Tquencer: A Tangible Musical Sequencer Using Overlays

Martin Kaltenbrunner
University of Art and Design
Linz, Austria
martin.kaltenbrunner@ufg.at

Jens Vetter
University of Art and Design
Linz, Austria
jens.vetter@ufg.at

ABSTRACT

This article discusses the design and implementation of the Tquencer, a tangible musical sequencer device. The tangible interaction paradigm for musical sequencers has been explored previously trough the arrangement of simple tangible tokens within time-based physical constraints. While this initial approach allows for the creation of basic rhythmic or melodic loops, it sometimes lacks the musical depth required for professional live performance. Therefore we introduce several additional physical design elements that allow the development of musical complexity, while maintaining the overall simplicity of tangible interaction. These tangible elements include single-step, multi-step and resizable tokens, which can be arranged to trigger musical events, multi-step sound effects and more complex sequence patterns. Furthermore the device also allows the dynamic assignment of the content and behavior of physical tokens through dedicated configuration tokens. We also introduce tangible overlays, which allow the handling and arrangement of multiple musical configurations in various layers. This token configuration and layer management is facilitated by an additional control step, which allows for the overall tangible configuration of a musical setup. Apart from its primary musical application scenario, the Tquencer also provides a generic tangible controller platform for time-based sequencing applications.

Author Keywords

Musical Instrument; Tangible Interaction; Interaction Design; Tangible Programming; Sequencer; RFID Sensing

INTRODUCTION

Token-based sequencing[10], which allows the arrangement of digital data embodied within physical tokens, is already a well established paradigm in tangible interaction design. Sequencing is conceptually grounded in the musical domain and has therefore been implemented in a vast variety of musical instruments, starting with the musical automata in the 18th century. Today this also includes several examples of tangible sequencer applications[1], since tangible interaction promised to resolve some of the performance limitations of contemporary digital music practice. Compared to commonly used software-based laptop instruments, tangible interfaces can provide a more expressive musical interaction bandwidth through direct physical manipulation to the performers, and therefore a more engaging performative experience to the audience. Nevertheless we can also observe that apart from a few exceptions such as the table-based modular synthesizer Reactable[7], most tangible musical interfaces often still lack the necessary musical depth for live performance, in spite of their innovative interaction design approach. Our Tquencer design therefore intends to provide a portable and low-cost musical device, which combines several advanced tangible interaction techniques in order to achieve sufficient musical complexity, while maintaining the overall simplicity of token-based interaction.

Our tangible interaction design primarily includes the sequential arrangement of physical tokens in combination with tangible overlays in order to expand the spatial limitations of a basic 8-step sequencer layout. We also focus on a fundamental tangible programming principle, which includes the static and dynamic assignment of musical content and functionality to tokens and overlays within the principal tangible domain. Therefore we provide single step tokens for musical events and resizable multi-step tokens e.g. for sound effects, as well as dedicated configuration tokens for dynamic content assignment. For the placement of these configuration tokens as well as the tangible overlays we expanded the eight sequencer steps with an additional control position.

RELATED WORK

There already exist several examples of tangible user interfaces, which individually implement some of the interaction design elements mentioned above. Verplank's et.al. early Tile-Sorting Board[11] from 1994 has been later also adapted to a musical application context. Bennet's Beat Bearing[2] device for example constitutes an elementary form of a tangible sequencer, where the positioning of uniform metal tokens within a physical multi-track configuration determines the content and timing of musical events. Hesse's Bubblegum Sequencer[5] on the other hand introduces colored tokens with statically linked musical content, which is independent of the vertical track position. Alonso's Scoretable[6] implements a circular token sequencer, which is a common alternative to the standard linear topology. Newton Dunn's BlockJam[8] design on the other hand allows the programming and arrangement of independent tangible sequencer blocks without the need for an external physical constraint. Costanza's Audio D-Touch[3] allows the swift exchange of various sequencer

TEI '18, March 18–21, 2018, Stockholm, Sweden

ISBN 978-1-4503-5568-1/18/03…$15.00

DOI: https://doi.org/10.1145/3173225.3173233

topologies by using application sheets. A further non-musical example for the sequential tangible programming paradigm is Gallardo's Turtan[4], which transfers the procedural Logo programming language into the tangible domain. Ullmer's Casiers[9] introduce the concept of tangible overlays, which constitute context-dependent and exchangeable physical constraints for interactive surfaces. This concept has been recently also adopted by the Sensel Morph[1] device, which provides various controller overlays for its force-sensitive surface.

