Air Quality and Noise Levels in Different Parts of Kamp-Lintfort City

1. Introduction

The German Environment Agency (UBA) and the European Environment Agency (EEA) reported that tens of thousands of people die because of high air pollution of early deaths and Year of Life Lost due to premature mortality. Because of its coal mining industry, North Rhine-Westphalia (NRW) has a long history of air pollution. Additionally, with its high population density of 1,170 inhabitants per square kilometer, NRW state corresponds to one of the dentist areas and, therefore, one of Germany's highest traffic areas. The NO_2 emissions exceed limiting values in several cities in NRW (L. Petry et al., 2020).
Air quality is essential when deciding where to live and pursue a life with good living standards. In particular, serious health issues are caused by air pollution, such as neurological, cardiovascular, and respiratory diseases. Noise coming mainly from dense traffic, industrial and construction activities can cause a significant impact on people's health. Kamp-Lintfort is a university town in NRW with a large community of academics and students. It is crucial as part of the scientific community to monitor the living quality in Kamp-Lintfort, particularly in residential areas. Low air quality levels or high levels of noise could affect students' brain functionality and performance at the university.
This project aims to measure mainly the concentrations of carbon dioxide (CO_2) and the sound intensity in three different locations in Kamp-Lintfort. The first one is Meisenweg, the second one is Moerserstraße, and the third one is Kamperdickstraße. Assumably, the three areas are selected in a way that would give one extreme case with high concentrations of the chosen toxic gas and high noise level, the second with medium levels, and the third with low ones. For that purpose, two different sensors are utilized and a microcontroller to read the resultant values and communicate data, hence drawing out well-derived conclusions.

2. Materials Description

- 1× Arduino Uno R3
- 1× MQ-135 gas sensor
- 1× LM-393 sound sensor
- 1× LCD display with I2C adapter (16 chars × 2 lines)
- 2×LED lights
- 1×220 Ohm Resistors
- 1×10k Ohm Resistor
- Jumper/ connecting wires
- 1× White breadboard

2.1 Arduino Uno R3

The Arduino Uno microcontroller board originated from the ATmega328P. It has 14 digital input/output pins, six analog inputs, a USB connection, a power jack, an ICSP header, and a reset button. It is connected to a computer with a USB cable.

Figure 1: Arduino Uno R3 microcontroller. Source: The most complete starter kit UNO R3 Project from Rhein-Waal Hochschule.

2.2 MQ135 Sensor

https://www.electronicoscaldas.com/datasheet/MQ-135_Hanwei.pdf
https://microcontrollerslab.com/interfacing-mq-135-gas-sensor-arduino/

MQ135 sensors are commonly used in air quality control equipment for buildings/offices. They are suitable for detecting NH_3, NO_x, alcohol, Benzene, smoke, CO_2, etc. MQ135 is selected to measure the air quality in residential areas in KL for several reasons: an affordable, easy-to-use sensor that measures the concentration of various toxic gases and has decently high sensitivity and fast response. Moreover, its operation is based on a simple drive circuit, making it suitable for controlling air quality levels on small scales like our project.

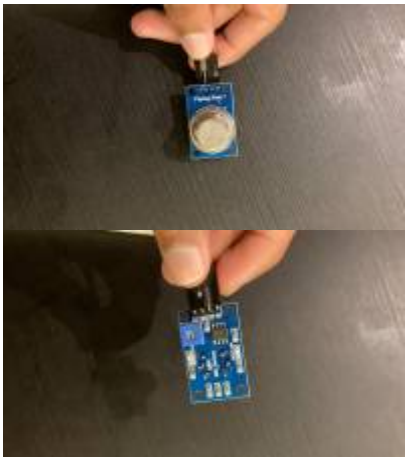

Figure 2: MQ135 Air Quality Sensor

Specifications: The working voltage is 5 V, and the analog output voltage is 0-4.2V.
MQ135 senses the air quality via a chemical-sensitive element covered by a steel exoskeleton. This element is subjected to a preheating current, where the gases to be measured later get ionized and

absorbed. The preheating time required is around 12-24 hours. After absorption of these gases, the resistance of the sensing substance changes, which in turn changes the amount of current going out. When the gas concentrations transcend specific safety limit values, an LED light is illuminated as an alarm.
The wire connections in the circuit are as followed:

- Arduino Analog Output pin A0 with the MQ135 Sensor Analog Output
- Arduino 5V pin with the MQ135 Sensor Vcc
- Arduino Ground (GND) pin with the MQ135 sensor Ground (GND)

2.3 LM393 Sensor

https://www.electronicshub.org/interfacing-sound-sensor-with-arduino/
https://components101.com/modules/lm393-sound-detection-sensor-module
https://5.imimg.com/data5/CY/QV/MY-1833510/sound-detection-sensor.pdf

