Measuring Noise Pollution by Utilizing Bluetooth Low Energy Beacons

Evan Fallis and Petros Spachos
School of Engineering, University of Guelph, Guelph, Ontario, Canada

2021 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE) | 978-1-6654-4864-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/CCECE53047.2021.9569083

***Abstract*—A major problem in several modern cities is noise pollution. Noise pollution is any disturbing or unwanted noise that interferes or even harms humans. It has several sources such as public events and vehicular traffic. Being able to characterize areas based on acoustic noise level is useful not only for proving danger in terms of damage to the human ear but also for general discomfort from living near a constantly noisy environment. Internet of Things (IoT) devices can help to collect, process, and characterize acoustic noise, while they offer advantages such as small size and low cost. This work uses small-size Bluetooth Low Energy (BLE) beacons to collect audio data in different parts of a city. The data are forwarded to a server station for further characterization and processing. Initial results show the promising performance of the proposed system, in terms of energy consumption and audio data characterization.**

***Index Terms*—Acoustic Noise Pollution, Bluetooth Low Energy, Internet of Things, Beacons**

I. Introduction

Noise pollution is a major concern for cities and urban environments [1]. Noise pollution comes from several sources around a city, such as construction sites, vehicular traffic, public events, and more [2]. When this noise is in high quantities and concentration, it can be damaging to the human ear, while it is also tied to generating stress and leads to general discomfort [3]. It is important to be able to collect, monitor, and properly characterize the audio data from a city in order to take the necessary actions. However, the noise level is different in different parts of the city, hence, it is necessary, any proposed solution to have several monitoring devices and be able to monitor large-scale areas.

The advantages of Internet of Things (IoT) devices can alleviate the problem. Small and inexpensive devices, equipped with microphones and communication capabilities can collect audio data and forward them to a server. However, as the number of those devices increases, so does the interference among them as well as with other wireless devices in the area.

Bluetooth Low Energy (BLE) beacons are hardware transmitters that use Bluetooth to broadcast their identifier to nearby devices [4]. Beacons can also be equipped with sensors and collect useful data. The use of Bluetooth helps to minimize the power requirement and extend the lifetime of each device. At the same time, the transmission range can be properly configured to limit any interference with nearby devices.

In this work, a noise pollution monitoring system is designed to measure the level of noise in different parts of a city. The system is equipped with BLE beacons for audio data collection, Raspberry Pis for data forwarding, and a server for the data analysis. The main contribution of this work is a wireless audio collection that uses Bluetooth beacons for data collection and WiFi communication for data forwarding. Overall system energy requirements are limited while the beacon firmware was optimized to reduce power consumption during data collection.

The rest of this paper is organized as follows: in Section II the related work is reviewed followed by Section III where the experimental design and the methodology are described. Section IV includes a discussion on the experimental results. The conclusion is in Section V.

II. Related Work

There is a number of research papers dealing with the collection of audio data in urban environments [5]–[14]. In [5], they created a WSN to collect audio data using a sound level meter which communicated mainly using XBees. Their prototype was based on a model created using the tinyLAB tool. The work in [6] outlined a plan to measure various types of pollution commonly found in urban cities, including acoustic sound. It can alert the user when noise levels surpass certain values. The work in [8] outlines a method to reduce noise caused by vehicles, specifically the sound generated between the tires and road. The method uses active noise control by installing actuators inside the vehicle fenders. In [7], artificial noise masking is implemented in Singapore. This is designed to combat noise pollution by adding white noise such as the sound of water to the environment.

In [10], they conducted a study involving noise level measurement around high traffic areas in Yavatmal, India. They were able to find that the residential prescribed limits for noise were being broken 75% of the time. In [9], a railway bridge was analyzed to determine the noise radiation emitting from it. The purpose was to check how much the acoustics of the bridge would affect the noise limits of the city. The idea of designing cities with noise considerations was discussed in [11]. This outlined many ways to design urban environments that help minimize unwanted noise. In [12], an IoT setup was utilized to collect various types of pollution, including audio. Their chosen sensor was the SEN-12642, an embedded analog microphone.

