Design and Development of a Low-Cost Noise Sensor-Network Based on the senseBox Platform

Masterarbeit

im Studiengang **Computing in the Humanities**
der Fakultät Wirtschaftsinformatik und Angewandte Informatik der
Otto-Friedrich-Universität Bamberg

Professur für Smart Environments

Verfasser: Lukas Schwarz
Matrikelnummer: 1234567
E-Mail: `mail@lukasschwarz.org`

Prüfer: Prof. Dr. Diedrich Wolter

30. März 2021

Contents

Abstract

This thesis describes the design and development of an easy to use and easy to build network of noise sensors for use with the senseBox platform. Its focus lies on usability by unexperienced citizens and journalists. A study with sensor journalists was conducted to find the requirements that are important for such a network aimed at a lay audience. The technical details of the software implementation as well as the hardware choices are briefly described and discussed. Additionally, ways of visualising acquired measurement data with accessibility for members of the general public in mind are explored and finally evaluated in a questionnaire.

1 Introduction

According to the Merriam-Webster dictionary, *noise* is any sound that is undesired or that interferes with one's hearing of something. Thus, noise can be considered a subset of sound that carries an unpleasant and often unwanted characteristic. Especially since mankind's move from an agricultural and more rural form of living to an industrialised, clustered lifestyle with highly populated cities, the level of noise from both humans and machinery has naturally increased.

This special subset of sound is not only unpleasant but can also be considered unhealthy, when exposed to for prolonged amounts of time. The *World Health Organisation (WHO)* names noise as a growing concern and one of the most important environmental risks to health. During night-time, prevalently annoyance and sleep disturbance – which in turn are linked to physiological and psychological effects – are of concern. Day time health outcomes include cardiovascular disease such as hypertension, general chest pain or more severely: heart attacks. Cognitive consequences include impaired oral and reading comprehension that especially pose a risk to children. They report that one third of people surveyed throughout Europe state having problems with noise [36].

While too much exposure to noise can evidently implicate adverse health risks, individuals can lack the means to inform themselves about the levels of sound in their surroundings. Some major cities and hot spots such as smaller settlements in proximity to an airport may have noise data readily available, yet exposure to noise may very well occur outside of those hot spots where data is usually not gathered in the first place. In addition, noise exposure is usually not mapped according to actual

measurements, but instead modelled or calculated based on sound sources. Moreover, those maps focus on transportation noise alone, although additional factors are involved [22].

This thesis aims to alleviate this lack of information by describing the design and development of an easy to use and low-cost noise sensor prototype, aimed at both interested citizens and members of the media industry in order to provide easily accessible data. Such a sensor can either be acquired or made by interested people themselves for little money with a reasonable degree of accuracy, thus enabling easy measurement at virtually any place of interest. Yet, data acquisition can not solely provide insight into noise pollution. Hence, in addition to the presented noise sensor, ways to provide meaningful visualisations accessible to the general public are explored and evaluated. Those visualisation proposals focus on the means to communicate inherent uncertainty in the acquired data and to discourage overinterpretation due to a lack of experience. Further, ways to intuitively convey the notion of loudness to the lay audience are included.

Noise levels are generally not interesting for only a single measuring point but rather for a larger area, say a cluster of small houses in a block. In order to allow for measuring these larger zones, the proposed sensor can work in a network with other remote sensors reporting their measurement data back to a master node.

As mentioned earlier, ease of use is a crucial factor in the design process. This ensures that successful implementation through untrained laymen is possible and that everyday citizens gain the ability to gather information about their environment without an overly steep learning curve or unnecessarily high cost. Building on a known and established platform such as the senseBox sensor framework, makes this goal more reachable. Additionally, utilisation of this known platform also reduces additional workload for tasks like data storage or server-side software implementations.

This work is carried out in cooperation with the Media Lab Ansbach to allow for development and testing in accordance to media needs. A preliminary two week interview study has evaluated the general idea, shown potential problems and introduced several criteria that will be described later on.

Both ease of use and affordable cost are generally hard to combine with high accuracy. In order to properly document and evaluate realistic accuracy values the built sensor shall be checked in a repeatable and systematic manner with an off-the-shelf sound level meter. While high resolution measurement is not of paramount importance, measurements should be reliable and distinct enough to allow proper

and useful interpretation.

Before focusing on the technical details and design considerations, some already existing projects with similar aims are explored. As citizen empowerment plays a significant role in this work, the field of *citizen science* and the senseBox framework are introduced. After outlining the requirements found during the preliminary study, the technical details of the implementation are shown more closely. An evaluation of the stated requirements concludes the first part of the thesis.

With the data acquisition aspect covered, suitable visualisation techniques are examined next and finally evaluated in a survey. A summary including ideas for future research concludes the thesis.

2 Background

This section introduces general terms and theoretical concepts that are referenced in the text later on. These explanations give a basic understanding of the ideas involved.

First, an overview over similar projects is given. Following this, the general notion of the term *Citizen Science* along with its different characteristics is explained, advantages and disadvantages of contributions made through citizen science are discussed and the motivations driving people to participate in such projects are examined. Next, the underlying framework for this thesis – the senseBox and openSenseMap – is explained and the section is concluded with an explanation of the requirements gathered in the preliminary study.

2.1 Similar projects

It goes without saying that there already exist numerous projects and approaches that have developed some sort of sound sensing device using more or less advanced software and hardware. Following is an overview over projects with comparable goals or techniques as this thesis.

Ivan Kostoski [13] developed a sound level meter with an ESP32 microcontroller and a digital I²S microphone. Their design supports several different microphone types to choose from which offers a high degree of flexibility. However, the design being based around the ESP32 microcontroller makes it considerably more expensive. Additionally, the ESP32's higher processing power and continuous WiFi connection leads to a higher power consumption making it less suitable for a battery powered installation. The software is exclusively utilising digital audio filters for frequency weighting that may be hard to grasp for lay people wishing to modify aspects of the code. The project also does not offer any means for weather protection.

The *reedu GmbH & Co. KG* – vendor of the senseBox – have recently made available a prototype for a sound level sensor built by *DFRobot* [5]. It features a measuring range of 30 to 130 dBA with an error margin of ±1.5 dBA. These specifications are sufficient for use in citizen science projects, yet the price of 69 € is too high for larger deployments and might be deterring for interested people with a small budget. In addition, no design files are supplied and due to the high number of electronic parts required, suitability for DIY assembly is limited. Again, no case or additional weatherproofing is supplied.

Risojević et al. [25] propose a noise sensor network for indoor sound level mea-

surement based on a cheap ARM Cortex-M0 microcontroller. Apart from an analogue MEMS microphone they include additional environmental sensors. The sensor nodes send their measurements via the ZigBee protocol to the master node, a Raspberry Pi. Notable is the low power consumption that allows the node to run for around seven days which is achieved by polling the sensors only at a fixed interval of ten seconds. This results in a vast reduction in temporal resolution leading to possibly missing loud events occurring between two measurements. The overall device cost amounts to 41€.

Noriega-Linares and Ruiz [18] base a sound level measuring device around a Raspberry Pi and an external USB microphone. Their design relies on readily available parts and involves little assembly. Weather protection is achieved through an off-the-shelf enclosure. Again, the final price of 100€ makes this implementation unsuitable for large citizen-driven deployments.

Tan and Jarvis [33] describe the design of a wireless sensor node powered through energy-harvesting techniques which in theory allows the nodes to run indefinitely. Due to power constraints stemming from the energy-harvesting approach they do not apply any frequency weighting and only record noise level peaks which are hard to compare against conventional sound level measurements.

Several authors have focused on mobile sound sensing instead of permanently installed sensors. Zamora et al. [38] describe the application of consumer smartphones for accurate noise level measurement. In an experimental study they conclude that such measurements can reach the accuracy of professional measurement devices. Picaut et al. [22] propose *NoiseCapture*, a smartphone application to perform continuous acoustic measurements while the owner is moving. The application allows to describe the path taken during the measurement to qualify the data. They report that measurements regularly differ by 10 dB and that GPS inaccuracy has a great effect on data quality. Further, data sparsity is a problem where only 6.6 % of participants have provided more than 9 measurements while the majority only uploads a single measurement after registration. Rana et al. [23] attempt to alleviate typical problems arising from mobile sensing approaches through context-aware sensing that modifies measured data depending on the mobile phone's position or refuses to record measurements altogether should the circumstances be too influential. Further, they employ various interpolation algorithms to overcome intermittent data availability. While the study concludes that both data reconstruction and context-sensing are feasible, overall accuracy is too low for reliable environmental sensing.

Mobile sensing shows great promise when enough participants are available. Yet, measurements need to be taken regularly and in a repeatable fashion to allow for usable results. Permanently installed measuring stations are able to supply continuous sensor data over great lengths of time without a constant change of variables.

While these projects share goals similar to the aims of this work, they rely on custom solutions for sharing data or do not include any form of data sharing at all. The collection of accurate and valid sensor data is important but cannot be made fully accessible without an easy way to share it. Further, suitable and descriptive methods of data presentation are required. Hence, this thesis builds on an already available sensing framework along with a study on sound level data visualisation.

2.2 Citizen Science

As this work is mainly concerned with the development of a cheap measuring device for layman use, its product falls into the category of *Citizen Science*. This term describes the process of scientific measurements taken by non-scientists, amateurs and generally people that are not conducting such measurements professionally. Furthermore, projects usually are initiated or led by professional scientists [9]. Exceptions occur, where data is simply collected and compiled into datasets without a scientist's involvement. A suitable example is the openSenseMap which is part of the senseBox platform. It simply allows users to upload their sensor data without necessarily being part of an overarching project. Uploaded data can be downloaded and used for various projects. In earlier years when technology was not as present as it is nowadays, typical examples for citizen science projects would have been in the fields of zoology or astronomy due to both relying more on observation than on expensive equipment [29]. Tasks such as the monitoring of celestial bodies or counting of birds can most easily be facilitated through a large number of participants. Due to the recent advancements in technology — especially since the omnipresence of the internet and internet-enabled devices — citizen science activities have also shifted to newer fields. The subject has been growing especially large since the introduction of cheap and easy to build sensors and microcontrollers such as those sold by *Adafruit* or *Arduino*. These cheap sensors allow for a plethora of measurements like light emission, air quality, motion, temperature and more. Together with microcontrollers or single board computers running fully fledged and easy to use operating systems like the *Raspberry Pi*, collection and storage of measurements is easier than ever. Various platforms for easy uploading and cataloguing have emerged, allowing

even simpler access to the field of citizen science.

It should be mentioned that sensor deployment is not the only way of participating in citizen science. Even nowadays some tasks can not be accomplished through technology and human workforce is required. However, as this thesis aims to develop a sensing device, the term will mainly focus on those technological aspects of citizen science.

2.2.1 Definitions of citizen science

The term citizen science is only a larger category that combines several definitions of the field, depending on the actual task performed. Since there are many different definitions as well as redefinitions, it is virtually impossible to name every single one. Hence, the most popular definitions are explained below.

Volunteer computing Volunteer computing defines one aspect of citizen science where participants provide spare computing resources on their personal devices [10]. Arguably the most famous example is *Berkeley Open Infrastructure for Network Computing* (BOINC)[1] that provides a general platform for distributed computing that has its origins in the *SETI@Home* [2] project, aiming to find signs of extraterrestrial intelligence by analysing radio signals. Unless participants are running large computational installations, involvement along with effort is generally low and passive compared to other forms of citizen science.

Volunteer thinking In contrast to the passive participation mentioned above, this sub-discipline requires more cognitive and active involvement from their members. Participants – usually registered on a website – are presented collected data and are asked to interpret that data by following a research protocol. Examples include identification of galaxies in photographies of space, annotation or transcription of historic diaries or the folding of proteins, amongst others [10].

Participatory sensing This aspect of citizen science is the one most fitting to the scope of this thesis. It involves participants performing sensory measurements of their environment. Some sources exclusively mention participation through the use of smartphones [10]. These devices have become ubiquitous and relatively cheap, leading to approximately 80% of the earth's population owning at least one smartphone [37]. In their simplest form, mobile devices may be used along with an application as a data entry device for basic tasks. In more advanced settings, the

inbuilt sensors themselves become part of the measuring device. Almost all smartphones nowadays ship with accelerometers, light sensors, good quality microphones, GPS receivers and high resolution digital cameras that can be employed for various measurement tasks. In addition to smartphone usage for participatory sensing applications, discrete sensing devices are commonly applied to measurement tasks [21]. This is especially true since the growth of the so called *maker culture* that emphasises open source hardware, open information and maker spaces to enable everyday people to build cheap and accessible sensors on their own [19].

2.2.2 Motivations for participation in citizen science projects

As citizen science is a voluntary activity, naturally no monetary compensation is involved. People generally spend time and money due to other motivating factors. This question about the driving factors of project participants has spurred several scientific studies on the topic [10]. While there may be numerous different incentives for people to participate in a project, some common findings about participant's motivations are reported below.

- Interest in the research topic [26]
- Learning new information or skills [20]
- Identification with the project's goals or values [26]
- Experiencing a sense of team spirit and helping others [26]
- Calling attention to environmental problems [20]

As a final remark it is fair to say that motivations of layman or citizen scientists are similar to those of professional scientists.

2.2.3 Advantages and disadvantages of citizen science contributions

The idea that people without formal scientific education are handling scientific endeavours may occur problematic at first and there are a few drawbacks to this idea. However, depending on the research goal and project design, contributions through citizen science can greatly benefit the scientific community.

Probably the advantage mentioned most often is the increased communication between scientists and the general public which in turn allows for a greater understanding of scientific or complex topics through laymen and furthers their curiosity

within the subject at hand. A study conducted by Crall et al. [4] that explored learning effects of citizen science participation suggests that attendees' engagement has a positive effect on science literacy and their attitude towards environmental protection. A survey carried out by Lewandowski et al. [14] asked participants of various butterfly conservation citizen science projects and concluded that 95% of the people asked have increased their participation action since they joined the programme. Another important factor for data acquisition is cost. Gardiner et al. [8] examined data quality of verified and unverified data in citizen science projects and found that the cost-effectiveness of data verified by a professional scientist can outweigh possible inaccuracies. This is due to a larger amount of samples collected through volunteers which may compensate erroneous entries. In terms of accuracy, datasets compiled through citizen science projects can yield similarly high quality results as those compiled by professional scientists [12]. Gardiner et al. list several measures for project design to improve overall data quality such as proper training and testing of volunteers, expert validation or accounting for random error or systematic bias.

While voluntary data seems like a good fit alongside traditional science, there naturally are some drawbacks. For instance, Thelen et al. [34] suggest that those tasks that are either very complex or require tedious or repetitive work are not well suited for volunteers. Another potential problem is the introduction of bias through participants. Such bias may occur when participants unwillingly measure something not as neutral as required for scientific purposes. One example of biased data is the experiment constructed by Farmer et al. [6]. A simulated online survey asked participants to identify bird calls. The results show that participants who identified themselves as experts were more likely to identify calls incorrectly.

At the very least, citizen science projects may be able to pave the way for future more in-depth research, even with imperfect data and results by pointing out topics worthy to investigate closer.

2.3 senseBox and the openSenseMap

The senseBox is a microcontroller based toolkit for environmental sensing and sharing of measurement data. It is distributed by *Reedu GmbH & Co. KG* and originated from a project at the University of Münster. The PCB is based around the ATSAMD21 microcontroller series but offers an abstraction layer by being programmable through the Arduino and Blockly framework. It is aimed at people with

an interest in environmental sensing and Citizen Science as well as school projects. Various sensor modules for phenomena such as temperature, humidity or air quality are purchasable off-the-shelf without any assembly required, thus fitting the senseBox mantra of "Plug-in-and-Go". This motto refers to the fact that every sensor sold, is tested for use with the board and has a library available for easy interfacing. Additionally, expansion cards, so called "Bees" plug into two headers on the main PCB to add LoRa capabilities or enable internet connection via ethernet or WiFi.

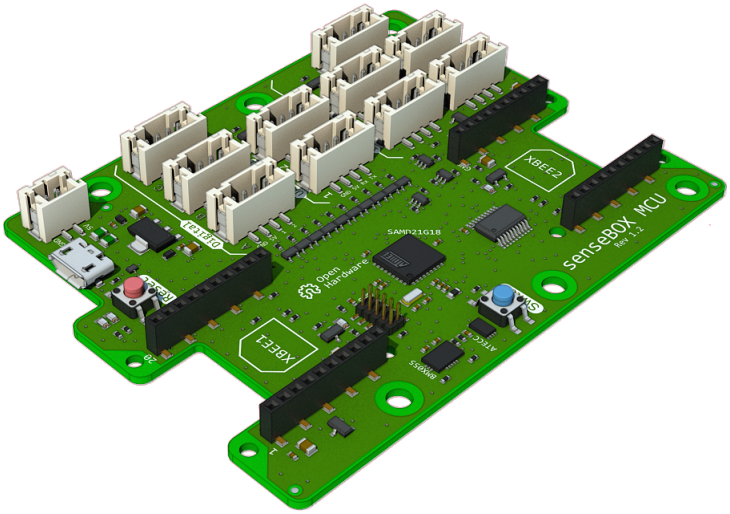


Figure 1: The bare senseBox PCB / CC BY-SA 4.0 [24]

For easy data storage and sharing, the senseBox uploads data to the openSenseMap (OSeM), another project of the University of Münster. It offers two main services: The API for uploading, requesting or deleting sensor data and the visualisation that consists of a map with registered senseBoxes shown at their location. The website further offers interpolation within a chosen bounding box and for one selected measurement phenomenon. By clicking on a senseBox, the installed sensors can be viewed along with their last measurement and a simple graph showing the recorded values over the last few hours. The OSeM is not exclusive to senseBoxes but is open for API requests coming from any registered device. It is especially popular with participants of the *luftdaten.info* project[20, p. 27]. As of March 2021 7777 devices are registered on the map.

