AI & SUSTAINABILITY IN VET EDUCATION

Nome dell'autore/i: Sucessos Criativos, Lda | P. Porto

Titolo del WP: WP3 – AI-SAVES@ProjectHub

Riferimento dell'attività: 3.2 – Modello di progetto per AI-SAVES@ProjectHub

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Indice dei contenuti

1. Titolo del progetto: 2

2. Categoria del progetto: 2

3. Descrizione del progetto: 2

5. Pubblico target: 2

6. Conoscenze preliminari: 2

7. Obiettivi del progetto: 3

8. Tempistica del progetto: 3

9. Risorse fornite: 3

10. Strumenti e tecnologie utilizzati: 4

11. Piano di implementazione passo dopo passo: 6

12. Criteri di valutazione: 13

13. Lingue disponibili: 13

14. Collaboratori: 13

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

1. **Titolo del progetto:**
   Sistema di rilevamento del movimento con illuminazione intelligente

---

2. **Categoria del progetto:**
   - IA nell'IoT

---

3. **Descrizione del progetto:**

   Questo progetto prevede lo sviluppo di un sistema intelligente di rilevamento del movimento utilizzando un Raspberry Pi, che controlla automaticamente l'illuminazione a strisce LED in base al rilevamento del movimento e alle condizioni di luce ambientale. Il sistema utilizza un sensore ad ultrasuoni per rilevare i movimenti nelle vicinanze e attiva l'illuminazione solo quando è buio, offrendo una soluzione di illuminazione automatizzata ed efficiente dal punto di vista energetico. Questo sistema intelligente contribuisce al risparmio energetico e aumenta la sicurezza in diversi ambienti, come corridoi, scale e ingressi.

---

4. **Parole chiave:**
   Rilevamento del movimento, Illuminazione intelligente, Raspberry Pi, IoT, Sensore ad ultrasuoni, Efficienza energetica, Sistemi automatizzati, Sostenibilità ambientale

---

5. **Pubblico target:**
   Sia docenti che studenti

---

6. **Conoscenze preliminari:**
   - Comprensione di base della legge di Ohm e dei circuiti elettrici
   - Introduzione alla programmazione Python (variabili, cicli, funzioni)
   - Conoscenza del controllo GPIO del Raspberry Pi
   - Esperienza di base con il protocollo di comunicazione SPI
   - Comprensione dell'interfacciamento dei sensori e dell'elaborazione dei segnali
   - Familiarità con la tecnologia e il controllo delle strisce LED

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

7. **Obiettivi del progetto:**
   - Introdurre gli studenti all'integrazione avanzata dei sensori IoT e all'automazione
   - Insegnare l'acquisizione di dati in tempo reale da più sensori simultaneamente
   - Sviluppare la comprensione dei sistemi di illuminazione intelligente ed efficienti dal punto di vista energetico
   - Promuovere la sostenibilità ambientale tramite l'automazione intelligente
   - Migliorare le competenze pratiche in elettronica, programmazione Python e integrazione di sistemi
   - Incoraggiare la creazione di soluzioni pratiche e scalabili per applicazioni reali
   - Favorire la comprensione degli algoritmi di rilevamento del movimento e della rilevazione della luce ambientale

---

8. **Tempistica del progetto:**

---

9. **Risorse fornite:**
   - Raspberry Pi 5 con SPI abilitato
   - Sensore di distanza ad ultrasuoni HC-SR04
   - Sensore LDR (Light Dependent Resistor)
   - Striscia LED NeoPixel (compatibile WS2812B)
   - Breadboard e cavi jumper
   - Alimentatore esterno per la striscia LED
   - Modelli di codice Python
   - Guida didattica completa passo-passo
   - Schemi di cablaggio e guida alla risoluzione dei problemi