In this paper, we extend our previous work in [13], [14]. Audio data are collected using a BLE beacon and forwarded to the server over a wireless network. The beacon has a microphone that is free from any power source aside from a simple coin cell battery, providing a very flexible solution

978-1-6654-4864-2/21/$31.00 ©2021 IEEE

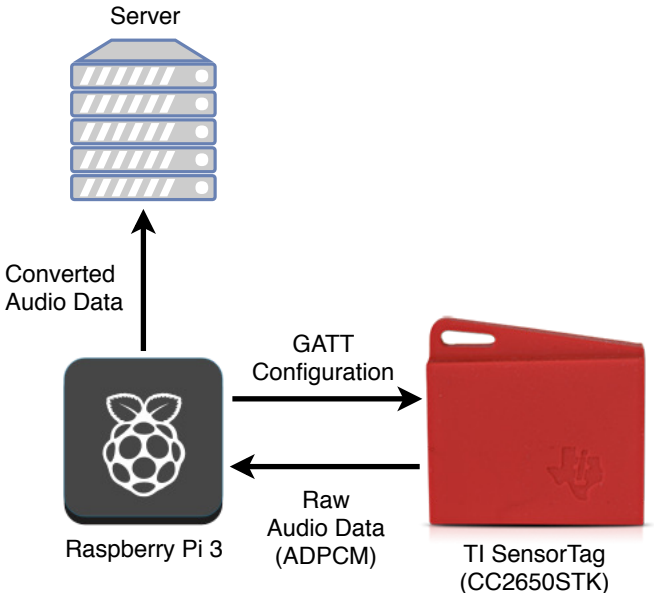

Fig. 1: Audio collection system using BLE beacons and Raspberry Pis.

for positioning audio sensors with the intent of acoustic noise modeling. The data is then forwarded to the server and analyzed using a script to find the envelope of an audio waveform.

III. Experiment Design and Methodology

The proposed system has three main components: the BLE beacon that collects the audio data, the Raspberry Pi that collects the data from the beacons and forwards them towards the server, and the server that does the final data analysis. A general overview of the testbed is shown in Fig. 1.

A. BLE beacon with microphone

For the audio data collection, the Texas Instruments SensorTag was used [15], shown in Fig. 2. It has a wide variety of sensors including an accelerometer, barometer, humidity sensor, luxometer, temperature sensors, magnet sensor, gyroscope, compass, and the necessary microphone. During data collection, only the microphone was activated and all the other sensors were off. SensorTag can communicate through ZigBee, WiFi, and BLE. In this work, BLE was preferred due to its low power requirements [13].

The SensorTag needed to be configured to work with the microphone. It uses Pulse Density Modulation (PDM) to transfer data from the microphone to the microprocessor. To program the firmware of the SensorTag, Code Composer Studio (CCS) was used. This was used to enable the microphone and disable all other services that were not related to the operation of transferring audio data. This was mainly done to save energy while streaming and reduce processing. The following code shows where the actual audio frames are being controlled in the firmware:

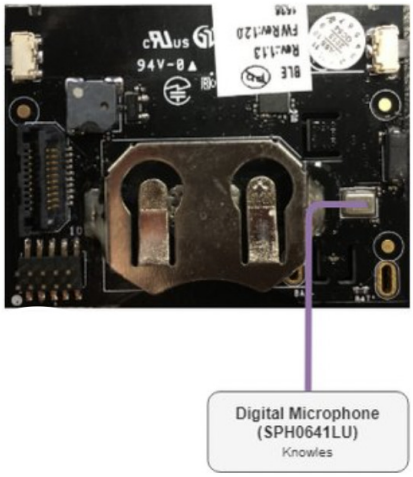

Fig. 2: SensorTag from Texas Instruments with a digital microphone.