A sound sensor with an LM393 comparator is used to measure sound intensity in the three selected areas in Kamp-Lintfort in this project. The sensor consists mainly of a microphone, a voltage comparator IC LM393, a sensitivity adjustment potentiometer, two LEDs (one power LED and one OUT LED), and a few other passive components (resistors and capacitors). The sound sensor module is a low-cost electronic sensor, and its circuit is easy to build. Moreover, it has an adjustable sensitivity, makes its operation flexible, and can be used in security and monitoring applications.
The microphone, a capacitor-based electronic component, detects the sound wave and sends electrical pulses to the circuit board. The potentiometer is used to adjust the sensor's sensitivity. At the same time, the voltage comparator IV LM393 processes the signal by comparing it to the threshold preset by the potentiometer. Thus, the digital output is obtained (Digital Output (0 or 1)). By rotating the trimmer knob of the potentiometer clockwise (counterclockwise), the sensor's sensitivity decreases (increases).
Specifications: The working voltage is 3.3-5V.

Figure 3: LM393 Microphone Sound Sensor

The wire connections in the circuit are as followed:

- Arduino Digital Output pin 12 with the LM393 Sensor Digital Output
- Arduino 5V pin with the LM393 Sensor Vcc
- Arduino Ground (GND) pin with the LM393 sensor Ground (GND)

2.4 LCD Screen Paired with I2C

https://core-electronics.com.au/tutorials/use-lcd-arduino-uno.html
http://www.handsontec.com/dataspecs/module/I2C_1602_LCD.pdf

Figure 4: Liquid Crystal Display (LCD) screen (16 x 2) paired with I2C

Liquid Crystal Display is a screen used to display values and data when working with Arduino IDE 1.8.15. It is composed of a liquid crystal inserted between 2 glass pieces, and it reacts when current is applied. A backlight is responsible for showing the values on the screen. The contrast between the values written and the dark background is controlled using the contrast control options by rotating the potentiometer knob.
The I2C or Inter-Integrated circuits make it easier to connect multiple “slaves” to a single master. Instead of utilizing numerous pins between the Arduino Uno and the LCD, only the following I2C pins are connected with the Arduino Uno.

- SDA pins
- SCL pins
- GND
- VCC to a 5V output in the Arduino Uno

Meanwhile, the LCD should ideally be soldered to the I2C. But in this project, the LCD and I2C were connected directly on the breadboard.

3. Setup Diagram

After testing each sensor alone, they are all set up together with the help of an Arduino white breadboard. The breadboard is powered by a 5V voltage supply on the positive red line, whereas the ground is the negative blue line. Since more than the pins provided by the Arduino microcontroller board were needed to suffice sensors, the LCD and its adapter, and alarm systems (with the LEDs and the buzzer), The white breadboard is used. For better visualization, see figure 5.

