ROBOT SEGUIDOR DE LINEA

Román H. Jhoan Cód.1094949367, Velásquez G. Juan Cód. 1098310202, Morcillo O. Esneyder Cod. 109495146

***Resumen*—** **En el presente informe se dará a conocer los aspectos más relevantes del diseño de un robot seguidor de línea, haciendo uso de códigos programados en lenguaje c en las tarjetas Arduino, de igual forma hallar el objetivo primordial de funcionamiento por medio de los componentes electrónicos a implementar.**

*Índice de términos*— Tarjeta Arduino, Módulos, Ensamble, .

# **Introducción**

Hoy en día en las empresas de mayor reconocimiento mundial, podemos observar la actividad y trabajo de robots; maquinas hechas por el hombre que se encargan de facilitar o incluso realizar completamente las tareas humanas de mayor riesgo, peligro y esfuerzo. Tareas que para el hombre resultan difíciles y tediosas de emprender, lo cual económicamente genera gastos, las empresas u organismos de cualquier índole lo que buscan es reducir gastos, pero no disminuir la calidad de sus trabajos. En contexto cada vez que surge un problema no tan grave, ya que cuentan con recursos modernos que nos permiten acceder a mecanismos fáciles de manejar construir y adquirir.

La robótica y la electrónica son áreas importantes en la historia del planeta ya que en todos los campos nos permite crear una cantidad ilimitada de aplicaciones prácticas, las podemos utilizar en nuestra vida cotidiana dándose por hecho. En el caso de los seguidores de línea nos

Permite aproximarnos a la programación y a la aplicación de la teoría estudiada el cual se presentan alternativas que puedan satisfacer las necesidades planteadas a nuestro proyecto.

.

Objetivos

Objetivo General

Implementar un robot seguidor de línea, cuyas características cumplan requerimientos con respecto a modos de programación, iniciativa, ensamble y funcionamiento concerniente a esta tecnología muy común en la ingeniería.

Objetivos Específicos

Obtener un seguidor de línea que siga un camino marcado con líneas negras utilizando códigos de programación en tarjeta Arduino.

Adaptar los sensores necesarios para detectar las líneas negras con respecto al fondo blanco y activar el funcionamiento correspondiente.

Marco teórico

Todos los rastreadores basan su funcionamiento en los sensores. Sin embargo, dependiendo de la complejidad del recorrido, el robot debe ser más o menos complejo (y, por ende, utilizar más o menos sensores).

Los rastreadores más simples utilizan 2 sensores, ubicados en la parte inferior de la estructura, uno junto al otro. Cuando uno de los dos sensores detecta el color blanco, significa que el robot está saliendo de la línea negra por ese lado. En ese momento, el robot gira hacia el lado contrario hasta que vuelve a estar sobre la línea. Esto en el caso de los seguidores de línea negra, ya que también hay seguidores de línea blanca.

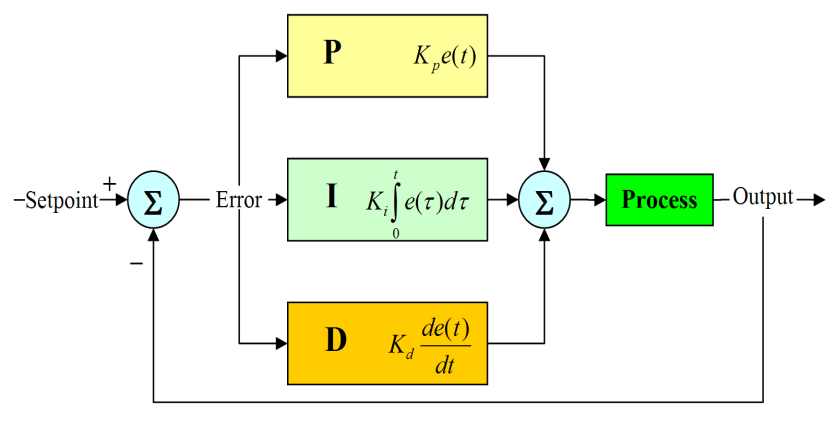
Estos robots tienen la capacidad de seguir una línea marcada en el suelo (normalmente una línea negra sobre un fondo blanco).

