Ben Gurion University

AISN

Auditory Imaging For Sightless Navigation

Application Requirements Document

Yakir Dahan
Royi Freifeld
Vitali Sepetnitsky

December 12, 2010

Revision: 1.0

Contents

1 Introduction

1.1 Vision

Most of our navigation in the everyday life, heavily depends on visual feedback that we get from our environment. When the ability to see the surroundings is missing due to visual impairments or hostile environment, the ability to navigate is also damaged. There are few options to help blind people with everyday navigation such as the white cane or a guide dog. But these two have lots of disadvantages:

- White canes cannot identify barriers above floor lever and their use requires many skill. In addition, while providing a degree of mobility they limit the freedom of their owner, i.e. restriction of the use of one hand.
- Guide dogs are extensive and expensive to train, moreover they can only work for an average of seven years and only a small fraction of the blind community can use their help.

Other options are for example the *optical radar* developed in BGU or *sensory substitution* research performed in the Hebrew university which relies on a *sonar-like images scanning* (for more information see the *Appendices* part). But both of these projects are limited in detecting obstacles only and are giving a visualization of the environment in a sonar-like manner. For the reasons stated here our project strives for the following goals:

- Sightless navigation by sensory substitution:

  Development of an application that allows a person to hear his environment for the purpose of navigating, relying primarily on the sense of hearing.
- Performing integration with a spatial auditory environment.
- Providing a flexible testing environment for future research:
  Our system will act as a research vehicle for the neuroscientists and especially for those that deal with *cognitive science*. The system will allow them to learn more about the step-by-step adaptive processes involved during the training towards development of a synthetic vision system which exploits the existing multisensory processing and of the human brain through training and education.

In this project we will develop an application which translates images received, in real time, from a video camera, into *3D audio environment* that can be easily perceived by people from both the blind and the sighted communities. The application will include an extensive and sophisticated training environment, which will guide a blind user through the process of adaptation to the system, understanding *3D sound* sensations and how they are produced from images. The system will use various algorithms for image processing and allow to install several types of 3D sound technologies. It will be flexible and easy to extend and change, thus adjustable to the needs of both blind users and researchers towards an effective and informative use.

In the near future, we plan to extend the application and allow it to be installed on a smart phone, towards a more friendly and common use, which will give the blind users an ability to use this application as a mobile guide, thus improving their quality of life (QoL).

1.2 The Problem Domain

Blindness is considered to be among the sensory disabilities that have a major impact on everyday life. There exists several sensory substitution techniques towards environmental imaging. Sonar is based on implicit preservation of only some degree of the environmental information, but not on mapping an image to another sensory. The systems designed today, use a left-to-right scanning which require a lot of training and good hearing. Employing 3D sound within the process can improve the sensory substitution process and make it more intuitive.

We are developing a system that converts visual information to a **3D sound**. Our system is designed to be used by visually impaired users, in order to help them getting a visualization of their surroundings. In addition, the system will supply a dynamic environment for the purpose of experiments performance in order to test the combination of visual information processing and 3D sound creation and positioning.

Figure 1: Basic System Architecture Diagram

The system consists of 3 major components involved in the process of converting the images to 3D sound:

- The Image Processing Unit
- The Sound Creation Unit
- The 3D Sound Positioning Unit

Visual information is obtained as a stream of images, by using a **web camera** connected to the machine on which the system is running. After analyzing a frame, by the Image Processing Unit, the extracted visual information is passed to the Sound Creation Unit. This unit creates the relevant sounds, according to the received information and transfers them to the 3D Sound Positioning Unit, which can be a 3rd party or our own implementation, and the sounds are positioned in the auditory environment. The created and positioned sounds are passed to **headphones** and played to the user.
The system stores user profiles for daily use and research purposes along with storing the 3D sound characteristics *HRTF datasets*. In addition, research experiments can be conducted using the training capabilities supported by the system, which allows to record the results to a data storage.

1.3 Stakeholders

1.3.1 Users

End users The system will be used, mostly, by visually impaired (or even blind) people (also called "end-users") for the purpose of hearing their physical environment.

Configuration users The system installation and initial tuning, such as user profiles creation, will be done by configuration users having the ability to see the operations they perform. These users will set all the settings needed for some end-user, thus allowing him to use the system properly.

Researchers In addition, the system will be designed for another type of users - researchers who wish to conduct experiments regarding 3D sound and image processing integration, using different image and sound processing methods. These users will be able to control all the system parameters and adapt them for their needs, during the experiments.

1.3.2 Experts

The system 3D sound engine, based on HRTF datasets is developed in coordination and under technical advisory of the acoustics laboratory, located at Electrical and Computer Engineering department in Ben-Gurion University, Beer-Sheba, Israel. Furthermore, this laboratory supplies us the sound equipment required for acoustic interface experiments we perform while deciding about 3D sound technologies.

1.3.3 Sponsors

The system is sponsored by Deutsche Telekom (DT) laboratories at Ben-Gurion University, Beer-Sheba, Israel. The design and the implementation of the system are made under technical advisory of **Dr. Rami Puzis** from DT labs. Design and implementation decisions will be discussed with Dr. Puzis, in coordination with the project team.
This project can be integrated with a project of a larger scale developed in Quality and Usability Lab, Deutsche Telekom Laboratories, TU Berlin.

1.4 Software Context

1.4.1 Image Processing Unit

This unit receives the obtained images and retrieves visual information relevant to the end user. The unit is responsible to produce an output to the Sound Creation Unit. It does so by implementing several image processing and computer vision methods (that can be replaced easily), that enable the retrieval of information. The system, should produce a sound as meaningful as possible for the blind. Thus, the Image Processing Unit has a role of *understanding* the frame that it grabs and return the best result as possible.
It's input, naturally, is a frame from a web camera. After grabbing a frame, and according to the selected method, the Image Processing Unit will produce, fairly quickly, data to visualize.
The Image Processing Unit utilizes the power of the *OpenCV* library (see *System Interfaces* below), for simplicity and camera interface. All image processing actions is being done in this Unit. Once the frame has gone through the Image Processing Unit, there will be no more 'image object' or 'image related' data. It is completely decoupled from the rest of the system.
The Image Processing Unit can be controlled by setting parameters associated with it, using the System UI Unit (see below).

1.4.2 Sound Creation Unit

This unit is responsible for creating appropriate sound according to the the recognized visual information. This unit can create two types of sounds:

- Discrete sound sources - for some chosen points.
- Continuous sound fields that suits a continuous set of points or a region in the input frame.

1.4.3 The 3D Sound Positioning Unit

This unit performs auditory spatialization of the sound and informs the user, through **headphones**, about the locations and movements of the detected objects. The auditory spatialization is performed on sound samples, generated by the Sound Creation Unit, by using a special **HRTF Data Storage**, which stores *HRTF datasets*. As it can be understood from above, this unit can be a 3rd party which simply gets a sound and a location in the auditory environment where it should be placed, and performs the spatialization. However, by using *HRTF datasets*, we seek to implement our own algorithm which will, possibly, perform more accurate spatialization along with an ability to be tuned personally for a specific user.

1.4.4 The System UI Unit

This unit enables the user to configure all the parameters associated with the three previous units and performs user related operations such as creating and updating user profiles. The created profiles and settings are stored in the **Settings Data Storage** from which they are obtained during the system operation and are presented to the user by the System UI Unit. In addition, the System UI Unit provides an environment for experiments conduction and recording, in a user-friendly manner.

1.5 System Interfaces

1.5.1 Hardware Interfaces

As described above, the system utilizes two hardware components:

- A **web-camera**, used for grabbing images which are used as an input for the Image Processing Unit.
  The hardware interface which is used for this reason is a standard driver for the operating system, on which our system is installed, supported by the camera's vendor. The interaction with the camera's driver will be performed using the OpenCV API.
  The OpenCV library enables communication and handling of webcams. It provides us with a comfortable and easy-to-use interface on handling camera and camera connection. After a quick check 'under the hood', it is actually a wrapper to another library named VideoInput (located here http://muonics.net/school/spring05/videoInput/). VideoInput is an open source C++ library.
  The Camera is initialized via a single function call: *cvCreateCameraCapture* (another way possible, since this function name is unclear, is using a macro *cvCaptureFromCAM*). It returns a struct of type *CvCapture** holding all information needed in order to control the camera. Once the camera is 'connected' to our system, the following function is used in order to retrieve a single frame *cvQueryFrame* returning an image frame of struct *IplImage**. All data structures are allocated (and released) with proper OpenCV API.
  Note: The scheme of grabbing a frame involves repeating the paradigm mentioned above in an infinite loop.

- **Headphones**, to which the created and positioned sounds are passed.
  The hardware interface which is used for this reason in a standard audio driver, supported by the soundcard's vendor. In that case, the interaction with the driver will be performed by using either the OpenAL API (which connects with the driver through DirectSound) or by using the Windows API functions which are supported within C/C++ (which again, use DirectSound in order to transfer the sound to the audio device).

1.5.2 Software Interfaces

OpenCV API As it was said above, OpenCV is an open source computer vision library that enables the developer to use built-in image processing algorithms. The library is written in C and C++, being as efficient as possible providing real-time results in both C and C++ API.
The Image Processing Unit will use the C language API and manage all data structures OpenCV provides (with respect to the C API).

OpenAL API OpenAL is a cross-platform 3D audio API appropriate for use with gaming applications and many other types of audio applications. The API models a collection of audio sources moving in a 3D space that are heard by a single listener somewhere in that space. The Sound Positioning Unit makes use with this API in order to perform real time sound positioning. OpenAL API functions that perform the sound spatialization, modification and retrieval of auditory environment settings, are prefixed with "AL_". All the data and settings regarding the auditory environment supported by OpenAL, are saved into 3 types of data structures named Listener, Buffer and Source. These data structures are allocated and freed using appropriate OpenAL API functions.

1.5.3 Events

User related events:

- **StartVisualization:** A blind user asks to start a visualization process.
  This event is a trigger to the whole system operation. It starts the process of receiving frames from the camera, processing them, creating 3D-sounds and positioning them.

- **PerformTraining:** A blind user asks to perform a training.
  This event is a trigger to a training session in which various types of training (that are supported by our system) are utilized by the user in order to improve the sensory substitution process made by his brain (for the purpose of improving the benefit from our system).

- **CreateNewUserProfile:** A user asks to create a new user profile.
  This event is triggered each time a new profile creation is required. The event causes the system to show a step-by-step wizard which guides the user through the new profile creation.

- **ChangeUserProfile:** A blind user changes a user profile.
  This event is triggered each time a user wants to change the profile by which the system is configured. The event causes the system to stop, in order to perform the configuration again according to the different profile that is chosen.

- **ViewOrModifyUserProfile:** A user asks to view and possibly modify an existing user profile.
  This event is triggered each time, a configuration in an existing profile is asked to be viewed or changed.

- **DeleteUserProfile:** A user asks to delete an existing user profile.
  This event is triggered each time, a user profile is required to be deleted, in case it is unnecessary or reached the limited number of supported profiles.

- **Install3DSoundTech:** A user of the system, probably a researcher, wants to install a new 3D-sound technology.
  This event is mostly triggered by researchers, in order to install a new technology and test its production of 3D-Sound in the system.

- **Terminate:** The user wants to terminate the system by sending a signal, telling that all system's operation should stop:
  This event triggers a system termination process which causes all the components to stop working after saving all necessary data..

Hardware related events:

- **VisualizeImage:** Grabbing a new frame.
  The Image Processing Unit grabs a new frame, starting the pipe of analyzing the frame and delivering it to the Sound Creation Unit for the purpose creating a meaningful sound that would be positioned in the spatial audio environment.

- **CameraError:** The Image Processing Unit ***didn't*** manage to interact with the camera.
  This is a rare event, but it can happen because of camera disconnection or other hardware error. In this case, the system should respond appropriately (see below).

- **CameraConnected:** A web camera has been connected.
  This event triggers the system to start its operation in case it was stopped due to camera disconnection or any other error.

- **HeadphonesDisconnected:** Headphones have been disconnected.
  This event makes the system stop its operation, until the headphones are connected again.

- **HeadphonesConnected:** Headphones have been connected.
  This event triggers the system to start its operation in case it was stopped, since it is allowed to output the created and positioned sound to headphones only.

2 Functional Requirements

***Note**: The current version of the functional requirements describes the initial system. Some of the requirements may be changed/removed during the development.*

2.1 Operational Functionality

2.1.1 Installation

- A binary installation file will be supplemented.
- The installation will include the following components:
  - The "AISN" application.
  - The *OpenCV* library.
  - The *Rapture3D* library.
  - Basic set of HRTF datasets created by *CIPIC* and *Listen* projects.
  - The required hardware drivers (for more details see *SE Project Constraints* in *Non-Functional Requirements* part).
- The installation will check the hardware compatibility to the AISN application which should include a check that there exists a web-camera connected to the machine.
- The installer should notify the user if certain hardware is missing or not compatible with system requirements.

2.1.2 User Profiles Managing

- A user will be able to create new user profiles that will specify the type of image processing and 3D sound technology that will be used. Each user profile will receive a unique name and will be stored in the *Settings Data Storage* for future use.
- The preferred HRTF dataset and other preferences regarding the image and sound processing parameters will be saved alongside the profile.
- Each user profile will contain of the following parameters (at least):
  - name
  - 3D sound technology
  - Type of image processing and its parameters (see Functional Requirements *Image Processing* section below).
  - Sound types and other parameters related to sound creation and positioning (e.g. volume range) defined below.
- A user will have the ability to view the existing profiles, to modify and delete them.
- 2 default user profiles will be supplied in the system's first installation. these profiles will be unchangeable and could not be deleted:

Default 1 This profile uses a default HRTF dataset alongside Image Processing based on feature points.

Default 2 This profile uses an OpenAL Rapture3D implementation alongside Image Processing algorithm based on contour detection.

- A user will be able to define up to 50 custom user profiles.

2.1.3 Switching Between Profiles

A user will have the ability to switch between profiles and so to initialize the system with the parameters predefined in each one of them.

2.1.4 Training

A user will be able to perform a staged and narrated training. Training will be performed while the system is connected to the computer, and will differentiate between two types of training:

- 3D sound perception - For the purpose of identifying the correct sound location (on all axes: elevation, azimuth, depth).
- Environment understanding - Allowing the user to gain experience with obstacles detection. Since this is the hardest part, the training will guide the user on several levels, from general obstacle positioning to contours understanding.

