TORNEO 2021 RF ROBOTIC FEST

TORNEO ROBOTIC FEST

¡UTILIZANDO SENSORES PARA SEGUIR UNA LÍNEA!

WWW.ROBOTICFEST.CL

Ministerio de Educación
Gobierno de Chile

FUNDACIÓN país digital

TORNEO ROBOTIC FEST

¡HOLA A TODAS Y TODOS!

Seguimos avanzando en el mundo de la simulación a través de Webots, esta vez, para complementar las instrucciones que descubrimos en la guía anterior, incluyendo un elemento clave para poder resolver los desafíos que serán parte de este torneo: seguir una línea. Son varios los algoritmos que podemos considerar al momento de seguir una línea, y es parte del desafío de este torneo el que junto a tu equipo puedan descubrir cuál es la mejor forma de hacerlo, sin embargo, y para lograr que todas y todos quienes participen en esta versión puedan comprender cómo se programa el robot e-puck de Webots, en esta guía abordaremos el uso de este elemento clave para construir un algoritmo seguidor de línea: **EL USO DE SENSORES.**

¿QUIERES CONOCER MÁS ACERCA DE ESTE TEMA?

¡VAMOS!

Anteriormente vimos cómo poder programar el robot e-puck para que pudiese avanzar dentro de los escenarios de pruebas, de forma autónoma, a partir de instrucciones de movimientos. Cuando queremos lograr que nuestro robot pueda seguir una línea de manera autónoma, debemos lograr un comportamiento similar al que obtuvimos cuando programamos las instrucciones de movimiento. En este caso, el robot debe ser capaz de reconocer una línea dibujada en el piso del escenario y según la trayectoria que describa esta línea, seguirla. Una alternativa posible podría ser entregar instrucciones de movimiento, una tras de otra, que permita describir todo el circuito dibujado por la línea, es decir, si conocemos cuál es la trayectoria de la línea podemos de forma manual, entregar todas las instrucciones que necesitamos para que el robot describa ese circuito. Si bien esta es una solución válida, tiene varias desventajas. La primera, debemos conocer el circuito previamente, por lo cual, ante cualquier cambio en el circuito, nuestro código ya no podrá seguir esta línea. Por otro lado, para poder seguir la línea de forma manual, debemos ser muy precisos con el conteo del tiempo, ya que probablemente en nuestra solución entregamos instrucciones del tipo "avanza por 2 segundos, luego gira medio segundo a la izquierda, luego avanza 2 segundos más, etc...". Esto podría resultar para cierto tipo de circuitos, pero no es la manera óptima de realizar la solución. Entonces, ¿qué podemos hacer?.

Como pudieron intuir, los robots disponen de ciertas herramientas que les permiten medir cosas del exterior, "sensar" estos datos. Un robot mediante una cámara podría mirar lo que hay en el escenario y a partir de eso, determinar cuál es el mejor camino a seguir. Cuando queramos medir algo del exterior para incorporar esta medición al procesamiento que puede hacer el robot, necesitaremos usar sensores. Existen varios sensores que se pueden incorporar a los robots como por ejemplo, sensores de ultrasonido (para detectar obstáculos o medir distancias), sensores infrarrojos (para detectar obstáculos o reconocer valores de luminosidad), cámaras, entre otros. En el caso de nuestro robot e-puck podemos encontrar varios de estos sensores, siendo los más importante para poder reconocer una línea, los sensores que encontramos en la parte inferior del robot, que se encuentran apuntando hacia el piso. Estos sensores corresponden a sensores infrarrojos (IR) que nos permitirán establecer una relación a partir de la cantidad de luz que refleja el piso hacia el robot. Por

un lado, estos sensores emiten una luz infrarroja la cual rebota en el piso y se devuelve al robot. Según el porcentaje de luz que logra capturar al volver puede determinar el "color" del piso. (Distintos colores absorben distintos valores de luz, por lo tanto, se devuelven en distintos niveles al sensor). Si bien esa es la forma clásica en la que trabajan estos sensores, en Webots debemos verlos de una forma un poco diferente (considerando que es una simulación). Por lo mismo es que estos sensores, junto a una correcta programación, te permitirán seguir cualquier tipo de línea. En esta guía te mostraremos cómo declarar 3 sensores IR de línea o de piso, y te entregaremos una versión bastante simple de un algoritmo para seguir una línea. ¡Veamos cómo se programa!