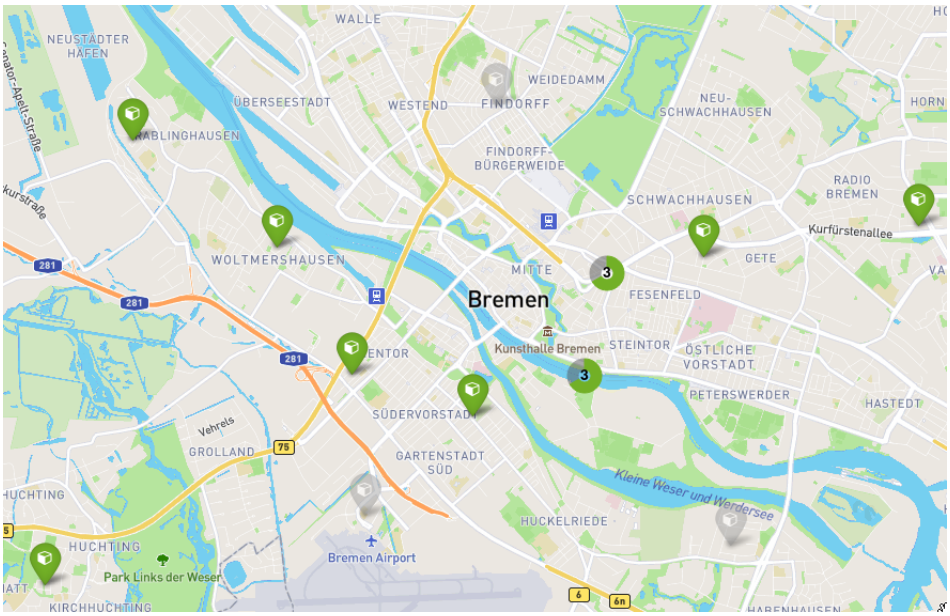


Figure 2: View of the openSenseMap

2.4 Preliminary study

For establishing requirements such a sensor should achieve, a qualitative survey with six people working in the field of journalism has been conducted beforehand. Each person was interviewed for approximately 30 minutes. Questions regarding access to and acquisition of statistical or measurement data, what they consider as *low-cost* and how much effort they deem reasonable when installing or assembling the devices have been asked.

Participants answered that easy access to measurement data is difficult. This is in part due to data encoding in proprietary or unsuitable formats such as PDF documents which are a challenge for machine processing. Additionally, data availability is restricted since official or governmental authorities do not always publish acquired data by default and gaining access is difficult and time consuming.

Respondents expect several requirements from a DIY low-cost sensor and the resulting data. First, collected data should be easily quantifiable. They wish for properly encoded data such as time series contained within accessible and versatile formats including CSV or JSON. In addition, they argued that real-time or near real-time access to the data would allow for easier storytelling and interaction with the phenomena observed.

The hardware should be easy to build without prior knowledge in the field of electronic engineering. For one, this enables individuals or even the journalists

themselves to deploy own sensors but more importantly makes possible hosting larger events with their readership where participants can build their own device in a subject they are passionate about. Naturally, this requires the means for building such a sensor to be available publicly which is another request that was voiced.

When asked about their notion of low-cost, participants responded within a price range of 10 to 20€. They argued that a higher price would be both deterring for interested people and prone to reducing the target audience as higher cost impedes accessibility for some demographic groups such as children.

While participants expressed a desire for high data resolution they added that professional accuracy is not necessarily required. Rather, data should be easily accessible for the public and be capable of arousing interest in the observed topic.

3 System description

This section goes into detail about the developed system and describes both software and hardware aspects. For clarity, the explanations are divided between the implementation on side of the microcontroller and on side of the senseBox.

3.1 Protocol

Compared to simple sensors that are directly connected to the I²C host, the described network consisting of several sensor nodes requires a more sophisticated means of communication. For this reason a simple time-based communication protocol is devised. It ensures that nodes only transmit when they are supposed to.

The protocol is based on a sync message that is transmitted by the master node at specified intervals. The sync message ensures that every slave node is always synchronised with the master node. This is necessary as the master node can only receive one transmission at a time. If two or more signals arrive simultaneously, transmitted data may be indecipherable or contain false values. While the internal clock of the microcontroller only suffers from an average drift of 0.05% [31, p. 75], even small differences can accumulate to a large deviation over time, eventually leading to overlapping transmissions and thus unusable data.

The protocol defines three main parameters:

1. *Tick count*: The number of measurements taken within one complete iteration. The default is set at 30 ticks per iteration.

2. *Tick duration*: The length of time each tick runs for. The default is set at 2 seconds per tick.

3. *Tx offset*: A factor (in ticks) that determines by much a packet transmission is delayed to avoid overlap at the master node. The default is set at 1 tick.

Therefore, the total runtime of the values described above amounts to 60 seconds per iteration.

At the start, the senseBox sends a 4-byte message via I²C to the master node containing the tick count, tick duration and tx offset. The master node is in idle up until the reception of the message. It configures the parameters according to the received values and sends a sync message containing these same parameters via LoRa for the slave nodes to receive. Immediately after sending the message, the master switches the LoRa module to reception mode as no transmission is required

until the next iteration and starts to measure the sound level for every tick. The tick duration is chosen such that one sound level measurement can comfortably fit into one tick with approximately 600 milliseconds to spare. The duration depends on the microcontroller's system clock (SYCLK) as well as LoRa parameters such as the channel bandwidth and the spreading factor. As the master's duty for this tick is completed, it listens for transmissions for the remaining time. At 0 microseconds of tick duration left, the next tick begins.

Similar to the master node waiting for the I²C message, the slave nodes are idling until the sync message is received. As the sync message's length is known to be 4 byte long, any message longer than that is ignored. This is an important check as otherwise messages exceeding the maximum expected packet length that have been sent from unrelated nodes can cause the reception buffer to overflow and stop the execution. Upon reception, the microcontroller processes the parameters and begins the measurement until the tick count is reached. Two ticks before the end of the iteration, the LoRa module switches back to reception mode and awaits the sync message that got initiated by the senseBox and repeated by the master node. Initially, reception mode was enabled immediately after the remote sensor has transmitted its packets. However, since the LoRa module consumes the most power while in reception mode, disabling it when it is not required substantially improves battery life. Reception of the sync message starts the measurement iteration anew but also sets a flag to indicate that one whole packet of measurements is ready for transmission at the next opportunity. The tx offset decides when the packet will be sent. Every node is aware of its device ID and can – by multiplying that ID with the offset value – know the number of ticks to wait before sending the measurements off to the master. For a device ID of 2 and a tx offset of 1, the slave node will send its packet at tick count 2, 2 seconds after node 1 and 2 seconds before node 3. To allow for delayed sending without overwriting previously stored sensor values, reception of the sync message also switches between two buffers such that one buffer is the buffer currently written to while the second buffer keeps its values until they are sent off.

Simultaneously, the master node has reached the tick count and waits until the next sync message is received via I²C to start the iteration anew. The delayed packets arriving between ticks 1 and 5 are received and copied into a separate buffer for transmission via I²C later on.

For high precision, the implementation utilises one of the microcontroller's internal timers and time units in microseconds.

The fact that the protocol parameters are contained in the sync messages has the added benefit of changing parameters on the fly at the start of every iteration. This means that those parameters can be adjusted from the senseBox alone which makes deployment of remote sensors considerably easier. If the slave nodes receive a sync message with protocol parameters differing from the previous ones, it discards its recordings as the new packet dimensions might be incompatible with the old setup.

Due to the delayed sending principle, values can the soonest be received via I²C when the last node has sent its package from one iteration earlier. This results in a delay of 71 seconds for the default values mentioned above including a tolerance of 1 second until the first packet can be received. The calculation is given in Equation 1.

$$\begin{aligned} Rx\ delay = &(tick\ count \cdot tick\ duration) + \\ &(5 \cdot Tx\ offset \cdot tick\ duration) + tolerance \end{aligned} \tag{1}$$

3.2 Microcontroller

The software implementation on side of the STM32 microcontroller uses the provided hardware abstraction layer (HAL) and C++ instead of directly programming the device in low level C. This mainly improves code readability and allows for easier debugging.

At the core of the algorithm lies a simple counter performing certain checks at predefined values. This explicit approach makes the code more transparent to the user and eases debugging, compared to an interrupt-based implementation. On device start up, necessary peripherals are initialized and the clock values are configured. By default, the microcontroller uses a *SYSCLK* of 16 MHz. Further, a general purpose timer is initialized for timekeeping later in the program. The SX1278 LoRa transceiver module is initialized with a channel bandwidth of 250 kHz and 868 MHz, the allowed frequency for use in the European region. The spreading factor SF8 is used. By choosing a higher spreading factor, both range and data integrity are increased, however at the cost of a slower transmission speed. For every spreading factor increment, airtime doubles for packages of the same size [28, p. 8]. This trade-off has to be considered when defining the protocol parameters.

Instead of polling for audio data from the microphone which would leave the microcontroller unusable for that time, I²S data is directly fed into a predefined buffer via the direct memory access controller (DMAC) in the background. This

way, no additional processing time is occupied and audio data can be grabbed from the buffer at any given time. The audio data from the microphone is encoded in a 24-bit integer on a 32-bit frame that gets split into two 16-bit integers by the I²S controller. Of those 24 bits only 18 are usable as per data sheet [11]. To save on RAM, the last two bits are omitted, thus resulting in a single 16-bit integer representation.

Next, the Analogue to Digital Converter (ADC) samples the *Device ID* voltage divider to obtain the device ID of the sensor the code is running on. This works by placing resistors of different values onto the soldering pads that reduce the 3.3 volt base voltage by a defined amount. Based on the ADC reading a different ID is assigned. Currently, five unique IDs with generous safety margins are configured but more could easily be added by reducing the margin for each value range. The configurations are shown in Figure 3. ID 0 is considered to be the master node while the remaining four IDs represent the slaves.

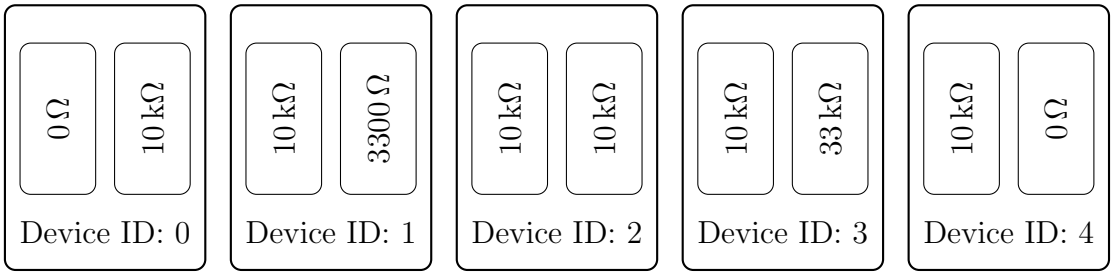


Figure 3: Voltage divider configurations and their resulting device IDs

To prevent the LoRa modem from interpreting other devices' messages as our own, some form of message identification is necessary. Three predefined bytes that are known among the sensors within the network are prepended at the start of every message. If those bytes don't match during reception, the message is disregarded completely. This also allows to have multiple networks operating near each other without cross-talk.

3.2.1 Battery level measurement

Slave devices should be aware of their battery level and inform the master if it reaches a critical threshold. Again, a voltage divider is used to lower the battery voltage to a safe value for the ADC. The reading is taken directly from the battery pack. The battery voltage is measured in the middle of one full protocol cycle to prevent false data readings due to the voltage dropping momentarily while sending data.

Measurements are averaged over a defined amount of cycles; 10 by default. As false readings could shorten the runtime considerably, the ADC is calibrated regularly against the microcontrollers provided internal voltage reference. The actual battery voltage V_{BAT} is calculated according to Equation 2.

$$V_{BAT} = \left(ADC_{BAT} + CAL) \cdot \frac{1.212V}{ADC_{VREF}} \right) \cdot 2 \tag{2}$$

ADC_{BAT} refers to the raw 12-bit ADC value of the battery voltage. CAL is the ADC calibration offset constant provided by *STMicroelectronics* during manufacture. The internal voltage reference 1.212 Volts [31, p. 59] is normalized with the ADC reading of the reference ADC_{VREF}. Since the voltage divider circuit divides the input current by half, doubling the reading results in the actual battery voltage. It is important to note, that a battery voltage measurement is no definitive metric for determining battery capacity. However, battery voltage can be used as an indicator for depleting capacity. Additionally, the important voltage threshold is defined by the voltage regulator's dropout voltage and both the microcontroller's and LoRa module's minimum operating voltage.

Once the ADC reading indicates a voltage lower than the defined threshold, the last two bytes of the LoRa packet – normally set to LOW – are flipped to *HIGH*. Upon reception, the senseBox reacts by informing the user about the impending sensor outage.

For receiving and transmitting data via I²C, the sensor uses the integrated I²C *AddrCallback* interrupt that executes a function whenever a device accesses the bus. Again, this makes the device more efficient than actively listening for incoming I²C connections. The microcontroller waits until the first sync message is received over I²C and transmits the protocol parameters via LoRa to the other nodes. Until the counter reaches the tick count setting received from the senseBox, it measures the sound pressure level in every tick, details of which are given below.

3.2.2 Sound level measurement

Each sensor node uses a Knowles SPH0645LM4H MEMS microphone on a breakout board for easier assembly. It provides data via the digital I²S interface. The measurement range lies between 29 and 120 dB, well within the expected noise range. The frequency range lies between 50 and 11000 Hz. The exact frequency response curve is shown in Figure 4. As the response curve is not perfectly flat and might

thus affect the accuracy of sound level measurements, frequency correction values are applied in software.

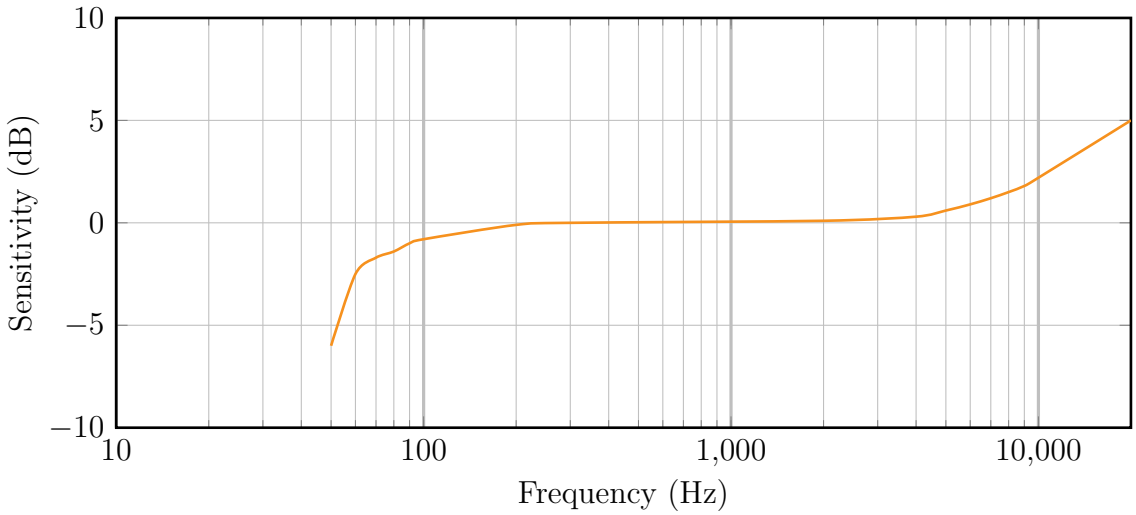


Figure 4: Knowles SPH0645LM4H frequency response

The measurement process starts by copying audio data from the I²S DMA buffer to an array of a predefined length. A Fast Fourier Transform (FFT) library[1] transforms the audio data from the time domain to the frequency domain to apply A-weighting later on. The parameters of the FFT determine both execution time and resulting resolution. The sample size is set at 1024 samples. The Nyquist theorem dictates that the sampling frequency needs to be at least double the maximum frequency measured. As the microphone has a maximum frequency range of 11 kHz, a sampling frequency of 22 kHz is chosen. A Hann window is applied to the transform. The FFT's result is given in bins where each bin represents a fraction of the whole spectrum. For a sampling frequency of 22 kHz and 1024 bins, the width of a single bin is:

$$\frac{22,000\ Hz}{1024} \;=\; 21.48\ Hz \tag{3}$$

Only half of the resulting 1024 bins are usable since their contents are mirrored around the centre bin. Thus, 512 frequency bins ranging from 0 Hz to 11,000 Hz – spaced 21.48 Hz apart – are the result. The first two bins represent a frequency below the microphones frequency range and are discarded. Every bin contains a

[1] `https://github.com/kosme/arduinoFFT`

magnitude value that represents the proportion of its frequency in the original signal. Equation 4 calculates the root mean square (RMS) of the magnitudes. Due to the use of the Hann window an energy loss correction factor of 1.63 needs to applied to the result [27].

