(and Sound) of SiMPE: Showcasing Outcomes of a Mobile Audio Programming Seminar

Cumhur Erkut
Medialogy
Aalborg University Copenhagen
Denmark
cer@create.aau.dk

Antti Jylhä
Dept. Computer Science
University of Helsinki
Helsinki, Finland
antti.jylha@helsinki.fi

Stefania Serafin
Medialogy
Aalborg University Copenhagen
Denmark
sts@create.aau.dk

ABSTRACT
As contemporary smartphones become ubiquitous, their nature changes from mere function to a tool for self-expression, creativity, or play. Mobile application development embraces this change, as evidenced by the popularity and profits of sound and music applications and games in various application stores. As proposed in the ACM Curriculum report, these application domains are attractive in education, especially in computer science and interaction design. The main question is how to systematically integrate the rapidly evolving knowledge, know-how, tools, and techniques of mobile (audio) programming and interaction design into university curricula. In this paper, we provide an account of our own development and teaching experience. We highlight the outcomes, which are exploratory applications challenging the state-of-the-art in mobile application development based on non-speech sound. The positive assessment of the participants and the appreciation of data providers are among the results that encourage in future development. Yet, there is room for improvement in combining the technical development with the design practices and user studies.

Author Keywords
Mobile Computing, Mobile Audio Programming, Sound, Sonic Interaction Design, Sound and Music Computing

ACM Classification Keywords
H.5.2. [Information interfaces and presentation]: Auditory (non-speech) feedback, Voice I/O.

INTRODUCTION
It has been more than a decade after the influential work AudioGPS by Holland and colleagues [1], which showed the mobile human-computer interaction (HCI) community how to design and evaluate location-aware, minimal-attention mobile interfaces utilizing non-speech sound. While this line of research is being actively pursued [2], there has been increasing focus on continuous, direct manipulation of the mobile device by human action, building on the early experiments in HCI [3], [4].

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
SiMPE' 13, a MobileHCI '13 WS, August 27, 2013, Munich, Germany
Copyright remains with the authors.

Fresh perspectives are gained with the closure of the interaction loop by multi-sensory feedback, where the non-speech sound is a distinctive component that guides the human action [5]. Sonic interactions demand in some cases attention, skill, and effort from the users, but these resources are justified with the opportunities in engagement, self-expression, creativity, or play. Developers need to be aware of these demands, besides the opportunities in using non-speech audio in mobile contexts. In addition, they need to combine expertise in mobile audio technology, programming and design.

Right at this interaction, we face a bottleneck similar to what has been manifested by Matthew Aylett (available at SiMPE wiki) in the context of combining speech technology and design: guidance is usually available in one or the other, but rarely in both. In our experience, the best way of nurturing the mobile audio technology/sonic interaction design intersection is to train the future developers early within multidisciplinary, project-based settings [6].

This paper showcases the outcomes of our training activity: a seminar on Mobile Audio Programming (MobiAP). Originally, the activity was intended as a first step in technology-design dialog and iteration, for constructing the most-advanced (yet acceptable) mobile audio processing codebase for subsequent build-up in the media technology and design curriculum. 13 seminar participants presented their selected topics at the final event of the seminar, on December 9, 2011 at the Aalto University, School of Electrical Engineering, and seven of them have completed their special assignment in 2012. They have demonstrated their projects on a mobile platform (or simulator) of their choice, which make up the showcases presented in this contribution.

We pursue several objectives in bringing these projects and demonstrations into the spotlight during the SiMPE workshop. First, the outcomes indicate, and in some case challenge the state-of-the-art in mobile application development based on non-speech sound. Yet, their design refinement and evaluation have not been completed. Our second objective, therefore, is to discuss with the Mobile HCI community how to extend them to interactive mobile applications that are meaningful for their users. Finally, we want to tackle the joint challenge of combining technology and design with the mobile speech community, and discuss the education opportunities at their intersection. Most applications will be demonstrated live or by video clips during SiMPE'13.

BACKGROUND: THE MOBIAP SEMINAR

When two authors of the present paper set to organize a seminar on mobile audio programming in 2011, the availability and substance of related work for a full-term course for the graduate and post-graduate students was an issue. The seminars typically require literature reviews of tutorials and state of the art papers, and other guidance so that the participants can read and understand the technical literature and develop their scientific writing and presentation skills. We have extended these requirements with practice-based modules from popular mobile music courses (e.g., Stanford Music 256b), and set with the participants the following learning objectives:

- to understand fundamentals of audio programming and how they can be utilized on different mobile platforms
- evaluate the opportunities and challenges of developing audio-based applications on mobile devices
- compare different control standards, such as OSC [7] and TUIO [8].
- tackle more advanced topics, such as streams, pipelines, multimedia frameworks, and specialized applications.

These objectives were aligned with the background and interests of the participants, resulting in seminar themes, as illustrated on Fig. 1. The first theme, *Interactive Audio Engines* includes mobile ports of interactive software such as pure data (PD) and STK, as well their extensions, such as libPD, RJDJ, MoMu or uRMUS.

