráficos del sistema.
- Módulos adaptables que permiten integrar código previamente creado por el usuario en lenguajes como C, Ada, MATLAB y Fortran.
- Posibilidad de realizar simulaciones de manera interactiva desde la línea de comandos de MATLAB.
- Los objetos de datos de Simulink facilitan la creación de tipos de datos específicos de MATLAB para los modelos de Simulink.

Función:

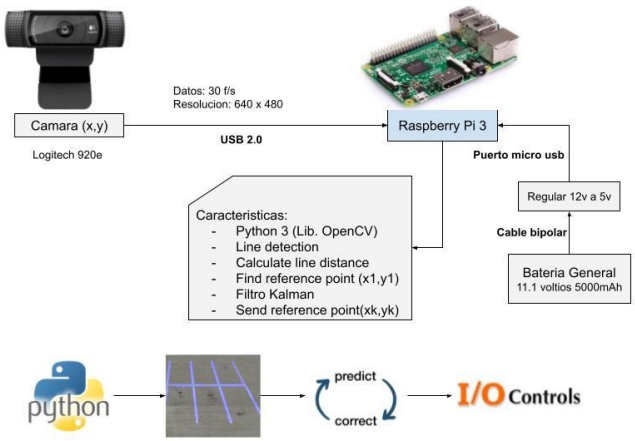
Simulink está estrechamente integrado con MATLAB. Requiere MATLAB para ejecutarse, dependiendo de MATLAB para definir y evaluar los parámetros del modelo y del bloque. Simulink también puede utilizar muchas características de MATLAB. Simulink puede usar MATLAB para:

- Definir las entradas del modelo.
- Almacene los resultados del modelo para su análisis y visualización.
- Realizar funciones dentro de un modelo, a través de llamadas integradas a operadores y funciones de MATLAB.

2.2.8. Procesamiento de video

El sistema de procesamiento se gestiona por el minicomputador Raspberry Pi, la cámara web es leído mediante un script en Python 3, los algoritmos utilizados son ***boundary-color*** que selecciona el color que se requiere reconocer calcular los puntos centros, ver Figura 13 para adquisición de datos (imágenes).

Figura 13: Procesamiento de video

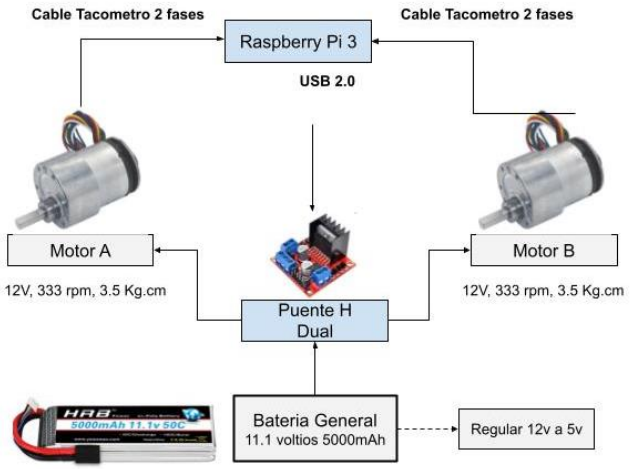


Elaborado por el equipo de trabajo.

2.2.9. Sistemas de control

El sistema de control se representa mediante una topología física de conexión, donde el minicomputador Raspberry Pi 3 es el control del sistema mediante el puente H dual que controla el giro y la velocidad de los motores, ver Figura 14.

Figura 14: Sistemas de control



Elaborado por el equipo de trabajo.

2.2.10. Cámara:

Una cámara digital es una cámara que almacena imágenes digitalmente en lugar de grabarlas en una película. Este tipo de cámara captura imágenes electrónicamente y las convierte en datos digitales que pueden almacenarse y procesarse en una computadora.

Función

Una cámara digital es un dispositivo fotográfico que adquiere y guarda imágenes de forma digital utilizando un sensor. Los sensores utilizados en las cámaras digitales son principalmente de dos tipos: el sensor CCD (Dispositivo de Carga Acoplada, por sus siglas en inglés) y el sensor CMOS (Semiconductor Complementario de Óxido Metálico, por sus siglas en inglés).

Justificación de uso

- Cuenta con entorno giratorio de 180° para captar un mejor ángulo.
- Tienen enfoque automático, las imágenes mantienen la nitidez y los detalles.
- Conexión USB para conectarse al ordenador.

Especificaciones Técnicas

- **Interfaz:** USB 2.0
- **Resolución de videos:** 1080p
- **Reducción de ruido:** Si
- **Longitud del cable:** 110 cm
- **Compatibilidad con sistema operativo:** Windows 7, Windows 10 y superior.
- **Modelo / marca:** WC - 1080
- **Alto/Ancho:** 8cm/8cm

Figura 15: Camara web Ebic WC-1080-HD

: www.coolbox.pe

2.2.11 Raspberry:

El Raspberry Pi fue desarrollado en Inglaterra por la Fundación Raspberry Pi es una placa pequeña, del tamaño de una tarjeta de crédito, que puede ser conectada a una computadora o televisor junto a un teclado y mouse para interactuar con ella. Este pequeño ordenador se puede utilizar en proyectos de electrónica a través de su puerto GPIO (General Purpose Input/Output o Entrada y Salida de uso General), ya que este puerto permite programar y facilitar el control de diferentes dispositivos a los cuales se puede conectar.

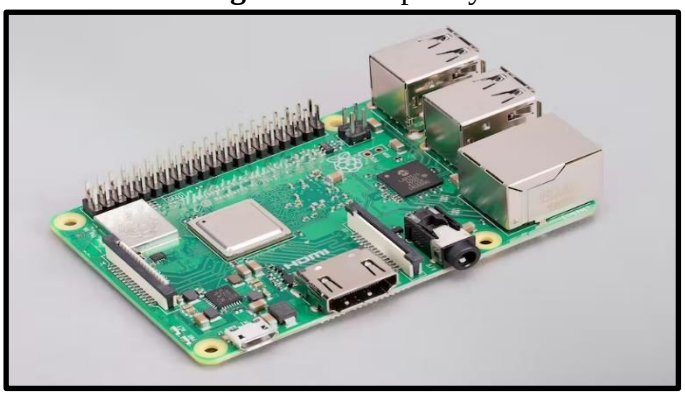
Función

Es la unidad de procesamiento que actúa como maestro para otros dispositivos inteligentes en el vehículo, basado en el sistema operativo Linux.

Justificación de uso

- Plataforma libre.
- Cuenta con un sistema operativo en tiempo real que permite la simultaneidad de procesos a velocidades diferentes.
- Ordenadores baratos y funcionales.

Figura 16: Raspberry Pi 3B



www.raspberrypi.com

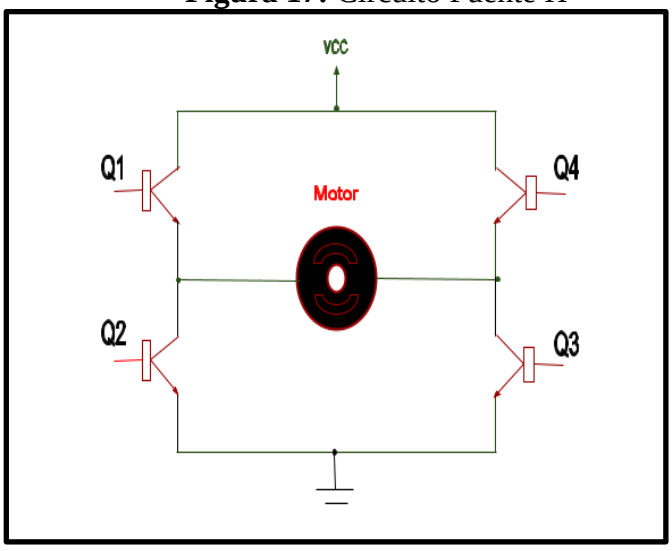
2.2.13. Controlador de motores Puente H

El circuito denominado puente H, el cual nos posibilita controlar la dirección de la corriente eléctrica en ambas direcciones mediante cuatro transistores.

Los componentes del puente H están dispuestos de tal manera que al activar dos de ellos se logra que el motor gire en una dirección, mientras que al activar otros dos se logra que gire en la dirección opuesta. Esta disposición se asemeja a la letra H, de ahí su nombre.

La configuración de los interruptores en el puente H permite que, al pasar la corriente por dos de ellos, el motor gire en sentido positivo. Al invertir el voltaje, se cierran los interruptores contrarios, lo que permite invertir el giro del motor.

Figura 17: Circuito Puente H



Elaborado por el equipo de trabajo.

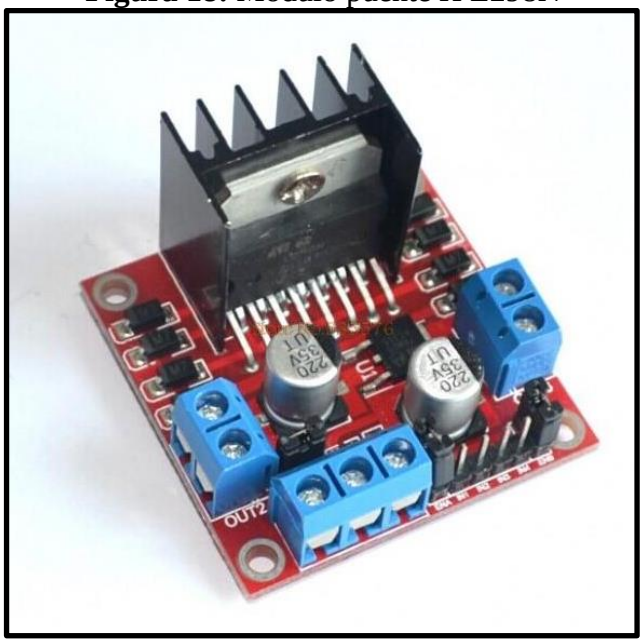
El driver puente propuesto para prototipo del módulo L298N es el más comúnmente utilizado para el control de motores DC con una corriente máxima de hasta 2 amperios. Internamente, el L298N cuenta con dos puentes H completos que permiten controlar dos motores DC o un motor paso a paso bipolar o unipolar.

El módulo posibilita la regulación de la dirección y velocidad de rotación de los motores mediante señales TTL que se pueden obtener de microcontroladores y tarjetas de desarrollo como Raspberry Pi, Arduino o Launchpads. La dirección de giro se controla mediante dos pines para cada motor, mientras que la velocidad se puede ajustar utilizando la técnica de modulación por ancho de pulso (PWM en inglés).

En su diseño, el L298N cuenta con un regulador de voltaje integrado LM7805 de 5V que se encarga de suministrar energía a la parte lógica del circuito. La activación de este regulador se lleva a cabo mediante un Jumper, y es posible utilizarlo para alimentar la etapa de control del L298N.

- Chip: L298N
- Canales: 2 (soporta 2 motores DC o 1 motor PAP)
- Voltaje lógico: 5V
- Voltaje de potencia (V motor): 5V - 35V DC
- Consumo de corriente (lógico): 0 a 36mA
- Capacidad de corriente: 2A (picos de hasta 3A)
- Potencia máxima: 25W
- Dimensiones: 43 * 43 * 27 mm
- Peso: 30g
- Admite entradas de señal PWM para el control de velocidad.
- Posee 8 diodos de protección contra corriente inversa.

Figura 18: Módulo puente H L298N



https://naylampmechatronics.com

Tabla 1: Tabla de verdad del Sistema de Control de motor DC con puente H

Entradas		**Polaridad**	
A	**B**	**A**	**B**
0	0	0	0
0	1	-	+
1	0	+	-
1	1	0	0

Elaborado por el equipo de trabajo.

2.2.14. Módulo regulador de voltaje

Los convertidores DC/DC modifican el nivel de voltaje de una fuente de alimentación a un nivel mayor o menor. Uno de estos convertidores es el LM2596, que se clasifica como un convertidor Step-Down DC-DC de 3A. Su función principal es proporcionar un voltaje de salida constante, que es menor que el voltaje de entrada, independientemente de las fluctuaciones de carga o voltaje de entrada. El LM2596 es

capaz de soportar una corriente de salida de hasta 3A, así como un voltaje de entrada de entre 4.5V y 40V, y un voltaje de salida que oscila entre 1.23V y 37V. El voltaje de salida deseado se ajusta a través de un potenciómetro multivuelta. Además, el LM2596 requiere muy pocos componentes externos para funcionar correctamente.

El LM2596 es un convertidor DC-DC de tipo reductor (Step-Down o Buck) que utiliza la técnica de conmutación para lograr una alta eficiencia de conversión, una regulación de línea precisa y un bajo nivel de rizado en el voltaje. Este módulo se caracteriza por su capacidad de reducir el número de componentes externos necesarios en la construcción de fuentes de alimentación, lo que simplifica el diseño y reduce los costos. Asimismo, es capaz de proporcionar un voltaje regulado que se estable a partir de una fuente de voltaje de entrada mayor.