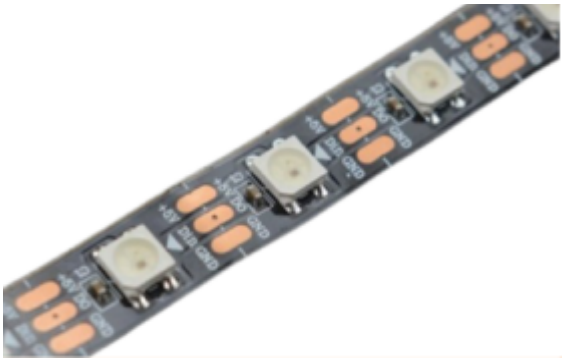
---

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

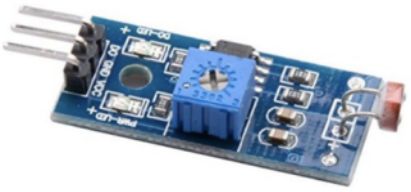
10. Strumenti e tecnologie utilizzati:

10.1 Hardware:

10 MATERIALE	
Raspberry Pi 5	
27W USB Type-C Power Supply for Raspberry Pi 5	
Flat Jumper Wires F/F	
Breadboard	
SINGLE-FILE WIRE FOR BREADBOARDS	
WS2812 RGB Programmable LED Strip 15x LED	

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

HC-SR04P - ULTRASONIC SENSOR 3V~5V	
LDR Sensor Module with Potentiometer	

10.2 **Software**:

- **Sistema operativo**: Raspberry Pi OS con SPI abilitato
- **Linguaggio di programmazione**: Python 3.7+
- **Ambiente di programmazione**: Python 3.7+ con Geany o IDE Python

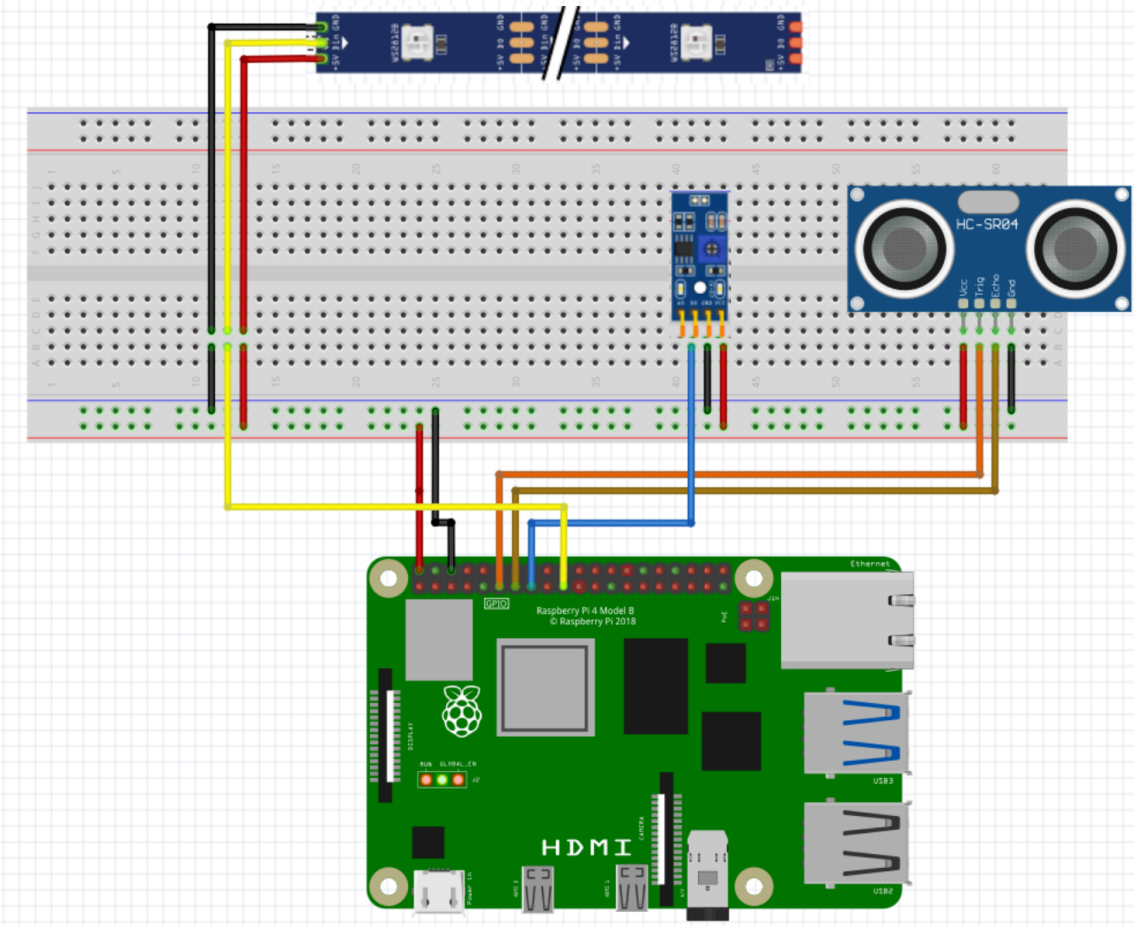
Librerie: RPi.GPIO, time, board, neopixel

