Tangible Auditory Interfaces

Combining Auditory Displays and Tangible Interfaces

Dissertation
zur Erlangung des akademischen Grades
Doktor der Ingenieurwissenschaften
der Technischen Fakultät der Universität Bielefeld

vorgelegt von Till Bovermann am 15. Dezember 2009

Acknowledgements

I would like to thank my supervisor Thomas Hermann for all the freedom, support and lively discussions. It is a pleasure to work with him; without his active support this work would not have been possible. To my co-workers and friends in the Ambient Intelligence Group – Florian Grond, Tobias Großhauser, Ulf Großekatthöfer, Sebastian Hammerl Christian Mertes, Eckard Riedenklau and René Tünnermann – warm thanks for inspiring and stimulating collaboration and for making this thesis possible. Although some of you are not co-authors of the papers included here, your ideas and comments certainly have had a great influence. I would also like to thank all members of the Neuroinformatics Group, Bielefeld for providing a pleasant workplace and a friendly atmosphere. Due to the patience and kindness of Helge Ritter, head of group, I had much freedom to work on my projects. I thank him for his ongoing support. Thanks also to friends, colleagues, and personnel at the CITEC Center of Excellence and the Institute of Electronic Music and Acoustics for creating such vibrant and interesting environments to work in. I am especially grateful to Alberto de Campo for his active and thoughtful support in many circumstances. I learned a lot. I want to thank the Central Lab facility of the CITEC for their support in the production of the ChopStix system. I am also greatful to the people directly involved in the production process of ChopStix: Jan Anlauff, Holger Dierker, Florian Grond, Felix Hagemann, Simon Schulz, René Tünnermann, and Sebastian Zehe. Without you all, it would not have happened. I want to thank René Tünnermann, Florian Grond and Thomas Hermann for their valuable contributions to Reim. I would also like to thank Bodo Lensch and the Animax, Bonn for the opportunity to show Durcheinander in their performance space. For linguistic and language support during writing, I would like to thank Katrin Kaup.

I especially want to thank Claudia Muhl and Sven Nieder for their support on a professional, yet also private level. For both their professional support and general friendliness, I would like to thank Christof Elbrechter, Marianne Egger de Campo, Jonas Groten, Oliver Lieske, Lucy Lungley, Julian Rohrhuber, Katharina Vogt, Arne Wulf, and the inhabitants of the red house in Graz. Many thanks also to all of you I did not mention personally, who have provided feedback on this work by commenting and criticising, discussing and contributing ideas.

Finally, I am deeply grateful to my family and I dedicate this work to my parents and grandparents for their devoted support given to me throughout my life. To Ulrike, thank you for your support, hope, trust, joy and everything else.

During the work on this thesis, I have been employed by the Neuroinformatics Group and the Ambient Intelligence Group of Bielefeld University as well as by the Institute of Electronic Music and Acoustics, Graz. This work has been partially supported by the DFG through the SFB 673, and by the Center of Excellence, CITEC, Bielefeld University. I would like to thank the *European Environment Agency* and *wunderground.com* for their kind provision of near realtime ozone, respectively weather data.

Contents

List of Figures

1. Introduction

The world around us is full of artificially gathered data. Upon that data we draw conclusions and make decisions, which possibly influence the future of our society. The difficulty hereby is not the data acquisition – we already have plenty – but our ability to process it [Gol97]. Arising from this circumstance, at least two demands for data preparation can be identified: gaining appropriate attention depending on the data domains' nature and the users' needs [Gol06], and finding representations that truly integrate data and algorithmic functionality into the human life-world. I argue that a thoughtful data representation, designed in a way that it benefits from the various aspects of the human's being-in-the-world [Hei27], i.e. the complex interplay between the human and its environment, can fulfil these requirements.

Our awareness of being-in-the-world is often caused by the intensiveness of multi-sensory stimuli. The experience of walking through a cavern, feeling a fresh breeze that contrasts with the pure solid rock under the feet, hearing echoes of footsteps and water drops serves as a good example for this: All the simultaneously sensed impressions make us aware of our body and its integration into the cavern. The lack of a single sense, or a misleading impression would change the overall impression. In traditional computer-related work, many senses such as Hearing, Taste or Smell are underused. Historically developed paradigms such as the prominent *Graphical User Interface* (GUI) are not able to fully embed the user into the information to be mediated. Possible explanations for their nevertheless widespread use should be searched more in their (historically developed) technical feasibility [Sut63], rather than in usability and user-oriented simplicity. For about the last ten years, though, there has been a shift towards better representations of computer-based processes and abstract data, which try to close the gap between the users' reality and the abstract environment of data and algorithms. These fields take advantage of both display and controlling strategies by primarily incorporating other modalities than vision. Currently, these systems take advantage of either alternative display technologies such as auditory or haptic displays, or advanced controlling approaches like multi-touch or tangibility. I argue that the already promising achievements will be even better, if auditory and tactile displays are complemented by direct controlling approaches. Furthermore, I believe that their combination will unfold the true potential of interfacing technologies [Roh08]. In this thesis, I present such an approach with Tangible Auditory Interfaces, a combination of Auditory Displays and Tangible Interfaces.

I herein argue that haptic feedback as well as rich controlling and display possibilities are essential for a sufficient interface. While Tangible User Interfaces provide rich and, at the same time, direct control over digital data, sound and therefore Auditory Displays are widely recognised as very direct and flexible in their dynamic allocation of user attention. A combination of both is considered promising, featuring an informational-rich interface where users can select, interpret and manipulate presented data based on their excellent structure recognition abilities. The aim of this thesis is therefore to provide insights into the fundamental research on a combination of Tangible and Auditory Interfaces as integral systems.

Interfacing

Dealing with data means to operate on it and to experience it. The best computing algorithm is useless without an appropriate system to mediate and control its behaviour; data has to be fed into it, and the results have to be shown. Interfacing between data and algorithms on the one hand – abstract notions without a fixed physical representation – and the human reality – where physicality plays an important role – is a difficult venture. Many aspects like transparency and comprehensibility as well as perceptual and technical considerations have to be taken into account. Two areas, *Human Computer Interaction* (HCI) and *Interaction Design* (IxD) focus their research on this field. Although focused on the same – the interface between human and computer – the research fields are inherently different: While the IxD community takes care of the design and production of interfaces, HCI research is mainly about its analysis. In other words, while HCI primarily deals with the analysis of existing systems regarding their performance in various contexts, IxD is accounted for their design and production.

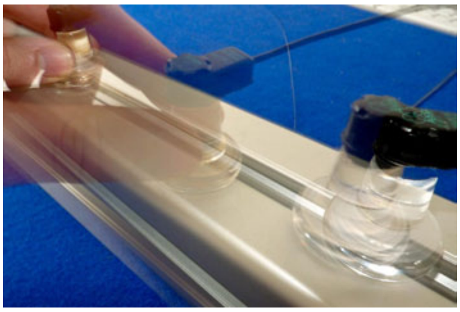


Figure 1.1.: Reim, a Tangible Auditory Interface for auditory augmentation.

For about the past ten years, both fields have experienced a change of their focus [LS04]. Originally oriented almost exclusively towards screen, mouse and keyboard as the central human computer interfaces, recent investigations cover also other interfacing techniques as they were developed for augmented- or virtual-reality systems [HHH+08] [KHS89]. Examples are tangible and multi-touch technologies: due to their interfacing capabilities, they provide a deep integration of the control of algorithmic systems into the user's environment. Multi-touch technology hereby frees users from manipulating complex digital systems with only a single pointing device (e.g. a mouse) by means of fingertip tracking. Tangible Interfaces go one step further by lifting actual parts of the digital data representation into the users' reality, making them graspable and manipulatable just like other physical objects [Ish08]. Also, display technologies other than vision were developed, leading to active discussions on multi-modality and its influence on display technologies [KACS03] [LCS03] [OCL04].

Tangible Interfaces

The young field of Tangible Interfaces (TI) picks up the concept of a more direct interfacing between users and computers that was not present in traditional GUI-based designs [UI00]. To achieve this, the community around TI introduced physical objects to the virtual world of the digital, fully aware of all their interaction qualities, but also of their ubiquitous limitations evolving out of their embedding in the real world. Tangible Interfaces exploit real world objects for the manipulation of digitally stored data, or – from a different point of view – enhance physical objects with data representations (either measured or rendered from artificial algorithms). This, on first sight very simple idea, turns out to be a powerful approach to the conscious development of complex, yet natural interfaces.

The user experience of a Tangible Interface is dominated by the incorporated physical objects. Their inherent natural features of which users already have a prototypical concept are valuable for the designer and make it easy to develop interfaces that are naturally capable of collaborative and multi-handed usage. Even further, the usage of tangible objects implicitly incorporates a non-exclusive application, so the system designer does not have to explicitly implement it.

Among other reasons, this shift was possible due to the availability of cheap sensor hardware. Former arguments against custom-built physical interfaces for software systems such as expensiveness, and lack of hardware liability were weakened and countered by the fact that the presence of dedicated physical properties such as position or extent of physical artefacts, together with the natural human knowledge about these properties even support the design of user-centred interfaces [PI07].

Auditory Displays

Not only research and perception of input technologies have changed over the last century, also the research in display technology has been taken a step further, discovering also non-visual modalities. The former focus on primarily visual displays has broadened to cover auditory [Kra94] and haptic cues [MS94] [BD97]. Particularly Auditory Displays (AD) have seen a strong uplift, since they support the human's excellent ability to perceive structures in a very different way than it is possible with the predominant visual display techniques. It turned out that sound rendering processes provide a way to display a reasonable amount of complexity. Therefore they are suited to display high-dimensional data. The benefit of sound, contrasting to other modalities besides vision, is that it can be technically rendered in a reasonable quality and spatial resolution.

The human perception of sound differs from visual perception. The human developed other structure detection and analysis techniques regarding the auditory sense, making it sensible for different structural information than they are recognised in the visual domain. These are among others timing aspects like rhythm, and the native support of time-based structures. The combination of Visual and Auditory Displays makes it possible to get a more complete interpretation of the represented data. Thus, the provision of the same data by more than one modality makes it possible to extend the usage of human capabilities in order to reveal the data's structure. Auditory Displays also natively support collaborative work [HBRR07], and allow for subconscious and ambient data representations [KL02].

Tangible Auditory Interfaces

Figure 1.2.: AudioDB, a Tangible Auditory Interface.

While both Auditory Display as well as Tangible Interface research are highly promising as individual fields of research, a combination of their techniques and experiences introduces valuable cross-links and synergies beneficial for both. In Tangible Auditory Interfaces, the tangible controlling component – focused on input capabilities of a system – is complemented by a display technology that supplements the existing haptic and visual cues. This combination forms an integral system for interactive representation of abstract objects like data or algorithms as physical and graspable artefacts. The primary modality of Tangible Auditory Interfaces for information and data mediation, therefore, is sound. During this thesis, I will point out that key features of Tangible Auditory Interfaces are interfacing richness, directness and flow, multi-person– as well as ambience and augmentation capabilities, and interface ergonomics. Furthermore, I state that specific requirements of Tangible Interfaces induce a certain gestalt or characteristic of the TAIs Auditory Display design and vice versa. Audio is a natural affiliate to physical objects; most of them already make sound, e.g. when touched or knocked against each other. Coming from the other direction, Auditory Displays in general and Sonification in particular profit from a direct control interface [HH05]; especially a highly interactive tangible input system allows a very close interaction loop between user and data representation.

1.1. Remarks

Abbreviations The following abbreviations will be used in this thesis:

AD Auditory Display

AR Augmented Reality

EDA Exploratory Data Analysis

GUI Graphical User Interface

HCI Human Computer Interaction

HID Human Interface Device protocol standard

IxD Interaction Design

L1 – L5, R1 – R5 Finger indices (see Figure 4.2 for details)

LOA Level of Abstraction

MBS Model-Based Sonification

RBI Reality-Based Interaction

RFID Radio-frequency identification

SETO SuperCollider Environment for Tangible Objects

TAI Tangible Auditory Interface

TDS Tangible Data Scanning

TI Tangible Interface

TUIO Tangible User Interface Object Protocol

Ubicomp Ubiquitous Computing

VR Virtual Reality

Figures All photos and images in this thesis are copyright by Till Bovermann. Exceptions are Figure 3.4(a) reprinted for exemplification from *Auditory Display* [Kra94], Figure 3.4(b) reprinted for exemplification from *Science By Ear: An Interdisciplinary Approach to Sonifying Scientifc Data* [dC09a], Figure 4.3 that is a reprint of video stills taken from the DVD *Rivers and Tides: Andy Goldsworthy working with time* [riv], Figure 4.6, printed by permission of the GNU Free Documentation License[1], Figure 5.6 courtesy by Eckard Riedenklau, Figure 9.2 and Figure 9.3 product images from various companies of digital audio workstations, and Figure 9.44 that is a collection of stills from video footage by Henrik Niemann.

[1] http://en.wikipedia.org/wiki/File:Planimeter.jpg

Code listings

Code listings are intended to exemplify specific aspects, and are therefore optimised for readability rather than computing time efficiency. They are written in SuperCollider [McC02] [WCC09], if not otherwise noted.

DVD

The accompanying DVD[2] contains a file hierarchy according to the chapters in this thesis. Videos and additional material are sorted into these folders. All videos are encoded using the H.264 standard, so that they should play back with any recent software video player.[3]

1.2. Document Structure

This work is divided into two parts. While the first part deals with theoretical deliberations on Tangible Auditory Interfaces and related fields, the second utilises the gained knowledge for practical investigations.

Part I is structured as follows: In Chapter 2, I provide an introduction to data on which all the other covered fields will be based. It describes the current role that data plays in our society, gives examples for its structure, and presents a formal definition. Chapter 3 then covers Exploratory Data Analysis (EDA), a research area that is part of data mining. EDA explicitly deals with the quest to find techniques and methods for exploring data for new information. The chapter focuses on data representation to enforce a variety of different data-driven experiences in order to help users to derive structural information from the data under investigation. Working with data in an explorative manner particularly means to use interfacing technology to bridge the gap between the virtual data environment and the human perceivable reality. The research fields *Human-Computer Interaction* and *Interaction Design* which will be described in Chapter 4. Their particular intention is to design and analyse the interface between human and machine, i.e. our reality and the digital realm of automated data processing. A relatively new field into which HCI and IxD investigate are Tangible Interfaces. Since they play an important role in this thesis, they are described and discussed in Chapter 5, in which I also develop considerations that I found essential for a theory-building for Tangible Interfaces in general and Tangible Auditory Interfaces in particular. To get the processed information derived from the underlying data to the user, display technologies are needed. They are examined in Chapter 6. Apart from a brief overview of the widely known visual display technology and its possibilities for data representation, I focus on Auditory Displays and their potential especially in closed-loop interfaces. This is followed by Chapter 7, in which the integration of Tangible Interfaces and Auditory Displays into *Tangible Auditory Interfaces* is introduced. Prior to a list of its key features, I will formulate a first proposal for a definition of Tangible Auditory Interfaces and its design guidelines.

Part II is structured as follows: Chapter 9 describes applications that I developed to support the ideas behind the TAI paradigm, illustrating their usefulness and potential regarding the features introduced in Chapter 7. An brief overview of these applications is given in Chapter 8. Chapter 10 introduces a software and a hardware framework that I have developed in co-operation with others over the last years to support the design and especially the implementation of TAIs. The thesis closes with a summary of my findings and an overall conclusion and outlook on further work in Chapter 11.

[2] The content of the DVD can also be found at `http://LFSaw.de/tai`.
[3] For example VLC: `http://www.videolan.org/vlc/`.

Part I.

Interfacing Digital Content with Auditory and Physical Devices

2. Data and its Central Role in Digital Environments

Data is omnipresent in our society. Everything is measured, and the resulting data is collected, analysed and used to help to control many aspects of the industrial world. One movement, enforced by the artificial need for speed and accuracy, is the shift of communication to use *digital* media. This trend turns communication away from the traditional face-to-face chat via low-tech media towards the use of digital media systems such as Voice over IP, internet-based chat, and electronic mail clients. This digital media, originally invented to be used for distance communications is increasingly used for communication in local settings, e.g. to distribute various information through an office.

It is, however, not only our communication that changed to the digital realm; our society itself depends increasingly on both digital data acquisition and automated analysis to gain new information. Essential application fields are for example marketing strategies, or the automated production of almost all devices and tools we use. In these applications, data on the production process has to be computed and analysed in both realtime and off-line to control the automated production units and monitor the build quality. The acquired data is digitally represented; coded either in numerical values or text. It often is composed into complex, non-linear structures in order to reflect the measurements and their interrelations. Data is not only processed by machines, but serves also as a resource for human analysis. The information extracted from that data is an integral part of our life that is used in both active and subconscious forms to understand and decide on living– and marketing strategies.

Data as a human resource

2.1. Examples for Common Data Domains

As a very abstract and general notion, *data* embraces a broad range of different shapes and meanings. Data taxonomy would surely differentiate between the two independent variables *Data Structure* and *Data Semantics*. While the semantic of data is particularly valuable for interpretation and reasoning, its structural appearance significantly influences which algorithmic exploration and analysis techniques may be applicable to get insights into its semantical meaning. Geospatial data sets, for example, usually contain data records that link a geographical location to observed values of measurements. These may be for example air pressure, temperature, humidity or wind speed. The structural relationship arises from the spatial distribution: possibly important information may be extracted from this data when considering relative distances between data points. Another example for data, now with a completely different semantic and structure is Electroencephalography (*EEG*) data, as it is measured by electrodes attached to a patient's scalp. It is used e.g. in medical applications for disease analysis, serving there as a base for medical findings. Also research in human cognition makes use of such data. EEG can be interpreted as a spatial

Geospatial data

EEG data

map of the scalp, using the electrodes' position as a three-dimensional location and the recorded current as a temporally changing feature vector. In addition to spatial relations, also timing aspects contribute to a meaningful interpretation of the data values, and should therefore be considered in their analysis.

2.2. Formal Definitions

Origins of the term Data

Data is plural for lat. *Datum*, *something given*; it is, however, today also used in singular to represent *a piece of information* [McK05]. The origins of today's interpretation of the term *data* as it is used in data mining and –exploration may be found in Riemann's definition of a *manifold* [Rie68]: a subset of a mathematical vector space with no dimensional semantics attached. This abstraction from semantics of the measured dimensions – in geospatial weather data these are the origins of the measured values, i.e. temperature, humidity, etc. – opens the possibility to apply (non-linear) operations to the manifold resulting in a new manifold where it is impossible to interpret the dimensions. However, the new representation possibly makes it easier to identify structural patterns such as clusters or data item dependecies.

Data as set

For data mining tasks, it is important to have a consistent mathematical representation of data sets. We define it as the set

$$\mathbf{X} = \{\mathbf{x}^\alpha\}_{\alpha=0,\ldots,m-1} \tag{2.1}$$

of m data items $\mathbf{x}^\alpha \in \mathbb{R}^n, \quad n \in \mathbb{N}$.

Sorted data sets contain additional information that is implicitly given by their ordering. This circumstance motivates their mathematical description as a series

$$\mathbf{X} = (\mathbf{x}^\alpha)_{\alpha=0,\ldots,m-1} \tag{2.2}$$

Data as matrix

Another way to represent data sets is to put them into a matrix

$$\mathbb{R}^{m\times n} \ni \mathbf{X} = \left(\mathbf{x}^0, \ldots, \mathbf{x}^{m-1}\right) = \begin{pmatrix} x_0^0 & \cdots & x_0^{m-1} \\ \vdots & \ddots & \vdots \\ x_{n-1}^0 & \cdots & x_{n-1}^{m-1} \end{pmatrix}, \tag{2.3}$$

where each column represents one data record. It is very close to the commonly used data structure in computers and has the benefit that all values are easily accessible by their indices. A disadvantage of such a representation, though, emerges, when trying to convert a discrete representation into an interpolated continuous vector field, since the dimensionality m of that matrix would be infinite. In this work, however, $\mathbf{X}$ will be used in the sense of (2.3). Exceptions are otherwise explicitly noted.

Data Domain

Data sets are usually embedded into domains. For a mathematical definition, we call the set $\mathbb{R}^n, \quad n \in \mathbb{N}$ together with the semantical description of its axes an *n-dimensional domain* $\mathcal{D}$. Since the semantical part of such a domain is very difficult to describe in mathematical terms, and therefore cannot make a contribution to this mathematical definition, I argue that an n-dimensional domain can by sufficiently described by $\mathbb{R}^n$.

In an arbitrary data set $\mathbf{X}$ that is encoded numerically (e.g. by a Shannon-coding), every data item can be described completely by an element $\mathbf{x} \in \mathbb{R}^n$. It can therefore be interpreted as a *point* in an n-dimensional domain.

For $\mathcal{D}, \mathcal{E}$ being two domains, and $k, l \in \mathbb{N}$, the concatenation given by

$$\mathcal{D} \circ \mathcal{E} = \{\mathbf{d} \circ \mathbf{e} | \mathbf{d} \in \mathcal{D}, \mathbf{e} \in \mathcal{E}\}, \tag{2.4}$$

with

$$\begin{aligned} \circ : \mathbb{R}^k \times \mathbb{R}^l &\rightarrow \mathbb{R}^{k+l} \\ \left(\begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_{k-1} \end{pmatrix}, \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_{l-1} \end{pmatrix} \right) &\mapsto \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_{k-1} \end{pmatrix} \circ \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_{l-1} \end{pmatrix} = \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_{k-1} \\ y_0 \\ y_1 \\ \vdots \\ y_{l-1} \end{pmatrix} \end{aligned} \tag{2.5}$$

forms a new $(k + l)$-dimensional domain.

Data storage and structures

On a computational level, data is often stored in structures that are closely related to the mentioned mathematical descriptions. Defining an array in SuperCollider, the computer language that is mainly used throughout the applications in this thesis [McC02], is done as follows:

```
a = ["value", 23];  // a new Array with 2 entries

a[0].postln;        // access a value and print it
a.do{|value, i|
    "At % there is %.\n".postf(i, value)
}
```

Among other possibilities to store data, there is also a more high-level representation, called *dictionary*. Such a dictionary associates arbitrary values with keys:

```
a = Dictionary.new; // a new Dictionary

a[\key] = "value";  // assign values to keys
a[\data] = [1, 2, 3, 4];

a[\key].postln;     // access a value and print it
a.keysValuesDo{|key, value, i|
    "The value of %(%) is %.\n".postf(key, i, value)
}
```

Another common approach for data representation in computers is implemented in *relational databases*. In this setup, data are hold in tables; two-dimensional arrays that are optimised for combining and filtering large, mostly numerical or textual databases.

Computation and algorithms

To handle data computationally, instructions in the form of algorithms and programs are needed. They determine how the data is treated, if and how it is filtered, sorted or processed to be finally presented to the user or stored in a database.

2.3. The Artificial Separation of Data and Algorithms

This section highlights the interdependencies between data and algorithms. It explains the difficulty in their separation, and outlines that, from a computational point of view, they are basically the same.

In our common environment, the reality, a clear separation between tool and material on the one side and its function on the other is noticeable: While the term *object*, be it considered a *tool* or its *material* is an abstract and general denomination for *something physical*, the notion of *function* denominates the idea of their specifics, their intended semantic.[1] The New Oxford American Dictionary describes *functionality* therefore as [McK05]

> the quality of being suited to serve a purpose well; practicality [. . .]
> the purpose that something is designed or expected to fulfill [. . .].

However, in digital environments these terms are different. In this case the separation between *tool*, *material*, or *data* but also *meaning* is blurred: An algorithm embodies both, the description of a process incorporating material, and the material itself, which makes it function, tool and material at the same time. My intent in this thesis is that the occurrence of data may always be interpreted as functionality, too. I think it is a good practice on the way to better understand the digital realm.

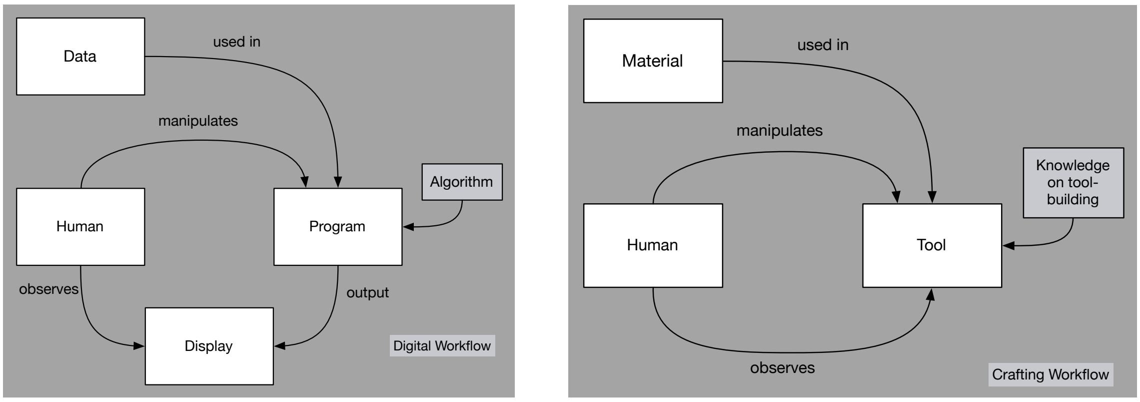
The fusion between algorithm and data is made explicit in Turing machines, where both the running algorithm and the data it operates on are stored on the same tape, making it possible to manipulate the program itself at runtime. This circumstance is for example used extensively in the Lisp programming language [Fod91].

Today, however, with mainly imperative programming languages dominating the field, it is rare that programs do change themselves at runtime. Exceptions can be found in artistic programming situations such as live coding and just in time algorithmic music performances [RdCW05]. Contrasting, the automated implementation of algorithms and customised functionality is widely known and used, e.g. in the production of serial letters with LaTeX. It's power can be compared to machines that are able to produce physical but customised tools like rapid prototyping systems.

2.4. Data Processing

As mentioned earlier, data and information are the dominating material for computers. They are designed to easily acquire, shape and display data. The diagram 2.1(a) shows a typical data workflow of a computing and exploration process. Similar to traditional crafting, it consists of acquisition (i.e. data acquisition or measurements), and manipulation with a tool (the program). In difference, though, the appearance of the resulting material can be chosen almost independently from the crafting process, because data has no human-perceivable gestalt by itself. Working on data with a computer can therefore be seen as closely related to the traditional way of hand-crafting physical material with the help of appropriate tools. The next paragraph, though, states that this is not the case on a closer examination.

[1] On a side note, this already incorporates a subjective view, since the function of an object depends on the user's *interpretation* of the object.



(a) Data Workflow (b) Crafting Workflow

Figure 2.1.: Similarities between the data mining workflow and the typical handcrafting workflow.

2.4.1. Data – the Non-materialistic Material

Implicit acceptance of data as a material

Due to its usage in digital environments, data is widely viewed as a material such as wood or stone. This implies certain materialistic characteristics and a way to treat it that is based on our common experience with reality. This circumstance has its origin in our often subconscious understanding of it. Already the phrases *data handling*, *data processing*, or *data mining* implicate that data is widely recognised as a basic, materialistic resource. The used words originate in crafting or other physical work.

Data, though, is immaterial, and *disembodied*. Its physical shape, the modality it is represented in, does by no means determine or affect its content; even more, data *is* pure content. Neglecting this fact, data mining and data analysis handle data as a material: they process, analyse, and shape it like other work fields process, analyse and shape stones. Nevertheless, the nature of data being a "non-materialistic material" has some inherent features, marking it different to material in the common sense. One of these features is that a data set is not bound to one phenotype. This implies that a change of its modality does to no extent change the data itself: the text of a written book contains no other information than the same text represented as bits and bytes on a hard disk. A change of representation does, however, change the way people perceive a data set, and therefore the data-inherent structure they are able to identify. This is due to the complex interplay of the data's representation and its perception by the human. So, data is independent of its representation type, but it is nevertheless bound to one (arbitrary regarding its meaning) physical representation.

Representation duality

If this representation is well-suited for an algorithmic processing by computers, it is – most of the time – not in a form that supports human perception or structure recognition. The reason for this is not that the machine-oriented representation is too complex to understand, moreover the pure physical representation (binary values coded to voltage in semiconductors or magnetic forces on hard discs) is completely inappropriate for the human senses and cannot be decoded without appropriate tools.

3. Exploratory Data Analysis

Human intelligence, particularly imagination and creativity prove to be important resources in everyday live. Intelligence enables us to discover unexpected structures, find previously unknown coherence, and gain new insights; all based on our knowledge of the world. These cognitive abilities can be seen as a result of human creativity, a unique way of thinking and creating, an ability to connect previously unconnected aspects of the world. Darwin's theory of evolution supports the argument that the characteristic shape of our imagination is based on the physical conditions of our immediate vicinity. Therefore, we are well-trained to unveil and manipulate structures in our everyday environment; spatial thinking and imagining future scenarios arising of current situations are easy for us. This can be exemplified with the help of the examination of a hypothetical passing of a hard-to-find pathway in the hills: We are not only able to easily spot its almost hidden course, but we can also manage to walk it without stumbling, and can at the same time anticipate how it possibly continues. This capability is the result of the harmonic interplay between the evolutionary deployed combination of body and brain.

However, the imagination respectively recognition of abstract structures in both theoretical and especially mathematically formalised spaces is considerably more difficult for us. An example for this is playing chess, a board game that requires thinking on a highly symbolic and abstract level. This game is generally accepted as difficult and highly complex, incorporating theoretical and non-linear thinking. Nowadays, it is played far better by computers than by humans, although these computers cannot be considered as intelligent, or even more intelligent than the people they beat in chess.

The usefulness of EDA

Disregarding the circumstance that we do not have a considerably good performance in the analysis of abstract data and algorithms, we still have to invest a huge and constantly growing part of our time into their analysis and exploration. Such tasks arise e.g. from working with pre-recorded data from investigations in a supermarket, astronomical measurements, computer programming or from the work on a quantum-physical experiment that helps us understanding our environment. The recorded data lack a physical representation that is easy to perceive. The measurements are, after all, primarily composed of abstract numbers attached with a description and a certain (probably not linear) correlation to each other. Our intent then is to find these structures and use them for decision making.

Riemanns Manifolds and its impact on data mining

As described in Chapter 2, the view on data, abstracting from its semantics (i.e. its description) can, according to Riemann, be interpreted as a manifold that is embedded into a high-dimensional vector space [Rie68]. Although this abstraction turns the data even more into the abstract, this techniques is essential for an algorithmic processing, which is necessary to either perform an automated analysis regarding known structural features, or prepare it for a representation that allows people to use their innate pattern recognition capabilities. This automated and computer-assisted information retrieval is called data mining [BH03] [FPSSU96]. It can be differentiated into *Confirmatory Data Analysis* (CDA) and *Exploratory Data Analysis* (EDA) [Tuk70].

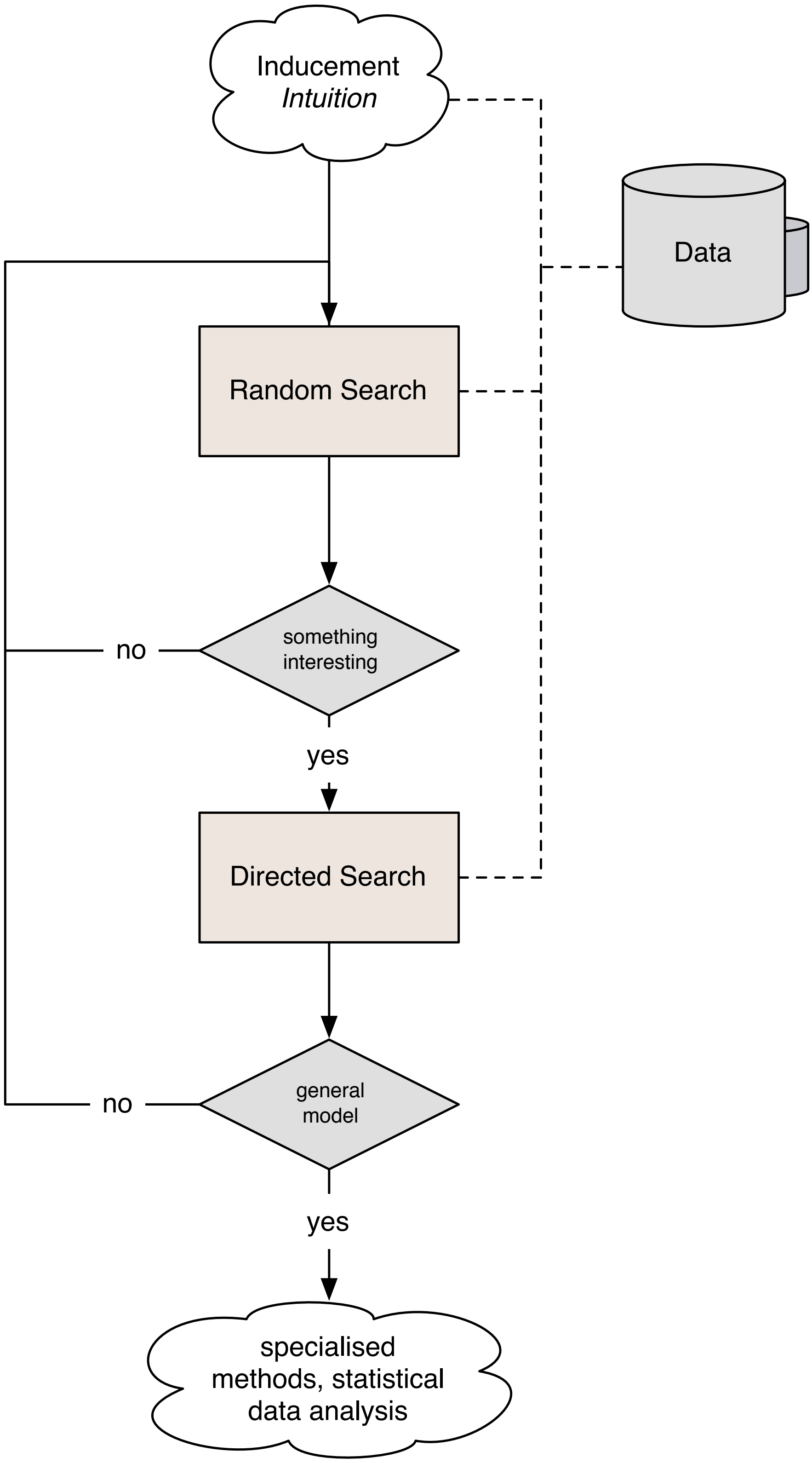


Figure 3.1.: Flowchart of Exploratory Data Analysis.

It is the aim of EDA to invent and further develop virtual tools and artefacts that can be used for data exploration. Therefore, one of EDAs dedicated tasks is to actively support human creativity by representing abstract data and algorithmic processes in such a way that is in line with the particular requirements of the human structure recognition system. One very common approach is the visual representation of data. It embeds otherwise abstract data into a humanly well-perceivable form, adding the possibility to use the visual sense and its characteristics in order to recognise structure and information. Other approaches use auditory and physical representations [BHR05] [HBRR07], or develop a completely virtual data space surrounding the user in a multi-modal fashion [dHKP02].

3.1. Workflow in Exploratory Data Analysis

Due to its exploratory nature, the EDA workflow tends to vary a lot. In order to still convey a general impression on how Exploratory Data Analysis usually works, a typical iteration will be described next and is also shown in the flowchart in Figure 3.1.

An EDA usually starts with data acquisition, followed by preliminary data preparation that includes steps like the elimination of missing data and sphering.[1] As a next step, the cleaned data is usually pre-processed by dimensional reduction methods (e.g. PCA [Jol86], or ICA [Com91]), or analysed for clusters [JD88] or other higher level features. The result is then presented to the user via display technologies such as visualisation or Sonification.[2] It is possible for the researcher to manipulate that representation and apply human-aided feature extraction algorithms. Usually, this workflow is repeated until an idea of a general model is discovered. In this situation, the researcher eventually changes his strategy from a random search to a directed search process. If a general model or source of structurally relevant information arises, he will leave the field of exploratory analysis in order to move on to more specialised methods that are dedicated to find significant validations for his theory.

3.2. Standard Techniques

In general, all approaches of data exploration change the representation of the data under exploration to an extent that specific features like local density, clustering or correlations can be better recognised by human perception. Main differences of these approaches, though, can be observed in (*a*) the used modalities, (*b*) the amount of possible user interaction, and (*c*) the level of abstraction from the original data. Technically, (*a*) has an important influence on how data can be transmitted technically – each medium has its specific character and therefore features – and which information can be carried out in what quality by the perceiving human. This aspect is covered in Chapter 6 in more detail. The amount of possible user interaction (*b*) highly depends on the way data is represented and how many parameters of this representation can be changed by the user in realtime. As stated before, Exploratory Data Analysis is the collection of techniques and methods to unveil unknown information and structure from data; a process in which creativity, intuition and

[1] Sphering is also known as *whitening*, and means to remove biases in all measured dimensions, followed by a variance normalisation. See standard literature on statistics like Steel [ST60] for further details.

[2] See Section 6 for more information.

curiosity plays an important role. The more direct and immediate activity is possible with data exploration systems, the better will be the results that can be achieved. The level of abstraction (c), finally, decides on the granularity of information presented to the user. In this case, eventually, a compromise between analogue and direct representation on the one side, and a symbolically higher representation that is based on automated analysis has to be found. While a quantitative and direct representation features a broad variety of possibly useful information at once that, however, may be difficult to understand since all feature extraction has to be done by the human and transfers the full complexity of the data under exploration, a more symbolic representation is based on the pre-processed and –analysed data. Although better to understand, it carries the risk to miss important structural information of which was never thought and therefore not searched for. By means of meta-controls, it is possible to change this level of abstraction to allow users to fluidly change between analogue and symbolic representations. This practice will be described for Tangible Interfaces in Section 5.4.3.

Why different approaches

As pointed out, the perception of display technologies carries specific characteristics that heavily depend on their modality. Each of them helps users to detect specific structures, whereas it does not feature other, maybe equally important aspects. It is fairly easy, for example, to use graphics and prepare data in a way that the viewer is able to recognise cluster-like structures. Such standard techniques for visual analysis are described in Section 6.2 A phase shift in a quasi-periodic signal on the contrary can only be made visible with considerable effort.

Although vision is clearly the preferred modality used for data exploration, audio-based displays gain more and more attention, since they focus on other aspects of the data under exploration. Aspects that are difficult to perceive visually. Up to now, there are no widely established standards in Sonification, however, the International Community for Auditory Displays is in the process to establish such standards [KWB+97]. Albeit, there are some promising fields aiming for attention regarding data exploration, since they are especially suited for multi-variate data. These fields include Parameter Mapping Sonification, Audification, and Model-Based Sonification. Among others, these techniques will be described in detail in Section 6.3.

3.3. Neighbour Fields

As mentioned in the last section, EDA is part of data mining. Closely related is analytical data analysis. Accompanied by statistical data analysis, it is most commonly used to get valid and verifiable results for structural relations. First hints into this direction are often found with the help of EDA. Other fields related to EDA are computer science and display technologies. They provide the algorithmic theory respectively hardware to be used in EDA applications. HCI and psychology, finally, are needed to get information on how a data exploration system should be designed in order to fulfil human requirements.

Apart from these direct interconnections, data monitoring is also related to EDA. It uses the same methods and mechanisms, but for a different purpose: While data exploration aims for new insights into (probably already known) data domains, data monitoring tasks are intended to increase the *perceivability* of data features. These features are, however, often already well understood. In a medical context, for example, data monitoring is often needed

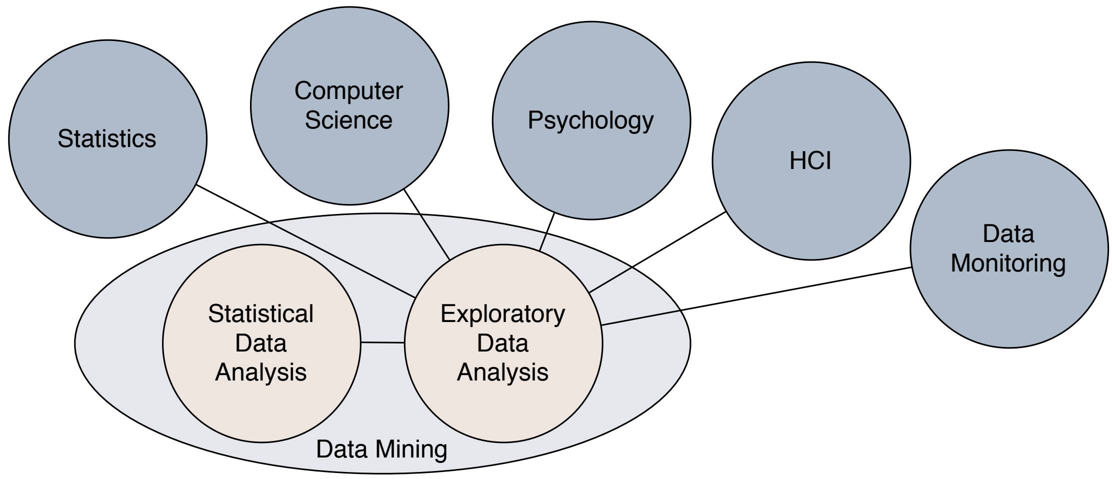


Figure 3.2.: Fields related to Exploratory Data Analysis.

to warn people according to the situation of patients. This situation can be supported by representation systems that represent e.g. EEG data by sound or vision that integrate into the staff's ambient environment.

3.4. Data Representations

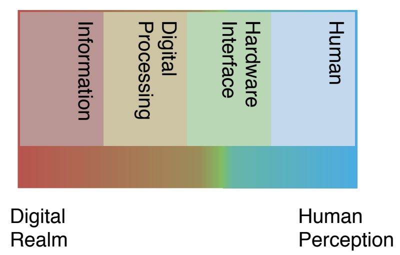


Figure 3.3.: Data transcribed from a digital storage to human perception has to go through several layers.

All data is represented in a certain way. The form of its representation hereby heavily relies on the context in which it is intended to be used. In a digital environment, for example, data should be optimised for digital data processing, whereas in a human related environment, data should be optimised according to human perceptual skills.

A digitally optimised data representation can be fully described as a valid element of a superset of symbols of a predefined alphabet. Take for example data from a digital photo camera that has to be saved for further usage and processing. This is done by filling a list of *Integer* variables with values ranging from 0 (black) to 255 (white) according to the brightness of points in the photography.[3] To be perceived by a human, however, the digital data[4] has to be transcribed into a human perceptible representation. For this, it has to pass at least one *digital* processing stage (usually a software that turns the value of a

[3] For the sake of simplicity, only the case of monochrome image processing is covered, data structures for coloured images are more complex, but are based on the same principals.

[4] *Digital data* actually is a wrong term: it is not the actual data that is digital but its form of representation. To not impede the reading flow by over-complex terms, I consider the term *digital data* to be an equivalent to *digitally represented data.*

list item into the value of a virtual pixel) and a *hardware* interface, which transcribes the prepared digital data (the pixel value) into a human perceivable event (the brightness of one point per list entry on the display's surface). This data transcription process from the digital realm to a perceptual stage is exemplified in Figure 3.4.

When examining data representations, it is essential to separate the following terms:

Data transcription is the formal act of moving data from one medium to another.

Data representation is how data is stored.

Data perceptualisation is how data is perceived.

Data transcription

Data representation

Data perceptualisation

An example for data transcription is the act of copying data from a hard disk to a DVD, but also the act of printing a visualisation of formerly digitally stored data to a sheet of paper. Each stage, be it the digital processing or the hardware interface, introduces specific properties that may hide, emphasise or even omit parts of the original data-inherent information. Each of the forms in which the data appears in these examples is referred to as data representation, i.e. their visual apearance on paper, their structure that is linked with the optical representation of bits on the DVD, or the structure and magnetically stored bits on the hard disk. Data always is represented with the help of a medium; its representation can be decoded with help of a (sometimes implicitly available) grammar. Data perceptualisation, finally, describes the way data is perceived. The perception heavily depends on the current representation and the perceiver. The perception process includes the perceiver's interpretation as well as his abilities for structure recognition and information retrieval based on the particular data representation. It also covers the perception of representation-inherent artefacts and their potential misleading.

3.4.1. Representation Classifications

Requirements

This section discusses and proposes indicators that may be used when describing data transcriptions, representations or perceptualisations. For this, I propose that a description of a data transcription into a new representation form should include information on

1. the incorporated sensorial modalities,
2. which data-inherent structures are emphasised,
3. the level of interaction between user and representation system,
4. the level of reality (as described in RBI, see Section 4.5.1),
5. the supported and preferred types of input data (e.g. sequential or cartesian),
6. the percentage of passed-through information, and
7. the symbolic level, i.e. whether high- or low-level symbols are used.

The requirements 1 through 5 are – given a specific transcription – more or less easy to deduce from the technical parts of the representation system. The requirements 6 and 7, though, need a closer look. In the following, I describe classification systems and other approaches that can be used as indicators for these parts. Because of the focus of this work, I chose most of them because of their close relation to Auditory Displays.

Sloman's *Analogical* and *Fregean* Representations

Sloman explains in his *Afterthoughts on Analogical Representations* the difference between *Analogical* and *Fregean* representations [Slo75]. While he defines analogical representations to be complex representations of complex data, obligatory having a structure that corresponds to the structure of the represented, Fregean representations do not need to have an obvious correspondence to the data's structure. For Sloman this especially means that the interpretation

> Analogical representations are continuous, Fregean representations discrete

as cited in his paper is a misinterpretation, because

> [there are] examples of discrete analogical representations, e.g. a list whose elements are ordered according to the order of what they represent.

However, a differentiation between continuous and discrete streams of information representations is often obvious in human-computer interaction contexts.

Kramer's Analogic/Symbolic Chart

A similar approach based on Sloman is Kramer's *Analogic/Symbolic placement scale* [Kra94], in which he claims that

> [an] analogic representation is one in which there is an immediate and intrinsic correspondence between the sort of structure being represented and the representation medium. The relations in the representation medium are a structural homomorph of the relations in the thing being represented. A change in the representation medium [. . .] has a direct correspondence with the thing being represented [. . .],

whereas

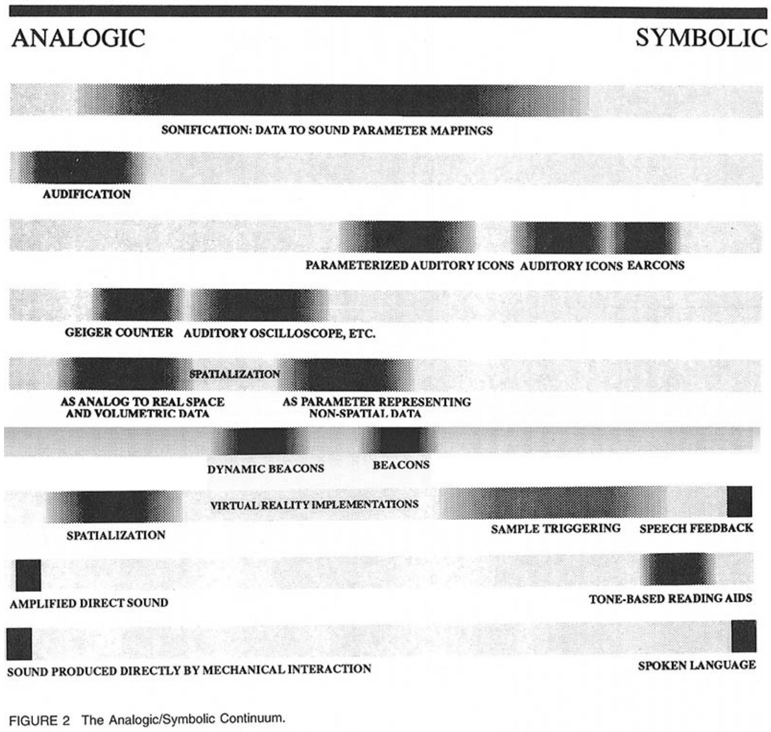
> [b]y symbolic representation we refer to those display schemes in which the representation involves an amalgamation of the information represented into discrete elements.

Kramer proposes that – in difference to Sloman – the classification and differentiation of Sonifications into his system is *continuous*. He proofs it by filing representative examples for Auditory Displays into his classification system. Although a continuous mapping space, Kramer's analogic/symbolic chart does not cover the above-mentioned, seemingly natural, discrimination between discrete and continuous data representations.

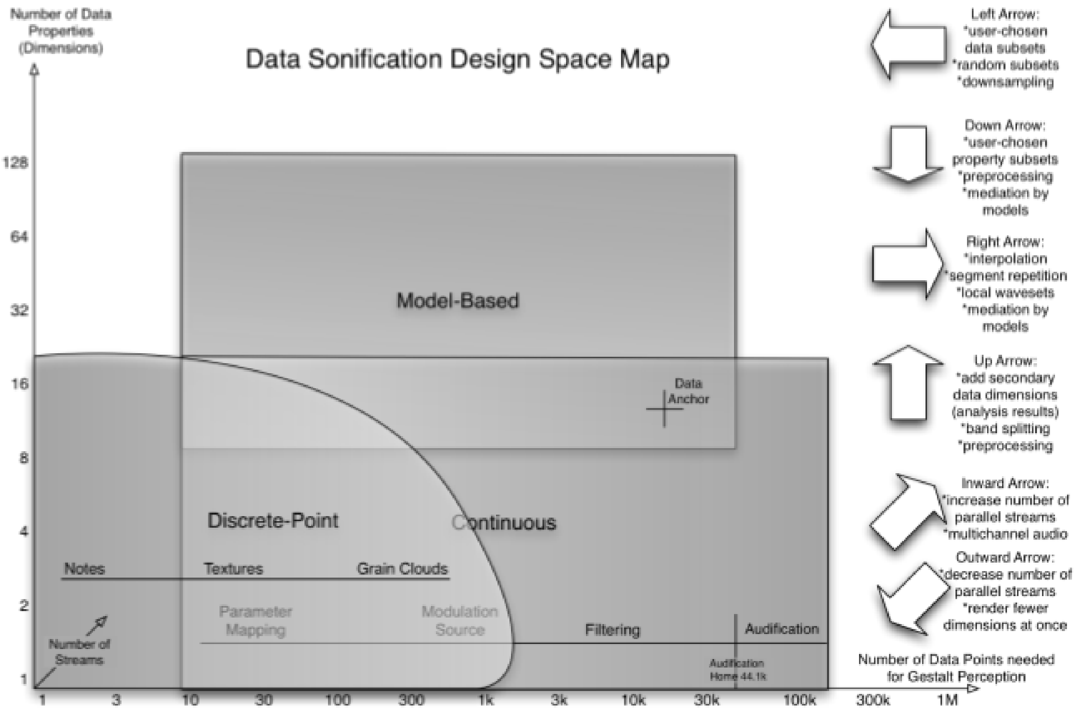
De Campo's Sonification Map

A third theory to classify – purely sonic – data representation is described by de Campo in his PhD thesis [dC09a]. The there-introduced *Sonification Design Space Map* (*SDSM*) draws a three-dimensional figure on how sound and meaning can be connected to render a sonic data representation. The aim of the SDSM is less to analyse existing data representations, furthermore, it should support to

> find transformations that let structures/patterns in the data (which are not known beforehand) emerge as perceptual entities in the sound which jump to



(a) Kramer's Chart on Analogic/Symbolic Continuum [Kra94].



(b) DeCampo's Sonification Design Space Map [dC09a].

Figure 3.4.: Schematics for data representations.

> the foreground, i.e. as identifiable 'interesting audible objects'[. . .]

Therefore, the SDSM can be used

> to achieve improvements to solve the most general task in data Sonification designs for exploratory purposes [, namely] to detect auditory gestalts in the acoustic representation, which one assumes correspond to any patterns and structures in the data one wants to find.

De Campo's intend can therefore be entitled as to guide the design of a data representation process in such a way that it fits the needs of the researcher. In his ibid. developed Sonification designs, he points out paths through the SDSM rather than concentrating on fixed points, which enables him to describe the actual design process as a continuous and intentional series of decisions based on user experience and the goals of the resulting system. This massively increases the usability of the *SDSM* and introduces an indicator for changing what I call the *level of abstraction* of a Sonification. Together with the definition of a level of abstraction for Tangible Interfaces (as described in Section 5.4.3) this forms a powerful toolbox for Tangible Auditory Interfaces.

3.4.2. Considerations based on the presented classification strategies

In the last subsection, an overview on common techniques to represent data and its structure was given. With Sloman's analogical-fregean, and Kramer's analogic-symbolic ranges, we get two closely related indicators that are based on subjective interpretations of the representation under exploration, since they rely on the characteristic of the information perceived by the human. De Campo's SDSM on the other hand introduces descriptive dimensions like the number of data points, the number of data properties, or the number of audio streams. Their combination is used to indicate an appropriate representation method. This strategy elegantly avoids the need to classify these methods according to their information preservation. Instead, the choice of Sonification strategies (and also strategies that include other modalities) is based on the experience of experts.

Many researchers, however, would prefer to actually use quantitative measures to compare representation techniques with each other in order to make decisions regarding their quality. Unfortunately, already the computation of the norm

$$\|\cdot\|_{\mathcal{S}} : \mathcal{S} \to \mathbb{R} \tag{3.1}$$

with $\mathcal{S}$ the set of all representations, determining the valuable information in that representation is impossible. At least when humans are incorporated into the perceptual process. The individual information content of a data representation is highly subjective; only when the states of all incorporated systems are known, the actual information content of a representation can be determined. In the case where the quality of a representation is based on human perception and analysis, only estimations based on quantitative and qualitative evaluations can be made. In these cases, it still remains difficult to generalise the performance of individuals. This makes indicators for e.g. the level of detail of a representation or its information to noise ratio unreliable.

Another aspect speaking against a quantitative measurement at least for representations that are intended for Exploratory Data Analysis is the aim of these representations: Estimations based on quantitative measurements abstract from the participants' individual performance in favour for a better generalisation. Although this might be effective in situations where the majority's performance is relevant, it does not make sense for explorative situations, in which it is essential to find *any* hint on any structure. If only one person can effectively use the representation to unveil unknown structural information, this has a significant impact and is considered as relevant. The general performance of the prototypical human backs out in favour to the individual. Qualitative methods, e.g. those based on grounded theory (as it will be described in Section 4.6 in more detail) are able to emphasise these aspects. They also build a solid basis for the analysis of the indicators by Sloman, Kramer and de Campo.

4. Interfacing Humans with Computers

Dealing with information means to operate on it and experience it. The best computing algorithm is useless without any interfacing; data has to be fed into it, and the results have to be shown, otherwise its operation does not make any sense.

Overview

In this chapter, I give a short overview on origins of interfaces that help to transfer human action to the digital realm as well as the other way around, where information has to be transferred from the digital into a human perceivable form. The first part of this chapter therefore is about observation and analysis of human action in reality and the origins of interfaces as they are ubiquitous today. This is followed by an introduction to Human Computer Interaction and Interaction Design, the central research fields for interfacing between humans and computers and an introduction to Graphical User Interfaces, the current state of the art technique for such interfaces. Since there are widely known limitations to this approach, I will then look at alternative approaches like Augmented Reality, Virtual Reality, and Tangible Interfaces, which can be integrated into the term of Reality-based Interaction. I reflect the idea as it was introduced by Jacob et al. and comment on it. The chapter ends with a reflection on evaluation methods for interface design.

4.1. Observation and Analysis of Human Action in Real-life Situations

When we look at our handling of digital information, our usual approach is rather limited. Limited to only a few general purpose interfacing devices. These input devices have separated functions: one is primarily used for symbolic input (usually a keyboard), and another one for pointing and manipulating on a more continuous level (typically a mouse). The complete system can be used to manipulate virtual elements like texts, images or visual representations of sound files, as long as they and their manipulation can be visually represented on a screen.[1] These input modalities add an abstraction layer between user and content that is the visual representation of text and controls. It cannot be touched and manipulated directly and is only accessible via a mouse or keyboard. This seems to be odd, but many people are so much trained to use only keyboard, mouse and screen for their day to day work on digital material that they do not consider it as a drawback or bottleneck. While complex, it is nevertheless possible to represent a specialised interface of a software system completely with virtual elements. Additionally, such a system is cheap and modular, and can therefore be recycled (the hardware interface) and copied (the software modules). Therefore, there is no need to buy or build expensive custom hardware. These Graphical User Interfaces (GUI) are populated by (virtual) copies of hardware machinery

[1] Although sound is also available at many workplaces, it is not commonly used in an interactive fashion, but more as a music playback device.

Figure 4.1.: Chatting people at the Grand Opening of the CITEC Graduate School in July, 2009. An example for Human-Human Interaction.

interface parts like sliders or buttons. These are popular and effective control interfaces for machines and probably originate from these. But reality has more to offer regarding control, interaction or manipulation, and while information is ubiquitous in western society and the operating on and with this information is an essential part of our day to day work, only few techniques found their way into interface design.

In the following, I will describe two scenarios in which people operate in and on their natural environment. These observations are intended to argue by their observed richness for the importance of research in human-computer interfaces and, in particular, for a richer and broader approach on human-data interfacing technology. They should be designed to actually make use of the insights and influences given by the observations of people's action, reaction and interaction with reality.

4.1.1. Human-Human Interaction

Inspired by the image of chatting people shown in Figure 4.1, we can identify two types of building-blocks of social interactions: the sensing and the acting part. While sensing is established by the five human senses – hearing, sight, taste, smell and touch – acting can be differentiated into speech, gesture, mimic and action.

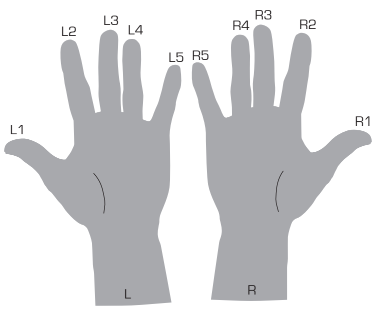


Figure 4.2.: Finger naming.

This complex system of interaction between humans is currently only understood in parts [Par37] [GDA91]. However, it is well-known that there is a direct coupling between interacting partners, established by common *interaction building blocks* as co-ordinated actions and reactions [GDA91]. The observation of such social behaviour can lead to deep insights for complex interfacing systems with algorithmic functionality. On a symbolical level, the artificial intelligence program *Eliza* by Weizenbaum [Wei66] managed to mirror social behaviour such that it was difficult to determine if it was a human chatting, or a computer. This is the intended behaviour that should be also possible on a subsymbolic level.

Figure 4.3.: Video stills in which Andy Goldsworthy explores leaves while crafting an art-piece [riv]. He sits under the tree from which the leafs are originating, assembles the artwork, and places it back to the tree. The second row shows stills from the sequence that is analysed in the main text.

4.1.2. Manipulating Objects

What can we take for human computer interface design from our day to day action on objects in reality, especially in natural environments? To investigate this question, I analysed a short video sequence, in which the British sculptor, photographer and environmentalist Andy Goldsworthy is shown working on an art-piece that is completely made of leafs and thorns [riv] (see also Figure 4.3). Goldsworthy's leafs are a good example for the big potential of coming tangible data exploration systems for three reasons: First, Goldsworthy is a skilled person, exploring material he is generally familiar with. Second, details of the material, i.e. the particular leafs that are in his hand and there actual configuration are unknown to him before he starts to explore them. Third, the activity of sorting and selecting leafs is unfamiliar to most of the observers (including me), the material. Nevertheless, it is sufficiently well known, making the exploration process comprehensible. This aspect supports an observation that is not too much bound on high-level symbolic meaning, as it would be the case when analysing a worker painting a wall. In the examined sequence, Goldsworthy works his way through a pile of leafs next to him. During this process, he masters very complex operations with his fingers; He for example feels their quality and selects some to use them in his sculpture. I claim that the analysis of his rich interaction patterns is beneficial for interface design considerations concerning future data exploration systems. I therefore underwent the video clip a qualitative visual analysis. The schema shown in Figure 4.2 hereby shows the finger coding used below.