HARD- AND SOFTWARE DESIGN

The central interactive element of the Tquencer device consists of an array of eight+one RFID reader modules, which are each capable of detecting the simultaneous presence of two ISO14443 (Mifare) tags operating at 13,561 MHz. The individual RFID reader modules are based on the MFRC-522 chipset, already include a PCB antenna and are interconnected through a serial SPI bus configuration. In order to avoid cross-detection problems between nearby readers, we employ a time-multiplexed sensing method by sequentially activating and reading from each module individually. The actual number of usable RFID tags is virtually unlimited, and we can statically (by design) and dynamically (by configuration) assign an individual tag UID to the actual token functionality.

Hardware Components

The overall interactive input and visual feedback is managed by an Arduino Micro (Leonardo compatible) micro-controller board, which includes an ATmega32U4 running at 16MHz. We use almost all available I/O pins for connecting the various hardware elements. As mentioned above the RFID reader array is connected through the SPI bus and uses nine additional digital selector pins. A complementary capacitive touch controller based on the MPR121 chipset is connected via the I2C bus and provides twelve individual touch points. Furthermore we use three analog pins for connecting the potentiometer knobs required for the control of global parameters. The visual feedback is provided by an array of nine Neopixel RGB LED modules, which are connected serially through single digital pin. The whole device setup is powered by a 5V2A power supply connected to a Micro USB port.

The micro-controller board processes the input data from all connected modules in real-time and also calculates the according states and events. This includes the detection of presence and removal of all RFID tokens, the analysis of capacitive touch button gestures as well as the precise sequencer timing and volume control from the potentiometer values. The resulting control values, events and states are encoded and transmitted through a custom serial protocol. Thus the micro-controller board already provides the full I/O functionality and can operate independently.

For the integrated musical functionality we employ a NanoPi Single Board Computer (SBC) running Ubuntu 16.04 with a real-time Linux Kernel. This devices sports a quad core ARM CPU running at 1GHz and 512MB RAM, and therefore provides sufficient computing power for real-time sound synthesis at a low cost and embedded form factor. The board also provides the necessary integrated sound output as well as an onboard Ethernet port, which we use for overall configuration purposes and external control capabilities. Additional USB ports also allow the connection of arbitrary USB devices, such as external sound cards or MIDI converters, while the integrated Micro USB port is used for the power supply. The overall internal hardware components are shown in Figure 1.

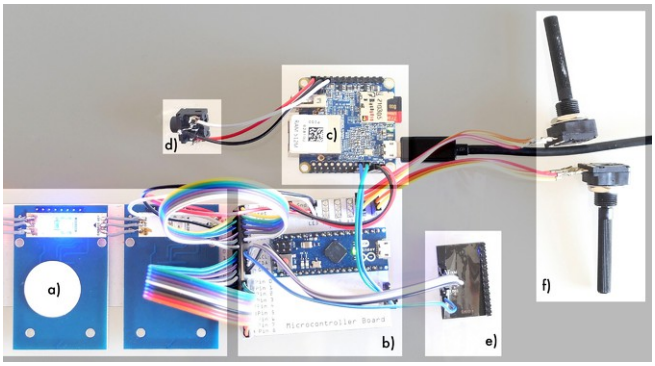

Figure 1. The internal hardware components of the Tquencer: a) RFID array b) micro-controller c) single board computer d) audio output e) touch controller f) controller knobs

Software Components

In conjunction with the basic I/O firmware on the micro-controller, we developed a simple background application, which constantly reads the provided interaction data from the serial port and translates the according states and events to a custom protocol based on Open Sound Control.[12] The according OSC messages are primarily used for the control of the internal synthesizer, and are simultaneously broadcasted to any device connected via Ethernet for alternative external control purposes.

The central synthesizer engine of the Tquencer has been implemented using the Supercollider[2] sound programming language, which not only provides the core musical sequencer functionality but also allows the design of simple programmable extensions. The audio output is managed by the Jack audio daemon, which in conjunction with the real-time Linux kernel ensures a minimal audio latency.

We also employ various integrated Linux system components for overall configuration purposes, such as the Samba daemon for file management via Ethernet as well as the Lighttpd server for a web-based configuration interface. Connected devices are automatically configured using DHCP and Zeroconf.