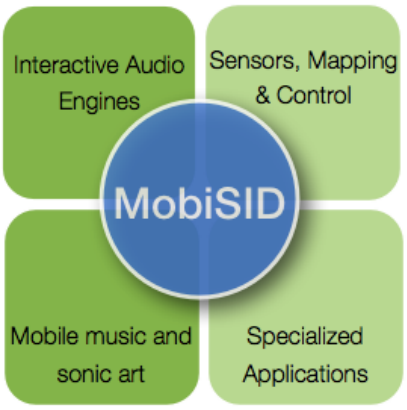

Figure 1. Organization of the Seminar.

Mobile music and sonic art includes music recognition by audio fingerprinting, mobile, collaborative, or networked music performance. *Sensors, mapping and control* includes a closer look at the affordances of mobile environment, e.g., accelerometer or microphone, both as a device and API. A special focus here was voice- and audio-driven applications, such as a mobile vocoder. *Specialized applications* include mobile audio-based context, activity, and environment recognition, way-finding and navigation, mobile game audio (procedural audio and effects).

The seminar papers that summarize state of the art in these themes are distributed under the terms of the Creative Commons Attribution License 3.0 Unported[1]. In addition, each post-graduate participant has completed a mini-project, complete with source-code, a working application on the device or simulator, and documentation. Most applications span multiple dimensions previously shown on Fig. 1, and used in the first column of Table 1, indicating the richness of the design space.

INTERACTIVE AUDIO ENGINES

When the mobile SDKs first available, the audio engines were mostly ported from the related desktop SDKs (see the History of iOS audio). In time, more power and interactive provisions were added. The popular interactive engines were ported on mobile platforms as native applications or libraries. A good example of this direction was libpd [9], a mobile implementation of the open-source visual programming language Pure Data (PD). The library has received interest and utilization during the contact lectures and practical work. In final implementations, however, the iOS support of OpenFrameworks (oF) has prevailed. oF is an open source C++ toolkit for creative coding, and three seminar apps were developed using this toolkit. In contrast, only one application has linked to libpd.

More recently, higher-level PD implementations became available (e.g., MobMuPlat and PdParty, both at GitHub with open source licenses). In addition, there has been two different implementation of a low-latency audio bus between separate applications on iOS: 1)AudioBus and 2) Jack iOS. While these special, efficient, but complicated APIs have pointed out many opportunities to the audio community, web applications with the help of HTML5 or scripting presents an alternative implementation direction for basic interactive use cases, where especially when latency is not a critical issue.

Main Theme	App/Project Name	Developer	Platform	Language(s)	Short Description
Interactive Audio Engines	Permafrost Backend	S. D'Angelo	Android	JavaScript	Web-based Audio DSP compiler
Sensors, Mapping & Control	Wdf2Go	R. Paiva	Android	Java	Mobile WDF tree
	BirdHound	R. Albrecht	Android	Java	Birdsong / bird sound recognition
	Tabula Rasa Augmentata	R. Pugliese	iOS	libpd	Sonic augmentation of surfaces
Mobile Music and Sonic Art	App for 18 Resonators	J. Parker	iOS	C++, oF	AV evolving sequence/drone synth
Spatial Applications	soundflux	A. Politis	iOS	C++, oF	Detection and display of sound direction
	DMS	S. Delikaris-Manias	iOS	C++, oF	A Double Middle-Side Decoder

Table 1: A short summary of the completed projects.

[1] Proceedings of Mobile Audio Programming Seminar Fall 2011, edited by Cumhur Erkut, Antti Jylhä, and Jussi Pekonen, http://blogs.aalto.fi/mobiap.

Permafrost Backend, within this context, is an experimental prototype back-end for a generic DSP compiler that takes a relatively low level description of a DSP module and can translate it to Matlab/GNU Octave code, plain C code and/or source code of an Android application. The compiler is written entirely in JavaScript. On a browser, an html page downloads the scripts (support for Data Flow Graphs, scheduler, and code generators for plain c, MATLAB, and android, see Fig. 2); scripts then parse the DSP elements, associate their parameters, and generate the desired code. The fundamental data structures in Permafrost Backend, as well its support classes are Audio DSP-based. This is a characteristic in many showcases presented here, and is related to the background and research/study interests of the participants.