Flip gesture

At 49.5s: The flip gesture is a fast scanning through the pile of leaves. The gesture starts by having leafs in each hands: One leaf a is in the left hand (L) and hold with the thumb (L1), while two other leafs (b and c) are stacked and hold with R1 in the right hand (R).

1. Let go leaf b with R1.

2. Move L1 between leaf *b* and *c*.
3. Align L1 with index finger such that leaf (b) now lies aligned to leaf *a*.
4. Move L2 such that leaf *a* and *b* lie together and are hold between L2 and L3.

The complete gesture takes about 2 seconds and is often repeated to scan through the pile, sometimes in the following variants.

Variants At 63.5s: Flip gesture with leaf. Goldsworthy picks a leaf and stores it between L2 and L3. This alters the gesture such that the previously performed action of L2 is now executed by the combination of L2 and L3, whereas the L4 is now responsible for the movement previously performed by L3.

At 65.0s: Flip gesture with changed hand roles. Goldsworthy alters his flip gesture such that the right hand (R) now has the part of the left (L) and vice versa.

Index finger flip At 77.5s: Goldsworthy puts R2 between the leafs and flips it repetitively from right to left, each time flipping one or more leafs from one side to the other. Each iteration takes about half a second. This gesture really looks like it is easy to perform. It is supported by the relatively big gap between the single leafs that is caused by their waviness.

Selection and texture test At 73.5s: Goldsworthy takes up a leaf with his right hand (R) while slightly rubbing it between R1 and R2. He may feels its structure/texture and then decides to keep it.

At 79.5s: Take and select the leaf next to L2. It is not visible (also not to Goldsworthy). He therefore uses only his tactile sense for the selection process.

Remarks Based on this analysis, I argue that Goldsworthy used his experience on the specific structure of leafs, not only for his sculptures, but also for their formation and building process.

Observations Action and inter-action in reality, possibly incorporating other people or objects is often a complex attempt. Goldsworthy's use of his fingers to sort, identify and select leafs gives us a hint of this circumstances. The complexity of his movements and their variety, though, are fundamentally different from the common manipulations we physically apply to interfaces designed for data processing. In the short example, already four different manipulation gestures can be identified. Sorting the leafs involves many subconsciously performed tasks and analyses of peripheral information. Not only their size or colour are of interest, but also their texture, their material, quality or stiffness. It is difficult to cover this highly direct coupling of sensing and understanding in one word, the German word *begreifen* as a polyseme for *to touch* and *to understand* may fit best.

In difference to the observed behaviours and actions of Goldsworthy, today's typical environments for data processing and exploration use only a rather limited part of our interaction and manipulation skills. In difference to Goldsworthy's leafs, they often feature a symbolic interface to mediate the users intends and the data representation, which, in addition, is uni-modal most of the time. It is an aim of this thesis to apply techniques to interface design learned from the leafs example.

4.2. Historical Considerations

One origin for the design of human-computer interfaces is surely the traditional tool use as it was viewed as a common and effective method to work on material such as wood or metal for centuries. The mechanisation of these working fields then required systems to trigger and control the evolving semiautomatic tasks. Trucks and motorised diggers, but also smaller tools like electric toothbrushes have controller boards and mechanical, pneumatic or electric switches, levers, or hand wheels to make automated action possible and controllable. The increasing tendency to miniaturisation transformed previously mechanical systems into electric and finally electronic devices. The user interface, however, remained roughly the same, such that today, we operate complex machines with interfacing technology having substantial origins in a long history. All interfacing sensors capture their degrees of freedom, and they feed the resulting information as control parameters into the digital process under control. Physical appearance thus turned into virtual (mostly graphical) symbols. The Graphical User Interface was born.

Analogue Computing

But there were also systems invented for mechanical data manipulation, long before electronically supported computation was invented. Due to their mechanical construction, their form directly corresponded to attributes and characteristics of the incorporated data. This makes them interesting concerning the design of new Tangible Interfaces for data exploration. In the following, I discuss two such machines; the slide rule and the planimeter. Before delving into these two examples, a short overview on analogue computing in general is given.[2]

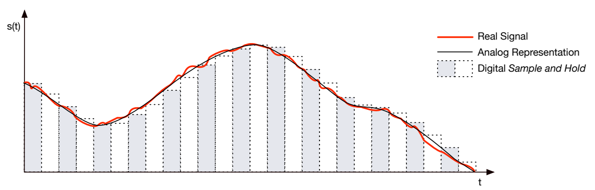


Figure 4.4.: Difference of continuous and discrete variables as they appear in analogue and digital systems.

Analogue computing has two distinguishing characteristics: *parallelism* and *continuity*. *Parallelism* means that operations can be performed in a truly parallel manner, i.e. many calculations are computed at the same time. Therefore, even sequenced modules can calculate their result in true realtime, making it easy to implement features like feedback control of sensor-actor systems. *Continuity*, in difference, is the use of continuous variables/parameters: Analogue Computing devices change their state not in discrete steps, but in a smooth and continuous manner. In contrast, a digital computer performs operations sequentially and operates on discrete numbers represented in floating point or integer values. Figure 4.4 exemplifies the difference between analogue and digital systems by means of typical signal representations. Although both types induce representation-specific artefacts, their nature are inherently different.

4.2.1. Slide Rule

The *slide rule*, invented by William Oughtred and others in the 1600s, is a mechanical analogue computing device that makes use of logarithmic scales to support numerical multiplication (see Figure 4.5). It is based on the work on logarithms by Napier [Nap19]. It is elongated and comprises of two main parts: an angular rod and a rail that guides its movement to be in parallel to its longest axis. On both elements, logarithmic scales are printed. Many slide rules additionally feature a glass window with either one or three

Hardware and functionality

[2] Much of the information on analogue computing originates from the excellent web-site *The Analogue Computing Museum* [Cow00].

hairlines printed perpendicular to the rod's movement. The window can be moved parallel to the rods movement on the rail. By adjusting the scales to each other, multiplications can be performed. (a) The first multiplication factor (on the rail) has to be lined up with the number *1* on the other scale. Figure 4.5 shows such a configuration for π. (b) The multiplication's result then can be read off the other scale: it is the number facing the second factor. Note that the slide rule makes it easy to compute all products for one given factor. This is achieved by glancing at the varying factors for each calculation, remaining the other factor fixed according to (a). The mechanical computing process is truly parallel, only the reading is done sequential.

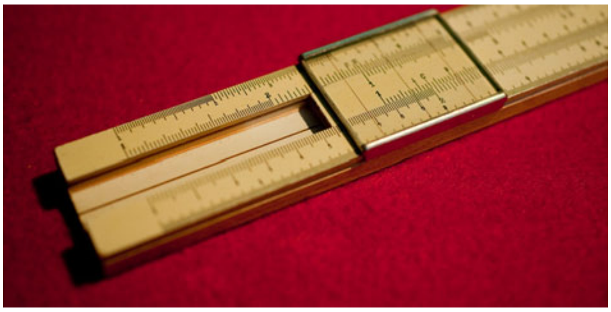


Figure 4.5.: A slide rule. In its current configuration it can be used to read off all results for $f(x) = \pi x$. The hairline on its sliding window indicates that it is used for $x = 1.16$.

Calculations using the slide rule are of limited precision due to their analogue inputs and outputs and possible mechanical imprecision. Conversely, because of the discrete numerical input and floating point electronic operations, even modest modern calculators have output resolutions of at least six significant figures. However, a slide rule tends to moderate the fallacy of false precision and significance. The typical precision available to a user of a slide rule is about three places of accuracy. This is in good correspondence with most data available for input to engineering formulas. When a modern pocket calculator is used, the precision may be displayed to seven or more decimal places, while in reality the results can never be of greater accuracy than the input data available.

Tangible experience

A slide rule features a characteristic tangible experience which heavily depends on the used material. Slide rules are build either from wood, metal or plastics. The mechanics – if such a simple mechanism can be named as such – of a well-built slide rule does make the user feel a smooth friction, allowing him to easily adjust the relative positions of the rod and the window on the rail. This is an important feature, which, although not affecting the general functionality, adds a substantial value to its operation because it supports an exact positioning of the parts to each other. Furthermore, slide rules do not depend on electricity and are, due to their mechanical nature, easy to replicate. From a given example of a slide rule, more can be constructed by a competent craftsman from rudimentary materials using non-industrial processes.

4.2.2. Planimeter

Hardware and functionality

A *planimeter* is an instrument that uses geometrical features to compute the area of graphically represented planar regions by tracing their boundaries. A planimeter has two arms with a freely moving elbow where one arm is a fixed to an anchor point. A needle traces the boundary of the region to be measured, while moving a wheel in the elbow, whose orientation is perpendicular to the elbow-to-needle arm. The net distance rolled by the wheel is exactly the same as the contoured area.[3] Planimeters exist in various forms, and can be classified into two main groups, (*a*) purely mechanical and (*b*) digitally enhanced ones. Their basic computation system is based on Green's Theorem, which proofs that the tangential line integral of a vector field around a curve equals the double integral of the curl of that vector field. Thus the distance travelled by the rolling wheel equals the double integral over the region of the curl of the relevant vector field.

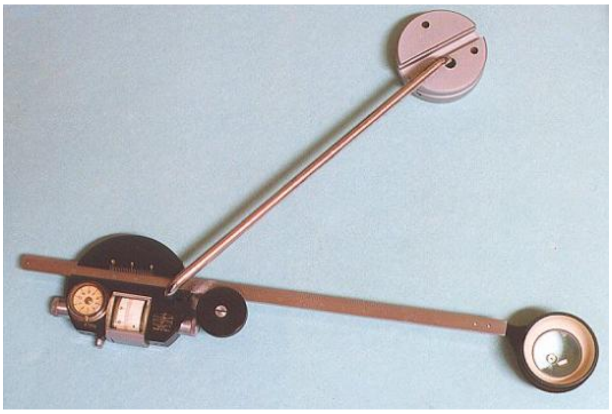


Figure 4.6.: A mechanical planimeter by the Gebrüder HAFF GmbH.
See Section 1.1 for © information.

Digital planimeters use the same technique to measure areas of arbitrary shape, but can also be used to perform other measurements, e.g. lengths, and volumes using contour lines. Their mechanism and mechanics, however, are exactly the same as used in analogue planimeters: only post-processing, interpretation and display of the measured values are done digitally by an integrated circuit board and are displayed by a digital display.[4]

Conclusion

Both planimeter and slide rule are good examples on how physical relations and features of our environment can be utilised for manipulation and computation of abstract data. Thus, one goal of this thesis is the promotion of these techniques to be integrated into Tangible Interfaces, helping to blur the gap between abstract data manipulation and reality with its rich feature-set.

4.3. Research in Human Computer Interaction and Interaction Design

Allmost all major improvements and insights into human computer interface design can be attributed to members or founders of the research community of *Human Computer Interaction* (HCI) or *Interaction Design* (IxD). One prominent example for this is the common graphical user interfacing originating from Stanford Research Institute (Engelbarts mouse-based hyper-link traversal for *NLS* [Eng62]), Xerox PARC (*PARC User Interface* [TML+79]) and MIT (Sutherlands *Sketchpad*) [Sut63]. The involved people are often called the founders of HCI. Other research fields are Shneidermans *direct manipulation* [Shn83], Ishii's *Tangible User Interfaces* [IU97], virtual and augmented reality originating in the sensorama by Heilig and coined by Lanier and Caudell [Hei92], and *ubiquitous computing*, a term phrased by Weiser [Wei91].

[3] Description according to "The Polar Planimeter" from `http://demonstrations.wolfram.com/ThePolarPlanimeter/`. Contributed by: Bruce Atwood (Beloit College) and Stan Wagon (Macalester College)

[4] For more information on the functionality of digital planimeters, please refer to `http://www.haff.com/digitalplanimeter_e.htm`

Both fields, HCI and IxD, reportedly see their focus on interface design research. Unsurprisingly, they are closely related to each other. However, HCI aims to analyse interactions between user and machine, and analyses existing technologies, whereas the focus of IxD is more on the actual interface design. IxD provides methods and guidelines for the concrete interface design, establishing itself as a more constructive discipline then HCI. In the next two paragraphs, a brief overview of the two areas will be given.

Interaction Design

Interaction Design (IxD) is a design-oriented research field. Many involved people are actually working on concrete interface designs for professional software. A good classification of the field is given by its description on the webpage of the *Interaction Design Association*:

> Interaction design (IxD) is a professional discipline that illuminates the relationship between people and the interactive products they use. While interaction design has a firm foundation in the theory, practice, and methodology of traditional design, its focus is on defining the complex dialogues that occur between people and interactive devices of many types – from computers to mobile communications devices to appliances. [...] Interaction design defines the structure and behaviors of interactive products and services and user interactions with those products and services [ixd].

Aside from HCI, IxD is also closely related to product design, industrial design, sociology and psychology. Especially the books by Verbeek [Ver05] and Dreyfuss [Dre03] had a substantial impact on this field, though targeted more on industrial design. Concerning this thesis, IxD contributes methods and guidelines for the conceptualisation of Tangible Auditory Interfaces.

Human Computer Interaction

Human Computer Interaction (HCI) is defined in the *Curricula for Human-Computer Interaction* [Hew92] as

> a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them.

It was founded by Carrol [Car09] and Myers [Mye98]. *HCI*'s contribution to this work is its provision of methods to be used in the analysis of Tangible Auditory Interfaces.

4.4. Graphical User Interfaces

Graphical User Interfaces visuo-haptically close the input/output loop of human computer operation by abstracting from the actual interfaces via a so-called *desktop metaphor*. They support a modular software design via the *Model-View-Controller* concept [Ree79]. Common input modalities and their integration into the GUI concept are tactile (keyboard, mouse), sound (microphone) and vision (camera), whereas output is typically mediated via graphics (monitors) and sound (speakers/headphones) [Mye98].

Critics

GUIs feature one general pointing device (typically a mouse), which serves as a meta-tool. It allows the user to operate virtual tools like selection frames or erasers to manipulate data. This concept relies on an abstraction layer between the user and the content, which allows to completely shift the actual interface and specialised operation of a software system into the virtual, while the actual physical interface and its operation is independent from the toolset and data manipulation process. To achieve this, GUIs are populated by virtual emulations and look-a-likes of common machine interface parts: virtual sliders or buttons are used that replace their hardware counterparts, fully operational with one pointing device. During the GUI high time, this was considered as a benefit, since the alternative – custom-built physical artefacts – was too expensive and possibly more prone to errors. In short, the generic pointing device made additional interfaces unnecessary, resulting in advantageous and modular set-ups that are easy to recycle and copy.

Consequences

Although there are benefits of GUIs, e.g. their mentioned general purpose design or there extreme modularity, which makes them easy to implement, they also have drawbacks, especially drawn from their relation to a fully virtual abstraction layer. It is for example very difficult if not impossible to establish an indirectly useable interface, enabling users to subconsciously focus on the operation of the system. This is not least because of the fact that GUI based interfaces lack a meaningful physical feedback; they are virtual abstractions of real and specialised hardware. Also, specialisation, e.g. for accurate work on graphical representations are only possible to a certain degree. Almost everywhere, where people rely on the operability of data functionality, they use specialised tools like graphics tablets (for drawing and photo processing) or fader boxes (for music production). This leads to the thesis that in certain application areas, a general purpose GUI is not sufficient for the appropriate use of the underlying system.

One important indicator for the fact that there is much potential still to be unveiled is the following: Principally it is possible to control and access current interfaces without more then two fingers and one hand: A mouse of a slightly different shape can be controlled easily by the palm, and also the keyboard we usually utilise to type text into computer programs does not require more then one or two fingers (for *shift*, *control*, or *command*) to be operated efficiently. Also, for typing and pointing, no special skills are needed such as dynamically adjusting pressure to objects in our hands.

One goal for today's research on HCI and IxD therefore is to find ways to find data representations that allow their differentiated handling and are open for newly established strategies during the operation. Operating such an interface may look similar to Goldsworthy's leafs as described in Section 4.1.2, however behind this operation, a complex data manipulation and exploration may hide. This circumstance is a starting point for many interface research fields, of which I will give a brief overview in the following section.

4.5. Alternative Approaches

GUIs are not the only approach used today to interface humans and digitally stored and manipulatable data. Many other fields like *Virtual Reality* (VR), *Augmented Reality* (AR), *Ubiquitous Computing* (UbiComp) or *Tangible Interfaces* (TI) were developed to overcome their mentioned drawbacks.

VR & AR

Virtual and augmented reality both originate in an interface for cinematographic application, the *Sensorama* by Heilig [Hei92]. Lanier then uses the term Virtual Reality first in an interview for the *Whole Earth Review* in 1989 [KHS89]. Coined by Caudell (during a stay at Boing) and described further by Wellner, *Augmented Reality* is meant as the opposite of VR [WMG93]:

> Instead of using computers to enclose people in an artificial world, we can use computers to augment objects in the real world. We can make the environment

> sensitive with infra-red, optical sound, video, heat, motion and light detectors, and we can make the environment react to people's needs by updating displays, activating motors, storing data, driving actuators, controls and valves.

UbiComp

Ubiquitous Computing on the other hand is a term by Weiser. He explained it in his article on *the computer for the 21st century* [Wei91]. His utopia of computers viewed them as parts of many artefacts surrounding us, thus fading into the background, and only add their specific value to the object's inherent functionality. As central elements of UbiComp, Weiser introduced *Tabs*, *Pads* and *Boards*; elements of different size, which provide access to information and algorithmic functionality either on a personal– , group– or public level. Technically, they mainly differ in their size and functional elements.

Tangible Interaction

Ishii's *Tangible User Interfaces* [IU97] use everyday artefacts for algorithmic data manipulation. Section 5 focuses on this approach in more detail.

Unifying Theories

Besides these concrete approaches, there are also efforts to find overlaps and relation between them, resulting in unifying theories. As a representative for these efforts I next discuss Jacob's *Reality based Interaction* (RBI) [JGH+08] since it is the closest to TAI, abstracting from the used interfacing modalities.

4.5.1. Reality-Based Interaction

In their publication *Reality-based Interaction: A Framework for Post-WIMP Interfaces*, Jacob et al. introduce the term Reality-Based Interaction (RBI) as a framework that unifies emerging human computer interaction styles [JGH+08]. As examples for these styles they name

> virtual, mixed and augmented reality, tangible interaction, ubiquitous and pervasive computing, context-aware computing, handheld, or mobile interaction, perceptual and affective computing as well as lightweight, tacit or passive interaction.

Their key statement for unifying these approaches into one field is that all of them – intentionally or unintentionally – utilise at least one of the four principles of RBI:

Naïve Physics

People have common sense knowledge about the physical world.

Body Awareness and Skills

People have an awareness of their own physical bodies and possess skills for controlling and coordinating their bodies.

Environment Awareness and Skills

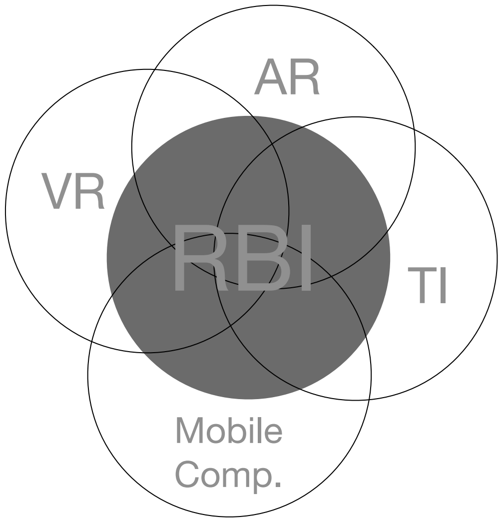
People have a sense of their surroundings and possess skills for negotiating, manipulating, and navigating within their environment.

Social Awareness and Skills

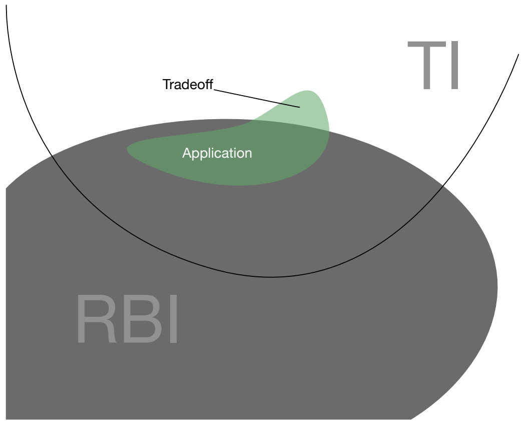
People are generally aware of others in their environment and have skills for interacting with them.

Design Implications

As the authors state, these principles – i.e. to base interaction techniques on pre-existing real world knowledge and skills – can help to reduce the overall mental effort that is required



(a) Venn diagram of RBI and its related research areas.



(b) The (hypothetical) location of an RBI application in the Venn diagram shown in Figure 4.7(a).

Figure 4.7.: Venn diagrams for Reality-based Interaction.

to operate a system because users already possess the needed skills by their being-in-the-world. They claim that this reduction of mental effort may speed up learning, improves performance, and encourages improvisation and exploration, since users do not need to learn interface-specific skills.

User-centered

Designing data monitoring systems via RBI implies the use of multi-modality in both directions, to and from the user. RBI forces to think problem–, respectively user centred, rather then tool-oriented. An example illustrating this is RBI's answer to the following question: *What is the typical reality-based approach to handle sounds?*[5] Sounds in reality are always connected to objects (re)acting with their environment. A loud bang, for example, always has a cause; be it an explosion or a slamming door. Auditive Displays on the other side give digital information a physical voice. There is no natural pendant for them, apart from – if it is the output of a Model-Based Sonification – a physical model completely rendered in the virtual. But this is, where the power of RBI comes into play: To be human-understandable and therefore closely linked to RBI themes, not only the sonic outcome of a physical model should be perceivable by the user. Moreover, RBI claims that the overall performance of the system will increase when its interface is part of the user's direct environment, be it integrated either via VR, AR or any other related interfacing technology. This is a strong argument for multi-modal interfaces for both, human sensing and acting with digitally represented data.

Classification of RBI into existing interfacing frameworks

On a more general level, Jacob et al. state that RBI can be seen as an umbrella term for all post-GUI Interfaces. However, I think that this is not true for all cases, since it is possible to construct interfaces that are clearly post-GUI, but not RBI. I therefore propose that fields like VR, AR or TI should be recognised to have *intersections* with RBI (as shown in the Venn diagram in Figure 4.7(a)) rather then being part of it.

[5] Discussion on this took place at the AmI journal club on March, 31st 2009.

Given the connection of fields, as different as AR, TI (acting in reality, using already existing phenomena, resp. augment them) and VR (acting in the virtual, re-modelling and manipulating existing phenomena), I doubt that RBI is a usable term to be of efficient use for applied interface development. However, the guidelines formulated in the description of RBI are worth to explore in all related fields. Although they tend to *mean* something different, depending on the utilised interfacing technology.

RBI Tradeoffs

One of these guidelines utilises *tradeoffs* regarding the above-described principles to sharpen the awareness of interface design. These tradeoffs are usually caused by the implementation of desired qualities of the system that cannot be implemented without automated algorithmic systems. Jacob et al. classify them into *expressive power* (users can perform a variety of tasks within the application domain), *efficiency* (users can perform a task rapidly), *versatility* (users can perform many tasks from different application domains), *ergonomics* (users can perform a task without physical injury or fatigue), *accessibility* (users with a variety of abilities can perform a task), and *practicality* (the system is practical to develop and produce). They further state that each tradeoff in an RBI-based system should be *explicitly* made. Tradeoffs are not only *optional* for RBI-related system design, moreover they deserve a central place: An application that makes use of dynamic/algorithmic data processing (e.g. have to use a computer) and is designed after the RBI framework has to have parts that result from these tradeoffs. Otherwise, the system could be built better *without* the use of computers (i.e. completely in reality).

A guideline for the design of an actual RBI interface can therefore be formulated as: Try to develop the desired application *strictly* according to the RBI principles, which especially means to avoid the mentioned tradeoffs. When desired features cannot be integrated without breaking rules, introduce tradeoffs, where each integration has to be accompanied by an explicit discussion of reasons and possible benefits. This results in an application that can be located in the Venn diagram (Figure 4.7(a)) as exemplified in Figure 4.7(b).

Socio-cultural aspects

According to Dourish and other thinkers of modern design theory, an essential aspect in interface design is the cultural and social environment in which the system will be used [Dou01]. Unfortunately, these socio-cultural aspects are somehow underrepresented in the RBI framework. Furthermore, it remains unclear how aspects, such as fun or pleasure – two factors that surely influence the usability of human-computer interfaces – should be integrated into RBI. An analysis and extension into these directions therefore is considered to be highly promising.

4.6. Methods for Interface Evaluation

How to evaluate human-computer interfaces? To get an overview on how to answer this question, I conducted an email survey in which I asked experts in TAI-related fields on how they would evaluate a software system for its interaction quality.[6] Twelve of twenty people were responding. They are experts in the fields of information technology, mobile technology, high-level programming languages, computer science, computer vision, Exploratory Data Analysis, sonification, computer interface design, virtual reality, ambient intelligence, media art, psychology, visual communication, and chemistry. After translating the (originally German) answers to English, I extracted the described approaches to identify steps of the

[6] The translation of the originally German text of the sent email can be found in Figure Figure 4.8.

Hello,

this is a little mini-survey.
Theme: *How to evaluate the Interaction Quality of an Application?*

Imagine that you designed and implemented an interactive part of a bigger software system (e.g. a photoshop plugin). Now, you want to know, if and how people are getting along with it, i.e. how well the implemented kind of interaction works.
What is your opinion on how to test for this?

Your suggestions and ideas are of great interest for me.
kind regards and happy thinking

Till

Figure 4.8.: The translated text of the email survey on the evaluation of Human-Computer Interfaces.

proposed methods.[7] In this way, I identified four modules that were considered necessary for a survey on interaction quality by almost all participants.[8] These are

Scenario In which environment is the interaction device tested?

Material Which survey-related media is used? Audio, video, questionnaire, subjective monitoring or (time-)measurements were named.

Methodology Which theoretical methods are used? Qualitative, quantitative, questionnaire, comparison, heuristics were named.

Indicators Which indicators are of importance for the analysis? Qualitative, quantitative or correlation-based indicators were named.

Although all respondents agreed in these general terms, their opinions regarding the concrete methods were very heterogeneous. This might be caused by the broad variety of expertise, however, even experts of the same field expressed very divers suggestions on how to cope with interaction quality evaluation. On a general level, their suggestions included qualitative and quantitative methods and indicators. Especially observation and interpretation of people's activity were considered useful. As measurable indicators for quantitative evaluation, besides measuring time and counting numbers of clicks needed for a predefined task, comparative stress tests before and after interface usage were named.

Quantitative Methods

These indicators can be identified as the basis for quantitative user studies. Such a study relies on the assumption that an intelligent combination of quantitative indicators and a structured/closed survey design can be developed, which can measure the *quality* of the explored interface. This quickly arises the question about the definition of the term *quality*; in many cases – at least in quantitative studies – it is defined as *good performance* in terms of the utilised underlying indicators for a concrete task that has to be accomplished by the participants. This means that the time user's need then is directly used as a criterion

[7] I used methods from grounded theory for this, a qualitative technique that will be described below [SC90].
[8] The curious reader can find the anonymised and translated wording of the participant's replies in Appendix A. The found codes are summarised in Appendix A.2.

for their performance, respectively the differences in stress tests before and after interface usage are interpreted as indicators for the quality of the interface. Although this might be a valid method for interface types in which a user task can be explicitly defined, this method is not sufficient for the evaluation of Exploratory Data Analysis systems due to their original intent. Their focus is to assist in finding *new* structures and insights, the outcome of such a session therefore cannot be determined beforehand and will differ from user to user. This in particular makes it very difficult to design a quantitative user study, since a big part of the outcome of a data exploration session depends on the intuition of the users.

Grounded theory

One qualitative method that can be used to evaluate user interfaces is grounded theory. It was also named by one of the participants. This explorative approach can be used to generate and proof hypothesises based on observations that are made when people use the system under exploration. Grounded theory was developed by Glaser and Strauss in the late 1950s as a sociological method for their studies in the Department of Nursing at the University of California in San Francisco. Travers explains in his book *Qualitative Research Through Case Studies* [Tra01] that

> [...] Glaser and Strauss accepted that the study of human beings should be scientific, in the way understood by quantitative researchers. This meant that it should seek to produce theoretical propositions that were testable and verifiable, produced by a clear set of replicable procedures, and could be used to predict future events.

Grounded theory relies on observations made in the collected data and thereof generated *codes* or *categories*. Its (possibly overlapping) phases are [Dic05]

1. data-collection,
2. note taking,
3. coding,
4. memoing,
5. sampling and sorting and
6. writing.

The research process itself relies on the translation of any collected material (1.), be it researcher notes, video or audio into written notes (2.). During this process, the writing of memos on hypothesises and possible sub- resp. super-categories is an essential part (4.), it helps to codify the observed behaviour into emerging categories (3.). As core categories emerge, their validity is proofed by sampling and sorting the data collection (5.). They later form the base of the theory and proof the validity regarding the data, since they can be used to name concrete sections in the original material in which the categorised behaviour can be observed.

Video analysis

Video interaction analysis is an emerging way to evaluate tasks in human-robot interaction [Hae09] and artificial conversation analysis [Kru09]. It can be combined with the grounded theory approach to form a qualitative analysis method that can be used to get insights into the recorded processes and actions. This suggests that video analysis is a valid tool for the initial evaluation of people's usage strategies when confronted with alternative human-computer interfaces.

The methods of grounded research combined with video analysis are applied to evaluate aspects of MoveSound, Reim and AudioDB. The results of the case studies (reported in Section 9.1.6, Section 9.3.6 and Section 9.4.3) indicate that this strategy can be considered useful to get an impression of user-related characteristics of TAIs. Due to the relatively small sample of four respectively five participants, though, its *quantitative* validity can be questioned. The case studies sufficiently confirm the usefulness of TAIs as interfaces for rich and nature-inspired representations of digital data and algorithmic processes on a *qualitative* level. Based on these investigations and their results, the systems may be undertaken further quantitative studies. However, this was not feasible during the work on this thesis.

5. Tangible Interfaces

In the previous chapter, *Tangible Interfaces* (TI) have already been mentioned as part of the *Reality-based Interaction* framework. However, TI is an independent research field, which investigates into the benefits of combining real-world objects with algorithmic data handling. Tangible Interfaces therefore exploit real world objects for the manipulation of digital information, or – in other words – enhance physical objects with digital functionality. What seems to be a rather simple idea eventually turns out to be a powerful approach to the conscious development of complex, yet natural interfaces.

The first three sections in this chapter introduce TI as an independent research field, provide definitions, and give a rough overview of current technologies and TI related applications. In the second part of the chapter (Sections 5.4 and 5.5) my view on TIs will be introduced. Furthermore, specific aspects will be pointed out which are of particular interest for the design of TAI.

The user experience of a Tangible Interface is dominated by the incorporated physical objects. Their inherent natural features of which users already have a prototypical concept are valuable for the designer and make it easy to develop interfaces that are naturally capable of collaborative and multi-handed usage. Even further, the usage of tangible objects implicitly incorporates a non-exclusive application, so the system designer does not have to explicitly implement it.

Although it is still possible to design uncomfortable systems, the users' everyday knowledge on physical objects and the handling of these encourage a nature-inspired interface design, which makes Tangible Interfaces a strong tool for HCI. Another benefit arising from a carefully designed system is the implicit display of its inner state. It makes it possible to observe the augmented artefacts and their relation to each other. The input device turns into a bidirectional component that is both controller and display at the same time. All this supports the user's development of a feeling of *flow*, just as it is common when playing a traditional musical instrument [Csi00].

Utopia

One rather utopical vision on how data and algorithmic functionality may be explored and handled in the future can be derived from Goldsworthy's leaf sorting (described in Section 4.1.1). Aspects that are worth imitating are the richness in the feedback and manipulation cues as well as the subtlety in its reactions. In this light, operating a Tangible Interface may not look different from manipulating non-augmented real-world objects for an external observer. The only difference to unaugmented artefacts, though, is the cause of its sonic, haptic and visual feedback, which is not only a result of pure physical reactions. Furthermore, it also derives from artificial reactions of algorithm-based changes in the data representation that is associated to the interface. In this case, it is not essential that the technical process is hidden, far more important is the fact that the human-computer interface is seamlessly integrated into the natural environment of its users.

5.1. What are Tangible Interfaces?

Tangible Interfaces (TIs) are used and designed by different research groups all over the world. Several disciplines work on and with them to get insights into their usability as well as the benefits and limitations compared to other approaches. Although 'Tangible Interface' and its synonyms 'Tangible User Interface', 'Tangible Interaction', 'Graspable Interface', 'Tangible Bits' or 'Tangible Computing' are widely accepted synonyms, various definitions of their meaning co-exist. This section presents descriptions of Tangible Interfaces by various experts. There will be comments on the definitions before I move on to a work in progress definition that will be used throughout the thesis.

Wikipedia

In 2009, *Tangible User Interfaces* were described in Wikipedia[1] as

> [. . .] a user interface in which a person interacts with digital information through the physical environment.

Furthermore, it described Ishii's *Tangible Bits* as

> [h]is particular vision for tangible UIs [give] physical form to digital information, making bits directly manipulable and perceptible. Tangible bits pursues seamless coupling between these two very different worlds of bits and atoms.

followed by a list of characteristics of Tangible User Interfaces:

> 1. Physical representations are computationally coupled to underlying digital information.
> 2. Physical representations embody mechanisms for interactive control.
> 3. Physical representations are perceptually coupled to actively mediated digital representations.
> 4. Physical state of tangibles embodies key aspects of the digital state of a system.

Nova

In 2006, Nicolas Nova held a talk at Nokia on *Tangible User Interface: misconception and insights* [Nov06], in which he defined a Tangible User Interface as

> [. . .] a user interface in which a person interacts with information through the physical environment.
> = Umbrella term

Fitzmaurice, Ishii, Buxton

In their paper *Bricks: Laying the foundation for Graspable User Interfaces* [FIB95], Fitzmaurice et al. introduce the new term Graspable User Interface as something that

> [. . .] allow[s] direct control of electronic or virtual objects through physical handles for control. These physical artifacts, which we call "bricks," are essentially new input devices that can be tightly coupled or "attached" to virtual objects for manipulation or for expressing action (e.g., to set parameters or for initiating processes).

Hornecker

On her website [Hor09], Hornecker introduces *Tangible Interaction* as a term that

> [. . .] denotes systems that rely on

[1] I am aware that Wikipedia cannot be considered a source of valid information. It does, however, show the broad interpretation of this term. Information taken from `http://en.wikipedia.org/wiki/Tangible_User_Interface` in August 2009.

> - tangibility / materiality
> - bodily / embodied interaction
> - physical representation of data embeddedness in real space, and augmentation of physical spaces.
>
> In different systems these characteristics can be of various intensity.

Regarding the scope of Tangible Interaction, she states that

> Tangible Interaction is an umbrella term, which encompasses approaches from HCI, computer science, product design and the interactive arts.

Dourish

In his book *Where the action is: The Foundation of Embodied Interaction* [Dou01], Dourish described *Tangible Interaction* as a generic term that covers the four fields (*a*) Ubiquitous Computing, (*b*) Augmented Reality, (*c*) a design perspective that investigates computing-tangiblitiy, and (*d*) the outcome of the Tangible Bits research investigation by Ishii. The term *Ubiquitous Computing* (*a*) is, according to Dourish, not only

> Weiser's vision of computationally enhanced walls, floors, pens, and desks, in which the power of computation could be seamlessly integrated into the objects and activities of everyday life.

Moreover, Dourish envisions the focus of attention of the ubicomp community in moving away from the computing machines and turning towards the actual computing process. The other field mentioned, *Augmented Reality* (*b*), is, according to him, the vision of

> computers in doorknobs and pens

dubbed by Weiser as "physical virtuality":

> It moves the computer into the real world. The site of interaction is the world of the user, not that of the system. That world [...] may be imbued with computation, but the computer itself takes a back seat.

The mentioned design perspective (*c*) is in Dourish's book represented by Bishop's design study of a *Marble Answering Machine*, where marbles represent incoming calls, and Jeremijenko's media art piece *Live Wire*, consisting of a plastic string that wiggles according to network traffic running through a connected network cable. Finally, Dourish quotes that the *Tangible Bits* research investigation

> [...] incorporates aspects of both the Ubiquitous Computing Program and the design perspective explored by Jereijenko.

Greenfield

Like Jacob with RBI, Greenfield views

> ubiquitous computing, pervasive computing, physical computing, tangible media, and so on [...] as facets of one coherent paradigm of interaction I call *everyware*. [emphasising by the original author]

As mentioned in Section 4.5, Greenfield's central argument for the Everyware concept is that the related fields do, from a user's observation, not have a distinctive difference to each other.

5.1.1. A Working Definition

All the above-quoted definitions describe Tangible Interfaces as a broad, yet diverse field that ranges from an interface where one person interacts with digital information through the physical environment to direct control of electronic or virtual objects through physical handles for control. According to the definitions, related fields of TI are HCI, computer science, product design and interactive arts as well as Ubiquitous Computing and Augmented Reality, while some believe that the latter two can be actually seen as parts of Tangible Interface research. On the other hand, there are at least two more general approaches to HCI research, RBI [JGH+08] and Everyware [Gre06], stating that TI is actually a part of them; be it as something that orients itself at real interactions, or be it as something that cannot be differentiated by users from other approaches. However, TI is widely recognised as an *umbrella term* for many research fields. I want to meet this with the following working definition for Tangible Interfaces.

Working definition

Definition *A* Tangible Interface *is a system that allows people to work with algorithms as well as abstract data just as if they are an integral, physical part of the users' reality. In order to control algorithmic behaviour, it utilises physical objects and thus allows people to operate and manipulate virtual (software-)objects.*

This definition highlights important characteristics of Tangible Interfaces. First, the natural availability of collaborative work with and via Tangible Interfaces is pointed out, secondly, it shows that physical objects and computational behaviour are closely coupled, and thirdly, it emphasises that a close interconnection between reality and the virtual manipulation of data is of importance.

5.1.2. Example Applications

To understand the usefulness of TI, beacon applications are described next. They are widely known in the TI community and often cited as representative examples for TI design.

Marble answering machine

The *Marble Answering Machine* [CS95] is a design study of an interface for a telephone answering machine. It represents recorded messages by marbles. Their location on a central element determines their current state. It is for example possible to play back a message by placing it on the dedicated area, or to clear it by putting the associated marble back to the (unassociated) marble stock. Being only a design study, the Marble Answering Machine clearly has its focus on the design of the actual user interface of a widely known automata for handling digitally stored audio recordings.

metaDesk

Clearly a different focus has the MIT's *metaDesk*, a surface-based Tangible Interface [UI97]. It is equipped with an *Active Lens*, a flat panel LCD that may be used as an overlay to data displayed onto the surface, a *Passive Lens* for interactively replacing covered parts of the visually shown information by other data, and *Phicons*, which represent landmarks and serve as handles into the projected information. This system was built as a proof of concept for both surface and vision based interaction with digital material such as maps or music controlling systems. Its technical realisation can be seen as a statement for the technical feasibility of such systems. Furthermore, it can also be considered a platform for fundamental research in position-based data representation controllers.

reacTable

A related system is the *reacTable*, a musical interface for collaborative control of sound synthesis processes [JKGB05]. The system allows users to arrange specialised objects on an active surface. Their configuration determines the parameter states of filters, sound generators, and controllers, all statically associated with the objects. Featured in many sound-related public journals, this implementation is widely known.

Urp is a system for collaborative environments, supporting the complex task of urban planning, usually done in teams [UI99]. In this setup, the focus is on the actual application rather than on TI research. Urp serves as a platform to exemplify implications that follow concrete urban planning scenarios. *Urp*

Live Wire is an interface to audio-visually represent network traffic in an unobtrusive manner [WB96]. The movement of a dangling red string is directly attached to the current traffic on a local network. This turns it into a human-perceivable, yet undisturbing representation. Although I have not seen the initial setup, I imagine its appearance was quite impressive. This feeling is drawn on the fact that it is cited as a good example system for the two research areas of Ubicomp and TI [Gre06]. *Live Wire*

The *Tangible Blocks* introduced by Anderson et al. can be used to physically build a geometric model that is translated into a virtual model [AFM+99]. It uses special brick-like objects that can be connected to form three-dimensional objects. Their positioning is then transferred into the computer, serving as a rough model for first-person shooter games or CAD programs. *Tangible Blocks*

5.1.3. Areas in Tangible Interface Research

The research in TI can be separated into several closely related fields. I will list main application areas that are widespreadly recognised and accepted in the TI community.[2]

Various research is being carried out to work on technical aspects that are needed to build tangible systems. Companies like Nokia and Microsoft invest into the research for tangible consumer-electronics and supporting technology.[3] Related to their market strategies, ongoing research features also mobile and embedded TIs. It focuses on possible applications of TI in remote environments. The used systems are designed to embed all the electronics, sensors and actors that may be needed. These systems do not require any additional computing power and are often wirelessly connected to a data server or to other embedded systems. *Tools and Techniques* *Embedded systems*

As explained in Section 5.1.2, there has also been significant development and research featuring active surfaces as central elements for TIs. Orientated towards desktop-work and workshop-like environments, this research investigates active environments particularly for work, art and play. Also, many custom applications have been developed incorporating Tangible Interfaces such as e.g. for urban planning [UI99] or assembly line design [Hor09]. They are often specialised for the needs and specifics of their environment and users. These prototypical applications are then evaluated for their general performance with respect to their initial task. Their research outcome are the findings of related user studies, which may be used for related applications, and identify usage trends. *Surface-based systems* *Niches*

A huge proportion of TI-related research can be summarised as Interface– and Interaction de-

[2] The compilation is inspired by the *Proceedings for the 2nd International Conference on Tangible and Embedded Interaction* [TEI08].

[3] I.e. both were sponsors of the Tangible Embedded Interface Conference in 2008 and 2009.

sign. One important example is the *Marble Answering Machine* as described in Section 5.1.2. It is from such works that theoretical aspects evolve which can be used to investigate into theoretical considerations on TI design. I investigated further into this direction and will present the results in Section 5.4. Other contributions to the TI research cover learning and didactics. A prototypical example for these efforts is the *LinguaBytes* [HHO+08] project,

> aimed at developing an interactive and adaptive play and learning system that stimulates the language development of toddlers [...] with multiple disabilities.

Related Disciplines

TI research is closely related to a broad variety of research fields. I now give a short overview about the main relations and their connection to TI.

Design

Both Interaction Design (IxD) and Industrial Design (ID) take part of TI as far as the design of the interface in hardware and interaction is concerned; they both put theoretical assumptions into action (see Section 4.3). Other philosophical notions of interface design and its interpretation concerning the influence on our environment is taken from philosophers like Husserl [HL65] and Heidegger [Hei27]. A prominent reading on their implications may be found in Dourish [Dou01] or Fernaeus et al. [FTJ08].

Implementation

Apart from design and theory, the implementation of TIs also requires knowledge in computer science, especially in the fields of robotics and ubiquitous computing as well as in electronics, computer vision and augmented reality.

Analysis

Finally, the analysis of TIs can be undergone in various ways, involving several disciplines. Essential in any case is the incorporation of the *human factor*. Possible science fields to draw from are psychology, sociology and human computer interaction. A detailed description was given in Section 4.3, followed by a report on methods for interface evaluation in Section 4.6.

5.2. Tools and Technologies utilised by Tangible Interfaces

5.2.1. Sensor technology

Position recognition and tracking

TI systems often require the position and orientation of physical objects. The two techniques that are mainly used for this are computer-vision-based tracking and triangulation via electromagnetic field sensors.

Camera-based tracking

Camera-based approaches are mostly used for multi-touch applications or to track visual markers attached to objects. They usually consists of one or more cameras, a data station and a Computer for data processing. The VICON system used for the JugglingSounds interface (see Section 9.6) is one example. It is a commercial system that is capable of 6DOF tracking of rigid bodies and human motion. It relies on reflective markers attached to the objects to be tracked. VICON is tailored towards applications in animation, biomechanics, and engineering, which requires them to be specialised in full-body motion tracking. However, it can also be used for the tracking of several rigid bodies in realtime at an update rate of approximately 120Hz.

Electromagnetic field sensors

Electromagnetic-based tracking on the other hand utilises the distance-variant strength of electromagnetic fields emitted by active markers that are attached to the objects to be tracked. Several sensors allow to get information about the current position of an object, either by triangulation [PHB00], or – when being in a matrix below an active surface – by getting the actual position in a grid [UI97].

ID

Apart from the location of objects, in many cases it is also essential to know the identity of an object, which is usually represented as unique ID. Position recognition and tracking technologies are often able to obtain the ID of an object. However, in situations in which the exact position of an object is not required, other systems such as RFID detection have been used.[4] These have proofed to be more reliable and less complex (at least in their set-up and cost) than marker-based tracking systems.

RFID

Radio-frequency identification (RFID) technology can turn "normal" objects like eggs, clothes, or even dogs into machine-recognisable objects. This is realised by attaching small chips, so-called *tags*, incorporating a unique ID. Objects placed near an RFID reader are recognised and send to a computer program that triggers arbitrary functionality on their appearance respectively disappearance. Appearance and disappearance of objects depend on the coverage area of both tag and reader. This defines an active space around the reader. RFID technology, in other words, allows digital systems to test the availability of specific physical objects at specific places. This functionality can be used for example for linkage-type applications that implement information containers as described by Ullmer [UI00]. The applications he describes utilise RFID technology to provide active ports on physical (electronic) devices like printers or screens, transforming the user's view of the tagged objects into *carrying containers* for digital information like texts, images, or other digital media. Placing tagged objects that were loaded with information on a specific device (like a printer) e.g. trigger the printer to produce a paper-based representation of the linked information.

Haptics and other object-inherent parameters

Apart from an object's location and identity, also object-inherent parameters are of interest for TI development. However, not only standard – electronically easy to sense – states such as button operations, the actual degree of a knob, or the position of an attached fader are of interest, but also measurements such as applied pressure, inherent orientation, acceleration, or the degree in which two parts are oriented to each other can be of value. To measure them, electronic sensor components can be used. These are e.g. Weiss-Foam [KWW03] [GWBW06] (pressure), triple axis accelerometer components like the MMA7260Q (orientation, acceleration), or regular potentiometers attached to embedded processors for both data acquisition and forwarding.

5.2.2. Processing

The processing of measurements, in which the augmenting functionality and data representation are included, is usually performed either in the used objects themselves, requiring digital components like processors and other electronic logic to be embedded into the objects, or are sent to an external computer system that processes the information. While embedded systems tend to be small and self-contained, and hence can be used at any place, systems attached to external computation power have the benefit of a lot more computing capability and are used for locational fixed applications.

[4] See for example `http://www.touchatag.com/`.

5.2.3. Actuating

Following the processing of physically sensed values and digitally stored information is the actual representation of the resulting data. Mainly the three modalities vision, haptic and audio are used for this.

Visual feedback

Visual feedback is usually given by internal displays and LEDs that are embedded into the interface, or by projectors that overlay information to the workspace. In some cases, information is also visualised on a standard computer display next to the TI.

Haptic feedback

Current technology also allows haptic feedback originating from embedded motors or other actuators. The range of different motors allow actions of the interface ranging from changing their shape or texture up to dynamics such as vibration of object parts.

Auditory feedback

The equipment of TI with an Auditory Display is usually achieved by loudspeakers. Their placement, either embedded or external, depends on the actual application. In some cases, sound is also transferred via headphones to provide an increased intimacy or a more sensual hearing experience (as it is the case in Tangible Data Scanning that will be described in Section 9.5).

5.3. Analysis and Classification

Dourish

Various theoretical classification systems for TIs have been developed until today. One approach to the analysis of TIs and related systems is given by Dourish in his book *Where the Action is* [Dou01]: He describes his view on Tangible Interaction and Social Computing by claiming that they are parts of a bigger field of research which he calls *direct interaction* and which covers the implications of technology-mediated inter-human communication.[5] His view is based on the observation that both fields have many aspects in common, especially the chance (and need) for inter-human collaboration and social interaction of the involved participants. He therefore introduces *ethnomethodology* as a toolset for the understanding and analysis of working paradigms that are supported by consciously designed interfaces. Among other aspects, this leads to a better understanding and possible guidance of the human-related part of TIs.

Ishii, Shaer & Holmquist

Contrasting analytical work on TIs was done e.g. by Ishii et al. [UI00], building a theoretical framework on a more descriptive theory on how objects are linked to functionality in existing applications. In the same way, Shaer et al. introduced *TAC*, a conceptual framework for Tangible User Interfaces that is mainly suited to analyse existing TIs for their semantical and functional relationships [SLCGJ04]. The paper *Token-Based Access to Digital Information* by Holmquist et al. introduced a taxonomy for physical representations with a focus to applications for accessing digital content such as music. Although the resulting taxonomies of such studies seem to sufficiently explain and sort already existing work in TIs, they do not explicitly comment on the human factor, a central aspect of interfaces between humans and computers.

To summarise, the described approaches help TI designers in different ways: While Dourish provides tools for the analysis of human relationships they want to mediate by TIs, Ishii, Shaer and Holmquist analyse existing applications for their use of objects as proxies for

[5] The definition of his term *Tangible Interaction* is exemplified further in Section 5.1.

digital functionality. However, all these frameworks and analytical systematics do not explicitly suggest ways for interface design of TIs. Another underrepresented part are the emergent possibilities when utilising TI object shapes, i.e. their inherent physical features and relationships. As already shown by Patten et al. [PI07], their explicit utilisation helps to increase the performance of TIs.

5.4. Crafting the Digital – Towards a Theory of Tangible Interface Design

The previous sections have given an overview of TI design, implementation and relations to other fields. The focus has been on the communities view and popular research trends. In this section, TI design guidelines will be introduced and some arguments for them will be presented. They have been deduced from observations made during the work on this thesis. I first describe how these observations and their consequences were developed, followed by more theoretical considerations and a proposed taxonomy for TIs in Section 5.5.

5.4.1. Turning Observations into Design Strategies

The following paragraph describes the iterative design process for the Reim Tangible Interface.[6] It serves as an example on how observations and considerations regarding usability and technical feasibility can influence the development of such a Tangible Interface.

A Designer's Perspective on TI Development

The original inspiration for the Reim system was an example for Model-Based Sonification of Thomas Hermann, realised by Jan Krause [HKR02]. Its primary user interface is a rigid object that senses its momentary acceleration and transfers it into an underlying physical model. The model itself simulates spring masses attached to certain points in space, determined by a connected data set that preserves the original data structure. *Origins*

Shaking the object and therefore the model causes a displacement of the data-objects and results in (virtual) collisions. These collisions render into sounds that are played back to the user in real time. The user experience when shaking the system is similar to that of shaking a rattle. However, it differs from an actual rattle, since its sound renders from a loudspeaker and is completely determined by the values measured by the excitation and the underlying data used to setup the Sonification model.

Another attempt to knock on data was made during the *Science by Ear* Workshop at the IEM in Graz, Austria, in March 2006. There, I took part in a small working group that was working on possible Sonification strategies of the "Materials" data set.[7] This data set contains measurements of the chemical features of wood, which we used as an inspiration for the Sonification process and its interface. We designed a system that imitated the sound of knocking on wood, except that the resulting sound was an artificial rendering of a data Sonification of one measurement entry. The Sonification was realised as a parameter *First iteration*

[6] In Section 9.3 a detailed description of Reim will be given.
[7] The Materials data set is provided at `http://sonenvir.at/workshop/problems/biomaterials/`.

mapping of each data item to a set of ringing filters (**DynKlank**). The frequencies were determined by the size of the incorporated molecules, whereas the decay-times corresponded to the intensity response of these molecules. The resulting filter bank was fed by an impulse that was triggered by the user interface. The users were able to tap on drum pads of a Trigger Finger interface.[8] There were also plans for further improvements, including the mapping of the tap velocity to the amplitude of the impulse in order to add dynamics to the display. In this system, the physical interaction between an exciter (e.g. the user's finger) and a surface (the Trigger Finger's pad) was filtered and processed to one value (a midi NoteOn message), and then used as the input to a complex reactive sound rendering system as will be pointed out later. The bottleneck of data processing between user input and model interface could have been avoided. The resulting system was missing a natural user-experience, also, we considered the inner complexity of the data to be not sufficiently reflected.

Second iteration In 2008, I worked with René Tünnermann on a similar concept. This time, we attached a contact microphone (a so-called transducer) to a tin can and implemented a *surface-trigger* in SuperCollider, which returned an audio-rate trigger when an onset event appeared in a relatively clean signal. This was observed from the can-attached transducer when tapping it.[9] We used a different data set for this interaction design, causing a different mapping of the data onto the free parameters of the filter bank as in the previous approach: The used glass data set is nine-dimensional; each dimension was mapped to the frequency of one resonator. The trigger-signal originating in the user interaction of knocking on the augmented can was then used as a gate for an audio rate envelope multiplied with pink noise, which, again, is fed into the filter-bank.

Turning point At this point, we realised that we had implemented an unnecessary layer of abstraction with this artificial exciter, imitating a complexity that had already been present in the captured signal: it already had a transient characteristic. We therefore decided to take a more direct approach by substituting the artificial exciter by its natural correspondent. We used a "physical model" as it appears in real life in order to produce input transients. After some attempts, we discovered that the software system can also be used with the transducer attached to glass solids. This extents the use case of tapping by scrubbing e.g. a metal surface. As in the setup before, the sound coming from the transducers was directly fed into a filter-bank, i.e. nothing in the actual software implementation was changed. However, its sonic characteristic changed fundamentally and with it the user's operating strategies.

Lessons learned The result of this design investigation is that an immediate connection between input sensing, the underlying algorithmic process, and the output rapidly increases the possible usage scenarios of an implemented system. However, none of the two first mentioned approaches allowed to directly manipulate the incorporated data, because the sensor information from the user manipulation was too limited.

This is a typical example of the morphogenesis of interface design through several iterations from which guidelines for Tangible Interface research have derived. Most of them were collected in the development of the applications described in Part II. Regarding the observed aspects of TIs as described above, I strongly support the point made by Williamson in his PhD thesis [Wil06]:

[8] A controller device by M-Audio.

[9] More complex approaches can be found in the SuperCollider UGen implementations of **Onsets**, **OnsetsDS** or **PV_JensenAndersen**.

> Where uncertainty is present in an interactive system, the full uncertainty should be preserved as far as possible and the inference done on the complete distribution of potential values when an action must be performed. Early, irreversible, filtering of the values to sequentialise the process should be avoided.

5.4.2. Utilising Features of Tangible Objects for Interface Design

Implicit State Display by Object Persistence

Any physical object has a perceivable state. They feature at least a position regarding a point of reference that can be recognised. Depending on their characteristics, they offer information on their current shape, colour, or orientation to their observers. Many TIs make explicit use of these features by sensing position and other states, and by displaying their current artificially augmented configuration through built-in lights or motors. Through manipulation of the objects,users can control the associated algorithmic processes. This technique has an inherent advantage concerning the usability of Tangible Interfaces: The state of the object implicitly indicates the state of the system without the incorporation of any active element. The interface itself is recognised by the user not only as a simple controller, but also serves as a display of the current control state.

Physical Constraints as a Chance for Tangible Interfaces

At first glance, the physicality and the resulting constraints of tangible objects implicates many drawbacks when compared to GUI-based systems. For instance, it is impossible to put two rigid objects onto the same position, and it takes manual effort to construct or deconstruct structures assembled by physical objects. However, physical constraints built into the interface are an opportunity rather than a drawback for its design. Let us consider for instance the difficulty to compute the optimal dense packaging of many objects in a given volume (like gravel in a glass). It is difficult to obtain a numerically result, even with the computing power available today. However, the physical system “computes” the solution in only a few seconds when shaking a glass filled with such objects. The question arising from this observation is how algorithms have to be designed to incorporate the hardware interface’s natural behaviour into the finding of a solution to the given task. This task requires a solid design process that is well informed about (*a*) the demands of the algorithmic process in order to be controlled, and also of (*b*) the constraints introduced by the interface. I will now focus on the various properties of manipulatable physical objects that can be considered to be used in TIs.

Continuity

Supposed that a TI utilises a set of rigid objects, e.g. the marbles shown in Figure 5.2(a), some observation can be made: In difference to virtual GUI objects like icons or windows, real objects maintain their spatial extend and thus cannot share a spatial volume. Furthermore, changes of their position, orientation or shape can only be made continuously. This means that there is always a trajectory without gaps between the start and the end position of an object’s movement. In consequence, moving an object from one position to another either displace all the objects that interfere with the movement trajectory, or – if this is not possible – the trajectory is detoured around them (see Figure 5.7 for such an example displacement). bearing in mind that this side-effect should be considered as an intrinsic

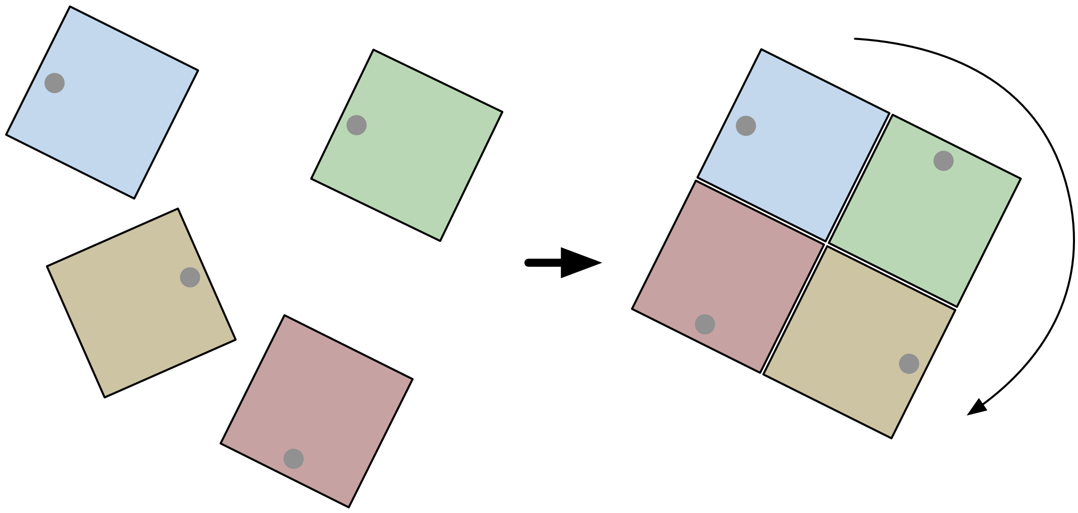


Figure 5.1.: As a side effect of physical constraints, the rotation of four audio-loaded cubes results in Shepard-Risset glissandi.

benefit, the interdependency of rigid objects as TI controllers can be used in various ways. Let us, for example, imagine an activity-driven auditory data display, where each data item is linked to a physical object. Depending on the objects position, a data-driven sound is rendered. By moving an object, the listener automatically experiences the properties of nearby objects (data items), and thus can understand local properties. Since acoustic rendition tends to be considered as directly connected to visual change [MM76], a binding of visual and auditory responses is naturally achieved. In my opinion, the value of a TI is very much dependant on how the human-observable properties of the used tangible objects are utilised. Especially their constraints should be reviewed and checked for potential benefit.

