System Requirements Specification

for

Audio Surveillance System

**Version 1.0 approved**

**Prepared by Zachary Tauscher, Jacob Attia, Caleb Leeb, and Jaclyn Welch**

**Embry Riddle Aeronautical University Senior Capstone Design I**

**26 October 2021**

**Table of Contents**

**Table of Contents ii**

**Revision History ii**

**1. Introduction 1**

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 1

**2. Overall Description 2**

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 2

2.5 Design and Implementation Constraints 2

2.6 User Documentation 2

2.7 Assumptions and Dependencies 3

**3. External Interface Requirements 3**

3.1 User Interfaces 3

3.2 Hardware Interfaces 3

3.3 Software Interfaces 3

3.4 Communications Interfaces 3

**4. System Features 4**

4.1 System Feature 1 4

4.2 System Feature 2 (and so on) 4

**5. Other Nonfunctional Requirements 4**

5.1 Performance Requirements 4

5.2 Safety Requirements 5

5.3 Security Requirements 5

5.4 Software Quality Attributes 5

5.5 Business Rules 5

**6. Other Requirements 5**

**Appendix A: Glossary 5**

**Appendix B: Analysis Models 5**

**Appendix C: To Be Determined List 6**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Initial Draft | 9/28/21 | Proposal of Draft Document | 1 |
| Second Draft | 10/26/21 | V2 Submission | 2 |
|  |  |  |  |

# Introduction

This product is designed to collect and process sound signals via a microphone array and send that data to a secure computer. This computer will utilize machine learning algorithms to identify the sounds recorded by this microphone array and ascertain the position of the sound relative to the microphone array. The secure computer will provide a GUI for security personnel to view the source location of the sound and the classifier given to that sound. The goal of this product is to provide cheap surveillance without the need for invasive camera systems.

## Purpose

The purpose of this document is to identify the requirements of a system which provides positional and identifying information on people within a room based solely on the audio data received by several arrays of microphones placed within the room. There are two experimental designs which are proposed by this document, each with their own unique requirements, which will be specified in this document below. However, for the most part these requirements and basic statements of intent are shared between the two designs. This product will encompass the scope of single-room surveillance and will be built to function in environments with common sources of background noise.

## Document Conventions

For future reference within this document, all mentions of the first experimental product design, multiple arrays placed at disjointed points in a room connected via wireless connectivity modules, will be referred to as design “A”. The second design, consisting of a ring of microphone arrays to be modularly placed at a central point within a room, will be referred to as design “B”.

## Intended Audience and Reading Suggestions

This document is intended to be read and reviewed by the product owner (Dr. Qi Cheng), the customer (Dr. Ilhan Akbas), and to be contributed to and continually modified by all members of the development team: Jacob Attia, Caleb Leeb, Zachary Tauscher, and Jaclyn Welch.

For those readers unfamiliar with general concepts of machine learning, audio processing, or triangulation, reviewing the included documents for reference before identifying the scope of this project may be helpful. For readers with a sufficient background in these concepts who feel comfortable with an integrated approach of these subjects to further surveillance, the following section regarding the product scope may be of more relevance.

To any reader of this document whose intention is to continue development in the pursuit of future research, both the product perspective (2.1) and our analysis models used (Appendix B) are of particular interest.

## Product Scope

The product is not intended to cover more than a single room, and it's scope is not to encompass more than positional information in more than two dimensions, as a more labor-intensive, intrusive, and indiscreet method of setting up a room would have to be employed to simulate three dimensions, robbing the product of any viable use case.

## References

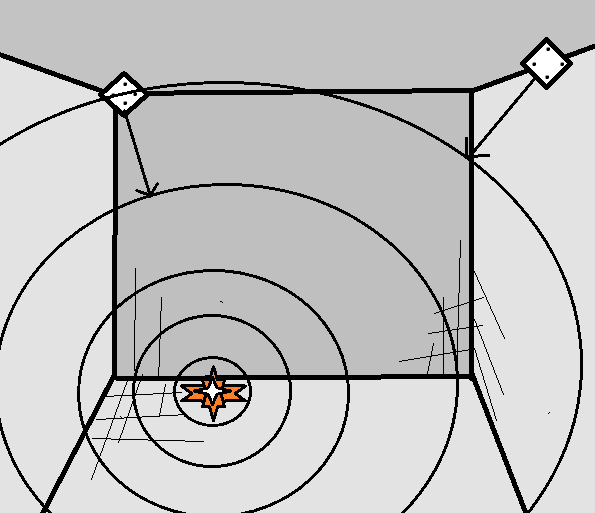
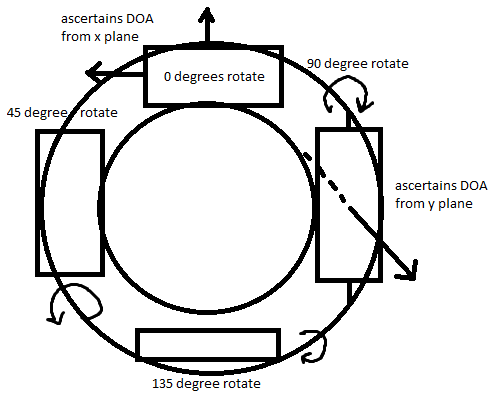
The following link: https://github.com/karatekidcaleb/CS490-Option-2-Audio/ will take the reader to the GitHub repository in which reside all source code and build-relevant working directories. As our product’s software is not currently maintained or protected by any license, redistributors and those intent on modification can expect to follow guidelines relevant to the GNU Lesser General Public License 3.0. For those interested in some literature relevant to the formation of this product’s research and a cursory literature review, please refer to the references to prior research in this AoC detailed later in 2.1.

