Adam Lewczuk and Michael Wasson
Electrical Engineering 250
Professor Bhaskar Krishnamachari
May 4, 2023

Air Conditioning and "Heating" System with the GrovePi Temperature and Humidity Sensor

For our project we decided to utilize the GrovePi temperature sensor to build a proof-of-concept temperature control system. This project used two nodes which communicated bidirectionally over TCP. Our first node was a Raspberry Pi that ran the TCP client code and read signals from three sensors: the temperature and humidity sensor, a potentiometer, and a button which were all attached to a GrovePi shield. Our second node was a laptop which ran the TCP server code to process incoming data, send output back to the client, and create a .png file at the end of the program. Our system calculated the heat index value from temperature and humidity and controlled a fan and LCD display based on this value's relation to a set threshold temperature.

Starting with our sensor readings, the GrovePi temperature and humidity sensor was the main source of data which measured temperature values in celsius and percent humidity values. We additionally included a potentiometer to allow the user to adjust a threshold value for their ideal room temperature. This value was displayed on an LCD screen attached to the GrovePi to let the user know what temperature they set it to. The last sensor was our button which would end the TCP communication and the program execution upon being pressed. All of these values were sent to our server to be processed.

On our server side, we started off by having code that would end TCP communication if the input indicated that the button was pressed. We then took our incoming temperature and humidity values and processed it to calculate the heat index. After converting the temperature to fahrenheit, we used the NOAA's heat index equation to calculate the desired value: https://www.wpc.ncep.noaa.gov/html/heatindex_equation.shtml. After finding the heat index, we subtracted the threshold temperature determined by the potentiometer from the heat index value. For each iteration of incoming data, we appended temperature, humidity, and heat index to their own arrays so that the data could be turned into a Pandas DataFrame, printed out by Plotly, and saved in a .png file at the end of the program running once the button was pressed.

After the data processing, we sent information back to the GrovePi client to its actuators. Based on testing, the mini fan took in a PWM input using the AnalogWrite() function using a value of 0-255. Since the speed of the fan rapidly approached its maximum in the first few numbers, we mapped a tested value to determine five distinct speeds of the fan. Taking the difference between the calculated heat index and threshold temperature, we sent “RED” and a “0” to the GrovePi if the heat index was below the specified threshold to disable the fan and turn the LCD red to convey “heating”. If the heat index was above the specified threshold, we sent “BLUE” to indicate “air conditioning” mode. While the fan was sent a “0” if the temperature was within 5 degrees of the threshold, non-zero integers were sent to the fan in 5-degree intervals to increase the fan-speed the higher the heat index value. Both the color label and the integer were sent back to the GrovePi which would control the LCD and fan respectively.

As we conclude the project, it is important to discuss our limitations. Most obviously, our mini fan was not large enough to have any real impact on ambient temperature. Additionally, we had no actuator to act as a heater. Because of this, we had to use fan modes and the color of the LCD screen as a proof-of-concept. Given better and more powerful actuators, we would change our processing in the server code to a PID controller to maintain a stable temperature of the environment. We also could have perfected our LCD display which only updated every second, the rate at which the while loop was running at. To make our LCD update seamlessly, we would include some type of interrupt in our future design which would support 2 processes running at once. Despite our limitations, our system was able to detect temperature and humidity, process the data, create a visualization, and control our mini fan to show that the concept was possible.