THE HELMHOLTZ RESONATOR TREE

Rafael C. D. Paiva and Vesa Välimäki[*]

Department of Signal Processing and Acoustics
Aalto University, School of Electrical Engineering
Espoo, Finland
rafael.dias.de.paiva@aalto.fi vesa.valimaki@aalto.fi

ABSTRACT

The Helmholtz resonator is a prototype of a single acoustic resonance, which can be modeled with a digital resonator. This paper extends this concept by coupling several Helmholtz resonators. The resulting structure is called a Helmholtz resonator tree. The height of the tree is defined by the number of resonator layers that are interconnected. The overall number of resonance frequencies of a Helmholtz resonator tree is the same as its height. A Helmholtz resonator tree can be modeled using wave digital filters (WDF), when electro-acoustic analogies are applied. A WDF tool for implementing Helmholtz resonator trees has been developed in C++. A VST plugin and an Android mobile application were created, which can run short Helmholtz resonator trees in real time. Helmholtz resonator trees can be used for the real-time synthesis of percussive sounds and for realizing novel filtering which can be tuned using intuitive physical parameters.

1. INTRODUCTION

Musical acoustics have many applications in understanding musical instruments and building computational tools. These computational tools may be used to create new digital musical instruments as well as to build digital effects based on intuitive physical phenomena. Some interesting phenomena derive from Helmholtz resonances [1]. These resonances are related to the body of acoustic string instruments, such as the guitar or violin, wind instruments, such as the flute or a simple ocarina, and the cavity of percussion instruments, such as the hang [2].

The physical description of musical phenomena use a variety of techniques [3]. Digital waveguides are used to model vibrating phenomena like strings and airflow in wind instruments [4]. The mass-spring models are used to create vibrating structures in an intuitive way [5, 3]. The modes of a system may be represented using coupled mode synthesis [6], modal synthesis [7], or the functional transformation method [8]. Various percussive musical instruments have been successfully modeled using a digital waveguide mesh [9, 10, 11], modal synthesis [12], and finite differences [13]. Additionally, there are a variety of models of electric circuits. These include state-space methods [14, 15], the K method [16] and wave digital filters (WDF) [17, 18] among others. WDF applications include the model of a piano hammer [19], models of vacuum tube amplifiers [20], and a model of an audio transformer in vacuum tube amplifiers [21].

This paper proposes the combination of several Helmholtz resonators in a tree-like structure, which is called a Helmholtz resonator tree. Such structures can be modeled using WDFs when electro-acoustic analogies are applied. This leads to real-time physical modeling of complex resonant structures, where the input and output points can be located at any of the interconnected resonators.

The Helmholtz resonator tree yields an intuitive way of creating physically-inspired resonating structures for synthesis. In this type of structure several resonators influence each other, and their parameters can be modified using acoustical parameters, which are intuitive even for non-technical people, although the exact resonance frequencies are unknown. Additionally, it provides the possibility of exciting the system at different points, resulting in timbre variations of the same musical instrument. This approach differs from the implementation of cascaded second order filters, since the resonators interact with each other, changing their resonance frequencies.

This paper is organized as follows. Section 2 presents a review of the acoustic-electric analogy for obtaining an equivalent model a Helmholtz resonator and extending it to a Helmholtz resonator tree structure. Section 3 reviews basic WDF concepts. Section 4 introduces the Helmholtz resonator tree tool with a VST plugin and an Android mobile application as implementation examples. Section 5 shows simulation results to evaluate the effect of the Helmholtz resonator tree structure and compares the computational cost of the implemented system with a commercial circuit simulator software. Section 6 concludes the paper.

2. HELMHOLTZ RESONATOR

2.1. Basic concept

The Helmholtz resonator, which is a prototype of a simple acoustic resonant system, can be thought to be a hollow shell, like a bottle, enclosing a volume connected to the external environment through an open pipe, or neck, as shown in Fig. 1(a). This structure is specified by the cavity volume V_0, the neck length l, the neck cross-sectional area S, and parameters of the surrounding environment [22], namely the speed of sound c and the density ρ of the surrounding gas where the resonator is located. These parameters are typically $c = 345$ m/s and $\rho = 1.2$ kg/m^3 at room temperature.

In order to derive the Helmholtz resonator model, the impedance of each acoustic part needs to be determined. The general impedance of an acoustic system is given by

$$Z = \frac{p}{U} = \frac{p}{Su}, \tag{1}$$

where p is the gas pressure, U is the volume velocity, u the particle velocity, and S the cross sectional area. By using the analogy pressure→voltage and volume flow→current, it is possible to

[*] This work was supported by the Nokia Foundation.

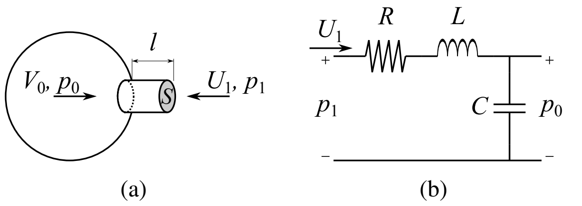


Figure 1: (a) Helmholtz resonator and (b) its equivalent circuit (adapted from [1]).

model an acoustic impedance as a circuit and use circuit simulation techniques to model the entire system.

