Printone: Interactive Resonance Simulation for Free-form Print-wind Instrument Design

Nobuyuki Umetani[1] Athina Panotopoulou[21] Ryan Schmidt[1] Emily Whiting[2]
[1]Autodesk Research [2]Dartmouth College

(a) (b) (c) (d) (e)

Figure 1: *(a) The user can design the shape of wind musical instruments while guided by sound simulation feedback. (b) The acoustic resonance is simulated by solving the wave equation in the frequency domain. (c) The hollow musical instrument has internal structure and finger holes (d) which produce target tones at specific finger configurations. (e) The user can design unconventional functional wind instruments.*

Abstract

This paper presents an interactive design interface for three-dimensional free-form musical wind instruments. The sound of a wind instrument is governed by the acoustic resonance as a result of complicated interactions of sound waves and internal geometries of the instrument. Thus, creating an original free-form wind instrument by manual methods is a challenging problem. Our interface provides interactive sound simulation feedback as the user edits, allowing exploration of original wind instrument designs. Sound simulation of a 3D wind musical instrument is known to be computationally expensive. To overcome this problem, we first model the wind instruments as a passive resonator, where we ignore coupled oscillation excitation from the mouthpiece. Then we present a novel efficient method to estimate the resonance frequency based on the boundary element method by formulating the resonance problem as a minimum eigenvalue problem. Furthermore, we can efficiently compute an approximate resonance frequency using a new technique based on a generalized eigenvalue problem. The designs can be fabricated using a 3D printer, thus we call the results "print-wind instruments" in association with woodwind instruments. We demonstrate our approach with examples of unconventional shapes performing familiar songs.

Keywords: wind instrument, acoustic simulation, computational fabrication, 3D printing, interactive design

Concepts: •**Computing methodologies** → **Physical simulation;** •**Applied computing** → *Sound and music computing;*

SA '16 Technical Papers,, December 05-08, 2016, , Macao
ISBN: 978-1-4503-4514-9/16/12
DOI: http://dx.doi.org/10.1145/2980179.2980250

1 Introduction

Wind instruments have been prominent in the musical history of all cultures since prehistoric times. They exhibit great diversity in structure and sound and many variations have been explored, such as flutes, horns, saxophones, and recorders. However, most wind instruments are restricted to cylindrical tube-like shapes, for which the resonance properties are well understood. Free-form design is challenging due to the complex relationship between geometry and resonance.

Wind instruments are characterized by an enclosed volume of oscillating air that radiates sound to the listener. Various tones are produced through acoustic resonance, where the specific frequencies of oscillation are magnified through interaction between sound wave propagation and the instrument's geometry. The player excites the oscillation by blowing air into the instrument and controls the resonance by changing the shape of the resonator (e.g., closing and opening finger holes).

Computational acoustics simulates how sound waves travel through air while interacting with solid objects. Resonance of arbitrarily-shaped objects can be computed through numerical simulation, however, this comes with high computational expense and is typically delegated to offline processing. Recently, Allen and Raghuvanshi [2015] presented a real-time interface that simulates resonance of two-dimensional instruments while the user edits the shape. In this paper we introduce a method to interactively simulate fully three-dimensional wind instruments which can then be fabricated and played in person.

We present *Printone*, a tool to design original functional three dimensional wind instruments. Printone aids the user's instrument creation via interactive simulation of the resonance frequency. The inputs are a three-dimensional shape that is to be converted into a musical instrument and a set of target frequencies that should be produced through different open/close patterns of the finger holes. To find a design which produces the target frequencies, the user interactively scales the input shape and edits the placement and size of the finger holes. During this process they are guided by interactive feedback from the simulation. We compute the resonance numerically from the free-form geometric shape, hence the user can explore unconventional musical instruments beyond simple cylin-

drical shapes.

The foremost technical challenge in interactive instrument design is efficient acoustic resonance simulation. Rather than solving for the entire frequency spectrum, we focus on the dominant tone, which is the lowest passive resonance frequency of the instrument's internal cavity. Additionally, we neglect the non-linear coupling effect from oscillation excitation of the mouthpiece.

Frequency-domain simulation provides an efficient way to simulate three-dimensional acoustics by solving only on the boundary. However, this approach solves the acoustics for a predefined frequency, and thus gives no direct clue about where the resonance frequency occurs. To overcome this problem, we first characterize the resonance as a nonlinear eigenvalue problem. Then, we present a technique to efficiently estimate the resonance frequency from a set of sparsely-sampled linear systems in the frequency domain by solving a generalized eigenvalue problem. We further accelerate simulation during geometry editing by introducing sensitivity-based first-order estimation and a local matrix update approach. Our interactive tool also provides an automatic hole size adjustment function, which makes it easy to satisfy multiple frequency targets with different finger configurations.

Our simulation provides interactive response times during design by limiting prediction to a discrete set of resonance frequencies, avoiding the expense of a full spectrum computation. This means we do not simulate the timbre of the instrument. However we show that predicting resonance frequencies is sufficient for hobby-grade fundamental frequency dominant wind musical instruments, where some degree of sound quality can be sacrificed to achieve novel designs. The resulting instruments can be fabricated with commonly available Fused Deposition Modeling (FDM) 3D printers. We validate our model through frequency comparisons between simulated and fabricated instruments, and we demonstrate a wide variety of novel instrument geometries. In summary our contributions include:

- A formulation of resonance as a nonlinear minimum eigenvalue of the system's coefficient matrix.
- Efficient resonance frequency estimation based on frequency-domain acoustic simulation at sparsely sampled frequencies.
- Sensitivity-based acceleration of resonance frequency prediction and finger hole size optimization.
- An interactive user interface for designing functional wind instruments.

2 Related Work

Physics of musical instruments. The physics of conventional acoustic instruments has been extensively studied. Please refer to Fletcher and Rossing [2010] for a comprehensive overview. For analysis specific to wind instruments, acoustic impedance models are commonly used to represent the frequency response of tube-like resonators [Nederveen 1998]. Based on the impedance model, real-time virtual instrument simulation systems have been explored [Cook 1991; Hirschman 1991]. However, impedance models typically simplify the pressure wave to be one-dimensional, and thus are limited to serial assembly of tube-like primitives such as cylinders or cones, where the analytic solution is known. In addition to the properties of resonators, the nonlinear properties of various mouthpieces have been studied [Hirschberg et al. 1996]. To investigate the relationship between more complex instrument shapes and the resulting sound, expensive simulation is employed involving the solution of the compressive Navier-Stokes equation (e.g., [Giordano 2014; Yokoyama et al. 2015]), the Helmholtz equation (e.g., [Lefebvre and Scavone 2012]), or hybrid formulations (e.g., [Da Silva 2008]). However, the computational cost of these methods limits their use in interactive design tools.

Computational free-form instrument design. A few recent works have tackled the problem of free-form musical instrument design. Umetani et al. [2010] presented interactive design of free-form metallophones by integrating real-time fundamental tone simulation into a geometric design system. This work was extended by Bharaj et al. [2015] to incorporate overtone simulation and geometry optimization for 2D and 3D shapes. These methods support the design of objects with specific *contact* sounds, while our goal is to design objects with a desired acoustic resonance frequency. Outside the computer graphics community, shape re-design of musical instruments has been applied to bell optimization [Schoofs et al. 1987] and customization of 3D-printable saxophone mouthpieces based on computational acoustic imaging [Lorenzoni et al. 2013].

Our work is greatly inspired by the work of Allen et al. [2015] which simulates sound for free-form wind musical instruments in 2D. They compute sound output by solving the wave equation in the time domain over a two-dimensional grid. However due to the 2D scope, the resulting designs are not physically realizable for fabrication. Extending to 3D wave propagation is feasible, but significantly more expensive than 2D simulation at the same resolution.

Recently Li et al. [2016] introduced the Acoustic Voxel method, in which a sequence of simple, parametric sound-filter modules are assembled to produce specific frequency-dependent impedance between input and output openings. The frequency response of the modules is pre-computed for the allowable space of parametric variation, making the offline stochastic sound optimization tractable. Although this method can be used to design basic instruments, it is not easily extended to the case of non-parametric free-form shapes as shown in our work, where the precomputation would need to be recomputed after every cavity shape change.

Sound synthesis in graphics. Research in computer graphics has also explored simulation of contact sounds to enhance the fidelity of virtual animation [van den Doel et al. 2001; O'Brien et al. 2001; Bonneel et al. 2008; Ren et al. 2013]. Modeling sound wave propagation in air is essential for realistic acoustic synthesis. Efficient representations of sound propagation have been proposed to model diffraction and reverberation in complex geometries [Funkhouser et al. 1999; Tsingos et al. 2001; Raghuvanshi et al. 2010]. Acoustic transfer accurately models how the object's steady-state vibration travels through air in the frequency domain, and thus is frequently applied in contact sound simulation [James et al. 2006; Zheng and James 2010; Zheng and James 2011]. These contact sound simulations generally take a two-step approach: vibration modes of the solid object are first computed, and then acoustic transfer is computed for each vibration mode discretely. In contrast, we tackle resonance simulation, where the vibration frequencies are not given from solid modal vibration. To identify the resonance frequencies it is necessary to scan the wave equation over a range of frequencies.

Asymptotic expansion of acoustic transfers was introduced by Li et al. [2015] to cover a range of frequencies, allowing interactive synthesis of contact sounds for various material parameters. However, precomputation costs are prohibitively high and specific to a particular shape, preventing applicability to interactive shape design. Further, sharp peaks in acoustic radiation at the resonance frequency are difficult to resolve by polynomial expansion. Rather than expanding the acoustic transfer, we directly compute the resonance frequency by exploring the spectral properties of the system's coefficient matrix.

Eigenformulation for acoustic resonance. To directly compute a resonance frequency, we model the resonance frequency as the frequency that gives the *smallest* minimum eigenvalue of the systems coefficient matrix. This approach is related to the Determinant Search Method (DSM), which models the resonant frequency as the frequency that give the *zero* minimum eigenvalue (i.e., the matrix is singular), and then finds a root of the determinant of the coefficient matrix using the Newton-Raphson iteration. Various approaches have been proposed to efficiently detect singularities of the coefficient matrix for acoustic problems inside fully enclosed cavities [Ali and Rajakumar 2004]. However, for our musical instrument design problem, we cannot employ DSM because the cavity is not fully enclosed and the sound is damped as it travels to infinity, thus the coefficient matrix is typically not singular.

Physics-based cavity optimization. We provide a design interface for creating functional wind instruments, which relies on the shape of internal cavities. Prior work has investigated optimizing cavity shapes to satisfy functional requirements, such as balance [Prévost et al. 2013], spin-ability [Bächer et al. 2014], buoyancy [Musialski et al. 2015; Wang and Whiting 2016] and strength-to-weight optimization [Lu et al. 2014]. Cavity shape optimization can also be performed efficiently using reduced-parameter shape spaces [Musialski et al. 2015]. We leave automatic optimization of internal cavity shape for future work, and instead focus on optimizing the finger holes that connect the cavity to the exterior.

