Proyecto 1

***Objetivo general:***

Implementar un vehículo seguidor de línea usando la plataforma de arduino y empleando un controlador PID que permita un movimiento acorde a las señales csensadas y ejecutadas mediante la tarjeta de desarrollo ATMEGA 2560.

***Objetivos específicos:***

* Construir o implementar la estructura física del vehículo seguidor de línea de acuerdo a parámetros de diseño tenidos en cuenta.
* Acondicionar la tarjeta de desarrollo al vehículo de forma que pueda ejercer la ejecución o intercambio de señales con el mismo, y los sensores que este tiene adaptados.
* Diseñar el código base PID que permita realizar la ejecución del movimiento de forma estable y acorde al comportamiento esperado.

-seguidor de línea, con arduino y el control PID.

Para este proyecto se pretende desarrollar e implementar un seguidor de línea, el cual será controlado por la placa de arduino, en nuestro caso, Mega2560, además de implementarle un control PID, el cual nos dará una precisión mayor en el seguidor. Por lo tanto, partimos de la elaboración del seguidor. Donde utilizamos los siguientes materiales

* 2 ruedas de arrastre
* 1 rueda loca
* 1 circuito puente H
* 8 Sensores de referencia Array QTR-8RC Reflectance
* 2 moto reductores
* 1 chasis auto inteligente
* Tarjeta de desarrollo Arduino Mega 2560



Fig.1. ruedas de arrastre

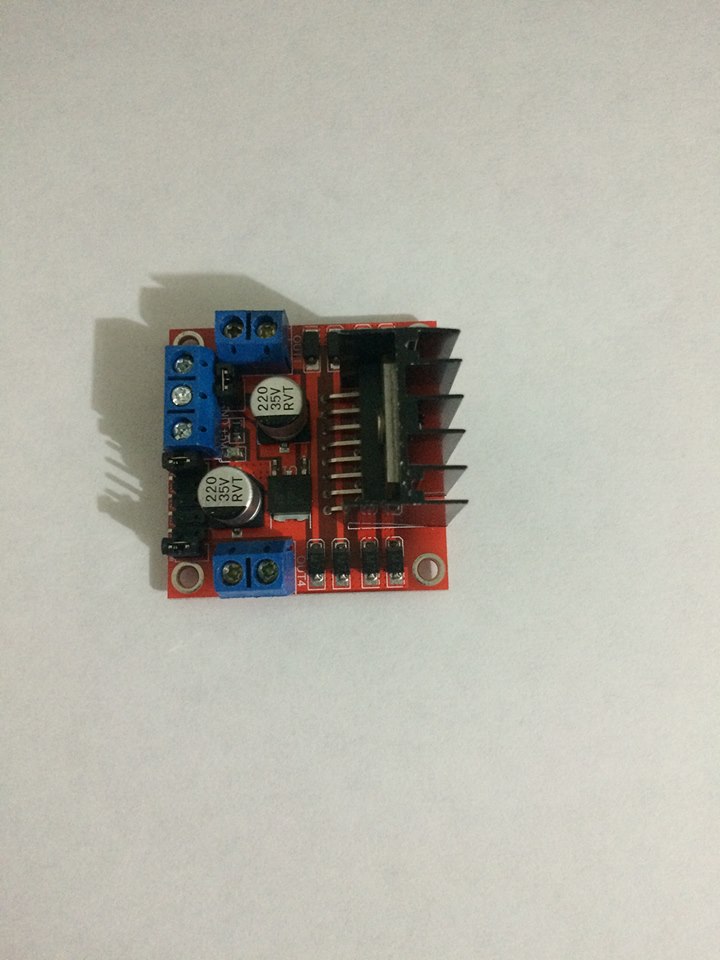


Fig.2.circuito puente H.