The following trainings will be supported:

- Visualizing random shapes, including circles, squares and triangles with different sizes which are located in various locations. The following settings will be allowed to configure:
  - The number of shapes of each type shown at a single time period.
  - The types of shapes, including circles, squares and triangles.
  - The sizes, including a constant size and size range.
  - The time period for performing a refresh of the view.
  - The colors of the shapes.
- Visualizing pre-defined image files:
  - Ability to load image files for the purpose of their visualization.
  - Ability to specify in which order images will be processed, including a pre-defined order and random order.
  - Ability to specify time period to show the next image, including a pre-defined time period or key typing.
- Fully immersive use of the system by emphasis of some feature (such as 3D sound positioning, color recognition, distance recognition etc.). This kind of training is the most advanced one and should be used by experienced users which tried other kinds of training and succeeded on them.

2.1.5 New 3D Sound Technologies Installation

A user will have the ability to add and install two types of 3D sound technologies:

- OpenAL implementations which conforms with the official API.
  API can be found in the official OpenAL website:
  `http://connect.creativelabs.com/openal/default.aspx`
  You can find it in the documentation section under the name *"OpenAL_ Programmers_ Guide.pdf"*

- *Binaural spatialization* using HRTF datasets -

  - Each dataset should be a '.mat' (MATLAB Data File) of the following structure:
    2 structs (one for each ear) named 'r_hrir_S' and 'l_hrir_S' each contains the following fields:

    type_s With a constant value 'FIR'. *Not a mandatory field*

    elev_v Available elevation array. Containing all the elevation values being supplied by the dataset

    azim_v Azimuth values array. Containing all azimuth values matching to the elevation values in *elev_ v* ($\forall i$, *elev_v*[i] *and azim_v*[i] defines a single measurement).

    sampling_hz The *sampling frequency. Not a mandatory field,* if not specified, default is 44100_{Hz}.

    content_m The measurements data arranged in an array where each line contains a single elevation-azimuth pair measurement.

  - For each installed HRTF dataset, the system should provide visual information about the positions in which sound sources can be placed, according to the measurements the dataset contains.

2.2 Core Functionality

The main requirement of the system is enabling visually impaired people to hear their environment. For that purpose, we define the following functional requirements from which the system behavior and abilities can be derived. The functionality of the system is divided to three main aspects, which suit to the major capabilities of the system.

2.2.1 Image Processing

The system will use several types of image processing and computer vision methods. The user will be able to switch between the different methods as he wishes. For this purpose we define a set of requirements:

- The system will provide a list of predefined methods for image processing over OpenCV library.

- The system will provide a utility to add technologies implementing the pre-specified API and constraints.

This unit will pass its result to the next unit, the Sound Creation unit. Each method will return a different set of results that can be divided into two types:

Point-wise Values Returns a set of points available for transforming into sound. The points are selected according to several traits, including brighter areas enclosed within a darker one, or a multiple line segment intersection. Each point in the set will include the following details:

- Coordinates in the frame it was extracted from.
- Color in that area.

Continuous-wise Values Returns a continuous segments and bounding polygons, representing regions in the frame it was extracted from. Each region in the set will include the following details:

- A data structure representing the segments and polygons.

2.2.2 Sound Creation

The sound creation system, creates sound samples dynamically - according to the processed image it receives. The following parameters are considered during the sound creation:

- **Information about the brightness of the color at the position of the sound source**. The sound will represent the brightness of the color, so that the user would be able to say whether the sound represents a bright area or dark one.
- **Information about corners.** The sound will tell if a certain area is a corner of an object, so that the user would be able to realize where the corners of the visualized object are.

The following features should be configurable:

- The number of sounds sources used to visualize a detected object.
- The source of the played sound. The system will create the sound sample it plays for the end-user, but will enable an expert-user to specify files of sound samples to play.

2.2.3 Sound Positioning

The 3D sound technologies that will be used, supply various sound spatial information and each generated sound source will support the following features:

1. The highest *resolution* available by the used technology will be provided. There are two options for the locations provision:
   (a) By using a 3D sound existing algorithm implementation, which performs interpolations based on some hard-coded HRTF datasets. The location of the provided sound source should be of high resolution, closed to 1 degree (according to the technology).
   (b) By using an HRTF dataset simple algorithm, in which the provided locations strongly rely on the locations supported by the dataset. Therefore, for each desired location, an approximation should be made in order to make the source be heard from the closest location available.
2. The system will provide a list of all installed 3D sound technologies, which includes at least OpenAL implementation and HRTF dataset based technology.

3 Non Functional Requirements

3.1 Performance constraints

3.1.1 Speed requirements

- **Switching between user profiles:** Switching between user-profiles should take no more than 5 seconds.
- **Response Time:** The system will response to a change occurred in the physical environment from which the camera takes the images, within 0.1 seconds at most (processing 10 frames per second), we will strive to ~0.034 seconds (processing 30 frames per second).
- **Training Speed:** This issue is very important, since the system is designated to researchers as well as end-users. Therefore we define the following requirements:
  - A simple training made by several steps, should take no more than 1 hour for an average user.
  - In order to perform an extensive training, more time will be required. The success of the training will be measured by a set of tests (see an example at appendices part 5.3). The training time depends on the user using the system - a visually impaired user is likely to succeed more quickly in the process of training, because of his well-developed sense of hearing. The process may take more time for a regular (non blind) user.
  - It should be noted that these requirements are based on experiments with different 3D sound technologies, used in this system. The results may vary in ±10% between different users and may be improved by using better 3D sound technology, e.g. choosing a better HRTF dataset which suits the physical parameters of the user body.
- In case of an equipment error, such as web-cam or headphones disconnection, the system will detect the problem after 1 second at most and respond accordingly.

3.1.2 Capacity requirements

- The system will support many different OpenAL implementations and HRTF datasets, the limit is the hard disk capacity only.
- The number of used 3D sound technologies during the system operation is restricted to 1 technology at a time and a single dataset in case the technology is based on HRTF datasets.
- The system is restricted to get the images stream from a single web-camera and will manage only 1 device of this type, at a time.
- The number of audio output devices to which the system output will be transformed, is restricted a single set of headphones and only 1 device of this type will be managed at a time.
- **RAM Capacity:** The system should work on machines with at least 1 GB of RAM.
- **Hard Drive Capacity:** The system should not require more than 100 MB of hard disk space (<u>not including</u> the 3D sound technologies and OpenCV drivers as well as sound samples and HRTF sets).

- **CPU Usage:** The CPU usage required by the system should not exceed 20%±5% during the visualization operation (can be 40%±10% during initialization and training).

3.1.3 Recoverability requirements

- After a web-camera or headphones related errors are repaired by connecting the appropriate equipment, the system will recover the last stable state and continue working with the active user profile.
- In case of an electrical black-out, the system will return to the last active user-profile after powering up again.
- In case of failure to get a frame from camera (while the camera is still connected), the system will not crash, and try getting another frame.

3.1.4 Reliability requirements

- As said before, we measure the accuracy of the system by using a special set of measurements, which will be specified later, after performing experiments with the whole system (see an example testing report at appendices part 5.3).
- The system should deal with errors of wrong inputs while parameters changing, present error messages and allow to change the parameters to correct values.
- The system should deal with equipment connection and disconnection in real time, present appropriate messages - alerting and notifying the performed operation along with sound alerts and notifications intended for visually impaired users.

3.1.5 Portability requirements

- Currently, the system is designed to be deployed on Microsoft Windows operating systems only.
- The supported versions of the operating system will be MS Windows XP (SP3 must be installed) / Vista / 7 and later versions (which support programs written under earlier versions of Windows).
- The system will be compatible with 32 / 64 bit machines having web-camera and audio drivers installed.

3.1.6 Availability requirements

- The system is required to be operable, in a functioning condition and in a stable state most of the time. The percentage of the time in which the system is unavailable to the user (in case it is initialized properly) should not exceed 5%.
- In case of an error in the user profiles database, the system should still be available to the user by using default settings.
- The system UI, viewed and heard by the user, should always stay usable and continue monitoring the equipment connections and disconnections.

3.2 Look, Feel and Use constraints

3.2.1 User interface requirements

- User interface will be in English.
- Simple, clear and readable Graphical User Interface.
- There will be no use of colors that associate with blind colors.
- All the error and information messages will be heard via the headphones for the blind user using the system.

3.2.2 Ease of use requirements

- The application will have a help menu that will include a detailed explanation for every operation and also a summary of the operation usage and functionality.
- Popular functionalities such as starting and stopping the system, will have a keyboard shortcut.
- Operations that cannot be done at a moment, due to the system's state, will not appear on the screen and will be blocked if executed from the keyboard.
- Complex operations will be implemented as wizards.

3.2.3 Ease of learning requirements

- The system should be easy to use even for users that are not well familiar to the computers technology.
- The training period regarding the basic operations required by the end users, in order to start/stop the system and switch between user profiles, should take a very short period of time, 20 minutes at most.

3.3 Platform constraints

- As mentioned above, the system is currently designed to run on MS Windows only. We chose this platform since the current OpenAL implementation we are using is Rapture3D of BlueRippleSound company which relies on DirectSound software installed on a computer with Windows operating system only. We will consider later to extend the implementation to support Unix based systems too, for the purpose of deploying it to portable devices with Android operating system.
- The components of the application core and the UI will be written in C++ language, using .NET 3.5 Framework with Graphical User Interface framework supported by Visual Studio 10.0 IDE. This VS IDE supplies all kinds of features that are relevant for our work and supports the various functionality that is required for the project. This functionality includes a simple interface to work with, project sharing systems and powerful mechanisms of auto-completeness, compiling and unit-testing.

3.4 SE Project Constraints

- At the end of the current academic year, the first version of the system will be presented. No demos are required, since the system will be interactive and the end-user will be able to activate, deactivate and configure it.

- During the development stage of the system, we will use a home-made cheap and simple device constructed from a PC web-camera strapped or taped to a top of stereo headphones (shown in figure 2). For the demo and testing purposes, a real device (shown in figure 3) might be supplied by DT labs. This device is a video sunglasses that offer a consistent alignment of the camera with the head by having the tiny camera hidden in the nose bridge of glasses. The glasses' legs behave like tiny headphones allowing to use the glasses as headphones along with the head-mounted camera. This device supports a more stable camera viewing direction along with an unobtrusive appearance of the system's user.

Figure 2: A home made device which will be used during the development period

Figure 3: A device like this might be used for the demo and testing purposes
Note: *This device is shown here for demonstration purposes only and the real device may look different*

- The system will employ MATLAB as a computation engine for performing the heavy math calculations required to produce 3D sound (in the HRTF based implementation we will support). The engine programs are standalone C/C++ or FORTRAN programs that communicate with a separate MATLAB process (via pipes on UNIX systems, or) through a Microsoft Component Object Model (COM) interface, on Microsoft Windows systems. Therefore, we are limited to work only on machines having MATLAB installed with an appropriate license.

- The system will be developed on several machines including our personal machines and machines in the university labs. The 3D Sound Positioning Unit will be developed on few machines only, since it requires a commercial version of Rapture3D OpenAL implementation which are a commercial software and therefore a license is needed for using it. In addition, for performing experiments regarding the 3D sound positioning, a high quality hardware will be used. These hardware is borrowed from the acoustics laboratory in BGU and includes a high quality external sound card and headphones.

- The system will be tested on both sighted and visually impaired users. For each user, several configurations will be tried - according to his personal feedback. The inputs will come from the physical environment in which the experiments will take place. But, in addition we will test specific inputs, according to accuracy measurements described in the appendices part. These inputs will be simulated in the enclosing environment of the participant user. During the tests, random data will be used, such as random shapes recognition, simulated by the training part of the system.

- The final system will be presented on one of the test machines having all the software installed on it and using the professional device described above.

3.5 Special restrictions & limitations

3.5.1 Technical documentation

The code written by AISN programmers must include some text along with it, to describe various aspects of its intended operation. Fine documentation will help making changes possible, as we expect from our client that basic and advanced level of requirements may change. Moreover, since our code is tested carefully, such explanations about the programmer intentions will help the tester to do his job. All the methods in the source code should be documented.

3.5.2 Implementation Constraints

- The system will be developed to be flexible, in order to support changes.
- Since our system will be used as a testing environment, there is a strong requirement for modularity, to allow researchers to make changes in different parts of the system. The system will be developed such that it will allow to replace each of the main parts easily and without the requirement to understand the whole code.

4 Usage Scenarios

4.1 User Profiles — The Actors

As it was described above, our system will support two classes of users intended to perform different operations using it: an end user and a researcher. The description below categorizes each user to one of these classes.

Note: *We treat the web-cam and the headphones as I/O application devices and therefore they were not used as actors within the described usage scenarios.*

4.1.1 End-Users

These users are the main users of the system, we can divide them into two subtypes:

Blind Users The system is mostly intended for users of this type. By the term "blind" we mean visually impaired users. Users of this type use the system as a sensory substitution device. The intention is to make this system serve them instead of their visual system and help them to feel their visual environment. The researchers also are assisted with these users in order to perform experiments about the ability of the hearing to come instead of the sight sense. However, for testing purposes, the system can be used by a person with a sight sense that is not damaged, whose eyes are covered with a blindfold.

Configuration Users A user of this type helps the blind user to perform initialization operations required for the system installation and first tuning. These operations include first installation, profile creating and setting of the required parameters. In addition, the blind user will be aided by a configuration user in order to successfully pass the training stage which will present him the system and make is suit his characteristics in the best way.

4.1.2 Researchers

As it was written above, one of the goals of the system is to provide a testing environment for researchers which want to test the ability of the hearing sense to replace the sight sense. The system allows researchers to manage user profiles which store different configurations and settings of the parameters regarding image analysis and 3D sound creation and positioning. In addition, the system allows the researcher to install a new 3D sound technology which conforms to some required interface, thus allowing him to test different kinds and implementations of 3D sound positioning technologies. By all this our system is an unprecedented experiment tool for the neuroscience researchers.