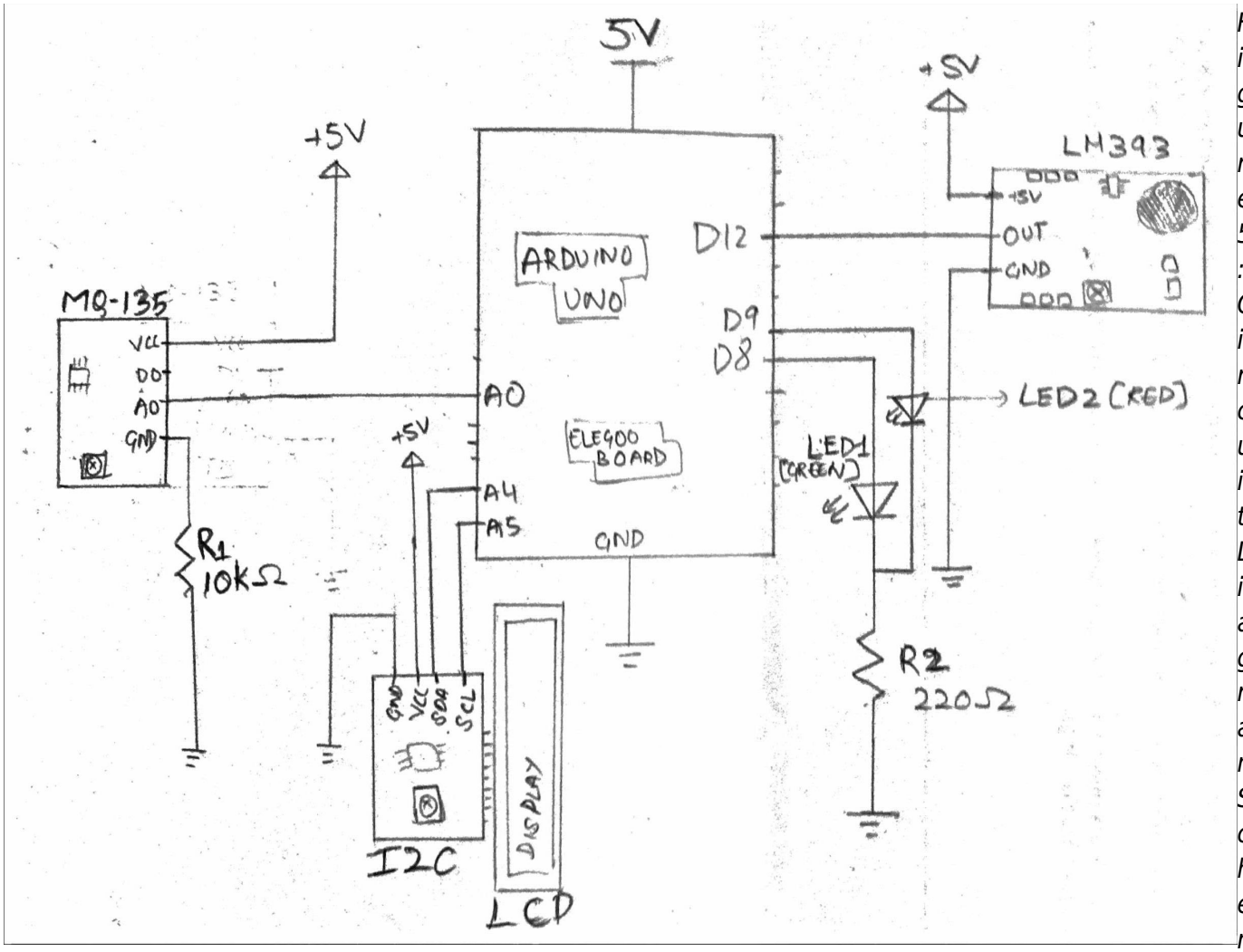

Figure5: Circuit Diagram Schematic of the Connections between the sensors MQ135 and LM393.

A tutorial explaining the circuit is also available here = Circuit Explained

4. Software

https://www.youtube.com/watch?v=PYkzJQhFNlA
https://arduino-tutorials.net/tutorial/drawing-sound-sensor-data-on-serial-plotter
https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://www.electroniclinic.com/co2-concentration-co2-ppm-or-co2-levels-using-mq135-sensor-arduino/

Code

Sound

Calibration and Testing:

The sensor's wiring is straightforward: its 5V to the Arduino 5V(via breadboard), the GND to Arduino GND (via breadboard), and the OUT of the sensor is connected to Arduino digital input D12. After powering up the sensor to the 5V, the threshold voltage for the IC LM393 comparator is set by

rotating the preset knob of the potentiometer. When the sensor microphone detects sound, a large amount of voltage (pulse) is transferred to the input of the IC LM393 comparator. The comparator processes the input by comparing it to the preset threshold voltage. If the input pulse exceeds the threshold, the sensor output will be 1 (HIGH). In contrast, when the sensor does not detect any sound (silence), then a small amount of voltage is transferred to the input of the comparator, which processes the signal, and If the input pulse is lower than the threshold, the sensor output will be 0 (LOW). This means that if the detected sound is loud (high noise), the sensor output goes to HIGH (digital output = 1), while the sensor output goes to LOW (digital output = 0) if the detected sound is less than the preset threshold.
First, the digital output of the sound sensor is checked directly on the serial monitor, and the sensitivity of the sensor is adjusted to a reasonable level (see figure 6).

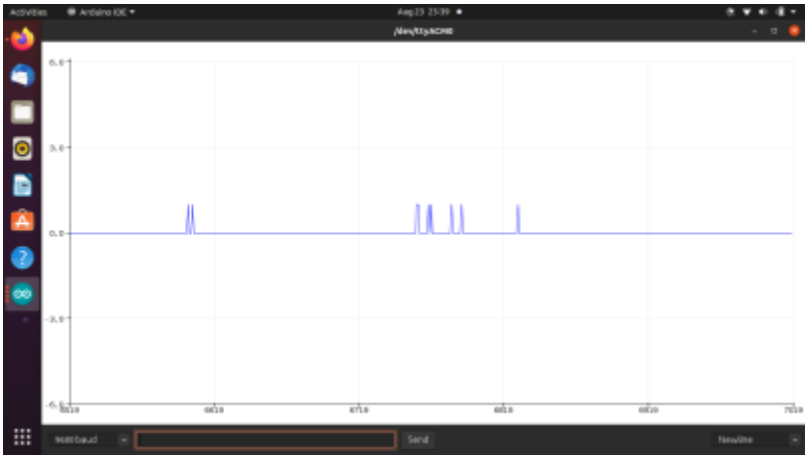

Figure 6: Serial Plotter: Adjustment of the sensor sensitivity using the potentiometer