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

11. Piano di implementazione passo dopo passo:

11.1 Configurazione hardware



Collegamenti del circuito

Sensore a ultrasuoni HC-SR04:

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

- VCC ⟶ 5V (Pin 2)
- GND ⟶ Massa (Pin 6)
- Trigger ⟶ GPIO 17 (Pin 11)
- Echo ⟶ GPIO 27 (Pin 13)

LDR Sensore:

- Un terminale ⟶ 3.3V (Pin 1)
- Altro terminale ⟶ GPIO 22 (Pin 15) e resistenza da 10kΩ verso massa (Pin 6)

NeoPixel LED Strip:

- VCC ⟶ Alimentazione esterna 5V
- GND ⟶ Massa (Pin 6) e massa dell'alimentazione esterna
- Data ⟶ GPIO 10 (MOSI/Pin 19)

Connessione di massa (GND) ⟶ Utilizzare il Pin 6 (GND) come massa comune per tutti i componenti

Passaggi di cablaggio:

1. Collegare il sensore HC-SR04: alimentazione 5V, massa, Trigger su **GPIO 17**, Echo su **GPIO 27**
2. Configurare il circuito LDR
3. Collegare la linea dati della striscia NeoPixel a **GPIO 10 (MOSI)**
4. Assicurarsi che tutte le connessioni di massa siano correttamente collegate alla **massa comune**

⚠️**Importante:** Prima di alimentare il sistema, verifica attentamente tutti i collegamenti. Un cablaggio errato può danneggiare i componenti o causare instabilità del sistema.

11.2 Configurazione del software

Ora che l'hardware è pronto, è il momento di implementare il sistema di controllo dell'illuminazione basato sul rilevamento del movimento e della luce ambientale.

1. Creare un ambiente virtuale

```
# Create project directory
mkdir motion_detection_system
cd motion_detection_system
```

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

```
# Create virtual environment
python3 -m venv venv

# Activate virtual environment
      source venv/bin/activate
```

2. Installare le dipendenze

```
# Install requirements
pip install -r requirements.txt
```

3. Aggiungere il codice

- Copia il codice Python del sistema di rilevamento del movimento nel file `main.py`
- Assicurati che il file si trovi nella directory del tuo progetto

4. Avviare il sistema

```
# Make sure virtual environment is activated
source venv/bin/activate

# Run the motion detection system
python main.py
```

5. **Arrestare il sistema**
   - Premi **Ctrl+C** per fermare il programma in modo sicuro
   - Tutti i componenti verranno ripristinati automaticamente in modo corretto.