4.2 Use-cases

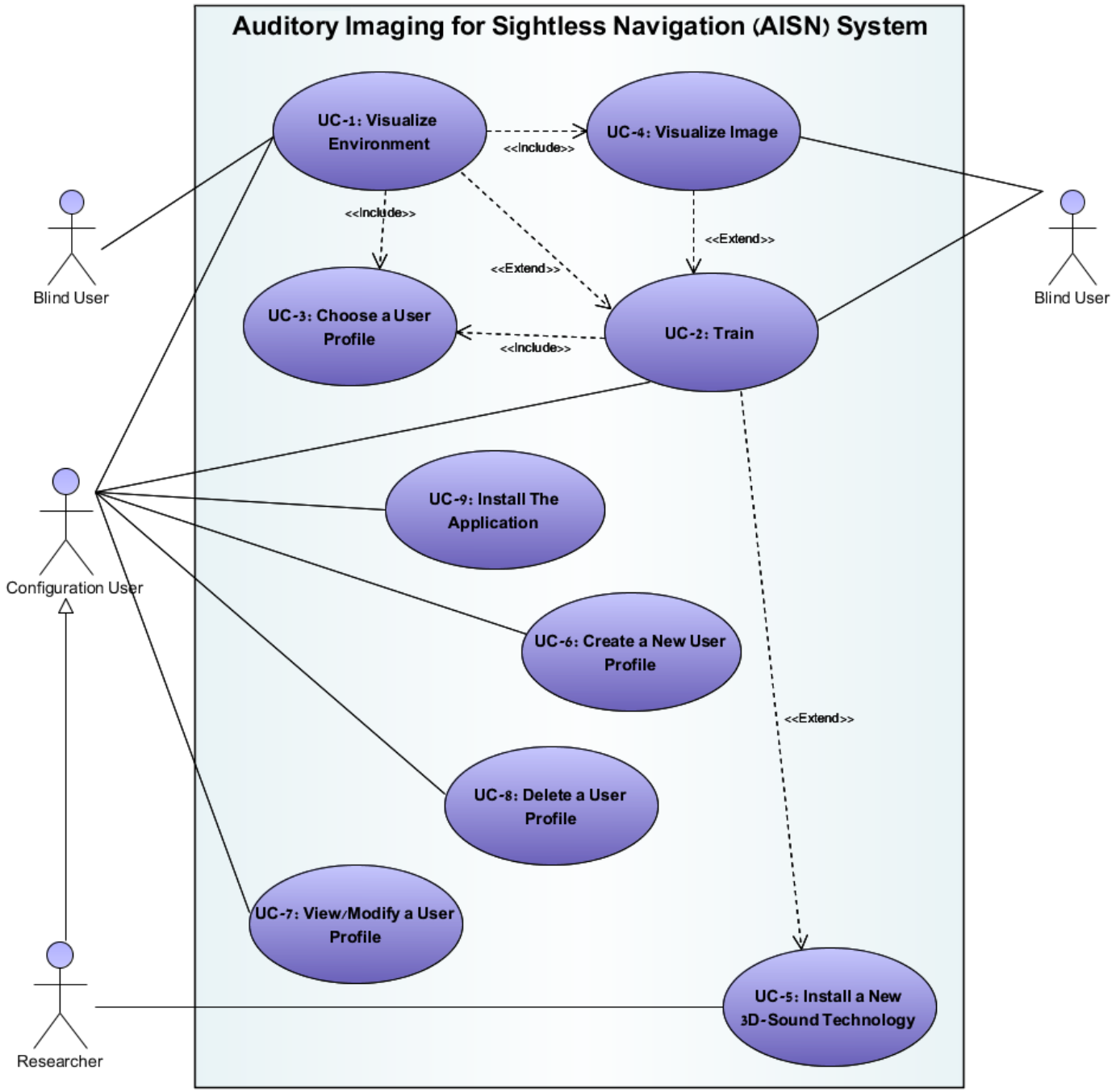


Figure 4: Use Case Diagram

4.2.1 UC-1: Visualize Environment

Summary A blind user wants to start a visualization process that will allow him to understand what is located in front of him at any time, by playing 3D-sounds in the appropriate positions, according to the physical environment around him. During the process, a frame is taken from the camera every constant period of time (predefined in the system settings) and a chain of processing events that ends up with an output of sound is carried out.

Description in a natural language An end-user presses the "Visualize" button located on the welcome screen and presented by the System UI. The user is asked to select a user profile, which contains all the requited parameters that are used in order to initialize the system. Then the user triggers the system to start a visualization process, which causes the processing units to operate in the following chain form: the Image Processing unit receives an image frame from the webcam, processes it and takes the relevant visual information out of it. The received information is transferred to the Sound Creation unit which generates sound samples. The created sounds are transferred to the Sound Positioning unit which outputs them to the end-user via headphones. This process is cyclic and repeats until the user stops its operation.

Actors

- **An end user** – who wants to experience a visualization of his environment and to whom the created sounds are transferred.

Trigger The blind user presses on the "Visualize" button located on the welcome screen and presented by the System UI. Probably, at first, the blind user is guided by a sighted person which serves as a configuration-user and initializes the appropriate configuration required for the visualization.

Preconditions

- The webcam is connected to the computer, properly installed and turned on.
- The headphones are connected to the computer, properly installed.
- The system is initialized properly - all its components are initialized and running.

Stakeholders and Interests

The end user wants to hear accurate information related to his physical environment, which is updated in real time and contains as much information as possible.

Main success scenario

1. The user presses the "Visualize" button located on the welcome screen and presented by the System UI.
2. UC-3 ("Choosing a User Profile") is executed.
3. The system takes an image frame from the webcam.

4. UC-4 ("Image Visualization") is executed with the retrieved frame as an input.
5. If the user asked to stop
   - 7.a. End of use case.
6. Else, Goto 3.

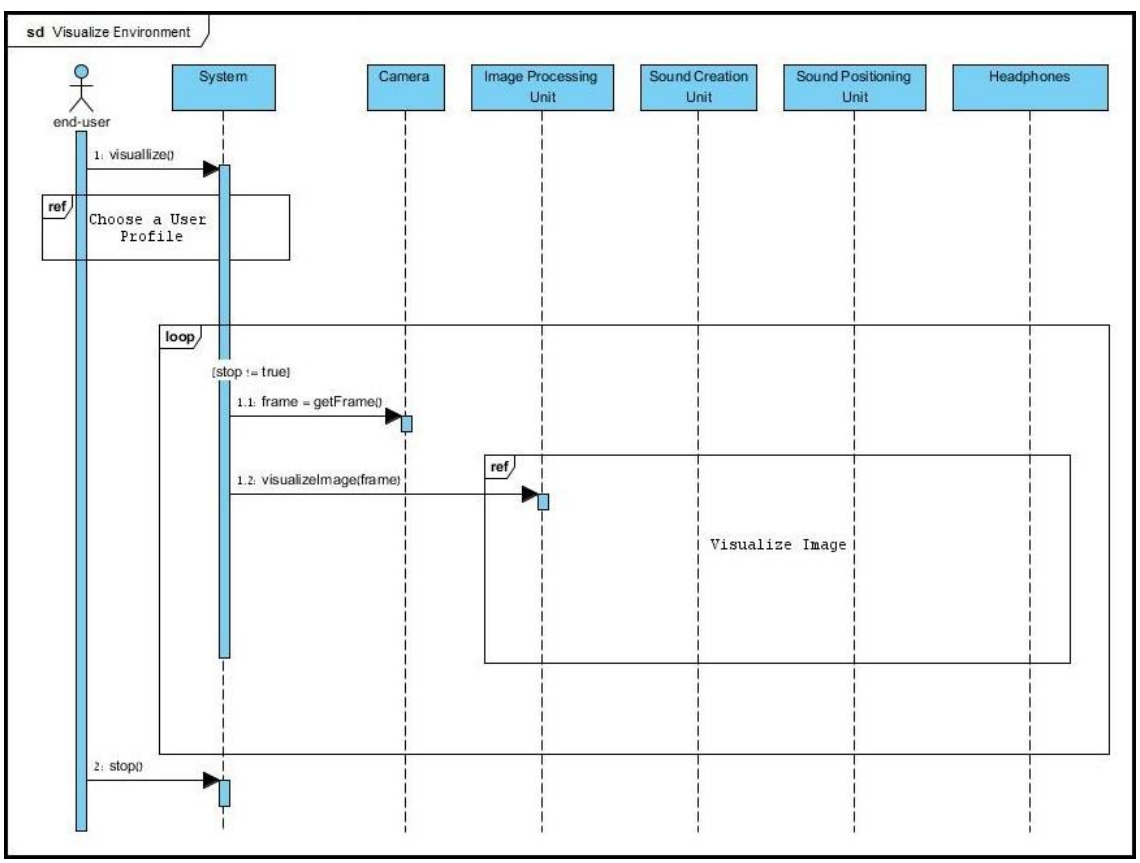
Alternative flows and Exceptions

- **<u>3.a.</u>**
  - <u>Case:</u> The image taking failed.
  - <u>Action:</u>
    1. An error message is presented to the user along with a sound which indicates a webcam recognition error.
    2. End of use case.

Priority and Frequency of occurrence This usage scenario is the main usage scenario of the system and lasts for a long period of time. This is the most frequent and important operation of the system and is also required to be performed in real time and with as little latency as possible. Therefore this use case has an urgent priority.

Covering Requirements 2.2

Sequence Diagram



4.2.2 UC-2: Train

Summary A blind user wants to perform a training before using the visualization functionality of the , which will allow him to learns about the operations required to start working with the system. This functionality allows the user to practice in recognizing the 3D-sounds the system creates and position, according to different inputs taken from the physical environment encloses the user. This training allows the user to exercise his conscious mental analysis and activate the visual areas of the brain for visual processing. More information about the support of training ability of the system can be found in the appendices part.

Description in a natural language The user presses the "Training" button located on the welcome screen and presented by the System UI. The user is asked to select a user profile which contains all the requited parameters that are used in order to initialize the system. After a profile is selected, the training screen is presented to the user and he is asked to select the type of training he wants to exercise with. The training types available by default are:

1. Visualizing random shapes, including circles, squares and triangles with different sizes which are located in various location.
2. Visualizing pre-defined image files.
3. Fully immersive use of the system by emphasis of some feature (such as 3D-sound positioning, color recognition, distance recognition etc.). This kind of training is the most advanced one and should be used by experienced users which tried other kinds of training and succeeded on them.

After choosing the desired training, the user is asked to fill the required settings of the training. In case of random shapes, these settings will include the minimal and maximal number of shapes, their kinds, size etc. In case of image recognition, the user is asked to choose between using the default set of images and providing his own set of images by giving a path. Then he will choose whether they will be visualized in sequence or randomly, the time period each image will be shown etc. Otherwise, general settings are required such as the latency period (after which the camera view is updated) and the like. After all the required parameters are filled, the training starts. During the training, the user can change the settings or stop the training and return to the welcome screen.

Actors

- **An end user** - who wants to perform a training of the system and gain an experience with mental analysis connecting between vision and hearing.

Trigger The user presses the "Training" button located on the welcome screen and presented by the System UI. Probably, the blind user is guided by a sighted person which servers as a configuration-user and initializes the appropriate configuration required for the training.

Preconditions

- The system is initialized properly - all its components are initialized and running.
- The UI of the system is presented on the screen.

Postconditions

- The selected user profile fits to the user characteristics (the user is expected to change it during the training phase and make it suitable for him as much as possible).

Stakeholders and Interests

The user who configures the training operation wants and easy and fast configuration process which won't put him through a complex process of parameters changing. In addition, he wants the configuration process to be error tolerated and notify him about bad values of parameters and other missing information required for the training.

The end user wants the system to provide him various kinds of training which will allow him to learn the system in stages with increasing levels of difficulty. This user also wants the system to give him as much feedback as possible, in order to allow him to perform training with as little effort as possible and with as little help of a sighed person as possible. The last requirement is necessary, since a feedback is the only way for the user to know whether his mental interpretation of the view is correct.

Main success scenario

1. The user presses the "Training" button located on the welcome screen and presented by the System UI.
2. UC-3 is executed.
3. The training screen is presented to the user and he is asked to select the type of training he wants to exercise with.
4. The user chooses the desired training type and presses "next".
5. The system asks the user to fill the required settings for the chosen training type.
6. The user fills the required parameters and presses "next".
7. If the training type is random shapes visualization:
    - 7.1. The system creates a sound that represents a random shape positioned in the 3D auditory space.
    - 7.2. The system outputs the created sounds to the user.
    - 7.3. Return to 7.a.
8. Else, if the training type is images visualization:
    - 8.1. The system retrieves the next image to play, according to the parameters set by the user.
    - 8.2. UC-4 ("Image Visualization") is executed with the retrieved image as an input.
    - 8.3. Return to 8.a.

9. Else

    - 9.1. The system creates a sound through emphasis of some feature.
    - 9.2. The system outputs the created sounds to the user.
    - 9.3. Return to 9.a.