Especificaciones Técnicas

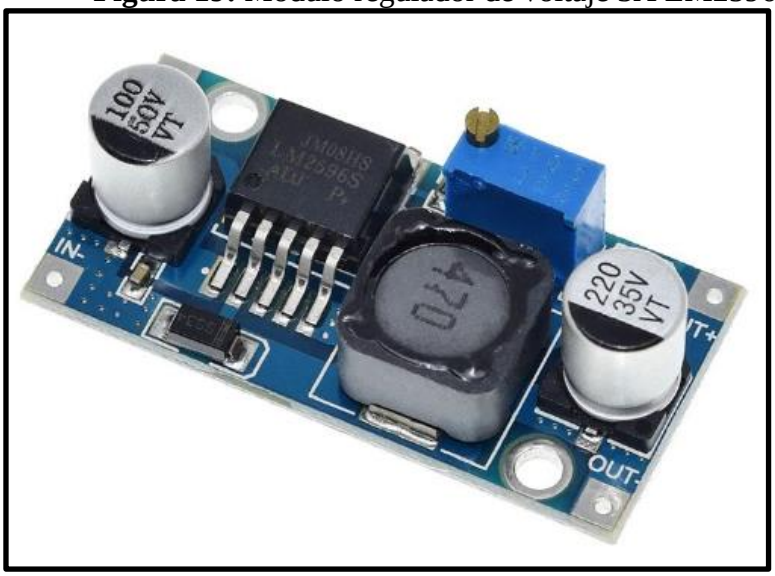
- Convertidor DC-DC Buck: LM2596
- Voltaje de entrada: 4.5V a 40V DC
- Voltaje de salida: 1.23V a 37V DC
- V. salida ajustable (el voltaje de entrada debe tener al menos 1.5V más que la salida).
- Corriente de Salida: máx. 3A, 2.5A recomendado (usar disipador para corrientes mayores a 2A).
- Potencia de salida: 25W
- Eficiencia de conversión: 92%
- Regulación de carga: S (I) ≤ 0.5%.
- Regulación de voltaje: S (u) ≤ 0.5%.
- Frecuencia de Trabajo: 150 KHz
- Ripple en la salida: 30mV (máx.) 20 M bandwidth

- Protección de sobre temperatura: SI (apaga la salida)
- Protección de corto circuito: SI (hasta 5A)
- Protección limitadora de corriente: SI
- Protección frente a inversión de polaridad: NO
- Dimensiones: 43mm*21mm*13mm

Justificación de uso

- No todos los componentes trabajan a 12V.
- Se deberá emplear un regulador para disminuir el voltaje a 5V que son requeridos por componentes como: Arduino, Raspberry Pi 3.

Figura 19: Módulo regulador de voltaje 3A LM2596



https://uelectronics.com

2.2.15. Batería

Las baterías o acumuladores que son dispositivos que permiten transformar energía química a energía eléctrica. Las baterías contienen electrodos positivos y negativos, se representan de varios tamaños y formas, y se emplean en una gran variedad

de usos como en aparatos eléctricos y electrónicos (audífonos, equipos médicos, ordenadores, etc.).

Justificación de su uso

El uso de una determinada cantidad de celdas en el vehículo se consideran dos aspectos importantes: la tensión y el amperaje. En cuanto a la tensión, se debe tener en cuenta que los drivers operan a 12 voltios y para el funcionamiento del resto de los dispositivos se requiere una tensión menor, por lo que se considera que el mínimo indispensable son 3 celdas. Aumentar la tensión implicaría la necesidad de un regulador para los drivers, lo cual no es conveniente. En cuanto al amperaje, se opta por una batería de alta descarga de 60 amperios, aunque el vehículo no consuma más de 10 amperios en su estado actual, ya que se justifica su uso en la disponibilidad de baterías.

Conexionado

En cuanto al conexionado, se deriva la salida de la batería directamente a dos tomas de alimentación: una para los drivers y otra para el regulador que reduce la tensión a 5 voltios y alimenta el resto de los dispositivos. Los drivers tienen una tolerancia alta en la tensión de alimentación, por lo que no es necesario utilizar un regulador con relación 1:1 y se puede realizar la conexión de manera directa.

Figura 20: Batería LI-PO 11.1V

Elaborado por el equipo de trabajo.

2.2.16. Motor DC

Los motores que funcionan con corriente continua que genera un campo magnético que actúa sobre el rotor o inducido del motor y convierte la energía eléctrica en energía mecánica. Una de las funciones consiste en medir la velocidad del vehículo mediante el encoder y luego transmitir estos datos a los controladores correspondientes.

Función

Se utilizó el motor de modelo "DC TT Motor 3v-6v Robot Smart Toy Car TT gear Motor", es un motor de propósito general, funciona perfectamente con baterías, se utilizan para ventiladores electrónicos, máquinas electrónicas, automóviles, robots, proyectos educativos.

Justificación de su uso

- Se pueden alimentar con energía almacenada en baterías.
- El control de velocidad es más simple y económico que los motores AC.
- Pueden trabajar a bajas velocidades, solo llevan dos cables.

- Tiene una gran capacidad anti-interferencias y no causa interferencias en el microcontrolador.

Especificaciones Técnicas

Marca/Modelo: Aokin / 1907_1

Torque: 800gf cm min.

No-load Speed: 1:48

RPM: 200 (cuando es de 4,5v)

Load current: 170 mA (cuando es 4.5v)

Figura 21: Motor DC for Arduino Remote Control

www.alibaba.com

2.2.17. Modulación PWM

La técnica de modulación conocida como modulación de ancho de pulso (Pulse Width Modulation) se encarga de modificar el ancho y la duración de los pulsos mediante un modulador de señal de información. Aunque esta técnica puede utilizarse para codificar información de señales de transmisión de datos, su principal función es controlar la cantidad de energía que se suministra a un dispositivo eléctrico, como un motor. La señal de suministro consiste en un tren de impulsos de voltajes, donde el ancho de los

pulsos individuales determina el nivel de voltaje efectivo que llega a la carga. En el caso de un motor de corriente continua, la modulación de ancho de pulso afecta el almacenamiento de energía en los devanados del motor, suaviza eficazmente las irrupciones de energía entregadas por el pulso de entrada para que el motor experimente un mayor o menor entrada de energía eléctrica dependiendo del ancho de los pulsos (Angalaeswari, 2016).

La modulación por ancho de pulso es una técnica que consiste en variar el ancho de una señal periódica para enviar información o controlar la cantidad de energía que se entrega a una carga. Esta técnica se utiliza para modificar el proceso de trabajo de una señal periódica con diferentes propósitos, como el control preciso de la energía entregada a una carga o la transmisión de datos.

Si el ciclo de trabajo es del 25%, significa que la señal estará en estado alto durante el 25% del período y en estado bajo durante el 75% restante. El período es la duración total de la onda antes de repetirse. Por lo tanto, la señal se repite a lo largo del tiempo, ya que el período se repite continuamente. El período se compone de la duración del estado alto y la duración del estado bajo, y cuando la señal vuelve a estar en estado alto, comienza un nuevo período y la onda comienza de nuevo. Las señales cuadradas o sinusoidales pueden variar en ancho relativo en relación con su período, lo que se conoce como ciclo de trabajo y se expresa en términos porcentuales. Para emular una señal analógica, se modifica el ciclo de trabajo para que el valor promedio de la señal se aproxime al voltaje deseado, lo que permite enviar voltajes entre 0 voltios y el máximo que soporte el dispositivo PWM utilizado.

CAPÍTULO III

MATERIALES Y MÉTODOS

3.1. MATERIALES

3.1.1. Hardware

- Raspberry Pi 3+
- Computadora Personal ACCER
- Regulador LM
- Router

3.1.2. Software

- Python 3
- Open CV
- Matlab
- Microsoft Word

3.2. MÉTODOS

3.2.1. Enfoque de la investigación

El método utilizado es descriptivo-experimental, ya que busca presentar detalles y características a través del estudio y análisis de los escenarios de experimentación y las muestras. El resultado esperado es una optimización de los materiales y software, estableciendo relaciones causa-efecto entre las variables con el fin de lograr una mejora en la eficiencia del sistema. Los algoritmos para identificación de patrones, objetos basados en su forma morfológica del objeto o patrón, son bases fundamentales para la creación de modelos de detección. (González y Woods, 2006).

3.2.2. Procesamiento digital de video

En procesamiento digital de imágenes, es mediante el testeo y aproximación e iteración de operaciones morfológicas, referencia de código en Anexo 2. Este código utiliza la biblioteca OpenCV, la biblioteca Numpy y una biblioteca "imutils" adicional para detectar y rastrear la posición de un objeto específico en un video. El video denominado "carretera.mp4", pero también se puede reemplazar con una transmisión en vivo desde la cámara de la computadora.

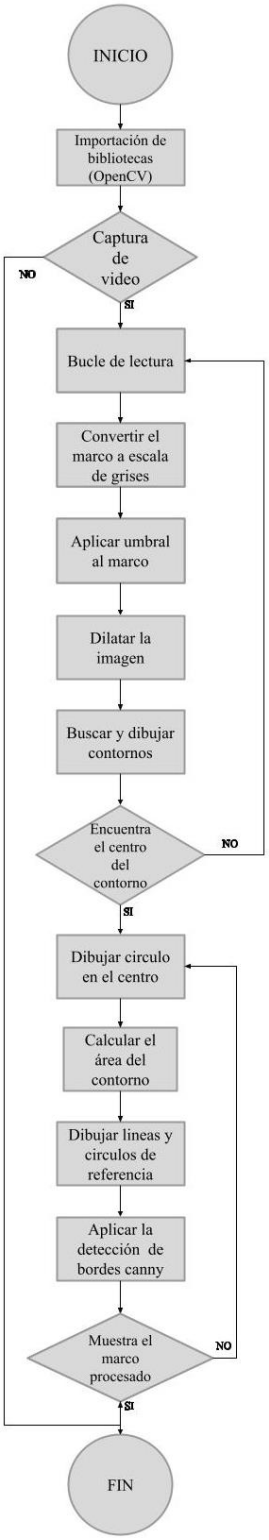
El código lee los cuadros del video y los convierte a escala de grises. Luego aplica un umbral al marco de escala de grises para crear una imagen en blanco y negro. Luego, el código dilata la imagen en blanco y negro y aplica el algoritmo de detección de bordes Canny para encontrar bordes en la imagen.

Luego, el código usa la función "findContours" de OpenCV para identificar y dibujar los contornos del objeto. También encuentra el centro del objeto y dibuja un círculo en ese punto. También calcula el área del contorno y lo imprime.

Además, el código también dibuja líneas de referencia en el marco y círculos alrededor del centro del objeto, que se pueden usar para rastrear la posición del objeto en el marco.

En general, este código se usa para rastrear el movimiento y la posición de un objeto específico en una transmisión de vídeo. A continuación, se visualiza un diagrama de flujo de este código:

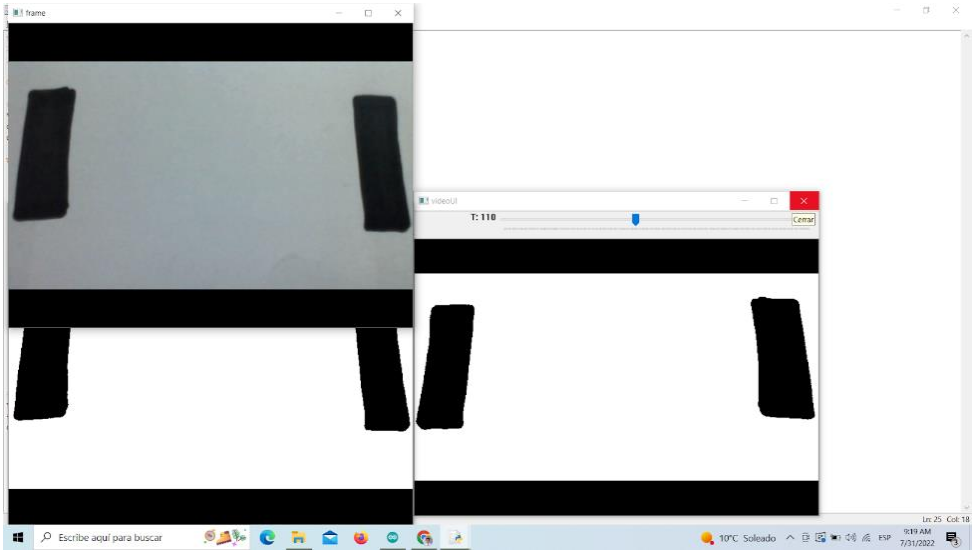
Figura 22: Diagrama de flujo procesamiento digital de imágenes



Elaborado por el equipo de trabajo

El resultado del algoritmo del reconcomiendo de líneas comienza primero con la extracción de color y binarización de la imagen.

Figura 23: Adquisición de imágenes de webcam.



Elaborado por el equipo de trabajo.

El primer paso del desarrollo se empieza a realizar el reconocimiento y detección de líneas por la semejanza con las vías. En la figura 21 se observa que las líneas paralelas mediante el algoritmo de detección de bordes se capturan las líneas negras. El algoritmo Canny es un algoritmo de detección de bordes utilizado en procesamiento de imágenes. Fue desarrollado por John F. Canny en 1986. El algoritmo funciona a través de varios pasos, incluyendo suavizado de la imagen, cálculo de la derivada de gradiente y aplicación de un umbral para determinar qué bordes son significativos y cuáles no lo son.

El algoritmo es conocido por ser muy preciso en la detección de bordes, especialmente en imágenes con ruido. En OpenCV, se puede utilizar cv2.Canny() para aplicar el algoritmo Canny a una imagen.

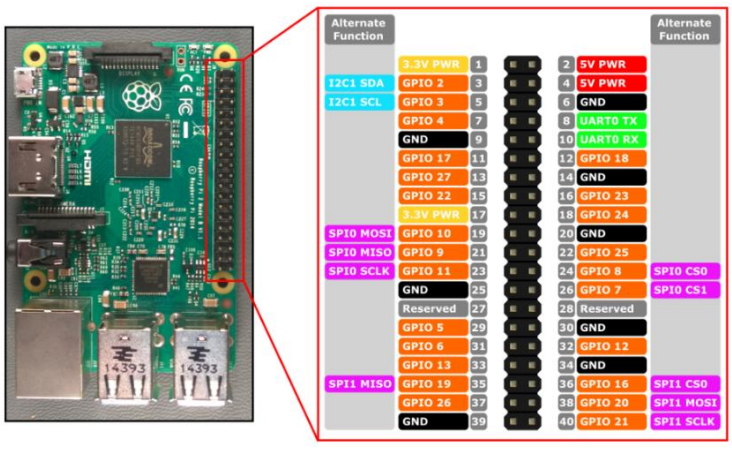
Figura 24: Algoritmo de reconocimiento y contornos

Elaborado por el equipo de trabajo.

