Sound Localization System

Bíró Mátyás-Péter
Student, Technical University of Cluj-Napoca, Computer Science (2021-2025)
E-mail: biromp6@proton.me

Soós Dávid
Student, Technical University of Cluj-Napoca, Computer Science (2021-2025)
E-mail: soos.sa.david@student.utcluj.ro

Sprencz Róbert
Student, Technical University of Cluj-Napoca, Computer Science (2021-2025)
E-mail: sprencz.zs.robert@student.utcluj.ro

Szabó Loránd
Student, Technical University of Cluj-Napoca, Computer Science (2021-2025)
E-mail: szabo.zo.lorand@student.utcluj.ro

Abstract. The purpose of our work was to create a system that can locate the origin of a sound signal. It is assumed that there is a single sound source during the localization process. This implementation uses an Arduino Uno board and R.22112 microphones.

1. Introduction

In the following paper we will discuss an implementation of a Sound Localization System, together with observations, ideas, and experiences. We also share our data and the assembly method of our circuit.

The assembly follows a simple logic: we use four microphones as sensors and four LEDs as actuators, all of them being connected to the main Arduino Uno board. To be able to monitor the data read from the sensors, the Arduino maintains a serial connection to a computer through a USB-port.

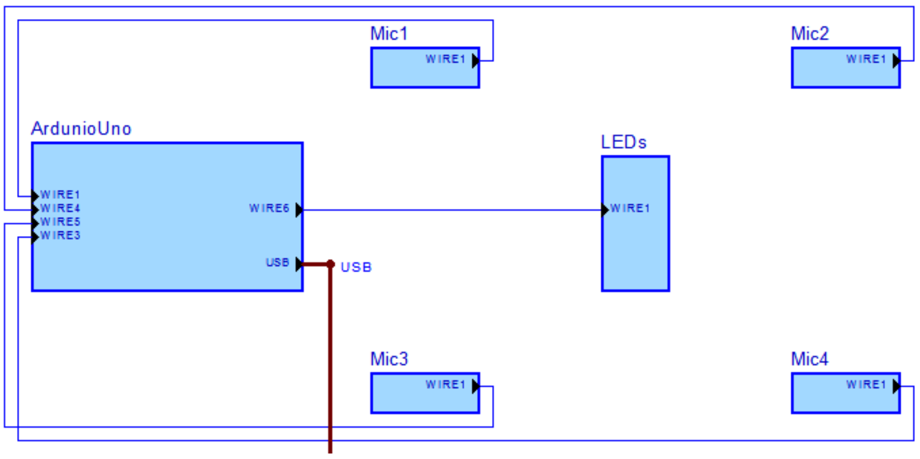


Fig. 1.1 Block Diagram

During each data read cycle, the Arduino measures the average voltage returned by each microphone for a given period and sends this information to the computer. The built-in serial plotter from the Arduino IDE plots these results, letting us obtain a live feed of the sound intensities. This data is later processed to determine the location of the sound source.