Further, every magnitude is weighted by a factor of the microphone frequency response correction and the A-weighting frequency weight. These values are calculated externally beforehand since the bin frequencies are constant. A modified version of the script in [15] is used for precalculate those correction values. They are stored as an array in the microcontroller's flash memory.

$$RMS = 1.63 \cdot \sqrt{2.0 \cdot \frac{\sum_{i=3}^{512} (bin_i)^2 \cdot corr_i}{sample\ size^2}} \tag{4}$$

Once the RMS is calculated, the A-weighted sound pressure level can be computed as depicted in Equation 5, where *OFFSET* is a constant to accommodate for losses through the case and wind screen. *REF DB* refers to the calibration value given in the microphone's data sheet and *REF AMPL* is the reference amplitude of the microphone.

$$dbA = OFFSET + REF\ DB + 20 \cdot log_{10}\left(\frac{RMS}{REF\ AMPL}\right) \tag{5}$$

Finally, the resulting level in dBA is stored in an array in fixed point representation. Choosing a fixed point instead of floating point representation reduces the number of bytes from four to two by using a 16-bit integer instead of a 32-bit float while still providing two decimal places.

For easier representation and comparison, the *equivalent sound level* [3] can be calculated as an average of the measurements within a given time frame as shown in Equation 6.

$$LA_{eq} = 10 \cdot \log_{10}\left(\frac{1}{n}\sum_{i=1}^{n} 10^{LA_i/10}\right) \tag{6}$$

3.2.3 Power consumption

Low power consumption is no hard requirement but greatly increases usability. For use with batteries, a low power usage results in fewer battery changes and thus less

user interference. When continuous operation without any intervention is important, adding a photovoltaic cell to each sensor is a viable option. Lower power usage results in a smaller cell footprint, making the sensor more versatile. When coupled with a solar battery charging circuit, a 1 Watt photovoltaic cell – slightly larger than the sensor case – is easily capable of supplying enough power for steady operation, given regular insolation. The biggest factor for power consumption is LoRa reception mode that keeps the module powered at all times. Having a high tick count results in fewer time spent in reception mode and thus reduces battery drain as is evident from Table 1. Similarly, lower tick duration requires more power as the microcontroller's clock has to be increased for the measurement calculation to fit into the shorter time frame.

The battery time values are calculated with a conservative battery capacity of 2640 mWh per cell and 7920 mWh for three cells in series. This value is based on a *Panasonic Eneloop Pro NiMH* rechargeable battery with a nominal voltage of 1.2 V and a minimum capacity of 2500 mAh. To accommodate for lower capacity cells, a decreased capacity of 2200 mAh is assumed.

Ticks	Tick dur.	Power consumption	Batt. life (hours)	Batt. life (days)
10	2	27.95 mW	283.36 hours	11.81 days
30	2	24.72 mW	320.39 hours	13.35 days
60	2	23.53 mW	336.59 hours	14.02 days
100	2	20.44 mW	387.48 hours	16.14 days
10	1	60.12 mW	131.74 hours	5.49 days
30	1	56.22 mW	140.88 hours	5.87 days
60	1	54.89 mW	144.29 hours	6.01 days
100	1	51.89 mW	152.63 hours	6.36 days

Table 1: Power consumption for different tick counts

3.3 senseBox

The three main tasks for the senseBox implementation are

- Supplying the sync message and keeping track of time
- Reading the data over I^2C and converting it into sensible values
- Uploading the data via the openSenseMap API

On device power-up the senseBox attempts to connect to the WiFi network specified by the user and retries the connection, should it fail. For accurate and reliable timekeeping, it configures and connects to a randomly selected network time protocol (NTP) server and ensures alignment with the server's time every 5 minutes. In an earlier software version that made use of the *NTPClient* library a server connection fault occurred intermittently leading to timing misalignment between the senseBox and the master node. The library has since been replaced by the *Arduino Time Library* that shows no such behaviour. After successful connection to the NTP server, the senseBox starts the protocol cycle with the parameters specified by the user by sending the sync message to the microcontroller via I^2C. Sensible defaults values are already predefined but can be changed if the actual scenario requires a different setup. Should the user-defined values violate the protocol's limits, the user is informed via the serial monitor.

The main loop regularly checks if sending the next sync message is due or if new data is available to read and acts accordingly. This manual check can later be moved to a time-based interrupt to reduce the amount of user-facing code. Measurement data is read into a *NoiseSensorNode* object via I^2C. Due to a hardware limitation with the senseBox's microcontroller, I^2C packages over 256 bytes size cannot be received and the device locks up. This limitation would result in an artificial tick count limit of only 10. Hence, each reading operation is split into five separate packages, one for each device, increasing the maximum possible tick count to 120. The node object also records the time stamp of the first measurement. The time information for consecutive measurements can be easily calculated later on. The reading step also checks the last two bytes of the received package that contain the sensor's battery state indicator. A message alerting the user of the low battery level is triggered if those bytes are set in the received packet.

After successful reception of all packets, a JSON array in the format specified by the senseBox API is assembled. The measurement values in their fixed point representation are converted back to regular floating point numbers and the time stamp for each measurement is calculated by incrementing the time of the first measurement by the tick duration for every recorded value.

```
[
  {
    "sensor": "6039d2cb754322003c53cd23",
    "value": "48.0",
    "createdAt": "2021-03-05T15:03:59Z"
  },
  {
    "sensor": "6039d2cb754322003c53cd23",
    "value": "48.3",
    "createdAt": "2021-03-05T15:04:01Z"
  }
]
```

Listing 1: Exemplary JSON array structure

To prevent a sudden lock-up of the senseBox when submitting large POST requests to the API, the request is split into chunks of 1000 characters at a time. This problem stems from an undocumented maximum length constraint in the senseBox's WiFi library. It is worth noting that the API is rate-limited to 15 requests per minute per IP.

If verbose mode is enabled, the API's answer is printed to the serial monitor.

It should be noted that the library described in this section has also been adapted to work with the ESP32 microcontroller. While it lacks the pluggable sensors of the senseBox, it also offers Internet connectivity, yet for a lower price.

3.4 PCB

Early circuit tests have mostly been carried out on prototyping breadboards that allow quick rearrangement of components. For a final product that is supposed to work for a long time and is easy to use and easy to assemble, a printed circuit board (PCB) is better suited. It allows easy and repeatable manufacture by interested individuals and defines fixed dimensions for use with the case. Several decisions have influenced the final design. To accommodate assembly through unexperienced users, the initial design made heavy use of through-hole components. However, for the PCB's relatively small footprint this led to too little space between components to allow for easy assembly. Additionally, non-SMD microcontrollers are difficult to source and generally more expensive than SMD types. Hence, mostly SMD components are used for wider availability and a less dense design. While considerably

smaller than through-hole parts, SMD components can still be soldered by hand. Larger SMD parts (1206) have been chosen for resistors and capacitors to make assembly easier.

The final dimensions are 59x41.6x1.6 mm and have been deliberately chosen to allow the PCB to sit on top of a standard 3xAA battery pack.

A rendering of the finished board is shown in Figure 5. A picture of the fully assembled prototype is shown in Figure 6. The design files are available in the online repository mentioned in Section 4.3.

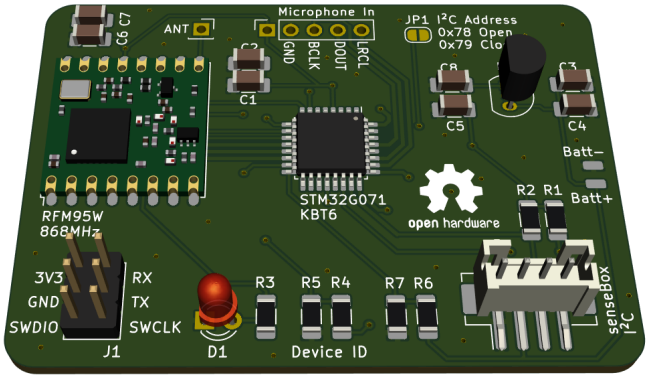


Figure 5: 3D Rendering of the fully populated PCB

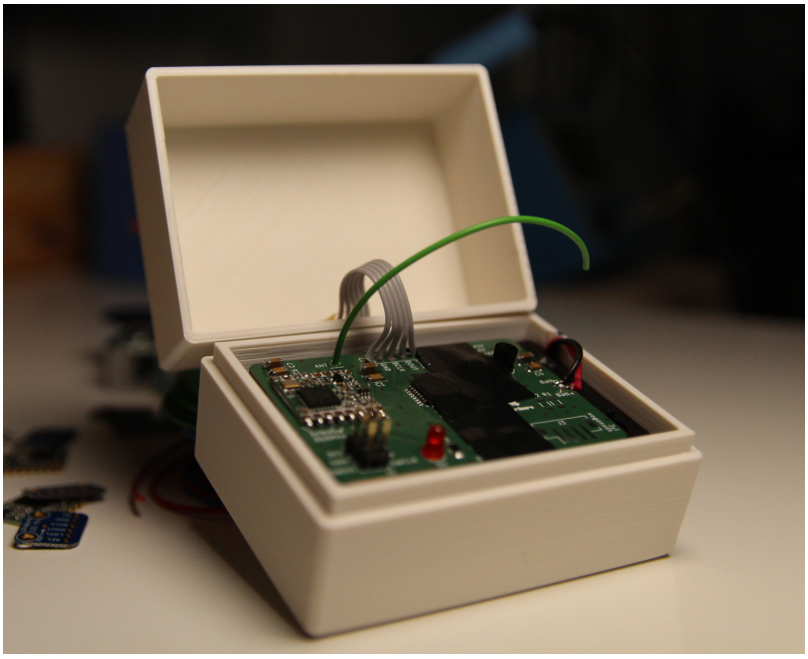


Figure 6: Fully assembled prototype

3.5 Case design

For general protection from the elements along with a more pleasant look, a simple enclosure was designed. It is supposed to guard the PCB and the microphone from direct water ingress as well as shielding against UV light from the sun.

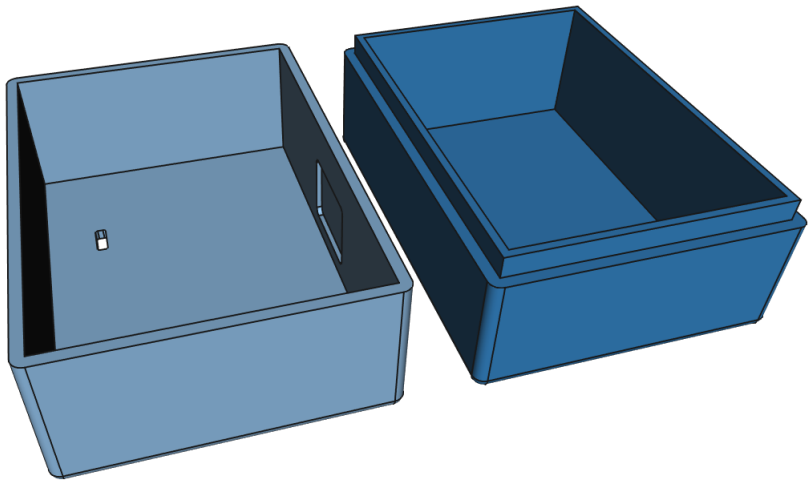


Figure 7: 3D Rendering of both master case halves

The case consists of two parts – a top (light blue) and a bottom (dark blue) part – depicted in Figure 7. The bottom part represents the main compartment that houses the battery pack and PCB. The height is chosen so that all parts sit just below the housing to minimize exposure. The case design for the master and slave nodes is identical except for a small cutout in the top of the master case that acts as a feedthrough for the I^2C connector cable. It can be glued shut after the cable has been inserted to prevent water ingress.

The main compartment's size is chosen according to the dimensions of the battery pack and provides a snug fit with 0.5 mm of tolerance on either side. This ensures that the contents are kept in place when gently pushed in but also allows to easily retrieve the enclosed parts for swapping the batteries or general maintenance. The bottom part also features a small tongue extending upwards to facilitate both the interlocking of the two halves as well as providing a water barrier. With earlier designs having the tongue on the upper case half, moisture running down the case would be able to creep along the extension and directly into the main compartment.

The upper case half is designed to accommodate the microphone breakout board. A small pocket in the front wall reduces the thickness by 1 mm. The dimensions

for the pocket are slightly oversized. This ensures that there is enough room for the glue used to hold the microphone PCB in place. A small hole is drilled in the centre of the pocket for the microphone port. For protection against water ingress into the microphone, an acoustically transparent acoustic vent[2] can be placed over the port. To alleviate wind noise, a small wind muff designed for video cameras[3] is placed on top of the microphone port. The case's walls have been designed thick enough to keep them from acting as a membrane possibly enhancing vibration and thus skewing results.

The easiest option to produce the case is by 3D printing. Recent 3D printers can operate with a plethora of different materials, yet some materials are better suited for this application than others. The prototype case is printed with a fused deposition modelling (FDM) printer with polylactic acid (PLA) filament. PLA shows a high resistance against UV light and was further chosen because it is easiest to work with and was readily available. However, having a relatively low melting point of approximately 200° C the print may start to warp or lose its shape when subjected to temperatures over 65° C such as in direct sunlight during summer time. Such low temperature stability is sufficient for a proof-of-concept case, yet for a final product a more resilient filament such as ABS or PTGE might be better suited.

The case's colour is no critical requirement, however very dark as well as transparent colours should be avoided to reduce excessive heating of the device in direct sunlight. The exact dimensions and the design files of the case are provided in the online repository mentioned in Section 4.3.

3.6 Cost

The overall cost of one prototype sensor is reported in Table 2. The cost in the table is for a combination between the master and the slave configuration. As the master requires no battery holder, the cost reduces by 2.25€. Slaves require no JST socket or pull-up resistors, reducing the cost by 0.36€. The total of 24.15€ is above the threshold that was determined during the preliminary study. However, cost can be further reduced by considering bulk prices and lower PCB manufacturing services when fast delivery is not required. Cheaper alternatives to the Adafruit MEMS breakout board are available, although with longer delivery times. Additionally, cheaper LoRa modules can be used instead of the RFM95W which has been se-

[2] `https://www.gore.de/node/17776`

[3] `https://rycote.com/microphone-windshield-shock-mount/micro-windjammers`

lected merely due to familiarity with the device. If a bare bones device is required, commodities such as the battery holder, LED, Pin header and JST plug can be omitted. Due to the simplistic design, etching the PCB at home is easily achievable.

Part	Cost
STM32G071KBT6	2.56€
HopeRF RFM95W	+ 6.12€
3xAA Battery Holder	+ 2.25€
Knowles SPH0645LM4H breakout board	+ 5.89€
Resistors 1206 (x6)	+ 0.21€
Capacitors 1206 (x8)	+ 0.20€
LP2950 Voltage Regulator	+ 0.49€
LED 3mm	+ 0.14€
2x3 Pin Header	+ 0.03€
JST PH 4-Pin socket	+ 0.29€
Subtotal: Electronic parts	18.18€
PCB manufacturing	+ 3.47€
3D printing filament	+ 1.00€
Wind muff	+ 1.50€
Subtotal: Other	5.97€
Total	24.15€

Table 2: Cost overview for the prototype

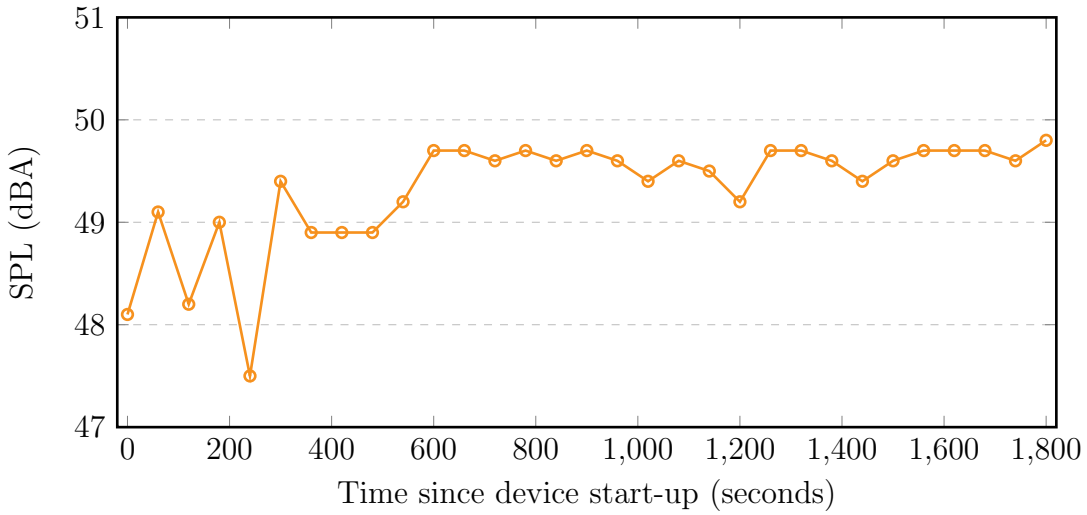


Figure 8: Sound level meter measurements over a period of 30 minutes for a white noise signal at 50 dBA

4 Evaluation

With the details of the system described, the following section focuses on the evaluation of the requirements elicited in Section 2.4.

