ReSilence

Retune the Soundscape of future cities through art and science collaboration

HORIZON- 101070278

D5.2

Prototypes

Dissemination level:	Public
Contractual date of delivery:	Month 19, 31 March 2024
Actual date of delivery:	Month 20, 30 April 2024
Workpackage:	WP5 Art-driven experimentation toolkit development
Task:	T5.1 Technical requirements and tools architecture T5.2 Integration of services at component level T5.3 Development of multisensory experiences in XR environments T5.4 Extending human senses to the digital world through tactile sound T5.5 Low-Intrusiveness Sound Design techniques for counteracting noise via interactive sonifications of the city waves of traffic
Type:	Demonstrator, pilot, prototype
Approval Status:	Final Draft
Version:	1.0
Number of pages:	46
Filename:	D5.2_resilence_prototypes.docx

Abstract

This document outlines technical requirements for ReSilence system development and

presents current status of artist-crafted prototypes. It details requirements analysis procedures, derived technical specifications, and prototype states.

funded by the European Union

History

Version	Date	Reason	Revised by
0.1	2 March 2024	ToC creation	CERTH
0.2	26 March 2024	Creation of 1st integrated version based on partner's and artist's contributions	CERTH
0.3	10 April 2024	Creation of 2nd integrated version based on partner's and artist's contributions	CERTH
0.4	22 April 2024	Internal review	AUTH
0.5	25 April 2024	Creation of 3rd integrated version based on internal review comments	CERTH
0.6	29 April 2024	Preparation and quality check of the final draft	CERTH
1.0	30 April 2024	Submission of the final draft	CERTH

Author list

Organization	Name	Contact Information
CERTH	Sotiris Diplaris	diplaris@iti.gr
CERTH	Nefeli Georgakopoulou	nefeli.valeria@iti.gr
CERTH	Paraskevi Kritopoulou	pakrito@iti.gr
CERTH	Eleftheria Layokapa	elagio@iti.gr
UNIGE	Antonio Camurri	antonio.camurri@unige.it
MPIEA	Melanie Wald-Fuhrmann	melanie.wald-fuhrmann@ae.mpg.de
MU	Beatrice de Gelder	b.degelder@maastrichtuniversity.nl

Executive Summary

This document presents a comprehensive overview of the technical requirements essential for the development of the ReSilence project's system, alongside a detailed status update on each prototype crafted by individual artists.

The document begins by outlining the procedures involved in technical requirements analysis, emphasizing the importance of this practice within the context of ReSilence. It elucidates various methods employed in gathering requirements, providing insight into the rationale behind their selection and application.

Subsequently, the document delves into the technical requirements derived from user and artist requirements, highlighting their interrelationships and dependencies. This section offers a comprehensive analysis of the interrelated technical specifications that form the foundation for the project's development roadmap.

Furthermore, the document provides a comprehensive overview of the current state of each prototype, furnishing detailed technical insights into their respective compositions and functionalities. Each prototype is meticulously described, including the input and output data utilised, and the specific requirements it aims to fulfil. This detailed breakdown is complemented by insightful screenshots capturing the essence of each experiment.

In conclusion, this document serves as a valuable resource for collaborators, offering clarity on the technical specifications of the ReSilence project's prototypes and their current status. It underscores the project's commitment to leveraging technology to create innovative sonic experiences while maintaining a user-centric approach throughout the development process.

Abbreviations and Acronyms

2D	Two Dimensional
3D	Three Dimensional
CPU	Central Processing Unit
FFT	Fast Fourier Transform
GDPR	General Data Protection Regulation
HLURs	High-Level User Requirements
Hz	Hertz
I/O	Input/Output
INCOSE	International Council on Systems Engineering
IP	Internet Protocol
KPI	Key Performance Indicator
MPIEA	Max Planck Institute for Empirical Aesthetics
MTCNN	Multi-task Cascaded Convolutional Neural Network
OS	Operating System
RAM	Random Access Memory
RGB	Red Green Blue
SBC	Single-Board Computer
SD card	Secure Digital card
TCP	Transmission Control Protocol
TR	Technical Requirements
UN	United Nations
UR	User Requirements
USS	Universal Source Separation algorithm
XR	Extended Reality
PUC	Pilot Use Case
FX	Special Effect
TAT	Tieranatomisches Theater

Table of Figures

Table of Tables

Table of contents

1 INTRODUCTION

This deliverable serves to document the comprehensive technical requirements essential for the development of the system, alongside providing an overview of the current status of each ReSilence prototype crafted by individual artists.

The document commences by elucidating the procedures involved in technical requirements analysis. This initial section not only introduces the practice of collecting and analysing technical requirements but also delineates its significance within the context of ReSilence. Furthermore, the section delves into a discourse on the various methods employed in gathering requirements, elucidating the rationale behind their selection and application within the project's framework.

In the next section, we present the technical requirements derived from the user/ artist requirements and its relations with each other. Finally, the current state of each prototype is presented with detailed technical information.

2 INTRODUCTION TO TECHNICAL REQUIREMENTS

2.1 Requirements Engineering

Requirements Analysis (Pohl et al., 2013), also known as Requirements Engineering, "*is the process of eliciting stakeholder needs and desires and developing them into an agreed set of detailed requirements that can serve as a basis for all other subsequent development activities*" These requirements serve as the cornerstone for all subsequent development endeavors. Each requirement is a carefully crafted statement aimed at describing operational, functional, or design aspects, ensuring they are clear, measurable, and indispensable for the acceptability of the product or process.

Initially, the system's comprehension may be nebulous, prompting the exploration of preliminary ideas, concepts, and end-user expectations. This evolves through the collaboration of technical experts, culminating in the establishment of a consensus-driven 'desired output'—a comprehensive set of detailed statements that serve as the basis for launching development activities.

Delving deeper into the process, the subsequent section describes the guiding principles underpinning system requirements.

2.1.1 Principles Governing System Requirements

Relation to User Requirements and Logical Architecture

The foundation of successful software development hinges on a clear distinction between user requirements and system requirements. This distinction is a key principle of requirements engineering.

User requirements, often captured as stakeholder/user statements, are expressed in natural language. These statements prioritise user needs and are readily understood by those without technical expertise. However, to facilitate proper system design by engineers, these user needs are translated into a more technical language using standard formats or templates. This refined version, known as system requirements, acts as an extended translation of user requirements. System requirements delve deeper, considering performance, quality, and other technical aspects while identifying the system's essential functionalities and modules.

Bridging the gap between user needs and technical implementation requires close collaboration between stakeholders/users and engineers. This collaborative effort ensures system requirements accurately portray the system's functions in an unambiguous, consistent, and verifiable way. This clarity is crucial for successful system development.

The logical architecture activity, which defines the system's boundaries and functionalities, also plays a vital role. This activity often leads to further refinement of the system requirements. By decomposing the system into logical components, the logical architecture lays the groundwork for implementation without introducing unnecessary constraints.

Regardless of whether the system is a completely new solution or an evolution of an existing product, ensuring user requirements, system requirements, and the logical architecture are all complete and consistent with each other is paramount. This consistency forms the

cornerstone for a well-defined system that meets user needs and translates effectively into a successful implementation.

Classification and Characteristics of System Requirements

System requirements can be categorised in various ways depending on the specific needs of the project. Here, we explore a classification based on the International Organization for Standardization (ISO) standard on 2011[1]. This classification groups requirements into distinct categories, each addressing a specific aspect of the system. The following table summarises these categories.

Types of System Requirements	Description
Functional Requirements	Define the system's functionalities and what it should do.
Performance Requirements	Specify how well the system must perform in terms of speed, accuracy, and other metrics.
Usability Requirements	Address how easy and user-friendly the system should be.
Interface Requirements	Define how the system interacts with other systems and users.
Operational Requirements	Specify how the system will be operated, maintained, and supported.
Modes and/or States Requirements	Detail the different operational modes or states the system can have.
Adaptability Requirements	Address the system's ability to adapt to changing environments or needs.
Physical Constraints	Define limitations imposed by physical factors like size or weight.
Design Constraints	Specify restrictions placed on the system's design due to technical or commercial reasons.
Environmental Conditions	Detail the environmental conditions (e.g., temperature, humidity) the system must operate in.
Logistical Requirements	Address the logistics of deploying, transporting, and installing the system.
Policies and Regulations	Specify compliance with relevant policies, regulations, or standards.
Cost and Schedule Constraints	Define limitations related to budget and development timeline.

Table 1: System requirements classification

Beyond classification, ensuring well-defined system requirements necessitates specific characteristics. The International Council on Systems Engineering (INCOSE) recommends a set of characteristics in their 2011 Systems Engineering Handbook[2]. These characteristics,

[1] ISO/IEC/IEEE. 2011. Systems and Software Engineering - Requirements Engineering. Geneva, Switzerland: International Organization for Standardization (ISO)/International Electrotechnical Commission/ Institute of Electrical and Electronics Engineers (IEEE), (IEC), ISO/IEC/IEEE 29148.

[2] INCOSE. 2011. Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities, version

outlined in the following table, guide the creation of clear, concise, and verifiable requirements.

Characteristics	Description
Necessary	The requirement defines an essential feature. If it is not included in the list of requirements, it causes the deficiency of conformance with regards to the need of a desired characteristic of the system, defined by users.
Unambiguous	The requirement is written in a way that produces a single and the same interpretation to all readers.
Complete	The requirement has to be sufficient without the need for additional information in order to understand it.
Singular	The requirement should describe a single need.
Feasible	The requirement has to take into account potential constraints with acceptable risk
Verifiable	The requirement has to be written in a such way that its realisation can be measurable or testable in order to verify it.
Correct	The requirement must be an accurate representation of the entity need from which it was transformed.
Conforming	Each requirement has to conform to an approved standard template and style for writing requirements, when applicable.

Table 2: Characteristics of requirements

2.2 Requirements analysis in ReSilence

ReSilence consists of various distinct and autonomous prototypes, each implemented by different artists, with its own unique audience and approach, all sharing a common goal.
Acknowledging the convergence of TRs towards shared goals, such as empowering the design process of urban environments with sound and enhancing sonic urban experiences, underscores the interconnection and mutual reinforcement among the various components. This shared thematic thread mirrors the collaborative ethos of ReSilence, where diverse stakeholders collaborate to achieve overarching objectives.
Incorporating this collaborative spirit into the TRs involves identifying common functionalities or specifications that contribute to the realisation of shared objectives. Just as artists and scientists collaborate to develop the tools, TRs can embody shared functionalities or interfaces that facilitate seamless integration and interoperability among system components.
The purpose of the requirements analysis in ReSilence aims to:

- Establish a clear and agreed-upon set of technical requirements for all stakeholders.
- Define the functionalities and user tasks.
- Specify the functionalities of each system component.

3.2.1. San Diego, CA, USA: International Council on Systems Engineering (INCOSE), INCOSE-TP-2003-002-03.2.1.

- Identify constraints and architectural implications.