3.2.3. Interconexión de Raspberry con driver

El sistema basado en una raspberry pi como el computador principal, este computador personal tiene pines o periféricos de salida, ver figura XX. Mediante estas salidas el sistema permite realizar comunicación serial, o pulsos, también para adquisición como el protocolo UART, y SPI. Un mapeo nos permite identificar los GPIO que pines de salida de propósito general.

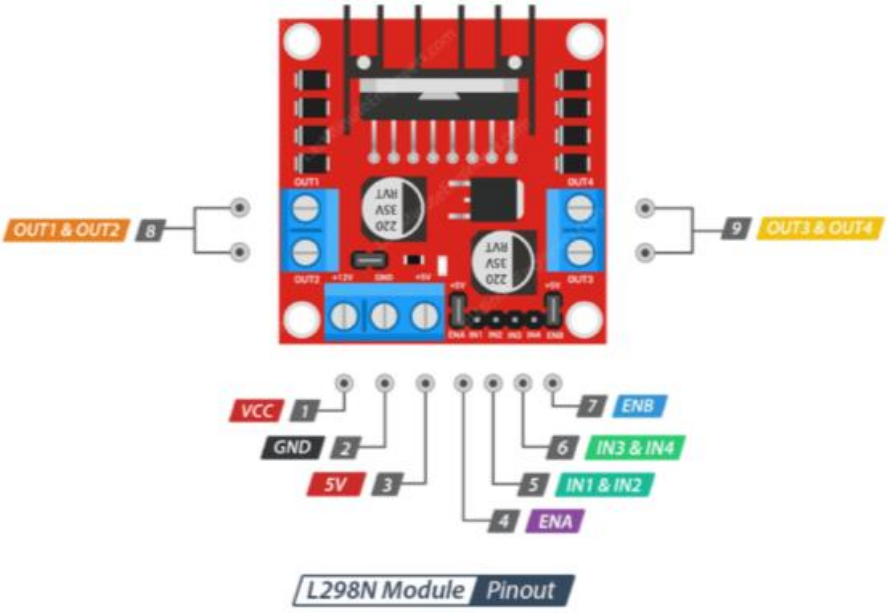
Figura 25: Mapeo de pines



Elaborado por el equipo de trabajo.

El driver LM298 de dos canales realiza el control de direcciones mediante la combinación de estados, además tiene la posibilidad de hacer el control de velocidad mediante PWM.

Figura 26: Driver LM298



Elaborado por el equipo de trabajo.

Tabla 2: Control de movimiento

Canal 1		Canal 2		En1/En2	Output
1	0	1	0	0 - 100	Adelante
0	1	0	1	0 - 100	Atrás
1	0	0	0	0 - 100	Derecha
0	0	1	0	0 - 100	Izquierda
0	0	0	0	0	Stop

Elaborado por el equipo de trabajo.

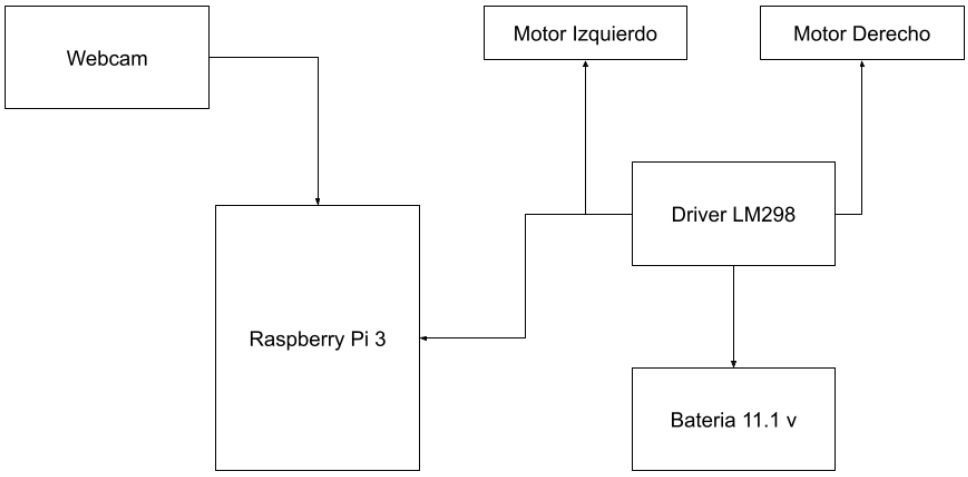
Por lo tanto, la topología de interacción entre el Raspberry y el drive se establece usando los pines de salida de propósito general, ver Tabla 3. La topología física de conexión es representada se muestra en la Figura 27.

Tabla 3: Conexión entre el Raspberry y el driver asignación de pines

Raspberry Pi 3		Driver
BOARD	BCM	-
7	4	EN1
16	23	IN4
18	24	IN3
3	2	IN2
5	3	IN1
7	4	EN2

Elaborado por el equipo de trabajo.

Figura 27: Topología de conexión



Elaborado por el equipo de trabajo.

3.2.4. Modelado matemático del sistema

Se puede formular un modelo matemático para 4 motores de DC utilizando las siguientes ecuaciones diferenciales. La primera ecuación se puede obtener a través del análisis de la malla del circuito:

$$v(t) = Ri(t) + L\frac{di(t)}{dt} + E_a(t) \tag{22}$$

Despejamos la derivada y obtenemos:

$$L\frac{di(t)}{dt} = v(t) - Ri(t) - E_a(t) \tag{23}$$

La ecuación de la sección mecánica es expresada por el siguiente modelo:

$$T_m(t) = J\frac{d\omega(t)}{dt} + B\omega(t) \tag{24}$$

- Tm (t) es el torque del motor
- B es el coeficiente de fricción equivalente al motor y la carga montados sobre el eje del motor
- J es el momento de inercia total del rotor y de la carga con relación al eje del motor
- ω(t) es la velocidad angular del motor

$$J\frac{d\omega(t)}{dt} = T_m(t) - B\omega(t) \tag{25}$$

La relación proporcional, Ka, establece que el voltaje inducido en la armadura es directamente proporcional a la velocidad angular del eje del motor, ver ecuación 26.

$$E_a(t) = K_a\omega(t); \tag{26}$$

La relación electromecánica que establece que el torque mecánico es proporcional a la corriente eléctrica, Km:

$$T_m(t) = K_m i(t) \tag{27}$$

Al obtener las 4 ecuaciones simplificadas que describen el comportamiento del motor. Luego se aplica la transformada de Laplace a las ecuaciones 25 al 27.

$$Lsi(s) = v(s) - Ri(s) - E_a(s) \tag{28}$$

$$Js\omega(s) = T_m(s) - B\omega(s) \tag{29}$$

$$E_a(s) = K_a\omega(s) \tag{30}$$

$$T_m(s) = K_m i(s) \tag{31}$$

Sustituimos la ecuación:

$$Ls\frac{T_m(s)}{K_m} = v(s) - R\frac{T_m(s)}{K_m} - K_a\omega(s) \tag{32}$$

$$v(s) = \frac{(R+L)T_m(s)}{K_m} + K_a\omega(s) \tag{33}$$

Podemos obtener la velocidad angular:

$$\omega(s) = \frac{T_m(s)}{J_s + B} \tag{34}$$

Reemplazamos en la ecuación:

$$v(s) = \frac{(R+L)T_m(s)}{K_m} + K_a\frac{T_m(s)}{J_s+B} \tag{35}$$

$$v(s) = \left(\frac{R+Ls}{K_m} + \frac{K_a}{J_s+B}\right) T_m(s) \tag{36}$$

$$v(s) = \frac{(R+Ls)(J_s+B) + K_aT_m}{K_m(J_s + B)}\ T_m(s) \tag{37}$$

Podemos obtener la función de transferencia que establece la relación entre la salida (torque) y la entrada (voltaje) de la siguiente manera:

$$\frac{T_m(s)}{v(s)} = \frac{K_m(J_s+B)}{LJs^2+(RJ+LB)s+RB+K_mK_a} \tag{38}$$

Torque voltaje:

$$\frac{T_m(s)}{v(s)} = \frac{K_m(J_s+B)}{LJs^2+(RJ+LB)s+RB+K_mK_a} \tag{39}$$

Fuerza contra electromotriz – voltaje:

$$\frac{E_a(s)}{v(s)} = \frac{K_mK_a}{LJs^2+(RJ+LB)s+RB+K_mK_a} \tag{40}$$

Corriente de armadura voltaje:

$$\frac{i(s)}{v(s)} = \frac{J_s+B}{LJs^2+(RJ+LB)s+RB+K_mK_a} \quad (41)$$

Velocidad angular – voltaje:

$$\frac{\omega(s)}{v(s)} = \frac{K_m}{LJs^2+(RJ+LB)s+RB+K_mK_a} \quad (42)$$

Posición - voltaje:

$$\frac{\theta(s)}{v(s)} = \frac{K_m}{LJs^2+(RJ+LB)s+RB+K_mK_a} \quad (43)$$

Denotemos la posición angular del i-ésimo motor por θi(t), la velocidad angular por ωi(t), el voltaje aplicado al motor por Vi(t), la corriente a través del motor por Ii(t), el par generado por el motor por Ti(t), y la inercia del motor por Ji. Se usan las siguientes ecuaciones diferenciales para modelar el comportamiento de los motores:

Para el primer motor: J1 dω1/dt = Ti - B1ω1 L1 di1/dt = Vi - Ri1I1 - Kt1ω1

Para el segundo motor: J2 dω2/dt = T2 - B2ω2 L2 di2/dt = V2 - Ri2I2 - Kt2ω2

Para el tercer motor: J3 dω3/dt = T3 - B3ω3 L3 di3/dt = V3 - Ri3I3 - Kt3ω3

Para el cuarto motor: J4 dω4/dt = T4 - B4ω4 L4 di4/dt = V4 - Ri4I4 - Kt4ω4

Donde: B1, B2, B3, B4 son los coeficientes de fricción y Ri1, Ri2, Ri3, Ri4 son la resistencia interna de los motores. Kt1, Kt2, Kt3, Kt4 son las constantes de par de los motores y L1, L2, L3, L4 son las inductancias de los motores. El par generado por cada motor se puede expresar como:

$$Ti = Ktii * Ii \quad (44)$$

Donde:

- Ktii es la constante de par del i-ésimo motor.

Las ecuaciones anteriores representan un sistema de cuatro ecuaciones diferenciales acopladas que describen el comportamiento de los cuatro motores de CC. La solución de este sistema nos dará la posición angular, la velocidad y la corriente de cada motor en función del tiempo. Para convertir el conjunto de ecuaciones diferenciales en una representación de espacio de estado, primero debemos definir las variables de estado. Se define las variables de estado de la siguiente manera:

x1 = θ1 (posición angular del primer motor)

x2 = ω1 (velocidad angular del primer motor)

x3 = I1 (corriente a través del primer motor)

x4 = θ2 (posición angular del segundo motor)

x5 = ω2 (velocidad angular del segundo motor)

x6 = I2 (corriente a través del segundo motor)

x7 = θ3 (posición angular del tercer motor)

x8 = ω3 (velocidad angular del tercer motor)

x9 = I3 (corriente por el tercer motor)

x10 = θ4 (posición angular del cuarto motor)

x11 = ω4 (velocidad angular del cuarto motor)

x12 = I4 (corriente por el cuarto motor)

Con estas variables de estado, se pueden reescribir las ecuaciones diferenciales en forma matricial de la siguiente manera:

[dx1/dt] [0 1 0 0 0 0 0 0 0 0 0 0][x1] [0]

[dx2/dt] [0 -B1/J1 0 0 0 0 0 0 0 0 0 0][x2] [Kt1/J1]

[dx3/dt] [-Kt1/L1 -Ri1/L1 0 0 0 0 0 0 0 0 0 0][x3] [1/L1]

[dx4/dt] [0 0 0 0 1 0 0 0 0 0 0 0][x4] [0]

[dx5/dt] = [0 0 0 0 -B2/J2 0 0 0 0 0 0 0][x5] + [Kt2/J2][u2]

[dx6/dt] [0 0 0 -Kt2/L2 -Ri2/L2 0 0 0 0 0 0][x6] [1/L2]

[dx7/dt] [0 0 0 0 0 0 0 1 0 0 0 0][x7] [0]

[dx8/dt] [0 0 0 0 0 0 0 -B3/J3 0 0 0 0][x8] [Kt3/J3]

[dx9/dt] [0 0 0 0 0 0 -Kt3/L3 -Ri3/L3 0 0 0][x9] [1/L3]

[dx10/dt] [0 0 0 0 0 0 0 0 0 0 1 0][x10] [0]

[dx11/dt][0 0 0 0 0 0 0 0 -B4/J4 0 0][x11] [Kt4/J4]

[dx12/dt] [0 0 0 0 0 0 0 0 -Kt4/L4 -Ri4/L4][x12] [1/L4]

donde u2 es el voltaje de entrada aplicado al segundo motor, y las matrices del lado derecho representan las matrices del sistema. En forma matricial, la representación del espacio de estados se puede escribir como:

- dx/dt = Ax + Bu y = Cx + Du

Donde:

- x es el vector de estado de tamaño 12x1, que consta de las variables de estado x1 a x12
- u es el vector de entrada de tamaño 1x1, que consiste en el voltaje aplicado al segundo motor
- y es el vector de salida de tamaño 12x1, que consta de todas las variables de estado
- A es la matriz del sistema de tamaño 12x12
- B es la matriz de entrada de tamaño 12x1
- C es la matriz de salida de tamaño 12x12
- D es la matriz feedforward de tamaño 12x1

Las matrices A, B, C y D están dadas por:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -B1/J1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -Kt1/L1 & -Ri1/L1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -B2/J2 & 0 & 0 & 0 & 0 & 0 & 0 & Kt2/J2 \\ 0 & 0 & 0 & -Kt2/L2 & -Ri2/L2 & 0 & 0 & 0 & 0 & 0 & 0 & 1/L2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -B3/J3 & 0 & 0 & 0 & Kt3/J3 \\ 0 & 0 & 0 & 0 & 0 & 0 & -Kt3/L3 & -Ri3/L3 & 0 & 0 & 0 & 1/L3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -B4/J4 & 0 & 0 & Kt4/J4 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -Kt4/L4 & -Ri4/L4 & 0 & 1/L4 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ 0 \\ 1/L1 \\ 0 \\ 0 \\ 1/L2 \\ 0 \\ 0 \\ 0 \\ 1/L4 \\ 0 \\ 0 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$D = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

La representación del espacio de estado proporciona una descripción matemática del sistema, que se puede utilizar para el diseño de análisis y control. Las variables de estado x1 a x12 representan el estado interno de los motores y su dinámica, mientras que la tensión de entrada u2 afecta el comportamiento del segundo motor.