One example for such constraint-based design is a set of rigid cubes that can be rotated on a surface. The design of the system herby links the angle of rotation to the frequency of a rendered sine wave, where a 360° rotation is equivalent to one ocatave. Having four cubes, some interesting stationary sounds can be adjusted, for instance chords, consonant and dissonant intervals. As an interesting side-effect of the physical constraints, the packaging of cubes as shown in Fig. 5.1 with 90° rotated objects gives an astonishing sound while the user is turning the whole meta-object: Shepard-Risset tones, an auditory illusion continuously rising in pitch, can be heard [She64]. This effect is caused by the geometric structure, which by design directly reflects the harmonic structure of the audio synthesis. However, it is the geometric constraint given by the cube-shaped objects that introduce the structural forms. This design is a nice example to explore, and to literally grasp, peculiarities of acoustic illusions.[10] Sometimes, physical constraints together with a well-designed mapping of controller values to free synthesis parameters generate a whole that is larger than the sum of the constituents.

[10] A video capturing this effect is part of the DVD.

(a) Marbles; simple rigid body artefacts with a Grain-like character.

(b) Interface design of an office application designed by students during a course on Tangible Interfaces.

Figure 5.2.: Examples for Tangible Interface Objects.

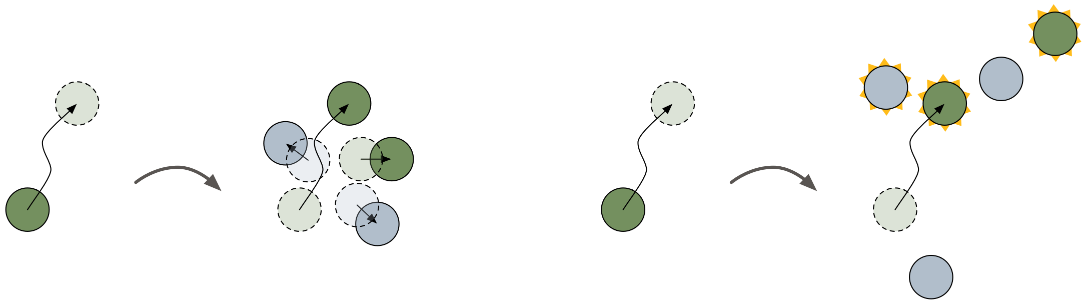
Clutter Confuses People – Minimal Tangible Interfaces

Any algorithmic functionality can be mediated by adding an extra button to an existing system. However, this strategy leads almost always to feature overload. The image in Figure 5.2(b) for example shows the outcome of this technique. During a course on Tangible Interfaces in 2006, a student group designed a Tangible Interface that was intended for the future office. In their design, many features that were meant to support task planning and staff management were represented by separate objects. Each of them represented either a specific person, or a functionality. Although each feature of the Tangible Interface was promising and thought-out well on its own, its overall appearance was cluttered. In the simulated *Wizard of Oz* scenario in which the students tested their TI, it turned out that it was difficult to remember the various functions of the incorporated objects. Additionally, the combination of modules that were not intended to be used together introduced inconsistencies.

A better strategy to integrate new functionality into a TI is to find a representation that incorporates already existing parts of the interface. Only if the addition of this feature by existing parts is not possible, new controls may be added. Another option is a complete redesign of the interface, which then considers also the new functionality. Often, however, a clever manipulation of the interface design holds completely new usage strategies and manipulation concepts, opening a bunch of new and beneficial options. The application of this strategy results in minimalist interfaces, i.e. interfaces that have as much controls as are needed for the operation of the underlying algorithmic process.

Limits of Purely Tangible States

As described in Section 4.2, it is possible to compute and manipulate abstract information with purely mechanical systems. People then interpret objects as data parts, their states and impact on each other as the data's manipulation. Taking this into account, parts of the processing realised in Tangible Interfaces do not have to be calculated by the underlying electronics, since it is already done by its physical components. An example for such an interface besides the mentioned slide rule and planimeter is the abacus, a tangible system



(a) Natural object reaction. Acting on an object causes physically related objects to react.

(b) Artificially induced object reaction. Acting on an object causes other objects to react. They do not have to be physically linked.

Figure 5.3.: Object reactions in Tangible Interfaces.

designed to support calculation. However, like all early figural systems of calculation, the abacus records the calculation result and not the process:

> At every new step, the previous one is lost, such that the result can only be validated by repeating the entire calculation [BH05].

Out of this evolves a chance for Tangible Interfaces to enhance such a non-technical system by augmenting it with a history.

The Concurrent Reality of Physical and Virtual Reaction

Tangible User Interface objects can be manipulated in three different ways. First and most obvious, the user can directly touch them and change their position, orientation, or all other internal states. Second, change of position may be caused by *physical displacement* caused by other objects. Third and most characterising for computer-assisted devices, objects in TI can be manipulated implicitly via underlying computing routines.

For example, the movement of an object can cause others to light up, indicating the change of an otherwise hidden state, or to show that they are implicitly connected. In other words, Tangible Interfaces augment physically well-known (i.e. trivial physical) reactions with virtually implied behaviour that otherwise is a unique feature of computer environments. This additional behaviour is achieved by adding a digital layer to the objects. Therefore, it is possible to affect physical objects that are not directly (i.e. not physically) related, i.e. to mediate a relation of physically unrelated objects to users (see also Figure 5.3). I call this *artificially induced reaction.*

From a design perspective, Jacob's theory of minimal addition can be applied, however, the decision for a specific augmentation modality implies certain considerations to be made. A visual augmentation for example requires that the augmenting visual stimuli can be reliably observed and identified with the corresponding interface parts. This requires either the objects or their canvas to be electronically enhanced.

An auditory augmentation on the other hand requires either loudspeakers built into the objects, or a spatial audio setup. In any way, a direct mapping of audio events to objects cannot be established, since the spatial resolution of the human ear is too fuzzy. As described before, this must not be a drawback, but has to be well considered in the design of the interface.

5.4.3. The Level of Abstraction in Tangible Interfaces

A desired behaviour of a software system that should react to a user's action may be realised between the following two extremes:

World model

One extreme is a system that implements all behaviour that is considered relevant right into its algorithmic part, and therefore abstracting from the used input. This approach can easily be used in combination with the model-view-controller [Ree79] design–, resp. architectural pattern, since both cultivate the separation of the algorithmic model from its input, respectively output. The parameters of the internal model are then connected to the actual user interface. Since all relevant information and processing is already integrated, the interface connection is pretty high-level, i.e. its appearance does not have to be connected to the functionality, but be ergonomic for the user. Strangely enough, such systems tend to offer users a huge number of control parameters (think for instance of common word or image processing software), and therefore need a high number of independent controls. Due to combinatorial reasons, this is a tough quest: already the combination of 80 buttons (of a usual keyboard), each with two states result in a space of possible input values of $2^{20} = 1.2089258196146291 * 10^{24}$. Moreover, considering continuos instead of discrete input dimensions as they are typical for Tangible Interfaces, this space is not easy to manage. Nevertheless, for each input vector, a valid strategy of action has to be explicitly implemented; most of them by introducing constraints that ignore certain input states (as pressing all 80 buttons at once). Even a tiny unconsidered action of the user (or the sensor system) may cause the system to proceed into an undefined state.

Reactive system

The other extreme is given by a *reactive system*, sensing information only on a very basic level and reacting to its sensor input with direct actions. Such a system does not make any use of an internal model or any other abstraction layer. Since the sensory input is not interpreted, its output is easy to explain. This results in a less error-prone behaviour due to misinterpretation of user input, which would be called "error" or "inappropriate behaviour". System reactions as well as input cannot be considered wrong anymore.

In the light of these observations, I define two extremes of tangible control; *Direct Control* and *Haptic Symbol*.

Definition *The term* Direct Controller *in TI denotes the direct one-to-one representation of quantitative data or algorithmic processes as tangible elements. Their state hereby controls basal data parameters or manipulations.*

Direct Control

Such a direct control of data and algorithmic functionality does not rely on a semantic interpretation of the underlying data or manipulation processes. Since the data (material), respectively the algorithmic process (tool) has a direct correspondence in reality, the user (and not the designer) interprets the activity. This heightens the communicability of unveiled information, previously covered in data, while at the same time supports users to develop a connection between (abstract) data and reality.

Considering the unintended

Many software and GUI applications can be used *wrong*, i.e. against the system designer's initial attempt. As stated in the description of haptic symbols, such a misuse often leads

(a) Ground layer. (b) First layer. (c) Second layer.

Figure 5.4.: Cuboro example setup.

into an undefined state, which then leads to undefined behaviour. In contrast, crafted artefacts of our everyday life have an intended functionality which they fulfil more or less appropriate. A chair, for example, is something to sit on, a paper-weight is an object to hold paper on a place, a bed is something that was built with the intention to let people sleep and lie on it. Beside these obvious and intended features, all these artefacts carry other (unintended) functions, which were not part of the designers' vision. E.g. a screwdriver can – besides his natural potential to support screw driving – more or less successfully be used as a pointer, a crowbar, or as something to get rid of staples, a bed can be used as a workplace, a chair can be used to stay on it or a paper-weight can be thrown at other people. Although all these functions are unintended by the artefact designers, they are, nevertheless, *valid*: it is possible to use them that way without breaking them. All the given example artefacts share, that they are basically low-level tools for object handling or manipulation, they can be seen as the correspondence of direct controllers for TIs. The designer of these direct controllers in turn should consider that they will be used in an unintended way, and enrich the possibilities to work with and represent algorithmic processes in the users' reality.

Haptic Symbol The other extreme of tangible control can be defined as follows:

Definition *A* Haptic Symbol *is a high-level physical representation for a possibly complex abstract data item or an algorithmic functionality.*

An example for this is a voice message represented as a marble in the *Marble Answering Machine* as described in Section 5.1.2: It represents a complex data item while offering only a limited set of operation (e.g. placing it on different functional parts of the answering machine to playback, store or format the associated message). Such a Haptic Symbol requires the interface designer to make some high-level assumptions regarding the data and the users' intention. At the same time, it develops an associated grammar that handles the validity of the assembled haptic symbols. It is the interface designer's responsibility to force the *right* usage of the symbols by the actual hardware; e.g. the playback mould of the answering machine should only allow to place one marble at a time, since the placement of several marbles in it would introduce an ambiguity that is difficult to manage; an interpretation of the user's intent has to be explicitly made: Does the user want to play back the associated messages in parallel (which is pretty strange for an answering machine), or in sequence, and if the latter, in which? Also, the hardware state of the system then does not completely reflect the actual software state anymore; additional information (about the currently speaking marble) has to be introduced.

To exemplify the difficulties evolving from a design that relies on haptic symbols, I next

(a) Valid track.

(b) Turning a track results in an invalid track.

(c) Different view of center figure.

Figure 5.5.: Cuboro cubes as an example for haptic symbols.

describe the limitations of a modular marble track design.[11] A standard Cuboro marble track construction kit consists of 54 wooden cubes, each having one or more carved tracks and tunnels. These haptic modules define a basis for possible marble tracks. Figure 5.4 shows examples of such a track.

Although the cubes do not have any haptic connections such as they exist in Lego or Fischer-Technik, they have to be put together in *just the right way*, otherwise the marble track is not valid, i.e. working.[12] By experimenting with the modular marble track system, I discovered that it is possible to define the grammar of a language based on the haptic symbols (the marble track modules). Using this language means to combine the symbols to sentences (tracks) and speak them (i.e. run a marble in the track).

As in other languages, it is nearly impossible to abuse the basic symbols and their associated, implicit grammar and still produce syntactically valid sentences: To illustrate this, let us envisage the following scenario: *Three cubes are placed on a desk as shown in Figure 5.5(a), assembling a valid marble track. Now, flip the whole aggregate by 90 degrees; it still looks like a valid marble track (Figure 5.5(b), 5.5(c)), apart from the fact that the marble will not run through it anymore. At least part number nine*[13] *feels wrongly placed, having its newly defined input at the wrong position (and with a un-runable curvature inside). Apart from this, also this flipped track is not usable in a bigger marble track scenario, since the connection points are on the wrong places. However, the combination of the blocks still feels to be right from a user's view.*

The un-runability of the built track (i.e. its invalidity concerning the marble track grammar) develop implicit restrictions to the marble track system. They are implicit, since the described (false) combination of objects seems to be valid unless it is used to run a marble. However, the way the limitations are communicated feel to be wrong. Especially with physical interfaces, it is possible to add physical restrictions that limit how objects can be assembled and therefore prevent the developer of a marble track to build un-runnable tracks. The cubes then would haptically inform about the validity of the built structure.

[11] I use a Cuboro marbletrack: `http://www.cuboro.ch/`
[12] For this experiment, I interpret the running marble as the evaluation of an algorithmic task.
[13] Numbering according to the Cuboro catalogue. Number nine e.g. is the block on which the marble is placed in Figure 5.4(b) and 5.4(c).

Level of Abstraction

Together, *Direct Controller* and *Haptic Symbol* identify the two extremes of the *Abstraction Level* of Tangible Interfaces:

Definition *The* level of abstraction *(*LoA*) of a TI describes the role of physical objects in the manipulation of data. This role is placed between the two extremes* Direct Control *(low) and* Haptic Symbols *(high).*

TIs have a high LoA, if they allow to control complex tasks with simple operation skills, e.g. by the press of a button. A low LoA on the other hand is given if the level of operation complexity corresponds to the level of complexity in the underlying data manipulation and algorithmic procession. The LoA of a TI is not quantifiable. Nevertheless, it is possible to compare the LoA of two interfaces as long as they operate on the same data type.

Shifting LoA A large part of a TI's appearance to its users is determined by its LoA. It is possible to integrate mechanisms into the interface that let the users shift the LoA of the TIs components during runtime. This unveils a great potential for interface and controller design, since it combines the strengths of both, haptic symbols and direct controllers and lets the user decide on the actual level he wants to work with. Shifting LoA is a way to cope with limitations introduced by features inherent to real physical objects.

Definition Shifting the Level of Abstraction *describes a meta-switch of a Tangible Interface that allows to change the functionality of a physical object in the interface setup. The selection process has to be user-controlled.*

Note that this is not possible in purely reality-based systems (as e.g. the marble track example described above), since there cause and reaction are fixed to each other. In TIs, however, the data manipulation part is actively programmed into the system. Therefore, the linkage between algorithms and their controls can be developed to be changed on the fly. Examples for such abstraction level shiftings are

Object-item assignment A controller object usually manipulates the state of one data item. A shift of the abstraction level extends its controller state to be linked to a selection of data items, which then can be operated all by one physical element.

Recording and playback of object manipulation The measured values of a manipulation of a physical control object are recorded and played back (e.g. in sequence, or determined by the user) on different data items.

The applications described in Part II often integrate such a feature; The proof of concept application *AudioDB*, for example, enables the user, on a per-object basis, to either operate one sound with one object or to associate many sounds with one object. The decision the user makes can be reverted at any point, which gives him a powerful tool for the manual clustering and sorting of sounds (see Section 9.4 for details).

5.5. Equivalents of Canvas, Point, Line, and Shape in Tangible Interfaces

This section draws a connection between elements of visual art and elements in TIs. It lists and describes associations of Canvas, Point, Line and Shape to their counterparts in

TIs. The collected observations can be used for the design of Tangible Interfaces, mainly comprising of a mix of independently tracked rigid bodies to represent digitally stored data.

5.5.1. Surface and Space – Canvasses for Tangible Interfaces

Surface

In Tangible Interface setups, surfaces can be used either as the actual interface object, or as the place where other manipulation objects are placed and operated. In this section I focus on the latter, featuring it as the *canvas of Tangible Interfaces*.

When designing surface-based Tangible Interfaces, it is a good habit to also actively decide on the used surface itself. Considerations regarding its shape, extent and mounting height as well as its placement are important factors on how the system will be used. Especially the surface's location is an important factor, since it will not only decide how the TI will be accessed, but also influences its assumed function, i.e. whether it is recognised as the central part or a (more or less) useful decoration of the environment.

Space

While a surface restricts the system operation to be two-dimensional, another possibility is to use the three-dimensional space as the TI's canvas. Both approaches have there specific features: While it is possible to place an object at an arbitrary location on a surface and it'll rest there, this is not possible in 3d space. Also, a surface-based system naturally shows the operational limits quite obviously by the surface's borders, whereas it is difficult to place borders around a part of a room without preventing a human operation in it. A limitation of a surface-based design, however, is that it provides a significantly decreased number of independent dimensions (two locational and one rotational dimension) compared to a space-based system (three locational and three rotational), even when only simple objects are used.

Container-like Artefact

There are of course other canvasses than only Surface and Space that can be used as a basis for TIs. E.g. the marble answering machine has a custom shaped object that serves as the central element on which all algorithmically augmented action takes place. Also, *ChopStix* as described in Section 9.2 makes use of a base object with moulds and glasses that hold the actual controlling elements. That custom shape together with its intended usage, however makes it difficult to clearly classify these objects as TI canvasses, e.g. it depends on the point of view if a ChopStix glass is recognised as a canvas for the sticks or as a tangible object placed on the base object-canvas.[14]

tDesk development

Not only the decision regarding the canvas' type has a big influence on the resulting system, but also its articular shape. An example for this are the design iterations of the *tDesk*, a tabletop interface for gestural, tangible and multi-touch-based user input and audio-visual output. It was initially developed in the iLab of the Neuro-informatics Group[15], and then transferred to the laboratory of the Ambient Intelligence Group[16] of CITEC.

The first setup named *gesture desk* was a standard working desk with one large glass surface equipped with two cameras. Its size was about 2×0.8m with a height of approx. 0.7m. It was placed right below a large projection wall sized 4×1.5m. Informal interviews of the people who set it up yielded that its intent was to provide a working environment for

[14] This ambiguity can be seen in Figure 9.17.
[15] `http://www.techfak.uni-bielefed.de/ags/ni`
[16] `http://www.techfak.uni-bielefed.de/ags/ami`

one sitting person in front of it, working with novel data exploration applications that incorporate gestural interaction with sound and graphics. At the same time, the system was intended to be used for data exploration with one person standing in front of the screen. Other people might stand in his back, looking at his interaction.

Hardware characteristics

For the second design iteration of the *tDesk* (originating from *Tangible Desk*), we had a slightly different use-case in mind, namely the development of a multi-user system, featuring four places from which the applications where equally good to be accessed. This especially meant that none of these places around the surface should have been in favour for the data exploration task. Furthermore, the tDesk's intent shifted towards the development of Tangible Interfaces to support direct human interaction rather than providing a hardware for gestural interfaces. The described demands required the following hardware characteristics to be considered:

Modularity The tDesk where intended as a modular system in hard- and software. This facilitates fast prototyping, redesign, and easy extraction of its components. A modular software approach improves scalability by allowing the distribution of modules on different computers.

Compactness All necessary computing and sensing elements should've been integrated into the table so that it could've been transported and used without much effort in different contexts.

Multimodality The tDesk should've integrated multi-modal sensing and displays, like graphics, Sonification, visual sensing and tactile sensing. Possible hardware for this where projectors for rear projection, cameras, microphones, transducers and force sensitive resistors.

Rich Interaction Types The tDesk should've allowed the parallel use of tangible and gestural interactions.

Collaborative Work The tDesk where intended to support collaborative work incorporating tangible interfacing paradigm introduced by Ullmer et al. [UI00].

We therefore chose to build a cube with 80cm edge length, located in the centre of the room, at least 1.5m away from all walls. This central position located it also in the sweet spot of an eight-channel surround sound system. Due to the surface's square shape, it offered equal access conditions from all sides. The resulting cube hosted a video projector facing downwards, such that – via a mirror – the whole glass surface was used as visual display. By its equipment with cameras, and various sensors like force-sensitive resistors or contact microphones it allowed to acquire information on user actions on and above the glass. The use of a mirror helped to keep the desk small in height despite a given (not so) minimal focus length of the used projector. It also allowed to use optics for the camera with a longer focal length incorporating less perspective distortions than wider angled lenses. In addition, we mounted four pressure sensors at the supporting corners of the glass top that allowed to measure how strong a user touched the table. Finally, the system provided enough space to integrate a small desktop computer and mounting space for additional hardware devices. The final tDesk design (see Figure 5.6) is an integral part of several applications that will be described in Part II.

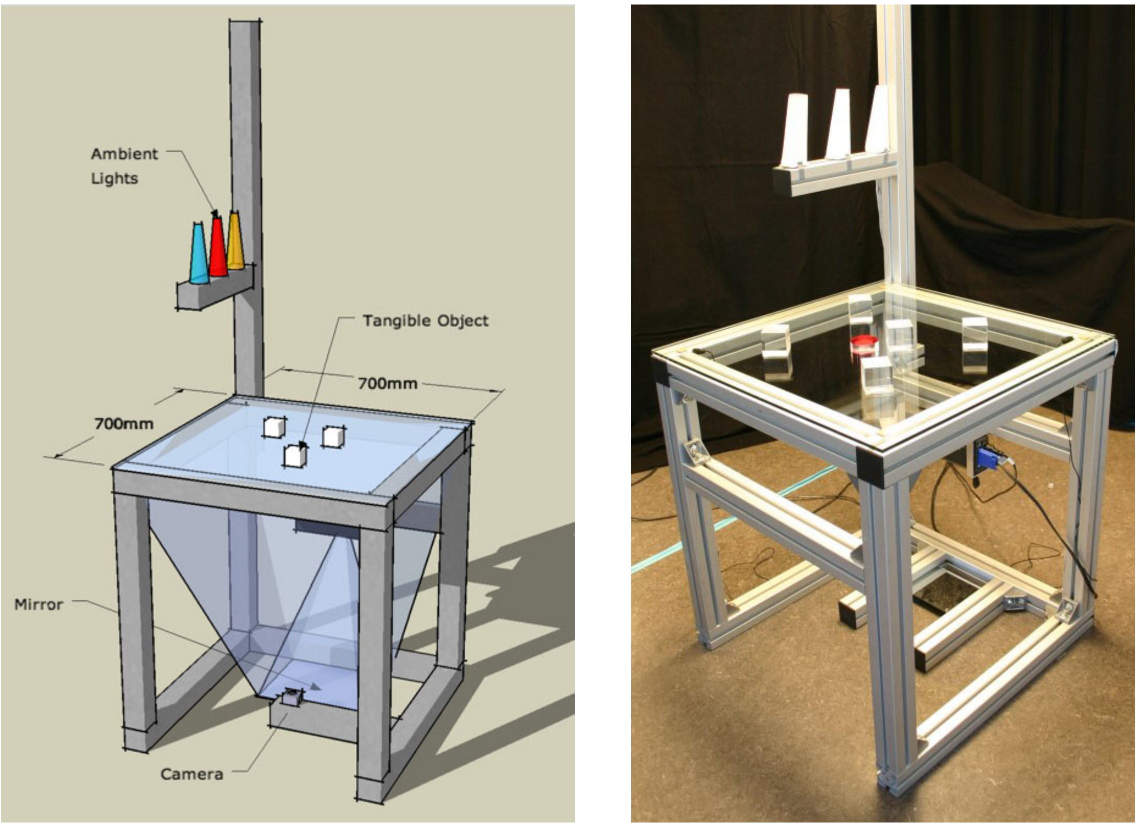


Figure 5.6.: The second iteration of the tDesk system. Images courtesy of Eckard Riedenklau.

Figure 5.7.: Lentil-shaped objects on a surface. Prototypical objects for the introduced object-class *Grains* and their manipulation.

5.5.2. Grains – Tangible Points in Space

The design of TIs often tend to use symbolic layers that bind abstract controls to specific objects. This design decision can be very powerful when dealing with a small set of fixed tasks, yet it does not naturally incorporate the strength of the human mind to build a loose coupling between objects and the corresponding information. The here introduced design concept for TIs is therefore based on simple physical objects called *Grains*. Although the very narrow set of possible operations the users may apply seems to restrict the system, it has the potential to appropriately mediate the complexity of the underlying algorithmic processes or data by the pure number of included objects. The applications *AudioDB* and *Durcheinander* are example systems that are based on this interface design. They will be described in Section 9.4, resp. Section 9.7.

Hardware Grains are small and indistinguishable objects, located either on a surface or in free space. For the different canvas types – surface and space – I differentiate between 2D- and 3D-grains. While 3D-grains are round like marbles, 2D-grains are more lentil-shaped because marble-shaped objects tend to unintentionally move around when placed on surfaces (see Figure 5.7). All Grains, however, have a round shape in the respective dimensionality of their canvas They therefore do not no have a notable orientation, neither for the human nor for the computer tracking system. Additionally, I assume that all Grains used for one application have the same colour and size, making them indistinguishable from each other. Only their particular position relative to each other or to the canvas' borders determines their current state. As the human, also the TI system should be able to track the Grains' trajectories in order to attach digital data. Due to their minimalist concept, Grains do not have object-inherent features like vibration or magnetic forces. Although these features are very interesting, their inclusion would heavily increase the complexity of a Grain's behaviour. Since the described parameter set is sufficient for the design of TIs to reflect complex relationships, I decided on a minimalist definition of Grains.

Grain-based TIs utilise basic physical constraints of material. They actively promote user exploration of the properties emerging from the combination of many Grains with appropriate data. To summarise the described features I define:

Definition *A* Grain *is a small physical artefact, which is not differentiable from another Grain. It is incompressible and has no user-observable orientation in space.*

Out of this definition, the following properties for Grain TIs can be derived immediately:

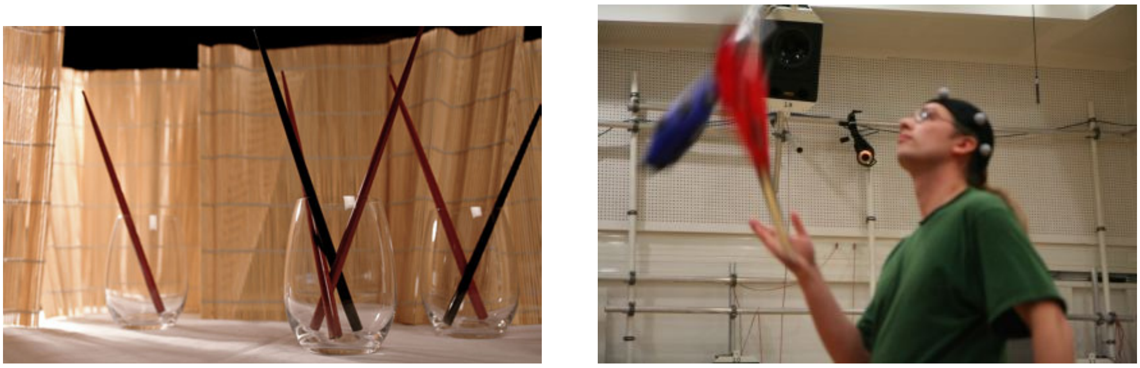
Placement A Grain is characterised by its position and extend. Its physical space is occupied and can therefore not be shared with other physical objects.

Movement As Grains are real-world objects they are restricted to steady movement.

Physical Relations Moving a Grain may cause mediated movements of other objects nearby.

Stacking Grains are not stackable.

Despite being very obvious, these properties are essential to the application of Grains in TIs and evoke in complex behaviour of the whole system. By turning the attention from one single Grain towards a set of ten or more, a whole bunch of additional features and constraints come into focus. The rigid nature of Grains for example forces them to rearrange their constellation when another object – a Grain, a human hand, or something else – is moved across their distribution. Figure 5.7 shows that behaviour. The increased amount of



(a) Chopsticks (b) Juggling Clubs

Figure 5.8.: Different types of Sticks.

Grains in operation noticeably increases the complexity and their possible configurations. This unfolding of possibilities by increasing the number of independent objects has its pendant in sound synthesis, where a similar emergent effect can be found in Granular Synthesis. In this case, stochastic distributions of simple acoustic elements (which are also called grains) suddenly create auditory gestalts and modes of control beyond the modes that exist for isolated grains [Roa01]. Section 6.4.1 goes into more detail of how to use these effects for information displays.

Speaking of the physical Grains, their dynamic physical interaction can be explained by physical laws. The laws' application, however, can also be considered as a "physical computation" that is based on the objects' interrelation, yielding in emergent reactions, which, due to its inherent complexity, can almost be called "behaviour".

Since this behaviour can be utilised to control a TI's underlying algorithmic process, this is, where it is possible to interface reality with the intended algorithmic augmentation. Obviously, a Grain-based TI may easily be put to use as an input interface to control the representation of similar typed items like photos, sounds or videos. But also e.g. the control of algorithms like (spatial) audio rendering processes, or determining the parameters of kernel seeds for density estimations of a data set can be controlled.

Two of the applications that will be described in Part II make extensively use of Grains. While *AudioDB* is designed to represent audio samples connected to Grains, and supports basic tasks like selection or grouping for collaborative interaction,[17] *Durcheinander* mediates the general functionality of clustering algorithms to learners by associating the Grains' position to positions of data items in a fictive data set on which a clustering process is applied.[18]

5.5.3. Sticks – Tangible Lines and Arrows

Going a step further with the help of the canvas and drawing analogy, I now move from points to lines that I call *Sticks* in TI.

[17] See Section 9.4 for details.
[18] See Section 9.7 for details.

Hardware Definition

As their name suggests, Sticks are elongated artefacts, mainly straight in their shape. Their extent ranges from about two centimetres to approximately twenty centimetres with a thickness of half a centimetre maximum. As Grains, they can be used by one or more persons either on a table or in free space. In difference to Grains, their design has not to cover special circumstances like a flat side when intended to be used on a surface, since–by means of their shape–unintended movement (rolling) is less a problem. A Stick has a notable orientation in space, but cannot be differentiated from another one by features other than position and orientation.[19]

Definition *A* Stick *is an elongated artefact, which is not differentiable from another Stick. It is incompressible, and not to bend. It has a user-observable orientation in space. Sticks may have differentiable tips. They are then called* directed.

Out of this definition, the following properties for Sticks (cf. to Figure 5.8) can be derived: It is obvious that the properties of placement, movement, and physical relations described for Grains hold true for Sticks, too. In difference to Grains, however, Sticks are stackable, though it is not easy to accomplish. As it was the case in the observation of Grains, turning the attention from one single Stick towards a set of ten or more adds many additional features, which all rely on their physical interrelationship. For example, graphical features like their orientation with respect to each other or the canvas are – unlike to Grains – far more obvious than their displacement behaviour. E.g. a collection of directed sticks placed on a surface has visual similarities to visualisations of flow fields. As shown in Figure 5.8(a), another feature of sticks is that they do not necessarily change their directional features when placed in containers; It only changes the interpretation of their current working state (*in use* when in the container, *off* when outside).

Section 9.2 will introduce *ChopStix*, a Tangible Auditory Interface that makes extensive use of Sticks as Tangible Interface objects. In *JugglingSounds*, juggling clubs are used as directed Sticks to control an Auditory Display for juggling (see Section 9.6).

5.5.4. Plates – Tangible Shapes

Introducing a second significant geometric extension to objects leads to Plates:

Definition *A* Plate *is a flat artefact with a dedicated shape and size. In difference to a surface, it is not used as a canvas for a TI, but is a manipulatable element that can be operated by users in the actual canvas. It has a dedicated position and orientation. It's interface design implications substantially differ whether it is used in 2D or 3D. It is effortlessly stackable.*

The usage of a Plate enables the user get an experience of his relation to the object and the room in which it is used. The implications of such a design and one possible use case are described at hand of TDS in Section 9.5. The there-featured application utilises a Plate as a medium to explore 3D data sets that are linked to the user-surrounding space. Another scenario would be to combine Plates with other Tangible User Interface objects. This would allow to use the Plates as temporary sub-canvasses for these other objects.

[19]For Sticks, the same holds true as for Grains: Of course they *are* always differentiable on a certain level of detail, but from the users point of view they are not.

5.5.5. Artefacts – Tangible Three-Dimensional Objects

Apart from the introduced object types, there are also other objects that can be utilised for TIs. As the diversity of objects is closely related to the accompanying object's complexity, it is impossible to give a complete classification of all object types in the scope of this thesis. These artefacts may be bend-able or have an internal state, making them complex and non-linear in their range of possible states. The curious reader may read on their utilisation in publications like Tangible Bits by Ishii et al. [IU97].

5.6. Conclusion

In this chapter I described the state of the art in Tangible Interfaces and their design as well as observations on their features and limitations. After a description of current state of the art in Tangible Interfaces, I proposed TI design guidelines, originating in my observations and other theoretical considerations. An essential part is the explicit utilisation of the level of abstraction (described in Section 5.4.3). By providing the user with the possibility to shift between different abstraction levels on the fly, Tangible Interfaces unfold their true potential lying in their status between the abstract digital (and therefore algorithmically processable) and the concrete physical reality.

Key features

With the design considerations described in this chapter, I identify the following features of TIs that should be considered during their design and implementation:

Iconic, Symbolic TI parts may either have a more iconic or symbolic binding to their associated data.

Container, Tokens, Tools TI parts represent data or algorithmic functions. Their degree of freedom is used to change the data's representation or other user-controlled parameters.

Spatiality Placement in space plays an important role in TIs since it is a prominent feature of objects that can be easily observed by the users. These positions are always seen as relative to certain landmarks, either provided by the TIs canvas or other hardware parts.

Constructive Users may be able to construct a meta-object out of different parts of the TI.

Active, Passive TIs can be either passive or active. The latter means that they may change their physical state without the user's intervention. However, an active interface is, due to its need for appropriate actuators like motors or electronically controlled magnetic forces, a very complex attempt. This makes TIs primarily an input medium for digital systems, though it naturally displays the current controller state.

User Awareness Designers of TIs have to consider the usage intents of the underlying algorithmic processes, i.e. whether it is used from time to time to adjust parameters of an ambient display, or it serves as the primary controller for i.e. active data exploration tasks.

Together these features form a good fundament for the integration of TIs into the TAI paradigm.

6. Information Displays

Information Displays mediate data to people. They are dedicated to transfer information on specific abstractly stored and processed aspects of the virtual realm into a physical and human-perceivable reality. Information Displays can be seen as *information media*: when used, their existence tends to fade into the subconscious. In the Heideggerian sense of our *Being-in-the-World*, they then change their role from being *Present-at-Hand* to *Ready-to-Hand* [Hei27].[1] Of course, these displays are still part of the communication process, adding their characteristics to the displayed data stream. Hence, they are not transparent as they are often imagined by their users. This circumstance is featured in more detail in Section 3.4, where indicators for data representations used in Exploratory Data Analysis are described. Closely related to this issue, an overview on ambient as well as directed information displays is given in this chapter, followed by a discussion of their usefulness in situations in which data has to be mediated from a technical and computer-optimised representation to a more human-oriented embodiment. This is followed by a brief overview on common information display types and by a section that explains why it makes sense to represent data by a diverse set of different display styles that also incorporate realtime user manipulation.

6.1. Display Types

Information displays make use of either the visual, auditory, or tactile senses. Despite the majority of systems that rely on a single sense, vision in most of the cases,[2] a combination of modalities is increasingly considered as useful by the display research community: Due to the extended number of dimensions compared to standard mono-modal displays, such multi-modal displays are able to distribute the data representation to more channels and therefore heighten the transparency of the data mediation system. Also, the multi-modality enables the display designer to concurrently feed the human perception system with different information streams, and exhaust their characteristics regarding structure recognition. In the following paragraphs, an overview of the integration and possible usage of the different modalities is given, showing how they are currently integrated into display techniques, and discussing their particular features.

Visual Information Displays Visual displays use the visual modality to transfer information. As the most common way of mediating digitally stored information, it is widely used for almost every data representation task. I will address its usage and application in Section 6.2.

[1] Heidegger's original terms where *In-der-Welt-sein* (*Being-in-the-World*), *Vorhanden* (*Present-at-Hand*) and *Zuhanden* (*Ready-to-Hand*).

[2] Sound is also used sometimes. Currently, there are only vague investigations into the presentation of information by the olfactory and gustatory senses.

Figure 6.1.: Design study of a dynamic information stream visualisation.

Auditory Information Displays As suggested by their name, Auditory Displays use sound to transfer information. Prominent examples are Earcons, Auditory Icons, Audification, Parameter Mapping Sonification and Model-Based Sonification. Since this work focuses on the connection of Tangible Interfaces with Auditory Displays, I dedicated Section 6.3 to the explanation of this field.

Tactile Information Displays The use of actuated physical objects as tactile information displays is a rather young technology, which has undergone active development, especially for supporting systems for visually impaired people. Various approaches exist, starting with Braille-based symbolic displays that aim to replace the usual computer screen, over more analogue devices that utilise mechanical actuators to represent dynamic images as reliefs, up to active Tangible Interface objects [FMI00]. Since this work does not focus on these interfaces, I will not go into further detail. An overview on these systems can be found in the thesis of Riedenklau [Rie09].

Smell and Taste As of today, smell and taste as display technology are underused due to technical reasons. However, there are basic attempts in producing digital-controlled interfaces to mediate information via these senses to humans [BMM06].

6.2. Visual Displays

The visual sense plays an important role in data display technologies. It has several independent parameters that can be classified into positional parameters (3 dimensions), time (1 dimension), colour, and texture (both multidimensional with a complex interrelationship).

Perception

All these freely adjustable parameters are perceptually very different. The three dimensions of the position for instance build a mathematical base, since their perception is completely interchangeable. Time, on the other hand, is a one dimensional variable that is perceived sequentially and cannot be reversed.[3]

On a different level, the colouring of a display has its origin in how we perceive light, a manifestation of electromagnetic energy. When interpreting it as a wave, each light source has a characteristic spectrum that may be filtered by partially absorbent material. However,the human light senses are only able to differentiate between three bins in this spectrum called red, green, and blue. Every other perceived colour is a result of a combination of energy in these bins, however, a changing distribution of energy inside the bins does not have a perceivable effect. This means, that it is possible to perceive two light sources to have the same colour, though they have a different spectrum. Even more complex, the three

[3] The illusion of reversing time is a result of reversing the time-based playback of a parameter stream.

dimensions of coloured light are not perceived as equivalent, which prevents it from being used as three independent parameters. This feature differentiates colour from position.

The two low-level visual qualities of texture and shape can be controlled with a high-dimensional parameter set. However, their dimensionality is difficult to determine and relies on the actual task. Perception-wise, there is a big dependency between the parameters of texture and shape. One parameter influences the perception of the others. Although the interdependency is a promising source for additional research, it will not be further elaborated in this thesis, since it does not contribute to the argumentation of this thesis. For more information, please refer to G. Stiny's book on *Shape* [Sti06].

6.2.1. Examples

There is a large number of widely used and known techniques for visual displays. However, it is clearly out of the focus of this work to provide a detailed overview. I therefore restrict the following description to only a few landmark techniques and recommend to Berthold et al. for a more detailed overview [BH03].

Geometric-based

Geometric-based displays are primarily used to represent multidimensional data sets. Representatives of this class of visualisations are e.g. *scatter plot matrices* [Cle93] [And72] or *projection pursuit* techniques [Hub85], by which users may define how data is geometrically shown to them, e.g. as a vector field or as parallel coordinates.

Iconic

In a completely different approach, *Iconic Displays* (or *Glyphs*) map attribute values have derived from multidimensional data to features of otherwise unrelated icons. Standard icon sets are e.g. *Stars* [War94], *TileBars* [Hea95] or *Chernoff Faces* [Che73].

Dense pixel

In the Dense Pixel display type, each dimension of a data item is mapped to the colour of one pixel element. All pixels for one data item are then grouped together and represented against other *Dense Pixels* rendered from the other data items [Kei00].

Stacked

Stacked display techniques represent data in a hierarchical concept. The *Dimensional Stacking* technique introduced by le Blanc et al. [LWW90] therefore embeds several visualisations into one bigger meta-visualisation. The display fields may be rendered according to one of the other visualisation techniques.

Dynamics

All above-described visualisation techniques can also be expanded by introducing time-based dynamics. An example for a Dense Pixel Display implemented by me as a design study is shown in Figure 6.1.

6.3. Auditory Displays and Sonification

In contrast to the human visual perception, the auditory senses are well developed concerning time varying structures like rhythms or patterns. Other features of the auditory modalities include a native multi-person involvement caused by its undirected nature, mass delivery of information, native support of time-based structures, and the possibility to flexibly change of being subconscious or alarming.

Auditory Displays utilise these features for information mediation by representing data and algorithmic structures via sound. As of today, six types of Auditory Display techniques are distinguished: Auditory Alarms, Auditory Icons, Earcons, Audification, Parameter

Mapping and Model-Based Sonification [Her08]. While the first three types can be classified as (static) Auditory Displays, the latter three are usually grouped under *Sonification*. The evolving discipline of *Sonification* is therefore a sub-field of Auditory Displays that uses the human auditory capabilities to (dynamically) represent data. It is defined by Hermann [Her08] as

> A technique that uses data as input, and generates sound signals (eventually in response to optional additional excitation or triggering) may be called *sonification*, if and only if
>
> **(C1)** The sound reflects objective properties or relations in the input data.
>
> **(C2)** The transformation is systematic. This means that there is a precise definition provided of how the data (and optional interactions) cause the sound to change.
>
> **(C3)** The Sonification is reproducible: given the same data and identical interactions (or triggers) the resulting sound has to be structurally identical.
>
> **(C4)** The system can intentionally be used with different data, and also be used in repetition with the same data.

Auditory Alarms In complex working environments such as medical operating rooms or hospitals in general, there are often alarm signals needed, indicating that a reaction to a certain incident is demanded. Research in *Auditory Alarms* specialised in the design and analysis of such alarms, focusing on their separation and differentiation into categories like urgency and type.

Earcons If discrete states of a system (e.g. a computer program) are given, each of them can be mapped onto a specific predefined sound to display its current state. This approach is called *Earcon* and was introduced 1994 by Brewster et al [BWE94].

According to the very limited and explicit mapping, there are only limited possibilities to apply this method to data exploration. For example the discrete results of a classification method could be mapped to specific sounds. In this case, the designer has to act according to psychological surveys of the human perception system.

Auditory Icons Sounds of everyday actions are mapped onto equivalent virtual events, like the click-sound that can be heard when pressing the shutter release of an analogue camera is mapped onto the button of its digital equivalent to let the user know that he took a picture [Gav94].

Like the earcons, this approach is only limited in use for data exploration since the choice of sounds depends to a large extend of the specific data domain.

Audification Let $(\mathbf{x}^\alpha)_{\alpha=0\ldots m}$ be an ordered list of multidimensional data records $\mathbf{x}^\alpha \in \mathcal{D}^n$, which can be represented as a discrete time series. Usually the number m of items is high, and an order is given by the measuring point in time. Examples for this data type are electroencephalograms (*EEG*) or seismographic data (see Section 2.1). Such data types are candidates for directly mapping onto loudspeakers as audio streams on several channels, each for one dimension of x_t. This is achieved by setting each multichannel-sample $s[t] \in \mathbb{R}^n$ to the appropriate data item x^t. If there are variations in the data set, they will be translated into variations in the samples, which leads to audible effects, providing that the variations lie in the human audible range. This method is commonly known as *Audification*. Audification of seismographic data was investigated by Hayward [Hay94] and Dombois [Dom01] [Dom02].

One prominent disadvantage of this approach is, however, that periodic patterns of the data have to be located in the audible frequency range of the human ear, ranging from approximately 40Hz to about 4kHz. It can be solved by pitching the whole data set, i.e. by compressing or stretching the time axis. This possibly leads to another problem based on the linkage of temporal and spectral components in audio streams: Given a sampling rate of 44.1kHz, a data set must consist of 44 100 data items to produce one second of sound. Therefore the data set has to consist of many data items in order to let the Audification produce any perceivable sound. To avoid these problems, many strategies have been developed. Utilising Granular Synthesis (cf. to Section 6.4.1) for audification, as described by de Campo [dCFH04] allows a completely independent control of temporal and spectral components. However, such technologies introduce additional algorithmic-based artefacts that may occlude the actual data representation.

Parameter Mapping Sonification Given an ordered multidimensional data set

$$\left(\mathbf{x}^0, \mathbf{x}^1, \ldots, \mathbf{x}^{m-1}\right) = \mathbf{X} \in \mathbb{R}^{m \times n}. \tag{6.1}$$

Each data point $\mathbf{x}^\alpha$ can be mapped by a function $\mathbb{R}^n \to \mathbb{R}^p$ to a p-dimensional parameter vector of sound attributes that are used to feed a predefined sound rendering process. Take for example three-dimensional data points

$$\mathbf{x}^\alpha = (x_0^\alpha, x_1^\alpha, x_2^\alpha)^\tau; \quad \alpha = 0 \ldots m-1 \tag{6.2}$$

A Parameter Mapping Sonification may characterise m sinusoidal grains[4] using the identity as mapping function. In this case, x_0^α is mapped onto the α'th grain frequency, x_1^α onto its amplitude and x_2^α onto its duration. This way, a Parameter Mapping Sonification can use all available audio signal parameters to convey the given data values to the user.

To specify the mappings in a meaningful way, there has to be a specific *mapping methodology* evolving from the data domain. If there is no domain-specific information available, it cannot be determined if one mapping is better than the other: Since the different sound synthesis parameters change the resulting sound in differently well perceptible ways, every mapping methodology may procures an unintended structure that possibly drown out the data inherent structural information.

Model-Based Sonification Why should data produce sound? Normally data can be perceived as passive elements of the world or of its description. Passive elements only act in response to another action; they are *reactive*. From this point of view, one has to excite these passive objects to get structure-born sounds. This suggests to design an excitable model in which one can include data to produce sound that is directly influenced

[4] A sinusoidal grain will be defined in Equation 6.3.

by the data and the stimulus, e.g. the interaction given by the user. This approach is called Model-Based Sonification. Thus the data becomes more or less directly the sounding instrument on which the user can operate [HR99]. According to Hermann, the following elements have to be defined for a *Sonification Model*:

Setup a model of dynamic elements in a vector space (model space),

Dynamics rules how elements in the model space interact and react to external triggers e.g. by motion equations, and their initial state;

Excitation option and parameters of the model that users can manipulate

Sound Link Variables variables linking dynamics of the model to physical audio signals

Listener sound wave transfer and receiver characteristics.

As the inventor of this technique, Hermann proposed several Sonification Models such as Principle Curve Sonification [HMR00] or Growing Neural Gas Sonification [HR04] featuring the Model-Based Sonification approach. In Section 9.5, I will present a Sonification Model as part of the TDS Tangible Auditory Interface.

6.4. Sound Synthesis Techniques for Auditory Displays

Every bit of information transferred to a listener via an Auditory Display has to be rendered by sound synthesis algorithms. Hence, the actual usage of these algorithms plays a major role in the design and implementation of Auditory Displays. This section describes fundamental sound synthesis techniques for Auditory Displays as they are used in the applications in Part II. I deliberately lay the focus on *Granular Synthesis*, *sound filtering*, and *spatial control*, as these are the main sound synthesis techniques used throughout this work. For a detailed overview of other sound synthesis techniques, I would recommend technical literature, e.g. by Moore [Moo90].

6.4.1. Granular Synthesis

> A complex sound may be imagined as a multicolored firework in which each point of light appears against a black sky [...] A line of light would be created by sufficiently large multitude of points appearing and disappearing instantaneously. [Xen71]

Grain cloud

One sound synthesis technique which is useful for Auditory Displays superimposes a lot of short sound events, so called *grains*, to compose a large sound cloud. This meta-event is called a *grain cloud* [Roa01], while the generic term for this synthesis process is *Granular Synthesis*.[5] In such a cloud, each grain typically lasts only for a short time that is close to the minimum of the human perceivable time for duration, frequency, and amplitude discrimination. When hundreds of these grains fill a cloud texture, even minor variations of their duration cause strong side effects in the spectrum of the cloud mass. Grain clouds therefore are predestined to be used as a synthesis technique for Auditory Displays of complex, multidimensional data. The advantage over other sound rendering techniques is the potential to change the sound quality according to a mass of nearly arbitrary

[5] For a more detailed introduction to Granular Synthesis, see also Roads [Roa85].

definable low level parameters in short intervals. This is possible because the source of the single grains can be rendered by almost any synthesis technique. This could be additive synthesis, subtractive synthesis, frequency modulation or similar as described for example by Roads [Roa96] or Moore [Moo90]. It is up to the sound designer to choose the appropriate technique. The multiplication of the synthesis' output by an amplitude envelope, e.g. a triangle function, forms a short grain. The synthesis mechanism, its control values and the quality of the envelope can be changed for every rendered grain.

In the following, I will describe only simple grain clouds. However, it is possible to extent the parameter range that can be used for Sonification by using other synthesis techniques for the grain rendering. Let

$$g(t) = \sum_{i=0}^{m-1} a_i \cdot (o_i - e(t)) \sin(2\pi f_i t) \tag{6.3}$$

with

$m \in \mathbb{N}$	number of mixed sine oscillators,
$e(t) : \mathbb{R} \to \mathbb{R}$	the amplitude envelope,
$f_i \in \mathbb{R}$	the oscillator's frequencies,
$a_i \in [0, 1]$	the maximum amplitude of the frequencies, and
$o_i \in \mathbb{R}$	the onset times delaying the amplitude envelopes

be a representation of a grain that is based on additive sound synthesis. A grain cloud c then is defined by

Definition

$$\mathbf{c} = (d_c, u_c, \mathbf{f}_c^\tau, \mathbf{a}_c^\tau, \mathbf{o}_c^\tau) \in (\mathbb{R} \times \mathbb{R} \times \mathbb{R}^m \times \mathbb{R}^m \times \mathbb{R}^m) \,. \tag{6.4}$$

This allows the sound designer to control the following parameters:

Cloud Density d the number of grains per second; a density of $d = 20$ means, that on average, each second of the grain cloud's life 20 grains are triggered,[6]

Grain Duration u the time, in which $e(t) \neq 0$,

Grain Oscillator Frequencies f_i

Grain Amplitudes a_i

Onset Delays o_i the duration that the amplitude envelopes of the single frequencies are delayed from the triggering event

With this simple grain cloud technique a wide range of dynamically changing sounds can be rendered. Its possibilities are e.g. shown in the Sections 9.2, 9.5 and 9.6.

Granular Resynthesis

A derivative sound synthesis technique is *Granular Resynthesis*, where an existing, pre-recorded sound is used as the base for granulation. Although based on a sound other than grain clouds, it has the same controllable parameters; in fact they only differ in the definition for the basic grain. The rendered base sound of standard Granular Synthesis is replaced by parts of the pre-recorded sound. This sound synthesis technique is implemented in *AudioDB* as described in Section 9.4.

[6] definition according to Roads [Roa01]

Explicit Granular Synthesis

Musical compositions that make use of Granular Synthesis often control their parameters by density functions that change dynamically in time, resulting in complex sound structures [dC09b].These statistical parameters like grain density or their granularity then describe the overall gestalt of the resulting soundscape. To get a certain complexity, each grain's parameters are determined by random values originating in the given densities. According to the law of the big number, the resulting grain cloud is then statistically correlated to the given density functions.

When dealing with data exploration, however, there are often many data items within a possibly high-dimensional vector space that have to be considered. Since one important goal of data exploration and thus also of its incorporated displaying technology is to find statistically interesting structures, it is possible to treat the actual data as the source for grain rendering parameters.

This makes the representation of data items as auditory grains a probate way to get first insights into the gestalt of data sets. Confronting the user with a soundscape that is completely based on the whole data set, he is able to get an overview of the overall structure, i.e. the density functions and possible correlations between data dimensions [FdC07]. In addition, it is still possible to focus on particular data items by changing the inner listening focus.

6.4.2. Sound Filtering

Signal filtering is the basis of subtractive synthesis. In order to create a parameterised – possibly pitched – sound $\hat{s}(t)$, a complex sound source $s(t)$ is extensively treated by both frequency and sound shaping filters:

$$\hat{s}(t) = f^1_{\Theta_1} \circ \ldots \circ f^n_{\Theta_n} \circ s(t) \tag{6.5}$$

with Θ_i being the parameter set for filter f_i. In this case $s(t)$ should be either aperiodic like white noise, pink noise, or brown noise, or (quasi-)periodic like a saw wave, a pulse wave, or an impulse stream. Filters are usually implemented either as spectral filters (using FFT) or by difference equations (FIR/IIR-filter). The most important filter classes are highpass, lowpass, bandpass and band-reject filters, yet the details would go beyond the scope of this section. For a more detailed introduction into sound synthesis, especially subtractive synthesis, please refer to Moore [Moo90].

Filtering can be applied to Parameter Mapping Sonification by mapping the synthesis parameters (e.g. pitch and Θ_i) to features that have to be extracted from the data under exploration. We utilise sound filtering in the Reim-setup (as will be described in Section 9.3) to augment structure-borne sounds with data-driven auditory feedback.

6.4.3. Spatial Control

> If you want the sound to come from a specific location, put a loudspeaker there.[7]

Spatialisation and its controlling-based equivalent Spatial Control are the terms for distributing sound in space in a controlled way. This sound rendering aspect is of particular

[7] Curtis Roads in a discussion on spatialisation; personal communication to Alberto de Campo, ca. 2000, quoted from memory.

interest for Auditory Displays, since it adds up to three dimensions to arbitrary sound sources. They can often be mapped intuitively, i.e. spatial measurements can be directly mapped to spatialisation parameters of sound sources. Thus, they can be used in all sub-fields of AD to increase the structural spreading of the data representation.

Spatialisation can be realised with several techniques:

Loudspeaker based The easiest and best solution to spatialise (positional static) sound sources is to place a loudspeaker at the point where the sound should appear. A prominent sound diffusion system for the performance of electroacoustic music is BEAST, the *Birmingham ElectroAcoustic Sound Theatre*.[8]

This technique is one option in the control paradigm of MoveSound (as described in Section 9.1).

Equal power panning In order to place sound sources between loudspeakers, and to be able to continuously move their position algorithmically, equal power panning implements a panning between several equidistantly arranged loudspeakers (either 2 in front (stereo setup), or more than two arranged in a circle, respectively sphere). VBAP is a multichannel approach utilising this technique for an arbitrary number of loudspeakers distributed equidistantly over a sphere [Pul01].

The sound synthesis of TDS (as described in Section 9.5) is an example for computationally cheap per-grain spatialisation based on equal power panning.

Wave-field synthesis An emulation of sound sources by a wall of loudspeakers in front of the virtual source. They simulate the sonic wave field as it would be at this position in case the virtual source actually emitted sound waves [Ber88] [Baa04].

Ambisonics Ambisonics is an algorithmic approach to spatial audio using multichannel reproduction systems. It is based on the idea that spherical harmonics can be used to encode and decode directions from which sound energy appears. It was developed independently by several researchers in the 1970s [Ger85].

Audio Spotlight An audio spotlight setup features directional sound emission. Particles in a beam originating in a parametric array of ultrasound emitters are excited by ultrasound waves using heterodyning. Since ultrasound has wavelengths much smaller than audible sound, it can be aimed in a much narrower beam. [YFKS83]

6.5. The Importance of Multi-Modal Displays

We experience our surrounding with the five human senses; sight, smell, hearing, taste, and touch. Due to evolutionary reasons, they all have their own right to exist, all of them play an important role in how we perceive our environment. Although it is possible not only to survive but live comfortably without one of the other senses, every sense has its specific application area in which it performs best. Its substitution by other senses always means to either lack valuable information, or to significantly increase cognitive load. Although we have five senses, it is obvious that the visual sense tends to be predominant in our conscious perception. Two examples for this phenomenon are the dominance of visual arts in our culture and, also, the current enthusiasm for taking pictures with digital cameras.

[8] `http://www.beast.bham.ac.uk/`

On the contrary, other senses are far less important in the view of many people, an observation that can be supported by the small number of people owning e.g. a sound recording device or a digital smell keeper. This is not only due to the technology for smelling devices that is not yet ready for portable devices, but also because people tend to actively perceive only visual cues; other senses are primarily processed in the subconscious part of our mind. This fact, however, does not imply that these modalities are less important for our decision-making, yet their influence is more subtle. From this perspective, also the history in the development of data analysis and data exploration tools make sense: Data is almost always represented in visual graphs or plots, even time-based measurements such as EEG data or audio are often decomposed into frequencies and then turned into visual representations.

The use of other modalities such as sound, however, is not widely accepted in the research community. It often is sufficient to show a graph of data to proof a thesis. This is by no means based in any objectively measurable significance of visual displays that our other senses might lack, but only supported by the fact that we tend to believe more in what we have visually perceived [Tuf83].

A multi-modal representation of data makes sense for exploratory tasks because every sense that is involved adds its characteristic structure recognition and analysis abilities. It allows users to perceive formative multi-modal experiences, rather than joining the value of all incorporated senses. The whole is perceived as something different than the sum of its parts.

7. Tangible Auditory Interfaces

In this chapter, I will introduce the *Tangible Auditory Interface* (TAI) paradigm, which combines controlling features of TIs with the display capabilities of ADs. I will show that the resulting human-data interface design can be used for powerful and productive work in various environments.

I therefore propose the following definition:

Definition *A* Tangible Auditory Interface *(TAI) combines Tangible Interfaces with Auditory Displays to mediate information back and forth between abstract data space and user-perceivable reality. The two parts form an integral system for the representation of abstract objects like data or algorithms as physical and graspable artefacts with inherent sonic feedback. The tangible part hereby provides the means for the manipulation of data, algorithms, or their parameterisation, whereas the auditory part serves as the primary medium to display the virtual dynamics to the users.*

Information flow

This definition implies the following information flow in TAIs (also visualised in Figure 7.1): The data or algorithmic functionality to be represented by the TAI is pre-processed by a (more or less advanced) model. The AD transforms the pre-processed data into sound that is perceived by the user. Depending on the perceived sounds and the user's imagination, he manipulates the TI, and thereby controls parameters of the data pre-processing, which results in a change of the auditory representation. From the user's point of view, the system directly reacts to his physical manipulations. Due to immediate and possibly diverse sonic reactions, a flow in operation is established, and the TAI is perceived to be Ready-to-Hand in the Heideggerian sense (as described in Chapter 6).

Design implications

Although technically every combination of AD and TI can be called a TAI, there are specific combinations that assemble into more powerful setups than others. This effect is based on the interconnection design that is established between the tangible control and the auditory output. The physical part of a TI hereby suggests possibilities on how and, particularly, in which detail the data set and its manipulation should be represented. Due to the inherent gestalt of the interface, the interaction designer is guided by object reactions he may find in the natural environment. In the same way, an AD representing a data set possibly arouses associations of the rendered sounds with the manipulation of physical objects. This observation can used to design a TI that complements the AD to an integral TAI. Both fields, TI and AD, therefore induce nature-inspired constraints to the design of their complementing part. This limits the number of possible representations for data and algorithmic processes such that only those are left that are based on commonly used associations. Let us consider for example the integration of an Auditory Display that represents a data item by a short auditory event into a TAI. The given association of one data item to one auditory event suggests the linkage of its manipulation to one physical object. The structure induced by the auditory representation then is reflected by the TI. Furthermore, the data-driven sound event can be linked with the object's physical

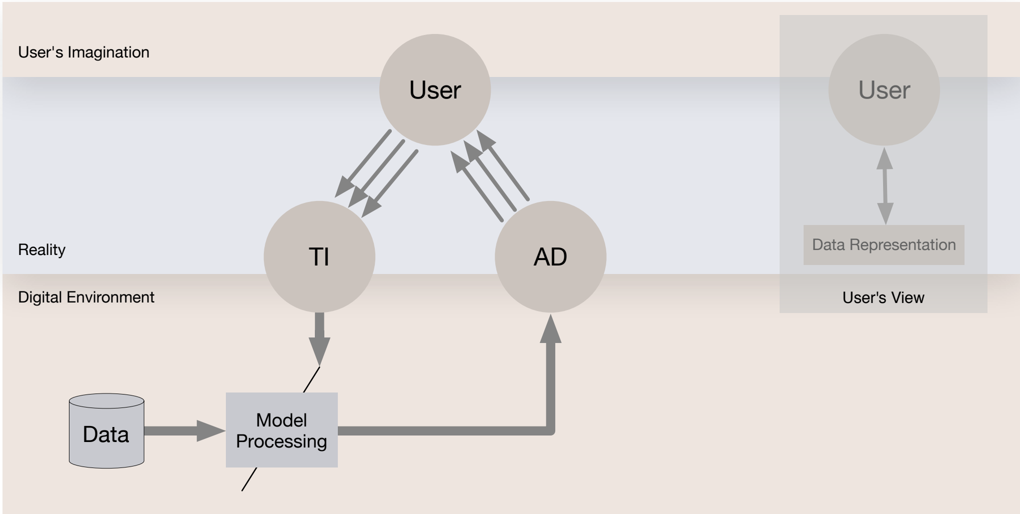


Figure 7.1.: Information flow in a Tangible Auditory Interface.

interaction with the TI's canvas. This effect then can be associated by the user with the physical interactions of rigid bodies, which are usually the cause for structure-borne sounds.