TANGIBLE INTERACTION DESIGN

The core musical interface of the Tquencer consists of a simple eight-step sequencer design, where the basic sonic events for each individual step can be configured through the placement of a tangible token. Due to the technical constraints of the chosen RFID sensing method described above, each step can detect two simultaneous tangibles, allowing a total of 16 musical events within each loop. While this allows the straightforward tangible programming of simple musical sequences

[1] http://www.sensel.com

[2] https://supercollider.github.io/

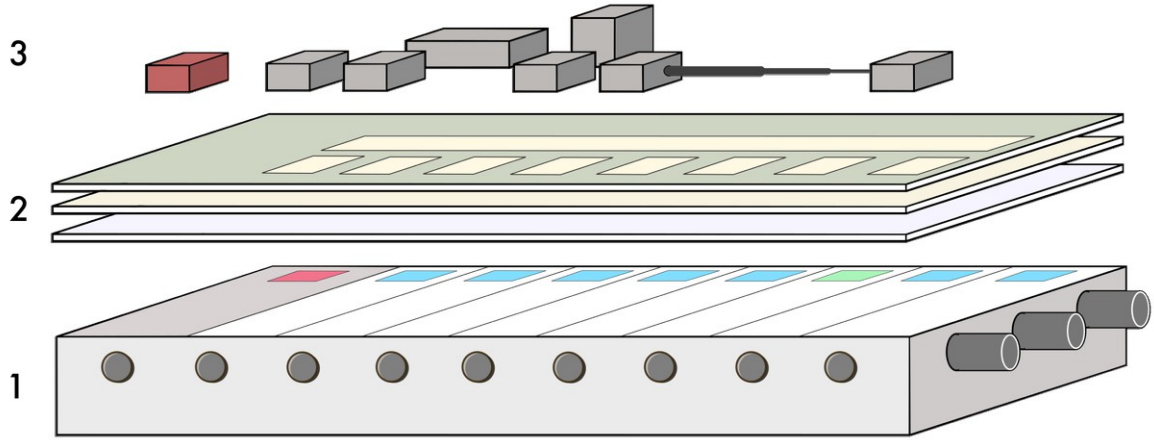

Figure 2. Diagram of interactive elements of the Tquencer: 1) sequencer 2) overlays 3) tokens

such as drum and melody loops, this of course only provides a rather minimal musical functionality in its basic configuration. Therefore we introduced several additional tangible design elements, which allow for the development of more complex musical structures, while maintaining the overall simplicity.

Tokens

From a musical point of view the Tquencer employs three different types of tokens: Event tokens, Effect tokens and Pattern tokens. Event tokens trigger simple musical events, such as sample playback, midi notes and algorithmic sound generators. Effect tokens on the other hand apply sound effects to single or multiple musical events, such as reverb, delay and distortion. Pattern tokens are altering the course of the sequencer sweep, such as reverse, repeat and local loop. In addition to that, we also provide dedicated configuration tokens, which allow the dynamic content assignment to Event and Effect tokens.

From a physical point of view, we provide three different token designs: Single-step tokens, Multi-step tokens and resizable Multi-step tokens.(see Figure 2.3) This allows the physical assignment of musical events and effects to single or multiple steps using a single token only, which covers the selected area. Resizable tokens are an extension of rigid two- and three-step tokens allowing the configuration of four- to eight-step tokens.

Overlays

Since a simple eight-step sequencer with a static configuration would not provide the necessary musical depth for professional musicians, we designed a physical approach for the selection of alternative musical programs and the management of multiple musical layers. We therefore employ physical overlays in the form of transparent plastic sheets, which are tagged with an RFID sticker on one end.(see Figure 2.2) Placing this overlay on the device with its sticker over the dedicated configuration step, thus will select the according musical program that has been associated with this overlay, such as a drum computer, note sequencer or more complex algorithmic behaviors.

Hence these overlays constitute a physical form of tangible windows, which allow the development of multi-layered musical structures, while maintaining the overall simplicity of our token-based sequencer design. Once such a configuration (e.g. drum loop) has been arranged, it can be converted to a continuous background loop by pressing a dedicated hold-button while removing the overlay including all active tokens. In order to facilitative the distinction and handling of the various overlays, we use differently colored sheets, was well as graphical (printed) and physical (engraved) design elements.

On the other hand the physical persistence of such a token-overlay configuration also imposes some conceptual constraints, since a previously configured background loop cannot be physically called back into operation. Thus we only can delete previous iterations of such a configuration, and have to rearrange a particular overlay if any changes are required.

