



PONTIFICIA
UNIVERSIDAD
CATÓLICA DE
VALPARAÍSO

EIE ___-__ Robótica e inteligencia artificial

pucv.cl

Módulo 3 Programación y lógica de funcionamiento S15

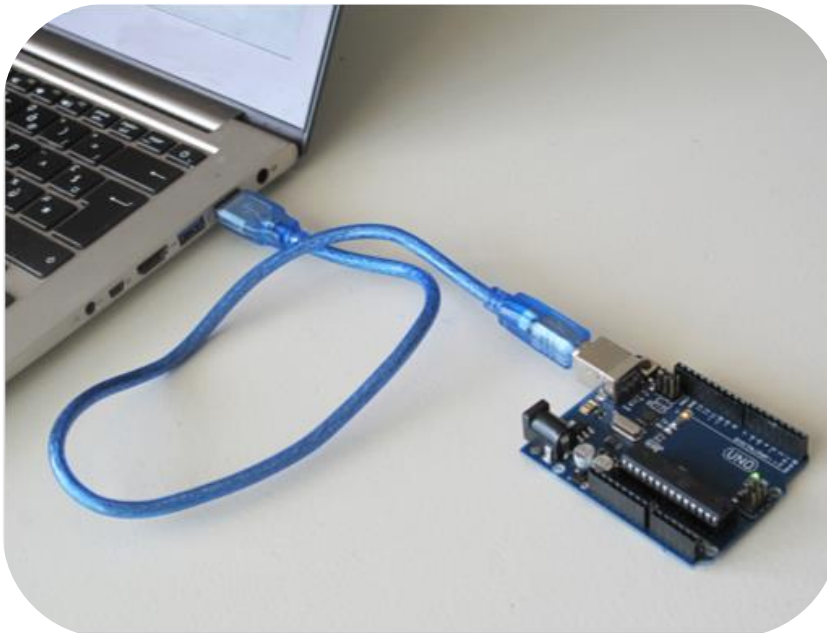
Valparaíso, 2022

PROGRAMACIÓN Y LÓGICA DE FUNCIONAMIENTO SESIÓN 15

Programación y lógica de funcionamiento

Las unidades de procesos centrales más utilizadas en robótica educativa y de investigación suelen ser altamente versátiles en tanto a la capacidad y compatibilidad de creación de distintos circuitos que den uso a distintos sensores y actuadores, permitiendo confeccionar sistemas robóticos que permitan pruebas y desarrollos.

La aplicación de robótica industrial suele utilizarse sistemas de control proporcionados por el fabricante del robot, donde se asegure un correcto funcionamiento y gran fiabilidad, dejando al operador del robot fuera del aborto unidad de programar al detalle las acciones de la máquina.

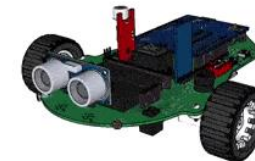
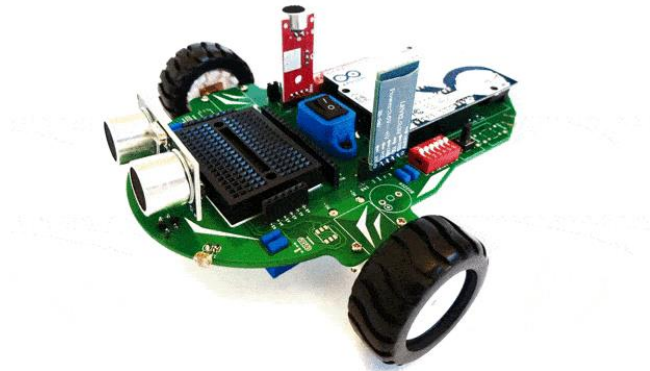


Programación y lógica de funcionamiento

Realidad y simulación

Como se ha mencionado anteriormente, la simulación permite una serie de ventajas sobre la implementación real, pero debe existir una relación fuertemente basada respecto a los sistemas simulados con el funcionamiento real de un sistema robótico. En efecto, gran cantidad de simulaciones se basa en un sistema robótico ya confeccionado.

