Bitácora de Avances Proyecto 2

Descripción de los Componentes a Utilizar y Funcionamiento del PID

Asignatura:

Microprocesadores

Presentado por:

Nombre: Cód.:

Darmael Alfredo Vargas Figueredo…………………………………1.118.560.349

Duvier de Jesús Bohórquez Palacio…………………………………1.094.948.815

Braulio Loaiza Castaño……………………………………………………1.096.646.602

Profesor:

Gerardo Andrés López Orozco

Programa de Ingeniería Electrónica

Facultad de Ingeniería

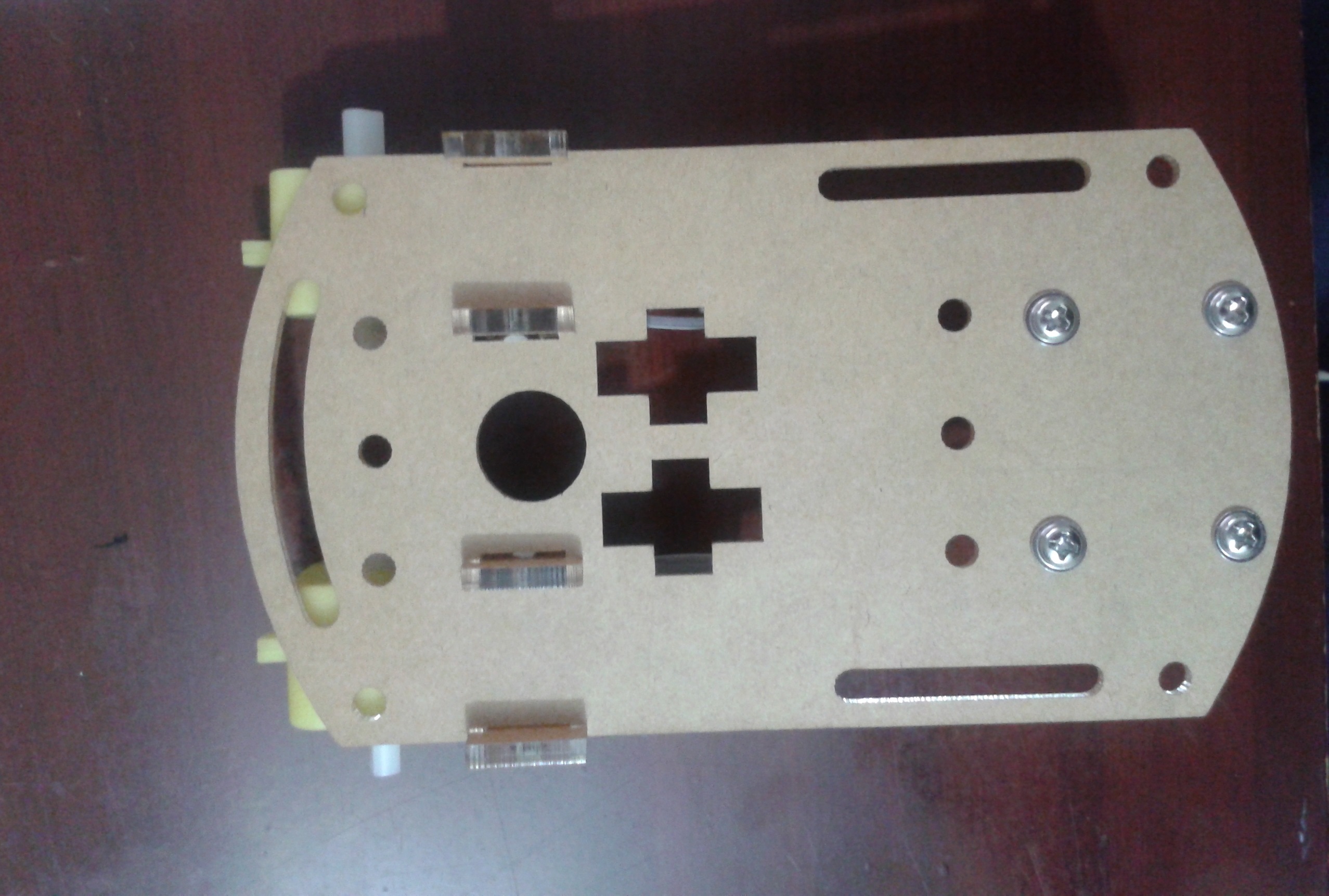
Universidad del Quindío

Armenia, Quindío

Marzo de 2016