Controls

In addition to the core token-overlay interaction, the Tquencer also provides several complementary controls, which consist of three lateral knobs (potentiometers), three lateral touch buttons (capacitive) as well as nine frontal touch buttons for each individual step (eight steps plus configuration step).

Two of the controller knobs are statically linked to the global tempo and volume controls, while the third knob is dedicated to a layer-specific global effect control, such as a reverb for example. The frontal touch buttons allow for additional gestural control of the tangible sequencer content, such as keeping an individual Event-token after removal (hold) or deleting all virtual content (double tap). The same touch gestures (hold and double tap) are applied to the button in front of the configuration step for the holding and deleting of overlays. The additional three lateral buttons are reserved for global configurations and mode changes.

Feedback

In order to facilitate its overall handling and interaction stability, the Tquencer also provides basic visual feedback in the form of an individual RGB color LED for each step including the configuration step. At the moment we defined several

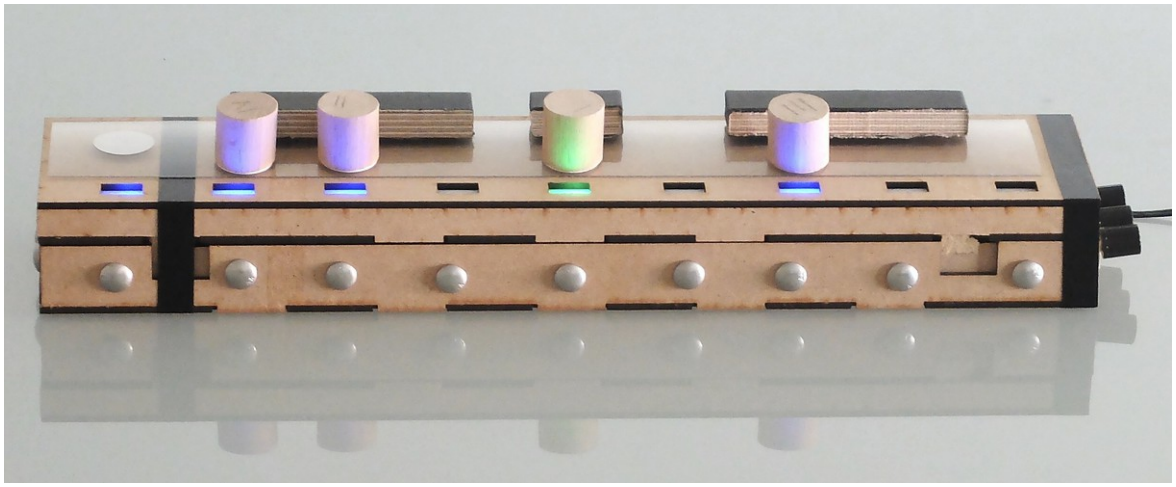

Figure 3. The current iteration of the Tquencer hardware prototype.

color codes which represent the various states of each individual step. This includes the basic feedback of the number of recognized tokens as well as the visualization of the moving sequencer sweep. The configuration step changes its color according to the currently active physical overlay.

As a musical instrument the Tquencer naturally provides acoustic feedback for all interactions, depending on the currently active musical configuration. It does not provide any additional acoustic feedback (such as tempo ticks), since this would interfere with the overall musical experience. Due to its asynchronous sequencer design, it also does not provide immediate acoustic feedback for each interaction, such as the placement of a new token. Therefore the visual cues play an important role for the real-time feedback for these interactions as well as for the overall tempo representation.

MUSICAL INTERACTION DESIGN

The Tquencer has been primarily designed as a standalone musical instrument, but can be also used as controller for external musical equipment, such as software applications (via OSC), digital synthesizers and effect devices (via MIDI) as well as analog synthesizers (via CV). In the following we will concentrate on the musical interaction design of the provided internal synthesizer, which is based on SuperCollider.

Tangible Programming

The initial default configuration after switching on the Tquencer is a generic drum sequencer. The sequencer loop starts with a default tempo, which is visualized by the moving sweep on the LED array, and can be adjusted through the primary tempo potentiometer on the right side.

Generic Sequencing

In this configuration all available tokens can be used to trigger various drum samples. Placing an Event token on an individual sequencer step will trigger the according pre-assigned drum sound as soon as the tempo sweep crosses that position. The overall arrangement of these drum tokens therefore allows the creation of simple rhythmic structures, with a maximum of two drum events per step. This physical control allows the direct manipulation of all present drum events, removing or adding further Event tokens affects the according drum sequence and results in immediate rhythmic changes.