3 Mechanics of Wind Instruments

In this section, we briefly describe the mechanics of wind musical instruments and breakdown the components simulated by our design tool. We use the word *tone* to refer to the sound produced by a single frequency, purely sinusoidal waveform. Natural sound is made up of many tones: the lowest frequency corresponds to the *fundamental tone*. Higher frequencies above the fundamental are called *overtones*. While overtones determine the *timbre* – the quality or color of a musical note – the pitch of the sound is determined by the fundamental tone.

Wind musical instruments typically consist of two components: a *mouthpiece* and a *resonator cavity*. The mouthpiece excites vibrations of air in a range of frequencies to initiate and maintain the resonance. The resonator is an enclosed cavity inside the instrument which amplifies oscillation of specific frequencies. Only the frequency corresponding to the resonance frequency of the cavity survives and will be audible.

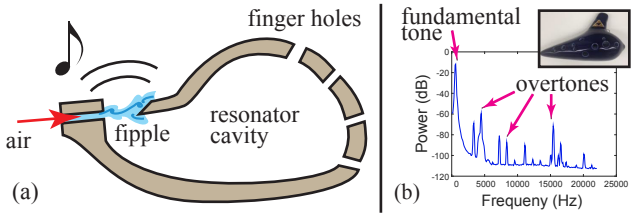


Figure 2: *Mechanism of a fipple-driven wind instrument. (a) Blown air reaches the fipple, creating an oscillation with a range of frequencies. The resonator cavity amplifies sounds at specific frequency tones. (b) In a typical ocarina spectrum, the fundamental tone is dominant over overtones.*

Fig. 2 illustrates the basic configuration of our musical instruments. Our mouthpiece is the *fipple*, which is commonly used for recorders or ocarinas. The fipple excites oscillation when a narrow airstream hits a sharp edge called the *bladed edge*, thus creating turbulence of many frequencies downstream. Other common types of mouthpiece exist, such as reeds, which are mechanically vibrating thin pieces of material (e.g. used by saxophones and clarinets). Our focus is primarily on fipples, since the vibration properties of reeds are very sensitive to material stiffness and thus are difficult to fabricate with commonly available FDM printers.

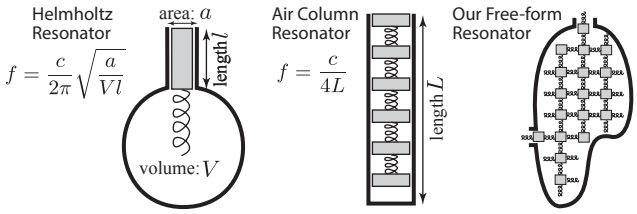


Figure 3: *The Helmholtz resonator and the air column resonator are the known resonator models for simple primitive geometries. The resonance frequencies f for these models are approximated with simple equations. When the geometry is more complex (e.g, free-form and multiple finger holes) we need to solve the wave equation to predict the resonance frequency.*

Resonance frequencies are determined by the shape of the resonator. There are two existing models of acoustic resonance for simple geometries. One is the Helmholtz resonator which applies to containers with near-spherical cavities, e.g. an empty bottle (see Figure 3-left). The air in the neck behaves as a mass in a mass-spring simulation, and the chamber as a spring. The frequency can be approximated simply from the volume of the resonator chamber and the neck dimensions. This model breaks down for more general shapes where the neck and the resonator chamber cannot be clearly separated. The other known resonance model is the air column resonator, in which the interference of the incident and reflected wave inside a cylinder constructs a standing wave (see Figure 3-middle). This can be modeled as a one-dimensional distributed mass-spring system, the frequency of which can be computed by the length of the cylindrical tube. For more complex geometry these models no longer apply, and thus computational simulation of acoustic wave propagation is required, which is the main focus of our paper.

While the resonator affects the resonance frequencies, the mouthpiece affects the amplitude of these frequencies. Typically the oscillation of the mouthpiece and the resonance are coupled. In other words, the oscillation introduced by the fipple is affected by the air movement driven by the resonance. Detailed analysis of the oscillation induced by the fipple requires expensive turbulence analysis using computational fluid dynamics [Giordano 2014]. In order to prioritize interactive functional instrument design, we ignore this coupling and only simulate the dominant frequency of the sound. Generally, the dominant frequency is one of the resonance frequencies and can be controlled by the player by changing the embouchure or the speed of the blowing. In fipple-driven wind instruments, the fundamental resonance frequency is typically dominant at moderate blowing speeds [Fletcher and Rossing 2010]. Thus, we found that predicting fundamental resonance frequencies is sufficient for designing hobby-grade musical instruments that can play simple melodies composed of a set of dominant tones. Incorporating timbre by simulating the coupling between the mouthpiece and resonator [Allen and Raghuvanshi 2015] would result in higher-quality sound, but is also significantly more computationally intensive for 3D geometries, and we leave it to future work to discover ways to do this at real-time rates.

4 User Interface

This section describes our wind instrument design tool. Fig. 4 shows a screen shot of our system. The 3D user interface consists of geometric editing tools to place finger holes and the fipple. Resonance simulation results are provided as feedback, allowing the user to adjust geometry and finger hole positions that generate desired resonance frequencies.

Our tool offers an interactive interface to convert an input 3D shape into a functional wind instrument. Interactivity is very important in free-form musical instrument design to satisfy multiple requirements. First, the resonance frequency should be as close to the target frequencies as possible. This is straightforward when each frequency corresponds to a configuration where only one hole is open – the user can adjust the one hole independently of the rest of the design. However, the design process becomes more complex when the user wishes to achieve a high number of notes with few holes, therefore needing to rely combinatorics of different hole configurations (e.g. 3 holes have 8 open/close combinations). Second, the original 3D shape needs to be preserved as much as possible. A free-form musical instruments should balance the functionality as a wind-instrument with the aesthetics of the original input shape. The user tries to minimize the number of finger holes and chooses their locations according to their design intent (e.g., do not place finger holes on a face). Finally, certain ergonomics need to be considered, e.g., the musical instrument needs to be graspable at any finger configuration. With an interactive feedback the user can explore solutions to these requirements.

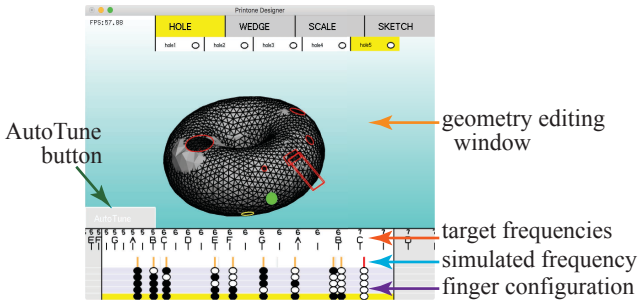


Figure 4: *Screen-shot of our wind instrument design tool. The user can manipulate the size and location of finger holes and can open and close these holes while listening to the simulated sound. The fundamental frequency is visualized together with finger configurations and target notes.*

Geometric editing. Our tool imports existing polygonal meshes, which specifies the exterior shape of the instrument. The tool automatically hollows out the shape to generate an internal cavity. Then, the user positions the fipple and finger holes by clicking on the object's surface. The user can also uniformly scale the entire instrument and adjust the orientation of the fipple. The size of the finger holes can be continuously adjusted to manipulate the frequency of the resonance. The user can also specify whether the finger holes are open or closed to test the sounds produced by their instrument design.

Simulation feedback. In the bottom of the window (Fig. 4), the tool shows finger hole open/close patterns along with their simulated resonance frequencies. At the same time, the speaker plays the sound with the simulated resonance frequency in the current finger configuration. During continuous adjustment of the size of an individual hole, the tool approximates the change of the resonance frequencies based on sensitivity analysis in real-time (>30FPS) and updates the audio and visual feedback. During idle moments, the system computes more accurate frequency estimations for each finger configuration. Each update takes only a few seconds, and thus it does not interfere with the user's interactive editing session.

Interactive finger hole size optimization. Our tool supports an optional *automatic* optimization of the finger hole sizing to simplify the design process. This is a helpful capability because achieving multiple resonance frequencies requires adjusting multiple finger holes in a coordinated way. To use the automatic optimization, the user first specifies finger configurations and their frequency targets. Then the user presses the "AutoTune" button, and the tool automatically resizes each finger hole such that it best achieves the frequency targets. This simple addition allows the user to save significant time and produce more accurate instruments. The optimization procedure is described in Sec. 7.

Design procedure. We assume the user has some prior knowledge about the design of wind instruments, namely that (i) larger cavities lower the resonance frequency, and (ii) larger and/or more holes increase the resonance frequency. When a finger hole is closed, the size and the position of that hole does not influence the resonance frequency. In other words, closing holes reduces the number of control parameters that affect resonance frequency. Hence, after the mouthpiece is positioned, the user first scales the instrument such that the lowest target note is produced when all holes are closed. Next, the user opens one hole and adjusts its size and location to achieve the second lowest frequency. The user then continues opening holes and adjusting their size and location to achieve sequentially-higher target frequencies, until all notes can be produced.

With this simple procedure, $\mathcal{N}$ holes are sufficient to produce $\mathcal{N}+1$ target tones, since each newly-opened hole can be individually adjusted to produce the next target tone. However, since there are actually $2^{\mathcal{N}}$ finger configurations for $\mathcal{N}$ holes, fewer holes can also be used. In practice, it is easy to achieve two or three higher target tones with each new hole. This involves some trial-and-error hole resizing and positioning, but interactive feedback makes the process straightforward. The optimization function is mainly used for fine-tuning instead of automatic discovery of new hole designs, since the user still needs to place the holes and specify the correspondence between target frequencies and finger configurations. The user is recommended to place the mouthpiece at the beginning of the design session, since the relocation of the mouthpiece alters all the achieved resonance frequencies. For the shapes in Fig.17, the resonance frequencies changes up to 20 Hz as a result of the mouthpiece relocation. The resonance frequencies are usually insensitive to mouthpiece location for the round cavity shapes. This is because the fipple is essentially a hole which is always open (see Helmholtz resonator model in Fig. 3). Our AutoTune also accommodates posterior relocation of the mouthpiece by adjusting hole sizes.

Using the approach described above, a novel wind instrument can be designed in about half an hour. This includes time spent designing aesthetics and ergonomics. For example, the mouthpiece must be unobstructed, the finger configurations should allow reliable grasping, and the mouthpiece and finger holes should not disrupt salient features. We leave it to future work to automatically guide the user towards "good" instrument ergonomics, for example using the graspability framework of [ElKoura and Singh 2003].

