Audio Engineering Society

Convention Paper

Presented at the 121st Convention
2006 October 5–8 San Francisco, CA, USA

Shape-changing Symmetric Objects for Sound Synthesis

Cynthia Bruyns[1 2], and David Bindel[2]

[1] *CNMAT, University of California at Berkeley, Berkeley, Ca. 94720, USA*

[2] *Computer Science Department, University of California at Berkeley, Berkeley, Ca. 94720, USA*

Correspondence should be addressed to Cynthia Bruyns (`cbruyns@cs.berkeley.edu`)

ABSTRACT

In the last decade, many researchers have used modal synthesis for sound generation. Using a modal decomposition, one can convert a large system of coupled differential equations into simple, independent differential equations in one variable. To synthesize sound from the system, one solves these decoupled equations numerically, which is much more efficient than solving the original coupled system. For large systems, such as those obtained from finite-element analysis of a musical instrument, the initial modal decomposition is time-consuming. To design instruments from physical simulation, one would like to be able to compute modes in real-time, so that the geometry, and therefore spectrum, of an instrument can be changed interactively. In this paper, we describe how to quickly compute modes of instruments which have rotational symmetry in order to synthesize sounds of new instruments quickly enough for interactive instrument design.

1. INTRODUCTION

Modal analysis involves computing the natural frequencies and mode shapes of a structure in order to determine how the structure moves when forced [2]. The modal method lends itself naturally to sound synthesis, because it allows one to model object sounds using oscillator properties computed automatically from a physical model, rather than by programming the oscillators directly. To compute the deformation of a vibrating system, we project the forces in each of the modal directions, and compute the responses of each mode separately. The overall motion of the object's surface is then the sum of contributions from each mode. Because the mode shapes can be precomputed for any fixed geometry, the main run-time cost of modal synthesis is the matrix multiplication used to project external forces onto the modes.

A limitation of the modal method is the cost of computing the eigen-information which affects the modifications that can be made to the object's geometry and material after the eigensolution has been determined. This research addresses the this limitation by exploiting the properties of symmetric systems.

Instruments such as smooth bells are rotationally symmetric. If we choose a finite element discretization which preserves this symmetry, we can perform a numerical separation of variables. We first change to a cylindrical coordinate system, in which the system matrices have a block circulant form. These circulant matrices can then be converted to block diagonal form using the Fast Fourier Transform, where each block corresponds to a separate angular wave number [1]. We then can compute a small modal decomposition for each of these independent sub-blocks instead of directly computing one large modal decomposition for the entire system. The eigenvalues of the original system are given by the eigenvalues of the sub-blocks, and the modal projections of interaction forces can be projected using the modal decompositions of the sub-blocks together with a Fourier transform.

We use this fast mode calculation technique in a synthesizing plug-in for playing rotationally symmetric percussive instruments. The geometry of the instrument is parameterized, and the position of the control points that determine the geometry is mapped to controllers of a MIDI device. In this way, we can modify the geometry using the same controllers we use to play the model.

By using symmetry in the modal analysis, each geometry can be analyzed quickly enough for interactive use. For example, for a bell model with over 2000 degrees-of-freedom, we can compute the complete modal decomposition in just under five seconds. Using an ordinary dense eigenvalue solver, the same computation would take over 16 minutes.

It is not strictly necessary to compute all the eigenvalues and eigenvectors of the system as only a few steady-state partials will be active after a period of resonance. However, for an arbitrary geometry, under arbitrary loading and boundary conditions, determining which resonant frequencies will be of interest is not straightforward. Because of this, we compute all the resonant frequencies in the audible range and use simulation to model the behavior of the object once it has been struck.

The result of this method is a system that allows a user to hear the result of modifying axis-symmetric object geometry interactively by allowing the user to strike the object while the shape is changing.

2. METHOD

Generally, the development of element mass and stiffness matrices follows the relationships:

$$M^e = \int_\Omega N^T \rho N d\Omega, \quad K^e = \int_\Omega B^T D B d\Omega \tag{1}$$

where the superscript e is used to denote the matrices per element (with domain Ω), ρ is the density of the material, D is a matrix representing the material stiffness, N is a matrix representing the element interpolation functions, and B is a matrix representing the operator on the interpolation functions. The exact form of the operator depends on the approximation method used and space of functions employed. Assembling these matrices into a representation for the entire system yields

$$M\ddot{u} + Ku = F(t) \tag{2}$$

where M is the systems mass matrix, K is the stiffness matrix, and $F(t)$is the applied force at time t.