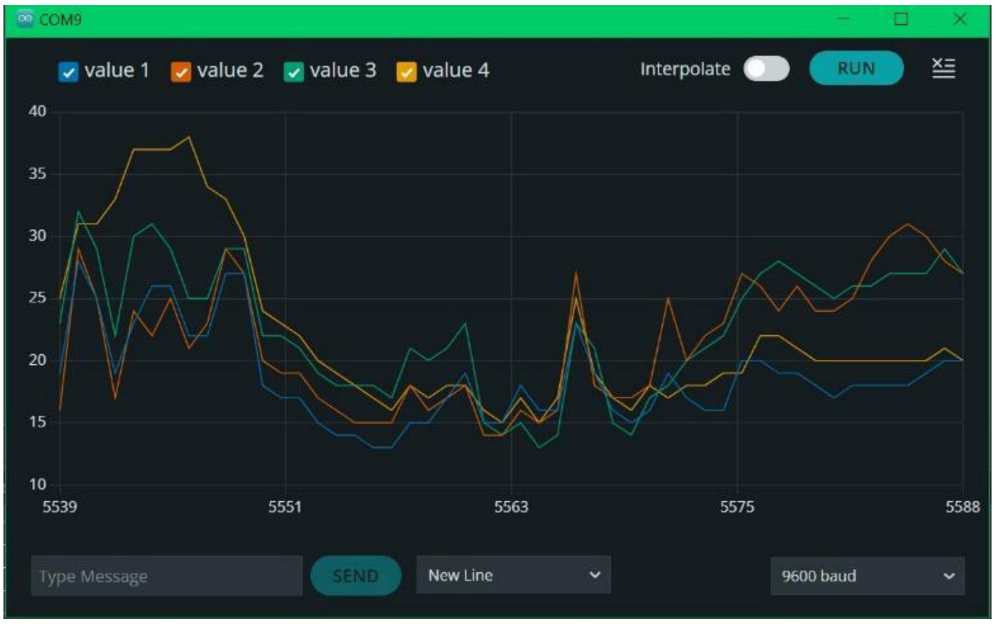


Fig. 1.2 Sound intensity measured by the sensors

2. Hardware

The required hardware elements consist of:

- 1 x Arduino Uno – *Figure 2.*
- 1 x USB A – USB B cable – *Figure 2.2*
- 4 x 22112 microphone modules – *Figure 2.3*
- 4 x LEDs, together with resistors – *Figure 2.4*
- 1 x breadboard for prototyping
- jumper wires
- a computer for the programming of the Arduino board

Fig. 2.1

Fig. 2.2

Fig. 2.3

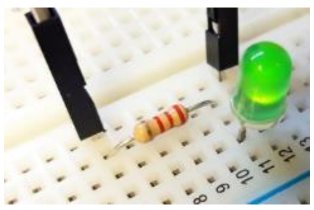


Fig. 2.4

3. Preparation and Configuration

The process of creating a Sound Localization System consists of the following steps:

1. Install Arduino IDE
2. Connect the microphones and the LEDs to the Arduino board
3. Connect the Arduino to the computer
4. Upload the code to the Arduino and run it
    - During this stage, the measured volumes can be read from the display of the computer
    - The position of the sound source can be determined graphically using the provided MATLAB code

4. Applications

Sound localization systems have a wide range of practical applications, including:

1. Sound source separation
2. Sonar (navigation and ranging using sound waves)
3. Wildlife localization
4. Gunshot detection
5. 3D sound localization applied in robotic hearing

5. Physical approach

5.1. Scientific background

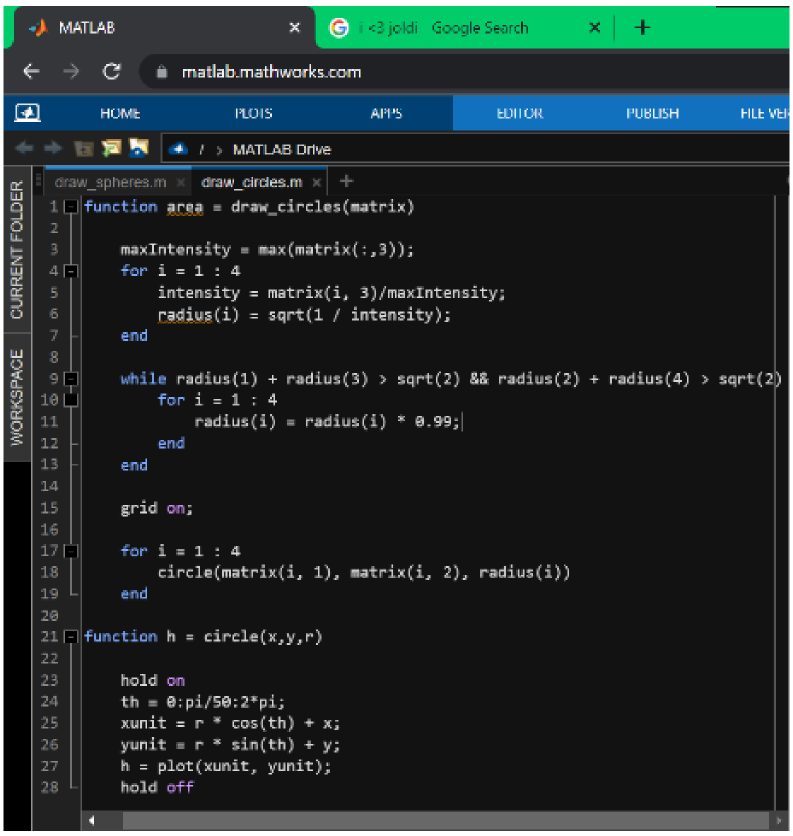
Sound waves propagate through any medium if there exists a source generating them. As the waves travel further from the source, their energy is spread on an even increasing surface. Thus, their intensity decreases depending on the distance. This property can be expressed using an inverse square law:

$$I = \frac{P_s}{4\,\pi d^2}$$

The symbols in the above equation represent:

- P_S – the acoustic power of the sound source
- d – the distance from the source to the destination

Due to this property, we can calculate the distances between the source and the sensors, allowing us to locate it.



```
function area = draw_circles(matrix)

    maxIntensity = max(matrix(:,3));
    for i = 1 : 4
        intensity = matrix(i, 3)/maxIntensity;
        radius(i) = sqrt(1 / intensity);
    end

    while radius(1) + radius(3) > sqrt(2) && radius(2) + radius(4) > sqrt(2)
        for i = 1 : 4
            radius(i) = radius(i) * 0.99;
        end
    end

    grid on;

    for i = 1 : 4
        circle(matrix(i, 1), matrix(i, 2), radius(i))
    end

function h = circle(x,y,r)

    hold on
    th = 0:pi/50:2*pi;
    xunit = r * cos(th) + x;
    yunit = r * sin(th) + y;
    h = plot(xunit, yunit);
    hold off
```

Fig. 5.1 Partial MATLAB code

5.2. Sensors and measurement methodology

First, the sensors' sensitivity is adjusted so that their outputs are equal when the sound source is located exactly in the middle of the map. This is done in order to avoid the offsets induced by different settings present on the microphones.

Once the system is initialized, it enters into a loop, endlessly performing the following steps:

1. Read the voltages returned by the R.22112 microphones
2. Send the measured results to the computer
3. Set the LED assigned to the microphone being the closest to the source ON

Processing the results requires more computational power, therefore it is executed on the computer:

1. Assume that the highest obtained value represents the intensity of the sound source
2. Calculate the distances using the formula from *Section 5.1*
3. Scale down the results until the smallest probable nonzero area for the location of the sound source is found
4. Display a simplified map containing the results

The above algorithm represents a simplified version of the actual algorithm, for the better understanding of the main ideas involved.