Comportamento del sistema

Comportamento dei LED:

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

- **LED bianchi:** Attivati quando viene rilevato movimento in condizioni di scarsa illuminazione
- **Tutti spenti:** Nessun movimento rilevato o luce ambientale sufficiente
- **Spegnimento automatico:** I LED si spengono dopo 5 secondi
- **Attivazione intelligente:** Risponde al movimento solo quando la luce ambientale è bassa

Output della console:

- Misurazioni della distanza in tempo reale
- Stato del rilevamento del movimento
- Letture del livello di luce ambientale
- Messaggi di stato del sistema

CODICE

```
import time
import board
import lgpio
import neopixel_spi as neopixel

# GPIO pin definitions
TRIGGER_PIN = 17    # GPIO pin for ultrasonic sensor trigger
ECHO_PIN = 27       # GPIO pin for ultrasonic sensor echo
LDR_PIN = 22        # GPIO pin for LDR (Light Dependent Resistor) sensor

# NeoPixel LED strip configuration
NUM_PIXELS = 12                     # Number of LEDs in the strip
PIXEL_ORDER = neopixel.GRB          # Color order (Green, Red, Blue)
LED_COLOR = 0xFFFFFF                # White color for motion detection
LED_BRIGHTNESS = 0.5                # LED brightness (0.0 to 1.0)

# Motion detection parameters
MOTION_THRESHOLD = 50.0             # Distance threshold in cm to detect motion
LIGHT_ON_DURATION = 5.0             # How long to keep lights on after motion
(seconds)
MEASUREMENT_INTERVAL = 0.1          # Time between distance measurements
(seconds)

# Initialize GPIO chip
h = lgpio.gpiochip_open(0)
lgpio.gpio_claim_output(h, TRIGGER_PIN)     # Set trigger pin as output
lgpio.gpio_claim_input(h, ECHO_PIN)         # Set echo pin as input
lgpio.gpio_claim_input(h, LDR_PIN)          # Set LDR pin as input

# Initialize NeoPixel LED strip
spi = board.SPI()
```

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

```
pixels = neopixel.NeoPixel_SPI(
    spi,
    NUM_PIXELS,
    brightness=LED_BRIGHTNESS,
    pixel_order=PIXEL_ORDER,
    auto_write=False
)

# Global variables for motion tracking
previous_distance = None
lights_on_until = 0  # Timestamp when lights should turn off

def measure_distance():
    """
    Measure distance using the ultrasonic sensor.

    Returns:
        float: Distance in centimeters, or None if measurement fails
    """
    try:
        # Send 10-microsecond pulse to trigger pin
        lgpio.gpio_write(h, TRIGGER_PIN, 1)
        time.sleep(0.00001)  # 10 microseconds
        lgpio.gpio_write(h, TRIGGER_PIN, 0)

        # Wait for echo to start (pin goes HIGH)
        start_time = time.time()
        timeout = start_time + 0.1  # 100ms timeout

        while lgpio.gpio_read(h, ECHO_PIN) == 0:
            start_time = time.time()
            if start_time > timeout:
                return None  # Timeout occurred

        # Wait for echo to end (pin goes LOW)
        end_time = time.time()
        while lgpio.gpio_read(h, ECHO_PIN) == 1:
            end_time = time.time()
            if end_time > timeout:
                return None  # Timeout occurred

        # Calculate distance based on time of flight
        pulse_duration = end_time - start_time
        distance = (pulse_duration * 34300) / 2  # Speed of sound: 343 m/s
```

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

```
        return distance

    except Exception as e:
        print(f"Error measuring distance: {e}")
        return None

def is_dark():
    """
    Check if ambient light level is low using LDR sensor.

    Returns:
        bool: True if it's dark (LDR reads 1), False if there's light (LDR
reads 0)
    """
    try:
        return lgpio.gpio_read(h, LDR_PIN) == 1
    except Exception as e:
        print(f"Error reading LDR sensor: {e}")
        return False

def detect_motion(current_distance):
    """
    Detect motion by comparing current distance with previous measurement.

    Args:
        current_distance (float): Current distance measurement in cm

    Returns:
        bool: True if motion is detected, False otherwise
    """
    global previous_distance

    if previous_distance is None:
        previous_distance = current_distance
        return False

    # Calculate the change in distance
    distance_change = abs(current_distance - previous_distance)

    # Update previous distance for next comparison
    previous_distance = current_distance

    # Motion detected if distance change exceeds threshold
    return distance_change > 5.0  # 5cm change threshold
```