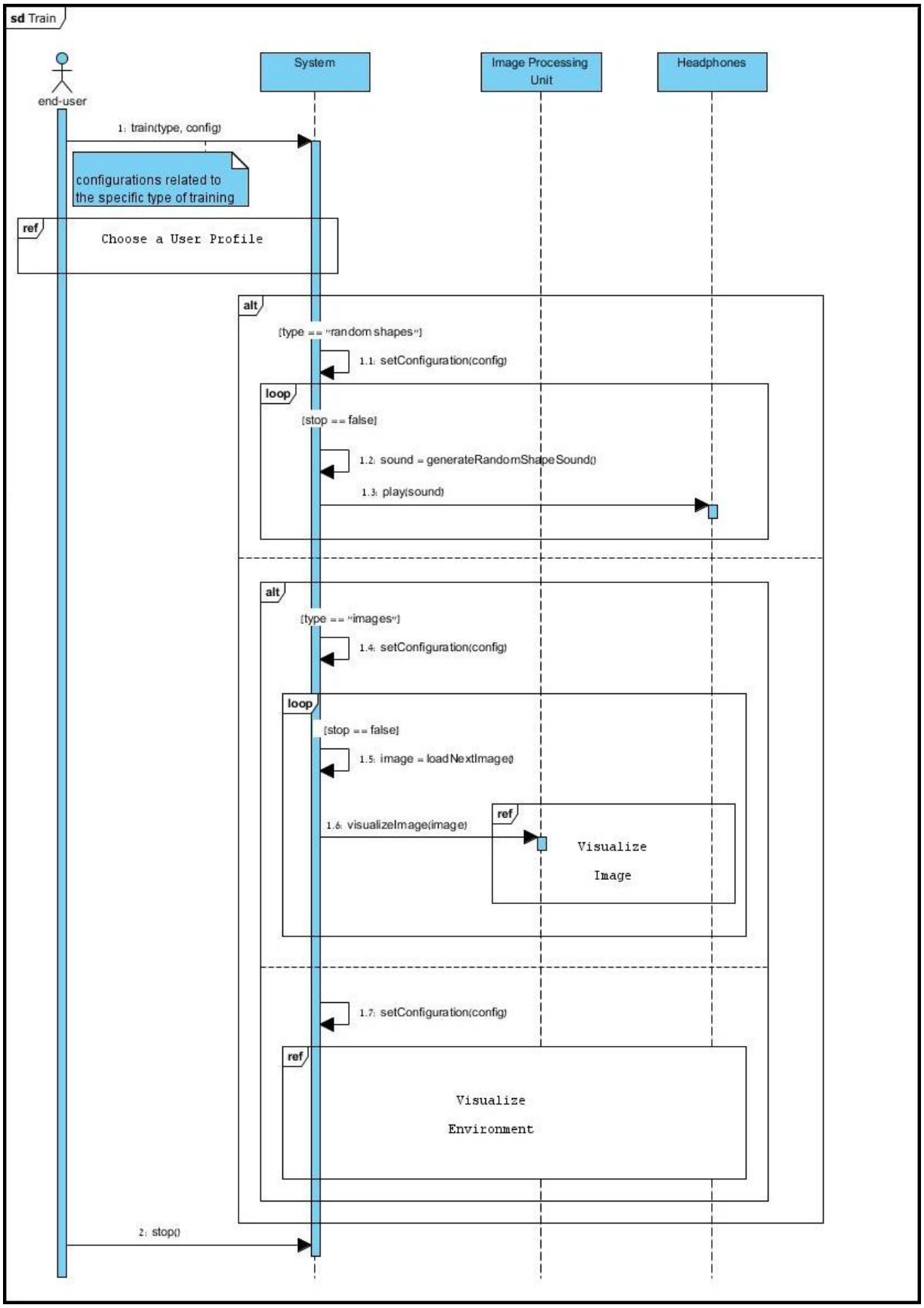
Alternative flows and Exceptions

- **6.a.**
    - Case: Not all required information has been provided - one of the fields is missing or incorrect
    - Action:
        1. The system asks the user to provide the missing information or to correct the mistake.
        2. Return to 5.
- **7.a. + 8.a. + 9.a.**
    - Case: The user presses on "Change Training Type" button
    - Action:
        1. The training stops.
        2. Return to 3.
- **7.b. + 8.b. + 9.b.**
    - Case: The user presses on "End of training button"
    - Action:
        1. The training stops and the system returns to the welcome screen.
        2. End of use case.

Priority and Frequency of occurrence This use case is expected to be performed very often and is essential in order to allow users to learn the system's way of operation and practice using it. The success or failure of using the system can't be checked without the training period, during which the user will learn how to use the system and gain experience. Only then, we can check whether the system is useful for this user and helps him to navigate or at least understand his environment.

Covering Requirements 2.1.4

Sequence Diagram



4.2.3 UC-3: Choose a User Profile

Summary A blind user chooses an existing user profile for the purpose of performing a training or in order to use the system as a sensory substitution mechanism, which is the main purpose of the system.

Description in a natural language This use case occurs as a part of the Visualization use case (see UC-1) or Training use case (see UC-2). After the user chooses the desired operation, the system presents to the user a list of the available profiles. The user chooses the desired profile and the system retrieves the profile details from the Settings Data Storage and initializes parameters accordingly. In case of an initialization error, the user gets an error message and is required to choose another profile or to stop the process of the visualization / training. In case of successful operation the system notifies the user by a message along with a suitable sound. Then, the system continues its operation from the appropriate use case.

Actors

- **A blind user** - who wants to use the system for sensory substitution in a "real" environment or for training. Here we note that the operation is initiated by the blind user but actually performed by a sighted person which serves as a configuration-user which helps the blind user.

Trigger The blind user which wants to choose a desired profile in order to start using the system

Preconditions

- The system is initialized properly - all its components are initialized and running.
- The user chose one of the options - visualization (UC-1) or training (UC-2).

Postconditions

- The system is initialized with the desired profile details.

Stakeholders and Interests

The blind user wants a fast process of profile choosing which takes a very short time and involves as little help of a configuration-user as possible.

The configuration user wants an easy process of profile choosing which will allow him to choose the required profile from a list of available ones without being needed to scroll through a long list of profiles, but by writing the first letters of the profile name. In addition, this user wants the system to return an informative message about the status of the initialization process, performed after the profile choosing.

Main success scenario

1. The system presents the user a list of the available profiles.
2. The user chooses the desired profile and presses "OK".
3. The system retrieves the required profile details from the Settings Data Storage and initializes according to them.
4. The system notifies the user about a successful profile choosing by a message along with a suitable sound.
5. Return to the appropriate use case (UC-1 or UC-2).

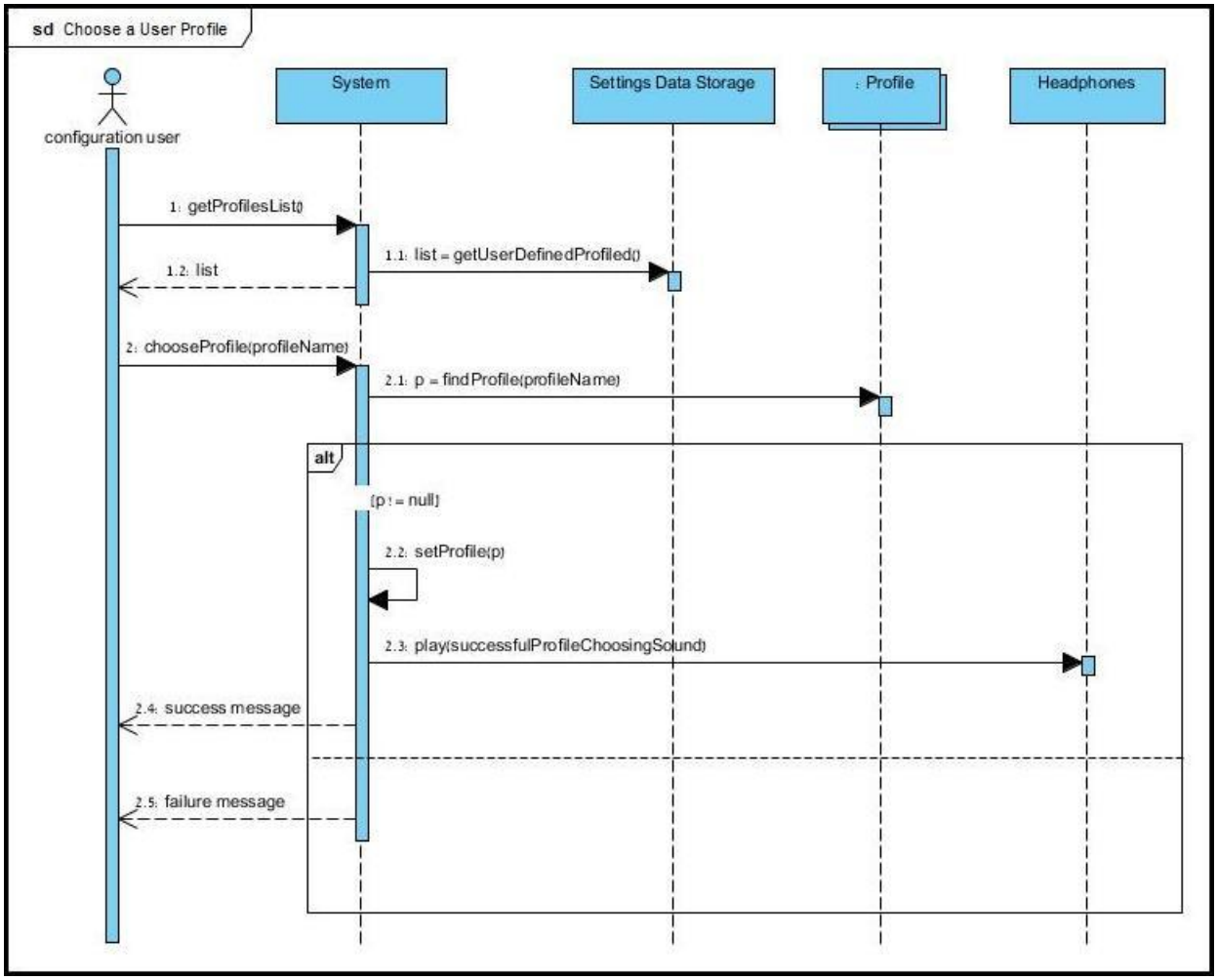
Alternative flows and Exceptions

- **3.a.**
  - Case: Database read error
  - Action:
    1. A message that apologizes for the inconvenience and indicates that an error has occurred is presented to the user along with an appropriate sound.
    2. The user is asked whether he wants to initialize the system with default parameters or to try later.
    3. If the user chooses default initialization:
       * 3.a The system initializes with default parameters.
       * 3.b Goto 4..
    4. Else, End of use case (the initial use case, UC-1 or UC-2 ends too and) the system returns to the welcome screen.
- **3.b.**
  - Case: The chosen user profile isn't found in the database.
  - Action:
    1. An appropriate error message is shown along with a generation of an appropriate sound.
    2. The user profile name is removed from the available user profiles list.
    3. An option to create a new user profile is presented.
    4. If the user chooses to create a new profile:
       * 4.a Goto UC-5.
    5. Return to 1.
- **3.c.**
  - Case: The initialization process according to the given user profile was failed.
  - Action:
    1. An appropriate error message is shown along with a generation of an appropriate sound.
    2. Return to 1.

Priority and Frequency of occurrence This use case occurs every time the user starts a visualization or training process, which requires an initialization of parameters. This is done by choosing a user profile from a list of existing profiles, which contains default profiles and some pre-defined ones.

Covering Requirements 2.1.3

Sequence Diagram



4.2.4 UC-4: Visualize Image

Summary This is the core of a visualization process, which allows the user to understand what appears in an image, by playing 3D-sounds in the appropriate positions around him. During the process, there is a chain of processing events on the given image, that ends up with an output of sound.

Description in a natural language This use case occurs as a part of the "Environment Visualization" use case (see UC-1) or "Training" use case (see UC-2). When the part of visualizing arrives, this use case is called with an image as its input and causes the processing units to operate in the following chain form: the Image Processing unit processes the received image and takes the

relevant visual information out of it. The received information is transferred to the Sound Creation unit which generates sound samples. The created sounds are transferred to the Sound Positioning unit which outputs them to the end-user via headphones.

Actors

- **An end user** – who wants to experience a visualization and to whom the created sounds are transferred.

Trigger This use case is called by some other use-case in order to perform a visualization of an image.

Preconditions

- The webcam is connected to the computer, properly installed and turned on.
- The headphones are connected to the computer, properly installed.
- The system is initialized properly - all its components are initialized and running.

Stakeholders and Interests

The end user wants to hear accurate information that contains as much information as possible.

Main success scenario

1. The system retrieves an image for visualization.
2. The system processes the image using the Image Processing unit, retrieves visual information and transfers it to the Sound Creation unit.
3. The Sound Creation unit creates the sounds which are relevant to the visual information and transfers them to the Sound Positioning unit.
4. The Sound Positioning unit manipulates the sound and places it in the appropriate position in the 3D auditory space.
5. The system outputs the created sounds to the user.

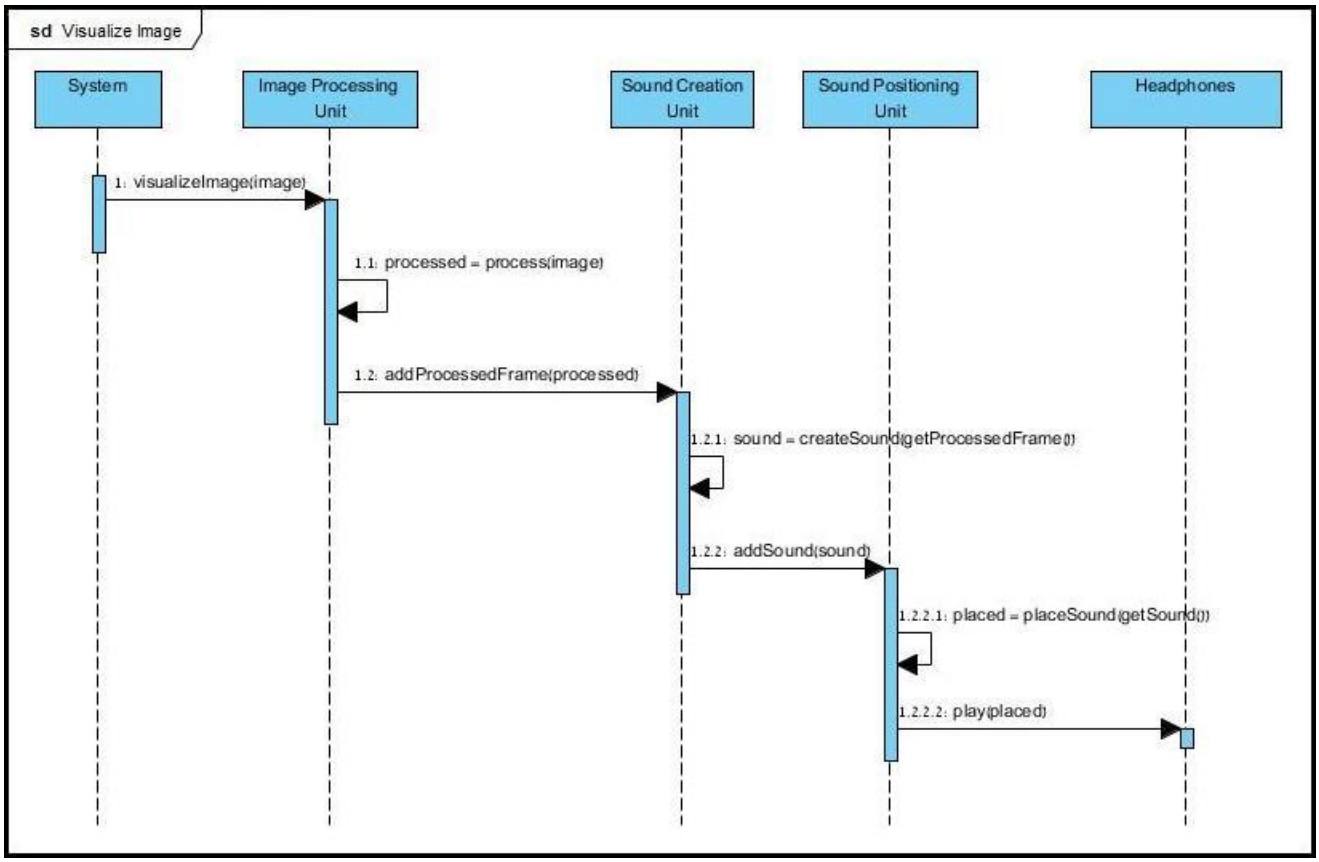
Alternative flows and Exceptions

- **5.a.**
  - Case: The headphones got cut off.
  - Action:
    1. An error message is presented to the user along with a sound which indicates headphones recognition error.
    2. End of use case.