Superimposed and separated layers

In their publication *Bricks* [FIB95], Fitzmaurice et al. propose a design space for brick-like Tangible Interfaces. Among other things, they differentiate between *superimposed* (i.e. directly coupled), or *separated* (i.e. indirectly coupled) physical and virtual layers of a brick-utilising interfaces. By looking at TAIs, however, this point of view turns out to be biased by the means of the visual dominance in their displays: Looking at TAIs unveils that other aspects also have to be considered: We interpret observed sounds often as being connected to a synchronous visual action. This does not necessarily have to be the case, since the sound may have been caused by something completely different, but we are used to interpret the temporal correlation as one common event that causes both the visual and the auditory part. A possible reason for this may be that physical processes almost always generate sounds while changing their visually observable state, and it is very unlikely for a synchronised audiovisual stimuli to be caused by two unrelated events. Therefore, our mind tends to bind these time-synchronous events together, even if other features like their origin are contradictive. Since it is technically possible to trigger sound events at almost exactly the same time when other (e.g. visual) events are observed, the human mind can be tricked by making it believe that it observes an actual sound-action coherence. This effect is called the *ventriloquist effect on the perception and identification of sound with other simultaneously perceived values* [VdG04], or – in short, referring to one of it's discoverer – the *McGurk effect* [MM76]. Having this in mind it can be said that, differing from the prerequisites needed for the visual augmentation of tangible objects where either the objects or the canvas have to be electronically enhanced, TAIs do not require the placement of an active feedback system at the same location as the incorporated objects. Moreover, it is sufficient to surround the canvas with a spatial audio system (see e.g. Section 9.4), or, assuming an implementation that features a close coupling between action and auditory feedback, even a mono loudspeaker setup near the canvas is sufficient (see e.g. Section 9.3).

Certainly, it is also possible to electronically enhance the objects or their canvas, just as in the visual augmentation pendant. In addition, none of these audio-based implementations lead to situations in which users physically occlude the system's (auditory) feedback, a dedicated problem of visual display systems. Taking all these observations into account, it can be sais that the *direct* and *superimposed* indicators for Tangible Interfaces as proposed by Fitzmaurice et al. cannot be interpreted as a *duality* where either the one or the other is true for a system. Moreover, they have to be recognised as independent from each other when dealing with TAIs.

7.1. Key Features of Tangible Auditory Interfaces

Apart from only merging features of TIs and ADs as they were described in Section 5 and 6, the TAI paradigm also incorporates additional characteristics that evolve from the tight interplay of sensing and acting in their respective domains. This has implications for the TAI design. The following paragraphs list and explain the evolving key features.

Augmentation As mentioned in Section 5.4.2, the McGurk effect makes it possible to implement acoustic augmentations by linking objects with sounds that are not rendered from the exact location where their manipulation takes place. Because of the ambient, less-directional perception of sound, no masking by user manipulation takes place, which is often considered an important problem in vision-based systems. Nevertheless, it is possible to add space relations to auditory augmentations of objects.

Interfacing Richness Chapter 3 explained that a representation system for digitally stored data should reflect the complexity of the data to be represented. Since both fields, TI and AD, provide a rich interface in control respectively display, TAIs naturally support continuous modalities and therefore can offer a highly analogue representation. Rather than being structured by an algorithmic pre-processing system that results in a more symbolical representation, the mediated quantitative data can be cognitively processed and directly controlled by the user. This is a valuable effect especially for data exploration.

Immediacy and Flow TAIs actively support flow [Csi00] in both sensing– and display modalities. This effect can be facilitated by the interface designer by reflecting user manipulations with prominently perceivable changes in the Auditory Display. Also, the controlling accuracy of Tangible Interfaces combined with an immediate and precise feedback fosters the flow in TAI usage. A good example for such a tight connection between controlling and auditory feedback is the Reim toolset (as it will be described in Section 9.3).

Collaborative Work Tangible Interfaces support multi-person control by nature. Sound on the other hand is broadcasted and can therefore be perceived by many people at the same time. These two attributes fit well together, and should be considered in TAI design.

Ergonomics Sounds caused by user-action should be related to real-world experiences, e.g. much pressure should result in loud sound. This way, user-confusion by unusual attitude is minimised [BHR06a].

Tight Coupling As stated in the previous paragraph, the options for display and sensors

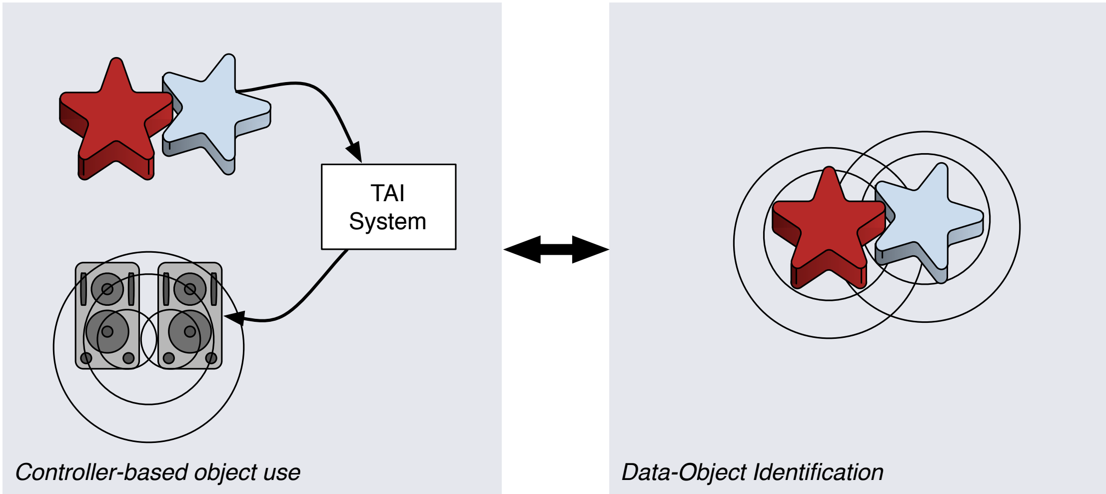


Figure 7.2.: Controller-based Object Use (left) vs. Data-Object Identification (right): The captured states of the objects are either used for real-time control of program parameters, or the users identify them directly with the referenced digital representation.

can and should be designed such that they are tightly coupled. The users' ability to understand how the interconnection is established should not be underestimated. True understanding of the system's reaction to user manipulation means that less reactions are considered to be wrong.

Ambience As stated before, ADs can be utilised for long-term monitoring. Such a subconsciously perceived display is optimally complemented by a lazy Tangible Interface that does not pro-actively change its state and does not need any active elements to keep that state.

7.2. Auditory Bindings for Tangible Interfaces

How can sound be systematically used in Tangible Auditory Interfaces? There are two different types of linkage that need to be considered: *controller-based object use* and *data-object identification*, as depicted in Figure 7.2.

Controller-based Object Use

Controller-based Object Use means that the controllable degrees of freedom of an object are used to control parameters of a data display. As an example application that shows the differences between the two linkage types, let us consider the design of an interface to control the spatial panning of a sound source. *Controller-based Object Use* then means to map e.g. the (x, y)-position of a tangible object to these parameters. The advantage opposed to classical GUI approaches (e.g. by instantiating sliders) is that (a) a potentially large number of parameters can be controlled simultaneously by manipulating one single object, and (b) –as in the example– parameter types can be mapped in a natural correspondence. In the same way, a binary parameter can be identified with object features with two natural states (e.g. a bottle that is either closed or open). Also, a sonic parameter ranging in a mathematical ring with values between 0 and 2π may be identified with the rotation of an object. Using this paradigm, auditory bindings for Tangible Interfaces are obvious in the light of existing Sonification techniques like *Parameter Mapping Sonification*, as in this case a large number of control parameters are needed to control the mapping.

Data-Object Identification

Data-Object Identification on the other hand means that a tangible object is actually identified with a data set. In other words, the object *becomes* the data and reacts on interactions and manipulations as convincing as possible. This linkage type is metaphorically stronger than the *Controller-based Object Use* discussed above, since the object is easily identified as the physical part of the data it is meant to represent. Under certain conditions, the above-described McGurk effect can enforce a data-object identification, even if the sound does not originate from the object itself: First, the acoustic response needs to be well synchronised to the object manipulation, and second, basic binding mechanisms need to be respected such as that a stronger physical interaction leads to a stronger response.

7.3. Application Fields

Data exploration

TAIs are useful in a broad range of application fields. Data exploration for example is a prominent application area for TAIs. They can serve as alternative interfaces, offering direct manipulation capabilities for data representation combined with a rich display that offers unusual insights into data via Sonification. Tangible Data Scanning (introduced in Section 9.5) is such a system. It is implemented to represent arbitrary three-dimensional data for EDA purposes.

Education

The multi-person capabilities of the TAI paradigm in conjunction with its interface richness in both control and display turns it into a comprehensive educational tool. In this situation, TAIs can be used to complement standard (visual) representations of data with Auditory Displays that mediate otherwise hidden aspects of underlying algorithmic or structural connections. The tangible part of the TAI can then be used to manipulate the visual representation, which in turn changes the auditorily represented parameters. Such an application will be described with Durcheinander in Section 9.7.

Monitoring

Action-based work such as physiotherapy, sports, or artistic activity, typically require the monitoring of body– and object motions. The TAI paradigm offers the technology to connect Auditory Display technology (incorporating a sense that is less often used in co-ordination tasks) with the used artefacts, adding an additional sonic augmentation. With JugglingSounds, we exemplify this by the example of juggling. It will be described in Section 9.6.

Soundscapes

Data-driven soundscapes will be ubiquitous in future home and marketing environments. Ambient data displays will mediate information on stock quotes or current and forthcoming weather situations. For such a scenario, TAIs offer building blocks to control such soundscapes. MoveSound and ChopStix (as will be introduced in Section 9.1 and Section 9.2) are proof of concept implementation for such use cases.

Subconscious data monitoring

The feature to augment everyday physical objects with data-driven Auditory Displays presents system designers with a toolset for the creation of data representations that mediate information subconsciously, while these representations can still be controlled by a Tangible Interface. Two applications covering this field, although with completely differing focuses in control- and display design, are ChopStix (Section 9.2) and Reim (Section 9.3).

Part II.

Systems Incorporating Tangible Auditory Interfaces

8. Overview

In the first part of this thesis, I introduced Tangible Auditory Interfaces as a new paradigm for user interfaces incorporating Tangible Interfaces and Auditory Displays. I described the background of related disciplines and showed how features of Tangible Interfaces and Auditory Displays fit together, forming a coherent gestalt as Tangible Auditory Interfaces, where both areas profit from the interconnection.

In this part, I describe applications and frameworks that I developed to support the ideas behind the TAI paradigm, illustrating their usefulness and potential. Chapter 9 covers a selection of seven applications that were developed over the course of this PhD research. I selected them, because they all incorporate parts of the theoretical considerations made in Part I, and therefore exemplify the introduced features of TAIs.

MoveSound MoveSound is an azimuth panning interface for up to 16 sources in a ring of an arbitrary number of loudspeakers. Both the position and width of arbitrary sound sources can be adjusted with it. By providing the user with an interface to select either one or more sources to operate on, the system allows to control several such sources at the same time.

ChopStix A Tangible Auditory Interface that is designed to display and control spatially distributed data. It reclaims the abstract of digitally processed information, and provides a different access to environmental data. Furthermore, it is a calm system dedicated to reflection by interacting with an auditory representation of this abstract data via everyday-like artefacts. Its users are immersed into weather data as measured throughout Europe in near real time. Hereby, the raw data provided by independent sources are stirred together by the user, forming an informative auditory stream. Its Tangible Interface allows users to express their idiosyncratic approach in understanding this data mix. The resulting configuration of the used Stix is a trace of each user's personal preference.

Reim This toolset incorporates sound as a basis to represent data. Its lightweight, modular concept intends to help creating data-driven object augmentations. Systems build according to Reim draw on peoples knowledge about every-day objects, whether they are simple like sticks and stones, or more specialised and integrated into daily, technical-driven systems as keyboards or other computer interfaces. Rather than manipulating the object's intentional usage to represent data, Reim transforms the object's sonic characteristics to augment it with external information. This means that the sonic reaction to e.g. an excitation of such an enhanced object does not only reflect its own structure, but also features the attached data: It is virtually changed to render an additional information layer by data-driven features.

AudioDB This surface-based Tangible Auditory Interface was designed to support collaborative navigation in information databases. It provides a tangible environment to sonically sort, group and select auditory representations of data, represented as physical artefacts on the canvas. *AudioDB*'s intend was to serve as a low-threshold interface to audio data that is used by several people during a discussion.

Tangible Data Scanning A TAI that is attached to a Sonification Model following the Model-Based Sonification approach. TDS provides direct access to abstract data sets using auditory augmentation and physical interaction. To achieve this, the space surrounding the user is augmented by the data-inherent vector space. Each data item is linked to a fixed location in real space. With TDS, the user is able to scan through these data items utilising a physical tool he holds in his hands. This Tangible Interface, a plate connected to a virtual geometric body, scans through acoustically active but virtual objects that represent the data.

JugglingSounds A system for real-time auditory monitoring of artistic juggling and swinging. It provides auditorily represented information on the juggling clubs' movement to the juggler and the audience. It was intended to help improve juggling skills by increasing their awareness for details in their movements, to unveil the nature of juggling patterns for scientific, kinesiological research, to mediate juggling to visual impaired people (whether as the audience or the artists) and to serve as an aesthetic element of the artistic performance on stage.

Durcheinander A system to help understand Agglomerative Clustering processes as they are used in various, often visually oriented data mining and exploration systems. It consists of several small objects on a tabletop surface, which represent data items in an artificially generated data set. A computer vision system tracks their position and computes a cluster dendrogram which is sonified every time a substantial change in this dendrogram takes place. Durcheinander may be used to answer questions concerning the behaviour of clustering algorithms under various conditions. We propose its usage as a didactical and explorative platform for single- and multi-user operation.

This detailed application overview is followed by Section 10, which contains descriptions of the hardware framework TUImod, and the software framework SETO. They were designed and implemented to support the implementation of TAIs as described in Chapter 9.

SETO The *SuperCollider Environment for Tangible Objects* is a modular software architecture based on the programming language SuperCollider for just in time development of objects-based Tangible User Interfaces. Since SuperCollider itself is a language especially suited for high- and low-level control of sound rendering processes, the addition of SETO makes it a valuable tool for rapid prototyping of Tangible Auditory Interfaces.

TUImod TUImod on the contrary is a modular hardware system with which custom tangible objects can be easily assembled. It provides stackable modules to build objects that can be tracked by a computer vision system, optionally has mechanical constraints, and a coloured top plate that indicates its functionality to the user.

9. Applications

9.1. MoveSound

MoveSound is an azimuth panning interface for up to 16 sources in a ring of an arbitrary number of loudspeakers. Both the position and width of arbitrary sound sources can be adjusted with it. By providing the user with an interface to select either one or more sources to operate on, the system allows controlling several such sources at the same time. Together with the integrated azimuth- and width-panning control, its functionality opens the field of dynamic sound spatialisation also to untrained users. MoveSound was designed as a software system that can be easily attached to human interface devices. Its usage scenarios are (a) spatial control of unobtrusive ambient soundscapes, and (b) dynamic spatial control of sound sources for artistic contexts.

Usage scenarios

Figure 9.1.: The MoveSound Logo.

Implementations for two devices were done, one for the *PowerMate* by *Griffin Technologies Inc.*, one for the *SpaceNavigator* by *3Dconnexion Inc.* This section describes the general setup, its original motivation, and use cases as well as technical details on the actual implementation.

9.1.1. State of the Art – Spatialisation Controls in Digital Audio Workstations

There are already many controlling interfaces for surround sound systems, mainly used in professional audio mixing systems (i.e. 5.1, 6.1, etc. formats). We differentiate here between hardware and GUI-based interfaces.

Hardware Interfaces

Like many other hardware interfaces for surround mixing, the two interfaces *Cakewalk VS-700C V-Studio* console and *DigiDesign ICON D-Control Surround Panner* shown in Figure 9.2 both feature one, respectively two joystick-like hardware controllers, which let the user control the position of one sound source in a surround mix. The Digidesign console also features a touch-screen interface for sound-position control and automation recording.

GUI Interfaces

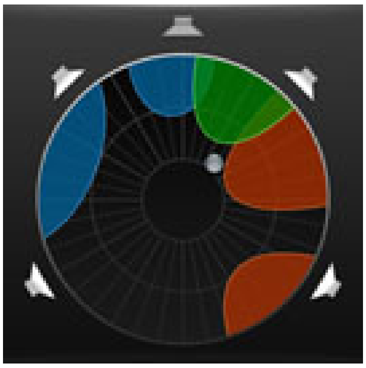
GUI-based interfaces for sound spatialisation work in a similar way. As Figure 9.3 shows, they typically feature a canvas on which the loudspeaker positions and phantom sources are shown. Their layout either is based on an ideal listening room (a square), or on a ring. Additional parameters like panoramic width or overall amplitude are represented by more classic GUI elements like sliders or knobs.

(a) Cakewalk VS-700C V-Studio Console.

(b) DigiDesign D-Control ES Multichannel panning section.

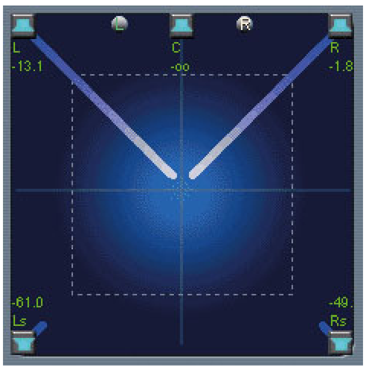
Figure 9.2.: Hardware surround panning interfaces.

(a) Logic Pro Audio surround panning GUI.

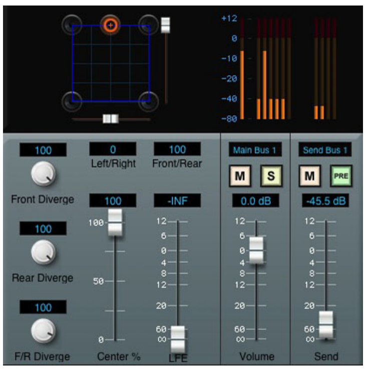


(b) Logic Pro Audio surround panning GUI.

(c) Logic Pro surround panning: trackview.



(d) Cubase 5.1 mixing interface.



(e) Neyrinck Monopanner 5.1.

(f) Cakewalk surround panner.

Figure 9.3.: Software surround panning interfaces.

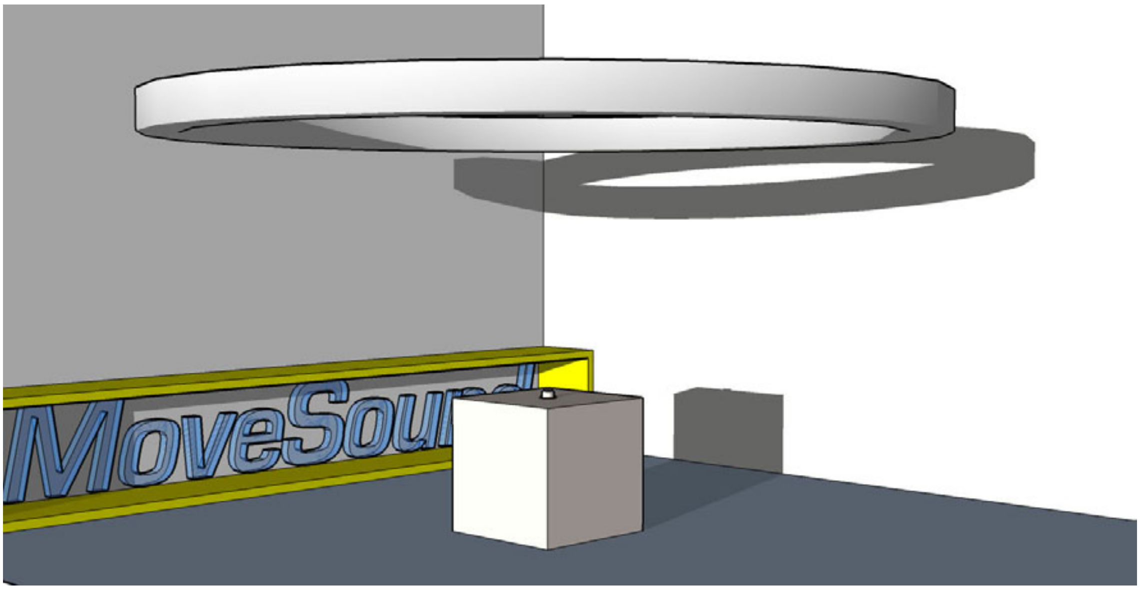


Figure 9.4.: Design study of the MoveSound environment. The tangible controller is located in the centre of the loudspeaker ring.

9.1.2. Motivation

We developed MoveSound to provide non-expert users with an accessible way to control spatial aspects of potentially complex soundscapes. Such a system can be useful in surround sound applications, in spatial Auditory Display systems, and in the media arts. Possible scenarios include the control of a spatial sound system in a living-room of a future family home, supporting the creation of artificial soundscapes or staging artistic soundscapes.

MoveSound's interfacing level focuses on two aspects: (a) the power of tangibility and directness complemented by algorithmic abstraction, and (b) its role as a test-system for the interplay between people and a minimalist Tangible Auditory Interface.

(a) MoveSound illustrates the power of *TAI*s regarding their potential in tangibility and directness in sound manipulation: It proves that even a minimalist interface (reduced both in hardware and controller-sound mapping) can provide full control over multiple spatial parameters of sounds. At the time of writing, it is part of the Ambient Intelligence Laboratory of the Ambient Intelligence Group in Bielefeld University. There it is used to dynamically control various soundscapes including Sonifications.

(b) Furthermore, we were interested in a test system for user studies in interaction design for sound-controlling systems. It was intended to be a platform to study user interaction patterns with a (simple) hardware controller. The research question here was whether the system could provide users with sufficient control over spatial and volumetric aspects of auditory information displays.

MoveSound distinguishes itself from other approaches to spatial control (as discussed in Section 9.1.1) by including the loudspeaker setup in the system's design.

(a) Griffin PowerMate (b) 3dConnexion Spacenavigator

Figure 9.5.: Tangible input devices for MoveSound.

9.1.3. Setup

Tangible control

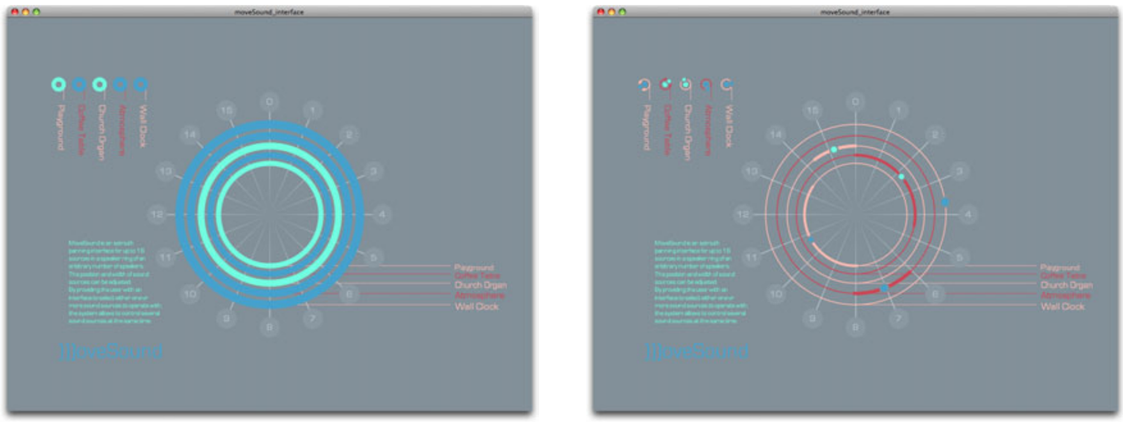
MoveSound provides a multi-modal front-end connected to a lightweight software interface to sound spatialisation and other controlling mechanisms. Plugins for two hardware interfaces were developed, each offering a different kind of interface to control the spatial parameters of interest: Depending on the chosen controller, the user is able to either adjust sound position and width via relative, or incremental adjustment. The PowerMate serves as an input device for relative adjustments (see Figure 9.5(a)). It changes the orientation of the activated sound source when turned, and when pressed and turned it controls the sound's spatial spreading. Incremental adjustment of the sound sources is supported by a SpaceNavigator, a 6DOF-capable 3D-mouse with two additional buttons (located left and right to the actual controller-cap. (see Figure 9.5(b))

Meta features

Besides the interface for sound source panning and spreading, MoveSound also offers a broad feature-set to control its general appearance and behaviour. One of these features switches MoveSound's audio spatialisation between continuous (sound sources can be placed anywhere on the ring) and discrete (sound source locations are fixed to an actual loudspeaker). Rejecting *phantom sources* (which are very common in sound production) establishes direct identification of a sound with a physical sound source, a loudspeaker. Some media artists prefer distributing sounds using such a single physical source concept. Additionally, a broad variety of virtual loudspeaker setups can be simulated, as MoveSound can also quantise locations to both real and virtual loudspeakers. This allows to perform artistic pieces that were originally designed for a different number of loudspeakers than available.

Graphical User Interface

MoveSound also offers visual feedback about its current state. Its visual display can be set up to either show detailed information (Figure 9.6(b)), or to display only the current source selection (Figure 9.6(a)). As described in Section 9.1.6 the choice of visual feedback determines the users focus. The detailed graphical display attracts their visual attention, and they report to feel fully in control. The reduced visual display tends to lead users into more awareness of the spatial situation created in the soundscape and the physical hardware setup. This tendency becomes even stronger when MoveSound quantises the sound locations to the physical loudspeakers, which then become a visual cue for the sounds' actual origin.



(a) Reduced graphical interface. (b) Full graphical interface.

Figure 9.6.: Graphical User Interface of MoveSound in full/reduced mode.

9.1.4. Level of Abstraction

As explained in Section 5.4.3, changing the level of abstraction is a central aspect of Tangible Auditory Interfaces: It unfolds their capabilities by combining physical and algorithmic features. MoveSound's design allows to change the level of abstraction in two ways: by *Sound Source Selection* and by *Controller Playback*.

Sound Source Selection means that the system lets the user control multiple sources at once, i.e. he can change orientation respectively width of selected sounds in one operation. This can be seen as a meta-switch to change the level of abstraction of MoveSound, since it changes the meaning of the central hardware controller to link to more, or a different set of sound sources.

Controller Playback means the system' s potential to record and play back all parameters such as orientation, width or volume of attached sound sources as they were adjusted by the user. All these non-programatic interactions are internally collected and can be played back later. It is also possible to change playback speed if desired.

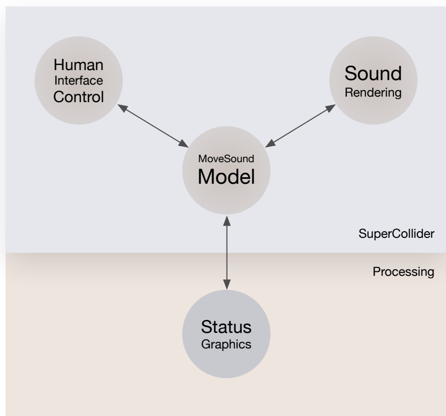


Figure 9.7.: MoveSound's modules.

9.1.5. MoveSound's Technical Aspects

MoveSound is realised in two programming languages, SuperCollider [McC02] [WCC09] for behaviour modelling and sound control, and Processing [RF07] for graphical output. Communication between these parts was realised via OSC [WF97] [WFM03]. As shown in Figure 9.7, the software implementation is divided into four primary parts:

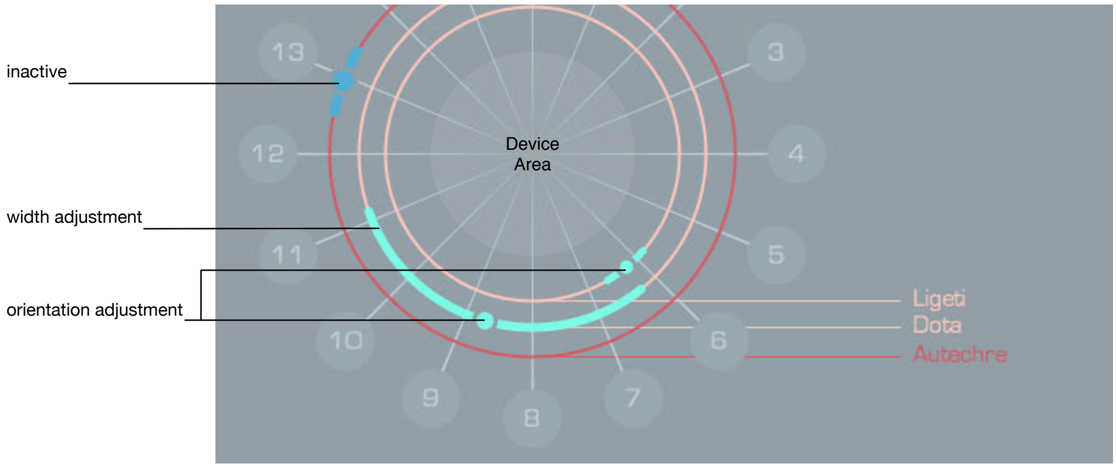


Figure 9.8.: Source Selection: *Ligeti* and *Dota* are set active.

MoveSound Model the unit in which the processing and distribution of all MoveSound-relevant information takes place,

Sound Rendering the interface to the sound rendering process,

Human Interface Control the connection to supported Human Interface Devices, and

Status Graphics the logic of the graphical display.

Before delving further into the actual implementation details, we first give an example of the usage of MoveSound under normal conditions, e.g. when prototyping a spatial soundscape.

Usage Example Code

Before operating with MoveSound, we have to create the basic interface. Please note that we have to specify the absolute number of sources we want to use at initialisation time. In the next code listings, a **MoveSound** instance is instantiated and basic parameters are set.

```
MoveSound.runExternalViz;// start external visualisation process

m = MoveSound.new(4, 0.5, 3, server: s);
m.virtualChannelOrientation = 0;
m.numVirtualChannels = 16;
m.isDiscrete = false;
```

To exemplify audio routing through MoveSound, we first create **NodeProxy** objects.

```
s.boot; // sound server
p = ProxySpace.push(s);  // create ProxySpace

// synthesis definitions
~a = {Impulse.ar(5)};
~b = {
    Klank.ar(
        `[{exprand(100, 700.0)}!24,
            {0.5.rand}!24,
            ({10.0.rand}!24)
        ],
```

```
        PinkNoise.ar(0.001)
    )
};
~c = {
    Klank.ar(
        `[
            [ 4800, 6426, 6918, 10338 ], nil, [5, 5, 5, 5]
        ],
        Dust.ar(1, 0.1)
    )
};
```

To play them via the MoveSound interface, we use the convenience method **playBy**. It's implementation is explained after this user-centred introduction.

```
~a.playBy([m, 0, "Impulse"])
~b.playBy([m, 1, "Steady Klank"])
~c.playBy([m, 2, "Crisp Chime"])
```

To control the width and orientation of the sounds in the loudspeaker ring, we now add a human interface device (HID) to it. As an example, we show here the creation of a *PowerMate* connection.

```
// setup an HCI MoveSound Controller
GeneralHID.buildDeviceList;
MoveSoundControlPower(m)
GeneralHID.startEventLoop
```

Besides controlling MoveSound via HID interfaces, it is possible to algorithmically change orientations, volume, and width either per source, or on a previously made selection of sources:

```
m.activeSources = [];        // none selected
m.activeSources = [0, 1]; // 0 and 1 selected

m.widths = {1.0.rand}!3; // change parameters
m.vols = {1.0.rand}!3;
m.orients = {2pi.rand}!3;
m.numVirtualChannels = 16;

m.name_(1, "Dota");  // set names of channels
m.name_(2, "Autechre");
m.name_(0, "Ligeti");

m.visionMode_(\full) // change visionMode
m.visionMode_(\reduced)
```

For demonstration purposes, MoveSound also provides the feature to display arbitrary text in one area of the *Status Graphics* module. This is also useful for experimental setups to provide information on the current task:

```
m.description("MoveSound is an azimuth
panning interface for up to 16
sources in a loudspeaker ring.");
```

As we had to test various colour schemes for the Status Graphics in order to find a variant that supports the user's recognition of active sources also on the limited colouring of the used projector, the system allows to change the colour scheme of the graphical interface on the fly, incorporating an interface to Adobe Kuler:[1]

```
m.colorKey = "337112"; // the themeID
```

Colour of functional items are set by their index:

```
m.backColor = 0;
m.activeColor = [3, 1]; //activated/deactivated
m.domainColor = [2, 4];
m.widthColor = [4, 2];
```

As mentioned above, it is possible to record and play back all control changes made via the MoveSound software interface.

```
m.startRecording;
m.stopRecording;
    // do some fancy stuff with the controllers
m.record.printAll; // print array of recorded data

a = m.replayTask;  // get data for playback
a.play;  // start playback
```

After focusing on the user's view of MoveSound, we examine its actual implementation by looking at the four core modules as they are shown in Figure 9.7.

MoveSound Model The *MoveSound Model* holds all relevant data for spatial control of sound sources, namely the link to the sound itself, its volume, panoramic width, and orientation. It also keeps track of which sources are currently active for manipulation. All other parts of the system are connected to it. This model is also responsible for recording and playback of control manipulations. Additionally, the model implements the sending part of the OSC interface to control the *MoveSound Graphics* (cf. Table B.1). The full interface is shown in Figure 9.9.

Sound rendering For the audio part, we decided to base on the *JITLib* **NodeProxy** framework as a flexible sound mapping and control architecture [RdCW05]. MoveSound was implemented such that it can be used to control spatial parameters of the **Monitor** implementation. To get this functionality, an extension of the **BusPlug** interface was required.[2] The added **playBy** method realises a convenient way to connect one control slot of a MoveSound instance with a specific **NodeProxy**:

```
+ BusPlug {
    playBy {arg whoAndHow, outs, amps, ins, vol, fadeTime, group, addAction;
        var who, how, name;
        whoAndHow.notNil.if({
            #who ... how = whoAndHow;
            who.isMemberOf(MoveSound).if({
                this.playN(outs, amps,
                    ins, vol, fadeTime, group, addAction, \az, who.numChannels);
                // add source to who at idx "how"
                who[how.first] = this;
```

[1] http://kuler.adobe.com
[2] **BusPlug** is a superclass of **NodeProxy**

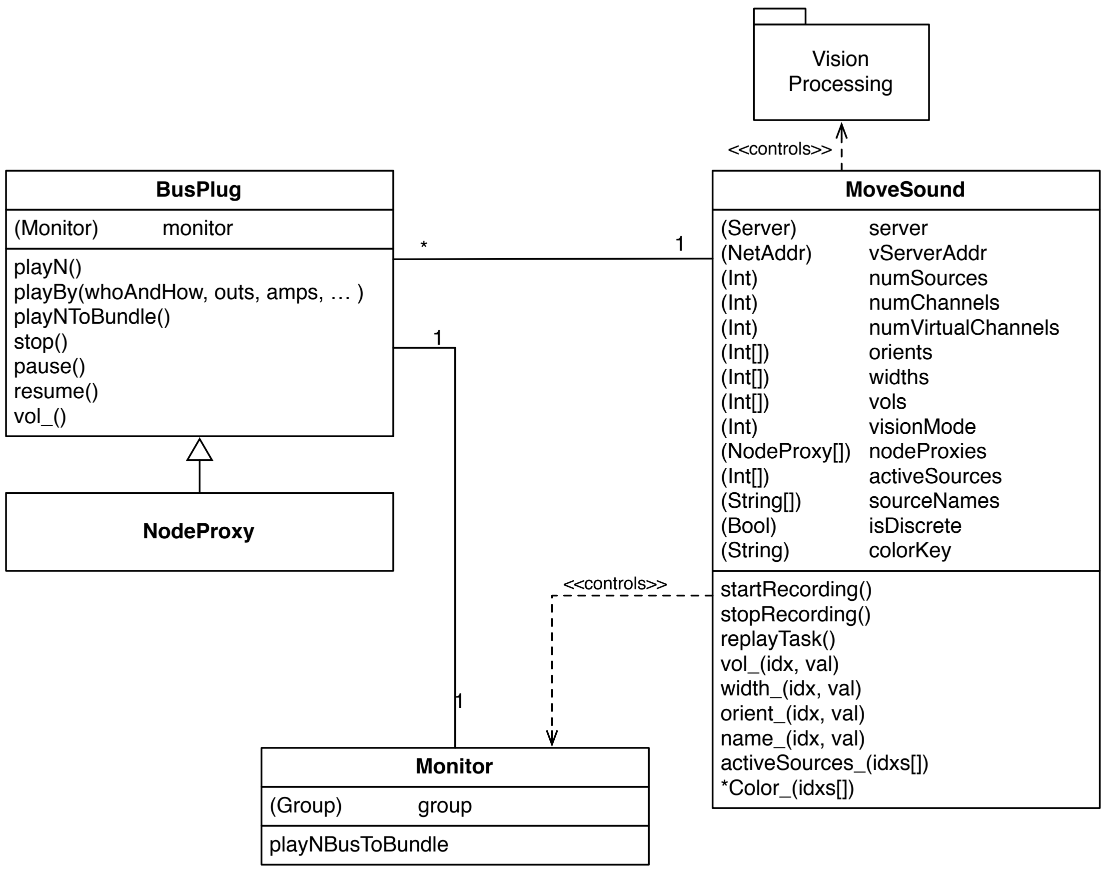


Figure 9.9.: UML diagram of the *MoveSound Model* and its connection to the *Sound Rendering*.

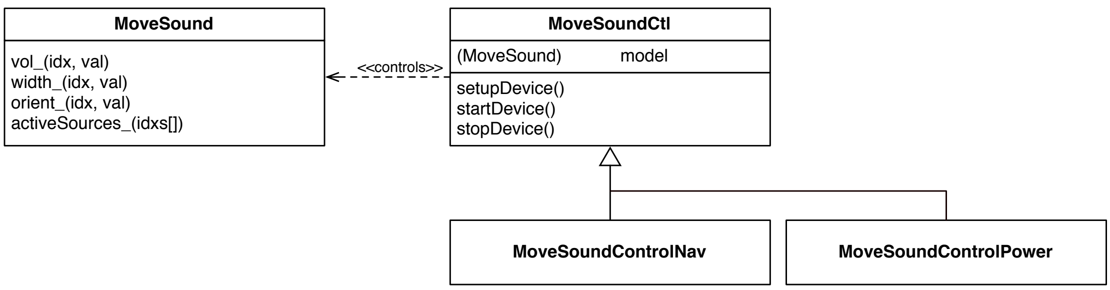


Figure 9.10.: UML diagram of the *Human Interface Control* and its relation to the model.

```
                who.name_(how.first, how.last);

                // break
                ^this;
            }, {
                warn("\"who\" is not known to playBy. Using playN instead.")
            })
        }); // fi
        this.playN(outs, amps, ins, vol, fadeTime, group, addAction)
    }
```

Its definition can be extended to use other control instances, e.g. to allow spatial control in an Ambisonics environment. For MoveSound's intended usage scenario, the adjustment of orientation and width in a loudspeaker ring, the following synthesis definition was developed and used:

```
var i = 1;
Array.geom(5, 2, 2).do{|j|
    SynthDef(("system_linkPan_audio_" ++ i ++ "_" ++ j), {
        arg out=0, in = 16, pos = 0, width = 2, vol = 1;
        arg orientation = 0.5, doneAction = 2;

        var env;
        var panner;

        env = (EnvGate( doneAction:doneAction, curve:'sin') * Lag.kr(vol, 0.05))
                    .unbubble;
        panner = PanAz.ar(
            j,
            InFeedback.ar(in, i),
            pos,
            env,
            width,
            orientation
        );
        Out.ar(out, panner);
    }, [\kr, \ir, \kr, \kr, \kr, \ir, \ir]);
}
```

Human Interface Control

We encapsulated the control mechanisms from the central model by implementing the abstract class **MoveSoundCtl** that serves as a linker between the model and the actual controllers. In order to get the actual hardware running, the three methods **setupDevice**, **startDevice** and **stopDevice** have to be implemented. We did this for the SpaceNavigator and the PowerMate input devices. A UML diagram of the controller interface dependencies is shown in Figure 9.10.

Status Graphics

In difference to the other parts, the *Status Graphics* is implemented in Processing [RF07], a simplified Java environment mainly suited for graphics programming. This module consists of three main classes: **SoundObj**s are responsible for displaying all information of Sound Sources and their spatialisation. They are collected in the **SoundObjHandler**, which also serves as the interface to the MoveSound Model via an OSC interface (cf. Table B.1). As its name suggests, the **ColorHandler** controls the colour scheme of the graphics environment.

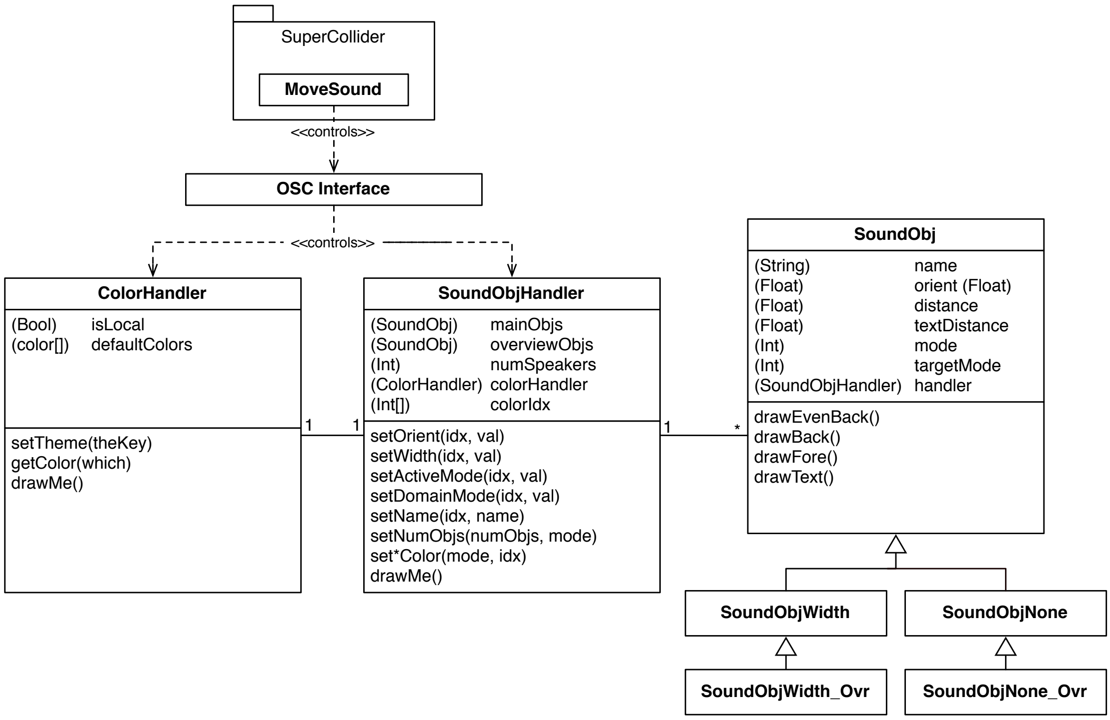


Figure 9.11.: UML diagram of *Status Graphics*.

For this, it can acquire colours from the Kuler API and append them to specific parts of the user interface.

9.1.6. A Qualitative Analysis of user Action by Means of MoveSound

We conducted a case study with five participants to gain insights into the usefulness of MoveSound as an interface to control spatial parameters of soundscapes. Its primary goal was to find out, if the minimalist TAI realised with MoveSound is sufficient for people to control spatial distribution of sound, and how they feel in operating it. Although the survey was designed to be explorative (as described in Section 4.6), we particularly searched for indicators regarding the questions:

1. Do people understand MoveSound's capabilities?
2. Do they experience any controlling limits?
3. Is there a difference in action that depends on the detail of the visual feedback system?

We did not intend to measure overall accuracy or performance; this might be a next step for such an interface design and requires a carefully selected user group operating in a dedicated scenario where accuracy is important.

For later analysis, we collected all user manipulations on the interface in data tables. We also recorded video footage of the participant's actions. The data files were processed into a visual representation (as shown exemplarily for the 4th challenge of Participant 4 in *Survey Tools*

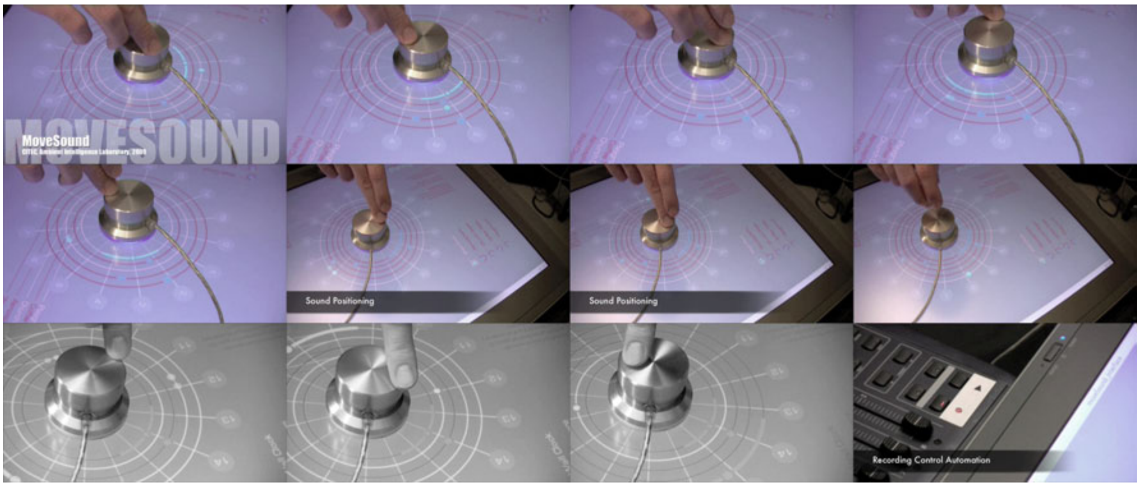


Figure 9.12.: Video stills of the MoveSound interface from the video demonstration on the DVD.

Figure 9.14).[3] Each sound source where associated with a coloured trajectory, starting from the centre. Sound source width is represented by the width of the trajectory.

Experimental Setup

The MoveSound survey consisted of six challenges divided into four stages, which each participant where asked to solve. In the first stage, the user was given the opportunity to get familiar with the interface and its capabilities (*Warm-up*), followed by two tasks that required skills in MoveSound's static usage paradigm (*1st* and *2nd Challenge*). In the third stage, the user was asked to perform a simple screenplay that was presented to him as an iconographical image (cf. Figure 9.13). In this situation, MoveSound's potential to change the position of sound sources should be used as a programmatic effect of the screenplay. We intended herby to get insights on how users co-ordinate their movements such that they are able to purposefully and dynamically control several sound sources at once. Sine the survey explicitly did not deal about time efficiency but exploration, there was no time constraint given. This resluted in a very broad spectrum in the duration of the single challenges as well as of the whole experiment: It varied between 30 and 60 minutes. In the fourth stage, the participants were asked for their ideas regarding the interface design. In this context, we offered to adapt MoveSound's interface design on the fly.

Challenges

Next, I will list the wording of single challenges and comment on them. The challenge text is reproduced in italics. The complete handout is reproduced in Appendix B.2.

Warm-up *Please adjust the position of the sound sources according to this sketch. I already adjusted the volume for you, so you do not have to care.*

This challenge primarily is primarily intended to get the participant familiar with the

[3] Visualisations for all participants can be found on the DVD. Due to data protection, the corresponding video footage is not part of the DVD. Upon request, I would be grateful to send a copy to the interested reader.

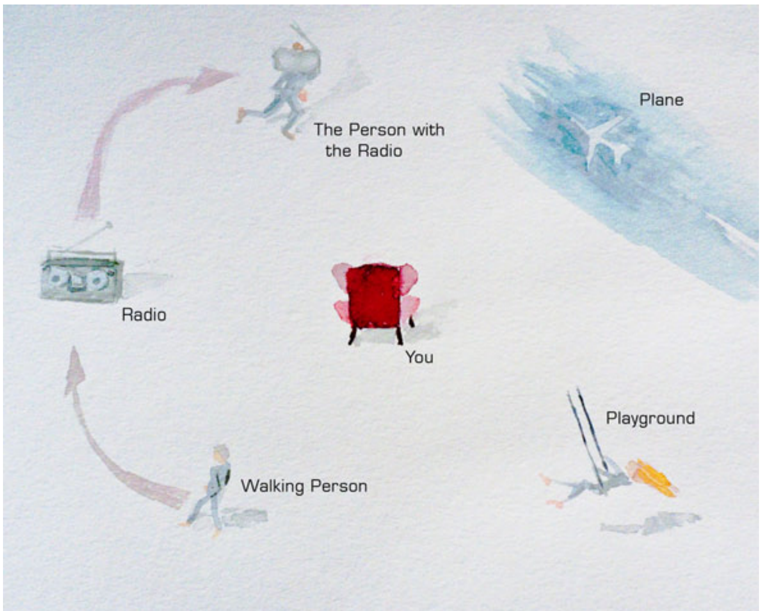


Figure 9.13.: Image of the screenplay as it was part of the MoveSound survey.

interface. While fulfilling a specific, yet simple task, i.e. placing sound sources to specific places, she is guided through MoveSound's capabilities and features.

1st Challenge *Done with the warmup? Nice! Can you think of a way how you can simulate the effect that you would hear when you turn your head? Please, try to emulate this with the MoveSound Interface and keep your head straight to the front.*

In this scenario the participant should activate all sources and turn them all at the same time. The result should sound for her, as if she would turn her head while the sources where fixed. The participant has to adapt to the situation and find a strategy to simulate a well-known situations.

2nd Challenge *Please, adjust the sound's position, width and amplitude such that you feel comfortable with it. Ask yourself, if you would like to listen to this soundscape for a longer period of time. Readjust the parameters eventually.*

Introducing a new set of sounds, the participant has to re-map sounds to positions in space. This task is less directive than the previous, since it is in the user's rating to find an appealing position of the sound sources.

3rd Challenge *Please try to emulate dynamic movements. In the sketch on the right, you see a plot of a little screenplay. With your skills in the MoveSound interface, you now should be able to perform it. Please practice a bit before proceeding to the next challenge.*

This example introduces dynamics not only for adjustment, but also for the actual performance. Control over positioning speed and co-ordination of several sources are now required.

4th Challenge *Please perform your little sound-piece and record it. After that, play it back to us. If you do not like what you have recorded, just re-press the record button and perform once more.*

Introduces the participant to control automation, and lets him record and play back his interpretation of the screenplay.

5th Challenge *Do you have any advice for me on how to change the behaviour of MoveSound, especially concerning the interaction design? Let's talk, maybe I can manage to adjust it to your needs.*

This challenge actually is no task, moreover it serves as a possibility to get ideas of the participants for the interaction design of MoveSound. We intended this as a possibility to acquire ideas for new features and controls and possibly test them for their usability with the particular individual person who raised it.

Questionnaire

1. *Gender*
2. *Age*
3. *Do You have a musical background? In which way?*
4. *Do you play video games? Which? How often?*
5. *Do you have experiences in sound engineering?*
6. *What are your personal skills and experiences with sound or audio?*
7. *Are you experienced in any craftsmanship?*

The experimental setup was completed by a small questionnaire evaluating the participant's background according to their knowledge of craftsmanship and sound manipulation.

Participants

Participant 1

Participant 1 is male and was 20 years old when attending the survey. He reported to have have a traditional (saxophone) and electronic music background and plays action-based and experimental musical video games on a regular basis. He stated that his experience in sound engineering is on a hobbyist level, while he plays the saxophone regularly at jazz concerts and attends new media and electro acoustic shows. Furthermore, he is experienced in electronics and some woodworking. This participant was presented with the *full visual feedback*.

Participant 2

Participant 2 is female, and was 28 years old when attending the survey. She reported to have a background in traditional music (piano, saxophone, vocal training, music courses, choir) and plays solitaire and ipod-games on a regular basis. She stated that she had no sound engineering experience apart from audio recording (without processing) and basic investigations into a sound editing program, however, she liked to "listen to sounds and to produce them with her own voice". For crafting experience, she told that she used to do fretwork, gardening, and often bakes and cooks. This participant was presented with the *full visual feedback*.

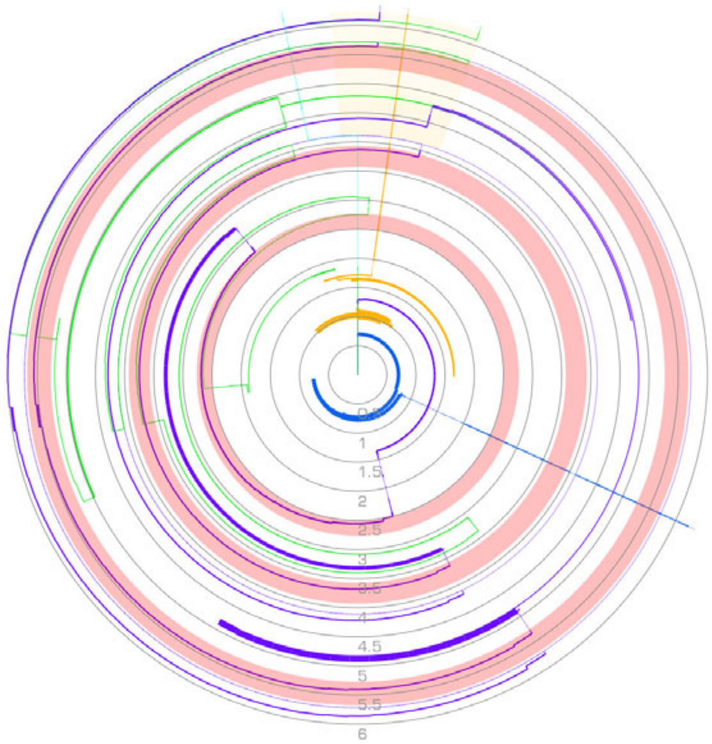


Figure 9.14.: MoveSound manipulation of Participant 4 during the 4th challenge. The blue line represents the “Playground” source, purple “Footsteps”, yellow “Airplane”, cyan “Table Soccer”, and green “Radio”. Playback of recorded material is indicated by a red overlay. For further explanation, see main text.

Participant 3

Participant 3 is male, and was 28 years old. He reported to have no musical background and rarely plays computer games (“none with spatial sound”). Furthermore, he claimed to have no experience in sound engineering apart from attending a sound synthesis lecture. This participant was presented with the *reduced visual feedback*.

Participant 4

Participant 4 is male, and was 26 years old when attending the survey. He reported to play the guitar, and is a casual player of role-play, adventure and first-person shooter computer games. Furthermore, he told us that he has basic knowledge in audio sample editors, but no crafting skills worth to report. This participant was presented with the *reduced visual feedback*.

Participant 5

Participant 5 is male, and was 26 years old. He reported that he used to attend guitar lessons for five years and used to sing songs at scout campfires. However, he stated to not have a professional musical background. Furthermore, he views himself as a casual player of adventure– and action/reflex games. He additionally reported to have no experience in sound engineering, and only basic “skills in sound”, and no crafting skills worth to report. This participant was presented with the *full visual feedback*.

Findings and Observations

During the analysis of the recorded video material, we identified four categories of insights: *acceptance*, *recording*, *display type*, and *bimanual operation*. In general, we found that the users were able to fulfil the given challenges without any exceptional problems.

Acceptance

All participants reported that they were able to solve the presented challenges to their satisfaction. None of them found exceptional weaknesses of the central interface design, i.e. the way how the sounds can be placed in the loudspeaker ring and their width can be adjusted. However, many of them complained that the source activation buttons are located at the wrong place; they proposed to integrate them into the visualisation. We therefore plan to replace the complete volume mixing section by a multi-touch setup where the source activation and amplitude control can be achieved directly on the surface.

Recording

Particularly the recording of soundscape dynamics has been understood exceptionally well, though participants had differing demands to fulfil the challenge. So, e.g. Participant 1 tried several times to achieve a control automation, Participant 5 fulfilled the task by recording a very basic automation.

Full vs. reduced display

I observed a clear difference in the participants' behaviour depending on the type of visual display they were confronted with. While the full display, mediating information on sound position and width, attracted the user's gaze completely to the control interface, the participants with the reduced visual display often gazed around and were not visually focusing on the manipulated tool. As a typically routine, they altered between volume control and position adjustment. They also did not use the capabilities of MoveSound to change the sounds' width. Afterwards, both Participant 3 and Participant 4 reported that they missed the visual correspondence. Participant 4 suggested that an adjustment by hearing only is difficult, and different volume settings for the sound sources decrease his sound separation capabilities such that he was often not aware of which sound he was controlling. He nevertheless proposed a possible usage scenario – the production of soundscapes where the exact position of sounds is not of particular interest; like the creation of jungle– or playground soundscapes from several sources – that he considered joyful and highly creative.

Bimanual operation

Most of the participants only used their preferred hand for MoveSound's operation. Only Participant 1 made extensive use of both hands in the 3rd and 4th challenge, in which he simultaneously adjusted volume and spatial parameters of sound sources.

Answers to survey questions

With these observations, we can answer the questions asked in the beginning of this section: People do understand MoveSound's capabilities, as far as the challenges of this case study are concerned. In most cases, they experienced controlling limits only regarding the sound source selection, a problem we want to address in future extensions of the MoveSound interface, Although all participants were able to fulfil the challenges to their satisfaction, we found differences in their action depending on the visual feedback system. While the full display exclusively attracted the user's gaze, the reduced visual display made the participants' gaze around. We believe that this turns them more to be present in the soundscape itself, then focusing on the model provided by the MoveSound interface.

9.1.7. Conclusion

We introduced MoveSound, a TAI that was designed to control spatial parameters of sound rendering in a loudspeaker ring. Movesound's Tangible Interface offers high accuracy controls that enable the user to change the rendering position of selected sound sources as well as their width. For the system's central controller, two different hardware interfaces were integrated that feature either a relative (PowerMate), or velocity-based control (SpaceNavigator). The minimalist Tangible Interface is complemented by a graphical display that mediates information on its state by back-projection onto the canvas. With its two modes, the visual display supports either a qualitative control that invites its users to playfully arrange sounds in their environment (reduced mode), or an explicit control in which the users are able to visually track the actual positions and widths of the sound sources (full mode). MoveSound's functionality relies on this visual display since it does not feature a completely physical state display. This tradeoff, though, introduces the flexibility to map sound source position and width with arbitrary precision to the controlling interface and to arbitrarily select which sound sources should be controlled in parallel.

Lessons learned

In the case study described in Section 9.1.6, we confronted participants with MoveSound. With this study, we found out that it took little effort for the participants to learn the capabilities and features of MoveSound. Not only audio professionals but also novices with little experience in sound control were able to use the system almost immediately. Another noteworthy finding is that MoveSound was used very different depending on the active graphical representation. The participants confronted with the reduced graphical interface were focusing more on the effect caused by their manipulation to the acoustic environment than on the visual display; an observation that contrasts with the behaviour of participants using MoveSound with the full visual display: They focused more on the effect their manipulation had to the visual model. The discussion with the participants yields that they had varying demands to the system: While the full display were considered to be used in situations that demand precise control, the potential of the reduced version that lacks a visual representation of sound position and width were considered by the participants for the intuitive and creative control of sound spatialisation.

