PhonHarp: A Hybrid Digital-Physical Musical Instrument for Mobile Phones Exploiting the Vocal Tract

Giorgio Presti
Daniele Adriano
Federico Avanzini
Adriano Baratè
Luca Andrea Ludovico
{name.surname}@unimi.it
Laboratory of Music Informatics (LIM), Department of Computer Science, University of Milan
Milan, Italy

ABSTRACT

This paper presents *PhonHarp*, a new interface for musical expression. The instrument can be described as a human-device loop, based on an Android app for mobile phones that produces sounds, which are modified by the vocal tract of the musician. After analyzing the state of the art, this work will provide details about functional requirements, design choices, and implementation aspects.

CCS CONCEPTS

• **Human-centered computing** → **Sound-based input / output**; **Mobile phones**; • **Applied computing** → **Sound and music computing**.

KEYWORDS

mobile devices, music expression, musical instruments, physical computing

ACM Reference Format:
Giorgio Presti, Daniele Adriano, Federico Avanzini, Adriano Baratè, and Luca Andrea Ludovico. 2021. *PhonHarp*: A Hybrid Digital-Physical Musical Instrument for Mobile Phones Exploiting the Vocal Tract. In *AM '21: Proceedings of the 16th International Conference on Audio Mostly, September 01–03, 2021, Trento, Italy.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3478384.3478413

1 INTRODUCTION

Nowadays, many virtual musical instruments are available as mobile applications, but little attention is paid to an extension of the interpretative possibilities of the musician. The goal of this work is to present and discuss a tool that primarily employs the phonatory apparatus as an extension of the musical instrument itself. Moreover, by using the smartphone's multi-touch interface, it allows for a multidimensional control of timbre parameters. Therefore, the performer has different techniques at her disposal to confer greater expressiveness to the generated melody. The name proposed for the instrument, *PhonHarp*, recalls the common root shared by the words *phonatory* and *phone*, and also recalls the *mouth harp*, which is played in a similar fashion.

The main source of inspiration for this work is the aim to create a hybrid instrument that embodies digital technology (the software running on the device), analog electroacoustic aspects (the phone loudspeaker), and physical components (the musician's mouth). In particular, the last two elements are variable and unpredictable (phone model, loudspeaker technology, mouth properties), letting the actual sound quality be unique, emerging from the possible elements combinations or tweaks (such as the use of external loudspeakers), a valuable aspect in terms of instrument *hackability* and *multiple interpretations design* [16, 24].

From a technological point of view, this work is similar to a talk-box, a pedal effect born in the '60s and mainly used in rock music. The goal of the talk-box is to let the performer modify the sound of a musical instrument (e.g., a guitar or a keyboard) with the mouth, thanks to a plastic tube that connects the acoustic audio output of the musical instrument to the oral cavity. The musician can form words, without actually pronouncing them, modifying the output sound. In this way, simply using the vocal tract, the spectral composition of the signal is modified, giving the audience the illusion of listening to a "speaking instrument" (which is basically a physical version of the Vocoder).

A complete talk-box system requires a musical instrument, an amplifier, a microphone, and a running implementation of the effect. These constraints have hampered the wide diffusion of such an effect beyond the community of professionals. While the availability of a sound source remains essential, the other aspects require skills and hardware components that can be emulated by a computer system. In this sense, the present work builds on the ubiquity of smartphones to make a talk-box-like effect available to anyone.

The remainder of the paper is organized as follows: Section 2 will review related approaches, Section 3 will present design goals and implementation details, and Section 4 will draw the conclusions.

2 STATE OF THE ART

This work is rooted in the long-standing development of innovative interfaces for controlling musical parameters. By analyzing

---

AM '21, September 01–03, 2021, Trento, Italy

ACM ISBN 978-1-4503-8569-5/21/09…$15.00
https://doi.org/10.1145/3478384.3478413

the scientific literature (e.g., the proceedings of the annual International Conference on New Interfaces for Musical Expression), and the most relevant applications available on the market, a first dimension to classify Digital Musical Instruments (DMI) emerged, distinguishing between the ones that require dedicated hardware and those running on mobile devices (e.g., smartphones, tablets or laptops). Another criterion that we took into consideration, due to the goals of the present work, is the possible involvement of the vocal tract. These independent dimensions will be further explored in the following subsections.

Within the category of musical instruments designed to be launched on mobile devices, such as smartphones, tablets, and laptops, two subcategories can be determined. These correspond to DMIs based on the possibility to apply effects and manipulate already available music only, or, conversely, to perform a tune generated by the user on-the-fly.