5 Acoustic Resonance

Acoustic resonance is a phenomenon where the output of an acoustic system becomes very large compared to the magnitude of input at specific input frequencies. Our goal is finding such resonance frequencies for a cavity shape, where the enclosed internal hull is connected to the exterior with small holes. Typical methods usually involve specifying locations of a sound-emitting point and a sound-observing point, and then find the ratio between input and output [Kirkup 1998]. Instead, we formulate resonance as a minimum eigenvalue problem of the system's coefficient matrix, which is solely defined by the object's geometry and independent of any source and observation points. This formulation leads to the fast approximate resonance computation presented in Section 6.

5.1 Background on Acoustic Simulation

This section briefly explains the background of the frequency domain acoustic simulation based on the notation used in [James et al. 2006]. We consider a problem where sound waves, which are emitted from external vibration sources, are reflected from an object's surface S. Resonance is a static phenomenon, i.e., the magnitude of a sound wave is maintained via excitation from an external vibration source. In this case it is more convenient to formulate the problem in the frequency-domain, where all the variables are converted to static via temporal Fourier transform. The pressure field is represented as a set of harmonically vibrating fields $p(\mathbf{x})e^{+i\omega t}$, where ω is the angular frequency of the vibration. The wavenumber is given as $k = \omega/c$, where c is the speed of sound in the medium (approximately 340 m/s in air). The spatial distribution of pressure is denoted $p(\mathbf{x}) \in \mathbb{C}$, and satisfies the Helmholtz equation

$$(-\nabla^2 - k^2)p(\mathbf{x}) = f_{ext}(\mathbf{x}), \quad \mathbf{x} \in \Omega, \tag{1}$$

where $f_{ext}(\mathbf{x})$ is the source term that represents a forced vibration from an excitation mechanism located inside the medium Ω. For example, a point source at $\mathbf{x}_{src}$ with unit magnitude results in $f_{ext}(\mathbf{x}) = \delta(\mathbf{x}_{src} - \mathbf{x})$, where δ is the Dirac delta function. We assume the object's surface is very hard and reflects the sound completely, i.e., the normal velocities of the air particles are zero at the surface. This can be written as a Neumann boundary condition

$$\frac{\partial p}{\partial \mathbf{n}} = 0, \quad \text{on} \quad S. \tag{2}$$

The Helmholtz equation can be solved with many different discretization methods. We use the Boundary Element Method (BEM), which compactly represents the three-dimensional pressure distribution in Ω with the pressure values on the object's surface mesh S. Since the BEM-based formulation naturally satisfies the *Sommerfeld radiation condition*, it can model the radiation from the surface to infinitely far away. The BEM discretizes the Helmholtz equation into a matrix formulation

$$\mathbf{A}(k)\mathbf{p} = \mathbf{f}_{ext}, \tag{3}$$

where $\mathbf{f}_{ext} \in \mathbb{C}^N$ is a vector storing incident pressure at vertices and $\mathbf{p} \in \mathbb{C}^N$ is the resulting reflection pressure value at the vertices of the object. The coefficient matrix $\mathbf{A}(k) \in \mathbb{C}^{N\times N}$ is a dense non-Hermitian matrix that varies with respect to the wavenumber k. Details of the matrix $\mathbf{A}$ are given in (A.5) in the supplemental material. Note that $\mathbf{A}$ changes nonlinearly with respect to k, since the first derivative of the coefficient matrix is k-dependent. Once the reflection pressure $\mathbf{p}$ is computed by solving (3), the pressure value at the observation point inside the medium $\mathbf{x}_{obs} \in \Omega$ is given by integrating pressure over the surface. See (A.4) in the supplemental material.

5.2 Resonance as a Small Minimum Eigenvalue

Typically, the resonance is defined as the frequency where pressure at the observation point $p(\mathbf{x}_{obs})$ has a local maximum for a specific source point $\mathbf{x}_{src}$. Such local maximums are typically detected with a *frequency sweep*, i.e., solving for the pressure at $p(\mathbf{x}_{obs})$ at many different wavenumbers. Frequency sweeps are very computationally intensive, we thus first formulate resonance in a different way such that we do not need to compute the pressure at a specific observation point.

The resonance frequency does not depend on the observation position, e.g., the fundamental frequency of a flute does not depend on a listener's position. Furthermore, the resonance frequency typically does not depend on the position of excitation. Hence, instead of comparing the values at these points, we compare the overall magnitude of incident excitation and reflection on the surface. Specifically, we define resonance as the frequency where the magnitude of reflection pressure on surface $|\mathbf{p}|$ takes the maximum value compared to the magnitude of the incident pressure $|\mathbf{f}_{ext}| = |\mathbf{A}(k)\mathbf{p}|$, i.e., the magnitude of the incident wave reaches a minimum relative to the magnitude of the reflection pressure.

$$k_{resonance} = \arg\min_k \left(\min_{\mathbf{p}} \frac{|\mathbf{A}(k)\mathbf{p}|}{|\mathbf{p}|} \right), \tag{4}$$

Note that the norm of the reflection pressure $|\mathbf{p}|$ corresponds to the sum of acoustic energy that the vertices emit and the norm of the incident pressure $|\mathbf{f}_{ext}|$ corresponds to the sum of acoustic energies that the vertices receive. The squared norm of the vector is defined as $|\mathbf{p}|^2 = \langle \mathbf{p}, \mathbf{p} \rangle = \mathbf{p}^T \bar{\mathbf{p}}$, where $\bar{\mathbf{p}}$ is the conjugate of $\mathbf{p}$.

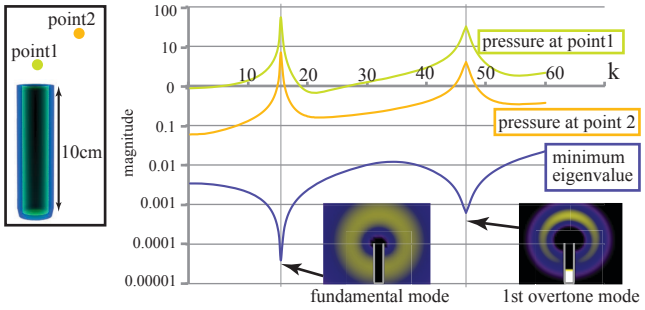


Figure 5: *Sound is emitted and observed at the two points near a tube with one end open. The magnitude of the observed reflection sound has a significant relationship with the minimum eigenvalue of the acoustic system.*

Given a fixed wavenumber k in (4), the problem of finding the minimal reflection pressure $\mathbf{p}$ is equivalent to finding the magnitude of the minimum eigenvalue of matrix $\mathbf{A}(k)$. If this minimum eigenvalue is small, the magnitude of reflection pressure becomes large compared to the magnitude of incident pressure (see a more detailed discussion in Appendix B). Figure 5 shows an air column vibration example, where the strong relationship between minimum eigenvalue and pressure wave magnitude is shown at several observation points. Since the matrix $\mathbf{A}(k)$ changes nonlinearly with respect to k, finding k that minimizes the smallest eigenvalue of $\mathbf{A}(k)$ by sampling many different wavenumbers is very time consuming. Section 6 describes our novel approximation to solve for this wavenumber efficiently.

Relationship to the modal vibration. Our resonance definition (4) is a generalization of a modal vibration, where the coefficient matrix can be decomposed into a positive-definite Hermitian matrix $\mathbf{K}$ and a squared-frequency-scaled identity matrix as

$\mathbf{A}(k) = \mathbf{K} - \omega^2\mathbf{I}$. Solving (4) for such problems is identical to the solution of the eigenvalue problem: $\mathbf{A}(k)\mathbf{p} = \mathbf{K}\mathbf{p} - \omega^2\mathbf{p} = 0$ for non zero $\mathbf{p}$. However, such a matrix decomposition is impossible for exterior acoustic simulations (the exterior acoustics means the medium Ω is unbounded). In our BEM simulation, the second derivative of the coefficient matrix with respect to the k is k-dependent (see (A.5) in the supplemental material). Moreover, the matrices for exterior acoustic problems are typically non-singular, thus there is no non-trivial solution for $\mathbf{A}(k)\mathbf{p} = 0$.

5.3 Spectral Properties of the Coefficient Matrix

We defined the resonance frequency as that where the minimum eigenvalue of the system's coefficient matrix is small. Before explaining how to find such a frequency, we briefly explain why cavity shapes produce resonance sound in terms of the spectral properties of the coefficient matrix. Acoustic resonance is a unique phenomena observed in cavities, i.e., convex shapes typically do not have resonance modes. Since the cavity resonance depends on the shape of the cavity, i.e., the exterior shape typically does not affect resonance, we only consider the properties of the coefficient matrix related to the interior. Note that this is an informal observation, rather than a mathematical proof, which we found to provide useful insight into the connection between cavity shape and the spectral properties of the matrix. A more precise discussion can be found in mathematics literature [Brüning et al. 2009].

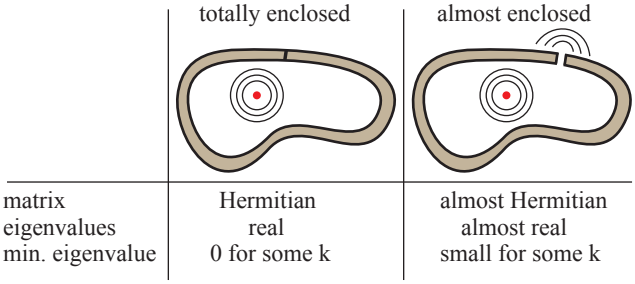


Figure 6: *Spectral properties of the coefficient matrix for totally enclosed cavity (left) and cavity with small holes (right). Since the difference in the coefficient matrix is small, the spectral properties of the cavity-with-holes inherits these of the totally enclosed cavity.*

The coefficient matrix $\mathbf{A}(k)$ is a discretization of the operator $\mathcal{A}(k) = \mathcal{L} - k^2\mathcal{I}$, where the $\mathcal{L}$ is the Laplace operator $(-\nabla^2)$ and the $\mathcal{I}$ is the identity operator. The domain of the acoustic medium Ω is a cavity where the internal hull and the outside region are weakly connected with small finger holes. Assuming that the holes are small relative to the overall cavity surface area, we expect that the spectral properties of the matrix are numerically similar to the properties for the matrix of a totally enclosed cavity.

In a fully enclosed region without holes, under the Neumann boundary condition (2), the Laplace operator is non-negative and Hermitian, which means it has real non-negative eigenvalues. When the squared wavenumber k^2 is identical to one of the eigenvalues of the Laplace operator $\mathcal{L}$, the operator $\mathcal{A}$ has a zero eigenvalue. The numerical discretization of an operator inherits its spectral properties, thus matrix $\mathbf{A}(k)$ is also expected to be Hermitian and have real eigenvalues. Furthermore, at some specific wavenumbers it has a zero eigenvalue.

