EarSaver: A device to detect dangerous audio noises

Lorenzo De Lauretis, Tiziano Lombardi and **Stefania Costantini** [1]

Abstract. The issue of noise pollution is becoming more and more relevant in our today's way of life. Studies have shown that some noise waves are especially damaging, triggering continuous harm to the nervous scheme with the resulting failure of listening capacity in some instances. Thanks to the latest technological findings, noises can be sampled and analyzed even on very tiny appliances that can possibly be carried anywhere. By testing the noise via a condenser microphone and evaluating the outcome of applying the Fast Fourier Transform to the sampled samples, we can identify the existence of frequencies that are considered detrimental to the auditory system, warning a person in real-time about the prospective risk to which (s)he is facing.

1 INTRODUCTION

Living in our culture today implies being continually surrounded by the voices and vibrations generated by the most disparate causes like smartphones, vehicles, automobiles, aerial transport, etc. Due to population growth and urbanization, this "noise pollution" will increase [3]. Noises should not be hazardous to the human ear in small doses and low volume, but they can be hazardous if they increase in volume and density. According to recent research, with long-lasting exposition to certain sound thresholds, there are specific sounds that can trigger serious hearing loss in rats for several days; these sounds have a frequency of 16kHz and stress of 115dB. There are also worst noises, with a frequency of 4kHz with a pressure of 125 dB, that cause permanent damage to the hearing of rats in both the low- and high-frequency range [7]. Sounds at so high pressure can be dangerous for the human ear too [4].

To advice the user of the occurrence of these dangerous noises, we built a very small device, called EarSaver. It samples sounds through a condenser microphone and analyzes the result of the application of the Fast Fourier Transform (FFT) to the sampled signals, thus detecting dangerous noises. When dangerous noises are detected, a led installed on EarSaver starts blinking, so as to warn the user of the potentially harmful noise. We tested the device with different noises at different frequencies/pressures, obtaining satisfactory results. This work is part of the eHealth project [2].

The paper is structured as follows. In Section 2 there is an excursus about similar works related to this topic. Section 3 presents background knowledge needed to better understand the work we have done. Section 4 describes our prototype, explaining the architecture, the firmware and their functionalities. Sections 5 and 6 discuss the testing phase and the achieved results. Then, Section 7 reports our conclusions based on data obtained using our prototype.

[1] Department of Information Engineering, Computer Science and Mathematics, University of L'Aquila, Italy, Email: lorenzo.delauretis@graduate.univaq.it, tiziano.lombardi@graduate.univaq.it, stefania.costantini@univaq.it

2 RELATED WORK

In [6], Rajagukguk et al. sampled the sound using a condenser microphone, analyzing the data using an Arduino Uno. They used an LCD display, a buzzer and LEDs to display the information about the danger level of the analyzed sound. Their device warns the user when the sound reaches the pressure of 75dB, considered by them harmful for the human ear. Our device is more precise, because it does not keep track only of the sound level, but also of the sound frequencies.

In [1], Bianchi et. al. explored the use of the Arduino platform, that can be a versatile audio processor. They treated the real-time signal processing, with particular emphasis on the FFT, realizing a sort of benchmark to discover the maximum length of an FFT that can be computed in real-time inside an Arduino. Their work is more about the operations that Arduino can make on audio, with the related benchmarks, while we focus more on the practical application of the results of FFT computation to prevent hearing loss.

In [8], Silva et. al. made a study on digital sound processing using Arduino and Matlab[2], using two different approaches: in the first one, all the computation is done on the Arduino, in the second one the sound samples are instead sent to a laptop where Matlab software performs the computations, before sending back to playback on the Arduino. Their work, differently from ours, is partly implemented on the Arduino and partly on a PC with complex sound analysis. Instead, we focus only on the Arduino without the need of the PC, thus devising a fully portable device.

3 BACKGROUND

This section discusses some heritage notions about Arduino and Fast Fourier Transform approach.

3.1 INTERNET OF MEDICAL THINGS

The Internet of Medical Things (also called the internet of health things) is an extension of the Internet of things (IoT)[3]. It is the specific field of application for medical and health purposes, such as data collection and analysis for research and monitoring. This 'Smart Healthcare' infrastructure facilitates the creation of digitized healthcare systems, connecting available medical resources and healthcare services. IoT healthcare systems manage chronic diseases and their prevention and control. The connectivity for remote monitoring enables health practitioners to capture data of the patients and applying complex algorithms in health data analysis.

