

Serra Automatizzata

Jabane Yassir, Venturelli Alice



Obiettivo del progetto

La serra automatizzata è un progetto innovativo finalizzato alla ricreazione di un ambiente ottimale per la coltivazione delle piante. Automatizzando il processo di irrigazione e illuminazione, si riesce a coltivare le piante anche in condizioni atmosferiche sfavorevoli. In questo modo la serra non necessita della cura di una persona, in quanto si autogestisce.

Monitorando i fattori ambientali in tempo reale, si può intervenire immediatamente evitando problemi alle colture, inoltre, c'è anche la possibilità di controllare tutti i fattori dal PC.

Obiettivo del progetto

L'idea che c'è dietro a questo progetto è quella di riportare l'agricoltura alla portata di tutti. Infatti disporre di un sistema di questo tipo, significa poter coltivare a casa tutto quello che si vuole riducendo di molto il tempo e la fatica da dedicarci.

È possibile, con un sistema ben sviluppato risparmiare anche più del 30% delle risorse naturali, ciò consente sia di abbattere i costi e sia di ridurre l'impatto ambientale.



Materiali utilizzati



- Sensore capacitivo di umidità
- Fotoresistenza
- Attuatori
- Arduino
- Pompa dell'acqua e tubo
- Striscia led RGB 5 V
- Cavi e componenti vari.

Funzionamento del progetto

Grazie al programma realizzato con linguaggio di programmazione C, implementato sul microcontrollore Arduino è stato possibile comandare un sensore di umidità ed una fotoresistenza per effettuare i controlli necessari al funzionamento ottimale del sistema.

Ciò permette di azionare un impianto a pioggia che irriga le piante ed un impianto di illuminazione per fornire la luce necessaria in modo costante per tutto il giorno.





Funzionamento del progetto

Raggiunto il livello desiderato di umidità del terreno l'impianto di irrigazione si ferma.

Il valore dell'umidità e l'eventuale necessità di irrigazione possono essere visualizzati su un piccolo display che garantisce all'utente un costante monitoraggio delle condizioni del sistema.

Quando viene rilevato un valore di luce inferiore alla soglia desiderata vengono azionate le luci in modo da fornire la luce necessaria alle piante.

L'illuminazione della serra e l'azionamento dell'impianto di irrigazione avvengono grazie all'impiego di energia elettrica.