An infinitesimal hole that connects the internal cavity and the exterior region produces only a small change to the matrix $\mathbf{A}(k)$ (see Figure 6). Since $\mathbf{A}(k)$ for an open boundary is a non-singular matrix, such perturbation *continuously* changes the eigenvalues. Hence, as long as the hole is small, we can still expect that $\mathbf{A}(k)$ is numerically close to Hermitian (meaning that its eigenvalues are numerically close to real-valued, i.e. the magnitude of the imaginary component is small relative to the real component). Hence, for some specific resonance wavenumber, the smallest eigenvalue of $\mathbf{A}(k)$ will still be much smaller than the second smallest eigenvalue. Figure 7 illustrates how a small hole in a cavity smoothly changes the profile of smallest and second smallest eigenvalues.

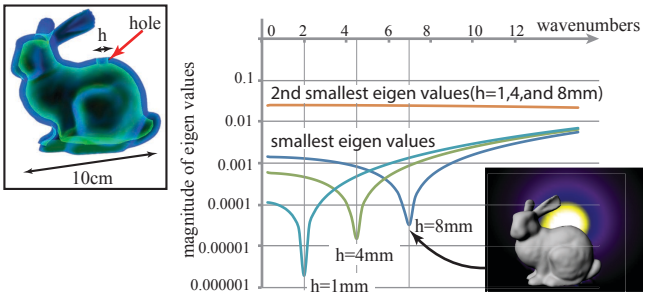


Figure 7: *The minimum and the second-smallest eigenvalues of the coefficient matrices for a cavity with a small hole. As the hole size grows, the profile of the eigenvalues continuously shifts.*

Origin of the Helmholtz resonance. The fully enclosed cavity with the Neumann boundary condition (2) has a trivial zero eigenvalue at $k = 0$ (i.e., zero frequency) where the interior pressure is constant. A small hole in the cavity perturbs this eigenmode, and thus there is always a small eigenvalue with low frequency. This low-frequency resonance corresponds to the Helmholtz resonance because as the hole becomes smaller the Helmholtz resonance frequency converges to zero [Mohring 1999].

6 Approximating Resonance

In the previous section, we defined the resonance frequency as the frequency which produces the smallest minimum eigenvalue of the coefficient matrix $\mathbf{A}(k)$. Computing the matrix $\mathbf{A}$ requires $\mathcal{O}(N^2)$ operations and is very expensive, thus we cannot compute $\mathbf{A}(k)$ for many different wavenumbers if our system is to run at interactive rates. In this section, we present a method to estimate resonance from a few discretely sampled wavenumbers.

6.1 Generalized Eigenvalue Problem Approximation

We sample the wavenumber at k_m $(m = 1, \ldots, M)$. At each discretely sampled wavenumber k_m, we approximate the k-varying coefficient matrix $\mathbf{A}(k)$ with a first-order Taylor expansion as

$$\mathbf{A}(k) \simeq \mathbf{A}_m + (k - k_m)\mathbf{D}_m, \tag{5}$$

where $\mathbf{A}_m = \mathbf{A}(k_m)$ and $\mathbf{D}_m = \partial\mathbf{A}(k)/\partial k$ is the derivative of the coefficient matrix at k_m. The derivative of the operator $\mathcal{A}(k)$ with respect to the wavenumber gives a scaled identity operator $\partial\mathcal{A}(k)/\partial k = -2k\mathcal{I}$. Since $\mathbf{D}_m$ is essentially a discretization of an identity operator, it is close to a real-valued diagonally-dominant matrix (similar to the mass matrix in a solid mechanics simulation).

Finding the resonance wavenumber (4) using this first-order approximation (5) is closely related to the generalized non-Hermitian eigenvalue problem:

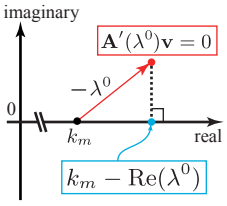
$$\mathbf{A}'(\lambda)\mathbf{v} = 0, \quad \mathbf{A}'(\lambda) = \mathbf{A}_m - \lambda\mathbf{D}_m, \tag{6}$$

where $\lambda \in \mathbb{C}$ is the eigenvalue and $\mathbf{v} \in \mathbb{C}^N$ is an eigenvector.

Since $\mathbf{A}'$ is just a shifted version of $\mathbf{A}$: $\mathbf{A}(k) = \mathbf{A}'(k_m - k)$, if $(k_m - k)$ is close to λ, the minimum eigenvalue of $\mathbf{A}(k)$ becomes small. In other words, when $(k - k_m)$ is the real part of λ, a vector $\mathbf{p}$ similar to $\mathbf{v}$ makes the $|A(k)\mathbf{p}|/|\mathbf{p}|$ small. Since the first order Taylor expansion (5) is valid only when the wavenumber is close to the sampled k_m, we are interested in finding a resonance wavenumber nearest to the sampled k_m. The eigenvalue of smallest magnitude λ^0 gives a resonance wavenumber as

$$k_{resonance} = k_m - \mathrm{Re}(\lambda^0). \tag{7}$$

This is a good approximation because the eigenvalue λ is close to a real number for the cavity resonance. The generalized eigenvalue problem (6) is identical to the conventional eigenvalue problem for matrix $\mathbf{A}_m\mathbf{D}_m^{-1}$. As we discussed in the previous section, $\mathbf{A}_m$ has nearly-real eigenvalues. Since the $\mathbf{D}_m$ is close to a real-valued diagonally-dominant matrix, multiplying by its inverse does not alter the properties of $\mathbf{A}_m$.



Inverse power method. The eigenvalue with smallest magnitude λ^0 can be easily computed using the inverse power method [Demmel et al. 2000]. More specifically, by repeatedly multiplying the inverse of the matrix $\mathbf{A}_m\mathbf{D}_m^{-1}$ with an arbitrary vector $\mathbf{v}$ as $\mathbf{v} \leftarrow \mathbf{D}_m\mathbf{A}_m^{-1}\mathbf{v}$, the Rayleigh quotient

$$\theta = \frac{\langle \mathbf{D}_m\mathbf{A}_m^{-1}\mathbf{v}, \mathbf{v}\rangle}{|\mathbf{v}|^2} \tag{8}$$

converges to the inverse of the minimum eigenvalue. Convergence is defined as $|\mathbf{w} - \theta\mathbf{v}|/|\mathbf{v}| < \epsilon$, we use $\epsilon = 10^{-5}$ in this paper. Once the iteration converges, the inverse of the Rayleigh quotient gives the minimum eigenvalue $\lambda^0 = \bar{\theta}/|\theta|^2$.

6.2 Sparse Wavenumber Sampling

We uniformly sample the wavenumbers such that the range of these frequencies covers the target frequencies. The computation of the matrix derivative $\mathbf{D}_m$ is as expensive as the computation of the coefficient matrix $\mathbf{A}_m$. Hence, as shown in Fig.8-left, we approximate the matrix derivative with a central differential of sampled coefficient matrices

$$\mathbf{D}_m = \frac{\mathbf{A}_{m+1} - \mathbf{A}_{m-1}}{k_{m+1} - k_{m-1}}. \tag{9}$$

For the first and last sampled wavenumber k_1 and k_M, we cannot compute the central difference and we must use forward and backward finite difference schemes.

Entry-wise accuracy of linearization. The accuracy of the linear approximation of (5) under the finite difference (9) depends on the density of the wavenumber sampling. Hence, we carefully determine sampling intervals such that the error of the linearization is below a certain threshold for all entries of the coefficient matrix. The (i, j)-entry of the coefficient matrix can be written as $a_{ij}(k) = a'_{ij}g(kr_{ij})$, where $a'_{ij} \in \mathbb{C}$ is a scaling factor independent of the wavenumber, $g(\gamma) = \exp(+i\gamma)(1 - i\gamma)$, and r_{ij} is the distance between point i and point j (see (A.5) in the supplemental material). The Figure 8-right illustrates the trajectory of the curve $g(\gamma)$. The second derivative of this curve ($|g''| = |1 + \gamma|$) indicates that the error of our linear approximation is on order of $\Delta k r_{ij}|1 + kr_{ij}|$, where Δk is the wavenumber sampling interval. Here, k is bounded by the wavenumber of maximum target frequency k_{max} and r_{ij} is bounded by the diameter of the circumscribed shape D. Hence, in our paper, we chose the wavenumber sampling interval as $\Delta k < 1/D|1 + k_{max}D|$, which can guarantee that the entry-wise linearization error is less than 3%, which we believe is a sufficient accuracy. For example, if the cavity shape is about 10 cm and the highest target frequency is 1000 Hz, then the wavenumber can be sampled at roughly 150Hz interval ($\Delta k = 3$).

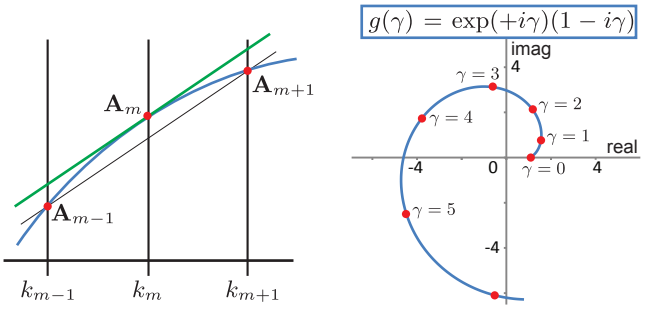


Figure 8: *(Left) our first-order approximation of the coefficient matrix at a wavenumber k_m (green line). (Right) an entry of a BEM coefficient matrix changes proportionally to the curve $g(\gamma)$ above. Denser sampling is required as curvature increase with γ.*

Culling inaccurate resonance estimation. We can estimate a resonance frequency for each of the sampled wavenumbers k_m $(m = 1, \ldots, M)$. After the M-number of resonance frequency estimations, we must determine which is the most accurate. The linear approximation in (5) and (9) is accurate if $k_{resonance}$ and k_m are close. Hence, we pick the most accurate estimation by choosing the one that has the minimum difference between $k_{resonance}$ and k_m. In other words, we choose a computed resonance wavenumber as the best estimation if the difference is smaller than half of the sampling interval $|k_{resonance} - k_m| < 0.5\Delta k$. Furthermore, the convergence ratio of the inverse power method is bounded by the ratio between the magnitudes of the first and second smallest eigenvalues $|\lambda^0|/|\lambda^1|$. Since the $|\lambda^0|/|\lambda^1|$ is very small ($< 10^{-3}$) around the resonance wavenumbers (see Fig. 7), several iterations are enough to reach convergence. If k_m is very far from $k_{resonance}$, convergence is slow. Hence, if a segment does not reach convergence after N_{iter} iterations, we regard this segment as far from the resonance frequency (we use $N_{iter} = 5$ in this paper).