[2] https://it.mathworks.com/products/matlab.html
[3] https://en.wikipedia.org/wiki/Internet_of_things

3.2 ARDUINO

Arduino[4] is an open-source hardware and software company that designs and manufactures single-board microcontrollers and microcontroller kits. They are used for building or prototyping digital devices and interactive objects, which make use of various sensors and actuators. Boards are equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various hardware components.

Arduino boards can be easily programmed using an Integrated Development Environment (IDE)[5] and a USB cable. A typical Arduino program is composed of two main functions:

- setup: it is the initialization phase of the board
- loop: it is a portion of code executed repeatedly.

3.3 FAST FOURIER TRANSFORM

The Fast Fourier transform (FFT[6]) algorithm is a method for computing the Finite Fourier transform of a series of N (complex) data points in approximately $N \log_2 N$ operations.

The Fast Fourier Transform can be useful in many fields, among which the one of our interest: sound analysis. In fact, it can be used to detect Speech Spectrogram [5] from the original waveform signal. Another use case is the Automatic Indexing of Musical Sounds [9] through their timbre recognition.

4 ARCHITECTURE AND FIRMWARE

4.1 SYSTEM STRUCTURE

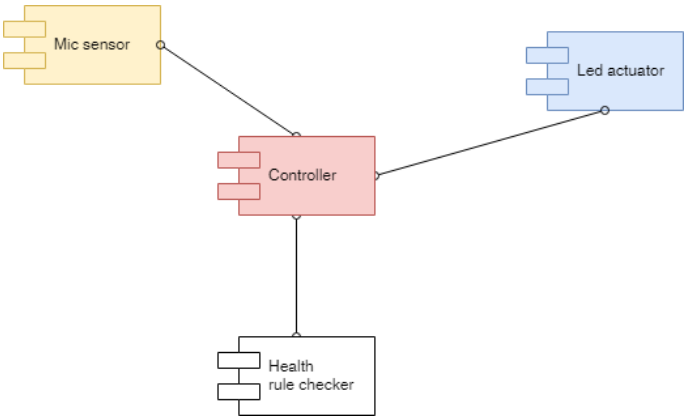

Figure 1. EarSaver architecture

This section discusses the architecture of the system using an UML Diagram Component in order to show relations among different stages composing our system. As shown in Figure 1, EarSaver is composed of five macro-components:

- Controller: the "brain" of EarSaver, it has the Firmware core inside. It receives input data from the Microphone Sensor and, when dangerous noises are revealed, it triggers the Alert LED.
- Health rule checker: it contains all the mathematical rules applied to collected microphone data in order to reveal dangerous situations.
- LED actuator: it is the component designed to drive an alert LED; when triggered it changes its state, from on to off and vice-versa.
- Mic sensor: is a sound sensor, which detects audio from the ambient and transmits data samples to the Controller component.

The presented Component architecture is designed to be Hardware independent, in the sense that the same designed system can be implemented also in different platform apart from the chosen Arduino framework; which means, it could be used also in other critical applications, such as the detection of ultrasonic-frequencies in a medical environment or very low frequencies in sonar applications (i.e. resonant testing of buildings).

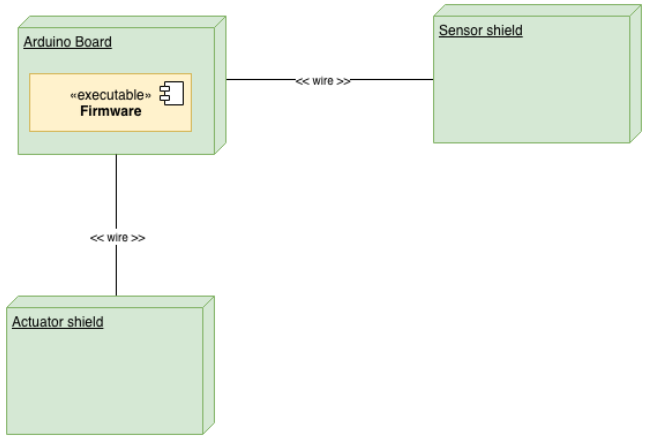

Figure 2. EarSaver Hardware Deployment

Figure 2 shows an example of the deployment of the system, putting in evidence the hardware structure and communication. In our Arduino example, we can see three hardware boards wired together. In particular:

- Arduino Board: the core board with the main controller installed into it. Here is where the main firmware runs.
- Sensor Shield: input peripheral board where sensors are installed (microphone in our case). This board collects data from the environment and sends them to the main controller.
- Actuator Shield output peripheral board, where actuators are installed (the LED in our case). This board is interfaced to the user.

4.2 SYSTEM BEHAVIOR

This system is characterized by three running states during its execution, related to the danger index.

- Normal: is the idle state in which the environment is in a safe state and nothing is prompted to the user.
- Danger: the environment is exiting the safe state for the presence of some dangerous frequency. The user is softly prompted of that.
- Critical: the environment is totally unsafe for the presence of dangerous frequencies at a high-pressure level. The user is strongly alerted.

When entering a state, the system properly sets its actuators to inform the user about environmental changes. Specific conditions allow the system to change state in a scale of danger; moreover, the system can return in a normal state if a critical situations end.

[4] https://en.wikipedia.org/wiki/Arduino
[5] https://www.arduino.cc/en/Main/Software
[6] https://en.wikipedia.org/wiki/Fast_Fourier_transform

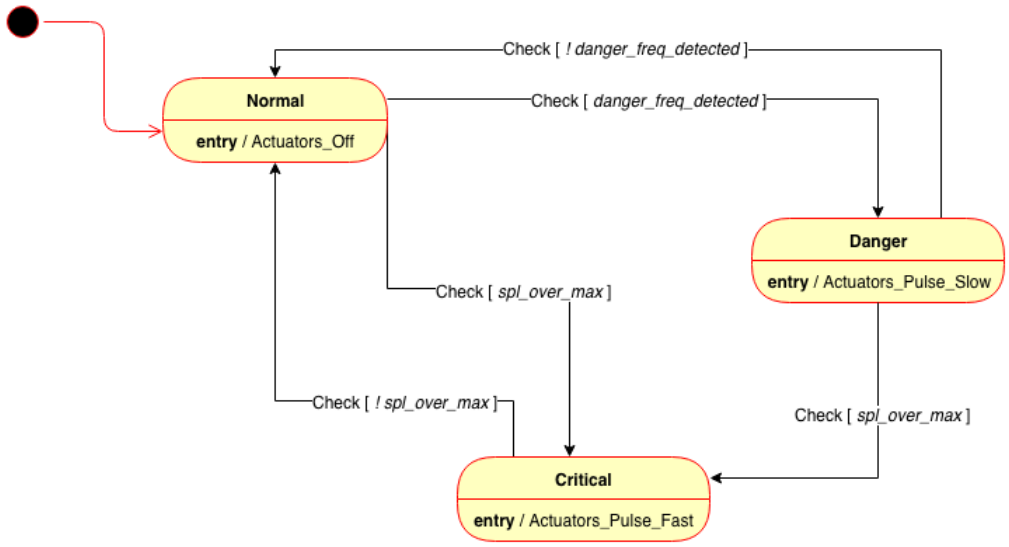

Figure 3. EarSaver Behavior evolution

Algorithm 1 FFT Implemented on Arduino

```
set sampling_max_frequency
set number_of_samples

set dangerous_frequencies_array
set dangerous_common_SPL
set dangerous_SPL

Loop {
 read number_of_samples in range (0,
        sampling_max_frequency)
           -> sample_array

 do FFT on sample_array -> FFT_array

 for each FFT_elem in FFT_array {
  if FFT_elem[SPL]>=dangerous_common_SPL{
   alert
  }
  for each dangerous_frequency in
           dangerous_frequencies_array {
   if FFT_elem[frequency] ==
            dangerous_frequency {
    if FFT_elem[SPL] >= dangerous_SPL {
     alert
    }
   }
  }
 }
}
```

In this first prototype, the state machine in Figure 3 realizes the Arduino Firmware, which allows EarSaver to be reliable and functional. It relies on the usage of the FFT algorithm, which is a polynomial approximation of the Fourier Transformation method. This method is parametrized on two parameters: the sampling maximum frequency, which determines the maximum detectable frequency; the number of samples to be stored and then analyzed, which describes our implementation is shown in Algorithm 1.