Listing 1: Transmit audio frame.

```
1  static void HIDAdvRemote_transmitAudioFrame(uint8_t *buf)
2  {
3   // Send 5 GATT notifications for every audio frame
4   for (int i = 0; i < BLEAUDIO_NUM_NOT_PER_FRAME;)
5   {
6    if (Audio_SetParameter(AUDIOPROFILE_AUDIO
7                ,BLEAUDIO_NOTSIZE
8                ,buf) == SUCCESS)
9    {
10    // Move on to next section of audio frame
11    buf += BLEAUDIO_NOTSIZE;
12    i++;
13   }
14  }
15  Task_sleep(DELAY * (1000 / Clock_tickPeriod));
16 }
```

The defined variable “DELAY”, as seen in the code above, will be changed to add deliberate time between samples. The Bluetooth connection remained in effect for the duration of the experiment. A flowchart with the software processes at the SensorTag is shown in Fig. 3. When the data are collected, they are forwarded from the SensorTag to the Raspberry Pi over Bluetooth.

B. Raspberry Pi

The Raspberry Pi utilized a script made specifically for the SensorTag [16]. The script collects data over Bluetooth and converts them to a waveform using Adaptive Differential Pulse Code Modulation (ADPCM), which is a common format for transferring audio data points. It was customized for this experiment and linked with a Python script to push data to a remote server through WiFi. Alternatively, the ADPCM conversion can happen on the server if preserving the raw data is desired. WiFi was preferred in this step due to the longer range in comparison with Bluetooth.

The Generic Attributes (GATT) helps configure Bluetooth devices by allowing the server device to essentially set up the client. The Raspberry Pi, in this case, will configure the SensorTag to only send audio data. It will then read the Bluetooth notifications whenever new data has been sampled

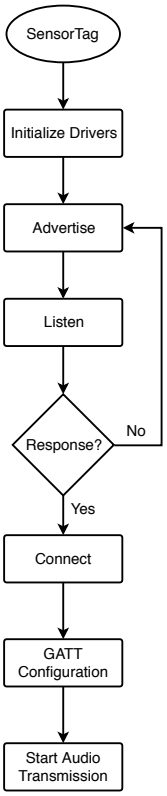

Fig. 3: Flowchart of the processes at the SensorTag.

on the SensorTag. An audio frame is sent in five discrete notifications, with 20 bytes of audio data per notification. The required throughput for full-speed audio transmission is 66670 bits per second. A flowchart with the software processes at the Raspberry Pi is shown in Fig. 4.

The in-depth block diagram of the testbed is shown in Fig. 5. This figure presents the three major devices of the system. The topmost block shows the parts of the SensorTag Kit that is used to collect the acoustic noise. During deployment, there will be multiple SensorTags that are connected to one Raspberry Pi gateway for larger coverage. The collected acoustic data will then be sent from the SensorTag to this device through Bluetooth. The middle block shows the components of the gateway that we used in our testbed. After receiving the SensorTag data, the Raspberry Pi will forward the aggregated information via WiFi to the server. The last block is a black box representation as it can be any device that is capable of analyzing the noise data. This block serves as the final destination of the data for processing.

C. Server

When the data is transferred to the main server, MATLAB is used to analyze all the data and to find the magnitude of each sample. A simple envelope detector is used to estimate the overall level of the audio. Since the data is on the server, it is possible to perform more advanced filtering to distinguish ambient noise from human voice, which will allow profiling an environment based on several categories. The use of a Fourier