III.1 Estructura b**á**sica

Estos robots pueden variar desde los más básicos (van tras una línea única) hasta los robots que recorren laberintos. Todos ellos, sin embargo, poseen (por lo general) ciertas partes básicas comunes entre todos sensores bonitos y baratos, como debe ser. Un rastreador detecta la línea a seguir por medio de sensores. Hay muchos tipos de sensores que se pueden usar para este fin; sin embargo, por razones de costos y practicidad los más comunes son los sensores infrarrojos (IR), que normalmente constan de un LED infrarrojo y un fototransistor.

PID

El PID (control proporcional, integral y derivativo) es un mecanismo de control por realimentación que calcula la desviación o error entre un valor medido y el valor que se quiere obtener (set point, target position o punto de consigna), para aplicar una acción correctora que ajuste el proceso.



*Fig. 1 Diagrama que representa el PID*

Para nuestro seguidor de línea el PID (que es una rutina basada matemáticamente), procesa los datos de los sensores infrarrojos y los usa para controlar la velocidad y posición del robot.

**Pasos para calibrar el PID**

Para poder calibrar el PID se siguen los siguientes pasos:

Paso 1: Acción Proporcional

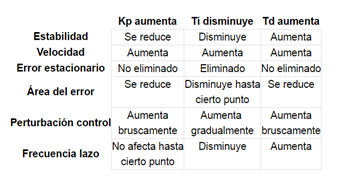
* Tiempo integral (TI), a su máximo valor.
* Tiempo derivativo (TD), a su mínimo valor.
* Empezando con ganancia baja se va aumentando hasta obtener las características de respuesta deseadas.

Paso 2: Acción integral

* Reducir el TI hasta anular el error en estado estacionario, aunque la oscilación sea excesiva.
* Disminuir ligeramente la ganancia.
* Repetir hasta obtener las características de respuesta deseadas.

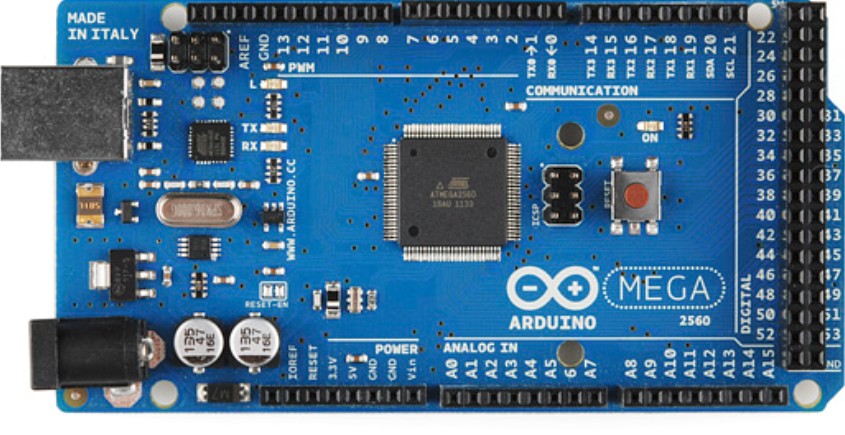
Paso 3: Acción Derivativa

* Mantener ganancia y tiempo integral obtenidos anteriormente.
* Aumentar el TD hasta obtener características similares, pero con la respuesta más rápida.
* Aumentar ligeramente la ganancia si fuera necesario.

