Diseño de un Carrito Seguidor de Línea de Código y Hardware Abierto

André Renato Leal Marroquín
Santa Tecla
ITCA - FEPADE
San Salvador, El Salvador
andre.leal17@itca.edu.sv

Edwin Salvador Leiva Alvarez
Santa Tecla
ITCA - FEPADE
San Salvador, El Salvador
edwin.leiva17@itca.edu.sv

Daniel Ernesto Molina
Santa Tecla
ITCA - FEPADE
San Salvador, El Salvador
daniel.molina17@itca.edu.sv

Javier Edgardo Trigueros Alvarez
Santa Tecla
ITCA - FEPADE
San Salvador, El Salvador
javier.trigueros17@itca.edu.sv

Abstracto - En este documento se presenta del desarrollo del proyecto final de Microcontroladores, un carrito seguidor de línea, su diseño y estructura, además, aquí se presenta el progreso que se ha tenido en las 5 semanas previas a la entrega, presentación y competencia de los carritos seguidores de línea, además de hablará de algunos problemas y ajustes que debimos hacer para el correcto funcionamiento del mismo incluyendo aspectos que debieron comprobarse ajustarse e incluso repararse el mismo día de la competencia.

I. SEMANA 1: INVESTIGACIÓN

Arrancamos con el proyecto la semana del 7 al 13 de mayo, con el conocimiento obtenido de parte del módulo de Microcontroladores, debimos empezar a buscar información sobre los posibles componentes, módulos, plataformas de programación, etc. Debiendo tomar en cuenta ciertos detalles y requerimientos pues se buscaba que el proyecto fuese de código y hardware libre, por lo que eso reducía las posibilidades, además debimos utilizar placar y circuitos electrónicos 100% creados de forma manual en el caso de los sensores de luz y para el control de los motores de las ruedas se necesitaban drivers que debían ser creados a mano también, a excepción de la placa de desarrollo, tomando en cuenta esto, encontramos lo siguiente:

A. *Sensores*

Para los sensores se encontró el CNY70 [1], que es un sensor óptico reflectante con salida de transistor, compuesto de forma sencilla de un led infrarrojo y un fotodiodo, se utilizó este y no su versión más sencilla (Infrarrojo y fotodiodo separados), por una sencilla razón, y es que, el CNY70 trae a ambos en el mismo encapsulado lo que facilita su manipulación y reduce la posibilidad de errores en el sistema, por señales inesperadas con la versión más sencilla. El enlace a la hoja de especificaciones del sensor se dejará en las referencias para su verificación.

Asimismo, se utilizarán para los sensores, una serie de componentes para la fabricación de un módulo sensor de luz:

- 4 Resistencias 220 ohm (Modulo de sensor CNY70)
- 4 Resistencias 330 ohm (Modulo de sensor CNY70)
- 4 Resistencias 100k ohm (Modulo de sensor CNY70)
- 4 resistencias 1k ohm (Modulo de sensor CNY70)
- 4 Transistores 2N3904[2] (Modulo de sensor CNY70)

B. *Drivers*

Se necesitaba un controlador para los motores, pues si se hubiese enviado una señal directa de los sensores hacia los motores habría sido factible debido al uso de 4 sensores y solo dos motores, aparte de eso, no se puede invertir el giro de los motores de forma directa, lo cual se necesita en caso de que el carrito pierda la línea, pues solo con la ayuda de esa inversión del giro en motores el carrito se puede reubicar para encontrar la línea, por esta razón se utiliza un puente de diodos conocido como puente en H, cuyos componentes principales son:

- 8 Diodos 1N4004[3]
- 1 IC L293D[4]

Con esto ya tendríamos cubierto el controlador.