4.1 Measurement accuracy

4.1.1 Sound level meter verification

During initial testing with the provided sound level meter (SLM) the suspicion arose that measurements are less consistent within the first few minutes of starting the device. A simple setup – identical to that mentioned in Section 4.1.2 – was arranged to verify this hypothesis. White noise was played for 15 seconds and values were recorded and averaged. Measurements were started every 60 seconds, with the test tone playing for 15 seconds before letting the device settle again for 45 seconds of silence.

Figure 8 shows the measured values in decibel for the given time, beginning with the first measurement right after start-up and ending the test after 30 minutes. It is evident that during the first five minutes after the device is first started, fluctuations of approximately $2dB$ occur. The values start to settle at around 10 minutes with a deviation of less than $1dB$, which is within the reported accuracy specifications of the meter.

4.1.2 Calibration against the sound level meter

With the goal of real-world use in mind, it is of paramount importance the device's reported values are as accurate as technically possible. To verify this, the SLM was used as a reference. While this SLM is not designed for high precision measurements it has been calibrated to $\pm 1dB$ of accuracy.

To ensure a sane calibration environment and repeatable measurements, several precautions have been taken. The sound level meter and the MEMS microphone were arranged between two loud speakers, one metre away from each. Both devices were positioned at the same height and with their microphone port openings facing the same way. During the measurement the room was completely silent with a sound pressure level between 30 to $35dB$. If the silence was interrupted by louder sounds from outside the room, the measurement was immediately voided and repeated. Further, cautions have been taken to eliminate echoes and sound reflections in the room and the path between the speakers and the devices was not obstructed. The room temperature was held constant at 25° Celsius as per the microphone datasheet's specifications [11]. Since a negative bias has been found during initial verification as mentioned in Section 4.1.1 the meter was left running for 10 minutes before starting the measurement.

Measurements were taken in increments of $5dB$. The starting value was chosen to be $35dB$ as values below this threshold are hard to achieve and are thus not stable enough for the measurement environment. Calibration was carried out to a maximum value of $80dB$. For every measurement white noise was played until the sound level meter stabilized on the chosen increment. White noise is used as it carries equal intensity at different frequencies. This represents real-world scenarios more realistically than a single frequency sine wave.

When the measurement is started, the noise sensor counts down from five before values are stored. This serves two purposes:

1. For approximately two seconds after powering on the device, the microphone output shows a large bias that slowly declines to a more stable value.

2. The countdown ensures that after starting the measurement there is enough time to move away from the setup so as not to skew the result by obscuring the path of the sound or by introducing sound reflections through moving or standing near the measuring devices.

Once those five seconds have passed, measurements are taken every second for 15 seconds and are averaged over this count. The results are shown in Figure 9.

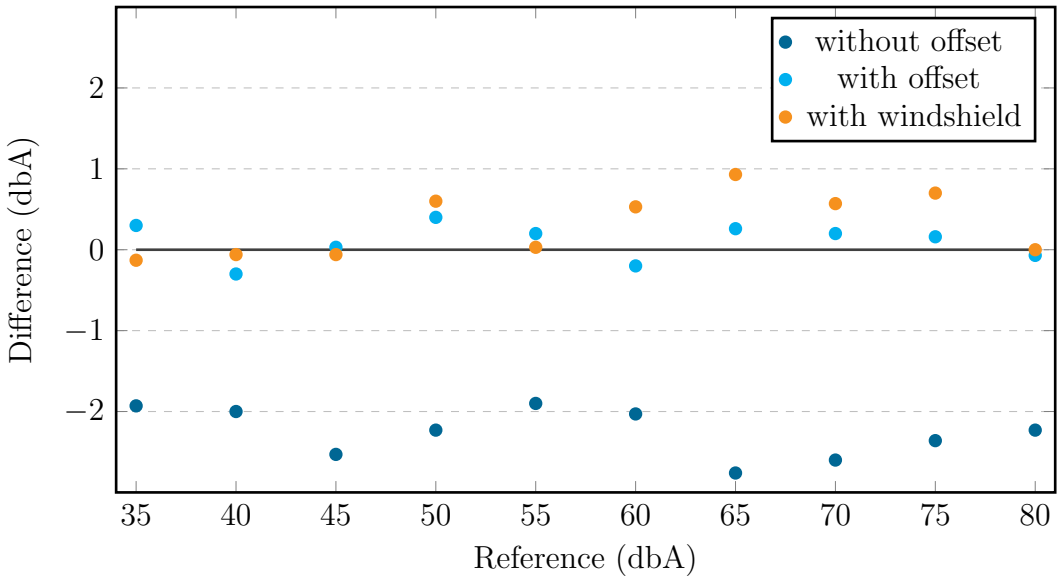


Figure 9: Deviation from the reference value for three iterations

The reference sound level in dBA is given on the x-axis. The y-axis shows the measurement difference between the reference and the sensor. The graph shows the measurements of three repeated runs. The first run (dark blue) has been conducted without any software corrections. It is evident that the line shows a steady -2 dBA deviation between the reported value and the reference. Hence, a +2 dBA offset was defined in software. The light blue line represents the second run with the corrected output. To ensure that the measurement is within reasonable accuracy with the windshield installed, a third measurement iteration was conducted (light blue), now with the windshield over the microphone port. No significant deviation can be observed between iterations two and three.

4.1.3 Cross-device validation

Every sensor in the proposed network should be reliable enough to produce credible measurements. With one random device verified, the remaining sensor nodes need to be tested for variance within the group. The same measurement setup was used for testing a total of 13 devices. Each sensor was tested separately since positions would be too different when verifying all devices at once. Results are shown in Figure 10. For values below 40 dBA a larger measurement difference of 2 dBA is evident. This is owed to the ambient sound in the room. Accuracy stabilises in the range over 40 dBA and the average deviation in this region is 1.3 dBA.

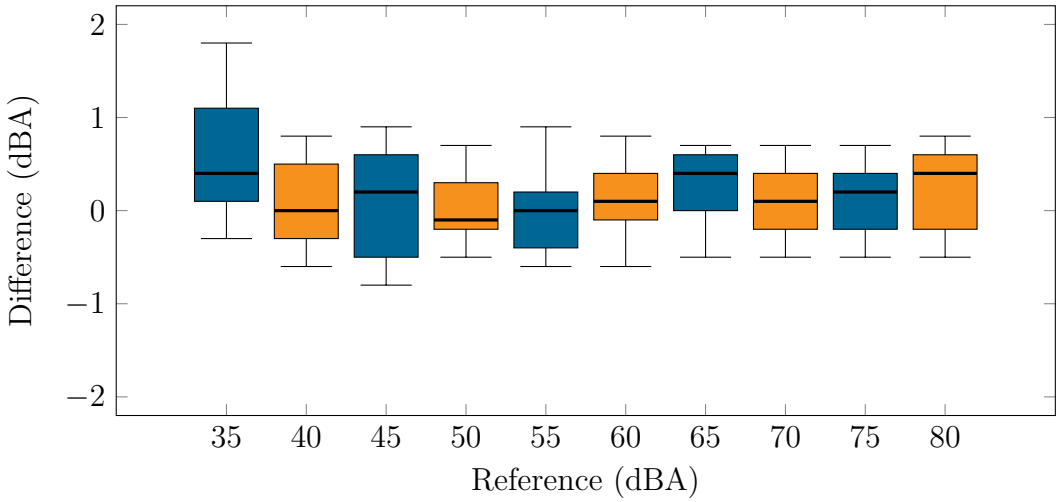


Figure 10: Boxplot outlining the characteristics of the tested devices

4.2 Low cost

The actual definition of the notion of "low-cost" is based on several aspects and cannot be answered definitively. In the preliminary study, participants named a final price range of 10 to 20€ for each sensor as acceptable. Table 2 shows that the final cost exceeds this threshold by more than 4€. As the designed device is no final product but merely a prototype, overall cost can be expected to be lower, especially when fast fabrication or delivery is no concern. For large organised sensor building events bulk pricing reduces the cost somewhat but is no option for citizens on their own. Several options for price reduction are also mentioned in Section 3.6. Additionally, final cost can be scaled to some extent as the number of deployed sensors can be chosen freely. Compared with other readily available senseBox sensor modules, the presented prototype fits well within their price range of 13 to 80€, depending on the number of nodes used. Finally, as only one senseBox is required for monitoring several locations simultaneously, the price for such a setup is considerably lower when compared to an installation with separate senseBoxes.

4.3 Easy to build

Again, a comprehensive definition of "easy to build" greatly depends on prior experience with the task at hand. Considering the target audience consisting of journalists and citizens, a threshold as low as possible is desirable. Firstly, the shown case is designed to be as simple as possible which allows easy fabrication with any 3D printer and no prior experience. The sensor is designed around an *Adafruit* breakout PCB

for the MEMS microphone which is easily available and allows for a "plug-and-play"-like setup instead of soldering the small microphone yourself. As outlined in the PCB design description in Section 3.4, only very few parts are required for the working sensor. While more parts to solder mainly increase the building time, a denser PCB layout also leads to harder to reach places which can be difficult for unexperienced users. Thus, the part count has been deliberately kept low by using a discrete linear power regulator instead of a switching power supply circuit. Due to the use of three AA batteries over a rechargeable battery pack, no additional charging circuitry is required. Omitting a rechargeable Lithium battery also makes the assembly process somewhat safer as careless handling of Lithium batteries can be quite dangerous.

While the PCB layout makes almost exclusive use of surface mount technology (SMT) parts, large footprints have been chosen to make the handling and soldering of components easier. Through hole components (THT) have been mostly avoided as they would crowd the PCB too much. In terms of component availability, all parts are easy to source from various vendors and are listed to be available in the future. All design files necessary for building the device are freely available under a permissible licence from the online repository[4] and can either be produced at home or via various PCB manufacturing and 3D printing services.

As an additional feature, the provided source code is both easily understandable and modifiable by using only the hardware abstraction layer (HAL) instead of low-level byte registers. The protocol mechanics are defined through simple control structures instead of timer interrupts that might be harder to grasp and debug for beginners.

Step-by-step assembly instructions are included with the design files in the mentioned online repository.

4.4 Easy to use

Apart from assembly, usage shall also be easy for the non-expert target audience. By building on top of the senseBox framework, a lot of simplifying abstractions – such as easy internet connectivity and near configuration-free upload capabilities – are already available. In accordance with the senseBox "plug in and go" mantra the provided software library makes using the sensor similarly easy as using the official sensor modules. The library provides sane and sensible protocol parameter defaults. Further, the protocol principle reduces setup complexity as remote sensor

[4] `https://gitlab.com/noise-sensor/noise-sensor`

nodes automatically receive and setup protocol parameters without involvement by the user. This aspect also allows to change the frequency of measurements without resetting involved devices in possibly hard to reach places. Further, additional nodes can be put into action at any given point without resetting other remote nodes simply by turning them on. Battery runtime is reasonably high and low batteries can be swapped quickly instead of waiting for an inbuilt battery pack to recharge. Yet, a completely independent setup without the need to replace batteries at all would be preferable.

Remote sensors inform the master node by themselves about impending failure due to low battery voltage. However, making the user aware of such alarms or other notices proves as a challenge when users don't have the senseBox display configured or are continuously monitoring the serial monitor. The weather proofing provided by the case design should theoretically ward against direct water ingress, however long term tests have not been conducted. For added weatherproofing the PCB can be coated in conformal coating.

Finally, a comprehensive user manual has been written and is available with the device.

4.5 Data access

Participants wished for easy and real-time data access in open formats. Again, by building on the senseBox framework and more importantly also employing the openSenseMap (OSeM), this requirement is easily satisfied. By default, sound level measurement packets are uploaded every 60 seconds. Past measurements can be retrieved either by calling the OSeM API or by downloading the daily backup. Both methods provide data packets in the open and standardised JSON or CSV formats.

4.6 Small user study

To provide some means of evaluation, a small user study was laid out. Three assembled nodes along with a senseBox were sent out for testing in a real-world scenario by a journalist. An assembly test was scheduled but could not be carried out.

The participant praised the easy to follow structure of the user manual including pictures and simply worded instructions. However, they noted that for low-threshold use with journalists, an even simpler solution is recommended. They suggested, that since a configuration-free plug-and-play solution is not entirely possible with

the presented design, a video tutorial could instead lower the threshold.

Further, they remarked that the WiFi Internet connection suffers from frequent dropouts. Although resetting the senseBox is easily done with the push of a button, disconnections are hard to notice when the device is not monitored constantly. A better solution for communicating this fault is necessary. Further, a more robust solution for automatically reconnecting to the WiFi should be put in place.

In general they consider the topic of sound pressure interesting as a medium for storytelling and emphasised the potential for journalistic use, especially in smaller local editorial teams.

4.7 Privacy

As citizens may be rightfully concerned about potential privacy issues with a recording device such as this sensor, this short section addresses these issue briefly. Any sound that is stored in the buffer is converted to an A-weighted decibel value within microseconds and discarded afterwards. The total length of the sound recordings is limited to only a few milliseconds which is not enough to capture meaningful contents of conversations. Additionally, those short snippets are only recorded every two seconds – for unmodified default values – which leaves too big a gap for the reconstruction of verbal exchanges.

5 Survey

As mentioned in Section 2.1, many sound measurement projects only focus on the technical aspects of measuring noise but fail to provide a helpful means to measurement data presentation for the average user. Therefore, three ideas for noise level visualisations that aim to respect non-expert user's requirements are proposed and examined in a questionnaire. Three hypotheses are tested through the visualisations:

1. The standard OSeM representation is not easily understandable and visually too busy.

2. Viewers do not understand when something is considered loud without guidance.

3. Viewers take reported numbers at face-value and don't consider measurement errors when reading graphs.

Hypothesis 1 is tested by providing the default OSeM representation as a baseline. Hypothesis 2 is tested by adding an interpretation guide to every proposed visualisation in the form of coloured sound level thresholds. Hypothesis 3 is tested through deliberately reducing the resolution and detailedness of the depictions thus keeping users from focussing too much on singular and possibly outlying values.

The survey consists of four separate depictions of the measured sound level over time. The values are shown for the time between 06:00 and 22:00 as defined by the emission guidelines of the municipality[32]. Night values are not included due to the fact that those generally exhibit very flat measurement curves with no peaks or valleys and are less suitable for testing the visualisation hypotheses. Each visualisation is shown twice, with one high and one low noise scenario – whereas the order is randomised – to assess whether participants are able to discern the difference between those scenarios.

Loudness values for every visualisation are presented as *equivalent A-weighted sound level* (LA_{eq}) that averages all measurements within a given time period to a single value for that window. The formula is shown in Equation 6 in Section 3.2.2.

Over all depictions – except for the default OSeM graphs – the same colour scheme is applied. It consists of green, yellow and red, the colours found in stop lights. This colour scheme works for two reasons: First, the overall amount of identifiable colour codes should not exceed ten and ideally be limited to no more than four different colours [17, p. 38]. Second, the green-yellow-red colour scheme

should be universally understood across viewers. When choosing a colour-coded representation, one has to consider that colour-codes can have different meanings in different cultures.

While representations could certainly be done in an interactive setting, this survey is conducted with non-interactive depictions such as graphs and animations to reduce the overall complexity.

Exemplary depictions of the visualisation scenarios are shown below. Full size pictures are available in the appendix.

5.1 OpenSenseMap default view

The openSenseMap provides a standard view for all measurements regardless of their unit. This view represents the baseline for the survey, testing hypothesis 1 and is depicted in Figure 11. Note that the graph shown to survey participants is a replica of the visualisation on the OSeM website since the export function does currently not work. As the OSeM representation allows interaction by the user while the survey is static, a second view with an enlarged portion of the data is provided to simulate the feature to some degree and is shown in Figure 12. The default view includes the time on the y-axis and the measurement value on the x-axis. Every value is plotted with a slightly translucent blue dot which leads to a gradually more opaque solid blue dot when multiple markers overlap each other.