The former group is composed by those mobile apps that do not offer direct control over played music or sound. This category embraces collaborative musical instruments, effect generators, and music games. *Echobo* [8], *WIJAM* [4], **12** [5] and *SWARMED* [7] are examples of collaborative musical instruments for mobile devices where the final melody is the result of real-time actions of geographically distributed users. Among effect generators, it is worth mentioning: modular synths such as *AuSynthAR* [10], *GrainTrain* [3], and *TC-11* [15]; sequencers such as *Gliss* [20] and *Pyxis Minor* [1]; and loopers such as *Twkyr* [23]. In most of these applications, audio effects are dynamically controlled through the multi-touch panel. *MMODM* [18] is an on-line drum machine connected to Twitter in order to create massive jam sessions. The app called *Two Turntables and a Mobile Phone* [2] employs the accelerometer and the gyroscope of the smartphone to mimic the record-style spinning plates used by DJs. Concerning musical games, mobile stores offer a great number of apps, often inspired by the top-selling game called *Guitar Hero* [1]. Among the tools discussed in the scientific literature, it is worth citing *Tok!* [9], a rhythmic game with advanced features, and *Scratch-Off* [6], where the user takes the role of a DJ; both tools are based on gesture detection, typically captured from the accelerometer of the multi-touch interface. Apparently, no tool belonging to the broad category of collaborative musical instruments, effect generators, and music-oriented games seems to support the phonatory apparatus as a way to process sound.

The latter group, i.e. instruments allowing direct control over music generation, is more interesting for the scope of this work, since the proposed musical instrument falls under this category. *Smule's Ocarina* [21] is an extended version of the traditional musical instrument, where notes are played by blowing into the microphone and plugging the virtual finger-holes using the multi-touch panel; additionally, the accelerometer lets the player apply a vibrato effect. Another work from the same programmer is *Magic Fiddle* [22], a musical instrument that recalls the violin interface and adds some gamification elements. *Laptop Accordion* [12] is meant to turn a laptop into an accordion: the gesture of expanding or compressing the bellows can be simulated by opening and closing the monitor panel, whereas the accordion's keys are activated by the computer keyboard. *Orphion* [19], *Pitch Canvas* [17], and *iPad Bladeaxe* [13] are innovative virtual instruments that take full benefit from large tablet screens to place notes onto a grid and to let the user control sound parameters through multi-touch gestures.

Instruments where the vocal tract is a relevant feature of the interface are also present, such as the *Mouthesizer* [11], where a live video of the mouth is used as a controller, *Pink Trombone*,[2] an articulatory speech synthesizer, and a wide range of tools developed in the context of the *SkAT-VG* project [14]. Nevertheless, none of the cited instruments physically involves the phonatory apparatus in the actual shaping of sound.

In this context, *ElectroSpit* [3] and *Pocket Talkbox* [4] are particularly relevant solutions, since they use the human vocal tract in order to create an effect similar to the talk-box. The former is a commercially-available product that, thanks to a dedicated device made of two loudspeakers, is able to redirect the sound produced by the app to the base of the throat. *Pocket Talkbox* is an Android app presenting a 8-element grid for musical notes, and a talk-box effect is obtained by bringing the smartphone loudspeaker near the mouth. *Pocket Talkbox* does not require dedicated hardware components, but it is limited as it regards pitch extension, timbre, and the dynamical control of the sound output. Conversely, our proposal aims to join usability, expressiveness, and configurability.

3 DESIGN AND IMPLEMENTATION

In this section we describe the design goals, the interface, and the implementation of *PhonHarp*.

Concerning the functional requirements, the basic idea of *PhonHarp* is letting users shape the sound through their vocal tract. To this end, two key aspects must be considered: i) concerning interface and controls, the smartphone must be held like a flute in order to bring the loudspeakers as close as possible to the mouth, and ii) the timbre has to present spectral richness in order to offer an articulated effect.

Other aspects to take into account are usability and accessibility. Users should be able to transform their smartphone into a musical instrument simply by installing a piece of software, with no need for external devices. Both left-handed and right-handed users should be able to play it.

The musical instrument is a monophonic one. Music-related functional requirements should include the possibility to perform melodic lines spanning across at least two octaves, playing all notes of the chromatic scale. The user should be allowed to pick a timbre from a predefined list. In addition to note-by-note frequency and intensity, some basic sound parameters should be configurable: harmonic content (filtering and pulse width modulation), vibrato, tremolo, equalization, portamento, and – finally – volume, and envelopes. Some parameters should be also dynamically adjustable during a single note performance: intensity, harmonic content, pitch bend, and the amount of tremolo, vibrato, and pulse width modulation effects.