The primary goal is to establish a clear understanding of the core functionalities and user interactions relevant to the entire project, even though individual prototypes may differ. This will involve:

- Identifying common user scenarios: Find the core interactions that the user will experience across prototypes

A close collaboration between the artists and the technical partners was necessary. A set of activities are undertaken during the requirement analysis process in ReSilence project. These activities include:

- Defining the technical requirements and their rationale.
- Classifying the technical requirements
- Incorporating the derived technical requirements
- Identifying potential risks

Artist-Specific Requirements Capture

Given the independent nature of the artistic prototypes, it is essential to capture the technical requirements specific to each artist's vision. This is achieved through:

- Individual Consultations: Detailed discussions with each artist to understand their prototype's technical needs (e.g., hardware integration, data processing).
- Prototype Analysis: Technical evaluation of each prototype to identify specific technical requirements not captured through consultations.

Consolidation and Consistency

While respecting individual artistic expression, ensuring a cohesive technical experience is vital. This will involve:

- Identifying common technical needs: Streamlining requirements where possible to reduce overall technical complexity and facilitate potential future integration between prototypes.
- Defining interoperability standards: If any data exchange or interaction between prototypes is envisioned, establishing communication protocols or data formats is essential.

Iterative Process and Open Communication

The technical requirements will be a living document, evolving as the project progresses. An open communication channel will be maintained:

- Regular Artist Check-Ins: Frequent consultations with artists to ensure their technical requirements remain aligned with their evolving artistic vision.
- Technical Review Sessions: Periodic meetings with the technical team to discuss challenges and opportunities arising from the evolving TR landscape.

This approach prioritises both artistic freedom and technical cohesion. By fostering open

communication, iterative refinement, and a balance between centralised guidelines and independent development, it is possible to achieve a successful multi-artist project with well-defined technical requirements.

3 TECHNICAL REQUIREMENTS

3.1 Technical Requirements Overview

This section details all the Specific User Requirements (UR) identified for the project. These URs were derived from the High-Level User Requirements (HLURs) established through collaboration with each artist. Each UR has been categorised as either a Functional Requirement or a Non-Functional Requirement, following established Software Engineering principles[3]. Finally, it relates each UR with a specific PUC:

- PUC 1: Musical Experience Design
- PUC 2: The New Silence (Sound and mobility)
- PUC 3: Sound of Urban Spaces
- PUC4: Sound and social inclusion

Artist	UR no	Role	Requirement Identified	Relevance per use case				Functional or Non-Functional
				PUC 1	PUC 2	PUC 3	PUC 4	
Tim Otto Roth	UR_1.1	As an artist I want	Aesthetic evaluation of music, perception, and experience.	X				
	UR_1.2	As an artist I want	Scientific methods to explain the psychological, neuronal and socio-cultural basis of aesthetic perceptions of sound	X				
	UR_1.3	As an artist I want	Web-based Systems including Multilingual Natural Language Generation for the assessment of user surveys	X				
PUC 2 - NO ARTIST EXCTRACTED FROM PUC ANALYSIS	UR_2.1		Isolation of the frequency range in which car related sounds occur		X			Functional
	UR_2.2		Use Audio to Image & Image to Audio for the visualisation of car sound recordings		X			Functional
	UR_2.3		Sonification of Human Movement both internally		X			Functional

[3] https://www.guru99.com/functional-vs-non-functional-requirements.html

			and externally of cars, synchronised with car sounds					
Andrea Cera	UR_3.1	As an artist I want	Sound files used for experiments to have good sound quality			X		Functional
	UR_3.2	As an artist I want	Sound files used for experiments to have a duration of at least 2/3 minutes			X		Functional
	UR_3.3	As an artist I want	Real -time Interactive Sonification of Human Movement Qualities, as measuring means of the efficacy of the sound design prototypes			X		Non-Functional
	UR_3.4	As an artist I want	Real-time Automated Analysis of Joint Action and Full-body Expressive Movement as measuring means of the efficacy of the sound design prototypes			X		Functional
	UR_3.5	As an artist I want	To be able to measure breathing / heart rate, electrodermal activity.			X		Functional
	UR_3.6	As an artist I want	Spectral - timbral analysis of sound files both recorded (raw) and manipulated			X		Functional
	UR_3.7	As an artist I want	To record movement and physiological data, as synchronised sequences with the playback of several sound files			X		Functional
	UR_3.8	As an artist I want	To be able to record subtle movements and changes of balance, with the least equipment possible to allow free movement			X		Functional
	UR_3.9	As an artist I want	To be able to analyse subtle movements and changes of balance, using minimally invasive technologies, to allow for a			X		Functional

			natural expression of movements					
Caroline Claus	UR_3.10	As an artist I want	Use methods to render out completely unintelligible voices or conversations that may incidentally be overheard by sensors			X		Functional
	UR_3.11	As an artist I want	To isolate the frequency range in which speech sounds occur			X		Functional
	UR_3.12	As an artist I want	To record triggering signals of movement in public space			X		
	UR_3.13	As an artist I want	To use Web-based Systems for Real-time and Mobile Feedback for collective and individual listening sessions			X		Functional
	UR_3.15	As an artist I want	To be able to Analyse comments on popular urban music videos performed and filmed in the area of interest			X		Functional
	UR_3.16	As an artist I want	Use tools/services for the detection of sonic events, patterns, and figures in public spaces			X		Functional
	UR_3.17	As an artist I want	Link audio data to Audio to Image & Image to Audio software To be able to abstractly visualise the recordings of sonic events, patterns, figures in public space			X		Functional
Joyce, Wendy, Gustavo (Collective)	UR_3.18	As an artist I want	To have a final reflection component that collects a few words from each participant (their post-experience reflection) to string into a collection of poems for advocacy.			X		Functional
	UR_3.19	As an artist I want	To be able to "clean" sound files found on the web and on ocean science			X		Functional

			databases of whale songs, sonic blasts from ships, engine noise from boats, airgun sonic disturbances from seismic surveyors for specific sounds to be more perceptible.					
	UR_3.20	As an artist I want	To translate scientific sea data (sonic activity or sea composition of algal blooms) into tangible, visceral and sonic experiences			X		Non- Functional
	UR_3.21	As an artist I want	To utilise spatial sound technologies in VR and synchronise them with haptic outputs			X		Functional
	UR_3.22	As an artist I want	To use multilingual technologies, such as concept extraction, sentiment analysis and text generation, to analyse the comments of the participants after the experience			X		Functional
	UR_3.23	As an artist I want	To detect of sonic emerging patterns of underwater sounds (recordings of whale songs, sonic blasts from ships, engine noise from boats, airgun sonic disturbances from seismic surveyors and algal blooms)			X		Functional
	UR_3.24	As an artist I want	To be able to collect sounds underwater			X		Functional
Loukia Tsafoulia Alfonso Sevrino	UR_4.1	As an artist I want	Use computer vision to track eye movements eye movements, pupil responses, and blink rates				X	Functional
	UR_4.2	As an artist I want	To connect real time feedback and inform the environment's sonic and light response.				X	Functional

	UR_4.3	As an artist I want	Real-time Analysis of Full-body Expressive Movement: To record, track, and trace body movement (i.e., body rocking) that could serve in the analysis of emotional state and anxiety levels.				X	Functional
	UR_4.4	As an artist I want	To collect data related to the psychological and emotional states of the occupant.				X	Functional
	UR_4.5	As an artist I want	Web-based Systems for Real-time and Mobile Feedback and Multilingual Natural Language Generation: To create a communication interface that facilitates feedback on the interactions between the occupant and the prototype space				X	Functional
	UR_4.6	As an artist I want	Body wearable sensors and biometric recording technologies can be adapted to the space created, allowing "non-invasive" experiences for the space occupants.				X	Functional
	UR_4.7	As an artist I want	To externalise the physiological and psychological states of the user				X	Functional
	UR_4.8	As an artist I want	Real-time Automated Analysis of Joint Action					

Table 3: User Requirements Identified

These User Requirements (URs) are then translated into Technical Requirements (TRs). This translation process establishes a concrete plan for the Key Performance Indicators (KPIs) the system should meet. First set of Technical Requirements analysed from the User Requirements are the following:

ID	TRs	Related URs
TR1	Support audio files with good sound quality	UR_3.1

TR2	Track human movement	UR_3.3, UR_3.4, UR_3.7, UR_3.8, UR_3.12, UR_4.3
TR3	Perform spectral-timbral analysis of sound files	UR_3.6, UR_3.16, UR_3.20
TR4	Synchronise movement and physiological data with the play-back of sound files	UR_2.3, UR_3.9, UR_4.4
TR5	Filter out unwanted noises from sound files	UR_3.10, UR_3.19
TR6	Isolate the frequency range in which speech sounds occur.	UR_2.1, UR_3.11
TR7	The system will be able to generate visual, spatial sound, and haptics	UR_2.3, UR_3.21
TR8	Real-time analysis of full-body expressive movement	UR_3.4, UR_3,5, UR_4.3
TR9	Measure parameters of breathing / heart rate, electrodermal activity	UR_3.5
TR10	Collect and analyse comments on popular urban music videos	UR_3.15
TR11	Link audio data to Audio to Image & Image to Audio software	UR_2.2, UR_3.17, UR_3.18, UR_3.20, UR_3.21
TR12	Collect data with body wearable sensors and biometric record-ing technology	UR_4.6, UR_4.7
TR13	Use computer vision to track eye movements, pupil responses and light in real time	UR_4.1, UR_4.2, UR_4.8
TR14	Collect and analyse data related to the psychological and emo-tional states of the occupants	UR_4.4, UR_4.7
TR15	Develop a Web-based system providing real-time Multilingual Natural Language Generation	UR_1.3, UR_3.13, UR_4.5
TR16	Develop a system to record and analyse sound files	UR_3.1, UR_3.2, UR_3.6, UR_3.7, UR_3.10, UR_3.11, UR_3.16, UR_3.17, UR_3.19, UR_3.21
TR17	Trigger signal of movement in public space	UR_3.12, UR_3.16
TR18	Collect sounds underwater	UR_3.24
TR19	Utilise spatial sound technologies in VR	UR_3.21
TR20	Develop multilingual technologies, with concept extraction, sentiment analysis and text generation	UR_1.3, UR_3.22, UR_4.5
TR21	Detect of sonic emerging patterns of sounds	UR_3.23, UR_4.2
TR22	Develop functionalities for analyzing and evaluating the aes-thetic perception of music, potentially using psychological and neurological principles.	UR_1.1 UR_1.2

Table 4: Technical requirements

4 REQUIREMENTS, SPECIFICATIONS AND FUNCTIONALITIES OF THE RESILENCE COMPONENTS

In this section the description of the prototypes and the components is included, together with Technical Requirements addressed and the input and output data (with samples). More information and demonstration URL's can be found here: **https://github.com/re-silencerepo/resilence_main**

4.1 Soft, a prototype for responsive environments and neurodivergence

Project Soft is a spatial wearable; an encapsulated, safe space with an adaptive interior environment where sound and light experience occurs via a full-body approach.