Outlook

An interesting extension for MoveSound is the integration of multichannel sound sources, a feature that was not implemented into the current version due to technical reasons (see Section 9.1.5). This would add many features and spatial effects like echo placements or stereo– and multichannel playback of pre-recorded sources.

The addition of multi-touch capabilities for sound source selection and volume adjustment as already mentioned in the previous section is also considered a natural addition. It would substitute the current (rather clumsy) MIDI fader box by controls that are directly integrated into the graphical display.

9.2. ChopStix

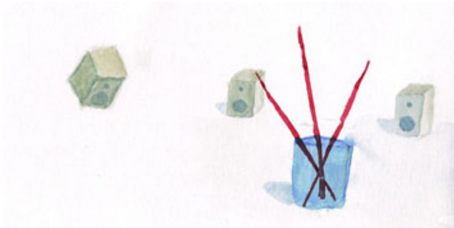


Figure 9.15.: Design concept of ChopStix.

ChopStix is a Tangible Auditory Interface that is designed to display and control spatially distributed data. It reclaims the abstract of digitally processed information and provides an alternative access to environmental data. Furthermore, it is a calm system dedicated to reflection by interacting with an Auditory Display by everyday-like artefacts. Its users are immersed into weather data as measured throughout Europe in near real time. The raw measurements provided by independent sources are stirred together by the user, forming an informative auditory stream. Its TI allows users to express their idiosyncratic approach in understanding this data mix. The resulting configuration of the used Stix is a trace of each user's personal preference.

By design, the ChopStix system is especially suited for near realtime streams of e.g. ozone measurements, weather-related data or traffic information. Its primary intent is to provide a human-computer interface to easily access and select parts of such streams for monitoring purposes. This is realised by the *ChopStix Auditory Display* (CAD), a spatial soundscape that reflects data streams auditorily in realtime, and the *ChopStix Tangible Interface* (CTI), a TI designed to control spatial aspects of these soundscapes. Likewise MoveSound (see Section 9.1), ChopStix is a loudspeaker centred design: It allows users to point small sticks (*Stix*) to directions of interest in the surrounding soundscape. However, in difference to MoveSound, the monitored data domain – near realtime data streams – provides update information on a regular basis. This implies a different approach to the control design, since the Auditory Display is tentatively used in concurrency to a primary task. To establish such a long-term usage, where users only sporadically make modifications to the soundscape, the CTI reflects its current state completely via physical objects and their relation to each other. This simplifies to keep track of the system's current state not only for the main user, but also for other people coming by. All of them are able to immediately interpret the current auditory configuration by visually observing the interface's current state.

Due to ChopStix reactive nature that does not include any complex internal controlling, it is easy to explain people, how a soundscape can be controlled with the CTI. The involved pressure- and magnetic field sensors directly control the amplitude of loudspeakers arranged in a ring; their location hereby determine their influence. This direct correspondence together with the system's immediate reaction completely explains the system's functionality.[4]

9.2.1. A User Story

ChopStix was designed with a clear view of it's possible usage in mind: As stated before, its primary intend was to (a) surround people with artificially rendered data-driven spatial soundscapes that mediate information on near real time data streams such as weather or

[4]Later in this section, we describe and discuss another software design approach, which incorporates a more abstract level of the interpretation of the tangible state and discuss the benefits and drawbacks of both.

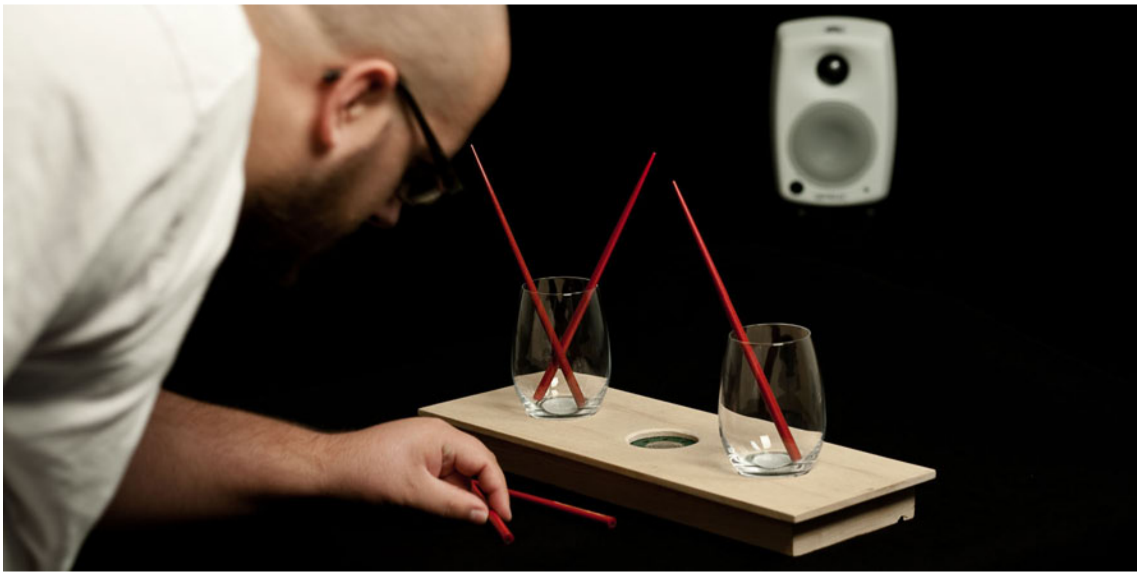


Figure 9.16.: *ChopStix Tangible Controller* is a plate with three sinks that are made to okace glasses. Each of the sinks are identified with one soundscape. Placing a glass activates the soundscape's playback with a spatial emphasis determined by *Stix*.

traffic information, and (*b*) to help people adjust the spatial distribution of the multichannel soundscapes according to their needs and preferences. The system's Tangible Interface therefore should serve the spatial parameters of predefined soundscapes. This especially means that not only the main user, but also other people, which occasionally drop by, should be able to easily and immediately recognise the system's current state, and possibly change it. This adjustment should be simple and located at the same place where the system's state is displayed. Obviously, simplicity of control plays an important role for the ease of use of such an interface. This explicitly does not mean that the system is simple in its design or based on simple algorithms, moreover its *usage* should be simple. To ensure a complete embedding into its environment, all possible degrees of manipulation should have a meaningful and appropriate effect on the system's state. Taking care of all these prerequisites led us to the following imaginary user experience:

User Story

A user enters a room, approaching the CTI. It is a rectangular-shaped plate surrounded by four glasses and several Stix. The plate is empty, rendering the artificial soundscape silent. On it, the plate has three sinks, each labelled with the name of a particular data-driven soundscape. They can be used to place a glass, possibly containing Stix*, which determine the amplitude distribution of the linked soundscape. Aware of this, the user places a glass on the sink labelled "Weather Condition". A calm sound fills the space around the person, equally distributed in the loudspeaker setup. To get information on only a specific direction, the user places a stick in the glass pointing to his hometown. The sound's ambience slowly changes, such that it is prominently heard from the direction the stick is pointing to. Data sources outside that scope are rendered less prominent. After a while, the user switches to a different soundscape – "Ozone Measurements" – by relocating the glass and its content to the sink labelled accordingly. He adds a second stick to the glass, which reshapes the ozone*

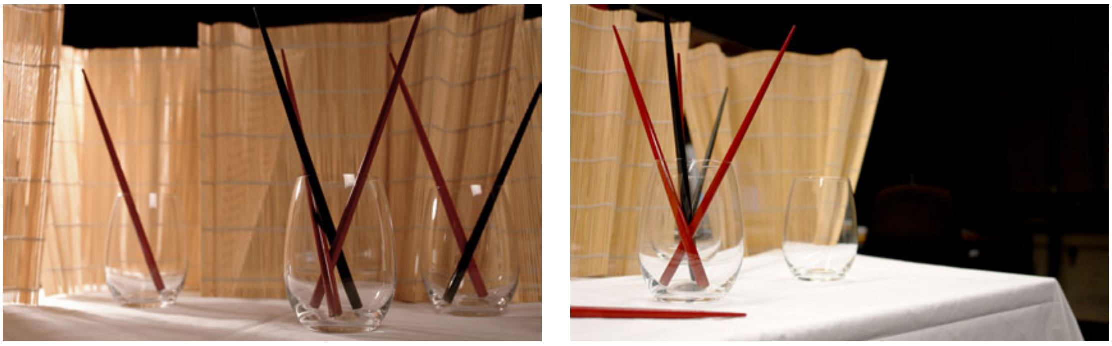


Figure 9.17.: The resulting design study of a ChopStix Interface mock-up session.

soundscape's amplitude distribution. By rotating the glass, the user coincidentally changes the directions of all contained sticks, while the soundscape's amplitude distribution follows the movement.

9.2.2. Data Domain

ChopStix is intended to serve as a monitoring system for spatially distributed near realtime data (NRT). The update rate of such data is typically between every minute and once per hour. The design goal to auditorily surround the user by the acquired data requires that its spatial configuration is algorithmically accessible. Queries for data items in a given area (e.g. Europe) should be possible. We were able to acquire and use the weather data stream provided by *Weather Underground*. According to their website, *wunderground.com*

> has developed the world's largest network of personal weather stations (almost 10,000 stations in the US and over 3,000 across the rest of the world) that provides our site's users with the most localized weather conditions available.

The weather data is provided via html files that are updated at least once per hour. We process this data and use it as the data stream in the ChopStix Auditory Display.

9.2.3. ChopStix Tangible Interface

Development Context The ChopStix Tangible Interface (*CTI*) was designed to offer long-term control of ambient spatial displays as they may be part of future living rooms. Since pointing is a widely known gesture to mediate attention to a specific location, be it by hand or by incorporating artefacts like signs or arrows, we decided to use Sticks (as described in Section 5.5) as controlling artefacts for the spatial aspects of the soundscape. *Enviroment* The audio-centric intent aims for a setup of CTI near the sweet spot of the multichannel loudspeaker setup, allowing the user to immediately observe the spatial dynamics while adjusting the Auditory Displays. However, the long-term aspect in the control data – near real time data streams change their values only a few times an hour at maximum – requires the Tangible Interface to be constantly available without disturbing by its prominence and placement. We therefore decided on a compromise and placed it near the edge of the used multi-speaker setup (as shown in Figure 9.19), where it may be surrounded by lounge chairs and can offer its functionality without being the centre of the room.

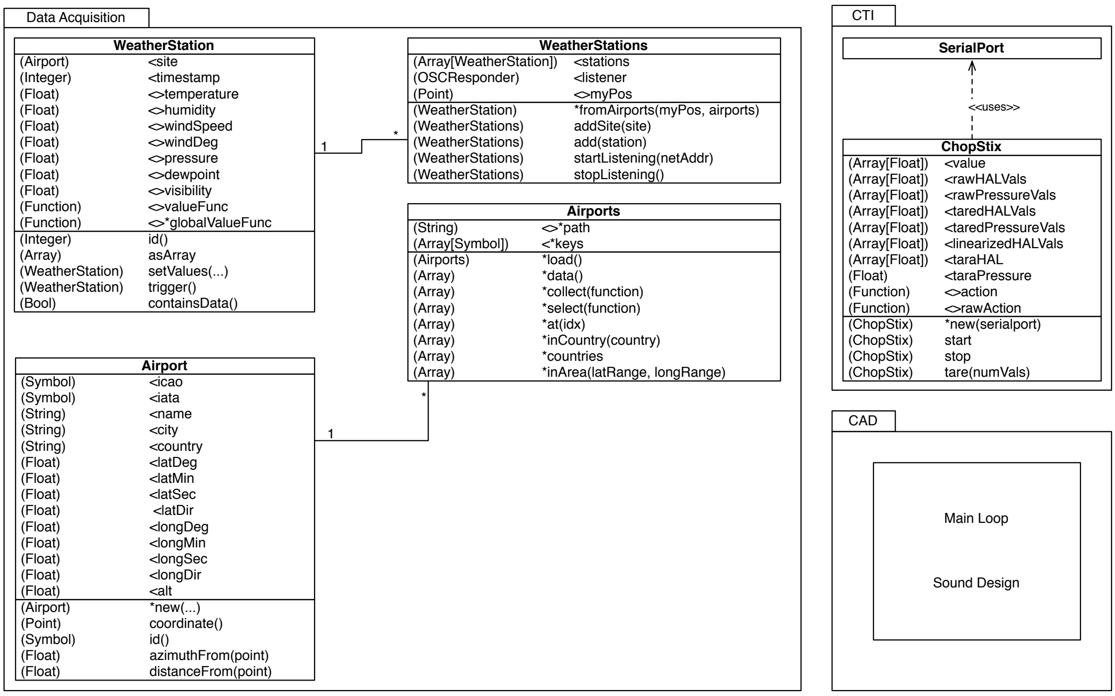


Figure 9.18.: UML diagram of ChopStix-relevant classes and their dependancies.

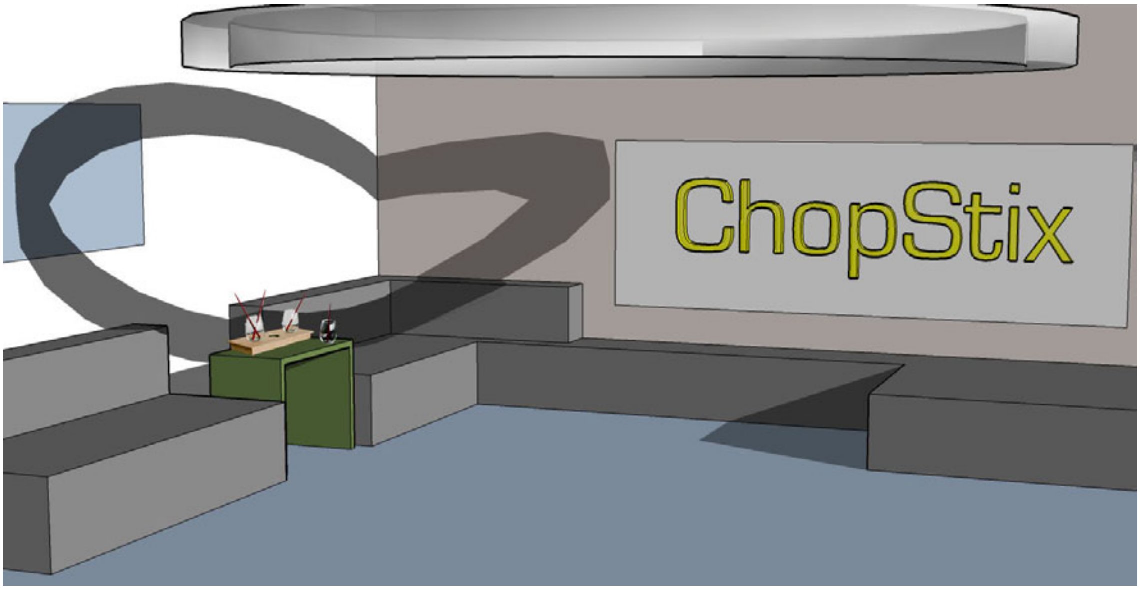


Figure 9.19.: A rendering of the location of the ChopStix Interface in a room. The spatial sound display is realised by the ring of loudspeakers on the ceiling. The long-term aspect in the control – near real time data streams change their values on an hourly basis – requires the interface to be constantly available, but not to be disturbing. Therefore it is placed near the edge of the used multi-speaker setup.

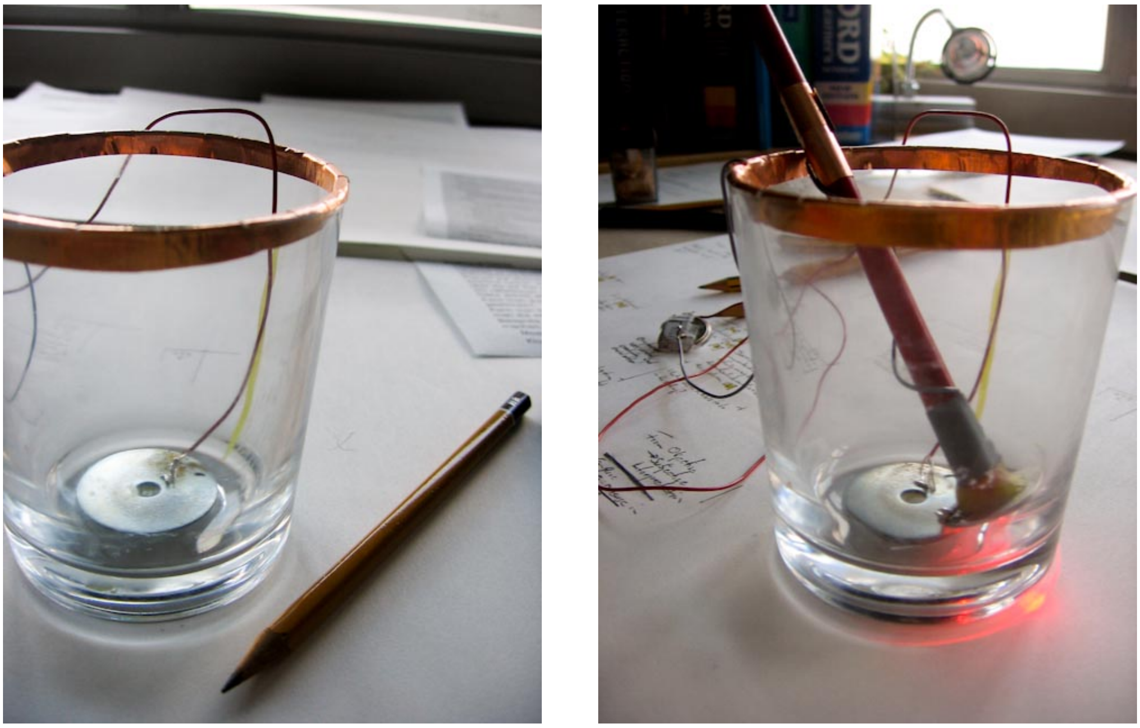


Figure 9.20.: Computer vision-based design of CTI.

Interface

CTI's hardware design draws on several mock-up tests, in which the aesthetic as well as the usability of the interface where tested. One of the test session results is shown in Figure 9.17. The correspondence of spatial parameters of a soundscape and the direction in which the used Stix point to requires automated sensing of the sticks position and orientation. Fortunately the feature space in which the recognition takes place is spatially limited to one of three *sinks* and physically constraint by a glass in which the Stix are placed: When left alone, these Stix tend to rest in a local energetic minimum, i.e. in this case that their bottom part is located on the opposite part of the glass' bottom as their pointing direction (for an example, see the configuration of the Stix in Figure 9.21). This means that sensing the position of the bottom end of a stick in a glass is sufficient (on a long-term basis) to pretend their pointing direction. To measure that position, we tested two different approaches; one using LEDs as light sources attached to each stick and a camera viewing from below the glass (Figure 9.20), and one with sensors for magnetic forces evoked by magnets and assembled into the Stix (Figure 9.21).

Design concepts

With the ChopStix setup, the difference between direct interaction design and abstraction-based interaction design for Tangible Auditory Interfaces can be exemplified. Both approaches serve specific benefits and drawbacks:

Abstraction-based interaction design

When each stick of a CTI corresponds to a unique sound source, we can speak of an abstraction-based interaction design. Each sink then could control one display type: sonic, visual or multi-modal, each highlighting a different aspect of the processed data. Placing a particular stick in a glass that stays on a sink would cause the associated data to be rendered in the display type attached to the sink. The stick's assumed position (extracted from the sensor data) would then be used to compute the display's amplitude or brightness

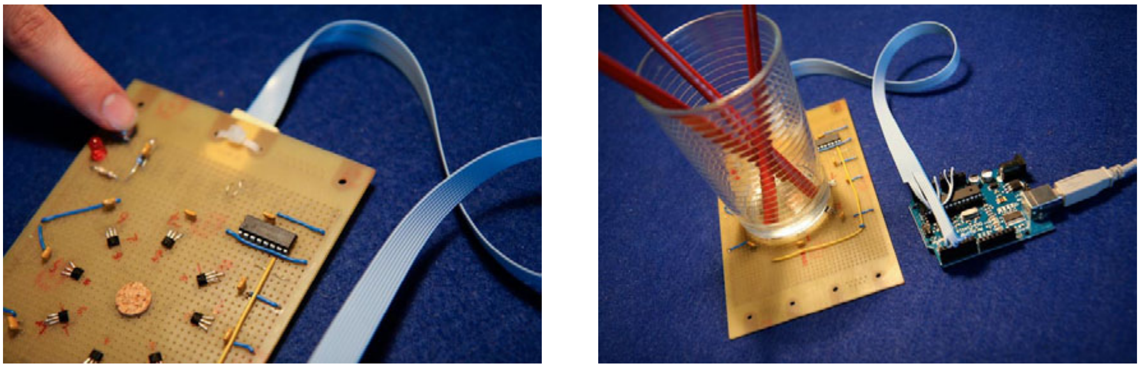


Figure 9.21.: First prototype of the Hall-effect-based design of CTI.

distribution regarding the represented data. Technically, such a system can be realised either by assembling different magnetic values into each stick that are sensed by Hall-effect sensors, or with coloured LEDs.

Direct interaction design

In a direct interaction design approach after the TAI key feature of Tight Coupling (see Section 7.1), the sinks represent sound sources. Placing a glass on such a sink influences the soundscape's playback; its overall amplitude is directly linked to the measured weight of the glass. Additionally placed Stix in the glass determine prominent areas in the soundscape; an amplitude distribution that reflects the sticks position is realised by directly coupling the sensor values of eight Hall-effect sensors to the amplitude of eight (possibly virtual) loudspeakers. Overall, the Stix' location and the overall weight of the glass determine the linked soundscape's overall amplitude. The used hardware by its static physical representation of the soundscape in combination with the static mapping of soundscapes and data to fixed places on the interface support an easy interpretation of the system's current state. The resulting minimalist interface can be seen as a representative for slow technology, a framework that allows people to leave their *information footprints* when walking by [HR01].

Actual implementation

After these preliminary design considerations and technical tests, we decided to use the direct interaction design approach, incorporating four drinking glasses, eight Stix prepared with the same load of magnets, and a surface with three sinks. The direct design was chosen due to technical feasibility and because of it's more simple user interface design. We believe that it is far better to understand for users than the abstraction-based design approach.

Pressure

Each of the three sinks on the surface has several sensors: By *Weiss-Foam* attached to the glass' bottoms, the overall weight throughout a sink can be measured. The deformable foam changes its resistance according to pressure and assembles – together with a voltage divider and ground and vcc plates integrated into the sinks – a reliable pressure sensor [KWW03].

Stix distribution

It is used to determine if something is placed on the sink, and how heavy it is. Sensing the presence and orientation of Stix in glasses on a sink was done with eight hall-effect sensors, equidistantly arranged in a circle below the glass. They sample the local magnetic field intensity and detect the magnetic forces originating in the magnets that are integrated into the Stix. All sensors are read into the central processing unit, either directly (as done for the pressure sensors) or via an eight channel analogue multiplexing IC (for the Hall-effect sensors).

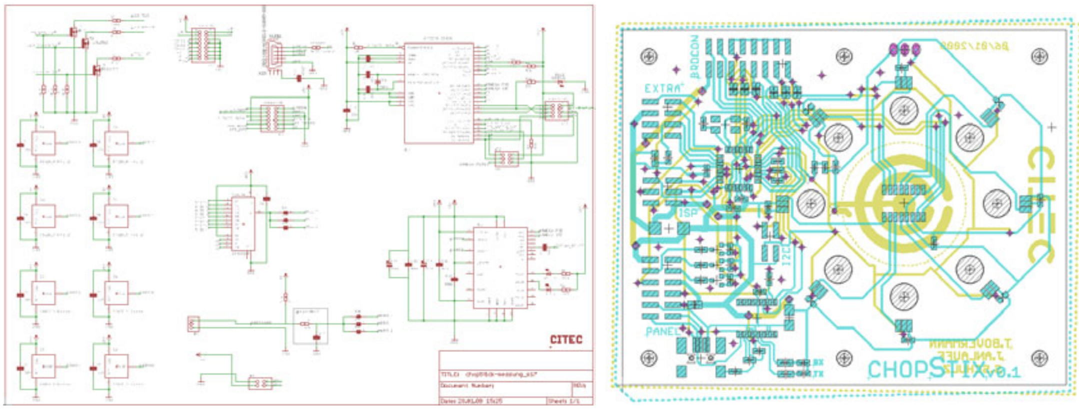


Figure 9.22.: Circuit diagram (left) and board layout (right) of the Hall-effect sensor based implementation of *CTI*.

For each sink, we designed and assembled one PCB board; two slaves with only sensors and multiplex parts, and one master that is additionally equipped with an Atmel ATMEGA168 CPU for sensor reading, basic processing, and communication with the host computer. Its serial interface sends the acquired data via an USB connection provided by an FTDI Serial2USB chip to a connected computer.[5] The circuit diagram and the board layout of the CTI are shown in Figure 9.22.

Software

On the host side, CTI's sensor values are captured by a standard serial connection based on ASCII coding. We chose SuperCollider [McC02] [WCC09] as the computer language for sound rendering and controlling. Utilising its native support for serial port interfacing, the incoming data stream was acquired from the CTI. The central class **ChopStix** with the interface as shown in the UML diagram in Figure 9.18 implements the serial port handling to be transparent for the user. It is only necessary for him to provide a valid **SerialPort** instance to which the used CTI is connected. As noted in the help file, **ChopStix** is a threaded controller that splits data acquisition from the corresponding hardware and the resulting action. This allows for a smoother integration of the system into a bigger scope and prevents it to actively wait for a CTI that is not properly responding.

Hall-Effect sensor linearisation

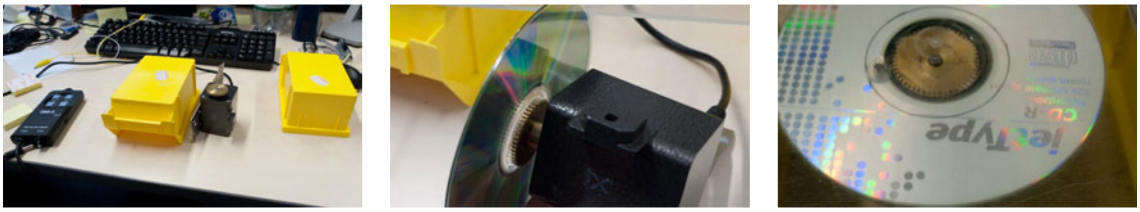
Since the used Hall-effect sensors are highly non-linear according to magnet positions, we calibrated and measured the ChopStix interface with the help of a magnet attached to a step-motor (see Figure 9.23). By help of a least square curve fitting algorithm we determined parameters for the mapping

$$\hat{x} = 1 - \sqrt{v \log \frac{a}{f(x) - b}} \tag{9.1}$$

We chose it because it had sigmoid qualities and reasonable parameterisations that fit the given optimisation problem.

The parameters acquired from applying the measured values to the optimisation allowed us to linearise the values of the Hall-effect sensors and feed them into the amplitude computation process needed for the spatial audio setup. The resulting curve is implemented in **ChopStix:pr_linearizeFunc** as

[5] The appropriate driver has to be installed to access it. You can get it at `http://www.ftdichip.com/FTDrivers.htm`.



(a) Overview of the measuring installation. The step motor for controlled movement of the magnet and two yellow boxes on which the CTI is placed during measurement.

(b) A CD attached to the step motor served as a fixation for the measuring magnet.

(c) Magnet connected to the CD.

Figure 9.23.: Setup for Hall-effect sensor data acquisition. The setup was used to calibrate the Hall-effect sensors.

```
+ ChopStix {
    pr_linearizeFunc {|y, v = 0.3348, a = 93, b= -5|

        y = y ? 0;
        y = y.clip(b, b+a);
        ^(((sqrt(log((y-b)/a).neg*v)).neg) + 1).clip(0, 1);
    }
}
```

Usage

The typical procedure to instantiate a ChopStix Tangible Interface is described next: First, the serial port has to be defined:

```
SerialPort.devicePattern = "/dev/tty.usbserial-*";
SerialPort.devices.first;    // look if there is a valid device

p = SerialPort(
    SerialPort.devices.first,
    baudrate: ChopStix.baudRate,
    crtscts: false
);
```

Then, a ChopStix instance listening to that port has to be created:

```
c = ChopStix(p); // create a new ChopStix instance listening to Serialport p
c.start;    // start data acquisition
```

To tare the sensors, the user has to make sure that nothing is placed on the CTI while it acquires data:

```
c.tare(50); // tare sensors. This takes a moment.
```

Finally, an action has to be defined that is evaluated on every update step. In this example, we simply print the measured values:

```
c.action = {|hVals, pressVals|
    "HALL:".postln;
```

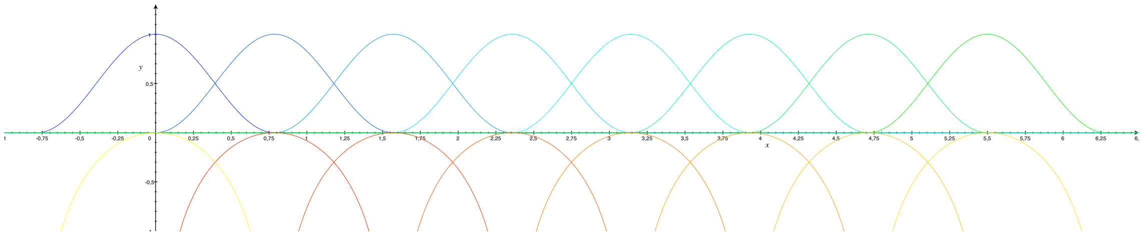


Figure 9.24.: Equal power panning as it is used in ChopStix. The x-axis represents the sound position in the ring. The green to blue curves are the normalised amplitudes of the loudspeakers, whereas the red to yellow curves represent the gain for each channel in dB.

```
    hVals.do{|pod|
        pod.do{|val| "%\t".postf((val*100).round)};
        "".postln;
        (pod.sum).postln;
    };
    "Pressure:".postln;
    pressVals.printAll
}
```

To stop data acquisition and close the serial port, the interface has to be stopped and the serial port has to be closed.

```
c.stop; p.close;
```

9.2.4. ChopStix Auditory Display

Hardware

Besides the CTI, there is also the ChopStix Auditory Display (*CAD*). Its intent is to present streams of spatially distributed data auditorily to the user. The used sound setup is – as in *AudioDB* and *Durcheinander* – a loudspeaker ring surrounding the CTI and its users. The system that was designed initially for a ring of eight loudspeakers reflects the CTI's hardware design with the eight magnetic hall sensors. By addition of virtual sound sources, it can be adapted easily to a different number of loudspeakers.

Software

The amplitude of each loudspeaker was determined according to the linearised Hall-effect sensor readings. The following formula for equal power panning additionally includes a width compensation depending on the number of loudspeakers.

$$\operatorname{amp}(x, p_i) = \begin{cases} \cos^2\left(\frac{n(2\pi x - p_i)}{4}\right) & \text{for } \|n(2\pi x - p_i)\| < 2\pi \\ 0 & \text{otherwise.} \end{cases} \tag{9.2}$$

with

$$\begin{aligned} n &\in \mathbb{N} && \text{number of loudspeakers, in this case 8,} \\ p_i &\in \{0, 2\pi\} && \text{position of loudspeaker } i, \\ x &\in \{0, 1\} && \text{linearised measurement for } i\text{-th Hall-effect sensor.} \end{aligned}$$

As shown in the corresponding plot in Figure 9.24, the overall gain [dB]

$$\mathrm{gain}(x_0, \ldots, x_{(n-1)}) = 10 \log_{10} \left(\sum_{i=0}^{n-1} \mathrm{amp}\left(x_i, p_i\right) \right) \tag{9.3}$$

with

$$p_i = \frac{2i\pi}{n}$$

equals the amplitude panning approach for an audio ring of n speakers [Pul01].

Basic setup

A basic setup that uses both CIT and CAD is shown in the following listing. It instantiates three soundscapes and stores them in an array.

```
Ndef(\leftSoundscape, {|amps = #[1, 1, 1, 1, 1, 1, 1, 1]|
    (LFSaw.ar(BrownNoise.ar.range(50, 100))*2).softclip!8 * amps
});
Ndef(\centerSoundscape, {|amps = #[1, 1, 1, 1, 1, 1, 1, 1]|
    (LFSaw.ar(BrownNoise.ar.range(100, 200))*2).softclip!8 * amps
});
Ndef(\rightSoundscape, {|amps = #[1, 1, 1, 1, 1, 1, 1, 1]|
    (LFSaw.ar(BrownNoise.ar.range(200, 400))*2).softclip!8 * amps
});

q.soundscapes = [
    Ndef(\leftSoundscape),
    Ndef(\centerSoundscape),
    Ndef(\rightSoundscape)
];

q.soundscapes.do(_.play);
```

The soundscapes' amplitude distribution then can be defined according to the measured Hall-effect values by

```
var tmpPressVals;
v = ControlSpec(2020, 0);
c.action = {|hVals, pressVals|
    tmpPressVals = v.unmap(pressVals);
    q.soundscapes.do{|scape, i|
        scape.group.setn(
            \amps, hVals[i].collect{|v|
                (0.1 + sin(v * 0.5pi)) * tmpPressVals[i]
})}}
```

To stop the playback of all soundscapes,

```
q.soundscapes.do(_.stop);
```

has to be evaluated.

Auditory Display

For the ChopStix system, we developed a near-realtime Auditory Display for the weather data described in Section 9.2.2. For each weather station, an audio stream was created, mirroring the provided measurements that were coming from the direction the station is located (according to the CTI): Its creation is linked to the update of **WeatherStations** as follows:

```
WeatherStation.globalValueFunc = {|me|
    me.containsData.if({
        "create Pdef for %\n".postf(me.id);
        Pdef(me.id, Pbind(
            \instrument, \chopTix,
            \freq, Pfunc{me.temperature.linlin(-10, 50, 200, 10000)},
            \dtEcho, 0.1,
            \azimuth, Pfunc{
                me.site.azimuthFrom(q.myPos)/pi;
            },
            \driftFac, Pfunc{me.windSpeed.linlin(0, 50, 0, 0.4)
                                * (me.windDeg-180).sign},
            \sustain, Pfunc{me.humidity.linlin(20, 100, 1, 8)},
            \durationBase, Pfunc{me.humidity.linlin(20, 100, 3, 9.9)},
            \dur, Pwhite(Pkey(\durationBase)*0.5, Pkey(\durationBase)*1.5),
            \rq, Pfunc{me.pressure.linlin(900, 20048, 2, 0.1)},
            \amp, Pfunc{me.site.distanceFrom(q.myPos).reciprocal},
            \out, q.outBusWeather.index,
            \group, q.synthGroup,
            \addAction, \addToHead
        )).play
    }, {
        "remove Pdef for %\n".postf(me.id);
        Pdef(me.id).stop
    })
};
```

This mapping is used to control the following **SynthDef**:

```
SynthDef(\chopTix, {
    arg out = 0;
    arg dtEcho = 0.1, freq = 14000,
        azimuth = 0, driftFac = 0.01,
        sustain = 0.7, rq = 1, amp = 0.1;

    var son, echoed, echoEnv, azimuthEnv, spatialized;
    son = Impulse.ar(0, mul: 0.25) + (SinOsc.ar(freq*0.5)
            * Line.ar(0.3, 0, 0.001));
    echoEnv = EnvGen.ar(
        Env([dtEcho, max(0.0001, dtEcho * 0.001), dtEcho],
            [sustain, 0], -3),
        doneAction: 2
    );
    azimuthEnv = EnvGen.ar(
        Env([azimuth, azimuth + (driftFac), azimuth],
            [sustain, 0.0001], \lin)
    ).wrap(0, 2);

    echoed = RHPF.ar(CombC.ar(son, 0.1, echoEnv, sustain), freq, rq);
    spatialized = PanAz.ar(q.numChans, (echoed.tanh) * amp , azimuthEnv);
    OffsetOut.ar(q.outBusWeather, spatialized)
}).store;
```

According to the implementation, high temperatures measured at the weather stations are reflected by high frequencies of the resonating filters, whereas the damping of the resonator and the duration of each event depends on the measured humidity. A dryer weather therefore corresponds to a dry sound.

Since the weather-related soundscape cannot be demonstrated easily because it changes only slowly with an update rate of once per hour, we also implemented a dummy soundscape that changes its characteristic more often. Both the weather display and the artificial soundscape are demonstrated in a video on the DVD.

9.2.5. Conclusion

Summary

With ChopStix, we introduced a stick-based TAI that is specialised in the long-term control of spatial Auditory Displays. It enables users to control the amplitude distribution of soundscapes on a long-term basis. These soundscapes are typically based on near-realtime acquired data. With ChopStix, users can mediate their current interest in certain regions. CTI, the ChopStix Tangible Interface, is an implementation of the Sticks scheme (as described in Section 5.5.3). Its direct manipulation-to-action mapping exemplifies one particular strength of TAIs; the power of a seamless interplay between a Tangible Interface and an Auditory Display.

Lessons learned

Due to the simple manipulation-to-action mappings, ChopStix' Tangible Interface (CTI) is easy to understand. Therefore, its technical explanation on the level of *what sensor is responsible for which functionality* completely covers its explanation on a functional level. In this context, a teacher can tell a new user that *the magnetic sensors inside the sink measure the current magnetic forces applied by the Stix' build-in magnets. The amplitude of each loudspeaker is directly controlled by the measuring of the corresponding sensor.* This easy explanation is sufficient to understand how the interface works. Since no internal model is needed for the controller, it does not do something unexpected that would be considered as wrong.

All user controls are represented by physical artefacts. They serve as both controller and implicit display of the system's current state. This makes CTI an *absolute* interface. All people that operated with the system understood this fact and used it that way.

During demonstrations, we experienced that the Stix' position can be interpreted in two different ways: Either their configuration is observed to point outwards, towards the loudspeakers, or inwards towards the sensors. These two interpretations need a different handling of the Stix to loudspeaker mapping (see Figure 9.16) We decided in favour to the sensor-oriented approach since it simplified the system's explanation to be closely related to the interface: *Each loudspeaker is connected to a magnetic sensor.*

As stated before, ChopStix is designed as a *tool* for spatial control of Auditory Displays. The design implicates certain use cases, however, it essentially provides a way to physically manipulate an artificially rendered soundscape's appearance. In our demonstrations, people quickly learned to press the glasses into the moulds in order to get the connected soundscape amplified; with this action, they expressed their subjective importance of that particular soundscape; ChopStix became ready-to-hand.

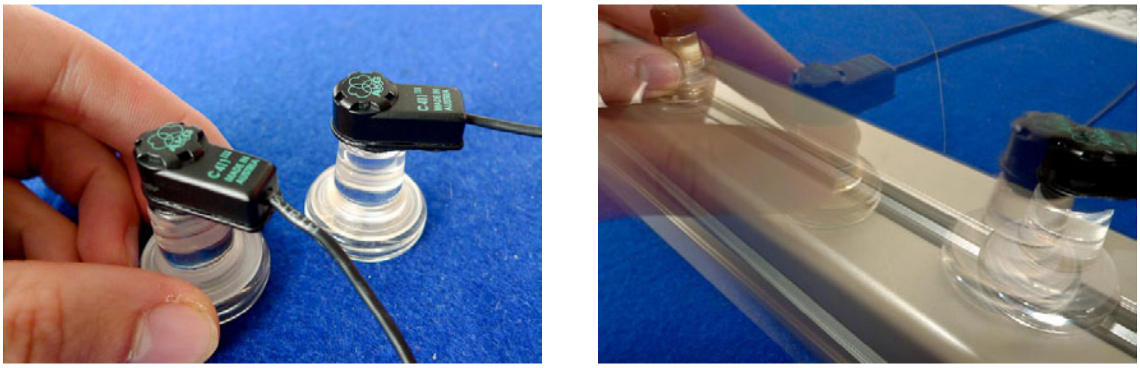


Figure 9.25.: Schüttelreim: One of several possible Reim setups.

9.3. Reim

The Reim toolset was developed in co-operation with René Tünnermann.

As stated in Chapter 2, digitally stored information has no human-perceivable phenotype by nature. Data is immaterial, since its formal information content does not change depending on its actual representation. There is for example absolutely no difference in a digital recording of Strawinsky's *Sacre du Printemps* whether it is represented as a series of magnetic forces on a rotating plate (i.e. a hard-drive), as states of electronic NAND-gates on computer chips (i.e. RAM), or as a series of high- and low-voltages in a copper-cable. On a perceptional level, however, where human senses and the interpretation of data is important, the effective representation does play an important role. It does make a fundamental difference to us and to the way we interpret content whether the recording is shown as coloured pixels on a surface, or played back by loudspeakers: We derive our understanding of data from its current representation. This circumstance makes it essential to look at representation processes and their influence on the human perception and interpretation when dealing with data exploration.

One of the human's natural qualification is his ability to easily *get a grip* of almost every physical object lying around. Technically speaking, a human is able to understand the basic features and possibly the inner structure of an object by exploring it with his sensors (ears, nose, skin, eyes) and actuators (arms, hands, legs, fingers, tongue, etc.). We propose that dealing with everyday data like temperature, humidity, and wind speed, or more technology-oriented measurements like CPU load and temperature should be as easy as discovering the current fill-level of a box with sweets. Taking this attempt literally motivates a more direct representation of data than it is state of the art: the augmentation of action feedback on everyday objects with appropriate data representations. *Reim*, the toolset introduced in this section incorporates sound as a basis for such a data representation. Its lightweight, modular concept intends to help creating data-driven object augmentations. Systems, build according to Reim, draw on peoples' knowledge about every-day objects, whether they are as simple as sticks and stones, or more specialised and integrated into daily, technology-driven systems as keyboards or other computer interfaces. Rather than manipulating the object's intentional usage to represent data, Reim transforms the object's sonic characteristics to augment it with external information. This means that the sonic

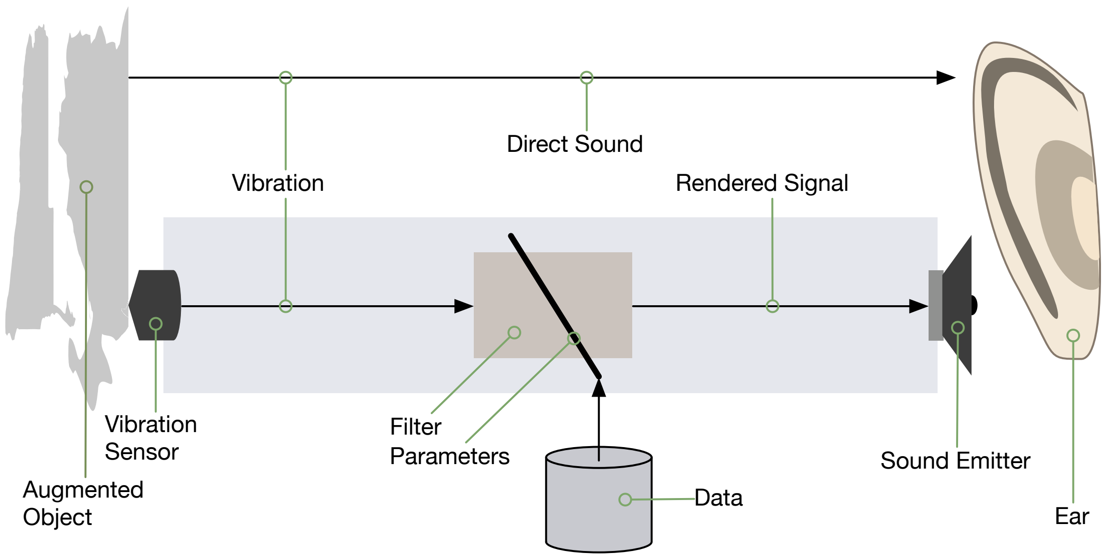


Figure 9.26.: General model of Reim-based auditory augmentations.

reaction to e.g. an excitation of such an enhanced object does not only reflect its own structure, but also features of attached data: It is virtually changed to render an additional information layer by data-driven features. We call this approach *Auditory Augmentation*. It can be formally described as the process of artificially shaping a manipulation-caused sound's gestalt according to external, data-driven parameters without changing its sonic presence and timing. *Auditory Augmentation*

Intents Reim helps to represent digital data as auditory characteristics of physical objects. When not attached to a specific object, a Reim-based system can alter the characteristics of arbitrary structure-borne sounds. Although powerful and built for non-linear analysis/exploration, this toolset is not intended and appropriate to systematically search for specific structure in data, or even to observe exact class labels for a data set. Moreover, it lifts the problem of observing structures up into perceivable reality, where the human ability to find and understand structural information can be utilised.

Assembly An auditory augmentation based on Reim consists of several parts: An audio-transducer captures structure-borne vibrations of objects, which are fed into an audio-filter respectively sound synthesiser. Additionally a mechanism to control the synthesiser's parameters according to external data is required. This, together with the filter turns the incoming signal into a perceivable sound, having information on (a) the source vibration and (b) the data under investigation. The resulting augmentation has no noticeable latency, and smoothly overlays with the original sound. The overall auditory character of the complete setup therefore depends on the input's audio characteristic, the filter, the data's state, and the sound rendering including possible distortion by the loudspeaker. Note that the resulting sound mixes with the real sound of the interaction, resulting in an enriched soundscape.

9.3.1. Usage Scenarios

To show the potential of Reim as a powerful toolset for data exploration that uses manipulation techniques as they are exemplified by Goldsworthy's leafs in Section 4.1.1, this section presents examples for the every-day usage of Reim as a an exploration system for abstract data.

Comparing data sets

Let us consider two data sets that share the same characteristics in distribution and local density. There are no obvious differences in their structure. A user wants to investigate if there are other, possibly non-linear structural differences between the data sets. By linking each data set to a Reim augmentation, he investigates into this direction.

Around him, the user collected surfaces of various characteristics: one of granite, one made of wooden, etc. He attaches the transducers of the Reim system to small glass objects and scrubs them over the surfaces. Each combination of surface, glass object/data set and scrubbing technique results in a characteristic sound. Exploring these combinations for differences between the sounds of each object enables the user to find structural differences between the data sets (see Figure 9.25). When he found interesting reactions, he captures and analyses the source vibrations (i.e. the sounds that appear when scrubbing the objects on the surfaces without the data-inherited overlay) for further analysis, because these sounds offer information on the non-linear structures in the data sets under exploration. It can be seen as a classifying discriminant. Instead of using only rigid bodies, it is also possible to attach the transducers to drinking glasses filled with grainy material of different sizes and shapes. The user then sequentially loads the data sets to the glass/tool aggregates and shakes them. This way he can test which of the glasses emit a characteristic sound augmentation that can be used to differentiate between the data sets. Both scenarios become more powerful by Reims feature to record and playback input sounds with different data sets. Also the feature to change the synthesis process as well as the range of the parameter mapping increases the flexibility of the system.

Monitoring Near Realtime Data

In a different scenario, the user wants to keep track of a slowly changing data stream like the weather situation around his working place. In order to acquire this information without being disturbed by a constantly sounding Auditory Display, or having to actively observe e.g. a webpage, he acquires the data automatically from the weather sensors and feeds them into a Reim object. After this, he attaches the connected transducer to a computer input interface that he is using regularly (e.g. the keyboard or the mouse) and then has an auditory augmentation of its structure-borne sound with the weather data. Every time the attached sensor values changes, the auditory character of the augmented device changes, giving the user a hint about current weather conditions.

Consequences

Adding auditory augmentation to structure-borne sounds means to insert a thin layer between people's action and an object's auditory re-action. The proposed auditory augmentation can be easily overlaid to existing sounds, and does not change prominent auditory features of the augmented objects like the sound's timing or its volume. In a peripheral monitoring situation, the data gets out of the way for the user if he is not actively concentrating on it. A characteristic change, however, tends to switch the users attention right back to it.

9.3.2. Related Work

Apart from the introduced auditory augmentation, also other human-computer interfaces where developed that utilise the user's familiarity with action-based sounds. The *audio-haptic Ball* for example senses properties like its velocity or force and feeds them into a Sonification Model resulting in an auditory and dynamic data representation [HKR02]. By this, it lets the user experience a physical model caused sound that displays, how data that is attached to it affects the models reaction to shaking or squeezing. Hereby, its auditory output directly corresponds to the users action and the model's reaction, which, again, relies on its data-driven configuration. The formal software development process for the audio-haptic Ball interface used for Model-Based Sonification can be described as (a) designing a physical model, (b) feed it with data-items, (c) shake it, and (d) render sounds according to the physical model. This approach especially requires the re-implementation of basic natural functionality, namely the dynamics of objects in a 3D-Space. Although this approach makes it possible to shake and squeeze data sets of higher dimensionality, it remains difficult to explain and understand what happens in such a space, and how the modelled n-dimensional object can be embedded into 3D reality to excite it via the audio-haptic Ball.

Model-Based Sonification and the Haptic Ball

The Pebblebox is another audio-haptic interface for the control of a granular synthesiser, which extracts information like onset, amplitude or duration of grain-like sounds captured from pebbles in a box. These high-level features derived from the colliding stones are used to trigger granular sounds of e.g. water drops or wood cracking to simulate rain or fire sounds [OE04]. The performance of the Pebblebox massively relies on the fact that the captured signal has to be a superposition of transient sound events. A change of the sound source like it is implemented in the Scrubber, another closely related interface also developed by the authors of the Pebblebox, has to extract a completely different feature set from the input signal. It is designed in assuming scrubbing sounds [EO04] in order to synthesise artificial scrubbing sounds. Auditory Augmentation does not rely on such assumptions: it directly uses the incoming sound source to drive a rather simple audio filter. Its output is directly played back to the user.

The Pebblebox

9.3.3. Level of Abstraction

Reim supports two different levels of abstraction:

Being-in-the-World incorporates mostly direct and physical manipulation with direct sonic feedback, whereas

Abstraction from RBI abstracts from the natural manipulation patterns.

In the first, more direct level of abstraction, the user's experience of an augmented object does not differ from handling non-augmented objects, apart from the fact that the object-emitted sounds are also data-driven. Due to his Being-in-the-World, the user feels familiar with the objects manipulation feedback. The user gets a feel for the process by gaining experience of the material-data compound's reaction over time. Non-linear complexity of material properties and their reactions to e.g. pressure and speed of action can therefore be used intuitively, i.e. without additional cognitive effort. Data therefore becomes integrated into everyday life.

Being-in-the-World

Abstraction from RBI

To gain assessment and increase repeatability in the explorative process of Reim, it is possible to capture the exact vibration of a physical excitation. It can be used to either repeat the data-representation process with the exact same prerequisites, or to sonify other data items with the same excitation sound.

This demand requires to capture the transducer's input and use it for the representation of several data sets as well as the addition of recording capabilities to the system such that the data's representation can be easily captured and replayed to others. Related to this are the offering of pre-recorded standard excitation sources, or the provision of a standard set of objects to add data-driven auditory augmentations.

9.3.4. Implementation

Reim is designed as an open toolset for sonic augmentation. Its implementation splits into hardware and software parts.

Hardware

According to the general model of Reim-based auditory augmentations (cf. Figure 9.26), the setup of a Reim-based system requires the following hardware: an audio transducer[6] (in the diagram referred to as *Vibration sensor*) for vibration sensing, an audio interface and a computer for capturing the sensed data and application of the filter model to the signal, and loudspeakers or headphones (*Sound Emitter*) for signal playback.

Software

We implemented some convenience classes that make it easy to apply data-based parameters to signal filter chains (*ReimData*), respectively to collect and store presets for the synthesis process (*ReimFilter*). Both, data processing and sound rendering is realised in SuperCollider [McC02] [WCC09].

ReimData; store data for Reim usage ReimData keeps track of the data under exploration. Since SuperCollider is divided into a synthesis process (scsynth) and a controlling part (sclang), which are interconnected by a network protocol, ReimData needs to synchronise the data resource between both sides. It's sclang interface is implemented as a class. It's programming interface is displayed in Figure 9.27. This code example shows how a ReimData object is instantiated:

```
q = q ? ();     // a dictionary
q.data = CSVFileReader.readInterpret( // load a data set
    "/localvol/data/share/testData/glass.csv"
);
q.data = q.data.flop[0..8].flop; // use first 9 Dimensions

// instantiate a ReimData object and fill it with the third row of the data
```

[6] Either a dynamic microphone like the AKG C411, or a piezo-based pickup system like the Shadow SH SB1.

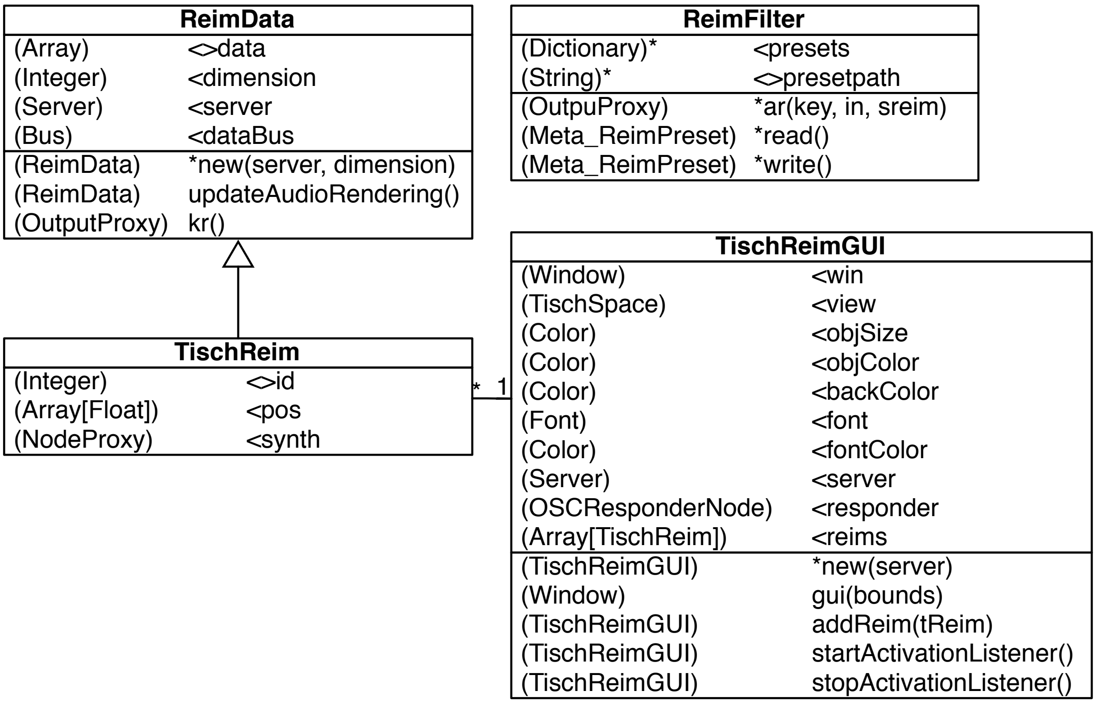


Figure 9.27.: UML diagram of Reim-related classes.

```
q.sData = ReimData(s, q.data.shape.last);
q.sData.data = q.data[2];
```

On the sound server side, where the sound rendering and processing takes place, it serves the assigned data on control buses:

```
s.options.blockSize = 32; // prepare Server for realtime control
s.options.hardwareBufferSize = 32;
s.options.numWireBufs = 64;
s.boot; // boot server
(
// create a Reim with the data
Ndef(\sReim, {|gThresh = 0.1, amp = 1, ringtime = 0.1, highFreq = 1000|
    var in;
    v-ar freqs, amps, rings;

    in = SoundIn.ar(0); // sound coming from the audio transducer

    // map Data to filter parameters
    freqs = Select.kr(q.sData.kr > 0, [100, q.sData.kr]) * 4000 + 2000;
    amps = q.sData.kr > 0;
    rings = q.sData.kr;

    DynKlank.ar( // The filter UGen: changes dynamically when data changes
        `[freqs.lag(0.1), DelayN.kr(amps, 0.1, 0.1), rings * ringtime],
        input: in * 0.25
    ).tanh; // clipping prevention
```

```
});
)
Ndef(\sReim).play;
```

Sound Processing and Filtering Presets with ReimFilter Central element for the sound processing is the global *preset* dictionary **Meta_ReimFilter:presets**, in which audio-filter definitions can be stored such that they survive system restarts. Per default, it contains one filter definition that can be used to augment data:

```
\reson -> {|in, sreim|
    // multichannel controls
    var freqs, amps, rings;

    var ringtime, highFreq;

    ringtime = \ringtime.kr(0.1);        // controls
    highFreq = \highFreq.kr(1000);

    freqs = Select.kr(sreim.kr.abs > 0, [100, sreim.kr.abs]) * 4000 + 2000;
    amps = sreim.kr > 0;
    rings = sreim.kr > 0;

    in = (in + HPF.ar(in, highFreq)) * 0.5;
    DynKlank.ar(
        `[freqs.lag(0.1), DelayN.kr(amps, 0.1, 0.1), rings * ringtime],
        in * 0.25
    ).tanh;
}
```

The *reson* preset is mainly a resonator bank, of which the frequencies, amplitudes and ringtimes are parameterised according to data values of the **ReimData** instance provided in the argument *to*sreim.

Other base filters for additional **ReimFilter** presets are convolution ugens or comb filters, as they can also be used to substantially change a sound signal without significantly changing its natural onset or duration. The following code snippet shows, how to create a new presets and store it in a personal repository:

```
ReimPreset.read; // read personal repository

ReimPreset.presets[\myReson] = {|in, sreim|
        // [...]
}
ReimPreset.write // write to personal repository
```

To get information on a preset's meta-parameter set, you may use

```
ReimFilter.controlNames(\reson)
```

(a) Emulating Paareim: Load data items into Stethoscope objects and move them over different materials.

(b) Emulating Schüttelreim: Place Stethoscope on a box filled with objects.

(c) Emulating Schüttelreim: Shake the box to hear the data augmentation.

Figure 9.28.: Usage scenarios for Reim *Stethoscope*.

9.3.5. Reim-based Applications

This section gives insights into systems utilising the Reim toolset for data exploration and ambient monitoring. Apart from the Reim Stethoscope that is a design study, all applications are demonstrated in videos on the DVD.

Reim Stethoscope

Augmented Object	Custom-built object with attached loudspeaker
Intended Usage	Data Exploration, Discussion on data-inherent structures
Data Domain	Arbitrary
Data Acquisition	Offline
Number of Objects	> 3

The *Stethoscope* features a self-contained hardware part that can be used to systematically gain new insights into the inner structure of arbitrary data sets. Its custom design shown in Figure 9.28 incorporates both, a vibration sensor and a loudspeaker. It's physical manifestation is closely based on the overview diagram in Figure 9.26. Together with a wireless access to load data and change the filter process, the resulting system is portable and self-contained since it does not need any additional hardware or software.

A typical scenario in which Reim Stethoscopes are used may look like this: *the user has three or more Stethoscope objects around him, each loaded with a different data set. By temporally attaching these objects to arbitrary artefacts such as stones, receptacles or surfaces, all their vibrations induced by operating them (e.g. by shaking or squeezing) cause an auditory augmentation, reflecting the characteristics of both, the augmented physical interaction and the augmenting data.*

Unfortunately, it was not feasible for this thesis to built such Stethoscope devices; instead we focused on two prototypes that are closely related to *Reim Stethoscope* and would be easy to realise with its hardware: *Paarreim* and *Schüttelreim* (See Figure 9.28). Their difference to Reim Stethoscope, though, is that they do not feature a direct Data-Object Identification as described in Section 7.2 but Controller-based Object Use. Their perceived auditory binding, however, is still perceived to be very strong.