The equivalent circuit of a Helmholtz resonator is given by an RLC circuit as Fig. 1(b). In this model, the impedance of an open tube is represented by the inductor L given by [1]

$$L = \frac{\rho l}{S}. \tag{2}$$

Additionally, the cavity is modeled by the capacitor C, which is given by [1]

$$C = \frac{\rho c^2}{V_0}. \tag{3}$$

Finally, the resistance is related to the cross-sectional area of the pipe aperture and the physical characteristics of the medium as [23]

$$R = \frac{\rho c}{S}. \tag{4}$$

Equations 2, 3 and 4 show that a Helmholtz resonator can be represented by an equivalent circuit as in Fig. 1(b). In this circuit, the external pressure is represented by the input voltage p_1, while the cavity pressure is represented by the capacitor voltage p_0. Additionally, the volume velocity in the neck is represented by the current U_1 flowing through the resistor and inductor.

2.2. Helmholtz resonator tree

The concept of the Helmholtz resonator can be expanded to build more complex structures as illustrated in Fig. 2. This structure is named the Helmholtz resonator tree. In Fig. 2(a), five Helmholtz resonators are connected through their necks. Hence, the pressure at each cavity will be the result of the integration of the volume flows from the tubes connected to this cavity. As a result, each cavity of Fig. 2(a) can be represented as a capacitor in Fig. 2(b), and the connections between cavities through the tubes are represented by circuit connections with resistors and inductors corresponding to the impedance of each tube.

The acoustic impedance of a Helmholtz resonator tree can be derived in an iterative way. The acoustic impedance of a single resonator is given by

$$Z_1(\omega) = \frac{(\mathrm{j}\omega)^2 LC + \mathrm{j}\omega RC + 1}{\mathrm{j}\omega C}, \tag{5}$$

where ω is the frequency in rad/s and $\mathrm{j} = \sqrt{-1}$ is the imaginary unit. The impedance of a Helmholtz resonator tree of height two and with N leaves can be determined as

$$Z_2(\omega) = \mathrm{j}\omega L + R + \frac{1}{\mathrm{j}\omega C + \sum_{n=1}^{N} \frac{1}{Z_{1,n}}}, \tag{6}$$

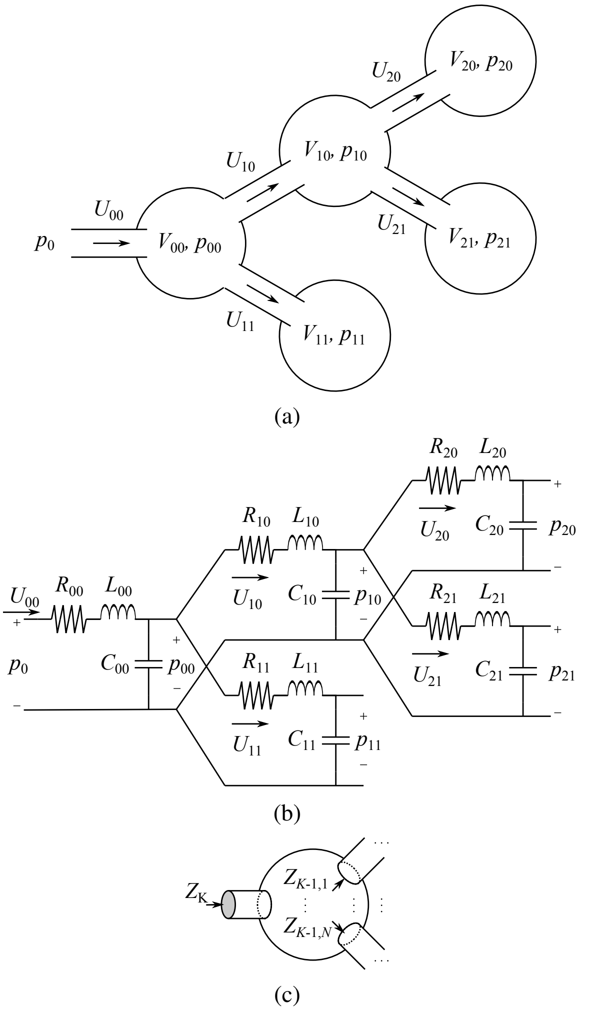


Figure 2: (a) Example of a Helmholtz resonator tree; (b) its equivalent circuit; and (c) impedance calculation.

where $Z_{1,n}(\omega)$ is the impedance of the n^{th} leaf, given by Eq. 5. The same procedure performed for a tree of height two can be generalized for a tree of height K, where the impedance of this tree $Z_K(\omega)$ is calculated based on the impedances of the subtrees $Z_{K-1,n}(\omega)$ connected to it, as illustrated in Fig. 2 (c). This procedure results in the general form of the Helmholtz resonator tree impedance

$$Z_K(\omega) = \frac{-\omega^2 LC + \mathrm{j}\omega RC + 1}{\mathrm{j}\omega C + \sum_{n=1}^{N} \frac{1}{Z_{K-1,n}}}. \tag{7}$$

Finally, the pressure-to-volume flow transfer function, can be obtained with the impedance $Z_K(\omega)$ as

$$H_K(\omega) = \frac{u(\omega)}{p(\omega)} = Z_K^{-1}(\omega). \tag{8}$$

This results in a complex resonating system where the resonances are given by the poles of $H_K(\omega)$ in Eq. 8. The poles of $H_K(\omega)$ are dependent on all the resonator parameters, and thus no simple closed-formula solution can be found for the resonating frequencies of this type of system. A demonstrative video illustrates the effect of changing a single resonator at `http://www.acoustics.hut.fi/go/dafx12-helmholtztree`.