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

```
def turn_on_lights():
    """
    Turn on all LEDs in the strip with the specified color.
    """
    global lights_on_until
    # Set all pixels to the motion detection color
    for i in range(NUM_PIXELS):
        pixels[i] = LED_COLOR

    pixels.show()

    # Set the time when lights should turn off
    lights_on_until = time.time() + LIGHT_ON_DURATION
    print("Motion detected in dark environment - Lights ON")

def turn_off_lights():
    """
    Turn off all LEDs in the strip.
    """
    # Set all pixels to black (off)
    pixels.fill(0x000000)
    pixels.show()

    print("Lights OFF")

def should_lights_be_on():
    """
    Check if lights should currently be on based on timer.
    Returns:
        bool: True if lights should be on, False otherwise
    """
    return time.time() < lights_on_until

def main():
    """
    Main program loop for motion detection system.
    """
    global lights_on_until

    print("Motion Detection System Starting...")
    print(f"Motion threshold: {MOTION_THRESHOLD} cm")
    print(f"Light duration: {LIGHT_ON_DURATION} seconds")
    print("Press Ctrl+C to exit")

    try:
```

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

```
        # Initialize with lights off
        turn_off_lights()

        while True:
            # Measure current distance
            distance = measure_distance()

            if distance is not None:
                # Check if object is within motion threshold
                object_detected = distance <= MOTION_THRESHOLD

                # Check ambient light level
                dark_environment = is_dark()

                # Print status information
                print(f"Distance: {distance:.1f}cm | Object detected:
{object_detected} | Dark: {dark_environment}")

                # Turn on lights if object detected in dark environment
                if object_detected and dark_environment:
                    turn_on_lights()

                # Turn off lights if timer expired (mandatory after timeout)
                if not should_lights_be_on() and lights_on_until > 0:
                    turn_off_lights()
                    lights_on_until = 0

            else:
                print("Distance measurement failed")
                # Turn off lights if timer expired even with failed measurement
                if not should_lights_be_on() and lights_on_until > 0:
                    turn_off_lights()
                    lights_on_until = 0

            # Wait before next measurement
            time.sleep(MEASUREMENT_INTERVAL)

    except KeyboardInterrupt:
        print("\nShutting down motion detection system...")

    finally:
        # Cleanup: turn off lights and close GPIO
        turn_off_lights()
        lgpio.gpiochip_close(h)
        print("System shutdown complete!")
```

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

```
if __name__ == "__main__":
    main()
```

Ora puoi eseguire il programma e vedere il risultato:

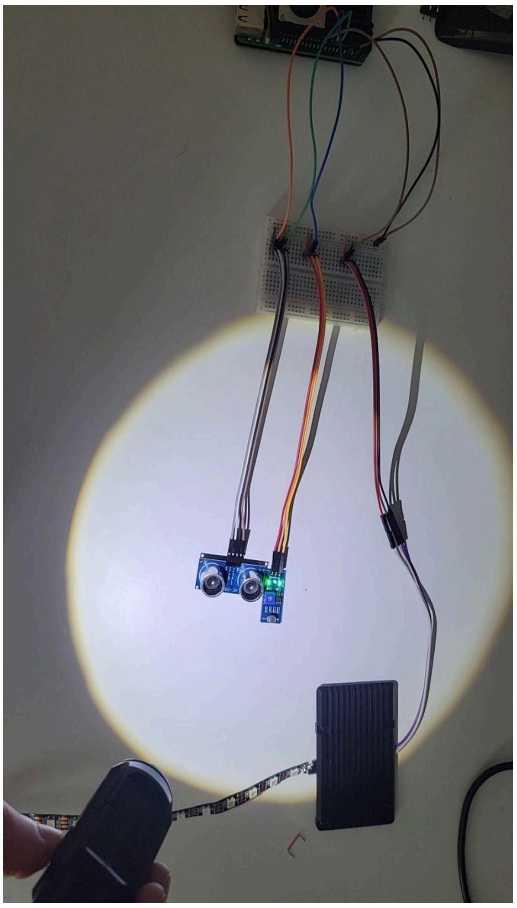
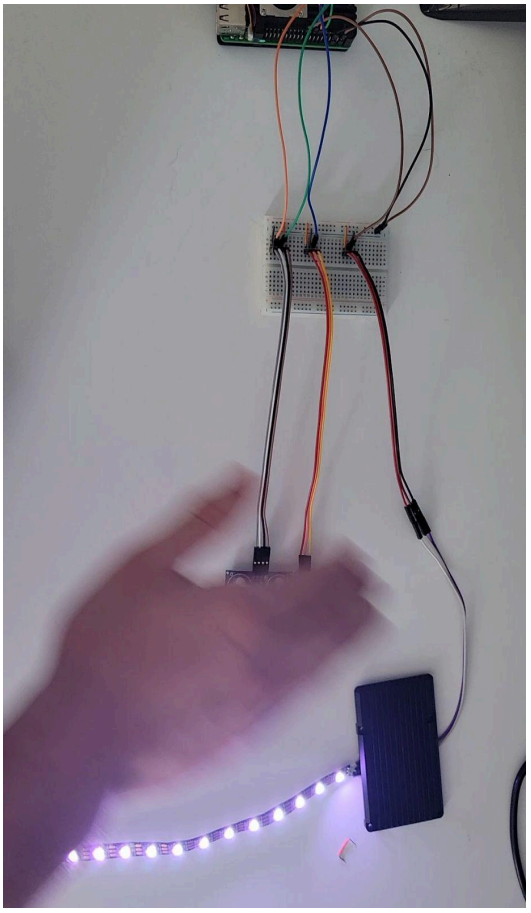
```
Distance: 8.9cm | Object detected: True | Dark: True
Motion detected in dark environment - Lights ON
Distance: 8.9cm | Object detected: True | Dark: True
Motion detected in dark environment - Lights ON
Distance: 9.2cm | Object detected: True | Dark: True
Motion detected in dark environment - Lights ON
Distance: 9.2cm | Object detected: True | Dark: True
Motion detected in dark environment - Lights ON
Distance: 7.8cm | Object detected: True | Dark: True
Motion detected in dark environment - Lights ON
Distance: 7.8cm | Object detected: True | Dark: True
Motion detected in dark environment - Lights ON
Distance: 9.5cm | Object detected: True | Dark: True
Motion detected in dark environment - Lights ON
Distance: 13.3cm | Object detected: True | Dark: True
Motion detected in dark environment - Lights ON
Distance: 16.0cm | Object detected: True | Dark: True
Motion detected in dark environment - Lights ON
Distance: 169.6cm | Object detected: False | Dark: True
Distance: 169.6cm | Object detected: False | Dark: True
Distance: 169.7cm | Object detected: False | Dark: True
Distance: 170.1cm | Object detected: False | Dark: True
Distance: 169.8cm | Object detected: False | Dark: True
Distance: 170.1cm | Object detected: False | Dark: True
```

```
Distance: 169.7cm | Object detected: False | Dark: False
Distance: 169.7cm | Object detected: False | Dark: False
Distance: 169.8cm | Object detected: False | Dark: False
Distance: 169.7cm | Object detected: False | Dark: False
Distance: 28.2cm | Object detected: True | Dark: False
Distance: 22.1cm | Object detected: True | Dark: False
Distance: 16.0cm | Object detected: True | Dark: False
Distance: 10.5cm | Object detected: True | Dark: False
Distance: 9.2cm | Object detected: True | Dark: False
Distance: 169.7cm | Object detected: False | Dark: False
Distance: 169.7cm | Object detected: False | Dark: False
Distance: 169.8cm | Object detected: False | Dark: False
Distance: 169.7cm | Object detected: False | Dark: False
Distance: 169.7cm | Object detected: False | Dark: False
Distance: 169.7cm | Object detected: False | Dark: False
Distance: 169.7cm | Object detected: False | Dark: False
Distance: 170.1cm | Object detected: False | Dark: False
Distance: 8.5cm | Object detected: True | Dark: False
Distance: 10.5cm | Object detected: True | Dark: False
Distance: 16.3cm | Object detected: True | Dark: False
Distance: 24.7cm | Object detected: True | Dark: False
Distance: 41.3cm | Object detected: True | Dark: True
Motion detected in dark environment - Lights ON
```

```
Distance: 170.1cm | Object detected: False | Dark: True
Distance: 170.1cm | Object detected: False | Dark: True
Distance: 170.1cm | Object detected: False | Dark: True
Distance: 169.7cm | Object detected: False | Dark: True
Distance: 170.1cm | Object detected: False | Dark: True
Distance: 169.8cm | Object detected: False | Dark: True
Distance: 170.1cm | Object detected: False | Dark: True
Distance: 169.7cm | Object detected: False | Dark: True
Distance: 169.7cm | Object detected: False | Dark: True
Distance: 170.1cm | Object detected: False | Dark: True
Distance: 170.1cm | Object detected: False | Dark: True
Distance: 170.1cm | Object detected: False | Dark: True
Distance: 170.1cm | Object detected: False | Dark: True
Distance: 170.1cm | Object detected: False | Dark: True
Lights OFF
Distance: 170.1cm | Object detected: False | Dark: True
Distance: 170.2cm | Object detected: False | Dark: True
Distance: 170.1cm | Object detected: False | Dark: True
Distance: 170.1cm | Object detected: False | Dark: True
Distance: 169.7cm | Object detected: False | Dark: True
Distance: 170.1cm | Object detected: False | Dark: True
Distance: 169.7cm | Object detected: False | Dark: True
Distance: 169.8cm | Object detected: False | Dark: True
```