Effect Sequencing

In addition to the generic sequencing of musical events, the Tquencer also allows three different ways to apply additional sound effects to the complete or parts of the sequence: Dedicated Effect tokens apply additional musical effects (such as delay or distortion) at their actual position in a sequence. Single-step Effect tokens only apply to an individual step, while multi-step Effect tokens cover a certain range of the sequence. Resizable multi-step Effect tokens allow for the rapid adjustment of the effect range. A global sound effect (such as reverb) can be adjusted with the dedicated effect potentiometer on the right side, while the frontal touch buttons can be used to de/activate that global effect for an individual step.

Pattern Sequencing

For the creation of more complex sequence topologies we provide a collection of preconfigured Pattern tokens, which are distinguishable from Event and Effect tokens by their shape. Pattern tokens can affect the course of an entire sequence with behaviors such as reversing, randomizing and resizing the overall sweep, but can be also employed locally for the creation of loops and repetitions.

Layer Management

As we can see, the maximum amount of simultaneously usable Event, Effect and Pattern tokens is significantly limited by the spatial and technical constraints of our basic device layout. Tangible overlays therefore allow the configuration and handling of various musical behaviors, which can be easily exchanged by the placement of an accordingly marked transparent sheet onto the device surface. Currently there are three default overlays available: a drum computer, a melody stave and an algorithmic synthesizer, which can be individually configured as explained further below. Additional blank

layers can be configured for the development of complex effect sequences and pattern behaviors in order to liberate the active tangible configuration for musical interaction. Holding the touch button in front of the configuration step when removing an overlay, allows to keep any musical content in the background, while new sequences can be arranged on top of these existing loops. Double-tapping the according touch button erases any background content, and only physically present tokens will continue to trigger musical output.

Token Assignment

While we can distinguish the fundamental token classes by their physical design (currently cylinders for Event tokens, cubes for Effect tokens, and hemispheres for Pattern tokens), the creation of a physical representation for token subclasses (for example based on color) would result in a very large collection of individual tokens to represent all possible musical content or sound effects. Therefore only Pattern tokens are actually hardcoded to their specific functionality, while we rely on the dynamic content association for Event and Effect tokens. Placing a special Programming token onto the configuration step turn the device into programming mode, which allows the assignment of context (layer) dependent musical content to Event tokens or individual sound effects to Effect tokens. Placing an Event or Effect token onto the device, will in this case assign the predefined content or effect of the selected slot to the token. This configuration is maintained after the removal of the Programming slot as is only valid in conjunction with the currently present overlay.

Software Configuration

While the core musical interaction and the fundamental configuration of the Tquencer can be fully handled within the tangible domain, we provide various additional configuration layers for the individualization and customization of the internal synthesizer.

Alternative musical content such as samples and sound fonts can be simply uploaded to the device via the integrated file sharing when connected to an external computer through Ethernet. This content can be then associated to the eight slots of an individual overlay via a simple web-based configuration interface, and then subsequently assigned to an individual token using the method described above.

Advanced users will be also able to extend the functionality of the internal synthesizer through a dedicated plugin interface for the realization of additional sound effects or algorithmic synthesizer elements. Since the actual source code of the complete synthesizer component is also exposed via file sharing, we also allow its full customization if required.

Performance Observations

As mentioned above tokens are placed on the surface of the Tquencer in order to arrange a musical sequence. Since this is a very direct way to generate and manipulate sequences, playing the Tquencer feels organic and familiar. This tangible approach to musical intervention immediately encourages advanced physical interactions, such as grabbing multiple tokens at once and moving them step by step. Involving the touch buttons e.g. for additional effects fits in easily, even while moving a token on the surface. The tempo controlled by the tempo potentiometer ranges from 30 bpm to 240 bpm, which allows for radical changes in the musical output. While may initially be a challenge to keep track of the various tokens, it quickly becomes clear that the interaction based on arranging tokens is very intuitive and playful.

Connecting external equipment (for example via MIDI, OSC) works simple and uncomplicated. Here the Tquencer can be used as a MIDI-sequencer and tempo-sync master device. The individual tokens then trigger MIDI notes, which can be assigned pre-configured either in the programming mode, or individually in the web-based configuration interface (see below). This allows software synthesizers, digital audio workstations, analog synthesizers and other devices to be included. Using the Tquencer along with other musical equipment and instruments is probably more common from a musical and performative point of view, rather than considering it as a standalone solo instrument.