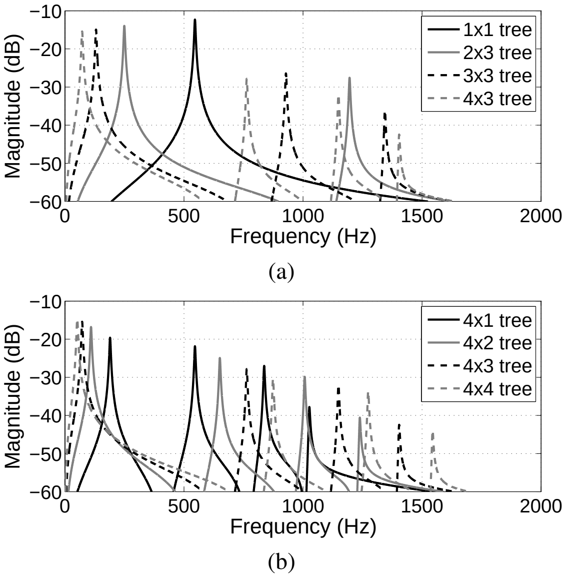
2.3. Helmholtz resonator tree structure effect

The parameters of the Helmholtz resonator tree were evaluated with Eq. 8 in two sets of results. In the first, the effect of the tree height and number of branch divisions is evaluated, while in the second set the effect of individual physical resonator parameters is evaluated.

Figure 3 shows the results for the evaluation of the Helmholtz resonator tree size parameters. In all the simulations, the curves are labeled $K \times B$, where K indicates the tree height and B indicates the number of branch divisions. Additionally, the parameters for individual resonators is kept constant, $V = 0.1$ m^3, $L = 10$ m, $S = 100$ m^2, $\rho = 1.2$ and $c = 343.2$ m/s, which results in a resonance frequency of 546.24 Hz. In Fig. 3(a), the height of the tree is modified, while the number of branch divisions at every height step is kept constant.

Figure 3(a) shows that for a tree of height two (2×3) only two resonances are observed, while for a height four (4×3) four resonances are observed. Additionally, even though the individual resonators have the same resonance frequency, the position of the magnitude response peaks is different when a different tree size is chosen.

The evaluation of the effect of the number of branch division is evaluated in Fig. 3(b). In this case, one notices that the number of resonances is the same, independently of the number of branch divisions. Additionally, as the number of branch division gets larger, the frequency difference between the resonant peaks increases.

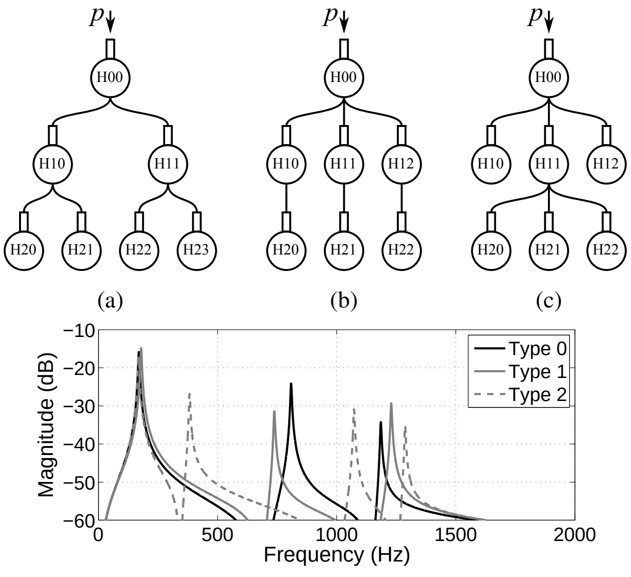


(a)

(b)

Figure 3: Magnitude response showing the effect of changing the Helmholtz resonator tree (a) height and (b) number of branch division.

Figure 4 shows the effect of different tree structures. Two different tree structures with the same number of resonators and the same resonator parameters are evaluated: $V = 0.1$ m^3, $L = 10$ m, $S = 100$ m^2, $\rho = 1.2$ and $c = 343.2$ m/s. The evaluated structures are shown in Fig. 4(a), (b) and (c), with type 0, type 1, and type 2 resonator trees, respectively. Figure 4(c) presents the frequency response for each tree type, where one sees that the type 0 and 1 trees have the resonances at 169.9 Hz, 809.1 Hz and 1185.5 Hz, and 179.5 Hz, 738.5 Hz and 1229.3 Hz, respectively. On the other hand, the type 2 tree has four resonances, at 168.35 Hz, 382.8 Hz, 1073.1 Hz, and 1287.5 Hz.



(a) (b) (c)

(d)

Figure 4: Evaluation of the tree structure type. Schematic view of a (a) type 0 tree, (b) type 1 tree, and (c) type 2 tree. (d) The magnitude response shows the effect of the tree structure.