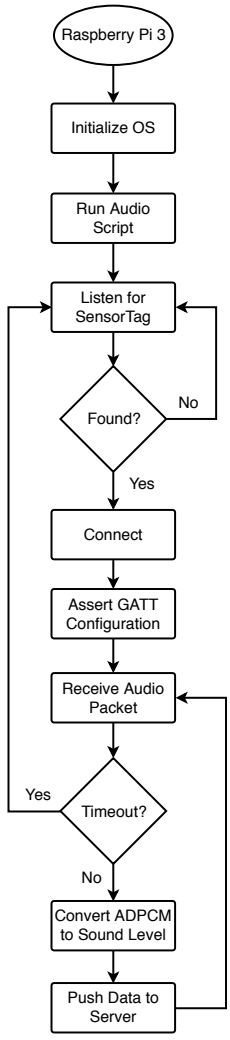

Fig. 4: Flowchart of the processes at the Raspberry Pi.

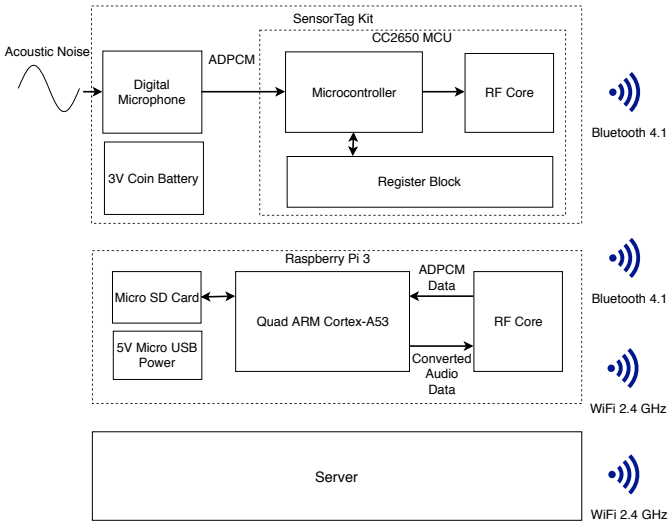

Fig. 5: Block diagram of the proposed system.

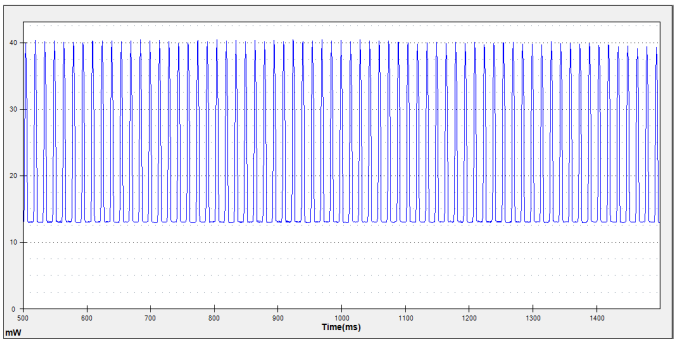

Fig. 6: Power consumption of SensorTag while collecting and transmitting audio data.

Delay (ms)	Power (mW)	Life (days)
0	19.77	0.99
50	9.14	2.14
200	7.66	2.56

TABLE I: Testbed power consumption.

Transform and a simple neural network can help to build a database of audio data proven to represent human voice.

During data collection, the power of the SensorTag was monitored to help characterize the energy requirements of the system as well as estimates of battery life. Reducing battery life concerns is important to keep the BLE beacons reliable, allowing them to be placed anywhere in the environment without the need for outlets. The Monsoon power tool was used to measure said power consumption [17]. It can be used to get very accurate readings at a high sample rate.

IV. Results and Analysis

The experiments focus on two aspects of the system: energy consumption and noise processing.

A. Battery life

Energizer reports a typical capacity of 235 mAh when tested at 2 V [18]. This can easily be translated to mWh by multiplying the two values, resulting in 470 mWh. Hence, the average power can be divided to find the estimated amount of hours, as:

$$Battery\ Life\ = \frac{Total\ Capacity}{Average\ Power} \tag{1}$$

The power usage of the SensorTag when streaming audio data continuously at the maximum rate is shown in Fig. 6. The spikes in power are from both the sampling of audio and the transmission of each frame.