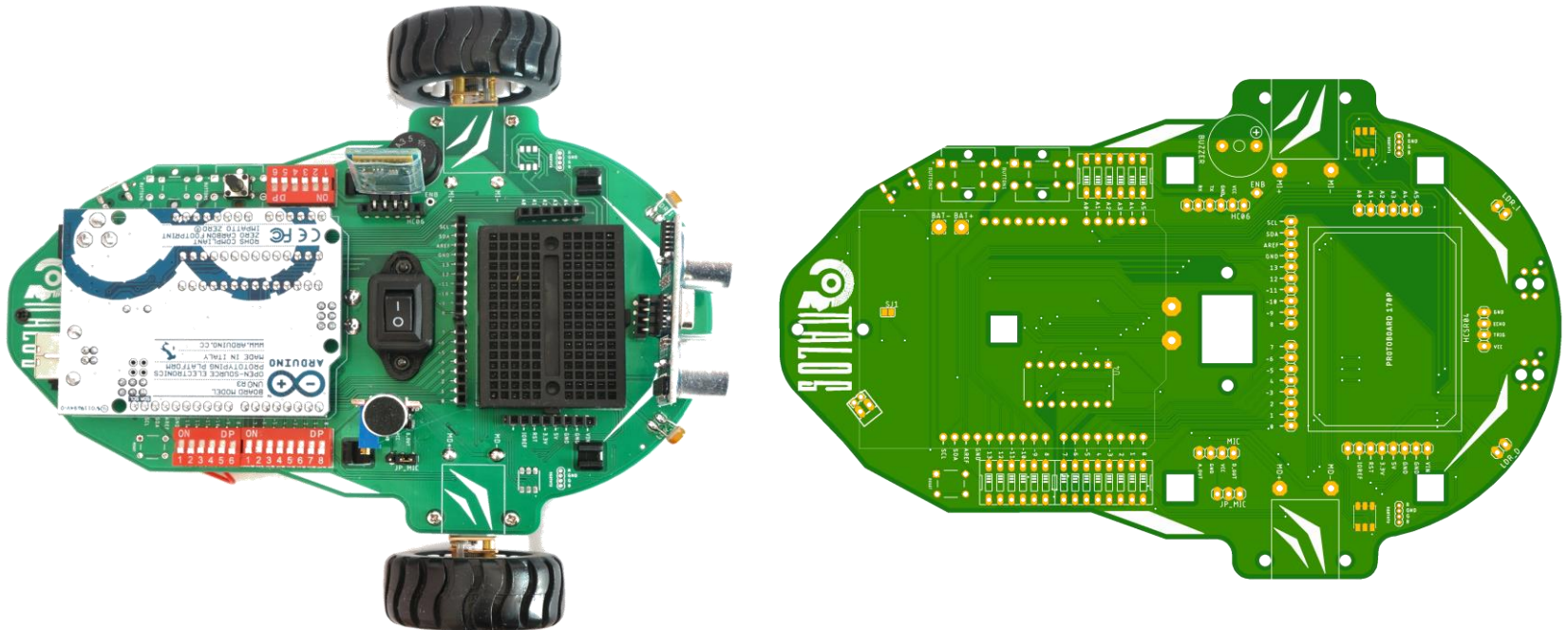
En este caso se analizará un pequeño robot diferencial de 2 ruedas con enfoque educativo, el cual posee CPU, Arduino UNO, actuadores y sensores que permiten programar una gran cantidad de pruebas de funcionamiento.



Programación y lógica de funcionamiento

Talos UNO

El robot Talos UNO utiliza una tarjeta Arduino UNO como CPU, además de sensores y actuadores que permiten programar algoritmos de prueba e investigación. El robot se considera una Shiel para Arduino, ya que la PCB que funciona como base del robot puse todas las conexiones necesarias para los sensores y el funcionamiento de los dos motores.

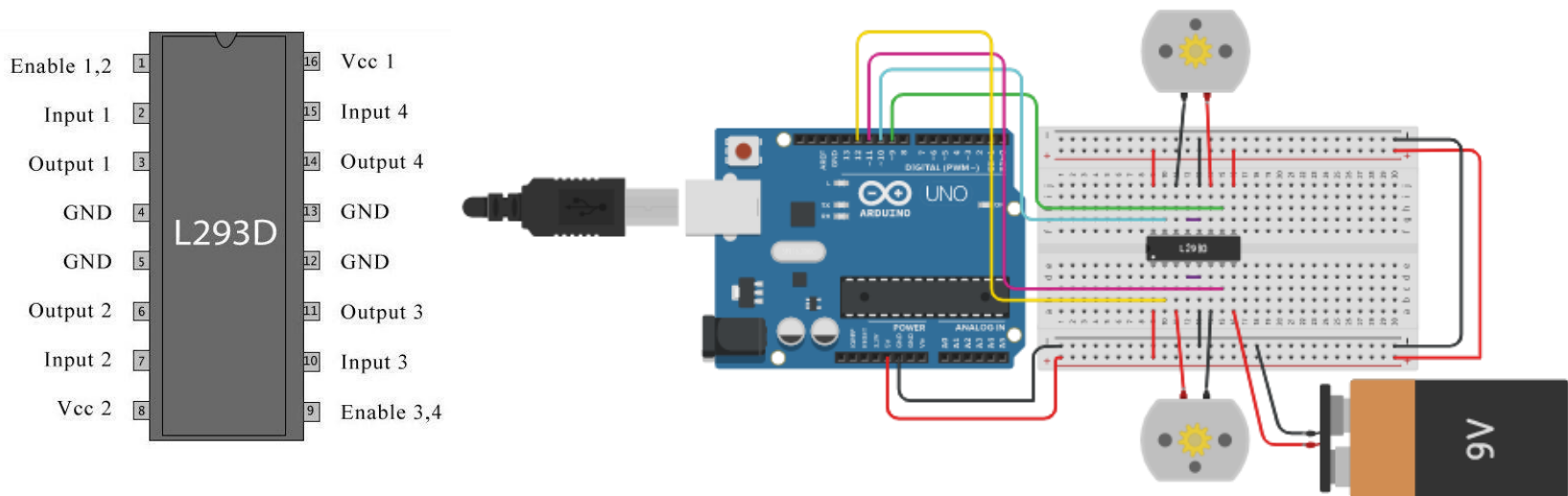


Programación y lógica de funcionamiento

Talos UNO

El robot hace uso de un puente h que permite controlar los dos actuadores que le dan movilidad al sistema, permitiendo programar las potencias de estos además de los sentidos de giros.