We use BiCGStab method [van der Vorst 1992] to solve the linear system. We did not use any preconditioner in the iterative solve, but we observed a stable convergence in the wavenumbers we use to simulate fundamental resonance frequency. The overall algorithm is shown in Algorithm 1.

Algorithm 1: resonance detection algorithm at wavenumber k_m

while $\#iteration < N_{iter}$ **do**
  $\mathbf{v} \leftarrow \mathbf{v}/|\mathbf{v}|$; /* normalizing */
  $\mathbf{x} \leftarrow \mathbf{A}_m^{-1}\mathbf{v}$
  $\mathbf{w} \leftarrow (\mathbf{A}_{m+1}\mathbf{x} - \mathbf{A}_{m-1}\mathbf{x})/(k_{m+1} - k_{m-1})$
  $\theta \leftarrow \langle \mathbf{w}, \mathbf{v}\rangle$; /* Rayleigh quotient */
  if $|\mathbf{w} - \theta\mathbf{v}| < \epsilon$ **then**
    $\lambda^0 = \bar{\theta}/|\theta|^2$
    $k_{resonance} = k_m - \mathrm{Re}(\lambda^0)$
    if $k_{resonance} \notin [k_{m-1/2}, k_{m+1/2}]$, discard this solution
  $\mathbf{v} \leftarrow \mathbf{w}$

7 Hole Size Optimization

We have described our novel framework to simulate a resonance frequency in response to geometry edits with a latency of several seconds. Our system is usable with these response rates, however, because the resonance frequencies are sensitive to hole sizes, the user may need to make many trial-and-error adjustments. We enhance this design workflow with a faster real-time tone estimation during hole resizing, and an automatic optimization of the finger hole sizes. We use sensitivity analysis for both of these tasks.

Sensitivity-based real-time response. Let ϕ be the diameter of a finger hole. When the system is idle, the tool computes a sensitivity $\partial k_{resonance}/\partial\phi$, which denotes the linear approximation of the resonance wavenumber change with respect to the change in hole size. When the user is interactively changing the hole diameter ϕ, the system instead estimates the resulting frequency change as

$$k'_{resonance} = k_{resonance} + \frac{\partial k_{resonance}}{\partial \phi}\Delta\phi. \tag{10}$$

Since this estimation does not involve expensive matrix operations, it can be computed at real-time rates (>30FPS). The user can then quickly adjust the hole size to achieve a specific frequency goal. The next time the system is idle, we compute the higher-accuracy resonance frequency solution with the technique described in Sec. 6, and then update the sensitivities for all open holes.

Sensitivity computation. The sensitivity of a resonance frequency can be approximated by observing how the solution changes during one iteration of Algorithm 1 under a small change in ϕ. The change in the Rayleigh quotient is approximated as

$$\frac{\partial\theta}{\partial\phi} = \left\langle \frac{\partial \mathbf{w}}{\partial\phi}, \mathbf{v} \right\rangle, \; where \; \frac{\partial \mathbf{w}}{\partial\phi} \simeq -\mathbf{D}_m\mathbf{A}_m^{-1}\left(\frac{\partial \mathbf{A}_m\mathbf{x}}{\partial\phi}\right). \tag{11}$$

Please refer to Appendix C for a detailed derivation. The term $\partial\mathbf{A}_m\mathbf{x}/\partial\phi$ is computed using finite differences, where a small change is applied to the hole size and the change in $\mathbf{A}_m\mathbf{x}$ is measured. Using this estimated Rayleigh quotient derivative, the eigenvalue and resonance wavenumber change is computed as

$$\frac{\partial\lambda^0}{\partial\phi} = \frac{1}{|\theta|^2}\overline{\frac{\partial\theta}{\partial\phi}} - \frac{2\bar{\theta}}{|\theta|^4}\,\mathrm{Re}\left(\langle\frac{\partial\theta}{\partial\phi}, \theta\rangle\right), \tag{12}$$

$$\frac{\partial k_{resonance}}{\partial\phi} = -\,\mathrm{Re}\left(\frac{\partial\lambda^0}{\partial\phi}\right). \tag{13}$$

Although this sensitivity approach is a first order approximation, it gives strong guidance as to how much a hole size must be changed to achieve a goal frequency. Because the relationship between the resonance wavenumber and the hole size is nonlinear, the error becomes large when the hole size changes dramatically. In our experiments, we observed that if the change in hole size is on the order of the frequency of a half note, then the error in sensitivity-based estimation is less than 10%.

AutoTune. When the user presses the "AutoTune" button, the tool automatically optimizes multiple hole sizes using the sensitivity information. The hole size updates are computed with the Newton-Raphson method to minimize the squared sum of wavenumber differences between the current and target wavenumbers. The hole size update can be written as

$$\Delta\Phi = \left[\sum_{g\in\mathcal{G}}\left(\frac{\partial k^g}{\partial\Phi}\right)\left(\frac{\partial k^g}{\partial\Phi}\right)^T\right]^{-1}\left(\sum_{g\in\mathcal{G}}\frac{\partial k^g}{\partial\Phi}(\hat{k}^g - k^g)\right), \tag{14}$$

where Φ is the set of all hole size parameters, $\mathcal{G}$ is the set of finger configurations where the goal is specified. For a finger configuration $g \in \mathcal{G}$, k^g denotes the current resonance wavenumber and $\hat{k}^g$ is the goal wavenumber. The number of holes and number of target frequencies can be arbitrary. If the number of target frequencies is larger than the number of holes, the system returns the best possible hole size to achieve the target.

8 Implementation Details

Our resonance simulation framework described in the previous section is designed such that it gives interactive feedback during incremental changes in the geometry. This section describes how we implement this computation.

Local matrix update. The coefficient matrix $\mathbf{A}_m$ is very expensive to compute. Thus, when the geometry is updated, we incrementally update the coefficient matrix computed from the previous geometry rather than recomputing from scratch. In other words, we only recompute the coefficients in the matrix that would be affected by the geometry change. As the size of the coefficient matrix depends on the number of vertices of the mesh, we constrain our mesh edits such that the total number of the vertices does not change.

After the user finishes a geometry edit (e.g, moving a finger hole), we determine which vertices are affected, and recompute the columns and rows which belong to the affected vertices. This computation is order $\mathcal{O}(Nn)$ where n is the number of the affected vertices.

We also locally update the matrix when the geometry is scaled. When the mesh is uniformly scaled by a factor s, we scale all the sampled wavenumbers by the factor $1/s$. This is correct because our coefficient matrix depends only on the relative distances between two points on the surface, scaled by wavenumber (see (A.5) in the supplemental material). The coefficient matrix and eigenvalues are unaffected and the resulting resonance frequency is just scaled by $1/s$. This agrees with the physical effect of scaling down a musical instrument, which produces a higher pitch sound. During the scaling operation, we maintain the size of the mouth piece, hence we update only the part of the matrix associated with the vertices around the fipple.

Mesh import and export. The user can import an arbitrary 3D manifold mesh into our design tool. Figure 9 shows the overall procedure from mesh importing to mesh exporting. The tool first creates an internal cavity with uniform shell thickness h (in all our examples, $h = 0.4$ cm). We use a discrete signed distance field to compute this internal cavity robustly. Next we prepare a simulation mesh by resampling the internal cavity mesh with a specific number of vertices and extruding it by the thickness h. After the user's mesh editing is finished, we join the cavity mesh and input surface mesh by inserting the finger holes and the fipple hole. Finally, we use a mesh Boolean operation to insert a nozzle for the fipple.

Mesh editing during simulation. During the interactive design session, the resonance simulation is performed on the simplified mesh (see Figure 9-(c)). Our tool allows the user to manipulate a finger hole by placing, moving, resizing and opening/closing it. The user can also manipulate the fipple by placing, moving and orienting it. We compute these operations with mesh stitching. We first add vertices to the mesh such that these vertices specify the outline of the finger holes or the fipple shape. Then, we change connectivity of the triangles in the mesh to insert these shapes. These operations only modify vertex positions and triangle connectivity

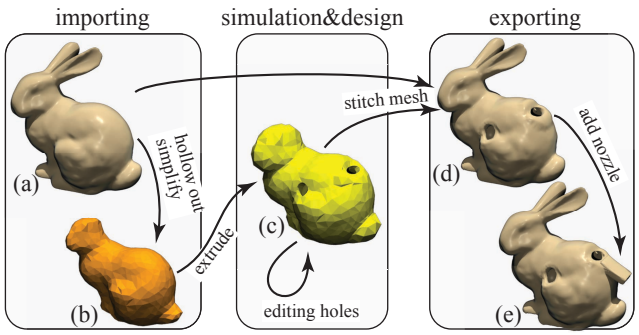


Figure 9: *Overall workflow of geometry processing. (a) the input mesh. (b) the simplified internal cavity mesh. (c) the mesh for acoustic simulation. (d) input mesh and cavity mesh are joined. (e) the final mesh with an air nozzle.*

in the neighbourhood of the edit. Furthermore, in order to make the coefficient matrix reusable, we maintain a constant vertex count by allowing floating/disconnected vertices. Although these floating vertices are not part of the simulation mesh, they have reserved spaces in the coefficient matrix. The entries of the coefficient matrix are adjusted for these floating vertices such that they are decoupled from the active vertices. By limiting the number of the finger holes, we constrain the maximum number of vertices that may be used in the mesh, thus we can maintain a constant-sized matrix for solving during shape editing.

Sound synthesis. During editing, the system plays the predicted resonance sound from a speaker. As our system only predicts fundamental frequencies, no overtones or timbre are simulated. Since a sound with only one frequency sounds artificial, we create a more natural sound by heuristically widening the peak using a normal distribution with 3.0 Hz standard deviation. The audio waveform is then generated by an inverse fast Fourier transform.

9 Results

Figure 1, 10, 15 and 17 show sixteen wind instruments created using our tool. Figure 1, 15 and 17 show various free-form instruments, which are designed to play a specific song. The input models were selected from a public 3D model-sharing website, with choices inspired by the theme of their songs. In addition, we made three instruments that have elongated flute-like shapes (see Figure 10), which were modeled using Teddy [Igarashi et al. 1999]. These free-form shapes would be difficult to approximate with an assembly of easily-analyzable primitives such as tubes, cones, or spheres. Specifically, several of our results have non-convex interior cavities, such as the star-shaped instrument in Fig. 1. We encourage readers to watch the accompanying video to hear the wind instruments' sound. We also include 3D-printable mesh files of these instruments' shapes in the supplemental material.

Figure 17 shows the simulated frequencies (blue lines) and the target frequencies (red lines) for various instrument shapes. With our design system equipped with the interactive simulation and optimization, the user can design instruments in which simulation frequencies agree with target frequencies. Furthermore, we have been able to achieve between 5 to 8 target frequencies for the shapes we have tested, with three to four finger holes. For example the BUNNY model can play eight tones with four holes. Without out interactive simulation feedback, it would be difficult to explore such designs since each hole cannot be tuned independently.