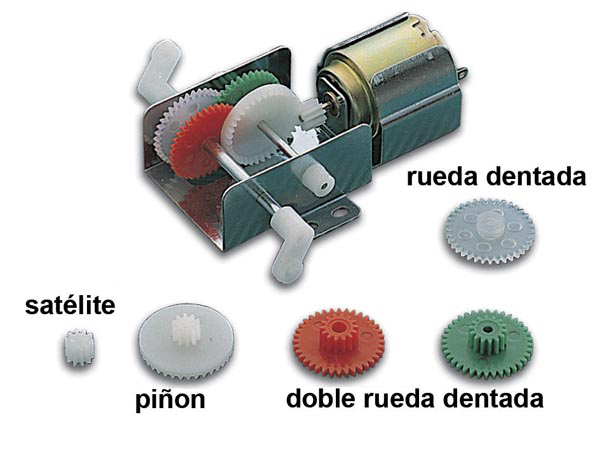
*Fig. 2 Esquema que representa el PID*

# **materiales**

- Arduino Mega (Atmega 2560)

  
*Fig. 3 Arduino MEGA*

**Tarjeta de control Arduino (MEGA):** La toma de decisiones y el control de los motores están generalmente a cargo de un micro controlador. La tarjeta de control contiene dicho elemento, junto a otros componentes electrónicos básicos que requiere el micro controlador para funcionar, para ello se hace una programación a un código en c para lograr que el micro funcione a lo requerido.



*Fig. 4 Motor acoplado con piñones*

**Motores:** El robot se mueve utilizando motores. Dependiendo del tamaño, el peso, la precisión del motor, entre otros factores, éstos pueden ser de varias clases: motores de corriente continua, motor paso a paso o servomotores.

  
*Fig. 5 Ruedas acopladas a los motores*

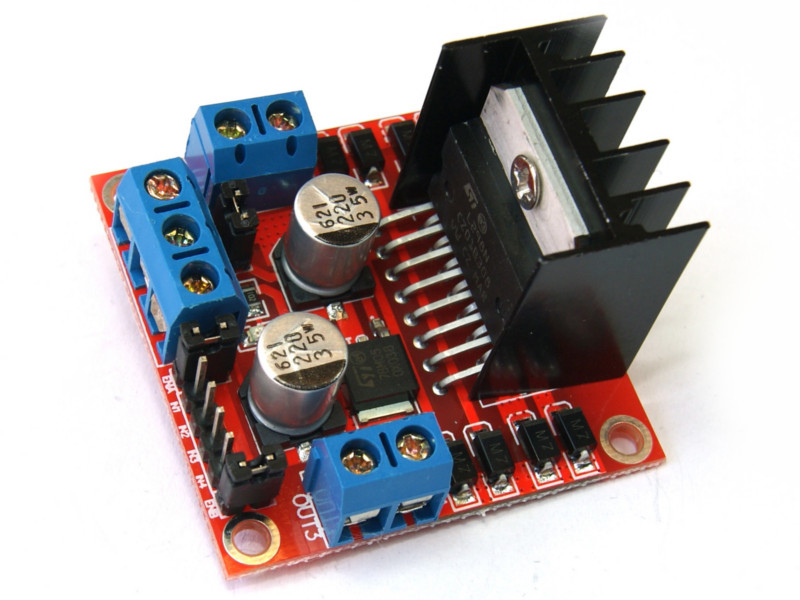
**Ruedas:** Las ruedas del robot son movidas por los motores. Normalmente se usan ruedas de materiales

anti-deslizantes para evitar fallas de tracción. Su tamaño es otro factor a tener en cuenta a la hora de armar el robot, sin embargo, las ruedas que fueron utilizadas tenían más fricción que como debía usarse.



*Fig. 6 Batería usada como fuente de energía*

**Fuente de energía:** El robot obtiene la energía que necesita para su funcionamiento de baterías o de una fuente de corriente alterna, siendo esta última menos utilizada debido a que le resta independencia al robot. En este caso se hizo uso de las baterías mostradas en la presente figura correspondiente a las baterías de un celular Nokia 1100, ya que generaban el voltaje correspondiente y necesitado para dar buen funcionamiento al módulo puente H previamente a los motores y a la tarjeta Arduino.



