Sistema de riego automático ARDUINO

(en tinkercad)

Participantes:
- Gutierrez Santino
- Martinez Lorenzo

Comisión:
TUP8 - Mañana

Informe:

Informe: Proyecto de Sistema de Riego Automático con Arduino

El proyecto consiste en la creación de un sistema de riego automático utilizando un Arduino y varios sensores para monitorear y controlar las condiciones ambientales que afectan el riego de las plantas. El sistema se encarga de medir la humedad del suelo, la intensidad de la luz, y la temperatura para automatizar el riego de las plantas.

Componentes utilizados:

1. Arduino Uno: Controla todo el sistema y recibe datos de los sensores para tomar decisiones.
2. Sensor de humedad del suelo (SOIL_MOISTURE_SENSOR) : Mide la humedad del suelo para determinar si las plantas necesitan agua.
3. Sensor de luz (LIGHT_SENSOR) : Permite detectar la intensidad de la luz ambiental para ajustar la actividad del sistema.
4. Sensor de temperatura (TMP_PIN) : Ayuda a monitorear la temperatura del entorno donde se encuentran las plantas.
5. Motor de riego (MOTOR_PIN) : Se encarga de administrar el flujo de agua para regar las plantas.
6. Servo motor (SERVO_PIN): Controla la apertura y cierre de la válvula de riego.
7. Pantalla LCD (conexión mediante LiquidCrystal) : Muestra datos relevantes del entorno y del sistema para el usuario.
8. LED (ledd_PIN) : Indica el estado del sensor de luz.

Funcionamiento:

El programa implementado en el Arduino tiene un bucle principal que se repite continuamente, realizando las siguientes tareas:

1. Lectura de los valores de los sensores de humedad del suelo, luz y temperatura.
2. Visualización de los datos en la pantalla LCD, mostrando la temperatura y la humedad del suelo.

3. Control del riego: Si la humedad del suelo es menor a un valor específico (en este caso, 500), el servomotor abre la válvula de riego durante un tiempo determinado (2 segundos), y luego la cierra.
4. Control de la iluminación: Si la intensidad de la luz es baja (menor a 5 en la escala), se enciende un LED. En caso contrario, el LED se apaga.
5. Control de la temperatura: Si la temperatura supera los 30 grados Celsius, se activa el motor de riego durante 2 segundos para refrescar las plantas.

Mejoras y Consideraciones:

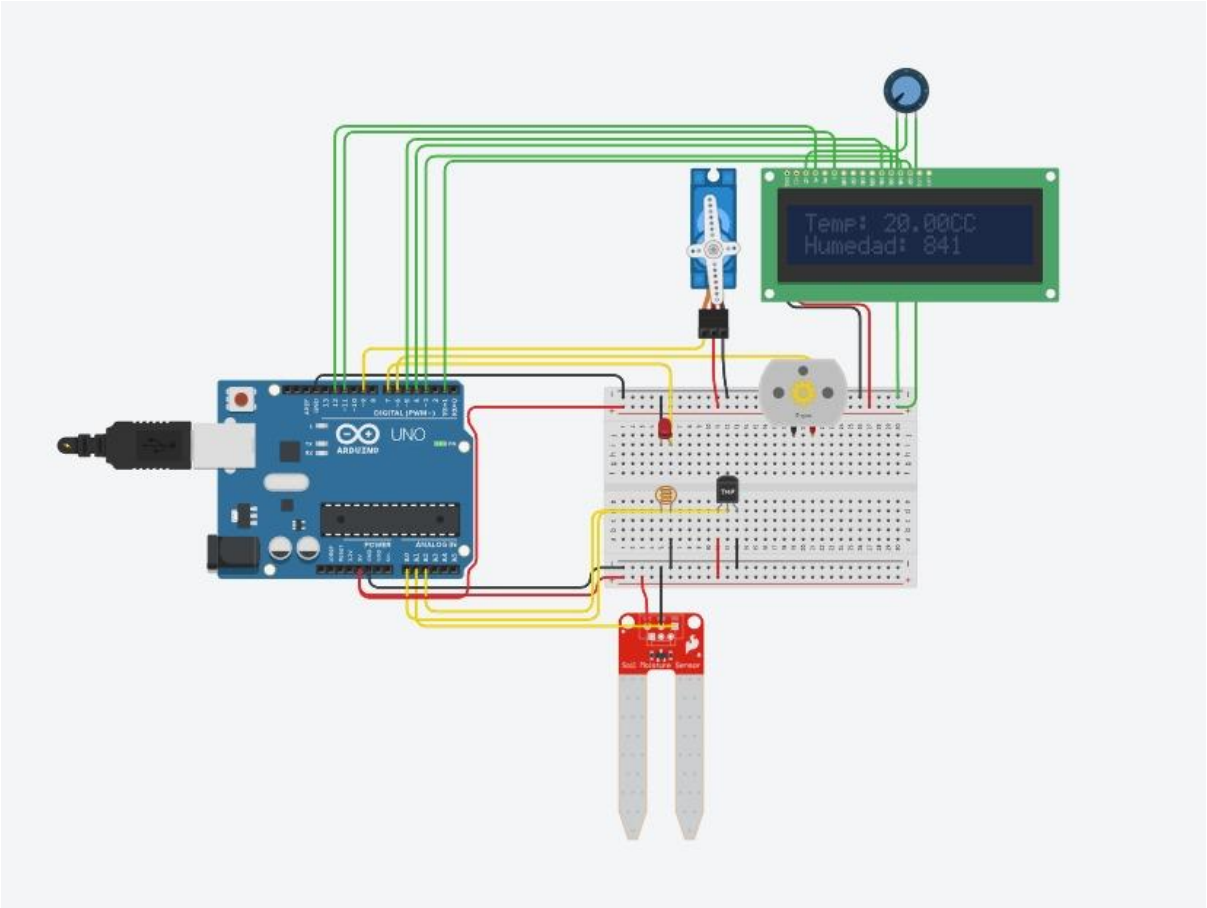
1. Calibración de sensores: Es crucial asegurarse de que los sensores estén correctamente calibrados para obtener lecturas precisas. Esto puede requerir ajustes en el código y pruebas en el entorno real.
2. Ajuste de umbrales: Los valores umbral para la humedad del suelo, la luz y la temperatura son arbitrarios en este código. Deben ajustarse según las necesidades específicas de las plantas que se están regando.
3. Optimización del riego: Se podría implementar un sistema más sofisticado que controle la duración del riego en función de la humedad del suelo y las condiciones climáticas.
4. Seguridad: Asegurarse de que el sistema sea seguro y no cause daños a las plantas o a la infraestructura. Por ejemplo, implementar medidas para evitar el exceso de riego.