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

12. Criteri di valutazione:

- **Configurazione hardware (30%)**: Cablaggio corretto e connessione stabile di Raspberry Pi, sensore ad ultrasuoni HC-SR04, sensore LDR e striscia LED NeoPixel; uso corretto dell'alimentatore esterno; tutte le masse collegate correttamente; nessuna connessione allentata o errata.
- **Funzionalità del codice (25%)**: Il sistema funziona senza errori; i LED si attivano solo nelle condizioni corrette (movimento rilevato + luce ambientale bassa); i sensori rispondono accuratamente; gli output della console sono informativi e in tempo reale.
- **Comprensione del sistema (20%)**: Spiegazione chiara di come i sensori ad ultrasuoni e LDR lavorano insieme per controllare il comportamento dei LED; comprensione della comunicazione SPI; uso appropriato delle risorse di alimentazione e GPIO.

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

- **Efficienza energetica e miglioramenti (15%)**: Dimostrazione della comprensione del risparmio energetico tramite illuminazione condizionata; idee di miglioramento come controllo tramite app mobile, registrazione dei dati o integrazione con alimentazione solare.
- **Interfaccia utente e interazione (10%)**: Il sistema gestisce casi limite (ad es. falsi positivi, variazioni improvvise di luce); procedure di avvio/arresto sicure; corretta gestione dei componenti e distribuzione sicura dell'alimentazione.

---

13. Lingue disponibili:

---

14. Collaboratori:

Co-funded by the European Union

This project has been funded with support from the European Commission. This communication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.