*Fig. 7 Modulo Puente H L298N*

**Modulo puente H** **L298N** es una tarjeta para el control de motores de corriente directa, motores a pasos, solenoides y en general cualquier otra carga inductiva. La tarjeta está construida en torno al circuito integrado **L298N**, el cual dispone en su interior de 2 puentes H independientes con capacidad de conducir 2 amperios constantes o 4 amperios en picos no repetitivos. La tarjeta expone las conexiones hacia el motor a través de bloques de terminales (clemas), mientras que las entradas de control y habilitación del puente H se exponen a través de headers macho estándar para facilitar todas las conexiones.

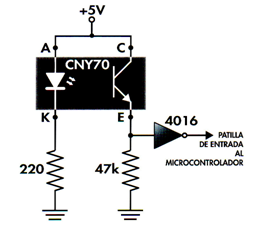
Esta tarjeta es ideal para controlar motores en pequeños robots como seguidores de líneas, zumos, robots de laberinto, etc. El L298N también es una excelente opción para manejar motores a pasos bipolares.

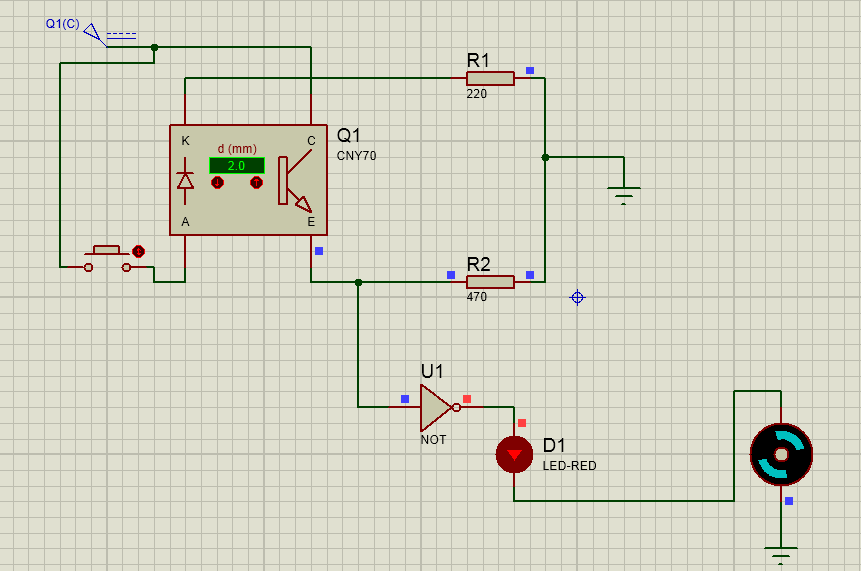


*Fig. 8 Sensores CNY70*

**Sensores CNY70** El fototransistor conducirá más, contra más luz reflejada del emisor capte por su base. La salida de este dispositivo es analógica y viene determinada por la cantidad de luz reflejada, así pues, para tener una salida digital se podría poner un disparador Trigger Schmitt y así obtener la salida digital, pero esto tiene un problema, y es que no es ajustable la sensibilidad del dispositivo y los puntos de activación de histéresis distan algunos mili voltios uno del otro.

Comúnmente utilizado en los robots rastreadores (Sniffers) para detección de líneas pintadas sobre el suelo, debido principalmente a su baja distancia de detección

  
*Fig. 9 Diagrama del sensor en módulo*



*Fig. 9 Simulación del módulo del opto acoplador*

El sensor detecta cuando no hay luz incidiendo sobre él y esto es posible gracias al color oscuro que proporciona la cinta negra, si la luz es detectada el sensor envía una señal que es traducida como una desviación del vehículo y aunque acá es directa la forma en que se ve es mediante algoritmos y un controlador de motor lo que permitirá manipular la señal para un eficiente sistema.

# **código, ensamble y funcionamiento**

Para el desarrollo de este proyecto se analizaron muchos factores como los motores a usar, como se iba a quemar la baquela que contenía los sensores seguidores de línea, los trimer para dar las constantes del PID.

Teniendo en cuenta los parámetros anteriores se diseño un código, tal que este pueda realizar los procedimientos necesarios para el correcto funcionamiento del proyecto.