Modelo matemático

Consideremos un sistema con cuatro motores de CC, donde cada motor se modela como una ecuación diferencial de primer orden. Supongamos que el voltaje de entrada a cada motor es u1(t), u2(t), u3(t) y u4(t), respectivamente. Supongamos también que la velocidad angular de cada motor está representada por ω1(t), ω2(t), ω3(t) y ω4(t), respectivamente.

El modelo matemático para cada motor DC está dado por:

- $J1d\omega1/dt = K1u1(t) - B1\omega1(t)$
- $J2d\omega2/dt = K2u2(t) - B2\omega2(t)$
- $J3d\omega3/dt = K3u3(t) - B3\omega3(t)$
- $J4d\omega4/dt = K4u4(t) - B4\omega4(t)$

Donde J1, J2, J3 y J4 son el momento de inercia de cada motor, K1, K2, K3 y K4 son las constantes de par de cada motor, B1, B2, B3 y B4 son los coeficientes de fricción de cada motor.

Supongamos que las entradas u1(t), u2(t), u3(t) y u4(t) están controladas por un controlador PID. La función de transferencia de un controlador PID viene dada por:

$$C(s) = Kp + Ki/s + Kd * s \quad (45)$$

Donde:

- Kp, Ki y Kd son los coeficientes de ganancia proporcional, integral y derivada del controlador, respectivamente.

La función de transferencia general del sistema se puede obtener tomando la transformada de Laplace de las ecuaciones diferenciales para cada motor y resolviendo la salida ω(s). La transformada de Laplace de la ecuación diferencial para el i-ésimo motor viene dada por:

- Jisωi(s) = KisUi(s) - Bisωi(s)
- ωi(s) = Ui(s)/(Jis + Bis)

La transformada de Laplace del voltaje de entrada ui(t) es Ui(s). La transformada de Laplace de la velocidad angular de salida ωi(t) es ωi(s). La función de transferencia global del sistema está dada por:

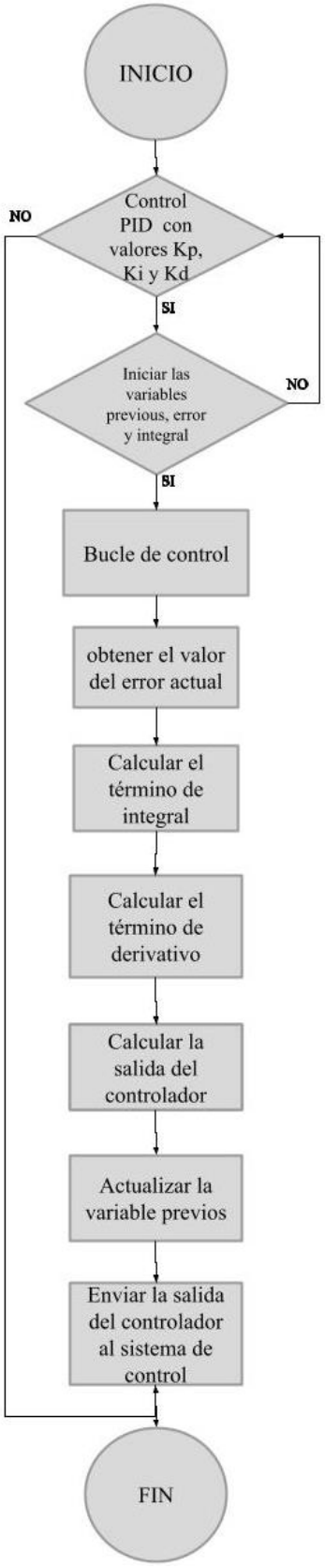
G(s) = ω1(s)/U1(s) = 1/(J1s + B1) * K1 * C(s) + ω2(s)/U2(s) = 1/(J2s + B2) * K2 * C(s) + ω3(s)/U3(s) = 1/(J3s + B3) * K3 * C(s) + ω4(s)/U4(s) = 1/(J4s + B4) * K4 * C(s)

3.2.5. Programación del Controlador PID

Para mejorar la conducción y toma de decisión del vehículo se implementa un controlador PID (proporcional, integral, derivativo). Un controlador PID es un algoritmo utilizado para controlar sistemas de control de procesos. En el método init, se establecen tres parámetros: kp, ki y kd, que son los tres coeficientes utilizados en el algoritmo PID. También se inicializan dos variables: previous_error e integral. El método update() es utilizado para actualizar el controlador con un nuevo valor de error y un intervalo de tiempo dt. En este método, se calcula el término integral sumando el error actual multiplicado por el intervalo de tiempo. También se calcula el término derivativo como la variación del error en función del tiempo, calculada como la diferencia entre el error actual y el error anterior, dividida por el intervalo de tiempo.

Finalmente, se calcula la salida del controlador como la suma de los términos proporcional, integral y derivativo, multiplicados por sus respectivos coeficientes kp, ki y kd. La variable previous_error se actualiza con el valor actual del error para su uso en el próximo cálculo del término derivativo. La función devuelve el valor de la salida del controlador. Todo este proceso se puede resumir en el siguiente diagrama de flujo.

Figura 28: Diagrama de flujos Programación del Controlador PID.

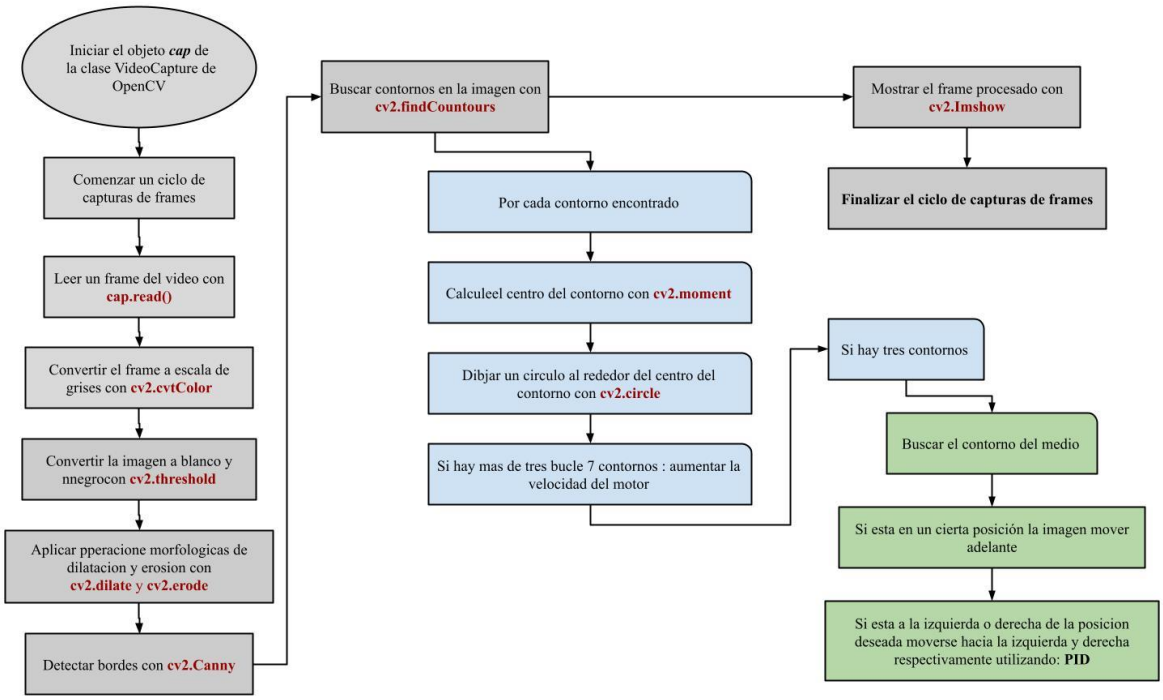


Elaborado por el equipo de trabajo.

3.3.6. Procesamiento de video y controlador PID

Este código usa la biblioteca de acceso libre OpenCV para procesar un video o una imagen. Utiliza un objeto "cap" de la clase ***VideoCapture*** de OpenCV para capturar un frame del video. Luego, convierte el frame a escala de grises con la función ***cvtColor***. Se utiliza la función threshold para convertir la imagen en blanco y negro, aplica operaciones morfológicas de dilatación y erosión para eliminar el ruido, utiliza la función ***Canny*** para detectar los bordes y busca los contornos en la imagen con ***findContours***. Por cada contorno encontrado, calcula el centro y dibuja un círculo alrededor del mismo. Si hay más de 3 contornos, aumenta la velocidad de un motor. Si hay 3 contornos, busca el contorno del medio, y si está en una cierta posición en la imagen, se mueve hacia adelante. Si está a la izquierda o derecha de la posición deseada, se mueve hacia la izquierda o derecha respectivamente utilizando el algoritmo PID.

Figura 29: Diagrama de flujos Procesamiento de video y Controlador PID



Elaborado por el equipo de trabajo.

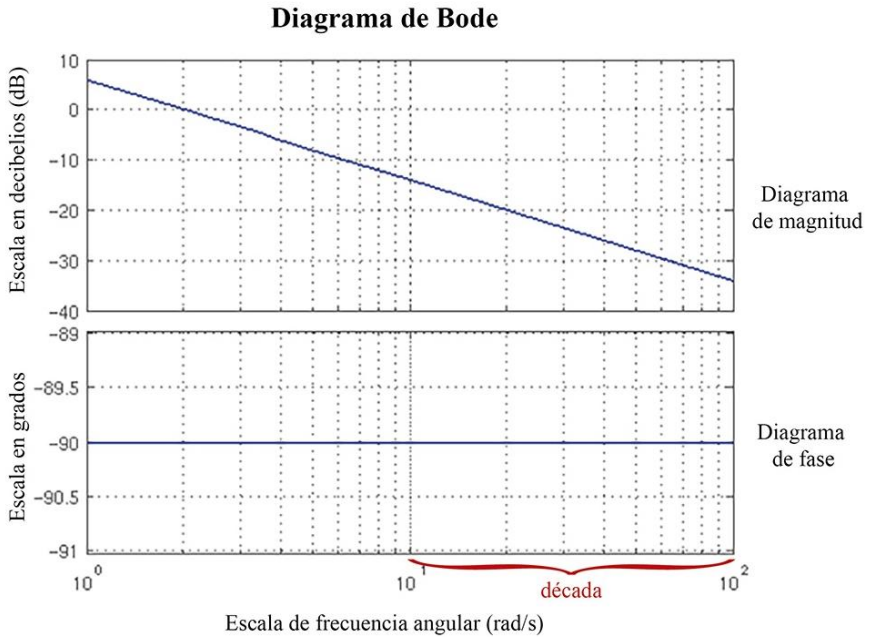
3.2.7. Diagrama de Bode

Un diagrama de Bode es una herramienta gráfica utilizada para representar la respuesta en frecuencia de un sistema. Está compuesto por un diagrama de magnitud en decibelios (dB) y un diagrama de fase en grados, ambos en función de la frecuencia. Al utilizar el diagrama de Bode, podemos analizar la estabilidad del sistema y diseñar controladores que operen dentro de las zonas de interés del proceso. Los diagramas de Bode permiten representar de manera gráfica el comportamiento de un circuito eléctrico en función de la frecuencia de la señal de excitación. Es decir, cuando un circuito es excitado con señales de frecuencia variable, los diagramas de Bode se utilizan para analizar la amplitud de la ganancia y la fase de las corrientes y voltajes en el circuito.

Los diagramas de Bode constan comúnmente de un diagrama de magnitud o ganancia y un diagrama de fase, los cuales varían según la frecuencia angular en forma de curvas por ω. Dado que esta frecuencia angular suele abarcar un amplio rango de valores, se utiliza frecuentemente una escala logarítmica en el eje horizontal para una representación más sencilla de analizar.

En cuanto al gráfico de amplitud, el eje vertical representa la ganancia del circuito expresada en decibelios (dB). En el gráfico de fase, el eje vertical muestra los ángulos de desfase expresados en grados.

Figura 30: Diagrama de Bode



https://panamahitek.com/

3.3. PLANTEAMIENTO DE LA SOLUCIÓN

Con el fin de alcanzar nuestro objetivo principal, la solución propuesta en este proyecto, como se mencionó previamente, implica la elaboración de un vehículo (solución) que pueda desplazarse de manera autónoma dentro de los límites de un carril.

Para poder lograrlo, se ha implementado una solución basada en la visión artificial, en la que una cámara adquiere imágenes de las líneas del carril. Luego, se analizan las imágenes capturadas y en base al resultado del análisis, se envía la señal de girar a la derecha, girar a la izquierda, seguir avanzando o retrocediendo. El programa desarrollado en Python ejecuta el análisis y envía las instrucciones a los motores, los cuales son programados en el lenguaje de programación C++. Todo esto es llevado a cabo por una Raspberry Pi 3 Modelo B (microcontrolador). Este microcontrolador permite analizar las imágenes capturadas utilizando las librerías OpenCV, así como generar las señales de control para el movimiento de los motores. La cámara es utilizada para capturar las imágenes y se puede utilizar junto con la Raspberry Pi 3. El vehículo es impulsado mediante una placa de control (shield) equipada con el circuito integrado L293D, el cual

permite el movimiento de los motores, el vehículo recibe señales directas desde la Raspberry Pi 3, lo que le permite moverse en la dirección adecuada.