Schüttelreim

Augmented Object	Bowl or Box filled with small objects
Intended Usage	Data Exploration
Data Domain	Arbitrary
Data Acquisition	Offline
Data Dimensionality	Arbitrary
Number of Objects	$1-2$

Schüttelreim is an alternative approach to the mentioned use case of active data exploration and comparison. In this setup, the transducers are attached to box-shaped objects filled with grainy material such as buttons or marbles. As shown in the Schüttelreim video on the DVD, the data-loaded filters can be excited by shaking the box. The attached transducer then captures the rattling of its content and feeds it into the **ReimFilter**. Speakers near the exploration area play back the augmentation in realtime. Similar to the Paarreim setup, Schüttelreim also allows to turn data into highly controllable sonic objects. By extensive use, people may learn to shake and manipulate the boxes in such ways that certain aspects of the data can be prominently perceived and possibly lead to a valid differentiation and classification of structural information.

Paarreim

Augmented Object	Small rigid object like a glass lid of wine bottle
Intended Usage	Data Exploration
Data Domain	Arbitrary
Data Acquisition	Offline
Number of Objects	$2-10$

With *Paarreim*, we designed an interface to actively explore and compare data sets. It is designed to cope with the *Comparing Data Sets* use case described in Section 9.3.1, which explains how active data exploration and comparison can be achieved with a Reim-based system. As an initial sound filter setup the *reson* example soundscape described in the **ReimFilter** description may be used. The interaction hardware in Paarreim envisions the transducers to be attached to small rigid objects with little natural resonance; for the prototype setup we took glass lids of wine bottles. Additionally, surfaces made of various material are needed, each having a characteristic structure. Last not least the sound synthesis can be made by any available loudspeakers. A possible system setup is shown in Figure 9.25; an example usage is shown in Figure 9.29 and in the example on the DVD. By scrubbing the transducer over other objects of various forms and materials, substantially different excitations of the data can be achieved, which in turn change the sound of the auditory augmentation. With Paarreim, the user gets detailed insights into the data retrieval structures and can learn to use specific material combinations that help him classify data into groups according to their sonic reaction.

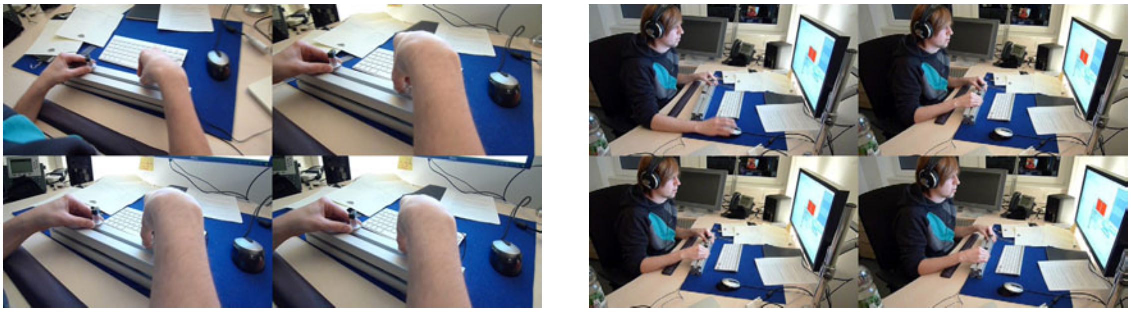


Figure 9.29.: Video stills from a Paarreim exploration session. It can also be found on the DVD.

Tischreim

Augmented Object	Surface
Intended Usage	Unobtrusive long-term Display with spatial resolution
Data Domain	Arbitrary
Data Acquisition	Arbitrary

Tischreim incorporates spatial relations into data augmentations with Reim setups. We illustrate its functionality by means of the following use case: *Consider a user sitting at a normal desktop in a typical working environment. He wants to have a certain information like the number of unread mails in his mailboxes, average CPU load, RAM, HD and swap usage permanently* at hand, *without either a possibly cluttering visual representation on his screen or on the surface that distracts him from his primary tasks, nor an Auditory Display that is permanently running.*[7] *Otherwise it should monitor the data streams on a subconscious level.* Tischreim's task then is to identify certain regions on the surface, and map the data-originating augmentation to surface excitations in these areas. To achieve this behaviour, a number of transducers were attached to the desktop, which capture all vibrations of its surface. A downstream signal processing unit extracts the location of the excitation and sends the raw signal to the *Tischreim* object that is responsible for that region. Each region therefore has a different meaning and a different sound, each connected to a separate **ReimData** object with a customised **ReimFilter**.

Software

Tischreim is the only Reim-based application that needs additional classes. As shown in the UML diagram in Figure 9.27, a **Tischreim** object inherits the main part of its functional range from **ReimData**. Additionally it holds an ID, information about the region it is linked to, and the connected **NodeProxy** for an easier setup of multiple audio augmentations. The following listing instantiates ten **TischReim** objects and fills each with one data row of a prepared data set:

```
q.sDatas = 10.collect{|i|
    TischReim(s, q.data.shape.last)
        .data_(q.data[i]).id_(i)
        .pos_({1.0.rand}!2)
};
```

[7]In this context, *at hand* means *available when needed.*

Now, the audio responder definitions for the Reim objects are defined, their region radii are set, and a control bus for the impact positions is allocated:

```
q.sDatas.do {|sdat|
    sdat.synth = NodeProxy.audio(s)
};
q.sDatas.do{|d| d.synth.play};

q.sDatas.do{|reim, i|
    reim.synth.source = {
        arg gThresh = 0.01, amp = 1, on = 0, radius = 0.1, myPos(#[0, 0]);
        var sqDistance, level;
        var in = DelayN.ar(SoundIn.ar(q.micInChannel), 0.1, 0.02);
        in = Gate.ar(in * amp, Amplitude.kr(in, releaseTime: 0.1)-gThresh)
                * on.lag(0.01);
        // compute squared distance from point of impact
        sqDistance = In.kr(q.posBus.index, q.posBus.numChannels).sum{|iPos, i|
            (myPos[i] - iPos).squared
        }.sqrt;
        sqDistance = sqDistance.linlin(0, 2, 0, 1).min(radius);
        level = sqDistance.linlin(0, radius, 1, 0);

        SendTrig.kr(Impulse.kr(5), i, level);

        ReimPreset.ar(\reson, in*level, q.sDatas[i]);
    }
};

q.sDatas.do{|reim|
    reim.synth.setn(\radius, 0.06, \on, 1, \gThresh, 0)
};

q.posBus = Bus.control(s, 2);
```

An **OSCresponder** captures the positional information that is send from an external program:

```
OSCresponder(NetAddr("localhost", 7000), \position, {|time, responder, message|
    q.posBus.set(message[1], message[2]);
}).add;
```

Last not least, a **TischreimGUI** is instantiated that shows the current configuration:

```
t = TischReimGUI(s);
t.gui(Window.screenBounds);

q.sDatas.do{|reim|
    t.addReim(reim)
};
t.startActivationListener;
```

Hardware

To acquire the excitation position, a custom hardware setup was needed. Principally, this can be done by *time delay* or *arrival or time reversal* algorithms, which compute runtime differences in the phase of signals that are captured by transducers attached to the edges of the surface [BCC+05]. Unfortunately, it was beyond the scope of this thesis to realise this scenario during Reim's development. Instead we used a graphics tablet with a stylus to identify and excite regions. We attached a transducer to the stylus in order to get a fairly good source signal. Moving the stylus around resulted in a scrubbing noise that was in turn processed by the responsible augmentation unit for that region. A video demonstrating this Tischreim setup can be found on the DVD.

Wetterreim

Augmented Object	Keyboard
Intended Usage	Unobtrusive long-term Display
Data Domain	Weather Data
Data Acquisition	1 update per hour

Communication shifts more and more to the digital: people post news to weblogs, twitter, chat or e-mail; even newspapers and flight schedules are browsable online. Therefore, the difficulty is not to acquire information, but our ability to process it. It is the attention that is the scarce resource [Gol97]. Information in the digital realm is often textually, i.e. symbolically represented: humans need to cognitively process it, even if they are only interested in a certain aspect. On the contrary, most natural phenomena represent information sub-symbolically, such that we are able to filter it easily for elements we consider relevant. Often this can be accomplished by subconscious processes. This especially holds true when the information transfer medium is sound. The manipulation of objects results in sounds that inherently transport information about the incorporated objects and the physical reaction. It is packed in a very dense form, yet is it easy to understand.

WetterReim utilises this feature for a dedicated scenario: the day-to-day work on a computer as it is common at almost any office workplace. As the source for the auditory augmentation, we chose the keyboard, one of the main interfaces for the daily work with computers. Typing on it results in a characteristic sound that is shaped by the design of the keyboard and its interplay with the writer's fingers. A contact microphone captured the keyboard's structure-borne sound, on which we based a Sonification of weather-indicating measurements. When filtering the captured sound by data-driven filter parameters, an audio stream is created, which is close to the characteristics of the original but features characteristics of the integrated data. The result is superimposed to the original sound such that it is perceived as one coherent auditory gestalt. The developed filter parameterisation for the weather data lets people perceive a drop in pressure or an approaching cloud front as a change in the object's auditory characteristic.

Implementation

The next code listings show how to set up a WetterReim system. Internally, a Python script [Lut06] is called, polling weather information to the SuperCollider implementation of WetterReim.

```
q = ();         //
q.server = s;
q.sourceChannel = 5;  // the transducer's channel
q.outChannel = 4; // output channel for the auditory augmentation
"%/WetterReim-DefaultSetup.scd".format(Document.current.path.dirname).load;
q.prepareServer; q.startServer; q.init;
q.startAcquireProcess;
```

```
q.addFilterPresets;
q.startAll
```

The actual synthesis process uses *multiFreqReso*, a **ReimFilter** preset based on a dynamic resonator bank that reserves two octaves for each data dimension in the sequence wind speed (lowest), humidity, temperature, pressure and (highest).

```
ReimFilter.presets[\multiFreqReso] = {|in, sreim|
    var freqs, amps, rings;
    var ringtime, highFreq, baseFreq;

    // controls that are accessible from outside
    ringtime = Control.names([\ringtime]).kr(#[0.1]);
    highFreq = Control.names([\highFreq]).kr(#[1000]);
    baseFreq = Control.names([\baseFreq]).kr(#[523]);

    freqs = (baseFreq.cpsmidi + sreim.collect{|reim, i|
         + ((i + reim) * 24)
    }).midicps;
    amps = sreim > 0;

    in = (in + HPF.ar(in, highFreq)) * 0.5;
    DynKlank.ar(
        `[freqs.lag(0.1), DelayN.kr(amps, 0.1, 0.1), rings * ringtime],
        in * 0.25
    ).tanh;
};

Ndef(\reims, {|amp = 1, vol = 2|
    var in = SoundIn.ar(q.sourceChannel);
    ReimFilter.ar(\multiFreqReso, in*amp, q.reimData) * vol;
}).playN(q.outChannel);

Ndef(\reims).set(
    \ringtime, 0.1, \amp, 0.2, \highFreq, 20000, \baseFreq, 100.midicps);
```

When finished listening (e.g. at the end of the day when packing up work), playback and data acquisition can be stopped by

```
Ndef(\reimsched).stop
Ndef(\reims).stop
q.stopAll
```

9.3.6. WetterReim Case Study

To gather feedback on the implemented auditory augmentation system, we conducted a qualitative user study where we asked three people to integrate WetterReim into their day-to-day work for a period of four or more days. After this period, we collected their statements in an unstructured interview.

Experimental Setup During the setup, the audio transducer was attached to the participant's commonly used keyboard (as shown in Figure 9.30). Its signal was fed into an external computer that was

Figure 9.30.: The hardware setup used by Participant 1. The transducer was attached to the external video adapter of her laptop. This made it easy for (dis-)assembly, since she only used WetterReim at her workplace, but carried her laptop with her.

exclusively used for data acquisition and sound rendering. The data that were augmented to the participant's keyboard were acquired from the nearest publicly available weather station. Its update rate varied between every half an hour and every hour. We used the filter setup *multiFreqReso* as described in Section 9.3.5. In an initial setup session, filter ranges were the adapted for each participant in order to reflect their individual preferences and the sonic character of their keyboard.

Observation

Overall, our observations based on the unstructured interviews unveiled the following aspects:

Sound design Participant 2 found the used ringing sound to be natural and pleasant. However, Participant 1 reported that the augmented sound irritated her in the beginning. Participant 1, Participant 2 and Participant 3 stated that they missed the sound when it was absent by accident.

Localization Participant 2 found it astounding that the sound seemed to originate from the keyboard although the loudspeaker was at a completely different position.

Data-sound mapping The differences in the rendered sound according to the data were considered by Participant 1 and Participant 2 to be reasonably distinguishable, even without direct comparison.

Exploration All participants reported that they also used the setup playfully; Participant 2 and Participant 3 stated to actively trigger it by purpose to hear the system's actual state.

Attention Regarding the subconsciousness of the sounds, participants reported mixed feelings. While Participant 1 found it difficult to shift her attention away from the sound, Participant 3 stated that a change in feedback was rising his attention even

User	Work Days	Weather Conditions
Participant 1	4 days	contrary weather, changes between 35°C, sunny and 20°C with thunderstorm and sometimes heavy rain in the evening.
Participant 3	10 days	constant over the time, no rain, around 20°C
Participant 2	8 days	20°C– 25°C, rainy and sunny

Figure 9.31.: The weather conditions for each participant during the WetterReim study.

when he was concentrating on something different. However, no participant mentioned the system to be bothersome.

Loudness The adjustment of the augmentation's volume was experienced by all users to be difficult. Especially Participant 1 reported to usually type relatively weak, making it difficult to properly adjust the amplitude of the augmentation.

Findings Based on this first study of an auditory augmentation setup in a real-world situation was a success. Its general application – unobtrusive monitoring of near realtime data – proofed to work for the participants. We especially found out that (*a*) users perceived the auditory augmentation and the original sound as a single natural sound, (*b*) they were not bothered by the Sonification, and (*c*) they had difficulties adjusting the volume of the auditory augmentation. For a future setup, we plan to investigate into this issue.

9.3.7. Conclusion

Summary In this section we introduced Reim, a toolset to help in the design of Tangible Auditory Interfaces for data representation. It utilises everyday objects and their interrelations to transform abstract data into physical manipulatable and auditory perceivable artefacts. The data is perceived as an auditory augmentation, complementing the natural structure-borne sounds with data-driven reactions. The proposed Reim model was applied to several design studies featuring different usage scenarios including active data exploration and subconscious monitoring situations.

Lessons learned During the setup of the different applications described in Section 9.3.5, we experienced that latency plays a prominent role in Reim-based applications. Long delays (bigger than about 20ms) between user action and system reaction broke the illusion of sonic identification and compactness of the object and its augmentation. Small scale locational separation between structure-borne sound (i.e. transducer location) and its augmentation (the loudspeaker), however, did not affect that illusion.

Because of Reim's simple technical assembly, it is easy to understand by users and turned out to be applicable for a long-term case study. During such a study on WetterReim, people used the auditory augmentation in their usual working environment. It augmented regularly updating local weather measurements onto structure-borne sounds of their computer keyboard. Participants reported that the augmentation worked well, though it turned out that the particular data domain was not of much use. However, the augmentation was perceived as part of the augmented object. Participants were also able to differentiate between several weather situations.

Many participants stated that they were not able to separate source sounds from data-driven sounds. Although this is an essential effect regarding the acceptance of the system, it uncovers an inherent issue of Reim-based applications: the sound of the data object combination is perceived as an entity; users are not able to split it into its components to separate the data-communicating part from the structure-borne sound. Long-term usage of a Reim-based system, though, should overcome this effect. People will adapt to the auditory specifics of the used objects and develop implicit knowledge on how to separate the physically induced sounds from the representation of the displayed data. This effect is supported by the fact that the physical part of the sound bases in a static set of parameters, reflecting the same characteristics in all excitations. Changes in the sound therefore always originate in a change of the data-driven augmentation.

These observations and considerations make us believe that Reim based systems are promising for future developments of TAIs for both data exploration and subconscious monitoring.

Figure 9.32.: AudioDB, a system for sound sample representation.

9.4. AudioDB

AudioDB, the TAI that will be presented in this section was first published together with Christof Elbrechter, Thomas Hermann and Helge Ritter in the *Proceedings of the 14th International Conference on Auditory Display*, and it was presented in a live demonstration at the accompanying conference in Paris [BEHR08].

Audio recording is common in analysis-based research areas to get information on people's behaviour during experiments. For their analysis, it is beneficial to first automatically label the recordings according to their content, i.e. to detect whether they contain artificial noise, a sneezing, rattling, speech or other characteristic sounds. This classification process can be automated by machine learning systems. The quality of such an automated classification usually is measured according to its performance against a *gold standard*, where human experts labelled (parts of) the data. To simplify the creation of such a gold standard, experts should be in the position to operate and manipulate sound snippets while discussing their classification. It is beneficial for the discussion process and the acceptance of its result to provide all attending people with the same opportunities and rights to operate the sounds. Providing a low entry-level for the user action shifts the limiting factor for possible user interaction from technical skills towards the group-inherent social hierarchy.

Another scenario that features the same prerequisites is the searching and sorting of digital audio as it is common for audio researchers and sound engineers. Searching and sorting of sounds that were collected in extensive databases (e.g. sampling libraries for musical production or seismographical studies) is a difficult task. The common technique of tagging sounds and other media files, as common in web-2.0 applications, like FreeSound[8] or

[8] http://www.freesound.org/

Flickr,[9] has the drawback that it needs descriptive words. Especially when dealing with abstract sounds, this technique turns out to be very difficult to handle because the auditory impression of a sound heavily depends on its semantic. The sound of frying bacon in a pan for example can easily be mistaken with raindrops on a ribbed roof [FBB05].

Motivation

For both use cases, a human computer interface for collaborative use of sound-based data is needed. *AudioDB*, a surface-based Tangible Auditory Interface was designed to support such collaborative navigation in information databases. It provided a tangible environment to sonically sort, group and select auditory representations of data that are represented as physical artefacts on a surface. AudioDB's intend was to serve as a low-threshold interface to audio data that is used by several people during a discussion. However, as can be seen by the very different use cases described, AudioDB was not intended for one dedicated field of work. Moreover, it served as a multi purpose and multi data tool, ready-to-hand to gain insight into various kinds of digital information represented by sounds. To reflect the outlined prerequisites, we paid particular attention to provide users with the same manipulation capabilities, independent from their location around the interface. This lead to a hardware design with no designated *front*, or primary place to stay. With grains as the primary interfacing objects (see Section 5.5.2), AudioDB served as a mediator for sonic content by augmenting sounds with tangible representation.

9.4.1. Intended Features and Behaviour

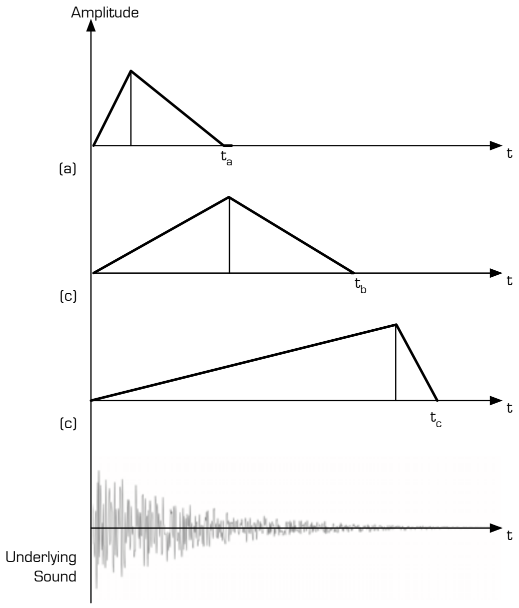
The functionality of AudioDB was designed to be simple in order to keep the entry-level for inexperienced users low. Every time an object is moved a certain amount, a part of the linked sound is played back. The speed of the movement hereby determines which part is played; slow movements cause playback of the transient first part of the sound, whereas a fast movement triggers playback of the (often more tonal) tail of the linked sound. Note that the implemented linkage is continuous, creating a linear mapping between the movement speed and the playback position in the sound file. This technique allows to present the information-stuffed transient onset as well as the tonal decay phase in a continuous stream (see Figure 9.33(a)).

Shifting the level of abstraction

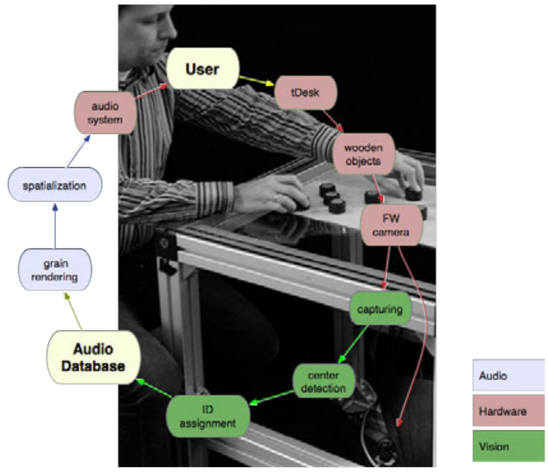
As described in Section 5.4.3, one particularly powerful feature of TIs is their ability to shift control between different layers of data abstraction. AudioDB takes advantage of this feature by providing grains with two states: each of its two sides (a and b) represents a specific data abstraction. If a grain is placed on the surface with side a turned upwards, exactly one sound is attached to it. The grain is in *Node Mode*. By moving the object, this sound is granulated and presented to the user. If turned upside-down, such that side b is facing upwards, the grain changes its mode to *Cluster Mode*, i.e. it is now attached to several sounds at a time. By moving the Grain (i.e. the virtual compound), all attached sounds are auditorily represented (Figure 9.35).

This meta-control allows the user to decide on a per-object basis, if he wants to manipulate either a single sound or a cluster of sounds. In the first case, he gains full control over its location representation and relative position, whereas the latter allows him to abstract from the sound–object relation, and operate with a whole sound cluster but with the drawback that he is not able to dynamically change a specific sound's cluster affiliation.

[9] http://flickr.com



(a) Dependency of duration and attack phase in processing the raw sound material.



(b) Conceptual figure of the data flow.

Figure 9.33.: Technology overview of AudioDB

Implications for user operation

The described simple functions of AudioDB lead to a variety of opportunities for users of which we here present a selection.

Cluster, Sort, and Order The identification of sounds to objects allow users to arrange them to clusters. As will be shown in Section 9.4.3, users tend to create such clusters on the fly, possibly with additional internal structure such as sequences or sub-clusters.

Distribute Work or tasks associated to clusters may be distributed to people standing around the surface by simply placing them in their front. The system itself hereby does not force a technical-caused dominance based on the skills or locations of people around the surface, such that social interrelationships that are not based on the technical system can take over.

Dynamics users are able to move grains either individually or together in one hand, blurring the abstraction between *cluster objects* and *node objects*.

9.4.2. Technology

Hardware

As a basis for AudioDB, we use the *tDesk* (described in Section 5.5.1), a tabletop system for Tangible Interfaces. Its dimensions suited well the desired multi-person set-up required by AudioDB, and intended to equally serve people standing around the desk, and provided each member direct access to the surface. By its dimension of 80×80cm all places on the surface can be captured easily by an adult person.

Below the surface, a digital camera captured the positions of objects. The location of the

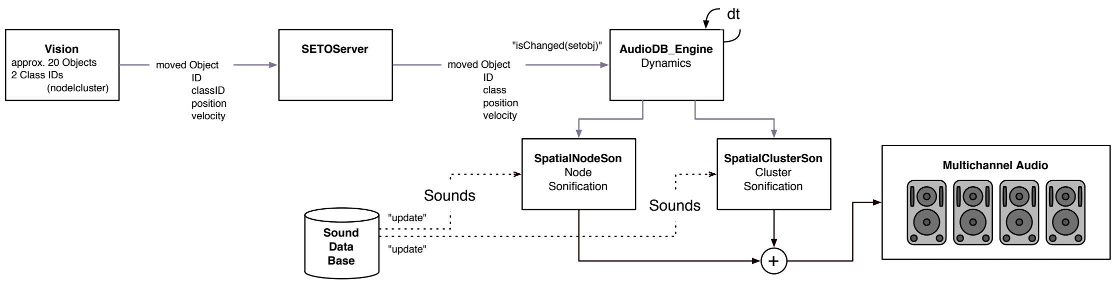


Figure 9.34.: Overview of the AudioDB software and its interdependencies.

camera prevented otherwise prominently appearing visual occlusions by the users' arms or other body parts.

The camera's image was processed by a blob tracking algorithm implemented by Christof Elbrechter. It detects number, colour and position of the grains' underside and additionally applies a unique ID according to the Algorithm by Cox et al. [CH96]. Our in-house implementation of this algorithm can process up to 50 objects in realtime on a recent computer system, while capturing and processing the image from the firewire camera in 20 fps. This is sufficient for a smooth interaction with AudioDB.

The system's Auditory Display was rendered to an 8- resp. 16-channel audio system arranged in a ring of equidistant loudspeakers, surrounding the tDesk. This allowed for a natural auditory interface, directly coupled to the users' action on the tabletop.

Software

Based on the described hardware, a system where implemented in SuperCollider [McC02] [WCC09]. As shown on Figure 9.34, it consists of a controlling and a sound synthesis part. The objects' motion is tracked by the system, send to SuperCollider, administered there by a **SETOServer**[10], and used as input to the data model called **AudioDBEngine**. In this object, each grain's positional information is linked to a corresponding data item. Mode, motion, speed and position of the object then determine the Auditory Display state as described in Section 9.4.1.

Shifting between modes

One aim of the system is to support users in sound classification. The user can achieve this by establishing clusters through the addition and removal of sounds to and from clusters. The system therefore implements rules to shift between the two abstraction modes as follows:

Node → Cluster

Turning an object from *Node Mode* into *Cluster Mode* triggers the system to collect sounds from all objects nearby. This means, that all these sounds are collected and implement the new sound set of the just established *Cluster Mode* object. The affected objects are assigned to new sounds, if they are in *Node Mode*, or otherwise – if in *Cluster Mode* – left empty. To decide which objects are affected by the restructuring process, the system invokes a hierarchical clustering process that builds a dendrogram of all the positions of the objects on the surface.[11] This dendrogram contains information about the distances between each object and the next cluster of objects nearby. Based on this information,

[10]See Section 10.2 for details.
[11] A short overview of Agglomerative Clustering is given in Section 9.7.1.

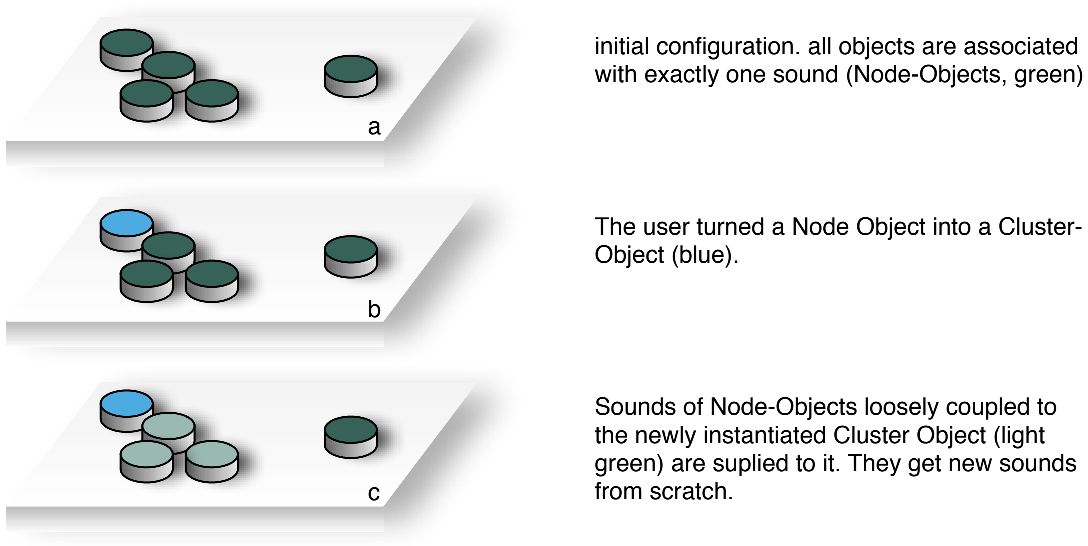


Figure 9.35.: Example layout for the transition from *Node Mode* to *Cluster Mode*.

AudioDB merges all objects in the sub-tree of the dendrogram that include the turned object and are separated from the rest of the objects by a given threshold. Figure 9.35 displays a step-by-step illustration of the transition process focussing on the clustering, whereas Figure 9.36 exemplifies the transition process from the view of the flipped object focusing on the actual algorithmic rules for sound distribution and collection. Turning an object from *Cluster Mode* into *Node Mode* distributes the contained sounds to the surrounding *Node Mode* objects.

Cluster → Node

Sound synthesis

The feedback of information to the user is realised by spatial granular re-synthesis based on the corresponding data item and its auditory representation. Each rendered audio grain is a part of the sound's onset multiplied by a curve with a sharp attack, or a longer part multiplied with a smoother envelope. Transient respectively decaying parts in the granular sound stream are chosen to be uniformly distributed over time. Information on attack and decay of the underlying sound therefore is kept in the resulting steady sound stream. To closely link the AD to its corresponding physical object, we render the sound to originate from the same direction the object is located with respect to the tDesk's centre.

As explained in the introduction to AudioDB, duration and attack of a single sound grain depended on the grain's speed of movement. As shown in Figure 9.33(a), these parameters are coupled with each other. The envelope's duration therefore determines its attack, i.e. how much of the transient part of the original sound is audible. For each grain in *Node Mode*, one synth is created according to the following **Synthdef**. The **bufnum** argument links to the sound file that is associated to the actual grain.

```
SynthDef(synthName, {|out=0, bufnum=0, dur = 0.1,
    amp = 0.05, orient = 0, width = 2|
    var player = PlayBuf.ar(1, bufnum);
    var env = EnvGen.ar(
        Env([0, 1, 0], [0.8, 0.2]*dur, [-1, 1] * ((dur*5).reciprocal-1)),
        levelScale: amp,
```

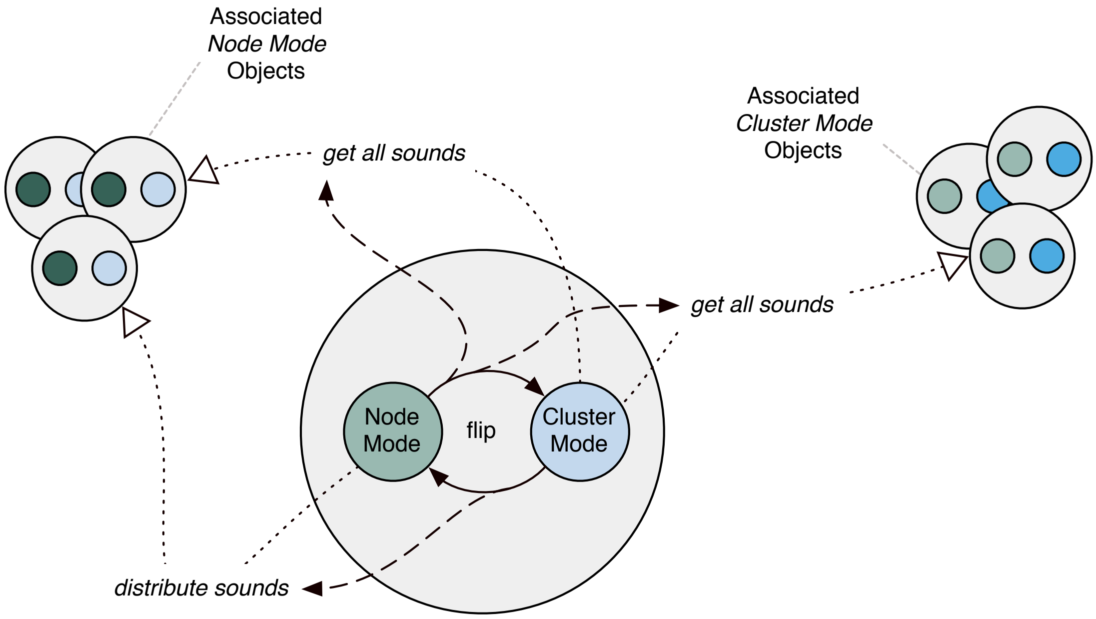


Figure 9.36.: The transition from *Node Mode* to *Cluster Mode* viewed from the object that flips. Depending on the state, it either collects all sounds from objects nearby, or distributes its sounds to the surrounding node-mode objects.

```
        doneAction: 2
    );

    Out.ar(
        out,
        PanAz.ar(numChans, player * env, orient, width: width)
    );
}).send(server);
```

The *Cluster Mode* uses the same synth definition for each associated sound as utilised in the *Node Mode* for one single sound. The resulting grains are spread in time and space:

```
buffers.do{|buffer, i|
    server.makeBundle( server.latency + ((i*0.005) + 0.01.rand), {
        Synth.grain(synthName, [
            \bufnum, buffer,
            \dur, #[
                 0.05,  0.01,  0.02,  0.04,  0.08,  0.16,
                 0.2 ,  0.32,  0.4,   0.64,  0.8 ,  1
            ].wchoose(#[
                12, 11, 10,  9,  8,  7,
                 6,  5,  4,  4,  4,  4
            ].normalizeSum),
            \orient, ((
                (pos).theta + 0.5pi
            )  * pi.reciprocal
            \width, 4 - (3 * pos.rho * 1/(2.sqrt)),
```

```
            \amp, 0.25 * numBuffers.reciprocal * speed
        ], target: server)
    })
}
```

This results in an asynchronous grain cloud as it is described in Section 6.4.1.

9.4.3. Case Study

Experimental Setup

Tools

We conducted a case study on AudioDB using the methods described in Section 4.6. In this study, four users were asked to solve a simple task: to arrange the presented sounds in an ordered fashion. For this, each participant had the same AudioDB setup. It consisted of 13 sounds, each linked statically to one grain (i.e. *Node Mode*). We did not include Cluster Mode, because we wanted to test for general feasibility of the object-sound linkage. With this open scenario, we aimed to generate hypotheses for further development and gain insights into how people may manipulate and sort sounds associated to Grains. We recorded the participant's tries with a camera from above, and analysed the videos according to the qualitative methods derived from grounded theory. The following paragraphs describe the observed behaviours. It is complemented by a report on findings resulting from this case study.

Participants

In the following descriptive part on the participants' performance, we use the same abbreviations to identify hands and fingers as in Section 4.1.2. A diagram explaining this coding is shown in Figure 4.2. The locations on the canvas is abbreviated by LL (lower-left), UL (upper-left), LR (lower-right), and UR (upper-right).

Participant 1

Participant 1 where using the interface for about four minutes. She developed object sequences in a short period of time. After half a minute, she started to arrange the 13 grains in two clusters, one horizontally oriented (LL to LR) containing basedrum samples, one vertically aligned (UR to LR) consisting of piano samples. From the beginning of their emergence, grains in clusters were arranged in rows, implicitely representing an order. During the clustering process, Participant 1 used the canvas' center for exploration.

At sec. 100, Participant 1 found two objects that did not fit into the other clusters; she grouped them between the other bigger clusters. After 105 seconds, she changed her strategy, and turned to intra-cluster exploration of the vertical cluster (UR-LR) with the piano sounds. During this process, she compared grains one by one and re-arranged them in the cluster's line-up. This intra-cluster exploration took about 1.5 minutes (until sec. 198). During this phase, there where moments lasting about two seconds, in which she did not manipulate anything. After rearranging the first big cluster, she applied the same strategy to the other cluster (LL-LR, basedrum). In the last 10 seconds, Participant 1 arranged the two big clusters to reflect their perpendicularity (as she explained afterwards). The two-grain cluster was placed between them.

Participant 2

Participant 2 used AudioDB for 2.5 minutes. He starts with very fast moves of one object each, exploring its audio by manipulating it in the middle of the canvas. He supplements each object manipulation by moving it to either left or right, depending on its sound.

During the first 45 seconds, he arranged the grains into four clusters, each located in one of the canvas' edges. Most of the clusters are sequenced similar to Participant 1. At sec. 45, Participant 2 changed his strategy towards the integration of the clusters into a large sequence near the lower edge (LL-LR). While in the first part of the case study a clear separation between exploration movement and placement movement could be observed, Participant 2 increases the efficiency of his movements such that a clear separation could not be observed anymore (se e.g. sec. 136.5). Participant 2 finished the object sorting with a sequential arrangement of all grains into a half-circle, with the open side facing towards him.

Participant 3

Participant 3 used AudioDB for 12 minutes. Her first movement was the relocation of all objects from the centre to LR, creating a "stock" of objects. The next three minutes, Participant 3 explored one grain after the other, picking it from the stock, shaking it in a free place of the canvas and sorted it into regions. She manipulated and relocated all grains that were not in the stock. Then, she took the next grain from the stock. After the three minutes, Participant 3 used up the stock. All grains were distributed over the canvas. She had manipulated and decided for each on its location. Then, she started to arrange the objects in linear-shaped, vertically aligned clusters. All objects hereby where explored concerning their placement in the cluster (*inter-cluster exploration*). During this restructuring, the clusters rearranged into vertical lines (sec. 380 onwards). The movements were more fluid than in the beginning.

Participant 4

Participant 4 used AudioDB for four minutes. He needed 30 seconds for initial sound exploration and rough sorting. After 196 seconds, he had all objects arranged in a sequence. The rest of time, he manipulated each object one by one (up to sec. 224), and rearranged the circle-like line-up into a diagonal line from UL to LR.

Observations

Hand usage

All participants used only one hand for operating the interface during the first minutes. This might originate in the circumstance that we did not explain the interface and its possible handling to the participants beforehand. After a while, though, Participant 1, and Participant 3 also used their left hand, sometimes exclusively (e.g. Participant 3: sec. 219 – sec. 235), sometimes in a true bimanual fashion (e.g. Participant 3: sec. 343; moving many objects with LP RP; both separated from each other, sec. 446.5; true bimanual manipulation).

Grasping

We identified four different grasping types during the case study. The most frequently used grasp was clearly the *precision grasp* (incorporating R1, R2 and sometimes R3). In this grasp, the grain is hold between the incorporated fingers and moved over the surface. It was used by all participants, however, Participant 2 and Participant 4 used it almost exclusively.

The next frequent movement type was the *fingertip movement* (incorporating R2 and sometimes R3), where a grain is manipulated by pressing the fingertip on its top tucking it between the finger and the surface. Participant 1 and Participant 3 often applied this technique (e.g. Participant 3 in sec. 2 with R2, sec. 51.5 with R2 and R3), Participant 2, Participant 4 did so only once, respectively twice.

A *flat hand movement* with the object between the surface and the fingers R2, R3, R4 where also sometimes applied. In this manipulation, the fingers where either closed or spread (e.g. Participant 1 in sec. 87, or Participant 3 in sec. 291.5 applying it for a two-object movement).

Participant 3 finally had a unique manipulation type. She pushed an object with R2 (sec. 161, 403) respectively R2R3 (sec. 403, 469.5) into the desired direction.

Movement types

In the analysis, we found two semantically distinguishable movement types:

Exploration Movement (EM) is undirected, slow, and often interrupted, the (often following), whereas

Positioning Movement (PM) is directed towards a determined position and usually executed with more speed.

we observed a prominent example for an EM in Participant 1's try: During seven seconds (from sec. 104 to sec. 111), she made an extensive and undirected fingertip movement (R2) with several breaks. This movement is not directed, but clearly intentional, thus it indicates that she moved the grain for sonic exploration of the associated sound. This kind of movement was observed in all other tries. Even further, Participant 2's EM's even can be interpreted as highly symbolic, since he always repeats a certain up-down ritual (see e.g. sec. 22, 24, 26).

Often an EM was followed by a short interruption, followed by a PM, in which the grain under exploration was put to a new place (e.g. Participant 1: sec. 15, Participant 2: sec. 24).

Multi-grain manipulation

After 87.5 seconds of single-grain manipulation, Participant 1 realised that she can also manipulate more than one object at a time: She moved three pre-clustered objects with her right hand, followed by a movement of a fourth object in the cluster towards the canvas' LL edge. From this time on, Participant 1 did similar movements, always for cluster re-location (e.g. at sec 103). At sec. 163, Participant 1 moved an object between two others causing their displacement. It is unclear if this was done intentional. Participant 3 on the other hand, started her try with a complete re-arrangement of the grains using both hands. This was followed by 50 seconds single hand usage after which she slowly started to use two hands simultaneously (sec. 50, sec. 216, sec. 291, sec. 316, sec. 320, sec. 377).

Not all participants ever manipulated more than one grain at a time: during the relatively short time in which Participant 2 used the interface, he only used his right hand and manipulated only one grain at a time. Participant 4 on the contrary, did a very complex EM incorporating three objects in sec. 62: He first manipulated one grain (A), followed by an EM of another object (B). During this manipulation, he collected A with the same hand. After three seconds, he let go B and further explored A.

Canvas usage

All participants made use of the complete active canvas. However, strategies varied between participants and also changed over time. While Participant 1 explored the sonic behaviour of the grains at their original place, and then decided to move them to a new position, Participant 2 and Participant 4 incorporated a dedicated exploration field at least in the first part of their tries. Yet, when turning more towards inter-cluster selection, they also changed their exploration strategy to make local manipulations. Participant 3, finally, used no dedicated exploration area, moreover, she used the place that was free of other grains. Furthermore, she created a "stock" of objects in the LR corner of the canvas, reserving the canvas' rest for exploration and ordering.

Airtime

Although we told the participants that the interface where not capable to track grain movements away from the surface, three of them occasionally moved the objects in free space over it. Participant 1 for example moved an object through the air in sec. 177, placing it at the other end of sequenced items, Participant 2: bridged a PM over other objects in sec. 19 and sec. 20, and Participant 3 tried to move objects in free space in sec. 434.

Besides different grasp– and manipulation modes, also a higher-level interpretation of the participant's manipulation can be observed. There is evidence for two such exploration modes: *Inter-cluster exploration* means exploration of grains that are not associated to a cluster, whereas *Intra-cluster exploration* means exploration of one cluster and its elements. While both modes may also includes grain comparisons, inter-cluster exploration is more open, whereas the sorting and positioning movements in intra-cluster exploration have a more formal approach. To proof the variety of user handling, we give three examples of the many different user actions:

Inter-cluster exploration

- Participant 1 used centre-based exploration, e.g. EM of grain A in the centre (sec. 20), followed by EM of grain B in the centre (sec. 22), followed by PM of grain A to left cluster (LR to UR) (sec. 23), followed by PM of grain C to centre, followed by EM at sec. 27.
- In the time from sec. 39 to sec. 47.5, Participant 2 made a long exploration movement of one object incorporating sweeping movements, possibly using it to also explore the TAI's spatial aspects.
- Participant 3 did an alternating operation of two objects A, B to compare them.

Intra-cluster exploration, on the other hand, splits into three distinguishable manipulation styles:

Intra-cluster exploration

Sequential comparison means that an existing (often linear arranged) cluster is moved one grain after the other. Typically, a grain's movement is short and perpendicular to the line up inside the cluster. After movement, the grain remains displaced from that line, and the participant continues with the next object. After moving all objects, their arrangement is slightly different than before. Examples for this movement can be found in Participant 1's try in sec. 69.5 or sec. 42, in Participant 2s try (sec. 30), or in Participant 4's try (sec. 110). The latter is particularly interesting, since here a sequential exploration of all 13 objects on the table is made, followed by a control exploration into the opposite direction.

Parallel comparison is another possibility that can be used to explore audio that is attached to grains. Since parallel movement of objects with one hand are not extensively done, there is only one example of such a movement by Participant 3in sec. 413. It lasts about 2.5 seconds and incorporates three grains moved in parallel with the right hand.

Restructuring is a PM in which objects are re-arranged. Typically, it is a follow-up to a previous sequential or parallel comparison. An example is given in Participant 2's try at sec. 126.5.

Findings

With the case study, we found out that all participants were able to connect the objects with their corresponding sounds without significant efforts. They were able to assemble the objects into an order and explain the meaning of the different object groups afterwards. The relatively long examination of Participant 3 and her increased fluidness in her movements let us believe that it is possible for users to turn their impression of AudioDB from present-at-hand to ready-to-hand, such that they are fully concentrated on the sounds and their handling rather than on the object manipulation.

After an initial phase, participants started to use both hands, and also moved several objects at a time. We believe that after a learning phase people will more extensively use of this feature. For additional studies, however, an explanation of the system is needed, otherwise much exploration of the system itself has to be done before users actually start to use it for the intended purpose.

Participants reported that they had problems to identify one particular sound that had both, sharp attack and tonal decay. Apart from this, all where able to differentiate the sounds from each other and sort them according to their auditory character.

9.4.4. Conclusion

In this section, we described AudioDB, a TAI to sort and organise short sounds. AudioDB makes use of Grains (as described in Section 5.5) and granular re-synthesis. Each audio sample under exploration is associated to a such a Grain that is located on a 2d canvas. AudioDB is intended as a technical rather than aesthetic representation of sounds. Its feature to shift the level of abstraction between node-mode and cluster-mode gives its users a powerful tool for the exploration of large-scale sound databases.

Lessons learned In the presented case study, we found out that people were able to correctly identify sounds with objects. The TAI allowed them to separate and sort even similar sounds. We also found out that users can easily adapt to the artificial mapping between the sounds and the tangible objects and can effortlessly solve tasks like sorting the presented sounds.

During many demonstrations to guests of our interaction laboratory, we found that also inexperienced users tended to grasp the tangible objects and started manipulating. They forgot about the technical system and seemed to manipulate the sounds directly, having the interface ready-to-hand. In these situations, we also experienced that the maximum number of objects on the surface should not be too big. Otherwise people get confused about the sound object identification. This restriction may be overcome either by additional visual annotations like IDs or names next to the grains, or with the usage of the introduced cluster mode.

Due to the round objects, people do not ask questions about the effect a rotation of an object may have to the linked sound sound; a question we had to answer often when replacing the grain-type objects with TUImod objects (described in Section 10.1).

On a perceptual level, we figured that spatial correspondence between objects and sounds is of particular value for the perception of sounds, though it was not missed by users when absent. Due to the human spatial separation and association capabilities, however, we think that the lack of a spatial relation between sounds and objects would have a negative impact on the users' object identification performance.

Outlook Although the case study results are promising regarding the user interface quality, AudioDB can be extended and improved, especially on a technical level. Currently, the complete controlling depends on the performance of the visual tracking system. An improvement of its frame rate, for example by developing an alternative to the visual subsystem, would massively increase the interface's usability. The current frame rate of 60Hz at maximum restricts the sound manipulation control to this (in auditory terms) slow controller rate. Values larger than 200Hz would allow to use the objects' speed to be mapped more reliably to the granulation of specific regions in the source sound, and could be used to introduce features like object shaking characteristics for the sound rendering control.

9.5. Tangible Data Scanning

Tangible Data Scanning was first published in the *Proceedings of the International Conference on Auditory Display, 2006* together with Thomas Hermann and Helge Ritter [BHR06a].

Tangible Data Scanning (TDS) is a *TAI* that is attached to a Sonification Model following the Model-Based Sonification approach of Hermann [HR99]. TDS provides direct access to abstract data sets using auditory augmentation and physical interaction. To achieve this, the space surrounding the user is augmented with a model space originating in the data. It comprises of virtual objects that are derived from the data items under exploration. They establish a fixed link between data set and real space. A plate that serves as the central Tangible Interface is connected to a virtual body that scans through the acoustically active objects that represent the data. Every time the virtual part of the exploration tool touches a data object it reacts with an audible excitation. This immediate feedback creates a strong spatio-auditive metaphor between the tool state and the sonic reaction of the system that helps the user to understand and relate his activity to the data set's inner structure. The connection of the real and the virtually linked data space enables the user to connect spatial landmarks to specific aspects of the data: A mentally link between physical artefacts and data-inherent structures like clusters or local density peculiarities is established. With the help of these associations, users are able to remember spatial data properties, even if the system is stopped.

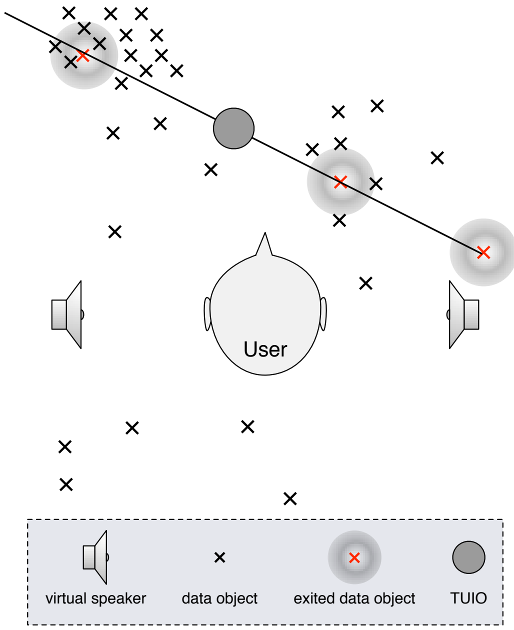
In the following sections, I will explain the concept, technology and implementation of TDS and will give usage examples for synthetic and real-world data sets.

9.5.1. Concept

TDS' intention was to integrate data into the users physical environment. It was designed to connect both space-related recognition and remembrance – both particular strengths of the human mind – with the abstract realm of data. The resulting setup should not only help to get insights into data-inherent characteristics, furthermore it should also support their communication between users.

The design resulting from these considerations allows to explore high-dimensional data by acting in a three-dimensional representation of the data under exploration. This data is represented as virtual, sounding objects that are fixed in space at locations determined by the data. The objects therefore literally become a (virtual) cloud that is anchored in the lab space. It surrounds the user and encourages him to recognise his environment as mental anchors that are augmented by the data. To explore the data objects, a physical object is introduced (e.g. a plate as described in Section 5.5.4). It is linked to a corresponding virtual object in the model space (e.g. a plane). The movements of the real object are retraced by the virtual object in model space which results in intersections with the data objects. This excitation causes them to produce a short percussive sound (like that of an excited spring-mass) that can be perceived by the user. An auditory effect therefore occurs only in response to user activity. Due to this immediate response of TDS to data item excitations, a tight coupling between user action and the system's reaction is established. This allows the user to reveal not only the data's inherent spatial distribution, but also to get insights into the more complex features like local density or topographic organisation. In their entity, the system's acoustic response represents the data set's internal structure.

(a) Tangible Data Scanning Sonification. The photo shows a user with an interaction tool, scanning the data cloud in data space that has been identified with the physical space. Headphones plus headtracking allow convincing spatial perception of intersection events.



(b) A conceptual overview of Tangible Data Scanning.

Figure 9.37.: Conceptional and real TDS layout.

In difference to approaches for analytical data exploration that compute density distributions or clusters, TDS displays only quantitative information about the data which has to be interpreted and cognitively processed by the user into higher-level structures. High or low data item density for example can be perceived in form of a dense respectively sparse acoustic texture. This linkage hereby is not computed explicitly, furthermore it results from the human cognition.

9.5.2. Related Work

TDS is closely related to Model-Based Sonification [HR99]. Conceptually closest to TDS is the data sonogram model [Her02] that implements a virtual mass-spring system for each data item. Their excitations are generated by local impacts into the model space. They cause spherical shock waves to run through the model space and excite the mass-spring systems. Their oscillations are turned into acoustic responses from which the user can learn about local densities and other coherence in the data under exploration.

9.5.3. The Sonification Model

According to the Model-Based Sonification approach, we describe the features of TDS' Sonification Model by the following elements:

Setup TDS is based on a spatial model. The model space, i.e. the virtual representation in which the data under exploration is located, can be represented by a manifold $\mathcal{V} \subseteq \mathbb{R}^3$ in which the set of objects

$$\mathcal{O} = \{o_i = (\mathbf{v}_i, w_i)^\tau \mid i = 1 \dots n\} \tag{9.4}$$

reside. Every o_i has a specified location

$$\mathbf{v}_i = (o_{i[1]}, o_{i[2]}, o_{i[3]})^\tau \in \mathcal{V} \tag{9.5}$$

and a weight $w_i \in \mathbb{R}$. The size n of the set and the object's characteristics are determined by a given data set

$$\mathbf{X} = \{\mathbf{x}_i \mid i = 1, \dots, n\} \tag{9.6}$$

and a pre-processing function

$$f : \mathbf{X} \to \mathcal{V} \times \mathbb{R} \tag{9.7}$$

The mapping from $\mathbf{X}$ to model space is then achieved by applying the mapping function f to each $\mathbf{x}$:

$$\forall\ \mathbf{x}_i \in \mathbf{X} :\ o_i = f(\mathbf{x}_i) \tag{9.8}$$

For example f maps a three-dimensional data set with data items out of two classes A, B

$$\mathbf{X} = \left\{\mathbf{x} = (x_1, x_2, x_3, x_l) \in \left[\mathbb{R}^3 \times \{A, B\}\right]\right\} \tag{9.9}$$

to

$$\begin{aligned} f \mapsto f(\mathbf{x}) &= (x_1, x_2, x_3, w)^\tau \text{ , with} \\ w &= \begin{cases} 0 & \text{, if } x_l = A \\ 1 & \text{, if } x_l = B \end{cases} \end{aligned} \tag{9.10}$$

In addition to the data objects, another special object T resides in the model space. It consists of the vectors in the set

$$\mathcal{T} = \{\mathbf{v} | T_\theta(\mathbf{v}) = 0\} \subseteq \mathbb{R}^3 \tag{9.11}$$

with $T_\theta : \mathbb{R}^3 \to \mathbb{R}$ test function and θ meta-parameters. It could be for example a plane T_p with

$$\begin{aligned} \mathcal{T}_\theta(\mathbf{v}) &= \mathbf{v}_t - (\hat{\mathbf{n}} \times \mathbf{v}) \\ \theta &= \{\mathbf{v}_t, \hat{\mathbf{n}}\} \end{aligned} \tag{9.12}$$

Dynamics By manipulating the physical object, the user is able to adjust the parameter set θ of T. Especially its position, orientation and size are of interest. Any intersection of T and o_i will cause a damped oscillation of the o_i depending on their weight.

Initial State All o_i's are in a state of equilibrium and do not produce any sound.

Excitation and Interactive Types The user is able to adjust the given parameters θ of T. This is done by a Tangible User Interface Object, which forces a direct interaction of the user with the system as motivated. Since the intersection-caused sound of the o_i's is damped, after a while TDS will again end in a state of equilibrium.

Model–Sound Linking There are at least two possibilities two describe the sound generation TDS. Both are based upon the collision of tool T and data objects o_i.

(a) The first approach expects the o_i's to be fixed in model space. The tool then is excited by each collision with a data item.

(b) The other point of view is to relate a masses that are connected to each object o_i by a spring. When a collision of o_i and T appears, the connected mass is deviated from its origin. Its return into equilibrium is then an audible process.

Since both model approaches are equal in their spatial output, because the produced sound is located at the same point in space and depends on both interaction partners. For that reason it is possible to use the one which allows the simpler explanation of a specific issue.

Listener The model aims at spatially surrounding the listener with object-caused impact sounds propagated to him directly from the intersection positions. To achieve this, a *virtual listener*, is introduced into the model space and characterised by the head location v_l and its orientation. As a basic choice, the listener is located in the origin of the model space with the ears aligned with the first axis.

Sound Synthesis In order to stay as close as possible to the model description, a physically inspired damped oscillator would have to be implemented for each possible intersection point. This directly conflicts with the fact that TDS unfolds its strength particularly when exploring data sets containing at least 150 or more data items, which is computationally unfeasible to be rendered by physical sound synthesis algorithms. The current implementation therefore is a compromise between computational load and being physically correctness regarding the sounds. It will be described in more detail in the next section.

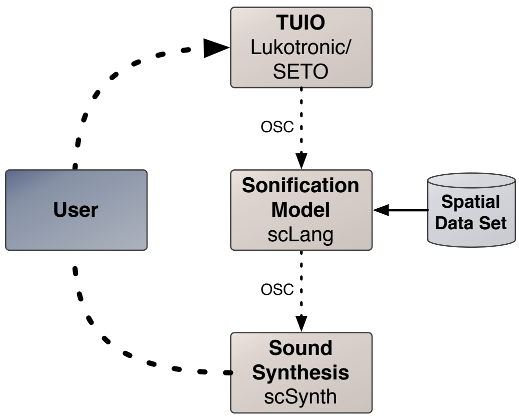
Data–Model Assignment As described above, every object o_i in model-space corresponds to a data item $\mathbf{x}_i$ by applying the transfer function $f(\mathbf{x})$ to it.

9.5.4. Technology

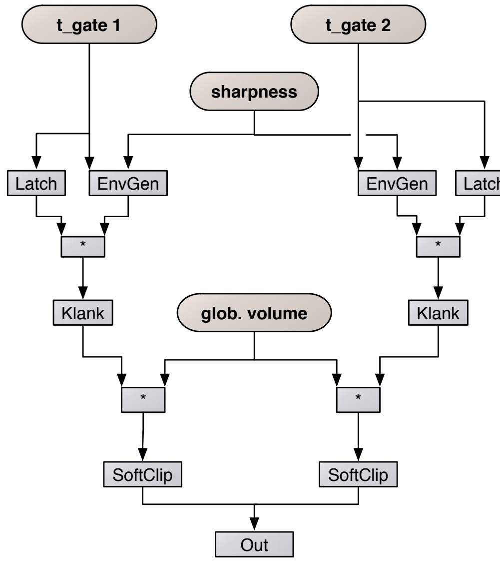
Hardware

TDS was realised in two derivations; one for three-dimensional data exploration in a room, the other for surface-based exploration on the tDesk (see Section 5.5.1). In the room-based setup, TDS were implemented with the help of a *Lukotronic motion capturing system*[12] for object motion capturing and headphones for sound rendering. It allowed to track the tool in six degrees of freedom at an update rate upto 100Hz. The other implementation utilised the tDesk surface and its computer vision capabilities. In this setup, a cube attached with a fiducial marker was tracked for its position and orientation. The measurements were attached to the virtual object in model space. The sound rendering was done with a stereo loudspeaker setup.

[12] `http://www.lukotronic.com`



(a) The implementation modules.



(b) Schema of the used synthesis engine.

Figure 9.38.: TDS schemata.

Software

TDS was implemented in the form of class extensions for the SuperCollider language [McC02], making use of *SonEnvir*[13], the just in time library *JITLib*[14], and other self-developed software building blocks. As shown in Figure 9.38(a), the system is divided into three parts, each running in a separated process. This ensures a de-coupling of sensor capturing, model computation, and sound synthesis, which prevents the system to be bound to the update rate of the sensor readings.

As described above, the user navigates the virtual object by a tangible object. When loading the data set into the model, it is scaled to the interval $[-1, 1]$ in all dimensions. After this, the Sonification Model is computed from the given data and the tracking data of the physical object. We exemplify this at hand of a computation for an exploration plane and two (virtual) pickups in model space that are located left and right to the user. Their virtual recording of the data excitations are rendered to the headphones respectively the stereo loudspeaker setup.

Let O be the basis of the model space, and P_t be the basis of the exploration tool at time t. Then ${}^{O}\mathbf{T}_{P_t}$ defines the homogeneous transform from O to P_t. For each time step Δt, the following algorithm has to be evaluated:

1. Get the current position of the TUIO and compute the homogeneous transformation ${}^{P_t}\mathbf{T}_O$
2. $\forall o_i$: compute its positions $\hat{o}_i^{(t)} = {}^{P_t}\mathbf{T}_O \; o_i$ with respect to P_t.