A continuación se presenta el código usado y unas breves especificaciones de lo que realizan algunas líneas especificas:

*int motor\_derecha = 2; // Con este pin, dirigimos el motor derecho*

*int motor\_izquierda = 3; // Con este pin, dirigimos el motor izquierdo*

*int pwm\_iz = 10; // pin de manejo de velocidad del motor izquierdo EL CUAL ES UN POCO MAS RAPIDO QUE EL DERECHO ya que nuentros motores no estaba calibrados*

*int pwm\_der = 11; // Pin de manejo de la velocidad del motor derecho*

*// Pines analogos*

*int sensor1 = 0;*

*int sensor2 = 1;*

*int sensor3 = 2;*

*// Variables iniciadas en cero*

*int posicion = 0;*

*int ultima\_posicion = 0;*

*float valor\_pwm = 0;*

*int velder=0,veliz=0;*

*int maxvelder=230;*

*int maxveliz=230;*

*int basevelder=130;*

*int baseveliz =130;*

*int lastError=0;*

*// Constantes del PID*

*float Kp=6,Kd=1,Ki=0.1; //Constantes del PID*

*void setup() {*

*delay(2000); // Hacemos una espera de 2 segundos para iniciar*

*Serial.begin(9600); // iniciamos comunicacion serial con 9600 bouds*

*pinMode(motor\_derecha,OUTPUT);*

*pinMode(motor\_izquierda,OUTPUT);*

*pinMode(pwm\_iz,OUTPUT);*

*pinMode(pwm\_der,OUTPUT);*

*}*

*void loop() {*

*// Lectura de los sensores en las entradas analogas*

*int s1 = analogRead(sensor1);*

*int s2 = analogRead(sensor2);*

*int s3 = analogRead(sensor3);*

*int s11 = 0;*

*int s12 = 0;*

*int s13 = 0;*

*// Calibracion de sensores para convertirlos a binario*

*if(s1>145){ //145 es el valor en que el sensor1 déjà de leer la lina negra*

*s11=1;*

*}*

*if(s2>55){ //55 es el valor en que el sensor2 déjà de leer la lina negra*

*s12=1;*

*}*

*if(s3>85){ //85 es el valor en que el sensor3 déja de leer la lina negra*

*s13=1;*

*}*

*ultima\_posicion = posicion; // Almacenamos las posiciones leidas*

*// posicion = (1\*s13)+(2\*s12)+(4\*s11); // Convertimos los numeros binarios obtenidos de los sensores en decimales*

*// Serial.println(posicion);*

*// Calculamos los valores proporcional, integral y derivativo*

*int error = posicion-5;*

*Serial.println(error);*

*int u = Kp \* error + Kd \* (error - lastError) + Ki \* (error + lastError) ;*

*lastError=error;*

*//En las siguientes condiciones se calibra la velocidad para mandarlos al arduino*