Second, to get the information about the intensity of the sound, we introduced a sampling time (SAMPLE_TIME) into the loop void of the code. The code will sum up the number of times the sensor's output is “==HIGH” (loud sound/high noise). The sum (sampleBufferValue) is tightly related to the intensity of the sound, and the sampling time can be set up by the user based on the expected noise in the tested area (frequency of the sound wave). Then, we added to the sensor's circuit a LED that blinks if the sampleBufferValue surpasses the predefined threshold. A SAMPLE_TIME of 10ms is chosen to extract the following results, motivated by the Arduino board's fast response. The blinking time of the alarm LED is adjusted to match the sampling time (the LED must stop blinking to be ready for the next sampling time).

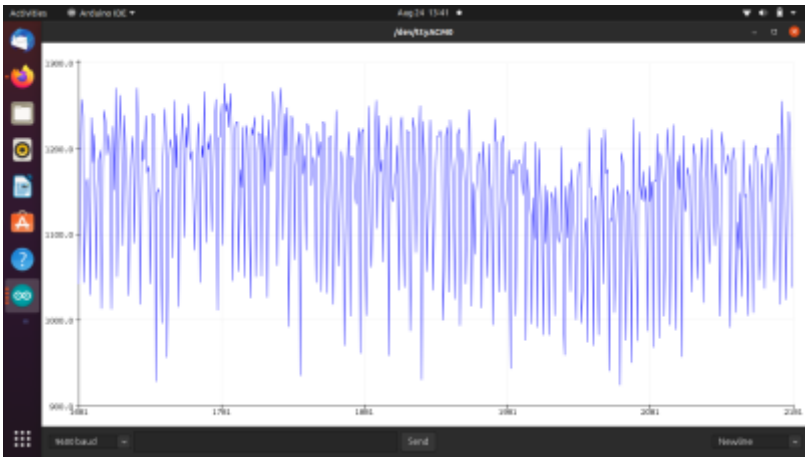

Figure 7: Serial Plotter: “sampleBufferValue” as a proxy to the intensity of sound detected by the LM393 for the case of vacuum cleaner as a source of sound.

Figure 7 shows an example of the Serial Plotter output for the Vacuum cleaner case as a source of continuous sound and detected by the digital sensor LM393.
Finally, to obtain a reasonable calibration of our sound sensor, we use an application called “Schall” from the Google play store to relate raw readings from the sensor to a decibel unit of sound intensity.

Table 1: Rough calibration of the LM393 sensor for noise detection

Test	Average dB (Schall/Google Apps)	Our LM393 sensor	Comments	Sound source
Test 1	30.3	0	LM393: Zeros at digital output of the sensor	Silence
Test 2	70	600	LM393: Cumulative digital output over 10ms of sampling time (MAX)	One person speaking Medium sound
Test 3	90	900	LM393: Cumulative digital output over 10ms of sampling time (MAX)	One person speaking Loud sound
Test 4	97	1000	LM393: Continuous digital output values above 1000	Vacuum cleaner

The four measured points in the table allow extracting the two fitting parameters through linear regression using Excel sheet. The two parameters are then used to convert the sensor's digital readings into decibel values.

The equation derived from the plotted curve: y = 0.67x + 30.223

The threshold is set to be 90 dB over the sampling time of 10s. Anything above 90 dB is considered noisy, and anything below 90 dB is considered acceptable.

A detailed explanation of the code can be found as a video tutorial here = LM393 Explained

The arduino code itself can be found here = Sound detection with the LM393 Sensor

Air

For air quality detections, carbon dioxide gas concentration was mainly measured using the MQ-135 sensor. The sensor's output pin was connected to the A0 pin of the microcontroller(the analog values were be fed via this pin). The voltage output of the sensor is connected to the 5V pin of the microcontroller via the breadboard. To complete the circuit, the MQ-135 ground pin was connected to a 10kΩ resistor(RLOAD), which was connected to the microcontroller ground pin.

As the output device, an LED bulb and an LCD screen are connected to the microcontroller. To glow when the CO_2 ppm crosses a threshold value and displays the ppm values, respectively.

The MQ-135 sensor detects the air quality and does not differentiate between the types of gases in the air. So to be able to detect CO_2 specifically, we need to find the parameters for CO2 from the calibration graph as shown in figure 8.