Priority and Frequency of occurrence This usage scenario is occasionally called by the "Training" use-case and always called by the "Environment Visualization" use-case which is the main usage scenario of the system and lasts for a long period of time. This is a very frequent operation of the system and is also required to be performed in real time and with as little latency as possible. Therefore this use case has an urgent priority.

Covering Requirements 2.2

Sequence Diagram



4.2.5 UC-5: Install a New 3D-Sound Technology

Summary A user of the system, probably a researcher, installs a new technology which will be used by the 3D-sound positioning part of the system.

Description in a natural language The user presses on the "Install a New 3D-Sound Technology" button presented by the System UI. The system presents a wizard which guides the user through the process of new technologies installation. First, the system asks the user whether he wants to install an OpenAL based technology or HRTF based one. After the user chooses the option he wants, he is asked to provide a full path to the *.DLL file which contains the implementation in case of OpenAL based technology, or the *.MAT file in case of a pure HRTF dataset. After the user enters the path, the system performs a check whether the installed file conforms to the supported 3D-sound technology interface. In case of a discrepancy, an error message which contains a description of the error is shown. Otherwise, the system saves the new technology in

the Sound Spatialization Data Storage and presents a message, notifying the user about a successful installation of the technology. Then, the system asks the user whether he wants to test the installed technology right now. In case of a positive answer, UC-2 is executed, otherwise the system returns to the welcome screen.

Actors

- **A researcher** - who is the most probably to perform an operation like that, since it involves a deep understanding of the system's structure and the requirements of the 3D-sound technologies it supports.

Trigger The researcher presses on the "Install a New 3D-Sound Technology" button presented by the System UI.

Preconditions

- The system is initialized properly - all its components are initialized and running.
- The welcome screen is presented by the System UI.

Postconditions

- The new technology is installed on the system and the Sound Spatialization Data Storage is updated with the required data.
- A new profile can be created and set to use the new technology.

Stakeholders and Interests

The researcher performing the installation wants an fast process that checks the compatibility of the technology with the system's requirements for 3D-sound technologies.

Main success scenario

1. The user presses the "Install a New 3D-Sound Technology" button presented by the System UI.
2. The system presents a wizard which guides the user through the process of new technologies installation.
3. The system asks the user whether he wants to install an OpenAL based technology or HRTF based one.
4. The user chooses the option he wants and presses "next".
5. The user is asked to provide a full path to the technology implementation / dataset file.
6. The user chooses the required path and presses "next".
7. The system performs a check, whether the installed file conforms to the supported 3D-sound technology interface.

8. The system saves the required technology in the Sound Spatialization Data Storage and presents a message notifying the user about a successful installation of a new technology.
9. The system asks the user whether he wants to test the installed technology right away.
10. In case of a positive answer, UC-3 is executed.
11. Otherwise the system returns to the welcome screen.

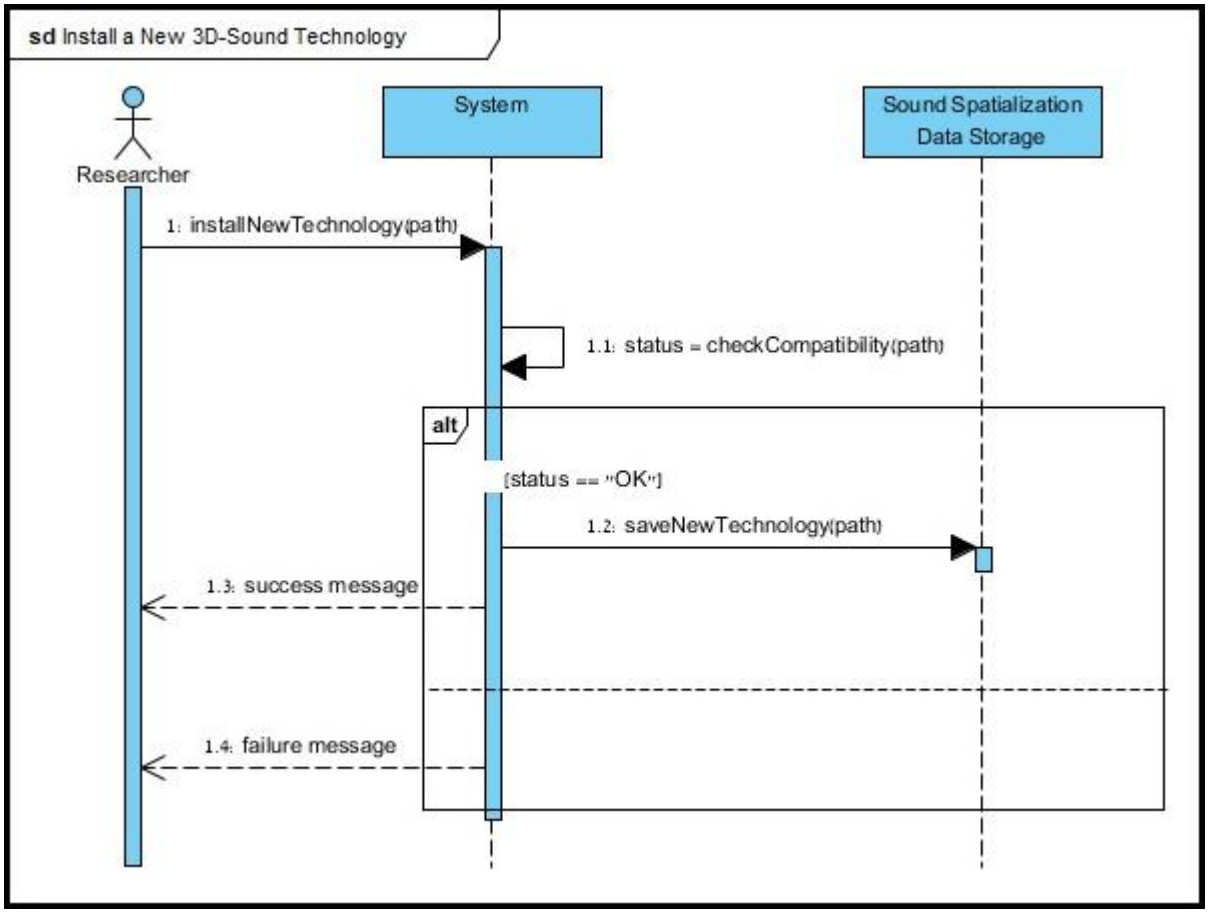
Alternative flows and Exceptions

- **6.a.**
  - Case: The required file wasn't found in the given path.
  - Action:
    1. The system presents a message notifying the user that the file does not exist.
    2. The user is asked to choose a file from another path or to cancel the technology installation.
    3. If the user wants to choose another path:
       * 3.a Return to 5.
    4. Else, Goto 11.
- **7.a.**
  - Case: The given technology doesn't conform with the interfaces the system supports.
  - Action:
    1. The system presents a message notifying the user about the differences between the given technology and the interfaces supported by the system.
    2. The user is asked to choose a different technology or to cancel the technology installation.
    3. If the user wants to choose another technology:
       * 3.a Return to 5.
    4. Else, Goto 11.
- **8.a.**
  - Case: Database write error
  - Action:
    1. A message that apologizes for the inconvenience and indicates that an error has occurred, is presented to the user.
    2. The user is asked whether he wants to try saving again, or to stop the process of the technology installation.
    3. If the user chooses to try again:
       * 3.a Return to 8.
    4. Else, Goto 11.

Priority and Frequency of occurrence This use-case is rarely executes since the system supplies two technologies (one of each type) in the first installation. Therefore it is unreasonable that a regular user will install a new 3D-sound technology. Probably, this use case will be utilized only by researchers and therefore it has low priority.

Covering Requirements 2.1.5

Sequence Diagram



4.2.6 UC-6: Create a New User Profile

Summary A user of the system wants to create a new profile which will store the preferred configuration of the system for a visually impaired user.

Description in a natural language The user presses the new profile creation button. The system presents the new profile creation wizard which will guide the user through the process of the profile creation. The user chooses a name for the new profile and the system asks him if he wants to create the new profile from scratch, or fill the parameters with an existing profile and then change some of them. In case of an empty profile, the system presents a screen in which all the parameters has empty values, otherwise a list of existing profiles is shown and the user chooses the desired profile from which the parameters will be taken. First, he is asked to fill parameters regarding the 3D-sound technology. These parameters include the preferred 3D-sound

technology, uploading pre-defined sounds or asking the system to create sounds according to the image analysis and some other parameters. Then, image processing screen is presented, in which the user is asked to fill parameters regarding the image analysis he prefers. The parameters include type of the prefers image analysis, which should be chosen from a list of available types, such as edge detection, feature points detection or solid objects detection. Other parameters are chosen according to the image analysis type. After the user chooses all the required parameters, the system saves the created profile in the Settings Data Storage and presents a message notifying the user about a successful profile creation. Then, the user is asked whether he wants to perform some training. In case he does, UC-2 is executed, otherwise the user is asked if he wants to start the visualization process immediately. In case of immediate execution, UC-1 is executed, otherwise the system closes the wizard and returns to the welcome screen.

Actors

- **A non-blind user** - who has the ability to perform a configuration of the system, through the UI.

Trigger The user presses the "Create Profile" button located on the welcome screen and presented by the System UI.

Preconditions

- The system is initialized properly - all its components are initialized and running.
- The UI of the system is presented on the screen.

Postconditions

- The new profile is created and the Settings Data Storage is updated with the new profile.
- The new profile is available from the profile choosing screen.

Stakeholders and Interests

The user which creates the profile wants an easy and fast process with interactive and error tolerated wizards which will guide him through the profile creation process and allow him to undo each performed operation.

The blind user for whom the profile is created wants a fast process of the profile creation which will allow him to use the application as fast as it possible, with as little effort as possible in order to fit the profile to his needs in the best way.

Main success scenario

1. A non-blind user presses on the new profile creation button located in the welcome screen and presented by the System UI.
2. The system presents the new profile creation wizard.
3. The user chooses a name for the new profile and presses "next".

4. The system asks the user if he wants to create the new profile from scratch or fill the parameters with an existing profile and then change some parameters.
5. The user chooses the preferred option and presses “next”.
6. In case of an empty profile, the system presents a screen in which all the parameters has empty values.
7. 3D-sound parameters screen is presented and the user is asked to fill parameters regarding the 3D-sound creation and positioning.
8. The user fills the preferred parameters and presses “next”.
9. Image processing parameters screen is presented, in which the user is asked to fill parameters regarding the image analysis he prefers.
10. The user fills all the required parameters and presses “next”.
11. The system saves the created profile in the Settings Data Storage and presents a message notifying the user about a successful profile creation.
12. The user is asked whether he wants to perform a training.
13. If a training is chosen:
    - 13.a UC-2 is executed.
14. Else, the user is asked if he wants to start the visualization process immediately.
15. If he chooses immediate visualization:
    - 15.a UC-1 is executed.
16. Else, the system closes the wizard and returns to the welcome screen.

Alternative flows and Exceptions

- **<u>3.a.</u>**
  - <u>Case:</u> There already exists a profile with the given name
  - <u>Action:</u>
    1. The system informs the user that a profile with the given name already exists and asks to change the name or cancel the profile creation.
    2. If the user chooses to change the name:
       * 2.a Return to 3.
    3. Else, Goto 16.
- **<u>3.b.</u>**
  - <u>Case:</u> The given profile name has an incorrect format (contains less than 4 characters).

- Action:
  1. The system asks the user to correct the profile name or cancel the profile creation.
  2. If the user chooses to correct the profile name:
     * 2.a Return to 3.
  3. Else, Goto 16.

- **5.a.**
  - Case: The user chooses to create the profile by modifying some of the settings of an existing one.
  - Action:
    1. The list of available profiles is presented to the user.
    2. The user chooses the desired profile and presses "next".
    3. The system retrieves the required profile from the Settings Data Storage and fills the empty values of the new profile with these settings.
    4. If a database error occurred while retrieving the required settings from the System Data Storage:
       * 4.a The system presents an appropriate message to the user and returns to 1 in this actions list.
    5. Else, The user is asked to change some settings.
    6. The user changes the required settings and presses "next".
    7. Goto 11.