01.

LIBRERÍAS

En esta oportunidad, mantendremos exactamente las mismas librerías que incluimos en el código para lograr que el robot se mueva de forma recta. De esta forma nuestro código debería verse de la siguiente manera:

```
#include <stdio.h>
#include <webots/distance_sensor.h>
#include <webots/light_sensor.h>
#include <webots/motor.h>
#include <webots/robot.h>
```

02.

DEFINICIONES

Mantendremos gran parte de las definiciones que creamos anteriormente (ya que nos irán sirviendo en la medida que agreguemos más elementos a nuestro código) y agregaremos algunas exclusivas para facilitar el uso del sensor de línea. Para mayor claridad, hemos reemplazado algunas de las declaraciones anteriores con nombres que describen mucho mejor su funcionamiento. El código se verá de la siguiente manera:

```
// Definiciones
#define TRUE 1
#define FALSE 0
#define IZQUIERDA 0
#define DERECHA 1
#define BLANCO 0
#define NEGRO 1
#define TIME_STEP 32  // [ms]

// Definiciones para uso de sensor IR de piso para seguir linea
#define NUMERO_SENSORES_LINEA 3
#define VALOR_BLANCO 600
#define SENSOR_LINEA_IZQUIERDO 0
#define SENSOR_LINEA_CENTRO 1
#define SENSOR_LINEA_DERECHO 2
```

Como se puede apreciar en el código anterior, hemos creamos un par de constantes que nos permitirán manipular de mejor manera estos sensores: número de sensores a utilizar (tres), valor en el cual podemos asumir que la medición del sensor nos entrega un color que podemos categorizar como “más claro”, y el identificador numérico correspondiente a cada sensor (cero, uno y dos).

03.

DECLARACIONES

a) WbDeviceTag:

Creamos una referencia a los elementos principales que utilizaremos del robot: motores y sensor de línea (o piso). Para ello, y tal como lo hicimos la vez anterior, declararemos esta referencia de tipo WbDeviceTag, como se puede apreciar en la imagen:

```
// Motores
WbDeviceTag motor_izquierdo, motor_derecho;

// Sensor de linea
WbDeviceTag sensor_linea[NUMERO_SENSORES_LINEA];
```

Notar que en el caso del sensor de línea, este es declarado como un arreglo cuyo tamaño es tres (dado por el valor de la constante NUMERO_SENSORES_LINEA). Debemos recordar que los arreglos funcionan a través de índices que indican su posiciones, para lo cual también utilizaremos las definiciones señaladas anteriormente. En este caso, usamos un valor de tres ya que podemos utilizar hasta 3 sensores de línea o piso.

b) Arreglo:

Adicionalmente, declaramos un arreglo de tipo entero (int) que nos permitirá almacenar el valor medido por cada uno de los sensores. Dado que tenemos tres sensores, crearemos un arreglo del mismo tamaño, donde el índice corresponde al valor medido por cada sensor (uno, dos o tres según el valor del índice).

```
int sensor_linea_valores[NUMERO_SENSORES_LINEA] = {0, 0, 0};
```

Entregaremos valores por defecto de cero a cada uno de estos sensores.

04.