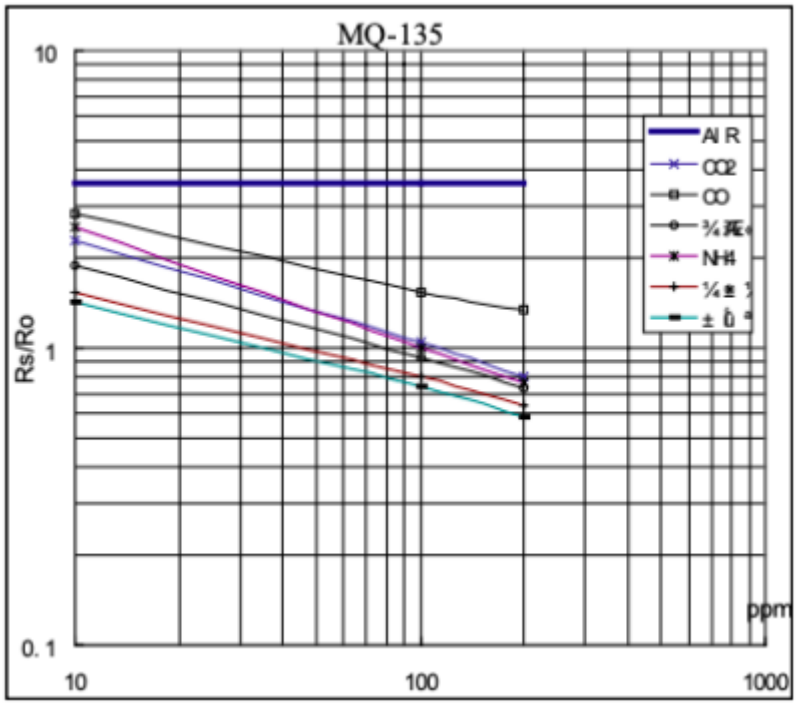

Figure 8: Calibaration graph for Mq-135

We can conclude from this graph and several other reports that the parameters *a* and *b* values for CO_2 are 116.6 and 2.76, respectively. The graph is basically the resistor ratio vs. the ppm value. This ratio is used since the MQ-135 does not give us the ppm value directly. Resistor ratio = Rzero/Rload. With this, we come to the next important parameter that is the Rzero, is the resistance of the sensor in the presence of clean air, i.e. in 400ppm of CO_2. This property changes for each resistor. A separate code was used to obtain the Rzero value, with the help of the MQ135.h library. And for this value to stabilize, the MQ-135 needs to be heated up for some time. After which, the Rzero value was updated in the MQ135.h library. Knowing these variables, the ppm of CO_2 was obtained from the sensor. Detailed information of which is given in the below hyperlinks.

A detailed explanation of the code can be found as a video tutorial here = Video Tutorial

The arduino code itself can be found here = Measuring Carbon Dioxide with MQ-135 sensor

Additional Libraries required

- The MQ135.h library is uploaded to the code using the Library Manager Tool to communicate with the MQ135 air quality sensor.
- LiquidCrystal_I2C library is uploaded to the code to communicate with the LCD screen.

5. Tests and Results

A detailed explanation of the location and a glimpse of the test results taken can be seen here under Measurements taken

1. Meisenweg, Kamp-Lintfort (Temperature = 22°C, Humidity = 20%)

At 19:00, the measurement is taken with open windows in a decently calm, clean residential neighborhood.

- **Using the MQ135 Air Quality Sensor:**

First the Rzero code was used to determine the value, which turned out to be 62. 42 Ohm. Then the value was fed to the main code for detecting CO_2 ppm levels. Using Python, the mean value of CO_2 levels was calculated from the 50 values taken. Mean CO_2 = 443.28 ppm, which is below the threshold value (443.28 ppm < 999 ppm). Hence, it is a safe environment.

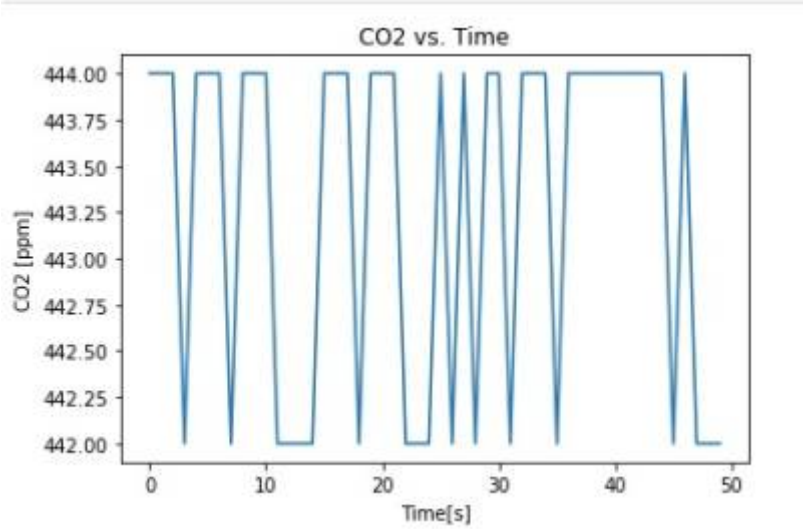

Figure 9: Screenshot of sample CO_2 levels in ppm using Python. Location: Meisenweg, Kamp-Lintfort.