In combining science, art, technology, and design expertise with the lived experience of neurodivergent advocates, this project investigates the challenges sonic environments present to neurodivergent individuals and the healing opportunities of sensory multimodal and responsive spaces for all.

Name of the component: Soft
Description of the component
Project Soft has dual goal: • To put forth a broader research framework for embodied and affective systems addressing design for inclusive environments and the critical role of architecture in engaging the UN Sustainable Development Goals • To offer a spatial solution that responds to the needs of neurodivergent individuals

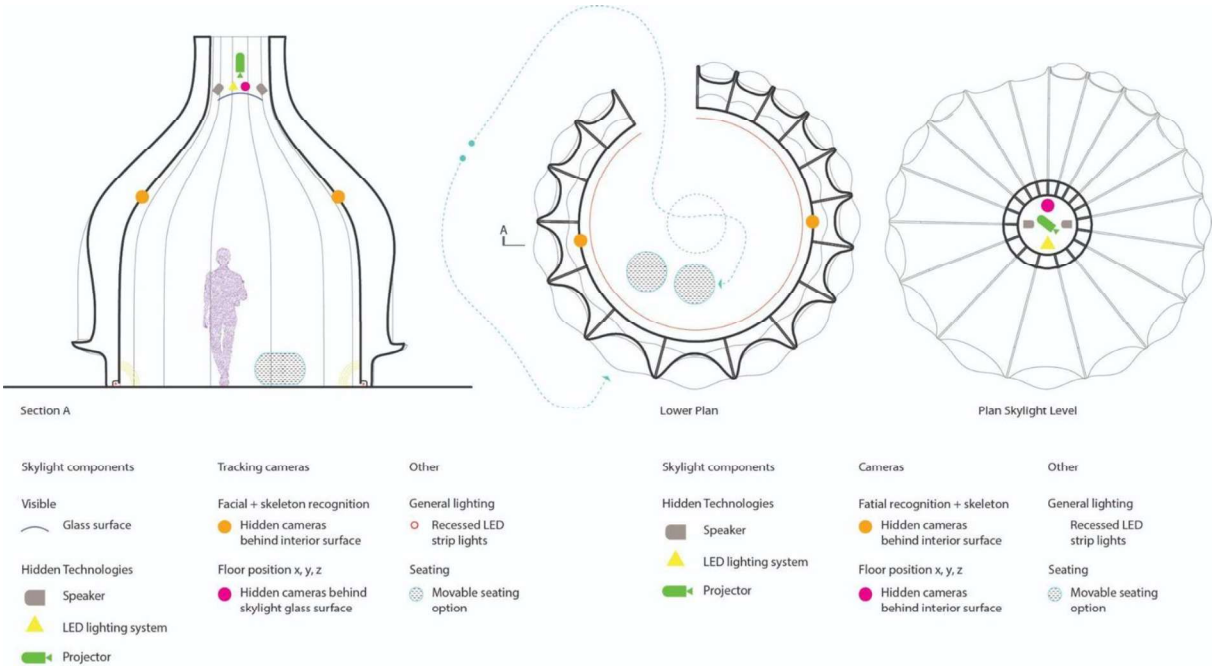


Figure 1: Soft Physical system

Spatial Actions

Create immersive experiences within the space to engage the visitor in sensory interconnected modalities.

Sound

Adapt sounds/music

(Wavelength, amplitude, frequency, time period, velocity/loudness, pitch, audible, and infra-sonic sound qualities)

Light

Change ambient lighting

(Light Coloration/frequency, intensity, oscillation)

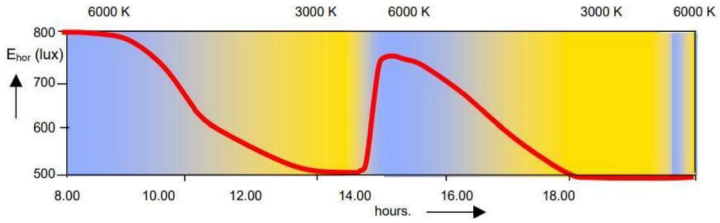


Figure 2: Light Coloration/frequency, intensity, oscillation

Scenario with gradually changing lighting level (red line) and color from 6000 K (blue area in the graph) to 3000 K (yellow area in the graph) according to the “human” rhythm.

Immersive Projection

Project images/animations

Create immersive experiences.

(visual representations, dynamic light projections, and soundscapes) within the space prototype

to engage the user in several sensory interconnected modalities.

User Interface

Create a communication interface to:

- Facilitate feedback on the interactions between the visitor and the prototype space and generate assessment reports
- Facilitate the support of medical experts and caregivers towards the space user
- Act as an evaluation mechanism for the prototype's impact
- Allow the visitor to override actions and voluntarily provide personal, demographic data

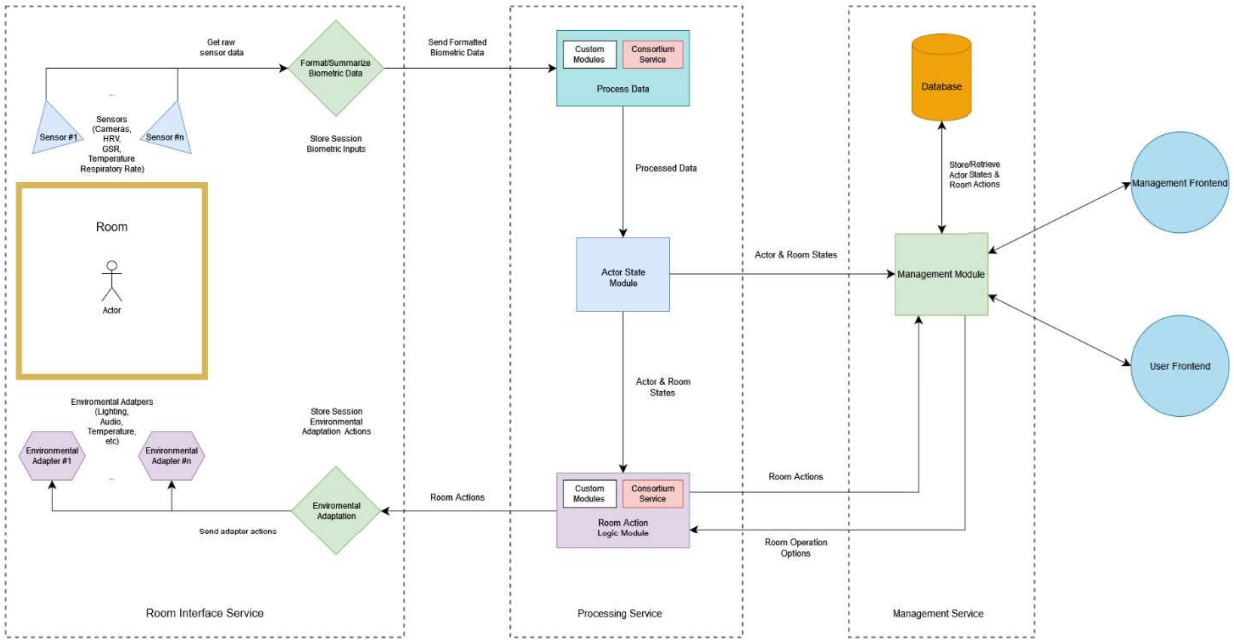


Figure 3: Preliminary Flow Diagram of Interactions

Analysis of the basic components

Sensors

Produces raw biometric data (Video, Images, Heart rate, Galvanic skin resistance)

Format/Summarise Biometric Data Module

Formats and summarises the partial incoming raw data to be ready for processing

- Summarise in a specific time frame since data from different sensors come at different time frames.
- Change raw data format(ex. binary format) from sensors to ideal format(ex json) for processing

Process Data Module

Uses Custom Modules and/or Consortium Services to process the data into a useful interpretation for the Actor & Room State Module

- A custom module uses input images to handle pose/body posture tracking
- A custom module uses input images to handle eye tracking
- A custom module uses input images to handle person 2D tracking

Actor & Room State Module

Uses the process data to determine the actor state.

- For example, actor has high heart rate and is pacing means actor is upset

Action Logic Module

Uses actor state information to determine actions

- For example, actor is upset. Calming music should play

Environmental Adaptation Module

Translates the action into an adapter command

- Communicate with sensor to play sound/music, change lighting, project images/videos

Environmental Adapter

Executes the action

- Play sound/music, change lighting, project images/videos

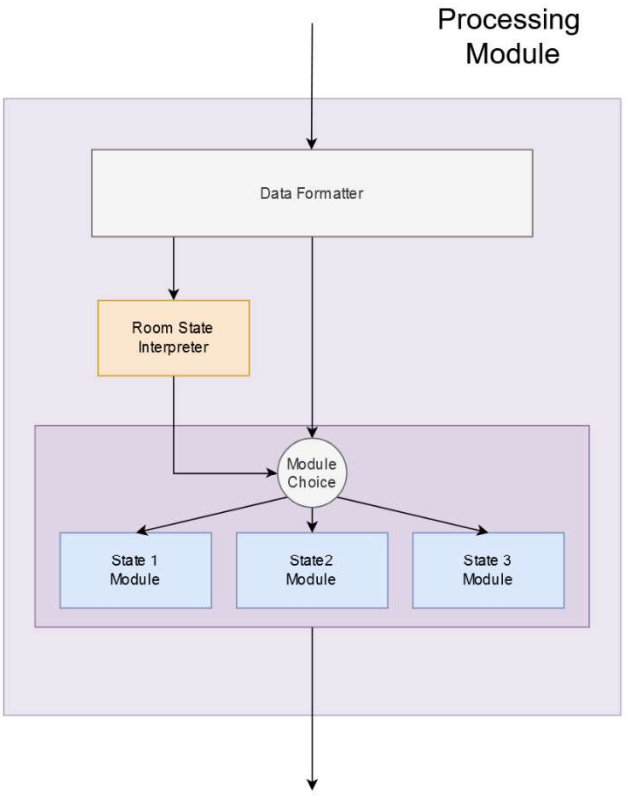


Figure 4: Processing Module flow

Frontend

Maintenance Frontend

An interface that a room manager can use to perform specific actions. Examples: Start a session, terminate a session, modify room operation, input user personal data, view database records, extract database records, etc

User Frontend

An interface where a user can perform specific actions. Examples: Enter personal data, answer questions about their experience in the room, provide feedback, etc

Backend

Management Service

Connects the frontend to the database. Communicates with the Processing Service to modify room operation functionality, like change operation mode or session termination.

Processing Service

Processes the raw data into actor states and room actions. Communicates with the room interface service to receive the sensor data. Sends room actions to the room interface service to be executed. Communicates with the management service to provide the data that should be stored in that database(actor states-room actions?) and to receive room operation functionality modifications, like change operation mode or session termination.

Room Interface Service

Communicates with the room sensors to collect the raw data and provides it to the processing service. Receives actions by the processing service that should be performed and uses the adapters to perform them.