Este proyecto proporciona una base sólida para un sistema de riego automático, pero puede ser expandido y mejorado para adaptarse a distintos entornos y necesidades de plantas específicas.

Código C++ (Tinkercad) :

```
#include <Servo.h>
#include <LiquidCrystal.h>


#define LIGHT_SENSOR A0
#define SOIL_MOISTURE_SENSOR A1
#define MOTOR_PIN 6
#define SERVO_PIN 9
#define TMP_PIN A2
#define ledd_PIN 7


Servo servo;
LiquidCrystal lcd(12, 11, 5, 4, 3, 1); // RS, E, D4, D5, D6, D7

void setup() {
  lcd.begin(16, 2);
  servo.attach(SERVO_PIN);
  pinMode(MOTOR_PIN, OUTPUT);

}
```

```
void loop() {
  float temperature = analogRead(A2);
  int soilMoisture = analogRead(SOIL_MOISTURE_SENSOR);
  int lightValue = analogRead(A0);




  lcd.setCursor(0, 0);
  lcd.print("Temp: ");
  lcd.print(temperature);
  lcd.print("C");

  lcd.setCursor(0, 1);
  lcd.print("Humedad: ");
  lcd.print(soilMoisture);

  if (soilMoisture < 500) {
    servo.write(90);
    delay(2000);
    servo.write(0);
  }



if (lightValue < 5) {
    digitalWrite(ledd_PIN, HIGH);
  } else {
          (lightValue > 5);
    digitalWrite(ledd_PIN, LOW);
  }


  if (temperature > 30.00){
   digitalWrite(MOTOR_PIN, HIGH);
    delay(2000);
    digitalWrite(MOTOR_PIN, LOW);

  }


  delay(1000);
}
```



Algo en particular que posee este sistema de riego automático, es que para no excederse con ninguna de las formas de nutrición que las plantas utilizan tenemos un sistema de chequeo constante cada aproximadamente varios segundos para que así el sensor brinde ya sea agua, viento o luz durante un tiempo determinado, luego chequea y si sigue necesitando, se le activará devuelta tanto la canilla, como el motor de viento, como la luz.

Conclusión...

Este proyecto nos llevó un aproximado de 4 tardes repartidas en 2 semanas, con prueba y error, además de unas horas en discord realizando el código para correr el mismo. Nos pareció un proyecto más que interesante y con un buen desafío que es hacer un sistema como corresponde y nos gusto mucho la idea que se explicó anteriormente sobre chequear constantemente los distintos sensores para no excederse en las necesidades de la planta. Es un sistema que tranquilamente podríamos adaptar a la necesidad de la planta que se necesite.

¡¡Esperamos que les haya gustado!!

PD:

- *Cables de color amarillo, son los cables de datos de los sensores/motores*

- *Los cables verdes, son de la pantalla LCD*
- *Por ultimo los positivos y negativos, rojo y negro*