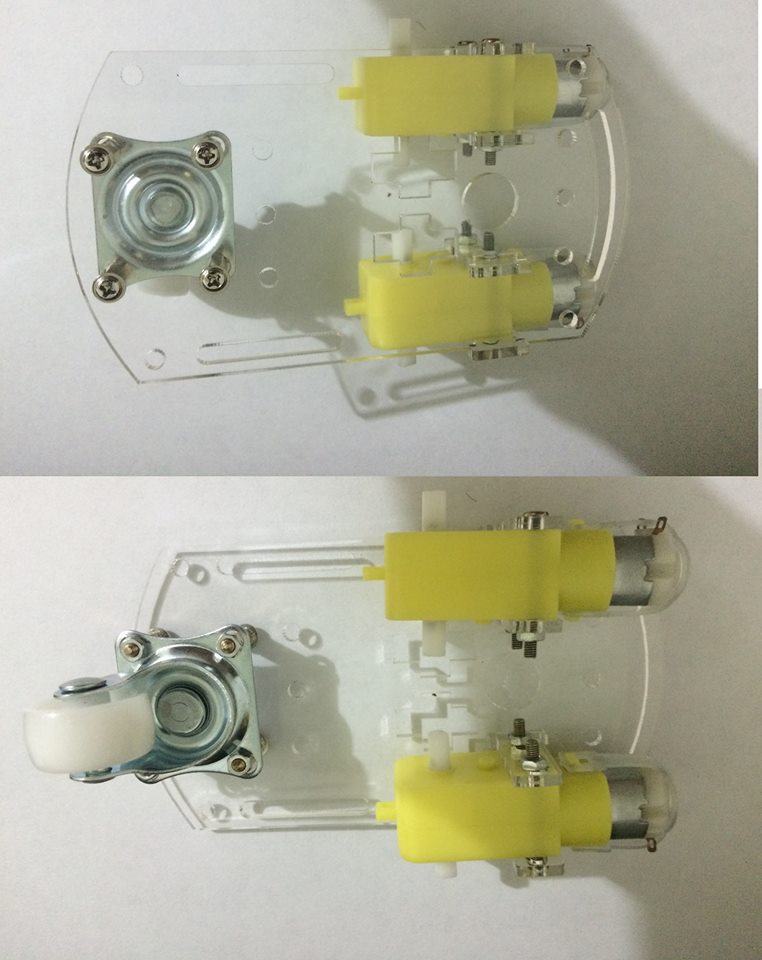


Fig.3. rueda loca, par de motores y base del chasis

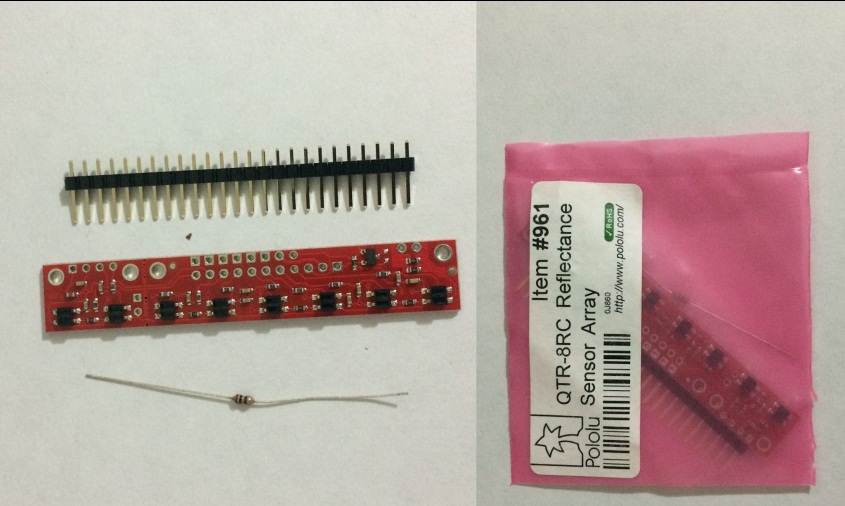


Fig.4. sensores para la detección de línea.



Fig.5. vehículo y diferentes materiales.

***Descripción del PID a emplear***

En primer lugar definamos que es un controlador PID.

Entonces un PID es un controlador de uso genérico, el cual emplea realimentación de lazo cerrado, es un sistema al que le ingresa un error correspondiente a la diferencia entra la salida deseada (set point) y el valor de salida obtenido en la salida final del sistema. El controlador intentara en todo momento minimizar el error ajustando la entrada al sistema

El PID es un sistema al que le entra un **error** calculado a partir de la salida deseada menos la salida obtenida y su salida es utilizada como entrada en el sistema que queremos controlar. El controlador intenta minimizar el error ajustando la entrada del sistema.

Este controlador tiene implícito 3 parámetros y cada uno influye en mayor o menor medida sobre alguna característica de la salida ya sea el tiempo de establecimiento, sobreimpulso, entre otros,

En este orden de ideas entonces los parámetros de los que se hablaba son

* Acción proporcional

Siendo la base de los tres parámetros, da una entrada de control proporcional con el error, usar solo este parámetro genera un error estacionario.

* Acción integral

Este da una suma de los errores previos a la entrada del sistema de control continuando hasta que el valor llegue al deseado, la mayoría de las veces el termino integral está asociado a un término proporcional, con lo cual se elimina el error estacionario.