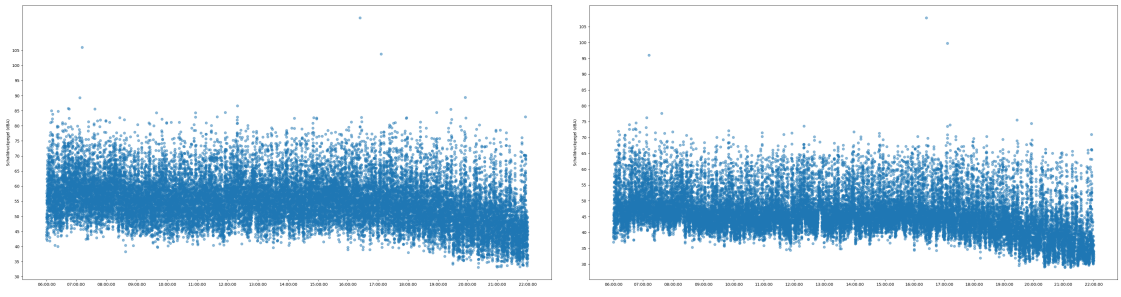


Figure 11: Default OSeM visualisation for both loudness scenarios

5.2 Fan chart visualisation

The first visualisation proposal consists of the equalised sound pressure level over the given time frame. As mentioned above, representations aim to actively prevent the user from overinterpretation of data and communicate data uncertainty. The graph representation in Figure 13 therefore includes an error margin around the actual

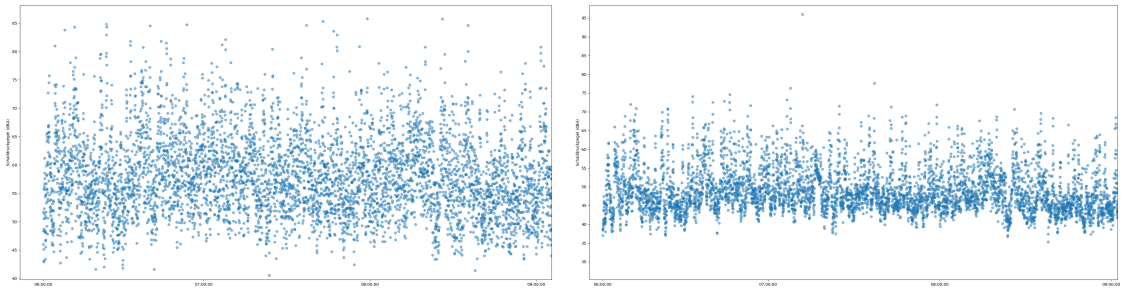


Figure 12: Default OSeM visualisation for both loudness scenarios zoomed in

measurement value to communicate that sensor data might be higher or lower than the centre line indicates. The error margin is derived from the measurement results in Section 4.1.2 and spans 2 dBA on either side of the centre line. To discourage viewers from concentrating on the main line, its contrast to the error margin is kept deliberately low [17, p. 42]. Sometimes referred to as *fan chart*, this type of visualisation is common in GDP projections [16] to communicate the possibility of future changes to the public. Galvão et al. [7] report that in a study of 3,000 participants, they found that the majority of participants did not take projections at face-value and instead expected data uncertainty when presented with a graph with error margins around its centre line. Additionally, two lines that correspond to the daytime thresholds for residential-only (50 dBA) and mixed-housing (60 dBA) areas defined by the municipality are included as visual guides for the reader and aim to test hypothesis 2. LA_{eq} levels are plotted for 30 minute intervals to smoothen the curve while tick marks are provided for 60 minute intervals in order to not overload the graph visually.

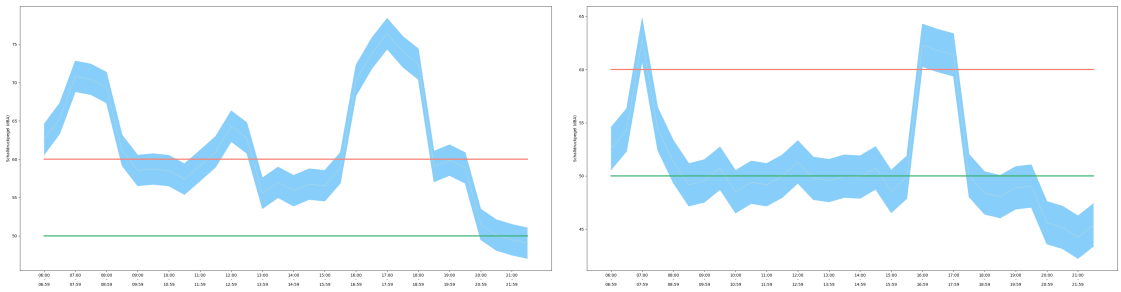


Figure 13: Graph depiction with error margin and thresholds for both loudness scenarios

5.3 Heat map visualisation

To explore a form of data representation different from a linear graph, Figure 14 is proposed. This approach resembles a heat map with intervals of one hour for each cell. LA_{eq} values are not reported directly but are encoded in the mentioned traffic-light colour scheme for testing hypothesis 2.

The resolution is kept low on purpose to make the chart readable at a glance. By deliberately reducing the number of distractors (cells), the information can be processed preattentively without conscious attention by the user [35]. In this way, a grid that exhibits mostly greenish colour can immediately be recognised as the representation of a lower sound level. Again, the same thresholds as mentioned above are applied. The colour map is defined such that values around the thresholds lie on colour transitions instead of hard edges, thus avoiding overinterpretation for values that are close to the edge due to measurement errors. Compared to conventional heat maps, this approach shows less variation in colour and values distinctly above the thresholds remain at the same solid colour.

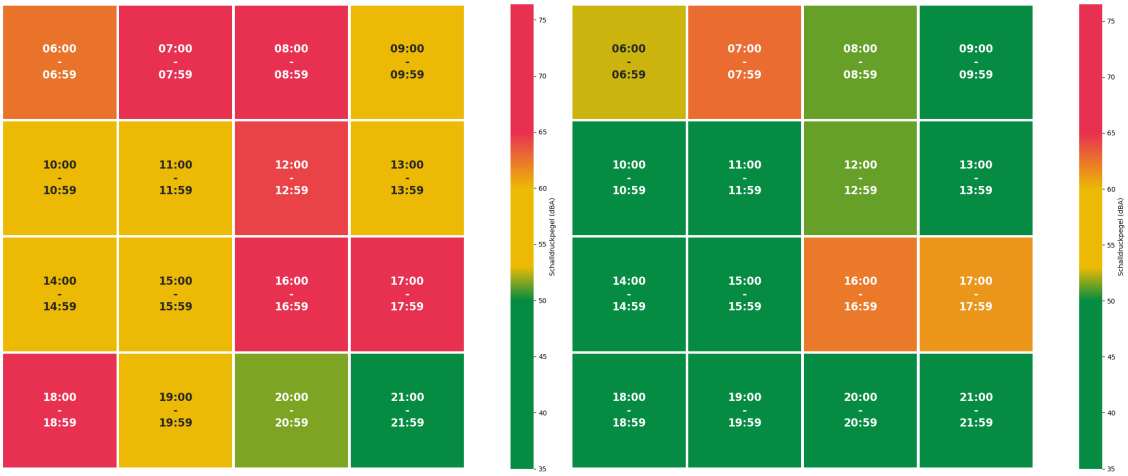


Figure 14: Heat map-like depiction of the sound level reported in hourly intervals for both loudness scenarios

5.4 Animated visualisation

The fourth visualisation approach disregards static images altogether and instead provides noise levels for each sensor node animated over time on a map. LA_{eq} values are given in 30 minute intervals to smoothen the animation while timestamps are updated in hourly intervals to reduce distraction. Again, the same three colour

scheme is employed for values above or below the defined thresholds. In addition, the circle radius grows or shrinks depending on the noise level. Uncertainty is communicated by fuzzy circle edges. A common problem in data visualisation lies in *Change Blindness* where viewers are unable to discern quick changes in value when briefly changing their focus[30, p. 59 et seq.]. To make changes in loudness more apparent and to allow the user to follow transitions more easily, every frame transition is cross faded over 600 ms which results in slower changes of colour and diameter. The total video length for the 16 hour period is 37 seconds.

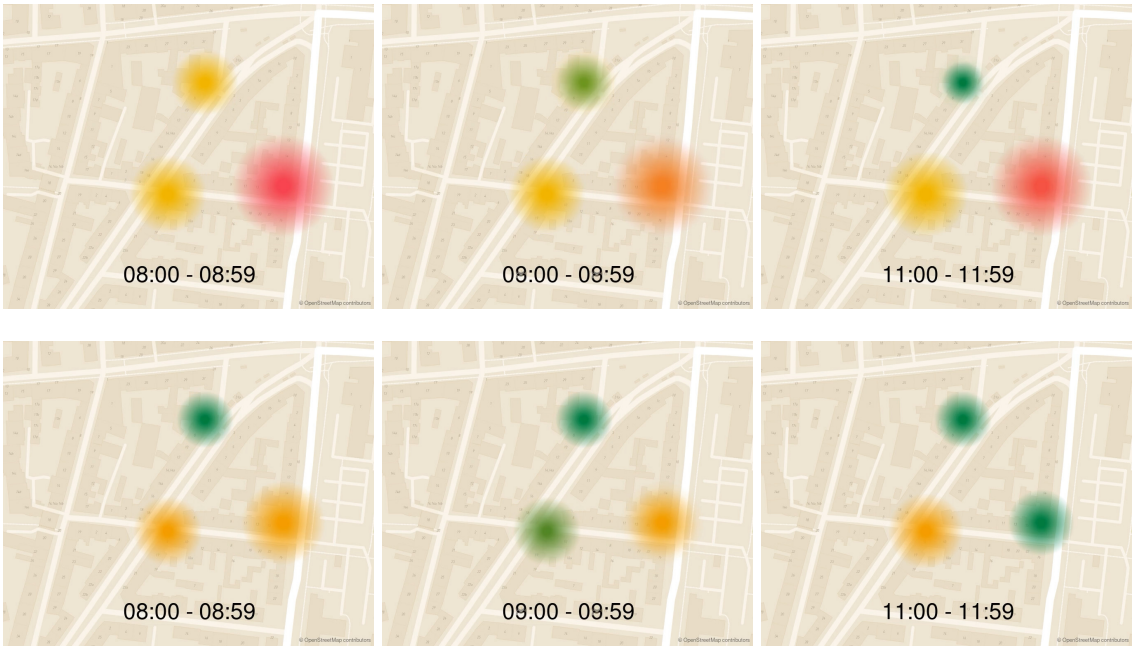


Figure 15: Three frames of the animation for each loudness scenario showing the cross fading of colours and differing circle radii

Finally, it should be noted that none of the proposed visualisations are accessible to people with achromatopsia. However, for use in production the colour schemes could be easily interchanged with other barrier-free pallets, for instance through a switch.

5.5 Survey questions

Before the visualisations are shown to the participants general demographic data such as their age and gender is collected. The users are further asked to describe

whether they have any prior experience with sensor data and how they would describe the noise within their living area on a scale from 1 to 5.

For every randomly selected visualisation, four questions – five for the animation type – are shown:

1. How loud would you describe the shown scenario?
2. How detailed do you think the presented measurements are?
3. How reliable do you consider the shown noise level regarding its effects on living at that location?
4. How helpful do you consider this noise representation when looking for a flat or house?
5. Would you expect to be bothered by noise if you lived at a location marked red? (animation only)

Due to having two rounds of identical questions per visualisation proposal, the same questions can be averaged should responses differ significantly in between them.

Each question except for the last one is rated on a five-point scale with the respective coding scheme listed in Table 3. At the end of the survey, a text field is provided for additional comments.

Score	Loudness	Detailedness	Reliability	Helpfulness
1	Very quiet	Not at all detailed	Not at all reliable	Not at all helpful
2	Quiet	Less detailed	Less reliable	Less helpful
3	Averagely loud	Neither	Neither	Neither
4	Somewhat loud	Somewhat detailed	Somewhat reliable	Somewhat helpful
5	Very loud	Very detailed	Very reliable	Very helpful

Table 3: Likert scale coding scheme

5.6 Results

Following is an analysis of the survey results. 48 people took part in the questionnaire, whereas only 24 completed all the questions. Most participants stopped when confronted with the first visualisation after filling out the demographic questions. Participants were predominantly male with 66.67% and 29.17% female. 4.16% did

not state their gender. The age distribution is depicted in Figure 16. 25% of participants stated to have prior experience with measurement data.

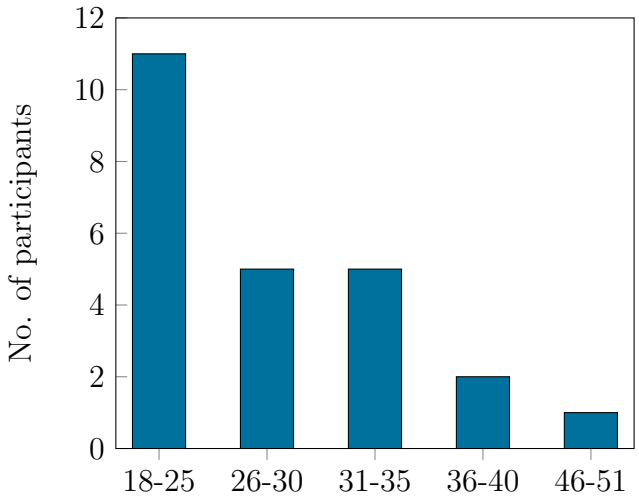


Figure 16: Age distribution amongst participants

Results for both scenarios of the visualisation proposals are detailed in the following subsections. The four metrics, loudness, detailedness, reliability and helpfulness are outlined for the respective representation. Loudness should be judged low for quiet and high for loud scenarios to satisfy hypothesis 2. Detailedness and reliability should be judged low to satisfy hypothesis 3, avoiding overinterpretation. Helpfulness should ideally be as high as possible but is not connected to any hypothesis.

5.6.1 OpenSenseMap default view

In general, respondents are able to roughly grasp the loudness of the visualisation. Nobody considered the loud scenario to be quieter than *averagely loud* but 37.5% rated the quiet setting as *somewhat loud* or *very loud*. 95% of participants deemed both visualisations *somewhat detailed* or *very detailed*. Two thirds rated both scenarios of the default visualisations to be *somewhat reliable* or *very reliable*. In terms of helpfulness when looking for a flat or house, the results are split at about 35% helpful to 50% not helpful.

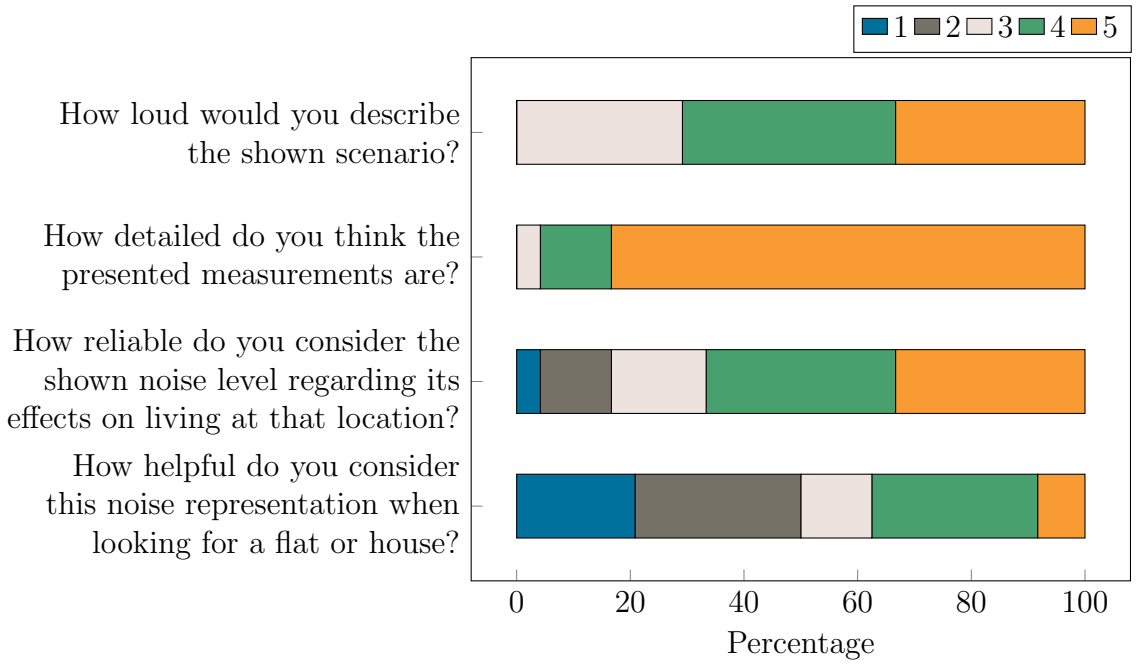


Figure 17: Results for the openSenseMap visualisation – loud scenario

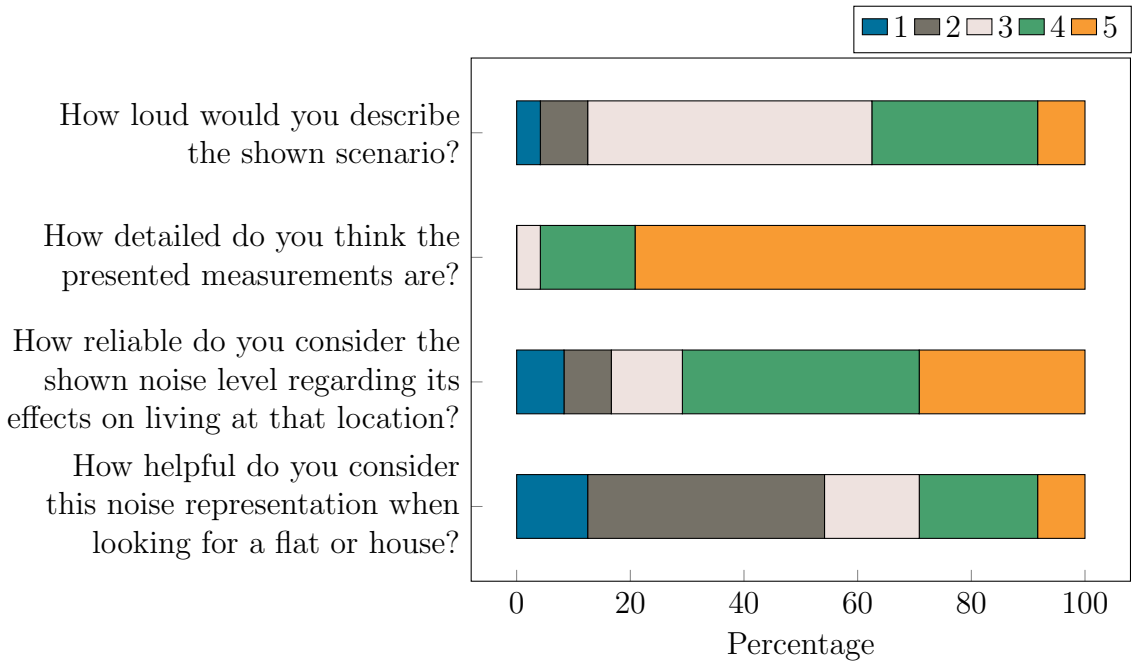


Figure 18: Results for the openSenseMap visualisation – quiet scenario

5.6.2 Fan chart visualisation

In terms of loudness, most participants (83.3%) rated the loud setting to be *somewhat loud* or *very loud*. The quiet setting was rated very similar to the openSenseMap default visualisation with participants rating the scenario too loud. In both the quiet and loud scenario, the visualisation was considered detailed with about 90% of agreement but overall less detailed, compared to the openSenseMap visualisation. Reliability was – when averaged between both depictions – rated at about 70%, again similar to the openSenseMap depiction and with only 8% deeming it *less reliable*. Nearly all responses deem the fan chart as helpful for the housing search with 80% of approval. However, 15% of participants deem the chart to be *less helpful* or *not at all helpful*.