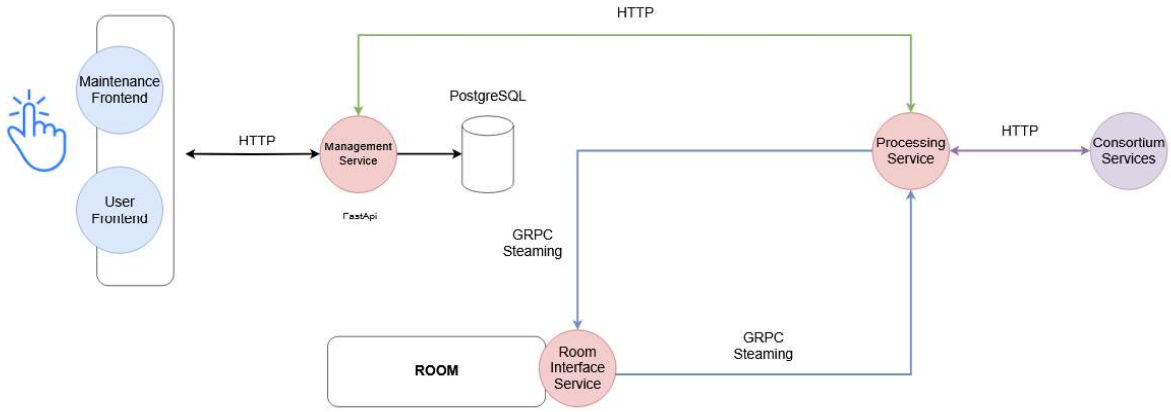


Figure 5: Room Interface Service flow

Technologies focused until this period:

Skeleton Pose tracking

Track the skeleton of a person using a depth or a typical RGB video camera. Skeleton is represented by 3D landmarks.

- Initial Research on available hardware, algorithms and software libraries for pose tracking. Devices (Kinect for Windows, Azure Kinect, Intel Realsense, Orbbec), SDKs (Nuitcracker, Lightbuss, Google Mediapipe)
- First implementation based on Google Mediapipe pose tracker model.

Face/Eye tracking

Detect and track the face and eyes of a person using a normal RGB camera.

- Initial Research on available algorithms hardware and software solutions and algorithms for face/eye tracking.
  Google mediapipe, OpenCV + HaarCascade, Gaze tracking library, Open-eye-detector, BlinkLinMuIT
- Implemented an eye tracking pipeline that is distance independent.

Top-down person tracking

Track the position of a person in 2D and extract motion statistics (translation, velocity, acceleration) using a depth camera.

- Initial Research on available algorithms hardware and software solutions and algorithms for face/eye tracking
- Implemented a top-down person tracking service based on Intel RealSense depth camera and OpenCV computer vision library

Face tracking pipeline and example

Eye tracking, facial landmarks and blendshapes require close-up images of a person's face. To overcome this we implemented the following pipeline:

- Predicts facial landmarks and blendshapes in the camera frame
- If the module fails to calculate facial landmarks, the camera frame is passed to a more robust face detector
- The robust face detector detects and crops the faces on the camera frame
- If a face is found and cropped, it is fed once more to the facial landmarker / blendshapes extractor.
- If the above process fails to detect a face, it is assumed that no faces exist in the camera frame.

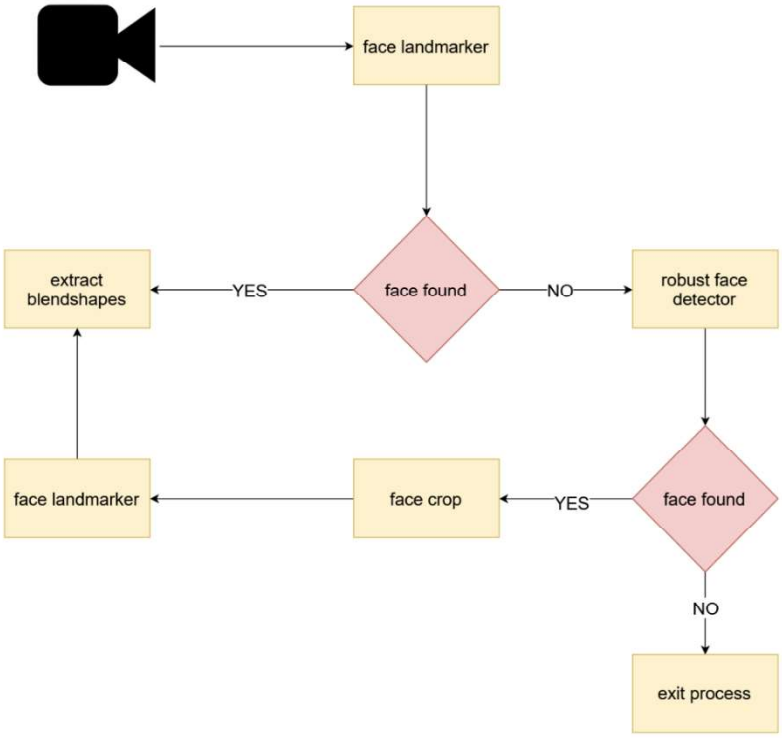


Figure 6: Face tracking pipeline

Modules

Different modules can be interchanged in the pipeline, allowing for modular experimentation. The list of currently available modules is presented below:

Camera	Face landmarker	Blendshape extractor	Robust face detector
Realsense	Google Mediapipe face landmarker	Google Mediapipe face landmarker	Google Mediapipe object detector
Conventional PC camera			Facenet (MTCNN) face detector

Camera

Currently two camera types are supported:

- Conventional pc camera: just the standard laptop camera. Performance varies based on camera specs
- Realsense camera: 1080p stream with additional depth data

Face Landmarker / Blendshape Extractor

Google Mediapipe face landmarker is capable of detecting facial landmarks as well as predicting facial blendshapes. It only works on small distances / zoomed images.

Robust face detector

A robust face detector is a face detector that can detect faces on greater distances. That way we can crop and re-feed a detected frame into the shorter-range face landmarker. Two robust face detectors are currently supported:

- Google Mediapipe object detector: a general object detector that has the capability of detecting faces on various distances
- Facenet (MTCNN): a state-of-art face detection and face landmarking model

Technical Requirements addressed

Soft addresses the following TR(s):

- TR2 Track human movement
- TR4 Synchronise movement and physiological data with the playback of sound files
- TR8 Real-time analysis of full-body expressive movement
- TR12 Collect data with body wearable sensors and biometric recording technology
- TR13 Use computer vision to track eye movements, pupil responses and light in real time
- TR14 Collect and analyse data related to the psychological and emotional states of the occupants
- TR15 Develop a Web-based system providing real-time Multilingual Natural Language Generation
- TR20 Develop multilingual technologies, with concept extraction, sentiment analysis and text generation
- TR21 Detect of sonic emerging patterns of sounds

Input Data with sample

The soft project used as input the video captured by:

- typical RGB video camera for **Skeleton Pose tracking - Face/Eye tracking**
- depth camera for top-down for **Top-down person tracking**

Output Data with sample

The output for soft is different for each component:

- **Skeleton Pose tracking**

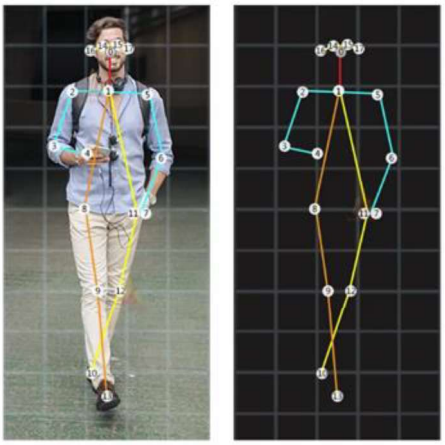


Figure 7: Skeleton Pose output

- **Face/Eye tracking**

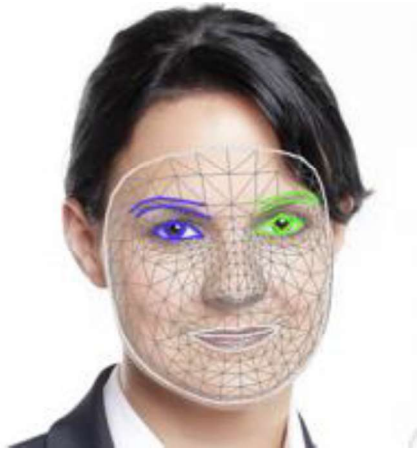


Figure 8: Face/Eye tracking output

- **Top-down person tracking**

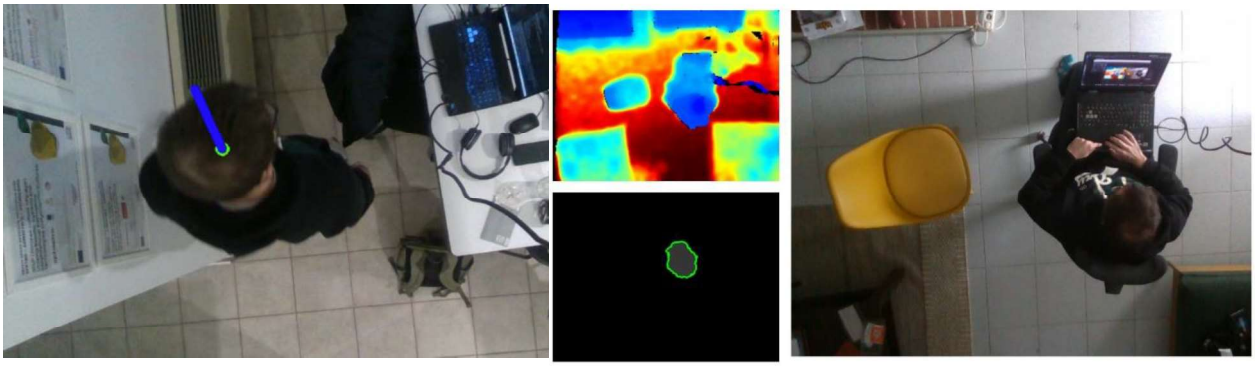


Figure 9: Top-down person tracking output

Screenshots

Figure 10: Skeleton Pose Tracking example based on Google Mediapipe

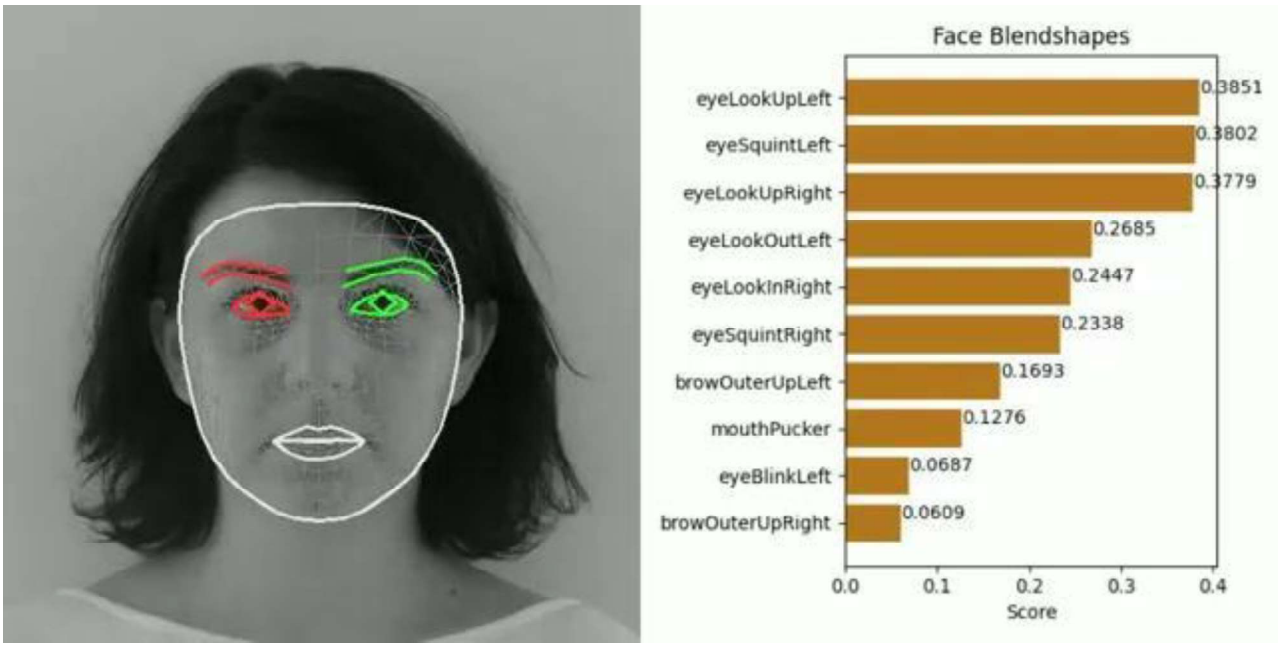


Figure 11: Face tracking and face blendshapes prediction (Google Mediapipe face landmarker)

- The face landmark detection model can process facial landmarks and predict facial blendshapes.
- Blendshapes represent high-level facial features (eyes closed, eyes squinting, mouth smiling, etc.).
- Each blendshape has a prediction confidence value between 0 and 1.

Figure 12: Top-down person tracking example 1

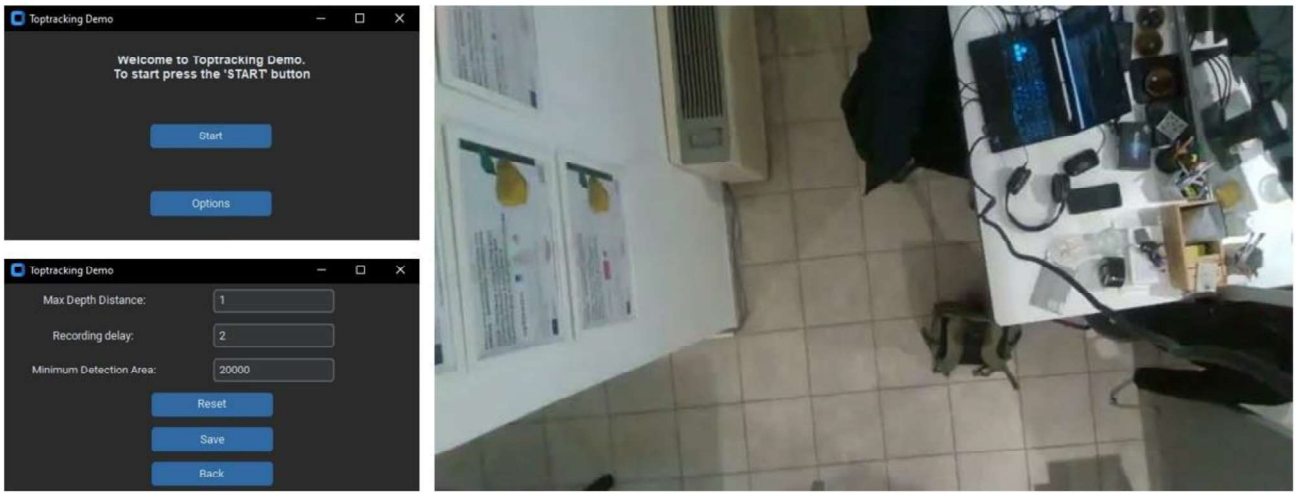


Figure 13: Top-down person tracking example 2

4.2 Audio Recording Toolbox

The Audio Recording Toolbox is comprised of specialised equipment that enables in real-time recording of urban soundscape while complying with GDPR. The main components are a Raspberry Pi 4B model[4] with an 8GB RAM used for audio signal recording and processing, paired with the AudioMoth (Hill et al., 2019) microphone for audio input.

Name of the component: Audio Recording Toolbox
Description of the component
The Audio Recording Toolbox consists of portable equipment, configured to support real-time GDPR-compliant sound analysis to ensure privacy in urban areas. The AudioMoth microphone is the input sensor used to capture the soundscape.

Figure 14: Audio Recording Toolbox

The collected sound is analysed to detect human voices/speech and then processed to subtract the human voice signal. Consequently, the human voice signal is filtered appropriately to blur the signal, rendering the identity of the speaker/speakers or the content of the speech incomprehensible.

The Raspberry Pi Single-Board Computer (SBC) records two audio files in .wav format. The first file contains the background sounds/noises as collected during the recordings, having subtracted the human voice signal. The second file contains the separated human voice signal though blurred.

The purpose of this component is to ensure that the collected sound data in urban environments comply with the GDPR legislation, and in more detail, that there is no breach in privacy of the persons located at urban spaces where soundscape recordings take place. The collected data will

[4] https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/

be used for further analysis with emotion recognition algorithms or may be processed /mixed by the artists to create new content.

Since the Audio Recording Toolbox component needs to be portable and autonomous equipment, in order to save power and increase its installation time to an area, the files are saved to the Raspberry Pi's local SD card. Since the Source Separation algorithm is developed in Python3, the PyAudio package is used to provide bindings for the audio I/O library PortAudio v19, to allow the communication between the devices (Raspberry Pi and AudioMoth).

Technical Requirements addressed

The Audio Recording Toolbox component addresses the TO1 Art-Driven Experimentation toolkit that aims to co-create products and services, and the following TR(s):

- TR6 Isolate the frequency range in which speech sounds occur
- TR16 Develop a system to record and analyse sound files

which are related to the following user requirements UR_3.10, UR_3.11, UR_3.16, UR_3.17

- Quadraphonic sound recordings (4 mics synchronised)

Input Data with sample

The input data of the Audio Recording Toolbox is the collected soundscape captured by the microphone, that is not saved (at its original form) at the device.
The Pyaudio package is used to establish communication with the AudioMoth device. The function that allows the audio signal collection from a microphone connected to a port of the Raspberry Pi is displayed below.

```
#Start recording:
def record_audio(RATE, RECORD_SECONDS):

    CHUNK = 512
    FORMAT = pyaudio.paInt16 #paInt8
    CHANNELS = 1

    p = pyaudio.PyAudio()


    stream = p.open(format=FORMAT,
                    channels=CHANNELS,
                    rate=RATE,
                    input=True,
                    frames_per_buffer=CHUNK) #buffer

    print("* recording")

    #frames = []
    audio = np.array([], dtype=np.int16)
    now = datetime.now()
    recording_time = now.strftime("%d-%m-%Y %H-%M-%S")

    for i in range(0, int(RATE / CHUNK * RECORD_SECONDS)):
        data = stream.read(CHUNK)
        audio = np.concatenate((audio,np.frombuffer(data, dtype=np.int16)))
```

```
    stream.stop_stream()
    stream.close()
    p.terminate()
    output_path = os.path.join("bin",
                               "separated_results",
                               Path(recording_time.split()[0]),
                               Path(recording_time.split()[1]))
    #output_path = os.path.join("separated_results")

    os.makedirs(output_path, exist_ok=True)
    #sf.write(os.path.join(output_path,
              "original_audio.wav"),
              audio/np.abs(audio).max(),
              RATE)

    return audio/np.abs(audio).max(), recording_time
```

The collected audio signal is not saved to the SD card, until the processing concludes.

Output Data with sample

Two audio files in .wav format are saved to the disk. Each file's duration is approximately 10 seconds. If the user selects to record for 60 seconds, 6 folders named sequentially from 0 to 5, will be created to include each save file.

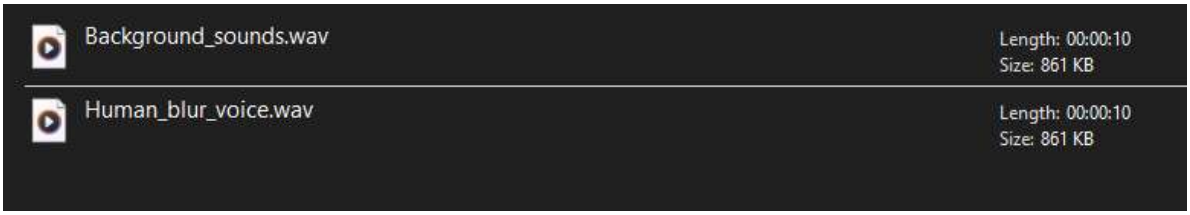


Figure 15: Saved .wav files at Raspberry Pi OS

These 5 folders are placed in folders named after the exact timestamp that the recording started. Also, a timestamp-named folder is saved inside a date-named folder, to organise the collected material.

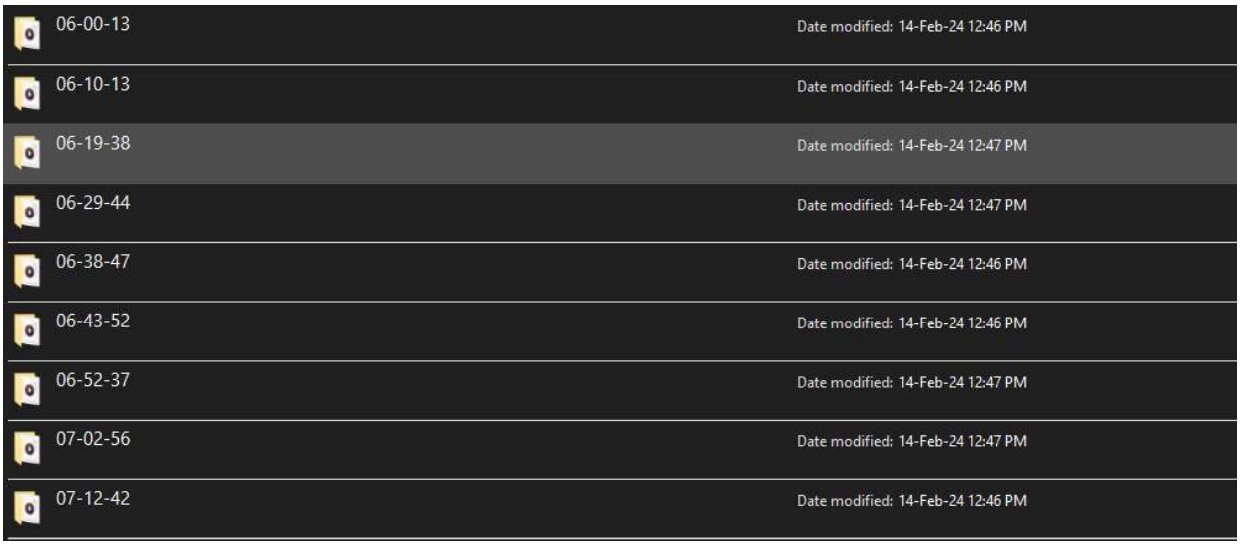


Figure 16: Folders containing the recordings that took place the same day, and are named as the exact timestamp the recording started

As mentioned previously, the saved data do not contain the original collected soundscape, but are the separated and filtered human voice plus the rest of the area's sounds.