* Acción derivativa

Este da una respuesta que es proporcional a la velocidad del cambio del error, con esta acción, y adicional a las demás se mejora la velocidad del sistema y contribuye a la disminución de oscilaciones excesivas.

**El código propuesto se deja para la próxima documentación puesto que aún se está estructurando mediante la ayuda de la clase teórica el manejo de los diferentes módulos del microcontrolador**

C:\Users\Alejandro Marmolejo\Desktop\descarga.png

Fig.6. Representación en bloques del sistema a emplear

Para realizar el montaje del seguidor de línea , primero necesitamos hacer una serie de pruebas las cuales fueron implementadas en el software de simulación de circuitos ISIS de PROTEUS , en ayuda de una librería de arduino , con lo cual se realizó la respectiva prueba de los motores y el puente h, de manera que se simulara el comportamiento de los sensores , y así saber si nuestros motores reaccionaban de manera correcta además de implementar un código en arduino, el cual está en prueba pues el total será mostrado al final del proyecto, para poder simular fue necesario el archivo en hexadecimal que contenía nuestro código base y con este así poder cargar al arduino simulado en ISIS , de manera que para nuestro código además se estableció de una vez el control pid.

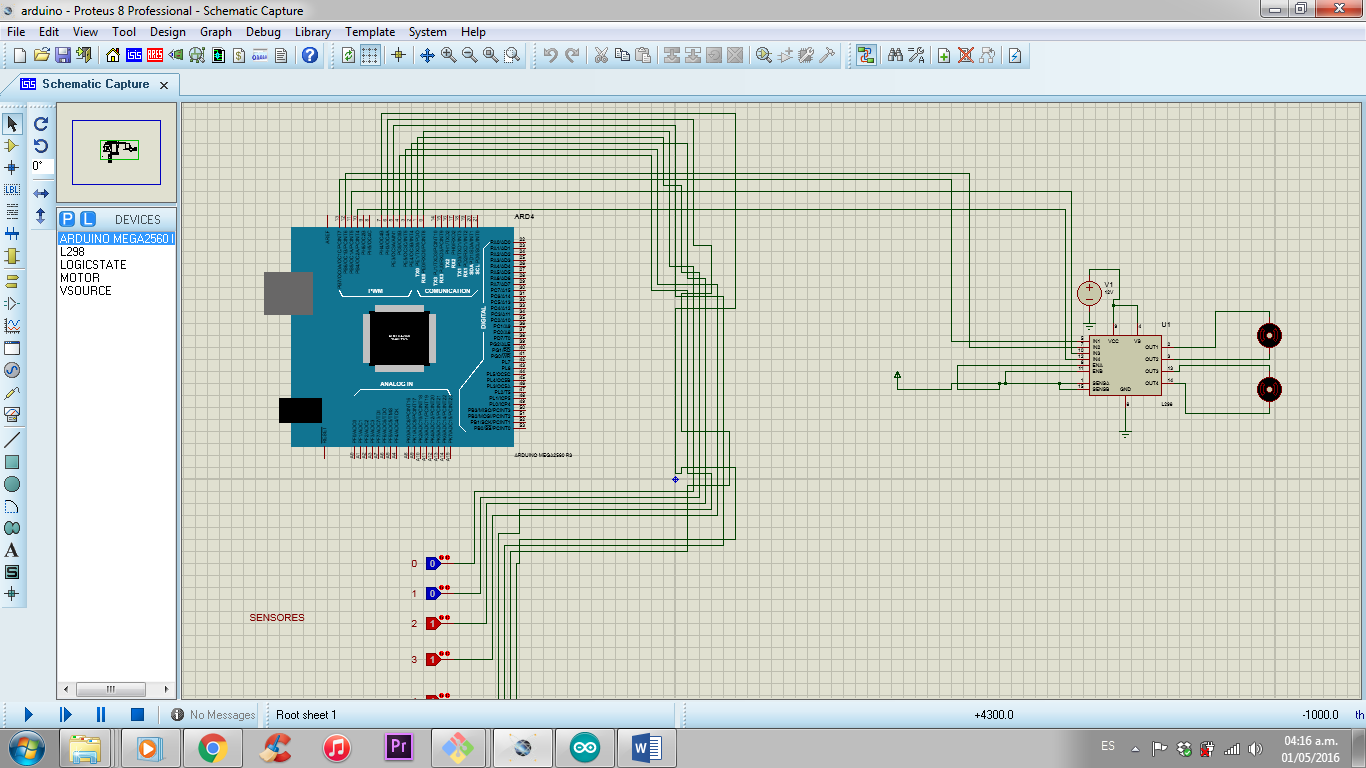


Fig.7. Implementación en proteus