Gila Kohanbash
Karla Vasquez
EE-250
3 May 2023

Parking Sensor Writeup

Project Description

We decided to name our IoT system the Parking Sensor. It will try to measure the distance of a car (for now represented by our hand) from the max threshold it can reach before “hitting the curb” when parking. The wall represents our curb. If it is at a safe distance from “the curb”, the green LED light will turn on, and as the object moves closer to the ultrasonic device, the yellow LED turns yellow, until it eventually turns on the blue LED when it has hit the curb (aka surpassed the max distance threshold).

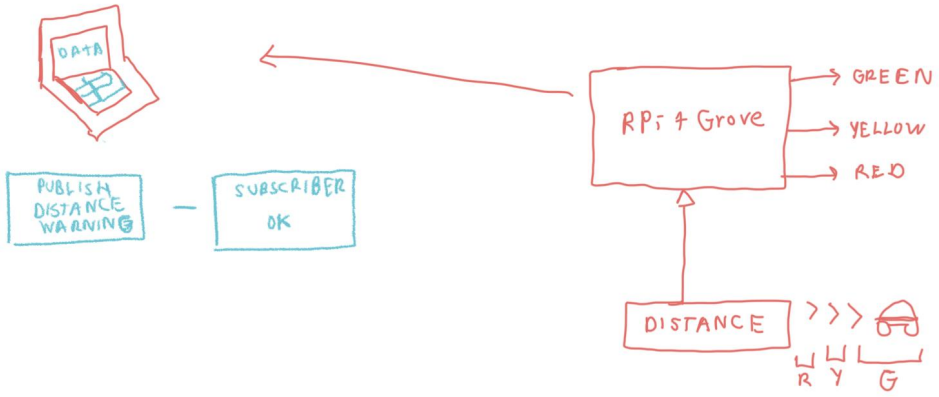
Components & Protocols Used

Parking Sensor used three LED lights, colored red, green, and blue, with less than 2.5k ohms in resistance. We used the GrovePi attached to the Raspberry Pi 3, along with a breadboard, jumper wires, and the ultrasonic sensor itself. Guided by our MQTT Lab 4, and using the nano editor on our terminal to run the code, implemented communication between the two nodes aside from outputting the distance onto the LED screen throughout the parking procedure. We used the MQTT as our Publish-Subscribe protocol to communicate the distances to our client, the driver, and inform them if the distance was okay. In other words, the MQTT protocol served as communication between an invisible traffic controller and the driver.

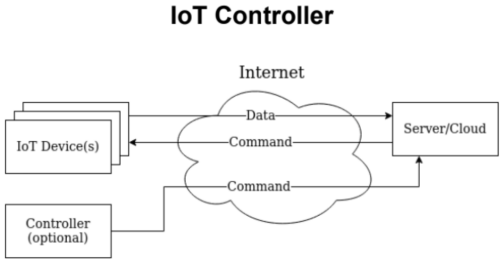
Final Project Reflection

Several of the challenges we faced were trying to combine the publisher code into the hardware code as one main file. We were getting connection errors with the subscriber that had to deal with a few mosquitto libraries that have not been downloaded yet. Another challenge was with our hardware code as we had to make sure the LEDs went into the correct pins, along with dealing with an appropriate brightness level for our LEDs (our LEDs don’t have the same brightness because we couldn’t find three of the same resistors). Furthermore, the X-Factor to our project is making the LED screen light up like a bumper car, and have it be able to detect the wall as a “curb” with an appropriate distance. Finally, the lessons learned is to make sure all libraries are downloaded before testing out the MQTT protocol, along with how to combine MQTT to our hardware code that deals with the LCD screen and LEDs. Before, MQTT had its own publish and subscribe files, so combining the publish code into one and making sure it was still compiling was a learning experience.

Block Diagram of Parking Sensor



Option We Decided to Go For



General Information about Parking Sensor:

1. Two or more physical nodes
    - Laptop
    - Raspberry Pi + GrovePi

2. Data Collection
    - Using Ultrasonic ranger

3. Data Processing
    - LCD Screen outputs warning messages regarding current parking situation; outputs
    - Going with option two, using an IoT Controller model

4. Node-to-Node Communication
    - In order to receive a response from the driver acknowledging it understands the instructions, we are implementing an MQTT publish-subscribe protocol to have the publisher reply with a command when distance is published

5. Visualization
    - MQTT protocol (data shown on laptop)