The estimations of battery life are shown in Table I. According to the results, the power consumption with a delay has an average power less than the minimum of the default, which can be seen in Fig. 6. This shows the effectiveness of decreasing the sampling rate in terms of energy consumption. In the future, disabling the Bluetooth radio will be done to reduce the power consumption much further. This will easily be able to extend the battery life to over a month.

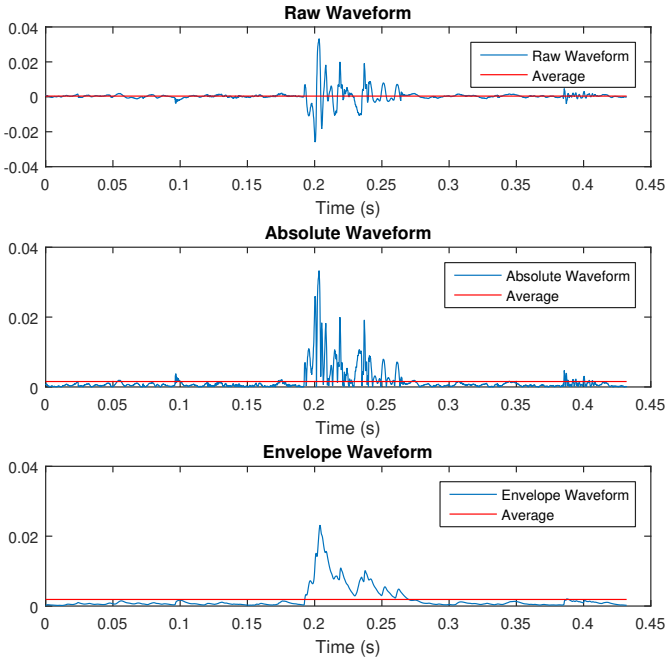

Fig. 7: Envelope analysis.

B. Waveform processing

A comparison of the waveform after using simple filters and processing is shown in Fig. 7. It is important to note that the waveform being used is sampled at full speed, meaning that the power savings technically do not apply to this scenario. Hence, the processing of the signal would be the same if the artificial delay between samples was present.

The raw waveform shows the basic values read from the .wav file. The average is close to zero as it considers both the positive and negative magnitude. The absolute waveform simply takes the raw data and converts it to positive numbers, then calculates the average of that, which ended up being 0.0016. The envelope filter attempts to track the magnitude of the waveform, this instance being slow. The average was 0.0019, slightly higher than the previous.

The purpose of this experiment was not to calculate the actual decibels measured, but rather to show how this technique can be used to find the relative noise level and then normalize the magnitude in later experiments. Storing the raw data on a server is typically a good technique since it can be hard to obtain the original after signal processing has taken place.

V. Conclusion

In this work, a system for audio collection and modeling is presented. The system includes a BLE beacon, a Raspberry Pi, and a server. The system uses Bluetooth to transfer the data from the beacons to the Raspberry Pi and then WiFi to forward them to the server. According to experimental results, the use of BLE helps the overall design to have low

power requirements, while the use of the server can help to better characterize the audio data. An estimation of the battery life was derived and then improved by adding artificial delays between samples. Initial testing with audio data show promising results, however, further processing and large-scale experimentation are necessary.

For future iterations, a type of wake-up sensor would like to be implemented as a low-resolution Analog to Digital Converter (ADC). This would trigger the high-power microphone when any audio is detected. It may also be configured to only listen to specific frequencies. This would use much less power than the current microphone and would be able to enable or disable the high resolution based on the current sound level.

REFERENCES

[1] N. Auger, M. Duplaix, M. Bilodeau-Bertrand, E. Lo, and A. Smargiassi, "Environmental noise pollution and risk of preeclampsia," *Environmental Pollution*, vol. 239, pp. 599 – 606, 2018.