La programación del puente H requiere 6 conexiones diseñar con el Arduino, las cuales permiten activar o desactivar los transistores que componen el circuito integrado.

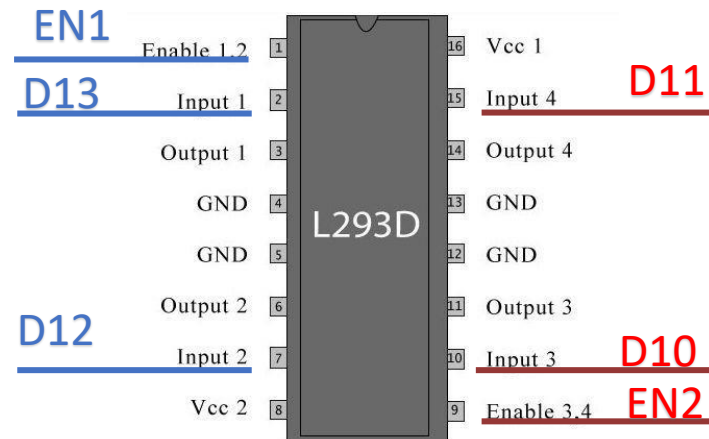


Programación y lógica de funcionamiento

Talos UNO

La programación en el IDE de Arduino no es compleja, se debe indicar que transistores se activarán para generar el movimiento deseado, además de la indicación de potencia de la acción.

```
void loop() {  
  
  analogWrite(EN1, 255); //potencia M1  
  digitalWrite(13, HIGH); //config sentido 1  
  digitalWrite(12, LOW);  
  analogWrite(EN2, 255); //potencia M2  
  digitalWrite(11, HIGH); //config sentido 1  
  digitalWrite(10, LOW);  
  delay(1000); //duración de la acción  
  
  analogWrite(EN1, 0); //potencia M1  
  digitalWrite(13, HIGH); //config sentido 2  
  digitalWrite(12, LOW);  
  analogWrite(EN2, 0); //potencia M2  
  digitalWrite(11, HIGH); //config sentido 1  
  digitalWrite(10, LOW);  
  delay(1000); //duración de la acción  
  
}
```



Este pequeño programa de ejemplo muestra al robot avanzando con potencia igual en ambos motores durante un segundo y luego deteniéndose (potencia cero) durante otro segundo, finalmente el algoritmo se reinicia por la función Loop.

Programación y lógica de funcionamiento

Talos UNO

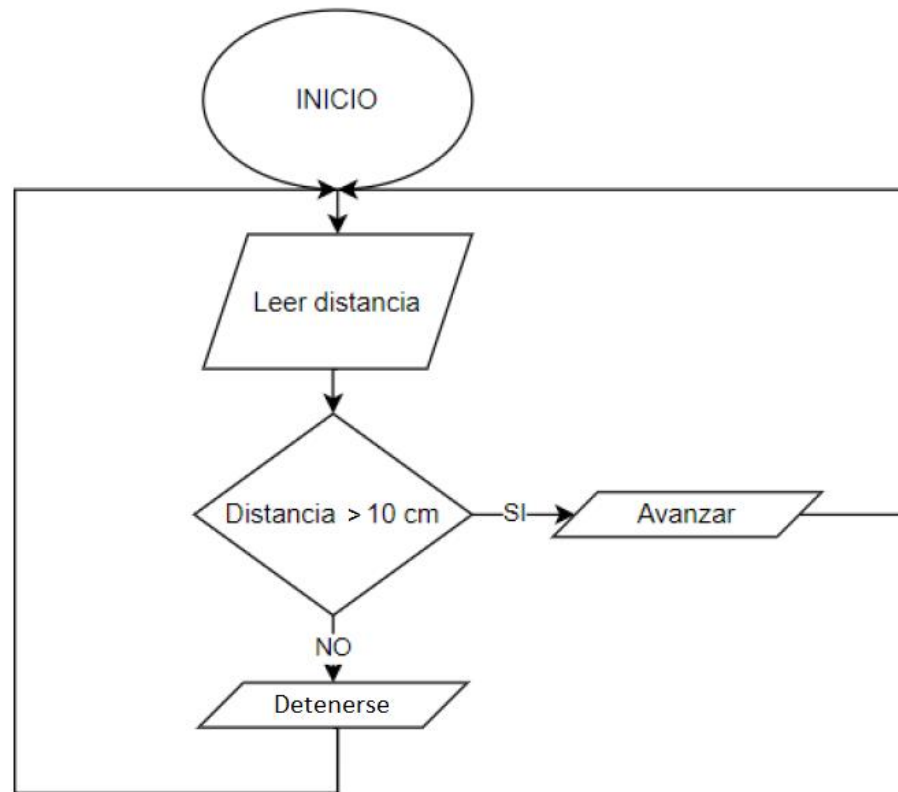
Si se desea implementar un algoritmo que lea el sensor de distancia ultrasónico para tomar la decisión de detenerse, se debe implementar el siguiente código.

```
void loop() {  
  
  if (ultrasonido > 10) {  
    analogWrite(EN1, 255); //potencia M1  
    digitalWrite(13, HIGH); //config sentido 1  
    digitalWrite(12, LOW);  
    analogWrite(EN2, 255); //potencia M2  
    digitalWrite(11, HIGH); //config sentido 1  
    digitalWrite(10, LOW);  
    delay(1000); //duración de la acción  
  }  
  else {  
    analogWrite(EN1, 0); //potencia M1  
    digitalWrite(13, HIGH); //config sentido 2  
    digitalWrite(12, LOW);  
    analogWrite(EN2, 0); //potencia M2  
    digitalWrite(11, HIGH); //config sentido 1  
    digitalWrite(10, LOW);  
    delay(1000); //duración de la acción  
  }  
}
```