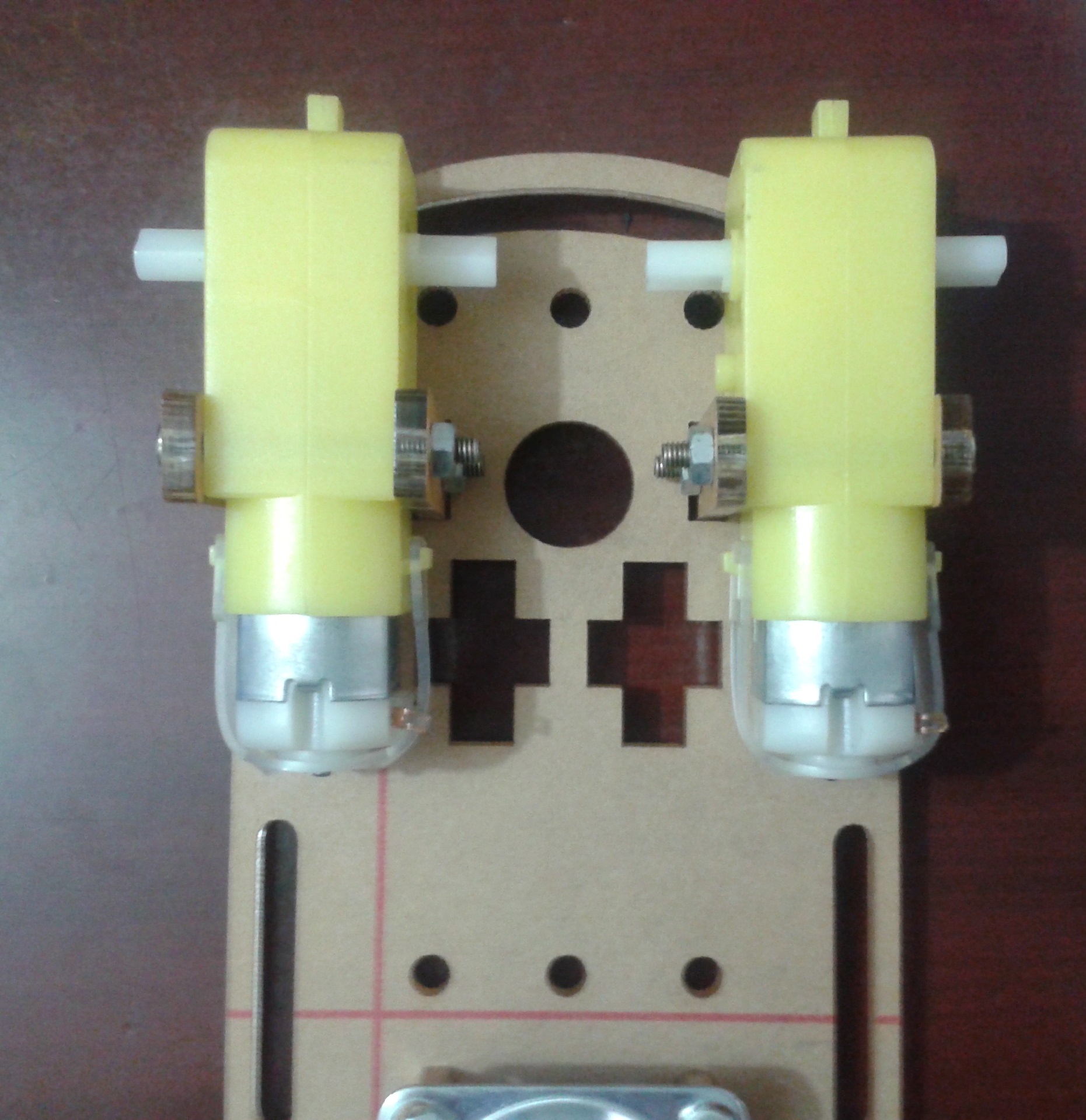
Descripción de los Componentes a Utilizar y Funcionamiento del PID

En el presente documento se mostraran y/o describirán, con la debida claridad del caso, la utilidad y el funcionamiento de cada componente requerido, para llevar a cabo la correcta implementación del carrito seguidor de línea, basado en el uso de algoritmos que definen el control PID, los cuales serán compilados por medio del microcontrolador Arduino MEGA 2560, como se ira mostrando a lo largo de esta bitácora.