Software architecture

As depicted on Figure 17 the recordings are scheduled and conducted, following the steps below:

1. Crontab schedules bash scripts activation
2. Bash scripts run for a specific time limit and they call the USS (Universal Source Separation algorithm) periodically to ensure synchronised Quadraphonic recordings
3. When called USS algorithm opens the AudioMoth mic for recordings (rec duration is set by the user)
4. USS separates the human voice from the background sounds
5. Then separated human-voice is blurred in real-time
6. Background sound and blurred human voice are saved separately at the SD card

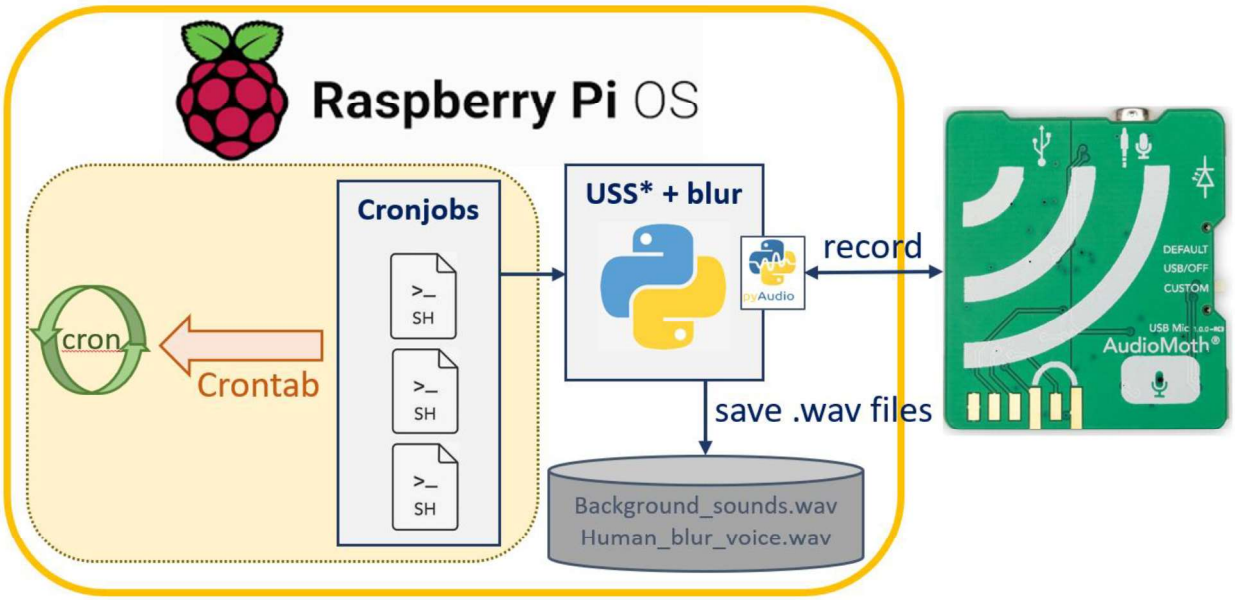


Figure 17: Software diagram

Prototypes and timelines

For the Audio Recording Toolbox component we have completed the prototype, within the time the contract of the artist stipulates:

- 1st Prototype (V1) – M18

The prototype of the Audio Recording Toolbox, that is described as a component of the ReSilence toolkit at the current deliverable (D5.2), is included in the 3rd milestone (MS3) at M20 of ReSilence project. The current deliverable (D5.2) contributes to MS3.

- Final Experiment – M21

According to Caroline Clauss's contract and the performance designed to conclude said artist's project, the final version of the Audio Recording Toolbox component will be deployed by M21 and the idea is to use the collected audio data as an input to an emotion recognition algorithm, either as they were recorded, either after being further processed. The final performance is related to addressing people to assess emotionally the audio files mentioned above, and then to compare these results to the output of the emotion recognition algorithm.

The data that will come from this analysis will be used for a socio-psychological assessment of the dataset annotation processes along with the dataset's inclusion.

4.3 Andrea Cera / ReSilent (Android app)

The smartphone app ReSilent (for Android OS) is a dissemination / proof-of-concept tool created to demonstrate the low-intrusiveness sound design techniques developed in ReSilence project.

Name of the component: Andrea Cera / ReSilent (Android app)
Description of the component

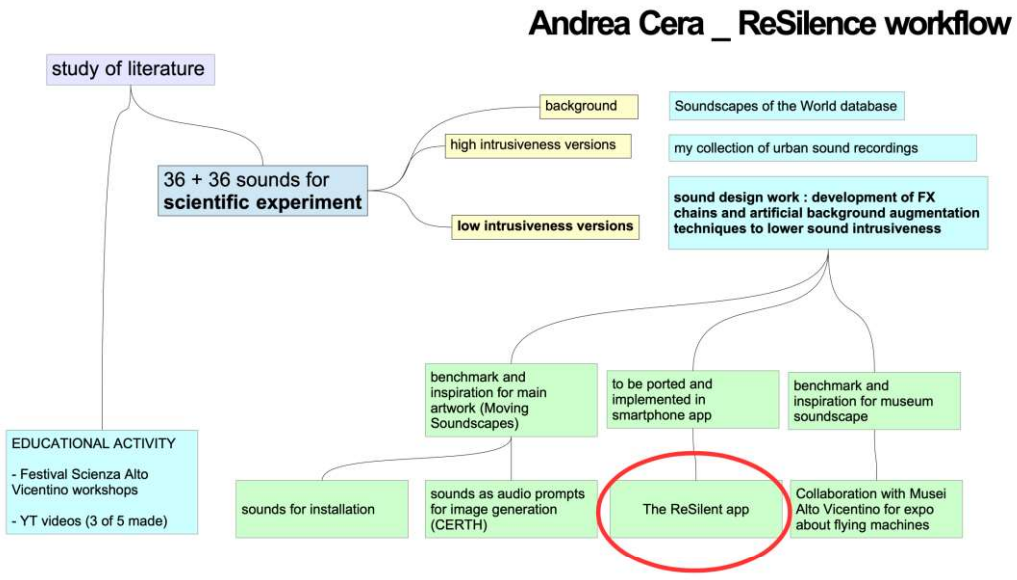


Figure 18: The ReSilent app in the ReSilence workflow

The ReSilent app allows to apply in real-time a subset of the sound design techniques used to lower intrusiveness in the sounds used for the scientific experiment.

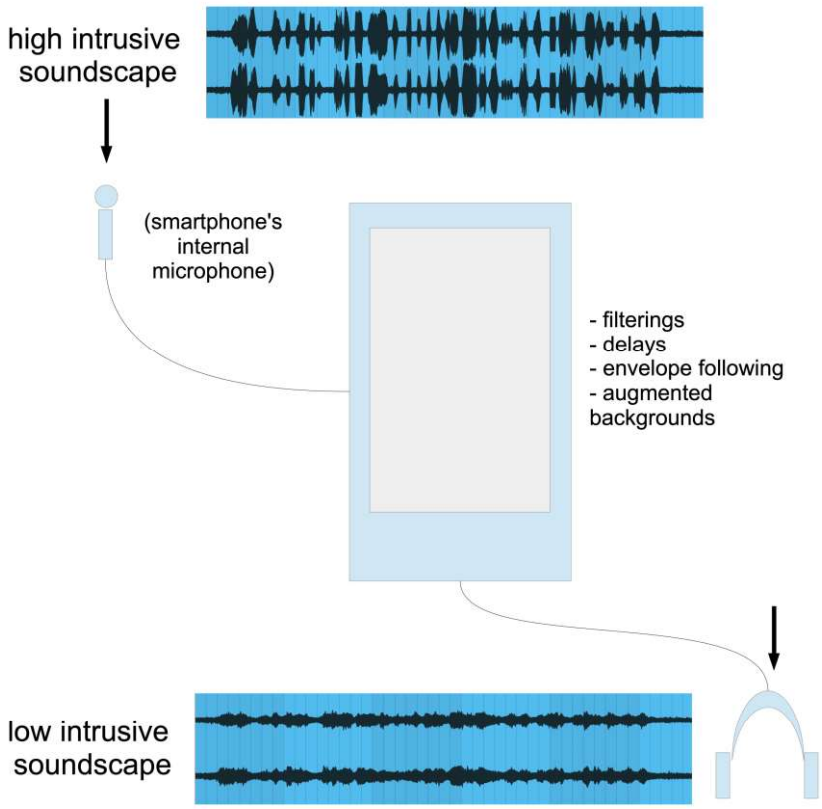


Figure 19: Application flow diagram

The app is intended to be activated when the user finds intrusive a certain soundscape (ex. people talking loudly on a train trip). Once the app has been launched, it starts transforming the sounds captured by the smartphone's microphone.
The transformations are a subset of the strategies used to prepare the “low-intrusiveness” version of the experiment sounds. The processed sounds are sent to the smartphone's audio output, when listened to with headphones, these sounds blend with the intrusive sounds from outside and create a smooth mixture of real and augmented background, longer temporal shapes, a sort of abstract and fluid granulated material, which makes difficult for intrusive sounds from outside to become salient.
The programming of the internal FX chain is based on the RNBO library of MaxMSP. In this following image it is possible to see a simple implementation of 1 filter + delays section and 3 players of background sounds – used for testing the overall infrastructure.