Programación y lógica de funcionamiento

Talos UNO

El diagrama de flujo del algoritmo implementado es el siguiente.

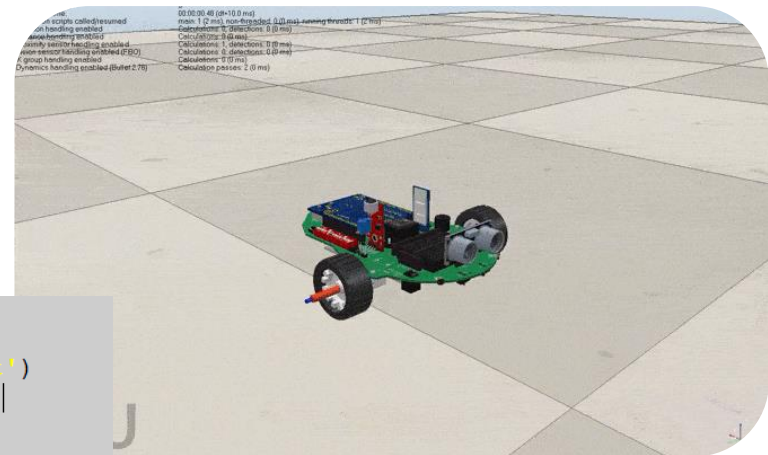


Programación y lógica de funcionamiento

Talos UNO

Al migrar al simulador CoppeliaSim, es posible simular el robot, considerando que el algoritmo implementado anteriormente debe ser programado con la estructura compatible del simulador (LUA).

```
function coroutineMain()  
  
    motorRight = sim.getObjectHandle('Right_joint')  
    motorLeft = sim.getObjectHandle('Left_joint')  
    sensor=sim.getObjectHandle("HCSR04")  
  
    while true do  
  
        sim.setJointTargetVelocity(motorRight,0) --stop  
        sim.setJointTargetVelocity(motorLeft,0)  
        sim.wait(1);  
  
    end  
end
```