Diagramma degli stati

L=Luce

SL= Valore letto dalla fotoresistenza

VS= Valore soglia luce

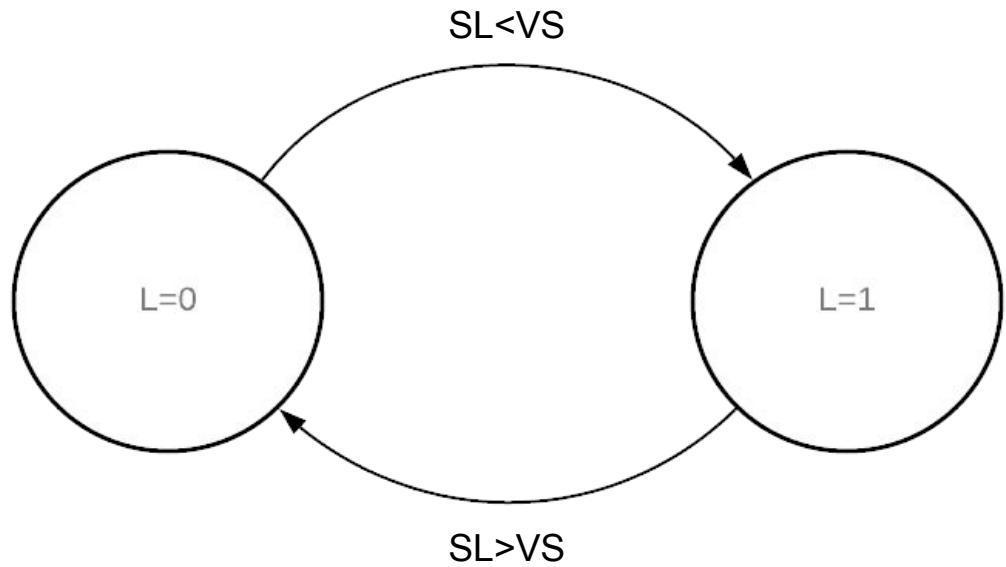


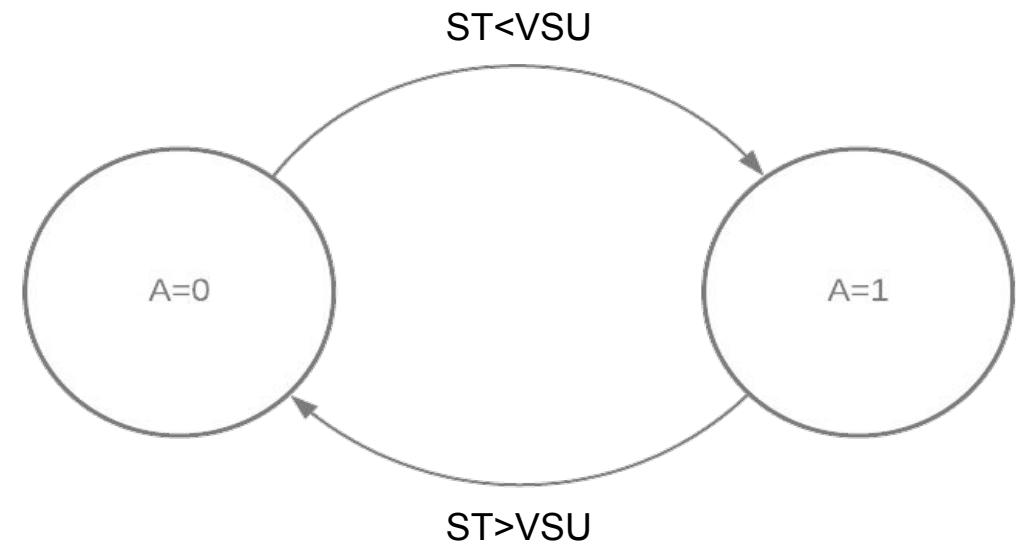


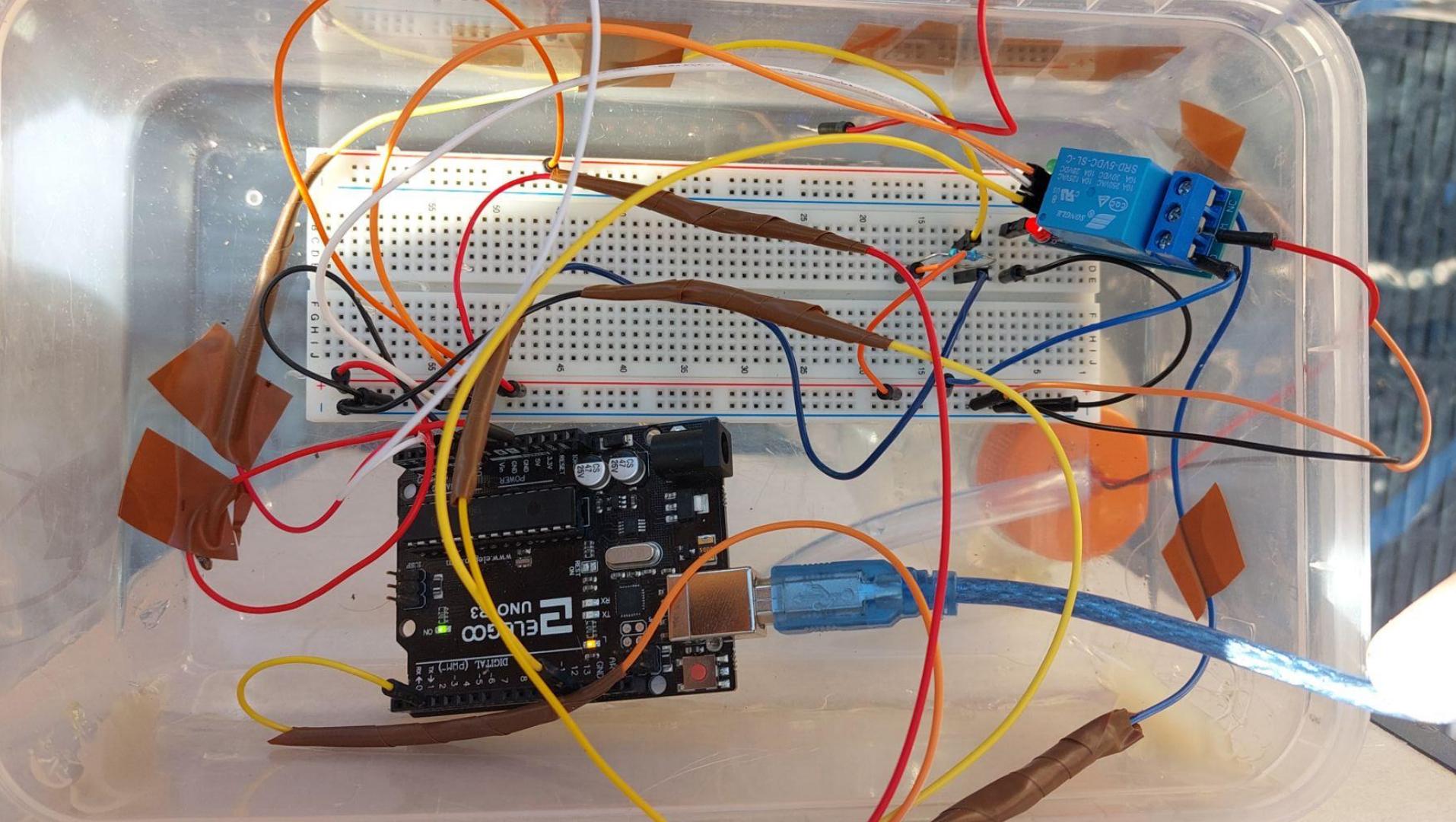
Diagramma degli stati

ST=Valore letto dal sensore di umidità

A=Pompa

V_{SUB}=Valore di soglia umidità







Migliorie future

- Aggiunta di altri sensori di umidità per gestire un impianto di irrigazione a zone, in modo da annaffiare solo le piante che hanno bisogno di acqua.
- Aggiunta di un impianto di riscaldamento e raffreddamento per poter controllare la temperatura interna della serra in modo da garantire sempre una temperatura ottimale.
- Aggiunta di un pannello HMI per fornire un'interfaccia grafica.
- Aggiunta di una scheda aggiuntiva per creare un sistema di controllo da remoto, per visualizzare grafici o dati relativi ai vari parametri di funzionamento.

CODICE COMMENTATO

```
// PROGETTO SERRA AUTOMATIZZATA JABANE, VENTURELLI

//DICHIARAZIONE LIBRERIE
#include <Arduino.h>
#include <dht11.h>
#include <LiquidCrystal.h>
#include <FastLED.h>
//DEFINIZIONE DI COSTANTI
#define LED_PIN 2      //PIN LED 2
#define NUM_LEDS 66    //NUMERO DEI LED
