Universidad de Costa Rica
Escuela de Ingeniería Industrial
TC-629 Kit Robótico

Taller #2:
GiraBot

Robokit

---

1. Metas

- Simular el movimiento de los girasoles para la captura de luz solar a partir de sensores y actuadores.
- Conocer la importancia de algunos sensores que permiten interactuar con el ambiente y la posible automatización de procesos.

2. Funcionamiento

Un fotoresistor es una resistencia que cambia su valor a medida que le incide luz, aprovechando este efecto se mide ese cambio para saber cuál fotoresistor está recibiendo más luz. Con el GiraBot, se colocan dos fotoresistores e implementando la debida programación se logrará hacer que el girasol se mueva en el sentido que haya más luz. Actualmente, muchos paneles solares se encuentran estáticos, sin posibilidad de girar como los girasoles. Este taller es una posible solución en pequeña escala que permitirá a un panel solar moverse desde el horizonte donde sale el Sol y lo acompañará hasta donde se oculte, aprovechando la máxima incidencia durante todo el día. Las resistencias que se colocan en serie con las fotoresistencias cumplen la función de evitar que ocurra un cortocircuito cuando la resistencia de la fotoresistencia es cercana a cero.

2.1. Fotoresistencia

Dado que la fotoresistencia es un elemento fundamental para este taller, se detallará su funcionamiento con mayor profundidad a continuación. Una manera de explicar su funcionamiento es pensando que cuando la resistencia no se expone a radiación luminosa los átomos se mantienen en sus niveles de energía base, haciéndolo resistivo naturalmente, pero cuando la superficie es incidida con fotones, estos son capaces de transmitir su energía a los electrones para que salten a la capa de conducción, de manera que al volverse más conductivo su valor resistivo baja.

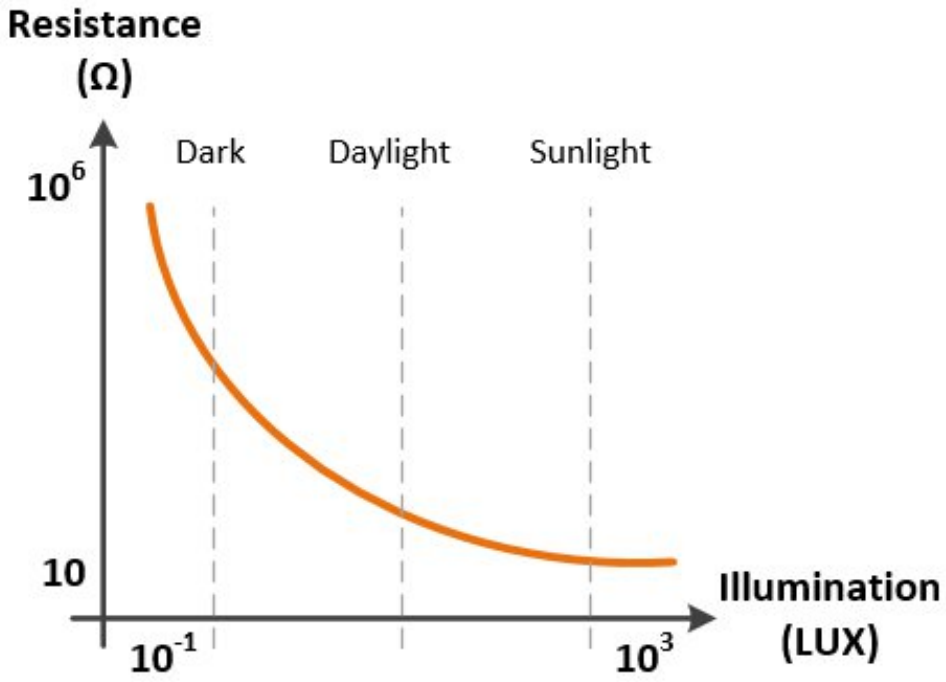


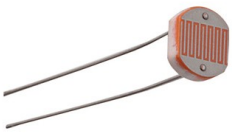
Figura 1: Curva de resistencia al cambiar la intensidad luminosa.