[13] `http://sonenvir.at`
[14] `http://swiki.hfbk-hamburg.de:8888/MusicTechnology/566`

3. Get the set of indices that intersect the plane in the time interval Δt:
$$\mathcal{I}_t = \left\{ i \,\middle|\, \operatorname{sgn}\left(o_{i[3]}^{(t)}\right) \neq \operatorname{sgn}\left(o_{i[3]}^{(t-\Delta t)}\right) \right\} \tag{9.13}$$
where $\operatorname{sgn} : \mathbb{R} \to \mathbb{R}$ is the signum function.
4. $\forall i \in \mathcal{I}_t$: compute true onset time $t + \Delta_i t$, with
$$\Delta_i t = \Delta t \cdot \left\| o_{i[3]}^{(t-\Delta t)} \right\|_2 \tag{9.14}$$
5. Get the amplitudes for the virtual microphones by using the $o_{i[1]}$ co-ordinate of the original data object.
6. Trigger all events at pre-computed time $t + \Delta_i t$ with its amplitude.

Sound Synthesis

As mentioned above, the sound design of TDS is constrained by two major aspects; first, the possibly high number of data items and therefore high computational load in the Sonification Model, and second, the goal to stay as close as possible to the sound of vibrating objects. Unfortunately, it is unavoidable to test for each data object o_i, whether an intersection with T takes place. This necessarily includes a matrix multiplication for every o_i. The computation of plane-object intersection and additionally its resulting physical sound therefore would be much too expensive for a significant number of data items. We therefore chose a computationally cheaper sound synthesis algorithm that still renders complex sound events: By adding virtual pick-up microphones at specific places into the space and directly rendering its input, we abstracted from the *one sound object per data impact* setup to the *one sound object per virtual pick-up* approach.[15] Each virtual pick-up is represented by a damped resonator bank (**Klank**). They are excited by trigger signals with an amplitude that corresponds to the location of the data impact.

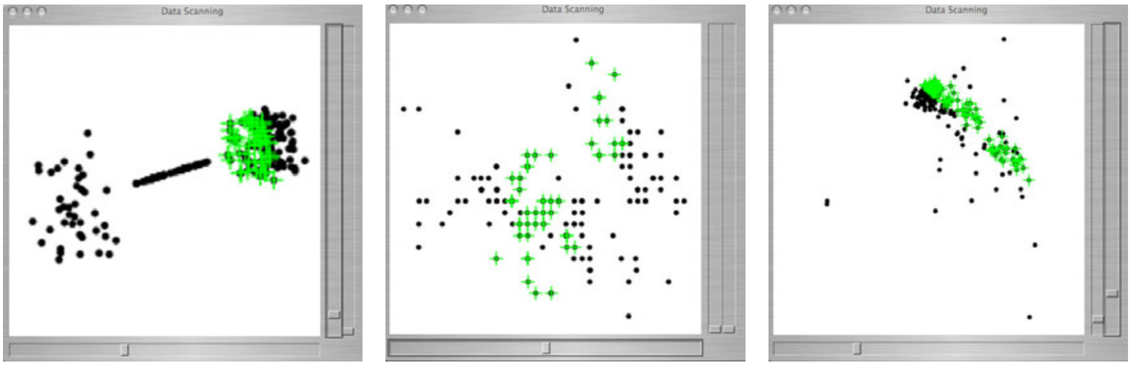
```
SynthDef(\reson, {|out=0, sharpness = 0.0001, freq = 0.2, globalAmp = 1|
	var t_gate, klank, harm, amp, ring, numReson = 5;

	t_gate = TrigControl.names([\t_gate]).kr([0,0]);
	harm = Control.names([\harm]).kr(1!numReson);
	amp = Control.names([\amp]).kr(1!numReson));
	ring = Control.names([\ring]).kr(1!numReson));

	klank = DynKlank.ar(
		`[harm,amp,ring],
		EnvGen.ar(
			Env.perc(sharpness, 2*sharpness),
			gate: t_gate ,
			levelScale: 0.1 * Latch.ar(t_gate, t_gate)
		),
		freq
	);
	klank = (klank * globalAmp).softclip;
	Out.ar(0, klank);
});
```

A stereo version of this setup is visualised in Figure 9.38(b).

Workload

The inter process communication between the three software parts is implemented in

[15] See Figure 9.37(b) for positioning in a stereo setup.



(a) Synthetic data set. (b) Iris data set. (c) Glass data set.

Figure 9.39.: Examples for data exploration with TDS. The green data objects are excited by moving the plane $\mathcal{T}$.

OSC [WF97] [WFM03]. This implies that the resulting system can be distributed over three computers to balance the workload in case of large data sets. Also, the sound rendering can be easily separated into several processes where each then would be responsible for one output audio channel.

As an extension to the currently implemented system, it is also possible to add *performance scaling* abilities [BHR05]. By computing a tool intersection for only a random subset of data items in each time step, the computational load could be decreased, whereas the relative information e.g. about local density is preserved. Detecting outliers or other singularities in the data, though, would not be possible anymore, since an impact of a data object would not strictly cause a sound.

9.5.5. Usage Examples

As benchmark data sets, we used a synthetically rendered 3D distribution and the measured data sets *MCI glass* and *iris*.[16] The exploration tool was a plane connected to the tangible object such that its normal vector pointed right out of the palm of the user's hand. The sound examples for the following description as well as a video of TDS in action can be found on the DVD.

Synthetic Data

For the qualitative evaluation we have used a synthetic data set, consisting of 3 clusters in series. One cluster is sparsely filled in three dimensions, the second is one-dimensional and the third has a high density in all three dimensions. A visualisation of the virtual data-tool interaction in the synthetic data set is shown in Figure 9.39(b).

When moving the plane along the third axis, in which all data clusters are lined up, the cluster borders as well as the dimensionality of each cluster is nicely separated by silence. The therefore can find class boundaries by moving the plane until it reaches a location at which no sound is produced.

The local density of the data set can be judged by scanning different regions of it. Regions with high local density produce a dense sonic grain cloud, whereas sparse regions are

[16] They both can be found at `http://www.ics.uci.edu/~mlearn/`.

rendered to more sparse clouds. The dimensionality of the cluster can be determined by the spatial spreading of the soundscape.[17]

Glass Data

Since the MCI glass data set is 9-dimensional, while TDS, in its current implementation, is only capable to represent three-dimensional data, we chose to explore the projection of the glass data set onto its first three principal components [Jol86] rather than using three arbitrarily chosen axes. This strategy preserves the maximal data variability into the three axes. As shown in Figure 9.39(c), TDS enables the user to explore the different densities of the data. In this particular setup it showed its strength especially in the outlier detection; they can be nicely separated from region with more data items.

Iris Data

The iris data set consists of three classes. Class (A) is linear separable from the others, whereas (B) and (C) mix into a lengthy cluster. By using the plane tool, we found out that class (A) can be separated easily from class (B) and (C). It was located in the upper front of the model space. A clear separation of (B) and (C) is not possible with the plane tool. Figure 9.39(b) shows a visual feedback of a user manipulation of this data set.

9.5.6. Conclusion

Tangible Data Scanning uses the spatial qualities of sound for an unobtrusive augmentation of digital data into physical space. The user is immersed into data sets that can be excited by an object that has a representation in both virtual and physical reality. TDS utilises sound to mediate focused data, whereas data items that are out of the user's focus remain hidden: they do not produce any sound until actively triggered by the object. At the same time, the physical augmentation of data into reality allows the user to create a spatial correspondence between his real environment and characteristic aspects of the data under exploration.

Lessons learned

As described in Section 9.5.5, it turned out that participants developed a strong associative relation between the explored data set and the room in which they used the system. Virtual data points get physically associable with real objects. The current design, particularly the chosen headphone-based approach, though, is more VR than AR.

On a technical level, we found out that the introduced sound rendering approach is effective for extensive spatial grain triggering, and can be extended easily for a more augmenting approach by rendering the sonic grains to a spatial loudspeakers setup rather than to headphones.

[17] At the moment it is necessary that the change in dimensionality is only in the first ordinate $o_{i[1]}$ because the current implementation of TDS is stereophonic. This constraint can be fixed by implementing the system for a spatial loudspeaker setup.

Figure 9.40.: Jonas Groten practising with JugglingSounds.

9.6. JugglingSounds

This chapter describes *JugglingSounds*, a system for real-time auditory monitoring of artistic juggling and swinging. It was designed and developed during a research stay in 2006 at IEM/KUG, Graz for the SonEnvir project in co-operation with Jonas Groten. A description of this system first appeared in the *Proceedings of the 2nd International Workshop on Interactive Sonification* [BGdCE07].

Juggling is a complex artistic activity; repeatedly throwing and catching of several (possibly different) objects in an aesthetic manner is quite difficult. Like any other art discipline that involves physical skills, a juggler needs to develop automatisms for movements. This allows her to spend less effort on monitoring single throws; she therefore has more cognitive capacities to focus on the flow of whole juggling patterns and their transitions. In juggling, progressing towards technical perfection therefore literally creates headroom for reflection on the articulated artistic statement. Especially rehearsal situations in which the actor trains herself in new patterns and moves requires to actively monitor the clubs' movement with several sensory modalities. The feedback is usually mediated via visual and haptic cues: Sight provides information on the clubs' state during the flight time, whereas haptic feedback can be used when the clubs are thrown and caught. Apart from these, there is also an auditory cue that offers an – albeit rather subliminal – feedback about the identity of the single clubs. The difference in the sounds originates in the variations of the club production process, which results in a unique set of resonances for each club. Catching the clubs during juggling makes them sound. This information, however is not of much interest to the juggler. We therefore assume the juggler's auditory cue to be the by far least used in training and performance situations. This makes it ideally suited to be used as a modality that can be augmented with additional information for monitoring dynamics and motions of juggling. JugglingSounds utilises this circumstance to provide auditory represented information to the juggler and the audience. The augmented information can be used for:

Training It can help jugglers to improve their juggling skills by increasing their awareness for details in their movements and the clubs' motions. For example monitoring overall precision or hand-to-hand symmetry can be achieved.

Science It can help to unveil the nature of juggling patterns for scientific, kinesiological research.

Support for the visually impaired It can be used to mediate juggling to visual impaired people (whether as the audience or being the artists).

Aesthetics JugglingSounds can be used as an aesthetic element of the artistic performance on stage.

9.6.1. Related Work

Many approaches for realtime monitoring by Sonification of data streams have been developed: While some of them use semantic-driven approaches where specific knowledge about the data is used to compute rather complex features [HBSR06], others tend to use simple, more arbitrary mappings to popular soundscapes, often as an amusement for the audience at public places [WGO+06]. Rather simple and direct mappings in a scientific context where introduced in the Sonification of human arm swinging, which uses vocal sounds [KWB06], or the EMG Sonifications as presented in [HP06]. Also, Hermann et al. developed a realtime monitoring of a virtual ball to be caught interactively [HHR06].

9.6.2. Design Decisions

Juggling

Juggling in general can be described as the *art of throwing and catching objects*. Against the common sense, it is not only a circus and performance art, but borrows aspects of dance, game, sports and even meditation. The way the juggler is throwing the juggling objects completely determines its motion in air-time, i.e. their trajectories and rotations simply follow the laws of gravity and inertia in free falling. When we look on the ratio of the time the objects held in the hand versus the time they are in the air we encounter something approximately like five parts airtime vs. two times in the hands.

Swinging

In swinging, only two objects (usually *Clubs* or *Pois*) are used. They more or less stay connected to the hand of the artist. Juggling and swinging, however can‘t be separated that strictly, since swinging moves are used in juggling with the clubs in the hand as well as throws are used in swinging routines. However, swinging movements are normally closer to dance movements; the requisites can be influenced at any time since they have always contact to the juggler.

Focus of the system

JugglingSounds was designed to represented aspects of the clubs' motion to the juggler and the audience in realtime. The sounds where rendered by a large audio loudspeaker set-up around the artist. This made it possible for all people in the room to hear the sonic monitoring at the same time as the performance took place.

We focused our work on club *juggling* and *swinging*, a technique where the artist has one club in each hand and moves them in a variety of patterns. Observation showed that these two techniques, though based on the same material, differ in two ways, the club's motion and which kind of information is interesting for the artist. While for example during juggling, the clubs are most of the time in the air and therefore have a parabola trajectory, in swinging, the actor is much more in control of the clubs, and therefore need more information on their actual trajectories. Because of this circumstance, we designed the Sonifications to be adjustable to the situation. Parameters like the length of sonic decay or the overall frequency range where adjusted to the artist's need. However, we had to limit the number of available clubs to three because the room in which JugglingSounds was developed and performed had only a height of about 5 meters.

9.6.3. Observations

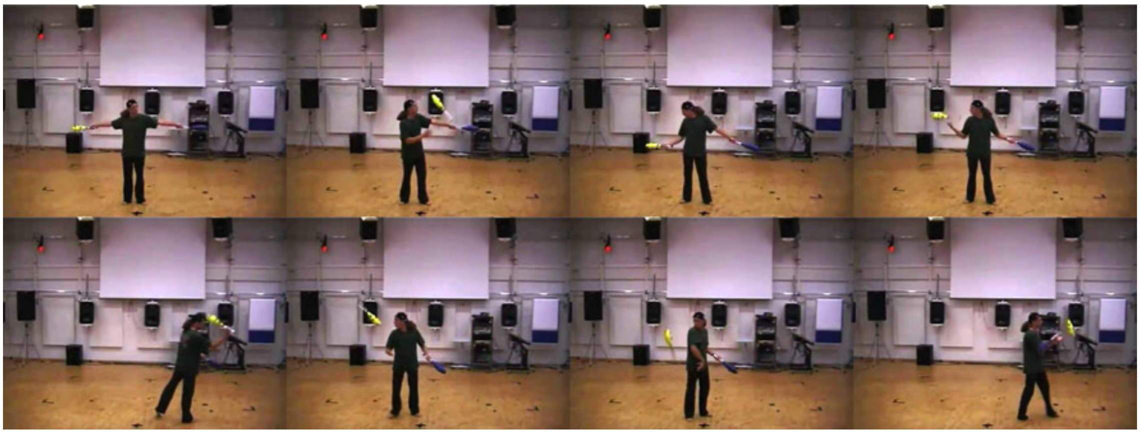
Although a natural feedback loop by haptics and vision is established while juggling and swinging, some of the movements remain difficult to practice. For example monitoring ambidextrous symmetry is of high interest in swinging patterns as their aesthetic impression drastically depend on exact symmetry in movements. Sometimes, the movement to observe cannot be visually observed by the artist. The swinging pattern *synchronous forward hand circles* [Jil94] gives a good example for this: During this trick, it is impossible for the artist to experience, if the clubs are moving correctly in phase. Concerning aesthetics, though, this is a very important aspect. A usual practice is to cast shadows of a light placed on the the side to a wall and look at them while practising. This method, however, requires that the artist deviates her posture away from the aesthetically optimum to see that shadow. Improving one aesthetic aspect therefore requires to give up another one. Another common technique to improve ambidextrous symmetry and precision swinging patterns, respectively throw time and height is video analysis. Unfortunately, this method only provides information *after* the performance, since it is impossible for the artist to anticipate additional visually presented information while juggling.

9.6.4. Systematic for Realtime Display Types

Approaches to realtime monitoring of motions may be found between the extremes of (a) strict *full analysis, then displaying the results* (referred to as qualitative display) and (b) displaying raw data in simple forms (referred to as quantitative display). While detailed analysis provides an appropriate view on already known features, by definition it does not allow to find unexpected or even unknown patterns or structures. Data analysis always requires one to know what to search for. Additionally, analysis heavily relies on the quality of its models used to determine the known patterns. Resulting exploration systems often use relatively simple displays with predefined sets of qualities; in Sonification this often leads to auditory icons, mapping arbitrary sounds (in the sense that their sounds are not directly data-driven) to events triggered by the analysis system.

In contrast, a direct mapping of given features – concerning juggling this would be the position, orientation or velocity of the clubs – provides a direct feedback. Analysis of the displayed data is shifted from machine-powered analysis to the pattern-recognition abilities of the human listener, who may or may not find structural information like the ones described in the full analysis approach, but also is able to unveil new, otherwise not found relationships and structures. Key factors in designing this type of exploration system is the decision for (a) the mapping between data-dimensions and Sonification parameters and (b) the used sounds.

During development of JugglingSounds we found that a direct mapping is necessary to



(a) Swinging with plane trigger Sonification.

(b) Swinging with plane trigger Sonification; Synchronicity.

(c) Exploring four regions with different Sonification approaches.

Figure 9.41.: Video stills of a JugglingSounds performance 2007 in Graz. The corresponding video is part of the accompanying DVD.

get reasonable information on the juggling process. Especially the realtime constraints of JugglingSounds limit the possibilities, since a proper analysis would have been too expensive by means of computational power. Nevertheless, we noticed that a simple mapping of the low-level streams to sound results almost always in an uninteresting and sonically overloaded soundscape where important parts are difficult to separate from unimportant parts. We argue that this is due to the fact that the motions of the clubs are deterministic and regular most of the time: their airtime trajectory can be fully described by the gravity under Newton's three laws of motion. By combining the data streams with relatively low-level events computed from the data, we managed this difficulty in a reasonable way. Figure 9.42(a) shows a schematic diagram of this approach.

9.6.5. Implications for JugglingSounds

The observations made in Section 9.6.3 together with the described systematic in Section 9.6.3 led to the following implications regarding the development of JugglingSounds.

Realtime First of all, the system should be able to represent captured juggling data in realtime such that specific events of interest as well as the overall continuous flow can be monitored and explored by the juggler and the audience. To serve its purpose to support the juggling performance, the design of the Sonifications should also feature an explicitly designed aesthetic quality. To fulfil these requirements, we designed several Sonifications, each with a dedicated sound design, forcing the audience and artists to focus on a specific set of juggling aspects and expressing a dedicated sound aesthetic. Although all set-ups mapped the acquired low-level information to sound parameters in a direct fashion, each Sonification approach aims to emphasise other aspects of the juggling procedure.

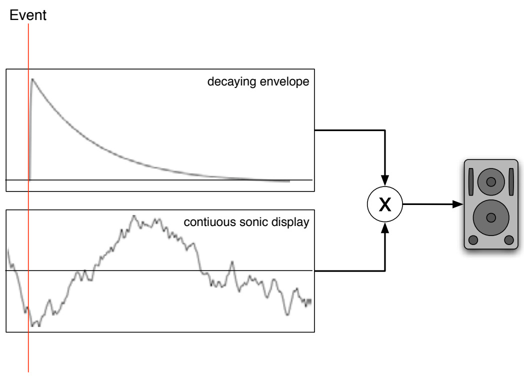
Spatiality To ensure a correct recognition and assignment of sounds to clubs, we decided to use a spherical Auditory Display. [fig: spherical Auditory Display] Each club is represented by a phantom sound source coming from the direction the club is pointing to with respect to the jugglers head.

Directness In order to cover as much of the available information as is needed, JugglingSounds used a mixture of direct mappings of low-level feature-streams and detected events for sound synthesis.

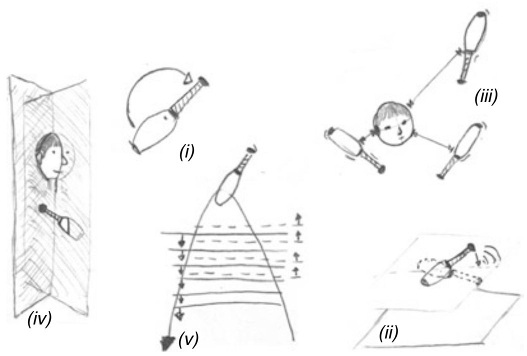
9.6.6. Setup

The juggler interacts with the system by throwing juggling-clubs. To ensure that the used real-time Sonification fully covers the club's motion, while not cluttering the soundscape with unnecessary sounds, JugglingSounds combines a direct mapping of low-level feature-streams with events of interest for controlling the sound synthesis. This way, the used features can roughly be grouped into discrete events and pseudo-continuos signals. With an update rate at about 120Hz and a latency below human perception,[18] they were perceived as realtime streams respectively immediate events. JugglingSounds allowed to display the clubs' motion by one of various Sonification styles. Each of these approaches aimed to emphasise different parts of the juggling procedure. Though very different in appearance,

[18] We did not measure this, yet the realtime Sonification felt very comfortable and direct. Also Jonas Groten as a juggling expert did not mention any latency-related artefacts.



(a) The Sonification strategy used in JugglingSounds. See main text for details.



(b) Trajectory features used for JugglingSounds.

Figure 9.42.: Sonification and feature extraction strategy.

they all had in common that they made use of direct mapping strategies to bridge the gap between the acquired motion data and the corresponding sound.

As pseudo-continuous features we computed and used (a) the rotation velocity around each club's flipping axis, (b) the distance of each juggling club to the juggler's head, (c) each club's position with respect to the room in world coordinates, (d) each club's position with respect to the juggler's head, and (e) each club's position with respect to the juggler head's position and orientation (floor level).

As shown in Figure 9.42(b) (iv) and (v), discrete information was translated into a trigger when a club crosses the coronal plane (behind/in front), or the lateral plane of the juggler's head (iv), or a club crosses one of six specified horizontal planes (v).

9.6.7. Sound Design Considerations

We aimed for clarity and timing sparsity of auditory components for both, concise monitoring and artistic purposes. We prevented for example a direct mapping of a club's rotation angle onto the frequency of a continuous tone, since this would have covered the complete timing spectrum, and is hard to locate. Instead, we mapped the rotation onto the frequency parameter of a trainlet synthesis process. This creates an effect that can be described best as bicycle spokes; there is still space between them, possibly used for other sounds, e.g. originating from the other clubs. Additionally, this sound design implicitly preserves a natural zero in sound, since no club rotation results in silence.

9.6.8. Sonification Design

We created five different Sonifications, each focusing on different aspects of the juggling performance. They are

Rotational Grain Train While the rotation speed of the clubs determine the frequency of a grain train, each grain's pitch is directly coupled to the height of the clubs. This emphasises possible symmetries in the juggler's motion: Similar rotation speeds create similar grain rates, and similar heights of the clubs result in similar pitch maxima in the respective streams.

```
~rotater = {|amp = 1|
    var rotVel;
    rotVel = ~rotVel.kr;
    BPF.ar(
        Impulse.ar((rotVel>0.5)*rotVel*5).lag(0.0001),
        (~height.kr * 120 + 36 + [[0, 7], [0, 12], [0, 16]]).midicps,
        0.2
    ).collect({ |pair| (pair * [1, 0.125]).sum })     * 6 * amp
};
```

Rotation Trigger Every full rotation cycle of a club triggers a sound, whose resonant pitch is determined by its distance to the ground. Note that adjusting the decay of the grain implicitly hides more or less information on the club's change in height. Since the sound is triggered when the club's rotational axis is at a specific angle (e.g. parallel to the floor), the timing pattern of identical angles for the different clubs is audible, and the juggler can get a clear impression of the throwing accuracy.

```
~planeTicker = {|saw2sin = 1, filterFreq = 2000, fSpread = 0.4, amp = 0.1|
    var src;
    var freq;

    freq = 3000*fSpread * (~height.kr*4-1).range(0.5, 2);
    src = SelectX.ar(
        saw2sin,
        [LFSaw.ar(freq) , SinOsc.ar(freq)]
    )
      * 0.1
      * Decay2.ar(
        Trig1.ar(~zeroCrossing.ar, 0.001) * 0.1,
        0.001,
        0.3
    );
    LPF.ar(src, filterFreq, mul: amp)
};
```

Distances to the Head This Sonification captures and mediates much of the inherent dynamics in juggling. Each juggling pattern creates its own characteristic sound pattern.

```
~distances = {|amp = 0.2795|
    LFSaw.ar(min((~dist.kr*2.5 * 90 + 20).midicps, 44100)) * amp
};
```

Left-right Trigger Each crossing of a club through the lateral plane triggers a sound whose pitch is directly coupled to the club's height above the ground and differs depending on its position in front of or behind the head.

```
~backCross = {|amp = 1|
    var trig;
    var numObj = 3;
    var in = ~isLeft.kr(numObj); // compute trigger for change of side
    var height = ~height.kr(numObj);
    var front = ~isFront.kr(numObj);
    var aEnv, fEnv;
    var noise, aEnvNoise;

    trig = Trig1.ar((in - Delay1.kr(in)).abs - 1, 0.00001) > 0.5;
    // only trigger if behind the body and near ground
    trig = trig * (front < 0) * (height < 0.26);

    aEnv = EnvGen.kr(Env.perc(0.05, 2), gate: trig) * 0.1;
    aEnvNoise = EnvGen.kr(Env.perc(0.01, 0.1), gate: trig);
    fEnv = EnvGen.kr(
         Env.perc(0.01, 0.1), gate: trig, levelScale: 900, levelBias: 50);

    noise = WhiteNoise.ar;

    aEnv * (0.2*noise*aEnvNoise +
         SinOsc.ar(fEnv * (height *8).squared * 0.4, 0, 1.5).softclip
       ) * amp
};
```

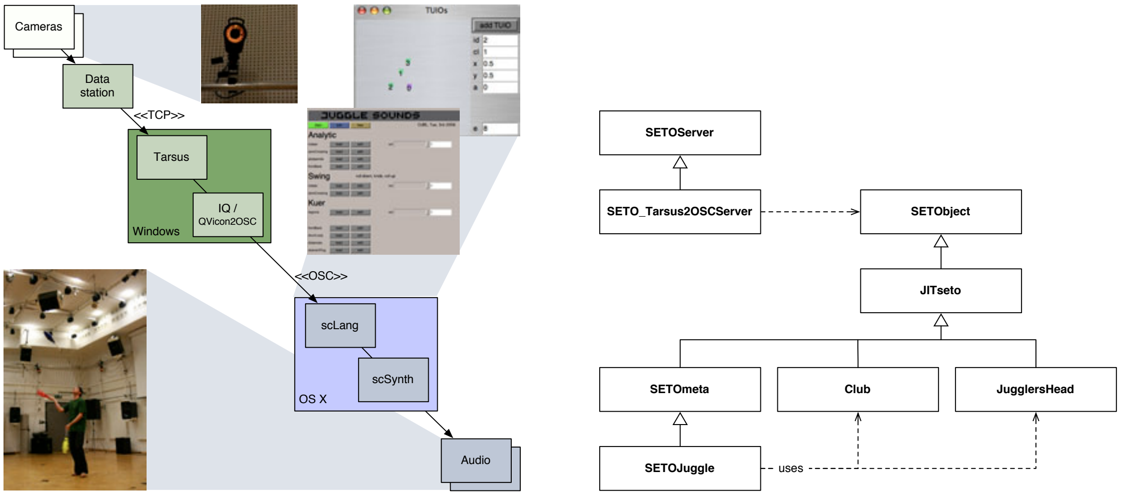
Rain on Bells We designed a discrete level indicator by placing several virtual horizontal planes in the air at equidistant heights, and linking each one to a differently pitched sound. Each crossing of a club results in a small sound grain which is different on the way up and down.

```
~clackUp = { |amp = 0.1|
    Formlet.ar(
        ~trigsUp.ar.lag(0.0004),
        (~height.kr).exprange(25, 2500).collect(_ * [1, 2.03, 4.1])
             .cpsmidi.round(2).midicps,
        0.0001,
        LinExp.kr(~rotVel.kr[i].abs, 0, 15, 0.02, 0.2).max(0.01)
    ).sum
    * (~height.kr*4 ** 2) * amp
}
```

9.6.9. Sonification Designs for Swinging

Rotation In this context, essentially the same mapping as in the corresponding juggle Sonification enables the artist to experience the amount of synchronicity in motion as well as the differences in height of the triggering points.

Rotation Trigger Especially tricks like counter-rotating clubs in front of the body or a 1-5-Circle may be monitored concerning their accuracy in execution for training purposes.



(a) Data flow.

(b) UML dependency diagram of JugglingSounds classes.

Figure 9.43.: Components of JugglingSounds.

To get an insight into the above described Sonification approaches consult the example videos (see Figure 9.6.2 for stills of the videos).

As understanding all these Sonification designs requires seeing the juggling performance and hearing the described Sonifications, please consult the example videos provided on the DVD. They give an insight into the described Sonification approaches.

9.6.10. Technical Aspects

For club tracking, we prepared 3 regular juggling clubs with several reflective markers. The actual data acquisition, i.e. gathering information on position and orientation of the three clubs and the jugglers head was done by a VICON motion tracking system. Although these systems are tailored towards applications in animation, biomechanics, or engineering and are specialised in full-body motion tracking, they are sufficient to track and distribute data on several rigid bodies in realtime. We used it to track rigid bodies (the juggling clubs and the jugglers head) in six degrees of freedom (6-DOF) on an update rate of approximately 120Hz. The tracking system consisted of 9 cameras, a VICON specific data station a PC running proprietary VICON software. Additionally an Apple Dual G5 Desktop system was integrated into the setup, serving the sound rendering and high-level control of JugglingSounds. All these computer systems were connected to each other over a dedicated ethernet connection. For sound rendering, we used the 24-channel audio setup arranged in a dome-like setup as it was provided by the IEM. All sounds where arranged according to the relative position of the clubs to the head of the juggling artist. This made the sound sources de-clutter; the display got much clearer in both, spatial occupancy of sounds to clubs and overall sound experience. The whole setup fit into half of a sphere with a radius of approximately 5m; enough space for a juggling performance with three clubs.[19]

[19]For more information on the performance space, please refer to `http://www.iem.at`.

Software In software, JugglingSounds consisted of two main parts, data acquisition via motion tracking and Sonification via SETO (as described in Section 10.2). The interconnection between these parts is shown in Figure 9.43(a). The above-described hardware setup was able to compute 6DOF positions of the clubs and the head at about 120Hz. The resulting pseudo-continuous stream of tracking data was translated into OSC messages by *QVicon2OSC*, and sent to the sound server. The sound synthesis as well as feature extraction were implemented in SuperCollider [McC02] [WCC09] using the SETO environment for tangible objects. In Section 10.2 the details of JugglingSounds SETO-related implementation will be described as an example for the usage of SETO in an actual application setup.

9.6.11. Conclusion

With JugglingSounds, we introduced a direct display for realtime monitoring of juggling moves to be used in both rehearsal and performance. By its utilisation of auditory representations for club trajectories, it supports jugglers in the training of timing aspects such as synchronism and rhythmics without influencing the performance by occupying the artist's most needed senses: haptics and vision. The representation of rhythmical patterns by audio supports both human processing and comparison skills for time-based structures. To facilitate this, the system combines direct mappings of continuous data streams with symbolical notion. This were done by the combination of analogue parameters like position, orientation and velocity of the clubs with highes-level triggers like zero-crossings and plane intersections. This combination supports the actor in focusing on the parts he considers as relevant, while still mediating other information for subconscious analysis.

Lessons learned With JugglingSounds, we exemplified that sound is an appropriate modality to mediate information to people who's other modalities are already occupied. As in the other applications that are presented in this thesis, the spatiality of the artificial sound sources that reflect their physical counterparts support the sound-object identification. The requirement of the club-attached sounds to be undifferentiable in their sonic character would otherwise make their separation very difficult. Although it looks complex, three-club juggling turned out to consist of relatively static movements. Due to their long airtime, their appearance does not substantially change over the course of a performance. This observation is reflected by JugglingSounds' sonic feedback. It also turned out that an essential part of a monitoring system for artistic purposes is the minimisation of latency. The linkage between the juggling movements and the rendered sounds otherwise would not be recognised correctly. In this context, Jonas Groten reported that the system sufficiently mediated him his (known) timing problems regarding left-right synchrony. Because of the immediate feedback, JugglingSounds allowed him to practise towards their minimisation.

Outlook One prominent extension to JugglingSounds would be its extension to track more than three clubs. This would add significant value to the system, since four– or five–club juggling adds many additional movements and patterns that are significantly different and more difficult to learn than three-club patterns. Also, adapting the system to use other juggling elements such as beanbags or rings would increase its functionality and would allow also less experienced jugglers to benefit from JugglingSounds.

Figure 9.44.: Video stills from the presentation of a prototype of Durcheinander at Animax, Bonn in late 2007. The corresponding video is part of the accompanying DVD.

9.7. Durcheinander

The system that will be presented in this section was developed in co-operation with Julian Rohrhuber in 2007. It was first published in the *Proceedings of the 14th International Conference on Auditory Display*, and it was presented in a live demonstration at the accompanying conference in Paris, France [BRR08].

With *Durcheinander* we present a system to help understand Agglomerative Clustering processes [ELL01b] as they are used in various, often visual oriented, data mining and exploration systems [Rit00] [Cor06]. Durcheinander consists of several small objects on a tabletop surface, which represent data items in an artificially generated data set. A computer vision system tracks their position and computes a cluster dendrogram, which is sonified every time a substantial change in this dendrogram takes place. Durcheinander may be used to answer questions concerning the behaviour of clustering algorithms under various conditions. We propose its usage as a didactical and explorative platform for single- and multi-user operation.

Agglomerative Clustering is a data mining approach that is mainly used to unveil structural relations in high-dimensional data sets [ELL01a]. It particularly facilitates the discovery of compact clusters of data items in high-dimensional vector spaces. Structures found by Agglomerative Clustering are assembled into a *dendrogram* that recursively interconnects single data items by means of their location (see Figure 9.45 for an example). Although the general behaviour of Agglomerative Clustering with a given set of meta-parameters (which includes the used distance metric) can be easily understood, the parameters' relation to the algorithm's result in a *specific* case is more difficult to grasp. Participants in data mining courses can achieve better understanding by trying to answer the following questions:

1. Under what variations in the data does the Agglomerative Clustering dendrogram

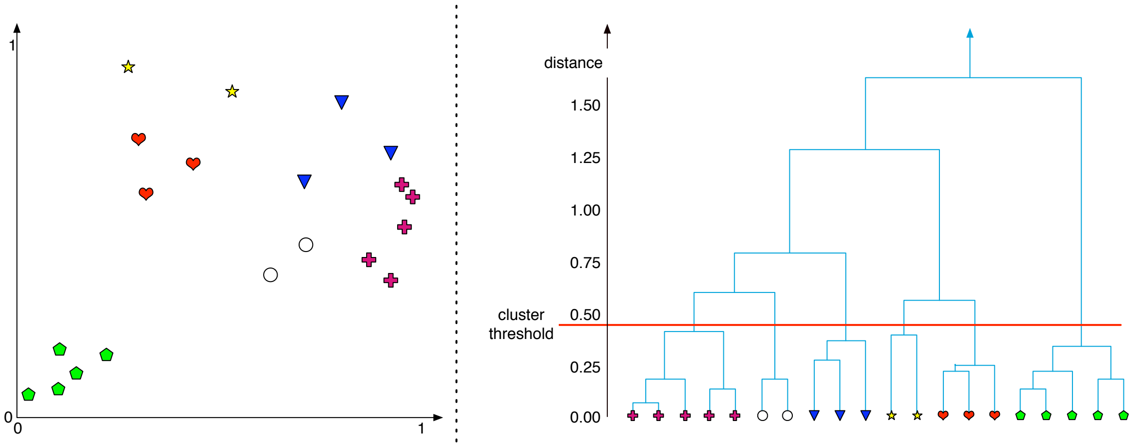


Figure 9.45.: A 2D-plot of a two-dimensional artificial data set and its corresponding dendrogram. The red line indicates a specific clustering that defines the shape of the data items in the scatterplot.

change its configuration and how does it change?

2. What happens when data items are in a special configuration?
3. What are the differences between the various distance metrics?
4. What are the differences between the various cluster metrics?

Durcheinander's purpose is to help answer these questions by means of the TAI paradigm. It provides the opportunity to physically *grasp* the data and, at the same time, allows auditory exploration of the effect of different clustering parameters. Durcheinander's tangible objects are laid out on a table and the sound is delivered in a spatial sound environment. Learners have turned out to particularly benefit from this collaborative multiuser nature of the system; it invites to discuss the results of Agglomerative Clustering *in the process* of co-operative exploration, instead of before and after. Furthermore, its *interactive programming* approach allows researchers to experiment with different Sonification methods during interaction.

Usage scenario

The following usage scenario describes a typical situation in which Durcheinander may be used:

Learners stay around the Durcheinander surface. A teacher configures a specific data set by outlaying it with objects. Now, all try out the stability of that specific layout by moving objects, while explicitly listening to changes in the configuration. After a while of trial and discussions, the teacher changes to a different cluster metric, and lets the students explore the resulting differences in the algorithm's behaviour.

In the interface, the current state of the cluster algorithm is mediated through the Auditory Display. Its behaviour depends on the used cluster metric. The clear separation of input data (objects) and processing layers (auditory part) can be used to explicitly transport the otherwise invisible clustering information and give hints on the importance of the interpretation regarding the cluster algorithm's output.

9.7.1. Agglomerative Clustering

Clustering can help to unveil hidden structures of a specific kind in possibly high-dimensional data sets. It is especially suitable for compact structures in the sense of the used distance metric. Agglomerative Clustering is a special approach for clustering and produces so-called *dendrograms* of inter-cluster distances by application of the following rules [ELL01b]:

1. Initially, all data items x_i are considered to be clusters c_i, so that $\forall x_i \in X : x_i = c_i$
2. Compute distances between all pairs of clusters and find the smallest distance:

$$\text{minpair} = \arg\min_{i \neq j} \; d(c_i, c_j) \tag{9.15}$$

$$\text{mindist} = \min_{i \neq j} \; d(c_i, c_j) \tag{9.16}$$

3. Join c_i and c_j at the distance mindist. This joint $\langle c_i, c_j \rangle$ represents the new cluster c_k
4. Add c_k to the list of clusters, remove c_i, c_j from this list
5. If more than one cluster is in the list of clusters GOTO 2, else END.

A cut at a specific distance in the resulting dendrogram represents one possible clustering of the given data set. For example applying Agglomerative Clustering to the data set shown in Figure 9.45 (a) results in the dendrogram shown in Figure 9.45(b). The red line represents a possible cut.

Although it seems natural to use the standard Euclidean metric to measure object distances, it is also possible to use other metrics which may fit better to the domain of the given data set. The choice of the inter-object metric as well as the choice of how to determine cluster distances heavily affects the structure of the Agglomerative Clustering outcome and therefore the resulting dendrogram. These metrics differentiate Agglomerative Clustering into e.g. single-linkage, complete-linkage, or average distance clustering:

Single Linkage:

$$d(c_i, c_j) = \min_{x \in c_i, y \in c_j} d(x, y) \tag{9.17}$$

Complete Linkage:

$$d(c_i, c_j) = \max_{x \in c_i, y \in c_j} d(x, y) \tag{9.18}$$

Average Distance:

$$d(c_i, c_j) = \text{avg}_{x \in c_i, y \in c_j} \, d(x, y) \tag{9.19}$$

Although it is relatively easy to understand the general global behaviour of the clustering algorithm, it is difficult to understand the way in which local variations such as the exact position of data items affect the algorithm's output. This is particularly interesting since Agglomerative Clustering is usually applied to data that incorporates measuring errors, which cause variations in data item locations.

A dynamically changing structure may not necessarily be best represented in form of a visual dendrogram; Sonification allows us to explore its recursive (re-)configuration without a projection onto the plane of geometry.

9.7.2. Implementation

Hardware

As a basis for Durcheinander, we use the tDesk, a tabletop tangible computing environment designed and built in the interaction laboratory at Bielefeld University (see Section 5.5). By design, the dimensions of the surface allow groups of people to work on tangible applications, providing each member direct access to the physical objects. We use a digital camera below the tDesk to capture the 2D positions of the objects used as the data set in our system. This method prevents possible visual object occlusions by the users such that all 20 objects are all the time recognisable by the vision-engine. A blob recognition algorithm then detects number and position of the objects, which is fed into the actual clustering algorithm which in turn computes the dendrogram.

Clustering

The dendrogram structure is translated into a corresponding sound synthesis graph which may be triggered externally by knocking on the surface of the tDesk. The resulting sound is rendered in real time to the users by the multi-channel audio system surrounding the table. Each physical data item produces a sound that is spatially related to its position on the surface; every object sound again consists of sub-sounds determined by other nodes of the dendrogram.[20] The graph structure is being continually updated, and whenever its configuration differs substantially from its predecessor, the system generates a trigger that propagates through the synthesis graph; a series of reconfigurations can be heard as a series of differing sounds in context.

Sound synthesis

The Sonification algorithm constructs a computation graph in which each node (representing a cluster c_i) takes an n-tuple of streams as input, provided by its enclosing cluster. In addition to this, a variable number of arguments allow parametric control and triggering of each node:

```
{|in, trig, dist, id, lagTime|
    freq = freq.lag(lagTime.max(0.05));
    freq = freq * (3 ** dist);
    [
        in + Decay2.ar(trig, 0.01, 2.5, 0.1) * SinOsc.ar(
            freq,
            SinOsc.kr(Rand(1, 4), 0, 0.05, Rand(0, pi))
        ),
        trig,
        freq
    ];
}
```

Each object is acoustically represented by a node that passes its own frequency response (**freq**) resulting from a trigger (**trig**) on to the next node's input (**in**). For this, each node passes an n-tuple of streams to both of its two adjacent nodes. The algorithm defining this flow-graph can be rewritten conveniently at runtime such that different synthesis techniques can be tested online.

[20] In order to realise such a framework, we implemented a modular sound architecture in SuperCollider, a higher-level programming language that is specially suited for real time sound rendering [McC02].

9.7.3. Conclusion

Durcheinander uses sound as a tool to represent structure and dynamics of Agglomerative Clustering algorithms. It's educational purpose is underlined by the separation of user-controllable data input and auditorily represented clustering results. A change in the cluster configuration triggers sonic events, indicating the momentary hierarchical configuration of the dendrogram on which the clustering process is based on. Durcheinander provides an additional perspective to clustering techniques by focusing on other aspects than the common visual representations. These aspects particularly include the spatial correspondence of clusters and its change under induced noise in data input, respectively under a change of the cluster metrics.

Lessons learned In late 2007, we presented Durcheinander at a workshop for children at the Animax in Bonn.[21] There, we had the chance to extensively work with visitors to adjust Durcheinander's Auditory Display (see Figure 9.44). At ICAD 2008, we presented a live demonstration of Durcheinander. In the light of these presentations, we can report that also inexperienced users tend to grasp the objects and start exploring without any uncertainty. Users tend to forget the technical system and manipulate the sounds directly, having the interface ready-to-hand. We view this as a valuable feature for systems dedicated for exploration and learning. These are the same insights as we experienced for AudioDB (see Section 9.4). However, we also realised that Durcheinander needs further development to be useful for actual didactical purposes. However, its current state clearly proofs that TAIs provide valuable methods for educational applications.

[21] This workshop was part of the DFG-funded research project *Artistic Interactivity in Hybrid Networks* of the German *Jahr der Geisteswissenschaften 2007*.

9.8. Discussion of the Presented Applications

In this chapter, we described various applications incorporating Tangible Auditory Interfaces for human computer interaction. This chapter concludes with an overview and critical discussion of specific aspects of the presented applications. It is structured into paragraphs that cover the main application field of the introduced TAIs.

Spatial Control The purpose of ChopStix and MoveSound is the manipulation of the spatial parameters of sounds. It is their main function to give low entry-level access to these parameters; the source sounds are considered to be either automated or controlled by other interfaces. AudioDB, TDS, JugglingSounds and Durcheinander, on the other side utilise spatiality to support the immersive effect of their inherent Auditory Display.

While the use cases associated with MoveSound require exact spatial control, the other spatialness-incorporating TAIs provide a rather loose control scheme; spatiality here is seen as a way to de-couple otherwise very dense audio streams, respectively as a feature to reflect the user's decision on importance (as in ChopStix). All these applications indicate that spatial control can be reflected fairly well with the TAI paradigm.

Implicit State Display Most of the presented TAIs use implicit state displays. In MoveSound, a visual layer is projected around the physical controller in order to display the system's state. It therefore does not completely reflect its state with tangible elements. This tradeoff is purposefully accepted because it adds the feature to control several sound sources with one physical handle, and it also adds the possibility to implement the playback of spatial parameters of some sound sources while concurrently adjusting others.

In ChopStix, on the other hand, the user controlled system state is completely represented by the physical state of the Tangible Interface. This is essential for its use as a long-term interface; users are subconsciously made aware of the system's current state, and are able to relate the physical configuration to the Auditory Display.

The same holds true for AudioDB, apart from the fact that the association of sounds to objects is not displayed. However, it is considered to be easily remembered in the rather short and intensive usage of the system. A visual assistance, though, is considered to be promising. This could be added for example by visually projecting source IDs or names next to the objects.

Due to its monitoring purpose, JugglingSounds' Auditory Display reflects parts of the motions and club movements. No other information is fed into the system, so there is no need for an additional representation. Similar to JugglingSounds, Durcheinander uses sound to represent the state of the controller objects. Because of its educational purpose, it does, however, add a layer of information about the current clustering of the objects that cannot be experienced by the other senses. The border of reality and algorithmic modelling that should be mediated is located between the modalities.

In Reim and TDS, finally, the tangible state is not of general importance; the users' action is the central element; they keep track of the data-object assignment (Reim), respectively explore the spatial relationships of linked data (TDS).

Data Exploration AudioDB offers a technical system to sort and organise sounds. TDS, on the contrary, places data items into the user's environment and enables him to explore their spatial relationship with a physical tool. The third application designed for data exploration, Reim, provides a very direct sonic representation of data items for everyday use; be it by active exploration and purposeful comparison, or on a subconscious level with an ambient auditory layer added to everyday objects. Although all three applications are exploration tools, they support explorative tasks with fundamentally different approaches. This fact emphasises the diversity in which TAIs can help in data exploration tasks to find evidence for structural or other information in data sets.

Monitoring JugglingSounds is developed to monitor juggling club movements, typically in juggle- respectively swing-training or -performance situations. Its broad range of Sonification styles are designed to give additional information cues about the current performance.

In contrast, the two other monitoring applications, ChopStix and WetterReim (the monitoring application of Reim) are intended to be used for a longer period of time. While ChopStix provides an interface to reflect the user's selection regarding interesting locations in a data-driven spatial soundscape, WetterReim's intention is the provision of a subconscious interface to near-realtime data that augment soundscapes originating from everyday objects in the user's environment.

Collaborative work In principle, all presented applications can be used collaboratively. However, some of them provide a user interface that was especially designed to be used by many people at a time. The tangible parts of AudioDB and Durcheinander for example are located on a horizontal surface that was designed especially for the use with many people. It does not feature a prominent direction from which it should be operated but users can stand around it while manipulating the tangible objects and experience the spatial auditory responses. Other TAIs such as ChopStix operate on a different timing level. Their interface is smaller, so it cannot be used easily by more than one person at a time. It, though, has a spatial display that is controlled by the Tangible Interface on a more long-term level. Considering the larger time-scale combined with the spatial display of four meters that surrounds the interface, ChopStix can also be considered as a multi-user system.

To summarise, all presented TAIs are designed to cope with the demands with respect to their main application area. Together, they form a toolbox of both case and design studies that can be used to orient oneself when developing new TAIs.

10. Software and Hardware Frameworks

10.1. TUImod

This project was done in co-operation with Risto Koiva, Thomas Hermann and Helge Ritter. It was originally presented at the Pervasive Conference in 2008 [BKHR08].

Tangible Interfaces add digital functionality to arbitrary physical objects. Many applications in this field rely on physical objects of various shapes that have to be recognised by a computer system to add algorithmic functionality. TUImod is a modular system of basic elements generated by rapid-prototyping techniques. Its modules can be combined in various ways into human distinguishable and computer trackable physical objects with specific physical properties.

TUImod supports fast prototyping of Tangible User Interfaces by providing a broad range of modules that can be assembled easily into a variety of objects exhibiting different features. The strength of this system lies in its modular structure, allowing a huge number of object designs. TUImod objects combine the following three element types:

UI User Interface Elements determine the object's identity in the user's view;

PF Physical Functionality Elements add physical functionality to the object;

CI Computer Interface Elements determine the object's identity, position and orientation for the computer.

10.1.1. Related Work

Recent publications already focused on design considerations for objects in tangible computing scenarios, though the authors therein focused mostly on a specific task such as controlling a musical interface [BKJ05] or testing new electronic interface technologies [NDNG03]. They all cover custom-build passive objects designed for that exact type of application.

This is contrasted by the *METADesk* system, an example for the integration of *active* objects that are capable to change their position on their own [PI07] . Although this is a promising and future-directed hardware design, its development and actual implementation remains expensive and its usage is potentially prone to errors due to the many electronic and mechanical components used. In contrast, the object design of TUImod consists only of passive objects, yet their feature set allows to build complex applications focusing on *direct interaction* with data structures and algorithmic functionality [Dou01]. Due to its modular design, however, TUImod objects are still open for the integration of such active components in future revisions. In this direction, Riedenklau developed TAO, active objects incorporating the modular approach of TUImods and their footprint, but adding electronic elements like motors LEDs and buttons [Rie09].

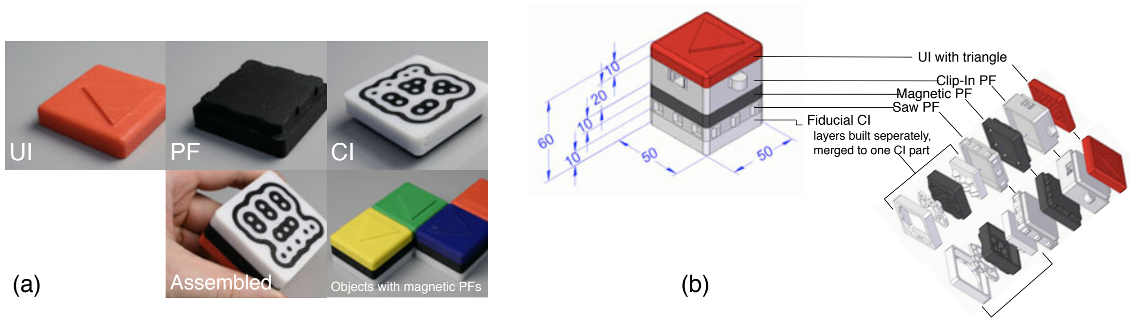


Figure 10.1.: Design and implementation of TUImod elements. (a) TUImod modular design. (b) TUImod object with all *PF* elements covered by example elements for *UI* and *CI*.

10.1.2. Object Design

TUImod objects consist at least of a "sandwich" of two basic elements: a *UI* responsible for the user's experience, and a *CI* for robust recognition by a computer. One or more intermediate layers of *PF* elements can be added, either to change the object height, or to add other physical properties like placement constraints, or magnetic forces. The single elements are designed to be stackable, resulting in both, robust interlocking and effortless disassembly. This allows a swift change of configuration and function of TUImod to be used in diverse applications.

Implemented object designs

We built 20 different *CI* elements, all based on the fiducial marker design, for object identification including position and orientation. We produced *UI* elements in five different colours (white, red, yellow, green, blue), each with four different reliefs (triangle, circle, square and plain surface). See Figure 10.2. The *PF* elements can be inserted between *CI* and *UI* to equip TUImod objects with different interfacing characteristics such as a different height, magnetic forces that allow only specific inter-object placements (Figure 10.2 (a)), clip-in functionality for mechanical object connection (Figure 10.2 (b)), or saw-shaped edges constraining inter-object placements to a discrete set (Figure 10.2 (c)). Additionally, we built two- and six-sided *CI* inter-connectors for state-representing objects as shown in Figure 10.2 (d), respectively (h). As can be seen in Figure 10.2 (f), the objects were designed to comfortably fit into a hand's palm.

Tracking

For the visual object tracking we use the fiducial tracking system [BKJ05]. It allows to simultaneously track 2D positions and orientations of up to 90 different markers at a time from below a glass surface.[1] This tracking system closely connects with TUIO, an OSC-based network interface protocol for tangible objects [KBBC05], which then is processed in our in-house developed framework for Tangible User Interfaces *SETO*.[2] All mentioned software systems are open-source.

Production

For the production of the TUImod elements, we used *Fused Deposition Modeling*, a rapid-prototyping 3D-Printing technology, where material is added in layers. With the *Stratasys Dimension SST 768 RP*-machine using acrylonitrile-butadiene-styrene (*ABS*) material,

[1] Please note that replacing the Fiducial-Tracking system, e.g. by a system based on electronic marker detection would not affect the user's experience.

[2] `http://tuio.lfsaw.de/seto.shtml`

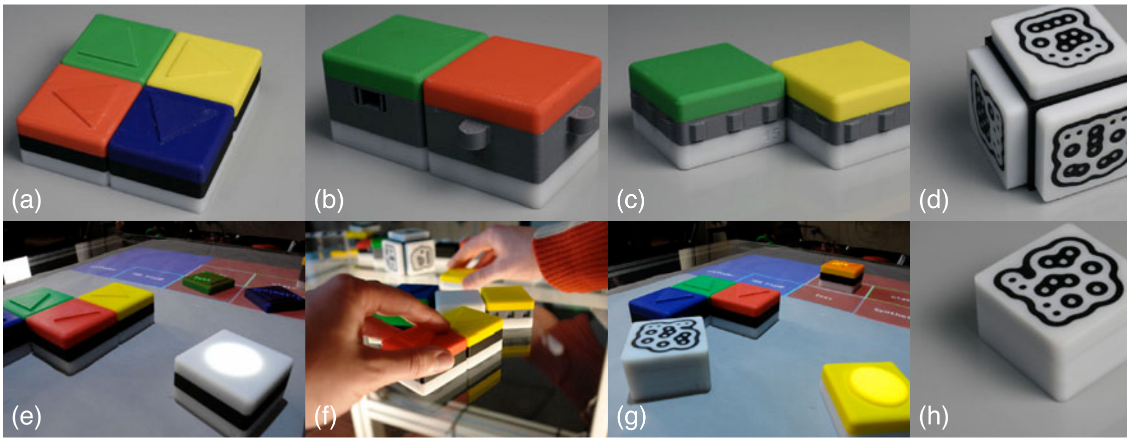


Figure 10.2.: TUImod objects with different *PF* elements. (a) magnetic, (b) clip-in, (c) saw, (d) cube, (h) two-sided. Images (e)–(g) show examples of the use in the tDesk environment with front-projection.

we were able to produce extremely durable objects that can withstand even the roughest handling. In addition, rapid prototyping allowed us for extremely short idea-to-product time-cycles while keeping the absolute costs for whole production relatively minimal. Unfortunately, the used machine allows only to produce single-coloured objects. To get the high contrast required for good visual detection of the *CI* elements, we developed a design with four interlocking parts of different colours (cf. Figure 10.1). The precision of the machine is good enough for the separate layers to be held together without any adhesive by just press fit.[3] All TUImod modules benefit from the ABS-typical crisp colours and elasticity, which is specially beneficial for the clip-in *PF* elements. Although the used material was not a decision of the design, but moreover given by the constraints of the 3D printer, it turned out that it produces a haptic feedback mediating a good quality.

10.1.3. Application

TUImod is part of tDesk [BHR06b], the tabletop system described in Section 5.5. Prior to the development of TUImod, we developed several tangible computing applications for the tDesk environment: E.g. we demonstrated a tangible computing system for the interactive control of real-time multi-channel data Sonification [HBRR07]. Physical objects here serve as graspable representations for signals like they can be derived from EEG channels. As a second system *AmbID*, an ambient interaction system [BHR06b] allows to control display properties of real-time data streams within an ambient multi-modal environment by changing relations between physical objects. Both applications use different object types to mediate their functionality. To distinguish the therein used acrylic cubes, we attached human-readable and computer-perceptible markers printed on paper. With the introduction of TUImod, the outwearing marker design was replaced by the much more robust TUImod objects.

[3] There are other machines for multi-colour 3D-Printing, where a special powder material is fixated in layers. Unfortunately this technology does not produce objects nearly as strong and flexible as those using Fused Deposition Modelling process and ABS material.

Although none of these applications utilise the constraints introduced by the *PF* elements, we regard their explicit usage for application design as highly promising. The TUImod system was also used in TDS (as described in Section 9.5, and in the implementation of the Shepard-Risset effect demonstrated in Section 5.4.2.[4] It's modular design also allows it to be used for systems like Durcheinander (see Section 9.7) or AudioDB (see Section 9.4). However, in these systems we preferred simpler grain-type objects featuring no dedicated orientation, since the orientation does not code any functionality.

A video demonstrating TUImod can be found on the accompanying DVD. All definitions of the TUImod elements are provided for free download.[5].

10.1.4. Conclusion

With TUImod, we introduced a modular, versatile yet extensible design for tangible objects aimed to be used in prototyping environments for tangible computing applications. We described the design and assembly of the elements as well as the resulting objects and their features. Adaptation of previously developed surface-based tangible computing applications is straightforward. The rapid prototyping allows to develop a collection of basic TUImod elements that offer the appropriate physical implementations for most needed functions (e.g. modules with moveable hardware sliders, or even malleable *PF* elements integrating springs). With such a set of building blocks, a wide range of different Tangible Interfaces can be prototyped without the need to develop and produce new hardware. Our current interest is to investigate relationships between physical shapes and auditory gestalts via interactive Sonification. Currently, the system is extended by Riedenklau to be used for active objects.

[4] You can find a video of the implemented example on the DVD.

[5] `http://tuio.lfsaw.de/tuimod.shtml`

10.2. SETO

The SuperCollider Environment for Tangible Objects (SETO) originates in the development of TUIO, the *Tangible User Interface Protocol* [KBBC05] and the need to interface it within SuperCollider [McC02] [WCC09]. Originally developed for surface-based interfaces, it was also used for 6DOF tracking in the implementation of JugglingSounds (see Section 9.6). During its development, SETO was extended to cope with the captured 6DOF movements of objects. While the first ideas of SETO were developed during my research stay as an STSM in Barcelona at the Music Technology Group of the Pompeu Fabra University,[6] I developed other parts at the Neuroinformatics Group of Bielefeld University and as a member of the SonEnvir project at IEM, Graz.[7]

Tangible Interfaces need some attention for their technical realisation. First of all, an object tracking system is needed that provides real-time information about physical object manipulation to the system. The type of tracking software depends on the objects, the frame-rate for pseudo-continuous data and the desired degrees of freedom that should be tracked. Independent from this, however, also the amount of money willing to spend has to be considered. A variety of systems may be taken into account, lasting from cheap, one camera, vision-based systems to many-camera or electronic-based marker tracking systems by far out of the financial scope for private use. A relatively low-cost variant is the open-source software *reacTIVision*,[8] for which a (fast) camera, appropriate lighting, objects to be tracked and a glass surface are required.

TUIO

Apart from the feature to track more or less DOFs, the tracking system does not influence the resulting sounds but the quality of the interaction. Thus the flow experience may drastically differ. This is mostly due to different tracking rates (10Hz vs. 120Hz) and latencies (2ms vs. 50ms). Since the tracking process is done in a separate process (probably even not on the same computer), an interface between the tracking system and SuperCollider is needed. TUIO is a protocol designed just for this purpose, i.e. to transmit tracked physical object states between an object recognition system and the application to be controlled, in our case SuperCollider. Strictly speaking, TUIO is a specification of a fixed set of URL-style commands and method names for OSC focusing particularly on reliability and performance.

Let us consider the following TUIO message:

```
1 /tuio/_ixya "set" 10 1 0.3 0.7 1.2
2 /tuio/_ixya "alive" 10
```

The command string (`ixya`) refers to the set of parameters each object provides. It is, in conjunction with the object's ID (10), the object's unique identifier. For a valid transfer, the object tracker has to send a set-message for each object change (including appearance and vanishing, line 1) and an alive-message containing the IDs of all available objects (line 2). All other messages (source, fsec) are optional. This makes it relatively easy to implement a basic interface for TUIO, especially in SuperCollider, since it has native OSC support. Assuming that the object tracking system sends TUIO messages of the type `ixya`, a patch that modulates the frequency of a corresponding sine wave according to the rotation angle

[6] This research stay was supported by the Cost287-ConGAS `http://www.cost287.org/`.