#define ON 1
#define OFF 0
#define LuminositaMin 700 //VALORE ANALOGICO MINIMO DI
LUMINOSITA'
#define TempMax 25      //VALORE TEMPERATURA MASSIMA
#define UmidMin 350     //VALORE ANALOGICO UMIDITA'
MINIMA

CRGB leds [NUM_LEDS];
```

```
LiquidCrystal lcd(7, 8, 9, 10, 11, 12); //PIN MONITOR LCD
dht11 DHT11;

//DEFINIZIONE DEI PIN DIGITALI

int PinPompa = 13;
int PinLuci = 2;
int PinVentole = 3;
int PinDht = 4;

//DEFINIZIONE DEI PIN ANALOGICI

int SensLuce = A1;
int PinUmid = A0;
```

CODICE COMMENTATO

```
void setup() {  
    lcd.begin(16, 2);      //16 COLONNE E 2 RIGHE MONITOR LCD  
    Serial.begin(9600);   //PORTA SERIALE A 9600 BAUD  
  
    //DEFINIZIONE DEI PIN DI INPUT E OUTPUT  
  
    pinMode(PinPompa, OUTPUT);  
    pinMode(PinLuci, OUTPUT);  
    pinMode(PinVentole, OUTPUT);  
    pinMode(PinDht, INPUT);  
    pinMode(PinUmid, INPUT);  
    pinMode(SensLuce, INPUT);  
  
    FastLED.addLeds<WS2812, LED_PIN, GRB>(leds, NUM_LEDS);  
}  
}
```

```
void loop() {  
    //ControlloTemp();  
    ControlloLuce();  
    ControlloUmid();  
    delay(1000);  
    //ESECUZIONE DELLE FUNZIONI DI CONTROLLO UNA VOLTA AL  
    SECONDO  
}
```