- **Using the LM393 Sound Sensor:**

The values were fed into the main code. The measurement of the sound intensity = 30 dB constant if no one is talking indoors. The neighborhood here is almost always completely silent; i.e., you can not hear any loud noises.

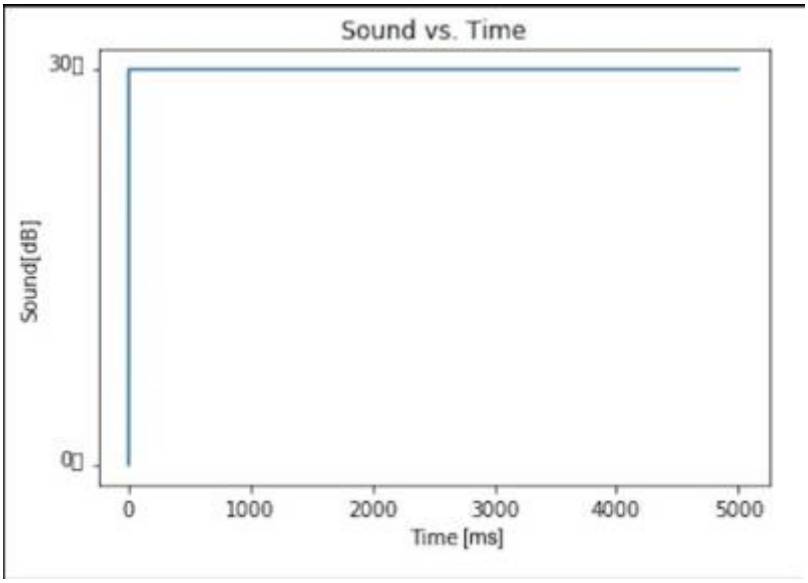

Figure 10: Screenshot of the sound intensity level in decibel using Python. Location: Meisenweg, Kamp-Lintfort

2. Moerserstraße, Kamp-Lintfort(Temperature = 16°C, Humidity = 60%)

At 15:40, the measurement is taken on the balcony next to the main road with a lot of traffic.

- **Using the MQ135 Air Quality Sensor:**

First the Rzero code was used to determine the value, which turned out to be 76 Ohm. Then the value was fed to the main code for detecting CO_2 ppm levels. Using Python, the mean value of CO2 levels was calculated from the 50 values taken. Mean CO2 = 401.98 ppm, which is below the threshold value (401.98 ppm < 999 ppm). Hence, it is a safe environment.

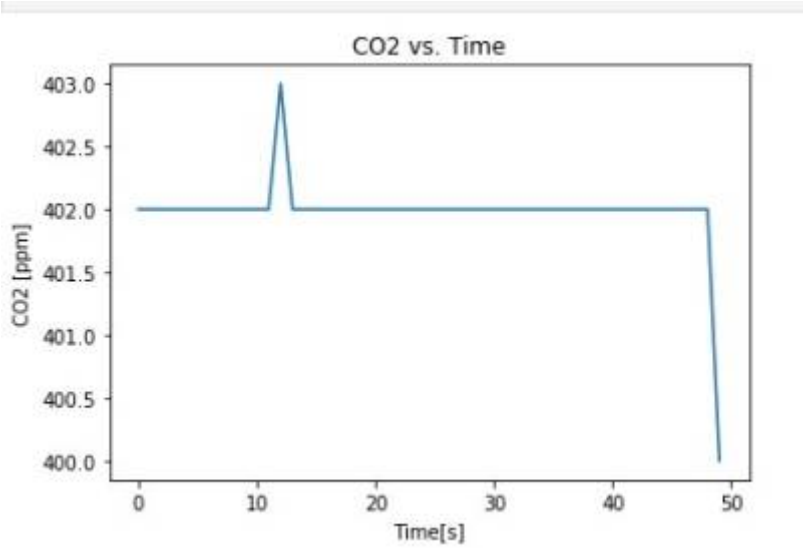

Figure 11: Screenshot of sample CO2 levels in ppm using Python. Location: Moerserstraße, Kamp-Lintfort.

- **Using the LM393 Sound Sensor**:

The values were fed into the main code.The measurement of sound intensity spiked up to 53 dB when sound of some bullets(cruiser bikes) was heard. Nevertheless, the sound intensity value is still within the safe range, i.e., below 90 dB.

