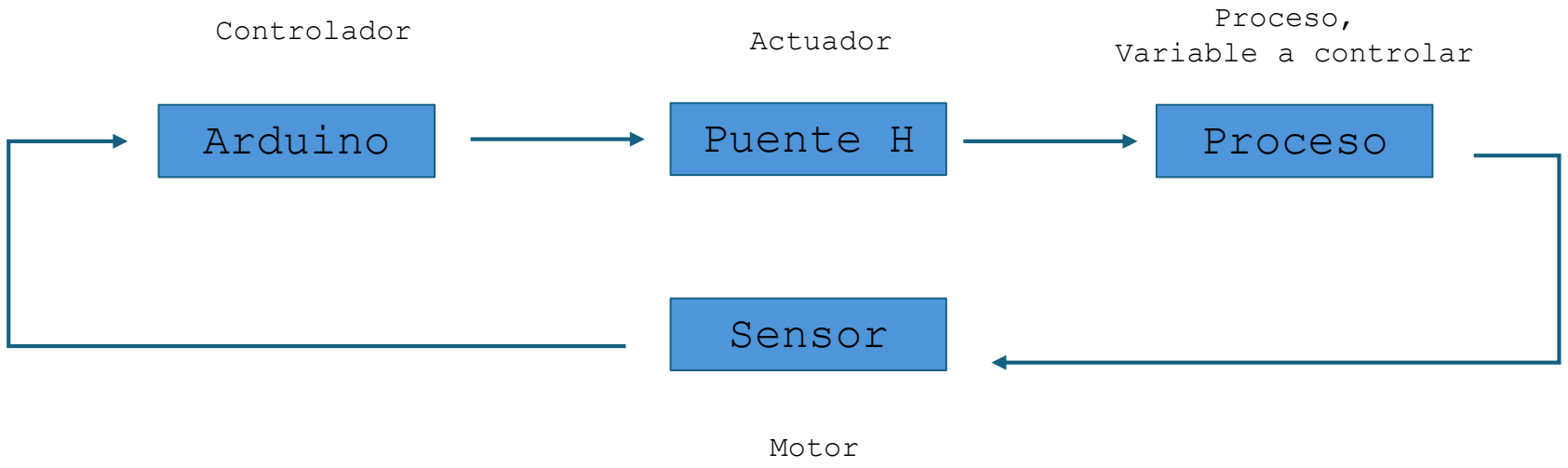
Taller de Programación I

SASIM

Mes de la robótica

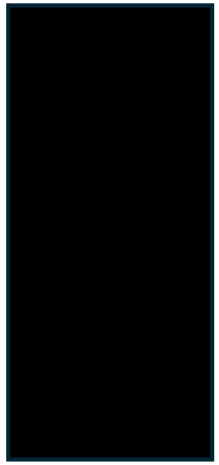
Entendiendo el sistema

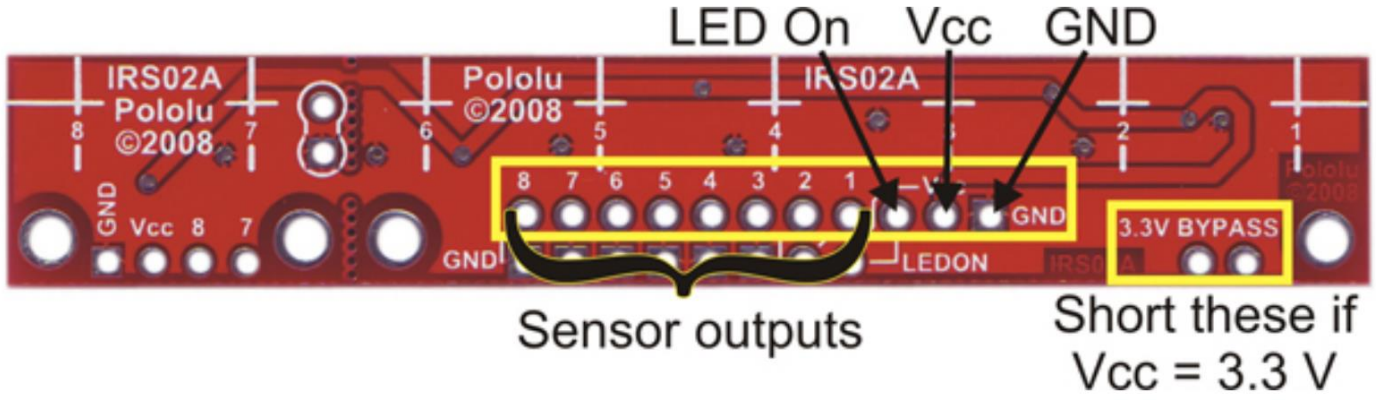
- El sistema cuenta con entradas (lecturas analógicas del sensor).
- Se procesa la señal.
- Se corrige la posición en función del error.