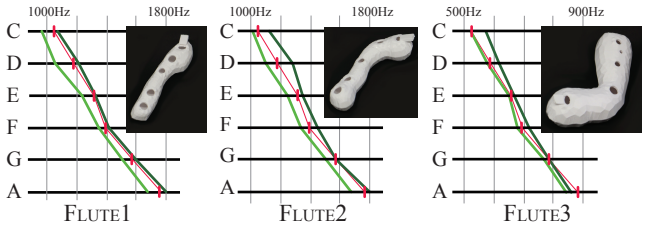


Figure 10: *Flute-like free-form musical instruments designed using our system. Red lines shows the frequencies of target notes and the light and dark green lines show ranges of the measured frequency.*

Accuracy. Figure 17 compares the target frequencies and actual measured frequencies of the instruments we designed. Red lines are the target frequencies, while the measured frequencies are shown as ranges between the light and dark green lines. This range occurs because the sound is influenced by the speed of blowing (higher speed produces higher frequencies). We gradually increased the blowing speed and recorded the lowest and highest frequencies that produce clear sound. We observe that the target frequencies generally fall within measured ranges, i.e., the user can play the exact tones with proper blowing speed. This type of fine-scale tuning by adjusting the blowing speed is very common in wind instruments.

Among the 104 target frequencies we created with various instrument shapes, we experienced only 4 instances where the instrument could not play a target frequency: at the highest frequencies of BEETHOVEN, BUNNY, LIZARD and FLUTE3. Each of these errors is observed where the instrument produces one octave higher

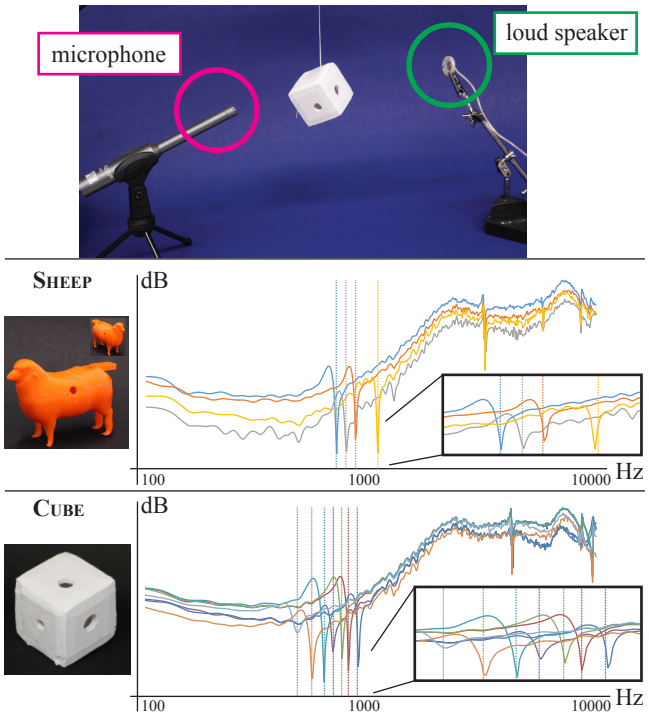


Figure 11: *(Top) passive resonance measurement setup. Sound from a loud speaker is recorded with a microphone for various frequencies. (Bottom) sharp pressure drops in the frequency response indicate resonance frequencies of the cavities. The measured frequencies (solid lines) agree well with the simulated resonance frequencies (dotted lines).*

than the lowest frequency. This occurs because the simulation loses accuracy when the total area of the finger holes becomes large compared to the cavity surface area. These frequencies are at most 3% lower than the target frequency, i.e., the deviation is smaller than half of the deviation of the flat tones. Although we did not tune these instruments at all, such small deviations can be easily fixed by tuning.

To eliminate the effect of the blowing speed, we also analyzed passive resonance frequencies of the cavities. We measure how the magnitude of reflecting sound emitting from a sound source changes with respect to the frequency. Figure 11-top shows our measurement setup, where we hang an instrument in the air, then emit sound of various frequencies with a speaker, and record the reflected sound with a microphone. This experimental set-up is common in finding the frequency response between input and output [Benade and Ibisi 1987]. With constant energy input, the output pressure drops at resonance because resonance mode requires higher energy to drive it. We used the software FuzzMeasure™ to sample the reflection magnitude across many different frequencies. Note that the resonance frequency is insensitive to the location of the microphone and sound source, thus we sampled the spectrum with several miscophone/speaker spatial configurations and averaged them. Figure 11-bottom shows the comparison of simulated resonance frequency against measured passive resonance frequency for the SHEEP model and CUBE models. The CUBE has three holes and an edge length of 4cm. We observed up to 10 Hz deviation between the simulation and the experiment.

In Figure 12, we explored the effect of simulation accuracy by varying the mesh resolution (i.e., the number of vertex N) and also the number of sampled wavenumbers M. The predicted resonance frequencies converge smoothly to the experimental result as these numbers are increased. All the designed examples shown in the paper are made with $N = 1700$ and $M = 5$ to maintain interactivity with reasonable accuracy. Since changes in the blowing speed can easily compensate for a 10Hz error, we believe the parameters we used are reasonable.

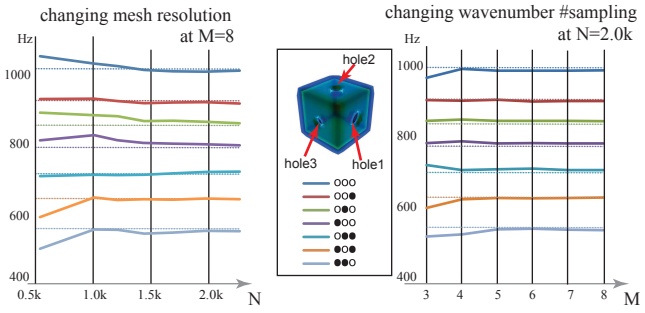
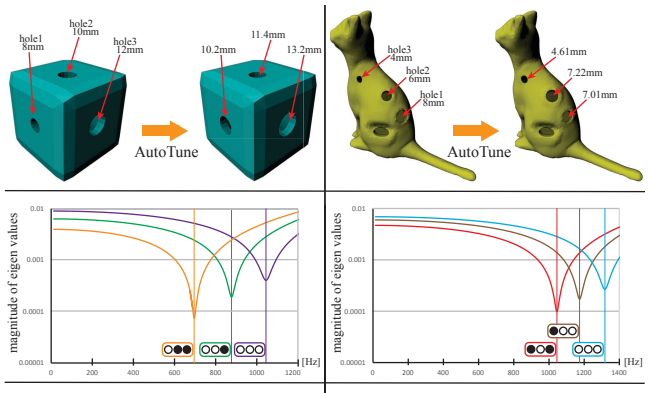


Figure 12: *Change in the resonance frequency with respect to the mesh resolution and the number of the sampled frequencies. Dotted lines shows resonance frequencies obtained from the actual physical measurement. In the legend, ○ ○ ● means that hole 1 and hole 2 are open and hole 3 is closed.*

We optimize the hole size of the CUBE and CAT models such that they can produce three different target tones. Our AutoTune optimizes the sizes of the holes such that the simulated frequency is within 2 Hz. The optimization took two iterations and 24 seconds in total for the CUBE model and three iterations and 41 seconds for the cat model. Figure 13-top shows close-ups of the holes before and after AutoTune.

Next, we verified the accuracy of our matrix interpolation approach (Sec. 6) in our fast approximation of the smallest minimum eigenvalue over a range of frequencies. We compared our frequency estimation result ($M = 5$) against the ground truth frequency sweep where the magnitude of eigenvalues sampled with a small frequency interval (0.1 Hz). Figure 13-bottom compares the ground truth and our resonance frequency prediction for the CUBE and CAT models. We observe that our resonance frequency estimation agrees well with the peaks of the eigenvalue spectrum (error for six tones was within 3 Hz.)



	initial	target	optim.	sweep
○●●	654.2	698.5 (F)	698.8	697.7
○○●	864.3	880.0 (A)	881.2	878.9
○○○	991.7	1046.5(C)	1046.3	1046.5

	initial	target	optim.	sweep
●○●	996.1	1046.5(C)	1045.7	1044.9
●○○	1109.6	1174.7(D)	1174.2	1173.7
○○○	1290.0	1318.5(E)	1318.1	1317.6

Figure 13: *(Top) the optimization of the hole sizes using the AutoTune for the CUBE (left) and CAT (right) model. (Middle) comparison between frequency sweep of the eigenvalues for very small sampling interval (curves) and our estimation of resonance frequency (vertical lines). (Bottom) comparison between the initial resonance frequencies, target frequencies, optimized resonance frequencies after AutoTune and ground truth resonance frequencies obtained from frequency sweep of the eigenvalue.*

Performance. We profile the speed of the each iteration of resonance simulation, total time to find the resonance frequency from scratch, and the speed of updating the resonance while editing the mesh. Table 1 lists statistics for our examples, measured on a MacBook Pro with Core i7 (3.0GHz) CPU. We observed that the performance is quadratic in the mesh vertex count, as the dense matrix update is the bottleneck. Since performance is determined by the mesh size, and we held mesh size fixed, all the models in Figure 17 exhibit timings similar to the CAT and PIG models in Table 1. Our tool is written in C++ and the entire resonance computation is performed by a single CPU thread. We use Lapack implemented on the Intel Math Kernel Library™ to efficiently compute the matrix-vector product for the linear solver. On average our tool runs at 5 frames-per-second and the resonance frequency can be computed in several seconds after the user completes geometry manipulations (opening/closing, relocating a hole). Compared to the frequency sweep approach with a 10 Hz frequency interval, our tool achieves at least a 50× speedup. Our tool provides sufficient interactivity to design functional free-form wind musical instruments shown as in Figure 17. Note that our implementation could be significantly faster with multi-threading or stream processing, since we can independently construct the coefficient matrix and compute the resonance for each wavenumber sampling point.