**Fig. 1** Placa base donde se adecuaran los elementos que componen al carrito

En primer lugar, en cuanto a los componentes a utilizar, se hará uso de la placa base mostrada, en la figura 1, la cual consiste en un chasis en acrílico, el cual se encuentra cubierto por una capa protectora superior e inferior con un adhesivo en color cartón que puede retirarse si se desea. Además este posee diversas perforaciones, las cuales son útiles para realizar una instalación de componentes, de manera más organizada, en términos del cableado que permite la interconexión de cada dispositivo a instalar en el mismo.



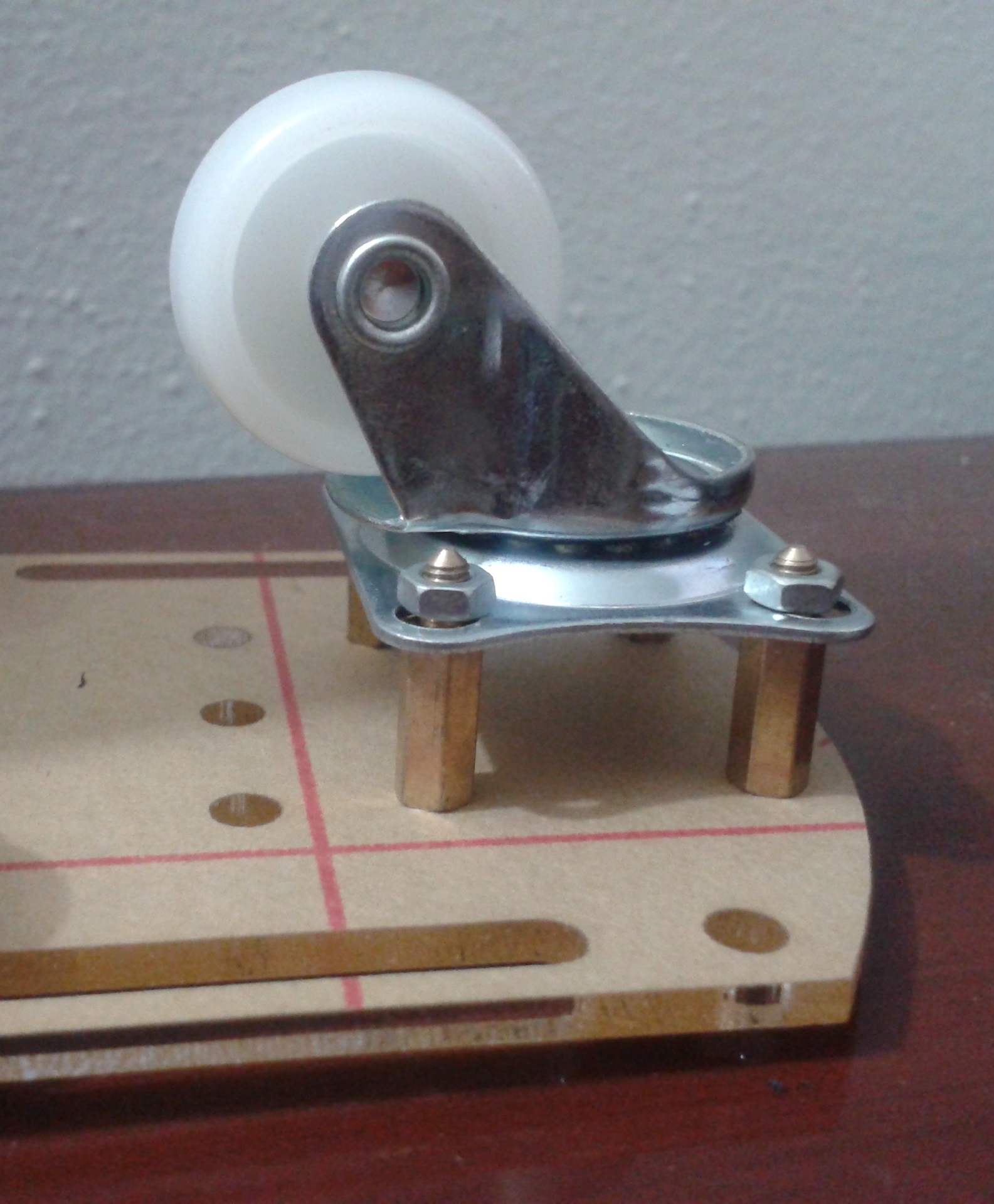
**Fig. 2** Motorreductores con caja reductora