PROGRAMA PRINCIPAL

El programa principal será bastante similar al presentado en la guía anterior y que fue entregado como ejemplo en los mundos de pruebas compartidos en los recursos del torneo. Principalmente, el código inicializará los elementos de Webots que permiten la simulación, así como también la inicialización de motores. La gran diferencia está en que en esta oportunidad, además de inicializar motores, debemos inicializar los sensores y programar el algoritmo que permita seguir la línea. Siguiendo con esto, la primera parte del código debería verse como sigue:

```
/* inicializamos la simulacion de Webots */
wb_robot_init();

/* Trabajando con los motores */
// motors
motor_izquierdo = wb_robot_get_device("left wheel motor");
motor_derecho = wb_robot_get_device("right wheel motor");


/* Inicializamos los motores */
wb_motor_set_position(motor_izquierdo, INFINITY);
wb_motor_set_position(motor_derecho, INFINITY);
wb_motor_set_velocity(motor_izquierdo, 0.0);
wb_motor_set_velocity(motor_derecho, 0.0);
```

En este ejemplo, lo primero que haremos será inicializar la simulación en Webots para luego utilizar la instrucción wb_robot_get_device() para asociar cada motor presente en el robot con su correspondiente WbDeviceTag. Por último, inicializamos los motores para ser controlados a través de su velocidad.

El siguiente paso será inicializar los sensores, que en este caso, serán 3. Para ello, utilizaremos la instrucción wb_robot_get_device(), tal como lo hicimos con los motores, pero esta vez, entregamos el identificador correspondiente a los sensores. Repetiremos este proceso 3 veces y utilizaremos como apoyo las definiciones creadas al comienzo del código.

```
sensor_linea[SENSOR_LINEA_IZQUIERDO] = wb_robot_get_device("gs0");
wb_distance_sensor_enable(sensor_linea[SENSOR_LINEA_IZQUIERDO], TIME_STEP);

sensor_linea[SENSOR_LINEA_CENTRO] = wb_robot_get_device("gs1");
wb_distance_sensor_enable(sensor_linea[SENSOR_LINEA_CENTRO], TIME_STEP);

sensor_linea[SENSOR_LINEA_DERECHO] = wb_robot_get_device("gs2");
wb_distance_sensor_enable(sensor_linea[SENSOR_LINEA_DERECHO], TIME_STEP);
```

La instrucción wb_distance_sensor_enable() es la que finalmente habilitará los sensores para que puedan leer los valores asociados a la línea. ¡Importante! Como podrán notar, estamos utilizando una función cuyo nombre hace referencia a un “sensor de distancia” (wb_distance_sensor_enable) cuando en realidad, lo que queremos hacer para seguir una línea es reconocer si estoy dentro de una superficie clara u oscura. Esto se debe a que tanto los sensores de distancia (que nos permitirán distinguir la proximidad hacia un objeto) como los sensores de línea o piso con los cuales trabajaremos en Webots, corresponden a sensores Infrarrojos (IR). La principal diferencia en la forma que utilizaremos estos

sensores está al momento de asociar el sensor a la referencia en Webots con el WbDeviceTag, donde utilizaremos los nombres gs0, gs2 y gs3 para referirnos al sensor de línea o piso, o ps0, ps1, ps2, ..., para referirnos al los sensores de proximidad. Más adelante, cuando revisemos en detalle cómo reconocer un obstáculo veremos esto de forma concreta. Lo que deben tener presente es que más allá de utilizar funciones que en sus nombres hacen referencia a un sensor de distancia, cuando asocian la función wb_robot_get_device() a uno de los sensores de línea (gs0, gs1 o gs2) están trabajando con los sensores que se encuentran apuntando al piso.