En resumen, la propuesta del proyecto se enfoca en el desarrollo de software y modelado, y también se incluye la descripción del ensamblaje del vehículo. Para validar que la solución funciona correctamente, se ha creado un circuito de líneas para simular un carril, con un circuito dibujado. El prototipo de vehículo autónomo se pondrá en movimiento en este circuito, con el objetivo de probar y demostrar que los vehículos autónomos serán parte de nuestra vida cotidiana.

CAPÍTULO IV

RESULTADOS Y DISCUSIÓN

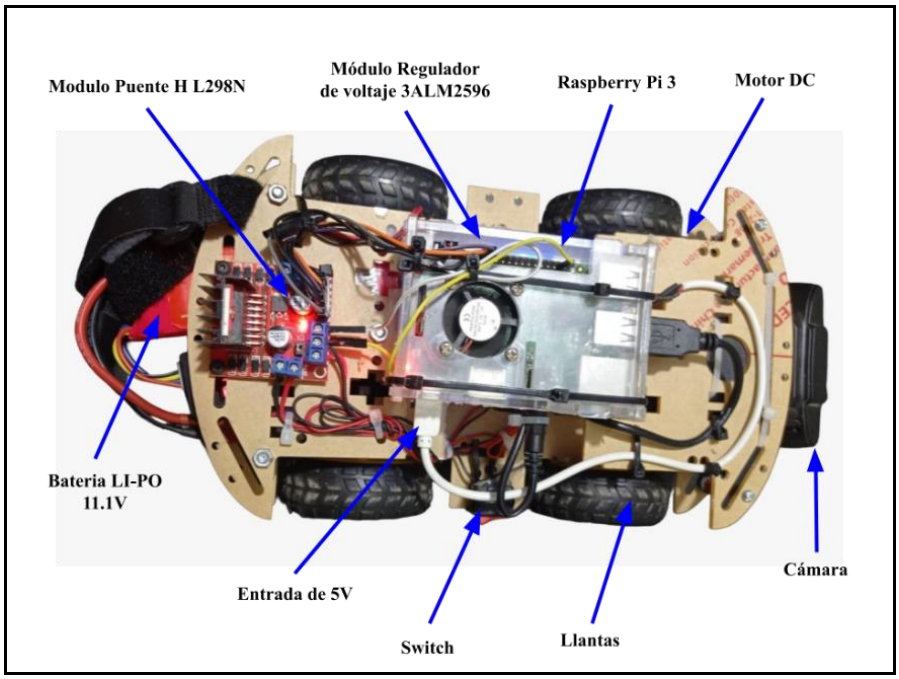
4.1. RESULTADOS

4.1.1. Prototipo Implementado

En esta sección se presenta y describe los elementos que componen el prototipo de vehículo autónomo, su funcionamiento y la arquitectura general del vehículo, se hace énfasis en el microcontrolador (Raspberry Pi 3B) y la cámara de video, como componentes fundamentales. En la figura 31, se muestra el prototipo desde una vista frontal en donde se visualiza los componentes visibles que son dispositivos electrónicos y partes mecánicas:

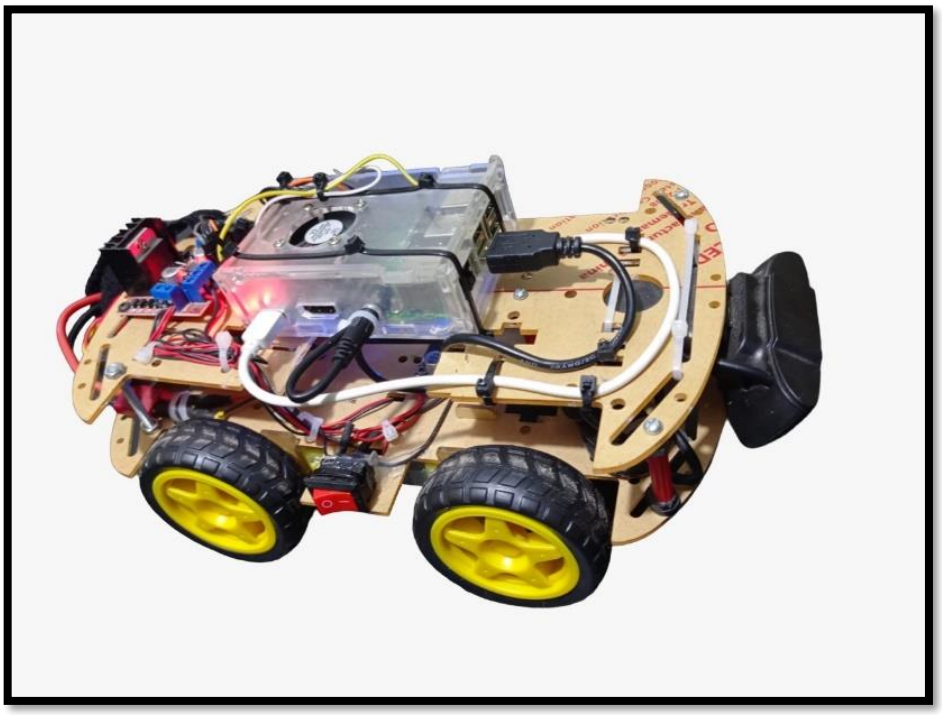
- Chasis del vehículo
- Conectores (cables)
- Soportes para el chasis
- Ruedas de goma
- Switch
- Modulo Puente H L298N
- Regulador de voltaje
- Motores DC
- Batería LI-PO
- Cámara Web
- Raspberry pi 3

Figura 31: Prototipo de vehículo Autónomo implementado



Elaborado por el equipo de trabajo.

Figura 32: Prototipo de vehículo lateral

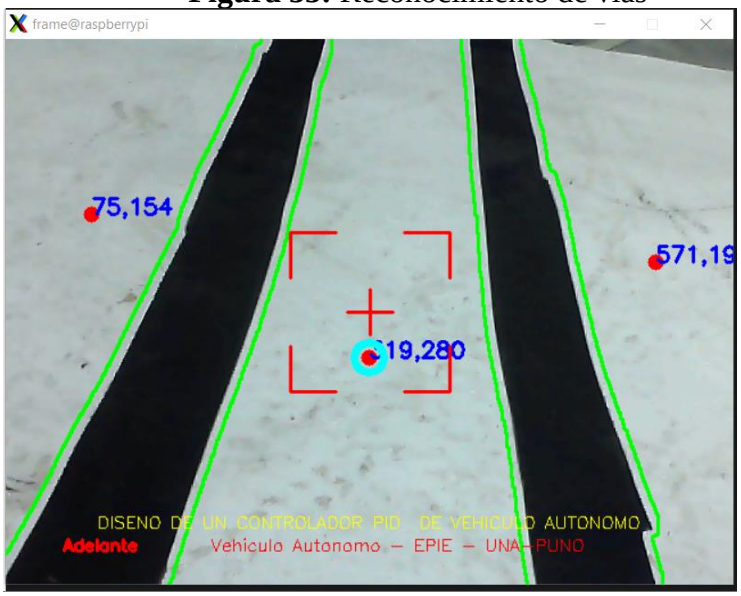


Elaborado por el equipo de trabajo.

4.1.2. Resultados preliminares

El código de reconocimiento de líneas negras, así como la identificación de puntos medios dentro de un video de resolución de 640 x 340 píxeles. Se identifica el punto central de la imagen. Ver figura 33.

Figura 33: Reconocimiento de vías



Elaborado por el equipo de trabajo.

4.1.3. Parámetros iniciales de reconocimiento

Para detección de líneas negras mediante primero el proceso de conversión de la imagen de RGB a escala de grises, se establece el rango de 50 siendo el máximo 255, esto permite eliminar ruidos externos. Los algoritmos morfológicos como la dilatación, erosión y bordes ***Canny*** identifican las líneas negras de la vía.

4.1.4. Parámetros iniciales de controlador PID

Para el vehículo para un arranque se establece de un rango de 0 a 100 valores PWM. Siendo el arranque en 27 %. Este valor es tomado como referencia como valor inicial para la marcha. La función de transferencia se define como la relación entre la salida y la entrada en el dominio de la frecuencia, típicamente representada por una transformada de Laplace. En el caso de un motor de CC, la entrada es el voltaje aplicado

al motor y la salida es la velocidad del rotor, mediante la ecuación (45) así como la repuesta es ploteada, ver Figura 34.

$$plant\_tf = Kt / (Js + b) \quad (46)$$

Calculando parámetros de motor tenemos:

Tabla 4: Recolección de datos

Parámetro	Símbolo	Valor	Unidad
Momento de Inercia	J	0.000672193	Kg.m^2
Constante de Fricción viscosa	B	0.000113	N.m.s/rad
Constante de Fuerza Electromotriz	Ka	0.2360	V/rad s
Constante del Par del Motor	Km	0.2360	N.m/A
Resistencia de Armadura	R	5.8	Ohms
Inductancia eléctrica	L	0.0015	H

Elaborado por el equipo de trabajo.

Se obtiene:

J = 6.7219e-04

b = 1.1300e-04

K = 0.2360

Donde:

- Kt: es la constante del par motor.
- Js: es el momento de inercia del rotor + carga.
- b: es el coeficiente de rozamiento viscoso del motor.
- Teniendo como función de transferencia de tiempo continuo:
- P_motor = 0.236 / (0.0006722 s + 0.000113)

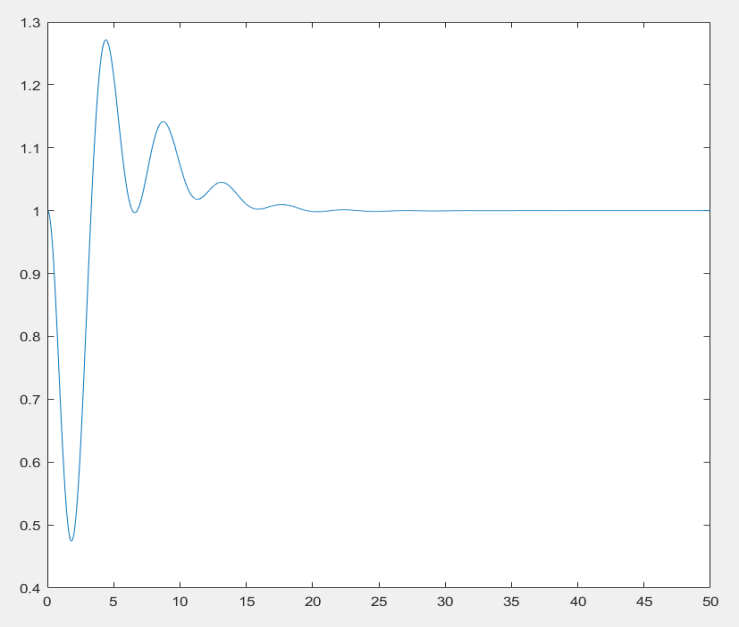
En MATLAB se ajusta el controlador PID usando el método Ziegler-Nichols, usando el siguiente comando, ver Anexo 4 donde de ejecuta programa de Matlab, usando el siguiente comando:

- C[pi,info] = pidtune(G,'PID')

Dando como resultando los valores de los parámetros:

Kp = 0.000685, Ti = 3.28, Td = 0.000334.

Figura 34: Reconocimiento de vías



Elaborado por el equipo de trabajo.

4.1.5. Prueba de movimiento en línea recta

Las pruebas se realizaron en una pista con dimensiones 1.22 m x 2.44 m visualiza la figura N° 35. Las pruebas del prototipo se realizaron usando una plataforma o pista diseñada para ejecutar el algoritmo de control.

Figura 35: Plataforma de pruebas.

Elaborado por el equipo de trabajo.

En la primer a prueba se realiza la prueba de recorrido en línea recta, en la siguiente tal como se observa las pruebas e intentos que el robot experimental.

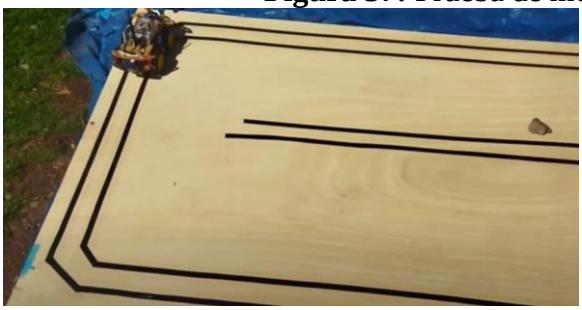
Figura 36: Prueba de movimiento en línea recta N°1

a) Punto de inicio

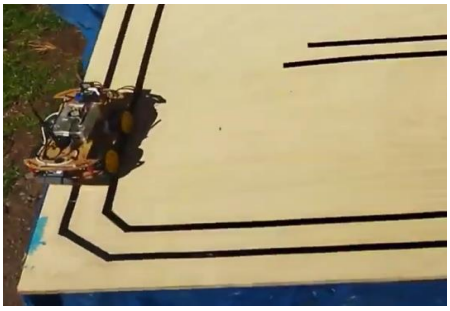
b) Punto final

Elaborado por el equipo de trabajo.

Figura 37: Prueba de movimiento en línea recta N°2



a) Punto de inicio

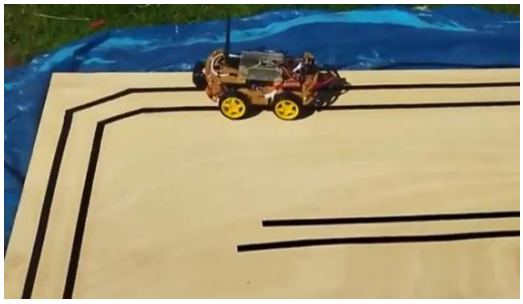


b) Punto final

Elaborado por el equipo de trabajo.