A dicha placa se le añaden dos motorreductores, como se muestra en la figura 2, los cuales cumplen con la función de ejecutar el proceso de movimiento que mostrara el carrito, cuando este se encuentre en funcionamiento. Es importante destacar el hecho de que estos elementos posen un voltaje de alimentación situado en un rango de 3 a 6 voltios, además de contar con una relación de caja de cambios de 1:48, lo que quiere decir que cada uno de los motores poseen una rotación de 48 vueltas, por lo que el tacómetro permite capturar 192 pulsos (48x4) y 8 rpm del motor, así pues, el módulo de velocidad permite capturar 15.306000 pulsos/rpm.



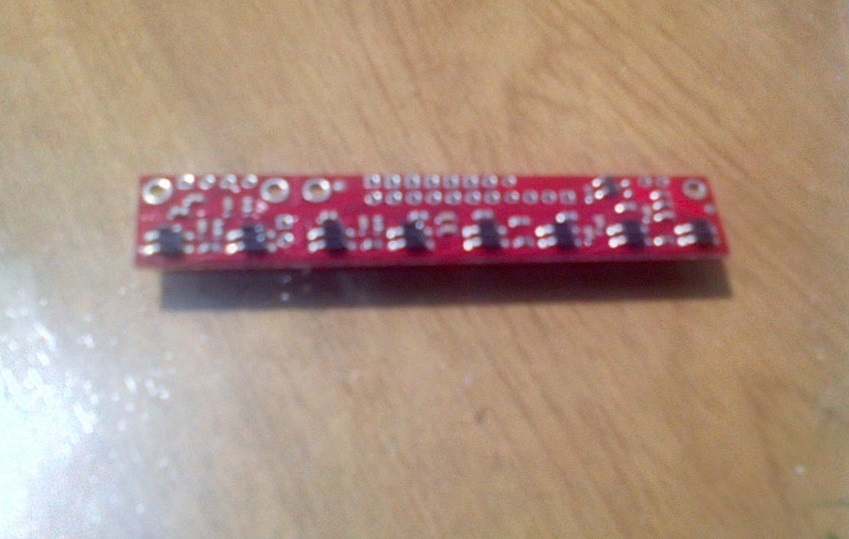
**Fig. 3** Ruedas

La figura 3 muestra el conjunto de ruedas que serán ubicadas en cada uno de los motores descritos anteriormente. En cuanto a sus dimensiones, estas poseen 6 cm de diámetro y 2.5 cm de ancho.