3. Materiales

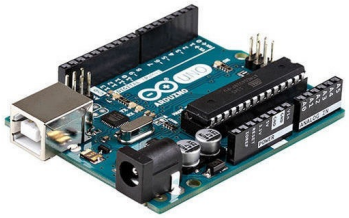
Aparte de la protoboard y los cables para hacer conexiones, el resto de los materiales a utilizar en este taller se muestran en la imagen 2. Además, se necesitan las piezas mostradas en la figura 3 para armar la estructura del GiraBot y sus componentes.

(a) Servomotor

(b) Resistencia

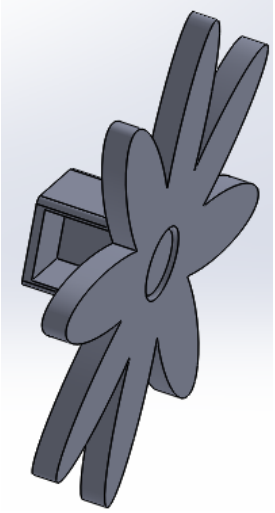


(c) Fotoconductor

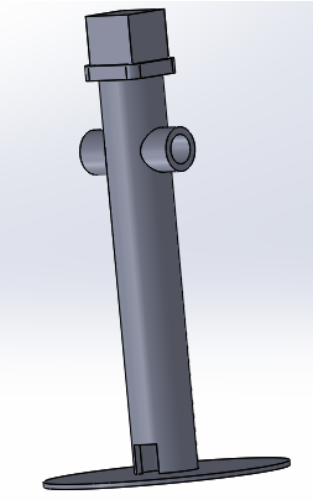


(d) Arduino

Figura 2: Equipo a utilizar.



(a) Flor



(b) Tallo

Figura 3: Piezas del GiraBot.

4. Actividades

1. Armar el circuito con las fotoresistencias como se muestra en la figura 4.
2. Los cables deben ingresar por el tallo de la flor para conectar las fotoresistencias con cable tipo hembra. Se puede alargar el cable mediante combinaciones de tipo hembra y macho si es necesario.
3. La base del tallo de la flor se debe adherir al servo con algún tipo de cinta, de forma sencilla.
4. Se recomienda que las resistencias sean del mismo valor y menores a 100 Ω.
5. Unir el tallo con los pétalos de la flor mediante la unión cuadrada.
6. Incluir la biblioteca para utilizar el servo de ser necesario y crear un objeto Servo.
7. Declarar e inicializar los pines y variables necesarios.
8. Establecer las configuraciones de los pines y la comunicación, y tasa de transmisión de datos del puerto serial.
9. Programar lo que se pide en cada uno de los siguientes casos, estudiando y entendiendo el código con ayuda de los diagramas de flujo en cada caso.