Figura 38: Prueba de movimiento en línea recta N°3

a) Punto de inicio



b) Punto final

Elaborado por el equipo de trabajo.

Figura 39: Prueba de movimiento en línea recta N°1

a) Punto de inicio

b) Punto final

Elaborado por el equipo de trabajo.

Tabla 5: Prueba de recorriendo en doble línea recta

Numero de prueba	Distancia de pista	Distancia recorrida	Tiempo del recorrido	Resultado
Prueba 01	1.97 m	1.97 m	8 segundos	Optimo
Prueba 02	80 cm	80cm	2 segundos	Optimo
Prueba 03	1.97 m	74 cm	4 segundos	Optimo
Prueba 04	1.97 m	60 cm	2 segundos	Optimo

Elaborado por el equipo de trabajo.

4.2. DISCUSIÓN

Los resultados obtenidos nos acercaron a nuestra hipótesis establecida, ya que podría ser factible que los vehículos autónomos empleen algoritmos de detección fundamentados en procesamiento digital de imágenes, para controlar la velocidad y dirección del vehículo que podría ser implementado en nuestra región de Puno.

Este proyecto tiene varias ventajas. El controlador PID es una técnica de control robusta y bien establecida que se utiliza en una variedad de aplicaciones. Al combinarlo con la realimentación de video, el sistema puede adaptarse en tiempo real a las condiciones cambiantes de la carretera y del tráfico, lo que mejora la estabilidad y la precisión en el seguimiento de la trayectoria deseada. Sin embargo, también hay algunos desafíos en este diseño. Por ejemplo, la realimentación de video puede ser afectada por condiciones de iluminación u otros factores adversos. Además, el sistema debe ser capaz de tratar con problemas de seguridad, tales como el uso de algoritmos robustos para detectar y evitar obstáculos. En general, el diseño de un controlador PID con realimentación de streaming de video es un paso prometedor hacia la implementación de sistemas de control avanzados para vehículos autónomos, pero todavía hay desafíos técnicos.

V. CONCLUSIONES

- Se implemento un controlador PID idóneo para un prototipo de un vehículo autónomo utilizando un streaming de video como realimentación para mantener una trayectoria proyectada utilizando el controlador PID para moderar la conducción del vehículo. Los motores DC emplean la realimentación de la webcam para control de posición y velocidad del vehículo.

- El diseño del controlador PID para el control de velocidad de motores y posición usando como realimentación una fuente de video. Donde se conectó el controlador PID con un sistema de procesamiento digital de imágenes para realimentación de posicionamiento. El lenguaje programación Python y la librería OpenCV facilitan una rápida implementación de algoritmos en el computador Raspberry Pi.

- En este trabajo se presenta el diseño de un controlador PID con realimentación de streaming de video para guiar un prototipo de vehículo autónomo. El controlador PID es un algoritmo de control de procesos ampliamente utilizado que permite controlar la velocidad y la posición del vehículo. La realimentación de video se utiliza para proporcionar al controlador información en tiempo real sobre la posición y el movimiento del vehículo.
- El sistema se ha probado en un prototipo de vehículo autónomo y se ha observado un buen desempeño en términos de estabilidad y precisión en el seguimiento de la trayectoria deseada. Este trabajo es un paso importante hacia la implementación de sistemas de control avanzados para vehículos autónomos.

VI. RECOMENDACIONES

La realimentación del sistema en una cámara web que envía píxeles del punto central. El problema general de la implementación es la velocidad de procesamiento, así como de la resolución del video. Para la aceleración del procesamiento del video se recomienda escalar a otro sistema embebido con superior procesamiento al Raspberry Pi. Al tener un sistema con procesador gráfico aumentaría la respuesta de procesamiento del reconocimiento de líneas en el sistema.

En la implementación que empezó de un diseño inicial con uso de un microcontrolador Arduino, se descartó su uso debido a que el puerto sería limitar el procesamiento de video y tiempos de respuesta. Por lo que se opta por simplificar el sistema para mejor respuesta de procesamiento. En el diseño de un controlador PID con realimentación de streaming de video para guiar un prototipo de vehículo autónomo, se recomienda considerar las siguientes recomendaciones:

Utilizar un algoritmo de detección de obstáculos robusto: para garantizar la seguridad del vehículo, es importante utilizar un algoritmo de detección de obstáculos confiable y preciso. Comience por investigar los diferentes tipos de controladores PID y su aplicabilidad en sistemas de control de vehículos autónomos. Continuar investigando en técnicas de aprendizaje automático para mejorar aún más el rendimiento del controlador y adaptarlo a diferentes entornos de conducción.

Realizar pruebas adicionales en diferentes condiciones de iluminación para evaluar la robustez del sistema. Investigar en la escalabilidad del sistema, para adaptarlo a diferentes tamaños y tipos de vehículos. Continuar documentando y compartiendo los resultados para contribuir al desarrollo de la comunidad de investigación en vehículos autónomos.

VII. REFERENCIAS BIBLIOGRÁFICAS

Ángel Molina García. (2017, September 8). Evaluación de Raspberry3 para adquisición de datos en entornos de laboratorio. Repository Home. https://repositorio.upct.es/bitstream/handle/10317/6470/tfg-sil-eva.pdf?sequence=1

Control tutorials for MATLAB and Simulink - Motor speed: PID controller design. (n.d.). https://ctms.engin.umich.edu/CTMS/index.php?example=MotorSpeed§ion=ControlPID

AprendePython. (2022). Python. Aprende Python. https://aprendepython.es/core/introduction/python

Bazanella, A. S.; Silva Júnior, J. M. G. Da. “Sistemas de Controle”: princípios e métodos de projeto. 1.ed. Editora UFRGS, 2005.

Contaval. (2016, February 18). ¿Que es la vision artificial Y para Que sirve? CONTAVAL. https://www.contaval.es/que-es-la-vision-artificial-y-para-que-sirve/

C, S. (2021, November 3). Modelo de motor DC. Control Automático Educación. https://controlautomaticoeducacion.com/analisis-de-sistemas/modelo-de-motor-dc/

Driver Puente H l298n 2a. (n.d.). Naylamp Mechatronics - Perú. https://naylampmechatronics.com/drivers/11-driver-puente-h-l298n.html

Gallardo García Hugo, Juárez Parga José Miguel. (2011, November 3). Instituto Tecnológico y de Estudios Superiores Monterrey Campus San Luís Potosí. San Luis Potosí, S.L.P. México.

Gaudenz Boesch M. (2022). The 12 Most Popular Computer Vision Tools in 2022. viso.ai. https://viso.ai/computer-vision/the-most-popular-computer-vision-tools/

Gonzalez, F. (2019). Propuesta de vehículo autónomo para discapacitados en la región Piura (Tesis para optar el título de Ingeniero Mecánico-Eléctrico). Universidad de Piura. Facultad de Ingeniería. Programa Académico de Ingeniería Mecánico-Eléctrica. Piura, Perú.

González, A. G. (2023, January 5). *Guía práctica para construir UN diagrama de bode*. Panama Hitek. https://panamahitek.com/guia-practica-para-construir-un-diagrama-de-bode/

IBM Corp. (2022). ¿Qué es la visión artificial? IBM. www.ibm.com

Juan Manuel Alonso Weber Leganés. (2020). Desarrollo de un Coche a Escala reducida para Aprendizaje de Conducción Autónoma mediante Técnicas de Deep Learning [Master's thesis].

Julio Labora Gómez. (2018, August). CONTROL DE UN VEHÍCULO AUTÓNOMO EN UN ENTORNO LIMITADO. Batería LiPo de 3 celdas de alta descarga, 26.

J. Canny, "A Computational Approach to Edge Detection," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679-698, Nov. 1986, doi: 10.1109/TPAMI.1986.4767851.

K. Piemngam, I. Nilkhamhang and P. Bunnun,(2019). "Development of Autonomous Mobile Robot Platform with Mecanum Wheels," 2019 First International Symposium on Instrumentation, Control, Artificial Intelligence, and Robotics

(ICA-SYMP), pp. 90-93, doi: 10.1109/ICA-SYMP.2019.8646085. Bangkok, Thailand.

Luis Alfaro1, José Roca, Enrique Poblet. (2003). Cámaras digitales y transferencia de imágenes al ordenador. http://www.patologia.es. http://www.patologia.es/volumen36/vol36-num2/36-2n02.htm

LM2596 Regulator step down 25W 3a. (2022, December 10). UNIT Electronics. https://uelectronics.com/producto/modulo-regulador-ajustable-lm2596-dc-dc-step-down-3a-1-25-30v/

Manuel Guillermo Quijano Ortega, Carlos Gerardo Hernendez Capacho. (2009). Obtención experimental de los parametros del motor que se utilizara en el sistema de locomocion de una esfera rodante. Universidad pontificia bolivariana. Escuela de ingenieria y administracion / facultad de ingenieria electronica

MATLAB® 9.2. (n.d.). Control tutorials for MATLAB and Simulink - Motor speed: PID controller design. https://ctms.engin.umich.edu/. https://ctms.engin.umich.edu/CTMS/index.php?example=MotorSpeed§ion=ControlPID

M. E. Abed, M. Aly, H. H. Ammar and R. Shalaby, (2020). "Steering Control for Autonomous Vehicles Using PID Control with Gradient Descent Tuning and Behavioral Cloning," 2020 2nd Novel Intelligent and Leading Emerging Sciences Conference (NILES), pp. 583-587, doi: 10.1109/NILES50944.2020.9257946. Giza, Egypt.

Mandy Sleight, M. (2021, June 22). How do self-driving cars work? Bankrate. https://www.bankrate.com/insurance/car/how-do-self-driving-cars-work/

Mrinal Tyagi, M. (2021, July 18). Segmentación de imágenes. Towardsdatascience. https://towardsdatascience.com/image-segmentation-part-1-9f3db1ac1c50

Muñoz, J. D. (2022, November 18). Introducción a python. PLEDIN 3.0. https://plataforma.josedomingo.org/pledin/cursos/python3/curso/u01/

Organización IBM. (n.d.). ¿Qué es la visión artificial? IBM - España. https://www.ibm.com/

Python. (2022). El tutorial de Python. Python documentation. https://docs.python.org/es/3/tutorial/

Roberto Carlos Sierra Obregón. (2020). Diseño y construcción de un sistema de control para un aparato rehabilitador físico. Repositorio Institucional del Tecnológico Nacional de México (RI-TecNM). https://rinacional.tecnm.mx/bitstream/TecNM/1296/1/_Roberto%20Carlos%20

Roberto Solé, R. (2021, July 18). Raspberry Pi: Que es, para Que sirve Y Que podemos hacer. Profesional Review. https://www.profesionalreview.com/2021/07/18/que-es-raspberry-pi/

Rubén Estrada Marmolejo. (2018, June 2). OpenCV findContours. HeTPro-Tutoriales. https://hetpro-store.com/TUTORIALES/opencv-findcontours/

Self-Driving Cars Introduction. Education ecosystem. (2020, 9 Abril). Recuperado en 12 de Mayo 2022 de https://educationecosystem.com/guides/artificial-intelligence/self-driving-cars/history

Siddartha Khastgir, Simon Brewerton, John Thomas, Paul Jennings, (2021) “Systems Approach to Creating Test Scenarios for Automated Driving Systems, Reliability

Engineering & System Safety", Volume 215, 107610, ISSN 0951-8320, https://doi.org/10.1016/j.ress.2021.107610. UK.

Sihuacollo, J. (2016). Diseño de un sistema de detección y conteo mediante el procesamiento digital de imágenes para ovas de trucha en el centro de investigación y producción(CIP) Chucuito UNA-PUNO (Tesis de pregrado). Universidad Nacional del Altiplano, Puno, Perú. pg.48

Siti nur wahidah / binti abdul wahab. (2014, july). Motor control system development using microcontroller based on pid controller. Universiti Tun Hussein Onn Malaysia. Faculty of Electrical and Electronic Engineering

Sleight, M., 2022. How Do Self-driving Cars Work? | Bankrate. [online] Bankrate.Recuperado de: https://www.bankrate.com/insurance/car/how-do-self-driving-cars-work/

Sociedad de Ingenieros Automotrices (SAE). (2022, September 6). What is an Autonomous Car? synopsys. https://www.synopsys.com/automotive/what-is-autonomous-car.html

Talos-robotics/Robowheels. (n.d.). GitHub. https://github.com/talos-robotics/Robowheels.

(n.d.). Education Ecosystem | The go-to platform for project-based learning. https://educationecosystem.com/guides/artificial-intelligence/self-driving-cars/history

B. Cardoso et al. (2020). "A Large-Scale Mapping Method Based on Deep Neural Networks Applied to Self-Driving Car Localization," 2020 International Joint

Conference on Neural Networks (IJCNN), pp. 1-8, doi: 10.1109/IJCNN48605.2020.9207449. Glasgow, UK.

Wikilibros. (2021, October 22). (https://es.wikibooks.org/wiki/Rob%C3%B3tica/Puente_H

ANEXOS

Anexo 1: Raspberry Configuración

UPDATE TIME:
sudo date -s "$(wget -qSO- --max-redirect=0 google.com 2>&1 | grep Date: | cut -d' ' -f5-8)Z"

UP: OpenCV Installation
1. sudo apt-get update && sudo apt-get upgrade && sudo rpi-update
2. sudo nano /etc/dphys-swapfile
CONF_SWAPSIZE=2048
3. sudo apt-get install build-essential cmake pkg-config
4. sudo apt-get install libjpeg-dev libtiff5-dev libjasper-dev libpng12-dev
5. sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libv4l-dev
6. sudo apt-get install libxvidcore-dev libx264-dev
7. sudo apt-get install libgtk2.0-dev libgtk-3-dev
8. sudo apt-get install libatlas-base-dev gfortran
9. wget -O opencv.zip https://github.com/opencv/opencv/archive/4.1.0.zip
10. wget -O opencv_contrib.zip https://github.com/opencv/opencv_contrib/archive/4.1.0.zip
11. unzip opencv.zip
12. unzip opencv_contrib.zip
13. sudo pip3 install numpy
14. cd ~/opencv-4.1.0/
15. mkdir build
16. cd build
17.
*cmake -D CMAKE_BUILD_TYPE=RELEASE *
*-D CMAKE_INSTALL_PREFIX=/usr/local *
*-D INSTALL_PYTHON_EXAMPLES=ON *
*-D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib-4.1.0/modules *
-D BUILD_EXAMPLES=ON ..
18. make -j4
19. sudo make install && sudo ldconfig
20. sudo reboot
Referencias de Instalación: https://raspberrypi-guide.github.io/programming/install-opencv

Anexo 2: Código en Python 3 implementado segmentado.