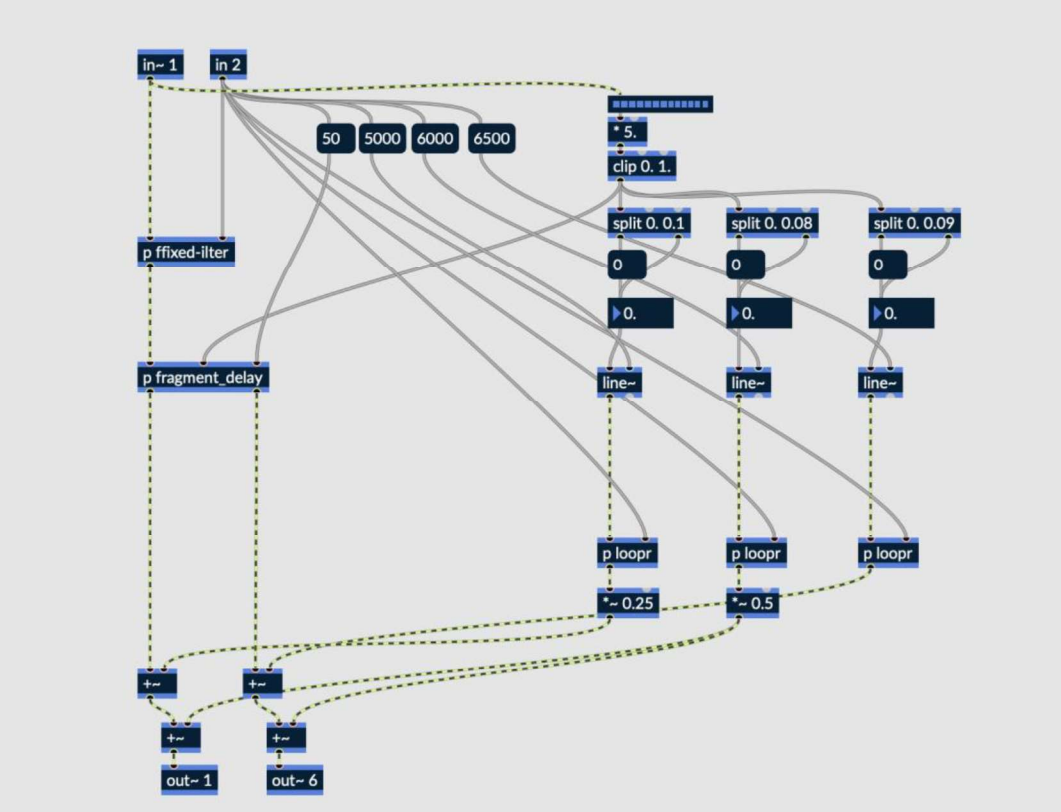


Figure 20 Single filter with delays section and three players of background sounds

The next iterations will considerably complexify this architecture, to make it as similar to the techniques used for the experiment as possible (according to the smartphone's CPU limits).

Technical Requirements addressed
The component addresses the following TR(s): • TR1 Support audio files with good sound quality • TR2 Track human movement • TR3 Perform spectral-timbral analysis of sound files • TR4 Synchronise movement and physiological data with the playback of sound files • TR8 Real-time analysis of full-body expressive movement • TR9 Measure parameters of breathing / heart rate, electrodermal activity • TR16 Develop a system to record and analyse sound files which are related to the following User Requirements: UR_3.1, UR_3.2, UR_3.3, UR_3.4, UR_3.5, UR_3.6, UR_3.7, UR_3.8, UR_3.9
Input Data
The input data of the module is audio sounds.
Output Data
The output data produced by the module is audio sounds.

Screenshots

The Android implementation of the RNBO patch is made by my collaborator Julien Bloit using the JUCE environment. The app interface will be quite simple, offering control and visualisation of input / output levels. In the following image, the working prototype app installed on a Motorola g32.

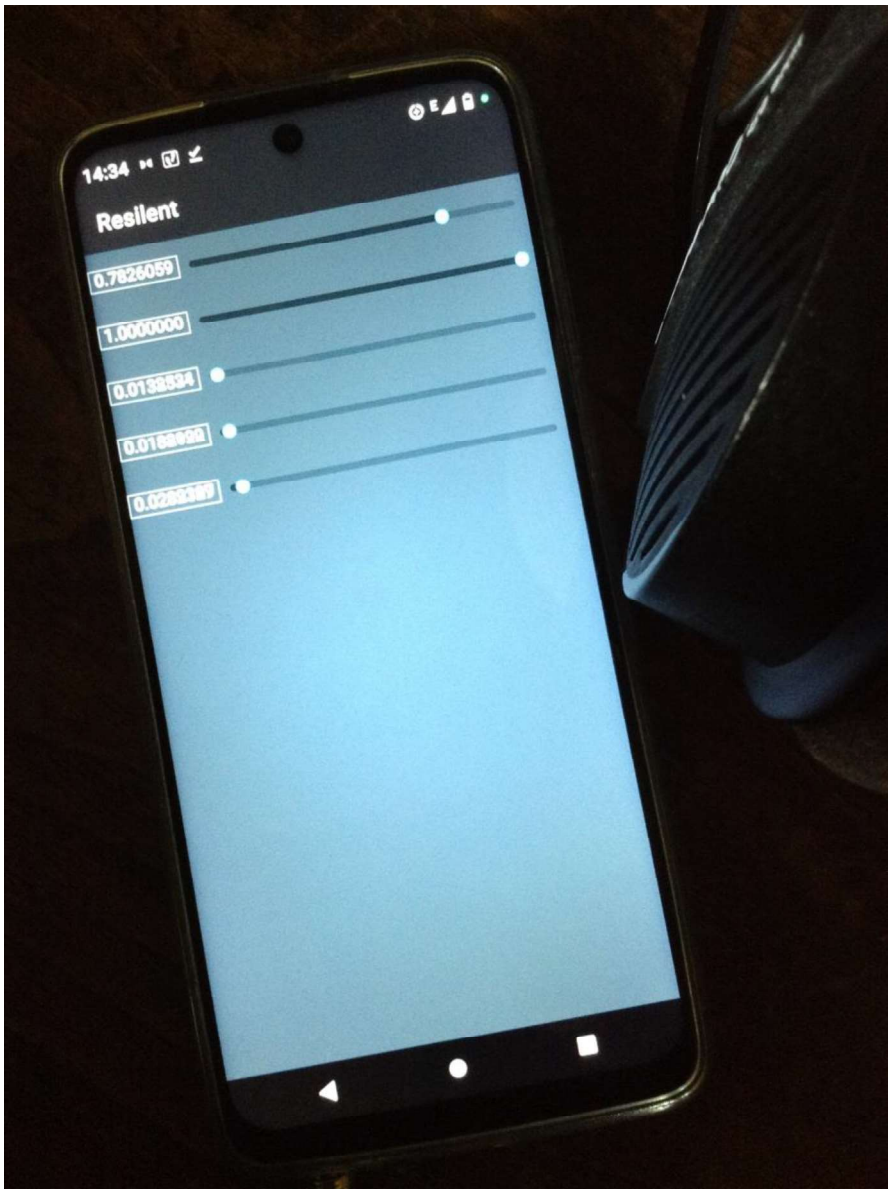
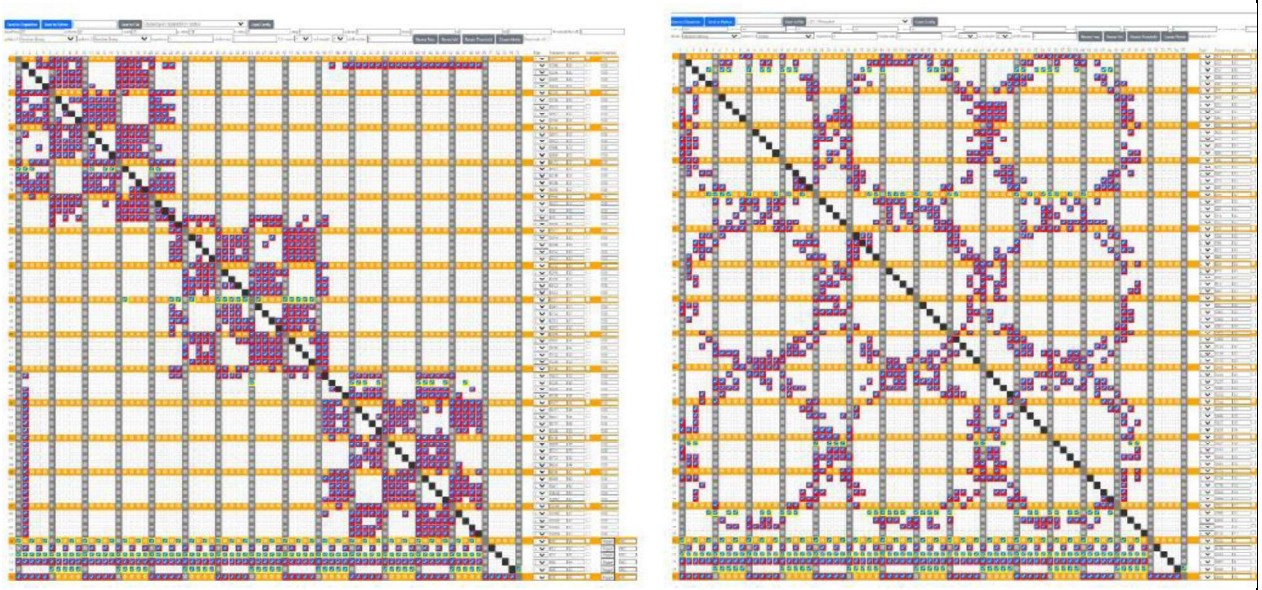


Figure 21: Prototype app installed android phone

4.4 Theatre of Memory

The Theatre of Memory is based on networks of interacting sounds.

Name of the component: Theatre of Memory
Description of the component
A crucial tool to design such networks is a connection matrix defining to which speakers an individual speaker is listening to. For 70 audio neurons this leads to 4900 potential connections which can't be checked and edited all manually. So, a Pythons based pattern generator was programmed to generate and to mix patterns, but also to add random noise.  Figure 22**:** Connectivity matrix with a circular pattern (right) or three loosely interconnected subnetworks based on a chequerboard pattern (left). Each row encodes a transmitting neuron and a column a receiving neuron The pattern processing was refined during the development phase adding also an import function to add bitmap drawings.

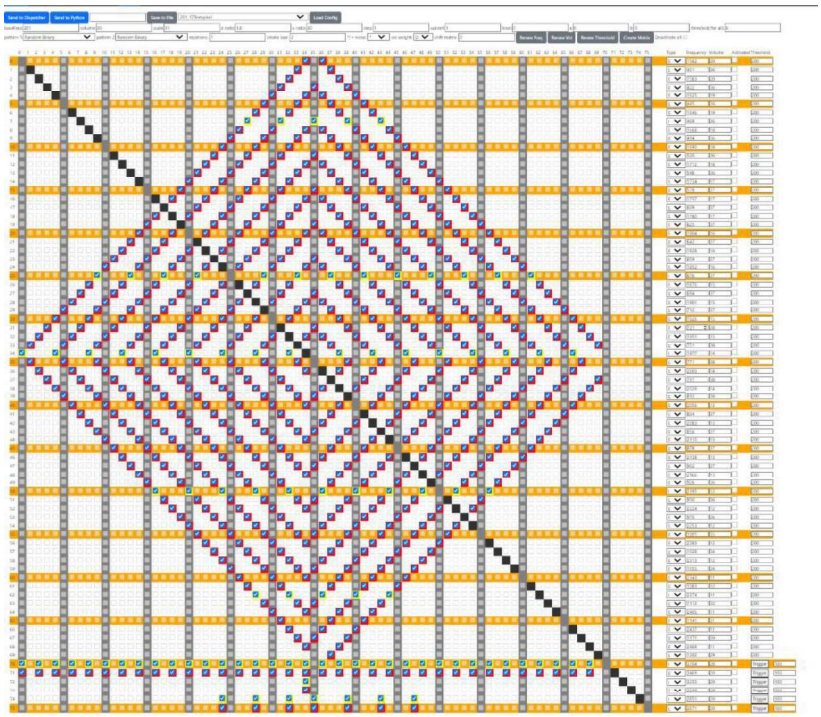


Figure 23: Spiral pattern based on bitmap drawing imported with the browser-based connection matrix tool. This pattern based on diagonal lines encodes two subnetworks

Figure 24: Simulation with 40 audio neurons for the lobby of the MPIEA including also sitting cushions

Hardware setup in Berlin The installation at the Tieranatomisches Theater (TA T) in total 1.5 t of rubber coated concrete plates were carried on the balcony to function as counterweight for the three major steel ropes. The ropes allowed a pending structure with three lines of 1-point-

trusses to hold the individual strings, the dispatchers and the cables.

