Introduction to Robotics: Homework #6

Pseudo-Smart Environment Monitor and Logger

Andrei Dumitriu andrei.dumitriu@fmi.unibuc.ro

V1.2 Last Updated: November 14th, 2023, 21:55

Due Date

Submit your completed assignment to your respective lab session during the week of November 20th - November 26th, 2023. Ensure your Git repository is ready and the assignment is submitted before the lab begins.

1 Objective

Develop a "Smart Environment Monitor and Logger" using Arduino. This system will utilize various sensors to gather environmental data, log this data into EEPROM, and provide both visual feedback via an RGB LED and user interaction through a Serial Menu. The project focuses on integrating sensor readings, memory management, Serial Communication and the general objective of building a menu. **See the partial example video (only of menu parsing) here:** `https://www.youtube.com/watch?v=mh0KYdul1Sk`

2 Components Required

- Arduino Uno Board
- Ultrasonic Sensor (HC-SR04)
- LDR (Light-Dependent Resistor) aka Photocell aka Photoresistor aka Light Sensor
- RGB LED
- Resistors as needed
- Breadboard and connecting wires
- (Optional) Additional sensors / compoennts for extended functionality

2.1 Menu Structure

1. Sensor Settings // Go to submenu

1.1 Sensors Sampling Interval. Here you should be prompted for a value between 1 and 10 seconds. Use this value as a sampling rate for the sensors. You can read a separate value for each or have the same for both.

1.2 Ultrasonic Alert Threshold. Here you should be prompted for a threshold value for the ultrasonic sensor. You can decide if that is the min or max value (you can signal that something is too close). When sensor value exceeds the threshold value, an alert should be given. This can be in the form of a message. If the LED is set to Automatic Mode (see section 4.2), it should also turn red if any of the sensors are outside the value.

1.3 LDR Alert Threshold. Here you should be prompted for a threshold value for the LDR sensor. You can decide if that is the min or max value (for example, it could signal that night is coming). When sensor value exceeds the threshold value, an alert should be given. This can be in the form of a message. If the LED is set to Automatic Mode (see section 4.2), it should also turn red if any of the sensors are outside the value.

1.4 Back // Return to main menu

2. Reset Logger Data. Should print a message, promting if you to confirm to delete all data. Something like "are you sure?", followed by the submenu with YES or NO. You can reset both sensor data at the same time, or you can do it individually. Your choice. Individually makes more sense, but I'm trying to simplify the homework.

2.1 Yes.

2.2 No.

3. System Status // Check current status and health

3.1 Current Sensor Readings. Continuously print sensor readings at the set sampling rate, from all sensors. Make sure you have a way to exit this (such as pressing a specific key) and inform the user of this method through a message.

3.2 Current Sensor Settings. Displays the sampling rate and threshold value for all sensors.

3.3 Display Logged Data. Displays last 10 sensor readings for all sensors. (or be creative and do it another way).

3.2 Back. Return to Main menu.

4. RGB LED Control // Go to submenu

4.1 Manual Color Control. Set the RGB colors manually. You decide how to input them, either by making an option for each channel, or by putting a string etc. If you expect a specific format, make sure to inform the user.

4.2 LED: Toggle Automatic ON/OFF. If automatic mode is ON, then the led color should be GREEN when all sensors value do not exceed threshold values (aka no alert) and RED when there is an alert (aka ANY sensor value exceeds the threshold). When automatic mode is OFF, then the LED should use the last saved RGB values.

4.3 Back // Return to main menu

2.2 Be Careful:

1. **EEPROM Write Cycles:** Avoid excessive writing to EEPROM to prevent wear. **DO NOT USE EEPROM.WRITE**. Only use Update() or Put().
2. **Sensor Calibration:** Ensure sensors are correctly calibrated for accurate readings. (aka you know their interval values)
3. **Serial Communication Errors:** Implement error handling for Serial Communication to manage incorrect inputs.

3 Submission Guidelines

Upload your code to GitHub and update the README with at least:

1. Task requirements. Include the menu structure in the description.
2. A photo of your setup
3. A link to a video demonstrating the functionality (preferred: YouTube)
4. Ensure the video is correctly oriented.

Submit your homework through MS Teams once your Git repository reflects the latest changes.

Coding Standards

Clean and readable code is essential for full credit. Just make sure you follow the (or a) coding standard.

4 Bonus Opportunities

1. **Advanced Data Analysis:** Include features like calculating averages or detecting trends in the sensor data.
2. **Additional Sensor Integration:** Incorporate more sensors (like temperature or humidity) for a more comprehensive monitoring system.
3. **Creative LED Feedback:** Innovate with the RGB LED to create intricate feedback patterns or color schemes based on environmental changes.
4. **Control individual Sensors:** While the requirements allow for sensors to be controlled in bulk (reset data etc), a system that allows for individual control, as well, is more robust.