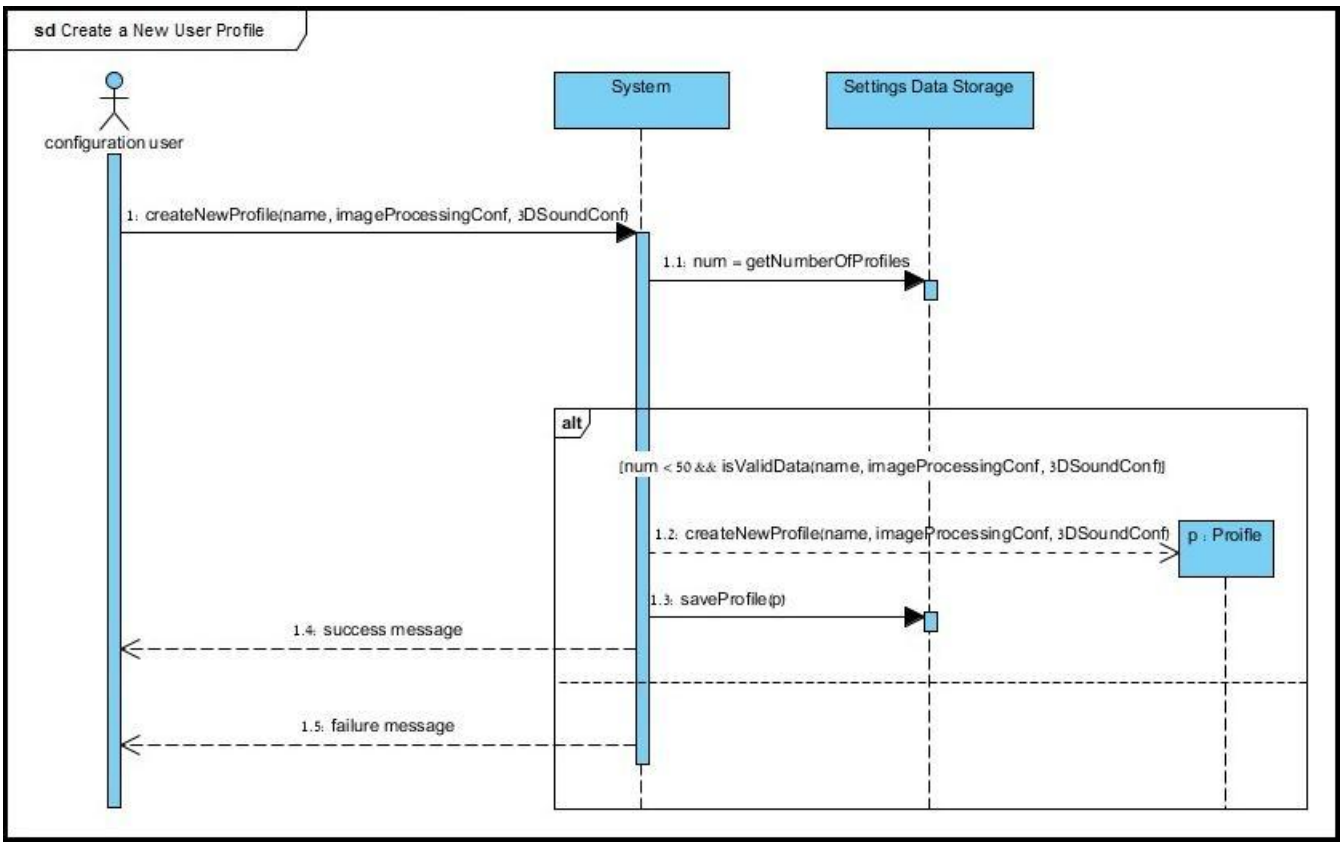
- **8.a. + 10.a.**
  - Case: Not all required information has been provided - one of the fields is missing.
  - Action:
    1. The system asks the user to provide the missing information.
    2. Return to 7 / 9 (accordingly).

- **11.a.**
  - Case: Database write error
  - Action:
    1. A message that apologizes for the inconvenience and indicates that an error has occurred is presented to the user.
    2. The user is asked whether he wants to try the saving again, or to exit the profile creation wizard..
    3. If the user chooses to try the saving again:
       * 3.a Return to 11.
    4. Else, Goto 16.

Priority and Frequency of occurrence The use case described above is performed each time a user wants to create a new user profile. This operation isn't a strong requirement for a new user since one of the default profiles, which are provided by the system, can be used. Nevertheless, creation of a new profile which suits best for each user is encouraged, since it allows the system to be optimized for the blind user in the optimal way. The use case is performed rarely in case the system is used by a blind user, but can be performed very often by a researcher. Thus, the use case has a middle priority.

Covering Requirements 2.1.2 (the first, second and third requirements)

Sequence Diagram



4.2.7 UC-7: View/Modify a User Profile

Summary A user of the system views existing configuration in a profile and possibly modifies it in order sets a new configuration of the system for a visually impaired user.

Description in a natural language A user which has the ability to see the system UI presses the profile view / modify button. The system presents the profile screen (also called "configuration screen") which will enable the user to view and change the existing settings of his profile. The user sees all the profile fields with their current values and gets the option to change them by will. The screen contains the following data:

- The name of the profile

- Parameters regarding the 3D-sound technology
- Parameters regarding the image analysis

After the user performs the desired changes, the system saves the changes in the Settings Data Storage and presents a message notifying the user about a successful profile modification. Then, the user is asked whether he wants to perform a training. In case he does, UC-2 is executed, otherwise the user is asked if he wants to start the visualization process immediately. In case he wants to start a visualization, UC-1 is executed, otherwise the system returns to the welcome screen.

Actors

- **A non-blind user** - who has the ability to see the configuration of the system via UI.

Trigger The user presses the "Modify Profile" button located on the welcome screen and presented by the System UI.

Preconditions

- The system is initialized properly - all its components are initialized and running.
- The UI of the system is presented on the screen.

Postconditions

- The new settings of the profile are saved in the Settings Data Storage.
- If changes were made, the new settings are available from the profile choosing screen.

Stakeholders and Interests

The configuration user wants an easy and fast process with interactive and error tolerated screen which will guide him through the profile modification process. In addition the system should prevent him from making undesired changes while viewing the profile and allow him to undo each performed operation.

The blind user for whom the profile is configured wants the process to be made with no mistakes, in order to fit the profile to his needs in the best way. He also wants a fast process which will allow him to use the application as fast as possible.

Main success scenario

1. A non-blind user presses on the profile view / modify button located in the welcome screen and presented by the System UI.
2. The system presents the profile screen.
3. The user changes the desired parameters and presses "save".

4. The system saves the modified profile in the Settings Data Storage and presents a message notifying the user about a successful profile modification.
5. The user is asked whether he wants to perform a training.
6. If a training is chosen:
   - UC-3 is executed.
7. Else, the user is asked if he wants to start a visualization process immediately.
8. If an immediate visualization is chosen:
   - UC-1 is executed.
9. Else, the system returns to the welcome screen.

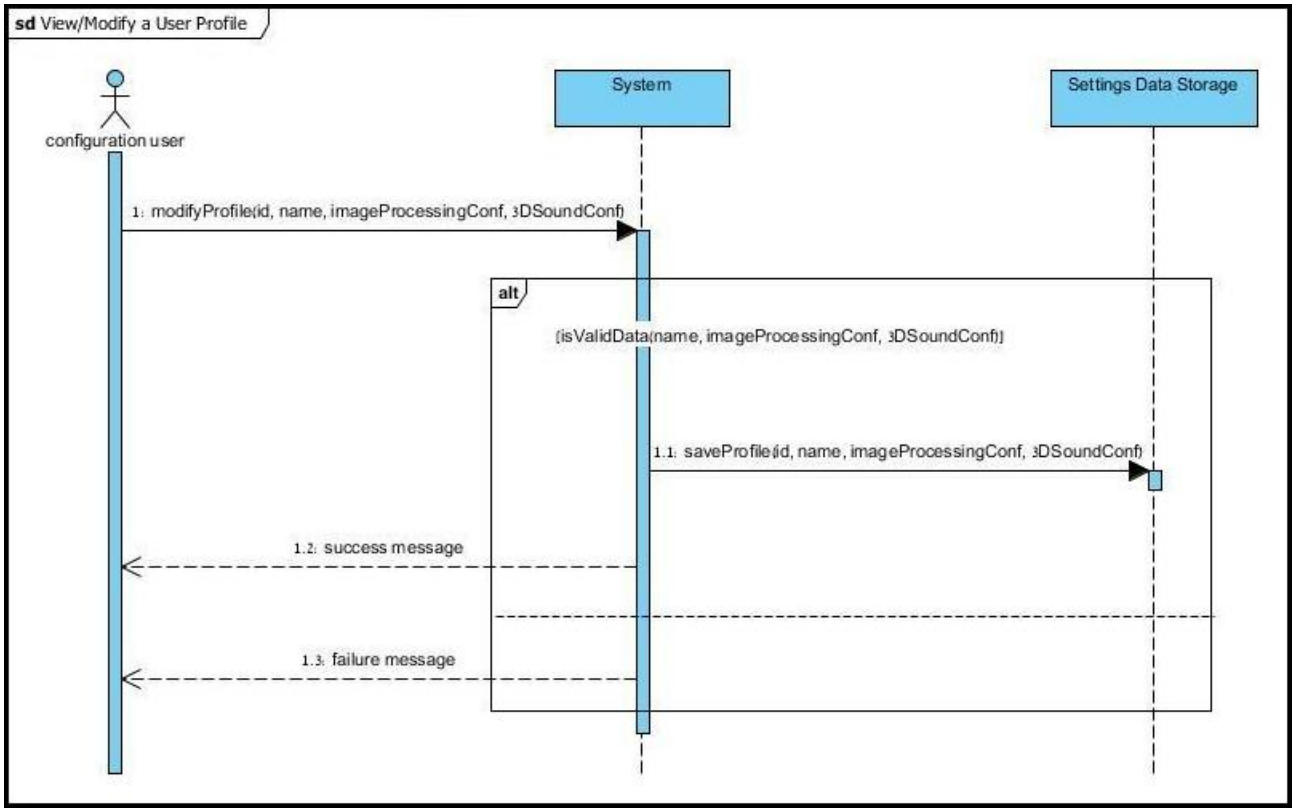
Alternative flows and Exceptions

- **2.a.**
  - Case: The user doesn't want to make any changes.
  - Action:
    1. The user presses "exit without saving".
    2. The system returns to the welcome screen.
- **3.a.**
  - Case: Not all required information has been provided - one of the fields is missing.
  - Action:
    1. The system asks the user to provide the missing information.
    2. Return to 3.
- **4.a.**
  - Case: Database write error
  - Action:
    1. A message that apologizes for the inconvenience and indicates that an error has occurred, is presented to the user.
    2. The user is asked whether he wants to try saving again, or to exit the profile modification screen..
    3. If the user chooses to try the saving again:
       * Return to 4.
    4. Else, End of use case.

Priority and Frequency of occurrence The use case described above is performed each time a user wants to view or modify an existing user profile. This operation is not a strong requirement for an existing user since usually only one profile is used by each user - the optimal one for him, according to the tests performed in the beginning of the application usage. Nevertheless, a modification of a profile is very common in the beginning of the process and is also very important in order to guarantee that the user gets the optimal settings for him. The use case is relatively rare in case the system is used by a blind user, but very common in case the system is used by a researcher. Thus, the use case has a middle priority.

Covering Requirements 2.1.2 (the fourth requirement)

Sequence Diagram



4.2.8 UC-8: Delete a User Profile

Summary A user of the system wants to delete some user profile, probably in order to clear some space for new user profiles to be created.

Description in a natural language A user which has the ability to see the system UI presses the profile deletion button, presented by the System UI. The system presents a list of all the existing user profiles and the user chooses a profile from it and presses the "Delete Profile" button. In case the profile required for deletion was provided by the system, the deletion operation is rejected and the system returns to the existing profiles screen. Otherwise, the system asks the user to confirm the profile deletion. In case the user confirms the deletion, the system deletes the

profile from the profiles list and updates the Settings Data Storage. Finally, the system returns to the existing profiles screen.

Actors

- **A non-blind user** - who has the ability to change the configuration of the system, add and delete profiles.

Trigger The user presses the "Delete Profile" button located on the welcome screen and resented by the system UI.

Preconditions

- The system is initialized properly - all its components are initialized and running.
- The UI of the system is presented on the screen.

Postconditions

- The chosen user profile was deleted from the existing profiles list.
- The Settings Data Storage was updated - the chosen profile was deleted from it.

Stakeholders and Interests

The user performing the deletion wants an easy deletion process which will not allow him to delete the default profiles, which were installed along with the system components and should not be deleted. This restriction is required in order to allow the basic operation of the system without creating new profiles.

Main success scenario

1. A non-blind user presses on the profile deletion button located in the welcome screen and presented by the System UI.
2. The system presents all the existing profiles in a list.
3. The user chooses a profile from the list and presses on the "Delete Profile" button.
4. The system asks the user to confirm the profile deletion.
5. The user confirms the deletion.
6. The system deletes the profile from the profiles list and updates the Settings Data Storage.
7. The user is notified about a successful deletion operation.
8. The system returns to the existing profiles screen.

Alternative flows and Exceptions

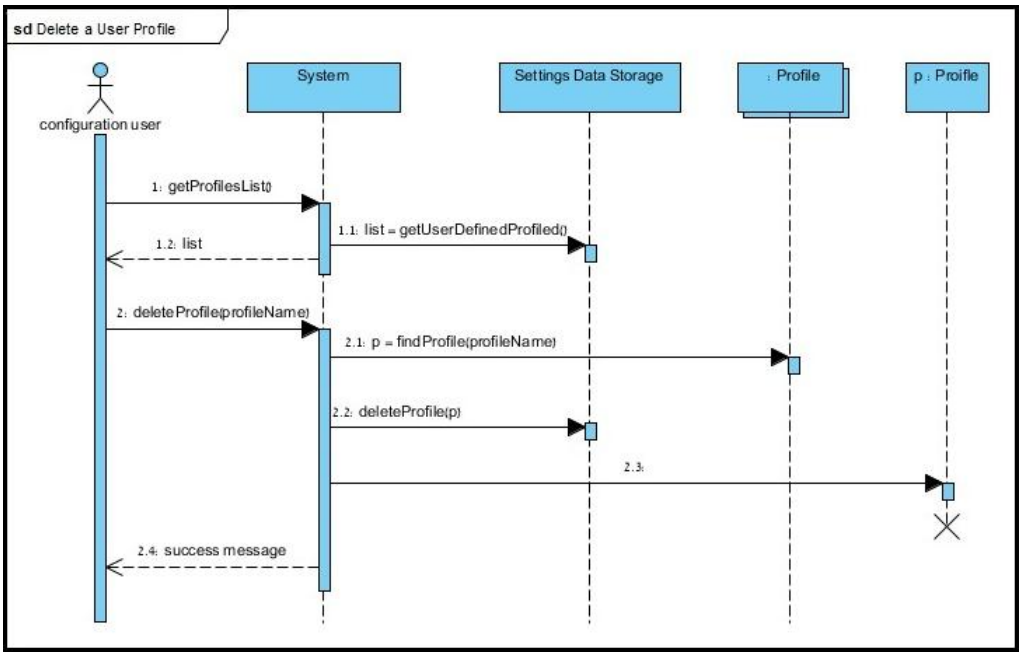
- **6.a.**
  - Case: Database write error
  - Action:
    1. A message that apologizes for the inconvenience and indicates that an error has occurred is presented to the user.
    2. The user is asked whether he wants to try the deletion again, or to stop the deletion operation.
    3. If the user chooses to try the deletion again:
       * Return to 6,
    4. Else, Goto 4.