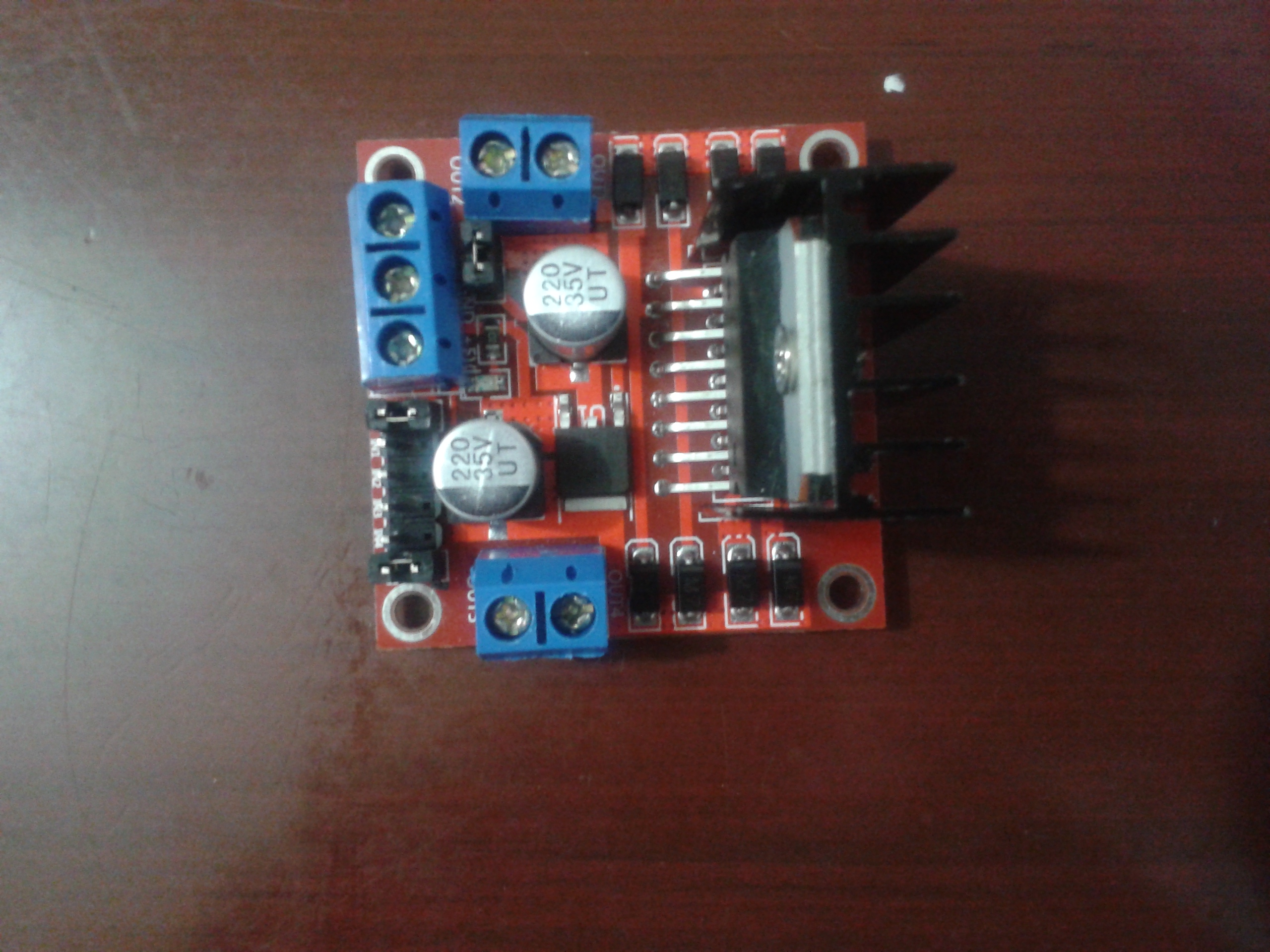
**Fig. 4** Rueda delantera

La rueda que se observa, en la figura 4, se debe utilizar para que el carrito conserve un equilibrio permanente, al tener la capacidad de orientarse en cualquier dirección. En cuanto a las dimensiones se debe destacar que esta posee 2.2 cm de diámetro y 1.2 cm de ancho. Por otra parte, observando esta imagen, se puede apreciar que esta se encuentra levantada por medio de un conjunto de tornillos y esto se hace con el fin de lograr un balance y/o nivelación del chasis, para que no ocurran problemas de estabilidad en su funcionamiento.



**Fig. 5** Sensor Qtr-8rc

En este proyecto es necesario hacer uso de un conjunto de sensores que permitirán la detección de la línea negra que seguirá el carrito. La figura 5 muestra dicho elemento, el cual es una placa que contiene un arreglo de 8 sensores montados a 9.5 mm uno del otro, por lo que es un gran detector para seguidores de línea. En este caso cada sensor proporciona una salida digital, la cual puede ser medida de forma separada. Las dimensiones de este objeto son de 2.5’’x 0.5’’x 0.125’’ sin pines y funciona a partir de un rango de voltajes situado entre 3.3 y 5 voltios. Además de esto, dicho dispositivo se debe alimentar con una corriente de 100 mA y una distancia optima de detección debe ser de 3 mm, aunque la máxima distancia recomendada de detección es de 9.5 mm.