Sensor QTR-8 A

- El sensor entrega valores entre 0 y 1023.
- Líneas blancas tienen valores cercanos a 0 y oscuras a 1023.
- Son 8 sensores, los de la esquina son sensibles a los cambios en curvas.



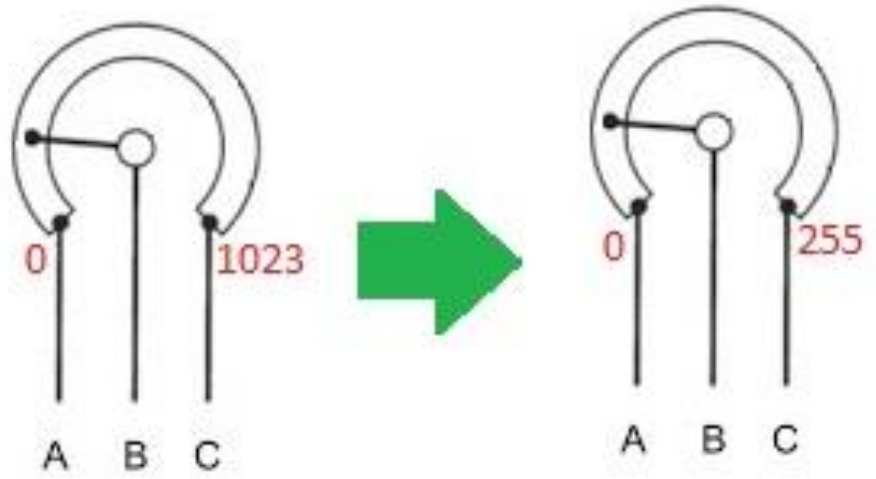


QTR-8x reflectance sensor array with 11×1 connection pins labeled.

Calibración de sensor

- En la práctica el sensor raramente dará valores sin error.
- Lo que se necesita es calibrar en función del mínimo y máximo.
- Utilizaremos la función map (es decir buscaremos los valores mínimos reales y máximos reales para hacer un rango entre estos).

```
//Nivel global antes de setup

int sensorPins[8] = {A0, A1, A2, A3, A4,
A5, A6, A7};

int sensorMin[8];
int sensorMax[8];
```



Calibración de sensor

- Dentro de nuestro setup, llenaremos los valores de los arreglos.
- Utilizaremos un ciclo for con el fin de llenar los arreglos.
- Posterior a eso vamos a imprimir los valores con el fin de observar los mínimos y máximos del sensor.