3D printing. We print the designed wind instruments using PLA filament on a MakerBot Replicator 2 printer, which is a widely available printer/material combination. In order to create an empty cavity inside the shape, we cut the model in half, print each half

Shape	#vtx.	#tri.	time/iter.	# iter.	time ini.	time open/close	time move	time freq. sweep
CAT	1.70k	3.40k	0.149s	15	14.6s	2.2s	2.0s	2m15s
PIG	1.70k	3.40k	0.151s	15	14.8s	2.1s	1.9s	2m16s
CUBE (60% res.)	0.97k	1.94k	0.047s	15	4.2s	0.85s	0.70s	39.2s
CUBE	1.65k	3.30k	0.127s	15	12.9s	2.1s	2.0s	1m59s
CUBE (150% res.)	2.55k	5.10k	0.305s	15	32.6s	6.2s	5.8s	2m57s

Table 1: *Statistics and timing; the number of vertex (#vtx.), the number of triangles (#tri.), average time for each iteration (time/iter.), maximum total number of iterations to find resonance frequency (#itr.), initialization time to compute coefficient matrices for all wavenumber points prior to the design session (time ini.), average time for updating the resonance frequency for opening/closing a finger hole (time open/close), average time for updating the resonance frequency for moving a finger hole (time move), and total computation time for updating resonance frequency using the frequency sweep approach (time freq. sweep).*

separately, and then glue the halves together. The cutting was done manually using a standard mesh editing software, with the cut placed such that we can remove the support material inside the cavity after printing. We carefully orient each part on the print bed so that the joint faces are not covered with support material, to produce a cleaner assembly. Note that with a powder-based 3D printer, this cutting step would be unnecessary, as we can extract the powder via the finger holes.

Saxophone mouthpiece. We also tested our approach with a saxophone mouthpiece, which uses a reed (i.e., a thin mechanically vibrating sheet) as the excitation mechanism. In the simulation, the saxophone mouthpiece is modeled as a completely closed geometry, whereas the fipples in our other examples are modeled as holes with a sharp edge. We used an actual saxophone mouthpiece by plugging it into a socket added to the 3D-printed cavity geometry. Figure 14 shows an example instrument with a saxophone mouthpiece. We observed that with this configuration the generated tones still agree well with the simulation.

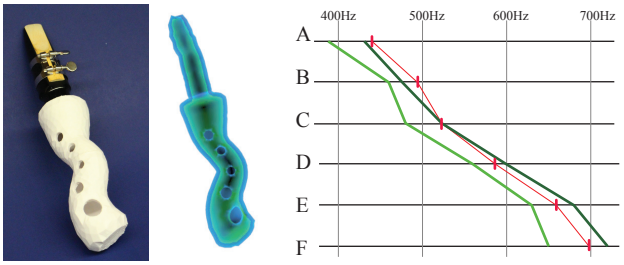


Figure 14: *(Left) free-formed instrument with saxophone mouthpiece. (Middle) simulation mesh for resonance frequency prediction. (Right) the measured tone agrees with the simulation except for the lowest two notes (A,B) by a small margin.*

10 Discussion

Material dependency. We solve the Helmholtz equation under the assumption that the surface reflects the sound wave perfectly (2). If the wall is soft, the resonance frequency is affected as the object surface vibrates, altering the phase and amplitude of reflection wave. To examine the effect of the printing material, we printed the SHEEP instrument with a Objet Connex 260 3D printer [http://stratasys.com] with the VeroWhitePlus^{TM}material (inset-middle), and with an Autodesk Ember^{TM}printer [https://ember.autodesk.com/] with UV-curable resin material (inset-right). Although these printers have significantly higher resolution than consumer FDM printers (inset-left), we did not observe any audible difference in fundamental tone between the three prints. Note that we did not sand, grind or smooth the interior of the 3D shapes, and our experimental results were still very accurate. We conclude that surface roughness has only a small effect on the dominant frequency. Also at the object scales we explored, fabrication error is small compared to the range of frequencies the mouthpiece can produce by changing the blowing speed. Please refer to the supplemental video for a demonstration.

Size dependency. To explore generality of our simulation across instrument sizes, we also made some quite large wind instruments. As discussed in Sec. 8 and Appendix A, the system matrix is invariant under scaling of geometry and wavenumber. This means that if the geometry is scaled by a factor of two, the resonance frequency is reduced by half (one octave lower). To validate this we printed models at scales of $3/2\times$, $2\times$, and $3\times$, and analyzed recordings to see whether they produced sound with frequencies $2/3\times$, $1/2\times$, and $1/3\times$ lower than the original frequency. Fig. 15 show the result of this analysis with the HEN model. The scaled models generate lower tones at the predicted ratios, implying that our simulation can be applied at varying scale.

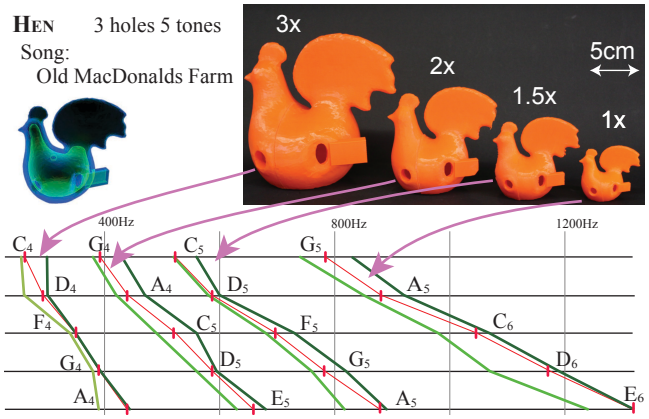


Figure 15: *Scaled instruments. The geometry of* HEN *model (5cm×5cm×7cm) is scaled by 1.5×, 2×, and 3× and the resulting sound is recorded.*

Helmholtz resonator model. Some of the qualitative behaviors of our free-form resonator (e.g., larger finger holes produce higher pitch) can be explained with the Helmholtz resonator model (see Figure 3-left). However, we cannot use the Helmholtz resonator model as a low order approximation, or for a quantitative prediction of resonance frequency. Figure 16 compares our resonance frequency prediction with that produced by the Helmholtz res-

onator model for the STAR instrument shown in Figure 1. For the Helmholtz resonator model, we employed Rayleigh's end corrections for a circular aperture on a thin wall [Howe 1998]. The Helmholtz resonator model does not take into account the cavity shape or the interference between multiple holes, and thus it is inapplicable for free-form wind instruments with multiple finger holes.

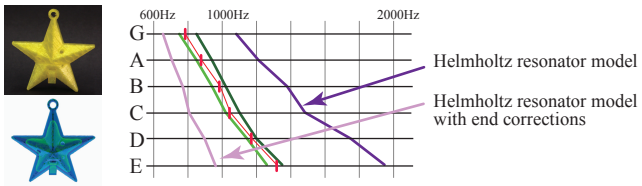


Figure 16: *Comparison of the resonance frequency prediction between our method (red) and the Helmholtz resonator model (dark purple) and the Helmholtz resonator model with end corrections (light purple). We used for the* STAR *instrument. While our prediction agrees with the measured ground truth data (light and dark green), the Helmholtz resonator model is significantly inaccurate.*

The three flute-like instruments in Fig. 10 are much longer along one axis and thus are known to be more accurately modeled by the air column resonator. For example, when all the holes are closed, the FLUTE1 produces 600 Hz sound whose wavelength (54cm) is approximately four times the flute's length (12cm). This agrees with the air column vibration model with one open end (see Figure 3-middle). Our model is based on the wave equation, thus is not limited to the primitive shapes that simple models can handle. Please see supplemental videos for sample sounds.

The bottlenecks in our resonance simulation are the construction of the coefficient matrices and the matrix-vector products. These computations are $\mathcal{O}(N^2)$ since the straightforward implementation of BEM requires dense matrices. To further improve performance, we applied the fast multipole method [Coifman et al. 1993] which reduces the order to $\mathcal{O}(N \log N)$. However, we only observed a 10-20% speedup at our problem scale $N = 1.6$k. This is because it is difficult to find a good decomposition (i.e., separating geometry with minimum overlap) of the smooth, featureless cavity surface.

Limitations and future work. To make interactive free-form wind-instrument design tractable, we simulated the dominant tone of a free-form musical instrument, and ignored the effects of timbre. Simulation of timbre requires the computation of the entire spectrum, including overtones and also modeling of the mouthpiece. As the mouthpiece greatly affects the timbre, the design and optimization of the mouthpiece is an interesting area for future work. Example-based modeling of the mouthpiece would be an interesting way to add timbre specification to our system.

The range of frequencies a wind instrument can achieve depends on the resonator's shape. For example, it is known that achieving a frequency range beyond one octave is difficult for a sphere-like resonator [Fletcher and Rossing 2010]. We are interested in optimizing the overall shape of resonators to achieve arbitrary target frequencies.

We presented a technique to predict resonance frequencies for the Helmholtz equation interactively in the context of free-form instrument design. Aside from music applications, resonance simulation is important in many engineering applications such as design of acoustic devices (e.g, speaker enclosure and mufflers) or electromagnetic devices (e.g., antenna and waveguides).

11 Conclusion

We presented an interactive interface to design free-form wind instruments. The dominant tones of the instruments' sound are modeled as the resonance frequency, which can be formulated as the smallest minimum eigenvalue of the nonlinear system matrix. Then we present a new discrete wavenumber sampling approach to efficiently compute such smallest minimum eigenvalues. We demonstrated the effectiveness of our approach with numerous functional free-form instruments designed with our tool, and with many quantitative experiments.

Acknowledgement

We thank the anonymous reviewers for their feedback. We also thank Dr. Alex Barnett for insightful advice. This project is partially supported by the National Science Foundation under Grant No. 1464267.

References

ALI, A., AND RAJAKUMAR, C. 2004. *The boundary element method: Applications in sound and vibration*. CRC Press.

ALLEN, A., AND RAGHUVANSHI, N. 2015. Aerophones in flatland: Interactive wave simulation of wind instruments. *ACM Trans. Graph. 34*, 4 (July), 134:1–134:11.

BÄCHER, M., WHITING, E., BICKEL, B., AND SORKINE-HORNUNG, O. 2014. Spin-it: Optimizing moment of inertia for spinnable objects. *ACM Trans. Graph. 33*, 4 (July), 96:1–96:10.

BENADE, A. H., AND IBISI, M. I. 1987. Survey of impedance methods and a new piezo-disk-driven impedance head for air columns. *The Journal of the Acoustical Society of America 81*, 4, 1152–1167.

BHARAJ, G., LEVIN, D. I. W., TOMPKIN, J., FEI, Y., PFISTER, H., MATUSIK, W., AND ZHENG, C. 2015. Computational design of metallophone contact sounds. *ACM Trans. Graph. 34*, 6 (Oct.), 223:1–223:13.

BONNEEL, N., DRETTAKIS, G., TSINGOS, N., VIAUD-DELMON, I., AND JAMES, D. 2008. Fast modal sounds with scalable frequency-domain synthesis. *ACM Trans. Graph. 27*, 3 (Aug.), 24:1–24:9.

BRÜNING, J., MARTIN, G., AND PAVLOV, B. 2009. Calculation of the kirchhoff coefficients for the helmholtz resonator. *Russian Journal of Mathematical Physics 16*, 2, 188–207.

COIFMAN, R., ROKHLIN, V., AND WANDZURA, S. 1993. The fast multipole method for the wave equation: A pedestrian prescription. *Antennas and Propagation Magazine, IEEE 35*, 3, 7–12.