C. *Plataforma de programación*

Se hizo una investigación de las plataformas de software libre arduino, raspberry pi, etc. Basándonos en el lenguaje de programación, determinamos que Python tiene una interfaz de código más fácil de comprender en nuestro entorno, pues para programarla debemos crear programas que debemos correr desde la consola de la raspberry pi, de hecho para funcionar se necesita que la raspberry tenga instalado un sistema operativo, para este fin se instaló Raspbian, este contaba con el compilador Thonny [5] y su respectiva IDE, el modelo de raspberry utilizado fue:

- Raspberry Pi modelo 3b[6]

Teniendo esto en cuenta la lista de materiales utilizados es:

- 1 Raspberry Pi Modelo 3b
- 4 Sensores CNY70
- 4 Resistencias 220 ohm (Modulo de sensor CNY70)
- 4 Resistencias 330 ohm (Modulo de sensor CNY70)
- 4 Resistencias 100k ohm (Modulo de sensor CNY70)
- 4 resistencias 1k ohm (Modulo de sensor CNY70)
- 4 Transistores 2N3904 (Modulo de sensor CNY70)
- 1 IC L293D
- 8 Diodos 1N4004
- 2 Placas de cobre 10cmx10cm
- 1g de percloruro
- 2 Motores con llantas integradas

Son los materiales que destinamos a comprar para la elaboración de nuestro proyecto.

II. Semana 2: Construcción de los Circuitos[7]

La segunda semana se utilizó para la construcción de las placas que ayudarían al control autónomo de carrito seguidor de línea, tales como los Sensores de luz y el driver de los motores aquí se incluyen los diagramas de los sensores así como, el del puente en H.

Las funciones de los circuitos son las siguientes:

El sensor va conectado a una serie de resistencias una de las cuales transforman el valor en un bit pequeño pero que sea capaz de enviar el valor a un transistor 2N3904 una ves el dato llega al transistor este lo procesa para enviar una 1 lógico legible por la raspberry pi la cual hará que se activen los motores para seguir la linea.

El puente H tiene más funciones, pues si bien es alimentada con 9 voltios o más para levantar los motores, de allí mismo logramos sacar la fuente de alimentación para los sensores que debía de ser de 5 voltios, pues contiene 3 pines de salida de voltaje y conexión de GND o tierra. Además tiene incluidos los pines de las 4 entradas que van 2 entradas a cada parte del puente para controlar los motores de cada lado y que necesita de una fuente de 5 voltios para alimentar las entradas para un mejor control de los motores mismos, debido a la complejidad de la explicación se recomienda observar los diagramas[7]

III. Semana 3: Código [7]

La lógica del carrito seguidor de línea es bastante sencilla, consta de 4 cuatro sensores que deben enviar cada uno un 1 lógico o un 0 lógico, se enviará ese dato al controlador de la placa de desarrollo a utilizar y este procesará dicha información de modo que active uno de dos motores, sea el de la izquierda o el de la derecha, para girar, o en caso necesario ambos para avanzar, y de esta manera hasta completar el recorrido.

Esa es la lógica detrás del funcionamiento, ahora bien la implementación es un poco más compleja, aquí se definió los pines de entrada y salida de la placa de desarrollo micro controlada una vez esto se definió procedimos a enviar los 1 y 0 lógicos a través de los puertos de la raspberry en base a la recepción de datos de los sensores.

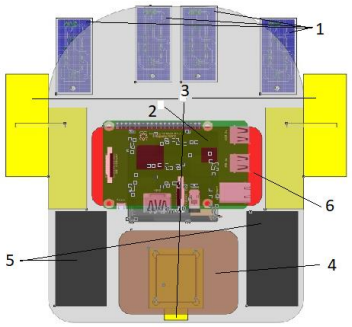
Hubo un punto de la programación en la que tuvimos que realizar varios cambios incluso, llegando al punto de ser necesario crear varios códigos de prueba hasta ver cual funcionaba mejor o en que podíamos depurar los que ya teníamos modificándolos o uniendo los códigos y la lógica con otros de las mismas pruebas que hacíamos.

Un dato importante es que se inició con un código de prueba de 2 sensores que solo probaba 2 sensores luego se agregó la funcionalidad de 4 sensores así como el funcionamiento de los motores en base a la información recopilada de los sensores, se han agregado las posibles funciones necesarias o problemas que pudiesen surgir en el recorrido.

En el caso que se utilice el mismo lenguaje de programación que el utilizado en el presente proyecto es necesario tomar en cuenta que en la raspberry es necesario definir, por nombre y apellido, cada uno de los puertos que se utilizara, siempre que se vaya a utilizar ese puerto en específico; sin embargo, la raspberry pi puede ser utilizada con otros leguajes como Java o C++.

