Universidad del Valle de Guatemala
Algoritmos y Estructura de Datos
Luis Rodrigo Urrutia Castellanos #16139
Kevin Sebastián Macario #17369
Antonio Reyes #17273

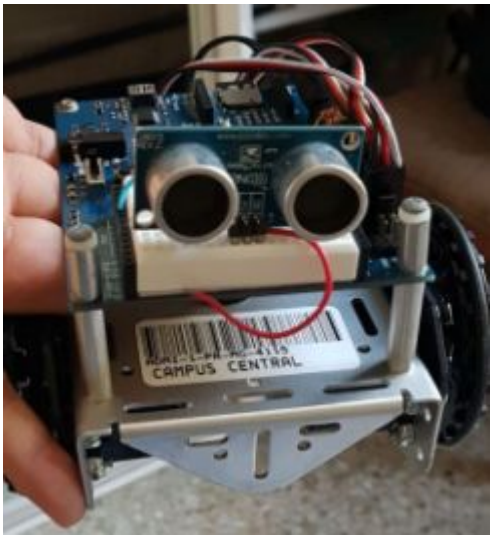
Fecha: 16 de febrero del 2018
Sección: 20
Instructor: Diego Enriquez
Auxiliar: Cristian Morales

Proyecto 1 - Fase 2

Para el primero proyecto de la clase de Algoritmos y Estructuras de Datos, se diseñó un algoritmo escrito en C, para que un robot Parallax fuera capaz de salir de cualquier laberinto. Se investigaron distintos métodos para resolver laberintos como El Algoritmo de Tremaux, de Garantía, de Cadena, de Mano Derecha, entre otros. A continuación se muestra una explicación a mayor detalle sobre el Algoritmo utilizado en el Proyecto y las preparaciones que debieron realizarse para implementar el algoritmo modificado de la Mano Derecha (*Wall follower).*

Preparación del robot

Se solicitó en la Facultad de Ingeniería en Ciencias de la Computación un robot Parallax. Dicho robot proveía un Sensor de movimiento “Ultrasonic Distance Sensor #28015” que es capaz de identificar un objeto entre una distancia entre 3 cm a 3 m dependiendo de la modificación (Inc, 2018).

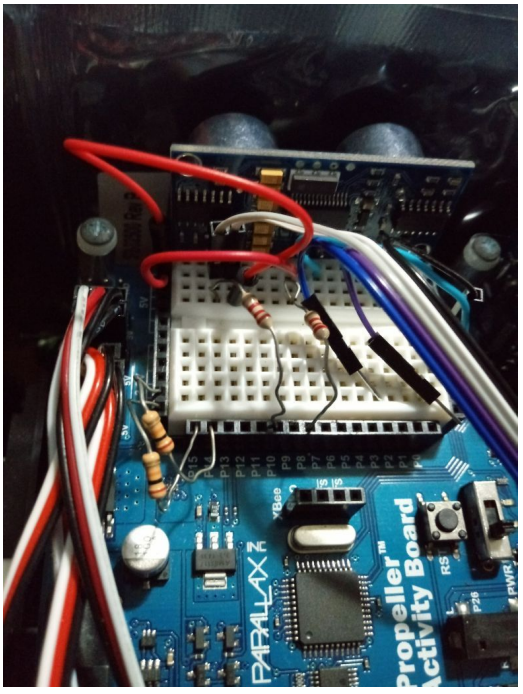


Materiales entregados por la facultad.

El Algoritmo realizado en el proyecto se basó en el “**Algoritmo Wall Follower**”, también conocido como Regla de la Mano Derecha o Izquierda, este algoritmo consiste en recorrer una pared que se encuentra a su lado.

Instructivo:

1. Al poseer el Parallax, para prepararlo se deben de seguir los siguiente pasos:
2. Revisar si todo los componentes del Parallax están conectados en los pines correspondientes, se adjunta imagen de cómo debe lucir el circuito:



Resistores de 2.2 ohm colocados en Pin 8 y 10. Además, se le conecta al sensor cable a tierra y hacia el voltaje (5V) ubicado en la parte izquierda del *board*.

3. Colocar 5 baterías AA al reverso del Parallax.
4. Descargar la interfaz de parallax en el sitio web oficial de Parallax.
5. Instalar interfaz de parallax conocida como Simple IDE. Link de descarga: https://learn.parallax.com/tutorials/language/propeller-c/propeller-c-set-simpleide
6. Leer la documentación para relacionarse con la sintaxis del lenguaje de programación C para empezar a programar.
7. Antes de empezar a programar, descargar código de calibración de sensores.
8. Conectar el parallax a la computadora por medio del cable brindado en la caja.
9. Colocar el switch en la posición 1, esto se debe de hacer siempre que se quiera compilar el exportar el código.
10. Compilar y exportar el código.
11. Colocar el switch en la posición 2, esto hará que el código que se exportó anteriormente sea ejecutado.
12. Realizando este instructivo está listo para realizar y crear códigos de su autoría.