5. Diagrama de conexiones

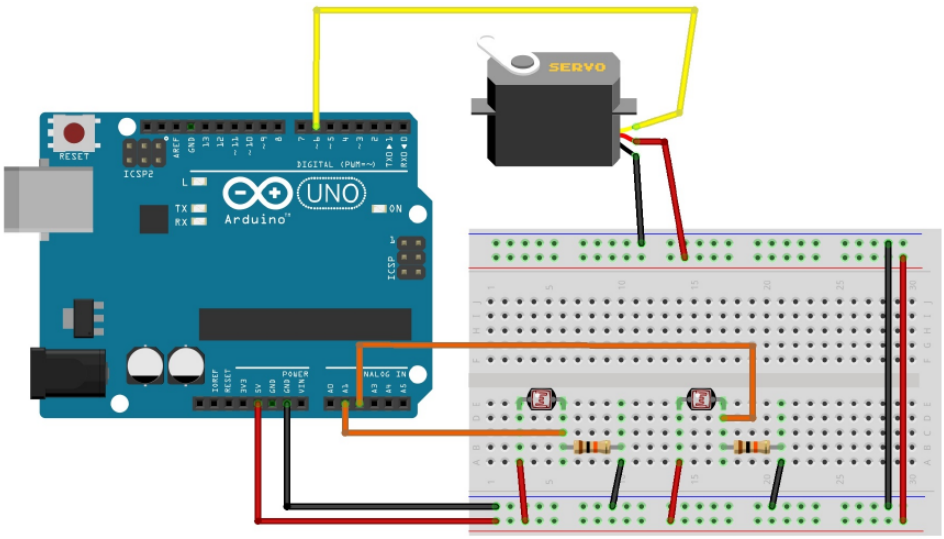


Figura 4: Conexiones del circuito.

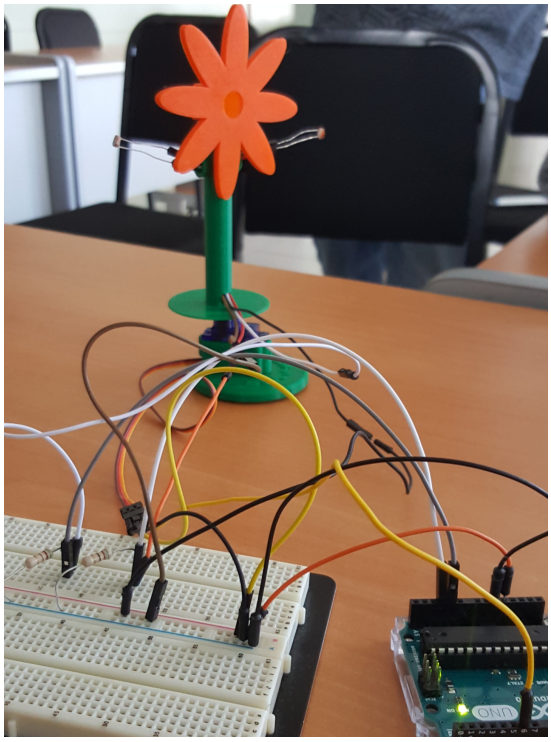


Figura 5: Estructura de GiraBot y sus conexiones.

6. Parte 1: Fotoresistores

Se debe verificar el funcionamiento de los fotoresistores. Para ello, se usa el siguiente código y con la linterna de un celular se verifica el cambio en las lecturas.

```
int dato1 = 0; // dato sensor 1
int dato2 = 0; // dato sensor 2
const int sensor1 = A1; //Pines fotoresistores
const int sensor2 = A2;
//const int --> variable que guarda un dato que no va a cambiar.
void setup() {
pinMode(sensor1,INPUT); // sensor --> entrada
pinMode(sensor2,INPUT); // sensor --> entrada
Serial.begin(9600); // iniciar el puerto Serial
}
void loop() {
dato1 = analogRead(sensor1); // guardo la lectura
dato2 = analogRead(sensor2);
// El rango de dato es entre 0 y 1023.
delay(1000); // 0.5s de esperar para no saturar.
Serial.print("Sensor 1 ----> "); Serial.print(dato1);
Serial.print(" Sensor 2 ----> "); Serial.println(dato2);
}
```

En el código anterior se pueden destacar varios detalles, entre ellos el hecho de que lo que se está haciendo es definir las variables y luego leer las entradas en los pines analógicos, las cuales tienen salidas entre 0 y 1023, pero no tienen correspondencia a un valor físico a menos que se caracterice el sensor y se pueda hacer esta conversión. Posterior a la lectura se despliegan los valores leídos por medio de las impresiones en pantalla.