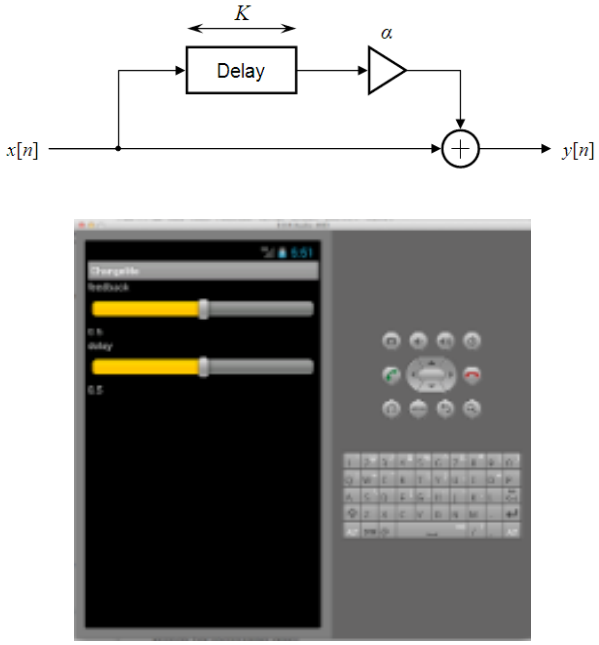

Figure 2. A simple comb filter with two parameters (top) implemented with scripting with Permafrost Backend, compiled, and launched on Android simulator (bottom).

SENSORS, MAPPING AND CONTROL

WDF2Go, is, to our knowledge, the first mobile implementation of a parametric block-based physical model utilizing wave-digital filters [10]. Like the Permafrost Backend, the compiled application makes use of standard mobile GUI widgets (buttons and sliders, see Fig. 2), as managed by the *HelmholtzTree* class. This class is an interface to the Helmholtz tree library implemented in C++, and sends control messages for modifying synthesis parameters in the library.

The user interface has 4 states: *START, PLAY, EDIT, and EDITAUDIO*. The *START* state initializes the native Helmholtz tree implemented in C++, as well as the audio engines used for controlling audio output. *PLAY* initiates the user interface shown on Fig. 3 (bottom). Here, it is possible to hit individual resonators (buttons H00, H10, etc.) as well as to apply an external excitation. *EDIT* controls the resonance parameters of each resonator individually. When entering this mode, the excitation set in the *PLAY* keeps running, and each resonator can be tuned by the user. The tuning parameters include the cavity volume, neck length and neck area. Additionally, the air density was included as a global parameter, i.e. when changing the air density, all the resonances will change. For all the resonators, the resonant frequency is displayed in order to help the user tuning its parameters. Within the *EDITAUDIO* state, the user is able to change the global parameters related to the audio interface (sampling frequency and the frame size).

Microphones and accelerometers

Besides the typical smartphone input modality of multi-touch and GUI, microphone(s) and accelerometer were also used in most applications. The two apps, **BirdHound** and **Tabula Rasa Augmentata**, which utilize the internal microphone for audio analysis and machine learning, are reported here. Other applications that listen to the accelerometer or external microphones are presented in the subsequent sections.

BirdHound is for identifying birds based on their song, simply by using a mobile phone. The algorithm used by BirdHound consists of three parts: segmentation, feature extraction by using mel-frequency cepstral coefficients (MFCC), and k-nearest-neighbors (kNN) classification. The segmentation is based on a short-time signal-energy-based method, which is explained in detail in [11]. In feature extraction, MFCCs are extracted from each segment and used as features for classification. In addition to MFCCs, different spectral and temporal features could also be used. Some preliminary tests were done using, e.g., the spectral centroid and zero-crossing rate in this project, showing a slight increase in the recognition rate compared with only using MFCC features. Classification has been carried out by a kNN classifier.

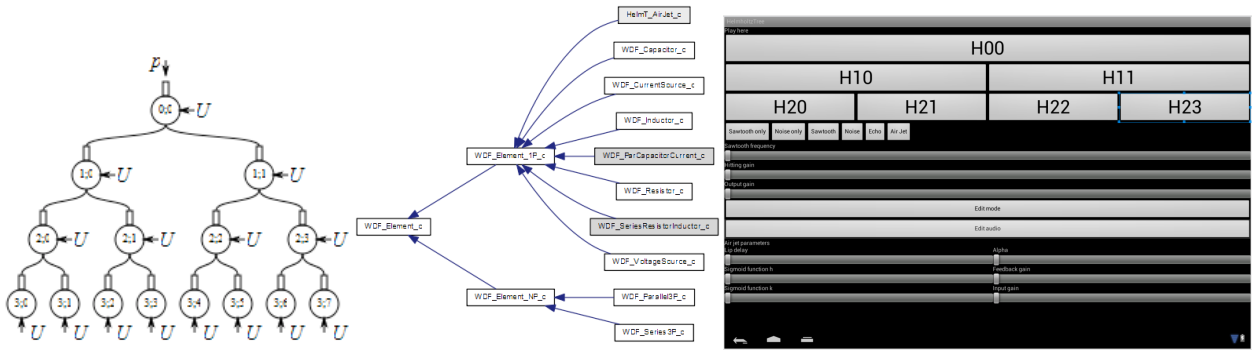

Figure 3. WDF2Go: a mobile, fully-parametric block-based physical modeling implementation utilizing wave digital filters. The library is developed in C++, and wrapped on Android OS [12]**.**