[7] `http://sonenvir.at`

[8] `http://reactivision.sourceforge.net/`

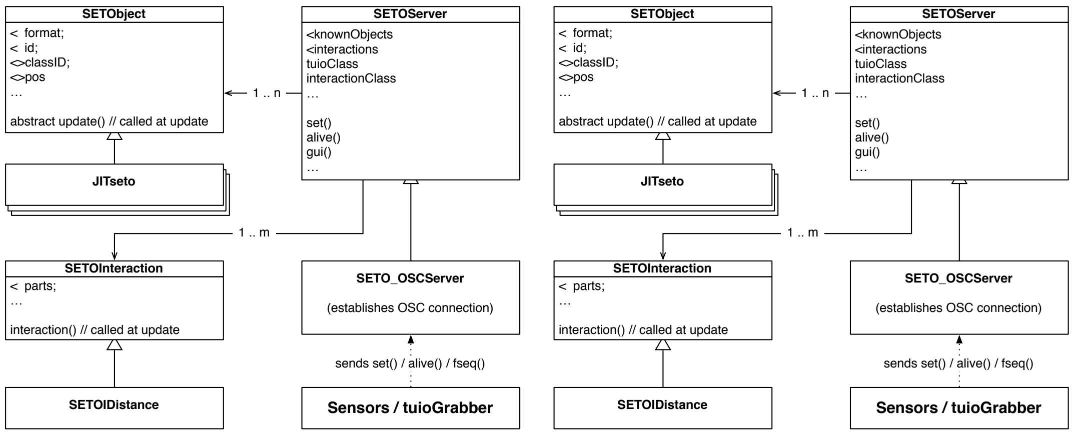
(a) of an object may be written as:

```
SynthDef(\testTUIO, {|freq = 400, out = 0, amp = 0, vol = 0.25, famp=1|
    Out.ar([out, out+1], SinOsc.ar(freq, 0, (amp.lag(0.01)*vol*famp)))
}).send(s);

q = ();
(
q.synths = IdentityDictionary.new; // a storage for synths

JITseto.action = {|me|
    s.bind{
        // make sure there is a synth
        q.synths[me].isNil.if{
            q.synths[me] = Synth(\testTUIO, [\vol, 0.2, \amp, 0])
        };
        s.sync;
        me.visible.if({
            q.synths[me].set(
                \freq, me.rotEuler[0].wrap(0, 2pi)
                       .linexp(0, 2pi, 400, 800),
                \amp, 1
            )
        }, {
            q.synths[me].set(
                \amp, 0
            )
        })}}
)

// instantiate SETOServer
t = SETO_OSCServer('_ixya', setoClass: JITseto);
t.gui;
t.start;
t.stop;
```

This approach does what is expected. However, it does not make use of the whole potential the TUIO interface definition provides for example to manage object representations and interactions. For this, SETO can be used, which de-couples functionality from implementation details by providing a higher abstraction level, such that the developer can concentrate more on the intended functionality than on the low-level features. SETO therefore takes care of object changes, including their visibility, and provides implementation interfaces to both actions and interactions of tangible objects.

10.2.1. Implementation

As shown in Figure 10.3(a), SETO mainly consists of three types of classes. Each tracked object is represented by an instance of a class derived by **SETObject**. Added algorithmic behaviour, e.g. the modulation of a corresponding sound depending on the physical object's motion can be implemented either as a fixed method of a custom class derived from **SETObject**, or by implementing the action of **JITseto**, a class providing just-in-time



(a) SETO classes

(b) JugglingSounds classes

Figure 10.3.: UML dependency diagrams related to SETO.

functionality for tangible objects. Interactions between physical objects may be defined alike by deriving from **SETOInteraction**. With SETO, the above shown example can be written as

```
(
q.synths = IdentityDictionary.new; // a storage for synths

JITseto.action = {|me|
    s.bind{
        // make sure there is a synth
        q.synths[me].isNil.if{
            q.synths[me] = Synth(\testTUIO, [\vol, 0.2, \amp, 0])
        };
        s.sync;
        me.visible.if({
            q.synths[me].set(
                \freq, me.rotEuler[0].wrap(0, 2pi).linexp(0, 2pi, 400, 800),
                \amp, 1
            )
        }, {
            q.synths[me].set(
                \amp, 0
            )
        })}}
)
```

Then, instantiate a **SETOServer**:

```
t = SETO_OSCServer('_ixya', setoClass: JITseto);
t.gui;
t.start;
t.stop;
```

In this listing, a **SETO_OSCServer** is instantiated that holds all representations of tangible objects for the programmer. All objects tracked by the tracking system are represented as **JITseto** objects, a child of **SETObject** that evaluates a global class function whenever they are updated. This is the place where the actual audio controlling takes place. At first sight this example might look as complex as the non-SETO one, however it has some benefits compared to it: The simple approach is limited to an explicit number of objects of which the identification numbers have to be successors of each other. The SETO example allows to use an arbitrary number of objects. While the mapping of object-behaviour to sound parameters is fixed once the system is set up, in the SETO example it is interchangeable at runtime.

10.2.2. Application

Together with colleagues, I implemented several TAIs with help of SETO: AudioDB (see Section 9.4), JugglingSounds (see Section 9.6), Ambit – a tangible environment for ambient data representation [BHR06b], and Tangible Interface for interactive Sonification of multivariate data [HBRR07]. To give an insight into the possibilities of SETO, I describe SETOs use by means of the implementation of JugglingSounds. For detailed information on its use and especially on the Sonification process, please refer to Section 9.6.

JugglingSounds consists of two main parts, data acquisition via motion tracking and Sonification via SETO as shown in Figure 9.43(a). For proper Sonification results, the used tracking system had to support capturing of Objects in 6DOF at an update rate of at least 40Hz. Fort his, we used an optical motion capturing system developed by Vicon Inc. . It consists of at least 6 cameras, a data station and a personal computer running the ViconiQ software as well as the server application (*Tarsus*), which is connected to the data station via Ethernet. This setup is able to compute the 6DOF position of rigid bodies (here three juggling clubs and the juggler's head) about 120 times per second. This pseudo-continuous multidimensional stream of tracking data is translated into OSC messages by QVicon2OSC, and sends to the Sonification part, running on a separate computer. The non-TUIO-conforming OSC interface of *QVicon2OSC* required to subclass **SETOServer**. Its definition (**SETOTarsusServer**) may be used as a guideline for implementing new tracking interfaces for SETO. Attaching the non-TUIO-conforming OSC interface of *QVicon2OSC* to SETO required the implementation of a dedicated **SETOServer** class. Its definition – **SETOTarsusServer** – may be used as a guideline for implementing new tracking interfaces in SETO.

SETO internally uses homogeneous transformations, consisting of a translation and a rotation part represented in a 4×4 matrix. The objects' 6DOF position, however, was provided by *QVicon2OSC* in axis notation. To save computational time, SETO only computes the transformation into a homogeneous form when it is needed.[9]

The complexity of the JugglingSounds system forced us to subclass from **SETObject** in order to represent juggling clubs and the juggler's head. The resulting class dependencies are shown in Figure 9.43(b). The class **Club** represents one tracked juggling club, while **JugglersHead** corresponds to the juggler's head. Each club's or head's motion can trigger the computation of Sonification-relevant features. Examples are the state of the club's

[9]For more information on the mathematical background on rotational representations, please refer e.g. to `http://en.wikipedia.org/wiki/Rotation_representation_(mathematics))`

symmetric axis (*symAxis*), or *zeroCrossing*: an indicator that returns true only if the symmetric axis has crossed the z-plane in the last update. The action methods of **Club** and **JugglersHead** were used to update the values of all incorporated **NodeProxies**. Examples are shown in the listing below.

```
Club.action = {|me|
    me.flipAngleVel.isNaN.not.if({
        ~rotVel.set(me.id, me.flipAngleVel);
    }, {
        ~rotVel.set(me.id, 0);
    });
    ~absX.set(me.id, me.pos[0]);
    ~absY.set(me.id, me.pos[1]);
    ~height.set(me.id, me.pos.last);

    ~relX.set(me.id, me.posRelHead[0]);
    ~relY.set(me.id, me.posRelHead[1]);

    ~zeroCrossing.set(me.id, me.zeroCrossing.binaryValue);
    ~catched.set(me.id, me.catched.binaryValue);
    ~posRelGPointX.set(me.id, me.posRelGroundPoint[0]);
    ~posRelGPointY.set(me.id, me.posRelGroundPoint[1]);
    ~posRelGPointZ.set(me.id, me.posRelGroundPoint[2]);
    ~posRelHeadX.set(me.id, me.posRelHead[0]);
    ~posRelHeadY.set(me.id, me.posRelHead[1]);
    ~posRelHeadZ.set(me.id, me.posRelHead[2]);
};

JugglersHead.action = {|me|
    ~regionChanged.set(0, me.regionChanged.binaryValue);
    ~region.set(0, me.region);
    me.regionChanged.if{
        q.regionChange(me.region);
    };
    ~headAbsX.set(0, me.pos[0]);
    ~headAbsY.set(0, me.pos[1]);
    ~headHeight.set(0, me.pos[2]);

};
JugglingInteraction.headClubAction = {|distance, isValid, head, club|
    ~dist.set(club.id, distance);
};
```

A basic implementation of a JugglingSounds setup can be found on the accompanying DVD.

10.2.3. Conclusion

In this section, I presented SETO, the SuperCollider Environment for Tangible Objects. It can be used to set up complex environments for tangible computing. It is especially suitable for environments where interaction with the code is needed. In this situation, it provides easy access to higher-level functionality. Based on an initial design, SETO was continuously developed over the implementations of the various Tangible Auditory Interfaces it was used for. Its software design proofed to be flexible, yet efficient for the various applications. In the future, we want to use it for a TAI that incorporates cube-shaped objects with an edge length of 60cm. It is developed in co-operation with Animax, Bonn.[10]

[10]More information, photos and a video of a proof of concept implementation can be found on `http://lfsaw.de/art/installations/SoundBlox.shtml`.

11. Conclusion

In this work, the integration of Tangible Interfaces (TI) with Auditory Displays (AD) into Tangible Auditory Interfaces (TAI) was introduced and investigated. This approach adds considerable value to both interface techniques, and it provides a reality-based interface to Sonification and Auditory Displays. Tangible Interfaces are strongly anchored in the physical environment. Therefore they need to be carefully integrated into the auditory representation system. If interface designers pay attention to this fact, their combination results in concise interfaces that blend data and algorithmic functionality into our everyday life. TAIs incorporate our manual and auditory skills and therefore turn data into a material in its original sense, incorporating physical qualities and (artificially induced) structure-borne sounds.

I have divided the thesis into two parts, *Interfacing Digital Content with Auditory and Physical Devices* and *Systems Incorporating Tangible Auditory Interfaces*. The first part concentrated on the theoretical background for TAIs, whereas the second part presents seven proof of concept applications for the TAI paradigm that are complemented by a description of software and hardware frameworks, which were developed to support their implementation. Together, the two parts illustrate in theory and practice that the TAI approach is a promising paradigm to produce interfaces between the abstract space of data and our everyday environment.

Part I started with a detailed description and analysis of TAI-related fields. For each of them, I reported on characteristics, motivation, and related research areas. I supplemented each description by insights into aspects and features that are valuable regarding their utilisation in TAIs. In the last chapter of the first part, I formulated the synthesis of TI and AD into the Tangible Auditory Interface (TAI) paradigm. Following the description of the research field, I proposed a definition and introduced their inherent characteristics and key features: (*a*) their potential to *augment* everyday artefacts, (*b*) their *interface richness* in both sensing and display, (*c*) their natural property to serve as a toolset for *collaborative work*, and (*d*) their tendency to easily support *immediacy and flow*. *Part I*

Motivated by these theoretical considerations, seven proof of concept applications were described in Part II. These are MoveSound (Section 9.1), ChopStix (Section 9.2), Reim (Section 9.3), AudioDB (Section 9.4), Tangible Data Scanning (Section 9.5), JugglingSounds (Section 9.6), and Durcheinander (Section 9.7). Each deals with one or more of the subjects Exploratory Data Analysis, multi-modality, Interactive Sonification, auditory augmentation, collaborative work, direct control, ambience, and ergonomics. In Section 9.8, I have discussed the described TAIs in relation to the characteristics introduced in Chapter 7. The introduced key features of the TAI paradigm were demonstrated by the proof of concept applications as follows: *Part II*

Augmentation With Reim, I have exemplified that the TAI paradigm can be used to develop data monitoring applications that augment structure-borne sounds by data-driven audio, such that the resulting sound event is perceived in combination with the haptic experience as a coherent gestalt. This allows to directly experience data as an auditory augmentation of everyday objects.

Interfacing Richness TDS, Durcheinander, JugglingSounds and ChopStix provide a rich interface in both user input and system output. Thus, data-inherent qualities are completely mediated to the users, and can be manipulated on a detailed level. By their potential to shift the level of abstraction, these applications allow users not only to manipulate data representations on different symbolic levels but also to decide on their preferred level on a per-object basis.

Immediacy and Flow The AudioDB case study indicates that it is possible to develop TAIs that incorporate both immediacy and flow. All participants were able to successfully manipulate the sounds that were linked to the tangible objects and, over a short learning period, reported fluidity in their operation.

Collaborative Work All but one of the presented applications can be used collaboratively. AudioDB and Durcheinander for example were designed especially for the use with many people and therefore serve as tools to support discussions and other team-based work on digital media.

Ergonomics Sound-object linkage as done e.g. in ChopStix, AudioDB and Reim were designed to be closely related to the users' common real-world experiences. The observations made in the case studies and in other public demonstrations support the validity of this strategy.

Tight Coupling Especially Reim, JugglingSounds and ChopStix were designed in order to establish a direct mapping between sensor measurements on the TI and the reaction of the incorporated AD. This supports the user in understanding the functionality of the underlying system, which in turn is one important factor to experience immediacy and flow.

Ambience The applications ChopStix, Durcheinander and Reim connect long-term monitoring with Tangible Interfaces that keep their state as long as they are not actively changed. The different approaches presented in these three applications exemplify the broad variety that the TAI paradigm supports for this feature.

A consolidated view of these features indicates that the TAI paradigm contributes a considerable progress in the human computer interface research towards nature-inspired and feature-rich data representations that fully integrate the human sensory and cognitive capabilities.

11.1. Further Work

Within the scope of this thesis, not all TAI-related aspects could be investigated due to time and capacity constraints. The main desirable future research aim with regards to theoretical considerations would be to establish an evaluation methodology for alternative human-computer interfaces. Especially TAIs would significantly benefit from such investigations, leading to evaluation results that are more comparable. However, the development of this methodology would require efforts in many related disciplines, such as Sociology, Psychology, Interaction Design, and Computer Science. I believe that the possibly extensive work that is needed to closely integrate these areas would be worth the effort because it would result in high-quality guidelines on how to properly evaluate alternative human-computer interface techniques. This is especially promising for the evaluation of TAIs concerning their collaborative feature.

These considerations could be combined with the exploration of the influence of object size and the incorporation of multi-touch features into TAIs. The latter would add algorithmic flexibility due to their close integration of manual control into visual displays. Regarding the specific TAIs introduced in Chapter 9, it would be a promising option to actually build custom interfaces originating in the design study of e.g. the Reim Stethoscope in order to test their feasibility and performance. In general, every investigation towards a closer coupling of data to our everyday environment would significantly add value to our everyday handling of information.

A. Measuring the Quality of Interaction

A.1. Replies

In this Appendix, the translated replies to the email-survey on measuring the quality of interaction are collected.

Participant 1

I would offer a hands-on workshop, but the question is more how I then would evaluate the interactive qualities?

I think I would stick to the workshop idea. Then, I would come up with a task which should be easily resolved with my application. Then, I would look, if the subjects get to the solution right away, and if they have fun in exploring the right way to solve the problem. On this observation, I then see, if I created an "intuitive" accessibility.

Participant 2

I would do a questionnaire. One could also present a certain task which the users have to solve with the application. Then, you may look how long they need to solve it and count their errors.

Participant 3

1. *idea: design a task–let people do the task with and without it–measure time/success*
2. *idea: shoot videos and evaluate them ethnomethodologically, i.e. which steps do the people undertake, where do they stick, what are they doing wrong, etc.*

Participant 4

brainstorming:
Interaction quality cannot be measured generally, but only in correlation to a distinct work process. I.e. it must be clear what to actually do with it (or that there is nothing to do with it, but that for true).

Then, the learning process can be understood (someone is not able to do something, and is now experiencing the way how to do it), or as a reinforcement of accepted "already learned". Which, again, cannot be known, but only formulated as an assessment by yourself or by others.

Thus, you surely have to ask what a subject anticipates from an application. E.g. should it be interesting, helpful, entertaining, sympathetic, powerful, secure, foolproof, instructive, open source, have a reliable future, etc.

potential procedure:

1. *show all the functionality without doing them. (Or omit this step)*
2. *Let the subjects describe what they could do, they should make a proposal what they anticipate of such a system.*
3. *Let the subjects act.*
4. *Ask the subjects how their plan correspond to the realisation.*

The quality then is not determined by the accordance between 2. and 4., but somehow different, more in relation to the above questions. Is there potential for getting new ideas? Does the subject learn something? Did he know something from somewhere else? Can the application be used somewhere else? Is it fun to be irritated? Is it awkward to have fun? Is it patronising? Is it fun to entrap the system or does that break something? Or, more general, is the misuse of functionality interesting?

Participant 5

Nun wollt ihr wissen ob und wie Nutzer mit dem Ding klarkommen,
If you have "real" users, assemble a task and let them (unexperienced users) solve it.

I.e.,sticking to the example of the photoshop plugin:

- *open an image*
- *add a layer*
- *apply plugin xyz on it (I don't have a clue of graphic software)*

Then, ask questions like

- *How easy were the single steps for you and why?*
- *Was it difficult to get the meaning of the adjustments out of the GUI?*
- *etc.*

Different Approach: Instead of evaluating the questions afterwards, you could sit next to the user and ask him to think aloud, i.e. as soon as something is unclear or they find something good/bad they should tell it, but you do not respond, but only note it down.

sozusagen wie gut die Art der Interaktion funktioniert.
With the right questions (see above) this is also possible.

Participant 6

I would test in two different ways. First, I would vary the interaction structure. Second, I would vary the tasks to be solved. Ask people which interaction they prefer according to the single tasks.

But first of all, just look at people interacting with the system and learn something in the sense of grounded theory.

Participant 7

Give a group of participants a defined set of tasks and look how they perform. At best, one has two GUI or such, which allow to compare the performance and the questionnaire's analysis. Otherwise, one has to get a way on how one evaluates the approaches of the participants, and if one can see parts were it crashes more than needed.

Participant 8

*I would give the user a Task, e.g. "Create a new picture with size 800*600px". Then measure the number of clicks and the needed time. As an addition, the user may give a rating from 1-6 regarding his opinion concerning the menu navigation.*

Participant 9

Generally, I see it like Joel Spolsky in "User Interface Design for Programmers":

The next step is to test your theories. Build a model or prototype of your user interface and give some people tasks to accomplish. The model can be extremely simple: sometimes it's enough to draw a sloppy picture of the user interface on a piece of paper and walk around the office asking people how they would accomplish x with the "program" you drew. As they work through the tasks, ask them what they think is happening. Your goal is to figure out what they expect. If the task is to "insert a picture," and you see that they are trying to drag the picture into your program, you'll realize that you had better support drag and drop. If they go to the Insert menu, you'll realize that you had better have a Picture choice in the Insert menu. If they go to the Font toolbar and replace the word "Times New Roman" with the words "Insert Picture", you've found one of those old relics who hasn't been introduced to GUIs yet and is expecting a command-line interface. How many users do you need to test your interface on? The scientific approach seems like it would be "the more, the better." If testing on five users is good, testing on twenty users is better! But that approach is flat-out wrong. Almost everybody who does usability testing for a living agrees that five or six users is all you need. After that, you start seeing the same results again and again, and any additional users are just a waste of time. The reason being that you don't particularly care about the exact numerical statistics of failure. You simply want to discover what "most people" think. You don't need a formal usability lab, and you don't really need to bring in users "off the street"–you can do "fifty-cent usability tests" where you simply grab the next person you see and ask them to try a quick usability test. Make sure you don't spill the beans and tell them how to do things. Ask them to think out loud and interview them using open questions to try to discover their mental model.

If you like it a bit more formal, I can think of the following methods with users: Benchmark-Tests: user gets a task and is evaluated for efficiency/performance and asked for comfort/-transparency. There are different phases, where investigators may intervene–this needs some time to prepare and is rather formal; in addition, user tend to feel like a guinea pig.

Thinking Aloud: user says what he currently wants to do and what he thinks of when he sees the GUI: fast and cheap. The output is excellent. Highly depends on the users. Some talk too much, others do not say a word without an actual request.

Constructive Interaction: Like think aloud, but with two participants. Also very interesting, but leads to unusable findings, if there is a bad user pair (e.g. one participant gets intimidated by the other, or one of them always asks the investigator).

Even without users, it is possible to test GUI's with certain heuristics (and/or with use-cases).

The Information Center for Social Science has an interesting report on evaluating software: Marcus Hegner, Methoden zur Evaluation von Software, Mai 2003, IZ-Arbeitsbericht Nr. 29

Participant 10

1. *user experience: Let people play with it and ask them. let them rate criteria on a 1-10 scale:*
2. *how convenient to use?*
3. *how fast can you handle things with it?*
4. *is it fun to use?*
5. *possibly ask for an open comparison or the-like?*
6. *Comparison with alternative approaches: assign a task to do with own software (A) or with an alternative one (B) and measure time.*
7. *then, ask again how convenient/efficient the users found the work with A resp. B.*
8. *which measured criteria correlate how well with the answers? (is 'experienced' efficiency really fast, etc.)*

Participant 11

There are various Factors:

Learnability *How fast learns the user the interaction, how does the reaction time changes over time of usage, in which frequency errors occur (wrong usage, difference between intended and obtained usage) (-> error-rate, reaction rate)*

Cognitive Load *how much needs the interaction cognitive control (to be measured by performance bumps in parallel secondary tasks). Related with the potential to be automated.*

Memory Intensity *how much memory is used to perform an action (RAM). E.g. Emacs needs the learning of many shortcuts until one can work with it in a suggestive way. The more Memory is consumed, the slower is the learning curve, perhaps at better performance...*

Flow-Experience *Interactions that cause a flow of action such that the agent is dissolving in the process, to be captured by flow.questionaires after Reichberg, at spontaneous interruptions of interactants*

Stress test *the more exhausting an interaction, the more will the interaction cause stress, it can be captured by physical measurements, from cutaneous electrical resistance to pulse rate to, etc. or by the impact of psychic distractors.*

Tiredness *like above, another aspect of pressure by an interaction*

Fun/Motivation *motivating interactions should find a subjectively more positive rating of the users, to be measured possibly by mapping associations (e.g. inkblots as good/bad animals)*

Latency/Dead-times *(connected with Flow) If there are Dead-times in Interaction processes (e.g. pauses between action and reaction) these may break the flow subjectively, this will affect flow and motivation (estimation: the more latency, the unpleasant the quality)*

Modality allocations / Naturalness *Interactions in real contexts always address a mixture of various modalities in a harmonic balance–a from "natural examples" strongly diverging concentration on only one modality could argue in favour of a bad interaction quality.*

[. . .]

Most of these aspects can be operationalised into comparison experiments between two interactions. For an absolute interaction quality scale I cannot see an all-too easy definition.

Participant 12

Qualitative (Though, I haven't got a clue of it)

Or something like

1. *reaction test*
2. *let them work 8 hours with the tool*
3. *reaction test*

Thesis: the higher the cognitive load, the more tired the subjects.

A.2. Generated Categories

```
scenario>workshop
scenario>task

method>qualitative
    method>ethnomethodology
    method>ethnomethodologic evaluation
    method>grounded theory
    method>User experience
    method>let describe
    method>let act
    method>Play and Ask
    method>ask for correspondence between action and imagination
    method>ask for anticipations regarding the application
method>quantitative
    method>thinking aloud
    method>constructive interaction
    method>test for cognitive load with a stress test
```

```
    method>vary interaction structure / task
method>questionnaire
    method>criteria rating on a scale (0..10) (1..6)
    method>user rating
method>compare
    method>compare two different approaches/applications
    method>compare w, w/o system
method>heuristics on typical users (without real users)


material>questionnaire
    material>questionnaire on which task with which interaction
material>interview
    material>video
    material>audio
    material>notes
material>quantitative measuring
    material>measurement of time
    material>measurement of clicks

indicator>qualitative
    indicator>observe user's action
    indicator>where do they stick
    indicator>what are they doing wrong
    indicator>fun
    indicator>potential for new ideas, creativity, learning, fun, misuse?
    indicator>interaction quality;
              can only be measured in correlation to a distinct work process.
    indicator>answers to convenience/efficiency questions
    indicator>ask for possible convenience
    indicator>ask for possible speed
    indicator>ask for possible fun
    indicator>open comparison to a different system
indicator>quantitative
    indicator>performance
    indicator>number of clicks
    indicator>needed time
    indicator>success rate
    indicator>efficiency
    indicator>error rate
    indicator>time to complete
    indicator>number of steps people need
    indicator>rating concerning menu navigation
indicator>correlation between measurable/quantitative
          indicators and qualitative user ratings
```

B. MoveSound Material

B.1. OSC Protocol

The following table lists the OSC commands to control MoveSound's visual display.

`/aMode (int) which (int) val`	set active mode of an object to val (either 0 or 1)
`/oAbs (int) which (float) val`	set orientation of a sound object to val
`/wAbs (int) which (float) val`	set width of a sound object to val
`/name (int) which (string) name`	set name of specified sound object to name
`/kulerKey (string) themeID`	acquire kuler theme with ID themeID
`/defaultColor`	set colorset to default colors

`/cmap (int)back (int)aLow (int)aHigh`
 	
`(int)dLow (int)dHigh (int)wLow (int)wHigh`
	set the mapping from the five colors
	in the colour-table to the functional parts

`/nSrcs (int) num (int) mode`	set number of sources to num; vision mode to mode
`/nSpeakers (int) num (float) orientation`	set number of loudspeakers and their orientation regarding the front
`/debug (int) trueFalse`	(de)activate debug mode
`/demo [(string) multiLineDescription]`	draw a description if provided a string, otherwise disable that feature

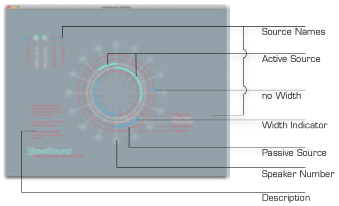
B.2. Case Study Handout

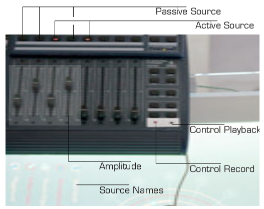
Welcome to MoveSound Experiments

MoveSound is an azimuth panning interface for playback of sound sources in a ring of speakers. Both, the position and width of arbitrary multichannel sound sources can be adjusted with it. By providing you with an interface to select either one or more sound sources to operate on, MoveSound allows you to control several sound sources at the same time. Together with the integrated azimuth- and width-panning capabilities it's functionality gains access to the field of dynamic sound spatialisation.

Below, you find two diagrams explaining MoveSounds capabilities; they are intended as a reference in the following experiment, so please use them whenever you need them. Please, also do not hesitate to ask me for further assistance.

The survey you are currently taking part is divided into six challenges. Please complete them in order, and fill out the questionaire at the end.





The MoveSound Interface

Above, you see a part of a faderbox with explanations, and the graphical interface that is on the table in front of you.[1]

MoveSounds interface is explained quickly: Its cenral element is the knob in the middle of the surface. You can use it to control two aspects of your sound sources, changing their position on the ring and their *width*. To give you information about currently selected sources, the light-red elements in the visual display around the knob indicate your current selection, whereas unselected sources are filled with a darker red.

Please try now to change the (pre-)selected source by rotating the Knob. Do you hear the effect?

Now, rotate the knob while pressing pressing it. This way, you are able to control the width of the source's playback. Can you hear the effect?

If you have any open questions, lets see if they are answered during the setup. For the experiment tracking, it would be nice if you could inform me, every time you complete a challenge, If you don't mind, tell me now that we proceed to the warm-up.

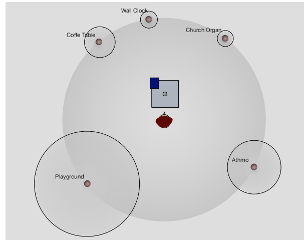
Enjoy your show!

1 If this is not the case, please tell me that I have prepared the wrong interface for you.

Warm-up

Please adjust the position of the sound sources according to this sketch.

I already adjusted the volume for you, so you do not have to care.



Your Audio Sources

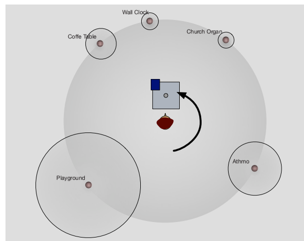
These five sound sources are currently linked to the interface:

1. children playing on a **Playground**,
2. chatting people on a **Coffe Table**,
3. an organist on a **Church Organ**,
4. the **Athmo** of a ventilation system, and
5. a large, ticking **Wall Clock**.

1st Challenge

Done with the warmup? Nice!

Can you think of a way how you can simulate the effect that you would hear when you turn your head? Please, try to emulate this with the MoveSound Interface and keep your head straight to the front.



Tip: It is possible to select more then one sound source.

MoveSound
Experiments

2

2nd Challenge

Please, adjust the sound's position, width and amplitude such that you feel comfortable with it. Ask yourself, if you would like to listen to this soundscape for a longer period of time. Readjust the parameters eventually.

Your new Audio Sources

These five sound sources are currently linked to the interface:

1. a **Playground** full of children,
2. calm **Footsteps** of a walking person,
3. a **Plane** from far away,
4. people playing **Table Soccer**, and
5. a **Radio** that plays music.

3rd Challenge

Please try to emulate dynamic movements. In the sketch on the right, you see a plot of a little screenplay.

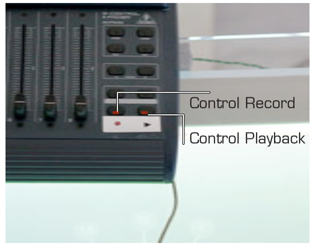
With your skills in the MoveSound interface, you now should be able to perform it. Please practice a bit before proceeding to the next challenge.

MoveSound Experiments

3

4th Challenge

Please perform your little sound-piece and record it. After that, play it back to us. If you do not like what you have recorded, just re-press the record button and perform once more.

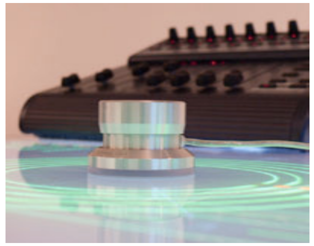


Recording

Do you see the two buttons in the right photo? The left one toggles recording, whereas the right one starts and stops playback.

5th Challenge

Do you have any advice for me on how to change the behavior of MoveSound, especially concerning the interaction design? Let's talk, maybe I can manage to adjust it to your needs.



Thank You!

I hope, you enjoyed using MoveSound. It would be great, if you could complete the questionnaire on the next page.

MoveSound Experiments

4

Bibliography

[AFM+99] D. Anderson, J. L. Frankel, J. Marks, D. Leigh, K. Ryall, E. Sullivan, and J. Yedidia. Building virtual structures with physical blocks. In *12th International Symposium on User Interface Software and Technology*, pages 71–72. Shire Publications, 1999. 45

[And72] D. F. Andrews. Plots of high-dimensional data. *Biometrics*, 29:125–136, 1972. 69

[Baa04] M. A. J. Baalman. Application of Wave Field Synthesis in electronic music and sound installations. In *Proceedings of the 2nd International Linux Audio Conference*, Karlsruhe, 2004. 75

[BCC+05] C. Bornand, A. Camurri, G. Castellano, S. Catheline, A. Crevoisier, E. Roesch, K. Scherer, and G. Volpe. Usability evaluation and comparison of prototypes of tangible acoustic interfaces. *Proceedings of ENACTIVE 2005*, 2005. 127

[BD97] S. Brave and A. Dahley. inTouch: a medium for haptic interpersonal communication. *Conference on Human Factors in Computing Systems*, pages 363–364, 1997. 3

[BEHR08] T. Bovermann, C. Elbrechter, T. Hermann, and H. Ritter. AudioDB: Get in touch with sounds. In *Proceedings of the 14th International Conference on Auditory Display*, 2008. 132

[Ber88] A. J. Berkhout. A holographic approach to acoustic control. *Journal of the Audio Engineering Society*, 36(12):977–995, 1988. 75

[BGdCE07] T. Bovermann, J. Groten, A. de Campo, and G. Eckel. Juggling sounds. In *Proceedings of the 2nd International Workshop on Interactive Sonification*, York, February 2007. 152

[BH03] M. Berthold and D. J. Hand, editors. *Intelligent Data Analysis: An Introduction, second edition.* Springer Verlag, New York, NY, USA, 2003. 15, 69

[BH05] P. Baines and A. Haslam. *Type and Typography.* Watson-Guptill Publications, Inc., New York, NY, USA, 2005. 54

[BHR05] T. Bovermann, T. Hermann, and H. Ritter. The local heat exploration model for interactive sonification. In *Proc. of the Int. Conf. on Auditory Display 2005.* ICAD, Jul 2005. 17, 150

[BHR06a] T. Bovermann, T. Hermann, and H. Ritter. Tangible data scanning sonification model. In Tony Stockman, editor, *Proceedings of the International Conference on Auditory Display (ICAD 2006)*, pages 77–82, London, UK, 6 2006. International Community for Auditory Display (ICAD), Department of Computer Science, Queen Mary, University of London. 79, 144

[BHR06b] T. Bovermann, T. Hermann, and H. Ritter. A tangible environment for ambient data representation. In *First International Workshop on Haptic and Audio Interaction Design*, volume 2, pages 26–30. multivis, Aug 2006. 171, 176

[BKHR08] T. Bovermann, R. Koiva, T. Hermann, and H. Ritter. TUImod: Modular objects for tangible user interfaces. In *Proceedings of the 2008 Conference on Pervasive Computing*, 2008. submitted. 169

[BKJ05] R. Bencina, M. Kaltenbrunner, and S. Jordà. Improved topological fiducial tracking in the reactivision system. In *Proceedings of the IEEE International Workshop on Projector-Camera Systems (Procams 2005)*, San Diego, USA, 2005. 169, 170

[BMM06] S. Brewster, D. McGookin, and C. Miller. Olfoto: designing a smell-based interaction. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 653–662, New York, NY, USA, 2006. ACM. 68

[BRR08] T. Bovermann, J. Rohrhuber, and H. Ritter. Durcheinander: Understanding clustering via interactive sonification. In *Proceedings of the 14th International Conference on Auditory Display*, Paris, France, June 24 - 27 2008. 162

[BWE94] S. A. Brewster, P. C. Wright, and A. D. N. Edwards. A detailed investigation into the effectiveness of earcons. In G. Kramer, editor, *Auditory Display*, pages 471–498. ICAD, Addison Wesley, 1994. 70

[Car09] J. M. Carroll. Human Computer Interaction (HCI). `http://www.interaction-design.org/encyclopedia/human_computer_interaction_hci.html`, 2009. 32

[CH96] I. J. Cox and S. L. Hingorani. An efficient implementation of Reid's multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(2):138–150, 1996. 135

[Che73] H. Chernoff. The use of faces to represent points in k-dimensional space graphically. *Journal of the American Statistical Association*, (68):361–368, 1973. 69

[Cle93] W. S. Cleveland. *Visualizing Data.* Hobart Press, 1993. 69

[Com91] P. Comon. Independent Component Analysis. In *Proceedings of the International Workshop on Higher-Order Statistics*, pages 111–120, Chamrousse, France, 1991. 17

[Cor06] G. Cormode. Some key concepts in data mining: Clustering. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 70, 2006. 162

[Cow00] D. Coward. Analog Computer Museum. `http://dcoward.best.vwh.net/analog/`, 2000. 29

[CS95] G. Crampton Smith. The hand that rocks the cradle. ID magazine, May/June 1995. 44

[Csi00] M. Csikszentmihalyi. *Beyond Boredom and Anxiety: Experiencing Flow in Work and Play.* Jossey Bass Wiley, 2000. 41, 79 [dC09a] A. de Campo. *Science By Ear. An Interdisciplinary Approach to Sonifying Scientifc Data*. PhD thesis, Institute of Electronic Music and Acoustics - IEM, University for Music and Dramatic Arts Graz, 2009. 4, 21, 22

[dC09b] A. de Campo. *The SuperCollider Book*, chapter Microsound. In Wilson et al. [WCC09], 2009. in press. 74

[dCFH04] A. de Campo, C. Frauenberger, and R. Höldrich. A generic, semantically based design approach for spatial auditory computer displays. In *Proc. of the Int. Conf. on Auditory Display 2004*. ICAD, 2004. 71

[dHKP02] G. de Haan, M. Koutek, and F.H. Post. Towards intuitive exploration tools for data visualization in VR. In *Proceedings of the ACM symposium on Virtual reality software and technology*, pages 105–112. ACM New York, NY, USA, 2002. 17

[Dic05] B. Dick. Grounded theory: a thumbnail sketch. `http://www.scu.edu.au/schools/gcm/ar/arp/grounded.html`, 2005. 38

[Dom01] F. Dombois. Using audification in planetary seismology. In Nick Zacharov Jarmo Hiipakka and Tapio Takala, editors, *Proc. of the Int. Conf. on Auditory Display 2001*, pages 227–230, Helsinki University of Technology, 2001. ICAD, Laboratory of Acoustics and Audio Signal Processing. 71

[Dom02] F. Dombois. Auditory seismology: on free oscillations, focal mechanisms, explosions and synthetic seismograms. In *Proc. of the Int. Conf. on Auditory Display 2002*. ICAD, July 2002. 71

[Dou01] P. Dourish. *Where the action is: The Foundation of Embodied Interaction*. MIT Press, 2001. 36, 43, 46, 48, 169

[Dre03] H. Dreyfuss. *Designing for People*. Allworth Press, November 2003. ISBN 158-115-312-0. 32

[ELL01a] B. S. Everitt, S. Landau, and M. Leese. *Cluster analysis*. Arnold, London, 4. edition, 2001. 162

[ELL01b] B. S. Everitt, S. Landau, and M. Leese. Hierarchical Clustering. *Cluster Analysis. Oxford University Press Inc., Publishers, New York, NY*, pages 55–89, 2001. 162, 164

[Eng62] D.C. Engelbart. Augmenting human intellect: A conceptual framework. *summary report AFOSR-3223 under contract AF*, 49(638):1024, 1962. 31

[EO04] G. Essl and S. O'Modhrain. Scrubber: an interface for friction-induced sounds. In *NIME '05: Proceedings of the 2005 conference on New interfaces for musical expression*, pages 70–75, Singapore, Singapore, 2004. National University of Singapore. 119

[FBB05] M. Fernström, E. Brazil, and L. Bannon. HCI design and interactive sonification for fingers and ears. *IEEE MultiMedia*, 12(2):36–44, 2005. 133

[FdC07] C. Frauenberger and A. de Campo. Analysing time series data. In *ICAD Proceedings*, page 504–508, Montreal, Canada, June 2007. International Community for Auditory Display, International Community for Auditory Display. 74

[FIB95] G. W. Fitzmaurice, H. Ishii, and W. Buxton. Bricks: Laying the foundations for graspable user interfaces. In *Proceedings of CHI 1995*, pages 442–449, 1995. 42, 78

[FMI00] P. V. S. Frei, B. Mikhak, and H. Ishii. Curlybot: Designing a new class of computational toys, 2000. 68

[Fod91] J. Foderaro. LISP: introduction. *Communications of the ACM*, 34(9):27, 1991. 12

[FPSSU96] U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors. *Advances in Knowledge Discovery and Data Mining*. MIT Press, 1996. 15

[FTJ08] Y. Fernaeus, J. Tholander, and M. Jonsson. Towards a new set of ideals: Consequences of the practice turn in tangible interaction. In *Proceedings of the Second International Conference on Tangible and Embedded Interaction (TEI'08)* [TEI08]. 46

[Gav94] W. W. Gaver. Using and creating auditory icons. In G. Kramer, editor, *Auditory Display*, pages 417–446. ICAD, Addison-Wesley, 1994. 70

[GDA91] A. Giddens, M. Duneier, and R.P. Appelbaum. *Introduction to sociology*. Norton New York, 1991. 26

[Ger85] M. A. Gerzon. Ambisonics in multichannel broadcasting and video. *Journal of the Audio Engineering Society*, 33(11):859–871, 1985. 75

[Gol97] M. H. Goldhaber. The attention and the net. *First Monday*, 2(4), April 1997. 1, 127

[Gol06] M. H. Goldhaber. How (not) to study the attention economy: A review of the economics of attention: Style and substance in the age of information. *First Monday*, 11(11), November 2006. 1

[Gre06] A. Greenfield. *Everyware: The Dawning Age of Ubiquitous Computing*. New Riders Press, March 2006. 44, 45

[GWBW06] D. Göger, K. Weiß, C. Burghart, and H. Wörn. Sensitive skin for a humanoid robot. *Human-Centered Robotic Systems*, 2006. 47

[Hae09] R. Haeussling. Video analysis with a four-level interaction concept: A network-based concept of human-robot interaction. In Kissmann [Kis09]. ISBN 978-3-631-57473-7. 38

[Hay94] C. Hayward. *Listening to the Earth Sing*, pages 369–404. Addison-Wesley, 1994. 71

[HBRR07] T. Hermann, T. Bovermann, E. Riedenklau, and H. Ritter. Tangible computing for interactive sonification of multivariate data. In *Proceedings of the 2nd Interactive Sonification Workshop*, February 2007. 3, 17, 171, 176

[HBSR06] T. Hermann, G. Baier, U. Stephani, and H. Ritter. Vocal sonification of pathologic EEG features. In Tony Stockman et al., editor, *Proc. of the Int. Conf. on Auditory Display 2006*, pages 158–163, London, UK, 6 2006. International Community for Auditory Display (ICAD), Department of Computer Science, Queen Mary, University of London. 153

[Hea95] M. A. Hearst. Tilebars: Visualization of term distribution information in full text information access. In *Proceedings of the Conference on Human Factors in Computing Systems, CHI'95*, 1995. 69

[Hei27] M. Heidegger. *Sein und Zeit.* Niemeyer, Halle a.d.S., 1927. 1, 46, 67

[Hei92] M. Heilig. The cinema of the future. *Presence: Teleoperators and Virtual Environments. MIT Press*, 1(3), 1992. 31, 33

[Her02] T. Hermann. *Sonification for Exploratory Data Analysis.* PhD thesis, Bielefeld University, Bielefeld, February 2002. 145

[Her08] T. Hermann. Taxonomy and definitions for sonification and auditory display. In *Proceedings of the 14th International Conference on Auditory Display*, Paris, France, June 24 - 27 2008. 70

[Hew92] T. T. Hewett. Curricula for human-computer interaction. Technical report, ACM SIGCHI, New York, NY, USA, 1992. Chairman-Hewett,, Thomas T. 32

[HH05] A. Hunt and T. Hermann, editors. *IEEE Multimedia, Special Issue Interactive Sonification.* IEEE, 04 2005. 4

[HHH+08] D. Hong, T. Höllerer, M. Haller, H. Takemura, A.D. Cheok, G.J. Kim, M. Billinghurst, W. Woo, E. Hornecker, R.J.K. Jacob, et al. Advances in Tangible Interaction and Ubiquitous Virtual Reality. *IEEE Pervasive Computing*, 7(2):90–96, 2008. 2

[HHO+08] B. Hengeveld, C. Hummels, K. Overbeeke, R. Voort, H. van Balkom, and J. de Moor. Let me actuate you. In *Proceedings of the Second International Conference on Tangible and Embedded Interaction (TEI'08)* [TEI08], pages 159–166. 46

[HHR06] T. Hermann, O. Höner, and H. Ritter. AcouMotion: An interactive sonification system for acoustic motion control. In S. Gibet, N. Courty, and J.-F. Kamp, editors, *Gesture in Human-Computer Interaction and Simulation: 6th International Gesture Workshop, GW 2005, Berder Island, France, May 18-20, 2005, Revised Selected Papers*, volume 3881 of *Lecture Notes in Computer Science*, pages 312–323, Berlin, Heidelberg, 2006. Springer. 153

[HKR02] T. Hermann, J. Krause, and H. Ritter. Real-time control of sonification models with an audio-haptic interface. In R. Nakatsu and H. Kawahara, editors, *Proc. of the Int. Conf. on Auditory Display 2002*, pages 82–86. International Community for Auditory Display (ICAD), 2002. 49, 119

[HL65] E. Husserl and Q. Lauer. *Phenomenology and the Crisis of Philosophy.* Harper & Row, 1965. 46

[HMR00] T. Hermann, P. Meinicke, and H. Ritter. Principal curve sonification. In P. R. Cook, editor, *Proc. of the Int. Conf. on Auditory Display 2000*, pages 81–86. Int. Community for Auditory Display, 2000. 72

[Hor09] E. Hornecker. The tangible interaction framework. `http://www.ehornecker.de/TangiblesFramework.html`, 2009. 42, 45

[HP06] A. Hunt and S. Pauletto. The sonification of EMG data. In *Proc. of the Int. Conf. on Auditory Display 2006*, London, UK, 2006. 153

[HR99] T. Hermann and H. Ritter. Listen to your Data: Model-Based Sonification for Data Analysis. In G. E. Lasker, editor, *Advances in intelligent computing and multimedia systems, Baden-Baden, Germany*, pages 189–194. Int. Inst. for Advanced Studies in System research and cybernetics, 1999. 72, 144, 145

[HR01] L. Hallnäs and J. Redström. Slow technology: Designing for reflection. *Personal and Ubiquitous Computing*, 5(3):201–212, 2001. 109

[HR04] T. Hermann and H. Ritter. Growing neural gas sonification for exploratory analysis of high-dimensional data. In *Proceedings of the Int. Symposium of Multimodal visualiztation*, 2004. 72

[Hub85] P.J. Huber. Projection pursuit. *The Annals of Statistics*, pages 435–475, 1985. 69

[Ish08] H. Ishii. Tangible Bits: Beyond pixels. In *Proceedings of the Second International Conference on Tangible and Embedded Interaction (TEI'08)* [TEI08]. 2

[IU97] H. Ishii and B. Ullmer. Tangible bits: towards seamless interfaces between people, bits and atoms. In *CHI '97: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 234–241, New York, NY, USA, 1997. ACM. 31, 34, 65

[ixd] Interaction design association. http://www.ixda.org/. 32

[JD88] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data.* Prentice-Hall, Englewood Cliffs, NJ, 1988. 17

[JGH+08] R.J.K. Jacob, A. Girouard, L.M. Hirshfield, M.S. Horn, O. Shaer, E.T. Solovey, and J. Zigelbaum. Reality-based interaction: a framework for post-WIMP interfaces. 2008. 34, 44

[Jil94] A. Jillings. *Modern Club Swinging and Pole Spinning.* Butterfingers, Bath, UK, 1994. 99 pages, 116 diagrams. 154

[JKGB05] S. Jorda, M. Kaltenbrunner, G. Geiger, and R. Bencina. The Reactable. In *ICMC Proceedings 2005*, 2005. 45

[Jol86] I. T. Jolliffe. *Principal Component Analysis.* Springer, New York, 1986. 17, 151

[KACS03] S. Kalyuga, P. Ayres, P. Chandler, and J. Sweller. The expertise reversal effect. *Educational Psychologist*, 38(1):23–31, 2003. 2

[KBBC05] M. Kaltenbrunner, T. Bovermann, R. Bencina, and E. Costanza. TUIO: A Protocol for Table-Top Tangible User Interfaces. In *Proceedings of Gesture Workshop 2005.* Gesture Workshop, 2005. 170, 173

[Kei00] D. A. Keim. Designing pixel-oriented visualization techniques: Theory and applications. *IEEE Transactions on Visualization and Computer Graphics*, 6(1):59–78, 2000. 69

[KHS89] K. Kelly, A. Heilbrun, and B. Stacks. Virtual reality: an interview with Jaron Lanier. *Whole Earth Review*, 64:108–120, 1989. 2, 33

[Kis09] U. T. Kissmann, editor. *Video Interaction Analysis: Methods and Methodology.* Peter Lang Publishing Group, Frankfurt am Main, 2009. ISBN 978-3-631-57473-7. 196, 199

[KL02] F. Kilander and P. Lönnqvist. A Whisper in the Woods: An ambient soundscape for peripheral awareness of remote processes. In *Proc. of the Int. Conf. on Auditory Display 2002*, 2002. 3

[Kra94] G. Kramer, editor. *Auditory Display.* Addison-Wesley, 1994. 3, 4, 21, 22

[Kru09] A. L. Krummheuer. Conversation analysis, video recording, and human-computer interchanges. In Kissmann [Kis09], pages 59 – 86. ISBN 978-3-631-57473-7. 38

[KWB+97] G. Kramer, B. Walker, T. Bonebright, P. Cook, J. Flowers, N. Miner, J. Neuhoff, R. Bargar, S. Barrass, J. Berger, G. Evreinov, W. T. Fitch, M. Gröhn, S. Handel, H. Kaper, H. Levkowitz, S. Lodha, B. Shinn-Cunningham, M. Simoni, and S. Tipei. Sonification Report: Status of the field and research agenda, 1997. 18

[KWB06] M. Kleimann-Weiner and J. Berger. The sound of one arm swinging: A model for multidimensional auditory display of physical motion. In *Proc. of the Int. Conf. on Auditory Display 2006*, pages 278–280. ICAD, June 2006. 153

[KWW03] O. Kerpa, K. Weiß, and H. Wörn. Development of a flexible tactile sensor system for a humanoid robot. In *Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)*, volume 1, page 6, 2003. 47, 109

[LCS03] W. Leahy, P. Chandler, and J. Sweller. When auditory presentations should and should not be a component of multimedia instruction. *Applied Cognitive Psychology*, 17:401–418, Feb 2003. 2

[LS04] J. Löwgren and E. Stolterman. *Thoughtful interaction design: A design perspective on information technology.* The MIT Press, 2004. 2

[Lut06] M. Lutz. *Programming python.* O'Reilly Media, Inc., 2006. 127

[LWW90] J. LeBlanc, M. O. Ward, and N. Wittels. Exploring n-dimensional databases. In *Proceedings of the 1st conference on Visualization '90*, pages 230–237. IEEE Computer Society Press, 1990. 69

[McC02] J. McCartney. Rethinking the computer music language: SuperCollider. *Computer Music Journal*, 26(4):61–68, 2002. 5, 11, 91, 110, 120, 135, 148, 161, 165, 173

[McK05] E. McKean, editor. *The New Oxford American Dictionary, Second Edition.* Oxford University Press, New York, 2005. 10, 12

[MM76] H. McGurk and J. MacDonald. Hearing lips and seeing voices. *Nature*, 264(5588):746–748, 1976. 52, 78

[Moo90] F.R. Moore. *Elements of Computer Music.* Prentice Hall, EngleWood Cliffs, 1990. 72, 73, 74

[MS94] T. H. Massie and J. K. Salisbury. The phantom haptic interface: A device for probing virtual objects. In *Proceedings of the ASME Winter Annual Meeting, Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, November 1994. 3

[Mye98] B. A. Myers. A brief history of human-computer interaction technology.

Interactions, 5:44–54, 1998. 32

[Nap19] J. Napier. *Mirifici logarithmorum canonis constructio.* Napier, Edinburgh, 1619. 29

[NDNG03] H. Newton-Dunn, H. Nakano, and J. Gibson. Block Jam: A tangible interface for interactive music. *Journal of New Music Research*, 32(4):383–393, 2003. 169

[Nov06] N. Nova. Tangible user interfaces: Misconceptions and insights. Presentation available at `http://tecfa.unige.ch/~nova/nokiatalk22092006.pdf`, September 2006. 42

[OCL04] S. Oviatt, R. Coulston, and R. Lunsford. When do we interact multimodally? cognitive load and multimodal communication patterns. *ICMI'04*, Oct 2004. 2

[OE04] S. O'Modhrain and G. Essl. PebbleBox and CrumbleBag: tactile interfaces for granular synthesis. In *NIME '04: Proceedings of the 2004 conference on New interfaces for musical expression*, pages 74–79, Singapore, Singapore, 2004. National University of Singapore. 119

[Par37] T. Parsons. *The Structure of Social Action.* McGraw-Hill, New York, 1937. 26

[PHB00] J. Paradiso, K. Hsiao, and A. Benbasat. Musical Trinkets: New pieces to play. In *SIGGRAPH 2000 Conference Abstracts and Applications, ACM Press, NY*, page 90, 2000. 46

[PI07] J. Patten and H. Ishii. Mechanical constraints as computational constraints in tabletop tangible interfaces. *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 809–818, 2007. 3, 49, 169

[Pul01] V. Pulkki. *Spatial sound generation and perception by amplitude panning techniques.* PhD thesis, Doctoral thesis, Helsinki University of Technology, Espoo Finland, 2001. 75, 113

[RdCW05] J. Rohrhuber, A. de Campo, and R. Wieser. Algorithms today: Notes on language design for just in time programming. In *Proceedings of International Computer Music Conference*, pages 455–458, Barcelona, 2005. ICMC. 12, 94

[Ree79] T. Reenskaug. Models-views-controllers. *Technical note, Xerox PARC, December*, 1979. 32, 55

[RF07] C. Reas and B. Fry. *Processing: a programming handbook for visual designers and artists.* The MIT Press, 2007. 91, 96

[Rie68] B. Riemann. Über die Hypothesen, welche der Geometrie zu Grunde liegen. *Abhandlungen der Königlichen Gesellschaft der Wissenschaftern zu Göttingen*, 13:1–21, 1868. 10, 15

[Rie09] E. Riedenklau. TAOs: Tangible active objects for table-top interaction. Master's thesis, Bielefeld University, June 2009. 68, 169

[Rit00] H. Ritter. The graphical simulation toolkit Neo/NST, 2000. 162

[riv] Rivers and tides: Andy goldsworthy working with time. DVD. Bonus Material: Leaf Works, Autumn, Scotland [1:33 – 3:20]. ix, 4, 27

[Roa85] C. Roads. *Foundations of Computer Music*, chapter 10, Granularsynthesis of sound, pages 145–159. MIT Press, Cambridge, 1985. 72

[Roa96] C. Roads. *The Computer Music Tutorial.* MIT Press, Cambridge, 1996. 73

[Roa01] C. Roads. *Microsound.* The MIT Press, Cambridge, 2001. 63, 72, 73

[Roh08] J. Rohrhuber. Implications of unfolding. In U. Seifert, J. H. Kim, and A. Moore, editors, *Paradoxes of Interactivity.* Transcript, Bielefeld, 2008. 1

[SC90] A.L. Strauss and J. Corbin. *Basics of qualitative research: Grounded theory procedures and techniques.* Sage Newbury Park, CA, 1990. 37

[She64] R. N. Shepard. Circularity in judgments of relative pitch. *The Journal of the Acoustical Society of America*, 36(12):2346–2353, 1964. 52

[Shn83] B. Shneiderman. Direct manipulation: A step beyond programming languages. *IEEE Computer*, 16(8):57–69, 1983. 31

[SLCGJ04] O. Shaer, N. Leland, E.H. Calvillo-Gamez, and R.J.K. Jacob. The TAC paradigm: specifying tangible user interfaces. *Personal and Ubiquitous Computing*, 8(5):359–369, 2004. 48

[Slo75] A. Sloman. Afterthoughts on analogical representations. In *TINLAP '75: Proceedings of the 1975 workshop on Theoretical issues in natural language processing*, pages 164–168, Morristown, NJ, USA, 1975. Association for Computational Linguistics. 21

[ST60] R. G. D. Steel and J. H. Torrie. *Principles and procedures of statistics.* 1960. 17

[Sti06] G. Stiny. *Shape.* MIT Press, Cambridge, Mass, 2006. 69

[Sut63] I. E. Sutherland. *Sketchpad, a man-machine graphical communication system.* Phd, Massachusetts Institute of Technology, 1963. 1, 31

[TEI08] TEI. *Proceedings of the Second International Conference on Tangible and Embedded Interaction (TEI'08)*, Bonn, Germany, February 18–20 2008. ACM. 45, 196, 197, 198

[TML+79] C.P. Thacker, E. M. McCreight, B.W. Lampson, R.F. Sproull, and D.R. Boggs. Alto: A personal computer. Technical report, Xerox PARC, 1979. 31

[Tra01] M. Travers. *Qualitative research through case studies.* Sage, 2001. 38

[Tuf83] E.R. Tufte. *The visual display of quantitative information.* Graphics Press Cheshire, CT, 1983. 76

[Tuk70] J. W. Tukey. *Exploratory Data Analysis*, volume 1 of *Addison-Wesley Series in Behavioral Science: Quantitative Methods.* Addison-Wesley, 1970. 15

[UI97] B. Ullmer and H. Ishii. The metaDESK: models and prototypes for tangible user interfaces. pages 223–232. ACM Press New York, NY, USA, 1997. 44, 47

[UI99] J. Underkoffler and H. Ishii. Urp: A luminous-tangible workbench for urban planning and design. In *Proceedings of CHI '99*, pages 386 –393, 1999. 45

[UI00] B. Ullmer and H. Ishii. Emerging frameworks for tangible user interfaces. *IBM Systems Journal*, 39(3 & 4):915 – 931, 2000. 2, 47, 48, 60

[VdG04] J. Vroomen and B. de Gelder. 9 Perceptual Effects of Cross-Modal Stimulation:

Ventriloquism and the Freezing Phenomenon. *The handbook of multisensory processes*, page 141, 2004. 78

[Ver05] P.-P. Verbeek. *What Things Do.* PSU Press, 2005. ISBN 978-0-271-02540-7. 32

[War94] M. O. Ward. Xmdvtool: Integrating multiple methods for visualizing multivariate data. In R. D. Bergeron and A. E. Kaufman, editors, *Proceedings of Visualization*, pages 326–333, October 1994. 69

[WB96] M. Weiser and J.S. Brown. Designing calm technology. *PowerGrid Journal*, 1(1):75–85, 1996. 45

[WCC09] S. Wilson, D. Cottle, and N. Collins, editors. *The SuperCollider Book.* MIT Press, Cambridge, 2009. in press. 5, 91, 110, 120, 135, 161, 173, 195

[Wei66] J. Weizenbaum. ELIZA: a computer program for the study of natural language communication between man and machine. 1966. 26

[Wei91] M. Weiser. The computer for the 21st century. *Scientific American*, 256(3):66–75, 1991. 31, 34

[WF97] M. Wright and A. Freed. Open Sound Control: A new protocol for communicating with sound synthesizers, 1997. 91, 150

[WFM03] M. Wright, A. Freed, and A. Momeni. OpenSound Control: State of the art 2003. In *Proceedings of the 2003 Conference on New Interfaces for Musical Expression (NIME-03), Montreal, Canada*, pages 153–159. NIME, 2003. 91, 150

[WGO+06] B. N. Walker, M. T. Godfrey, J. E. Orlosky, C. Bruce, and J. Sanford. Aquarium sonification: Soundscapes for accessible dynamic informal learning environments. In *Proc. of the Int. Conf. on Auditory Display 2006*, pages 238–241, June 2006. 153

[Wil06] J. Williamson. *Continuous Uncertain Interaction.* PhD thesis, University of Glasgow, 2006. 50

[WMG93] P. Wellner, W. Mackay, and R. Gold. Back to the real world. 1993. 33

[Xen71] I. Xenakis. *Formalized Music.* Indiana University Press, Bloomington, 1971. 72

[YFKS83] M. Yoneyama, J. Fujimoto, Y. Kawamo, and S. Sasabe. The audio spotlight: An application of nonlinear interaction of sound waves to a new type of loudspeaker design. *The Journal of the Acoustical Society of America*, 73(5):1532–1536, 1983. 75