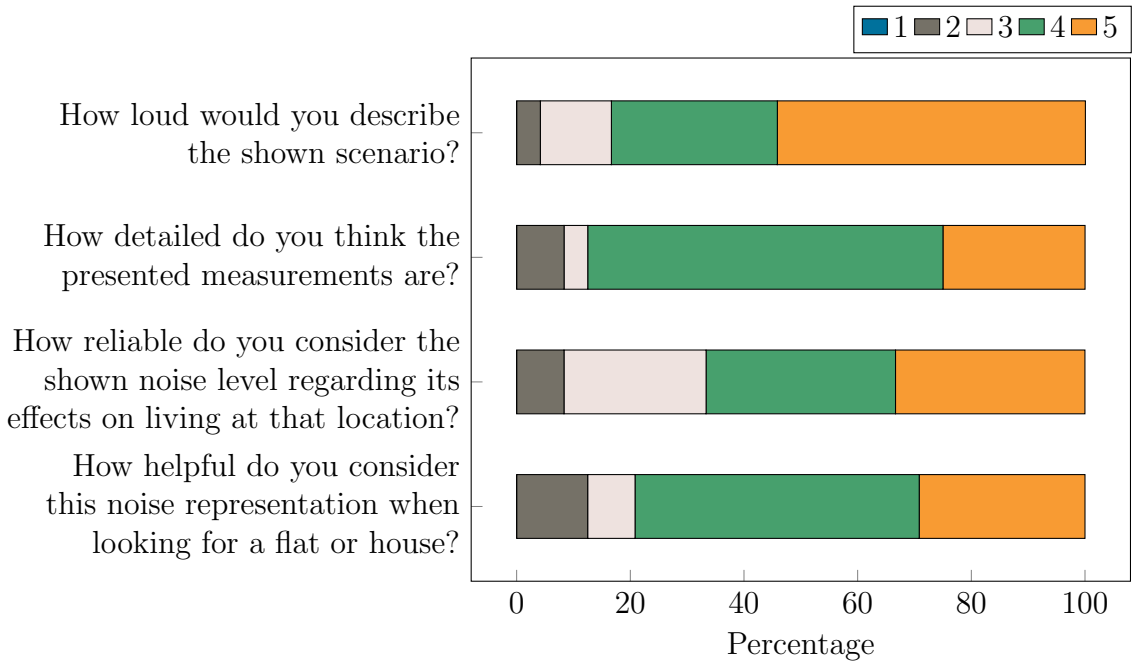


Figure 19: Results for the fan chart visualisation – loud scenario

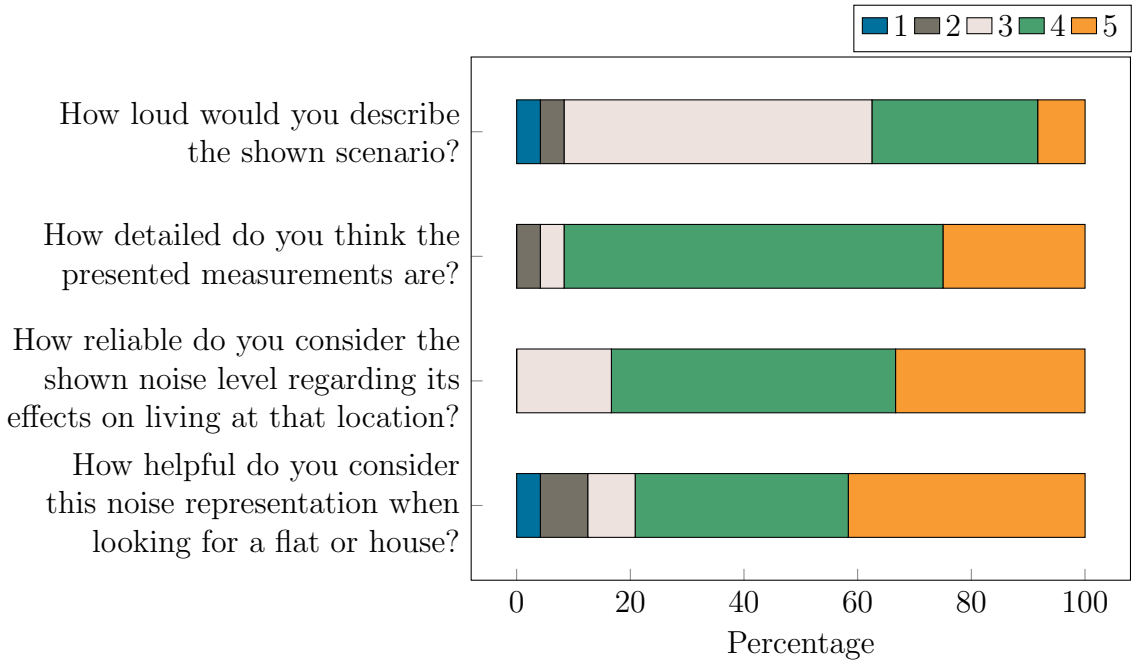


Figure 20: Results for the fan chart visualisation – quiet scenario

5.6.3 Heat map visualisation

The heat map visualisation showed the most concise response to the change in loudness between depictions. Over 91% correctly recognised the higher noise level. In the quiet heat map scenario, 45% of respondents accurately classified the quiet setting as *quiet* or *very quiet* with another 33% judging the depiction to be of average loudness. This depiction was the least wrongly rated in terms of loudness, with only 20% rating it *somewhat loud* or *very loud*. Detailedness shows a slight deviation between both scenarios. The mean for *somewhat detailed* and *very detailed* is 55% and shows no distinct trend. Heat map reliability is classified as *somewhat reliable* and *very reliable* by approximately 75%. The heat map visualisation reaches the highest score in terms of helpfulness with over 90% in both scenarios.

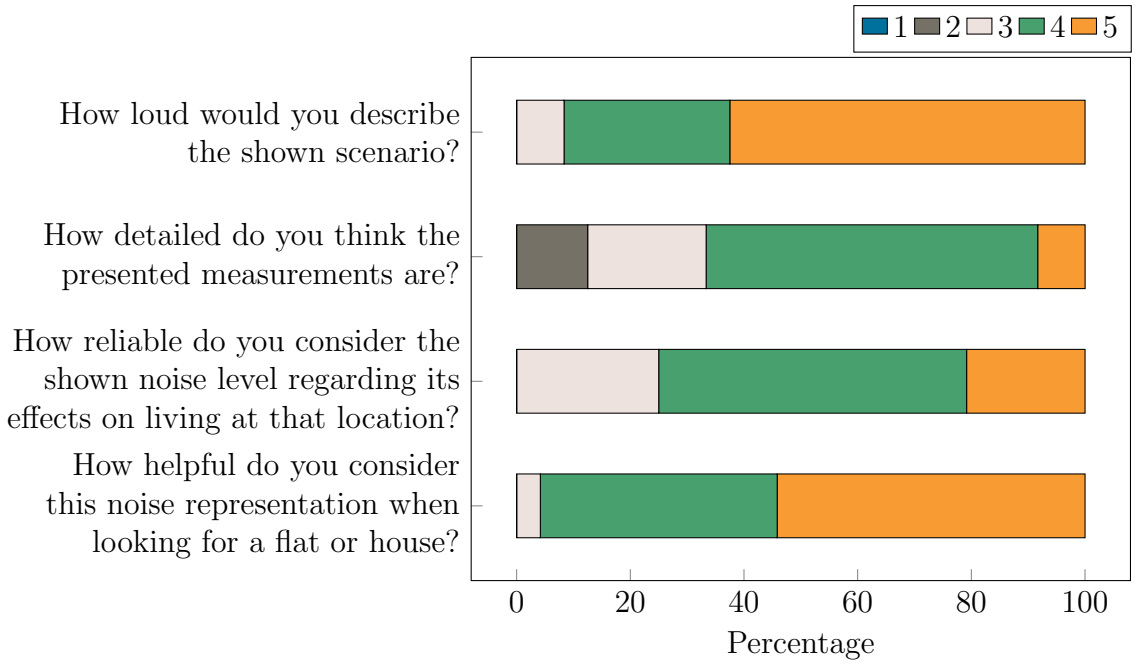


Figure 21: Results for the heat map visualisation – loud scenario

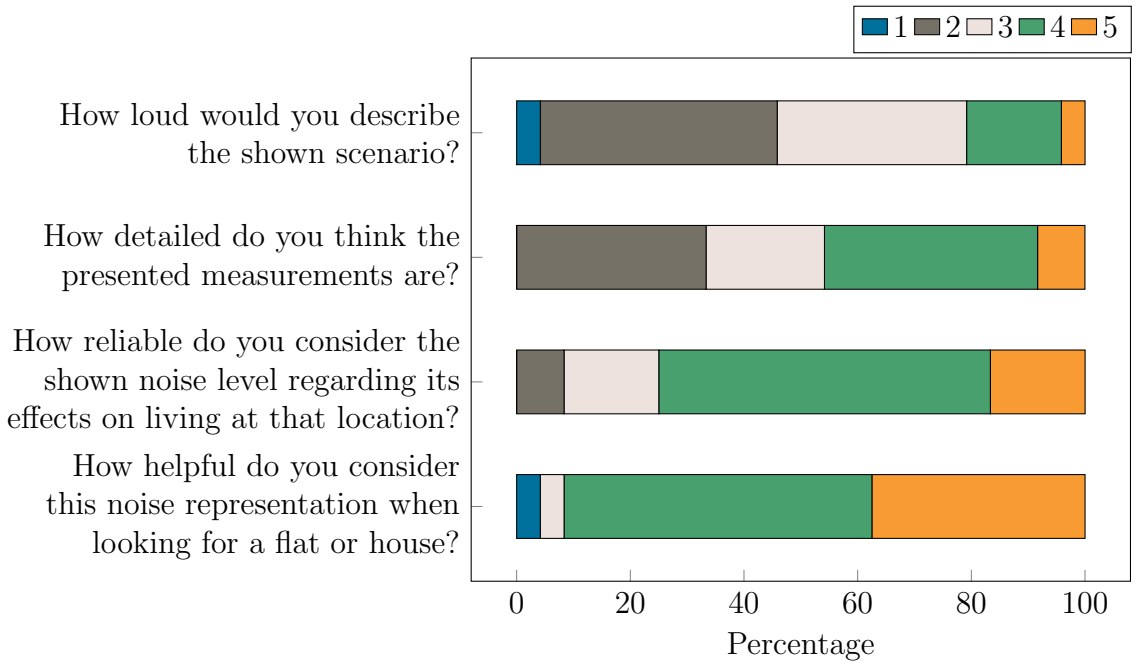


Figure 22: Results for the heat map visualisation – quiet scenario

5.6.4 Animated visualisation

It is evident that participants could not discern any difference between the quiet and loud animation scenario. Compared to other visualisations, respondents considered the animation to be the least detailed, albeit 45% considering it to be *somewhat detailed*. Reliability was still judged relatively high with two thirds giving it a score of *reliable* or *very reliable*. While considered *helpful* or *very helpful* by three quarters of participants, the animation is classified as less helpful than the heat map or fan chart but more helpful than the senseBox representation. For the fifth question, asking whether users would expect to be bothered by noise if they lived at a red marked location, only one participant responded with *no*.

Participants noted that watching the animations takes too much time and that diagrams instead allow to understand the noise level with less effort and within a shorter time.

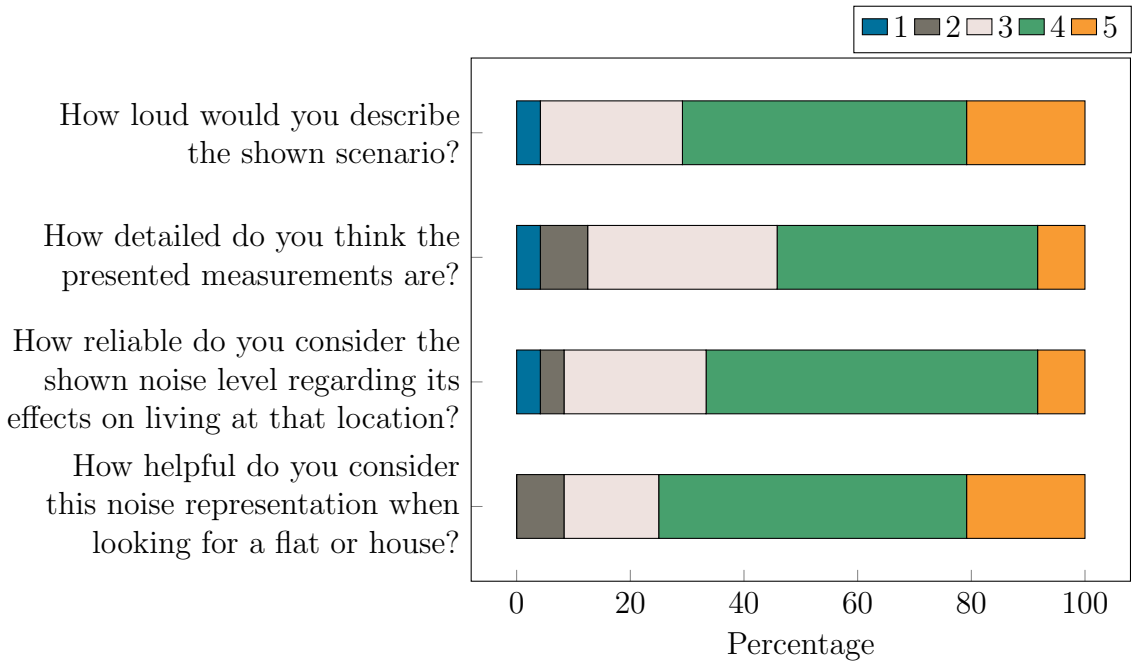


Figure 23: Results for the animation visualisation – loud scenario

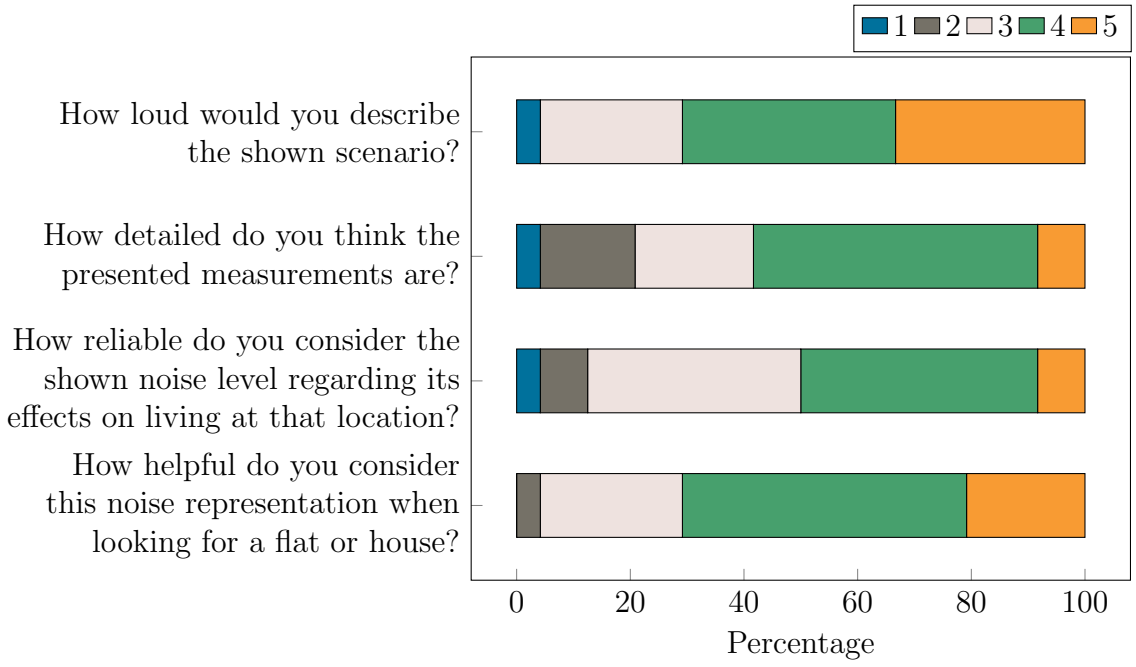


Figure 24: Results for the animation visualisation – quiet scenario

5.7 Survey conclusion

In general, the openSenseMap visualisation scored the worst. Participants were not able to easily identify the level of noise and further rated it to be very detailed, making it prone to overinterpretation. However, reliability was rated amongst the lowest. Participants commented that this representation was the hardest to understand. This comes as no surprise as this visualisation is not fitted for any specific measurement phenomenon such as sound level.