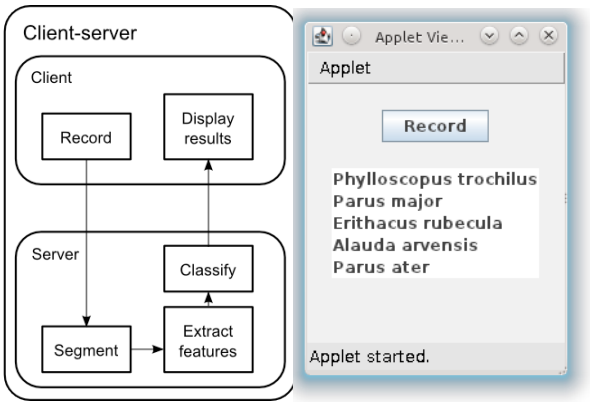

Figure 4. Client-server implementation of BirdHound (top), simulation with a desktop client (bottom).

A client-server architecture has been chosen for implementation (Fig. 3) The client consists of a button for starting to record and for stopping the recording and identifying the recorded birdsong. After recording, the audio is sent to the server, which sends back a list of the five most commonly identified birds among the syllables found in the recording. The client displays them all (there might be more than one bird in the recording). The server can be run in training, test, or regular operation modes, with a possibility saving and loading the trained classifier, so that it does not have to be retrained each time the server application is started.

Originally planned for Android OS, BirdHound is demonstrated on desktop, because mainly the server carries the computational load, and the Android simulator did not allow real-time audio input. The future work will focus on contextual (sonic) interaction design to improve the usability and user experience.

Tabula Rasa Augmentata (TRA) investigates the sonic augmentation of daily objects by the use of the audio-input available on mobile phone. The focus is on audio-tactile interaction with daily objects as input for music creation by sonically augmenting surfaces. The audio and control flow in the application is illustrated on Fig 5 (top).

Following the audio input, TRA uses audio-tactile input to capture and recognize different *sonic gestures* [13]. It then implements audio-driven sound synthesis. Rather than triggering a set of samples, continuous tracking of sonic gestures provide control parameters to an audio-synthesis module. It includes a machine learning strategy that recognizes the intention of the user in terms of sonic gestures that keeps the sound very tight to the *tangible interaction* with the material properties of the augmented surface (roughness, hardness, hollow, etc.). This (near) real-time, tightly-coupled sonic interaction is intended to create an illusion of transforming the audio-tactile properties of the surface.

MOBILE MUSIC AND SONIC ART

App for 18 Resonators is an audiovisual sequence/drone synthesizer for iOS, built with openFrameworks. The synthesis employs a large parallel bank of 18 resonators, each excited by a pulse train at a particular frequency. The pulse train frequencies are offset slightly over the bank to produce interesting 'phasing' effects. Each pulse train is visualized by a moving block on the screen. At higher frequencies, interesting patterns appear in the collection of blocks, which mirror the changes in the synthesized sound.

A number of red bars are overlaid on the screen, which allow the user to draw in distributions of frequencies for the resonator bank. The accelerometer is used to manipulate various parameters of the system, with care taken to make the interaction feel sufficiently rich.

The core of the DSP structure is a library of C++ classes containing various DSP primitives. The library allows appropriate structures to be re-used for purposes other than direct audio generation or processing. For example, a lowpass filter class can be re-used to provide control smoothing at, for example, the screen refresh rate. For this project, only a small number of primitives were necessary - a simple phase counter, an appropriate resonator filter structure, and a simple leaky integrator. A transposed direct form II biquad filter is used to implement the resonators.