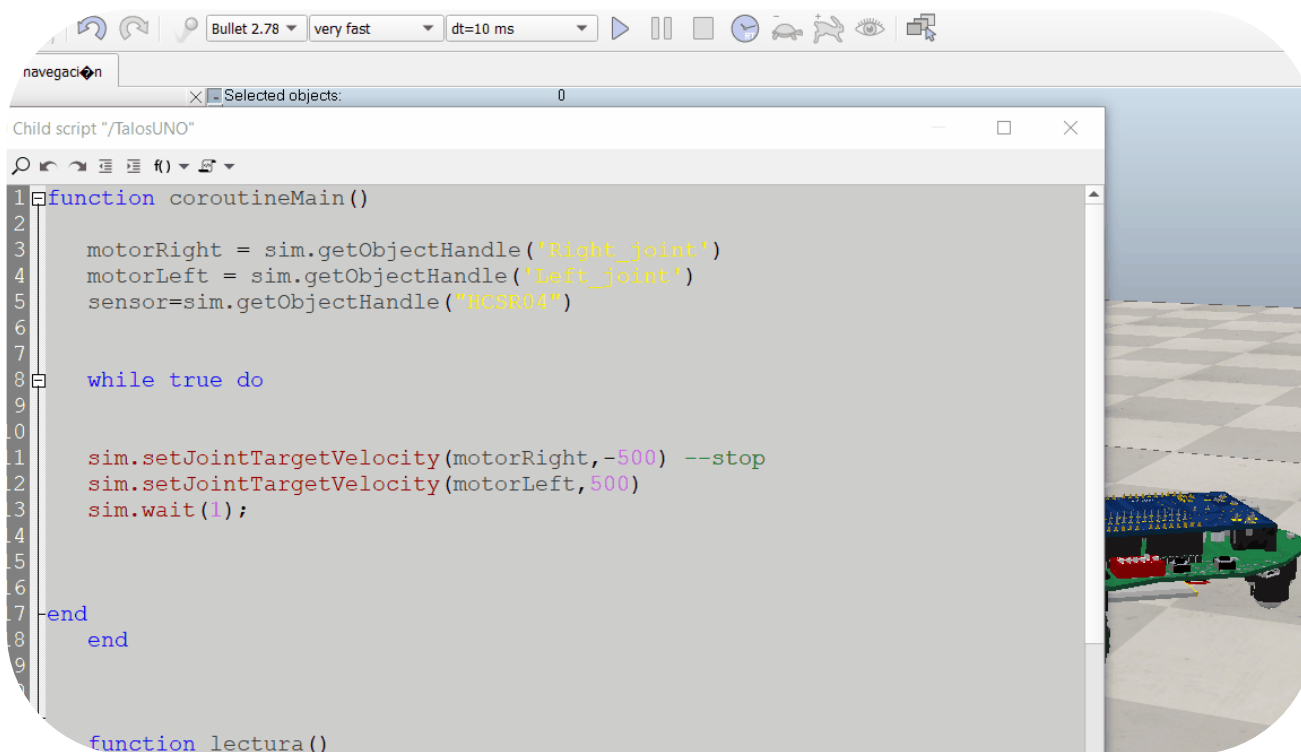


Programación y lógica de funcionamiento

Talos UNO

Notar que el diagrama de flujo sigue siendo el mismo, pero la implementación del programa ha cambiado, debido a que en este caso no existe una comunicación directa con la CPU del robot.

Incluso es posible indicar parámetros que el robot no podrá alcanzar realmente, como una potencia exageradamente alta.



The screenshot shows a simulation window with a toolbar at the top containing icons for navigation and simulation control. Below the toolbar, a status bar indicates 'Bullet 2.78', 'very fast', and 'dt=10 ms'. A 'navegación' window shows 'Selected objects: 0'. The main area displays a 3D model of a Talos UNO robot on a checkered floor. Overlaid on this is a 'Child script "/>

```

1 function coroutineMain()
2
3     motorRight = sim.getObjectHandle('Right_joint')
4     motorLeft = sim.getObjectHandle('Left_joint')
5     sensor=sim.getObjectHandle("HCSR04")
6
7
8     while true do
9
10
11         sim.setJointTargetVelocity(motorRight,-500) --stop
12         sim.setJointTargetVelocity(motorLeft,500)
13         sim.wait(1);
14
15     end
16 end
17
18 function lectura()

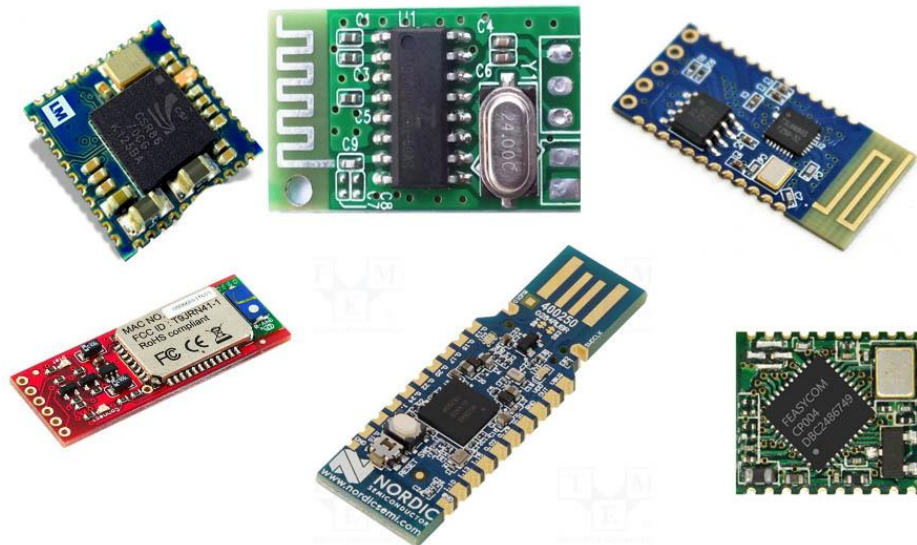
```

Programación y lógica de funcionamiento

Desventajas

Algunas plataformas de desarrollo son tan básicas que presentan desventajas cuando se busca implementar alguna aplicación un tanto más compleja. En el caso del mencionado Arduino, el microcontrolador no cuenta con una capacidad de procesamiento elevado, por lo que no es posible implementar sistemas de reconocimiento de imágenes ni una transmisión de datos con un ancho de banda adecuado.

