System Design Document

For

Audio Surveillance System

Team members: Zachary Tauscher, Jacob Attia, Caleb Leeb, Jaclyn Welch, Jorge Garcia

Version/Author	Date
Version 1	28 September 2021
Version 2	26 October 2021
Version 3	30 November 2021
Version 4	01 February 2022

TABLE OF CONTENT

1 INTRODUCTION 3
1.1 Purpose and Scope 3
1.2 Project Executive Summary 3
1.2.1 System Overview 3
1.2.2 Design Constraints 3
1.2.3 Future Contingencies 3
1.3 Document Organization 3
1.4 Glossary 4
2 SYSTEM ARCHITECTURE 4
2.1 System Hardware Architecture 4
2.2 System Software Architecture 4
2.3 Internal Communications Architecture 4
3 HUMAN-MACHINE INTERFACE 4
3.1 Inputs 5
3.2 Outputs 5
4 DETAILED DESIGN 5
4.1 Hardware Detailed Design 6
4.2 Software Detailed Design 6
4.3 Internal Communications Detailed Design 7
5 EXTERNAL INTERFACES 7
5.1 Interface Architecture 7
5.2 Interface Detailed Design 8
6 SYSTEM INTEGRITY CONTROLS 8

INTRODUCTION

1.1 Purpose and Scope

This section provides a brief description of the Audio Surveillance System's purpose and scope.

1.2 Project Executive Summary

This section provides a description of the Audio Surveillance System from a management perspective and an overview of the framework within which the conceptual system design was prepared.

1.2.1 System Overview

The goal of the Audio Surveillance System is to record sounds with at least three microphone arrays, each consisting of at least three microphones, and transfer the audio to a Raspberry Pi with bluetooth capabilities. The Raspberry Pi will transfer this information to a computer with the Audio Surveillance System's application, which will visualize the data for the user to view. This view will display a basic map of the room under surveillance, indicate voices and footsteps in the room, and where these sounds are detected in the room. Two methods for the set up of the microphone arrays are currently being considered: omnidirectional microphone arrays using signal strength to localize sound, and directional microphone arrays that triangulate the location of the sound.

1.2.2 Design Constraints

The restraints of this system are as follows:
Cost: The quality and amount of hardware we use will affect the cost and the appeal to the customer conversely. Hardware such as a data acquisition board, which would increase quality of data and assist in data processing, were removed from the system design due to cost constraints. The microphone arrays purchased for the system will be the majority of the budget, as the team was able to procure a Raspberry Pi 4 8GB CanaKit, which would have been costly to procure otherwise.
Disguise: The purpose of the Audio Surveillance System is to replace traditional video surveillance with audio surveillance to make surveillance more discreet and cost effective. Using small hardware and being able to place it around a room in a discreet way will limit the hardware we can use and the arrangement of the microphone arrays. It will also limit the positioning of the arrays within the system itself as a more distantly spread series of arrays will be significantly more noticeable.
Hardware Limitations: Hardware capability will determine how accurate the visualization of the sounds in a room can be. The ability of the microphone arrays to determine the strength and direction of sound accurately will affect the accuracy of the system.
Availability of Training Data: Lastly, to test the design and provide data for the machine learning software to differentiate between sounds, pure sounds will have to be provided to the system. The physical ability to access pure sound is a constraint that will limit the accuracy of our system.

1.2.3 Future Contingencies

The current design of the system includes the integration of several microphone arrays with a Raspberry Pi to a central computer for processing using bluetooth. If the Raspberry Pi processor is not able to handle sending the data to the computer via bluetooth, we will adjust to a wired setup for the Raspberry Pi and computer. If the Raspberry Pi does not have the computational power to support the system, we have access to additional Raspberry Pis and will combine them with the current one to increase the computational power.

1.3 Document Organization

This document begins by describing the individual elements of the system, then describing the whole system from a high level, then breaking the specific subsystems down to include every detail of the system. It will cover the purpose, components, communication, architectures, and design of the system, as well as contingencies and possible replacements for possible future issues of the system.