# Overall Description

## Product Perspective

#the following no longer viable considering we have decided Design B will be our primary working model

Design A Design B

 ****

## Product Functions

* The product must be able to discern/label what types of sound are coming in (such as footsteps, normal conversation)
* The product must be able to determine the location of a sound source in a room
* The product must be able to display the locations and differentiations visually

## User Classes and Characteristics

**2.3.1: Security Guard in Command**

These personnel could be tasked with the monitoring of associated entities producing sound within a room for some security purposes. They would need access only to basic user-level functionality (the GUI display). This is the critical end user, as they will be interacting with the system most frequently.

**2.3.2: Administrator of Audio Surveillance System**

These personnel would need root-level access to the system and have all authentication keys necessary to make, modify, and delete files associated with the system and respective firmware. This user would primarily keep records of key events, previous displays on the GUI, and specified timeframes of activity. This user would ideally have their own meta-monitor program to capture displays form the system for future use and provide repudiation.

**2.3.3: Technical Support (IT)**

These personnel would provide any necessary maintenance to the system via their intimate knowledge with the system and at least admin level privileges.

## Operating Environment

## 2.4.1 User Interface

The designed interface runs on a secure host computer in communication with the Raspberry Pi microphone array and utilizes the processed data to display to the user the location and identity of multiple sound sources. This display will be in the form of a continually updating two-dimensional window with text labels and colored polygons on the screen to display associated characteristics of a sound source.

## 2.4.2 Data Collection

A microphone array of 4 microphones is linked to a Raspberry Pi and is used to collect audio files and time delay of arrival data. This data is processed using Python to turn the audio .wav file into a spectrogram for use in machine learning and package the time delay of arrival to be used for locational calculation. It is then transmitted to the secure host computer.

## 2.4.3 Data Processing

The data processing environment takes in the collected data received from the Raspberry Pi microphone array. It uses the spectrogram of the recorded audio as an input to Python machine learning libraries, then classifies what the audio is of. It separately uses the time delay of arrival data to calculate locational coordinates to be projected on a map of the room.

## Design and Implementation Constraints

1. Design Constraints:

This product is intended for surveillance use, and therefore discretion of implementation is priority. However, the more compact and less obtrusive the product is designed, the less precise the outputs and telemetry data will be. As the microphone arrays specifically are less spaced out, the computation of the intersection of several DOAs relies more and more on the precision of individual microphone arrays, losing precision in the resulting noise/inaccuracy. The product is proposed to have approximately half a meter between the centroids of each microphone array to balance these two concerns.

1. Implementation Constraints

The size and shape of the room is dependent on the use case of the product, and therefore eliminating the interference from things like sound bouncing off walls and reverberations in more vacant locations is a concern and somewhat important to minimize. The spectrograms of each waveform file produced by the microphone arrays will not be affected, as only a general noise profile is required. However, sound echoing from the ceiling to the microphone arrays will skew the DOA to a mild extent, sacrificing some precision.

## User Documentation

https://wiki.seeedstudio.com/ReSpeaker\_4-Mic\_Linear\_Array\_Kit\_for\_Raspberry\_Pi/

http://www.pygame.org/docs/ref/

https://www.analyticsvidhya.com/blog/2021/06/introduction-to-audio-classification/

## Assumptions and Dependencies

*<List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).>*

2.7.1 Assumptions

Python libraries in the source code for our project may not be maintained in the future or ensured that they are compatible with the most current version of python. For prudence, the version of Python which is compatible with all libraries we use is 3.8.8.

# External Interface Requirements

## User Interfaces

The User Interface subsystem is comprised of a window on the desktop of the secure computer connected to the Raspberry Pi which displays any points and associated labels of activity within the room under surveillance. These points will be representative of a scaled top-down view of the room and will update as new sound signals are picked up by the microphone arrays.

## Hardware Interfaces

The Raspberry Pi and secure laptop or computer system must be connected via Bluetooth. By running a python server on the secure computer connected to a client program in the Raspberry Pi, a 6-channel relay from each microphone array will be communicated and processed further on the computer system. Each microphone must also be configured to communicate with the Raspberry Pi via USB and USB-C connections (they will be in close proximity to one another).

## Software Interfaces

*<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.>*

Our software interfaces consist of the Raspberry Pi 4 Operating System, in which all drivers for the microphone arrays, a working python environment for the basic routing of signals, and connectivity to the secure host computer all take place. It also consists of the secure monitoring computer with our visualization software, also running a working python environment, and with the appropriate Bluetooth drivers.

# System Features

*<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>*

## System Feature 1

*<Don’t really say “System Feature 1.” State the feature name in just a few words.>*

4.1.1 Description and Priority

*<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>*

4.1.2 Stimulus/Response Sequences

*<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>*

4.1.3 Functional Requirements

*<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use “TBD” as a placeholder to indicate when necessary information is not yet available.>*

*<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>*

REQ-1:

REQ-2:

## System Feature 2 (and so on)

# Other Nonfunctional Requirements

## Performance Requirements

*<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>*

## Safety Requirements

*<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product’s design or use. Define any safety certifications that must be satisfied.>*

## Security Requirements

There are a number of security and privacy requirements that must be met. As the system will be recording audio the files and data must not be contained on the device itself but rather sent to a host system. As the system will be left in a room the hardware must not allow for tampering and should be completely contained. A potential security measure is to allow the system to identify if it is being tampered with and send a security alert and shut off connection to the host. For the scope of this project we will restrict access to only users who have a valid ID.

*<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>*

## Software Quality Attributes

*<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>*

## Business Rules

*<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.>*

# Other Requirements

*<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>*

**Appendix A: Glossary**

*<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>*

**Appendix B: Analysis Models**

*<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams*.>

**Appendix C: To Be Determined List**

*<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>*