One can then transform the stiffness matrix into cylindrical polar coordinates, by multiplying the appropriate sub-blocks by the matrix:

$$QQ = I \otimes Q \tag{3}$$

where

$$Q = \begin{pmatrix} cos(\theta) & -sin(\theta) & 0 \\ sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{4}$$

In this way:

$$A = K_{11}, \quad B = K_{12}QQ, \quad B^T = QQ^T K_{21} \tag{5}$$

where the subscripts indicate sub-blocks of the original stiffness matrix.

After this transformation, K, is then a block circulant matrix, and it can be represented as:

$$K = \begin{pmatrix} A & B & 0 & ... & ... & B^T \\ B^T & A & B & 0 & ... & 0 \\ 0 & B^T & A & B & 0 & ... \\ ... & ... & ... & ... & ... & ... \\ ... & ... & 0 & B^T & A & B \\ B & 0 & ... & 0 & B^T & A \end{pmatrix} \quad (6)$$

This form can also be expressed as:

$$K = I \otimes A + P \otimes B + P^T \otimes B^T \quad (7)$$

Where P is a cyclic permutation matrix and $\otimes$ is the Kronecker product such that [**?**]:

$$I \otimes A = \begin{pmatrix} A & 0 & ... & ... & ... & ... \\ 0 & A & 0 & ... & ... & ... \\ ... & 0 & A & 0 & ... & ... \\ ... & ... & ... & ... & ... & ... \\ ... & ... & ... & 0 & A & 0 \\ ... & ... & ... & ... & 0 & A \end{pmatrix} \quad (8)$$

Noting that P has an eigenbasis represented by a Fourier matrix (Z) with eigenvalues (ω) that are a partition of unity of the order determined by the dimension (n) of P, we can also represent the system as:

$$K = \quad (9)$$
$$(Z \otimes I) * (I \otimes A + L \otimes B + L^* \otimes B^T) * (Z \otimes I)^T$$

Where L is a diagonal eigenvalue matrix associated with Z:

$$L = \begin{pmatrix} 1 & ... & 0 & ... & ... & 0 \\ 0 & \omega & 0 & 0 & ... & 0 \\ 0 & 0 & \omega^2 & 0 & 0 & \\ ... & ... & ... & ... & ... & ... \\ ... & ... & 0 & 0 & \omega^{n-1} & 0 \\ 0 & 0 & ... & 0 & 0 & \omega^n \end{pmatrix} \quad (10)$$

Now computing the eigensystem for the original system can be done sub-block by sub-block by radial wave number on the pencil.

$$K = A + L(k,k)B + L(k,k)^* B \quad (11)$$

Thus, the system has been reduced from a $(mn) \times (mn)$ system to a $(n \times n)$ one thereby greatly reducing the cost of eigendecomposition from $O((mn)^3)$ to $O(mn^3)$.

The eigenvalues of the original system are the same as the eigenvalues of the reduced system and can be recovered by concatenating the eigenvalues at each iteration. The eigenvectors are recovered by the projection:

$$X_{original} = (X_{reduced})(Z \otimes I) \quad (12)$$

3. RESULTS

We can employ this reduction technique to in the example shown in Figure 1. In this example, we can parametrically control the height, and radii defining the curve of a bell. Using the reduction, the original stiffness matrix of the system can be represented as $40, 66 \times 66$ blocks (40 radial divisions, 11 lateral divisions, 6 degrees-of-freedom per node) instead of one large 2,640 x 2,640 matrix.

Using this method, the time to compute the reduced system is 4.97 seconds as compared to 16.69 minutes needed for the full system. By exploiting the symmetries of the object, we can find the resonant information of an object in a fraction of the time it would take using traditional modal analysis methods.

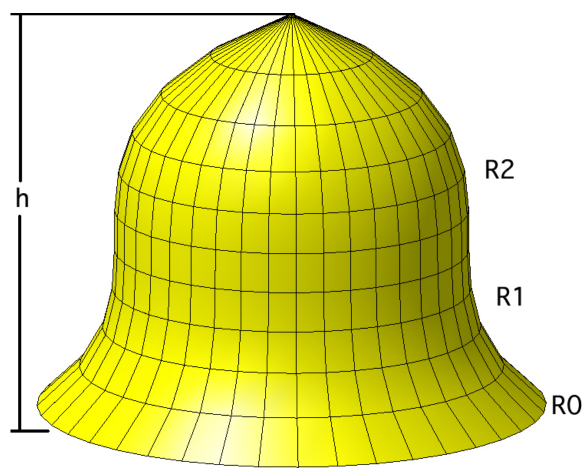

Fig. 1: Undeformed bell with parametric control.