Concerning the interface, *PhonHarp* basically presents a keyboard consisting of five buttons, indicated by white circles in Figure 1. Users must rotate the smartphone in order to bring the speaker

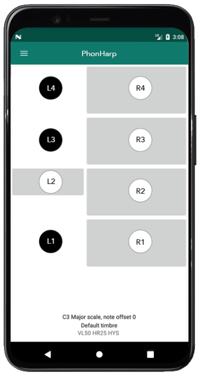
---

[1] https://www.guitarhero.com

[2] .https://dood.al/pinktrombone/

[3] https://www.electrospit.com

[4] https://play.google.com/store/apps/details?id=com.Bright_Blue_LED.PocketTalkbox



C3 Major scale, note offset 0
Default timbre
VL50 HR25 HYS

Figure 1: The interface of *PhonHarp*.

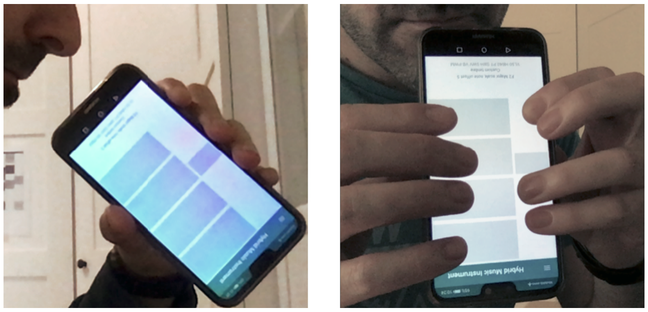


Figure 2: Two alternative ways to hold the device.

towards the mouth. Figure 2a shows the suggested orientation, while Figure 2b presents an alternative way to hold the phone.

For the sake of simplicity, from here on we will refer to right-handed users. In order to have a firm grip, it is necessary to hold the device with the left hand by using the index, ring and little fingers in the upper part (L1, L3, and L4, respectively) and the thumb blocking the other side of the device. Musical notes are determined by the index, middle, ring and little fingers of the right hand (R1, R2, R3 and R4 circles, respectively), while the middle finger of the left hand (L2) acts as a modifier of the current note by lowering it by a semitone. For a firmer grip, the thumb of the right hand can be used to support the bottom of the smartphone, too.

Although *PhonHarp* may recall a flute, the relationship between the fingering and the notes played makes it more like a trumpet. To identify the note triggered by keys R1 to R4, we will use a 4-bit encoding of numbers in the base-2 numeral system, in the interval 1 to 15 (0 has been excluded). These notes are mapped onto a diatonic musical scale, thus determining an extension of the instrument equal to 2 octaves. During the configuration phase, it is possible to specify the type of scale, choosing between major and natural minor, the first grade, and the reference octave. Figure 3 shows the relationship between the combination of pressed keys and the note played. In this case, the instrument is tuned to play the *G Major* scale spanning from G3 to G5. Optionally, a user-defined hysteresis time can be activated in order to avoid multiple triggers when pressing many keys at once.

Persistent settings can be configured in a dedicated area of the app. Conversely, dynamical timbre parameters (e.g., tremolo and vibrato) can be controlled using swipe gestures on the horizontal and vertical axis of the smartphone screen.

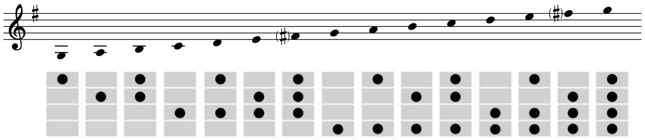


Figure 3: Mapping between notes and keys.

The application has been designed to be compatible with Android version 5 (codenamed *Lollipop*) or higher.

The core of the app is the software synthesizer, whose block diagram is shown in Figure 4. It consists in a semi-normalized subtractive architecture, composed of a variable pulse-width (PW) square-wave oscillator (OSC), followed by an amplitude-modulation block (AMP), and a simple equalizer (EQ). Typically subtractive synthesizers also carry a filter module, but, in this case, the filter is provided by the mouth of the musician, thus involving a physical process in the sound-generation chain.

Using a variable PW oscillator allows the user to select in a simple way among odd-only harmonics typical of blowed instruments (a 50% duty cycle); all harmonics, typical of plucked string instruments (very short duty cycle, generating a train of impulses); and a PW-modulated sound, similar to bowed string sounds.