Si bien existen módulos que permiten la conexión de este dispositivo con protocolos wifi, Bluetooth, red, LoRa, Xbee, entre otros, los datos transmitidos seguirán siendo paquetes básicos y sencillos.



Programación y lógica de funcionamiento

Desventajas

algunos sistemas robóticos utilizan Arduino para ejecutar el control de los actuadores y sensores involucrados en la estructura del robot, transmitiendo estos datos en tiempo real a un computador como controlador principal.

En este caso se debe identificar que la unidad de procesos centrales del robot se centra en el computador de control, y la plataforma de desarrollo a bordo solo se considera una etapa de comunicación.



Programación y lógica de funcionamiento

Desventajas

La consideración anterior no suele ser siempre una desventaja, ya que existen aplicaciones que requieren un control elaborado en una estación de monitoreo que se encarga de más de un solo equipo robótico, como lo es en el caso industrial.

Algunos robots industriales cuentan con una dirección directa a un computador de control y en su estructura de funcionamiento no cuentan con una CPU principal, más bien es una etapa de comunicación entre los controles de los actuadores y sensores, y la estación de control principal.



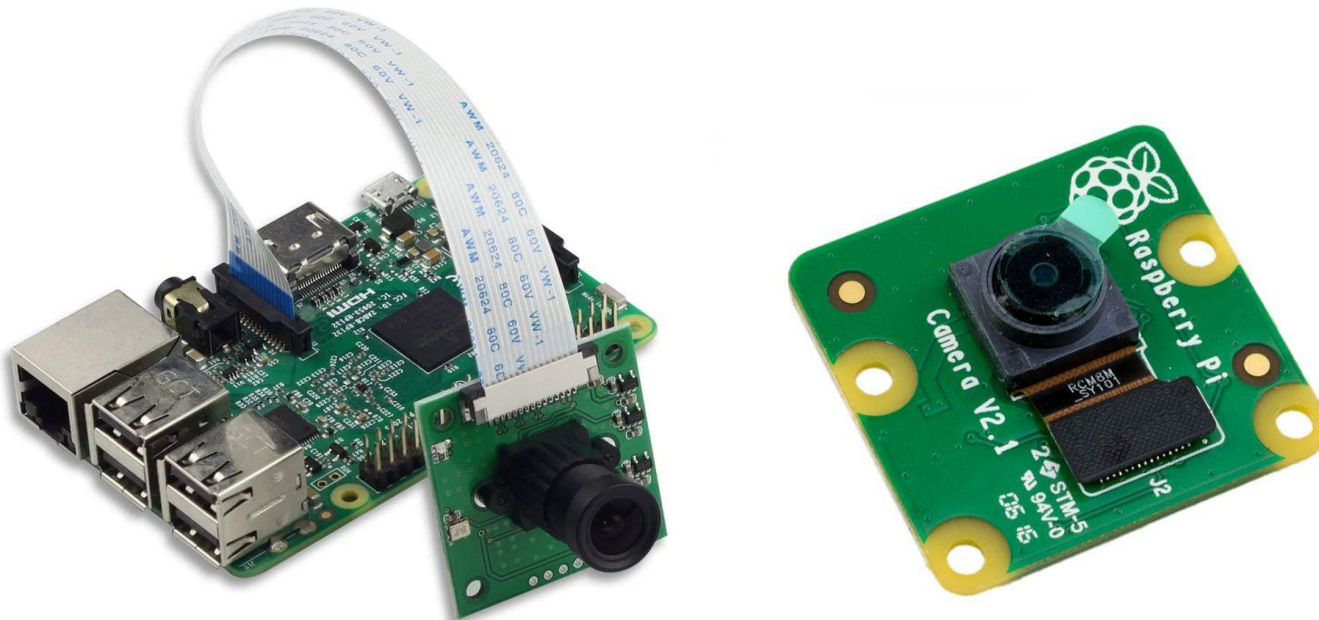
Programación y lógica de funcionamiento

Raspberry

La plataforma raspberry presenta una gran cantidad de ventajas en robots con otras aplicaciones, ya que esta compone un computador capaz de soportar un sistema operativo, dentro de este sistema es posible instalar software y herramientas útiles para ciertas aplicaciones.

Además, como plataforma posee mucha más capacidad de procesamiento, permitiendo trabajar con distintos tipos de archivos, conexiones, y anchos de banda elevados.