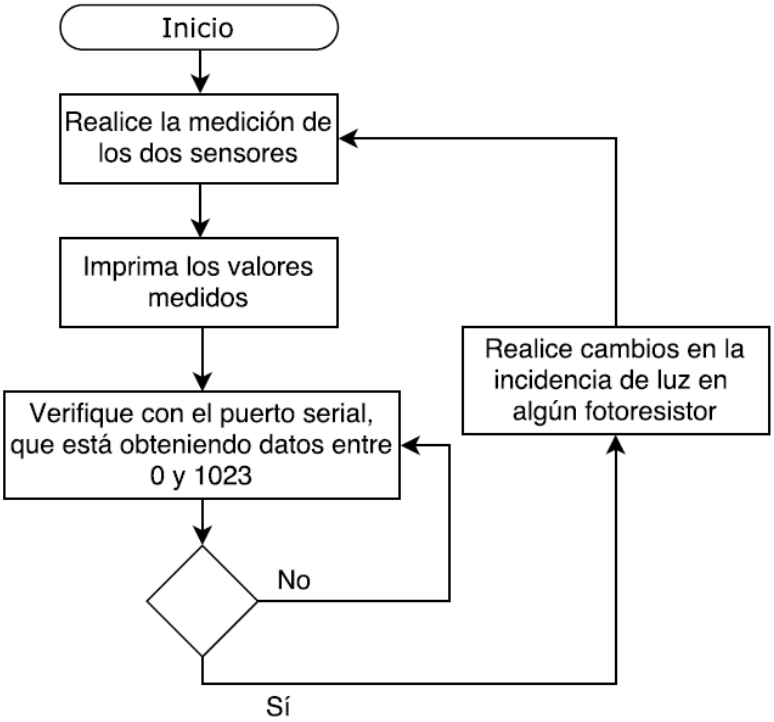


Figura 6: Diagrama de flujo para prueba de fotoresistencias.

7. Parte 2: Servomotor

Se debe verificar el funcionamiento del servo. Para ello, se usa el siguiente código y se verifica visualmente el giro del servo.

```
#include <Servo.h> // Bibliotecas
Servo servo; // creacion del objeto
const int motor = 6; // Variables
int posicion = 0;
void setup(){ // Configuraciones
servo.attach(motor); // motor --> pin 6
servo.write(90); // centrar el motor
Serial.begin(9600); // iniciar el puerto Serial
}
// Creamos una funcion que mueva al servo desde 10 grados hasta 170
   grados.
void explorar(){
//** Sentido Antihorario **//
```

```
for(posicion = 10; posicion <= 170; posicion++){
servo.write(posicion); // Se mueve hacia el angulo "posicion"
delay(15); // Esperamos para no saturar.
      } // fin del for
} // fin de la funcion explorar
void loop(){
explorar(); // Llamamos nuestra funcion.
}
```

En este extracto de código lo que se hace es centrar el servomotor poniéndolo a 90 grados que es el centro de un servo de 180 grados y luego se utiliza un bucle para hacer mover el motor lentamente hasta otra posición (170 grados), esto para ver en qué dirección se desplaza.

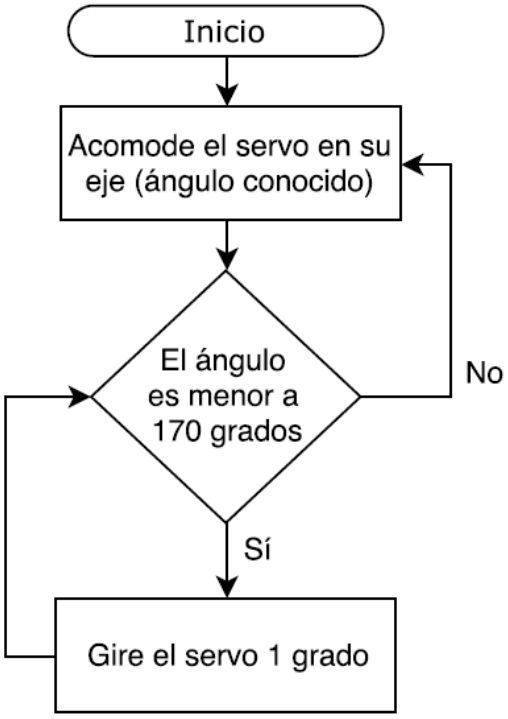


Figura 7: Diagrama de flujo para prueba de servomotores.