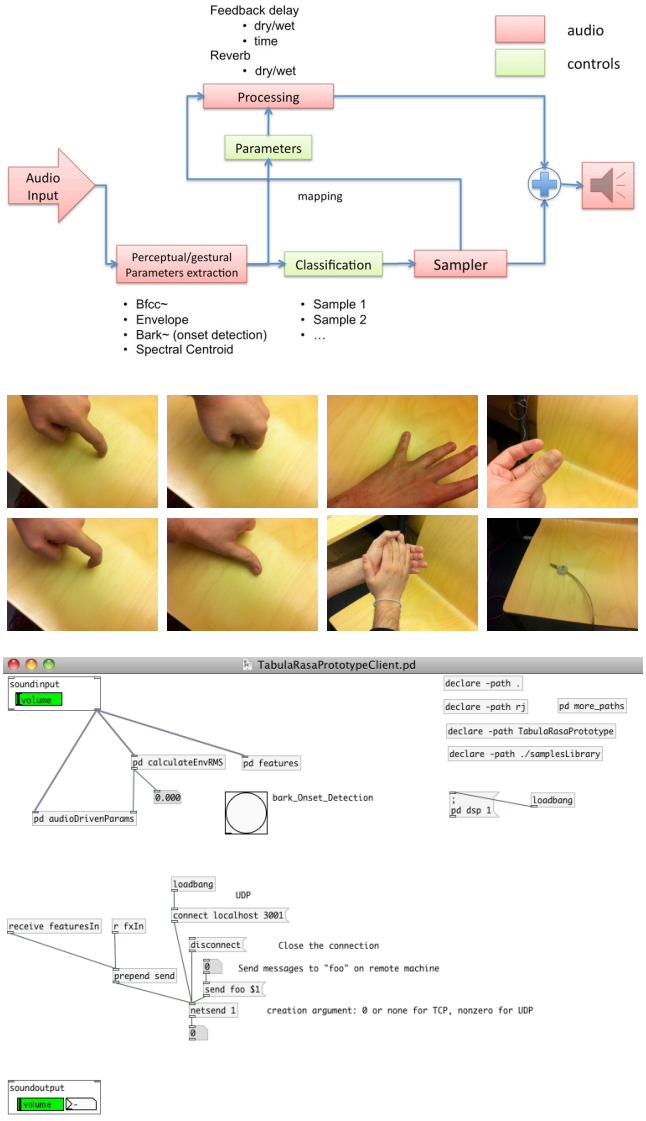

Figure 5. (Top) The audio and control data flow of TRA. (Middle) Seven sonic gestures detected by the application, together with the small mic used in prototyping: finger tap, knock, nail tap, thumb, little finger, snap, and clap. (Bottom) Client to be ported to the mobile device via libpd.

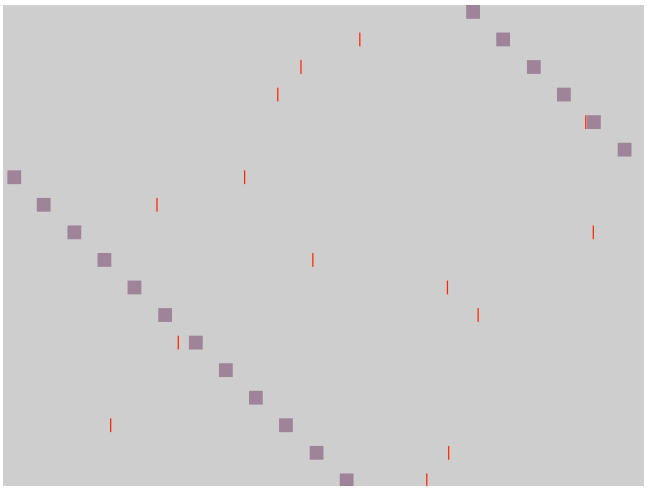

Figure 6. The user interface of App for 18 Resonators. Compare the invisible sliders affording direct manipulation of sound by red tick marks with the GUI sliders of Figs 2 & 3.

The intention for the interaction design was to be immediately engaging and allow the production of a variety of sounds, whilst being simple enough that little conscious thought would be necessary. Two main modes of interaction are employed:

1. **Multi-touch** allows the red bars within the above mentioned 18 sub-divisions of the screen to be moved to different positions, and hence the the frequency of the resonators to be altered. The change in resonator frequency is smoothed, so that no transients are produced if the user suddenly changes the frequency of a particular resonator by a large amount. It's possible to interact meaningfully with the sound both by manipulating individual bars/resonators and by making broad sweeping gestures.
2. **Accelerometer** controls, after smoothing, a number of synthesis parameters simultaneously. The smoothing produces an effect where the movement of the accelerometer feels less like a twitchy control, and more like a way of pushing the system slowly into a different equilibrium. The parameters manipulated include the base speed of the impulse trains, the amount of noise injected into the resonators, and the decay time of the resonators. The parameter ranges and mappings are chosen so that interacting with the accelerometer feels like changing the whole system rather than simply tweaking one parameter linearly.

SPECIAL 3D-AUDIO APPLICATIONS

The extension of the internal microphone with additional hardware were envisioned in the following projects, and the applications indicate how useful these extensions could be.

Soundflux is a 2-D sonic map visualizer of strong directional sounds captured with two peripheral cardioid microphones attached to a mobile phone. The microphones should be oriented with at least 90 degrees of separation between them, similar to a stereo X-Y recording arrangement to form an an acoustic "torchlight". This way, the direction of arrival of prominent sound events in the acoustical scene can be separated from diffuse background noise and reverberation.

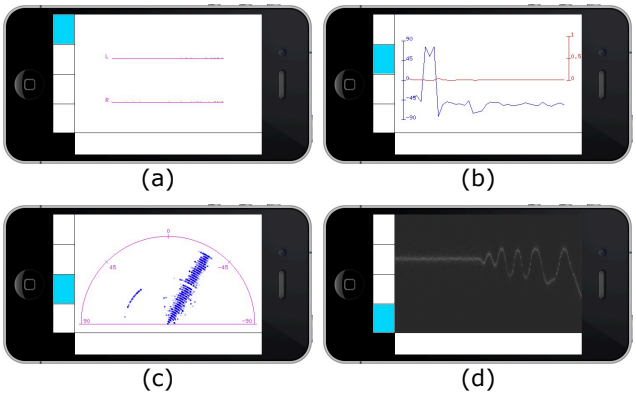

Figure 7. The user interface of and visualization of Soundflux.