- **6.b.**
  - Case: The chosen profile was not found in the Settings Data Storage
  - Action:
    1. The profile is deleted from the existing profiles list.
    2. Goto 7.

Priority and Frequency of occurrence This use-case is supposed to occur rarely, since the number of profile which can be stored simultaneously is very high. Likely, profile deletion will be performed only by researchers, after performing large experiments. Therefore this use case has a very low priority.

Covering Requirements 2.1.2 (the fourth requirement)

Sequence Diagram



4.2.9 UC-9: Install The Application

Summary A configuration user wants to install the system in order to give a blind user the ability to use it or for research purposes.

Description in a natural language The User clicks twice on the installation file icon and the system presents an installation wizard which guides the user through the installation process. The system performs a check of minimal hardware requirements and presents a message notifying that all the requirements are satisfied. Then, the system installs the AISN application, the OpenCV library, Rapture3D and HRTF datasets, all this alongside a progress bar which visualizes the installation progress. The system notifies the user about a successful installation and the installation ends.

Actors

- **A non blind user -** who performs the installation for a blind user.

Trigger The user double-clicks on the installation file on his machine.

Preconditions

- The machine, on which the system is installed, is turned on and has the installation file and all other files required for the installation.

Postconditions

- The AISN system is installed on the user's machine.
- The OpenCV library is installed on user's machine.
- Two 3D-sound technologies are installed on user's matching, including Rapture3D and basic HRTF datasets.

Stakeholders and Interests

The user who installs the system wants a simple installation process which guides him through the installation and informs him about the installation progress or when something goes wrong.

The blind user wants a fast installation of the system on his machine, with as little effort and room for mistakes as possible, in order to allow him to start using the system quickly.

Main success scenario

1. The User clicks twice on the installation file icon.
2. A "welcome" screen is presented notifying the user that the AISN system will be installed on his computer.
3. The user presses "next".

4. The system performs a check of minimal hardware requirements and presents a message notifying that all the requirements are satisfied.
5. The user presses "next".
6. The system installs the AISN application, the OpenCV library, Rapture3D and HRTF datasets through notifying the user about the installation progress.
7. The system presents a message notifying the user about a successful installation.
8. The user presses "finish".

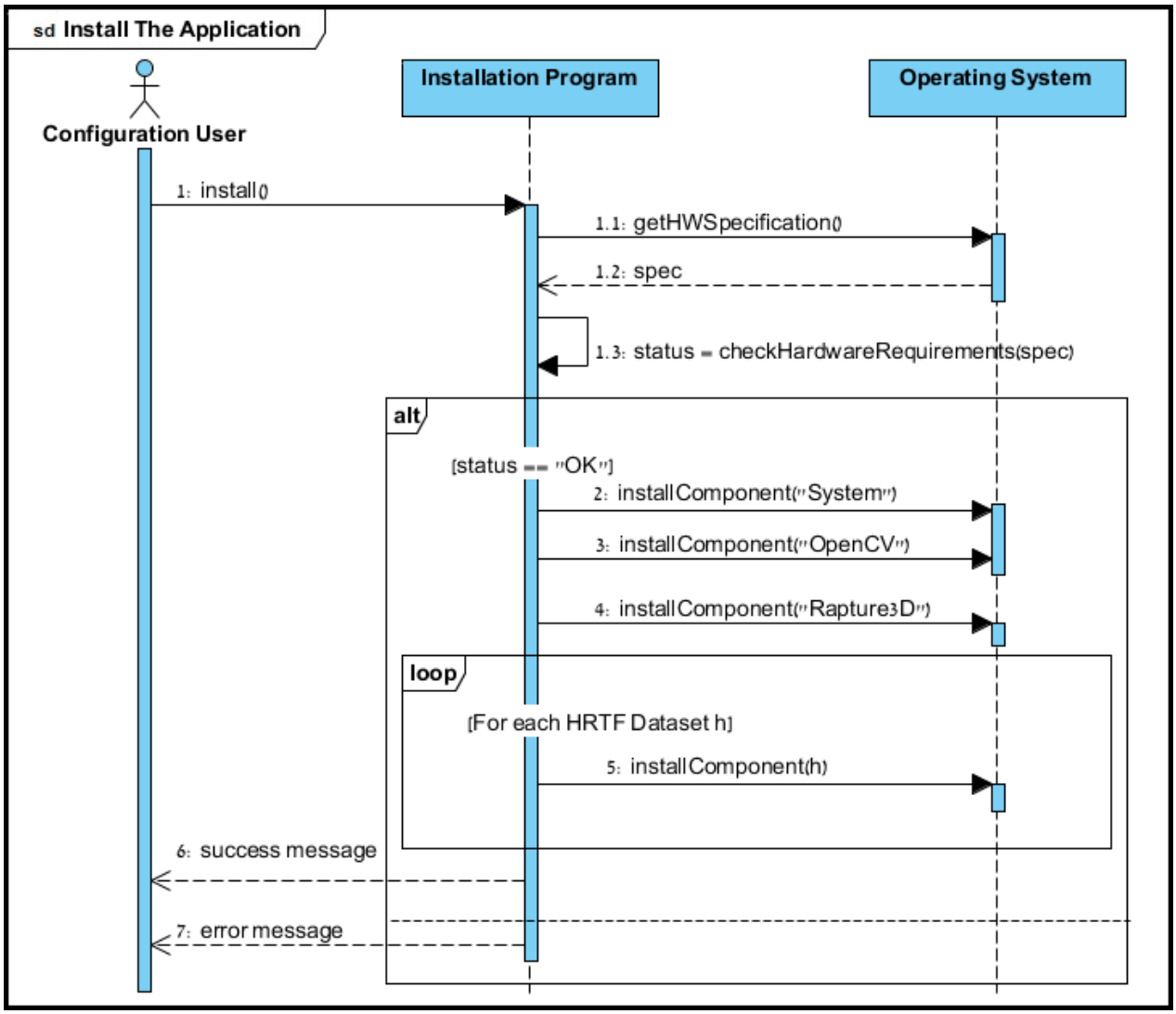
Alternative flows and Exceptions

- **4.a.**
  - Case: The computer, on which the system is installed, doesn't have the minimal required hardware.
  - Action:
    1. The user is notified about the missing requirements and gets a suggestion of what are the missing hardware.
    2. End of use case.
- **general (can happen at each step):**
  - Case: Electrical black-out or other type installation failure occurred.
  - Action:
    1. In case of an electrical black-out, end of use case (in the next installation attempt, all the components should be reinstalled).
    2. In case of installation failure:
       (a) The user is notified that a failure occurred.
       (b) End of use case.

Priority and Frequency of occurrence This use case is executed each time the system is installed on a new machine. Since, the installation process is crucial in order to allow the proper operation of the system, this use case has a critical priority and must be implemented in order to allow the use of the system.

Covering Requirements 2.1.1

Sequence Diagram



4.3 Special usage considerations

At any time, if the system crashes (e.g. because of an electrical blackout), it should be rebooted, the user should choose a profile again and the data that wasn't saved should be reentered.

5 Appendices

5.1 Glossary

AISN Auditory Imaging For Sightless Navigation, the official name of this SE project.

Binaural Spatialization A technique that aims at reproducing a real sound environment using only two channels (like a stereo recording). It is based on the assumption that our auditory system has only two receivers, namely the ears. If it is possible to deliver a signal equal (or nearly equal) to the one which a subject would receive in a real environment, this will lead to the same perception. Our auditory system performs various tasks to obtain a representation of the acoustic environment. Binaural spatialization can be achieved through various processes, such as using the impulse response of the head (HRIR).

Blind User A visually impaired user of the system for whom the main system's functionality is intended. As it was written above, the main purpose of the system is allowing this type of users to "see" the environment surrounding them with exploiting the sense of hearing, towards the goal of navigating in the physical environment.

CIPIC HRTF Database CIPIC is an abbreviation for Center for Image Processing and integrated computing university of California. The CIPIC HRTF Database is a public-domain database of high-spatial-resolution HRTF measurements for 45 different subjects. The database includes 2,500 measurements of head-related impulse responses for each subject. These measurements were recorded at 25 different azimuths and 50 different elevations. The database includes anthropometric measurements for use in HRTF scaling studies, technical documentation, and a utility program for displaying and inspecting the data.

Cognitive Science An interdisciplinary scientific study of how information concerning faculties such as perception, language, reasoning, and emotion, is represented and transformed in a (human or other animal) nervous system or machine (e.g., computer). It consists of multiple research disciplines, including psychology, artificial intelligence, philosophy, **neuroscience**, learning sciences, linguistics, anthropology, sociology, and education. It spans many levels of analysis, from low-level learning and decision mechanisms to high-level logic and planning; from neural circuitry to modular brain organization.

Computer Vision The science and technology of machines that see. See in this case means that the machine is able to extract information from an image that is necessary to solve some task. As a scientific discipline, computer vision is concerned with the theory behind artificial systems that extract information from images. The image data can take many forms, such as video sequences, views from multiple cameras, or multi-dimensional data from a medical scanner. Typical tasks of computer vision are feature extraction and object recognition.

Configuration User A non-blind user which uses the system mainly while configuring it and/or creating user profiles, for a blind user that wants to use the system. Our intention is to decrease the requirement to involve the configuration users during the process of the system tuning.

Field Of Vision (FoV) The angular extent of the observable world that is seen at any given moment (also abbreviated field of view). Different animals have different fields of view,

depending on the placement of the eyes. Humans have an almost 180-degree forward-facing horizontal field of view, while some birds have a complete or nearly-complete 360-degree field of view. In addition, the vertical range of the field of view in humans is typically around 100 degrees. In machine vision the lens focal length sets up the fixed relationship between the field of view and the working distance. Field of view is the area of the inspection captured on the camera's imager. The size of the field of view and the size of the camera's imager directly affect the image resolution (one determining factor in accuracy). Working distance is the distance between the back of the lens and the target object.

HRTF The HRTF can be described as the modifications to a sound from a direction in free air to the sound as it arrives at the eardrum. These modifications include the shape of the listener's outer ear, the shape of the listener's head and body, the acoustical characteristics of the space in which the sound is played, and so on. All these characteristics will influence how (or whether) a listener can accurately tell what direction a sound is coming from. The modifications may be captured via an impulse response which relates the source location and the ear location. This impulse response is termed the head-related impulse response or HRIR. The HRTF is the Fourier transform of the HRIR.

HRTF Dataset Because of the unique physical characteristics of human bodies, heads and pinna, everyone's characteristics are unique. Although there are some similarities between persons, typically individualized HRTFs lead to a better, more accurate and realistic auditory image. An HRTF dataset is a measurement of an individualized HRTF which improves the sound localization and reduces front/back and up/down confusions of the measured person and people with same anthropometric characteristics.

Image Processing Any form of signal processing for which the input is an image, such as a photograph or video frame; the output of image processing may be either an image or, a set of characteristics or parameters related to the image. Most image-processing techniques involve treating the image as a two-dimensional signal and applying standard signal-processing techniques to it.

Listen HRTF Database A database created by measurement sessions which took place in the frame of the Listen project, which is a shared-cost project in the Information Society Technologies (IST) Program of the European Commission's Fifth Framework Program. As a part of the Listen project measurements on relationships between users' head and ear dimensions had to be made. Two partners of the Listen project, AKG and Ircam, which are involved in the audio part of the project, have both performed the measurement sessions whose results are arranged as a HRTF database. These measurements were recorded at 10 different azimuths and 24 different elevations and were performed at Paris, France.

Object Tracking Detection of interesting moving objects, tracking of such objects from frame to frame, and analysis of object tracks to recognize their behavior. In its simplest form, tracking can be defined as the problem of estimating the trajectory of an object in the image plane as it moves around a scene. In other words, a tracker assigns consistent labels to the tracked objects in different frames of a video. Additionally, depending on the tracking domain, a tracker can also provide object-centric information, such as orientation, area, or shape of an object.

OpenAL OpenAL is a cross-platform 3D audio API appropriate for use with gaming applications and many other types of audio applications. The API models a collection of audio sources moving in a 3D space that are heard by a single listener somewhere in that space.

Optical Radar A system developed at BGU that helps blind people maneuver around obstacles. The system incorporates a computer, video cameras and a scanning light source to warn the blind of obstacles with audible alerts. The system detects obstacles by scanning the depth of its surroundings, taken from two different angles, similar to that of the human eye.

Rapture3D A commercial OpenAL implementation that is distributed by BlueRippleSound company and was intended originally for game development. The game tells Rapture3D what sounds should be playing where and when, and Rapture3D gets on with the maths to perform the required sound placement.

Researcher One of the types of users for whom the system is intended. Those researchers are cognitive science and especially neuroscience researchers who want to test the ability of the hearing sense to replace the sight sense.

Sampling Frequency Defines the number of samples per second (or per other unit) taken from a continuous signal to make a discrete signal. For time-domain signals, the unit for sampling rate is hertz (inverse seconds, 1/s). The inverse of the sampling frequency is the sampling period or sampling interval.

Sensory Substitution A replacement of one sensory input (vision, hearing, touch, taste or smell) by another, while preserving some of the key functions of the original sense. In particular, research in this area aims at providing some equivalent of vision via hearing or touch, or some equivalent of hearing via vision or touch. Blindness and deafness are generally considered to be among the sensory disabilities that have the greatest impact on everyday life, and the quest for good sensory substitution devices or prostheses for blindness and/or deafness therefore presents a great challenge.