The trailing EQ consists in a low-shelving and a high-shelving equalizers. Cutoff frequencies are fixed to 400 Hz and 8 kHz respectively, in order to provide control over those parts of the spectrum which are outside the band modulated by the mouth shape. Modulation signals are 2 low-frequency oscillators (LFO), and 2 attack-sustain-release (ASR) envelope generators. Modulation destinations are hardwired to the pitch and PW parameters of OSC, and to the overall amplitude (controlled by AMP).

Monophonic expression signals are the note selected by the keyboard, the horizontal and vertical swiping positions over the pressed keys, and the gate information (is some key pressed or not). The possible destinations that the user can select for these controls are: Volume, PW, pitch, tremolo depth, vibrato depth, and PW modulation depth. Other user-selectable preferences saved in the timbre settings are: a timbre name (many timbre can be stored, but only one at time can be played), portamento, equalizer gain values, LFO frequences, modulation amounts, envelopes parameters, OSC PW, expression signals routing, and keyboard hysteresis time.

Finally, a feature for sharing timbre settings with user contacts is provided via standard Android sharing practices. This is achievable by long-pressing the timbre name when in the timbre list screen.

For the synthesizer implementation, we chose the *JSyn* library, written in Java language. Alternative solutions were evaluated (e.g., *Superpowered*, *JUCE*, and *Oboe*), but this library proved to achieve good performances in terms of latency and resource occupation, also guaranteeing easy integration within the Java environment. Another plus is the very small size of the resulting Android Package (APK), namely 4 MB approx.

CPU, memory, and energy consumption were monitored through the *Android Studio Profiler*, performing a number of tests on a *HUAWEI P20 Lite* device equipped with Android 9. On average, the CPU utilization reached about 20% during the launch phase,

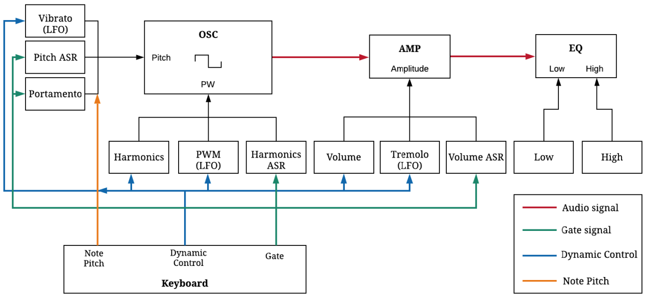


Figure 4: Block diagram of the synthesizer.

then dropped to 10% when the synthesizer was active, regardless of whether or not the sound was reproduced. Energy consumption followed a similar trend, with average values in the start phase and remaining remarkably low for other activities. Measured memory usage was around 80 MB: 60% for the allocation of native objects of the Android framework, 22% for Java object instances, and 18% for the bytecode.

4 CONCLUSIONS AND FUTURE WORK

PhonHarp aims to exploit the broad diffusion and the ease of use of smartphones to create a musical instrument whose characteristics span across an acoustic one (due to the involvement of the vocal apparatus), a digital electrophone one (due to the software sound synthesizer), and an analog electrophone one (due to the presence of the phone loudspeakers). Its 2-octaves extension and tuning options allow the performance of a vast musical repertoire. The learning curve necessary to play simple tunes is expected to be comparable to that of the recorder, thus addressing also beginners and people with no previous musical knowledge. On the other side, skilled users will be able to confer greater expressiveness to the musical performance by customizing the timbre and dynamically controlling its parameters. It is worth underlining that *PhonHarp* presents additional control aspects with respect to similar apps; considering also its peculiar keyboard layout, *PhonHarp* opens the way to elements of mastery as an instrument in its own right.

Concerning future work, the expressive possibilities of *PhonHarp* could be further expanded using the gyroscope and the accelerometers. In addition, some buttons (for example, volume controls) could become shortcuts to load predefined scenarios, or they could be associated to special functions such as quick tuning. Social interaction potential should be investigated as well; e.g., when configurations, timbres, and tablatures are shared among users.

PhonHarp is currently a working prototype, so an extensive test campaign should be conducted in order to assess its usability for musicians and non musicians, and to evaluate the accessibility by different types of users. After practicing the instrument, players are expected to adapt to its interface and develop other performance techniques, thus exploiting such a technology in original and even unpredictable ways.

REFERENCES

[1] Timothy J. Barraclough, Dale A. Carnegie, and Ajay Kapur. 2015. Musical Instrument Design Process for Mobile Technology. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Louisiana State University, 289–292.