The implementation is related to acoustic scene analysis and direction-of-arrival (DOA) estimation. More importantly, it is a necessary building block for more advanced applications such as adaptive steering of beamformers for real-time dereverberation and noise-suppresion for speech enhancement, segregation and identification of acoustical scenes, and source separation. There are novel audio DSP algorithms in the implementation, however here we focus on the GUI and interaction.

The GUI consists of a main screen screen showing the instantaneous spectrum of each microphone (Fig. 7 a, which shows the spectra of white noise). The the big buttons on the left side of the screen allows access to three modes of direction-of-arrival visualizations, each providing complementary information (b-d). The active button is highlighted.

The first type of visualization are the instantaneous direction and current difuseness estimate for each frequency. The horizontal axis is the frequency axis (logarithmically spaced), while the left vertical axis (blue) is the DOA axis between ± 90°, and the right vertical axis (red) is the diffuseness axis between [0-1]. Fig. 7b presents an example of a strong sinusoidal source at around 50° at about 450Hz, in the presence of another broadband noise sound coming from around -45°.

The sonar-like plot of Fig. 7c illustrates the directions as polar coordinates (radial dimension is the frequency axis). This visualization presents a more intuitive representation of where the sound is coming from. Finally Fig. 7d presents a rolling spectrogram, where instead of frequency content, the broadband energy of the signals at different directions are tracked in time. The vertical axis are the DOAs between ± 90°, and the horizontal axis is the time axis, with a full window covering around 2 seconds. The example in the figure presents a broadband source that is oscillating from left-to-right around 30° and then stays still. A sensitivity threshold based on diffuseness determines the luminosity of the energies, in order to omit random directions and make clear directional sources more visible. This sensitivity can be changed on runtime by touch-and-drag vertically on the drawing rectangle.

Double Middle-Side Decoder (DMS) is another utility application implemented with openFrameworks that has interactive controls for enhanced stereo matricing of a peripheral system of three microphones: a cardioid facing the centre of the sound source, another cardioid with the null to the centre of the sound source and a figure of eight sideways again with the null to the centre of the sound

source. By arranging the microphones in this way the figure of eight microphone picks up information from the left and right side equally. In order to get a stereo sense of these microphones the signals have to be decoded into a sum and difference matrix. This operation is carried out by four sliders, and the resulting microphone pattern is visualized in real-time in addition to the stereo sound output, as illustrated on Fig. 8. The sound examples of demonstration recordings and renderings are at http://soundcloud.com/surequta/sets/double-middle-side-dms-demo

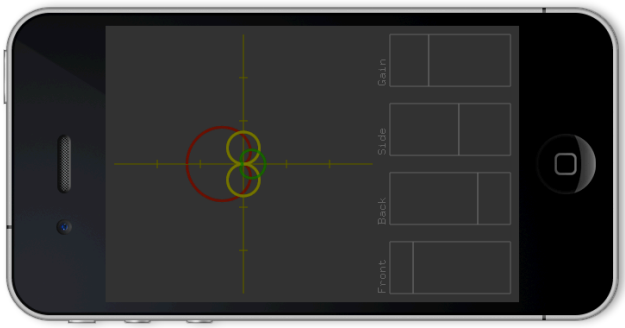

Figure 8. The user interface of and visualization of DMS.

DISCUSSION

As stated in the introduction, the project-based learning activity showcased here was a first step in technology-design exchange and iteration. Now that the advanced audio technologies embedded in the applications are available, we discuss here possible directions relevant for Mobile HCI and interaction design. These touch-points are product sound design, research through design, and ubiquitous computing.

Product (Sound) Design

Product design is a process where user-centered design methods are applied starting from user needs with the aim of a commercially feasible product. Various app stores provide the infrastructure needed for deploying mobile applications. In this respect, the spatial sound applications **soundflux** and **DMS** afford joint mobile hardware-software design, possibly in collaboration with the audio equipment manufacturers. The loudspeaker placement applications distributed e.g., by Genelec (the leading active loudspeaker company from Finland) rely currently entirely on geometrical methods, by listening to the gyroscope of a smartphone/tablet, and by providing visual cues in aligning loudspeakers for a good listening experience. **soundflux** readily demonstrates the feasibility of combining the gyroscope input with audio.

There are several applications available in the market that aim for auditory augmentation of the real-world sonic interactions (see for instance http://www.tabledrum.com). **Tabula Rasa Augmentata** could aim for a similar musical product. Within the music category, **Wdf2Go** and **Permafrost Backend** could aim for constructive musical synthesizers, **App for 18 Resonators** to an entry-level music making tool, and **BirdHound** towards an ecological equivalent of music discovery applications SoundHound and Shazam. But for a commercial deployment, these applications should be nurtured with novel product/service design practices. This is probably the reason why major mobile manufactures support the application development with campus-wide programs.