TANGIBLE INTERACTION PLATFORM

The overall modular design of the Tquencer also allows its usage as a generic tangible controller device for arbitrary time-based sequencing application scenarios. Although our current configuration includes an integrated synthesizer for our specific musical application, the generic OSC encoding of the overall sequencer state and user interactions also allows for the control of alternative internal (on the SBC) or external applications (via Ethernet). One possible example would be the control of a DMX driven light performance through tangible sequencing.

Following the Open Design philosophy we are also planning to provide all software and hardware elements including a comprehensive documentation of the Tquencer internals under an open licensing scheme. This includes the open source code of the micro-controller firmware, interaction daemon and synthesizer application as well as the open hardware design of the employed PCB boards and other physical components.

This open design approach not only allows the full reproduction of our device, but more importantly will also facilitate modifications of our original design for customized application scenarios as well as the improvement of our design itself by the community.

CONCLUSION AND FUTURE WORK

We presented the hardware, software and interaction design of the Tquencer device. The current iteration of this tangible musical sequencer is already quite mature, but is still subject to ongoing development and improvement both from a technical and musical perspective. The future work will mainly concentrate on the performative aspects of our tangible interaction design, which will most-importantly include its on-stage evaluation in a live performance scenario. Furthermore we are planning to refine the overall hardware design of the device towards production quality, including the possibility of a subsequent crowd-funding campaign.

REFERENCES

1. R. Arar and A. Kapur. 2013. A History of Sequencers: Interfaces for Organizing Pattern-Based Music. In *Proceedings of the Sound and Music Computing Conference (SMC2013)*. Stockholm, Sweden.
2. P. Bennett. 2010. The Representation and Control of Time in Tangible User Interfaces: (Summary of PhD Research). In *Proceedings of the Fourth International Conference on Tangible, Embedded, and Embodied Interaction (TEI '10)*. Cambridge, Massachusetts, USA.
3. E. Costanza, S. B. Shelley, and J. Robinson. 2003. D-touch: A Consumer-Grade Tangible Interface Module and Musical Applications. In *Proceedings of the Conference on HumanComputer Interaction (HCI03)*. Bath, UK.
4. D. Gallardo, C. F. Julià, and S. Jordà. 2008. TurTan: A tangible programming language for creative exploration. In *Third IEEE International Workshop on Tabletops and Interactive Surfaces (Tabletop 2008)*. Amsterdam, The Netherlands.
5. H. Hesse and A. McDiarmid. 2008. *The Bubblegum Sequencer*. Technical Report. UC Berkeley School of Information.
6. S. Jordà and M. Alonso. 2006. Mary Had a Little scoreTable* or the reacTable* Goes Melodic. In *Proceedings of the 2006 Conference on New Interfaces for Musical Expression (NIME '06)*. Paris, France.
7. S. Jordà, G. Geiger, M. Alonso, and M. Kaltenbrunner. 2007. The reacTable: Exploring the Synergy Between Live Music Performance and Tabletop Tangible Interfaces. In *Proceedings of the 1st International Conference on Tangible and Embedded Interaction (TEI '07)*. Baton Rouge, Louisiana.
8. H. Newton-Dunn, H. Nakano, and J. Gibson. 2002. Block Jam. In *ACM SIGGRAPH 2002 Conference Abstracts and Applications (SIGGRAPH '02)*. San Antonio, Texas.
9. B. Ullmer. 2011. Casier: Structures for Composing Tangibles and Complementary Interactors for Use Across Diverse Systems. In *Proceedings of the Fifth International Conference on Tangible, Embedded, and Embodied Interaction (TEI '11)*. Funchal, Portugal.
10. B. Ullmer and H. Ishii. 1999. mediaBlocks: Tangible Interfaces for Online Media. In *CHI '99 Extended Abstracts on Human Factors in Computing Systems*. Pittsburgh, Pennsylvania.
11. M. Withgott. 2015. What Do Objects Mean?: Early Tangibility in and Around Interval Research. In *Proceedings of the Ninth International Conference on Tangible, Embedded, and Embodied Interaction (TEI '15)*. Stanford, California, USA.
12. M. Wright, A. Freed, and M. Ali. 2003. OpenSound Control: State of the Art 2003. In *Proceedings of the 3rd Conference on New Instruments for Musical Expression (NIME '03)*. Montreal, Canada.