For interactive sound rendering, we created a software plug-in. Using this design allows for integration with music interfaces such as a piano keyboard. The user-interface for the plug-in provides visual feedback, showing changes to the material parameters and object geometry used in the computations. Figure 2 shows the user interface for the audio unit. The top portion the user interface is contains the sliders for parameter adjustment, and the bottom portion gives a three-dimensional view of the model to define the strike position and examine the mode shapes.

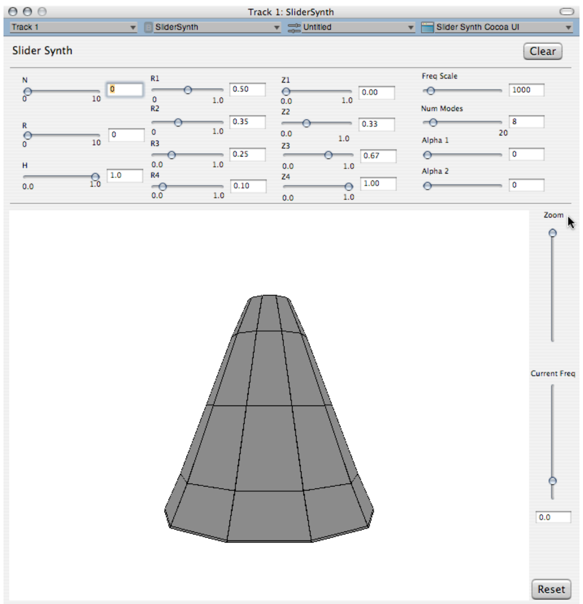

Fig. 2: Synthesizer user interface.

The parameters that the user can control, corresponding to the sliders at the top the plug-in, are as follows: the material parameters such as damping (α_1 and α_2) and stiffness (freq scale). Audio parameters such as number of resonators used. And geometric parameters such as number of radial (R) and lateral (N) segments, as well as height (Z's), and radii (R's) of the segments.

The user interacts with the object my selecting locations on the object's surface with a mouse click. These locations are mapped to keys on a MIDI keyboard. Once the location and key are mapped, the velocity of the key press determines the intensity of the impulse applied to the model.

The software is written in C++ and Objective-C and uses Cocoa and OpenGL APIs for the user interface. The audio engine for the plug-in uses the Core Audio and Altivec APIs. Calls to the synthesizer are made by the host software, which also processes the MIDI events. In this way, the synthesizer acts as a black box, receiving MIDI data and producing an audio stream.

Using this technique, we can generate sounds from objects as geometric modifications are made and then hear the changes in frequency spectrum as a function of shape. Figure 3 shows four different shapes made from modifying the object parameters.

Figure 4 shows that as the radii of the different segments are changed, the peaks in the spectrum change in interesting ways.

By examining Figure 5, we can also see the shift in sustained partials.

These results suggest that there is a subtle interplay between shape and material that will determine the overall timbre of such shapes.

4. DISCUSSION

These results show that changes in an object's geometry will move peaks in its frequency spectrum. By using fast decomposition methods, one can explore how the peaks move and begin to investigate visual and aural aesthetic combinations for instrument design.

Extensions to this research will include non-homogeneous element cross-sections such as those found in many bell designs. The excess material in the non-homogenous cross-sections are often beneficial for bell tuning and it would be another parameter to add to our system.

Using an additional optimization, it is also possible to solve the original system in $O(n^2 log(n))$ time using a a Fast Fourier Transform (FFT) to diagonalize the system. In this way Equation 2 becomes:

$$(ZMZ^T)(Zu)_{tt} + (ZKZ^T)(Zu) = (Zf) \qquad (13)$$

similar to normal modal analysis where Z would be the matrix of eigenvectors. However, in this case, Z is the FFT that diagonalizes the system.