When it comes to sounding products and mobile applications, the current practices of product design have little to offer, although this is changing with efforts such as the Product Sound Design Summer Schools the authors have organized in 2010 and 2012 (see also [14]).

In product design, communication with potential users and their experiences in use are important factors. Experience prototypes [15], which can be constructed in many ways including video production, are a good way to probe sounding products and mobile applications.

Experience and Video Prototyping

In the mobile world, the original methods of video prototyping, such as visual overlays and special effects, are common practice. *Sonic overlays*, however, have been significantly under-utilized [6]. In this technique, a particular interaction is isolated and filmed, and alternative sound designs, be they voice, everyday objects or parametric sound synthesis, are overlaid on the visuals. The technique works well with mobile applications [16], since in many cases the point to overlay the sound is distinctly recognizable in videos. Similarly, the sounds generated or processed with the applications can be overlaid on explorative, scenario-based videos that provide contextual ideas to designers.

Research Through Design (RTD)

Different from product design, RTD starts with research questions and aims for communicable knowledge. The rationale behind most of the mobile applications presented here was related to the research questions and methods of the audio DSP engineering: can a sophisticated algorithm developed on a desktop platform made to be run real-time on a constrained mobile device? The applications showcased here demonstrate this aspect of feasibility, yet they should next add to the design knowledge on mobile sonic interaction, through a process where both the design and evaluation become important. Mobile audio applications afford, besides experience and video prototypes, other ways of design inquiries, for instance *mobile probes* [17], [18] or *reflection* [19]. This issue of using working mobile applications for RTD will be elaborated further in a companion paper, in the context of developing mobile applications for special needs.

Ubiquitous Computing

The field of ubiquitous computing seeks natural and context-aware applications [20] and to these ends mobile audio can be a tremendous asset. Sound is an integral part of our environment and can provide contextual information for computational analysis. Sound-based context recognition can be fused with information from other sensors for more robust techniques, to be used as the basis of adaptive interfaces.

On the other hand, mobile phones obviously also afford utilizing sound as feedback in ubiquitous computing applications. In addition to attention-driving and UI sounds, more *exploratory* and *performative* sound solutions are also conceivable (as in the case study of **TRA**). Furthermore, sound can be used to facilitate social connectivity in ubiquitous applications. Multiple people can act upon the same microphone-equipped device, and vice versa, the sounds captured by one device can be transmitted to others.

CONCLUSION AND FUTURE WORK

In this contribution, we have tackled how to integrate the rapidly evolving knowledge, know-how, tools, and techniques of mobile (audio) programming and interaction design into university curricula. Our suggested solution is to interleave technology and design within multidisciplinary, project-based settings. We have showcased the applications developed in the first leg of this process, and discussed three directions relevant for Mobile HCI and interaction design: product sound design, research through design, and ubiquitous computing.

There are two immediate future directions we will consider: 1) designing for special needs and 2) combining mobile audio with open APIs and service design. These directions were mentioned implicitly above, now we explicate them.

The general goal of *universal and accessible design* [21] is also relevant for mobile use, where physical, cognitive, or attentional capacity of a user is reduced because of mobility [22]. Besides that, mobile audio applications hold promise for users with reduced sight or visual impairments. Beyond accessibility, considering the interests and engagement of *people with special needs* requires shifting the focus from users' lack of abilities to their strengths. The challenge in this shift is the diversity of the user group, which makes it difficult to come up with universal, "one-size-fits-all" solutions. The rapid intake of tablets and smartphones among populations with special needs is a promising direction to pursue in mobile HCI concerning the auditory modality [23], [24]. This direction is explored further in a companion contribution submitted to SiMPE'13. Suffice to say here that currently, the mobile accessibility applications are targeting more to physical conditions and less so to the cognitive and attentional resources.

This observation has guided us towards an RTD exercise in control and awareness of time [16], which facilitated open APIs (Helsinki Regional Transport and Google Weather) together with GPS and minimal GUI input, and used auditory display to provide information. The application has received two awards[2], and its source code has been made publicly available. We note that both the public and the private sector strategically consider mobile applications as additional touch points besides the physical front and back offices. To help user involvement and participation in value creation, they also provide open data and APIs. In Denmark, for instance, there is a dedicated portal enlisting such APIs (http://digitaliser.dk). We therefore plan to harness this data and APIs for designing novel services and applications, equipped with advanced auditory displays empowered by the mobile audio technologies showcased here, and made resonate with the user needs by design and evaluation.

ACKNOWLEDGMENTS

We thank all our students and participants for their work showcased here. C. Erkut acknowledges the Academy of Finland (#120583, 2007-2012) for the support of the work described here.

REFERENCES

[1]Holland, S. and Morse, D., "AudioGPS: Spatial audio navigation with a minimal attention interface," Personal and Ubiquitous Computing, vol. 6, pp. 253–259, 2002.

[2] McGookin, D. and Magnusson, C., "Extreme navigation: introduction to the special issue," Personal and Ubiquitous Computing, Sep. 2011.