* Importación de librerías y declaración de variables

➔ *Librerías*

```
numpy  //  Librería Numpy para operaciones matemáticas
cv2        //  Librería OpenCV
imutils   // Utiitario
serial     //comunicación
time      //tiempo de acceso y conversiones
struct    //Interpreta byte como paquete de datos binarios
```

➔ *Declaración de variables parámetros*

```
x = 640
y =  480
```

➔ *Llamado del archivo entrenado y fuente de video.*

```
Cascade = cv2.CascadeClassifier("cascade8x825s.xml")
camera = cv2.VideoCapture('recortado.mp4')
```

* Funciones de procesamiento

➔ *Lectura de la fuente de video*

```
ret, frame = vid.read()
if frame is None:
break
```

➔ *Captura de video de la cámara web*

```
def __init__(self):
    self.video_capture = cv2.VideoCapture(0)
    self.current_frame = self.video_capture.read()[1]
  def start(self):
    threading.Thread(target=self._update_frame, args=()).start()

  def _update_frame(self):
    while True:
      self.current_frame = self.video_capture.read()[1]

  def get_current_frame(self):
    return self.current_frame

webcam_capture = WebcamCapture()
webcam_capture.start()
```

➔ ***Conexión de los pines del Raspberry pi***

```
GPIO.setmode(GPIO.BCM)
Motor1A = 24
Motor1B = 23
Motor1E = 4
Motor2A = 3 # 27
Motor2B = 2 # 28
Motor2E = 17

GPIO.setup(Motor1A,GPIO.OUT)
GPIO.setup(Motor1B,GPIO.OUT)
GPIO.setup(Motor1E,GPIO.OUT)
```

```
GPIO.setup(Motor2A,GPIO.OUT)
GPIO.setup(Motor2B,GPIO.OUT)
GPIO.setup(Motor2E,GPIO.OUT)

pi_pwm_1 = GPIO.PWM(Motor1E,1000)          # create PWM instance with frequency
pi_pwm_1.start(0)                          # start PWM of required Duty Cycle

def izquierda(speed):        #Motor 2 positive and negative rotation
  pi_pwm_1.ChangeDutyCycle(speed)
  dir = "Izquierda"
  GPIO.output(Motor1A,GPIO.LOW)
  GPIO.output(Motor1B,GPIO.LOW)
  GPIO.output(Motor2A,GPIO.HIGH)
  GPIO.output(Motor2B,GPIO.LOW)
  print(dir)

def atras(speed):        #Motor 2 positive and negative rotation
  pi_pwm_1.ChangeDutyCycle(speed)
  dir = "Atras"
  GPIO.output(Motor1A,GPIO.LOW)
  GPIO.output(Motor1B,GPIO.HIGH)
  GPIO.output(Motor2A,GPIO.LOW)
  GPIO.output(Motor2B,GPIO.HIGH)
  print(dir)

def adelante(speed):        #Motor 2 positive and negative rotation
  pi_pwm_1.ChangeDutyCycle(speed)
  dir = "Adelante"
  GPIO.output(Motor1A,GPIO.HIGH)
  GPIO.output(Motor1B,GPIO.LOW)
  GPIO.output(Motor2A,GPIO.HIGH)
  GPIO.output(Motor2B,GPIO.LOW)
  print(dir)

def derecha(speed):        #Motor 2 positive and negative rotation
  pi_pwm_1.ChangeDutyCycle(speed)
  dir = "Derecha"
  GPIO.output(Motor1A,GPIO.HIGH)
  GPIO.output(Motor1B,GPIO.LOW)
  GPIO.output(Motor2A,GPIO.LOW)
  GPIO.output(Motor2B,GPIO.LOW)
  print(dir)

def stop(speed):        #Motor 2 positive and negative rotation
  pi_pwm_1.ChangeDutyCycle(speed)
  dir = "Stop"
  GPIO.output(Motor1A,GPIO.LOW)
  GPIO.output(Motor1B,GPIO.LOW)
  GPIO.output(Motor2A,GPIO.LOW)
  GPIO.output(Motor2B,GPIO.LOW)
  print(dir)
```

➔ *Introducir losparametros PID*

```
class PIDController:
  def __init__(self, kp, ki, kd):
```

```
        self.kp = kp
        self.ki = ki
        self.kd = kd
        self.previous_error = 0
        self.integral = 0

    def update(self, current_error, dt):
        self.integral += current_error * dt
        derivative = (current_error - self.previous_error) / dt
        output = self.kp * current_error + self.ki * self.integral + self.kd * derivative
        self.previous_error = current_error
        return output

desired_position = 320
speed = 0;

# Set the PID controller constants for the position controller
#  Kp = 0.0168, Ki = 0.0377, Kd = 0.00188
kp_position = 0.0168
ki_position = 0.0377
kd_position = 0.00188
pid_position = PIDController(kp_position, ki_position, kd_position)
```

➔ *Deteccion del objeto*

```
while True:
    #ret, frame = cap.read()
    frame = webcam_capture.get_current_frame()
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    bw = cv2.threshold(gray,50, 255, cv2.THRESH_BINARY)[1]
    mask = cv2.dilate(bw, None, iterations=1)
    mask = cv2.erode(mask,None, iterations=8)
    edged = cv2.Canny(mask, 60, 200)
    contours, hierarchy = cv2.findContours(edged,cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_NONE)
    center = None
    cv2.drawContours(frame, contours,-1, (0, 255, 0), 2)
    cnts = cv2.findContours(mask.copy(), cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
    cnts = imutils.grab_contours(cnts)
    for c in cnts:
        M = cv2.moments(c)
        #M = cv2.moments(c)
        if M["m00"] != 0:
            cX = int(M["m10"] / M["m00"])
            cY = int(M["m01"] / M["m00"])
        # draw the center of the shape on the image
            cv2.circle(frame, (cX, cY), 7, (0, 0, 255), -1)
            cv2.putText(frame,str(cX) + ','+ str(cY),(cX,cY),cv2.FONT_HERSHEY_SIMPLEX, 0.65, (255,0,0), 2)
        # compute the center of the contour
```

➔ *Seguimiento de objeto y control de movimiento*

```
if len(cnts) > 3:
        speed = 25
        speed += 2
        if(speed > 30):
            speed = 30
        atras(speed)
```

```
        if len(cnts) == 3:
            N = cv2.moments(cnts[1])
            sX = int(N["m10"] / N["m00"])
            sY = int(N["m01"] / N["m00"])
            radius = 13
            cv2.circle(frame, (sX,sY), radius, (255, 255, 0), 6)
            if(sX >= 310 and sX <= 330):
                d = "Adelante"
                cv2.putText(frame,d,(50,450), cv2.FONT_HERSHEY_SIMPLEX, 0.50, (0,0,255), 2)
                speed = 28
                speed += 1
                adelante(speed)
                if(speed > 70):
                    speed = 70

            if(sX < 310):
                d = "Izquierda"
                cv2.putText(frame,d,(50,450), cv2.FONT_HERSHEY_SIMPLEX, 0.50, (0,0,255), 2)
                dt = 0.2
                error_position = desired_position - sX #current_position
                output_position = pid_position.update(error_position, dt)
                if(output_position > 40):
                    output_position = 40
                izquierda(30)
                #izquierda(output_position)
                time.sleep(dt)

            if(sX > 330):
                d = "Derecha"
                cv2.putText(frame,d,(50,450), cv2.FONT_HERSHEY_SIMPLEX, 0.50, (0,0,255), 2)
                cv2.putText(frame,d,(50,450), cv2.FONT_HERSHEY_SIMPLEX, 0.50, (0,0,255), 2)
                dt = 0.2
                error_position = desired_position - sX #current_position
                output_position = pid_position.update(abs(error_position), dt)
                if(output_position > 40):
                    output_position = 40
                derecha(30)
                time.sleep(dt)

    print("|","Coord:",sX,",",sY,"|","Dirección:",d,"|","Error:",error_position,"|","Vel_PWM",output_positio
    n,"|") #,line)
            # Puntero de referencia
```

➔ *Puntero de referencia*

```
cv2.line(frame, pt1=(320,220), pt2=(320,260), color=(0,0,255), thickness=2)
cv2.line(frame, pt1=(300,240), pt2=(340,240), color=(0,0,255), thickness=2)
cv2.line(frame, pt1=(250,170), pt2=(290,170), color=(0,0,255), thickness=2)
cv2.line(frame, pt1=(250,170), pt2=(250,210), color=(0,0,255), thickness=2)
cv2.line(frame, pt1=(250,310), pt2=(290,310), color=(0,0,255), thickness=2)
cv2.line(frame, pt1=(250,310), pt2=(250,270), color=(0,0,255), thickness=2)
cv2.line(frame, pt1=(350,170), pt2=(390,170), color=(0,0,255), thickness=2)
cv2.line(frame, pt1=(390,170), pt2=(390,210), color=(0,0,255), thickness=2)
cv2.line(frame, pt1=(350,310), pt2=(390,310), color=(0,0,255), thickness=2)
cv2.line(frame, pt1=(390,310), pt2=(390,270), color=(0,0,255), thickness=2)
```

➔ *Lectura de la fuente de vídeo y conversión a formato HSV*

```
vid_gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
bw = cv2.threshold(vid_gray,170, 255, cv2.THRESH_BINARY)[1]
mask = cv2.dilate(bw, None, iterations=6)
edged = cv2.Canny(mask, 60, 200)
contours, hierarchy = cv2.findContours(edged,cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_NONE)
center = None
cv2.drawContours(frame, contours,-1, (0, 255, 0), 2)
cnts = cv2.findContours(mask.copy(), cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
cnts = imutils.grab_contours(cnts)
```

➔ *Cálculo del centro del contorno*

```
cX = int(M["m10"] / M["m00"])
cY = int(M["m01"] / M["m00"])
```

➔ *Dibujar el centro de la forma de la imagen*

```
cv2.circle(frame, (cX, cY), 7, (0, 0, 255), -1)
cv2.putText(frame,str(cX) + ','+ str(cY),(cX,cY),cv2.FONT_HERSHEY_SIMPLEX, 0.65, (255,0,0), 2)
if len(cnts) == 3:
        area_0 = cv2.contourArea(cnts[0])/1000
        area_1 = cv2.contourArea(cnts[1])/1000
        area_2 = cv2.contourArea(cnts[2])/1000
        N = cv2.moments(cnts[1])
        sX = int(N["m10"] / N["m00"])
        sY = int(N["m01"] / N["m00"])
        radius = 14
        cv2.circle(frame, (sX,sY), radius, (255, 255, 0), 6)
```

➔ *Posición del vehículo y etiquetado*

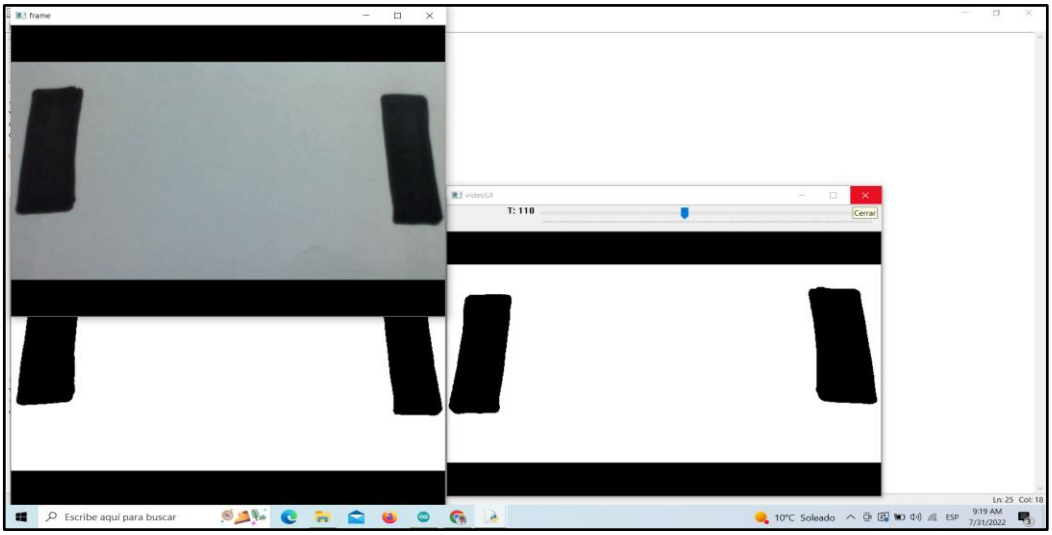
```
if(sX >= 310 and sX <= 330):
        d = "Adelante"
        cv2.putText(frame,d,(50,450), cv2.FONT_HERSHEY_SIMPLEX, 0.50, (0,0,255), 2)
if(sX < 310):
        d = "Izquierda"
        cv2.putText(frame,d,(50,450), cv2.FONT_HERSHEY_SIMPLEX, 0.50, (0,0,255), 2)
if(sX > 330):
        d = "Derecha"
        cv2.putText(frame,d,(50,450), cv2.FONT_HERSHEY_SIMPLEX, 0.50, (0,0,255), 2)
cv2.putText(frame,"Vehículo Autónomo - EPIE - UNA-PUNO",(180,450),
        cv2.FONT_HERSHEY_SIMPLEX, 0.50, (0,0,255), 1)
cv2.putText(frame,"DISEÑO DE UN CONTROLADOR PID  DE VEHÍCULO  AUTÓNOMO",(80,430),
        cv2.FONT_HERSHEY_SIMPLEX, 0.50, (0,255,255), 1)
```