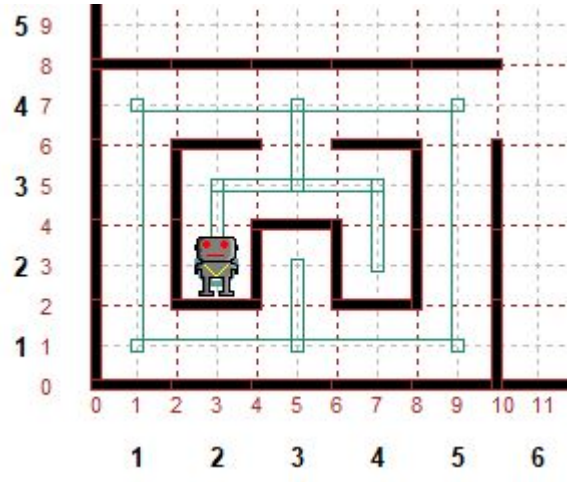
Ventajas del Algoritmo Wall Follower:

- No presenta dificultad en la implementación.
- Es posible manipular el código de una manera sencilla para hacer que la eficiencia del algoritmo incrementa.
- El algoritmo ocupa menos memoria que los otros Algoritmos.

Desventajas del Algoritmo Wall Follower:

- Si el algoritmo comienza dentro de un *loop*, una “isla”, no será posible encontrar una solución.

La imagen a continuación muestra el problema que debe solucionarse al implementar el Algoritmo Wall Follower:



Se implementó otro sensor en el lado derecho del robot, este se utilizará para poder identificar si existe una pared al lado derecho de Parallax, en caso de que el sensor no encuentre una pared, este girará a la derecha. En caso de que tanto el sensor frontal y derecho identifiquen una pared, el Parallax se moverá a la izquierda. El link a continuación muestra la implementación de ambos sensores, demostrando el comportamiento del Parallax al detectar una pared en el sensor derecho y en ambos sensores.

Link: https://www.youtube.com/watch?v=I5ncSujGOAY

Métodos elaborados en código:

Main: Método principal.
Consiste de un ciclo con el cual trabaja con el sensor frontal y derecho del robot. El sensor derecho trabaja con una distancia de alcance de 26 cm, este busca una pared a la cual seguir, cuando encuentra una camina derecho hasta un cruce o intersección del laberinto. El sensor frontal se utiliza para identificar si existe algún obstáculo frente a él, si encuentra uno realiza un giro de 45° hasta que ya no haya obstáculo alguno. El algoritmo implementado se denota como “Algoritmo de la mano derecha”.

Contador: Agrega una unidad de movimiento cada vez que el robot se mueve hacia delante.
Coordenadas: Método que controla el sentido y dirección del robot en todo momento.

Ambiente de programación: SIMPLE IDE

Comparación con Fase 1 del Proyecto

En teoría y de manera similar al algoritmo realizado en Rurple, el algoritmo realizado en C funciona. Sin embargo, el robot no realiza los giros de manera consistente. Por ello, se le asignaron los siguientes *specs*: Se estableció que el sensor derecho tendría que identificar una pared a una distancia de 15 cm con el fin de que el robot tuviera el espacio suficiente para poder realizar un giro. También se estableció que el sensor frontal tendría que identificar una pared a una distancia de 10 metros, de esta manera, se puede realizar un cruce sin que el robot raspe las paredes y hay espacio suficiente para que el sensor derecho pueda identificar la siguiente pared.

Las velocidad de las ruedas para realizar el giro fueron establecida utilizando las variables “giro” cuyo valor es 40, para poder realizar el cruce a la derecha, debe especificarse que la velocidad de la rueda derecha debe ser -40 mientras que la velocidad de la rueda izquierda debe ser 40. En caso de querer cruzar a la izquierda, el valor de la velocidad de la rueda izquierda debe ser negativo.

Solución de laberinto (Prueba)

https://www.youtube.com/watch?v=2xrjs4Y6k8k

Solución de laberinto (En clase)

https://youtu.be/HPaG7nd1uLA

Link a Repositorio

https://github.com/RodrigoUrrutiaC/Proyecto1-AED

Bibliografía

Inc, P. (2018). *Parallax Inc*. Obtenido de https://www.parallax.com/product/28015