[3]Fitzmaurice, G., Ishii, H., and Buxton, B., "Bricks: laying the foundations for graspable user interfaces," in Proc. CHI, Denver, CO, USA, 1995, pp. 442–449.

[4]Rekimoto, J., "Tilting operations for small screen interfaces," presented at the the 9th annual ACM symposium, Seattle, Washington, USA, 1996, pp. 167–168.

[5]Franinovic, K. and Serafin, S., eds. Sonic Interaction Design. Cambridge, MA: MIT Press, 2013.

[6]Rocchesso, D., Serafin, S., and Rinott, M., "Pedagogical Approaches and Methods," in Sonic Interaction Design, Chapter 4, Franinovic, K. and Serafin, S., Eds. Cambridge, MA: MIT Press, 2013, pp. 125–150.

[7]Wright, M., "Open Sound Control: an enabling technology for musical networking," Org. Sound, vol. 10, no. 3, pp. 193–200, Nov. 2005.

[8]Kaltenbrunner, M., Bovermann, T., Bencina, R., and Costanza, E., "TUIO - A Protocol for Table Based Tangible User Interfaces," 2005.

[9]Brinkmann, P., Kirn, P., Lawler, R., McCormick, C., Roth, M., and Steiner, H.-C., "Embedding Pure Data with libpd: Design and Workflow," presented at the Pure Data Convention, Weimar-Berlin, Germany, 2011.

[10]Rabenstein, R., Petrausch, S., Sarti, A., De Sanctis, G., Erkut, C., and Karjalainen, M., "Block-based physical modeling for digital sound synthesis," IEEE Signal Proc Mag, vol. 24, no. 2, pp. 42–54, 2007.

[11]Fagerlund, S., "Automatic Recognition of Bird Species by Their Sounds," Master's Thesis, Helsinki University of Technology, 2004.

[12]de Paiva, R. C. D. and Välimäki, V., "The Helmholtz Resonator Tree," presented at the International Conference on Digital Audio Effects, 2012, pp. 1–7.

[13]Jylhä, A., "Sonic Gestures and Rhythmic Interaction between the Human and the Computer," Doctoral Thesis, Aalto University, School of Electrical Engineering, Espoo, Finland, 2012.

[14]Langeveld, L., van Egmond, R., Jansen, R., and Özcan, E., "Product Sound Design: Intentional and Consequential Sounds," in Advances in Industrial Design Engineering, Ch. 3, Coelho, D. A., Ed. InTech, 2013.

[15]Bucheneau, M. and Suri, J. F., "Experience prototyping," in Proc. DIS 2000, Brooklyn, NY, pp. 424–433.

[16]Kostiainen, J., Erkut, C., and Piella, F. B., "Design of an Audio-based Journey Planner Application," presented at the MindTrek'11, Tampere, Finland, 2011, pp. 107–113.

[17]Hulkko, S., Mattelmäki, T., Virtanen, K., and Keinonen, T., "Mobile probes," presented at the NordiCHI, Tampere, Finland, 2004, ACM. pp. 43–51.

[18]Barrass, S., Schaffert, N., and Barrass, T., "Probing Preferences between Six Designs of Interactive

[2] Second prize in the category of "Innovative Interfaces" by Helsinki Regional Traffic (HSL) in 2011 and a special prize from the Helsinki City Data Center in December 2012. The code-base "intrance-sounds: Auditory Journey Planning on iOS" is hosted at http://code.google.com/p/intrance-sounds/.

Sonifications for Recreational Sports, Health and Fitness," in Proc. ISon, Stockholm, Sweden, 2010.

[19]Obrenović, Ž., "Rethinking HCI education," interactions, vol. 19, no. 3, pp. 66–70, May 2012.

[20]Abowd, G. D. and Mynatt, E. D., "Charting past, present, and future research in ubiquitous computing," ACM Transactions on Computer-Human Interaction, vol. 7, no. 1, pp. 29–58, Mar. 2000.

[21]Gribbons, W. M., "Universal Accessibility and Functionally Illiterate Populations: Implications for HCI, Design, and Testing," in The Human-Computer Interaction Handbook, no. 44, Lawrence Erlbaum Associates, 2008.

[22]Obrenović, Ž., Abascal, J., and Starcevic, D., "Universal accessibility as a multimodal design issue," Communications of the ACM, vol. 50, no. 5, pp. 83–88, 2007.

[23]Sampath, H., Indurkhya, B., and Sivaswamy, J., "A communication system on smart phones and tablets for non-verbal children with autism," presented at the ICCHP'12: Proceedings of the 13th international conference on Computers Helping People with Special Needs, 2012, vol. Part II , Volume Part II.

[24]Hourcade, J. P., Bullock-Rest, N. E., and Hansen, T. E., "Multitouch tablet applications and activities to enhance the social skills of children with autism spectrum disorders," Personal and Ubiquitous Computing, vol. 16, no. 2, pp. 157–168, Feb. 2012.