Habiendo completado todo esto en nuestro código, el siguiente paso será programar el ciclo infinito principal que se encargará de controlar la ejecución del robot. En este ciclo incluiremos los siguientes elementos: por un lado, la lectura de los sensores de piso y por otro, el algoritmo para seguir una línea:

a) Lectura de sensores:

Para utilizar los sensores de línea utilizaremos la instrucción wb_distance_sensor_get_value(), cuyo retorno nos dirá cuánto es el valor que entrega el sensor que estemos consultando. Guardaremos este resultado en el arreglo que hemos creado anteriormente, manteniendo los índices que hemos definido (0 para el sensor izquierdo, 1 para el sensor de centro y 2 para el sensor derecho). La imagen a continuación muestra cómo debería verse esta sección del código:

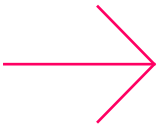
```
/* Leemos los valores del sensor */

sensor_linea_valores[0] = wb_distance_sensor_get_value(sensor_linea[0]); // gs0
printf("Sensor izquierdo = %d | ", sensor_linea_valores[0]); // valor leido por gs0

sensor_linea_valores[1] = wb_distance_sensor_get_value(sensor_linea[1]); // gs1
printf("Sensor central = %d | ", sensor_linea_valores[1]); // valor leido por gs1

sensor_linea_valores[2] = wb_distance_sensor_get_value(sensor_linea[2]); // gs2
printf("Sensor derecho = %d\n", sensor_linea_valores[2]); // valor leido por gs2
```

Hemos agregado una instrucción adicional (printf()) en cada lectura del sensor, que nos permitirá escribir en la consola cuál es el valor leído por el sensor. Te recomendamos que constantemente leas los valores de los sensores y los vayas mostrando en la consola, de esa manera podrás saber con exactitud cuales son los valores medidos y cómo vas a poder interpretar estos resultados. Notar que a la función wb_distance_sensonr_get_value(), le debemos decir cual sensor que quiero utilizar. Por comodidad, hemos omitido el ciclo infinito para enfatizar solamente en la lectura de los sensores.



b) Algoritmo para seguir una línea

Existen diversos tipos de algoritmos para seguir una línea. Algunos serán más rápidos, otros más precisos. Algunos usarán un sensor, otros usarán dos o tres. El tipo de solución que implementarán dependerá exclusivamente de cómo quieran resolver el desafío del Torneo. Para el ejemplo que entregaremos en esta guía estamos utilizando una versión simplificada de una forma clásica de resolver un seguidor de línea: utilizaremos un solo sensor del robot (el central) y evaluaremos si estamos posicionados en un color claro o en un color oscuro. En caso de estar en un color claro realizaremos un leve giro hacia un lado (izquierda), mientras que si estamos posicionados sobre un color oscuro, realizaremos un giro hacia el otro lado (derecha). Este sencillo procedimiento debería lograr que el robot sea capaz de seguir una línea. Si todavía tienen dudas, los invitamos a que tomen un papel y lápiz y dibujen posibles trayectorias simulando este movimiento, girar a la izquierda y derecha dependiendo del valor que nos entregue nuestro sensor. Es importante precisar que el sensor que estaremos utilizando no nos permite distinguir con exactitud diferencias de colores, sin embargo, podemos detectar fácilmente contrastes o zonas más claras u oscuras. Te recomendamos mostrar en la consola qué es lo que imprimen los sensores, para que de esa forma seas tu mismo quen interpreta qué valores entregados por el sensor corresponden al negro de la línea o al blanco del piso. El código presentado a continuación resume esta solución:

```
if (sensor_linea_valores[SENSOR_LINEA_CENTRO] > VALOR_BLANCO){

  /* Izquierda */
  wb_motor_set_velocity(motor_izquierdo, 0.00628 * 100);
  wb_motor_set_velocity(motor_derecho, 0.00628 * -50);

} else{

  /* Derecha */
  wb_motor_set_velocity(motor_izquierdo, 0.00628 * -50);
  wb_motor_set_velocity(motor_derecho, 0.00628 * 100);
}

wb_robot_step(250);
```

05.

CÓDIGO COMPLETO

A continuación les compartimos la solución completa que muestra cómo utilizar los sensores infrarrojos (IR) ubicados en la parte inferior del robot, para poder seguir una línea.