At the beginning of the code, it is possible to see the declaration of the two parameters cited above.

- Sampling maximum frequency: it is chosen high enough to catch all wanted frequencies with respect to hardware capabilities.
- The number of samples: it is chosen large enough to have a certain number of samples which can emphasize the characteristic of the audio signal, accordingly with hardware capabilities.

After the declaration of the two parameters described above, there is the declaration of global variables to describe the samples' list of data and the initialization of our algorithm variables. In the Loop method, there is the cyclic core of the algorithm. First of all, it collects the chosen number of samples, storing them in dedicated memory locations. Then Fast Fourier Transform is applied on the set of data, identifying the spectrum characteristic of the detected audio signal, which can be compared with the target to possibly trigger an alert, which turns on/off the output LED of the circuit. This algorithm is iterated every second in order to have real-time continuous monitoring of the environment.

5 DETECTING MALICIOUS FREQUENCIES

The main objective of our device is to warn the user of potentially harmful noises that are in her/his surrounding, helping them to prevent hearing loss. To do that, we first have to understand which are the harmful noises, and what they can cause to the auditory system during short and long period expositions.

As discussed in [7], there are two main kinds of sound that are very dangerous for the auditory system of the rat: one is 16kHz at 115dB and one is 4kHz at 125dB. The exposition to the noise with 16kHz and 115dB can cause severe hearing loss for several days, while the exposition to the noise with 4kHz at 125dB can cause permanent damage to the auditory system of the rat. Also, as argued in [4], sounds at so high pressure are dangerous for the human auditory system too, and they should be avoided.

The noises at 115dB and 125dB are at very high pressure, so they can be identified by the human ear; the frequency recognition, instead, can be more difficult for the human ear, so we need a device that helps the user in recognizing also these potentially harmful frequencies. As the first step, we built our prototype with with Arduino as the microcontroller, as shown in the previous section. The result-

ing device can be used for audio sampling and analysis. The device is very small and is battery powered, so it is perfect to be carried around potentially everywhere.

As next step, we wrote the code that permits to the Arduino to work, written entirely in C++[7]: through this code, we were able to sample and analyze the audio surrounding our device and to make a led blink.

Our device helps the user in the recognition of frequency and pressure, blinking a led to alert the user when one of the following conditions are met:

- Analyzed sound is at a frequency of 4kHz or 16kHz and the pressure is greater equal 75dB.
- Analyzed sound has a pressure greater equal 100dB.

The led blinking frequency is proportional to the pressure of the analyzed noises. When the sound pressure is lower than 75dB, the led is switched off, when the sound pressure is greater equal 75dB it blinks at a 2Hz frequency, when the sound pressure increases even more, the led blinks with an increasing frequency, until it reaches the 100Hz blinking frequency, at the 125dB.

The device was tested using a wave generator, an object that is able to produce sounds at particular frequencies and pressures. We performed forty noises tests, with four different pressure levels: 60dB, 80dB, 90dB and 105dB, varying the frequency from 2kHz to 20kHz. The results of our tests are shown in Table 1: if the led blinks at the corresponding frequency and pressure, in the table is reported "Yes", "No" otherwise. We tested our device in a silent room, without external sounds, to prevent interferences given from them.

Table 1. Led blinking testing results

	60dB	80dB	90dB	105dB
2KHz	No	No	No	Yes
3KHz	No	No	No	Yes
4KHz	No	Yes	Yes	Yes
5KHz	No	No	No	Yes
6KHz	No	No	No	Yes
10KHz	No	No	No	Yes
14KHz	No	No	No	Yes
16KHz	No	Yes	Yes	Yes
18KHz	No	No	No	Yes
20KHz	No	No	No	Yes

6 DISCUSSION

This paper started with an introduction to Arduino and why it is so useful to cope with problems that are always more relevant nowadays. We choose Arduino for our project for a number of reasons: it is open-source, it is cheap, it is all-in-one and can be programmed using C++. Via the combination of all these features, we got a dedicated microcontroller that is easily expandable. The sensors and actuators are relatively cheap, permitting, with a very low budget, to build one's own devices. There are a lot of Arduino libraries available online, that permit rapid development of code for any device. Without Arduino and all its features, a work such as the one described in this paper would have been more complicated and, in some cases, it would have not been possible.