Making the substitutions:

$$\begin{aligned} \hat{M} &= \mathrm{FFT}(\mathrm{FFT}(M)^T)/n, \\ \hat{K} &= \mathrm{FFT}(\mathrm{FFT}(K)^T)/n, \\ \hat{f} &= \mathrm{FFT}(f) \end{aligned} \qquad (14)$$

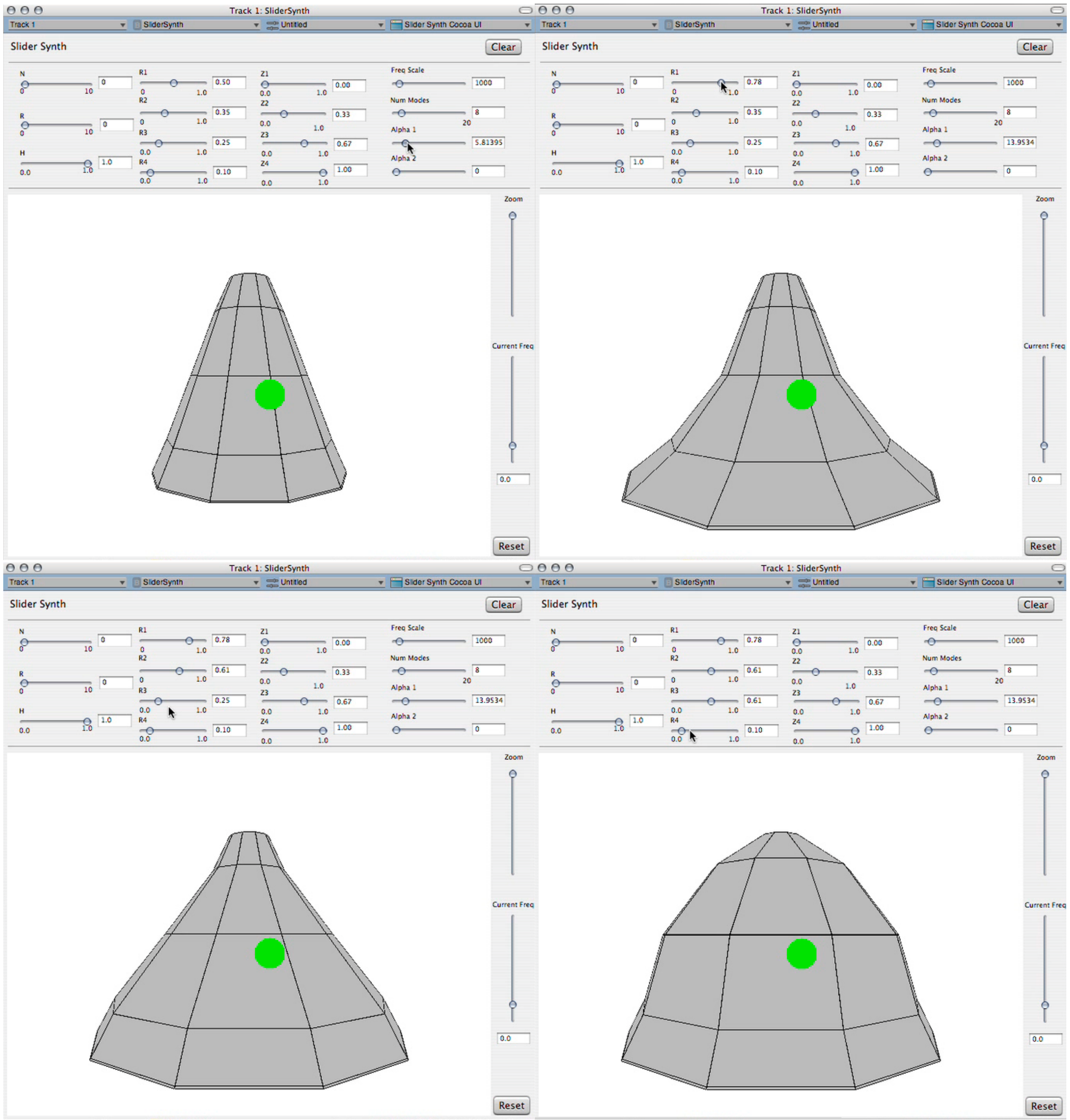

Fig. 3: Deformed parametric shape.

we can express the original system as a set of uncoupled ordinary differential equations as:

$$\hat{M}\hat{u}_{tt} + \hat{K}\hat{u} = \hat{f} \tag{15}$$

We use these equations to solve for $\hat{u}$, and take its Inverse Fast Fourier Transform (IFFT) to recover the original solution to the problem.