*if (error>=(-1))*

*{*

*velder = basevelder - u;*

*veliz = baseveliz + u;*

*if(error==0)*

*{*

*velder = maxvelder;*

*veliz = maxveliz;*

*}*

*}*

*if (error<(-1))*

*{*

*velder = basevelder + u;*

*veliz = baseveliz - u;*

*}*

*if (velder > maxvelder ) velder = maxvelder;*

*if (veliz > maxveliz ) veliz = maxveliz;*

*if (velder < 0) velder = 10;*

*if (veliz < 0) velder = 10;*

*//A continuacion se Mandan los valores de velocidad al arduino*

*if(posicion == 5){ // Avanza normalmente*

*digitalWrite(motor\_derecha\_at,HIGH);*

*analogWrite(pwm\_der,150);*

*digitalWrite(motor\_izquierda\_at,HIGH);*

*analogWrite(pwm\_iz,250);*

*}*

*if(posicion == 0){ // Nunca debe ocurrir (STOP)*

*digitalWrite(motor\_derecha\_at,HIGH);*

*analogWrite(pwm\_der,40);*

*digitalWrite(motor\_izquierda\_at,HIGH);*

*analogWrite(pwm\_iz,40);*

*}*

*if(posicion == 3){*

*digitalWrite(motor\_derecha\_at,HIGH);*

*analogWrite(pwm\_der,120);*

*digitalWrite(motor\_izquierda\_at,HIGH);*

*analogWrite(pwm\_iz,50);*

*}*

*if(posicion == 1){ //Se supone nunca pasará*

*digitalWrite(motor\_derecha\_at,HIGH);*

*analogWrite(pwm\_der,120);*

*digitalWrite(motor\_izquierda\_at,HIGH);*

*analogWrite(pwm\_iz,100);*

*}*

*if(posicion == 6){ //Se supone nunca pasará*

*digitalWrite(motor\_derecha\_at,HIGH);*

*analogWrite(pwm\_der,50);*

*digitalWrite(motor\_izquierda\_at,HIGH);*

*analogWrite(pwm\_iz,120);*

*}*

*if(posicion == 4){ //Se supone nunca pasará*

*digitalWrite(motor\_derecha\_at,HIGH);*

*analogWrite(pwm\_der,100);*

*digitalWrite(motor\_izquierda\_at,HIGH);*

*analogWrite(pwm\_iz,120);*

*}*

*if(posicion == 7){*

*if((ultima\_posicion==6)||(ultima\_posicion==4)){*

*digitalWrite(motor\_derecha\_at,HIGH);*

*analogWrite(pwm\_der,40);*

*digitalWrite(motor\_izquierda\_at,HIGH);*

*analogWrite(pwm\_iz,140);*

*}*

*if((ultima\_posicion==3)||(ultima\_posicion==1)){*

*digitalWrite(motor\_derecha\_at,HIGH);*

*analogWrite(pwm\_der,140);*

*digitalWrite(motor\_izquierda\_at,HIGH);*

*analogWrite(pwm\_iz,40);*

*}*

*}*

*}*

En cuanto al ensamble este fue muy simple ya que los 3 sensores infrarrojos se conectaban a los pines del arduino que están declarados en el anterior código y todos estos tenían un Vcc común y un Gnd común que se conectan a la el pin por el cual el puente H suministra los 5V.

Para la conexión del puente H, este se alimentaba con 8V a el llegaban 2 señales pwm procedentes del Arduino y del puente H salen las conexiones para los dos motores.

Para el funcionamiento lo que pasa es que se tiene un numero todo el tiempo que representa la posición, para este caso en específico la posición ideal es 5, todas las demás posiciones van a contener un error y este error es usado en la ecuación del pid para concebir un valor, el cual sirve para manipular las velocidades que se imprimen en los motores

# **CONCLUSIONES**

* El seguidor de línea es una herramienta que estimula la creatividad y desarrollo de los conocimientos adquiridos por el ingeniero durante su proceso de formación académica basándose en programación, análisis e invención.
* El diseño adoptado para el montaje del robot fue el más cómodo a nuestro alcance ya que se facilitó el funcionamiento por medio del código y el ensamble físico de manera correcta, teniendo en cuenta que en el plano físico los motores desempeñaban un gran papel
* Todos los sensores basan su funcionamiento en la detección de luz sin embargo, dependiendo de la complejidad del recorrido, el robot debe ser

Más o menos complejo (y, por ende, utilizar más o menos sensores).

* Al utilizar 3 sensores, ubicados en la parte inferior de la estructura, uno junto al otro. Cuando uno de los 3 sensores detecta el color blanco, significa que el robot está saliendo de la línea negra por ese lado, en ese momento, el robot gira hacia el lado contrario hasta que vuelve a estar sobre la línea.

# **BIBLIOGRAFIA**

[1]https://es.wikipedia.org/wiki/Robot\_seguidor\_de\_l%C3%ADnea

[2]http://www.buenastareas.com/materias/informe-seguidor-de-linea/0

[3]https://es.scribd.com/doc/78191609/SEGUIDOR-DE-LINEA-INFORME

[4]http://seguidordelineaups.blogspot.com.co/