Within the data obtained from the testing, it is possible to verify that the devices works well, correctly identifying the potentially dangerous frequencies/pressures. When the noises produced from our wave generator had a pressure of 60dB, the led on our device had not blinked, because the analyzed sound was not potentially harmful. When we tested the device with noises with the pressure set at 80dB, the led blinked when the noises reached 4kHz and 16kHz frequencies, correctly advising the user of the analyzed harmful frequencies. When the noises produced from our wave generator had a pressure of 90dB, the led blinked when the noises reached 4kHz and 16kHz frequencies, correctly warning the user of the potential danger for his ears. When the noises had a pressure of 105dB, the led blinked within all the frequencies reached from our wave generator, because we consider dangerous noises at so high pressure. The led blinking frequency is proportional to the pressure of the analyzed noises.

A problem that is possible to identify in our system is the microphone sensibility, that, in some cases, cannot be enough to sample very high sound pressures: the microphone we used for testing is a consumer one, that can be bought from everybody to build own devices, and not a professional one; so, the precision is not so high for professional use; it may lead to a loss of precision during measurements. From the testing results, we can ensure that our device works correctly with respect to the workflow, properly identifying the potentially dangerous frequencies/pressures.

7 CONCLUSIONS AND FUTURE WORK

In this work, we created a tool that warns the user when potentially harmful noises are detected in its near surroundings. It is very small, meaning that it can be carried potentially everywhere. It is cheap and is easy to build, meaning that it can be constructed potentially by everybody. It is functional because it is able to sample and analyze the noises coming from the near surroundings, warning the user with the blink of a led if the analyzed noises are potentially harmful to them. In the near future, we will build another version of our system, with a better professional microphone, that will be able to sample the sounds with better precision, even at higher sound pressures. Using the design guidelines illustrated here, everybody can build its own system with a very low budget, safeguarding their own auditory system and protecting their ears from auditory diseases.

REFERENCES

[1] André J. Bianchi, 'Real time digital audio processing using arduino', (2013).

[2] Lorenzo De Lauretis, Stefania Costantini, and Ivan Letteri, 'An ontology to improve the first aid service quality', in *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, pp. 1479–1483. IEEE, (2019).

[3] Lisa Goines and Louis Hagler, 'Noise pollution: A modern plague', *Southern Medical Journal*, **100**(3), (2007).

[4] M. Maassen, W. Babisch, K. Bachmann, H. Ising, G. Lehnert, P. Plath, P. Plinkert, E. Rebentisch, G. Schuschke, M. Spreng, G. Stange, V. Struwe, and H. Zenner, 'Ear damage caused by leisure noise', *Noise and Health*, **4**(13), 1–16, (2001).

[5] A. V. Oppenheim, 'Speech spectrograms using the fast fourier transform', *IEEE Spectrum*, **7**(8), 57–62, (Aug 1970).

[6] Juniastel Rajagukguk and Nurdieni Eka Sari, 'Detection system of sound noise level (snl) based on condenser microphone sensor', *Journal of Physics: Conference Series*, **970**(1), 012025, (2018).

[7] Amanda C. Reed, Tracy M. Centanni, Michael S. Borland, Chanel J. Matney, Crystal T. Engineer, and Michael P. Kilgard, 'Behavioral and neural discrimination of speech sounds after moderate or intense noise exposure in rats', *Ear Hear*, **35**(6), e248–e261, (2014). 25072238[pmid].

[8] Sergio Silva, Salviano Soares, and Antonio Valente, 'Digital sound processing using arduino and matlab', (11 2012).

[9] X. Zhang and W. R. Zbigniew, 'Analysis of sound features for music timbre recognition', in *2007 International Conference on Multimedia and Ubiquitous Engineering (MUE'07)*, pp. 3–8, (April 2007).

[7] https://it.wikipedia.org/wiki/C%2B%2B