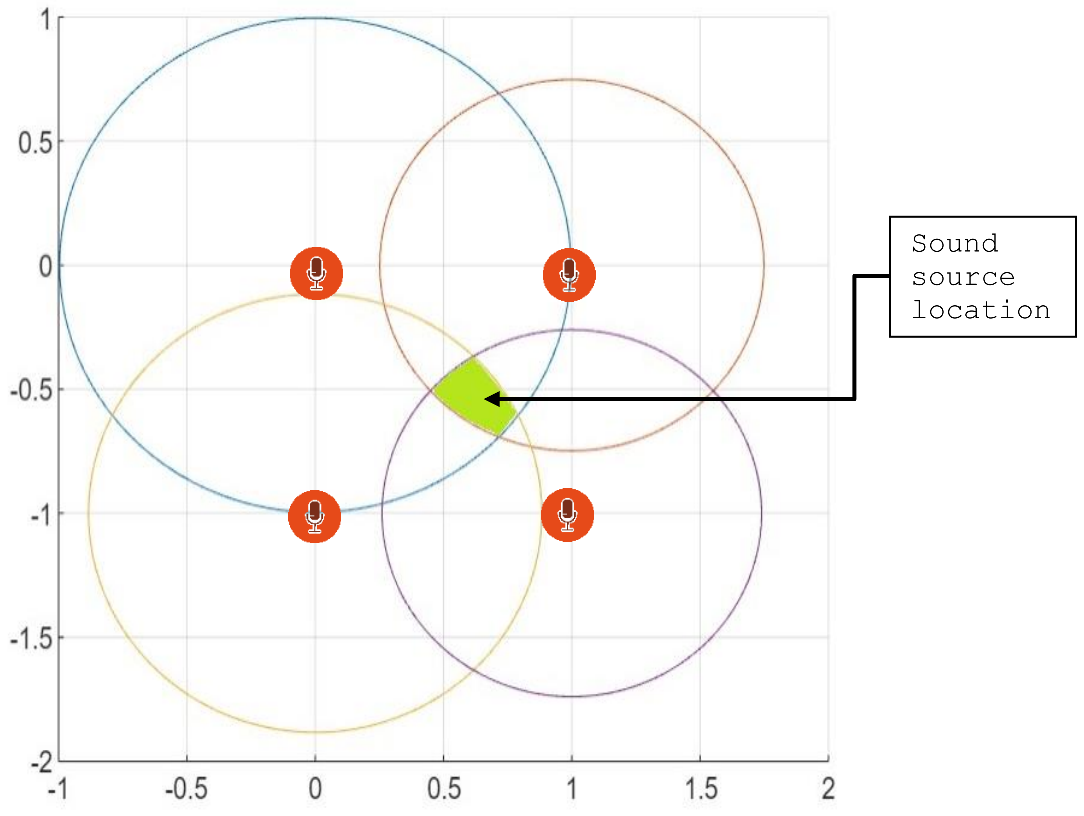


Fig. 5.2 Graphical representation of the results

6. Further development

The presented system is able to instantly determine which microphone is situated the closest to the sound source, displaying this information on the four LEDs assigned to the sensors. However, to obtain the graphical representation, the received data is fed manually into the MATLAB module so that it can process it accordingly.

Given the efficiency, precision, and measurement speed of this system, we recommend the following improvement to be implemented:

- Use a device set with a higher sample rate
  → i.e., finer microphones with a more advanced microcontroller board
- Automate the data transfer between the Arduino and the MATLAB module
  → Real-time visual feedback of the measurements
- Use more microphones
  → Increased precision
- Take into consideration the Time of Arrival for the peaks in the analysed sound
  → The microphones can be positioned further away from the source

7. Bibliography

[1] https://en.wikipedia.org/wiki/Sound_localization
[2] https://www.sigmanortec.ro/Modul-microfon-senzor-sunet-p126025149
[3] https://pressbooks.pub/sound/chapter/intensity-and-distance-april-2019-version/
[4] https://www.youtube.com/watch?v=6ZQYALZqZNQ

8. Appendix

This section provides information for the better understanding of this work.