CODICE COMMENTATO

```
//FUNZIONI GESTIONE SENSORI

//CONTROLLO TEMPERATURA, NON USATO MA GIA' PROGRAMMATO

/*void ControlloTemp () {

    //LETTURA VALORI DAL DHT11

    int chk = DHT11.read(PinDht);
    float temp = DHT11.temperature;

    //SCRITTURA DELLA TEMPERATURA SUL MONITOR LCD

    if (chk == 0) {
        lcd.setCursor(0, 0);
        lcd.print("temperatura");
        lcd.print(":");
        lcd.print(temp, 1);

    }
}
```

```
//SE LA TEMPERATURA SUPERÀ LA SOGLIA MASSIMA SI ACCENDE
IL SISTEMA DI VENTILAZIONE

if (temp > TempMax) {
    Ventole(ON);
}
else {
    Ventole(OFF);
}
*/
}
```

CODICE COMMENTATO

```
//FUNZIONE CONTROLLO UMIDITA'

void ControlloUmid() {
    float umid = analogRead(PinUmid); //LETTURA E SCRITTURA
DEL VALORE ANALOGICO LETTO DAL SENSORE DI UMIDITA'
    //Serial.println(umid);          USATO PER SCRIVERE I VALORI
SUL MONITOR SERIALE PER EFFETTUARE I CONTROLLI INIZIALI PER
DEFINIRE IL VALORE MINIMO

//SCITTURA VALORI UMIDITA' SULL'LCD

/*
lcd.setCursor(0, 1);
lcd.print("umidita'");
lcd.print(":");
lcd.print(umid);
lcd.print('%');
*/
```

```
//SE L'UMIDITA' SUPERA IL VALORE MINIMO (ORDINE DEI VALORI
INVERSO, ASCIUTTO=VALORI ALTI) SI ACCENDE LA POMPA PER UN
SECONDO

if (umid > UmidMin) {
    digitalWrite(PinPompa, LOW); //CONTROLLO INVERSO DELLA
POMPA, OUTPUT BASSO, POMPA ACCESA
    delay(1000);
    digitalWrite(PinPompa, HIGH); //CONTROLLO INVERSO
DELLA POMPA, OUTPUT ALTO, POMPA SPENTA
    delay(10000);
} else {
    digitalWrite(PinPompa, HIGH);
}
```

CODICE COMMENTATO

```
//FUNZIONE CONTROLLO LIVELLO LUCE

void ControlloLuce() {
    float lumin = analogRead(A1); //LETTURA E SCRITTURA DEL
VALORE ANALOGICO LETTO DALLA FOTORESISTENZA
    //Serial.println(lumin);      USATO PER SCRIVERE I VALORI
SUL MONITOR SERIALE PER EFFETTUARE I CONTROLLI INIZIALI PER
DEFINIRE IL VALORE MINIMO

    //SE LA LUMINOSITA' SUPERA IL VALORE MINIMO (ORDINE DEI
VALORI INVERSO, BUIO=VALORI ALTI) SI ACCENDONO LE LUCI

    if (lumin > LuminositaMin) {
        LuciON();
    }
    if (lumin < LuminositaMin) {
        LuciOFF();
    }
}
```

```
//FUNZIONE GESTIONE VENTOLE
/*
void Ventole(char EnVentole){
    switch(EnVentole){
        case OFF: //ventole off
            digitalWrite(PinVentole, LOW);
            break;

        case ON: //ventole on
            digitalWrite(PinVentole, HIGH);
            break;
    }
}
*/
```

CODICE COMMENTATO

```
//FUNZIONI GESTIONE LUCI

//FUNZIONE PER ACCENDERE I LED AD UNO AD UNO CON UN DELAY DI
//20ms TRA LE ACCENSIONI

void LuciON() {
    for (int i = 0; i <= NUM_LEDS; i++) {
        leds[i] = CRGB(104, 50, 227); //VALORE HEX PER IL
        COLORE "BURPLE", COLORE OTTIMALE PER LA CRESCITA DELLE
        PIANTE
        FastLED.show();
        delay(20);
    }
}
```

```
//FUNZIONE PER SPEGNERE I LED AD UNO AD UNO CON UN DELAY DI
//20ms TRA LE ACCENSIONI (ORDINE DI SPEGNIMENTO INVERSO
//RISPETTO ALL'ACCENSIONE)

void LuciOFF() {
    for (int i = NUM_LEDS; i >= 0; i--) {
        leds[i] = CRGB(0, 0, 0); //VALORE HEX PER LO
        SPEGNIMENTO DEI LED
        FastLED.show();
        delay(20);
    }
}
```

FINE

Grazie per l'attenzione