COOK, P. R. 1991. Tbone: An interactive waveguide brass instrument synthesis workbench for the next machine. In *Proceedings of the International Computer Music Conference*, 297–297.

DA SILVA, A. R. 2008. *Numerical studies of aeroacoustic aspects of wind instruments*. PhD thesis, McGill University.

DEMMEL, J., DONGARRA, J., RUHE, A., AND VAN DER VORST, H. 2000. *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA.

ELKOURA, G., AND SINGH, K. 2003. Handrix: animating the human hand. In *Proceedings of the 2003 ACM SIG-*

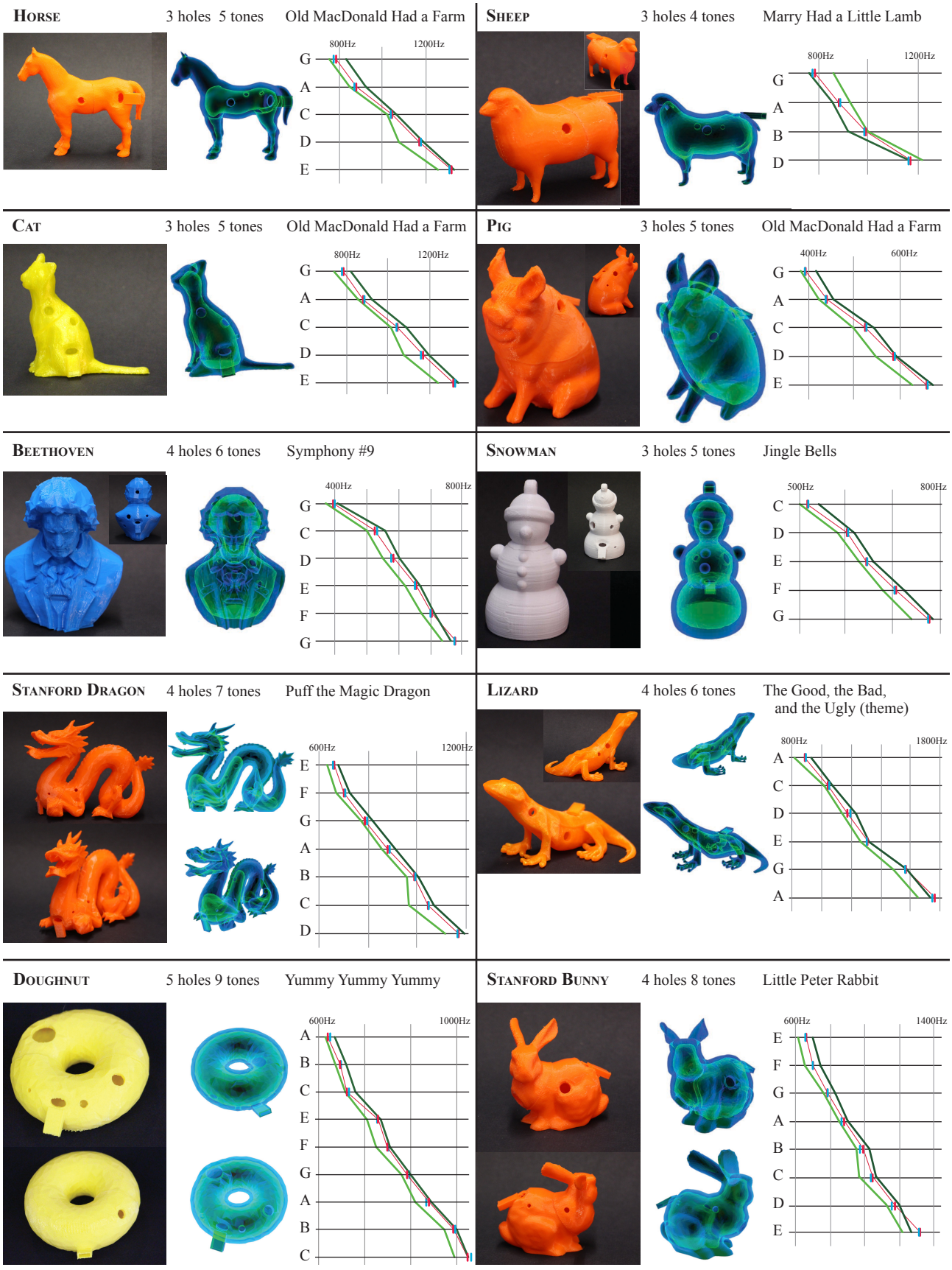


Figure 17: *Examples of free-form wind instrument design. The name of the model, number of holes, number of target notes and the target songs are listed together with frequencies of each note. For each instrument, the red lines show the target frequencies, the blue lines show the simulated frequencies, the light green shows the minimum and the dark green line shows the maximum measured frequencies.*

GRAPH/Eurographics symposium on Computer animation, Eurographics Association, 110–119.

FLETCHER, N. H., AND ROSSING, T. 2010. *The Physics of Musical Instruments*. Springer, 12.

FUNKHOUSER, T., MIN, P., AND CARLBOM, I. 1999. Real-time acoustic modeling for distributed virtual environments. In *Proceedings of SIGGRAPH*, 365–374.

GIORDANO, N. 2014. Simulation studies of a recorder in three dimensions. *The Journal of the Acoustical Society of America 135*, 2, 906–916.

HIRSCHBERG, A., PELORSON, X., AND GILBERT, J. 1996. Aeroacoustics of musical instruments. *Meccanica 31*, 2, 131–141.

HIRSCHMAN, S. 1991. *Digital Waveguide Modeling and Simulation of Reed Woodwind Instruments*. PhD thesis, Stanford University, Stanford, CA.

HOWE, M. 1998. Acoustics of fluid-structure interaction. *Cambridge University Press*.

IGARASHI, T., MATSUOKA, S., AND TANAKA, H. 1999. Teddy: A sketching interface for 3d freeform design. In *Proceedings of SIGGRAPH*, 409–416.

JAMES, D. L., BARBIČ, J., AND PAI, D. K. 2006. Precomputed acoustic transfer: Output-sensitive, accurate sound generation for geometrically complex vibration sources. *ACM Trans. Graph. 25*, 3 (July), 987–995.

KIRKUP, S. M. 1998. *The Boundary Element Method in Acoustics: A Development in Fortran (Integral Equation Methods in Engineering)*. Integrated Sound Software, 10.

LEFEBVRE, A., AND SCAVONE, G. P. 2012. Characterization of woodwind instrument toneholes with the finite element method. *The Journal of the Acoustical Society of America 131*, 4, 3153–3163.

LI, D., FEI, Y., AND ZHENG, C. 2015. Interactive acoustic transfer approximation for modal sound. *ACM Transactions on Graphics 35*, 1.

LI, D., LEVIN, D. I., MATUSIK, W., AND ZHENG, C. 2016. Acoustic voxels: Computational optimization of modular acoustic filters. *ACM Trans. Graph. (to appear) 35*, 4.

LORENZONI, V., DOUBROVSKI, E., AND VERLINDEN, J. 2013. Embracing the digital in instrument making: Towards a musician-tailored mouthpiece by 3d printing. In *Proceedings of the Stockholm Music Acoustics Conference*.

LU, L., SHARF, A., ZHAO, H., WEI, Y., FAN, Q., CHEN, X., SAVOYE, Y., TU, C., COHEN-OR, D., AND CHEN, B. 2014. Build-to-last: Strength to weight 3d printed objects. *ACM Trans. Graph. 33*, 4 (July), 97:1–97:10.

MOHRING, J. 1999. Helmholtz resonators with large aperture. *Acta Acustica united with Acustica 85*, 6, 751–763.

MUSIALSKI, P., AUZINGER, T., BIRSAK, M., WIMMER, M., AND KOBBELT, L. 2015. Reduced-order shape optimization using offset surfaces. *ACM Trans. Graph. 34*, 4 (July), 102:1–102:9.

NEDERVEEN, C. J. 1998. *Acoustical Aspects of Woodwind Instruments, Revised Edition*, rev sub ed. Northern Illinois University Press, 7.

O'BRIEN, J. F., COOK, P. R., AND ESSL, G. 2001. Synthesizing sounds from physically based motion. In *Proceedings of SIGGRAPH 2001*, 529–536.

PRÉVOST, R., WHITING, E., LEFEBVRE, S., AND SORKINE-HORNUNG, O. 2013. Make it stand: Balancing shapes for 3d fabrication. *ACM Trans. Graph. 32*, 4 (July), 81:1–81:10.

RAGHUVANSHI, N., SNYDER, J., MEHRA, R., LIN, M., AND GOVINDARAJU, N. 2010. Precomputed wave simulation for real-time sound propagation of dynamic sources in complex scenes. *ACM Trans. Graph. 29*, 4 (July), 68:1–68:11.

REN, Z., YEH, H., AND LIN, M. C. 2013. Example-guided physically based modal sound synthesis. *ACM Trans. Graph. 32*, 1 (Feb.), 1:1–1:16.

SCHOOFS, A., VAN ASPEREN, F., MAAS, P., AND LEHR, A. 1987. I. computation of bell profiles using structural optimization. *Music Perception: An Interdisciplinary Journal 4*, 3, 245–254.

TSINGOS, N., FUNKHOUSER, T., NGAN, A., AND CARLBOM, I. 2001. Modeling acoustics in virtual environments using the uniform theory of diffraction. In *Proceedings of SIGGRAPH 2001*, 545–552.

UMETANI, N., MITANI, J., AND IGARASHI, T. 2010. Designing custom-made metallophone with concurrent eigenanalysis. *NIME*.

VAN DEN DOEL, K., KRY, P. G., AND PAI, D. K. 2001. Foleyautomatic: Physically-based sound effects for interactive simulation and animation. In *Proceedings of SIGGRAPH 2001*, 537–544.

VAN DER VORST, H. A. 1992. Bi-cgstab: A fast and smoothly converging variant of bi-cg for the solution of nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput. 13*, 2 (Mar.), 631–644.

WANG, L., AND WHITING, E. 2016. Buoyancy optimization for computational fabrication. *Comput. Graph. Forum 35*, 2, 49–58.

YOKOYAMA, H., MIKI, A., ONITSUKA, H., AND IIDA, A. 2015. Direct numerical simulation of fluid–acoustic interactions in a recorder with tone holes. *The Journal of the Acoustical Society of America 138*, 2, 858–873.

ZHENG, C., AND JAMES, D. L. 2010. Rigid-body fracture sound with precomputed soundbanks. *ACM Trans. Graph. 29*, 4 (July), 69:1–69:13.

ZHENG, C., AND JAMES, D. L. 2011. Toward high-quality modal contact sound. *ACM Trans. Graph. 30*, 4 (July), 38:1–38:12.