8.1. The code

The source code of the final setup is as follows:

```
int sensorPin1 = A0;
int sensorPin2 = A1;
int sensorPin3 = A2;
int sensorPin4 = A3;
int micValue = 0;
int period = 100;
int LED1 = 6;
int LED2 = 7;
int LED3 = 8;
int LED4 = 9;
const int threshold = 20;

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);

  pinMode(sensorPin1, INPUT);
  pinMode(sensorPin2, INPUT);
  pinMode(sensorPin3, INPUT);
  pinMode(sensorPin4, INPUT);

  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
  pinMode(LED3, OUTPUT);
  pinMode(LED4, OUTPUT);
}

char* leds(int par1, int par2, int par3, int
par4) {
  if (par1 > par2 && par1 > par3
      && par1 > par4 && par1>threshold) {
    return "1000";
  }
  else if (par2 > par1 && par2 > par3
      && par2 > par4 && par2>threshold) {
    return "0100";
  }
  else if (par3 > par1 && par3 > par2
    && par3 > par4 && par3>threshold) {
    return "0010";
  }
  else if (par4 > par1 && par4 > par2
    && par4 > par3 && par4>threshold) {
    return "0001";
  }
  else {
    return "0000";
  }
}

void handleLeds(char ledsArray[4]) {
  if (ledsArray == "1000") {
    digitalWrite(LED1, HIGH);
  }
  else if (ledsArray == "0100") {
    digitalWrite(LED2, HIGH);
  }
  else if (ledsArray == "0010") {
    digitalWrite(LED3, HIGH);
  }
  else if (ledsArray == "0001") {
    digitalWrite(LED4, HIGH);
  }
  delay(10);

  digitalWrite(LED1, LOW);
  digitalWrite(LED2, LOW);
  digitalWrite(LED3, LOW);
  digitalWrite(LED4, LOW);
}

void loop() {

  int sum1 = 0, sumPrev1 = 0;
  int sum2 = 0, sumPrev2 = 0;
  int sum3 = 0, sumPrev3 = 0;
  int sum4 = 0, sumPrev4 = 0;
  for(int i = 0; i < period; i++){
      sum1 += analogRead(sensorPin1);
      sum2 += analogRead(sensorPin2);
      sum3 += analogRead(sensorPin3);
      sum4 += analogRead(sensorPin4);
  }

  sum1 /= period;
  sum2 /= period;
  sum3 /= period;
  sum4 /= period;

  Serial.print(sum1);
  Serial.print(",");
  Serial.print(sum2);
  Serial.print(",");
  Serial.print(sum3);
  Serial.print(",");
  Serial.print(sum4);
  Serial.print(",");


  char* ledsArray = leds(sum1, sum2, sum3,
sum4);
  handleLeds(ledsArray);

  Serial.println("");
  //delay(10);

  sumPrev1 = sum1;
  sumPrev2 = sum2;
  sumPrev3 = sum3;
  sumPrev4 = sum4;
}
```

8.2. The final setup

Connecting the components together, we obtained the following setup:

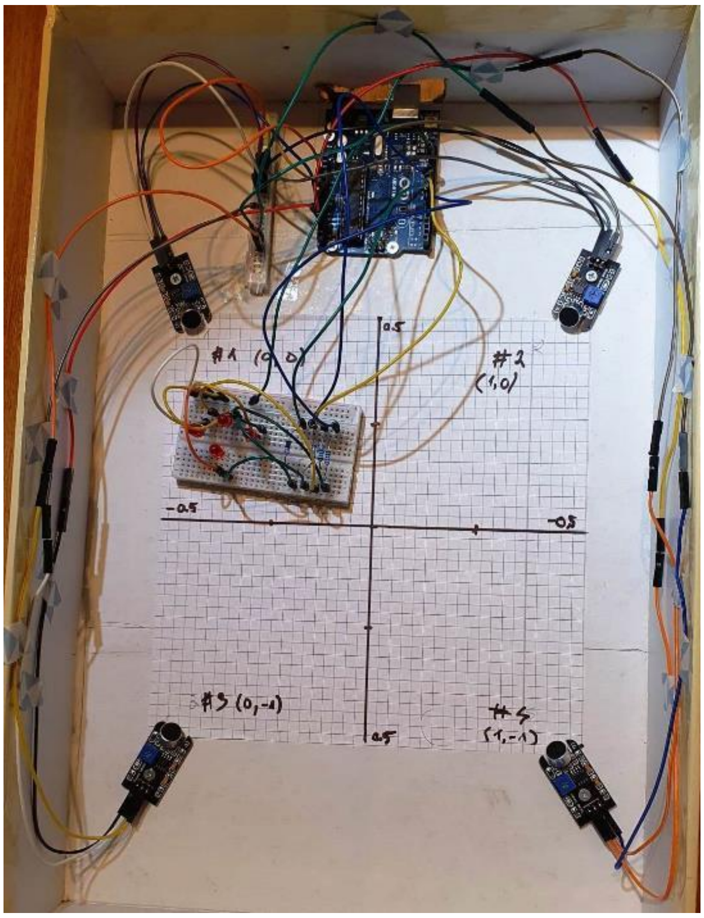


Fig. 8.2 The final setup of the Sound Localization System