The major tasks to track the sine tones played by other speakers at high precision are:

- Adapting the firmware to the new processor generation allowing a frequency tracking at a higher resolution including the full audio spectrum and even parts of low-frequency ultra sound.
- Interfering noises caused for instance by talking visitors or by movements in the wooden auditorium had to be taken into consideration.
- A solution needed to be found for the extremely long reverberation time of four seconds caused by the theatre cupola. Such a long time bears the risk that one and the same acoustic signal could trigger several times.
- Finding a way to control the quality of the frequency analysis and how precise sine tones played by other speakers are tracked.

The tasks were successfully tackled:

- The new micro controller ESP32 S3 comprises 16 times more RAM allowing massive improvements in the frequency tracking based on a FFT (Fast Fourier Transform). An adequate precision of the tracking could be realised for the frequency spectrum of 20-2000 Hz measuring the audio spectrum in time windows of 200-250 milliseconds at a resolution of 4 Hz. The sensitivity can be adapted by frequency dependent thresholds.
- Most noises can be filtered out by comparing the amplitude with the two neighbouring frequency windows. Low frequency noises can be filtered out by that local amplitude comparison. Clapping hands were one of the few acoustic interferences which still have an effect – possibly also a way to be used later for human interactions with the system. Another trade-off is that only every 4th frequency window can be used by that method, so the minimal frequency interval is 16 Hz. But this linear propagation of frequency windows has only an effect on the intervals for the lower audible spectrum.
- The reverberation issue could be solved by comparing the amplitude of a frequency with the previously measured amplitude.
- To control the analytic quality of individual units they can be accessed via Wifi to monitor the spectrum detected. Here a special Javascript based plug-in was developed for the browser-based control environment.

Control environment Controlling the environment needs two major components: the browser-based matrix editor as visual interface to design the interaction networks and the Python based script doing the math behind. The Python scripts also manage the transfer of the network parameters via TCP/IP to the electronics telling the units which frequency to play and which frequencies do have an excitatory or inhibitory effect.

A third component which was programmed is a six-channel sequencer controlling six extra speakers functioning just as external triggers for the 70 speakers. This sequencer was realised with the visual programming environment Puredata (Pd Vanilla), which was used also to control parameter changes via cue lists.

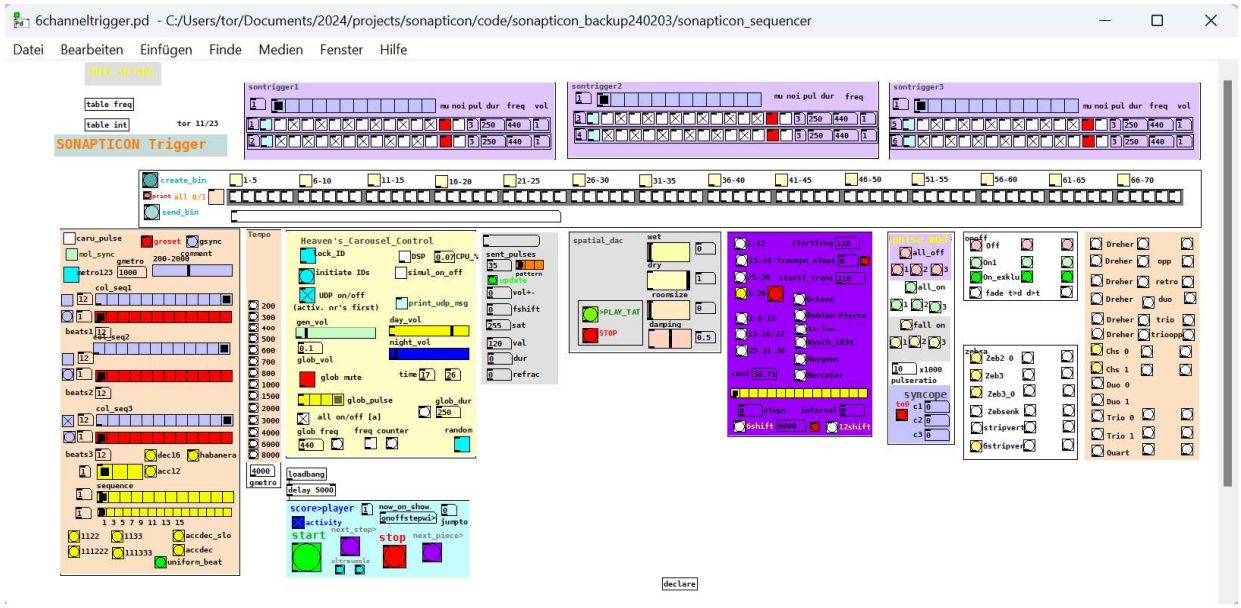


Figure 25: Sequencer like GUI based on Pd

Technical Requirements addressed
Theater of Memory addresses the following TR(s): • TR15 Develop a Web-based system providing real-time Multilingual Natural Language Generation • TR20 Develop multilingual technologies, with concept extraction, sentiment analysis and text generation • TR22 Develop functionalities for analyzing and evaluating the aesthetic perception of music, potentially using psychological and neurological principles which is related to the following User Requirements: UR_1.1, UR_1.2, UR_1.3
Input Data with sample
The Theater of Memory uses as input data audio sounds
Output Data with sample

The Theater of Memory produces as an output audio sounds through the multiple loudspeakers

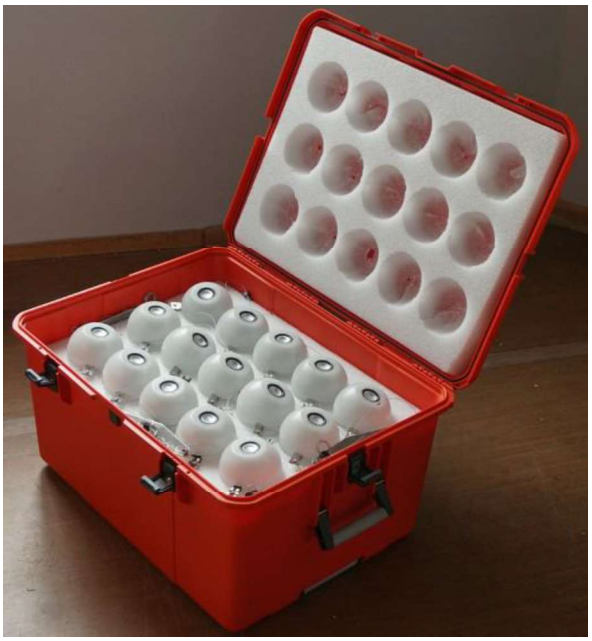


Figure 26: Speakers used in Theatre of Memory

Screenshots

Figure 27: Happening like atmosphere in Frankfurt with people sitting on little circular carpets

Figure 28: Opening at the TA T with a full house theatre

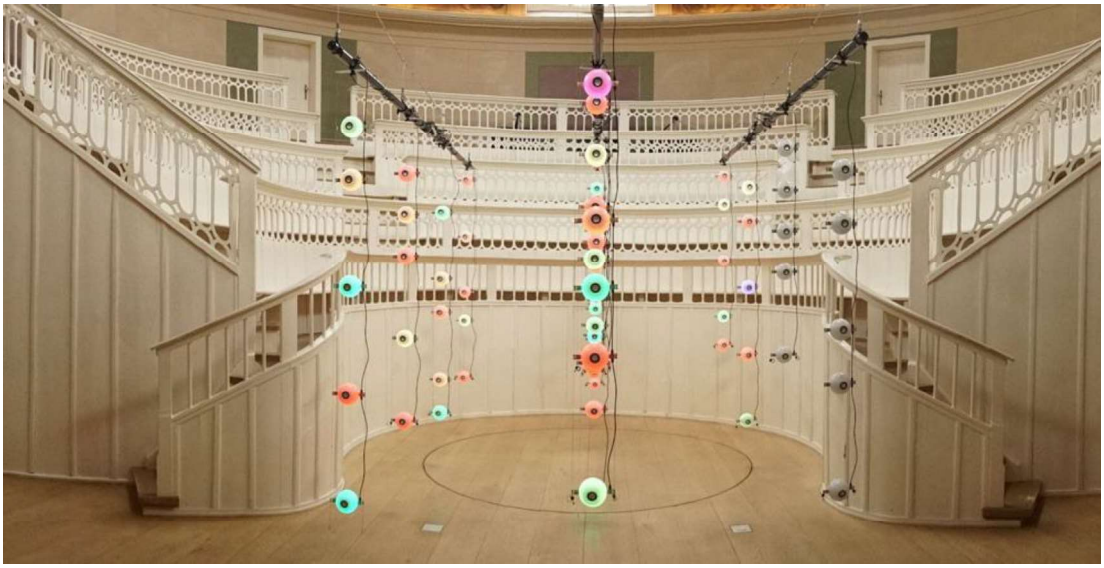


Figure 29: Daylight impression of the Theatre of Memory in March 2024

5 CONCLUSIONS

In this deliverable, we delved into the technical specifications of the ReSilence project's current state prototypes. We began with a foundational overview of established practices for requirements collection and analysis. Following this, we explicitly defined the scope of this activity within the context of ReSilence, ensuring a targeted approach.

Next, we embarked on a detailed discussion of the technical requirements derived from user requirements. This analysis encompassed services, machines, and user tools, providing a comprehensive picture of the technical needs for the project.

To further solidify understanding, we provided an in-depth overview of the current state for each prototype and its underlying architecture. Each prototype received individual attention, including a thorough description, the input and output data it utilises, and the specific requirements it aims to fulfil. This detailed breakdown was further enhanced by insightful screenshots capturing each experiment's essence.

6 REFERENCES

Hill, A. P., Prince, P., Snaddon, J. L., Doncaster, C. P.,and Rogers, A. 2019. *"AudioMoth: A low-cost acoustic device for monitoring biodiversity and the environment"*, HardwareX,vol *6*, e00073.

Pohl, K., and Ulfat-Bunyadi, N. (2013). *"The three dimensions of requirements engineering: 20 years later",* Seminal contributions to information systems engineering: 25 Years of CAiSE, p. 81-87.