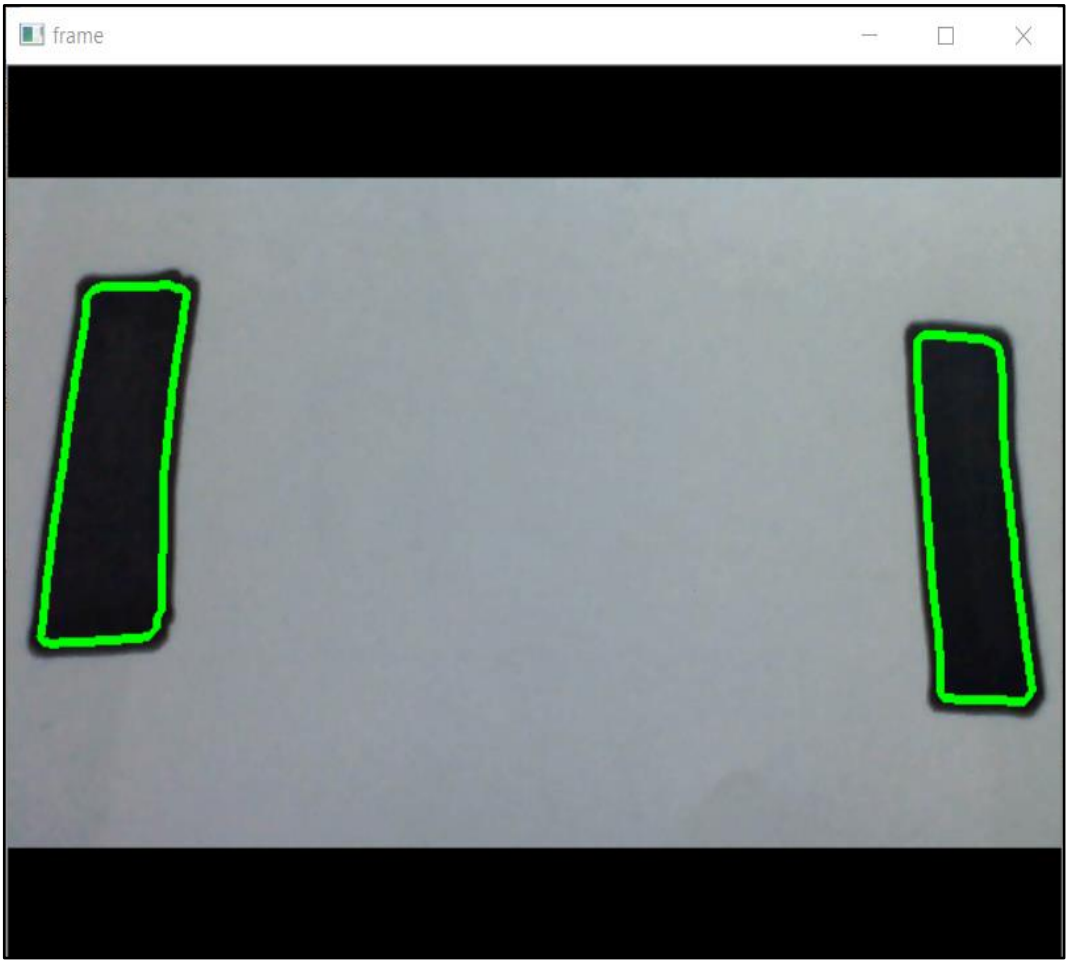
* Resultado de procesamiento

➔ *Visualización con función de terminación de proceso y salida.*

```
cv2.imshow("Frame", frame)  // Visualización
key = cv2.waitKey(10) & 0xFF // Función de salida
if key == ord("q"):
    break
cv2.destroyAllWindows()     // Fin de programa
```

Anexo 3: Extracción de color e identificación de línea negras.

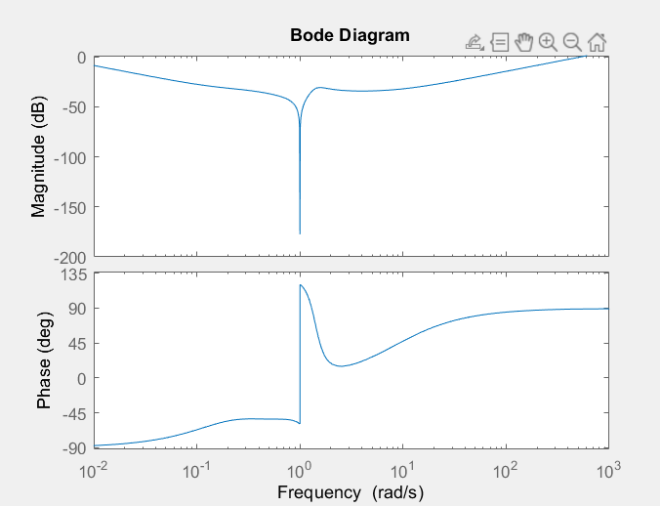


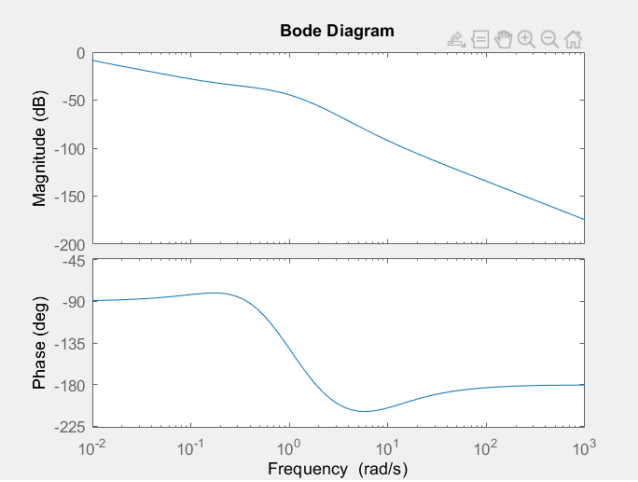


Anexo 4: Calculo de los parámetros PID

```
% Define las constantes del sistema
J1 = 0.000672193; % momento de inercia del motor 1
J2 = 0.000672193; % momento de inercia del motor 2
J3 = 0.000672193; % momento de inercia del motor 3
J4 = 0.000672193; % momento de inercia del motor 4
K1 = 0.2360; % constante de torque del motor 1
K2 = 0.2360; % constante de torque del motor 2
K3 = 0.2360; % constante de torque del motor 3
K4 = 0.2360; % constante de torque del motor 4
B1 = 0.000113; % coeficiente de fricción del motor 1
B2 = 0.000113; % coeficiente de fricción del motor 2
B3 = 0.000113; % coeficiente de fricción del motor 3
B4 = 0.000113; % coeficiente de fricción del motor 4
Kp = 0.0168; % ganancia proporcional del controlador PID
Ki = 0.0037; % ganancia integral del controlador PID
Kd = 0.00188; % ganancia derivativa del controlador PID

% Define la función de transferencia del controlador PID
s = tf('s');
C = Kp + Ki/s + Kd*s;

% Define la función de transferencia del sistema
G = tf([K1*J2*J3*J4, K2*J1*J3*J4, K3*J1*J2*J4, K4*J1*J2*J3],
[(J1*J2*J3*J4*B1*B2*B3*B4),
((J1*J2*J3+B1*J2*J3+B2*J1*J3+B3*J1*J2)*B4 +
(J1*J2+J1*J3+J2*J3)*B1*B2*B3), ((J1+J2+J3)*B1*B2*B3*B4 +
(J1*J2*B3+J1*J3*B2+J2*J3*B1)*B4 +
(J1*J2*J3+B1*J2*J3+B2*J1*J3+B3*J1*J2)*K1*K2*K3*K4),
K1*K2*K3*K4*B1*B2*B3*B4]);

% Define el sistema de lazo cerrado
T = feedback(G*C, 1);

% Ajusta el controlador PID usando el método Ziegler-Nichols
% [C_pi,info] = pidtune(G,'PID');
% C = C_pi;

% Simula el sistema con una entrada de paso
t = 0:0.01:50;
u = ones(size(t));
[y, t] = lsim(T, u, t);

% Grafica la respuesta del sistema
%plot(t, y);
%bode (G);
sisotool (G);
```

Anexo 5: Diagrama de Bode, **l**a respuesta en frecuencia del sistema. Una que corresponde con la magnitud de dicha función y otra que corresponde con la fase.





Universidad Nacional del Altiplano Puno | VRI Vicerrectorado de Investigación | Repositorio Institucional

DECLARACIÓN JURADA DE AUTENTICIDAD DE TESIS

Por el presente documento, Yo Mariela Laqui Huanca, Flor Brighitt Pampa Silca, identificado con DNI 70095238, 76866825 en mi condición de egresado de:

☒ **Escuela Profesional,** ☐ **Programa de Segunda Especialidad,** ☐ **Programa de Maestría o Doctorado**

INGENIERIA ELECTRONICA,

informo que he elaborado el/la ☒ **Tesis** o ☐ **Trabajo de Investigación** denominada:

"DISEÑO DE UN CONTROLADOR PID CON REALIMENTACION DE STREAMING DE VIDEO PARA GUIADO DE UN PROTOTIPO DE VEHICULO AUTONOMO"

Es un tema original.

Declaro que el presente trabajo de tesis es elaborado por mi persona y **no existe plagio/copia** de ninguna naturaleza, en especial de otro documento de investigación (tesis, revista, texto, congreso, o similar) presentado por persona natural o jurídica alguna ante instituciones académicas, profesionales, de investigación o similares, en el país o en el extranjero.

Dejo constancia que las citas de otros autores han sido debidamente identificadas en el trabajo de investigación, por lo que no asumiré como suyas las opiniones vertidas por terceros, ya sea de fuentes encontradas en medios escritos, digitales o Internet.

Asimismo, ratifico que soy plenamente consciente de todo el contenido de la tesis y asumo la responsabilidad de cualquier error u omisión en el documento, así como de las connotaciones éticas y legales involucradas.

En caso de incumplimiento de esta declaración, me someto a las disposiciones legales vigentes y a las sanciones correspondientes de igual forma me someto a las sanciones establecidas en las Directivas y otras normas internas, así como las que me alcancen del Código Civil y Normas Legales conexas por el incumplimiento del presente compromiso

Puno 15 de mayo del 2023

FIRMA (obligatoria) Huella

FIRMA (obligatoria) Huella

Universidad Nacional del Altiplano Puno | Vicerrectorado de Investigación | Repositorio Institucional

AUTORIZACIÓN PARA EL DEPÓSITO DE TESIS O TRABAJO DE INVESTIGACIÓN EN EL REPOSITORIO INSTITUCIONAL

Por el presente documento, Yo Mariela Laqui Huanca , Flor Brighitt Pampa Sulca, identificado con DNI 70095238, 76866825 en mi condición de egresado de:

☒ **Escuela Profesional,** ☐ **Programa de Segunda Especialidad,** ☐ **Programa de Maestría o Doctorado**

INGENIERIA ELECTRONICA,

informo que he elaborado el/la ☒ **Tesis** o ☐ **Trabajo de Investigación** denominada:

"Diseño de un controlador PID con realimentación de streaming de video para guiado de un prototipo de vehículo autónomo"

para la obtención de ☐**Grado,** ☒ **Título Profesional** o ☐ **Segunda Especialidad.**

Por medio del presente documento, afirmo y garantizo ser el legítimo, único y exclusivo titular de todos los derechos de propiedad intelectual sobre los documentos arriba mencionados, las obras, los contenidos, los productos y/o las creaciones en general (en adelante, los "Contenidos") que serán incluidos en el repositorio institucional de la Universidad Nacional del Altiplano de Puno.

También, doy seguridad de que los contenidos entregados se encuentran libres de toda contraseña, restricción o medida tecnológica de protección, con la finalidad de permitir que se puedan leer, descargar, reproducir, distribuir, imprimir, buscar y enlazar los textos completos, sin limitación alguna.

Autorizo a la Universidad Nacional del Altiplano de Puno a publicar los Contenidos en el Repositorio Institucional y, en consecuencia, en el Repositorio Nacional Digital de Ciencia, Tecnología e Innovación de Acceso Abierto, sobre la base de lo establecido en la Ley N° 30035, sus normas reglamentarias, modificatorias, sustitutorias y conexas, y de acuerdo con las políticas de acceso abierto que la Universidad aplique en relación con sus Repositorios Institucionales. Autorizo expresamente toda consulta y uso de los Contenidos, por parte de cualquier persona, por el tiempo de duración de los derechos patrimoniales de autor y derechos conexos, a título gratuito y a nivel mundial.

En consecuencia, la Universidad tendrá la posibilidad de divulgar y difundir los Contenidos, de manera total o parcial, sin limitación alguna y sin derecho a pago de contraprestación, remuneración ni regalía alguna a favor mío; en los medios, canales y plataformas que la Universidad y/o el Estado de la República del Perú determinen, a nivel mundial, sin restricción geográfica alguna y de manera indefinida, pudiendo crear y/o extraer los metadatos sobre los Contenidos, e incluir los Contenidos en los índices y buscadores que estimen necesarios para promover su difusión.

Autorizo que los Contenidos sean puestos a disposición del público a través de la siguiente licencia:

Creative Commons Reconocimiento-NoComercial-CompartirIgual 4.0 Internacional. Para ver una copia de esta licencia, visita: https://creativecommons.org/licenses/by-nc-sa/4.0/

En señal de conformidad, suscribo el presente documento.

Puno 15 de mayo del 2023

FIRMA (obligatoria)

Huella

FIRMA (obligatoria)

Huella