[2] Nicholas J. Bryan and Ge Wang. 2011. Two Turntables and a Mobile Phone. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. 179–184.

[3] Anıl Çamcı. 2018. GrainTrain: A Hand-drawn Multi-touch Interface for Granular Synthesis. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Virginia Tech, 156–161.

[4] Jun-qi Deng, Francis Chi Moon Lau, Ho-Cheung Ng, Yu-Kwong Kwok, Hung-Kwan Chen, and Yu-heng Liu. 2014. WIJAM: A Mobile Collaborative Improvisation Platform under Master-players Paradigm. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Goldsmiths, University of London, 407–410.

[5] Eran Egozy and Eun Young Lee. 2018. *12*: Mobile Phone-Based Audience Participation in a Chamber Music Performance. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Virginia Tech, 7–12.

[6] Nicholas Gillian, Sile O'Modhrain, and Georg Essl. 2009. Scratch-Off: A Gesture Based Mobile Music Game with Tactile Feedback. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. 308–311.

[7] Abram Hindle. 2013. SWARMED: Captive Portals, Mobile Devices, and Audience Participation in Multi-User Music Performance. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Graduate School of Culture Technology, KAIST, 174–179.

[8] Sang Won Lee and Jason Freeman. 2013. echobo: Audience Participation Using The Mobile Music Instrument. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Graduate School of Culture Technology, KAIST, 450–455.

[9] Sang Won Lee, Ajay Srinivasamurthy, Gregoire Tronel, Weibin Shen, and Jason Freeman. 2012. Tok!: A Collaborative Acoustic Instrument using Mobile Phones. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. University of Michigan.

[10] Pedro Pablo Lucas. 2019. AuSynthAR: A simple low-cost modular synthesizer based on Augmented Reality. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. UFRGS, 405–406.

[11] Michael J Lyons, Michael Haehnel, and Nobuji Tetsutani. 2001. The mouthesizer: a facial gesture musical interface. In *Conference Abstracts, Siggraph 2001*, Vol. 230. Citeseer.

[12] Aidan Meacham, Sanjay Kannan, and Ge Wang. 2016. The Laptop Accordion. In *Proceedings of the International Conference on New Interfaces for Musical Expression (2220-4806, Vol. 16)*. Queensland Conservatorium Griffith University, 236–240.

[13] Romain Michon, Julius Orion Smith III, Matthew Wright, and Chris Chafe. 2016. Augmenting the iPad: the BladeAxe. In *Proceedings of the International Conference on New Interfaces for Musical Expression (2220-4806, Vol. 16)*. Queensland Conservatorium Griffith University, 247–252.

[14] Davide Rocchesso, Davide A Mauro, and Stefano Delle Monache. 2016. miMic: The microphone as a pencil. In *Proceedings of the TEI'16: Tenth International Conference on Tangible, Embedded, and Embodied Interaction*. 357–364.

[15] Kevin Schlei. 2012. TC-11: A Programmable Multi-Touch Synthesizer for the iPad. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. University of Michigan.

[16] Phoebe Sengers and Bill Gaver. 2006. Staying open to interpretation: engaging multiple meanings in design and evaluation. In *Proceedings of the 6th conference on Designing Interactive systems*. 99–108.

[17] Bradley Strylowski, Jesse Allison, and Jesse Guessford. 2014. Pitch Canvas: Touchscreen Based Mobile Music Instrument. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Goldsmiths, University of London, 171–174.

[18] Basheer Tome, Donald Derek Haddad, Tod Machover, and Joseph Paradiso. 2015. MMODM: Massively Multipler Online Drum Machine. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Louisiana State University, 285–288.

[19] Sebastian Trump and Jamie Bullock. 2014. Orphion: A Gestural Multi-Touch Instrument for the iPad. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Goldsmiths, University of London, 159–162.

[20] Jan T. von Falkenstein. 2011. Gliss: An Intuitive Sequencer for the iPhone and iPad. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. 527–528.

[21] Ge Wang. 2009. Designing Smule's Ocarina: The iPhone's Magic Flute. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. 303–307.

[22] Ge Wang, Jieun Oh, and Tom Lieber. 2011. Designing for the iPad: Magic Fiddle. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. 197–202.

[23] Karl Yerkes and Matthew Wright. 2014. Twkyr: a Multitouch Waveform Looper. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Goldsmiths, University of London, 375–378.

[24] Victor Zappi and Andrew McPherson. 2018. Hackable Instruments: Supporting Appropriation and Modification in Digital Musical Interaction. *Frontiers in ICT* 5 (2018), 26.