[2] P. E. K. Fiedler and P. H. T. Zannin, "Evaluation of noise pollution in urban traffic hubs—noise maps and measurements," *Environmental Impact Assessment Review*, vol. 51, pp. 1 – 9, 2015.

[3] T. Münzel, M. Sørensen, F. Schmidt, E. Schmidt, S. Steven, S. Kröller-Schön, and A. Daiber, "The adverse effects of environmental noise exposure on oxidative stress and cardiovascular risk," *Antioxidants & redox signaling*, vol. 28, no. 9, pp. 873–908, 2018.

[4] P. Spachos and K. Plataniotis, "BLE Beacons in the Smart City: Applications, Challenges, and Research Opportunities," *IEEE Internet of Things Magazine*, vol. 3, no. 1, pp. 14–18, 2020.

[5] J. G. Cantuña, S. Solórzano, and J. Clairand, "Noise pollution measurement system using wireless sensor network and ban sensors," in *2017 Fourth International Conference on eDemocracy eGovernment (ICEDEG)*, April 2017, pp. 125–131.

[6] B. Siregar, A. B. A. Nasution, and F. Fahmi, "Integrated pollution monitoring system for smart city," in *2016 International Conference on ICT For Smart Society (ICISS)*, July 2016, pp. 49–52.

[7] H. Shu, Y. Song, and H. Zhou, "High annoyance urban noise masking," in *2017 IEEE 2nd International Conference on Signal and Image Processing (ICSIP)*, Aug 2017, pp. 380–384.

[8] M. A. Sahib and S. Streif, "Design of an active noise controller for reduction of tire/road interaction noise in environmentally friendly vehicles," in *2017 Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA)*, Sept 2017, pp. 59–62.

[9] Y. Tong, Y. Jiang, Z. Zhou, and D. Ma, "The study of railway bridge noise monitoring and prevention," in *2014 Fourth International Conference on Instrumentation and Measurement, Computer, Communication and Control*, Sept. 2014, pp. 244–249.

[10] P. B. Nagarnaik, V. M. Mohitkar, and D. K. Parbat, "Evaluation of noise pollution annoyance at uninterrupted traffic flow condition," in *2011 Fourth International Conference on Emerging Trends in Engineering Technology*, Nov 2011, pp. 156–163.

[11] X. Zhao, B. Cheng, S. Liang, and Y. Tang, "Design for human aural health: Noise pollution prevention and control in urban dwellings design of china," in *2009 3rd International Conference on Bioinformatics and Biomedical Engineering*, June 2009, pp. 1–4.

[12] P. Patil, "Smart iot based system for vehicle noise and pollution monitoring," in *2017 International Conference on Trends in Electronics and Informatics (ICEI)*, May 2017, pp. 322–326.

[13] E. Fallis, P. Spachos, and S. Gregori, "A power-efficient audio acquisition system for smart city applications," *Internet Things*, vol. 9, p. 100155, 2020.

[14] E. Fallis, M. Lipski, A. Mackey, M. J. Baucas, M. James, P. Spachos, and S. Gregori, "A testbed for adaptive microphones in ultra-low-power systems," in *2019 IEEE Sustainability through ICT Summit (StICT)*, 2019, pp. 1–6.

[15] SensorTag Audio Details. (retrieved: 2021-07-29). [Online]. Available: https://www.ti.com/tool/TIDC-CC2650STK-SENSORTAG

[16] GitHub Code for Audio Collection. (retrieved: 2021-07-29). [Online]. Available: https://github.com/fommike/node-sensortag

[17] Monsoon Power Monitor. (retrieved: 2021-07-29). [Online]. Available: http://www.msoon.com/LabEquipment/PowerMonitor/

[18] CR2032 Energizer Datasheet. (retrieved: 2021-07-29). [Online]. Available: http://data.energizer.com/pdfs/cr2032.pdf