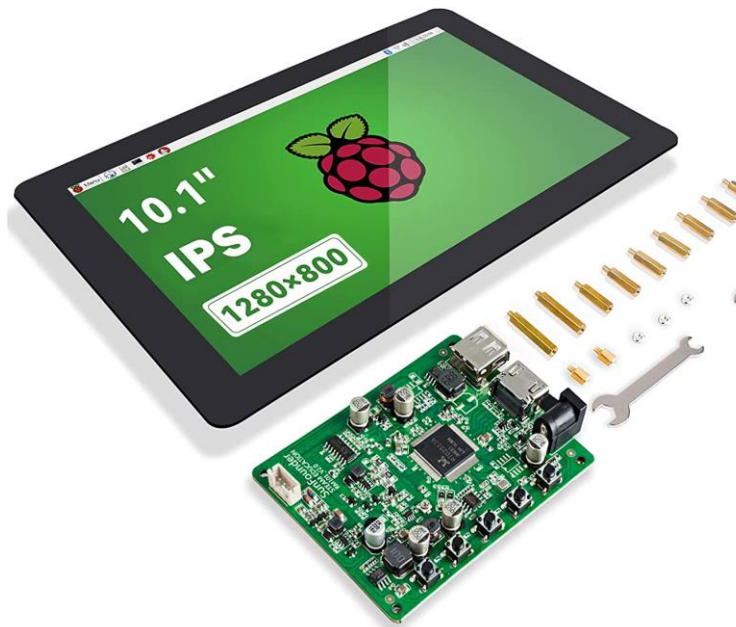
Es posible encontrar cámaras para conectar a la tarjeta y aplicar algoritmos de visión artificial, expandiendo las capacidades del sistema robótico



Programación y lógica de funcionamiento

Raspberry

Raspberry tiene la capacidad de implementar una interfaz gráfica que permita interactuar con un usuario o público objetivo, de esta manera si se requiere crear un robot asistente, a través de una raspberry y una pantalla Touch será posible mostrar imágenes, vídeos, y contenido multimedia para entregar información dentro del proceso de interacción.



Programación y lógica de funcionamiento

Compatibilidad

Se sabe que es comúnmente utilizado el circuito puente H para el control de motores, mientras que otros actuadores como lo son los servomotores ya poseen un driver interno que permite controlar su potencia, sentido de giro, y el ángulo a alcanzar a través de un tipo de señal PWM.

Para el caso de sensores, se depende de que tan elaborado es el funcionamiento electrónico del sensor, es decir, qué tipo de señal puede entregar a la CPU después de una lectura.

Sensores de funcionamiento básico



Sensores de funcionamiento avanzado



Programación y lógica de funcionamiento

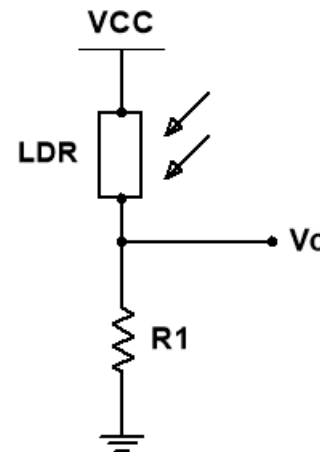
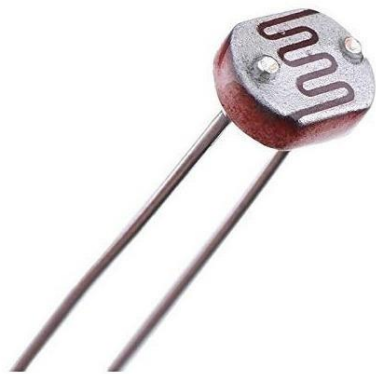
Compatibilidad

Para el caso de los sensores de luz (LDR), los cuales tienen estrías de cadmio dónde este es susceptible a la luz cambiando la resistencia eléctrica del material, es posible realizar una interpretación análoga de la cantidad de luz presente donde se realiza la medición.

Para esto es importante confeccionar un circuito que permita realizar esta interpretación.

El circuito comúnmente utilizado para los LDR es un sencillo divisor de tensión, donde si el sensor cambia su resistencia tendrá una mayor o menor caída de tensión, lo que será detectado por un puerto análogo de lectura (ADC).

La medición recibida es un nivel de voltaje, el cual no posee una unidad de medida de luz, por lo tanto se considera que no está interpretada.

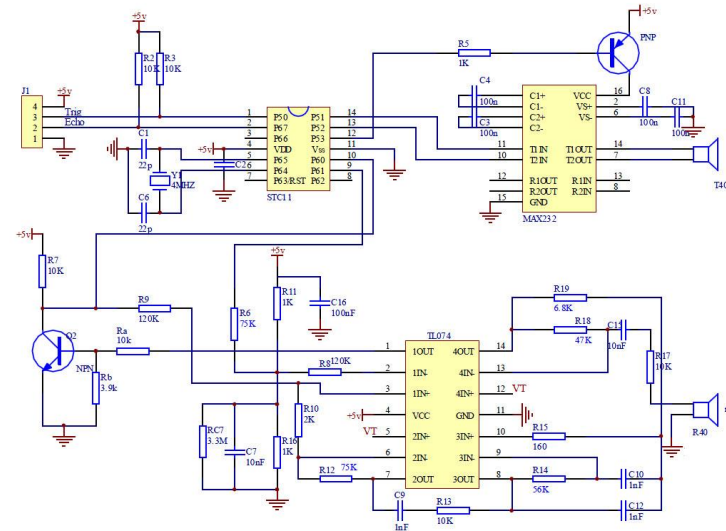
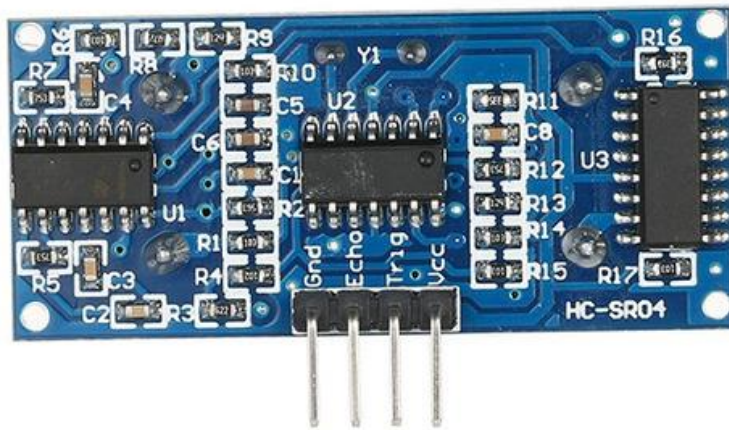