IV. Semana 4: Diseño del chasis[7]

En este punto del documento describiremos o más bien se mostrará el diseño final que tuvo el chasis y se explicará las razones de algunos puntos en el diseño del mismo:



1. Sensores, la posición de los sensores se decidió en curva pues brinda mayor precisión en una curva repentina o en un cambio brusco de dirección en que dos de los sensores estén desfasados permite que logren captar la línea negra y se mantengan dentro del circuito.

2. Raspberry Pi, se definió esta posición por comodidad, si bien es cierto que sirven cualquier parte del chasis, al estar en una posición facilita la conectividad hacia el resto de elementos del carrito.

3. Llantas, como se puede observar se hizo el uso de 3 llantas 2 controladas por motores y 1 rueda "loca" que facilita el giro de las llantas y brinda estabilidad, por esa precisa razón se definió en "V" la posición de las llantas, además brinda un balance casi perfecto que permite el buen desempeño del carrito.

4. Puente en H, conectividad es lo que defina la posición pues es del puente en H que se alimentan 3 de los sensores y los motores.

5. Pack de baterías, a su vez el puente en H es alimentado por 4 packs de baterías, que convenientemente están situados a los lados del Puente en H para mayor facilidad de energizarlo.

6. PowerBank 5V, es la que nivela el peso del carrito por eso va al medio del chasis para balancear el peso del carrito.

V. Semana 5: Pruebas

Empiezan las pruebas, a este respecto, fue complicado al inicio, ya teníamos todos los elementos montados en el chasis y lamentablemente no funcionó debido a una línea de código que se definió mal:

Definimos que si los 4 sensores estaban en 1, avanzara, sin embargo, así quedaba directo y no detectaba la línea negra que debía seguir:

```
if(IO.input(14)==True and IO.input(15)==True and
IO.input(4)==False and IO.input(17)==True):

    IO.output(5,True) #1A+
    IO.output(6,False) #1B-

    IO.output(13,True) #2A+
    IO.output(19,False) #2B-
```

Al darnos cuenta editamos esa línea de código e hicimos una nueva función donde se definía directamente que si no detectaba ninguna condición de giro sea derecha o izquierda fuese hacia adelante, y luego lo único que se hizo mandar a llamar la función:

```
def adelante():                    #Funcion adelante
    IO.output(5,True) #1A+
    IO.output(6,False) #1B-

    IO.output(13,True) #2A+
    IO.output(19,False) #2B-
```

VI. Día de la competencia

Llega el día de la competencia, aquí empezamos con el pie izquierdo nuevamente, no tuvimos problemas ni con el Hardware ni el Software, sin embargo, no funcionó el carrito en la primera vuelta por un detalle mínimo, olvidamos conectar la fuente de voltaje de 3 de los sensores. Una vez se conectó la fuente de alimentación de esos 3 sensores, el carrito ya funcionó he hizo los siguientes tiempos:

Vuelta 1	Vuelta 2	Vuelta 3	Vuelta 4	Promedio
Fallo	26.55 s	29.81 s	26.45 s	20.7 s

Como notamos se pudo bajar el tiempo record 10 milésimas más esto se hizo modificando los sensores de manera física, en la tercera vuelta pues la pista estaba desnivelada en una parte, y la calibración de los sensores había sido establecida de forma que uno de los sensores, específicamente el de la izquierda estaba más abajo que el otro.

Por ese motivo tuvimos que hacer la re calibración de los sensores y el sensor central izquierdo, lo nivelamos al sensor central derecho, así logramos hacer que funcionara nuevamente y para la última vuelta logramos bajar los 0.10 s

References

[1] Datasheet CNY70: www.alldatasheet.com/datasheet-pdf/pdf/26332/VISHAY/CNY70.html

[2] Datasheet 2N3904: pdf1.alldatasheet.com/datasheet-pdf/view/11470/ONSEMI/2N3904.html

[3] Datasheet 1N4004: pdf1.alldatasheet.com/datasheet-pdf/view/14621/PANJIT/1N4004.html

[4] Datasheet L293D: pdf1.alldatasheet.com/datasheet-pdf/view/22432/STMICROELECTRONICS/L293D.html

[5] http://thonny.org/

[6] Datasheet Raspberry Pi 3b: http://docs-europe.electrocomponents.com/webdocs/14ba/0900766b814ba5fd.pdf

[7] Para más información, y la verificacion de diagramas, esquemas y diseño del chasis: https://github.com/LineFJAED/LineFollowerJAED