The animation could not reliably convey the loudness level either, with participants judging both settings similarly loud. While low detailedness and reliability ratings ward against overinterpretation, the visualisation is considered amongst the least helpful and was additionally described at hard to follow.

The fan chart representation helped considerably with classifying the noise level but also achieved the second highest detail rating. Reliability is considered the second lowest. Its helpfulness was rated the second highest. Especially the results in loudness recognition are encouraging but more effective techniques for discouraging overinterpretation are required.

The most promising results are reported for the heat map visualisation. Almost all respondents were able to correctly discern the difference in loudness. Participants rated the heat map to be less detailed than other depictions but considered it to be the most reliable, again making it susceptible to overinterpretation. In terms of helpfulness, this visualisation was rated the highest. In addition, participants commented that they would like to see such a indication on realty websites.

Hypothesis 1 – the OSeM visualisation is hard to understand – can be verified from the results shown above. The positive results in noise level classification for the fan chart and heat map visualisations along with the negative results for the OSeM default representation suggest that hypothesis 2 can be verified as well. Finally, a reduction in detailedness did not yield any reduction in reliability. Thus, hypothesis 3 cannot be considered verified.

In conclusion, graphs are preferred and understood easier by viewers. The heat map presentation allowed for the most intuitive understanding of loudness levels and is rated the most helpful but may be prone to overinterpretation. Further and more focussed research is required to find ways to dissuade users from trusting the heat map values too much.

6 Conclusion and outlook

This thesis outlines the implementation of a low-cost and easy to use noise sensor network that can be easily built with the files provided. The noise sensor builds on top of the established *senseBox* framework that empowers interested citizens to easily start measuring noise on their own in places where no noise data is available. For reliable data transmission between sensors, a time-based communication protocol was devised. The software library that was developed for the *senseBox* and *ESP32* microcontroller is designed with sensible defaults that require minimal intervention by the user while still allowing fine grained adjustments if needed. The low building cost ensures a low entry threshold for a large audience. The device was designed with users' and journalists' demands in mind and has been evaluated against those requirements. In a small user test those demands have been tested in a real-world setting.

In addition to the technical side, viable measurement visualisations are provided. The thesis contributes three ideas for providing a data representation that is easily understandable by a lay audience. The visualisation proposals have been designed with respect to established data representation techniques and have been evaluated in a survey. A fan chart and heat map for loudness indication have received high approval ratings by study participants, while it was found that an animation approach did not convey the noise level reliably.

The thesis provides a solid foundation for future research. While the results of the survey are promising, a larger-scale study would provide more substantial results. Further, focussing a second study iteration on the most helpful data representations could provide an even better insight with more detailed results for those visualisations. Moreover, an interactive setting allowing to manipulate certain aspects of the visualisations may offer a different view along with new insights. Finally, an actual integration of the proposals into the openSenseMap could be an interesting addition.

In terms of technical aspects, the proposed communication protocol would benefit greatly from added robustness features such as acknowledgment of received packets and resending capabilities. The used MEMS microphone offers a frequency range between 50 Hz and 11 kHz which is sufficient for most types of noise. Other models that allow measurements above 11 kHz would be an interesting addition to the system. While the system itself offers accurate enough measurements, a study looking into mounting options and location choice – both important factors for the

final result – could offer additional accuracy. Currently, the sensor only reports the sound level in dBA. Another possibility would be to make use of raw audio data or the frequency composition to identify various sources and locations of appearing sounds along with a suitable depiction. Although the cheapest possible microcontroller has been chosen for this project, there may be cheaper and more efficient parts available. The device might benefit from a deeper look into code optimization in terms of memory and processing power usage. The same goes for a detailed analysis of battery usage as well as optimization. Adding a photovoltaic panel for real self-sufficient operation would be another interesting addition worth looking into.

References

[1] Anderson, D. BOINC: A system for public-resource computing and storage. In *Fifth IEEE/ACM International Workshop on Grid Computing* (2004), IEEE.

[2] Anderson, D. P., Cobb, J., Korpela, E., Lebofsky, M., and Werthimer, D. SETI@home. *Communications of the ACM 45*, 11 (nov 2002), 56–61.

[3] Can, A., Leclercq, L., Lelong, J., and Defrance, J. Capturing urban traffic noise dynamics through relevant descriptors. *Applied Acoustics 69*, 12 (dec 2008), 1270–1280.

[4] Crall, A. W., Jordan, R., Holfelder, K., Newman, G. J., Graham, J., and Waller, D. M. The impacts of an invasive species citizen science training program on participant attitudes, behavior, and science literacy. *Public Understanding of Science 22*, 6 (apr 2012), 745–764.

[5] DFRobot. Gravity analo sound level meter sku sen0232-dfrobot. `https://wiki.dfrobot.com/Gravity__Analog_Sound_Level_Meter_SKU_SEN0232`.

[6] Farmer, R. G., Leonard, M. L., and Horn, A. G. Observer effects and avian-call-count survey quality: Rare-species biases and overconfidence. *The Auk 129*, 1 (jan 2012), 76–86.

[7] Galvao, A. B., Mitchell, J., and Runge, J. Communicating Data Uncertainty: Experimental Evidence for U.K. GDP. EMF Research Papers 30, Economic Modelling and Forecasting Group, 2019.

[8] Gardiner, M. M., Allee, L. L., Brown, P. M., Losey, J. E., Roy, H. E., and Smyth, R. R. Lessons from lady beetles: accuracy of monitoring data from US and UK citizen-science programs. *Frontiers in Ecology and the Environment 10*, 9 (aug 2012), 471–476.

[9] Jennett, C., and Cox, A. L. Digital citizen science and the motivations of volunteers. In *The Wiley Handbook of Human Computer Interaction*. John Wiley & Sons, Ltd, dec 2017, pp. 831–841.

[10] Jennett, C., Kloetzer, L., Schneider, D., Iacovides, I., Cox, A., Gold, M., Fuchs, B., Eveleigh, A., Mathieu, K., Ajani, Z., and

Talsi, Y. Motivations, learning and creativity in online citizen science. *Journal of Science Communication 15*, 03 (apr 2016), A05.

[11] Knowles. *SPH0645LM4H-B I2S Output Digital Microphone*, Feb. 2017. https://www.knowles.com/docs/default-source/model-downloads/sph0645lm4h-b-datasheet-rev-c.pdf?Status=Master&sfvrsn=c1bf77b1_4.

[12] Kosmala, M., Wiggins, A., Swanson, A., and Simmons, B. Assessing data quality in citizen science. *Frontiers in Ecology and the Environment 14*, 10 (dec 2016), 551–560.

[13] Kostoski, I. Sound level meter with arduino ide, esp32 and i2s mems microphone. https://github.com/ikostoski/esp32-i2s-slm, Nov. 2019. Accessed: 30-03-2021.

[14] Lewandowski, E. J., and Oberhauser, K. S. Butterfly citizen scientists in the united states increase their engagement in conservation. *Biological Conservation 208* (apr 2017), 106–112.

[15] Loos, L. Script for precalculating microphone correction values. https://gitlab.waag.org/lodewijk/amsterdam-sounds-kit/-/blob/master/Tools/PrecalculateBinScaleTable/precalculate_bin_scale_table.rb, Dec. 2019. Accessed: 25-03-2021.

[16] Manski, C. F. Communicating uncertainty in policy analysis. *Proceedings of the National Academy of Sciences 116*, 16 (nov 2018), 7634–7641.

[17] Mazza, R. *Introduction to Information Visualization*. Springer London, 2009.

[18] Noriega-Linares, J., and Ruiz, J. N. On the application of the raspberry pi as an advanced acoustic sensor network for noise monitoring. *Electronics 5*, 4 (oct 2016), 74.

[19] Papavlasopoulou, S., Giannakos, M. N., and Jaccheri, L. Empirical studies on the maker movement, a promising approach to learning: A literature review. *Entertainment Computing 18* (jan 2017), 57–78.

[20] Pesch, M. Volunteer's interaction in technology driven citizen science projects. mathesis, University of Muenster, Münster, Aug. 2018.

[21] Peterová, R., and Hybler, J. Do-it-yourself environmental sensing. *Procedia Computer Science 7* (2011), 303–304.

[22] Picaut, J., Fortin, N., Bocher, E., Petit, G., Aumond, P., and Guillaume, G. An open-science crowdsourcing approach for producing community noise maps using smartphones. *Building and Environment 148* (jan 2019), 20–33.

[23] Rana, R., Chou, C. T., Bulusu, N., Kanhere, S., and Hu, W. Ear-phone: A context-aware noise mapping using smart phones. *Pervasive and Mobile Computing 17* (feb 2015), 1–22.

[24] Reedu GmbH & Co. KG. sensebox MCU. `https://sensebox.github.io/books-v2/home/de/komponenten/sensebox-mcu.html`, Jan. 2020. Accessed: 23-03-2021.

[25] Risojević, V., Rozman, R., Pilipović, R., Češnovar, R., and Bulić, P. Accurate indoor sound level measurement on a low-power and low-cost wireless sensor node. *Sensors 18*, 7 (jul 2018), 2351.

[26] Rotman, D., Preece, J., Hammock, J., Procita, K., Hansen, D., Parr, C., Lewis, D., and Jacobs, D. Dynamic changes in motivation in collaborative citizen-science projects. In *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work - CSCW '12* (2012), ACM Press.

[27] Schaldenbrand, P. Window correction factors. `https://community.sw.siemens.com/s/article/window-correction-factors`, Aug. 2019. Accessed: 23-03-2021.

[28] Semtech. *LoRa® and LoRaWAN®: A Technical Overview*, Feb. 2019. `https://lora-developers.semtech.com/uploads/documents/files/LoRa_and_LoRaWAN-A_Tech_Overview-Downloadable.pdf`.

[29] Silvertown, J. A new dawn for citizen science. *Trends in Ecology & Evolution 24*, 9 (sep 2009), 467–471.

[30] Spence, R. *Information Visualization*. Springer International Publishing, 2014.

[31] ST Microelectronics N.V. *STM32G071x8/xB*, Nov. 2018. https://www.st.com/resource/en/datasheet/stm32g071kb.pdf.

[32] Stadt Erlangen — Amt für Umweltschutz und Energiefragen. *Nachbarschaftslärm — Rechte und Pflichten.* Bräuningshof, Aug. 2017.

[33] Tan, W. M., and Jarvis, S. A. On the design of an energy-harvesting noise-sensing WSN mote. *EURASIP Journal on Wireless Communications and Networking 2014*, 1 (oct 2014).

[34] Thelen, B. A., and Thiet, R. K. Cultivating connection: Incorporating meaningful citizen science into cape cod national seashore's estuarine research and monitoring programs. *Park Science 25*, 1 (2008), 74–80.

[35] Ware, C. Chapter five - visual salience and finding information. In *Information Visualization (Third Edition)*, C. Ware, Ed., third edition ed., Interactive Technologies. Morgan Kaufmann, Boston, 2013, pp. 139–177.

[36] WHO Regional Office for Europe, Ed. *Environmental Noise Guidelines for the European Region* (Copenhagen, 2018), World Health Organization.

[37] Wigginton, C., and Deloitte & Touche LLP. Global mobile consumer trends, Apr. 2017.

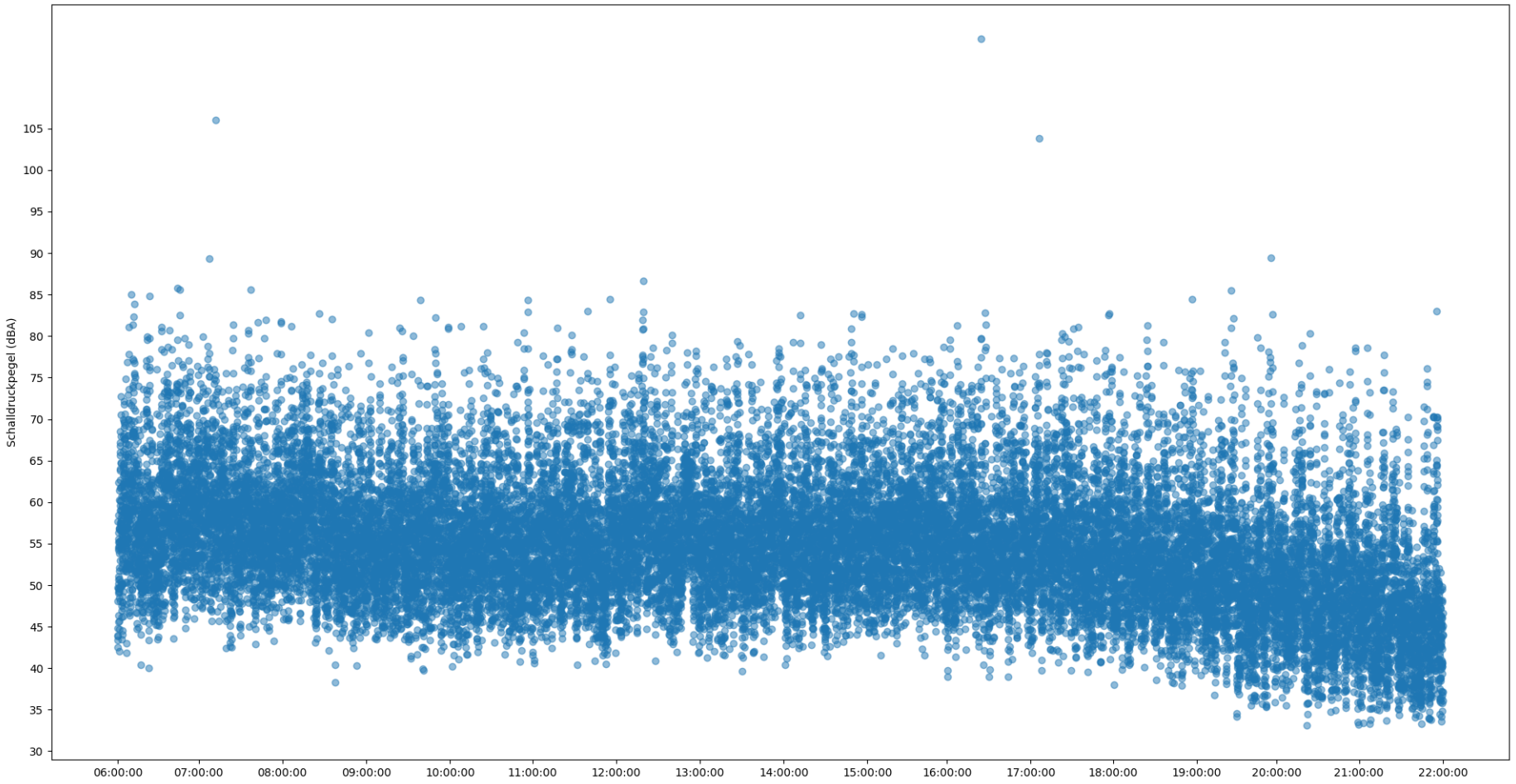
[38] Zamora, W., Calafate, C., Cano, J.-C., and Manzoni, P. Accurate ambient noise assessment using smartphones. *Sensors 17*, 4 (apr 2017), 917.