8. Parte 3: GiraBot

Uniendo los dos códigos anteriores se debe lograr que el girasol siga a la luz más intensa que detecta dentro de su rango de giro. Un modelo guía para las conexiones se muestra en la figura 5. El código se muestra a continuación.

```
#include <Servo.h>
Servo servo; // creacion del objeto para usar la libreria
const int motor = 6;
int posicion = 90;
int dato_Izquierda = 0; // dato sensor 1
```

```
int dato_Derecha = 0; // dato sensor 2
const int sensor1 = A1; // Sensor Izquierda
const int sensor2 = A2; // Sensor Derecha
void setup() { // Configuraciones
servo.attach(motor); // motor --> pin 6
servo.write(posicion); // centrar el motor (90*)
pinMode(sensor1,INPUT); // sensor --> entrada
pinMode(sensor2,INPUT); // sensor --> entrada
Serial.begin(9600); // iniciar el puerto Serial
}
void loop() {
Serial.print("Posicion: "); Serial.print(posicion); Serial.println("*")
   ;
dato_Izquierda = analogRead(sensor2); // guardo la lectura
dato_Derecha = analogRead(sensor1);
Serial.println(dato_Izquierda);
Serial.println(dato_Derecha);
   if( dato_Izquierda > dato_Derecha){
      if(posicion >= 10){
         posicion = posicion - 10; // Girar sentido antihorario
       }
   }
   if( dato_Izquierda < dato_Derecha){
      if(posicion <= 170){
         posicion = posicion + 10; // Girar sentido horario
      }
   }
servo.write(posicion);
delay(10);
}
```

Este código que es el que rige el movimiento del girasol lo que hace es hacer uso de dos condicionales en los cuales se pueda saber si la lectura del fotoresistor derecho es menor o mayo que la del izquierdo, una lectura mayor de este implicaría que la luz incide en mayor medida sobre el otro fotoresistor y por lo tanto debe girar en la dirección contraria a la del de mayor valor, para que siga la fuente luminosa, este cambio lo hace de 10 en 10 grados, para que no sea muy brusco, se puede cambiar para hacer el movimiento más suave, este sigue cambiando la posición porque el void loop se mantiene en constante repetición. Los condicionales internos de mayor o menor a un valor son para limitar el movimiento del servo en su rango de operación.