3. WAVE DIGITAL FILTERS

Wave digital filter (WDF) is an efficient technique for implementing circuit models [17]. By using this technique, each circuit element is represented by one block with one port through which the voltages and currents are mapped onto the wave domain, where the computation of the iteration between circuit components is simplified. Hence, a WDF provides a way to simulate circuit models without matrix inversions and iterative processes for circuits with one nonlinear element.

The main WDF paradigm is mapping the Kirchhoff variables, i.e. voltages and currents, onto the wave variables used in the computation. Each WDF port has an incoming wave A and an outgoing wave B, which are related to the port voltage V and current I by

$$\begin{aligned} A &= k\,(V + R_\mathrm{P} I) &(9)\\ B &= k\,(V - R_\mathrm{P} I)\,, &(10)\end{aligned}$$

where R_P is the port impedance and k is a real-valued scaling factor [17, 3]. On the other hand, the wave variables can be easily converted again into Kirchhoff variables as

$$V = \frac{A+B}{2k} \quad (11)$$

$$I = \frac{A-B}{2kR_\mathrm{P}}. \quad (12)$$

It is important to notice that the port impedance R_P is not necessarily related to the impedance of the element being simulated. In most of the cases, R_P is adjusted in order to provide reflection-free elements, but this is not necessarily mandatory for all elements.

Reflection-free ports constitute an important issue in WDF. Elements using reflection-free ports are also called adapted elements. When this kind of port is used, the output wave of an element has no instantaneous dependency on the incoming wave. This means that the computation of the outgoing wave for that element will imply in no recursive method for the computation of the circuit response. Typically, multiport elements can only have one reflection-free port. One example of how this is represented in the drawings of this work is presented in Fig. 5(a), where a generic tree-port element X is connected to other elements through ports with impedance $R_{\mathrm{p}0}$, $R_{\mathrm{p}1}$ and $R_{\mathrm{p}2}$. In this representation, the dash at the beginning of the line representing the port connection $R_{\mathrm{p}0}$ indicates that this port is adapted.

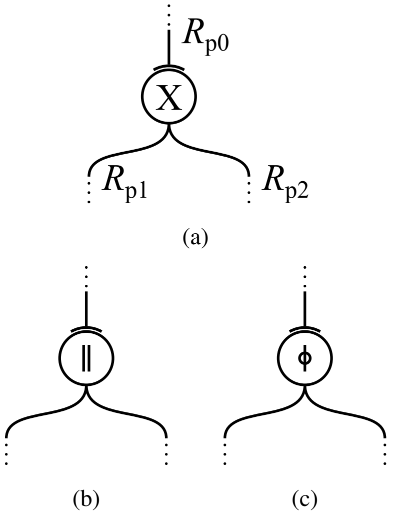


Figure 5: Drawing nomenclature used in this paper. (a) A generic multi-port element with one reflection-free port and tree port (b) series and (c) parallel adaptors with one reflection-free port.

4. HELMHOLTZ RESONATOR TREE TOOL

In order to encapsulate the behavior of Helmholtz resonators, a C++ tool for simulating these resonators was implemented. This tool is built using WDF classes to simulate circuit elements and uses its own classes for generating Helmholtz resonator tree structures and managing the connections between resonators.

Figure 6 shows the basic elements of a Helmholtz resonator implemented in the C++ tool. The acoustic representation of a generic Helmholtz resonator is presented in Fig. 6(a), which may be connected to the cavity of another resonator through its neck, and it may be connected to several other resonators in its cavity. In this system, each resonator can be excited with an external source of volume flow U_0, which simulates something hitting that resonator. The equivalent circuit of the resonator is shown in Fig. 6(b), which includes a capacitor simulating the resonator cavity, the neck model with a resistor and an inductor, and a current source simulates the volume flow disturbance at the cavity. The WDF model for this resonator is shown in Fig. 6(c). This model is rendered generic for the connection of other resonators at the neck and cavity. Additionally, the model combines a series inductor and resistor, and a parallel current source and a capacitor in order to reduce the computational complexity of the system. It is important to notice that, for avoiding delay-free loops in the WDF implementation, the neck can contain only one neighbor.

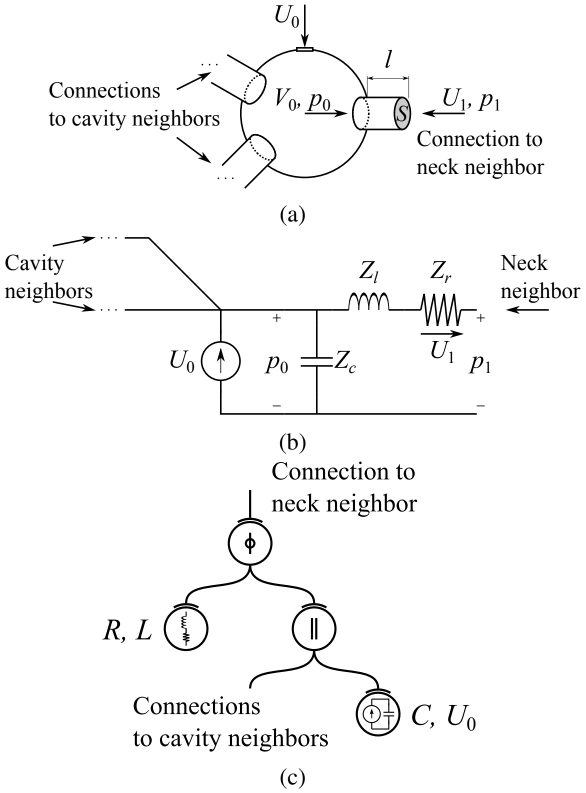


Figure 6: Helmholtz resonator circuit elements: (a) general resonator with neck and cavity connections, (b) equivalent circuit and (c) WDF implementation.