It is also possible, since we are simulating a far field listening scenario with a fixed source and listener, to directly obtain the aggregate acoustic pressure by projection instead of adding up the contribution of each oscillating point on the surface [3]. This would

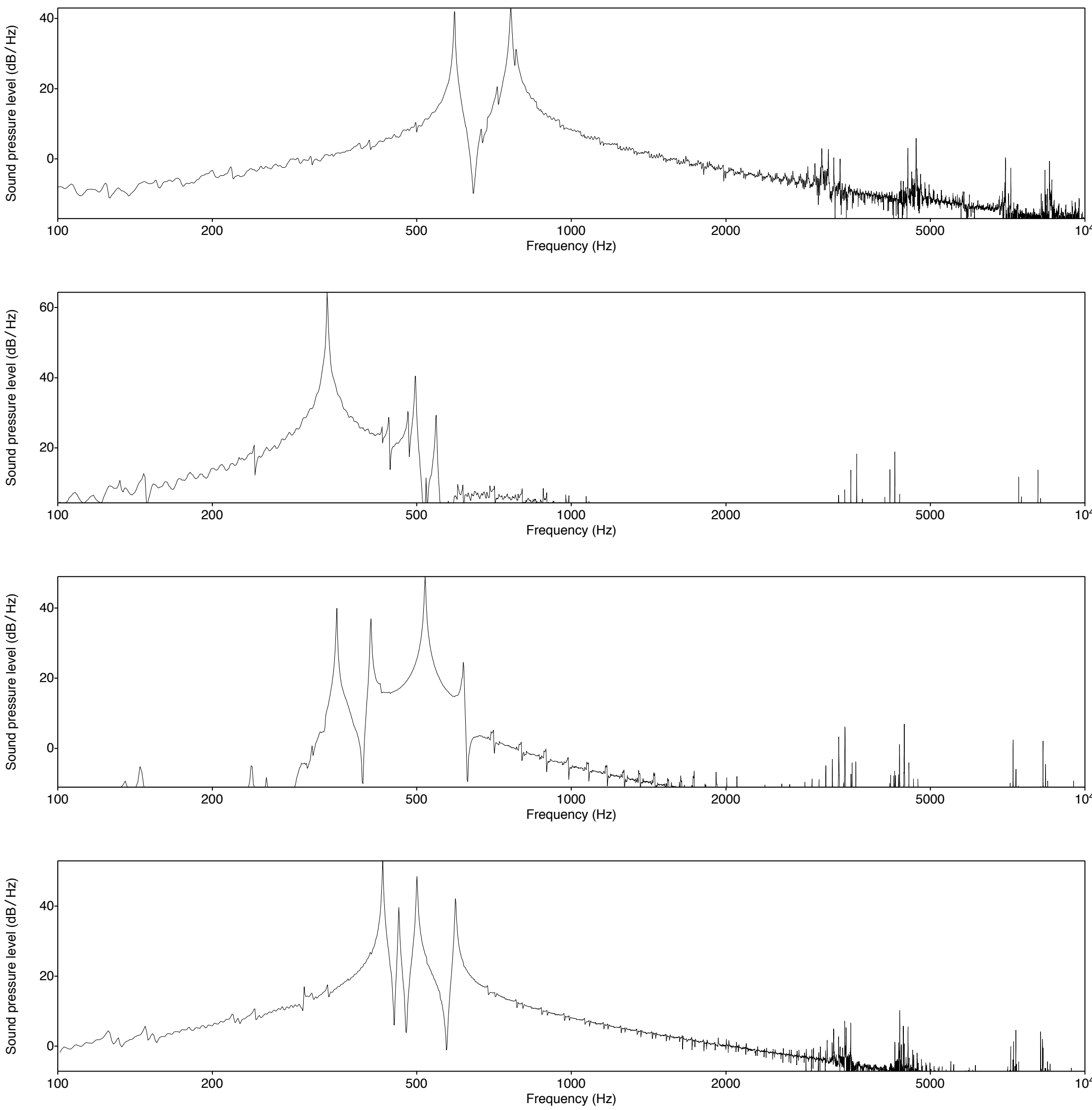

Fig. 4: Frequency spectrum comparison.

be accomplished by forming the linear operator L that weights the contributions of each point and then by performing the projection:

$$u_{total} = \hat{L} * \hat{u} \tag{16}$$

5. ACKNOWLEDGMENTS

Thanks to Jonathan Dorfman for insights into circulant matrix properties. Also thanks to David Wessel, Adrian Freed and Justin Maxwell for their valuable feedback.