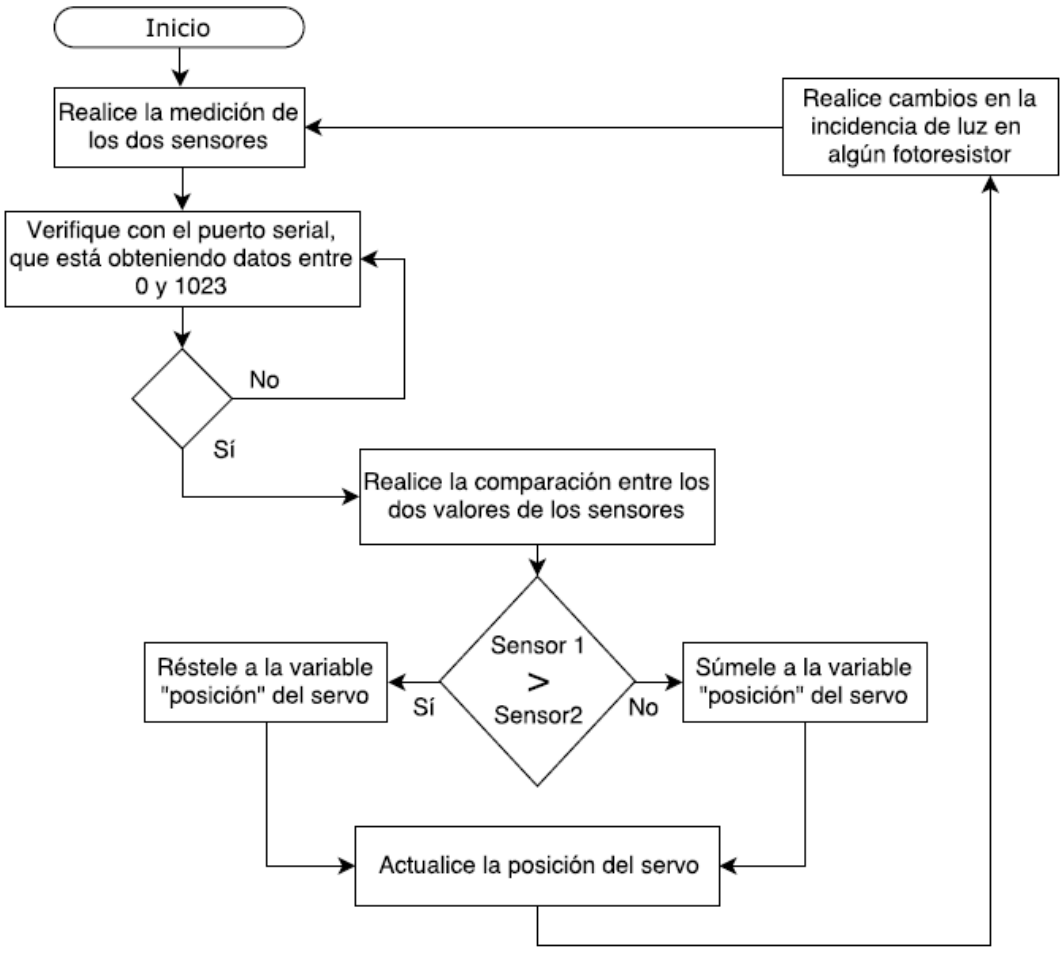


Figura 8: Diagrama de flujo completo del GiraBot.

9. Descripción de instrucciones importantes utilizadas

1. Tener un valor fijo durante todo el código. Puede ser de distintos tipos, no solo entero (*int*). Es muy útil cuando se usa un mismo valor en varias partes del código y en algún momento se quiere cambiar su valor, entonces basta con cambiarlo en una línea y no en cada lugar que se usó.

```
1    const int nombre // variable que guarda un dato que no va a
       cambiar.
```

2. Abrir el puerto serial y establecer la tasa de datos en 9600 bps.

```
1    Serial.begin(9600);
```

3. Leer un valor analógico en un rango de datos entre 0 y 1023.

```
1    dato1 = analogRead(sensor1);
```

4. Imprimir el valor que toma una variable.

```
1     Serial.print("Sensor␣1␣---->␣"); Serial.print(dato1);
```

5. Biblioteca para usar el servo y creación de un objeto tipo Servo.

```
1     #include <Servo.h>
2     Servo servo;
```

6. Usar los métodos *attach()* y *write()* de la clase servo. El primero para asignar el pin donde está conectado el servo y el segundo para darle una posición al servo.

```
1     servo.attach(motor); // motor --> pin 6 servo.write(90); //
         centrar el motor
```

7. Crear una función que no retorna ningún valor *void* (vacía) y sin ningún argumento.

```
1     void explorar(){
2     //** Instrucciones **//
3     }
```

8. Ciclo *for*. Para repetir una instrucción un determinado número de veces.

```
1     for(int i=0; i< 10; i++){
2     //** Instrucciones **//
3     }
```