Programación y lógica de funcionamiento

Compatibilidad

En el caso del sensor ultrasonido, es posible apreciar que cuenta con una PCB que involucra circuitos integrados, encargados de realizar la medición de tiempo entre la emisión del frente ultrasonido y la recepción del eco, interpretando una distancia a través de la demora de tiempo del eco recibido.

Este tipo de sensor entrega una señal tipo digital que ya se encuentra interpretada, es decir, que la señal de distancia ya cuenta con una unidad de medida acorde.



Programación y lógica de funcionamiento

Compatibilidad

De esta manera es posible encontrar una gran gama de sensores resistivos que funciona con el mismo principio del LDR. También es posible encontrar sensores elaborados que entregan una señal interpretada sin la necesidad de confeccionar circuitos ni realizar mediciones de voltaje.



Sensor de temperatura
LM35



Sensor de pH

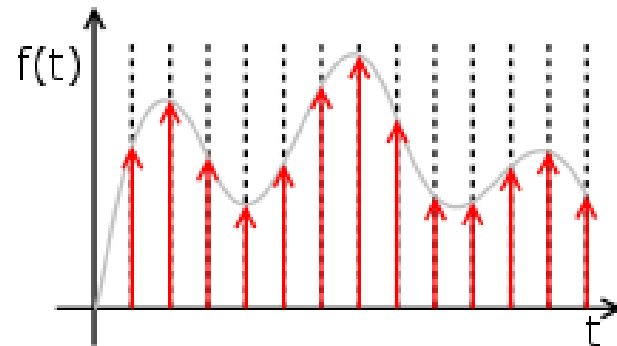
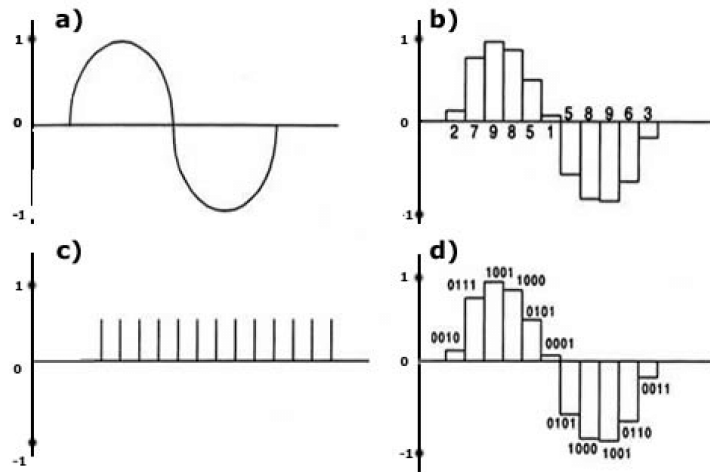
Programación y lógica de funcionamiento

Muestreo de señales análogas

Las señales análogas corresponden a niveles de voltaje análogos a la variable física medida, es decir, si se mide intensidad de luz, es posible observar un cambio de voltaje con la misma forma que el cambio de intensidad de luz.

Para que la medición del sensor sea precisa, se identifica el concepto de muestreo.

El muestreo de una señal Consiste en la selección de ciertos valores de una señal análoga continua para obtener una discreta. Mientras estos valores se encuentren más próximos entre ellos, la medición será más cercana a la variable real



Programación y lógica de funcionamiento

Muestreo de señales análogas

Algunas plataformas de desarrollo poseen frecuencias de muestreo Más elevadas que otras, además de una resolución mayor.

Por ejemplo, arduino UNO presenta una resolución de muestreo de 10 bits, esto equivale a 1024 valores entre 0 V y 5 V (voltaje de operación).

Por otro lado Arduino DUE presenta una resolución de muestreo de 12 bits, esto equivale a 4096 valores entre 0 V y 3 V (voltaje de operación)

En comparación, Arduino DUE presenta una mayor ventaja respecto a la medición de sensores análogos, considerando que su voltaje de operación es de 3 V.