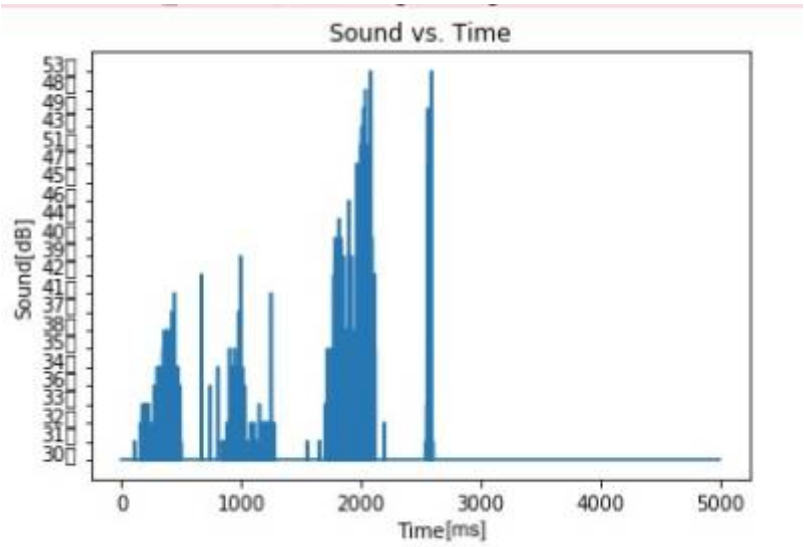

Figure 12:Screenshot of the sound intensity level in decibel using Python. Location: Moerserstraße, Kamp-Lintfort

3. Kamperdickstraße, Kamp-Lintfort (Temperature = 16°C, Humidity = 60%)

At 13:30, the measurement is taken on the balcony next to the main road and Postbank with some traffic.

- **Using the MQ135 Air Quality Sensor:**

First the Rzero code was used to determine the value, which turned out to be 76 Ohm. Then the value was fed to the main code for detecting CO_2 ppm levels. Using Python, the mean value of CO2 levels was calculated from the 50 values taken. Mean CO2 = 411.48 ppm, which is below the threshold value (411.48 ppm < 999 ppm). Hence, it is a safe environment.

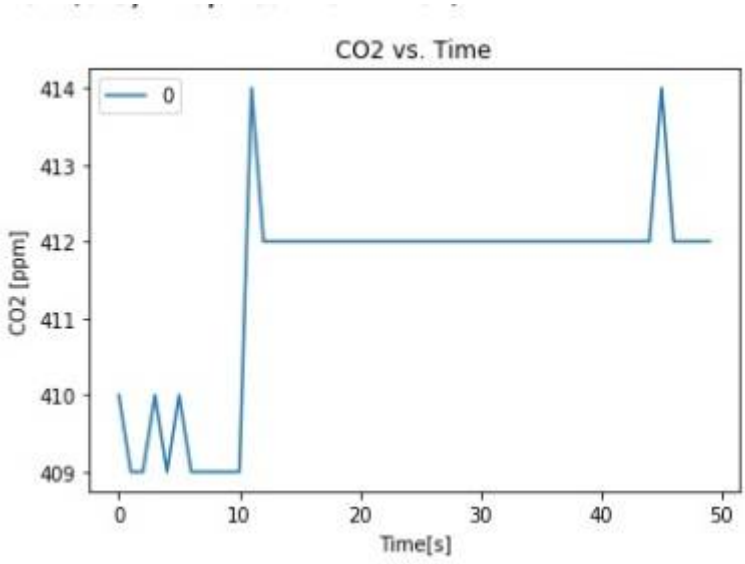

Figure 13: Screenshot of sample CO2 levels in ppm using Python. Location: Kamperdickstraße, Kamp-Lintfort.

- **Using the LM393 Sound Sensor:**

The values were fed into the main code.The measurement of the sound intensity = 30 dB constant, as there was no traffic passing by at the moment of measurement. Hence, it is rather calm street with sound intensity < 90 dB.