1.4 Glossary

NLP - Natural Language Processing
ASR - Automatic Speech Recognition
.wav - Waveform Audio File Format
.mp3 - MPEG-1 Audio Layer 3 Format
.wma - Windows Media Audio Format
MIR - Music Information Retrieval
STFT - Short Term Fourier Transform
MFCCs - Mel-Frequency Cepstral Coefficients
ANN - Artificial Neural Networks
SoC - System-on-a-Chip

2 SYSTEM ARCHITECTURE

This section describes an overview of the hardware and software architecture for the Audio Surveillance System and subsystems.

2.1 System Hardware Architecture

There are several hardware components that make up the Audio Surveillance System, that are listed as follows:

- Seeed Studio ReSpeaker Mic Array V2.0 Microphones XMOS's XVF-3000 (Quantity 4)
- UNYEE USB Bluetooth Wireless Micro Adapter EDR V5.0
- ASUS USB-BT500 USB 2.0 Bluetooth 5.0 USB Adapter
- USB 2.0 Micro USB Male to USB Female OTG Adapter (Quantity 2)
- Raspberry Pi 400 All-in-One Quad-Core 64 bit 4 GB Ram Dual Band WiFi Bluetooth 5.0 BLE Dual 4K Output Complete Personal Computer Kit

The system also utilized a computer that can connect to the Raspberry Pi via Bluetooth or wired connection, and can run the software required of the system. For our project, any of the personal computers our teammates had were used for this purpose. The user would have to connect a personal or desktop computer with the necessary software and Bluetooth capabilities that can connect to the other components to complete the system.

2.2 System Software Architecture

Once .wav files are taken in by the microphone arrays in the system, they are distributed to three different internal subsystems. One subsystem, consisting of the firmware present in the microphone array and a python script which utilizes it as a library, is responsible for the DOA calculations. Another subsystem generates spectrograms to be used in further ML processing. The final subsystem receives this data to be picked apart for solely the time-delay of arrival, furthering the precision of the coordinates produced by later telemetry outputs.

2.3 Internal Communications Architecture

Sound is recorded by the microphone arrays to begin the system, which is fed to the Raspberry Pi via USB cables. The Raspberry Pi records this sound as .wav files. Communication within the system then flows from the .wav files created on the Raspberry Pi to the host computer via the Bluetooth adapters. These files are received by the main program that uses the directional information from the microphone array and processes the information in two different ways. One part of the program utilizes the directional information to plot the location of the sound in the room with relation to the microphone arrays. The other part of the program converts the audio waves to a spectogram, and uses machine learning classification methods to classify the sound. The location and classification are then visualized for the user to provide a diagram of the room to the user using only the audio waves.

3 HUMAN-MACHINE INTERFACE

This section provides details as to how the user of the system will interact with the system, including inputs and outputs to be expected.

3.1 Inputs

The input to the system will be the sound in the room, which will be received by the microphone arrays, then uploaded to the Raspberry Pi. The Raspberry Pi will send this data to the secure computer as a serialized stream of values representing the strength and direction of the sounds, which will be processed using aforementioned Python scripts into interpretable data.

3.2 Outputs

The output of the system will be a diagrammatic display on the screen of the computer, representing the sounds in the room for the user to view. This will include the categorization of the sound and a scaled approximation of its location.

4 DETAILED DESIGN

This section details exactly how to build and integrate the system, including the hardware components, code, software components, and connecting them to have a cohesive and functional production.

4.1 Hardware Detailed Design

The three main hardware components of the system are on the microphone arrays, Raspberry Pi, and the host computer.