The complexity of implementing the WDF resonator of Fig. 6(c) is given as follows. The series combination of a resistor R and inductor L is implemented with two multiplications, one sum, and one memory element. The parallel combination of a capacitor C and a current source U_0 has one multiplication, one sum, and one memory element. The three-port series and parallel adaptors implies one multiplication and four additions [24]. This results in four multiplications and six additions for each resonator. If an additional voltage/current source is connected to the root element of the Helmholtz resonator tree, it may be implemented using one multiplication and one addition [17, 3].

4.1. VST Plugin

In order to evaluate the performance of the Helmholtz resonator tree, a Virtual Studio Technology (VST) plugin was created. VST plugins are based on a technology developed by Steinberg, which allows the creation of plugins that are easily used in a audio host program [25]. The plugin creates a Helmholtz resonator tree of variable size. The size of the tree is controlled by two parameters, the height of the tree and the number of branch divisions at each height step. Additionally, the plugin enables modification of the physical parameters of the resonators where all the resonators have the same physical dimensions. In this plugin, the input signal is fed as an input pressure at the root resonator and the output signal is the volume flow at this resonator.

4.2. Mobile implementation

A mobile implementation of the Helmholtz resonator tree was developed for the Android tablet. The main interface used was developed using Java in the Android Software Development Kit. The interface with the existing C++ Helmholtz Tool was developed using Android's Native Development Kit framework, which uses the Java Native Interface.

Two approaches were tested for implementing the audio interface. The first used the AudioTrack library. Although this is a straightforward method, when tested with the Android 3.1 OS in a Samsung Galaxy tablet, the maximum allowed frame size was 180 ms. This created a minimum algorithm delay, which is not desirable in real-time applications. In the second approach, the audio interface used the OpenSL ES library to access audio output. When using this library no restrictions on the minimum frame size were observed. Since it was verified that OpenSL ES has reduced algorithm latency compared to AudioTrack, this library was chosen to implement the final mobile application. Additionally, the sampling rate for the application could be modified to between 8 kHz and 44.1 kHz, with audible artifacts being observed only when using the 44.1-kHz sampling rate.

As an example implementation, a seven-element Helmholtz resonator tree was developed, where the parameters of individual resonators can be modified in the user interface. The resonator tree structure is presented in Fig. 7. In this structure, the user may feed a constant pressure signal p into the first resonator's neck or hit any of the resonators applying a short noise burst of volume flow in the cavity of the resonator.

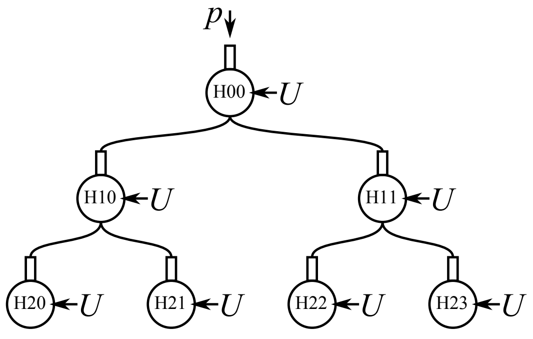


Figure 7: Helmholtz resonator tree structure used in the mobile application.

5. RESULTS

5.1. Computational complexity evaluation

The computational complexity of the Helmholtz resonator tree implementation was evaluated by comparing the simulation time of the VST plugin against the commercial circuit simulation software LTSpice. For this purpose, a tree with a height of four steps and two branch divisions per step was built using the Helmholtz resonator tree tool using a 48-kHz sampling frequency. The same circuit was simulated in LTSpice with the maximum simulation step size set to 1/48000 in order to enable a fair comparison. Additionally, a stereo signal was simulated, meaning that the same network was simulated twice, using both approaches and the input signal consisted of impulses spaced at 10 s with a 120-s long file. The computer used in this test is equipped with a Intel^{R} Core^{TM}2 Quad CPU of 3 GHz, 8 GB RAM and running the Windows 7 operating system. The host for the VST plugin was Audacity 1.3 Beta.

The simulation time was measured for both cases and the CPU usage was monitored while the simulation was running. A first evaluation was performed using a 120-s random noise input signal. When using LTSpice, the measured simulation time was 30 min 56 s (1856 s) with 75% CPU usage. On the other hand, when using the implemented VST plugin, the simulation time was 1 min 52 s (112 s) with 25% CPU usage. In order to evaluate the effect of different input signals in LTSpice, a second test was conducted using a 500-Hz sine wave as input. In this case, the simulation time using LTSpice decreased to 6 min and 42 s (402 s). This shows that for the tested computer configuration LTspice takes 3.35 to 15.5 s to simulate each second of a four steps high Helmholtz resonator tree with two divisions per branch while using 75% of the processing power. For the same circuit, the Helmholtz resonator tree plugin took 0.93 s to simulate each second of the same Helmholtz resonator tree while using only 25% of the processing power. For the case of 100% of CPU power, the time to process 1 s with LTSpice would be between 2.5 and 11.6 s, while for the Helmholtz resonator tree tool it would be 0.21 s. This indicates that the Helmholtz resonator tree implemented as a VST plugin is 10 to 55 times faster than LTSpice for the circuit under test.