Appendix

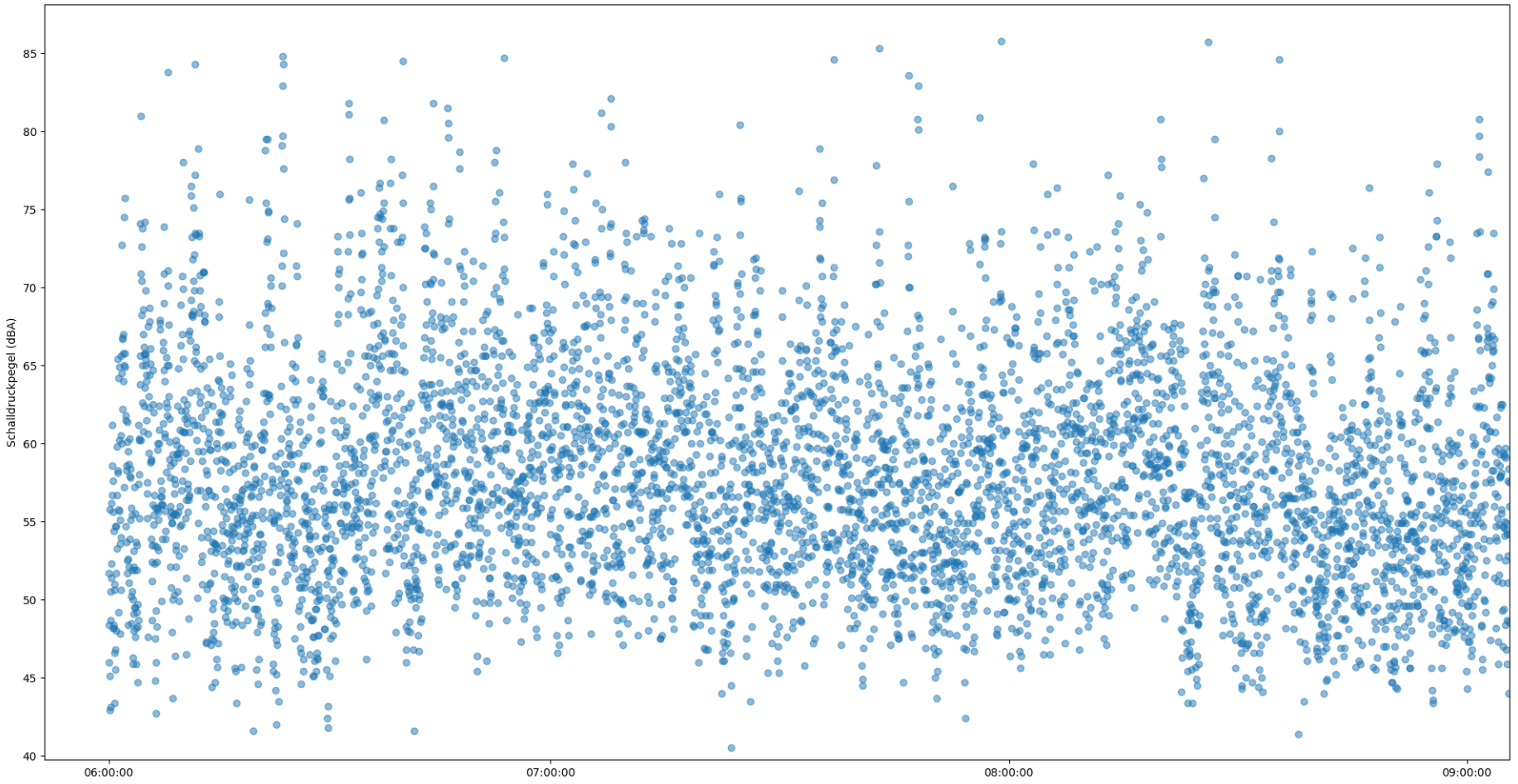
Full size visualisations

Assembly manual (German)

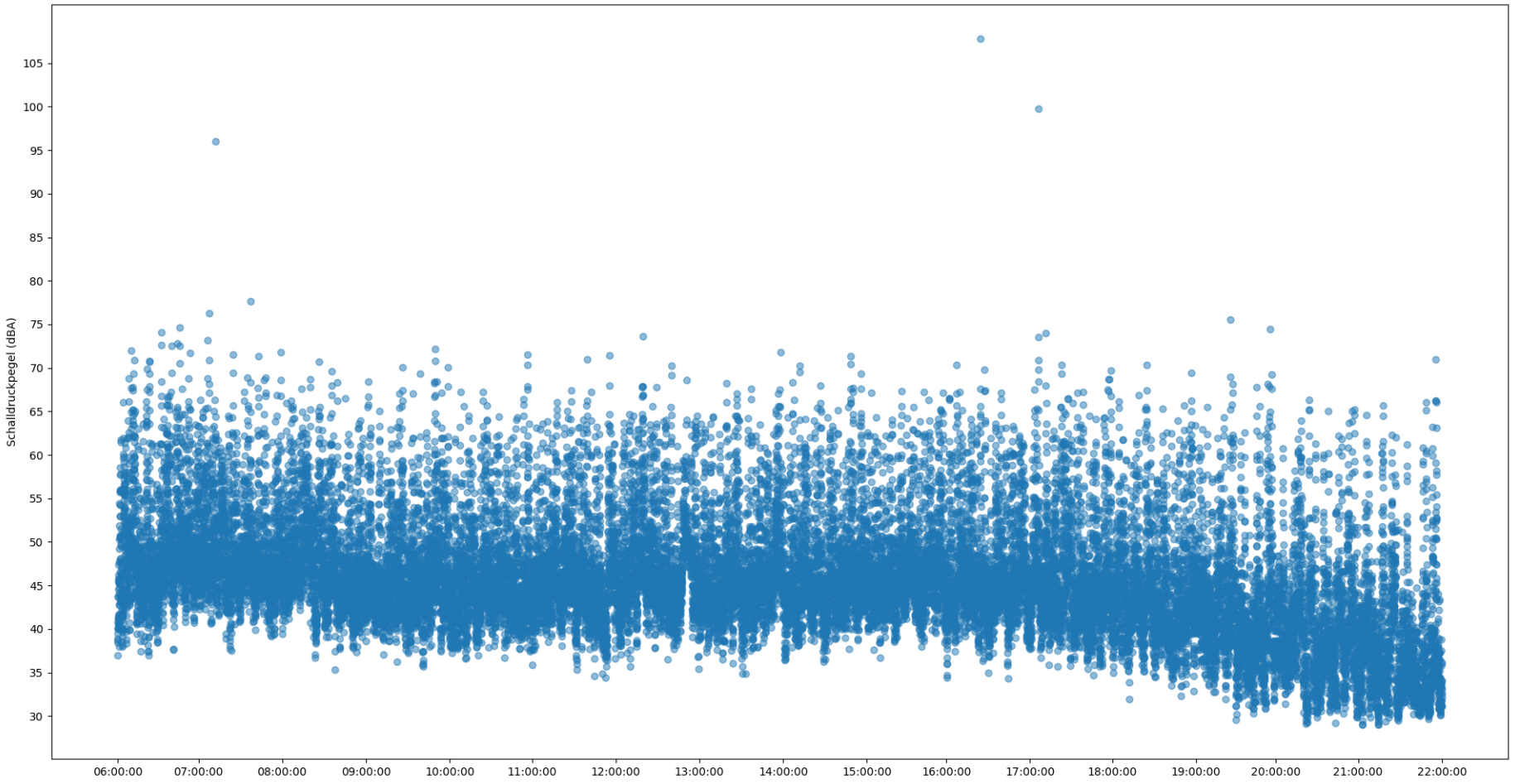
User manual (German)



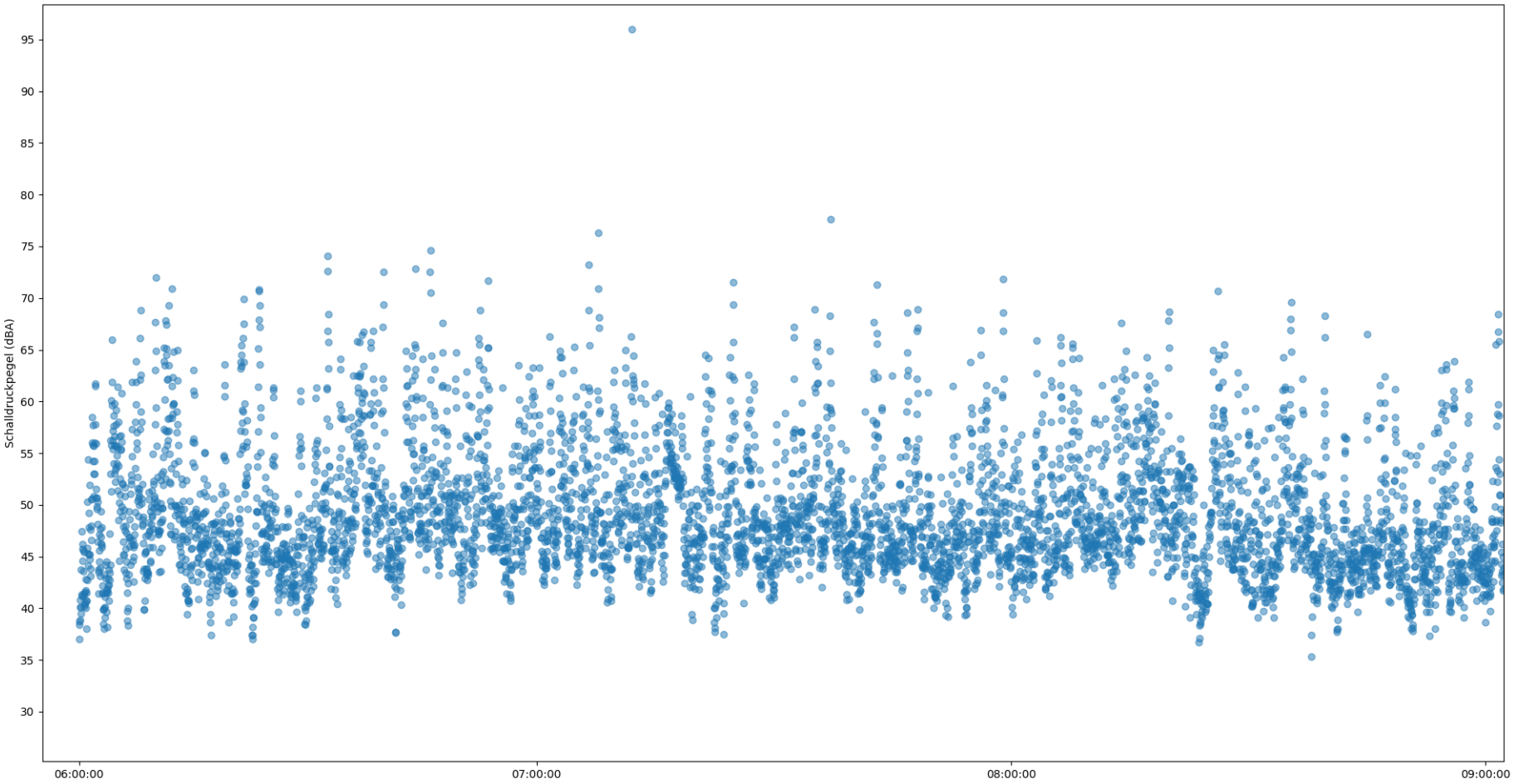
OSeM default visualisation – loud scenario



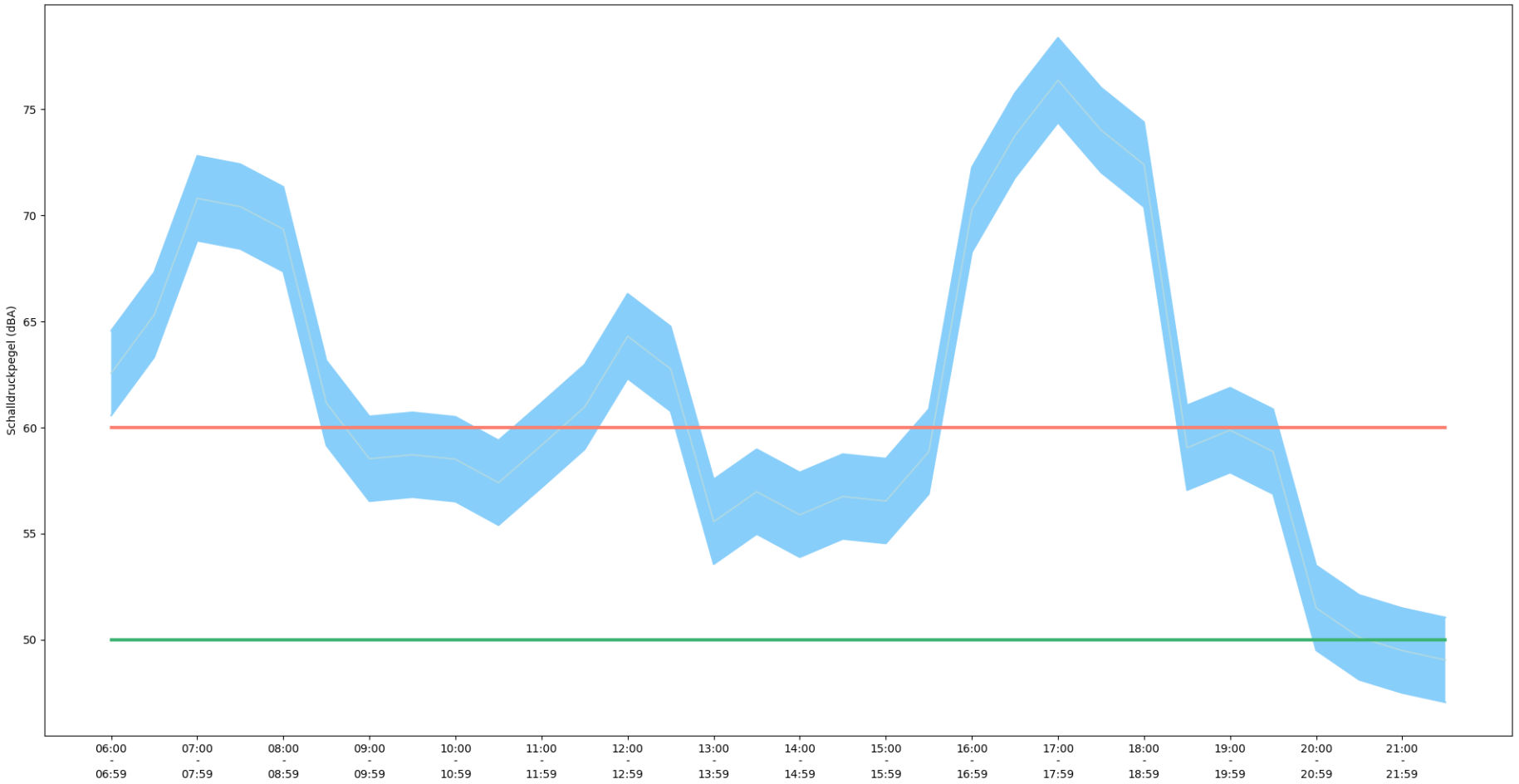
OSeM zoomed in default visualisation – loud scenario



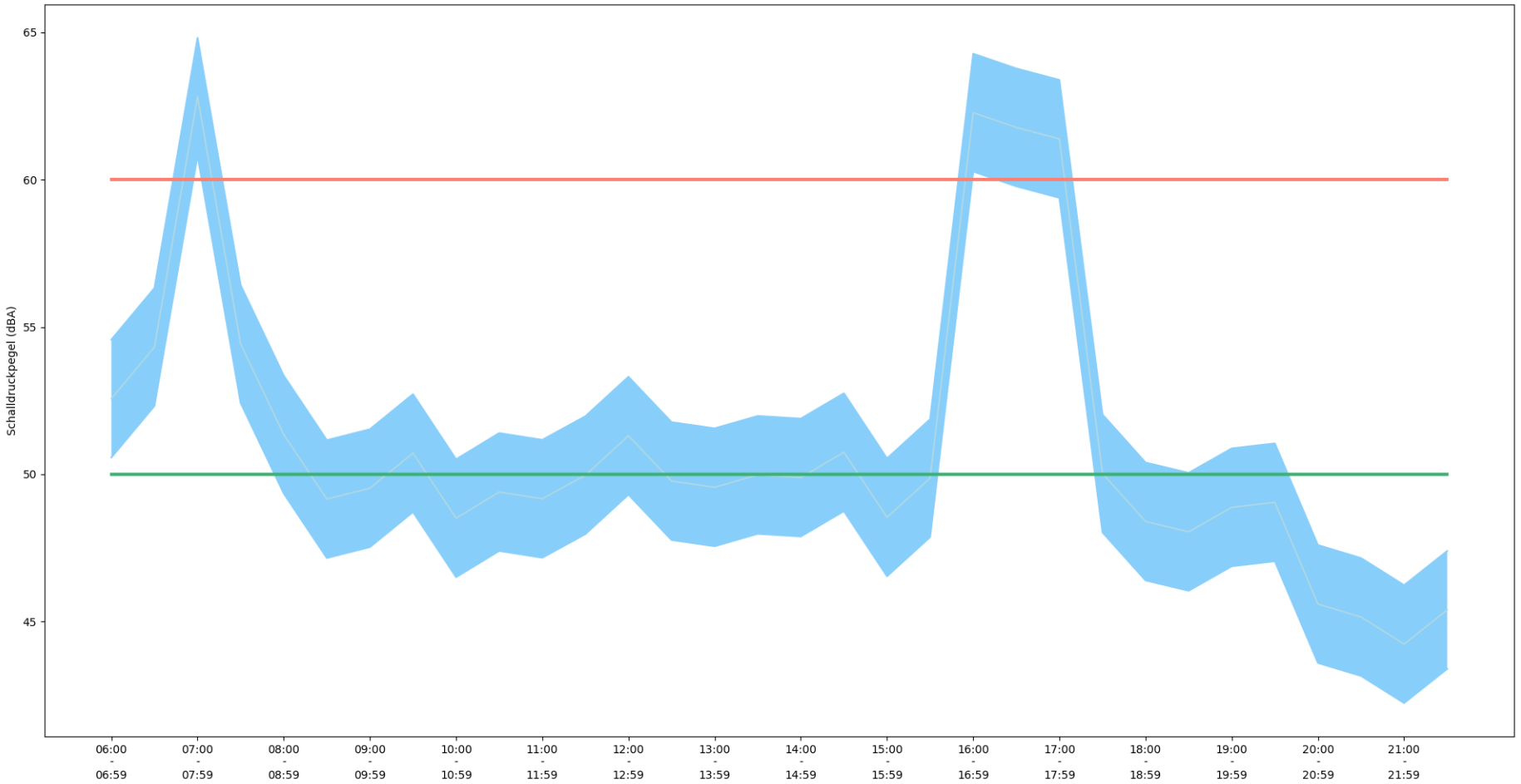
OSeM default visualisation – quiet scenario



OSeM zoomed in default visualisation – quiet scenario



Fan chart visualisation – loud scenario



Fan chart visualisation – quiet scenario

Heat map visualisation – loud scenario

Heat map visualisation – quiet scenario