The microphone arrays in this system are Seeed Studio ReSpeaker Mic Array v2.0 Microphones XMOS's XVF-3000. These arrays contain 4 high performance digital microphones and have 12 programmable RGB LED indicators. They require a power supply of 5V DC from Micro USB or Expansion Header. Each array has a 70 mm diameter and a 3.5mm audio jack output socket, which we will not be connecting with this design. Each microphone has a -26 dBFS sensitivity and is omnidirectional. If the microphone arrays needed to be replaced, ones with the same number and orientation of microphones and of comparable sensitivity would be needed. It is also important that they have omnidirectional reception capability, and that they have USB compatibility.

The Raspberry Pi used in this system is the Raspberry Pi 400 which comes as part of a complete personal computer kit, all the components of which are integrated in our system. The Raspberry Pi has a Broadcom BCM2711 quad-core Cortex-A72 (ARM v8) 64-bit SoC processor with a speed of 1.8 GHz. The Raspberry Pi has Bluetooth capability, but for the working system now we are using both USB 3.0 ports to connect the system. The Raspberry Pi with these specifications is crucial to the system due to its size, allowing it to be inconspicuous, and it's power, allowing it to play a critical processing part of the system. If the Raspberry Pi was not functional or available, the data processing it performs could be done on the host computer, but it would require the microphone arrays to be connected directly to the computer, making it more difficult to disguise as a surveillance device.

The host computer and Raspberry Pi are connected using a USB to USB cable. The Raspberry Pi is connected to the microphone array with a USB to USB cable as well, but with a USB 3.0 male to female OTG adapter.

4.2 Software Detailed Design

This section will detail the design of the software used for the Audio Surveillance System. We will be setting up the Raspberry Pi with Raspberry Pi Imager to be able to intake the data and send it to the computer in a serialized stream. This will utilize pip for package management, and Thonny as the IDE while in testing and development. This system will use Scikit-Learn as a library to enable machine learning in Python.

5 EXTERNAL INTERFACES

External systems are any systems that are not within the scope of the system under

development, regardless whether the other systems are managed by the State or another agency. In this section, describe the electronic interface(s) between this system and each of the other systems and/or subsystem(s), emphasizing the point of view of the system being developed.

5.1 Interface Architecture

In this section, describe the interface(s) between the system being developed and other systems; for example, batch transfers, queries, etc. Include the interface architecture(s) being implemented, such as wide area networks, gateways, etc. Provide a diagram depicting the communications path(s) between this system and each of the other systems, which should map to the context diagrams in Section 1.2.1. If appropriate, use subsections to address each interface being implemented.

5.2 Interface Detailed Design

For each system that provides information exchange with the system under development, there is a requirement for rules governing the interface. This section should provide enough detailed information about the interface requirements to correctly format, transmit, and/or receive data across the interface. Include the following information in the detailed design for each interface (as appropriate):

- The data format requirements; if there is a need to reformat data before they are transmitted or after incoming data is received, tools and/or methods for the reformat process should be defined
- Specifications for hand-shaking protocols between the two systems; include the content and format of the information to be included in the hand-shake messages, the timing for exchanging these messages, and the steps to be taken when errors are identified
- Format(s) for error reports exchanged between the systems; should address the disposition of error reports; for example, retained in a file, sent to a printer, flag/alarm sent to the operator, etc.
- Graphical representation of the connectivity between systems, showing the direction of data flow
- Query and response descriptions

If a formal Interface Control Document (ICD) exists for a given interface, the information can be copied, or the ICD can be referenced in this section.

6 SYSTEM INTEGRITY CONTROLS

Sensitive systems use information for which the loss, misuse, modification of, or unauthorized access to that information could affect the conduct of State programs, or the privacy to which individuals are entitled.

Developers of sensitive State systems are required to develop specifications for the following minimum levels of control:

- Internal security to restrict access of critical data items to only those access types required by users
- Audit procedures to meet control, reporting, and retention period requirements for operational and management reports
- Application audit trails to dynamically audit retrieval access to designated critical data
- Standard Tables to be used or requested for validating data fields
- Verification processes for additions, deletions, or updates of critical data

Ability to identify all audit information by user identification, network terminal identification, date, time, and data accessed or changed.