5.2. Accuracy evaluation

The accuracy of the Helmholtz resonator tree tool was evaluated comparing the VST plugin's results with the ones obtained with LTSpice. Figure 8 shows the comparison of the frequency response obtained with a Helmholtz resonator tree with a height of 4 steps and 2 branch divisions per step. Additionally, the Helmholtz resonator was built with a cavity $V = 0.1$ m^3, $L = 10$ m, $S = 100$ m^2, $\rho = 1.2$, and $c = 343.2$ m/s. In this result, both the Helmholtz resonator tree tool and the LTSpice results have three resonances at the same frequencies, although some amplitude deviation is observed for the high frequency peaks and the amplitude between these peaks. Overall, this result shows good agreement between the LTSpice reference simulation and the Helmholtz resonator tree tool.

5.3. Mobile application results

The frequency representation of recorded examples using the mobile application is shown in Figs. 9 and 10. These examples were

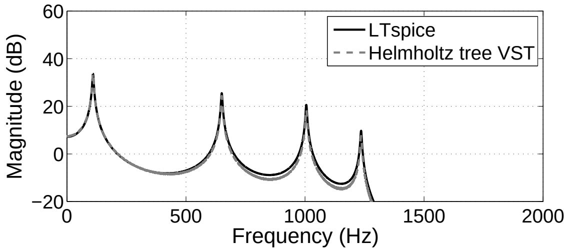


Figure 8: Helmholtz resonator tree tool and LTSpice frequency response comparison for a four steps high Helmholtz resonator tree with two branch divisions per height step.

collected by recording the output of a Samsung Galaxy tablet, operating at a 24-kHz sampling frequency and with a frame size of 10 ms.

Two types of excitation were used. The first one was constant noise pressure applied at the root element of the tree. This is modeled as a voltage source applied at the neck connection in the model of Fig. 6 (c). The second one is a windowed white noise burst of 10 ms simulating the volume flow disturbance caused by hitting a resonator.

Figure 9 shows the results for the default parameters of the application. These parameters include $\rho = 1.2$, $V = 0.1$ m^3, $A = 100$ m^2 and $L = 100$ m for all the resonators. Figure 9 shows four main resonances at 114.5 Hz, 247.2 Hz, 843.2 Hz and 1071.5 Hz. For one excitation signal at H00, Fig. 9(a) shows that most of the energy is concentrated at the higher frequency resonances and small variations when exciting the resonating structure at different positions.

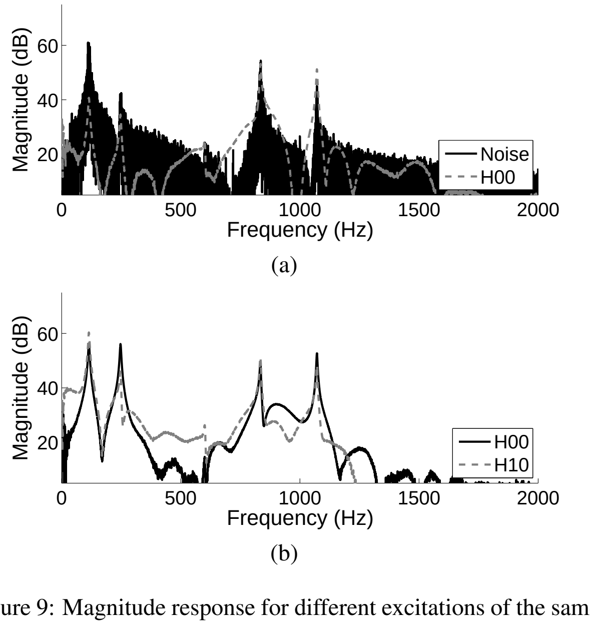


Figure 9: Magnitude response for different excitations of the same Helmholtz resonator structure. (a) Noise pressure excitation at H00 and volume flow disturbance at H00 and (b) volume flow disturbance at H10 and H20.

The results for the second configuration are shown in Fig. 10. In this configuration the air density was set to $\rho = 398$, while the individual resonator parameters were H00 with $V = 0.0575$ m^3, $A = 870$ m^2 and $L = 20.8$ m; H10 with $V = 0.131$ m^3, $A = 1000$ m^2 and $L = 5.75$ m; H11 with $V = 0.301$ m^3, $A = 758$ m^2 and $L = 10$ m; H20 with $V = 0.0912$ m^3, $A = 1000$ m^2 and $L = 100$ m; H21 with $V = 1$ m^3, $A = 1000$ m^2 and $L = 83.1$ m; H22 with $V = 0.173$ m^3, $A = 1000$ m^2 and $L = 100$ m; and H23 with $V = 0.1$ m^3, $A = 870$ m^2 and $L = 10$ m.