**Fig. 6** Driver puente H L298n doble motor

La figura 6 muestra un driver puente H, el cual será utilizado en la implementación del seguidor de línea, puesto que este permite dar funcionamiento a los motores, a partir de la información que suministre el controlador arduino. Este elemento es, básicamente, un controlador de motores de alto voltaje, alta corriente y es diseñado para aceptar estándares de niveles lógicos TTL. Este dispositivo se encuentra en la capacidad de manejar cargas inductivas tales como relés, solenoides, motores DC y de paso. En cuanto a sus especificaciones técnicas, este dispositivo cuenta con un voltaje de control de 5 voltios, una corriente de control situada entre 0 y 36 mA, soporta temperaturas desde -20 hasta 135 grados centígrados, presenta un voltaje de salida desde 5 voltios hasta 35 voltios, soporta una corriente máxima de 2 A, una potencia máxima de 25 Watts, sus dimensiones externas son de 43x 43x 27 mm y pesa 30 gramos.



**Fig. 7** Microcontrolador Arduino Mega 2560

Por último se hará uso del microcontrolador mostrado, en la figura 7, el cual será útil para compilar los algoritmos a implementar, con respecto al control PID, y gestionara todas las operaciones que se irán a ejecutar sobre el carrito seguidor de línea.

Por otra parte, para esta semana, se implementó el control PID en el IDE de Arduino y se desarrolló un algoritmo base para el funcionamiento del robot seguidor de línea, se implementará el montaje y se realizarán las respectivas pruebas para así corregir posibles errores y también llegar a posibles mejoras.

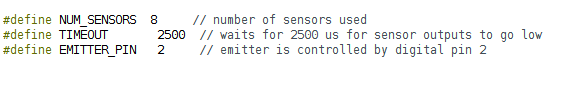
Para el código se usó la librería QTRSensors creada por Pololu , empresa que desarrolla los sensores que estamos usando (QTR-8RC) , básicamente es un conjunto de sensores infrarrojos los cuales responden de diferente manera , ya sea que están sobre una línea oscura o una línea blanca, se crea una variable entero llamada posición, en la cual se deberá profundizar más para saber con exactitud el error en el lazo típico de control y así implementar debidamente el control PID.

El programa se divide en varias partes, para ser más exactos, se divide en setup y loop, en el setup se puede definir la naturaleza de los pines y en este caso particular, se calibra el conjunto de sensores con respecto a la distancia en la que están ubicados frente a la superficie, para ello se usa la función qtrrc.calibrate(); dentro de un ciclo for, esto antes de que empiece a ejecutar el loop del programa, se mantiene un delay de 2 segundos y los motores apagados gracias a la función wait.

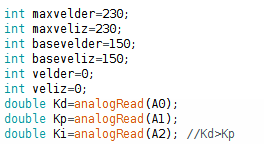


Luego se entra al loop de el programa pero no sin antes definir nuestra función que reconocerá los valores mapeados por los sensores



Con

También antes de iniciar el setup se definen variables como: la velocidad base a la que va a estar el robot sobre linea continua, el tope máximo de velocidad (buscando estabilidad en el sistema), su velocidad inicial de 0 , y finalmente como se dijo en clase los valores de las constantes Kp ,Kd y Ki son tomados a partir de la lectura de los valores dados por potenciómetros conectados a el ADC del Arduino.



Ahora se entra al loop, en donde se encuentra nuestro control PID , primero se adquiere la posición de los sensores dando esto como resultado un valor (falta determinar), éste valor se puede manipular y así se podrá obtener el error que posteriormente la función u tratará de eliminar, u estará compuesto por una sumatoria de : la constante proporcional Kp\*e, la constante diferencial Kd multiplicada por la diferencia entre el error anterior y el error actual y finalmente se adiciona Ki multiplicada por la integral discreta del error actual y el error pasado, esto en conjunto da origen a nuestro controlador PID , finalmente el error actual se convierte en el error anterior para la siguiente iteración, luego se aplica la lógica pura para darle dirección a el robot en los diferentes casos donde se encuentra con curvas, debido a que no se conoce el rango de valores se define condiciones temporales que serán modificadas.

La idea es que cuando el robot encuentre una curva, éste de acuerdo a la orientación de la curva adicione o reste a la velocidad base de los motores la señal u que se menciona anteriormente, superando así este obstáculo.



parte final del código es destinada más que todo a la estabilidad del sistema, en donde se busca que nuestro robot no se vuelva “loco” en determinados casos , como por ejemplo que la velocidad a la que un motor gira en alguna dirección supere la velocidad máxima definida y también en los casos donde el valor de velocidad en algún motor alcance valores negativos, para este caso el motor se detiene, por último se escriben los valores finales en los pines de salida.