Sonar-Like Images Scanning Sonar (an acronym for SOUND NAVIGATION AND RANGING) is a technique that uses sound propagation (usually underwater, as in submarine navigation) to navigate, communicate with or detect other vessels. Several products were created while trying to apply this technology for auditory imaging, like "Sonic Glasses", or "Sonic Guide" that represents object distance by pitch, but also represents surface texture through timbre. Later, in 1991 Peter Meijer from Utrecht university (Germany) developed a device, nicknamed "The vOICe", that maps grey-scale images from a video camera into corresponding visual sounds. But, it remains difficult to settle on a good measure for equivalent image resolution as offered by sonar approaches, because sonar is not based on mapping an image matrix to another sensory modality, but on the implicit preservation of some degree of detailed environmental information. Moreover, mapping the images left-to-right according to the sonar approach, does not make the user to feel his environment in a natural way, but in a step-by-step manner.

3D-Audio Environment Three-dimensional virtual environment which surrounds the user's head and makes possible the virtual placement of any number of sound sources around it, thus simulating sounds coming from locations where the sounds are placed.

3D-Sound A stereo image produced by stereo headphones or by two loudspeakers in order to create the illusion of sound sources placed anywhere in the 3 dimensional space, including behind, above or below the listener.

3D-Sound Technology A technology that exploits 3D positional audio using head-related transfer functions and reverberation, the changes of sound on its way from the source (including reflections from walls and floors) to the listener's ear can be simulated. The technologies include localization of sound sources behind, above and below the listener and have been incorporated in music and video-game style arts.

User Profile Contains user-specific parameters configuration about the image processing and the 3d-sound technology used. Each user that uses the AISN system can create a personal user-profile which will be saved in a special XML-based file and contains his personal tuning of the system's parameters. The personal user-profile allows the user to get the most benefit from the system.

5.2 OpenAL

5.2.1 General Description

OpenAL is an abbreviation for ***O**pen **A**udio **L**ibrary.* which was created and supported by Creative Lab and Loki Entertainment companies. This is a cross-platform, free license and open-source library with **3D audio API**, which suits games and many other types of audio applications that use 3D-Sound. OpenAL supports Mac OS 8/9, Linux, BSD, Solaris and Microsoft Windows XP/Vista/7 operating systems. Today it is hosted (and largely developed) by Creative Technology with on-going support from Apple, Blue Ripple Sound, and free software/open source enthusiasts.

Actually OpenAL is an audio API only, not a working library and there are 3 main implementations:

1. OpenAL SI (OpenAL Simple Implementation): is the original implementation, from Loki, and is not currently maintained.

2. OpenAL Soft: is an LGPL-licensed, cross-platform, software implementation. This library replaces the deprecated OpenAL SI.

3. Rapture3D OpenAL Driver: Windows only, software implementation, based on DirectSound3D.

The library is simple and easy to use and therefore very convenience for programmers. The programming style (like functions names) is similar to OpenGL library which is popular in computer graphics programming. The library models a collection of audio sources moving in a 3D space and heard by a single listener somewhere in that space. The three fundamental objects of OpenAL are *Listener*, (sound) *Source* and *Buffer* (There are other objects like *audioDevice* and *audioContext*). The Listener object represents the position of the human listener, which hears the sounds from different points in the space around him. The Buffer file is an abstraction for an audio file – it is filled with audio data. Lots of Buffers can be used in the same application and each one can be attached to one or more Source objects. A Source object represents a point in a 3D space that produces sounds. A Source object contains a list of buffers and can be positioned and played. Creating a single listener and a number of sources and buffers and then updating the positions, the orientations of the sources and the listener dynamically can present a convincing 3D audio world. OpenAL allows defining different parameters of listener and sound sources in the 3D medium (like position, velocity and mutual influence of the sound source according to the listener position), which affects the generated sound. The OpenAL library adapts sounds according to these parameters and mixes sounds from different sources, in order to generate the appropriate sound for each of the output sound ports. The Listener object contains the velocity, position and direction of the listener, and the general gain applied to all sound. Buffers contain audio data in PCM format, either 8 or 16-bit, in either monaural or stereo format. The rendering engine performs all necessary calculations like distance attenuation, Doppler effect, etc.

OpenAL Utility Toolkit (ALUT) ALUT is a set of higher level semi-portable 'convenience' functions that were mixed up in the OpenAL library distribution. The intention was that ALUT would be a separated library that is portable between systems. It is hoped that it will be well suited to producing succinct demo programs and to help new developers to get started with OpenAL. This was done by removing the annoying details of getting an audio application started - allowing developers to learn OpenAL without distractions such as loading sound samples from disk.

More information can be found in the following links:

- OpenAL main homepage: `http://connect.creativelabs.com/openal/default.aspx`.
- APIs:
  - The OpenAL Utility Toolkit (ALUT) Reference Manual in PDF Format: `http://connect.creativelabs.com/openal/Documentation/The%20OpenAL%20Utility%20Toolkit.pdf`.
  - Quick reference guide to using OpenAL 1.0 or 1.1: `http://connect.creativelabs.com/openal/Documentation/OpenAL_Programmers_Guide.pdf`.
  - The official OpenAL 1.1 Specification and Reference in PDF format: `http://connect.creativelabs.com/openal/Documentation/OpenAL%201.1%20Specification.pdf`.

5.2.2 Rapture3D Description

Rapture3D is a non-free, OpenAL based driver that was developed by Blue Ripple Sound company. Rapture3D can place sounds to the left, to the right - in front of you, behind you, above and below: it receives as an input what sounds should be played where and when, and gets on with the maths to make that happen.
Two versions are supported:

- User version
- Advanced version

In the "user" version of the software, the decoder synchronizes with Windows settings. In the "advanced" version other settings are supported, and can be controlled by the user. Other features in the advanced version are:

- Access to audio hardware via ASIO (ASIO bypasses the normal audio path from the user application through layers of intermediary Windows operating system software, so that the application connects directly to the sound card hardware) which can allow simple synchronized access to large numbers of channels of extremely high quality and low latency audio hardware.
- Decoder generation for custom layouts. The user enters where the various speakers are located, into the set up program and the generator then generates a new decoder optimized for these arrangements.

Rapture3D has the following features:

- 32bit floating point audio path.
- High quality effects and filters.
- Support for multi-channel sound sources.
- No limit on the number of sources or effects except CPU power. Able to render hundreds of sound sources and multiple effects on relatively old hardware.
- Access to audio hardware via DirectSound.
- **3D sound over headphones. This uses psychoacoustic techniques based on head shape. Five different "HRTFs" are provided.**

More information can be found in the Blue Ripple Sound website, Rapture3D page located here.

5.3 3D-Sound technologies testing report

Here is a report of 3D sound technologies experiment we will perform in order to choose an appropriate 3D sound technology to be used with the system implementation. In addition, this report monitions accuracy measures to be used while testing the technologies, striving to deduce the experiments set for testing the whole system.

Date (of experiment conduction): _____

Purpose: The purpose of this experiment is to test the different 3D-Sound technologies we investigated during the research period and choose the technology that sounds better, in order to use it in future development of the AISN and AudioWebInterface projects. The technologies quality will be determined by measuring users' ability to detect the properties of 3D-Sound that are stated below.

Technologies: In this experiment we test two different kinds of technologies:

1. 3D-Sound existing algorithms (DirectSound3D and Rapture3D).
2. Our own implementations of binaural spatialization using different HRTF datasets.

Here we provide a complete list of the technologies that will be tested, with a brief definition for each one of them:

1. Existing implementations:
   (a) DirectSound3D – An interface between applications and sound card drivers on Microsoft Windows OS, enabling applications to produce sound. In addition provides 3D-audio spatialization by using software 3D audio algorithm.
   (b) Rapture3D – A commercial implementation of the OpenAL interface. Rapture3D uses psychoacoustic techniques based on head shape by providing five different HRTF datasets hardly coded in the implementation.
2. Our implementation:
   We implemented our own simple algorithm which produces 3D sound. The algorithm uses different HRTF datasets and thus the spatialization quality strongly depends on the dataset being used. Currently we support two kinds of HRTF datasets:
   (a) CIPIC HRTF dataset – A public-domain database of high-spatial-resolution HRTF measurements for 45 different subjects.
   (b) LISTEN HRTF dataset – 51 different subjects

Equipment and software needed: The equipment we need for conducting the experiment is:

- For generating 3d-sound using the described technologies:
  - A computer on which Rapture3D and HRTF datasets (along with the algorithm implementation) are installed (currently only one laptop has all these features).

  - An external sound card which can be connected to USB 2.0 or IEEE 1394 port.
  - A high quality headphones.

- For measuring the results:
  - A white board.
  - A projector – in order to project a location pointer on the board.
  - Two laser pointers:
    * A red pointer used for pointing the sound location.
    * A green (or any other color) pointer used for head tracking.

Experiment setup: After entering the lab, an equipment setup is needed:

1. The computer should be started and the appropriate applications (include MATLAB) should be opened and configured: **~10 min**
2. The computer should be connected to a working projector which may need to be tuned: **~5 min**
3. The sound card should be connected to the computer and appropriately tuned: **~5 min**
4. The web-camera should be placed in a constant place in front of the white board, and tested to capture the laser pointer: **~5 min**.
5. An appropriate place for the subject under test should be chosen: **~5 min**.
6. A little 3d-sound sample should be generated and played in order to assure that equipment setup was completed successfully.

According to the approximation made above, the total time we need for the setup is about 30 minutes.

HRTF choosing: According to the technologies description, two of the described technologies are based on HRTF datasets which are created according to measured parameters of different subjects' heads. Due to the individuality of each subject a strong requirement for the experiment relevance in pre-choosing an appropriate HRTF dataset among the existing datasets; otherwise the experiment results may point on technology failure which may be a false decision. Currently we don't have any wise procedure for choosing the best suitable dataset. The only reliable procedure is to test all the available datasets (which may be quite irritating). The criterion for a suitability of an HRTF should be of course the quality of simulation of the elevation (especially bottom degrees). However, according to research made at New York and Michigan universities , front / back confusion rates increase when non-individualized HRTFs are used to simulate spatial audio. Therefore, the procedure of choosing an appropriate HRTF will be as follows:

1. A set of points corresponding to XY (in front of the listener head) and XZ circle (around the listener head) will be generated.

2. The spatial audio of the created circles will be simulated using each HRTF filter. The subject will listen to each result and tell us which HRTF dataset provided him the best virtual reality of an XY and XZ circle. Assuming the time required to hear the samples using a single HRTF filter is 30 sec. and assuming there exists max of 50 HRTF measurements, the time for an HRTF choosing is ~25 min. for a single dataset . In order to decrease the time required for the experiment, we suggest use few computers, which have MATLAB installed along with the HRTF datasets. In this way, many subjects will be allowed to choose HRTFs simultaneously.

Experiment procedure: The experiment will be conducted on several subjects. When testing technology X, we'll produce 3D spatialized sounds using X, controlled by several parameters described below and ask the subject to describe us what he had heard.

Observations and Data: We'll use four kinds of tests, each one measures different feature of the technology. The tests are described in the table below, for each test the parameters being controlled and the measures being done are listed (next page).

Test name	What is heard	Parameters (the possible options are specified in the brackets) that the subject should tell	Measures	Feature measured
1 Identifying a shape in XY plane (This includes detection of spinning direction)	A randomly generated shape from the list below, spinning clockwise or counterclockwise The possible shapes are: Circle, Square and Triangle	**Shape type** T (Circle, Square, Triangle) **Shape size** S (bigger, smaller) **Direction** D (clockwise, counterclockwise)	Type will get 37.5%, Size will get 25% and Direction will get 37.5% Percent of correct guesses out of X tries is calculated: $val = \frac{T'}{X} \cdot 37.5 + \frac{S'}{X} \cdot 25 + \frac{D'}{X} \cdot 37.5$ Where T', S' and D' denote number of correct guesses for type, size and direction accordingly	Quality and accuracy of 3D sound movement simulation according to a given route
2 Spinning direction in XZ plane	A sound spinning randomly around the listener head (XZ plane) in clockwise or counterclockwise directions	The spinning direction D	Percent of correct guesses D' of the direction, out of X tries is calculated: $val = \frac{D'}{X} \cdot 100$	Quality of 3D sound movement simulation in XZ plane
3 Pinpointing a sound source	A sound located somewhere in the XY plane	The sound position	The average distance between the real position to the perceived position of the source (in cm), calculated after X measurements: $val = \frac{Sum\ of\ all\ distances}{X}$	Accuracy of 3D sound location in XY plane

Figure 5: A table summarizing the experiments

5.4 References

Here we list the sources of information which were used while writing this document:

1. Bradski, G.; Kaehler, A. (2008), "Learning OpenCV: Computer Vision with the OpenCV Library".

2. OpenCV official website, `http://opencv.willowgarage.com/wiki/Welcome`.

3. Durand R. Begault , "3-D sound for virtual reality and multimedia", : National Aeronautics and Space Administration, Ames Research Center ; Hanover, MD, 2000.
   Online version is available here.

4. Audio Education Series, "3D Sound What Is It?", vol. 4, Audio Products Division of National Semiconductor, Analog Products, Santa Clara, California.

5. Creative Technology Limited, "OpenAL Programmer's Guide, OpenAL Versions 1.0 and 1.1", 2007,
   Online version is available here.

6. Gardner,W. G. and Martin, K. D., "HRTF Measurements of a KEMAR," J. Acoust. Soc. Amer., Vol. 97, 3907-3908, 1995.
   See also `http://www.sound.media.mit.edu/KEMAR.html`.

7. A. Roginska, G. Wakefield, T. Santoro, "User Selected HRTFs: Reduced Complexity and Improved Perception", Naval Submarine Medical Research Lab, Groton, CT, USA; New York University, New York, NY, USA; University of Michigan, Ann Arbor, MI, USA.

8. V. R. Algazi, R. O. Duda, D. M. Thompson and C. Avendano, "The CIPIC HRTF Database", Proc. 2001 IEEE Workshop on Applications of Signal Processing to Audio and Electroacoustics, pp. 99-102, Mohonk Mountain House, New Paltz, NY, Oct. 21-24, 2001.
   See also `http://interface.idav.ucdavis.edu/CIL_html/CIL_HRTF_database.htm`.

9. Olivier Warusfel, Room Acoustics Team, IRCAM, PARIS, France, "LISTEN HRTF DATABASE" homepage,
   `http://recherche.ircam.fr/equipes/salles/listen/`.

10. Peter B.L. Meijer, 1996 - 2010, "Artificial Vision for the Totally Blind", "The vOICe" homepage,
    `http://www.artificialvision.com`.

11. Wikipedia (used while writing the glossary).