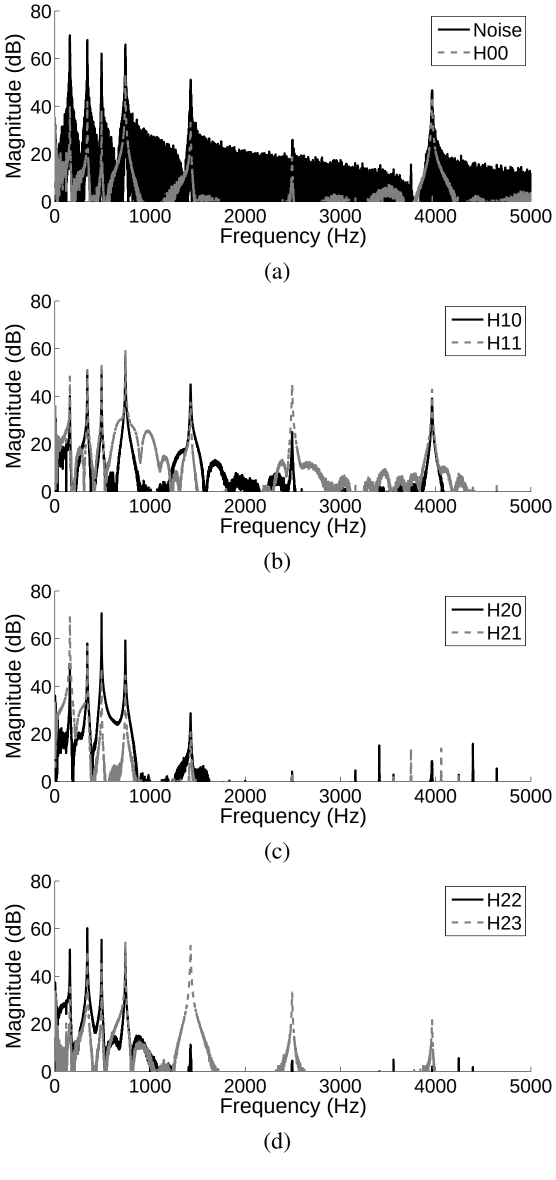


Figure 10: Magnitude response for different excitations of the same Helmholtz resonator structure. (a) Noise pressure excitation at H00 and volume flow disturbance at H00; (b) volume flow disturbance at H10 and H11; (c) volume flow disturbance at H20 and H21; and (d) volume flow disturbance at H22 and H23.

Figure 10 shows that, independently of how the Helmholtz resonator tree is excited, seven distinct resonances are visible at 159 Hz, 342.5 Hz, 491.5 Hz, 742 Hz, 1426 Hz, 2493 Hz and

3962.2 Hz. Moreover, the output energy is observed to be concentrated at some resonances depending on which resonator is excited. When exciting the resonator H20, the energy is concentrated at 491.5 Hz, whereas when exciting H21 energy concentrates at 159 Hz. This results in a distinct timbre every time a different part of the resonator tree is excited, which can be interpreted as hitting different parts of the same musical structure. When comparing the results of Figs. 10 and 9, the resonators with different parameters are seen to yield larger differences when exciting the different resonators. Additionally, the number of resonances increases when setting different parameters for individual resonators.

6. CONCLUSIONS

This paper has introduced the new idea of a Helmholtz resonator tree and has shown how the new structure can be implemented. The equivalent circuit analogy for acoustic systems was reviewed, and the model of a Helmholtz resonator was extended to include connections of several resonators.

Even when all the resonators of a tree have the same parameters, a complex resonator with many resonance frequencies is obtained. In this case, the height of the Helmholtz resonator tree determines the number of resonances observed in the frequency response of the tree. Furthermore, the number of branch divisions per layer influences the spacing between the frequency response peaks.

WDFs can be used to grow Helmholtz resonator trees. A C++ tool called HelmTree was implemented for real-time emulation of Helmholtz resonator tree structures. This tool consists of a WDF block library extended with features to simulate acoustic phenomena in Helmholtz resonator trees. The tool was used to create a VST plugin and a mobile application using the Android operating system. These pieces of software can emulate Helmholtz resonator trees in real time.

The Helmholtz resonator tree tool was compared against a commercial circuit simulator LTSpice. The comparison showed that the simulation results obtained with the Helmholtz resonator tree are in line with results obtained with standard circuit simulators, because no approximations are done. The VST plugin using the Helmholtz resonator tree tool was 10 to 55 times faster than LTSpice. Moreover, the computing times indicate that the Helmholtz resonator tree tool is suitable for real-time simulation of short Helmholtz resonator trees.

Since the Helmholtz resonator tree creates resonant filters related to physical objects, it is useful for synthesizing percussive sounds. Testing with the mobile application revealed that exciting different resonators of the tree leads to a different timbre. This can be directly applied to modeling of a musical instrument whose timbre depends on the position where it is excited. The complex frequency responses obtained with these resonators also appear suitable for filtering musical signals.

The Helmholtz resonator tree introduced in this paper can serve as a method for real-time synthesis and effects processing. One advantage of having parameters related to Helmholtz resonators to build filters is the close relation to understandable physical properties. These parameters are intuitive for users with no training, since it is easy to learn what happens to the sound when the volume of a bottle or its neck length is changed. Supplementary material to this paper, including sound examples, is available at http://www.acoustics.hut.fi/go/dafx12-helmholtztree.