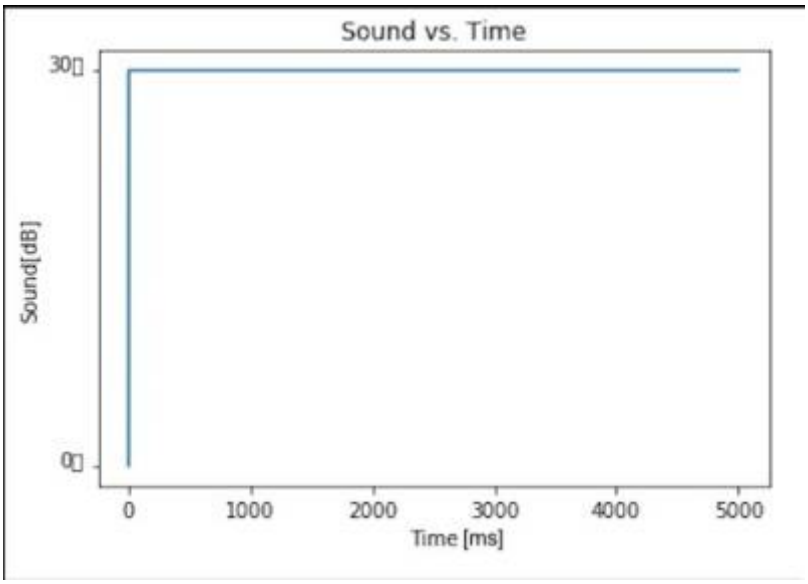

Figure 14: Screenshot of the sound intensity level in decibel using Python. Location: Kamperdickstraße, Kamp-Lintfort

6. Discussion and Conclusion

The LM393 sensor narrowly detected enough noise in a minimal range at one particular moment. Since it had only digital outputs (zeros or ones), selecting a specific sample time interval and collecting the digital results over that period helped obtain numerical readings to compare them between the three different locations. Additionally, to acquire a legitimate unit out of those raw readings, the application “Schall” calibration was a very decent option to do so and receive roughly the sound intensity values in a decibel unit.
Considering the precision limitation, both Meisenweg and Kamperdickstraße showed the lowest sound intensity levels with only 30 dB detected over the time interval. Moerserstraße came second with around 54 dB sound intensity. However, all locations checked the criteria of having noise below the threshold value (90 dB), which means the three environments are suitable for the student's life.
The MQ135 sensor was able to produce analog values in the three locations, giving more valid values that continuously can vary over time. With the help of the MQ135 library, CO_2 gas was selected as a parameter and used for calibration in order to convert those analog signals to concentration values of the unit particles per million (ppm). Since the MQ135 sensor is more sensitive, the results could be more reliable than those generated by the LM393 sound sensor. Therefore, the comparison between the three measurements is well-justified. The air quality in Moerserstraße was the best with a mean value of 401.98 CO_2 ppm, followed by the one in Kamperdickstraße with a mean value of 411.48 CO_2 ppm, and last is With relatively higher CO_2 concentration in the air corresponding to a mean value of 443.28 ppm.
For a clearer comparison, we used Python to find out the ratio of the mean measurement to the standard clean air ppm was calculated for the three locations. The concentration of standard clean air is chosen to be 400 ppm.

Table 2: The mean measurement-to-standard clean air ratio at the three locations:

Location	1	2	3
Mean[ppm]	443.28	401.98	411.48
Ratio(Mean/400ppm)	1.1082	1.00495	1.0287

From table 2, the closest condition to the standard clean air condition is location 2, having the nearest value to 1, whereas the furthest would be location 1, having the furthest ratio from 1. Nevertheless, the variation was barely detectable, as the three locations showed good air quality under the threshold value set. Hence, those minor differences may be negligible when deciding on a place to live healthily with clean air quality.

Limitations to the experiment: 1. The sensor LM393 was not as sensitive to sounds as expected. An example which can be seen in location 2, albeit being the busiest street, the LM393 could only detect loud sounds like that of a truck's engine or a cruiser bike starting. A better sensor can be used in place of this to make a more accurate reading.

2. CO_2 readings taken were justified given the location of the three places, where there were more green patches, a higher concentration of CO_2 that was detected. In order to assess the pollution levels better, a different sensor could have been used to detect a more prominent pollutant, for e.g, Carbon Monoxide or Nitrogen Dioxide. Though when this was tried with the MQ-135 sensor, the values received were strangely off the charts, making them unreliable.

3. The MQ-135 needed to heat up for over 24 hours to get a stable value, due to time limitations of having to test in three different locations, the time to heat up was not achieved. So only a few hours of heating up could possibly cause the sensor to get odd values.

4. Test results should have been taken at the same time, in order to overcome the limitation of difference in temperature and humidity conditions.

7. References

1. L. Petry, H. Herold, G. Meinel, T. Meiers , I. Müller , E. Kalusche , T. Erbertseder , H. Taubenböck 2, E. Zaunseder, V. Srinivasan, A. Osman, B. Weber, S. Jäger, C. Mayer, C. Gengenbach, “AIR QUALITY MONITORING AND DATA MANAGEMENT IN GERMANY – STATUS QUO AND SUGGESTIONS FOR IMPROVEMENT”, The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume XLIV-4/W2-2020, 2020 5th International Conference on Smart Data and Smart Cities, 30 September – 2 October 2020, Nice, France, 7 pages. Accessed on 20th Aug 2021:
2. https://www.electronicoscaldas.com/datasheet/MQ-135_Hanwei.pdf
3. https://microcontrollerslab.com/interfacing-mq-135-gas-sensor-arduino/

1)

1)

Main Page