¡Mucho éxito!

```
#include <stdio.h>
#include <webots/distance_sensor.h>
#include <webots/light_sensor.h>
#include <webots/motor.h>
#include <webots/robot.h>

// Definiciones
#define TRUE 1
#define FALSE 0
#define IZQUIERDA 0
#define DERECHA 1
#define BLANCO 0
#define NEGRO 1
#define TIME_STEP 32  // [ms]


// Definiciones para uso de sensor IR de piso para seguir linea
#define NUMERO_SENSORES_LINEA 3
#define VALOR_BLANCO 600
#define SENSOR_LINEA_IZQUIERDO 0
#define SENSOR_LINEA_CENTRO 1
#define SENSOR_LINEA_DERECHO 2


// Motores
WbDeviceTag motor_izquierdo, motor_derecho;

// Sensor de linea
WbDeviceTag sensor_linea[NUMERO_SENSORES_LINEA];


int sensor_linea_valores[NUMERO_SENSORES_LINEA] = {0, 0, 0};


int main() {


  /* inicializamos la simulacion de Webots */
  wb_robot_init();


  /* Trabajando con los motores */

 // motors
  motor_izquierdo = wb_robot_get_device("left wheel motor");
  motor_derecho = wb_robot_get_device("right wheel motor");


  /* Inicializamos los motores */
  wb_motor_set_position(motor_izquierdo, INFINITY);
  wb_motor_set_position(motor_derecho, INFINITY);
  wb_motor_set_velocity(motor_izquierdo, 0.0);
  wb_motor_set_velocity(motor_derecho, 0.0);


 sensor_linea[SENSOR_LINEA_IZQUIERDO] = wb_robot_get_device("gs0");
 wb_distance_sensor_enable(sensor_linea[SENSOR_LINEA_IZQUIERDO], TIME_STEP);

 sensor_linea[SENSOR_LINEA_CENTRO] = wb_robot_get_device("gs1");
 wb_distance_sensor_enable(sensor_linea[SENSOR_LINEA_CENTRO], TIME_STEP);

 sensor_linea[SENSOR_LINEA_DERECHO] = wb_robot_get_device("gs2");
 wb_distance_sensor_enable(sensor_linea[SENSOR_LINEA_DERECHO], TIME_STEP);
```

```
    /* Leemos los valores del sensor */

    sensor_linea_valores[0] = wb_distance_sensor_get_value(sensor_linea[0]); // gs0
    printf("Sensor izquierdo = %d | ", sensor_linea_valores[0]); // valor leido por gs0

    sensor_linea_valores[1] = wb_distance_sensor_get_value(sensor_linea[1]); // gs1
    printf("Sensor central = %d | ", sensor_linea_valores[1]); // valor leido por gs1

    sensor_linea_valores[2] = wb_distance_sensor_get_value(sensor_linea[2]); // gs2
    printf("Sensor derecho = %d\n", sensor_linea_valores[2]); // valor leido por gs2


    /* evaluamos el valor del sensor de linea central, si este valor es blanco,
    | giramos a la izquierda, sino, a la derecha */


    if (sensor_linea_valores[SENSOR_LINEA_CENTRO] > VALOR_BLANCO){

      /* Izquierda */
      wb_motor_set_velocity(motor_izquierdo, 0.00628 * 100);
      wb_motor_set_velocity(motor_derecho, 0.00628 * -50);

    } else{

      /* Derecha */
      wb_motor_set_velocity(motor_izquierdo, 0.00628 * -50);
      wb_motor_set_velocity(motor_derecho, 0.00628 * 100);
    }

    wb_robot_step(250);
  }

  return 0;
}
```

TORNEO ROBOTIC FEST

¡MUCHO ÉXITO!

:)

TORNEO ROBOTIC FEST

WWW.ROBOTICFEST.CL

Ministerio de Educación
Gobierno de Chile

FUNDACIÓN país digital