7. ACKNOWLEDGMENTS

The authors would like to thank Nokia Foundation for funding, and Dr. Jyri Pakarinen, Dr. Henri Penttinen, Dr. Antti Jylhä, and Dr. Cumhur Erkut for their helpful comments.

8. REFERENCES

[1] N. H. Fletcher and T. D. Rossing, *The Physics of Musical Instruments*, Springer-Verlag, 1998.

[2] A. B. Morrison and T. D. Rossing, "The extraordinary sound of the hang," *Physics Today*, vol. 62, no. 3, pp. 66–67, Mar. 2009.

[3] V. Välimäki, J. Pakarinen, C. Erkut, and M. Karjalainen, "Discrete-time modelling of musical instruments," *Reports on Progress in Physics*, vol. 69, no. 1, pp. 1–78, Jan. 2006.

[4] J. O. Smith, "Physical modeling using digital waveguides," *Computer Music Journal*, vol. 16, no. 4, pp. 74–91, 1992.

[5] A. Kontogeorgakopoulos and C. Cadoz, "Cordis Anima physical modeling and simulation system analysis," in *Proc. SMC'07, 4th Sound and Music Computing Conference*, Greece, Jul. 2007, pp. 275–282.

[6] S. A. Van Duyne, "Coupled mode synthesis," in *Proc. ICMC'97, International Computer Music Conference*, Thessaloniki, Greece, 1997, pp. 248–251.

[7] J. D. Morrison and J. Adrien, "MOSAIC: A framework for modal synthesis," *Computer Music Journal*, vol. 17, no. 1, pp. 45–56, 1993.

[8] L. Trautmann and R. Rabenstein, *Digital Sound Synthesis by Physical Modeling Using the Functional Transformation Method*, Springer, 2003.

[9] S. A. Van Duyne and J. O. Smith, "Physical modeling with the 2-D digital waveguide mesh," in *Proc. Int. Computer Music Conference*, Tokyo, Japan, 1993, pp. 40–47.

[10] F. Fontana and D. Rocchesso, "Physical modeling of membranes for percussion instruments," *Acta Acustica united with Acustica*, vol. 84, no. 14, pp. 529–542, May 1998.

[11] L. Savioja and V. Välimäki, "Interpolated rectangular 3-D digital waveguide mesh algorithms with frequency warping," *IEEE Trans. Speech and Audio Processing*, vol. 11, no. 6, pp. 783 – 790, Nov. 2003.

[12] F. Avanzini and R. Marogna, "A modular physically based approach to the sound synthesis of membrane percussion instruments," *IEEE Trans. Audio, Speech, and Language Processing*, vol. 18, no. 4, pp. 891–902, May 2010.

[13] S. Bilbao, "Time domain simulation and sound synthesis for the snare drum," *Journal of the Acoustical Society of America*, vol. 131, no. 1, pp. 914–925, Jan. 2012.

[14] K. Dempwolf, M. Holters, and U. Zölzer, "Discretization of parametric analog circuits for real-time simulations," in *Proc. DAFx'10, 13th International Conference on Digital Audio Effects*, Graz, Austria, September 2010, pp. 1–8.

[15] I. Cohen and T. Hélie, "Real-time simulation of a guitar power amplifier," in *Proc. DAFx'10, 13th International Conference on Digital Audio Effects*, Graz, Austria, September 2010.

[16] D. T. Yeh, J. S. Abel, and J. O. Smith, "Automated physical modeling of nonlinear audio circuits for real-time audio effects – part I: Theoretical development," *IEEE Trans. Audio, Speech, and Language Processing*, vol. 18, no. 4, pp. 728–737, May 2010.

[17] A. Fettweis, "Wave digital filters: Theory and practice," *Proc. of the IEEE*, vol. 74, no. 2, pp. 270–327, Feb. 1986.

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

[19] G. De Sanctis and A. Sarti, "Virtual analog modeling in the wave-digital domain," *IEEE Trans. Audio, Speech, and Language Processing*, vol. 18, no. 4, pp. 715–727, May 2010.

[20] J. Pakarinen and M. Karjalainen, "Enhanced wave digital triode model for real-time tube amplifier emulation," *IEEE Trans. Audio, Speech, and Language Processing*, vol. 18, no. 4, pp. 738–746, 2010.

[21] R. C. D. Paiva, J Pakarinen, V. Välimäki, and M. Tikander, "Real-time audio transformer emulation for virtual tube amplifiers," *EURASIP Journal on Advances in Signal Processing*, vol. 2011, pp. 1–15, 2011.

[22] A. D. Pierce, "Basic linear acoustics," in *Springer Handbook of Acoustics*, T. D. Rossing, Ed., chapter 3, pp. 25 – 111. Springer, 2007.

[23] T. D. Rossing and N. H. Fletcher, *Principles of Vibration and Sound*, Springer-Verlag, 2004.

[24] A. Fettweis and K. Meerkotter, "On adaptors for wave digital filters," *IEEE Trans. Acoustics, Speech, and Signal Processing*, vol. 23, no. 6, pp. 516–525, Dec. 1975.

[25] Steinberg, "VST SDK 2.4 documentation," Nov. 2006.