6. REFERENCES

[1] Philip J. Davis, "Circulant Matrices," John Wiley & Sons Inc. 1979.

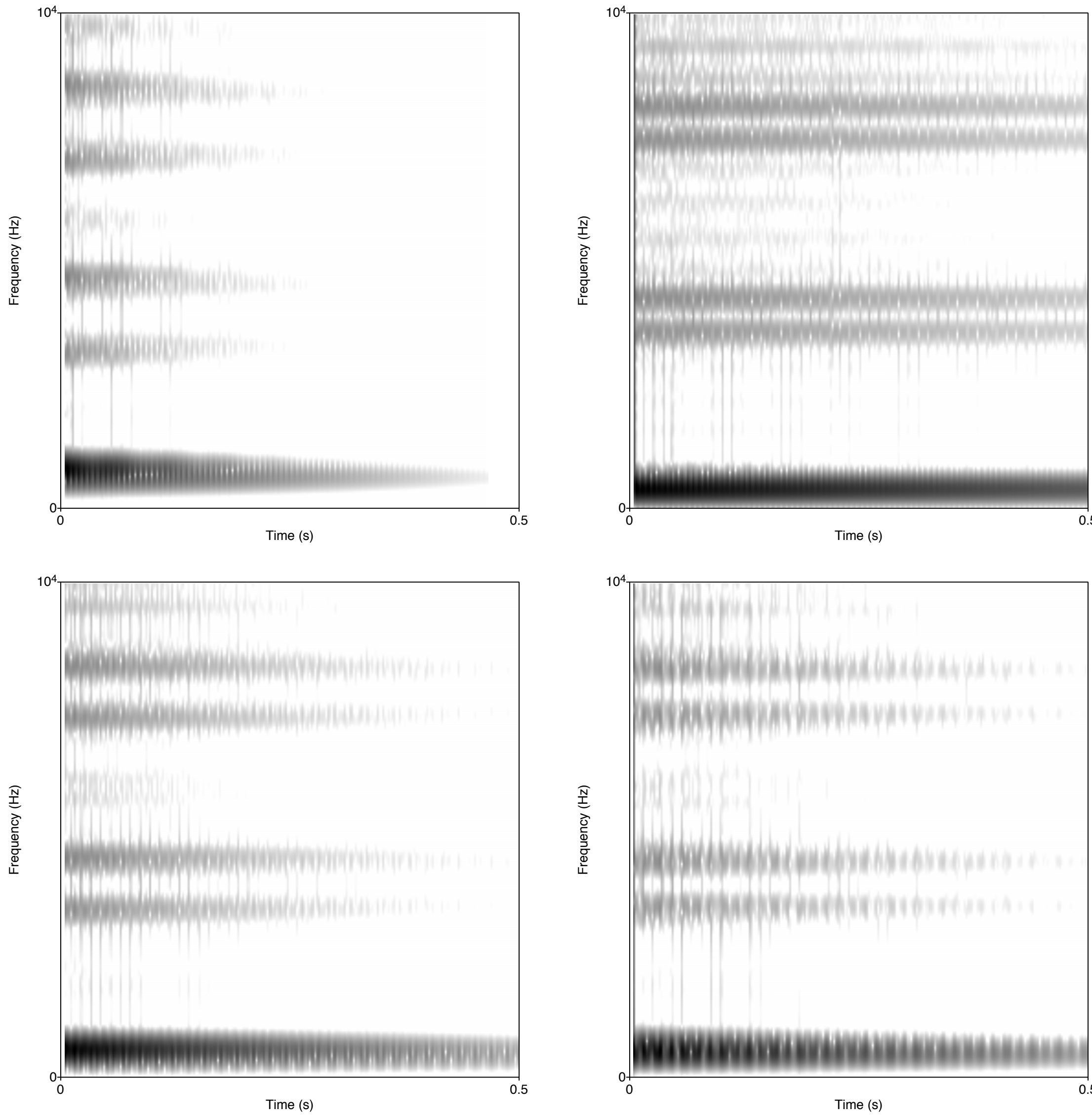

Fig. 5: Sustained resonant frequencies.

[2] Zhi-Fang Fu and Jimin He, "Modal Analysis ," Butterworth-Heinemann. 2001.

[3] Bruyns, C; "Sound Synthesis and Instrument Design for Computer Music." 2005 Master's Thesis, Department of Computer Science, University of California at Berkeley.