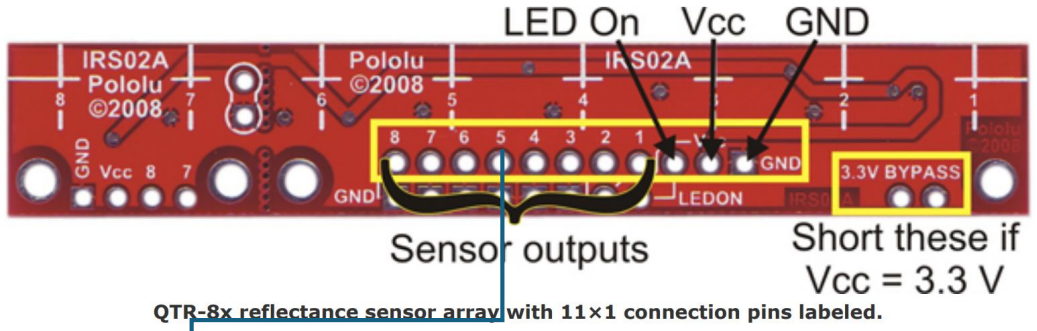
```
void setup() {
  Serial.begin(9600);

for (int i = 0; i < 8; i++) {
    pinMode(sensorPins[i], INPUT);
    sensorMin[i] = 1023; // Valor máximo posible al inicio
    sensorMax[i] = 0;    // Valor mínimo posible al inicio
  }
```

```
Serial.println("=== CALIBRANDO SENSORES ===");
  delay(2000);
  unsigned long tInicio = millis();
  while (millis() - tInicio < 5000) {
    for (int i = 0; i < 8; i++) {
      int lectura = analogRead(sensorPins[i]);
      if (lectura < sensorMin[i]) sensorMin[i] = lectura;
      if (lectura > sensorMax[i]) sensorMax[i] = lectura;
    }
    delay(5);
  }

for (int i = 0; i < 8; i++) {
    Serial.print("Sensor "); Serial.print(i);
    Serial.print(" - Min: "); Serial.print(sensorMin[i]);
    Serial.print(" | Max: "); Serial.println(sensorMax[i]);
  }
}
```

Calculo de posición de línea

- Para la posición primero normalizaremos los valores.
- Después de esto asignaremos pesos equivalentes (asumir que no todos los sensores tienen misma importancia).
- Realizaremos una ponderación que nos ayudará a detectar lo alejado que estamos del objeto.

```
int calcularPosicionLinea() {
  long sumaPonderada = 0;
  int sumaTotal = 0;

  for (int i = 0; i < 8; i++) {
    int lectura = analogRead(sensorPins[i]);

    // Normalizar valor entre 0 y 1000
    int normalizado = map(lectura, sensorMin[i], sensorMax[i], 1000, 0);
    normalizado = constrain(normalizado, 0, 1000);

    sumaPonderada += (long)normalizado * (i * 1000);
    sumaTotal += normalizado;
  }

  if (sumaTotal == 0) return setPoint; // No hay línea detectada

  return sumaPonderada / sumaTotal; // Devuelve la posición ponderada
}
```



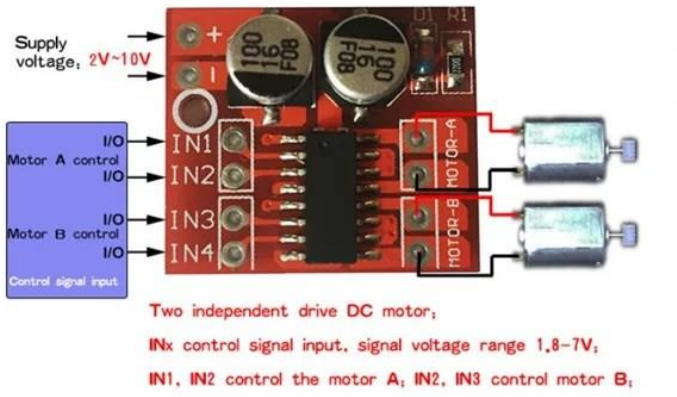
Posición 0, se toma como inicio

Puente H y control de motores

- El puente H es un driver utilizado para controlar dirección y velocidades de un motor de CD.
- Este servirá para mantener un control de la velocidad con el controlador.
- Utilizaremos voltaje PWM.

Nivel global

```
// Pines del motor izquierdo
const int ENIzq = 5;  // PWM para velocidad
const int IN1 = 7;    // Dirección
const int IN2 = 8;    // Dirección

// Pines del motor derecho
const int ENDer = 6;  // PWM para velocidad
const int IN3 = 9;    // Dirección
const int IN4 = 10;   // Dirección
```



Setup

```
pinMode(ENIzq, OUTPUT); pinMode(IN1, OUTPUT); pinMode(IN2,
OUTPUT);
pinMode(ENDer, OUTPUT); pinMode(IN3, OUTPUT); pinMode(IN4,
OUTPUT);

  digitalWrite(IN1, HIGH); digitalWrite(IN2, LOW);
  digitalWrite(IN3, HIGH); digitalWrite(IN4, LOW);
```

¿Dudas?